TR-O-0111 25

# 非線形有機光学材料の設計論

大田原　一成　　新上　和正

1996. 3. 7

ATR光電波通信研究所

# 非線形有機光学材料の設計論

大田原 一成　新上 和正

## 概論

超高速通信などの光デバイスに応用が期待されている３次非線形光学材料について、非線形性を向上させるための材料の設計論を議論する。従来の試行錯誤的手法や知識，経験に基づく手法でのアプローチに拠らず、積極的に設計するという立場で、経験的手法を用いない新しい設計論を展開する。この設計論を適用することで３次非線形光学材料の設計原理を導くとともに、既存の非線形光学材料の特徴を議論する。

本報告書の目次

本報告書の構成*

第 1 章 材料の設計の序論

有機非線形光学材料の特性をどのように向上させるかという具体的問題に対し、従来の試行錯誤的または知識データベースを基にした経験的な手法を使わない新しい設計の手法を導入する。

第 2 章 対称性のある分子の場合

新しい設計手法に基づき、対称性のある分子に 3 次非線形光学材料の設計原理を導くとともに、現在までに提案されている設計指針との比較する。また、得られた設計原理から見た材料の評価について述べる。

第 3 章 対称および非対称な分子の場合

対称性のある分子に加え、非対称な分子を含めた場合の設計原理を導くとともに、各々の分子種の特徴について述べる。

第 4 章 電子相関効果と分子設計論

3 次非線形光学材料の特性の理論計算は、励起状態を含む電子状態の計算によるため、その電子相関効果が重要となる。設計原理に含まれる物質パラメータ値に対する電子間相互作用の影響について述べる。

第 5 章 結論

*本文中に参照する参考及び文献は、各章ごとに読みやすくするため、重複することがある。

# 第 1 章 材料の設計の序論

3 次非線形感受率の大きい物質は、超高速通信, 画像処理, 電気光学的信号処理, 全光的信号処理など[1]の光デバイスを構成するための鍵となる物質である。実用的なデバイスとして十分に小型で、微弱な光でも動作するためには、3 次非線形感受率を向上させることが必要である。今までに、有機物質および半導体のような無機物質とも、3 次非線形感受率の物理的, 化学的性質について、多くの研究がなされて来た。最近、ナノ構造の技術開発が注目され、特に、量子ドット[2], 有機分子[3]といったナノスケール物質の非線形感受率に興味が注がれている。この分野における中心的課題は、非線形感受率をどのようにして高めるかである。しかし、この問題についての研究は進んでいない。

非線形感受率の測定に用いられる第 3 高調波発生についての 2 次超分極率 ($\gamma$) は、時間依存摂動法から各状態間の仮想遷移経路の総和として次の様に記述される[4]:

$$
\begin{aligned}
\gamma_{ijkl}(-3\omega;\omega,\omega,\omega) &= P\sum_{mnv}\Bigg\{\frac{\mu^i_{gm}\mu^j_{mn}\mu^k_{nv}\mu^l_{vg}}{(E_{mg}-3\hbar\omega)(E_{ng}-2\hbar\omega)(E_{vg}-\hbar\omega)} \\
&+ \frac{\mu^j_{gm}\mu^i_{mn}\mu^k_{nv}\mu^l_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}-2\hbar\omega)(E_{vg}-\hbar\omega)} \\
&+ \frac{\mu^j_{gm}\mu^k_{mn}\mu^i_{nv}\mu^l_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}+2\hbar\omega)(E_{vg}-\hbar\omega)} \\
&+ \frac{\mu^j_{gm}\mu^k_{mn}\mu^l_{nv}\mu^i_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}+2\hbar\omega)(E_{vg}+\hbar\omega)}\Bigg\}. \qquad (1)
\end{aligned}
$$

ここで、$P$ は交換演算子、添字 $m,n,v$ は励起状態 (基底状態を含む), $g$ は基底状態, $i,j,k,l$ はテンソル成分, $E_{mg}$ は基底状態から励起状態 $m$ への励起エネルギー, $\mu_{mn}$ は $m$ $n$ 間の遷移双極子モーメント, $\omega$ は入射光振動数、$\hbar$ は Plank 定数を $2\pi$ で割った量を表す。

3 次非線形感受率の大きくするということは式 (1) に与えられた $\gamma_{ijkl}(-3\omega;\omega,\omega,\omega)$ を大きくするということである。この問題は、一見最適化問題に帰着されるように見えながらそうではない。物質 (システム) 設計は、ある所望の性質を最大にする物質 (材料ではない: 既に機能という性質を含んでいる) を得るための考えと方法を与えることであるが、「ある所望の性質を最大にする」というところは確かに通常の最適化問題となる。しかし、この問題で更に肝心なことは最適化問題に対する式 (1) に含まれる $E_{mg}$ や $\mu_{mg}$ などの物質パラメータが取り得る領域を、実際に合成出来るか自然に存在する物質であるかなどを拠り所として確定することにある。従ってより一般的に '最適化問題を解く' とは解を求めることと定義域を確定することを共に考えなければならない。

最適解が一意的に存在するのではなく屡々 'より効率的' にするためには物質変数をどう動かせばよいかを決めることが可能となるのみである。一般に、関数 $\gamma_i$ は、ある物質パラメータ $\{x_i\}$ ($1\leq i \leq N$) から構成され、従属パラメータと独立パラメータ**を用いて、式 (2) の形で書かれる。ここで、$x_1,\cdots,x_m$ は従属パラメータで、$x_{m+1},\cdots,x_N$ が独立パラメータ、$\gamma_i$ は独立パラメータを固定した時に、$x_1^0,\cdots,x_i^0$ に対する

**従属パラメータと独立パラメータの性質は式 (2)-(4) の性質を持つものと定義される。

最大値を持つと考えると、

$$\gamma_i(x_1,\cdots,x_m;x_{m+1},\cdots,x_N)=\gamma_i^\dagger\Psi(x_i,\cdots,x_m;x_{m+1},\cdots,x_N), \tag{2}$$

が得られる。ここで、

$$\gamma_i^\dagger=\gamma_i(x_1^0,\cdots,x_m^0;x_{m+1},\cdots,x_N), \tag{3}$$

$$\Psi(x_1,\cdots,x_m;x_{m+1},\cdots,x_N)=\frac{\gamma_i(x_1,\cdots,x_m;x_{m+1},\cdots,x_N)}{\gamma_i(x_1^0,\cdots,x_m^0;x_{m+1},\cdots,x_N)} \tag{4}$$

である。ここで、$x_1^0$, $\cdots$,$x_m^0$ で、$\Psi$=1 となる。$\gamma_i(x_1^0,\cdots,x_m^0;x_{m+1},\cdots,x_N)$ は、式 (2) における $\gamma_i^\dagger$ に対応する独立パラメータだけを含んでいる。

纏めれば、(最大化させる) コスト関数は二種類の変数 $x_i$,$y_j$ ($i$=1,..,$m$;$j$=$m$+1,.,$N$) を使って

$$\gamma(x_i,y_j)=\gamma^\dagger(x_i)\Psi_{\{x_i\}}(y_j)\ \ (\Psi_{\{x_i\}}(y_j)\leq 1) \tag{5}$$

と分解された。但し $\Psi_{\{x_i\}}(y_j)$ は $x_i$ をパラメータに $y_j$ を変数とする関数である。$y_j$ は $x_i$ に依存した値 $y_j^*(x_i)$ を取る時 $\Psi_{\{x_i\}}(y_j)$=1 となる。この式から不等式

$$\gamma(x_i,y_j)\leq\gamma^\dagger(x_i) \tag{6}$$

を得る。上記の分解は $\Psi_{\{x_i\}}(y_j)$ は $y_j$ がどんな値を取ろうが $\gamma(x_i,y_j)$ の最大値 (最小値) は $\gamma^\dagger(x_i)$ と $x_i$ のみで決まることを言っている。上の分解式は式の存在についてであり、具体的な分解手続きを与えているのではない。また分解の仕方は $m=0$ 又は $N=0$ 以外の場合には一意的に決まらない。

Operations research (OR) のテキストでは専ら定義域は予め与えられたものとしている通常の最適化問題であり最適解が一意的に存在する。この最適化問題は $\Psi_{\{x_i\}}(y_j)$=1 となる $y_j$ を求めることに対応する。(この際 $m=0$ であり $x_i$ なる変数は現れない) 従って、分解式は最適化問題を解くことより寧ろ二つのクラスの変数に分類される場合に意味が出てくる。効率的に変数を動かすかを決める指針は、変数 $x_i$ に対しては $\gamma^\dagger(x_i)$ を増大させるように、また変数 $y_j$ に対しては $\Psi_{\{x_i\}}(y_j)$=1 になるように、つまり $y_j$ を $y_j^*(x_i)$ に近付けることになる。OR でも定義域自身が問題となる問題群が幾らでもあるように思える。

本報告書では、(5)-(6) で与えられた簡単な式から出発して、式 (1) で与えられる非線形有機光材料の非線形感受率を高める設計論を議論しよう。

# 第 2 章 対称性のある分子の場合

有機化合物の 3 次非線形光学特性を向上するための物質設計原理を、2 タイプ遷移パスモデルを基に検討した。2 タイプ遷移モデルの信頼性は、種々の有機化合物についての電子状態計算により調べた。3 次非線形性を最大化する物質パラメータの組み合わせを、他の物質パラメータを固定して取り出し、設計原理を立てた。実存する有機化合物に対する分子軌道法による電子状態計算の結果は、それらの物質パラメータが 3 次非線形光学特性を最大化していることを示唆する。

## 2.1 始めに

有機物質に対する設計原理を議論する。これを行うため、3 次非線形性を表す簡単なモデルとして、2 タイプ遷移パスモデルを調べた。我々は、ここで、物質の持つパラメータが、あらゆる値も取り得ることを仮定し、そのパラメータ空間の中に於ける 3 次非線形性の挙動を調べた。物質パラメータは従属パラメータと独立パラメータの 2 つのタイプに分類される。従属パラメータとは、他の独立パラメータを固定した時に、3 次非線形性を最大化するようなパラメータである。3 次非線形性を高める物質設計原理は、従属パラメータと独立パラメータについての次の 2 つの条件から得られる。即ち、従属パラメータを、3 次非線形性が最大化させるように動かす, 独立パラメータを 3 次非線形性が増大するように動かすことである。この物質設計原理を議論するとともに、今まで提案されていた設計指針と比較を行う。種々の有機化合物の電子状態は、半経験的分子軌道法を用い、1 電子および 2 電子励起配置を含む配置間相互作用 (Single and Double Configuration Interaction:SDCI) を取り入れて計算した。CI 配置は、CNDO/S(Complete Neglect Differential Overlap/Spectroscopic) 近似により、SCF(Self Consistent Field) 法で得られた基底関数のうち、$\pi$ 電子基底関数から生成した。実存する有機化合物に対する電子状態計算の結果、独立パラメータを固定した時、従属パラメータが、3 次非線形性を最大化する値に近い値を示し、その結果、より簡単化された物質設計原理が得られた。

## 2.2 3 次非線形感受率のモデル

### A. 2 タイプ遷移パスモデル

3 次非線形感受率は、振動数 $\omega$ の光を物質に入射し、発生する $3\omega$ の光強度を測定する第 3 高調波発生法 (Third Harmonic Generation:THG) により評価されることが多い。第 3 高調波発生についての 3 次非線形感受率 $\chi^{(3)}$ は、単位分子あたりの 2 次の超分極率 $\gamma$ を用いて

$$\chi^{(3)}_{ijkl}(-3\omega;\omega,\omega,\omega) = N\gamma_{ijkl}(-3\omega;\omega,\omega,\omega) \tag{7}$$

と書かれる。ここで、添字 $ijkl$ はテンソル成分, $N$ は単位体積あたりの分子数,$\omega$ は入射光の振動数を示す。この 2 次超分極率 $\gamma_{ijkl}(-3\omega;\omega,\omega,\omega)$ を中心に考えて行く。

まず、中心対称性を持つ分子の場合について、2 タイプ遷移モデルを基に、物質設計原理を議論しよう。2 タイプ遷移パスモデルでは、2 つのタイプの仮想的な電子遷移だけが、2 次超分極率へに寄与するものと考える (図 2-1)。2 つのタイプうち、第 1 の遷移タイプは、$g$-1*-$g$-1*-$g$ の遷移パスとする。ここで、$g$-$m$-$n$-$v$-$g$ の表記は、基底状態 $g$ から $m$ 番目, $n$ 番目, $v$ 番目の電子状態を経て、基底状態 $g$ へ戻る遷移パス

を示す。また、$g$-1*-$g$-1*-$g$ にある 1* は、2 次超分極率に対し、負で最大の寄与をする励起状態を表す。従って、1* 状態は第一励起状態と同じであるとは限らない。第 2 の遷移タイプは、$g$-1*-$i$-1*-$g$ の遷移パスで、$i$ は、基底状態 $g$ と 1* 状態を除いた ($i \neq g, 1^*$) 全ての励起状態を表す。

ここで、入射光のエネルギーが電子励起エネルギーに対して無視できるほど小さく、非共鳴の場合を考えよう。この時、時間依存摂動論 [4] から導かれる 2 次超分極率は、

$$\gamma_i = \frac{\mu_{1^*g}^2}{E_{1^*g}^2}\left[\sum_{i\neq,1^*}\frac{\mu_{1^*i}^2}{E_{ig}}\right] - \frac{\mu_{1^*g}^4}{E_{1^*g}^3}, \tag{8}$$

ここで、$\mu_{mn}$ は $m$ 番目と $n$ 番目の状態間の遷移モーメント，$E_{mg}$ は基底状態から $m$ 番目状態への励起エネルギーを表す。我々は、式 (1) で、有機分子の分子軸方向に沿った最も大きい超分極率成分である $\gamma_{iiii}(0;0,0,0)$ だけを考慮することとした。今後、$\gamma_{iiii}(0;0,0,0)$ を $\gamma_i$ と略記する。

B. モデルの信頼性

このモデルの信頼性を確かめるために、種々の有機分子の電子状態を半経験的分子軌道法で、1 電子および 2 電子励起を含む配置間相互作用の方法 (CNDO/S-SDCI)[5] を用いた計算により調べた。配置間相互作用に於いて、1 電子および 2 電子の $\pi$ 配置を取り入れることは、電子相関の性質を記述するのに重要であることが知られており、特にベンゼンなどの小さな分子に対して重要である。そして、CI 効果は分子が大きくなると共に減少する [6,7,8]。入射光エネルギー $\hbar\omega = 0.65$ eV ($\hbar$ は Planck 定数) の時の電子遷移パスの計算結果を図 2-2$a$-$b$ に示す。ここで、各プロットは、2 次超分極率への寄与が大きい順番に並べた。また、我々のモデルに含まれる遷移パスを、小さな円で囲んで示す。図 2-2$a$ で、遷移パス $g$-2-$g$-2-$g$, 図 2-2$b$ で、$g$-1-$g$-1-$g$ は、第 1 の遷移タイプに対応する。これ以外で、小さな円で囲んで示したパスは、第 2 の遷移タイプに分類される。図 2-2$a$ は鎖状構造の $n-$ オクタテトラエンの場合で、図 2-2$b$ は平面構造のスチルベンの場合の結果である。この結果から、有機分子の形状に関係なく、2 タイプ遷移パスモデルが成り立っていると考えられる。

同様にして、ポリエン類, 芳香族類, 縮合環系化合物, キノイド型化合物など種々の有機分子の電子状態について調べたところ、殆どの有機分子について [9]、2 タイプ遷移パスが支配的となるモデルは成り立っていることが分かった。Wu ら [10] は、ポリエンの超分極率に対して 2 個の遷移パスが約 70% の寄与であることを報告している。またこのモデルには、3 レベルモデルの全ての遷移パスを含んでいる。3 レベルモデルは、3 つの電子状態だけを考慮するもので、超分極率を解析するためによく用いられる [11,12]。図 2-2 に、3 レベルモデルの遷移パスを大きな円で示した。図 2-2$a$ に見られるように、3 レベルモデルは、超分極率を表すには不十分である。2 タイプ遷移パスモデルはまた、式 (8) で 1* 状態に種々の励起状態を含む点で、3 タイプモデル [13,14] とも異なる。

## 2.3 設計原理

式 (8) の物質パラメータ $E_{1^*g}, \mu_{1^*g}^2, E_{ig}, \mu_{1^*i}^2$ ($i \neq g, 1^*$) をどのように変化させれば $\gamma_i$ を増大さることができるかという問題を考えよう。各々の有機分子について物質パラメータは独自に決まっているが、この問題を解くために、これらのパラメータが

任意の値を取ることを許して調節可能なパラメータとしてみなすこととする。そして、これらのパラメータ空間における $\gamma_i$ の挙動を調べよう。もし、パラメータ空間で、$\gamma_i$ に絶対的な最大値が存在するなら、設計原理は単純に $\gamma_i$ を最大化させるように各パラメータを動かせばよい。

しかしながら、式 (8) では、$\gamma_i$ はパラメータ空間に最大値を持たない。即ち、$\gamma_i$ はパラメータ $E_{1^*i}$(または $\mu^2_{1^*g}$) の関数として単純に増加 (または減少) するが、一方で、$\mu^2_{1^*g}, E_{1^*}$ は、各々 $\gamma_i$ に対して最大値を持ち、$\mu^2_{1^*g}$ についてのその解は $\partial\gamma_i/\partial\mu^2_{1^*g}$=0 を満たすことであり、$E_{1^*g}$ についての解は $\partial\gamma_i/\partial E_{1^*g}$=0 を満たすことであるが、$\mu^2_{1^*g}$ と $E_{1^*g}$ について、同時に満足する解の組は無く、$\gamma_i$ に対して、$\mu^2_{1^*g}$ と $E_{1^*g}$ を両方同時に最適化することができない。

そこで、$\gamma_i$ を最大化するただ一つのパラメータを取り出すこととする。これを従属パラメータと呼び、残りのパラメータを独立パラメータと呼ぶ。従属パラメータは、独立パラメータを固定した時、$\gamma_i$ を最大化できるパラメータである。従属パラメータの個数は、$\gamma_i$ を同時に最大化できるパラメータの最大個数を取るものとする。一方、各独立パラメータは、$\gamma_i$ を最大化できる値を持っていない。もし、全てのパラメータが、最大化できる値を持ってないなら、従属パラメータ (または独立パラメータ) を選ぶのに曖昧さが残るが、その場合には、設計原理が単純になるように選択することが重要になる。今回の場合、従属パラメータとして $\mu^2_{1^*g}$, 独立パラメータとして $E_{1^*g}, E_{ig}, \mu^2_{1^*i}$ $(i \neq g, 1^*)$ を選ぶのが都合が良い。他の選択として $E_{1^*g}$ を従属パラメータにできるが、設計原理が複雑になる[§]。この様にすると、式 (8) は次のように書き換えできる。

$$
\begin{aligned}
\gamma_i &= \frac{\mu^2_{1^*g}}{E^2_{1^*g}}\left[\sum_{i\neq g,1^*}\frac{\mu^2_{1^*i}}{E_{ig}}\right] - \frac{\mu^4_{1^*g}}{E^3_{1^*g}}, \\
&= \frac{1}{4E_{1^*g}}\left[\sum_{i\neq g,1^*}\frac{\mu^2_{1^*i}}{E_{ig}}\right]^2\left(2[\frac{2\mu^2_{1^*g}/E_{1^*g}}{\sum_{i\neq,1^*}\mu^2_{ig}/E_{ig}}] - [\frac{2\mu^2_{1^*g}/E_{1^*g}}{\sum_{i\neq,1^*}\mu^2_{ig}/E_{ig}}]^2\right), \\
&= \gamma^\dagger_i\Psi(\frac{2\mu^2_{1^*g}/E_{1^*g}}{\sum_{i\neq g,1^*}\mu^2_{ig}/E_{ig}}),
\end{aligned} \tag{9}
$$

ここで $\gamma^\dagger_i$ は次のように定義する。

$$
\gamma^\dagger_i = \frac{1}{4E_{1^*g}}\left[\sum_{i\neq g,1^*}\frac{\mu^2_{1^*i}}{E_{ig}}\right]^2, \tag{10}
$$

また、

$$
\Psi(x) = x(2-x) \quad (0 < x) \tag{11}
$$

で定義される。関数 $\Psi(x)$ は $x$=1 の時に最大値 (=1) を持つ凸関数である。$\Psi(x)$ は、$\mu^2_{1^*g}$ の従属パラメータが、他の独立パラメータを固定した時に、$\gamma_i$ の最大値にどの程度近いかを表す評価関数として見ることができる。一方、$\gamma^\dagger_i$ は独立パラメータだけを

§例えば、$\gamma_i = \mu^4_{1^*g}/E^3_{1^*g}$ について、ここで人工的な場合を考えて見る。各物質パラメータを変化させて $\gamma_i$ を増大させる方法は、$\mu^2_{1^*g}$ を増加させ、$E_{1^*g}$ を減少させることである。他の従属パラメータの選択方法として、ここで、$\alpha = \mu^2_{1^*g}/E^2_{1^*g}$ とすると $\gamma_i = E_{1^*g}\alpha^2$ となり、設計原理は、$\alpha$ を増加させ、$E_{1^*g}$ を増加させることとなる。物質パラメータ $E_{1^*g}$ の変化のさせかたは、2つの方法で異なることとなる。

含み、従属パラメータが任意に変化した時々での、最大値を与える。式 (11) の $\Psi(x)$ の性質から、独立パラメータのみから成り、物質種に関係なく成立する不等式 $\gamma_i \leq \gamma_i^\dagger$ を得る。このように処理すると、求める設計原理は、独立パラメータと従属パラメータに対する次の 2 つの条件から得られる。即ち、

<u>独立パラメータについて、</u>
$(I_1)$ $\gamma_i^\dagger$ を増加する,
<u>従属パラメータについて、</u>
$(I_2)$ $\Psi(x)$ が 1 $(x \to 1)$ になるように近づける, 即ち、 $\mu_{1^*g}^2$ を次の関係を満たすように動かす:

$$\frac{\mu_{1^*g}^2}{E_{1^*g}} = \frac{1}{2}\sum_{i\neq,1^*}\frac{\mu_{1^*i}^2}{E_{ig}}. \tag{12}$$

条件 $(I_1)$ を達成させる、即ち、 $\gamma_i^\dagger$ を増加する方法は、式 (10) から分かるように、
$(i_1)$ $E_{jg}$ $(j \neq g)$ を減少させる,
$(i_2)$ $\mu_{1^*i}^2$ $(i \neq g, 1^*)$ を増加させることである。

ここで、これらの条件 $i_1$ と $i_2$ は、各パラメータの変化が、条件 $(I_1)$ に対して同等でない場合や、$(i_1)$ と $(i_2)$ のような求められる条件が、独立パラメータの選び方に依存するような場合には、どのようにして $\gamma_i^\dagger$ を最も効果的に増大させるかという点について注意する必要がある。

$(I_1)$ と $(I_2)$ で示される設計原理を図 2-3 に示す。この結果を分かりやすく示すために、条件 $(i_1)$ と $(i_2)$ と $(I_2)$ を 3 レベルモデルに当てはめて考える。ここで、 3 つの状態を、 $g$, $1^*$, $2^*(E_{1^*g} < E_{2^*g})$ とすると、求めるシステムは、
$(i_1)$ から、 $1^*$ 番目と $2^*$ 番目の励起状態が縮退し、且つ、基底状態のエネルギーに近づけることであり、
$(i_2)$ から、 $\mu_{1^*g}^2$ を大きくすることであり、
$(I_2)$ から、遷移モーメント $\mu_{1^*g}^2$ が $\mu_{1^*g}^2 = \mu_{1^*2^*}^2/2$ の関係を満たすことである。

## 2.4 結果と考察

### A. 他の設計原理との比較

今までに、 3 レベルモデルを基にした他の設計原理も報告されている [11,12]。それは、 $(P_1)$ 基底状態の 1 次分極率を増加させる, $(P_2)$ 基底状態と $1^*$ 番目の励起状態の分極率の差を増加させることである。ここでは、今回得られた設計原理と今までに報告された設計原理を比較する。これを行うために、得られた設計原理に、 3 つの状態を、 $g$, $1^*$, $2^*(E_{1^*g} < E_{2^*g})$ とした 3 レベルモデルを適用する。基底状態の分極率 $\alpha_g$ と $1^*$ 番目の励起状態の分極率 $\alpha_{1^*}$ は、各々

$$\alpha_g = \frac{2\mu_{1^*g}^2}{E_{1^*g}}, \tag{13}$$

$$\alpha_{1^*} = \frac{2\mu_{1^*2^*}^2}{E_{2^*1^*}} - \alpha_g \tag{14}$$

$(E_{2^*1^*}=E_{2^*g}-E_{1^*g},(>0))$ と定義される。これを、式 (9) に適用すると、

$$\gamma_i = \frac{\alpha_g}{4E_{1^*g}}\{r\alpha_{1^*}-(1-r)\alpha_g\} \tag{15}$$

が得られる。ここで、$r$ は $r=E_{2^*1^*}/E_{2^*g}$ (0<r<1)。従属パラメータを $\alpha_g$, 独立パラメータを $\alpha_{1^*}$, $E_{1^*g}$, $r$ に取り、前節で述べた解析方法を適用すると、最終的に設計原理として、

独立パラメータの$\alpha_{1^*}, E_{1^*g}, r$に対して、
$(II_1)$

$$\gamma_i^\dagger = \frac{r^2\alpha_{1^*}^2}{16(1-r)E_{1^*g}} \tag{16}$$

を増加させること,

従属パラメータである$\alpha_g$に対して、
$(II_2)$ $\alpha_g$ を

$$\alpha_g = \frac{r\alpha_{1^*}}{2(1-r)} \tag{17}$$

の関係を満足させるよう動かすこと

が得られる。本設計原理と今までの設計原理の非常に大きな違いは、独立パラメータを選ぶことにある。条件 $(P_1)$ と $(P_2)$ は、式 (15) における $r = 1/2$ となる特殊な場合であり、その時、$\gamma_i$ は、$\alpha_g$ と $\alpha_{1^*}-\alpha_g$ の積に比例することとなる。この 2 つの量は、独立パラメータとして考えた時に得られるもので、$\gamma_i$ は独立パラメータだけを含む。従って、条件 $(P_1)$ と $(P_2)$ は、前節の第 1 の条件 $(i_1)$ と $(i_2)$ に類似したものである。前節で注意したように、それらの条件は、各パラメータを動かした時、最も効率良く、どのように $\gamma_i$ を増大させるかを示すものであり、それは、独立パラメータの選び方に依存する。一方、条件 $(II_1)$,$(II_2)$ は、独立パラメータとして $\alpha_{1^*}$, 従属パラメータとして $\alpha_g$ を選ぶことで得られる条件である。そして、$(II_1)$,$(II_2)$ は、式 (15) の右辺第 1 項と第 2 項の正負の相反する関係を取り込むことから得られたものである。このような比較から、前節 $(I_2)$ の条件は、今回初めて導かれたものであることが分かる。

B. 実存する物質の評価

次に、物質設計原理の観点から見た、実際の物質に於ける物質パラメータ値について議論しよう。これを行うために、種々の有機分子について 1 電子および 2 電子励起を含む配置間相互作用を取り入れた半経験的分子軌道法 (CNDO/S-SDCI) を用いて電子状態を計算した。電子状態から求める 2 次超分極率の計算は、THG の厳密な表式[4] である $\gamma_{ijkl}(-3\omega;\omega,\omega,\omega)$ の $\hbar\omega = 0.65$ eV の時と、2 タイプ遷移パスモデルで $\hbar\omega = 0$ の時、つまり、式 (8) の値について行った。表 2-1 に示した計算結果から次のことが分かる。

(1) 2 タイプ遷移パスモデルは、表 1 の 2 列目に示されるように、THG の厳密な表式である $\gamma_i(-3\omega;\omega,\omega,\omega)$ を良く再現する。$\gamma_i(-3\omega;\omega,\omega,\omega)$

の計算値はエチレン (1) とベンゼン (10) の場合を除き、実験値と良く一致する。これらの分子は、電子相関効果が重要なため、定量的に $\gamma_i(-3\omega;\omega,\omega,\omega)$ を求めることは難しく、非経験的な *ab-initio* 計算ですら、$\gamma_i(-3\omega;\omega,\omega,\omega)$ を再現しない。

(2) 電子受容性基や電子供与性基を導入した電子的な置換基効果は、分子 7 と 8, 4 と 9, 16 と 17 の比較から、計算を行った有機分子に対しては小さい。これらの分子は、$\gamma_i$ に大きく影響する最高被占軌道 (HOMO) と最低空軌道 (LUMO) が、置換基の導入によって若干の変化しか生じないため、$\gamma_i$ に対する大きな効果を示さなかったと考えられる。

(3) 式 (9) の関数 $\Psi(x)$(表中では $\Psi$ と略) は、分子 1- 6 に見られるように、有機分子の長さが増加するにつれ、1 に近づく傾向がある。これは、他の分子でも同様で、構成原子数が増加していくと $\Psi$ が 1 に近づいて行くことを示唆する。この意味を調べるため、式 (9) の $\sum_{i\neq g,1^*}\mu_{1^*i}^2/E_{ig}$ を $\Psi$ を使って書き直すと次のようになる。

$$\begin{aligned}\gamma_i &= \gamma_i^*\frac{\Psi}{(1-\sqrt{1-\Psi})^2} && \frac{\mu_{1^*g}^2}{E_{1^*g}} \leq \frac{1}{2}\sum_{i\neq g,1^*}\frac{\mu_{1^*i}^2}{E_{ig}} \text{ の場合 ,}\\ &= \gamma_i^*\frac{\Psi}{(1+\sqrt{1-\Psi})^2} && \frac{\mu_{1^*g}^2}{E_{1^*g}} > \frac{1}{2}\sum_{i\neq g,1^*}\frac{\mu_{1^*i}^2}{E_{ig}} \text{ の場合 ,}\end{aligned} \tag{18}$$

ここで、$\gamma_i^*$ は次のように定義する。

$$\gamma_i^* = \frac{\mu_{1^*g}^4}{E_{1^*g}^3} \simeq \frac{\alpha_g^2}{4E_{1^*g}}. \tag{19}$$

式 (19) では、基底状態の 1 次分極率が $\alpha_g \simeq 2\mu_{1^*g}^2/E_{1^*g}$ と近似できる関係を用いた。式 (18) において $\Psi(x) \simeq 1$ と成る、または、式 (12) の関係が関係が成り立つ場合、$\gamma_i \simeq \gamma_i^*$ となる。即ち、$\gamma_i$ が基底状態と $1^*$ 番目の励起状態間の電子的性質にのみ依存することを意味する。これは、$\gamma_i$ が僅かな物質パラメータだけで決定されるという大変注目すべき結果である。$\Psi(x) \simeq 1$ となる有機分子に対する分子設計指針は、$\gamma_i \simeq \gamma_i^*$ と式 (19) の $\gamma_i^*$ の関係から、前節の $(i_1)$ と $(i_2)$ 同様に、

$(ii_1)$ $1^*$ 番目の励起状態エネルギー $E_{1^*g}$ を減少させる,

$(ii_2)$ 基底状態と $1^*$ 番目の励起状態間の遷移モーメントを増加させることとなる。

この条件は、アントラセン (12) に対し、キノイド型ナフタレン (14) が、$\mu_{1^*g}^2$ はあまり大きくないが、$E_{1^*g}$ が小さく、$\gamma_i$ が大きくなっていることに対応する。即ち、アントラセンはキノイド型ナフタレンより遷移モーメントが大きいにもかかわらず、$\gamma_i$ は小さい。しかしながら、キノイド型の有機分子は、$\hbar\omega \ll E_{ig}(i{=}1,2,\cdots)$ という非共鳴領域の条件から外れて行くため、実用的光デバイスの材料として適するとは限らないと考えられる。一方、$\gamma_i$ の、分子長さの変化に対する影響は、分子 1-6 と 10-12 に見られるように電子受容性基, 電子供与性基の置換をしなくとも大きく増加する。これは、励起エネルギー $E_{1^*g}$ の減少というよりも、遷移モーメント $\mu_{1^*g}^2$ の増加に対

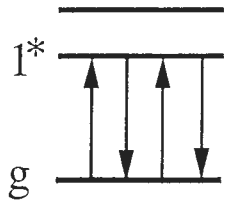
応する。多くの有機分子に対して、なぜ、物理的に依存関係の無い、式 (12) の各パラメータによって、式 (12) の関係が満足するのか、または $\Psi(x) \simeq 1$ が成立するのか、理由は不明である。計算結果から示唆されるように、式 (18) における $\gamma_i \simeq \gamma^*$ の関係から、非常に簡単で実用的な設計原理を与える。

Rustagi と Ducuing[15] は、簡単な 1 次元自由電子系の理論的検討から、$\gamma_i$ と $\alpha_g$ の分子長依存性を $\gamma_i \propto \alpha_g^{5/3}$ と見い出した。このべき則は、我々の導いた関係式に近い。幾つかの有機化合物について $\gamma_i$ のサイズ依存性を確認するための実験は [16] 行われているが、$\gamma_i \propto \alpha_g^{5/3}$ の関係は得られていない。また、$\gamma_i$ と $\alpha_g$ および $\beta$ (2 次非線形感受率) の関係は、非対称な有機分子に対しては、理論的に検討されている [17]。我々の計算結果が示唆する $\gamma_i \simeq \gamma_i^*$ の関係を実験的に確認するために $E_{g1^*}$, $\alpha_g$, $\gamma_i$ の測定データが必要である。しかし、それらの実験データは、$\Psi \simeq 1$ があまり成り立たない、ベンゼンやエチレンなどの小さな有機化合物に対する測定データはあるものの、$\Psi \simeq 1$ が成立する大きな有機化合物についての測定データはなく、サイズの大きな有機化合物について、式 (18) の $\gamma_i \simeq \gamma_i^*$ の関係を実験的に確認することが望まれる。

参考及び文献


1) B.E.A. Saleh and M.C. Telch: *Fundamentals of photonics*, (John-Wiley, New York, 1991).
2) D.W. Hall and N.F. Borrelli, "Absorption saturation in commercial and quantum-confined $CdSe_xS_{1-x}$-doped glasses," J. Opt. Soc. Am. B, **5**, 1650-1654 (1988).
3) T. Yoshimura, S. Tatsuura, and W. Satoyama, "Quantum wire and dot formation by chemical vapor deposition and molecular layer deposition of one-dimensional conjugated polymer," Appl. Phys. Lett., **60**, 268-270 (1992).
4) 例えば、 *Nonlinear Optics* by R.W. Boyd, (Academic press, 1992).
5) 例えば、 J.A. Pople and D.L. Beveridge, *Approximate Molecular Orbital Theory* (McGraw-Hill, Yew York, 1971).
6) A.F. Garito, J.R. Heflin, K.Y. Wong, and O. Zamani-Khamiri, *Nonlinear Optical Properties of Polymers*, ed. by A.J. Heeger, D. Uhlich, and J. Orenstein (Mater. Res. Soc. Proc. 109, Pittsbrugh, PA, 1988)
7) E. Perrin, P.N. Prasad, P. Mougerot, and M. Duppis, "*Ab initio* calculations of polarizability and second hyperpolarizability in benzene including electron correlation treated by Møller-Plesset theory," J. Chem. Phys., **91**, 4728-4732 (1989).
8) B.M. Pierce, "A theoretical analysis of third-order nonlinear optical properties of linear polyene and benzene," J. Chem. Phys., **91**, 791-811 (1989).
9) 幾つかの有機化合物は、２タイプ遷移パスモデルが優勢とならない。例外は、幾つかの励起状態が縮退に近く、ほぼ同じ遷移モーメントを持っている場合に起きる。これは、ピレンのように２次元系で高い対称性を持っている化合物でよく現れる。しかし、本論文で取り扱っている２タイプ遷移パスモデルは、これらの化合物に合うように容易に拡張できる。
10) J.W. Wu, J.R. Heflin, R.A. Norwood, K.Y. Wong, O. Zamani-Khamiri, and A.F. Garito, "Nonlinear-optical processes in lower-dimensional conjugated structures," J. Opt. Soc. Am. B, **6**, 707-720 (1989).
11) B.M. Pierce, "A theoretical analysis of third-order nonlinear optical properties of $\pi$-electron conjugated molecules," SPIE **971**, 25-41 (1988).
12) C.W. Dirk, L. Cheng, M.G. Kuzyk, "A simplified three-level model describing the molecular third-order nonlinear optical susceptibility," Int. J. Quant. Chem., **43**, 27-36 (1992).
13) A.F. Garito, J.R. Heflin, K.Y. Wong, and O. Zamani-Khamiri, *Organic Materials for Non-linear Optics*, ed. by R.A. Hann and D. Bloor (Royal Sciety of Chemistry, 1988)
14) M. Nakano, M. Okamura, K. Yamaguchi, and T. Fueno, "CNDO/S-CI calculations of hyperpolarizabilities. III. Regular polyenes, charged polyenes, disubstituted polyenes, polydiacetylene and related species," Mol. Cryt. Liquid, **A182**, 1-15 (1990).
15) K.C. Rustagi and J. Ducuing, "Third-order optical polarizability of conjugated organic molecules," Opt. Commun., **10**, 258-261 (1974).
16) J.P. Hermann and J. Ducuing, "Third-order polarizabilities of long-chain molecules," J. Appl. Phys., **45**, 5100-5102 (1974).
17) R. Morita and M. Yamashita, "Ralationship between second- and third- order nonlinear optical susceptibilities due to electronic polarization," Jpn. J. Appl. Phys., **32**, L905-L907 (1993).

< The 1st type >




< The 2nd type >

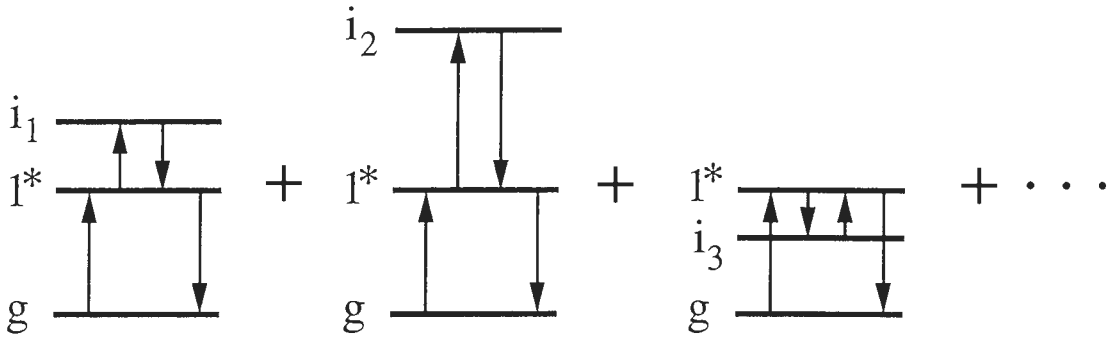
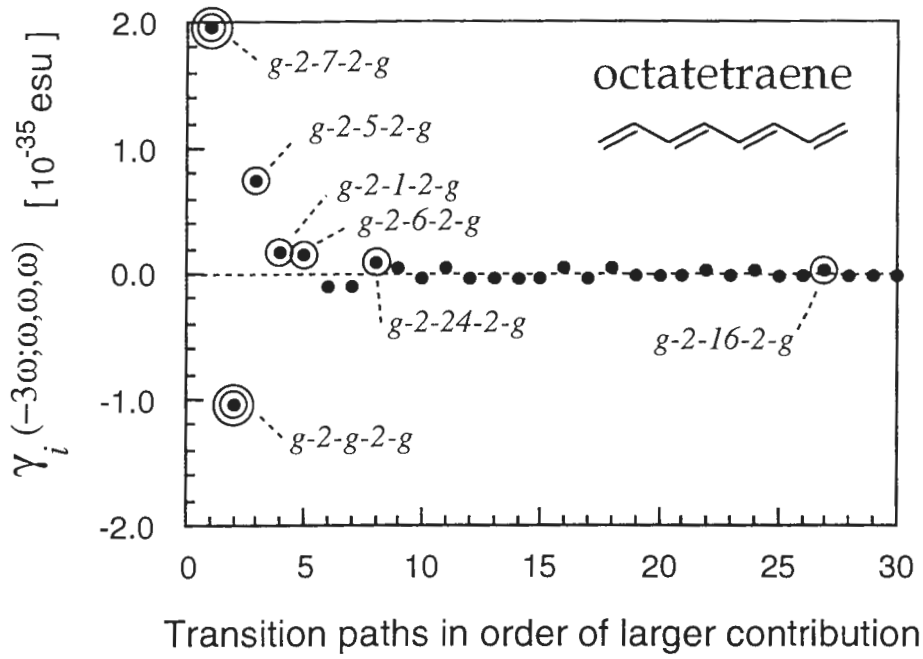



図 2-1. 2 次超分極率に支配的となる 2 つの型の遷移パス。第 1 の遷移タイプは、$g$-1*-$g$-1*-$g$ の各励起状態を経由する。ここで 1* は、2 次超分極率に負で最大の寄与をする励起状態。第 2 の遷移タイプは、$g$-1*-$i$-1*-$g$ $(i \neq g, 1^*)$ を経由する過程。

(a)




(b)

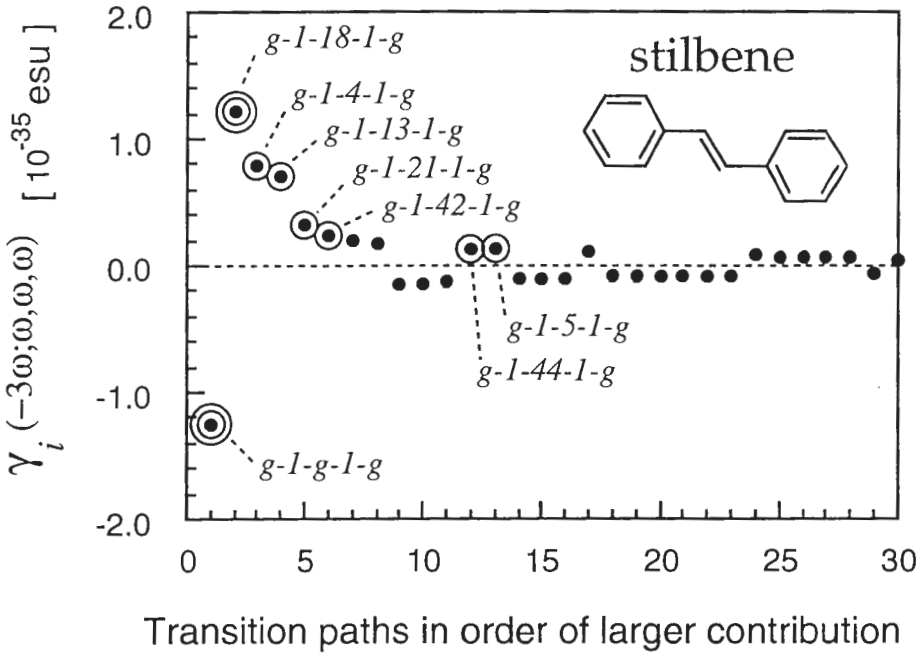



図 2-2. 代表的な有機分子の遷移パス成分。各遷移パスは 2 次超分極率に大きく寄与する順番に並べた。2 タイプ遷移パスは、小さな円で囲んで示す。計算は、1 電子および 2 電子励起の配置間相互作用を含めた半経験的分子軌道法 CNDO/S-SDCI 法により行った。(*a*) *n*- オクタテトラエン。遷移パス *g*-2-*g*-2-*g* は第 1 のタイプ、これ以外の小さな円で記した遷移パスは、第 2 のタイプに属する。(*b*) スチルベン。遷移パス *g*-1-*g*-1-*g* は第 1 のタイプ、他の小さな円で記した遷移パスは、第 2 のタイプに属する。

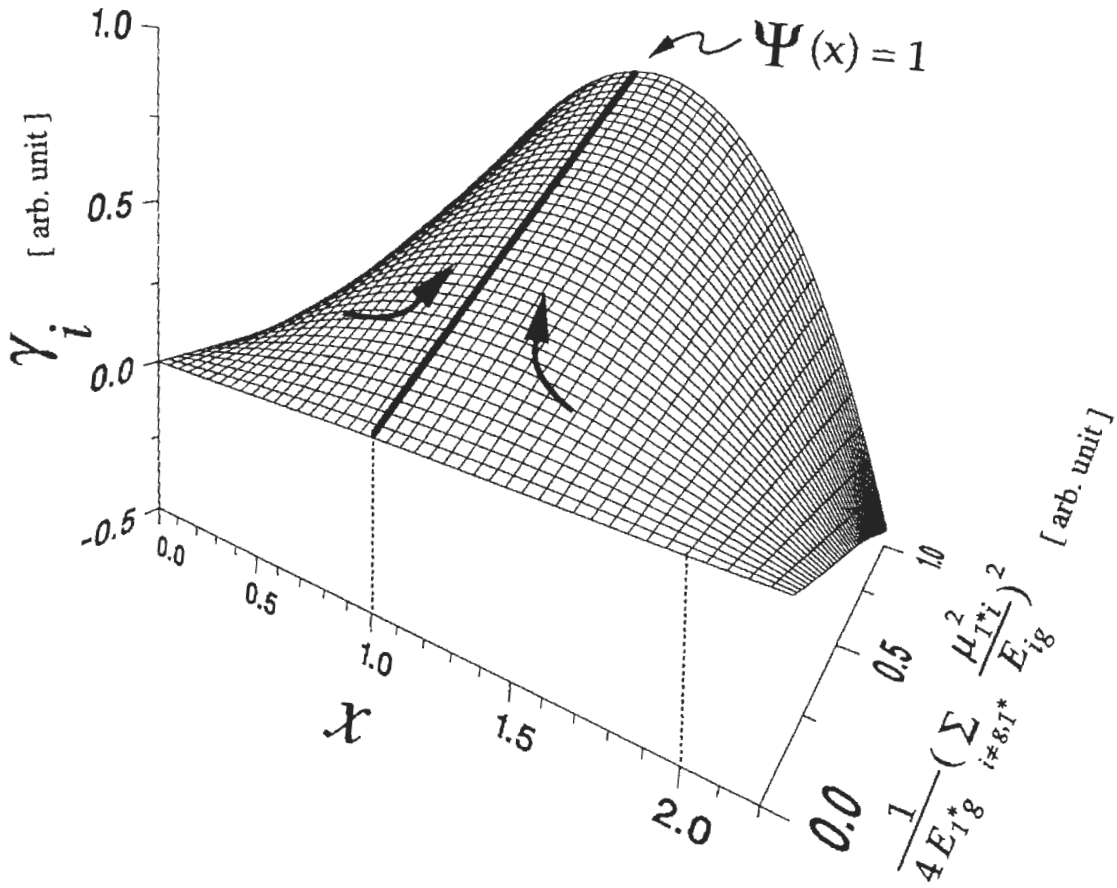
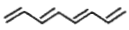
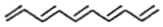
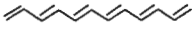
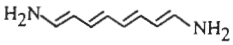
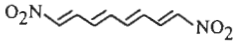
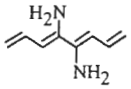
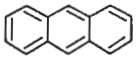
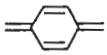
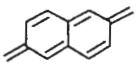
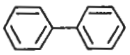
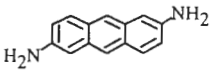
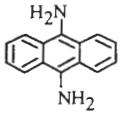



図 2-3. $(I_1)$, $(I_2)$ で記述した設計原理。$\Psi = 1$ の線上は、$\gamma_i$ が式 (10) で与えられる $\gamma_i^\dagger$ に等しい。設計原理は矢印で示した方向に物質パラメータを変化させることである。

表 2-1. 幾つかの有機分子に対する 2 次超分極率と物質パラメータの計算値。計算は $\hbar\omega = 0.65$ eV 時の、$\gamma_{ijkl}(-3\omega, \omega, \omega, \omega)$ の厳密な表式と、$\hbar\omega = 0$ 時の、式 (9) による表式について行った。式 (10) で定義した $\gamma^{\dagger}$ は、2 次超分極率の上限を与える。

| | Compound | $\gamma_i^{(-3\omega;\omega,\omega,\omega)}$ [$10^{-36}$ esu] | $\gamma_i^{\text{Two types paths}}$ [$10^{-36}$ esu] | $\Psi$ | $\gamma_i^{\dagger}$ [$10^{-36}$ esu] | $\mu_{1^*g}$ [Debye] | $E_{1^*g}$ [eV] |
|---|---|---|---|---|---|---|---|
| 1 |  | 0.003 | 0.03 | 0.748 | 0.04 | 3.77 | 8.03 |
| 2 |  | 0.8 | 0.98 | 0.866 | 1.13 | 4.85 | 6.66 |
| 3 |  | 4.9 | 5.0 | 0.942 | 5.3 | 6.70 | 5.43 |
| 4 |  | 17.0 | 17.4 | 0.890 | 19.5 | 8.26 | 5.06 |
| 5 |  | 49.2 | 47.7 | 0.904 | 52.8 | 9.89 | 4.52 |
| 6 |  | 111.7 | 103.3 | 0.911 | 113.4 | 11.32 | 4.15 |
| 7 |   | 34.1 | 32.9 | 0.899 | 36.7 | 8.99 | 4.54 |
| 8 |   | 38.2 | 45.6 | 0.964 | 47.3 | 10.08 | 4.33 |
| 9 |   | 18.5 | 18.5 | 0.906 | 20.4 | 7.93 | 4.61 |
| 10 |  | 0.35 | 0.26 | 1.000 | 0.26 | 4.36 | 7.02 |
| 11 |  | 3.3 | 2.7 | 1.000 | 2.7 | 7.28 | 6.29 |
| 12 |  | 9.7 | 13.7 | 0.990 | 13.8 | 10.35 | 5.50 |
| 13 |  | 6.6 | 4.9 | 0.984 | 4.9 | 6.55 | 4.91 |
| 14 |  | 33.2 | 22.1 | 0.983 | 22.4 | 8.03 | 3.91 |
| 15 |  | 8.0 | 6.4 | 0.772 | 8.2 | 5.99 | 5.19 |
| 16 |   | 11.9 | 17.2 | 0.985 | 17.5 | 10.68 | 5.23 |
| 17 |   | 9.8 | 11.0 | 0.995 | 11.1 | 9.46 | 5.35 |

# 第 3 章 対称および非対称な分子の場合

有機物質の 3 次非線形光学特性を増大させる物質設計原理を 5 タイプ遷移パスモデルを基に提案する。5 タイプ遷移パスモデルの信頼性は、種々の有機化合物の電子状態計算から確認を行った。物質設計原理は物質パラメータのうち、独立パラメータと呼ぶパラメータを固定した時に、3 次非線形性を最大化する従属パラメータと呼ぶパラメータの組を抽出することで打ち立てた。分子軌道法による計算結果は、対称な有機分子では、従属パラメータが非線形性をほぼ最大化しているが、一方で、非対称な有機分子では、従属パラメータが非線形性の最大値から離れていることを示唆する。

## 3.1 始めに

3 次非線形感受率を高めるための研究は、有機物質, 無機物質を問わず、多くの研究がなされて来た。最近、ナノレベルの微細加工技術が注目されているが、特に、量子ドット [1] や有機分子 [2] といったナノスケールの物質についての非線形性に興味が持たれている。この分野の中心的課題は、どのようにすれば 3 次非線形感受率を増大できるかという問題であるが、あまり研究がなされていない。

本論文は、中心対称および非対称な有機分子の両方についての物質設計原理を議論する。これを行うため、我々は、5 タイプ遷移モデルを調べる。そして、物質パラメータがどんな値も取り得ることを許したパラメータ空間で、3 次非線形感受率の挙動を調べる。物質パラメータは、従属パラメータと独立パラメータの 2 つの型に分類する。3 次非線形感受率を増大させる物質設計原理は、独立パラメータを固定した時に、従属パラメータを非線形性を最大化するように動かすこと、および、独立パラメータを非線形性を増大するように動かすことと得られた。この設計原理は、実際の $\pi$- 共役系直鎖炭化水素と芳香族化合物について、一重項 $\pi$ 電子配置についての全 1 電子および全 2 電子配置間相互作用を取り入れた半経験的分子軌道法 (CNDO/S-SDCI)[3] と状態和法による 3 次非線形感受率の計算から検討した。

## 3.2 モデルと設計原理

### A. 5 タイプ遷移パスモデル

5 タイプ遷移パスモデルを基にした中心対称性のある分子と非対称な分子についての物質設計原理について議論しよう。まず、1 分子についての取り扱いを行うため、非線形感受率は、単位分子当たりの 3 次非線形感受率 $\chi^{(3)}$ を表す 2 次超分極率 $\gamma$ について考えて行く。5 タイプ遷移パスモデルは、ただ 5 つのタイプの仮想的電子遷移だけが、3 次非線形感受率に対して寄与するものと仮定するモデルである。この 5 つのタイプの遷移パスを図 3-1 に示す。ここで、中間状態 $m^*$ は、$\gamma$ に対して最も大きな寄与をする遷移パスに含まれる 1 番目の状態とする (基底状態 $g$ は含まない $m^* \neq g$)。状態 $i$ は、$m^*$ 状態と基底状態を除いた ($i \neq m^*, g$) 種々の励起状態を表す。このモデルは、非線形感受率を解析するのによく用いられる 3 レベルモデル [4,5] の全ての遷移パスを含む。この 3 レベルモデルでは、3 つの電子状態だけを考えているが、非線形感受率を再現するには不十分である。我々のモデルは、また、他に提案されている 3 タイプモデル [6,7] とも異なる。3 タイプモデルは、I,II と III の遷移タイプだけを含むが、$m^*$ については、種々の励起状態を含んでいる。この 3 タイプモデルも、後に議論するように、非線形感受率を再現するのに十分とは言えない。

入射光エネルギー $\hbar\omega$($\hbar$: Plank 定数) が、電子励起エネルギーに対して、十分小さく、非共鳴の場合を考え、有機分子の長軸方向にあって 2 次超分極率の最大となるテンソル成分だけを考えることとしよう。この場合、2 次超分極率は [8]

$$\begin{aligned}\gamma &= \frac{\mu_{m^*g}^2}{E_{m^*g}^2}\sum_{i\neq g,m^*}\frac{\mu_{m^*i}^2}{E_{ig}} - \frac{\mu_{m^*g}^4}{E_{m^*g}^3} + \eta\Big\{\frac{\mu_{m^*g}^2\bar{\mu}_{m^*m^*}^2}{E_{m^*g}^3} \\ &- \frac{\mu_{m^*g}^2}{E_{m^*g}^2}\sum_{i\neq g,m^*}\frac{\mu_{ig}^2}{E_{ig}} + \frac{\mu_{m^*g}\bar{\mu}_{m^*m^*}}{E_{m^*g}^2}\sum_{i\neq g,m^*}\frac{\mu_{m^*i}\mu_{ig}}{E_{ig}}\Big\}, \qquad (20)\end{aligned}$$

ここで、$\mu_{mn}$ は $m$ 番目と $n$ 番目の状態間の遷移モーメント, $\bar{\mu}_{mn}$=$\mu_{mn}$-$\mu_{gg}\delta_{mn}$, $E_{mg}$ は基底状態から $m$ 番目の状態への励起エネルギー, $\eta$ は 2 値関数で対称な系では $\eta$=0, 非対称な系では $\eta$=1 とする。

B. モデルの信頼性

このモデルの信頼性を確認するために、種々の有機分子について、半経験的分子軌道法 (CNDO/S-SDCI) を用いて電子構造を調べた。入射光エネルギーが $\hbar\omega = 0.65$ eV の場合について計算した電子遷移パスを、2 次超分極率に対して寄与の大きな順番に並べた結果を図 3-2 に示す。我々のモデルに取り入れられない遷移パスを白丸でプロットした。この結果から、5 タイプ遷移パスモデルは有効であると考えられる。図 3-2 に見られるように、3 タイプ遷移パスが支配的であるモデル [6,7] に含まれない遷移パスも $\gamma$ を良く再現するのに必要であることが分かる。ポリエンや芳香族化合物についての電子構造を調べた結果、5 タイプ遷移パスが支配的となるモデルは、多くの対称および非対称な有機分子とも成り立っていることが分かった。

3.3 設計原理

式 (20) の物質パラメータ $E_{m^*g}$, $\mu_{m^*g}$, $\bar{\mu}_{m^*m^*}$, $E_{ig}$, $\mu_{ig}$, $\bar{\mu}_{m^*i}$, $(i \neq g, m^*)$ をどのように変化させれば、$\gamma$ を増大することが出来るか考えよう。これを行うため、実際には、物質によって固有に決まるこれらのパラメータ値がどんな値も取り得ることを許し、調節可能なパラメータであると仮定する。そして、これらのパラメータの取り得る全範囲からなるパラメータ空間の中で、$\gamma$ の挙動を見て行こう。もし、$\gamma$ がこのパラメータ空間中に唯一の最大値を持っているなら、設計原理は $\gamma$ を最大化させるように物質パラメータを動かすことと決定できる。しかしながら、式 (20) からすぐに分かるように、$\gamma$ はこのパラメータ空間に最大値を持っていない。そこで、物質パラメータの種類を従属パラメータと独立パラメータと呼ぶ 2 つのタイプに分類する。従属パラメータとは、他のパラメータを固定した時に、$\gamma$ に対し、最大値を持っているパラメータである。他の残りは、独立パラメータで、これは、$\gamma$ を単純に増加または減少させるパラメータである。従属パラメータの個数は、同時に $\gamma$ を最大化できる最大個数を取ることと定義する。もし、全てのパラメータが従属パラメータで無い場合には、従属パラメータ (または独立パラメータ) を選択するのに曖昧さが残るが、その場合には、設計原理が簡単になるように選択することが重要である。

従属パラメータとして $\mu_{m^*g}$ と $\mu_{ig}$(対称分子の場合は $\eta$=0 で $\mu_{ig}$ は消える) を選ぶと、式 (20) は、

$$\gamma = \gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}_{m^*m^*}^2, \mu_{m^*i}^2)\Psi(\mu_{m^*g}, \mu_{ig}). \qquad (21)$$

ここで、$\gamma^\dagger$ は独立パラメータだけを含み、次のように定義される。

$$\gamma^\dagger = \frac{1}{4E_{m^*g}^3}\Big(\sum_{i\neq g,m^*}\frac{E_{m^*g}\mu_{m^*i}^2}{E_{ig}} + \eta\bar{\mu}_{m^*m^*}^2\Big)^2 + \frac{\eta}{4}\sum_{i\neq g,m^*}\frac{\bar{\mu}_{m^*m^*}^2\mu_{m^*i}^2}{E_{m^*g}^2E_{ig}}, \tag{22}$$

$\Psi$ は従属パラメータの関数として与えるものと考え、

$$\Psi = \frac{1}{\gamma^\dagger}\Big\{\frac{\mu_{m^*g}^4}{E_{m^*g}^3}(2x^2-x^4) - \frac{\eta}{2}\sum_{i\neq g,m^*}\frac{\bar{\mu}_{m^*m^*}^2\mu_{m^*i}^2}{E_{m^*g}^2E_{ig}}\Big(\frac{x^2{y_i}^2}{2} - xy_i\Big)\Big\}, \tag{23}$$

ここで、$x$ と $y_i$ は各々次のように定義する。

$$\mu_{m^*g} = x\mu_{m^*g}^*,\ \mu_{ig} = y_i\frac{\bar{\mu}_{m^*m^*}\mu_{m^*i}}{2\mu_{m^*g}^*}, \tag{24}$$

$$\mu_{m^*g}^*{}^2 = \frac{1}{2}\Big(\sum_{i\neq g,m^*}\frac{E_{m^*g}}{E_{ig}}\mu_{m^*i}^2 + \eta\bar{\mu}_{m^*m^*}^2\Big). \tag{25}$$

$\Psi$ は $\mu_{m^*g}$ と $\mu_{ig}$ が両方とも $\gamma$ を最大化した時に最大値 (=1) となる関数である。$\Psi$ $\leq$1 であることから、この関数値は独立パラメータを固定した時に、従属パラメータが $\gamma$ の最大値にどの程度近いかを見る評価の指標となる。一方、$\gamma^\dagger$ は、従属パラメータが任意に変化したある時に於ける $\gamma$ の最大値を与える。これは $\Psi$=1 の場合に対応する。$\Psi$ の性質から、物質の種類に依存しない

$$\gamma \leq \gamma^\dagger \tag{26}$$

なる不等式を得る。物質設計原理は、独立パラメータと従属パラメータの 2 つの条件から得られる。

<u>独立パラメータに対して、</u>
$(I)$ $\gamma^\dagger$ を増加する,

<u>従属パラメータに対して、</u>
$(II)$ $\Psi$ を 1 に近づける, 即ち、$\mu_{m^*g}$ と $\mu_{ig}$ を式 (24) の $x$=1, $y_i$=1 が成り立つように動かすことである。

条件 $(I)$ を達成する方法は、式 (22) で $\gamma^\dagger$ を増加させることであるから、

$(i_1)$ $E_{jg}$ $(j \neq g)$ を減少させる,

$(i_2)$ $\mu_{m^*i}^2$ と $\bar{\mu}_{m^*m^*}^2$ $(i \neq g, m^*)$ を増加させる

ことである。ここで注意しなければならないのは、各々のパラメータが条件 $(I)$ に対して等価に変化しない場合や、$(i_1)$ と $(i_2)$ のような求められる条件が、独立パラメータの選び方に依存するような場合には、どのようにして条件 $(i_1)$ と $(i_2)$ が、最も、効率良く $\gamma^\dagger$ を増大させるかという点に注意しなければならない。

### 3.4 結果と考察

物質設計原理の観点から、実際の有機化合物の物質パラメータについて議論しよう。この議論のため、幾つかの有機分子の電子構造を半経験的分子軌道法 (CNDO/S-SDCI) で計算した。計算した数値を表 3-1 に示す。表中、$\bar{y}$ は式 (23) の $y_i$ の加重平均で、次のように定義する。

$$\sum_{i\neq g,m^*} \frac{\bar{\mu}_{m^*m^*2}\mu^2_{m^*i}}{E^2_{m^*g}E_{ig}}\left(\frac{x^2\bar{y}^2}{2} - x\bar{y}\right) = \sum_{i\neq g,m^*} \frac{\bar{\mu}_{m^*m^*2}\mu^2_{m^*i}}{E^2_{m^*g}E_{ig}}\left(\frac{x^2y_i^2}{2} - xy_i\right). \tag{27}$$

これから次のことが分かる:

$(a)$ 5 タイプ遷移パスモデルは $\gamma(-3\omega;\omega,\omega,\omega)$ を良く再現する。
$\gamma(-3\omega;\omega,\omega,\omega)$ の計算値は実験値と一致している。

$(b)$ 対称な分子と非対称な分子で、物質パラメータ値に顕著な違いがある。対称な有機分子 (1-5) は、$\Psi \simeq 1$, 同等に $x \simeq 1$ であるのに対し、電子供与性置換基や電子受容性置換基を導入して電子的に非対称な有機分子 (6-8) は、$\Psi \simeq 0.35$, $x \simeq 0.5$ で $y_i \simeq 0$ の値を持つ。これは、独立パラメータを固定した時に、対称な場合 (非対称な場合) の従属パラメータが、$\gamma$ に対する最大値に近い (遠い) ことを意味する。特に、対称な場合は $\Psi \simeq 1$ を満足するので、$\gamma$ は、式 (22) と (25) を用いて、

$$\gamma \simeq \frac{\mu^4_{m^*g}}{E^3_{m^*g}} \tag{28}$$

と書き直すことができ、簡単で実用的な物質設計原理が得られる。

$(c)$ 一方、式 (22) の $\gamma^\dagger$ は、対称な有機分子 (1-5) については小さい値であるが、非対称な有機分子 (6-8) に対しては大きい値を示す。これは、式 (25) の右辺第 1 項よりも大きな第 2 項が、非対称な有機分子では、対称な有機分子に比べ、大きいな値であることに由来する。

$(d)$ 非対称な有機分子 (9-10) の $\gamma^\dagger$, $\Psi$ は、対称な有機分子と同等な値を示す。これは、ベンゼン (4) に対して導入した置換基の電子的な効果があまり大きくないことによると考えられる。

$(e)$ 有機分子 (6-8) と分子 (9-10) についての $\Psi$ に対する $x$ と $\bar{y}$ の影響を比較すると、$x$ の $\Psi$ に対する効果の方が、$\bar{y}$ (または $y_i$) の $\Psi$ に対する効果よりも大きいことが分かる。分子 (9-10) は、$x$ のその最適値 $x$=1 に近づいているが、$\bar{y}$ の最適値 $\bar{y}$=1 からは遠く離れている。

上述した考察から、実存する対称および非対称な有機分子の物質設計原理 $(I)$ と $(II)$ は、図 3-3 の矢印で示す方向に物質パラメータを動かすことである。特に、非対称な分子は、従属パラメータの関係が設計原理 $(II)$ を満たすように変えれば、元々 $\gamma^\dagger$ が大きいので、$\gamma$ を向上させることが出来る。

参考及び文献

1) D. W. Hall and N. F. Borrelli, J. Opt. Soc. Am. B, 5 (1988) 1650.
2) T. Yoshimura, S. Tatsuura, and W. Satoyama, Appl. Phys. Lett. 60 (1992) 268.
3) J. A. Pople and D. L. Beveridge, *Approximate Molecular Orbital Theory* (McGraw-Hill, New York, 1971).
4) B. M. Pierce, SPIE 971 (1988) 25.
5) C. W. Dirk, L. Cheng, M. G. Kuzyk, Int. J. Quant. Chem., 43 (1992) 27.
6) A. F. Garito, J. R. Heflin, K. Y. Wong, and O. Zamani-Khamiri, in *Organic Materials for Non-linear Optics* ed. by R. A. Hann and D. Bloor (Royal Sciety of Chemistry, 1988) 16.
7) M. Nakano, M. Okamura, K. Yamaguchi, and T. Fueno, Mol. Cryt. Liquid, A182 (1990) 1.
8) R. W. Boyd, *Nonlinear Optics* (Academic press, 1992).

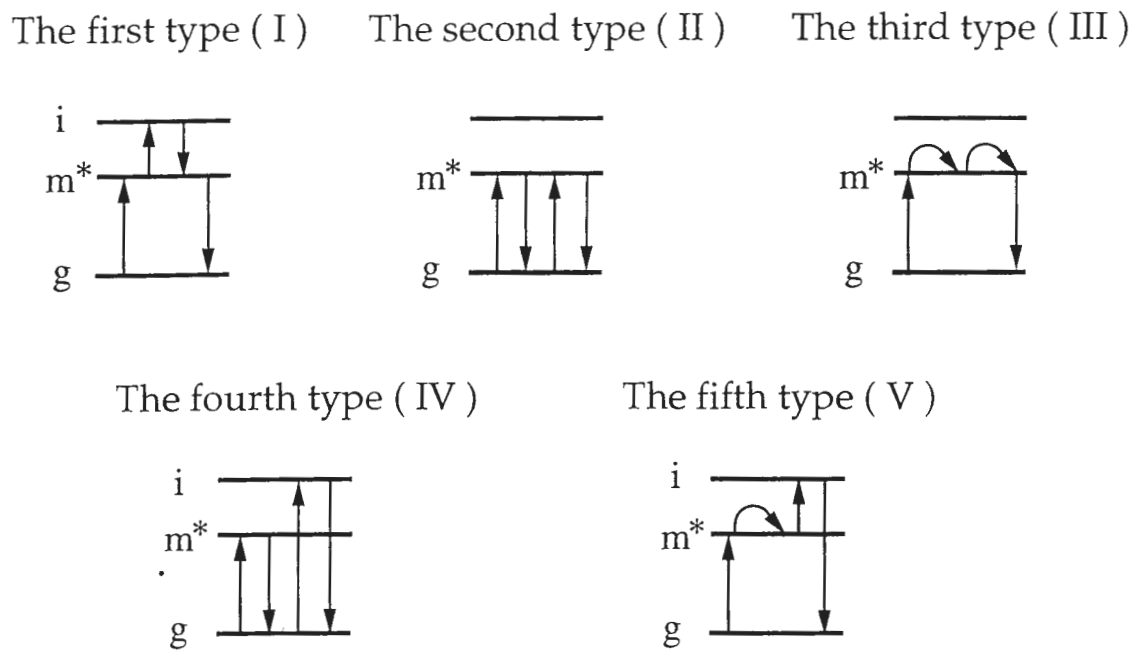



図 3-1. 3 次非線形感受率に支配的な 5 つの遷移パス。

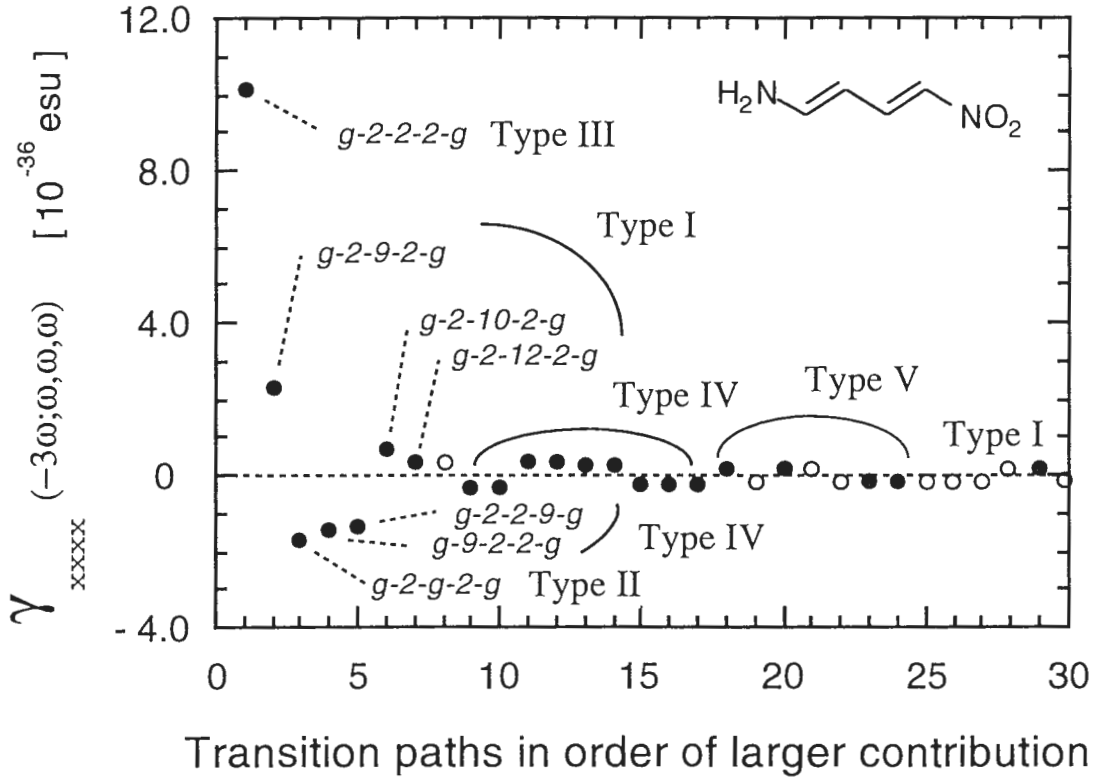



図 3-2. 非対称である 1- アミノ -4- ニトロ -1,3 ブタジエンの遷移パス成分。ここで、電子遷移パスは 2 次超分極率への寄与の大きな順番に並べた。モデルに含まれる 5 つのタイプ遷移パスは、黒丸 (●) で、モデルに含まれない遷移パスを白丸 (○) で表す。*g-m-n-v-g* の表記は基底状態 *g* から *m* 番目, *n* 番目, *v* 番目の励起状態を経由して、基底状態へ戻る遷移パスを意味する。

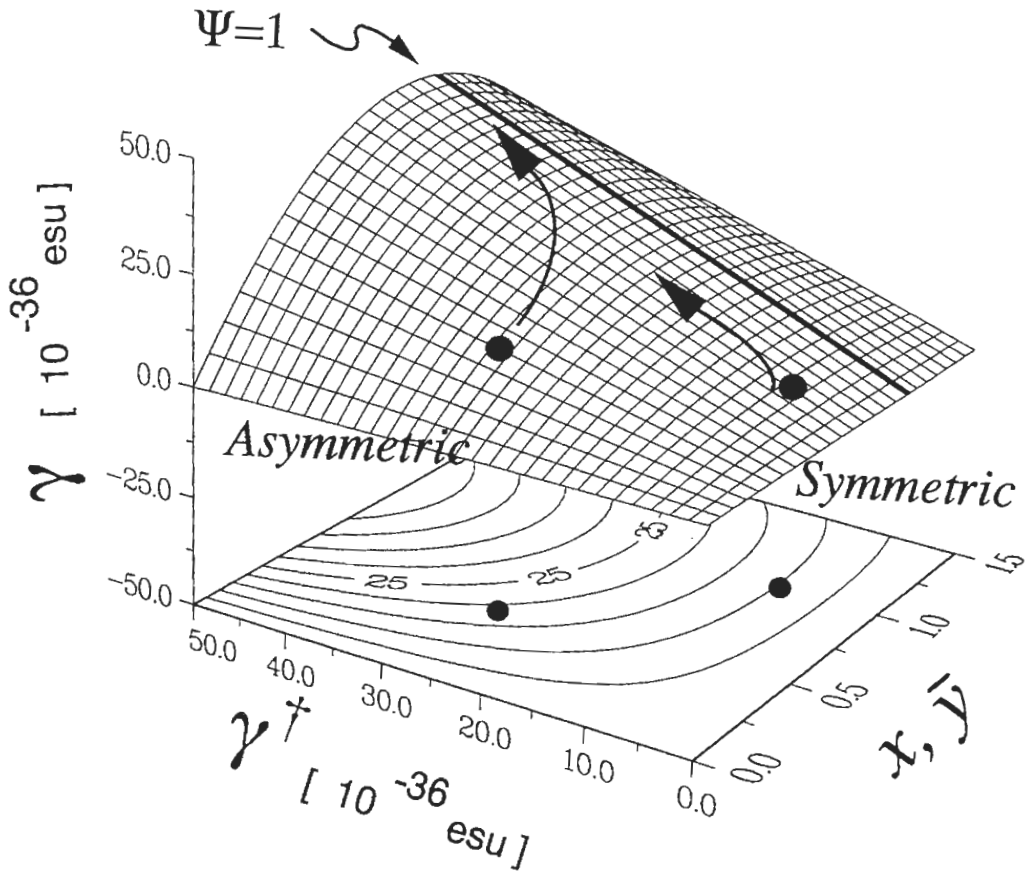
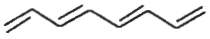
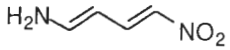
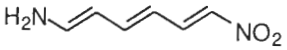
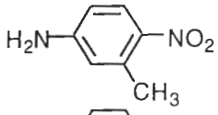
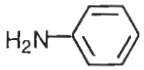
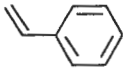



図 3-3. 実存する対称および非対称な有機分子に対する物質設計原理 $(I)$ および $(II)$。

表 3-1. 幾つかの対称および非対称有機分子の 2 次超分極率と物質パラメータの計算値。計算は $\hbar\omega = 0.65$ eV 時の、$\gamma_{ijkl}(-3\omega,\omega,\omega,\omega)$ の厳密な表式と、$\hbar\omega = 0$ 時の、式 (20) による 5 タイプ遷移パスモデルでの表式について行った。分子 1-5 は対称なので、従属パラメータとして $x$ だけを示す。

| | Compound | $\gamma_{xxxx}(-3\omega;\omega,\omega,\omega)$ [$10^{-36}$esu] | $\gamma$ Five types [$10^{-36}$esu] | $\gamma^{\dagger}$ [$10^{-36}$esu] | $\Psi$ | $x$ | $\bar{y}$ |
|---|---|---|---|---|---|---|---|
| 1 |  | 1.4 | 1.0 | 1.1 | 0.84 | 0.80 | — |
| 2 |  | 6.3 | 4.6 | 6.1 | 0.76 | 0.73 | — |
| 3 |  | 16.8 | 12.7 | 18.2 | 0.70 | 0.69 | — |
| 4 |  | 0.9 | 0.3 | 0.3 | 0.92 | 1.10 | — |
| 5 |  | 9.1 | 2.7 | 4.0 | 0.67 | 0.83 | — |
| 6 |  | 12.7 | 7.7 | 21.7 | 0.35 | 0.51 | -0.38 |
| 7 |  | 35.7 | 11.9 | 34.4 | 0.35 | 0.45 | -0.05 |
| 8 |  | 7.0 | 3.4 | 10.5 | 0.32 | 0.48 | -0.13 |
| 9 |  | 1.7 | 0.2 | 0.3 | 0.45 | 1.01 | -14.3 |
| 10 |  | 3.7 | 1.3 | 2.4 | 0.53 | 0.67 | -7.42 |

# 第 4 章 電子相関効果と分子設計論

3 次非線形感受率の微視的な起源である 2 次超分極率 ($\gamma$) を分子設計が容易となるような物質変数で記述し物質設計の方針を得ることは、3 次非線形光学材料を開発する上で重要である。2 次超分極率は、各仮想遷移経路ごとに、励起状態のエネルギー, 遷移双極子モーメント, 入射光エネルギーの関係から与えられる量の総和で与えられるが、全ての遷移経路を考慮したままでは、$\gamma$ は多くの物質変数で記述され、容易で実用的な分子設計指針を得ることはできない。そこで我々は、遷移経路を $\gamma$ に対して寄与の大きな幾つかの型に限定したモデルを用いて $\gamma$ の記述を簡易化すると共に、$\gamma$ を記述する物質変数をどのように変化させれば、$\gamma$ を増大することができるか検討してきた。ここでは、3 次非線形光学効果を持つ物質について分子設計のための物質変数と電子相関効果について検討する。

## 4.1 始めに

超分極率の計算における電子相関効果は、今までに幾つかの報告がある。Hurst らによるポリエンについての *ab-intio* 計算では、diffuse な基底関数が必要であると指摘しているが、拡がった基底関数を使っても実験値との差があり、この理由を電子相関によるものと考えている [1]。Perrin らはベンゼンの *ab-intio* 計算で Møller-Plesset 摂動論で電子相関を取り込み、$\gamma$ に対する電子相関効果が 3 割程度であることを示している [2]。一方で、比較的大きな分子を扱うために半経験的分子軌道法を用いて計算が行なわれている。Pierce は INDO (Intermediate Neglect of Differential Overlap) 法に $\pi$ 電子配置について 1 電子励起配置間相互作用 (SCI: single-excitation configuration interaction) と 1 電子および 2 電子励起配置間相互作用 (SDCI: single- and double-excitation configuration interaction) で電子相関を取り扱い、比較した [3]。SCI 法では $\gamma$ の正負さえも逆転する結果を与え、2 電子励起配置を含めることの重要性を示した。また、Heflin らはポリエンについて CNDO/S (Complete Neglect of Differential Overlap/Spectroscopic)-SDCI 法を用い、$\gamma$ に寄与する重要な励起状態を解析した結果、1 光子励起状態に加え、2 光子励起状態が重要であることを示した [4]。このように、2 次超分極率を議論するには、電子相関の効果を充分に取り入れなければならない。

本論文の主題は、直接に $\gamma$ に対する電子相関効果ではなく、分子設計のために簡易化した $\gamma$ を記述する物質変数に対して、電子相関効果の程度を知ることである。そのため、幾つかの対称および非対称を含めた有機分子について、CNDO/S-SDCI 法により電子相関を取り扱い、物質変数に与える影響について検討した。

## 4.2 分子設計

物質設計が可能になるようにモデル化するため、時間依存摂動論によって導かれる表式 [5] に現れる多数の遷移経路のうち、$g$-$m^*$-$i$-$m^*$-$g$、$g$-$m^*$-$g$-$m^*$-$g$、$g$-$m^*$-$m^*$-$m^*$-$g$、$g$-$m^*$-$g$-$i$-$g$、$g$-$m^*$-$m^*$-$i$-$g$ (ここで、$m^*$ は $\gamma$ に最も大きく寄与する遷移経路に含まれる一つの励起状態を、$i$ は他の全ての励起状態を表す: $i \neq m^*, g$) の 5 つの型の遷移経路だけが $\gamma$ に寄与するとし、入射振動数 $\omega \to 0$、および分子軸方向のテンソル成分だけを考えることとすると次のように書ける [6]:

$$\gamma = \frac{\mu_{m^*g}^2}{E_{m^*g}^2} \sum_{i \neq g, m^*} \frac{\bar{\mu}_{m^*i}^2}{E_{ig}} - \frac{\mu_{m^*g}^4}{E_{m^*g}^3} - \frac{\mu_{m^*g}^2}{E_{m^*g}^2} \sum_{i \neq g, m^*} \frac{\mu_{ig}^2}{E_{ig}}$$

$$+ \quad \eta\left\{\frac{\mu_{m^*g}^2\bar{\mu}_{m^*m^*}^2}{E_{m^*g}^3} + \frac{\mu_{m^*g}\bar{\mu}_{m^*m^*}}{E_{m^*g}^2}\sum_{i\neq g,m^*}\frac{\bar{\mu}_{mi}\mu_{ig}}{E_{ig}}\right\}. \tag{29}$$

ここで $\bar{\mu}_{mn}$=$\mu_{mn}$-$\mu_{gg}\delta_{mn}$、 $\eta$ は非対称な場合に $\eta$=1、 対称な場合に $\eta$=0 の値を取る2値関数である。

今、物質変数が自由に変化させられることを前提に、$\gamma$ を増大させるには、各物質変数をどの様にすれば良いかをこれらの変数空間の中で考える。ここで物質変数を $\gamma$ に対して、他の変数を不変とした時に最大値を持つ従属変数 $(\mu_{m^*g}, \mu_{ig})$ と最大値を持たない独立変数 $(E_{m^*g}, E_{ig}, \bar{\mu}_{m^*m^*}^2, \bar{\mu}_{m^*i}^2)$ の2種類に分類すると式(29)は

$$\gamma = \gamma^\dagger(E_{m^*g}, E_{ig}, \bar{\mu}_{m^*m^*}^2, \bar{\mu}_{m^*i}^2)\Psi(\mu_{m^*g}, \mu_{ig}) \tag{30}$$

と書ける。$\gamma^\dagger(E_{m^*g}, E_{ig}, \bar{\mu}_{m^*m^*}^2$、$\bar{\mu}_{m^*i}^2)$ は独立変数のみを含む関数で

$$\begin{aligned}\gamma^\dagger(E_{m^*g}, E_{ig}, \bar{\mu}_{m^*m^*}^2\bar{\mu}_{m^*i}^2) &= \frac{1}{4E_{m^*g}^3}\Big(\sum_{i\neq g,m^*}\frac{E_{m^*g}\bar{\mu}_{m^*i}^2}{E_{ig}} + \eta\bar{\mu}_{m^*m^*}^2\Big)^2 \\ &+ \frac{\eta}{4}\sum_{i\neq g,m^*}\frac{\bar{\mu}_{m^*m^*}^2\bar{\mu}_{m^*i}^2}{E_{m^*g}^2E_{ig}}\end{aligned} \tag{31}$$

である。一方、$\Psi(\mu_{m^*g}, \mu_{ig})$ は独立変数を含むが従属変数の関数と見なされ

$$\Psi(x, y_i) = \left\{\frac{2\mu_{m^*g}^{*4}}{E_{m^*g}^3}x^2 - \frac{\mu_{m^*g}^{*4}}{E_{m^*g}^3}x^4 - \frac{\eta}{2}\sum_{i\neq g,m^*}\frac{\bar{\mu}_{m^*m^*}^2\bar{\mu}_{m^*i}^2}{E_{m^*g}^2E_{ig}}\Big(\frac{x^2{y_i}^2}{2} - xy_i\Big)\right\}\times \frac{1}{\gamma^\dagger(E_{m^*g}, E_{ig}, \bar{\mu}_{m^*m^*}^2\bar{\mu}_{m^*i}^2)} \tag{32}$$

で与えられる。ここで $x$、$y_i$ $(i \neq m^*, g)$ は各々

$$\begin{aligned}\mu_{m^*g} &= x\mu_{m^*g}^*, \\ \mu_{ig} &= y_i\frac{\bar{\mu}_{m^*m^*}\bar{\mu}_{m^*i}}{2\mu_{m^*g}^*}, \\ \mu_{m^*g}^{*2} &= \frac{1}{2}\Big(\sum_{i\neq g,m^*}\frac{E_{m^*g}}{E_{ig}}\{\bar{\mu}_{m^*i}^2 + \eta\bar{\mu}_{m^*m^*}^2\}\Big)\end{aligned} \tag{33}$$

で定義される。

関数 $\Psi(x, y_i)$ は、従属変数 $(x, y_i)$ を動かした時の $\gamma$ の最大値にどの程度近いかを示す評価関数と考えられ、任意の $x, y_i$ に対して $\Psi(x, y_i) \leq 1$ であり、$\Psi(0, y_i)$=0 及び $\Psi(1, 1)$=1 なる性質を持っている。式(30)-(33)で $\gamma$ を記述することによって、$\gamma$ を増大させるためにはどの物質変数をどの様に変化させれば良いかが分かる。即ち、分子設計指針は

$(I)$ 独立変数に対して $\gamma^\dagger(E_{m^*g}, E_{ig}, \bar{\mu}_{m^*m^*}^2\bar{\mu}_{m^*i}^2)$ を増加させる、

$(II)$ 従属変数に対して $\Psi(x, y_i)$ (或いは $\Psi(\mu_{m^*g}, \mu_{ig})$) を1に近づける

ように各々を構成する物質変数 $(E_{m^*g}, E_{ig}, \bar{\mu}_{m^*m^*}^2, \bar{\mu}_{m^*i}^2, \mu_{m^*g}, \mu_{ig})$ を変化させることである。

### 4.3 分子設計のための物質変数と電子相関

2 次超分極率 ($\gamma$) は前節に述べた様に、物質変数の関数である $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ と $\Psi(x, y_i)$ の積で書けた。こららの関数にどの程度電子相関効果が影響するかについて調べるため、幾つかの対称および非対称を含めた有機分子について、 CNDO/S-SDCI 法を用いて、配置間相互作用 (CI) に含める励起配置の種類と数を変えることで、電子相関を取り入れる程度を変え物質変数に対する影響を調べた。

A. $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ への電子相関効果

配置間相互作用 (CI) は、 1 電子近似で求まった Hartree-Fock 軌道の最高被占軌道 (HOMO) と HOMO−n 軌道にわたる分子軌道から最低空軌道 (LUMO) と LUMO+n 軌道にわたる分子軌道への 1 電子及び 2 電子励起配置について、 CI 数を変えるために考慮する軌道数 n を 1 ～ 3 にとり、各々 2 × 2、 3 × 3、 4 × 4 の SDCI を行った。 $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ の CI 数に対する変化を図 4-1, 4-2 に示す。対称分子の場合、 CI 数の変化に対する $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ の変化は 30-40% 程度で、 CI 数の増加に伴なって $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ は減少する傾向にある。一方、非対称分子では、数百 % の変化し、 CI 数の増加に対して $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ は著しく増加する。

B. $\Psi(\mu_{m^*g}, \mu_{ig})$ への電子相関効果

同様に $\Psi(\mu_{m^*g}, \mu_{ig})$ の CI 数に対する変化を図 4-1, 4-2 から見ると、対称分子では、 CI 数の変化に対して $\Psi(\mu_{m^*g}, \mu_{ig})$ の変化は 20% 程度で、 CI 数の増加に伴なって $\Psi(\mu_{m^*g}, \mu_{ig})$ は減少する。一方、非対称分子では、 100% 近くの変化があり、 CI 数の増加に対して $\Psi(\mu_{m^*g}, \mu_{ig})$ は著しく減少する。

### 4.4 議論と結論

物質変数の関数である $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$、 $\Psi(\mu_{m^*g}, \mu_{ig})$ についての電子相関効果は、対称分子では影響が小さいのに対し、非対称分子では大きく影響した。非対称分子に対して大きく影響した理由は、主に ${\mu^*_{m^*g}}^2$ が大きく変化したことによる。 CI 数の増加に対して、 ${\mu^*_{m^*g}}^2$ が著しく大きくなることによって、 $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ が増加している。これは、 $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ を与えている式 (4) で右辺の第一項目の括弧内の左項が大きくなるためである。一方、 ${\mu^*_{m^*g}}^2$ の増加に比較して、 $\mu_{m^*g}$ の変化が小さいため、 $\Psi(\mu_{m^*g}, \mu_{ig})$ が減少した。このように非対称分子の場合には、 $\gamma^{\dagger}$ $(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$、 $\Psi(\mu_{m^*g}, \mu_{ig})$ の個々に対する電子相関効果は大きいが、 $\gamma$ については $\gamma$ が $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$ と $\Psi(\mu_{m^*g}, \mu_{ig})$ の積で書かれるために効果が小さくなっている。

物質設計への CI 効果は、対称分子に対してはある程度の CI を考慮することで $\gamma^{\dagger}(E_{m^*g}, E_{ig}, \bar{\mu}^2_{m^*m^*}\bar{\mu}^2_{m^*i})$、と $\Psi(\mu_{m^*g}, \mu_{ig})$ は収束しているようにように思われるが、非対称分子の場合には、これらの量は CI 効果をどの程度考えるか敏感であるように思われる。

参考及び文献

1) G. B. Hurst, M. Dupuis, E. Clementi, J. Chem. Phys., 89, 385 (1988).
2) E. Perrin, P. N. Prasad, P. Mougenot, M. Dupuis, J. Chem. Phys., 91, 8 (1989).
3) B. M. Pierce, in *Organic Materials for Non-linear Optics* ed. by R. A. Hann and D. Bloor (Royal Sciety of Chemistry, 1988) 48.
4) J. R. Heflin, K. Y. Wong, O. Zamani-Khamiri, A. F. Garito, Phys. Rev. B, 38, 1573 (1988).
5) R. W. Boyd, *Nonlinear Optics* (Academic press, 1992).
6) K. Ohtawara and K. Shinjo, Jpn. J. Appl. Phys., 34 Suppl. 34-1, 203 (1995).

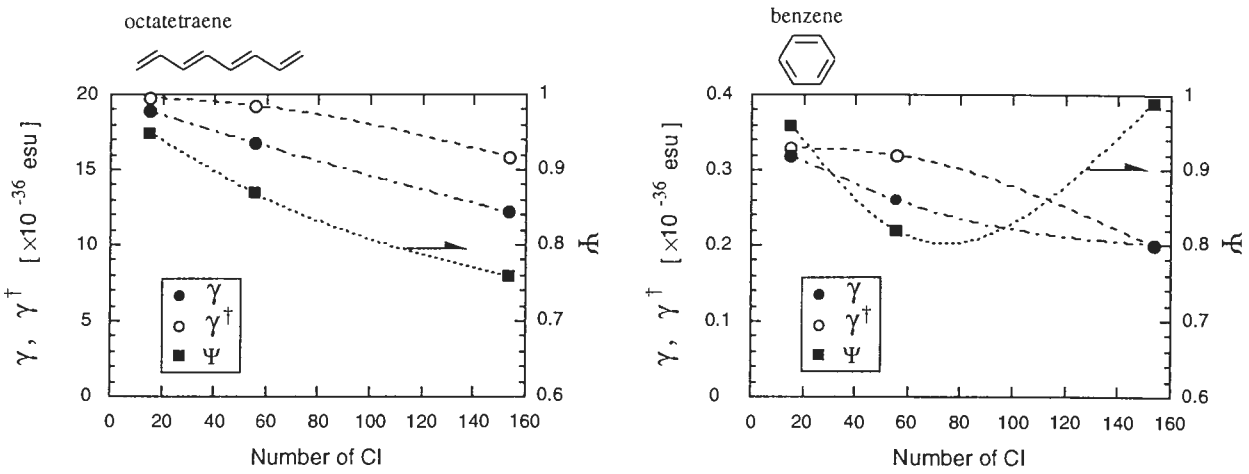



図 4-1. 対称な分子の電子相関効果。

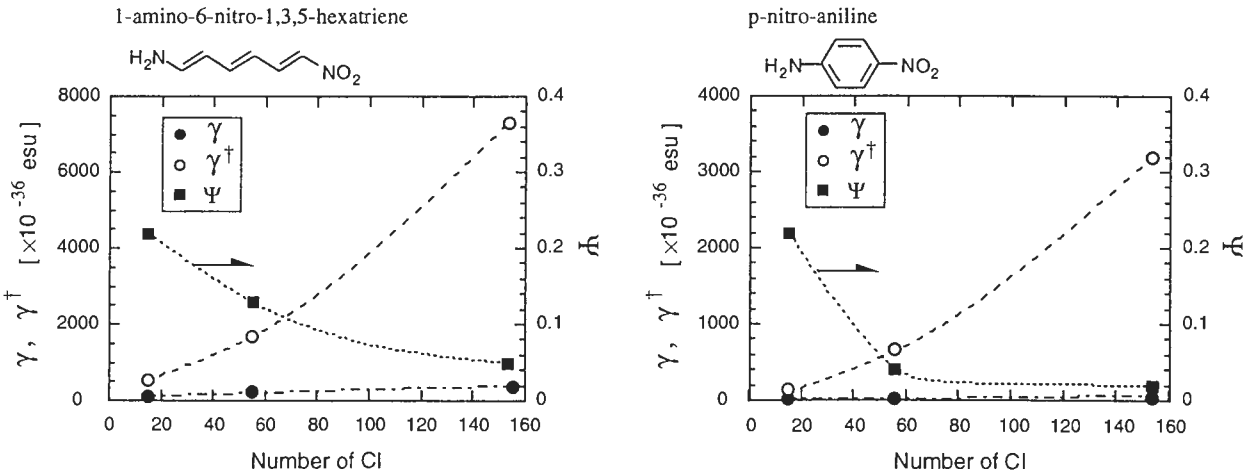



図 4-2. 非対称な分子の電子相関効果。

# 第 5 章 結論

超高速通信などの光デバイスに応用が期待されている 3 次非線形光学材料について、非線形性を向上させるための材料設計指針を提案するために、理論的研究を行った。新しい設計の方法論として、試行錯誤や知識, 経験を用いない方法論を検討し、一般的に存在する分解式を得た。そして、分解式の適用によって、3 次非線形性を支配している物質のパラメータを、どのように設計すれば良いかを示す設計原理を得た。また、設計原理に含まれる物質のパラメータを指標とした評価により、物質による特徴づけができ、対称な分子に対しては、簡略化され、より設計し易い設計原理を得た。一方、電子相関の検討を行い、評価に用いた物質パラメータが、ある程度の電子相関を考慮することで、大きく変化しないことを確認し、計算結果に対する信頼性を確保した。

今回、設計の第一段階として、物質 (システム) の変数をどのように動かせば良いかについて、仮想的な変数空間での答えを得ることができたが、具体的な 3 次非線形光学材料の提案には至っていない。従って、設計の第二段階として、その変数値を満たし、実在する具体的な物質 (システム) を求めるまたは探す手法が重要であり、現在、仮想空間の解を実空間に展開する方法を検討している。

今後、得られた知見が、3 次非線形光学材料の開発, 物質設計システムへの展開, システム設計への展開が期待できると思われる。

謝辞

本研究の遂行に当たりその機会を与えて頂いたエイ・ティ・アール光電波通信研究所、猪股英行社長、渡辺敏英室長に深謝いたします。また計算物理グループの方々に感謝いたします。

佐々田友平氏との議論は大変有意義なのもでありました。

## 付録 非線形感受率の計算プログラム

### A. 概要

有機分子についての2次超分極率の計算は、表A-1に示す計算工程で行った。ここでは、骨格部分であるCNDO/S-SDCI法による分子軌道計算プログラムと各励起状態のエネルギーおよび遷移モーメントから2次超分極率を計算するプログラムについて記述する。

表A-1. 非線形感受率計算プログラムと計算工程

| 処理内容 | プログラム名 a) | 計算機 | 出力ファイルと形式 |
|---|---|---|---|
| 化学構造式の描画,<br>2次元構造 → 3次元構造変換 | ChemDraw b)<br>Chem3D b) | Macintosh | ChemDraw 形式 (*)<br>内部座標形式 (*.dat) |
| mopac コマンドの付与 | chem2mop | Convex | mopac 入力 (*.dat) |
| 分子構造最適化 | MOPAC93 c) | Convex | mopac 出力 (*.out) |
| 分子構造および<br>分子軌道の確認 | Xmol d)<br>read_mopac e) | Sun<br>Convex | |
| 最適構造の抽出と<br>mopac コマンドの付与 | out2dat | Convex | mopac 入力 (*.dat) |
| Finate Field 法による2次超分極率<br>計算と分子軸中心への座標移動 | MOPAC93 c) | Convex | mopac 出力 (*.out) |
| CNDO/S 計算の入力データ生成 | out2mos | Convex | mopac 入力 (*.dat) |
| CNDO/S-SDCI 計算,<br>遷移モーメント計算 | mosf6 | Convex | 遷移双極子モーメントと<br>励起エネルギー (*.tr) |
| Sum of states 法による2次<br>超分極率と物質パラメータの計算 | g100 | Convex | 超分極率, 物質変数値 (*.g)<br>遷移パス解析結果 (*.g_type) |

a) 注釈の無いプログラムは、自作または改造を行ったプログラム
b) ChemDraw, Chem3D, Cambridge Scientific Computing, Inc., Massachusetts. (http://www.camsci.com)
c) MOPAC93[1], J. J. P. Stewart, Fujitsu Limited, Tokyo.
d) Xmol, Research Equipment Inc. and dba Minnesota Supercomputer Center, Inc. (http://www.msc.edu)
e) Application Visialization System, Convex Computer Corp. and Advanced Visual Systems Inc.

B. CNDO/S-SDCI 分子軌道計算プログラム

JCPE(日本化学プログラム交換機構)[2] により頒布されている半経験的分子軌道法 CNDO/S の計算プログラム MOS-F[3,4] に 1 電子および 2 電子励起配置の配置間相互作用 (SDCI) を含めたプログラムを以下の表式により作成した。

B1. 分子軌道法

分子の全電子エネルギーを最小にする最も良い 1 電子波動関数 ($\psi$), 即ち分子軌道 (MO) は、次の Hartree-Fock-Roothaan の方程 (HF 方程式) の解である。

$$\mathbf{F}\psi = \varepsilon\psi \tag{1.1}$$

分子軌道を原子軌道 ($\chi_r$, AO) の 1 次結合とし (LCAO 近似),

$$\psi_i = \sum_r C_{ir}\ \chi_r \tag{1.2}$$

これを (1.1) 式に入れると、各原子軌道に掛かる係数 ($C_{ir}$) は、次の永年方程式と規格化条件から求められる。

$$\begin{aligned} C_{11}(F_{11}-\varepsilon)+C_{12}(F_{12}-S_{12}\varepsilon)+C_{1n}(F_{1n}-S_{1n}\varepsilon)&=0\\ C_{21}(F_{21}-S_{21}\varepsilon)+C_{22}(F_{22}-\varepsilon)+C_{2n}(F_{2n}-S_{2n}\varepsilon)&=0\\ &\vdots\\ C_{n1}(F_{n1}-S_{n1}\varepsilon)+C_{n2}(F_{n2}-S_{n2}\varepsilon)+C_{nn}(F_{nn}-\varepsilon)&=0 \end{aligned} \tag{1.3}$$

行列で書くと、

$$\mathbf{FC} = \mathbf{SC}\varepsilon \tag{1.4}$$

規格化条件は、

$$\int \psi_i^2 d\tau = \sum_r \sum_s C_{ir}C_{is}S_{rs} = 1 \tag{1.5}$$

永年方程式の 1 つの解は、$C_{i1} = C_{i2} = \cdots = C_{in} = 0$ だが、これでは $\psi_i = 0$ となってしまう。そうならないためには、

$$\begin{bmatrix} F_{11}-\varepsilon & F_{12}-S_{12}\varepsilon & \cdots & F_{1n}-S_{1n}\varepsilon \\ F_{21}-S_{21}\varepsilon & F_{12}-\varepsilon & \cdots & F_{2n}-S_{2n}\varepsilon \\ \vdots & & & \\ F_{n1}-S_{n1}\varepsilon & F_{n2}-S_{n2}\varepsilon & \cdots & F_{nn}-\varepsilon \end{bmatrix} = 0 \tag{1.6}$$

とならないといけない。即ち,

$$Det\,[\mathbf{F}-\mathbf{S}\varepsilon] = 0 \tag{1.7}$$

(1.7) 式を解いて $n$ 個の $\varepsilon$を求め、これを(1.4)式に代入して各$C_{ir}$ の比を求め、(1.5) 式より各 $C_{ir}$ を求める。求めた各 $C_{ir}$ を用いて、再び $\mathbf{F}$ を計算し、新たな $\varepsilon$ を求めるのを $\varepsilon$ が収束するまで行ない (変分法)、目的の分子軌道 $\psi_i$ と分子軌道のエネルギー $\varepsilon_i$ を求める。

B2. CNDO/S 法

CNDO/S 法では HF 方程式を解く際に、以下の近似を行なう。

1) 原子軌道は価電子のみを考慮する。例えば、 H では $1s$ 軌道, C, N, O では $2s, 2p_x, 2p_y, 2p_z$ 軌道, Si では $3s, 3p_x, 3p_y, 3p_z$ を対象とする

2) 永年方程式中の重なり積分は

$$S_{rs} = \int \chi_r(1)\ \chi_s(2)\ d\tau = \delta_{rs} \tag{2.1}$$

$\delta_{rs}$ は Kronecker のデルタ $(\delta_{rs} = 1\ (r{=}s),\ \delta_{rs} = 0\ (r \neq s))$

3) Fock の Matrix element は

$$F_{rr} = H_{rr} - \frac{1}{2}P_{rr}\ (rr|rr) + \sum_{t \neq r} P_{tt}\ (rr|tt) \tag{2.2}$$

$$F_{rs} = H_{rs} - \frac{1}{2}P_{rs}\ (rr|ss) \qquad (r \neq s) \tag{2.3}$$

$P_{rs}$ は原子軌道 $r, s$ 間の結合次数で,

$$P_{rs} = \sum_{i}^{all} \nu_i C_{ir} C_{is} \tag{2.4}$$

$\nu_i$ は $i$ 番目の軌道にある電子数。従って, 被占軌道 (occ, 以下 $2n$ 電子系で $n$ と書く) までの和で,

$$P_{rs} = 2\sum_{i}^{occ} C_{ir} C_{is} \tag{2.5}$$

4) Core 積分の対角成分 (クーロン積分) は,

$$H_{rr} = \int \psi_r(1)\Big\{-\frac{h^2}{8\pi^2 m}\Delta + V(1)\Big\}\ \psi_r(1)\ d\tau \tag{2.6}$$

ここで, $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$, $V$ はポテンシャルエネルギー。
(2.6) 式を 1 中心と 2 中心に分けて書き,

$$H_{rr} = \int \psi_r(1)\Big\{-\frac{h^2}{8\pi^2 m}\Delta + V_A(1)\Big\}\ \psi_r(1)\ d\tau\ + \sum_{B(\neq A)} \psi_r(1) V_B(1) \psi_r(1)\ d\tau$$

$$= U_{rr} + \sum_{B(\neq A)} V_{AB} \tag{2.7}$$

$$U_{rr} = -\frac{1}{2}(I_r + A_r) - (Z_A - \frac{1}{2})\ \gamma_{AA}, \quad V_{AB} = -Z_B\ \gamma_{AB} \tag{2.8}$$

と近似する。 A,B は原子, $I_r$ はイオン化ポテンシャル, $A_r$ は電子親和力, $Z_B$ は原子核 B の核電荷から内殻電子数を引いた数, $\gamma$ は電子間反発積分。従って,

$$H_{rr} = -\frac{1}{2}(I_r + A_r) - (Z_A - \frac{1}{2})\ \gamma_{AA} - \sum_{B(\neq A)} Z_B\ \gamma_{AB} \tag{2.9}$$

5) Core 積分の非対角成分 (共鳴積分) は,

$$H_{rs} = \frac{1}{2}K\ S_{rs}(\beta_A^0 + \beta_B^0) \tag{2.10}$$

$K$ は Scaling Factor で、$\sigma$ 電子の場合 $K$=1.0, $\pi$ 電子の場合 $K$=0.585, $\beta^0$ は原子固有の parameter。$S_{rs}$ は重なり積分で、永年方程式中では $\delta_{rs}$ と近似したが、ここでは、*Slater* 型原子軌道を用いて計算する。($S_{rs}$=$\int \chi_r \chi_s d\tau$)

6) 電子間反発積分 $(rs|tu)$ は, 微分重なりを無視 (ZDO 近似) し $r$=$s$, $t$=$u$ 以外は 0 とする。

$$(rs|tu) = (rr|tt)\ \delta_{rs}\delta_{tu} \tag{2.11}$$

このうち,
a) 1 中心電子間反発積分†は,

$$(rr|rr) = \gamma_{rr} = \gamma_{AA} = I_r - A_r \qquad (\ r\ 軌道\ on\ A\ 原子\ ) \tag{2.12}$$

b) 2 中心電子間反発積分‡は、1 中心積分と原子間距離などから以下の計算方法で算出する。これらは選択できるようにする。

$$(rr|ss) = \gamma_{rs} = \gamma_{AB} \qquad (\ r\ 軌道\ on\ A\ 原子\ ,\ s\ 軌道\ on\ B\ 原子\ ) \tag{2.13}$$

*i*) 西本 - 又賀の式 (NM-$\gamma$)

$$(rr|ss) = \gamma_{rs} = \frac{e^2}{\frac{2e^2}{\gamma_{rr}+\gamma_{ss}} + R_{rs}} \tag{2.14}$$

ここで、$R$ は原子間距離 [Å], $e^2$=14.3986 [eV · Å] である。

*ii*) 大野 -Klopman の式 (OK-$\gamma$)

$$(rr|ss) = \gamma_{rs} = \frac{e^2}{\sqrt{R_{rs}^2 + (\frac{\gamma_{rr}+\gamma_{ss}}{2})^2}} \tag{2.15}$$

以上のようにすると、Fock の Matrix element は,

$$F_{rr} = -\frac{1}{2}(I_r + A_r) + \left[(P_{AA} - Z_A) - \frac{1}{2}(P_{rr} - 1)\right]\gamma_{AA} + \sum_{B(\neq A)}(P_{BB} - Z_B)\ \gamma_{AB} \tag{2.16}$$

$$F_{rs} = \frac{1}{2}K\ S_{rs}(\beta_A^0 + \beta_B^0) - \frac{1}{2}P_{rs}\ \gamma_{AB} \qquad (r \neq s) \tag{2.17}$$

†CNDO/2 法では、主量子数 2 の場合、$(rr|rr)$ を原子軌道 $r$ の属する原子 $A$ の 2$s$ 軌道とおき、$(s_A s_A|s_A s_A)$ を非経験的に *Slater* 関数から計算するが、CNDO/S 法では $I_r - A_r$ を使う。

‡CNDO/2 法では、主量子数 2 の場合、$(rr|ss)$ を原子軌道 $r$ の属する原子 $A$ の 2$s$ 軌道, $s$ の属する原子 $B$ の 2$s$ 軌道とおき、$(s_A s_A|s_B s_B)$ を非経験的、即ち *Slater* 関数から計算するが、CNDO/S 法では次の方法のいずれかを使う。MOS-F には、sgint.f に次の 7 方法が組み込まれている。1) Analytic, 2) Pariser-Parr(帯電球近似), 3) Nishimoto-Mataga, 4) Nishimoto-Mataga-Weiss, 5) Ohno, 6) Ohno-Klopman, 7) Dasgupta-Huzinaga これらは例えば、*i*), *ii*) で記述される。

ここで,$P_{AA}$ は原子 A についての結合次数 ($P_{rs}$) 総和, 即ち, 電子密度で,

$$P_{AA} = \sum_{r}^{on A} P_{rr} \tag{2.18}$$

7) 重なり積分は *Slater* 軌道を用いて解析的に計算する [5,6]。

8) 全エネルギーは次のように計算する。

a) 全電子エネルギー

$$\begin{aligned} E_{elec} &= \sum_{i}^{n} \varepsilon_i + \frac{1}{2} \sum_{r,s} P_{rs} H_{rs} \\ &= \frac{1}{2} \sum_{r,s} P_{rs} (H_{rs} + F_{rs}) \end{aligned} \tag{2.19}$$

b) 全核エネルギー

$$E_{core} = \sum_{A>B}^{n} \frac{Z_A Z_B}{R_{AB}} \tag{2.20}$$

c) 全エネルギー

$$E_{total} = E_{elec} + E_{core} \tag{2.21}$$

以上、§1 ～§2 に記述した部分は、 MOS-F に組み込まれている。

B3. 配置間相互作用 (CI)

前節のようにして求まった分子軌道をもとに、配置間相互作用の方法で電子相関を取り入れる。まず、 HF 法で求まった電子配置 $\Psi_0$ をもとに、いろいろな電子配置の各励起配置を作る。そして、求める全波動関数 $\Psi$ は、これらの電子配置の線形結合からなるものとして考え、前節で分子軌道を求めたのと同様の方法 (変分法) で、これら電子配置の線形結合の係数を求める。

全波動関数は各電子配置の線形結合で与えられ、

$$\Psi = \sum_{i} \lambda_i \Psi_i \tag{3.1}$$

求める $\lambda_i$ は次の固有値問題を解いて得られる。

$$(\mathbf{H} - E\mathbf{1})\lambda = 0 \tag{3.2}$$

ここで $E$ は求める全波動関数のエネルギー, $\mathbf{1}$ は単位行列, $\lambda$ は求める係数 $\lambda_i$ の縦ベクトル, $\mathbf{H}$ はハミルトニアン行列で

$$H_{ij} = \int \Psi_i^* H \Psi_j d\tau = \langle \Psi_i | H | \Psi_j \rangle \tag{3.3}$$

ここで、各行列要素 $\langle \Psi_i | H | \Psi_j \rangle$ は次ページ以降に記述する。

[ 1 ] 配置の種類

SDCI 法で取り扱う配置は基底配置, 1 電子励起配置, および 2 電子励起配置である。以下、取り扱う配置は一重項状態¶だけを考える。この場合、基底配置はただ 1 つ, 1 電子配置は 1 タイプ, 2 電子配置は 5 タイプがある。ここで言う配置とは、電子の詰め方である。

a) 基底配置

$2n$ 個の電子を持つ分子について求めた分子軌道は、基底配置そのものであり、各軌道にエネルギーの低い順に各々 2 個の電子が入った $n$ 個の被占軌道‖と電子の入っていない $n$ 個の空軌道からなる。

b) 1 電子励起配置

被占軌道から空軌道へ電子が 1 つ上がった状態である。励起の仕方は 1 タイプしかないが、全ての 1 電子励起を HOMO, LUMO を基準に考えると、

HOMO→LUMO, HOMO→LUMO+1, HOMO→LUMO+2, $\cdots$, HOMO→LUMO+$n$-1,
HOMO−1→LUMO, HOMO−1→LUMO+1, $\cdots$, HOMO−1→LUMO+$n$-1,

$\vdots$

HOMO−$n$+1 →LUMO, HOMO−n+1→LUMO+1, $\cdots$, HOMO−$n$+1 →LUMO+$n$-1

だけある。これらの配置を $\Psi_{i\to k}$ と表記する。1 電子励起の中だけでも全ての配置を取り入れると、計算が膨大になるため、プログラム中では、これらの $i, k$ が 個別に指定できるようにし、例えば, 全電子が 10 個で HOMO が 5 番目の軌道の場合に、被占軌道の 3,5 番目と空軌道の 6,7 番目から、 3→6, 3→7, 5→6, 5→7 を発生するように個々に 1 電子励起配置に取り入れる軌道を指定する。

c) 2 電子励起配置

被占軌道から空軌道へ電子が 2 つ上がった状態である。励起の仕方は次の 5 タイプある。先程と同様な表記で書くと、

Type 1: $\Psi_{\substack{i\to k\\ i\to k}}$ $i$ 番目の被占軌道から $k$ 番目の空軌道へ 1 つの電子が、$i$ 番目の被占軌道から $k$ 番目の空軌道へもう 1 つの電子が励起する配置。

Type 2: $\Psi_{\substack{i\to k\\ j\to k}}$ $i$ から $k$ へ, $j$ から $k$ へ 2 電子励起する配置。

Type 3: $\Psi_{\substack{i\to k\\ i\to l}}$ $i$ から $k$ へ, $i$ から $l$ へ 2 電子励起する配置。

Type 4: $\Psi^{a}_{\substack{i\to k\\ j\to l}}$ $i$ から $k$ へ, $j$ から $l$ へ 2 電子励起する配置。スピン配置が Type 5 と正反対。

Type 5: $\Psi^{b}_{\substack{i\to k\\ j\to l}}$ $i$ から $k$ へ, $j$ から $l$ へ 2 電子励起する配置。スピン配置が Type 4 と正反対。

1 電子励起配置と同様に、プログラム中では、これらの $i, j, k, l$ が個別指定ができるようにする。

¶一重項状態とは電子スピンが反平行な状態で、三重項はスピン平行な状態である

‖電子の詰まっている一番上 (エネルギーが) の軌道を最高被占軌道 (Highest Occupied Molecular Orbital: HOMO), 電子の詰まっていない一番下の軌道を最低空軌道 (Lowest Unccupied Molecular Orbital: LUMO) と呼ぶ

[ 2 ] 行列要素

式 (3.3) の行列要素を以下に記述する。

1) 対角項 $\langle\Psi_i|H|\Psi_i\rangle$

a) 基底配置 $(\Psi_0)$

$$\langle\Psi_0|H|\Psi_0\rangle = E_0 = 2\sum_i^n H_i + \sum_{i,j}^n (2J_{ij} - K_{ij}) \tag{3.4}$$

b) 1 電子励起配置 $(\Psi_{i\to k})$

$$\langle\Psi_{i\to k}|H|\Psi_{i\to k}\rangle = E_{i\to k} = E_0 + F_{kk} - F_{ii} - J_{ik} + 2K_{ik} \tag{3.5}$$

c) 2 電子励起配置 $(\Psi_{\substack{i\to k\\ j\to l}})$

$$\langle\Psi_{\substack{i\to k\\ i\to k}}|H|\Psi_{\substack{i\to k\\ i\to k}}\rangle = E_0 + 2E_{i\to k} - 2J_{ik} + J_{ii} + J_{kk} - 2K_{ik} \tag{3.6}$$

$$\langle\Psi_{\substack{i\to k\\ j\to k}}|H|\Psi_{\substack{i\to k\\ j\to k}}\rangle = E_0 + E_{i\to k} + E_{j\to k} - J_{ik} - J_{jk} + J_{kk} + J_{ij} - K_{ik} - K_{jk} + K_{ij} \tag{3.7}$$

$$\langle\Psi_{\substack{i\to k\\ i\to l}}|H|\Psi_{\substack{i\to k\\ i\to l}}\rangle = E_0 + E_{i\to k} + E_{i\to l} - J_{ik} - J_{il} + J_{ii} + J_{kl} - K_{ik} - K_{il} + K_{kl} \tag{3.8}$$

$$\langle\Psi^a_{\substack{i\to k\\ j\to l}}|H|\Psi^a_{\substack{i\to k\\ j\to l}}\rangle = E_0 + E_{i\to k} + E_{j\to l} - J_{kj} - J_{li} + J_{kl} + J_{ij} - \frac{3}{2}(K_{ik} + K_{jl}) + \frac{1}{2}(K_{kj} + K_{li}) + (K_{kl} + K_{ij}) \tag{3.9}$$

$$\langle\Psi^b_{\substack{i\to k\\ j\to l}}|H|\Psi^b_{\substack{i\to k\\ j\to l}}\rangle = E_0 + E_{i\to k} + E_{j\to l} - J_{kj} - J_{li} + J_{kl} + J_{ij} + \frac{3}{2}(K_{ik} + K_{jl}) - \frac{1}{2}(K_{kj} + K_{li}) - (K_{kl} + K_{ij}) \tag{3.10}$$

2) 非対角項 $\langle\Psi_i|H|\Psi_j\rangle$

a) 基底配置と 1 電子励起配置

$$\langle\Psi_0|H|\Psi_{i\to k}\rangle = \sqrt{2}\ F_{ik} \tag{3.11}$$

b) 基底配置と 2 電子励起配置

$$\langle\Psi_0|H|\Psi_{\substack{i\to k\\ i\to k}}\rangle = K_{ki} \tag{3.12}$$

$$\langle\Psi_0|H|\Psi_{\substack{i\to k\\ j\to k}}\rangle = -\sqrt{2}\ (kj|ki) \tag{3.13}$$

$$\langle\Psi_0|H|\Psi_{\substack{i\to k\\ i\to l}}\rangle = \sqrt{2}\ (ki|li) \tag{3.14}$$

$$\langle\Psi_0|H|\Psi^a_{\substack{i\to k\\ j\to l}}\rangle = -\{(ki|lj) + (kj|li)\} \tag{3.15}$$

$$\langle\Psi_0|H|\Psi^b_{\substack{i\to k\\ j\to l}}\rangle = \sqrt{3}\ \{(kj|li) - (ki|lj)\} \tag{3.16}$$

c) 1 電子励起配置間

$$\langle\Psi_{i\to k}|H|\Psi_{r\to t}\rangle = \delta_{ir}F_{kt} - \delta_{kt}F_{ir} + 2(ik|rt) - (ir|kt) \tag{3.17}$$

d) 1 電子励起配置と 2 電子励起配置

$$\langle\Psi_{i\to k}|H|\Psi_{\substack{r\to t\\ r\to t}}\rangle = \sqrt{2}\ \{\delta_{ri}\delta_{tk}F_{rt} - \delta_{tk}(ir|tr) + \delta_{ri}(rt|kt)\} \tag{3.18}$$

$$\langle\Psi_{i\to k}|H|\Psi_{\substack{r\to t\\ s\to t}}\rangle = -\Big\{\delta_{kt}\delta_{ir}F_{st} + \delta_{ri}(st|kt) - \delta_{tk}\{(st|ri) + (rt|si)\}\Big\} \tag{3.19}$$

$$\langle\Psi_{i\to k}|H|\Psi_{\substack{r\to t\\ r\to u}}\rangle = \delta_{kt}\delta_{ri}F_{ru} + \delta_{ri}\{(rt|ku) + (kt|ru)\} - \delta_{tk}(ri|ru) \tag{3.20}$$

$$\langle\Psi_{i\to k}|H|\Psi^a_{\substack{r\to t\\ s\to u}}\rangle = -\sqrt{\frac{1}{2}}\ \Big\{\delta_{ri}\delta_{tk}F_{su} - \delta_{tk}\{(ri|su) + (si|ru)\} + \delta_{ri}\{(st|ku) + (kt|su)\}\Big\} \tag{3.21}$$

$$\langle\Psi_{i\to k}|H|\Psi^b_{\substack{r\to t\\ s\to u}}\rangle = -\sqrt{\frac{3}{2}}\ \Big\{\delta_{ri}\delta_{tk}F_{su} - \delta_{tk}\{(ri|su) - (si|ru)\} - \delta_{ri}\{(st|ku) - (kt|su)\}\Big\} \tag{3.22}$$

e) 2 電子励起配置間

$$\langle\Psi_{\substack{i\to k\\ i\to k}}|H|\Psi_{\substack{r\to t\\ r\to t}}\rangle = \delta_{ir}(kt|kt) + \delta_{kt}(ir|ir) \tag{3.23}$$

$$\langle\Psi_{\substack{i\to k\\ i\to k}}|H|\Psi_{\substack{r\to t\\ s\to t}}\rangle = \sqrt{2}\ \delta_{kt}\ \Big\{\delta_{ri}\ \Big\{F_{rs} + 2(tt|rs) - (ts|tr)\} - (ir|is)\ \Big\} \tag{3.24}$$

$$\langle\Psi_{\substack{i\to k\\ i\to k}}|H|\Psi_{\substack{r\to t\\ r\to u}}\rangle = \sqrt{2}\ \delta_{ri}\ \Big\{\delta_{ku}\ \Big\{F_{tu} - 2(rr|tu) + (ru|rt)\} + (kt|ku)\ \Big\} \tag{3.25}$$

$$\langle\Psi_{\substack{i\to k\\ i\to k}}|H|\Psi^a_{\substack{r\to t\\ s\to u}}\rangle = \delta_{ri}\delta_{ku}\ \Big\{2(ut|sr) - (ur|st)\ \Big\} \tag{3.26}$$

$$\langle\Psi_{\substack{i\to k\\ i\to k}}|H|\Psi^b_{\substack{r\to t\\ s\to u}}\rangle = \sqrt{3}\ \delta_{ri}\delta_{ku}(ur|st) \tag{3.27}$$

$$\langle\Psi_{\substack{i\to k\\ j\to k}}|H|\Psi_{\substack{r\to t\\ s\to t}}\rangle = \delta_{kt}\ \Big\{\delta_{ri}\ \{-F_{sj} - 2(tt|sj) + (tj|ts)\} + (ir|js) + (is|jr)\} + \delta_{ri}\delta_{sj}(kt|kt) \tag{3.28}$$

$$\langle\Psi_{\substack{i\to k\\ j\to k}}|H|\Psi_{\substack{r\to t\\ r\to u}}\rangle = \delta_{uk}\ \Big\{\delta_{ri}\ \{2(rj|ut) - (rt|uj)\} + \delta_{rj}\{2(ri|ut) - (rt|ui)\}\Big\} \tag{3.29}$$

$$\langle\Psi_{\substack{i\to k\\ j\to k}}|H|\Psi^a_{\substack{r\to t\\ s\to u}}\rangle = \sqrt{2}\,\delta_{si}\ \Big\{\delta_{rj}\{(kt|ku) + \delta_{uk}\{F_{ut} - (ss|ut) + \frac{1}{2}(st|su)\}\Big\} + \delta_{uk}\{\frac{1}{2}(rt|ju) - (rj|ut)\}\Big\} \tag{3.30}$$

$$\langle\Psi_{\substack{i\to k\\ j\to k}}|H|\Psi^b_{\substack{r\to t\\ s\to u}}\rangle = -\frac{\sqrt{3}}{2}\ \delta_{ri}\delta_{uk}\ \Big\{(st|uj) + \delta_{sj}\{\ (\sqrt{2}-1)(st|su) - \sqrt{2}\ (rt|ru)\}\Big\} \tag{3.31}$$

$$\langle\Psi_{\substack{i\to k\\ i\to l}}|H|\Psi_{\substack{r\to t\\ r\to u}}\rangle = \delta_{ri}\ \Big\{\delta_{tk}\ \{F_{lu} - 2(rr|lu) + (ru|rl)\} + (ku|lt) + (kt|ul)\} + \delta_{tk}\delta_{ul}(ir|ir) \tag{3.32}$$

$$\langle\Psi_{\substack{i\to k\\ i\to l}}|H|\Psi^a_{\substack{r\to t\\ s\to u}}\rangle = \sqrt{2}\,\delta_{tk}\ \Big\{\delta_{ri}\{(rs|lu) - \frac{1}{2}(su|lr) + \delta_{ul}\{F_{rs} + (tt|rs) - \frac{1}{2}(ts|tr)\}\} - \delta_{ul}(ri|si)\Big\} \tag{3.33}$$

$$\langle\Psi_{\substack{i\to k\\ i\to l}}|H|\Psi^b_{\substack{r\to t\\ s\to u}}\rangle = -\sqrt{\frac{3}{2}}\ \delta_{ri}\delta_{tk}\ \Big\{(su|lr) + \delta_{ul}\{\ (\sqrt{2}-1)(su|ru) - \sqrt{2}(tr|ts)\}\Big\} \tag{3.34}$$

$$\langle\Psi^a_{\substack{i\to k\\ j\to l}}|H|\Psi^a_{\substack{r\to t\\ s\to u}}\rangle = \delta_{ri}\delta_{sj}\ \Big\{\delta_{tk}\{F_{lu} - (rr|lu) + (1 - \frac{1}{2})(ru|rl)\} + (kt|lu) + (ku|lt)\}$$
$$+\delta_{tk}\delta_{ul}\{\delta_{ri}\{-F_{sj} - (tt|sj) + (1 - \frac{1}{2})(tj|ts)\} + (ri|sj) + (rj|si)\} + \delta_{tk}\delta_{ri}\{\ (1 - \frac{1}{2})(su|lj) - (sj|lu)\Big\} \tag{3.35}$$

$$\langle\Psi^a_{\substack{i\to k\\ j\to l}}|H|\Psi^b_{\substack{r\to t\\ s\to u}}\rangle = \sqrt{\frac{3}{2}}\ \delta_{ri}\delta_{tk}\{(su|lj) - \delta_{sj}(ru|rl) - \delta_{ul}(st|tj) + \delta_{sj}\delta_{ul}(rt|rt)\} \tag{3.36}$$

$$\langle\Psi^b_{\substack{i\to k\\ j\to l}}|H|\Psi^b_{\substack{r\to t\\ s\to u}}\rangle = \delta_{ri}\delta_{sj}\ \Big\{\delta_{tk}\{F_{lu} - (rr|lu) + (1 + \frac{1}{2})(ru|rl)\} + (kt|lu) - (ku|lt)\}$$
$$+\delta_{tk}\delta_{ul}\{\delta_{ri}\{-F_{sj} - (tt|sj) + (1 + \frac{1}{2})(tj|ts)\} + (ri|sj) - (rj|si)\} + \delta_{tk}\delta_{ri}\{\ (1 + \frac{1}{2})(su|lj) - (sj|lu)\Big\} \tag{3.37}$$

ここで、

$$F_{ij} = H_{ij} + \sum_{p}^{n} \{2\,(pp|ij) - (pi|pj)\} \tag{3.38}$$

$$J_{ij} = (ii|jj) \tag{3.39}$$

$$K_{ij} = (ij|ij) \tag{3.40}$$

また、2電子励起配置のうち、$\langle \Psi_{\substack{i\to k\\ j\to l}} | H | \Psi_{\substack{r\to t\\ s\to u}} \rangle$ の $i, j, k, l$ および $r, s, t, u$ には、$\Psi_{\substack{i\to k\\ j\to l}}$ に対し $\Psi_{\substack{j\to l\\ i\to k}}$, $\Psi_{\substack{r\to t\\ s\to u}}$ に対し $\Psi_{\substack{s\to u\\ r\to t}}$ があるが、これらは、各式中にある $\delta_{ij}$ などが残る (=1) ように決める。

## B4. 遷移モーメントの計算

電子スペクトルの吸収波長は、2つの電子状態間のエネルギー差で与えられる。一方、吸収強度は、電子状態間の遷移確率で与えられ、遷移確率は、遷移モーメントの2乗に比例する。遷移モーメント ($\mu_{mn}$) は、状態 $m$ と $n$ の間の双極子モーメントに相当する量で、$\mathbf{r}$ を電子の位置ベクトル, $e$ を電子の電荷とすると次式で表される。

$$\mu_{mn} = e \int \Psi_m \sum_i \mathbf{r}_i \Psi_n d\tau \tag{4.1}$$

各状態間 ($\Psi_m \to \Psi_n$) の遷移モーメントは次の様に、1電子遷移の遷移モーメントに帰して求められる。

1) 基底配置間 (即ち、双極子モーメント)

$$\langle \Psi_0 | M | \Psi_0 \rangle = L \tag{4.2}$$

2) 基底配置と1電子励起配置間

$$\langle \Psi_0 | M | \Psi_{i\to k} \rangle = \sqrt{2}\, m_{ik} \tag{4.3}$$

3) 基底配置と2電子励起配置間

$$\langle \Psi_0 | M | \Psi^{i\to k}_{j\to l} \rangle = 0 \tag{4.4}$$

4) 1電子励起配置間

$$\langle \Psi_{i\to k} | M | \Psi_{r\to t} \rangle = \delta_{ri}\delta_{tk} - \delta_{tk} m_{ri} + \delta_{ri} m_{tk} \tag{4.5}$$

5) 1電子励起配置と2電子励起配置間

$$\langle \Psi_{i\to k} | M | \Psi^{r\to t}_{r\to t} \rangle = \sqrt{2}\, \delta_{ri}\delta_{tk} m_{rt} \tag{4.6}$$

$$\langle \Psi_{i\to k} | M | \Psi^{r\to t}_{s\to t} \rangle = -\delta_{ri}\delta_{tk} m_{st} \tag{4.7}$$

$$\langle \Psi_{i\to k} | M | \Psi^{r\to t}_{r\to u} \rangle = \delta_{ri}\delta_{uk} m_{rt} \tag{4.8}$$

$$\langle \Psi_{i\to k} | M | \Psi^{a\ \ r\to t}_{\ \ s\to u} \rangle = -\sqrt{\frac{1}{2}}\, \delta_{ri}\delta_{tk} m_{su} \tag{4.9}$$

$$\langle \Psi_{i\to k} | M | \Psi^{b\ \ r\to t}_{\ \ s\to u} \rangle = -\sqrt{\frac{3}{2}}\, \delta_{ri}\delta_{tk} m_{su} \tag{4.10}$$

6) 2 電子励起配置間

$$\langle\Psi_{i\to k}^{i\to k}|M|\Psi_{r\to t}^{r\to t}\rangle = \delta_{ri}\delta_{tk}(L+2m_{tt}-2m_{rr}) \tag{4.11}$$

$$\langle\Psi_{i\to k}^{i\to k}|M|\Psi_{s\to t}^{r\to t}\rangle = \sqrt{2}\ \delta_{ri}\delta_{tk}m_{rs} \tag{4.12}$$

$$\langle\Psi_{i\to k}^{i\to k}|M|\Psi_{r\to u}^{r\to t}\rangle = \sqrt{2}\ \delta_{ri}\delta_{tk}m_{tu} \tag{4.13}$$

$$\langle\Psi_{i\to k}^{i\to k}|M|\Psi^{a}{}_{s\to u}^{r\to t}\rangle = 0 \tag{4.14}$$

$$\langle\Psi_{i\to k}^{i\to k}|M|\Psi^{b}{}_{s\to u}^{r\to t}\rangle = 0 \tag{4.15}$$

$$\langle\Psi_{j\to k}^{i\to k}|M|\Psi_{s\to t}^{r\to t}\rangle = \delta_{ri}\delta_{tk}\{\delta_{sj}(L+2m_{tt}-m_{rr})-m_{sj}\} \tag{4.16}$$

$$\langle\Psi_{j\to k}^{i\to k}|M|\Psi_{r\to u}^{r\to t}\rangle = 0 \tag{4.17}$$

$$\langle\Psi_{j\to k}^{i\to k}|M|\Psi^{a}{}_{s\to u}^{r\to t}\rangle = \sqrt{2}\ \delta_{ri}\delta_{sj}\delta_{tk}m_{tu} \tag{4.18}$$

$$\langle\Psi_{j\to k}^{i\to k}|M|\Psi^{b}{}_{s\to u}^{r\to t}\rangle = 0 \tag{4.19}$$

$$\langle\Psi_{i\to l}^{i\to k}|M|\Psi_{r\to u}^{r\to t}\rangle = \delta_{ri}\delta_{tk}\{\delta_{ul}(L+m_{tt}-2m_{rr})+m_{lu}\} \tag{4.20}$$

$$\langle\Psi_{i\to l}^{i\to k}|M|\Psi^{a}{}_{s\to u}^{r\to t}\rangle = \sqrt{2}\ \delta_{ri}\delta_{tk}\delta_{ul}m_{rs} \tag{4.21}$$

$$\langle\Psi_{i\to l}^{i\to k}|M|\Psi^{b}{}_{s\to u}^{r\to t}\rangle = 0 \tag{4.22}$$

$$\langle\Psi^{a}{}_{j\to l}^{i\to k}|M|\Psi^{a}{}_{s\to u}^{r\to t}\rangle = \delta_{ri}\delta_{tk}\{\delta_{sj}\delta_{ul}(L+m_{tt}-m_{rr})+\delta_{sj}m_{ul}-\delta_{ul}m_{sj}\} \tag{4.23}$$

$$\langle\Psi^{a}{}_{j\to l}^{i\to k}|M|\Psi^{b}{}_{s\to u}^{r\to t}\rangle = 0 \tag{4.24}$$

$$\langle\Psi^{b}{}_{j\to l}^{i\to k}|M|\Psi^{b}{}_{s\to u}^{r\to t}\rangle = \delta_{ri}\delta_{tk}\{\delta_{sj}\delta_{ul}(L+m_{tt}-m_{rr})+\delta_{sj}m_{ul}-\delta_{ul}m_{sj}\} \tag{4.25}$$

ここで、 CI のハミルトニアン行列と同様に、 2 電子励起配置のうち、 $\langle\Psi_{j\to l}^{i\to k}|H|\Psi_{s\to u}^{r\to t}\rangle$ の $i,j,k,l$ および $r,s,t,u$ には、 $\Psi_{j\to l}^{i\to k}$ に対し $\Psi_{i\to k}^{j\to l}$, $\Psi_{s\to u}^{r\to t}$ に対し $\Psi_{r\to t}^{s\to u}$ があるが、これらは、各式中にある $\delta_{ij}$ などが残る (=1) ように決める。
また、 $M$ は $M^x, M^y, M^z$ で、遷移モーメントの $x, y, z$ 成分を表す。

$L$ も同様に $L^x, L^y, L^z$ で、

$$L^x = 2e\sum_p^n m_{pp}^x + \sum_q e_q x_q \tag{4.26}$$

ここで、 $p$ は被占軌道についての総和を取り、 $q$ は核についての総和を取る。

$m_{ij}$ も同様に $m_{ij}^x, m_{ij}^y, m_{ij}^z$ で

$$m_{ij}^x = \langle\psi_i(1)|M^x(1)|\psi_j(1)\rangle \tag{4.27}$$

(4.27) 式の 行列要素 (dipole 積分) は、以下の 2 通りで計算する。

a) 一中心積分うち、同じ原子上の同じ原子軌道のみ考慮する (ZDO 近似)。また、二中心積分は無視する。

$$\langle\psi_i|r|\psi_j\rangle = \sum_t C_{it}C_{jt}\langle\chi_t|r|\chi_t\rangle = \sum_t C_{it}C_{jt}R_t \tag{4.28}$$

ここで、 $t$ は各原子の原子軌道で ( $1s, 2s, 2px, 2py, 2pz, \cdots$), $R_t$ は核電荷を原点にして移動した時の、原子軌道 $\chi_t$ の属する原子の座標。

b) 一中心積分は、同じ原子上の各原子軌道間の積分を考慮し、*Slater* 軌道を用いて解析的に[††]計算する。また、二中心積分は無視する。

$$\langle\psi_i|r|\psi_j\rangle = \sum_{t,u} C_{it}C_{ju}\langle\chi_t|r|\chi_u\rangle \tag{4.29}$$

例えば、次の２つの分子軌道があり、$\chi_{2s} \sim \chi_{2pz}$ までが $A$ 原子上にあり、$\chi_{1s}$ が $B$ 原子上にある場合の

$$\psi_i = C_{i1}\chi_{2s} + C_{i2}\chi_{2px} + C_{i3}\chi_{2py} + C_{i4}\chi_{2pz} + C_{i5}\chi_{1s} \tag{4.30}$$

$$\psi_j = C_{j1}\chi_{2s} + C_{j2}\chi_{2px} + C_{j3}\chi_{2py} + C_{j4}\chi_{2pz} + C_{j5}\chi_{1s} \tag{4.31}$$

$x$ 方向成分の遷移モーメントは

$$\begin{aligned}\langle\psi_i|x|\psi_j\rangle &= C_{i1}C_{j1}\langle\chi_{2s}|x|\chi_{2s}\rangle + C_{i1}C_{j2}\langle\chi_{2s}|x|\chi_{2px}\rangle + C_{i2}C_{j1}\langle\chi_{2px}|x|\chi_{2s}\rangle \\ &+C_{i2}C_{j2}\langle\chi_{2px}|x|\chi_{2px}\rangle + \cdots + C_{i5}C_{j5}\langle\chi_{1s}|x|\chi_{1s}\rangle\end{aligned} \tag{4.32}$$

7) CI 計算した場合の遷移モーメント

CI 計算した時は、各状態は、種々の配置の線形結合で書かれるため、状態間の遷移モーメントは各 CI 係数が掛かった形になる。CI 計算した状態の波動関数 $\Psi_m$ は、

$$\Psi_a = \lambda_{a0}\Psi_0 + \lambda_{a1}\Psi_1 + \lambda_{a2}\Psi_2 + \cdots \tag{4.33}$$

$$\Psi_b = \lambda_{b0}\Psi_0 + \lambda_{b1}\Psi_1 + \lambda_{b2}\Psi_2 + \cdots \tag{4.34}$$

CI 計算で求まった状態 $\Psi_a$,$\Psi_b$ 間の遷移モーメントは、

$$\begin{aligned}\langle\Psi_a|M|\Psi_b\rangle &= \lambda_{a0}\lambda_{b0}\langle\Psi_0|M|\Psi_0\rangle + \lambda_{a0}\lambda_{b1}\langle\Psi_0|M|\Psi_1\rangle + \lambda_{a0}\lambda_{b2}\langle\Psi_0|M|\Psi_2\rangle + \cdots \\ &+\lambda_{a1}\lambda_{b0}\langle\Psi_1|M|\Psi_0\rangle + \lambda_{a1}\lambda_{b1}\langle\Psi_1|M|\Psi_1\rangle + \cdots\end{aligned} \tag{4.35}$$

ここで右辺の $\langle\Psi_m|M|\Psi_n\rangle$ は式 (4.2) ~ (4.32) に従って求める。

以上のように、半経験的分子軌道法 CNDO/S による基底状態の波動関数とエネルギー, 1 電子および 2 電子励起配置を含む配置間相互作用の方法 (SDCI) による励起状態の波動関数とエネルギー, および励起状態の波動関数と SDCI の係数を用いて求める基底状態および各励起状態間の遷移モーメントを計算するプログラムのソースリスト, 入力例, 出力例を付録の末に付ける。

[††] $\langle\chi_t|r|\chi_u\rangle$ は MOS-F のにある getdx.f, getdy.f, getdz.f の subroutine を用いる。

C. 2次超分極率の計算

非線形感受率の測定に用いられる第3高調波発生についての2次超分極率 ($\gamma$) は、時間依存摂動法から各状態間の仮想遷移経路の総和として次の様に記述される [7]:

$$\gamma_{ijkl}(-3\omega;\omega,\omega,\omega) = P\sum_{mnv}\Bigg\{\frac{\mu^i_{gm}\mu^j_{mn}\mu^k_{nv}\mu^l_{vg}}{(E_{mg}-3\hbar\omega)(E_{ng}-2\hbar\omega)(E_{vg}-\hbar\omega)}$$
$$+\frac{\mu^j_{gm}\mu^i_{mn}\mu^k_{nv}\mu^l_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}-2\hbar\omega)(E_{vg}-\hbar\omega)}$$
$$+\frac{\mu^j_{gm}\mu^k_{mn}\mu^i_{nv}\mu^l_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}+2\hbar\omega)(E_{vg}-\hbar\omega)}$$
$$+\frac{\mu^j_{gm}\mu^k_{mn}\mu^l_{nv}\mu^i_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}+2\hbar\omega)(E_{vg}+\hbar\omega)}\Bigg\}. \qquad (C.1)$$

ここで、$P$ は交換演算子、添字 $m,n,v$ は基底状態と各励起状態, $g$ は基底状態, $i,j,k,l$ はテンソル成分, $E_{mg}$ は基底状態から励起状態 $m$ への励起エネルギー, $\mu_{mn}$ は $m$ $n$ 間の遷移双極子モーメント, $\omega$ は入射光振動数、$\hbar$ は Plank 定数を $2\pi$ で割った量を表す。しかし、この式は、入射光振動数が電子励起エネルギーに対して無視できるほど小さい非共鳴の場合、分母がゼロとなる divergence を起こすため、この問題を解消するために書き換えられた次の Orr の式 [8,9] を用いて計算する。

$$\gamma_{ijkl}(-3\omega;\omega,\omega,\omega) = P\sum_{mnv(\neq g)}\Bigg\{\frac{\mu^i_{gm}\bar{\mu}^j_{mn}\bar{\mu}^k_{nv}\mu^l_{vg}}{(E_{mg}-3\hbar\omega)(E_{ng}-2\hbar\omega)(E_{vg}-\hbar\omega)}$$
$$+\frac{\mu^j_{gm}\bar{\mu}^i_{mn}\bar{\mu}^k_{nv}\mu^l_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}-2\hbar\omega)(E_{vg}-\hbar\omega)}$$
$$+\frac{\mu^j_{gm}\bar{\mu}^k_{mn}\bar{\mu}^i_{nv}\mu^l_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}+2\hbar\omega)(E_{vg}-\hbar\omega)}$$
$$+\frac{\mu^j_{gm}\bar{\mu}^k_{mn}\bar{\mu}^l_{nv}\mu^i_{vg}}{(E_{mg}+\hbar\omega)(E_{ng}+2\hbar\omega)(E_{vg}+\hbar\omega)}\Bigg\}$$
$$-P\sum_{mn(\neq g)}\Bigg\{\frac{\mu^i_{gn}\mu^j_{ng}\mu^k_{gm}\mu^l_{mg}}{(E_{ng}-3\hbar\omega)(E_{ng}-\hbar\omega)(E_{mg}-\hbar\omega)}$$
$$+\frac{\mu^j_{gn}\mu^i_{ng}\mu^k_{gm}\mu^l_{mg}}{(E_{ng}-\hbar\omega)(E_{mg}+\hbar\omega)(E_{mg}-\hbar\omega)}$$
$$+\frac{\mu^j_{gn}\mu^k_{ng}\mu^i_{gm}\mu^l_{mg}}{(E_{ng}+3\hbar\omega)(E_{ng}+\hbar\omega)(E_{mg}+\hbar\omega)}$$
$$+\frac{\mu^j_{gn}\mu^k_{ng}\mu^l_{gm}\mu^i_{mg}}{(E_{ng}+\hbar\omega)(E_{mg}-\hbar\omega)(E_{mg}+\hbar\omega)}\Bigg\}. \qquad (C.2)$$

ここで、$\bar{\mu}_{mn}=\mu_{mn}-\mu_{gg}\delta_{mn}$, 添字 $m,n,v$ は基底状態を含まず、励起状態だけであることに注意する。

基底状態および各励起状態間のエネルギーと遷移モーメントから2次超分極率の各テンソル成分, 2次超分極率に対する遷移パスの寄与順序, 遷移パスのタイプ別の寄与の度合い, 物質パラメータ値を計算するプログラムのソースリスト, 入力例, 出力例を付録の末に付ける。

参考及び文献

1) J. J. P. Stewart, *MOPAC 93.00 Manual*, Fujitsu Limited, Tokyo, Japan(1993).
2) JCPE, Japan Chemistry Program Exchange, Tokyo, Japan.
3) A. Matsuura and T. Hayano, Fujitu Sci. Tech. J., 28, 402(1992).
4) A. Matsuura, JCPE P078, 同プログラムは、 CNDO/2-3R[10], MOPAC[11], Gaussian80[12,13] のコードの一部を流用している。
5) 菊池 修, 分子軌道法 電子計算機によるその活用, 講談社, Tokyo(1971)
6) R. S. Mulliken, C. A. Rieke, D. Orloff and H. Orloff, J. Chem. Phys., 17, 1248(1949).
7) 例えば、 *Nonlinear Optics* by R.W. Boyd, (Academic press, 1992).
8) B. J. Orr and J. F. Ward, Mol. Phys., 20, 513(1971)
9) B.M. Pierce, J. Chem. Phys., 91, 791(1989).
10) H. L. Hase and A. Schweig, CNDO/2-3R, QCPE[14] #261
11) J. J. P. Stewart, MOPAC, QCPE #455
12) C. K. Foley and D. B. Chesnut, Gaussian80(modified version of QCPE #406), QCPE #500
13) J. S. Binkley, R. A. Whiteside, R. Krishnan, R. Seegar, H. B. Schlegel, D. J. Defrees and J. A. Pople, Gaussian80, QCPE #406
14) QCPE, Quantum Chemistry Program Exchange, Indiana Univ., Bloomington, USA. (ftp://qcpe6.chem.indiana.edu)

## D1. CNDO/S-SDCI 計算プログラム

## 1. 分子軌道法

複数個の原子が結合してできた分子は、偶数個 ( 2n 個 ) の価電子が結合に寄与する。これらの電子系の Hamiltonian は、次式で与えられる。

$$H(\xi_1, \xi_2, \cdots, \xi_{2n}) = \sum_{v=1}^{2n}\left[-\frac{\hbar^2}{2m}\nabla_v^2 + \sum_b V_b(\xi_v)\right] + \sum_{v'>v}\frac{e^2}{|r_v - r_{v'}|} \tag{1.1}$$

$\xi_v : (r_v, \sigma_v)$

ここで、$\sigma$ と b はそれぞれスピン座標と原子核を表わす。したがって、この電子系の定常状態は、以下のSchroedinger 方程式をとけばもとまる。

$$H\Psi(\xi_1, \xi_2, \cdots, \xi_{2n}) = E\Psi(\xi_1, \xi_2, \cdots, \xi_{2n}) \tag{1.2}$$

$\Psi$ を 1 個の行列式で近似した場合、その状態における、エネルギー期待値$\langle\Psi|H|\Psi\rangle$が停留値となる様にすると、この場合の最良の近似となる。これは、以下の変分問題を解くことを、意味する。

$$\delta\left(\langle\Psi|H|\Psi\rangle - \sum_i \varepsilon_i \sum_\sigma \int dr|\varphi_i|^2\right) = 0 \tag{1.3}$$

$$\Psi(\xi_1, \xi_2, \cdots, \xi_{2n}) = N|\varphi_1\overline{\varphi}_1\varphi_2\overline{\varphi}_2 \cdots \varphi_n\overline{\varphi}_n| \equiv N\begin{vmatrix} \varphi_1(\xi_1) & \overline{\varphi}_1(\xi_1)\cdots & \varphi_n(\xi_1) & \overline{\varphi}_n(\xi_1) \\ \varphi_1(\xi_2) & \overline{\varphi}_1(\xi_2)\cdots & \varphi_n(\xi_2) & \overline{\varphi}_n(\xi_2) \\ \vdots & \vdots \quad \ddots & \vdots & \vdots \\ \varphi_1(\xi_{2n}) & \overline{\varphi}_1(\xi_{2n})\cdots & \varphi_n(\xi_{2n}) & \overline{\varphi}_n(\xi_{2n}) \end{vmatrix}$$

$$\left(\varphi_i(\xi_v) \equiv \psi_i(r_v)\alpha(\sigma_v),\ \overline{\varphi}_i(\xi_v) \equiv \psi_i(r_v)\beta(\sigma_v)\right) \tag{1.4}$$

$$\sum_\sigma \int dr|\varphi_i|^2 = 1 \tag{1.5}$$

ここで、N は規格化因子、(1.5) は拘束条件、また$\varepsilon_i$は Lagrange の乗数である。(1.3) となる$\varphi_i$の軌道部分 $\psi_i$ は、次式を満たす。

$$\left[-\frac{\hbar^2}{2m}\nabla^2 + V(r) + 2\sum_{j=1}^{n}\int dr'\frac{e^2|\psi_j(r')|^2}{|r - r'|}\right]\psi_i(r) \qquad \left(V(r) = \sum_b V_b(r)\right)$$
$$-\sum_{j=1}^{n}\int dr'\frac{e^2\psi_j(r')\psi_i(r')}{|r - r'|}\psi_j(r) = \varepsilon_i\psi_i(r) \tag{1.6}$$

なお、(1.4) は Slater 行列式、また、(1.6) は、Hartree-Fock 方程式と呼ばれる。$\psi_i(r)$は分子全体に拡がりを持つ 1 電子状態で分子軌道(MO)と呼ばれる。(1.6) は記号的に表わすと、次式の固有値問題となる。

$$F|\psi\rangle = \varepsilon|\psi\rangle \tag{1.7}$$

原子に局在する 1 電子状態の線形結合でMOを近似する方法は、LCAOMOと呼ばれる。つまり、MO を次式で近似し、(1.7) に新たに変分法を適用する。





$$\psi(r) = \sum_{s=1}^{m} C_s\chi_s(r) \tag{1.8}$$

$$\langle\psi|\psi\rangle = 1 \tag{1.9}$$

$\chi_s(r)$には規格化された Slater 型軌道関数が用いられる。$C_s(1\leq s\leq m)$は変分パラメーターであり、拘束条件(1.9)を考慮すると、次式を満たす。

$$\delta\left(\langle\psi|F|\psi\rangle - \varepsilon\langle\psi|\psi\rangle\right) = 0 \tag{1.10}$$

(1.10) に (1.8) を代入して、次式を得る。

$$\sum_{s=1}^{m}(F_{rs} - \varepsilon S_{rs})C_s = 0 \qquad (1 \leq r \leq m) \tag{1.11}$$

$$F_{rs} = \langle\chi_r|F|\chi_s\rangle = \int dr'\chi_r(r')F\chi_s(r') \tag{1.12}$$

$$S_{rs} = \langle\chi_r|\chi_s\rangle = \int dr'\chi_r(r')\chi_s(r') \tag{1.13}$$

(1.9) に (1.8) を代入して、次の拘束式を得る。

$$\sum_{r,s} C_rC_sS_{rs} = 1 \tag{1.14}$$

次式の固有方程式の解の固有値 $\varepsilon$ をもとめ、それを用いて(1.11) と (1.14) を連立して (1.11) の自明でない解$C_s$が、もとめられる。

$$\det(F - \varepsilon S) = 0 \tag{1.15}$$

以上より、分子軌道の添え字を i とすると、n個の分子軌道$\psi_i$とそのエネルギー準位$\varepsilon_i$がもとまり、分子軌道は次式で与えられる。

$$\psi_i(r) = \sum_{s=1}^{m} C_{is}\chi_s(r) \qquad (1 \leq i \leq n) \tag{1.16}$$

(1.12) の行列要素は、以下のように表わされる。

$$\begin{aligned} F_{rs} &= \langle\chi_r|F|\chi_s\rangle \\ &= \int dr\chi_r(r)\left[-\frac{\hbar^2}{2m}\nabla^2 + V(r)\right]\chi_s(r) \\ &\quad + 2\sum_{j=1}^{n}\int drdr'\frac{e^2\chi_r(r)\chi_s(r)|\psi_j(r')|^2}{|r - r'|} \\ &\quad - \sum_{j=1}^{n}\int drdr'\frac{e^2\chi_r(r)\chi_s(r')\psi_j(r)\psi_j(r')}{|r - r'|} \\ &= \hat{H}_{rs} + 2\sum_{t,u}\sum_{j=1}^{n} C_{jt}C_{ju}\int drdr'\frac{e^2\chi_r(r)\chi_s(r)\chi_t(r')\chi_u(r')}{|r - r'|} \end{aligned}$$

$$- \sum_{t,u} \sum_{j=1}^{n} C_{jt} C_{ju} \int d\mathbf{r} d\mathbf{r}' \frac{e^2 \chi_r(\mathbf{r}) \chi_t(\mathbf{r}) \chi_s(\mathbf{r}') \chi_u(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|}$$

$$= \hat{H}_{rs} + \sum_{t,u} P_{tu} \left[ (rs|tu) - \frac{1}{2} (rt|su) \right] \tag{1.17}$$

ここで、次の記号が用いられている。

$$\hat{H}_{rs} \equiv \int d\mathbf{r} \chi_r(\mathbf{r}) \left[ -\frac{\hbar^2}{2m} \nabla^2 + V(\mathbf{r}) \right] \chi_s(\mathbf{r}) \tag{1.18}$$

$$(rs|tu) \equiv \int d\mathbf{r} d\mathbf{r}' \frac{e^2 \chi_r(\mathbf{r}) \chi_s(\mathbf{r}) \chi_t(\mathbf{r}') \chi_u(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} \tag{1.19}$$

$$P_{tu} \equiv 2 \sum_{j=1}^{n} C_{jt} C_{ju} \tag{1.20}$$

$\hat{H}_{rs}$, $(rs|tu)$はそれぞれcore積分と原子積分であり、また$P_{tu}$は結合次数と呼ばれる。

通常 m>n であり、m 個の MO

$$\psi_1, \psi_2, \cdots, \psi_n, \psi_{n+1}, \cdots, \psi_{m-1}, \psi_m \quad (\varepsilon_1 \le \varepsilon_2 \le \cdots \le \varepsilon_{m-1} \le \varepsilon_m)$$

がもとまる。MO $\psi_1, \psi_2, \cdots, \psi_n$ は、被占軌道と呼ばれ、これらの MO からなる Slater 行列式は、ここでの近似における基底状態を与える。(1.20) の評価は、被占軌道からの寄与のみ考慮されてなされる。MO $\psi_{n+1}, \cdots, \psi_{m-1}, \psi_m$ は空軌道と呼ばれ、励起状態を構成するのに用いられる。





## 2. CNDO (Complete Neglect of Differential Overlap)

Slater 型軌道関数 $\chi$, 同士の空間的重なりを無視し、(1.11) と (1.15)を次式で近似する。

$$\sum_s (F_{rs} - \varepsilon \delta_{rs}) C_s = 0 \tag{2.1}$$

$$\det(F - \varepsilon I) = 0 \tag{2.2}$$

この近似は、CNDO と呼ばれる。ここで、$\delta_{rs}$と I はそれぞれ、Kronecker のデルタと単位行列である。さらに、以下の近似を適用して、(1.17) の行列要素を簡略化する。

$$(rs|tu) \approx (rr|tt)\delta_{rs}\delta_{tu} = \gamma_{rt}\delta_{rs}\delta_{tu} \tag{2.3}$$

$$\gamma_{rt} \equiv (rr|tt) \tag{2.4}$$

(2.3) と (2.4) を (1.17) に代入して、次式が得られる。

$$F_{rs} = \hat{H}_{rs} - \frac{1}{2} P_{rs}\gamma_{rs} + \delta_{rs} \sum_t P_{tt}\gamma_{rt} \tag{2.5}$$

さらに、上記近似に加えて、以下の近似を用いる。以後、$r \in a$は原子軌道 r が原子 a に属すことを表わす。まず、$\gamma_{rt}$は、添え字が表わす原子のみに依存するパラメーターで与えられ次式で近似できるとする。

$$\gamma_{rr} \approx \Gamma_a \quad (r \in a) \tag{2.6}$$

$$\gamma_{rt} \approx \Gamma_{ab} \quad (r \ne t, r \in a, t \in b) \tag{2.7}$$

$\Gamma_a$と$\Gamma_{ab}$は経験的に与えられるパラメーターである。また、$\hat{H}_{rs}$は以下の様に近似される。

a) r=s

$$\begin{aligned}
\hat{H}_{rr} &\equiv \int d\mathbf{r} \chi_r(\mathbf{r}) \left[ -\frac{\hbar^2}{2m} \nabla^2 + \sum_b V_b \right] \chi_r(\mathbf{r}) \\
&= \int d\mathbf{r} \chi_r \left[ -\frac{\hbar^2}{2m} \nabla^2 + V_a \right] \chi_r + \sum_{b \ne a} \int d\mathbf{r} \chi_r V_b \chi_r \\
&= U_{rr} - \sum_{b \ne a} Z_b \int d\mathbf{r} \frac{e^2 \chi_r(\mathbf{r}) \chi_r(\mathbf{r})}{|\mathbf{r} - \mathbf{r}_b|} \\
&\approx U_{rr} - \sum_{b \ne a} Z_b \Gamma_{ab} \quad (r \in a)
\end{aligned} \tag{2.8}$$

$$U_{rr} \equiv \int d\mathbf{r} \chi_r \left[ -\frac{\hbar^2}{2m} \nabla^2 + V_a \right] \chi_r \approx -\frac{1}{2}(I_r + A_r) - \left( Z_a - \frac{1}{2} \right) \Gamma_a \quad (r \in a) \tag{2.9}$$

ここで、$I_r$と$A_r$は原子軌道 r のイオン化エネルギーと電子親和力であり、$Z_a$は原子番号から内殻電子の数を引いたものである。

b) r≠s

$$\begin{aligned}
\hat{H}_{rs} &\equiv \int d\mathbf{r} \chi_r(\mathbf{r}) \left[ -\frac{\hbar^2}{2m} \nabla^2 + \sum_b V_b \right] \chi_s(\mathbf{r}) \\
&\approx \frac{1}{2} \left[ \int d\mathbf{r} \chi_s \left[ -\frac{\hbar^2}{2m} \nabla^2 + V_a \right] \chi_r + \int d\mathbf{r} \chi_r \left[ -\frac{\hbar^2}{2m} \nabla^2 + V_{a'} \right] \chi_s \right]
\end{aligned}$$

$$
\begin{aligned}
&\approx \frac{K}{2}\left[\hat{H}_{rr}\int d\boldsymbol{r}\chi_s\chi_r + \hat{H}_{ss}\int d\boldsymbol{r}\chi_r\chi_s\right] \\
&= \frac{K}{2}S_{rs}\left(\hat{H}_{rr} + \hat{H}_{ss}\right) \\
&\approx \frac{K}{2}S_{rs}\left(\beta_a^0 + \beta_{a'}^0\right) \quad (r \in a, s \in a')
\end{aligned}
\tag{2.10}
$$

ここで、K, $\beta_a^0$ は経験的パラメーターであり、$S_{rs}$は (1.13) によって数値的に与えられる。

以上より、F の行列要素は、次の様に表わすことができる。

a) r=s

$$
\begin{aligned}
F_{rr} &= \hat{H}_{rr} - \frac{1}{2}P_{rr}\gamma_{rr} + \sum_t P_{tt}\gamma_{rt} \\
&= -\frac{1}{2}\left(I_r + A_r\right) - \left(Z_a - \frac{1}{2}\right)\Gamma_a - \sum_{b\neq a} Z_b\Gamma_{ab} \qquad (r \in a) \\
&\quad - \frac{1}{2}P_{rr}\Gamma_a + \sum_{t\in a} P_{tt}\gamma_{rt} + \sum_{t\in a} P_{tt}\gamma_{rt} \\
&= -\frac{1}{2}\left(I_r + A_r\right) - \left[Z_a - \frac{1}{2}\left(1 - P_{rr}\right)\right]\Gamma_a - \sum_{b\neq a} Z_b\Gamma_{ab} \\
&\quad + \sum_{t\in a} P_{tt}\Gamma_a + \sum_{b\neq a}\sum_{t\in b} P_{tt}\Gamma_{ab} \\
&= -\frac{1}{2}\left(I_r + A_r\right) + \left[P_a - Z_a - \frac{1}{2}\left(P_{rr} - 1\right)\right]\Gamma_a + \sum_{b\neq a}\left(P_b - Z_b\right)\Gamma_{ab}
\end{aligned}
\tag{2.11}
$$

$$
P_a = \sum_{t\in a} P_{tt} \tag{2.12}
$$

b) r≠s

$$
\begin{aligned}
F_{rs} &= \hat{H}_{rs} - \frac{1}{2}P_{rs}\gamma_{rs} \\
&\approx \frac{K}{2}S_{rs}\left(\beta_a^0 + \beta_{a'}^0\right) - \frac{1}{2}P_{rs}\Gamma_{aa'} \quad (r \in a, s \in a')
\end{aligned}
\tag{2.13}
$$

(2.11)と(2.13)より、F の行列要素の計算に必要なパラメーターは、$K, I_r, A_r, \beta_a^0, \Gamma_a, \Gamma_{ab}$であり、$S_{rs}$は数値積分によって与えられる。





## 3. 分子積分

LCAO 近似において、MO は次式で与えられる。

$$
\psi_i(\boldsymbol{r}) = \sum_{s=1}^{m} C_{is}\chi_s(\boldsymbol{r}) \quad (1 \le i \le n) \tag{3.1}
$$

$$
\sum_{r,s} C_{ir}C_{is}S_{rs} = 1 \qquad (1 \le i \le n) \tag{3.2}
$$

ここで、$\psi_i$と$\chi_s$はそれぞれMOとAOである。 (3.1) に関する積分は、分子積分とよばれ、以下の様に定義される。

$$
\tilde{H}_{ij} \equiv \int d\boldsymbol{r}\psi_i(\boldsymbol{r})\left[-\frac{\hbar^2}{2m}\nabla^2 + V(\boldsymbol{r})\right]\psi_j(\boldsymbol{r}) \tag{3.3}
$$

$$
[ij|k\ell] \equiv \int d\boldsymbol{r}d\boldsymbol{r}'\,\frac{e^2\psi_i(\boldsymbol{r})\psi_j(\boldsymbol{r})\psi_k(\boldsymbol{r}')\psi_\ell(\boldsymbol{r}')}{|\boldsymbol{r} - \boldsymbol{r}'|} \tag{3.4}
$$

$$
J_{ij} \equiv [ii|jj] \tag{3.5}
$$

$$
K_{ij} \equiv [ij|ij] \tag{3.6}
$$

これらに、(3.1) を代入して、分子積分と(1.18), (1.19) の原子積分の関係が、以下の様に得られる。

$$
\tilde{H}_{ij} = \sum_{r,s} C_{ir}C_{js}\hat{H}_{rs} \tag{3.7}
$$

$$
[ij|k\ell] = \sum_{r,s,t,u} C_{ir}C_{js}C_{kt}C_{\ell u}(rs|tu) \tag{3.8}
$$

$$
J_{ij} = [ii|jj] = \sum_{r,s,t,u} C_{ir}C_{is}C_{jt}C_{ju}(rs|tu) \tag{3.9}
$$

$$
K_{ij} = [ij|ij] = \sum_{r,s,t,u} C_{ir}C_{is}C_{jt}C_{ju}(rt|su) \tag{3.10}
$$

ところで、軌道部分が(3.1) の MO となる状態 (1.4) における、エネルギー期待値を計算すると、次式が得られる。

$$
\langle H\rangle \equiv \langle\Psi|H|\Psi\rangle = 2\sum_{i=1}^{n}\tilde{H}_{ii} + \sum_{i=1}^{n}\sum_{j=1}^{n}\left(2J_{ij} - K_{ij}\right) \tag{3.11}
$$

(3.11) に (3.7) 等を代入して、原子積分で表わすと、次式となる。

$$
\langle H\rangle = 2\sum_{i=1}^{n}\sum_{r,s} C_{ir}C_{is}\hat{H}_{rs} + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{r,s,t,u} C_{ir}C_{is}C_{jt}C_{ju}\left[2(rs|tu) - (rt|su)\right] \tag{3.12}
$$

(3.12) の拘束条件(1.5) のもとでの停留点は、Lagrange の乗数法を用いると、(3.12) を (1.3) に代入して得られる方程式を解いてもとまる。$C_{ir}$を変分パラメーターと解釈すると、この方程式は、次式となる。

$$\frac{\partial}{\partial C_{ir}}\left(\langle H\rangle - \sum_{k=1}^{n}\varepsilon_k \sum_{\sigma=\pm\frac{1}{2}}\int dr|\varphi_k|^2\right) = 0 \quad (1 \le i \le n, 1 \le r \le m) \tag{3.13}$$

(3.13) に (3.12) を代入して(1.11) と等価な次式を得る。

$$\sum_{s=1}^{m}(F_{rs} - \varepsilon_i S_{rs})C_{is} = 0 \quad (1 \le i \le n, 1 \le r \le m) \tag{3.14}$$

(3.14) に $C_{ir}$ をかけ r についての和をとると、(3.2) を考慮して次式となる。

$$\begin{aligned}\sum_{r,s}(F_{rs} - \varepsilon_i S_{rs})C_{ir}C_{is} &= \sum_{r,s}(F_{rs}C_{ir}C_{is} - \varepsilon_i S_{rs}C_{ir}C_{is}) \\ &= \sum_{r,s}F_{rs}C_{ir}C_{is} - \varepsilon_i = 0\end{aligned} \tag{3.15}$$

したがって、MO $\psi_i$ のエネルギー固有値 $\varepsilon_i$ は以下の様に表わされる。

$$\begin{aligned}\varepsilon_i &= \sum_{r,s}F_{rs}C_{ir}C_{is} \\ &= \sum_{r,s}C_{ir}C_{is}\hat{H}_{rs} + 2\sum_{j=1}^{n}\sum_{r,s,t,u}C_{ir}C_{is}C_{jt}C_{ju}\left[(rs|tu) - \frac{1}{2}(rt|su)\right] \\ &= \tilde{H}_{ii} + \sum_{j=1}^{n}(2J_{ij} - K_{ij})\end{aligned} \tag{3.16}$$

(3.11) と (3.16) より次式を得る。

$$\langle H\rangle = 2\sum_{i=1}^{n}\varepsilon_i - \sum_{i=1}^{n}\sum_{j=1}^{n}(2J_{ij} - K_{ij}) \tag{3.17}$$

## 4. CI 法

LCAOMO により、MO

$$\psi_1, \psi_2, \cdots, \psi_n, \psi_{n+1}, \cdots, \psi_{m-1}, \psi_m \quad (\varepsilon_1 \le \varepsilon_2 \le \cdots \le \varepsilon_{m-1} \le \varepsilon_m) \tag{4.1}$$

がもとまり、次式で与えられるHartree-Fock 法での基底状態が得られる。

$$|\Psi_0\rangle = \frac{1}{\sqrt{(2n)!}} |\varphi_1\overline{\varphi}_1\varphi_2\overline{\varphi}_2 \cdots \varphi_n\overline{\varphi}_n| \equiv \frac{1}{\sqrt{(2n)!}} \begin{vmatrix} \varphi_1(\xi_1) & \overline{\varphi}_1(\xi_1) \cdots & \varphi_n(\xi_1) & \overline{\varphi}_n(\xi_1) \\ \varphi_1(\xi_2) & \overline{\varphi}_1(\xi_2) \cdots & \varphi_n(\xi_2) & \overline{\varphi}_n(\xi_2) \\ \vdots & \vdots \quad \ddots & \vdots & \vdots \\ \varphi_1(\xi_{2n}) & \overline{\varphi}_1(\xi_{2n}) \cdots & \varphi_n(\xi_{2n}) & \overline{\varphi}_n(\xi_{2n}) \end{vmatrix}$$

$$\xi_\nu : (\boldsymbol{r}_\nu, \sigma_\nu) \tag{4.2}$$

$$\varphi_i(\xi_\nu) \equiv \psi_i(\boldsymbol{r}_\nu)\alpha(\sigma_\nu), \ \overline{\varphi}_i(\xi_\nu) \equiv \psi_i(\boldsymbol{r}_\nu)\beta(\sigma_\nu) \tag{4.3}$$

ここで、$\sigma_\nu$は$\nu$番目の電子のスピン座標であり、$\alpha$と$\beta$はスピン状態である。

$|\Psi_0\rangle$からの励起状態は、空軌道$\psi_k(n < k)$を$|\Psi_0\rangle$の被占軌道$\psi_i(i \le n)$と入れ替えることによりもとまる。たとえば、1 重項の 1 電子励起状態は次式で与えられる。

$$|\Psi_{i\to k}\rangle \equiv \frac{1}{\sqrt{2}} \frac{1}{\sqrt{(2n)!}} \left[ |\varphi_1\overline{\varphi}_1\varphi_2\overline{\varphi}_2 \cdots \varphi_i\overline{\varphi}_k \cdots \varphi_n\overline{\varphi}_n| - |\varphi_1\overline{\varphi}_1\varphi_2\overline{\varphi}_2 \cdots \overline{\varphi}_i\varphi_k \cdots \varphi_n\overline{\varphi}_n| \right] \tag{4.4}$$

$$(i \le n; n < k)$$

Hartree-Fock 法では、(1.1) の固有状態$\Psi$を$|\Psi_0\rangle$のみとした近似を採用している。しかし、(4.4) の様な無限個の励起状態が混合された状態が真の固有状態$\Psi$なので、$\Psi$は次式となる。

$$\Psi(\xi_1, \xi_2, \cdots, \xi_{2n}) = \sum_{\omega=1}^{\infty} D_\omega \Phi_\omega(\xi_1, \xi_2, \cdots, \xi_{2n}) \tag{4.5}$$

ここで、$\omega$は電子配置に対応する添え字であり、$\Phi_1 = |\Psi_0\rangle, \Phi_2 = |\Psi_{i\to k}\rangle, \cdots$は正規直交系をなし、以下の関係が成立する。

$$\langle \Phi_\omega | \Phi_\tau \rangle = \delta_{\omega\tau} \tag{4.6}$$

(4.5) を有限個の電子配置による展開で次式の様に近似し、(1.1) に有限個の変分パラメーター$D_\omega$を用いる変分法を適用する近似は、CI 法と呼ばれる。

$$\Psi = \sum_{\omega=1}^{\Omega} D_\omega \Phi_\omega \tag{4.7}$$

CI 法は、

$$\langle \Psi | \Psi \rangle = 1 \tag{4.8}$$

を拘束条件とした$\langle \Psi | H | \Psi \rangle$の停留点をもとめる問題に帰着される。これは、Lagrange の乗数法を適用すれば、次の条件と等価となる。

$$\frac{\partial}{\partial D_\omega} \left( \langle \Psi | H | \Psi \rangle - E \langle \Psi | \Psi \rangle \right) = 0 \tag{4.9}$$

よって、(4.7) を (4.9) に代入して次式を得る。

$$\sum_{\tau=1}^{\Omega} (H_{\omega\tau} - E\delta_{\omega\tau}) D_\tau = 0 \tag{4.10}$$





$$H_{\omega\tau} = \langle \Phi_\omega | H | \Phi_\tau \rangle \tag{4.11}$$

(4.10) の固有値問題を解けば、CI 法による多電子状態がもとまる。

## A.1 原子積分の評価

CNDOにおけるFの行列要素は、(2.11) と (2.13) で与えられる。これらの要素の中の原子積分の評価の近似のやりかたにより、CNDOはCNDO/2やCNDO/S等に分けられる。表A1に原子積分の評価の方法が示される。

表A1

| 近似法 | $\hat{H}_{rs}\ (r \neq s)$ † | $\Gamma_a$ | $\Gamma_{ab}$ |
|---|---|---|---|
| CNDO/2 | $\frac{1}{2} S_{rs}(\beta_a^o + \beta_b^o)$ †† | $(2s_a 2s_a \rfloor 2s_a 2s_a)$ ††† | $(2s_a 2s_a \rfloor 2s_b 2s_b)$ ††† |
| CNDO/S | $\frac{K}{2} S_{rs}(\beta_a^o + \beta_b^o)$ †† | $I_r - A_r$ | P-P法 †††† |

† $\hat{H}_{rs} \equiv \int d\tau \chi_r(\tau)\left[-\frac{\hbar^2}{2m}\nabla^2 + \sum_b V_b\right]\chi_s(\tau)$

†† $r \in a, s \in b$

††† 2sのSlater型軌道関数によるクーロン積分

†††† Pariser-Parr の方法で、以下の様に与えられる。

1) $R_{ab}>R_0$ ($R_{ab}$ : 原子核 a と b の距離, $R_0$ : 経験的パラメーター)

$$\Gamma_{ab} = \frac{7.1975}{R_{ab}}\left[\frac{1}{\sqrt{1+\left(\frac{R_a - R_b}{2R_{ab}}\right)^2}} + \frac{1}{\sqrt{1+\left(\frac{R_a + R_b}{2R_{ab}}\right)^2}}\right] \quad \text{(eV)} \tag{a1.1}$$

$$R_a, R_b = \frac{4.597}{Z^*} \quad (\text{Å}) \tag{a1.2}$$

($Z^*$ : Slater 型軌道が感じる原子の有効電荷)

2) $R_{ab} \leqq R_0$

a) Pariser-Parr のオリジナルな方法

$$\Gamma_{ab} = \frac{1}{2}(\Gamma_a + \Gamma_b) + c_1 R_{ab} + c_2 R_{ab}^2 \tag{a1.3}$$

b) 西本-又賀 の方法

$$\Gamma_{ab} = \frac{e^2}{R_{ab} + c} \tag{a1.4}$$

c) 大野の方法

$$\Gamma_{ab} = \frac{e^2}{\sqrt{R_{ab}^2 + c^2}} \tag{a1.5}$$

ここで、$\frac{e^2}{c} = \frac{1}{2}(\Gamma_a + \Gamma_b)$である。





## A.2 CI 法の行列要素

2n 電子系に CI 法を適用する場合の電子配置 ( (4.7) の $\Phi_\omega$ ) の基底を、以下に示す。

Hartree-Fock法の基底状態、1 重項状態の一電子励起状態、そして1 重項状態の 2 電子励起状態を用いることにする。ここで、MO を示す添え字 i, j は被占軌道 ($i \leqq n$, $j \leqq n$) を、k, $\ell$ は空軌道 ( $k>n$, $\ell>n$ ) を表わす。また、対応するエネルギー順位を $\varepsilon_i, \varepsilon_j, \varepsilon_k, \varepsilon_\ell$ と表わすことにする。なお、被占軌道で最高のエネルギー順位の MO は HOMO ( Highest Occupied Molecular Orbital ) そして空軌道で最低エネルギー順位の MO は LUMO ( Lowest Unoccupied Molecular Orbital ) と呼ばれる。

a) 基底状態 (Hartree-Fock状態)

$$\Phi_1 = |\Psi_0\rangle = \frac{1}{\sqrt{(2n)!}}|\varphi_1\bar{\varphi}_1\varphi_2\bar{\varphi}_2 \cdots \varphi_n\bar{\varphi}_n|$$

$$(\varphi_i(\xi_v) \equiv \psi_i(r_v)\alpha(\sigma_v),\ \bar{\varphi}_i(\xi_v) \equiv \psi_i(r_v)\beta(\sigma_v),\ \omega = \Omega_1 = 1) \tag{A2.1}$$

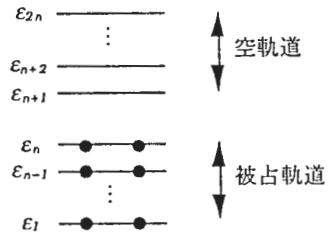



図 A2.1 基底状態 (Hartree-Fock状態) $|\Psi_0\rangle$

b) 1 電子励起状態

$$\Phi_\omega = |\Psi_{i\to k}\rangle \qquad (\Omega_1 < \omega \leq \Omega_2) \tag{A2.2}$$

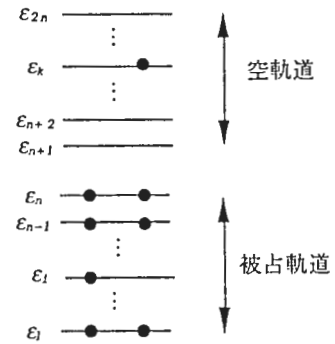



図 A2.2 1 電子励起状態 $|\Psi_{i\to k}\rangle$

c) 2 電子励起状態

(i) type 1

$$\Phi_\omega = \left| \Psi^{\phantom{a}}_{\substack{i\to k\\ i\to k}} \right\rangle \qquad (\Omega_2 < \omega \le \Omega_3) \tag{A2.3}$$

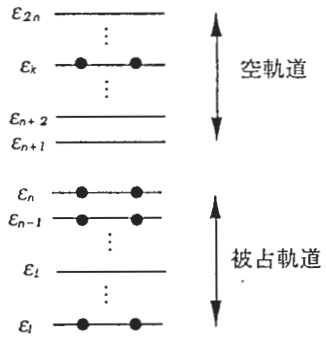



図 A2.3　2 電子励起状態 $\left| \Psi^{\phantom{a}}_{\substack{i\to k\\ i\to k}} \right\rangle$

(ii) type 2

$$\Phi_\omega = \left| \Psi^{\phantom{a}}_{\substack{i\to k\\ j\to k}} \right\rangle \qquad (\Omega_3 < \omega \le \Omega_4,\ i \ne j) \tag{A2.4}$$

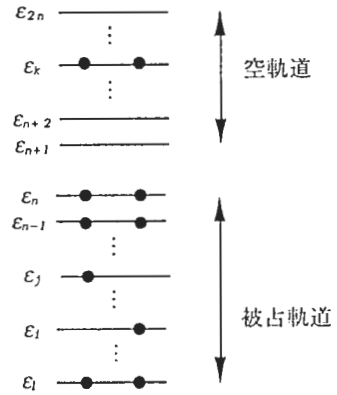



図 A2.4　2 電子励起状態 $\left| \Psi^{\phantom{a}}_{\substack{i\to k\\ j\to k}} \right\rangle$





(iii) type 3

$$\Phi_\omega = \left| \Psi^{\phantom{a}}_{\substack{i\to k\\ i\to \ell}} \right\rangle \qquad (\Omega_4 < \omega \le \Omega_5,\ k \ne \ell) \tag{A2.5}$$

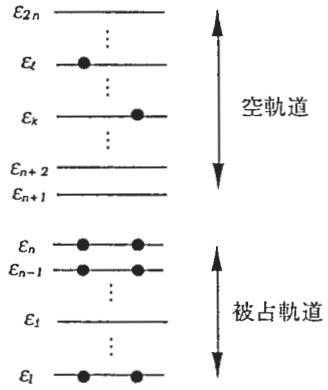



図 A2.5　2 電子励起状態 $\left| \Psi^{\phantom{a}}_{\substack{i\to k\\ i\to \ell}} \right\rangle$

(iv) type 4

$$\Phi_\omega = \left| \Psi^{a}_{\substack{i\to k\\ j\to \ell}} \right\rangle \qquad (\Omega_5 < \omega \le \Omega_6,\ i \ne j,\ k \ne \ell) \tag{A2.6}$$

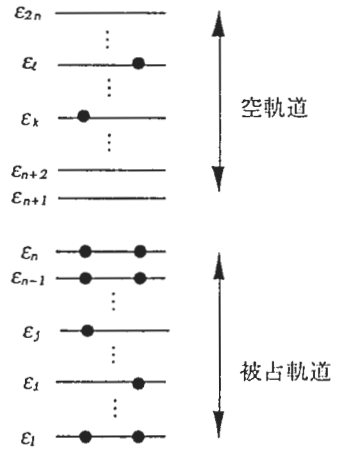



図 A2.6　2 電子励起状態 $\left| \Psi^{a}_{\substack{i\to k\\ j\to \ell}} \right\rangle$

(v) type 5

$$\Phi_\omega = \left| \Psi^b_{\substack{i\to k \\ j\to \ell}} \right\rangle \qquad (\Omega_6 < \omega \le \Omega_7,\ i \ne j,\ k \ne \ell) \tag{A2.7}$$

($\left| \Psi^a_{\substack{i\to k \\ j\to \ell}} \right\rangle$ とスピン配置が逆)

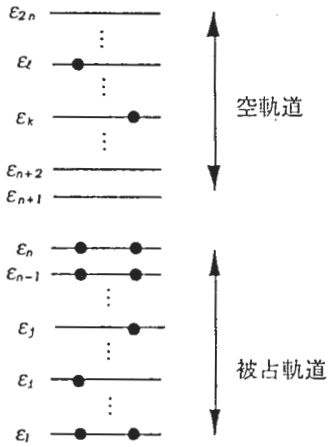



図 A2.7　2 電子励起状態 $\left| \Psi^b_{\substack{i\to k \\ j\to \ell}} \right\rangle$

以上のCI 法の基底$\Phi_\omega, \Phi_\tau$に対するCI 法の行列要素 (4.11) の$H_{\omega\tau}$は、以下の様に表わされる。ただし、1 電子 Hamiltonian F のMO 間の行列要素$\tilde{F}_{ij}$は次式で定義される。

$$\tilde{F}_{ij} \equiv \langle \psi_i | F | \psi_j \rangle = \varepsilon_j \langle \psi_i | \psi_j \rangle = \varepsilon_i \delta_{ij} \tag{A2.8}$$

I. 対角要素

a) 基底状態間

$$\langle \Psi_0 | H | \Psi_0 \rangle = 2\sum_{i=1}^{n} \tilde{H}_{ii} + \sum_{i=1}^{n}\sum_{j=1}^{n} \left(2J_{ij} - K_{ij}\right) = 2\sum_{i=1}^{n} \varepsilon_i - \sum_{i=1}^{n}\sum_{j=1}^{n} \left(2J_{ij} - K_{ij}\right) \equiv E_0 \tag{A2.9}$$

b) 1 電子励起状態間

$$\begin{cases} E_{i\to k} \equiv \varepsilon_k - \varepsilon_i - J_{ik} + 2K_{ik} \\ \langle \Psi_{i\to k} | H | \Psi_{i\to k} \rangle = E_0 + E_{i\to k} \end{cases} \tag{A2.10}$$

c) 2 電子励起状態間

(i) type 1 間

$$< \Psi_{\substack{i\to k \\ i\to k}} \mid H \mid \Psi_{\substack{i\to k \\ i\to k}} > = E_0 + 2E_{i\to k} - 2J_{ik} + J_{ii} + J_{kk} - 2K_{ik} \tag{A2.11}$$

(ii) type 2 間

$$< \Psi_{\substack{i\to k \\ j\to k}} \mid H \mid \Psi_{\substack{i\to k \\ j\to k}} > = E_0 + E_{i\to k} + E_{j\to k} - J_{ik} - J_{jk} + J_{kk} + J_{ij} - K_{ik} - K_{jk} + K_{ij} \qquad (i \neq j) \tag{A2.12}$$

(iii) type 3 間

$$< \Psi_{\substack{i\to k \\ i\to \ell}} \mid H \mid \Psi_{\substack{i\to k \\ i\to \ell}} > = E_0 + E_{i\to k} + E_{i\to \ell} - J_{ik} - J_{i\ell} + J_{ii} + J_{k\ell} - K_{ik} - K_{i\ell} + K_{k\ell} \qquad (k \neq \ell) \tag{A2.13}$$

(iv) type 4 間

$$< \Psi^a_{\substack{i\to k \\ j\to \ell}} \mid H \mid \Psi^a_{\substack{i\to k \\ j\to \ell}} > = E_0 + E_{i\to k} + E_{j\to \ell} - J_{kj} - J_{\ell i} + J_{k\ell} + J_{ij} - \frac{3}{2}\left(K_{ik} + K_{j\ell}\right) + \frac{1}{2}\left(K_{kj} + K_{\ell i}\right) + K_{k\ell} + K_{ij} \qquad (i \neq j, k \neq \ell) \tag{A2.14}$$

(v) type 5 間

$$< \Psi^b_{\substack{i\to k \\ j\to \ell}} \mid H \mid \Psi^b_{\substack{i\to k \\ j\to \ell}} > = E_0 + E_{i\to k} + E_{j\to \ell} - J_{kj} - J_{\ell i} + J_{k\ell} + J_{ij} + \frac{3}{2}\left(K_{ik} + K_{j\ell}\right) - \frac{1}{2}\left(K_{kj} + K_{\ell i}\right) - K_{k\ell} - K_{ij} \qquad (i \neq j, k \neq \ell) \tag{A2.15}$$





II. 非対角要素

a) 基底状態－1 電子励起状態間

$$< \Psi_0 \mid H \mid \Psi_{i\to k} > = \sqrt{2}\tilde{F}_{ik} = 0 \tag{A2.16}$$

b) 基底状態－2 電子励起状態間

(i) 基底状態－type 1 間

$$< \Psi_0 \mid H \mid \Psi_{\substack{i\to k \\ i\to k}} > = [ki \mid ki] \tag{A2.17}$$

(ii) 基底状態－type 2 間

$$< \Psi_0 \mid H \mid \Psi_{\substack{i\to k \\ j\to k}} > = -\sqrt{2}[kj \mid ki] \tag{A2.18}$$

(iii) 基底状態－type 3 間

$$< \Psi_0 \mid H \mid \Psi_{\substack{i\to k \\ i\to \ell}} > = \sqrt{2}[ki \mid \ell i] \tag{A2.19}$$

(iv) 基底状態－type 4 間

$$< \Psi_0 \mid H \mid \Psi^a_{\substack{i\to k \\ j\to \ell}} > = -[ki \mid \ell j] - [kj \mid \ell i] \tag{A2.20}$$

(v) 基底状態－type 5 間

$$< \Psi_0 \mid H \mid \Psi^b_{\substack{i\to k \\ j\to \ell}} > = \sqrt{3}\{[kj \mid \ell i] - [ki \mid \ell j]\} \tag{A2.21}$$

c) 1 電子励起状態－1 電子励起状態間

$$< \Psi_{i\to k} \mid H \mid \Psi_{r\to t} > = \delta_{ir}\tilde{F}_{kt} - \delta_{kt}\tilde{F}_{ir} + 2[rt \mid ki] - [ir \mid kt] \tag{A2.22}$$

d) 1 電子励起状態－2 電子励起状態間

(i) 1 電子励起状態－type 1 間

$$< \Psi_{i\to k} \mid H \mid \Psi_{\substack{r\to t \\ r\to t}} > = \sqrt{2}\left\{\delta_{ir}\delta_{kt}\tilde{F}_{rt} - \delta_{tk}[ir \mid tr] + \delta_{ir}[rt \mid kt]\right\} \tag{A2.23}$$

(ii) 1 電子励起状態－type 2 間

$$< \Psi_{i\to k} \mid H \mid \Psi_{\substack{r\to t \\ s\to t}} > = -\left\{\delta_{ir}\delta_{kt}\tilde{F}_{st} + \delta_{ri}[st \mid kt] - \delta_{tk}\left([st \mid ri] + [rt \mid si]\right)\right\} \tag{A2.24}$$

(iii) 1 電子励起状態－type 3 間

$$< \Psi_{i\to k} \mid H \mid \Psi_{\substack{r\to t \\ r\to u}} > = \delta_{ir}\delta_{kt}\tilde{F}_{ru} + \delta_{ri}\{[rt \mid ku] + [kt \mid ru]\} - \delta_{tk}[ri \mid ru] \tag{A2.25}$$

(iv) 1 電子励起状態－type 4 間

$$< \Psi_{\substack{i\to k}} \mid H \mid \Psi^{a}_{\substack{r\to t\\ s\to u}} > =$$

$$-\sqrt{\frac{1}{2}}\left\{\delta_{ir}\delta_{tk}\tilde{F}_{su} - \delta_{tk}\{[ri \mid su] + [si \mid ru]\} + \delta_{ri}\{[st \mid ku] + [kt \mid su]\}\right\} \tag{A2.26}$$

(v) 1 電子励起状態－type 5 間

$$< \Psi_{\substack{i\to k}} \mid H \mid \Psi^{b}_{\substack{r\to t\\ s\to u}} > =$$

$$-\sqrt{\frac{3}{2}}\left\{\delta_{ir}\delta_{tk}\tilde{F}_{su} - \delta_{tk}\{[ri \mid su] - [si \mid ru]\} - \delta_{ri}\{[st \mid ku] - [kt \mid su]\}\right\} \tag{A2.27}$$

e) type 1－2 電子励起状態間

(i) type 1－type 1 間

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid H \mid \Psi_{\substack{r\to t\\ r\to t}} > = \delta_{ir}[kt \mid kt] + \delta_{tk}[ir \mid ir] \tag{A2.28}$$

(ii) type 1－type 2 間

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid H \mid \Psi_{\substack{r\to t\\ s\to t}} > = \sqrt{2}\delta_{tk}\left\{\delta_{ir}\left[\tilde{F}_{rs} + 2[tt \mid rs] - [ts \mid tr]\right] - [ir \mid is]\right\} \tag{A2.29}$$

(iii) type 1－type 3 間

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid H \mid \Psi_{\substack{r\to t\\ r\to u}} > = \sqrt{2}\delta_{ri}\left\{\delta_{ku}\left[\tilde{F}_{tu} - 2[rr \mid tu] + [ru \mid rt]\right] + [kt \mid ku]\right\} \tag{A2.30}$$

(iv) type 1－type 4 間

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid H \mid \Psi^{a}_{\substack{r\to t\\ s\to u}} > = \delta_{ri}\delta_{ku}\{2[ut \mid sr] - [ur \mid st]\} \tag{A2.31}$$

(v) type 1－type 5 間

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid H \mid \Psi^{b}_{\substack{r\to t\\ s\to u}} > = \sqrt{3}\delta_{ri}\delta_{ku}[ur \mid st] \tag{A2.32}$$

f) type 2－2 電子励起状態間

(i) type 2－type 2 間





$$< \Psi_{\substack{i\to k\\ j\to k}} \mid H \mid \Psi_{\substack{r\to t\\ s\to t}} > =$$

$$\delta_{tk}\left\{\delta_{ir}\left[-\tilde{F}_{sj} - 2[tt \mid sj] + [tj \mid ts]\right] + [ir \mid js] + [is \mid jr]\right\} + \delta_{ri}\delta_{sj}[kt \mid kt] \tag{A2.33}$$

(ii) type 2－type 3 間

$$< \Psi_{\substack{i\to k\\ j\to k}} \mid H \mid \Psi_{\substack{r\to t\\ r\to u}} > =$$

$$\delta_{uk}\left\{\delta_{ir}[2[rj \mid ut] - [rt \mid uj]] + \delta_{rj}[2[ri \mid ut] - [rt \mid ui]]\right\} \tag{A2.34}$$

(iii) type 2－type 4 間

$$< \Psi_{\substack{i\to k\\ j\to k}} \mid H \mid \Psi^{a}_{\substack{r\to t\\ s\to u}} > =$$

$$\sqrt{2}\delta_{si}\left\{\delta_{rj}\left[[kt \mid ku] + \delta_{uk}\left(\tilde{F}_{tu} - [ss \mid tu] + \frac{1}{2}[st \mid su]\right)\right] + \delta_{uk}\left(\frac{1}{2}[rt \mid ju] - [rj \mid tu]\right)\right\} \tag{A2.35}$$

(iv) type 2－type 5 間

$$< \Psi_{\substack{i\to k\\ j\to k}} \mid H \mid \Psi^{b}_{\substack{r\to t\\ s\to u}} > =$$

$$-\frac{\sqrt{3}}{2}\delta_{ri}\delta_{uk}\left\{[st \mid uj] + \delta_{sj}\left[(\sqrt{2} - 1)[st \mid su] - \sqrt{2}[ru \mid rt]\right]\right\} \tag{A2.36}$$

g) type 3－2 電子励起状態間

(i) type 3－type 3 間

$$< \Psi_{\substack{i\to k\\ i\to \ell}} \mid H \mid \Psi_{\substack{r\to t\\ r\to u}} > =$$

$$\delta_{ri}\left\{\delta_{tk}\left[\tilde{F}_{\ell u} - 2[rr \mid \ell u] + [ru \mid r\ell]\right] + [ku \mid \ell t] + [kt \mid u\ell]\right\} + \delta_{tk}\delta_{u\ell}[ir \mid ir] \tag{A2.37}$$

(ii) type 3－type 4 間

$$< \Psi_{\substack{i\to k\\ i\to \ell}} \mid H \mid \Psi^{a}_{\substack{r\to t\\ s\to u}} > =$$

$$\sqrt{2}\delta_{tk}\left\{\delta_{ri}\left[[rs \mid \ell u] - \frac{1}{2}[su \mid \ell r] + \delta_{u\ell}\left(\tilde{F}_{rs} + [tt \mid rs] - \frac{1}{2}[ts \mid rt]\right)\right] - \delta_{u\ell}[ir \mid si]\right\} \tag{A2.38}$$

(iii) type 3－type 5 間

$$< \Psi^{\ }_{\substack{i\to k\\ j\to \ell}} \mid H \mid \Psi^{b}_{\substack{r\to t\\ s\to u}} > =$$
$$- \frac{\sqrt{3}}{2}\,\delta_{rt}\delta_{tk}\Big\{[su \mid \ell r] + \delta_{ut}\Big[(\sqrt{2}-1)[su \mid ru] - \sqrt{2}[tr \mid ts]\Big]\Big\} \tag{A2.39}$$

h) type 4－2 電子励起状態間

(i) type 4－type 4 間

$$< \Psi^{a}_{\substack{i\to k\\ j\to \ell}} \mid H \mid \Psi^{a}_{\substack{r\to t\\ s\to u}} > =$$
$$\delta_{rt}\delta_{sj}\Big\{\delta_{tk}\Big[\tilde{F}_{\ell u} - [rr \mid \ell u] + \frac{1}{2}[ru \mid r\ell]\Big] + [kt \mid \ell u] + [ku \mid \ell t]\Big\}$$
$$+ \delta_{tk}\delta_{ut}\Big\{\delta_{rt}\Big[-\tilde{F}_{sj} - [tt \mid sj] + \frac{1}{2}[tj \mid ts]\Big] + [ri \mid sj] + [rj \mid si]\Big\} \tag{A2.40}$$
$$+ \delta_{tk}\delta_{rt}\Big\{\frac{1}{2}[su \mid \ell j] - [sj \mid \ell u]\Big\}$$

(ii) type 4－type 5 間

$$< \Psi^{a}_{\substack{i\to k\\ j\to \ell}} \mid H \mid \Psi^{b}_{\substack{r\to t\\ s\to u}} > =$$
$$\sqrt{\frac{3}{2}}\delta_{rt}\delta_{tk}\Big\{[su \mid \ell j] - \delta_{sj}[ru \mid r\ell] - \delta_{ut}[st \mid tj] + \delta_{sj}\delta_{ut}[rt \mid rt]\Big\} \tag{A2.41}$$

i) type 5－2 電子励起状態間

(i) type 5－type 5 間

$$< \Psi^{b}_{\substack{i\to k\\ j\to \ell}} \mid H \mid \Psi^{b}_{\substack{r\to t\\ s\to u}} > =$$
$$\delta_{rt}\delta_{sj}\Big\{\delta_{tk}\Big[\tilde{F}_{\ell u} - [rr \mid \ell u] + \frac{3}{2}[ru \mid r\ell]\Big] + [kt \mid \ell u] - [ku \mid \ell t]\Big\}$$
$$+ \delta_{tk}\delta_{ut}\Big\{\delta_{rt}\Big[-\tilde{F}_{sj} - [tt \mid sj] + \frac{3}{2}[tj \mid ts]\Big] + [ri \mid sj] - [rj \mid si]\Big\} \tag{A2.42}$$
$$+ \delta_{tk}\delta_{rt}\Big\{\frac{3}{2}[su \mid \ell j] - [sj \mid \ell u]\Big\}$$





## A.3　対称な添え字の組み合わせの通し番号

ある量Xの要素$X_{ij}$が添え字に対して対称であるとする。この場合、添え字の組み合わせ(i,j)に通し番号nを割与えることができ、以下の1対1の対応が定義できる。

$$(i_n, j_n) \leftrightarrow n(i, j) \tag{A3.1}$$

$$\begin{cases} n = \dfrac{\kappa(\kappa-1)}{2} + \kappa' \\ \kappa = max(i, j) \\ \kappa' = min(i, j) \end{cases} \tag{A3.2}$$

特に、対角成分の通し番号は次式となる。

$$n = \frac{i(i+1)}{2} \tag{A3.3}$$

(A3.1)の対応は、図A3.1のアルゴリズムで定義できる。

例として下記のものがある。

1) 原子積分$\Gamma_{ab}$

(2.7)の$\Gamma_{ab}$においてaとbは原子を示す通し番号である。要素$\Gamma_{ab}$を示す通し番号をnとすると、$\Gamma_{ab}$は以下の様に参照できる。

$$\begin{cases} \Gamma_n \equiv \Gamma_{ab} \\ n = \dfrac{\kappa(\kappa-1)}{2} + \kappa' \\ \kappa = max(a, b) \\ \kappa' = min(a, b) \end{cases} \tag{A3.4}$$

2) 分子積分$[q_i q_j \mid q_k q_\ell]$

ここでi,j,k,$\ell$は遷移に関係するMOを対応し、$q_i$等は実際のMOの番号であり、以下の関係がある。

$$1 \le i, j, k, \ell \le n_{sd} = n_{ocs} + n_{vcs} \tag{A3.5}$$

ここで、$n_{sd}, n_{ocs}, n_{vcs}$はそれぞれ、遷移に関係するMOの総数、励起元の被占軌道の総数、励起先の空軌道の総数である。

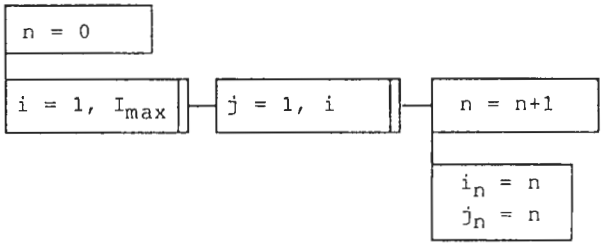



図 A3.1 $(i_n, j_n) \leftrightarrow n(i, j)$の対応の定義

分子積分の要素の通し番号を$\mu$とし、遷移に関係するMOの組合わせの通し番号を$\nu$とすると、分子積分は、以下の様にして参照できる。

$$\begin{cases} I_\mu \equiv I_{\nu(i,j),\nu(k,\ell)} \equiv \left[q_i q_j \mid q_k q_\ell\right] \\ \mu = \dfrac{\lambda(\lambda-1)}{2} + \lambda' \\ \lambda = max(\nu(i,j), \nu(k,\ell)) \\ \lambda' = min(\nu(i,j), \nu(k,\ell)) \\ \nu(i,j) = \dfrac{\kappa(\kappa-1)}{2} + \kappa' \\ \kappa = max(i,j) \\ \kappa' = min(i,j) \\ \nu(k,\ell) = \dfrac{\kappa''(\kappa''-1)}{2} + \kappa''' \\ \kappa'' = max(i,j) \\ \kappa''' = min(i,j) \end{cases} \tag{A3.6}$$

3) 電子配置に関する対称演算子の要素 $< \Phi_\omega \mid A \mid \Phi_\tau >, < \Xi_\lambda \mid A \mid \Xi_{\lambda'} >$

CI法の電子配置の基底とCI法の固有状態をそれぞれ$\Phi_\omega$, $\Xi_\lambda$とすると、それらの演算子の要素は、以下の様にして参照できる。ここで、要素の通し番号をnとする。

$$\begin{cases} A_n \equiv < \Phi_\omega \mid A \mid \Phi_\tau > \\ n = \dfrac{\kappa(\kappa-1)}{2} + \kappa' \\ \kappa = max(\omega, \tau) \\ \kappa' = min(\omega, \tau) \end{cases} \tag{A3.7}$$

$$\begin{cases} \tilde{A}_n \equiv < \Xi_\lambda \mid A \mid \Xi_{\lambda'} > \\ n = \dfrac{\kappa(\kappa-1)}{2} + \kappa' \\ \kappa = max(\lambda, \lambda') \\ \kappa' = min(\lambda, \lambda') \end{cases} \tag{A3.8}$$





## A.4　原子積分から分子積分への変換

分子積分$\left[q_i q_j \mid q_k q_\ell\right]$は、以下の様にして原子積分から得られる。ただし、

$$i' \equiv q_i \tag{A4.1}$$

である。ここで、iと$q_i$はそれぞれ遷移元遷移先の指定の番号とそれの示すMOの番号であり、i等は(A3.5)を満たす。

$$\begin{aligned} [i'j' \mid k'\ell'] &= \sum_{r,s,t,u} C_{i'r} C_{j's} C_{k't} C_{\ell'u} (rs \mid tu) \\ &= \sum_{r,s,t,u} C_{i'r} C_{j's} C_{k't} C_{\ell'u} \delta_{rs} \delta_{tu} \gamma_{rt} \\ &= \sum_{r,t} C_{i'r} C_{j'r} C_{k't} C_{\ell't} \gamma_{rt} \\ &= \sum_{r,t} C_{i'r} C_{j'r} C_{k't} C_{\ell't} \Gamma_{a(r)a(t)} \\ &= \sum_{r} C_{i'r} C_{j'r} \left( \sum_{t} C_{k't} C_{\ell't} \Gamma_{a(r)a(t)} \right) \end{aligned} \tag{A4.2}$$

ここで、a(r), a(t)はAOr, tの属す原子の通し番号である。この分子積分を(A3.6)の様に通し番号$\mu$の順番に格納する。プログラムのなかでは、a(r)と$q_i$はibtoa(r)とlmo(i)によって参照される。

## A.5　遷移モーメント

遷移モーメントは、(A3.8)のAが次式の双極子モーメントの要素である。

$$M = e\sum_{q} r_q \tag{A5.1}$$

CI法の固有状態は、電子配置の基底により次式の様に展開されるとする。

$$\Xi_\lambda = \sum_{\omega=1}^{\Omega} D_{\lambda\omega}\Phi_\omega \qquad (1 \le \lambda \le \Omega) \tag{A5.2}$$

よって、遷移モーメントは(A5.1)の電子配置の基底間の要素により、次式で与えられる。

$$\begin{aligned}\tilde{M}_{\lambda\lambda'} &= < \Xi_\lambda \mid M \mid \Xi_{\lambda'} > \\ &= \sum_{\omega,\tau} D_{\lambda\omega}D_{\lambda'\tau} < \Phi_\omega \mid M \mid \Phi_\tau > \\ &= \sum_{\omega,\tau} D_{\lambda\omega}D_{\lambda'\tau}M_{\omega\tau}\end{aligned} \tag{A5.3}$$

$$M_{\omega\tau} = < \Phi_\omega \mid M \mid \Phi_\tau > \tag{A5.4}$$

なお、電子配置の基底$\Phi_\omega$については、A.2を参照するのがよい。さらに、分子積分$m_{ij}$を用いて$M_{\omega\tau}$は以下の様に評価できる。

$$m_{ij} = e < \psi_i \mid r \mid \psi_j > \tag{A5.5}$$

A) Hartree-Fock 状態と電子配置の基底間の要素

(1) Hartree-Fock 状態間

$$\langle \Psi_0 | M | \Psi_0 \rangle = L \equiv 2e\sum_{p=1}^{n} m_{pp} + \sum_{q=1}^{N} e_q R_q \tag{A5.6}$$

ここで、pは被占軌道のMOを表わし、$e_q$は原子qの有効電荷を表わし、$R_q$は原子qの位置ベクトルである。

(2) Hartree-Fock 状態と1電子励起状態間

$$< \Psi_0 \mid M \mid \Psi_{\substack{i\to k\\ i\to k}} > = \sqrt{2}m_{ik} \tag{A5.7}$$

(3) Hartree-Fock 状態と2電子励起状態間

$$\begin{aligned}< \Psi_0 \mid M \mid \Psi_{\substack{i\to k\\ i\to k}} > &= < \Psi_0 \mid M \mid \Psi_{\substack{i\to k\\ j\to k}} > = < \Psi_0 \mid M \mid \Psi_{\substack{i\to k\\ i\to l}} > \\ &= < \Psi_0 \mid M \mid \Psi^a_{\substack{i\to k\\ j\to l}} > = < \Psi_0 \mid M \mid \Psi^b_{\substack{i\to k\\ j\to l}} > = 0\end{aligned} \tag{A5.8}$$

B) 1電子励起状態と電子配置の基底間の要素

(1) 1電子励起状態間

$$< \Psi_{i\to k} \mid M \mid \Psi_{r\to t} > = \delta_{ri}\delta_{tk}L - \delta_{tk}m_{ri} + \delta_{ri}m_{tk} \tag{A5.9}$$

(2) 1電子励起状態と2電子励起状態間

$$< \Psi_{i\to k} \mid M \mid \Psi_{\substack{r\to t\\ r\to t}} > = \sqrt{2}\delta_{ri}\delta_{tk}m_{rt} \tag{A5.10}$$





$$< \Psi_{i\to k} \mid M \mid \Psi_{\substack{r\to t\\ s\to t}} > = -\delta_{ri}\delta_{tk}m_{st} \tag{A5.11}$$

$$< \Psi_{i\to k} \mid M \mid \Psi_{\substack{r\to t\\ r\to u}} > = \delta_{ri}\delta_{tk}m_{rt} \tag{A5.12}$$

$$< \Psi_{i\to k} \mid M \mid \Psi^a_{\substack{r\to t\\ s\to u}} > = -\frac{1}{\sqrt{2}}\delta_{ri}\delta_{uk}m_{su} \tag{A5.13}$$

$$< \Psi_{i\to k} \mid M \mid \Psi^b_{\substack{r\to t\\ s\to u}} > = -\sqrt{\frac{3}{2}}\delta_{ri}\delta_{tk}m_{su} \tag{A5.14}$$

C) 2電子励起状態間の要素

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid M \mid \Psi_{\substack{r\to t\\ r\to t}} > = \delta_{ri}\delta_{tk}(L + 2m_{tt} - 2m_{rr}) \tag{A5.15}$$

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid M \mid \Psi_{\substack{r\to t\\ s\to t}} > = \sqrt{2}\delta_{ri}\delta_{tk}m_{rs} \tag{A5.16}$$

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid M \mid \Psi_{\substack{r\to t\\ r\to u}} > = \sqrt{2}\delta_{ri}\delta_{tk}m_{tu} \tag{A5.17}$$

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid M \mid \Psi^a_{\substack{r\to t\\ s\to u}} > = 0 \tag{A5.18}$$

$$< \Psi_{\substack{i\to k\\ i\to k}} \mid M \mid \Psi^b_{\substack{r\to t\\ s\to u}} > = 0 \tag{A5.19}$$

$$< \Psi_{\substack{i\to k\\ j\to k}} \mid M \mid \Psi_{\substack{r\to t\\ s\to t}} > = \delta_{ri}\delta_{tk}\left\{\delta_{sj}(L + 2m_{tt} - m_{rr}) - m_{sj}\right\} \tag{A5.20}$$

$$< \Psi_{\substack{i\to k\\ j\to k}} \mid M \mid \Psi_{\substack{r\to t\\ r\to u}} > = 0 \tag{A5.21}$$

$$< \Psi_{\substack{i\to k\\ j\to k}} \mid M \mid \Psi^a_{\substack{r\to t\\ s\to u}} > = \sqrt{2}\delta_{ri}\delta_{sj}\delta_{tk}m_{tu} \tag{A5.22}$$

$$< \Psi_{\substack{i\to k\\ j\to k}} \mid M \mid \Psi^b_{\substack{r\to t\\ s\to u}} > = 0 \tag{A5.23}$$

$$< \Psi_{\substack{i\to k\\ i\to l}} \mid M \mid \Psi_{\substack{r\to t\\ r\to u}} > = \delta_{ri}\delta_{tk}\left\{\delta_{ul}(L + m_{tt} - 2m_{rr}) + m_{ul}\right\} \tag{A5.24}$$

$$< \Psi_{\substack{i\to k\\ i\to l}} \mid M \mid \Psi^a_{\substack{r\to t\\ s\to u}} > = \sqrt{2}\delta_{ri}\delta_{tk}\delta_{ul}m_{rs} \tag{A5.25}$$

$$< \Psi_{\substack{i\to k\\ i\to l}} \mid M \mid \Psi^b_{\substack{r\to t\\ s\to u}} > = 0 \tag{A5.26}$$

$$< \Psi^a_{\substack{i\to k\\ j\to l}} \mid M \mid \Psi^a_{\substack{r\to t\\ s\to u}} > = \delta_{ri}\delta_{tk}\left\{\underbrace{\delta_{sj}}_{\delta_{ul}}(L + m_{tt} - m_{rr}) + \delta_{sj}m_{ul} - \delta_{ul}m_{sj}\right\} \tag{A5.27}$$

$$< \Psi^{a}_{\substack{i\to k\\ j\to l}} \mid \boldsymbol{M} \mid \Psi^{b}_{\substack{r\to t\\ s\to u}} > = 0 \tag{A5.28}$$

$$< \Psi^{b}_{\substack{i\to k\\ j\to l}} \mid \boldsymbol{M} \mid \Psi^{b}_{\substack{r\to t\\ s\to u}} > = \delta_{ri}\delta_{tk}\left\{\overset{\delta_{ul}}{\delta_{sj}}(\boldsymbol{L} + \boldsymbol{m}_{tt} - \boldsymbol{m}_{rr}) + \delta_{sj}\boldsymbol{m}_{ul} - \delta_{ul}\boldsymbol{m}_{sj}\right\} \tag{A5.29}$$

ところで、(A5.5)の$\boldsymbol{m}_{ij}$は以下のようにして、評価される。

[1] ZDO近似

$$\begin{aligned}
\boldsymbol{m}_{ij} &= e < \psi_i \mid \boldsymbol{r} \mid \psi_j > \\
&= e\sum_{t,u} C_{it}C_{ju} < \chi_t \mid \boldsymbol{r} \mid \chi_u > \\
&\approx e\sum_{t,u} C_{it}C_{ju}\delta_{tu}\boldsymbol{R}_{a(t)} \\
&= e\sum_{t} C_{it}C_{jt}\boldsymbol{R}_{a(t)} \\
&= e\sum_{a}\sum_{t\in a} C_{it}C_{jt}\boldsymbol{R}_{a}
\end{aligned} \tag{A5.30}$$

[2] 解析的積分

$$\begin{aligned}
\boldsymbol{m}_{ij} &= e < \psi_i \mid \boldsymbol{r} \mid \psi_j > \\
&= e\sum_{t,u} C_{it}C_{ju} < \chi_t \mid \boldsymbol{r} \mid \chi_u > \\
&\approx e\sum_{t,u} C_{it}C_{ju}\delta_{a(t),a(u)} < \chi_t \mid \boldsymbol{r} \mid \chi_u > \\
&= e\sum_{a}\sum_{k=1}^{g}\sum_{k'=1}^{g} C_{it(k,a)}C_{ju(k',a)} < \chi_{t(k,a)} \mid \boldsymbol{r} \mid \chi_{u(k',a)} >
\end{aligned} \tag{A5.31}$$

$$k, k' = s,\ p_x,\ p_y,\ p_z,\ d_{3z^2-1},\ d_{xy},\ d_{yz},\ d_{x^2-y^2},\ d_{zx} \tag{A5.32}$$

ここで、$t(k, a)$, $u(k', a)$は原子aの軌道$k$, $k'$に対応するAOの通し番号であり、次式をみたす。

$$t(k, a) = i_{a0} + k - 1,\ u(k', a) = i_{a0} + k' - 1 \tag{A5.33}$$

$i_{a0}$は原子aに属すAOの通し番号の最小のものである。また、

$$\boldsymbol{r}^{a}_{k,k'} = < \chi_{t(k,a)} \mid \boldsymbol{r} \mid \chi_{u(k',a)} > \tag{A5.34}$$

はサブルーチンgetdx, getdy, getdzによって求められ、配列要素$dip(k, k')$に格納される。

```
>>>>> subprogram calling tree map <<<<<

 *-->used more than once @-->& with child
 =-->recursive

 mosf     ---precal  ---setcom  ---openf
                      +-prtitl  ---time     *
                                 +-cuts
                                 +-strlen  ---len      *
                                 |          +-char     *
                                 +-char    *
                                 +-max0    *
                                 +-min0    *

          +-jctrl   ---rdopt   ---toupr   *
                               +-merror  *

                    +-optset  ---setflg  ---ipos    @
                               +-ipos    @
                               +-getval  ---ichar   *
                                          +-merror  *
                                          +-dfloat  *
                                          +-idnint  *

                               +-chkkey  ---merror  *
                               +-merror  *
                               +-min0    *
                               +-getenv
                    +-prtopt
                    +-clockm  @

          +-pstcal  ---etime   *
                    +-nint    *
                    +-tohms   *
                    +-time    *

          +-minput  ---rdtitl  ---merror  *
                    +-rdgeom  ---rcoord  ---toupr   *
                               +-merror  *
                               +-reada   ---ichar   *

                               +-ztoc    ---dcos
                                          +-dsin
                                          +-dsqrt   *
                                          +-dabs    *
                                          +-abs
                                          +-merror  *

                               +-ctoz    ---xyzgeo  ---bangle  ---dsqrt   *
                                                     +-dacos   *

                                          +-dihed   ---dsqrt   *
                                                     +-dang    ---dabs    *
                                                                +-dsqrt   *
                                                                +-dacos   *
                                          +-dsqrt   *
                               +-czprnt
                               +-ctrns   ---dfloat  *
                               +-symm
                               +-cprint
                               +-cform   ---iclear
                                          +-cclear
                                          +-cappnd  ---len     *
                                          +-itoc    ---len     *
                                           +-iabs    *
                                           +-fld     ---mod     *
                                          +-csort2

                               +-ancmol  ---anchd   *
                                          +-ancout  @

                               +-atdist  ---min0    *
                                          +-dsqrt   *

                    +-prmset  ---getep
                               +-getbp
                               +-getzp   ---dfloat  *
                               +-getgp
                               +-getpk
                               +-getscp
                               +-rwprml  ---len     *
```

```
                                          +-toupr   *
                                          +-merror  *

                               +-cndos3  ---dsqrt   *
                               +-rwprm2  ---merror  *
                               +-merror  *
                               +-scpout
                    +-pcnoe
                    +-clockm  @

          +-sgint   ---sginfo  ---merror  *
                               +-dfloat  *
                               +-getfct  ---dfloat  *
                               +-ancbas  ---ancout  @

                    +-ovlp    ---relvec  ---dsqrt   *
                               +-dsqrt   *
                               +-ss      @
                               +-max0    *
                               +-harmtr  ---dsqrt   *
                               +-ssmout  *

                    +-eri     ---anl2e   ---dsqrt   *
                                          +-ss      @
                                          +-dfloat  *

                               +-pp2e    ---dfloat  *
                                          +-dsqrt   *

                               +-nm2e    ---dsqrt   *
                               +-ohno2e  ---dsqrt   *
                               +-ok2e    ---dsqrt   *
                               +-dh2e    ---dsqrt   *
                                          +-dexp    *
                               +-asmout
                    +-clockm  @

          +-guess   ---ssmout  *
                    +-diagl   @
                    +-eigout  *
                    +-dminl
                    +-dmaxl
                    +-clockm  @

          +-rhfclo  ---mod     *
                    +-getd
                    +-conclo  ---dsqrt   *
                               +-dfloat  *
                               +-dabs    *
                    +-getf
                    +-scfeng
                    +-diagl   @
                    +-epsln
                    +-geth0   ---max0    *
                               +-min0    *
                    +-ssmout  *
                    +-clockm  @
                    +-merror  *

          +-gprpty  ---dsqrt   *
                    +-engout
                    +-eigout  *
                    +-dmout
                    +-ssmout  *
                    +-acout   *
                    +-getdx   @
                    +-getdy   @
                    +-getdz   @
                    +-ancgp   ---ancout  @
                    +-clockm  @

          +-molint  ---packmo
                    +-moinfo  ---merror  *
                               +-iread   ---rdcard  ---toupr   *
                                          +-chktyp
                                          +-merror  *
                                          +-intchr  ---merror  *
                    +-intcil
                    +-intci2
                    +-moiout  ---mod     *
                    +-mkcibs
                    +-intci0  ---max0    *
                               +-min0    *
```

```
|          +-clockm  @
|
+-sci     ---cimout
|          +-diag1   @
|          +-civout
|          +-clockm  @
|
+-tmom    ---packcv
|          +-dipzdo  @
|          +-diplcn  @
|          +-settm
|          +-tmomci  ---dsqrt   *
|          |          +-msmout  *
|          +-clockm  @
|
+-eprpty  ---min0    *
|          +-dabs    *
|          +-idnint  *
|          +-cichrg
|          +-ancout  @
|          +-acout   *
|          +-dsqrt   *
|          +-ancep2  ---ancout  @
|          |          +-anco2   ---anchd   *
|          |          |          +-ancr    *
|          |          |          +-anci    *
|          |          +-mod     *
|          |          +-anchd   *
|          |          +-dsqrt   *
|          +-clockm  @
+-polar
+-hypol1  ---dlshg   ---ancbet  @
|          |          +-shg     ---betxyz  *
|          |          +-chgunt  *
|          |          +-dsqrt   *
|          |          +-dfloat  *
|          |          +-merror  *
|          |
|          +-dleope  ---ancbet  @
|          |          +-eope    ---betxyz  *
|          |          +-chgunt  *
|          |          +-dsqrt   *
|          |          +-dfloat  *
|          |          +-merror  *
|          |
|          +-dlor    ---ancbet  @
|          |          +-or      ---betxyz  *
|          |          +-chgunt  *
|          |          +-dsqrt   *
|          |          +-dfloat  *
|          |          +-merror  *
|          +-clockm  @
+-hypol2
+-sdcil   ---cimtss  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|          |
|          +-cimt10  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|          |
|          +-cimt1s  ---max0    *
|          |          +-min0    *
|          |          +-dsqrt   *
|          |          +-clockm  @
|          |
|          +-cimt11  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|          |
|          +-cimt30  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|          |
|          +-cimt3s  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|          |
|          +-cimt31  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
```

```
|
|          +-cimt33  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt20  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt2s  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt21  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt22  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt32  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt40  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt4s  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt41  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt42  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt43  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt44  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt50  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt5s  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt51  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt52  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt53  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
|
|          +-cimt54  ---dsqrt   *
|          |          +-max0    *
|          |          +-min0    *
|          |          +-clockm  @
```

```
|          |
|          +-cimt55  ---max0    *
|          |          +-min0    *
|          |          +-clockm  @
|          +-fckout  @
|          +-diag1   @
|          +-cidump  ---civo02  ---min0    *
|          +-clockm  @
|
+-tmsdci  ---packci
           +-dipzdo  @
           +-diplcn  @
           +-tmsdbs  ---tmbs0   ---dsqrt   *
           |          |          +-max0    *
           |          |          +-min0    *
           |          |          +-clockm  @
           |          |
           |          +-tmbss   ---max0    *
           |          |          +-min0    *
           |          +-tmb10
           |          +-tmb1s   ---dsqrt   *
           |          |          +-max0    *
           |          |          +-min0    *
           |          +-tmb11
           |          +-tmb30
           |          +-tmb3s   ---max0    *
           |          |          +-min0    *
           |          |
           |          +-tmb31   ---dsqrt   *
           |          |          +-max0    *
           |          |          +-min0    *
           |          |
           |          +-tmb33   ---max0    *
           |          |          +-min0    *
           |          +-tmb20
           |          +-tmb2s   ---max0    *
           |          |          +-min0    *
           |          |
           |          +-tmb21   ---dsqrt   *
           |          |          +-max0    *
           |          |          +-min0    *
           |          |
           |          +-tmb22   ---max0    *
           |          |          +-min0    *
           |          +-tmb32
           |          +-tmb40
           |          +-tmb4s   ---dsqrt   *
           |          |          +-max0    *
           |          |          +-min0    *
           |          +-tmb41
           |          +-tmb42   ---dsqrt   *
           |          |          +-max0    *
           |          |          +-min0    *
           |          |
           |          +-tmb43   ---dsqrt   *
           |          |          +-max0    *
           |          |          +-min0    *
           |          |
           |          +-tmb44   ---max0    *
           |          |          +-min0    *
           |          +-tmb50
           |          +-tmb5s   ---dsqrt   *
           |          |          +-max0    *
           |          |          +-min0    *
           |          +-tmb51
           |          +-tmb52
           |          +-tmb53
           |          +-tmb54
           |          +-tmb55   ---max0    *
           |          |          +-min0    *
           |          +-tmdump
           |          +-clockm  @
           |
           +-tmsdst  ---max0    *
           |          +-min0    *
           |          +-fckout  @
           +-clockm  @
```

```
#
# mosf Makefile
#
CPPFLAGS        =
#CPPFLAGS       =       -Dssl2vp
#LDFLAGS        =       -lm -lF77 -lssl2vp
#LDFLAGS        = -O
#FC     =       f77
FC      =       fc
FFLAGS =
#FFLAGS  =      -cxdb
FFLAGS  =       -O3
#EXE     =      mosf
EXE     =       ./mosf

# cut for save memory
#OBJS    =       trout.o intout.o

OBJS    =       mosf.o\
                acout.o\
                aintgs.o\
                ancbas.o\
                ancbet.o\
                ancep2.o\
                ancgp.o\
                anchd.o\
                anci.o\
                ancmol.o\
                anco2.o\
                ancout.o\
                ancr.o\
                anl2e.o\
                asmout.o\
                atdist.o\
                bangle.o\
                betxyz.o\
                binom.o\
                bintgs.o\
                cappnd.o\
                cclear.o\
                cform.o\
                chgunt.o\
                chkkey.o\
                chktyp.o\
                cichrg.o\
                cidump.o\
                cimout.o\
                cimt10.o\
                cimt11.o\
                cimt1s.o\
                cimt20.o\
                cimt21.o\
                cimt22.o\
                cimt2s.o\
                cimt30.o\
                cimt31.o\
                cimt32.o\
                cimt33.o\
                cimt3s.o\
                cimt40.o\
                cimt41.o\
                cimt42.o\
                cimt43.o\
                cimt44.o\
                cimt4s.o\
                cimt50.o\
                cimt51.o\
                cimt52.o\
                cimt53.o\
                cimt54.o\
                cimt55.o\
                cimt5s.o\
                cimtss.o\
                civo00.o\
                civo01.o\
                civo02.o\
                civout.o\
                clockm.o\
                cndos3.o\
                coeff.o\
                coeffs.o\
                conclo.o\
                cprint.o\
                csort2.o\
                ctoz.o\
                ctrns.o\
                czprnt.o\
                dleope.o\
                dlor.o\
                dlshg.o\
                dang.o\
                dh2e.o\
                diagl.o\
                diagd.o\
                dihed.o\
                diplcn.o\
                dipzdo.o\
                dmout.o\
                dummy.o\
                dump00.o\
                ehobkd.o\
                ehousd.o\
                eigout.o\
                elmntf.o\
                engout.o\
                eope.o\
                eprpty.o\
                epsln.o\
                eqrt2d.o\
                eri.o\
                fld.o\
                fckout.o\
                getbp.o\
                getd.o\
                getdx.o\
                getdy.o\
                getdz.o\
                getep.o\
                getf.o\
                getfct.o\
                getgp.o\
                geth0.o\
                getpk.o\
                getscp.o\
                getval.o\
                getzp.o\
                gprpty.o\
                guess.o\
                harmtr.o\
                hypoll.o\
                iclear.o\
                intchr.o\
                intci0.o\
                intcil.o\
                ipos.o\
                iread.o\
                itoc.o\
                jctrl.o\
                llcn00.o\
                lzdo00.o\
                merror.o\
                minput.o\
                mkcibs.o\
                moinfo.o\
                moiout.o\
                moldip.o\
                molint.o\
                msmout.o\
                nm2e.o\
                ohno2e.o\
                ok2e.o\
                openf.o\
                optset.o\
                or.o\
                ovlp.o\
                packci.o\
                packcv.o\
                packmo.o\
                pcnoe.o\
                pdmix.o\
                pp2e.o\
                precal.o\
                prmset.o\
                prtitl.o\
                prtopt.o\
                pstcal.o\
```

```
                rcoord.o\
                rdcard.o\
                rdgeom.o\
                rdopt.o\
                rdtitl.o\
                reada.o\
                relvec.o\
                rhfclo.o\
                rwprm1.o\
                rwprm2.o\
                scfeng.o\
                sci.o\
                scpout.o\
                sdci1.o\
                setcom.o\
                setflg.o\
                settm.o\
                sginfo.o\
                sgint.o\
                shg.o\
                spmix.o\
                ss.o\
                ssmout.o\
                strlen.o\
                symm.o\
                timef.o\
                tmb10.o\
                tmb11.o\
                tmb1s.o\
                tmb20.o\
                tmb21.o\
                tmb22.o\
                tmb2s.o\
                tmb30.o\
                tmb31.o\
                tmb32.o\
                tmb33.o\
                tmb3s.o\
                tmb40.o\
                tmb41.o\
                tmb42.o\
                tmb43.o\
                tmb44.o\
                tmb4s.o\
                tmb50.o\
                tmb51.o\
                tmb52.o\
                tmb53.o\
                tmb54.o\
                tmb55.o\
                tmb5s.o\
                tmbs0.o\
                tmbss.o\
                tmdump.o\
                tmom.o\
                tmsdbs.o\
                tmsdst.o\
                tmomci.o\
                tmsdci.o\
                tohms.o\
                tosec.o\
                toupr.o\
                uname.o\
                xyzgeo.o\
                ztoc.o

HDRS    =       MSSIZE

all:    $(EXE)

$(EXE): $(OBJS)
        $(FC) $(FFLAGS) $(CPPFLAGS) -o $(EXE) $(OBJS) $(LDFLAGS)

$(OBJS): $(HDRS)

clean:
        rm -f $(OBJS) $(EXE)

# end of Makefile
c@(#)=MSSIZE ... For include statements
c     Program size
c     ------------------------------------------------------------------
c                                          Fujitsu**          sun***
```

```
c      MaxAt  MaxBas  MCIOcc  MCIVac  ---------------------------------
c                                          bss    Total      bss    Total
c     ---------------------------------------------------------------------
c       100     200     10      10      3.74     4.51     1.42     1.86
c       100     200     20      20      6.33     7.10     4.01     4.44
c       100     200     40      40     64.09    64.86    61.77    62.20
c       100     200     60      60    313.91   314.69   311.59   312.03
c       200     400     20      20      7.87     8.64     5.54     5.97
c       400     800     20      20     23.95    24.72    21.61    22.04
c       800    1600     20      20     87.79    88.56    85.43    85.87
c      1200    2400     20      20    193.82   194.60   191.50   191.93
c      1200    2400     60      60    321.91   322.70   319.60   320.03
c     ---------------------------------------------------------------------
c      *   Units in Mbytes. 1 Mbytes = 1,000,000 bytes here.
c      **  Fujitsu UXP/M Fort77 EX/VP
c      *** Sun Fortran Version 1.4 ... text + data = 0.42 Mbytes
c
c     Parameter list of MOS-F (V01L05).
c       MAXAt  ... Maximum no. of atoms.
c       MAXBas ... Maximum no. of basis functions.
c       MAXIAN ... Maximum no. of atomic numbers (fixed to 99).

c     For CI calculation(1).
c       MCIOcc ... Maximum no. of occupied MOs.
c       MCIVac ... Maximum no. of vacant MOs.
c     For CI calculation(2).
c       MCIMOs ... Maximum no. of active MOs.
c       MCIPck ... Maximum size of a molecular dipole matrix.
c       MCICSF ... Maximum no. of spin adapted CSFs.
c       MCIInt ... Maximum no. of lower triangle elements of a CI
c                  matrix - same as molecular integrals, OVOV.
c$$$       Parameter (MCICSF=MCIOcc*MCIVac)
c$$$       parameter( mcics0 = mciocc*mcivac )
c$$$       parameter( mcics0 = 5000 )

      parameter( maxat  = 50, mxbase = 200 )
      parameter( mciocc = 30,  mcivac = 30 )

c$ohtawara
c       parameter( mcics0 = 500 )
c       parameter( maxbas = max( mxbase, mcics0 ) )

c$ohtawara
c       parameter( mcicsf = maxbas )

c     CI      nconf
c    2 * 2      15
c    3 * 3      55
c    4 * 4     153
c    5 * 5     351
c    6 * 6     703
c    7 * 7    1275
c    8 * 8    2145
c    9 * 9    3403
c   10 *10    5151 (over 2GB = LD error)
c       parameter( mcicsf = 710 )
      parameter( mcicsf = 360 )

      parameter( maxbas = max( mxbase, mcicsf ) )


      parameter( msdcib = mcicsf )
      parameter( msdcim = msdcib*( msdcib + 1 )/2 )
c----------
      Parameter (MaxIAN= 99)
      Parameter (MAPck =MaxAt *(MaxAt +1)/2)
      Parameter (MBPck =MaxBas*(MaxBas+1)/2)
      Parameter (MaxB2 =MaxBas*MaxBas)
      Parameter (MCIMOS=MCIOcc+MCIVac)
      Parameter (MCIPck=MCIMOS*(MCIMOS+1)/2)
      parameter( mumax0 = mcipck*( mcipck + 1 )/2 )

      Parameter (MCIInt=MCICSF*(MCICSF+1)/2)
      Parameter (MCICS2=MCICSF*MCICSF)

c
c------------- b a s i n f
c
      common/ basinf / nlbas(maxat), nubas(maxat), nibas(maxat),
     $                 ibtoa(maxbas)
c
c---------- d g r n d
c
```

```
      common/ dgrnd / dxpt,   dypt,   dzpt,   dxap,   dyap,   dzap,
     $                dx,     dy,     dz
c
c--------- e n e r g y
c
      common/ energy / energy
c
c--------- s d c i 0 0
c
      common/ sdci00 / cimat(msdcim),h000,ncibas,ncimat,isdci0
c$$$     $                 lomg(  msdcim ), ltau( msdcim ),
c$$$     $                 ncibas, ncimat,  isdci0

c$ohtawara
      parameter(idebug=  0)
cc      parameter(idebug=  1)
cc       parameter(idebug= 2)
c       parameter(idebug= -1)

c$ohtawara  to read cimatrix from file
c       parameter(ireadci= 1)

c$ohtawara  to read cimatrix from file
c same as sdci01.h sdci01.h are used by tmb**.f
c       parameter(iprint_yes0 =  0)
c       parameter(iprint_yes0 =  1)
       parameter(iprint_yes0 = -1)

c
c--------- s d c i 0 1
c
      common/ sdci01 / mcio( msdcib ),   momega( 7 ),
     $   lci1( msdcib ), lci2( msdcib ), lci3( msdcib ), lci4( msdcib ),
     $   ici1( mcipck ), ici2( mcipck ), nsdci,  numax

c$ohtawara  to read cimatrix from file
c same as sdci01.h sdci01.h are used by tmb**.f
c       parameter(iprint_yes =  0)
c       parameter(iprint_yes =  1)
       parameter(iprint_yes = -1)

c
c--------- s d c i 0 2
c
      common/ sdci02 / fockmt( mbpck ), ediff( mcipck ),
     $                 gintgl( mumax0 ), mumax
c$$$      common/ sdci02 / fockmt( mbpck ), fckmol( mcipck ),
c$$$     $                 gintgl( mumax0 ), mumax
c
c--------- t m o m n t
c
      common/ tmomnt / tlx,    tly,    tlz,
     $     tmbx( msdcim ), tmby( msdcim ), tmbz( msdcim ),
     $     tmsx( msdcim ), tmsy( msdcim ), tmsz( msdcim )
c
c---------- m o l
c
      common/ mol   / coord(3,maxat), ian(maxat), natoms, icharg,
     $                multip, nae,    nbe,    ne,     nbasis
c
c------------- w i n d o w
c
      common/ window / iomo( mciocc ), ivmo( mcivac ), lmo( mcimos ),
     $                 nocs,   nvcs,   ncsfs
c
c--------- w o r k 0 2
c
      common/ work02 / c(maxb2), eig(maxbas), cieig(mcicsf),
     $                 scrtch( mcicsf, 19 )
c
C@(#)=MOSF(Main)
      Program MOSF
C
C     ------------------------------------------------------------
C     MOS-F (V01L05) ... Semiempirical MO package for spectroscopy.
C        By A. Matsuura   Fujitsu Laboratories Ltd.
C     ------------------------------------------------------------
C
C     Methods.
C      1. CNDO/2    By Pople and Segal.
C      2. CNDO/S    By Del Bene and Jaffe.
C      3. CNDO/S2,3 By Lipari and Duke.
C      4. INDO/S    By Ridley and Zerner ... not available at present.
```

```
C
C     References.
C      1. J.Chem.Phys.,44,   3289(1966)
C      2. J.Chem.Phys.,48(4),1807(1986)
C         J.Chem.Phys.,48(9),4050(1968)
C      3. J.Chem.Phys.,63(5),1748(1975)
C      4. Theor.Chim.Acta,32,111(1973)
C         Theor.Chim.Acta,53,21(1979)
C
C     Note.
C      This program contains the FORTRAN source codes of the following
c      QCPE programs.
C       1. CNDO/2-3R (QCPE #261) by H.L.Hase AND A.Schweig
C       2. MOPAC (QCPE #455) by J.J.P.Stewart
C       3. Gaussian 80 (QCPE #500) by C.K.Foley and D.B.Chesnut (This
C          program is the modified version of QCPE #406 written by
C          J.S.Binkley, R.A.Whiteside, R.Krishnan, R.Seeger,
C          H.B.Schlegel,D.J.Defrees, and J.A.Pople)
C      In order to code this program, the author referred to the
C      FORTRAN source codes of the following QCPE programs. But this
C      program does not contain their source codes.
C       1. CNDO/M (QCPE #315) by H.H.Jaffe
C       2. CNDUV99 (QCPE #333) by H.Baumann
C       3. CINDO/SHIFT/UV (QCPE #382) by M.Rajzmann and P.Francois.
C       4. MELDF (QCPE #580) by E.R.Davidson
C
C     Program limitations.
C      1. Only the closed shell SCF, singlet-SCI and triplet-SCI
C         calculations are available in the current version of MOS-F.
C      2. The maximum number of basis functions, atoms and CI-size are
C         adjustable (using the include statement).
C      3. Atomic parameters.
C        CNDO/2     ... H,Li,Be,B,C,N,O,F,Na,Mg,Al,Si,P,S,Cl,Ge,As,Se,Br
C        CNDO/S     ... H,Li,Be,B,C,N,O,F,Na,Mg,Al,Si,P,S,Cl
C        CNDO/S2,S3 ... H,C,N,O,F
C        INDO/S     ... H,C,N,O ... not available at present.
C
C     This routine is composed of main routines listed below.
C      1. PreCal ... Preset some labeled common blocks and print out
c                    the title of this program.
C      2. JCtrl  ... Set the computational procedure and options.
C      3. MInput ... Get atomic and molecular information.
C      4. SGInt  ... Get overlap integrals, 1 center ERIs and 2 center
C                    ERIs.
C      5. Guess  ... Get initial guess by a Huckel-type calculation.
C      6. RHFClo ... Restricted Hartree-Fock-Roothaan calculation for
C                    a closed shell system.
C      7. GPrpty ... Molecular properties for ground state.
C      8. MolInt ... Integral transformation (from AO to MO).
C      9. SCI    ... Singly excited ci calculation.
C     10. TMom   ... Get transition moments.
C     11. EPrpty ... Molecular properties for excited states.
C     12. Polar  ... Get frequency-dependent polarizabilities, alpha.
C                     ... not available at present.
C     13. Hypo11 ... Get frequency-dependent 1st hyperpolarizabilities,
C                    beta.
C     14. Hypo12 ... Get frequency-dependent 2nd hyperpolarizabilities,
C                    gamma.   ... not available at present.
C     15. PstCal ... Printing of total CPU-time.
c
c     Difference between V01L01 and V01L02.
c      1. Computation of the total atomic charges for the SCI excited
c         states.
c      2. The use of lower cases in input.
c
c     Difference between V01L02 and V01L05.
c      1. Output module for ANCHOR II or other post-process graphic
c         packages.
c      2. The use of lower cases in output (partly).
c      3. Printing of chemical formula and molecular weight.
c      4. Increase of the number of keyword lines.
C     ------------------------------------------------------------------
      implicit real*8 (a-h,o-z)

      include 'MSSIZE'
      include 'sdci00.h'
      Logical IfCI,IfBeta
c     ------------------------------------------------------------------
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c      open(6,file='fort.6',form='formatted',status='unknown')
```

```
      open(16,file='fort.16',form='formatted',status='unknown')

      CALL PRECAL

   10 CALL JCTRL(IRDEND,IfCI,IfBeta)

      IF(IRDEND.EQ.1) THEN
        CALL PSTCAL
      END IF

   20 CALL MINPUT(JRDEND)

      IF(JRDEND.EQ.2) THEN
        CALL PSTCAL
      ELSE IF(JRDEND.EQ.1) THEN
        GO TO 10
      END IF

      CALL SGINT
      CALL GUESS
      CALL RHFCLO
      CALL GPRPTY

c$ohtawara
c      If (.Not. IfCI) Goto 20
      If (.Not. IfCI) Goto 99

      call molint
      if( isdci0 .eq. 0 ) then
        call sci
        call tmom
        call eprpty
        call polar
        if( ifbeta ) then
          call hypo11
          call hypo12
        endif
      elseif( isdci0 .ge. 1 ) then
        call sdci1
        call tmsdci
      else
        write(iout,*)' **** bad choice of ci bases'
        stop
      endif

c$ohtawara
c      go to 20
c$ohtawara
 99   stop

      end

C
C@(#)=ACOUT(SUB)
      SUBROUTINE ACOUT(ATMCHG,IAN,NATOMS)
C
C     PRINTING OF TOTAL ATOMIC CHARGES.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*2 ELMNT
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      COMMON /PTbl2 / ELMNT(99)
      DIMENSION ATMCHG(*),IAN(*)
 2000 Format(1H ,I4,2X,A2,1X,F9.5,5(2X,I4,2X,A2,1X,F9.5))
C     ****************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=4
      ELSE
        IW=6
      END IF
C
      ILOWER=1
      IUPPER=IW
C
   10 IF(IUPPER.GT.NATOMS) THEN
        IUPPER=NATOMS
      END IF
C
      WRITE(IOUT,2000) (I,ELMNT(IAN(I)),ATMCHG(I),I=ILOWER,IUPPER)
C
      IF(IUPPER.EQ.NATOMS) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
C
C@(#)=AINTGS(SUB) ... CNDO2/3R(AINTGS)
      SUBROUTINE AINTGS(X,K)
C
C     ==============================================================
C     CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R
C     ==============================================================
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /AUXINT/ A(17),B(17)
C     ****************************************************************
      A(1) =DEXP(-X)/X
      DO 10  I=1,K
   10 A(I+1) =(A(I)*DFLOAT(I)+DEXP(-X))/X
      RETURN
      END
c
c@(#)=AncBas(Sub)
      Subroutine AncBas(Zeta,Dmy,NAtoms,NBasis,NLBas,NUBas,IUnit)
c
c     Output module of basis functions for post-process graphic package.
c     On input:
c      Zeta   ... Slater-exponent.
c      Dmy    ... Dummy array.
c      NAtoms ... The number of atoms.
c      NBasis ... The number of basis functions.
c      NLBas  ... Atom-basis number information.
c      NUBas  ... Atom-basis number information.
c      IUnit  ... Fortran logical unit number for output.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H, O-Z)
      Dimension Zeta(*),Dmy(*),NLBas(*),NUBas(*)
      Data NF/6/
c     ****************************************************************
c
c     Slater-type basis functions.
      n = 0
      Do 20 iAt = 1, NAtoms
        Z   = Zeta (iAt)
        iLw = NLBas(iAt)
        iUp = NUBas(iAt)
        Do 10 iBas = iLw, IUp
          n = n + 1
          Dmy(n) = Z
   10   Continue
   20 Continue
c
      Call AncOut ('Exponents',9,Dmy,1,1,NBasis,NF,IUnit,0,Junk)
c
      Return
c
      End
c
c@(#)=AncBet(Sub)
      Subroutine AncBet (String,LenStr,WEV,BVec,BTotal,IUnit,IFType,
     &                   Init)
c
c     Output module of frequency-dependent first hyperpolarizabilities
c     for post-process graphic package.
c     On input:
c      String ... String which indicates item to output.
c      LenStr ... Length of string.
c      WEV    ... Frequency of incident light in units of eV.
c      BVec   ... Vector component of first hyperpolarizability.
c      BTotal ... Norm of BVec.
c      IUnit  ... Fortran logical unit number for output.
c      IFType ... Format type for output.
c      Init   ... Flag for printing header.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H,O-Z)
      Character*256 String
```

```
      Logical Init
      Dimension BVec(1)
      Dimension BOut(5)
      Data NRow/1/, NClm/5/, NF/6/
c     ****************************************************************
      If (Init) Call AncHd (String,LenStr,NRow,NClm,NF,IUnit)
      If (Init) Return
c
      BOut(1) = WEV
      BOut(2) = BVec(1)
      BOut(3) = BVec(2)
      BOut(4) = BVec(3)
      BOut(5) = BTotal
c
      Call AncR (BOut,1,NRow,NClm,NF,IUnit,IFType)
c
      Return
c
      End
c
c@(#)=AncEP2(Sub)
      Subroutine AncEP2 (NCSFs,NState,NOCS,NVCS,NMOs,IUnit,MultCI,
     &                   MCIMOs,
     &                   LMO,IOcc,
     &                   CIEig,CIVec,Osc,
     &                   TGNX,TGNY,TGNZ,TMomSq,
     &                   TNNX,TNNY,TNNZ,
     &                   AUToDb)
c
c     Output module of excited states properties for post-process
c     graphic package.
c     On input:
c      NCSFs  ... The number of Configuration state functions.
c      NState ... The number of excited states printed to output file.
c      NOCS   ... The number of occupied MOs in CI calculation.
c      NVCS   ... The number of virtual MOs in CI calculation.
c      NMOs   ... The number of MOs in CI calculation.
c      IUnit  ... Fortran logical unit number for output.
c      MultCI ... Spin multiplicity in CI calculation.
c      MCIMOS ... Maximum number of MOs in CI calculation.
c      LMO    ... MO number in CI calculation.
c      IOcc   ... Scratch integer array for output of electron
c                 configurations.
c      CIEig  ... Eigenvalues of CI matrix.
c      CIVec  ... Eigenvectors of CI matrix.
c      Osc    ... Oscillator strength.
c      TGNX   ... X components of G-E transition moments in atomic unit.
c      TGNY   ... Y components of G-E transition moments in atomic unit.
c      TGNZ   ... Z components of G-E transition moments in atomic unit.
c      TMomSq ... Norm of G-E transition moments in atomic unit.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H,O-Z)
      Integer a, b
      Dimension LMO(1),IOcc(MCIMOs,1)
      Dimension CIEig(1),CIVec(1),Osc(1)
      Dimension TGNX(1),TGNY(1),TGNZ(1),TMomSq(1)
      Dimension TNNX(1),TNNY(1),TNNZ(1)
      Data NF/6/, NF2/40/
 3010 Format(4(f12.6, 1x))
c     ****************************************************************
c
c     Window information.
      Call AncOut('CI_MO_Numbers',13,LMO,1,1,NMOs,NF,IUnit,1,1)
c
c     Electron configuration in CI matrix.
      Call AncO2('CI_Configuration',16,IOcc,MCIMOS,NMOs,NCSFs,NF2,
     &           IUnit,1,2,.True.)
c
        Do 30 k = 1, NF2
          Do 10 i = 1, NOCS
            IOcc(i,k) = 2
   10     Continue
          Do 20 a = NOCS+1, NMOs
            IOcc(a,k) = 0
   20     Continue
   30   Continue
c
        n = 0
        Do 80 i = 1, NOCS
          Do 70 a = NOCS+1, NMOs
            n = n + 1
            IOcc(i,n) = 1
            IOcc(a,n) = 1
            if (Mod(n,NF2) .Eq. 0) Then
              Call AncO2('CI_Configuration',16,IOcc,MCIMOS,NMOs,n,NF2,
     &                   IUnit,1,2,.False.)
              n = 0
              Do 60 k = 1, NF2
                Do 40 j = 1, NOCS
                  IOcc(j,k) = 2
   40           Continue
                Do 50 b = NOCS+1, NMOs
                  IOcc(b,k) = 0
   50           Continue
   60         Continue
            EndIf
   70     Continue
   80   Continue
c
        if (Mod(n,NF2) .NE. 0)
     &        Call AncO2('CI_Configuration',16,IOcc,MCIMOS,NMOs,n,NF2,
     &                   IUnit,1,2,.False.)
c
c     Eigenvalues for CI matrix.
      Call AncOut('CI_Energies',11,CIEig,1,1,NState,NF,IUnit,0,Junk)
c
c     Eigenvectors for CI matrix.
      Call AncOut('CI_Vectors',10,CIVec,NCSFs,NCSFs,NState,NF,IUnit,0,
     &            Junk)
c
      If (MultCI .NE. 1) Return
c
c     Oscillator strength.
      Call AncOut('CI_OscillatorStrength',21,Osc,1,NState,1,NF,IUnit,0,
     &            Junk)
c
c     Transition moments between ground state and Excited states.
c     Unusual manipulation of output modules.
      Call AncHd('CI_TransitionMoments',20,NState,4,NF,IUnit)
        Do 90 i = 1, NState
          Total = DSqRt(TMomSq(i))
          TX    = -TGNX(i)
          TY    = -TGNY(i)
          TZ    = -TGNZ(i)
          Write(IUnit,3010) TX,TY,TZ,Total
   90   Continue
c
c     Dipole moments for excited states.
c     Unusual manipulation of output modules.
      Call AncHd('CI_Dipoles',10,NState,4,NF,IUnit)
        DO 100 ISTATE=1,NSTATE
          MA = ISTATE*(ISTATE+1)/2
          XE =-AUTODB*TNNX(MA)
          YE =-AUTODB*TNNY(MA)
          ZE =-AUTODB*TNNZ(MA)
          TE = DSQRT(XE*XE+YE*YE+ZE*ZE)
          WRITE(IUnit,3010) XE,YE,ZE,TE
  100   CONTINUE
c
      Return
c
      End
c
c@(#)=AncGP(Sub)
      Subroutine AncGP(NAtoms,NBasis,MaxBas,IUnit,
     &                 DX,DY,DZ,DT,
     &                 AtmChg,Eig,C,GChrg)
c
c     Output module of ground state properties for post-process graphic
c     package
c     On input:
c      NAtoms ... The number of atoms.
c      NBasis ... The number of basis functions.
c      MaxBas ... The maximum number of basis functions (used as an
c                 adjustable dimension size).
c      IUnit  ... Fortran logical unit number for output.
c      DX     ... X-Dipole component in units of Debye.
c      DY     ... Y-Dipole component in units of Debye.
c      DZ     ... Z-Dipole component in units of Debye.
c      DT     ... Norm of Dipole mements in units of Debye.
c      AtmChg ... Total atomic charges.
c      Eig    ... Orbital energies in units of Hartree.
c      C      ... Molecular Orbital coefficients.
```

```
c      GChrg  ... Atomic orbital densities (equivalent to diagonal
c                 elements of density matrix).
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H,O-Z)
      Dimension AtmChg(1),Eig(1),C(MaxBas,1),GChrg(1)
      Dimension D(4)
      DATA NF/6/
c     ****************************************************************
c
c     Dipole moments
      D(1) = DX
      D(2) = DY
      D(3) = DZ
      D(4) = DT
      Call AncOut('Dipole',6,D,1,1,4,NF,IUnit,0,Junk)
c
c     Atomic charges.
      Call AncOut('Charges',7,AtmChg,1,NAtoms,1,NF,IUnit,0,Junk)
c
c     Orbital energies.
      Call AncOut('MO_Energies',11,Eig,1,1,NBasis,NF,IUnit,0,Junk)
c
c     Molecular orbital coefficients.
      Call AncOut('MO_Coefficients',15,C,MaxBas,NBasis,NBasis,NF,
     &            IUnit,0,Junk)
c
c     Atomic orbital densities.
      Call AncOut('AO_Densities',12,GChrg,1,1,NBasis,NF,IUnit,0,Junk)
c
      Return
c
      End
c
c@(#)=AncHd(Sub)
      Subroutine AncHd(String,LenStr,NRow,NClm,NField,IUnit)
c
c     Header output for Anchor II.
c     On input:
c      String ... String which indicates item to output.
c      LenStr ... Length of string.
c      NRow   ... Array size in row.
c      NClm   ... Array size in column.
c      NField ... The number of data in a row.
c      IUnit  ... Fortran logical unit number for output.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H, O-Z)
      Character*1 String(*)
      Logical First
      Data First/.True./
 3010 Format('%%',77a1)
 3020 Format('%NumRow =', i6)
 3030 Format('%NumColumn =', i6)
 3040 Format('%NumField =', i6)
c     ****************************************************************
      If (LenStr .GT. 77) Stop 'Abend in AncOut'
c
      If (.not.First) Write(IUnit,*) ' '
c
      Write(IUnit,3010) (String(i), i=1, LenStr)
      If (NRow .GT. 1) Write(IUnit,3020) NRow
      If (NClm .GT. 1) Write(IUnit,3030) NClm
      If (NRow .GT. 1 .And. NClm .GT. NField) Write(IUnit,3040) NField
c
      If (First) First=.False.
c
      Return
c
      End
c
c@(#)=AncI(Sub)
      Subroutine AncI(IArray,MaxDim,NRow,NClm,NField,IUnit,IFtype)
c
c     Output module of Integer data for post-process graphic package.
c     On input:
c      IArray ... Integer Array to output.
c      Maxdim ... Adjustable dimension size of 'IArray'.
c      NRow   ... Array size in row.
c      NClm   ... Array size in column.
c      NField ... The number of data in a row.
c      IUnit  ... Fortran logical unit number for output.
c      IFType ... Format type for output.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H, O-Z)
      Dimension IArray(MaxDim,1)
 3010 Format(6(i8, 1x))
 3020 Format(40i2)
c     ****************************************************************
      If (IFType .Eq. 1) Assign 3010 to IFmt
      If (IFType .Eq. 2) Assign 3020 to IFmt
      If (IFType .NE. 1 .And. IFType .NE. 2) Stop 'Abend in AncI.'
c
      Ilower = 1
      IUpper = NField
c
   10 If (IUpper .GT. NClm) IUpper = NClm
c
        Do 20 i = 1, NRow
          Write(IUnit,IFmt) (IArray(i,j), j = ILower, IUpper)
   20   Continue
c
        If (IUpper .Eq. NClm) Return
c
        ILower = ILower + NField
        IUpper = IUpper + NField
c
      Goto 10
c
      End
c
c@(#)=AncMol(Sub)
      Subroutine AncMol (NAtDum,NAtoms,IUnit,
     &                   ChmFrm,NChrs,WMol,
     &                   ModelH,
     &                   IAtN,NA,NB,NC,IAN,
     &                   ZCoord,Coord,
     &                   Elmnt)
c
c     Output module of molecular structure information for post-process
c     graphic package.
c     On input:
c      NAtDum ... The number of atoms containing dummy atoms.
c      NAtoms ... The number of atoms in a molecule.
c      IUnit  ... Fortran logical unit number for output.
c      ChmFrm ... Chemical formula.
c      NChrs  ... Number of characters in ChmFrm.
c      WMol   ... Molecular weight.
c      ModelH ... Model Hamiltonian.
c      IAtN   ... Atomic numbers in Z matrix.
c      NA     ... Indices for Z matrix.
c      NB     ... Indices for Z matrix.
c      NC     ... Indices for Z matrix.
c      IAN    ... Atomic numbers of constituent atoms.
c      ZCoord ... Z matrix elements.
c      Coord  ... Cartesian coordinates of a molecule.
c      Elmnt  ... Array of atomic symbols.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H,O-Z)
      Character*256 ChmFrm
      Character*8 ModelH
      Character*2 Elmnt
      Dimension ZCoord(3,1),Coord(3,1)
      Dimension IAtN(1),NA(1),NB(1),NC(1),IAN(1)
      Dimension Elmnt(1)
      Data NF/6/
 3010 Format(80a1)
 3020 Format(a8)
 3030 Format(a2, 1x, 3(f18.12,1x),3i6)
 3040 Format(a2, 1x, 3(f20.12,1x))
c     ****************************************************************
c
c     Chemical formula.
c      ... Unusual manipulation of output modules.
      Call AncHd('ChemicalFormula',15,1,1,NF,IUnit)
      Write(IUnit,3010) (ChmFrm(i:i),i=1,NChrs)
```

```
c
c     Molecular weight.
      Call AncOut('MolecularWeight',15,WMol,1,1,1,NF,IUnit,0,Junk)
c     Model Hamiltonian.
c      ... Unusual manipulation of output modules.
      Call AncHd('ModelHamiltonian',16,1,1,NF,IUnit)
      Write(IUnit,3020) ModelH
c
c     Z Matrix.
c      ... Unusual manipulation of output modules.
      Call AncHd('ZMatrix',7,NAtDum,1,NF,IUnit)
        Do 10 i = 1, NAtDum
          Write(IUnit,3030) Elmnt(IAtN(i)),(ZCoord(j,i),j = 1, 3),
     &                      NA(i),NB(i),NC(i)
   10   Continue
c
c     Cartesian coordinate.
c      ... Unusual manipulation of output modules.
      Call AncHd('Coordinate',10,NAtoms,1,NF,IUnit)
        Do 20 i = 1, NAtDum
          Write(IUnit,3040) Elmnt(IAN(i)),(Coord(j,i),j = 1, 3)
   20   Continue
c
      Return
c
      End
c
c@(#)=AncO2(Sub)
      Subroutine AncO2(String,LenStr,IArray,MaxDim,NRow,NClm,NField,
     &                 IUnit,Iflg,IFType,Init)
c
c     Output module for post-process graphic package ... Type II.
c     On input:
c      String ... String which indicates item to output.
c      LenStr ... Length of string.
c      IArray ... Array to output (double precision).
c      Maxdim ... Adjustable dimension size of 'IArray'.
c      NRow   ... Array size in row.
c      NClm   ... Array size in column.
c      NField ... The number of data in a row.
c      IUnit  ... Fortran logical unit number for output.
c      IFlg   ... Flag for data type.
c                 0 ... Data type of 'IArray' is Real*8.
c                 1 ... Data type of 'IArray' is Integer.
c      IFType ... Format type for output.
c      Init   ... Flag for printing header.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H, O-Z)
      Character*256 String
      Logical Init
      Dimension IArray(*)
c     ****************************************************************
c
c     Write out header.
      If (Init) Call AncHd (String,LenStr,NRow,NClm,NField,IUnit)
      If (Init) Return
c
      if (IFlg .Eq. 0) Call AncR (IArray,MaxDim,NRow,NClm,NField,IUnit,
     &                            IFType)
      If (IFlg .Eq. 1) Call AncI (IArray,MaxDim,NRow,NClm,NField,IUnit,
     &                            IFType)
c
      If (IFlg .NE. 0 .And. IFlg .NE. 1) Stop 'Abend in AncOut.'
c
      Return
c
      End
c=>   92/Oct A.M.   ...   New Routine is appended.
c
c@(#)=AncOut(Sub)
      Subroutine AncOut(String,LenStr,IArray,MaxDim,NRow,NClm,NField,
     &                  IUnit,Iflg,IFType)
c
c     Output module for post-process graphic package.
c     On input:
c      String ... String which indicates item to output.
c      LenStr ... Length of string.
c      IArray ... Array to output (double precision).
c      Maxdim ... Adjustable dimension size of 'IArray'.
c      NRow   ... Array size in row.
c      NClm   ... Array size in column.
c      NField ... The number of data in a row.
c      IUnit  ... Fortran logical unit number for output.
c      IFlg   ... Flag for data type.
c                 0 ... Data type of 'IArray' is Real*8.
c                 1 ... Data type of 'IArray' is Integer.
c      IFType ... Format type for output.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H, O-Z)
      Character*256 String
      Dimension IArray(*)
c     ****************************************************************
c
c     Write out header.
      Call AncHd(String,LenStr,NRow,NClm,NField,IUnit)
c
      if (IFlg .Eq. 0) Call AncR (IArray,MaxDim,NRow,NClm,NField,IUnit,
     &                            IFType)
      If (IFlg .Eq. 1) Call AncI (IArray,MaxDim,NRow,NClm,NField,IUnit,
     &                            IFType)
c
      If (IFlg .NE. 0 .And. IFlg .NE. 1) Stop 'Abend in AncOut.'
c
      Return
c
      End
c
c@(#)=AncR(Sub)
      Subroutine AncR(Array,MaxDim,NRow,NClm,NField,IUnit,IFtype)
c
c     Output module of Real*8 data for post-process graphic package.
c     On input:
c      Array  ... Real*8 Array to output.
c      Maxdim ... Adjustable dimension size of 'Array'.
c      NRow   ... Array size in row.
c      NClm   ... Array size in column.
c      NField ... The number of data in a row.
c      IUnit  ... Fortran logical unit number for output.
c      IFType ... Format type for output ... not used at present.
c     On output:
c      None.
c
c     ****************************************************************
      Implicit Real*8 (A-H, O-Z)
      Dimension Array(MaxDim,1)
 3010 Format(6(f12.5, 1x))
c     ****************************************************************
      IFType = 1
      If (IFType .Eq. 1) Assign 3010 to IFmt
c
      Ilower = 1
      IUpper = NField
c
   10 If (IUpper .GT. NClm) IUpper = NClm
c
        Do 20 i = 1, NRow
          Write(IUnit,IFmt) (Array(i,j), j = ILower, IUpper)
   20   Continue
c
        If (IUpper .Eq. NClm) Return
c
        ILower = ILower + NField
        IUpper = IUpper + NField
c
      Goto 10
c
      End
c
C@(#)=ANL2E(SUB) ... CNDO2/3R
      SUBROUTINE ANL2E(COORD,ZETA,GAB,GAA,IAN,NATOMS,NC,MDIM)
C
C     CALCULATION OF ERIS OVER S-TYPE SLATER FUNCTIONS.
C     FROM CNDO2/3R.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON(17)
      COMMON /FCTRL / FACT(0:50)
      DIMENSION COORD(3,*),ZETA(*),GAB(*),GAA(*),IAN(*),NC(MDIM,*)
      DATA ZERO/0.0D+00/,TWO/2.0D+00/
C     ****************************************************************
```

```
      DO 50 IAT=1,NATOMS
        IA   =IAN(IAT)
        NCI  =NC(IA,1)
        NCI2 =NCI+NCI
        ZI   =ZETA(IAT)
        MX   =IAT*(IAT-1)/2
        IATM1=IAT-1
C
        IF(IATM1.EQ.0) GO TO 30
C
C     2 CENTER ERIS
        CIX=COORD(1,IAT)
        CIY=COORD(2,IAT)
        CIZ=COORD(3,IAT)
C
        DO 20 JAT=1,IATM1
          JA  =IAN(JAT)
          NCJ =NC(JA,1)
          NCJ2=NCJ+NCJ
          ZJ  =ZETA(JAT)
C
          RX  =COORD(1,JAT)-CIX
          RY  =COORD(2,JAT)-CIY
          RZ  =COORD(3,JAT)-CIZ
          RIJ =DSQRT(RX*RX+RY*RY+RZ*RZ)
C
          TRM1=SS(0,0,0,(NCJ2-1),0,ZERO,(TWO*ZJ*RIJ))*(RIJ/TWO)**NCJ2
          TRM2=ZERO
          DO 10 I=1,NCI2
            TMU1=DFLOAT(I)*(TWO*ZI)**(NCI2-I)*(RIJ/TWO)**(NCI2+NCJ2-I)
            TMU2=SS((NCI2-I),0,0,(NCJ2-1),0,(TWO*ZI*RIJ),(TWO*ZJ*RIJ))
            TMU =TMU1*TMU2
            TMD =TWO*DFLOAT(NCI)*FACT(NCI2-I)
            TRM2=TRM2+TMU/TMD
   10     CONTINUE
          MA=MX+JAT
          GAB(MA)=AUTOEV*(TRM1-TRM2)*((TWO*ZJ)**(NCJ2+1))/FACT(NCJ2)
   20   CONTINUE
C
C     1 CENTER ERIS
   30   TRM1=FACT(NCI2-1)/((TWO*ZI)**NCI2)
        TRM2=ZERO
        DO 40 I=1,NCI2
          TMU =DFLOAT(I)*FACT(NCI*4-I-1)*(TWO*ZI)**(NCI2-I)
          TMD =TWO*DFLOAT(NCI)*FACT(NCI2-I)*(TWO*(ZI+ZI))**(NCI*4-I)
          TRM2=TRM2+TMU/TMD
   40   CONTINUE
        MA=IAT*(IAT+1)/2
        GAA(IAT)=AUTOEV*(TRM1-TRM2)*(TWO*ZI)**(NCI2+1)/FACT(NCI2)
        GAB(MA) =GAA(IAT)
   50 CONTINUE
C
      RETURN
      END
C
C@(#)=ASMOUT(SUB)
      SUBROUTINE ASMOUT(A,IAN,NATOMS)
C
C     OUTPUT MODULE FOR ATOM BASED SYMMETRIC MATRIX.
C     PRINT OUT LOWER TRIANGLE MATRIX.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*2 ELMNT,EL
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      COMMON /PTbl2 / ELMNT(99)
      DIMENSION A(*),IAN(*)
 2000 Format(1H ,5X,11(8X,I3))
 2010 Format(1H ,I3,2X,A2,1X,11(1X,F10.6))
C     ****************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=6
      ELSE
        IW=11
      END IF
C
      ILOWER=1
      IUPPER=IW
C
   10 IF(IUPPER.GT.NATOMS) THEN
        IUPPER=NATOMS
      END IF
C
      WRITE(IOUT,2000) (I,I=ILOWER,IUPPER)
C
      DO 20 I=ILOWER,NATOMS
        NCLMS=I-ILOWER+1
        IF(NCLMS.GT.IW) THEN
          IU=ILOWER+IW-1
        ELSE
          IU=I
        END IF
C
        MA=I*(I-1)/2
C
        IA=IAN(I)
        EL=ELMNT(IA)
        WRITE(IOUT,2010) I,EL,(A(MA+J),J=ILOWER,IU)
C
   20 CONTINUE
C
      IF(IUPPER.EQ.NATOMS) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
C
C@(#)=ATDIST(SUB)
      SUBROUTINE ATDIST(C,IAN,NATOMS)
C
C     PRINTING OF INTERATOMIC DISTANCES.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*2 ELMNT2,EL
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PTbl2 / ELMNT2(99)
      COMMON /PRWDTH/ IWDTH
      DIMENSION C(*),IAN(*)
      DIMENSION RIJ(11)
 2000 Format(1H ,21('=')/
     &       1H ,'Distance matrix (Ang)'/
     &       1H ,21('='))
 2010 Format(1H ,4X,11(8X,I3))
 2020 Format(1H ,I3,2X,A2,1X,11(1X,F10.6))
C     ****************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=6
      ELSE
        IW=11
      END IF
C
      ILOWER=1
      IUPPER=IW
C
      WRITE(IOUT,2000)
C
   10 IF(IUPPER.GT.NATOMS) THEN
        IUPPER=NATOMS
      END IF
C
      WRITE(IOUT,2010) (I,I=ILOWER,IUPPER)
C
      DO 30 I=ILOWER,NATOMS
        IU =MIN0((I-ILOWER+1),IW)
        MA =(I-1)*3+1
        CXI=C(MA)
        CYI=C(MA+1)
        CZI=C(MA+2)
        DO 20 K=1,IU
          J =ILOWER+K-1
          MA=(J-1)*3+1
          CX=CXI-C(MA)
          CY=CYI-C(MA+1)
          CZ=CZI-C(MA+2)
          RIJ(K)=DSQRT(CX*CX+CY*CY+CZ*CZ)
   20   CONTINUE
        IA=IAN(I)
        EL=ELMNT2(IA)
        WRITE(IOUT,2020) I,EL,(RIJ(K),K=1,IU)
   30 CONTINUE
C
      IF(IUPPER.EQ.NATOMS) RETURN
C
```

```
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
C
C@(#)=BANGLE(SUB) ... MOPAC(BANGLE)
      SUBROUTINE BANGLE(XYZ,I,J,K,ANGLE)
C
C     ==========================================================
C     MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC
C     ==========================================================
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION XYZ(3,*)
C*****************************************************************
C
C BANGLE CALCULATES THE ANGLE BETWEEN ATOMS I,J, AND K. THE
C        CARTESIAN COORDINATES ARE IN XYZ.
C
C*****************************************************************
      D2IJ = (XYZ(1,I)-XYZ(1,J))**2+
     1       (XYZ(2,I)-XYZ(2,J))**2+
     2       (XYZ(3,I)-XYZ(3,J))**2
      D2JK = (XYZ(1,J)-XYZ(1,K))**2+
     1       (XYZ(2,J)-XYZ(2,K))**2+
     2       (XYZ(3,J)-XYZ(3,K))**2
      D2IK = (XYZ(1,I)-XYZ(1,K))**2+
     1       (XYZ(2,I)-XYZ(2,K))**2+
     2       (XYZ(3,I)-XYZ(3,K))**2
C*MOS XY = SQRT(D2IJ*D2JK)
      XY = DSQRT(D2IJ*D2JK)
      TEMP = 0.5D0 * (D2IJ+D2JK-D2IK) / XY
      IF (TEMP .GT. 1.0D0) TEMP=1.0D0
      IF (TEMP .LT. -1.0D0) TEMP=-1.0D0
C*MOS ANGLE = ACOS( TEMP )
      ANGLE = DACOS( TEMP )
      RETURN
      END
C
C@(#)=BETXYZ(FUN)
      DOUBLE PRECISION FUNCTION BETXYZ(W1,W2,I,J,K,TGN,TNN,CIEIG,NCSFS,
     &                                 MDIM1,MDIM2)
C
C     Get hyperpolarizabilities for incident light W1 and W2.
C     On input:
C      W1      ... ENERGY OF INCIDENT LIGHT (ATOMIC UNIT).
C      W2      ... ENERGY OF INCIDENT LIGHT (ATOMIC UNIT).
C      TGN     ... <G/R/E> (ATOMIC UNITS)
C                   G : TOTAL WAVE FUNCTION OF GROUND STATE.
C                   E : TOTAL WAVE FUNCTION OF EXCITED STATES.
C                   R : POSITION VECTOR OF ELECTRON.
C      TNN     ... <E/R/E'>
C      CIEIG   ... TRANSITION ENERGIES (ATOMIC UNITS).
C      MDIM1   ... ADJUSTABLE DIMENSION SIZE OF ARRAY TGN.
C      MDIM2   ... ADJUSTABLE DIMENSION SIZE OF ARRAY TNN.
C     On output:
C      BETXYZ ... BETA(-(W1+W2);W1,W2) (ATOMIC UNITS).
C                 HYPERPOLARIZABILITY FOR INCIDENT LIGHT W1 AND W2.
C                  SHG  ... BETA(-2W; W, W).
C                  EOPE ... BETA( -W; 0, W).
C                  OR   ... BETA(  0; W,-W).
C
C     REFERENCES.
C      J.A.MORRELL AND A.C.ALBRECHT, CHEM.PHYS.LETT.,64,46(1979)
C      S.P.KARNA,P.N.PRASAD ,AND M.DUPUIS, J.CHEM.PHYS.,94,1171(1991)
C      J.E.RICE ET AL, J.CHEM.PHYS.,93,8828(1990)
C      J.F.WARD, REV.MOD.PHYS.,37,1(1965)
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION TGN(MDIM1,*),TNN(MDIM2,*),CIEIG(*)
      DATA ZERO/0.0D+00/,PT25/0.25D+00/,ONE/1.0D+00/
C     ***************************************************************
      BIJK=ZERO
C
      DO 20 IS=1,NCSFS
C
        RGN1I =TGN(IS,I)
        RGN1J =TGN(IS,J)
        RGN1K =TGN(IS,K)
        WGN1  =CIEIG(IS)
        W1PW1 =ONE/(WGN1+W1)
        W1PW2 =ONE/(WGN1+W2)
        W1MW12=ONE/(WGN1-(W1+W2))
C
        DO 10 JS=1,NCSFS
C
          RGN2I =TGN(JS,I)
          RGN2J =TGN(JS,J)
          RGN2K =TGN(JS,K)
          WGN2  =CIEIG(JS)
          W2MW1 =ONE/(WGN2-W1)
          W2MW2 =ONE/(WGN2-W2)
          W2PW12=ONE/(WGN2+(W1+W2))
C
          IF(JS.GT.IS) THEN
            MA=JS*(JS-1)/2+IS
          ELSE
            MA=IS*(IS-1)/2+JS
          END IF
C
          RNNI =TNN(MA,I)
          RNNJ =TNN(MA,J)
          RNNK =TNN(MA,K)
C
          RIJK=RGN1I*RNNJ*RGN2K
          RIKJ=RGN1I*RNNK*RGN2J
          RJIK=RGN1J*RNNI*RGN2K
          RJKI=RGN1J*RNNK*RGN2I
          RKIJ=RGN1K*RNNI*RGN2J
          RKJI=RGN1K*RNNJ*RGN2I
C
          BIJK=BIJK
     &         + RIJK*W1MW12*W2MW2 + RIKJ*W1MW12*W2MW1
     &         + RJIK*W1PW1*W2MW2  + RJKI*W1PW1*W2PW12
     &         + RKIJ*W1PW2*W2MW1  + RKJI*W1PW2*W2PW12
C
   10   CONTINUE
C
   20 CONTINUE
C
      BETXYZ=BIJK*(-PT25)
C
      RETURN
      END
C
C@(#)=BINOM(FUN) ... CNDO2/3R(BINOM)
      FUNCTION BINOM(N,I)
C
C     ==========================================================
C     CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R
C     ==========================================================
C     DOUBLE PRECISION FUNCTION BINOM ... FROM CNDO 2/3R
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /FCTRL / FACT(0:50)
C     ***************************************************************
      J=N-I
      BINOM=FACT(N)/(FACT(I)*FACT(J))
      RETURN
      END
C
C@(#)=BINTGS(SUB) ... CNDO2/3R(BINTGS)
      SUBROUTINE BINTGS(X,K)
C
C     ==========================================================
C     CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R
C     ==========================================================
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /AUXINT/ A(17),B(17)
      COMMON /FCTRL / FACT(0:50)
C     FILLS ARRAY OF B-INTEGRALS. NOTE THAT B(I) IS B(I-1) IN THE
C     USUAL NOTATION
C     FOR X.GT.3                    EXPONENTIAL FORMULA IS USED
C     FOR 2.LT.X.LE.3 AND K.LE.10   EXPONENTIAL FORMULA IS USED
C     FOR 2.LT.X.LE.3 AND K.GT.10   15 TERM SERIES IS USED
C     FOR 1.LT.X .E.2 AND K.LE.7    EXPONENTIAL FORMULA IS USED
C     FOR 1.LT.X.LE.2 AND K.GT.7    12 TERM SERIES IS USED
C     FOR .5.LT.X.LE.1 AND K.LE.5   EXPONENTIAL FORMULA IS USED
C     FOR .5.LT.X.LE.1 AND K.GT.5    7 TERM SERIES IS USED
C     FOR X.LE..5                    6 TERM SERIES IS USED
C     ***************************************************************
      I0=0
      ABSX=DABS(X)
```

```
      IF(ABSX.GT.3.   ) GO TO 120
   10 IF(ABSX.GT.2.   ) GO TO 20
   40 IF(ABSX.GT.1.   ) GO TO 50
   70 IF(ABSX.GT..5   ) GO TO 80
  100 IF(ABSX.GT..000001  ) GO TO 110
      GO TO 170
  110 LAST=6
      GO TO 140
   80 IF(K.LE.5) GO TO 120
   90 LAST=7
      GO TO 140
   50 IF(K.LE.7) GO TO 120
   60 LAST=12
      GO TO 140
   20 IF(K.LE.10) GO TO 120
   30 LAST=15
      GO TO 140
  120 EXPX=DEXP(X)
      EXPMX=1.D0/EXPX
      B(1)=(EXPX-EXPMX)/X
      DO 130 I=1,K
  130 B(I+1)=( FLOAT(I)*B(I)+(-1.D0)**I*EXPX-EXPMX)/X
      GO TO 190
  140 DO 160 I=I0,K
      Y=0.D0
      DO 150 M=I0,LAST
  150 Y=Y+(-X)**M*(1.D0-(-1.D0)**(M+I+1))/(FACT(M)*DFLOAT(M+I+1))
  160 B(I+1)=Y
      GO TO 190
  170 DO 180 I=I0,K
  180 B(I+1)=(1.D0-(-1.D0)**(I+1))/DFLOAT(I+1)
  190 CONTINUE
      RETURN
      END
c
c$(#)=CAppnd(Sub)
      Subroutine CAppnd (Str1,Str2,NChrs)
      Implicit Integer (a-z)
c
c     Append character string.
c     On input:
c      Str1   ... Character string to be appended.
c      Str2   ... Character string to append.
c      NChrs  ... Number of characters in Str1.
c     On output:
c      Str1   ... Character string after appended.
c      NChrs  ... Number of characters in Str1 after appended.
      Character*(*) Str1,Str2
      Character*1 iBl
      Save iBl
      Data iBl/1h /
c
      LenSt2=Len(Str2)
c
      Do 10 i=LenSt2,1,-1
        If (Str2(i:i) .ne. iBl) Then
        Len2=i
        GoTo 20
        EndIf
   10 Continue
      Stop 'CAppnd.  Number of characters in Str2 is zero'
c
   20 If (NChrs .eq. 0) Then
        Str1=Str2
      ELse
        Str1=Str1(1:NChrs)//Str2
      EndIf
c
      NChrs=NChrs+Len2
c
      Return
      End
c
c@(#)=CClear(Sub)
      Subroutine CClear(CArray,ISize)
      Implicit Integer (a-z)
c
c     Clear CArray.
c     On input:
c      CArray ... Character variable to clear.
c      ISize  ... Size of CArray.
c     On output.
c      CArray ... Blank-cleared character variable.
c
      Character*(*) CArray
      Dimension CArray(*)
      Character*1 Blank
      Save Blank
      Data Blank/1h /
c
      Do 10 i=1,ISize
        CArray(i)=Blank
   10 Continue
c
      Return
      End
c
c@(#)=CForm(Sub)
      Subroutine CForm(IOut,IAN,NAtoms,AtMass,Elmnt,ChmFrm,NChrs,WMol)
      Implicit Real*8 (a-h,o-z)
      Include 'MSSIZE'
c
c     Get Chemical formula and molecular weight.
c     On input:
c      IOut   ... Fortran logical unit for standard output.
c      IAN    ... Array for atomic number.
c      NAtoms ... Number of atoms.
c      AtMass ... Array for atomic weight.
c      Elmnt  ... Array for atomic symbols.
c     On output:
c      ChmFrm ... Chemical formula.
c      NChrs  ... Number of characters in ChmFrm.
c      WMol   ... Molecular weight.
c
      Character*256 ChmFrm
      Character*4 Chr4
      Character*2 Elmnt,CArray,ChrTmp
      Dimension NIAN(MaxIAN),IArray(MaxIAN),CArray(MaxIAN)
      Dimension IAN(*),AtMass(*),Elmnt(*)
      Data Zero/0.0d+00/
 2000 Format(1h ,'Chemical formula ... ',68a1)
 2010 Format(1h ,'Molecular weight ... Containing unknown atoms.')
 2020 Format(1h ,'Molecular weight ...',f10.4)
c
      Call IClear(NIAN,MaxIAN)
      Call CClear(ChmFrm,1)
      NChrs=0
      iFlg=0
      WMol=Zero
c
      Do 10 i=1,NAtoms
        NIAN(IAN(i))=NIAN(IAN(i))+1
        WMol=WMol+AtMass(IAN(i))
        If (AtMass(IAN(i)) .lt. Zero) iFlg=1
   10 Continue
c
c     First is carbon atom.
      IANum=6
      If (NIAN(IANum) .gt. 0) Then
        Call CAppnd(ChmFrm,Elmnt(IANum),NChrs)
      EndIf
      If (NIAN(IANum). gt. 1) Then
        Call IToC(NIAN(IANum),Chr4)
        Call CAppnd(ChmFrm,Chr4,NChrs)
      EndIf
c
c     Second is hydrogen ...
      IANum=1
      If (NIAN(IANum) .gt. 0) Then
        Call CAppnd(ChmFrm,Elmnt(IANum),NChrs)
      EndIf
      If (NIAN(IANum). gt. 1) Then
        Call IToC(NIAN(IANum),Chr4)
        Call CAppnd(ChmFrm,Chr4,NChrs)
      EndIf
c
c     Then alphabetic ...
      n=0
      Do 20 i=1,MaxIAN
        If (i .eq. 1) GoTo 20
        If (i .eq. 6) GoTo 20
        If (NIAN(i) .gt. 0) Then
          n=n+1
          IArray(n)=i
          CArray(n)=Elmnt(i)
        EndIf
```

```
   20 Continue
c
      Call CSort2(CArray,IArray,ChrTmp,n)
c
      Do 30 i=1,n
        IANum=IArray(i)
        If (NIAN(IANum) .gt. 0) Then
          Call CAppnd(ChmFrm,Elmnt(IANum),NChrs)
        EndIf
        If (NIAN(IANum). gt. 1) Then
          Call IToC(NIAN(IANum),Chr4)
          Call CAppnd(ChmFrm,Chr4,NChrs)
        EndIf
   30 Continue
c
c     Then print.
      Write(IOut,2000) (ChmFrm(i:i),i=1,NChrs)
      If (iFlg. eq. 1) Then
        Write(IOut,2010)
      Else
        Write(IOut,2020) WMol
      EndIf
c
      Return
      End
C
C@(#)=CHGUNT(SUB)
      SUBROUTINE CHGUNT(A,N,CONST)
C
C     UNIT TRANSFORMATION.
C     A(I)=A(I)*CONST FOR I=1 TO N
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(*)
C     ****************************************************************
      DO 10 I=1,N
        A(I)=A(I)*CONST
   10 CONTINUE
C
      RETURN
      END
C
C@(#)=CHKKEY(SUB)
      SUBROUTINE CHKKEY(IPRM,NPRMS,IVAL)
C
C     CHECK IF KEYWORD IS MULTI-DEFINED IN OPTION CARDS.
C     ****************************************************************
      COMMON /IO    / IN,IOUT,IFILE(10)
      DIMENSION IPRM(*)
 9910 Format(1H ,50('X')/
     &       1H ,'KEY-WORD WAS MULTI-DEFINED. CHECK YOUR INPUT DATA.'/
     &       1H ,50('X'))
C     ****************************************************************
      N=0
      DO 10 I=1,NPRMS
        IF(IPRM(I).NE.0) THEN
          N=N+1
          MA=I
        END IF
   10 CONTINUE
C
      IF(N.EQ.0) THEN
        IVAL=0
        RETURN
      ELSE IF(N.EQ.1) THEN
        IVAL=MA
        RETURN
      ELSE
        WRITE(IOUT,9910)
        CALL MERROR
      END IF
C
      END
C
C@(#)=CHKTYP(FUN)
      INTEGER FUNCTION CHKTYP(CARD,IB)
C
C     CHECK THE TYPE OF INPUTED DATA BLOCK.
C
C     On input:
C      CARD ... CARD CONTAINS DATA BLOCK.
C      IB   ... CHECK DATA BLOCK THAT START FROM IB-TH COLUMN.
C     On output:
C      CHKTYP
C      1 ... INTEGER TYPE.
C      2 ... REAL TYPE.
C      3 ... CHARACTER TYPE.
C
C     ****************************************************************
      CHARACTER*80 CARD
      CHARACTER*1  NUMBER(0:9),PLUS,MINUS,POINT,SPACE,CHR1
      DATA NUMBER/'0','1','2','3','4','5','6','7','8','9'/
      DATA PLUS/'+'/,MINUS/'-'/,POINT/'.'/,SPACE/' '/
      DATA MAXCLM/80/
C     ****************************************************************
      ITYPE=1
      IS=IB
      CHR1=CARD(IS:IS)
      IF(CHR1.EQ.PLUS.OR.CHR1.EQ.MINUS) THEN
        IS=IS+1
      END IF
      IF(CHR1.EQ.POINT) THEN
        ITYPE=2
        IS=IS+1
      END IF
      CHR1=CARD(IS:IS)
      IF(CHR1.EQ.SPACE) THEN
        ITYPE=3
        GO TO 30
      END IF
      DO 20 I=IS,MAXCLM
        CHR1=CARD(I:I)
        IF(CHR1.EQ.SPACE) THEN
          GO TO 30
        END IF
        DO 10 J=0,9
          IF(CHR1.EQ.NUMBER(J)) THEN
            GO TO 20
          END IF
   10   CONTINUE
        ITYPE=3
        GO TO 30
   20 CONTINUE
C
   30 CHKTYP=ITYPE
C
      RETURN
      END
c
c@(#)=CIChrg(Sub)
      Subroutine CIChrg (CIVec, CMO, CI2Occ, CI2Vac, EChrg, NBasis,
     &                   NOCS, NVCS)
c
c     ****************************************************************
c     Computation of total atomic orbital charges for the i-th SCI
c     excited state.
c
c     On input:
c      NBasis ... No. of basis functions.
c      NOCS   ... No. of occupied MOs considered in CI calculation.
c      NVCS   ... No. of virtual MOs considered in CI calculation.
c      CIVec  ... i-th CI vector (packed form).
c      CMO    ... MO coefficients (packed form). Only containing
c                 NOCS plus NVCS vectors.
c      CI2Occ ... Scratch array.
c      CI2Vac ... Scratch array.
c     On output:
c      EChrg  ... Total atomic orbital charges for the i-th SCI
c                 excited state.
c     ****************************************************************
      Implicit Real*8 (a-h, o-z)
      Integer*4 a
      Dimension CIVec(*), CMO(*),CI2Occ(*), CI2Vac(*), EChrg(*)
      Data Zero /0.0d+00/
c     ****************************************************************
      Do 10 a = 1, NVCS
        CI2Vac(a) = Zero
   10 Continue
c
      n = 0
      Do 30 i = 1, NOCS
        x = Zero
        Do 20 a = 1, NVCS
          n = n + 1
          vec2 = CIVec(n)*CIVec(n)
          x = x + vec2
```

```
          CI2Vac(a) = CI2Vac(a) + vec2
   20     Continue
          CI2Occ(i) = x
   30 Continue
c
      Do 60 iAO = 1, NBasis
        x = Zero
        Do 40 i = 1, NOCS
          ma = (i - 1)*NBasis + iAO
          x = x - CI2Occ(i)*CMO(ma)*CMO(ma)
   40     Continue
        Do 50 a = 1, NVCS
          ma = (a + NOCS -1)*NBasis +iAO
          x = x + CI2Vac(a)*CMO(ma)*CMO(ma)
   50     Continue
          EChrg(iAO) = X
   60 Continue
c
      Return
      End
      subroutine cidump
c=====================================================================*
c                                                                     *
c       @@@@      @    @@@@@   @    @  @    @  @@@@@                 *
c      @    @     @    @    @  @    @  @@  @@  @    @                *
c      @          @    @    @  @    @  @ @@ @  @    @                *
c      @          @    @    @  @    @  @    @  @@@@@                 *
c      @    @     @    @    @  @    @  @    @  @                    *
c       @@@@      @    @@@@@    @@@@   @    @  @                    *
c=====================================================================*
c
c-------------   dump sdci eigen vectors
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c
      common /io    / in,iout,ifile(10)
      common /lvci  / locc(mcicsf),lvac(mcicsf),
     &                mocc(mcicsf),mvac(mcicsf)
      common /work01/ ovov(mciint),oovv(mciint),civec(mcicsf,mcicsf),
     &                dummy(1)

c cut for low mem ********************************************************
cc$ohtawara for my trmom calc. (in tmsdst.f)
c      common /mywork01/ aci(mcicsf,mcicsf)
cc$ohtawara for my trmom calc. (in tmsdst.f)
c      do istart=1,mcicsf
c         do iend=1,mcicsf
c            aci(iend,istart) = civec(istart,iend)
c         enddo
c      enddo
c cut for low mem ********************************************************

      nend = ncibas
      call civo02( civec,  cieig,  mcicsf, nend )
c
c$$$      do 100 nomg = momega(1)+1, momega(2)
c$$$        i = lci1( nomg )
c$$$        k = lci3( nomg ) + nocs
c$$$        n = nomg - momega( 1 )
c$$$        locc( n ) = lmo( i )
c$$$        lvac( n ) = lmo( k )
c$$$  100 continue
c$$$c
c$$$      nscibs = momega( 2 ) - momega( 1 )
c$$$      if( isdci0 .eq. 1 ) then
c$$$        nend = ncibas
c$$$      else
c$$$        nend = momega( 3 )
c$$$      endif
c$$$      write( iout, 1000 )
c$$$      call civo00( civec,  cieig,  locc,   lvac,   mcicsf, nscibs,
c$$$     $             nend )
c$$$      write( iout, 1100 )
c$$$      call civo01( civec,  cieig,  locc,   lvac,   mcicsf, nscibs,
c$$$     $             nend )
c
```



```
c
c$$$ 1000 format(1h ,//
c$$$     $       1h ,37('=-')/
c$$$     &       1h ,'eigenvectors of ci hamiltonian matrix'/
c$$$     $       1h ,'(contribution from Hartree-Fock and',
c$$$     $           ' single excitaion states)'/
c$$$     &       1h ,37('=-'))
c$$$ 1100 format(1h ,//
c$$$     $       1h ,37('=-')/
c$$$     &       1h ,'eigenvectors of ci hamiltonian matrix'/
c$$$     $       1h ,'(contribution from double excitaion state type I)'/
c$$$     &       1h ,37('=-'))
c
c
      return
      end
C
C@(#)=CIMOUT(SUB)
      SUBROUTINE CIMOUT(A,LOCC,LVAC,NCSFS)
C
C     OUTPUT MODULE FOR CI HAMILTONIAN MATRIX.
C     PRINT OUT LOWER TRIANGLE MATRIX.
C     CAUTION.
C      MAXIMUM MO NO. PRINTED IN THIS ROUTINE IS LESS THAN 100 (I2).
C      IN CASE OF PRINTING MO WHOSE NO. IS GREATER THAN OR EQUAL TO 100,
C      FORMAT OVERFLOW WILL OCCUR.
C     ******************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      DIMENSION A(*),LOCC(*),LVAC(*)
 2000 Format(1H ,9X,11(5X,I2,'->',I2))
 2010 Format(1H ,I2,'->',I2,4X,11(1X,F10.6))
C     ******************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=6
      ELSE
        IW=11
      END IF
C
      ILOWER=1
      IUPPER=IW
C
   10 IF(IUPPER.GT.NCSFS) THEN
        IUPPER=NCSFS
      END IF
C
      WRITE(IOUT,2000) (LOCC(I),LVAC(I),I=ILOWER,IUPPER)
C
      DO 20 I=ILOWER,NCSFS
        NCLMS=I-ILOWER+1
        IF(NCLMS.GT.IW) THEN
          IU=ILOWER+IW-1
        ELSE
          IU=I
        END IF
C
        MA=I*(I-1)/2
C
        WRITE(IOUT,2010) LOCC(I),LVAC(I),(A(MA+J),J=ILOWER,IU)
C
   20 CONTINUE
C
      IF(IUPPER.EQ.NCSFS) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
      subroutine cimt10
c=====================================================================*
c                                                  @     @@@          *
c       @@@@      @    @    @   @@@@@   @@     @   @                 *
c      @    @     @    @@  @@     @    @ @    @     @                 *
c      @          @    @ @@ @     @      @    @     @                 *
c      @          @    @    @     @      @    @     @                 *
c      @    @     @    @    @     @      @     @   @                 *
c       @@@@      @    @    @     @    @@@@@    @@@                  *
c=====================================================================*
c
c  get ci matrix elements between type I and Hartree-Fock state
```

```
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
      ntau = 1
      do 1 nomg = momega(2)+1, momega(3)
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nu   = kpp*( kpp - 1 )/2 + kppx
        mu   = nu*( nu + 1 )/2
        ncimat = nomg*( nomg - 1 )/2 + ntau
        cimat( ncimat ) = gintg1( mu )

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,i,k,nomg,ntau,cimat(ncimat)
 100  format('( 0 -> 0) -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    1 continue
c
c
c     all done. return to caller.
      call clockm('cimt10.   ')
c
c
      return
      end
      subroutine cimt11
c=====================================================================*
c                                          @       @                  *
c       @@@@      @    @    @   @@@@@   @@      @@                   *
c      @    @     @    @@  @@     @    @ @     @ @                   *
c      @          @    @ @@ @     @      @       @                   *
c      @          @    @    @     @      @       @                   *
c      @    @     @    @    @     @      @       @                   *
c       @@@@      @    @    @     @    @@@@@   @@@@@                 *
c=====================================================================*
c
c  get ci matrix elements between type I states
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)
c
c
c off diagonal part
      do 1 nomg = momega(2)+1, momega(3)
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        do 2 ntau = momega(2)+1, nomg-1
          ir = lci1( ntau )
          it = lci3( ntau ) + nocs
          cc0 = 0.0d0
          if( i .eq. ir ) then
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            mu   = nu*( nu + 1 )/2
            cc0  = cc0 + gintg1( mu )
          endif
          if( k .eq. it ) then
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            mu   = nu*( nu + 1 )/2
            cc0  = cc0 + gintg1( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,i,k,ir,it,ir,it,
     #nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue

coht      write(iout,10) ncimat,cimat(ncimat)
coht 10   format('11 off diag ',i5,f10.5d3 )

c
c
c diagonal part
      do 3 nomg = momega(2)+1, momega(3)
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nu   = kpp*( kpp - 1 )/2 + kppx
        mu   = nu*( nu + 1 )/2
        cc0  = 2.0d0*( ediff( nu ) - gintg1( mu ) )
        nuii = i*( i + 1 )/2
        nukk = k*( k + 1 )/2
        muii = nuii*( nuii + 1 )/2
        mukk = nukk*( nukk + 1 )/2
        lmd  = max0( nuii, nukk )
        lmdx = min0( nuii, nukk )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - 2.0d0*gintg1( mu )
     $        + gintg1( muii ) + gintg1( mukk )
        ncimat = nomg*( nomg + 1 )/2
        cimat( ncimat ) = cc0
c========( Thu Oct 19 18:11:25 JST 1995 )======
c$$$          cimat( ncimat ) = cimat( ncimat ) + h000
c========( Thu Oct 19 18:11:25 JST 1995 )======

c$ohtawara
      if(idebug.eq.1) write(iout,110) i,k,i,k,  i,k,i,k,
     #nomg,nomg,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    3 continue
c
c
      call clockm('cimt11.   ')
c
c
      return
      end
      subroutine cimt1s
c=====================================================================*
c                                          @                          *
c       @@@@      @    @    @   @@@@@   @@      @@@@                 *
c      @    @     @    @@  @@     @    @ @     @                     *
c      @          @    @ @@ @     @      @      @@@@                 *
c      @          @    @    @     @      @          @                *
c      @    @     @    @    @     @      @     @    @                *
c       @@@@      @    @    @     @    @@@@@    @@@@                 *
c=====================================================================*
c
c  get ci matrix elements between type I and single excitaion states
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
```

```
      common / io / in, iout, ifile(10)
c
c
      do 1 nomg = momega(2)+1, momega(3)
        ir = lci1( nomg )
        it = lci3( nomg ) + nocs
        do 2 ntau = momega(1)+1, momega(2)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
          cc0 = 0.0d0
          if( i .eq. ir ) then
            kpp  = max0( i, it )
            kppx = min0( i, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
          endif
          if( k .eq. it ) then
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( k, ir )
            kppx = min0( k, ir )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - gintgl( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = dsqrt( 2.0d0 )*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,ir,it,ir,it,nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,')',' -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimtls.   ')
c
c
      return
      end
      subroutine cimt20
c=====================================================================*
c                                    @@@@@    @@@                     *
c       @@@@      @    @    @   @@@@@ @      @  @   @                 *
c      @    @     @    @@  @@     @          @ @     @                *
c      @          @    @ @@ @     @    @@@@@  @     @                *
c      @          @    @    @     @   @       @     @                *
c      @    @     @    @    @     @   @        @   @                 *
c       @@@@      @    @    @     @   @@@@@@@   @@@                  *
c=====================================================================*
c
c  get ci matrix elements between type II and Hartree-Fock state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)
c
      root2 = dsqrt( 2.0d0 )
      ntau  = 1
      do 1 nomg = momega(3)+1, momega(4)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
c        j = lci1( nomg )
c        i = lci2( nomg )
        k = lci3( nomg ) + nocs
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( j, k )
        kppx = min0( j, k )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        ncimat = nomg*( nomg - 1 )/2 + ntau
        cimat( ncimat ) = -root2*gintgl( mu )

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,j,k,nomg,ntau,cimat(ncimat)
 100  format('( 0 -> 0) -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    1 continue
c
c
c     all done. return to caller.
      call clockm('cimt20.   ')
c
c
      return
      end
      subroutine cimt21
c=====================================================================*
c                                    @@@@@      @                     *
c       @@@@      @    @    @   @@@@@ @      @   @@                  *
c      @    @     @    @@  @@     @          @  @ @                  *
c      @          @    @ @@ @     @    @@@@@      @                  *
c      @          @    @    @     @   @           @                  *
c      @    @     @    @    @     @   @           @                  *
c       @@@@      @    @    @     @   @@@@@@@  @@@@@                 *
c=====================================================================*
c
c  get ci matrix elements between type II and type I
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)
c
c
      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momega(3)+1, momega(4)
c$ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )
c        it = lci3( nomg ) + nocs

        do 2 ntau = momega(2)+1, momega(3)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
c zero except for (delta_ir.eq.1).and.(delta_kt.eq.1)
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
c          go to 4
 3        continue
```

```
          cc0 = 0.0d0
          if( k .eq. it ) then
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( i, is )
            kppx = min0( i, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - gintgl( mu )
            if( i .eq. ir ) then
              nu   = it*( it + 1 )/2
              kpp  = max0( ir, is )
              kppx = min0( ir, is )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + 2.0d0*gintgl( mu )
              kpp  = max0( it, is )
              kppx = min0( it, is )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( ir, it )
              kppx = min0( ir, it )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - gintgl( mu )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = root2*cc0

c$ohtawara
c 4         continue
      if(idebug.ge.1) write(iout,110) i,k,i,k,ir,it,is,it,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt21.   ')
c
c
      return
      end
      subroutine cimt22
c=====================================================================*
c                                        00000   00000                *
c       0000      0    0    0   00000 0      0 0      0               *
c      0    0     0    00  00     0          0        0               *
c      0          0    0 00 0     0    00000   00000                 *
c      0          0    0    0     0   0        0                     *
c      0    0     0    0    0     0   0        0                     *
c       0000      0    0    0     0   0000000 0000000                 *
c=====================================================================*
c
c  get ci matrix elements between type II and type II
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)
c
c
c off diagonal part
      do 1 nomg = momega(3)+1, momega(4)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
c         j = lci1( nomg )
c         i = lci2( nomg )

        k = lci3( nomg ) + nocs
        do 2 ntau = momega(3)+1, nomg-1
c$ohtawara
          ir = lci1( ntau )
          is = lci2( ntau )
c          is = lci1( ntau )
c          ir = lci2( ntau )

          it = lci3( ntau ) + nocs
          cc0 = 0.0d0
          if( k .eq. it ) then
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( j, is )
            kppx = min0( j, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            kpp  = max0( j, ir )
            kppx = min0( j, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( i, is )
            kppx = min0( i, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            if( i .eq. ir ) then
              nu   = it*( it + 1 )/2
              kpp  = max0( is, j )
              kppx = min0( is, j )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - 2.0d0*gintgl( mu )
              kpp  = max0( it, is )
              kppx = min0( it, is )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( it, j )
              kppx = min0( it, j )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + gintgl( mu )
            endif
          endif
          if( i .eq. ir  .and.  j .eq. is ) then
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            mu   = nu*( nu + 1 )/2
            cc0  = cc0 + gintgl( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = cc0
c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,j,k,  ir,it,is,it,
     #nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
c diagonal part
      do 3 nomg = momega(3)+1, momega(4)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
c         j = lci1( nomg )
c         i = lci2( nomg )
```

```
        k = lci3( nomg ) + nocs
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nuik = kpp*( kpp - 1 )/2 + kppx
        muik = nuik*( nuik + 1 )/2
        kpp  = max0( i, j )
        kppx = min0( i, j )
        nuij = kpp*( kpp - 1 )/2 + kppx
        muij = nuij*( nuij + 1 )/2
        kpp  = max0( j, k )
        kppx = min0( j, k )
        nujk = kpp*( kpp - 1 )/2 + kppx
        mujk = nujk*( nujk + 1 )/2
        cc0  = ediff(  nuik ) + ediff(  nujk )
     $        - gintgl( muik ) - gintgl( mujk ) + gintgl( muij )
        nuii = i*( i + 1 )/2
        nujj = j*( j + 1 )/2
        nukk = k*( k + 1 )/2
        mukk = nukk*( nukk + 1 )/2
        lmd  = max0( nuii, nukk )
        lmdx = min0( nuii, nukk )
        muik = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( nukk, nujj )
        lmdx = min0( nukk, nujj )
        mujk = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( nujj, nuii )
        lmdx = min0( nujj, nuii )
        muij = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( muik ) - gintgl( mujk )
     $             + gintgl( mukk ) + gintgl( muij )
        ncimat = nomg*( nomg + 1 )/2
        cimat( ncimat ) = cc0
c========( Thu Oct 19 18:11:25 JST 1995 )======
c$$$            cimat( ncimat ) = cimat( ncimat ) + h000
c========( Thu Oct 19 18:11:25 JST 1995 )======

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,j,k,  i,k,j,k,
     #nomg,nomg,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    3 continue
c
c
      call clockm('cimt22.   ')
c
c
      return
      end
      subroutine cimt2s
c=====================================================================*
c                                        @@@@@                        *
c       @@@@      @    @    @   @@@@@ @      @   @@@@                 *
c      @    @     @    @@  @@     @          @  @                     *
c      @          @    @ @@ @     @    @@@@@    @@@@                 *
c      @          @    @    @     @   @             @                 *
c      @    @     @    @    @     @   @         @    @                *
c       @@@@      @    @    @     @   @@@@@@@   @@@@                 *
c=====================================================================*
c
c  get ci matrix elements between type II and single excitaion states
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

      do 1 nomg = momega(3)+1, momega(4)
c$ ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )

        it = lci3( nomg ) + nocs

        do 2 ntau = momega(1)+1, momega(2)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ ohtawara
          ir = lci1( nomg )
          is = lci2( nomg )
          i1 = ir
          i2 = is
          if(is.eq.i) then
             ir = i2
             is = i1
          endif

          cc0 = 0.0d0
          if( k .eq. it ) then
            kpp  = max0( is, it )
            kppx = min0( is, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            kpp  = max0( i, is )
            kppx = min0( i, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( ir, it )
            kppx = min0( ir, it )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
          endif
         if( i .eq. ir ) then
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( is, it )
            kppx = min0( is, it )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - gintgl( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.gt.0) write(iout,100) i,k,ir,it,is,it,
     &nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,')',' -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     &3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)


    2   continue
    1 continue
c
c
      call clockm('cimt2s.   ')
c
c
      return
      end
      subroutine cimt30( momgs,  momge  )
c=====================================================================*
c                                        @@@@@    @@@                 *
c       @@@@      @    @    @   @@@@@ @      @  @   @                 *
c      @    @     @    @@  @@     @          @ @     @                 *
c      @          @    @ @@ @     @    @@@@@  @     @                 *
c      @          @    @    @     @          @ @     @                 *
c      @    @     @    @    @     @   @      @  @   @                 *
c       @@@@      @    @    @     @    @@@@@    @@@                  *
c=====================================================================*
c
c  get ci matrix elements between type III and Hartree-Fock state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
```

```
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

      root2 = dsqrt( 2.0d0 )
      ntau  = 1
      do 1 nomg = momgs, momge
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( i, l )
        kppx = min0( i, l )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        ncimat = nomg*( nomg - 1 )/2 + ntau
        cimat( ncimat ) = root2*gintgl( mu )

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,i,l,nomg,ntau,cimat(ncimat)
 100  format('( 0 -> 0) -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    1 continue
c
c
c     all done. return to caller.
      call clockm('cimt30.   ')
c
c
      return
      end
      subroutine cimt31( momgs,  momge,  mtaus,  mtaue  )
c=====================================================================*
c                                        @@@@@     @                  *
c       @@@@      @    @    @   @@@@@ @      @   @@                  *
c      @    @     @    @@  @@     @          @  @ @                  *
c      @          @    @ @@ @     @    @@@@@     @                  *
c      @          @    @    @     @          @    @                  *
c      @    @     @    @    @     @   @      @    @                  *
c       @@@@      @    @    @     @    @@@@@  @@@@@                  *
c=====================================================================*
c
c  get ci matrix elements between type III and type I
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momgs, momge
c$ohtawara
c        ir = lci1( nomg )
c        it = lci3( nomg ) + nocs
c        iu = lci4( nomg ) + nocs

        do 2 ntau = mtaus, mtaue
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci3( nomg ) + nocs
          i3 = lci4( nomg ) + nocs

          ir = i1
          it = i2
          iu = i3
          if(ir.eq.i .and. iu.eq.k)  go to 3
          ir = i1
          it = i3
          iu = i2
          if(ir.eq.i .and. iu.eq.k)  go to 3
c zero except for (delta_ir.eq.1).and.(delta_ku.eq.1)
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
c          go to 4
 3        continue

          cc0 = 0.0d0
          if( i .eq. ir ) then
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( k, iu )
            kppx = min0( k, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            if( k .eq. iu ) then
              nu   = ir*( ir + 1 )/2
              kpp  = max0( iu, it )
              kppx = min0( iu, it )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - 2.0d0*gintgl( mu )
              kpp  = max0( iu, ir )
              kppx = min0( iu, ir )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( ir, it )
              kppx = min0( ir, it )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + gintgl( mu )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = root2*cc0

c$ohtawara
c 4          continue
      if(idebug.ge.1) write(6,110) i,k,i,k,  ir,it,ir,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt31.   ')
c
c
      return
      end
      subroutine cimt32( momgs,  momge,  mtaus,  mtaue  )
c=====================================================================*
c                                        @@@@@   @@@@@                *
c       @@@@      @    @    @   @@@@@ @      @ @      @               *
c      @    @     @    @@  @@     @          @        @               *
c      @          @    @ @@ @     @    @@@@@   @@@@@                *
c      @          @    @    @     @          @ @                     *
c      @    @     @    @    @     @   @      @ @                     *
c       @@@@      @    @    @     @    @@@@@  @@@@@@@                *
c=====================================================================*
c
c  get ci matrix elements between type III and type II
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
```

```
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)
c
      do 1 nomg = momgs, momge
c$ohtawara
c         ir = lci1( nomg )
c         it = lci3( nomg ) + nocs
c         iu = lci4( nomg ) + nocs

        do 2 ntau = mtaus, mtaue
          i = lci1( ntau )
          j = lci2( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci3( nomg ) + nocs
          i3 = lci4( nomg ) + nocs

          ir = i1
          it = i2
          iu = i3
          if( iu.eq.k .and. (ir.eq.i .or. ir.eq.j) ) go to 3
          ir = i1
          it = i3
          iu = i2
          if( iu.eq.k .and. (ir.eq.i .or. ir.eq.j) ) go to 3
c zero except for delta_uk, etc...
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
 3        continue

          cc0 = 0.0d0
          if( k .eq. iu ) then
            if( i .eq. ir ) then
              kpp  = max0( j, ir )
              kppx = min0( j, ir )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( it, iu )
              kppx = min0( it, iu )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + 2.0d0*gintgl( mu )
              kpp  = max0( j, iu )
              kppx = min0( j, iu )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( it, ir )
              kppx = min0( it, ir )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - gintgl( mu )
            endif
            if( j .eq. ir ) then
              kpp  = max0( i, ir )
              kppx = min0( i, ir )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( it, iu )
              kppx = min0( it, iu )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + 2.0d0*gintgl( mu )
              kpp  = max0( i, iu )
              kppx = min0( i, iu )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( it, ir )
              kppx = min0( it, ir )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - gintgl( mu )
            endif
          endif
```



```
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,j,k,  ir,it,ir,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt32.   ')
c
c


      return
      end

      subroutine cimt33( momgs,  momge,  mtaus  )
c=====================================================================*
c                                          @@@@@   @@@@@              *
c       @@@@      @    @    @   @@@@@ @     @ @     @                 *
c      @    @     @    @@  @@    @         @       @                 *
c      @          @    @ @@ @    @    @@@@@   @@@@@                 *
c      @          @    @    @    @         @       @                 *
c      @    @     @    @    @    @   @     @ @     @                 *
c       @@@@      @    @    @    @    @@@@@   @@@@@                  *
c=====================================================================*
c
c  get ci matrix elements between type III and type III
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c off diagonal part
      do 1 nomg = momgs, momge
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
        do 2 ntau = mtaus, nomg-1
          ir = lci1( ntau )
          it = lci3( ntau ) + nocs
          iu = lci4( ntau ) + nocs
          cc0 = 0.0d0
          if( i .eq. ir ) then
            kpp  = max0( l, it )
            kppx = min0( l, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( k, iu )
            kppx = min0( k, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( l, iu )
            kppx = min0( l, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            if( k .eq. it ) then
              nu   = ir*( ir + 1 )/2
              kpp  = max0( iu, l )
              kppx = min0( iu, l )
              nux  = kpp*( kpp - 1 )/2 + kppx
```

```
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - 2.0d0*gintgl( mu )
            kpp  = max0( iu, ir )
            kppx = min0( iu, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( ir, l )
            kppx = min0( ir, l )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
          endif
        endif
        if( k .eq. it  .and.  l .eq. iu ) then
          kpp  = max0( i, ir )
          kppx = min0( i, ir )
          nu   = kpp*( kpp - 1 )/2 + kppx
          mu   = nu*( nu + 1 )/2
          cc0  = cc0 + gintgl( mu )
        endif
        ncimat = nomg*( nomg - 1 )/2 + ntau
        cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,i,l,  ir,it,ir,iu,
     #nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
c diagonal part
      do 3 nomg = momgs, momge
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nuik = kpp*( kpp - 1 )/2 + kppx
        muik = nuik*( nuik + 1 )/2
        kpp  = max0( i, l )
        kppx = min0( i, l )
        nuil = kpp*( kpp - 1 )/2 + kppx
        muil = nuil*( nuil + 1 )/2
        kpp  = max0( k, l )
        kppx = min0( k, l )
        nukl = kpp*( kpp - 1 )/2 + kppx
        mukl = nukl*( nukl + 1 )/2
        cc0  = ediff(  nuik ) + ediff(  nuil )
     $       - gintgl( muik ) - gintgl( muil ) + gintgl( mukl )
        nuii = i*( i + 1 )/2
        nukk = k*( k + 1 )/2
        null = l*( l + 1 )/2
        muii = nuii*( nuii + 1 )/2
        lmd  = max0( nuii, nukk )
        lmdx = min0( nuii, nukk )
        muik = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( nuii, null )
        lmdx = min0( nuii, null )
        muil = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( nukk, null )
        lmdx = min0( nukk, null )
        mukl = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( muik ) - gintgl( muil )
     $             + gintgl( muii ) + gintgl( mukl )
        ncimat = nomg*( nomg + 1 )/2
        cimat( ncimat ) = cc0
c========( Thu Oct 19 18:11:25 JST 1995 )======
c$$$            cimat( ncimat ) = cimat( ncimat ) + h000
c========( Thu Oct 19 18:11:25 JST 1995 )======

c$ohtawara
      if(idebug.ge.1) write(out,110) i,k,i,l,  i,k,i,l,
     #nomg,nomg,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)
```

```
    3 continue
c
c
      call clockm('cimt33.   ')
c
c
      return
      end
      subroutine cimt3s( momgs,  momge,  mtaus,  mtaue  )
c=================================================================*
c                                        @@@@@                    *
c       @@@@      @    @    @   @@@@@ @     @   @@@@              *
c      @    @     @    @@  @@     @         @  @                  *
c      @          @    @ @@ @     @    @@@@@    @@@@              *
c      @          @    @    @     @         @       @              *
c      @    @     @    @    @     @    @    @  @    @              *
c       @@@@      @    @    @     @    @@@@@    @@@@               *
c=================================================================*
c
c  get ci matrix elements between type III and single excitaion states
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c
      do 1 nomg = momgs, momge
        ir = lci1( nomg )
c$ohtawara
c       it = lci3( nomg ) + nocs
c       iu = lci4( nomg ) + nocs

        do 2 ntau = mtaus, mtaue
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
          cc0 = 0.0d0
c$ohtawara
          it = lci3( nomg ) + nocs
          iu = lci4( nomg ) + nocs
          i1 = it
          i2 = iu
          if(iu.eq.k) then
             it = i2
             iu = i1
          endif

          if( i .eq. ir ) then
            kpp  = max0( i, it )
            kppx = min0( i, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( k, iu )
            kppx = min0( k, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            kpp  = max0( i, iu )
            kppx = min0( i, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
          endif
          if( k .eq. it ) then
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( iu, ir )
```

```
            kppx = min0( iu, ir )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - gintgl( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,ir,it,ir,iu,
     &nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,')',' -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     &3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt3s.   ')
c
c
      return
      end
      subroutine cimt40
c=====================================================================*
c                                        @         @@@                 *
c       @@@@      @    @    @   @@@@@ @    @   @   @                  *
c      @    @     @    @@  @@     @   @    @  @     @                 *
c      @          @    @ @@ @     @   @    @  @     @                 *
c      @          @    @    @     @   @@@@@@@ @     @                 *
c      @    @     @    @    @     @        @   @   @                  *
c       @@@@      @    @    @     @        @    @@@                   *
c=====================================================================*
c
c  get ci matrix elements between type IV and Hartree-Fock state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
      ntau  = 1
      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
c        j = lci1( nomg )
c        i = lci2( nomg )
c        l = lci3( nomg ) + nocs
c        k = lci4( nomg ) + nocs

        kpp  = max0( i, k )
        kppx = min0( i, k )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( j, l )
        kppx = min0( j, l )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = - gintgl( mu )
        kpp  = max0( i, l )
        kppx = min0( i, l )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( j, k )
        kppx = min0( j, k )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        ncimat = nomg*( nomg - 1 )/2 + ntau
        cimat( ncimat ) = cc0 - gintgl( mu )
```



```
c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,j,l,nomg,ntau,cimat(ncimat)
 100  format('( 0 -> 0) -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    1 continue
c
c
c     all done. return to caller.
      call clockm('cimt40.   ')
c
c
      return
      end


      subroutine cimt41
c=====================================================================*
c                                        @         @                   *
c       @@@@      @    @    @   @@@@@ @    @    @@                   *
c      @    @     @    @@  @@     @   @    @   @ @                   *
c      @          @    @ @@ @     @   @    @    @                    *
c      @          @    @    @     @   @@@@@@@    @                    *
c      @    @     @    @    @     @        @    @                    *
c       @@@@      @    @    @     @        @   @@@@@                 *
c=====================================================================*
c
c  get ci matrix elements between type IV and type I
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c

      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )
c        it = lci3( nomg ) + nocs
c        iu = lci4( nomg ) + nocs

        do 2 ntau = momega(2)+1, momega(3)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
        i1 = lci1( nomg )
        i2 = lci2( nomg )
        i3 = lci3( nomg ) + nocs
        i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if(ir.eq.i .and. iu.eq.k)  go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if(ir.eq.i .and. iu.eq.k)  go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if(ir.eq.i .and. iu.eq.k)  go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if(ir.eq.i .and. iu.eq.k)  go to 3
c          write(iout,*) '=== Error in cimt41.f ===='
c          stop
```

```
c zero except for delta_is, etc...
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
 3        continue

          cc0 = 0.0d0
          if( (k .eq. iu).and.(i .eq. ir) ) then

            kpp  = max0( it, iu )
            kppx = min0( it, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( ir, is )
            kppx = min0( ir, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + 2.0d0*gintgl( mu )
            kpp  = max0( it, is )
            kppx = min0( it, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( ir, iu )
            kppx = min0( ir, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - gintgl( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,i,k,  ir,it,is,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt41.   ')
c
c
      return
      end
      subroutine cimt42
c=====================================================================*
c                                        @         @@@@@               *
c       @@@@      @    @    @   @@@@@ @    @  @      @                *
c      @    @     @    @@  @@     @   @    @         @                *
c      @          @    @ @@ @     @   @    @   @@@@@                 *
c      @          @    @    @     @   @@@@@@@ @                       *
c      @    @     @    @    @     @        @  @                       *
c       @@@@      @    @    @     @        @  @@@@@@@                 *
c=====================================================================*
c
c  get ci matrix elements between type IV and type II
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c
      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
c         ir = lci1( nomg )
c         is = lci2( nomg )
c         it = lci3( nomg ) + nocs
c         iu = lci4( nomg ) + nocs
```

```
          do 2 ntau = momega(3)+1, momega(4)
            i = lci1( ntau )
            j = lci2( ntau )
            k = lci3( ntau ) + nocs
c$ohtawara
            i1 = lci1( nomg )
            i2 = lci2( nomg )
            i3 = lci3( nomg ) + nocs
            i4 = lci4( nomg ) + nocs

            ir = i1
            is = i2
            it = i3
            iu = i4
            if( is.eq.i .and. ir.eq.j .and. iu.eq.k ) go to 3
            ir = i1
            is = i2
            it = i4
            iu = i3
            if( is.eq.i .and. ir.eq.j .and. iu.eq.k ) go to 3
            ir = i2
            is = i1
            it = i3
            iu = i4
            if( is.eq.i .and. ir.eq.j .and. iu.eq.k ) go to 3
            ir = i2
            is = i1
            it = i4
            iu = i3
            if( is.eq.i .and. ir.eq.j .and. iu.eq.k ) go to 3
c zero except for delta_is, etc...
            ncimat = nomg*( nomg - 1 )/2 + ntau
            cimat( ncimat ) = 0
 3          continue

            cc0 = 0.0d0
            if( i .eq. is ) then
              if( j .eq. ir ) then
                kpp  = max0( k, it )
                kppx = min0( k, it )
                nu   = kpp*( kpp - 1 )/2 + kppx
                kpp  = max0( k, iu )
                kppx = min0( k, iu )
                nux  = kpp*( kpp - 1 )/2 + kppx
                lmd  = max0( nu, nux )
                lmdx = min0( nu, nux )
                mu   = lmd*( lmd - 1 )/2 + lmdx
                cc0  = cc0 + gintgl( mu )
                if( k .eq. iu ) then
                  nu   = is*( is + 1 )/2
                  kpp  = max0( it, iu )
                  kppx = min0( it, iu )
                  nux  = kpp*( kpp - 1 )/2 + kppx
                  lmd  = max0( nu, nux )
                  lmdx = min0( nu, nux )
                  mu   = lmd*( lmd - 1 )/2 + lmdx
                  cc0  = cc0 - gintgl( mu )
                  kpp  = max0( it, is )
                  kppx = min0( it, is )
                  nu   = kpp*( kpp - 1 )/2 + kppx
                  kpp  = max0( is, iu )
                  kppx = min0( is, iu )
                  nux  = kpp*( kpp - 1 )/2 + kppx
                  lmd  = max0( nu, nux )
                  lmdx = min0( nu, nux )
                  mu   = lmd*( lmd - 1 )/2 + lmdx
                  cc0  = cc0 + 0.5d0*gintgl( mu )
                endif
              endif
              if( k .eq. iu ) then
                kpp  = max0( it, ir )
                kppx = min0( it, ir )
                nu   = kpp*( kpp - 1 )/2 + kppx
                kpp  = max0( j, iu )
                kppx = min0( j, iu )
                nux  = kpp*( kpp - 1 )/2 + kppx
                lmd  = max0( nu, nux )
                lmdx = min0( nu, nux )
                mu   = lmd*( lmd - 1 )/2 + lmdx
                cc0  = cc0 + 0.5d0*gintgl( mu )
                kpp  = max0( ir, j )
                kppx = min0( ir, j )
                nu   = kpp*( kpp - 1 )/2 + kppx
```

```
              kpp  = max0( it, iu )
              kppx = min0( it, iu )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - gintgl( mu )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = root2*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,j,k,  ir,it,is,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt42.   ')
c
c
      return
      end
      subroutine cimt43
c=====================================================================*
c                                          @        @@@@@              *
c       @@@@       @     @     @   @@@@@ @     @  @      @             *
c      @    @      @     @@  @@      @   @     @         @             *
c      @           @     @ @@ @      @   @     @   @@@@@               *
c      @           @     @    @      @   @@@@@@@        @             *
c      @    @      @     @    @      @          @  @      @             *
c       @@@@       @     @    @      @          @   @@@@@              *
c=====================================================================*
c
c  get ci matrix elements between type IV and type III
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
c         ir = lci1( nomg )
c         is = lci2( nomg )
c         it = lci3( nomg ) + nocs
c         iu = lci4( nomg ) + nocs

         do 2 ntau = momega(4)+1, momega(5)
           i = lci1( ntau )
           k = lci3( ntau ) + nocs
           l = lci4( ntau ) + nocs
c$ohtawara
           i1 = lci1( nomg )
           i2 = lci2( nomg )
           i3 = lci3( nomg ) + nocs
           i4 = lci4( nomg ) + nocs

           ir = i1
           is = i2
           it = i3
           iu = i4
           if( it.eq.k .and. ir.eq.i .and. iu.eq.l ) go to 3
           ir = i1
           is = i2
           it = i4
           iu = i3
           if( it.eq.k .and. ir.eq.i .and. iu.eq.l ) go to 3
           ir = i2
           is = i1
           it = i3
           iu = i4
           if( it.eq.k .and. ir.eq.i .and. iu.eq.l ) go to 3
           ir = i2
           is = i1
           it = i4
           iu = i3
           if( it.eq.k .and. ir.eq.i .and. iu.eq.l ) go to 3
c zero except for delta_is, etc...
           ncimat = nomg*( nomg - 1 )/2 + ntau
           cimat( ncimat ) = 0
 3         continue

           cc0 = 0.0d0
           if( k .eq. it ) then
             if( i .eq. ir ) then
               kpp  = max0( ir, is )
               kppx = min0( ir, is )
               nu   = kpp*( kpp - 1 )/2 + kppx
               kpp  = max0( l, iu )
               kppx = min0( l, iu )
               nux  = kpp*( kpp - 1 )/2 + kppx
               lmd  = max0( nu, nux )
               lmdx = min0( nu, nux )
               mu   = lmd*( lmd - 1 )/2 + lmdx
               cc0  = cc0 + gintgl( mu )
               kpp  = max0( iu, is )
               kppx = min0( iu, is )
               nu   = kpp*( kpp - 1 )/2 + kppx
               kpp  = max0( l, ir )
               kppx = min0( l, ir )
               nux  = kpp*( kpp - 1 )/2 + kppx
               lmd  = max0( nu, nux )
               lmdx = min0( nu, nux )
               mu   = lmd*( lmd - 1 )/2 + lmdx
               cc0  = cc0 - 0.5d0*gintgl( mu )
               if( l .eq. iu ) then
                 nu   = it*( it + 1 )/2
                 kpp  = max0( ir, is )
                 kppx = min0( ir, is )
                 nux  = kpp*( kpp - 1 )/2 + kppx
                 lmd  = max0( nu, nux )
                 lmdx = min0( nu, nux )
                 mu   = lmd*( lmd - 1 )/2 + lmdx
                 cc0  = cc0 + gintgl( mu )
                 kpp  = max0( it, is )
                 kppx = min0( it, is )
                 nu   = kpp*( kpp - 1 )/2 + kppx
                 kpp  = max0( it, ir )
                 kppx = min0( it, ir )
                 nux  = kpp*( kpp - 1 )/2 + kppx
                 lmd  = max0( nu, nux )
                 lmdx = min0( nu, nux )
                 mu   = lmd*( lmd - 1 )/2 + lmdx
                 cc0  = cc0 - 0.5d0*gintgl( mu )
               endif
             endif
             if( l .eq. iu ) then
               kpp  = max0( i, ir )
               kppx = min0( i, ir )
               nu   = kpp*( kpp - 1 )/2 + kppx
               kpp  = max0( i, is )
               kppx = min0( i, is )
               nux  = kpp*( kpp - 1 )/2 + kppx
               lmd  = max0( nu, nux )
               lmdx = min0( nu, nux )
               mu   = lmd*( lmd - 1 )/2 + lmdx
               cc0  = cc0 - gintgl( mu )
             endif
           endif
           ncimat = nomg*( nomg - 1 )/2 + ntau
           cimat( ncimat ) = root2*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,i,l,  ir,it,is,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
```

```
c
c
      call clockm('cimt43.   ')
c
c
      return
      end
      subroutine cimt44
c=====================================================================*
c                                          @        @                 *
c       @@@@       @    @    @   @@@@@ @    @  @    @                 *
c      @    @      @    @@  @@     @   @    @  @    @                 *
c      @           @    @ @@ @     @   @    @  @    @                 *
c      @           @    @    @     @   @@@@@@@ @@@@@@@                *
c      @    @      @    @    @     @         @        @               *
c       @@@@       @    @    @     @         @        @               *
c=====================================================================*
c
c  get ci matrix elements between type IV and type IV
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c off diagonal part
      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
c        j = lci1( nomg )
c        i = lci2( nomg )
c        l = lci3( nomg ) + nocs
c        k = lci4( nomg ) + nocs
        do 2 ntau = momega(5)+1, nomg-1
c$ohtawara
          ir = lci1( ntau )
          is = lci2( ntau )
          it = lci3( ntau ) + nocs
          iu = lci4( ntau ) + nocs
c          is = lci1( ntau )
c          ir = lci2( ntau )
c          iu = lci3( ntau ) + nocs
c          it = lci4( ntau ) + nocs
          cc0 = 0.0d0
          if( i .eq. ir  .and.  is .eq. j ) then
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( l, iu )
            kppx = min0( l, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            kpp  = max0( k, iu )
            kppx = min0( k, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( l, it )
            kppx = min0( l, it )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            if( k .eq. it ) then
              nu   = ir*( ir + 1 )/2
              kpp  = max0( iu, l )
              kppx = min0( iu, l )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - gintgl( mu )
              kpp  = max0( ir, iu )
              kppx = min0( ir, iu )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( ir, l )
              kppx = min0( ir, l )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + 0.5d0*gintgl( mu )
            endif
          endif
          if( k .eq. it  .and.  iu .eq. l ) then
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( j, is )
            kppx = min0( j, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            kpp  = max0( j, ir )
            kppx = min0( j, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( i, is )
            kppx = min0( i, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            if( i .eq. ir ) then
              nu   = it*( it + 1 )/2
              kpp  = max0( is, j )
              kppx = min0( is, j )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - gintgl( mu )
              kpp  = max0( it, is )
              kppx = min0( it, is )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( it, j )
              kppx = min0( it, j )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + 0.5d0*gintgl( mu )
            endif
          endif
          if( i .eq. ir  .and.  k .eq. it ) then
            kpp  = max0( iu, is )
            kppx = min0( iu, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( l, j )
            kppx = min0( l, j )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + 0.5d0*gintgl( mu )
            kpp  = max0( j, is )
            kppx = min0( j, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( l, iu )
            kppx = min0( l, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - gintgl( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,j,l,  ir,it,is,iu,
```

```
     #nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
c off diagonal part
      do 3 nomg = momega(5)+1, momega(6)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
c        j = lci1( nomg )
c        i = lci2( nomg )
c        l = lci3( nomg ) + nocs
c        k = lci4( nomg ) + nocs
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nuik = kpp*( kpp - 1 )/2 + kppx
        muik = nuik*( nuik + 1 )/2
        cc0  = ediff( nuik ) - 1.5d0*gintgl( muik )
        kpp  = max0( l, j )
        kppx = min0( l, j )
        nujl = kpp*( kpp - 1 )/2 + kppx
        mujl = nujl*( nujl + 1 )/2
        cc0  = cc0 + ediff( nujl ) - 1.5d0*gintgl( mujl )
        kpp  = max0( j, k )
        kppx = min0( j, k )
        nujk = kpp*( kpp - 1 )/2 + kppx
        mujk = nujk*( nujk + 1 )/2
        cc0  = cc0 + 0.5d0*gintgl( mujk )
        kpp  = max0( l, i )
        kppx = min0( l, i )
        nuli = kpp*( kpp - 1 )/2 + kppx
        muli = nuli*( nuli + 1 )/2
        cc0  = cc0 + 0.5d0*gintgl( muli )
        kpp  = max0( l, k )
        kppx = min0( l, k )
        nukl = kpp*( kpp - 1 )/2 + kppx
        mukl = nukl*( nukl + 1 )/2
        cc0  = cc0 + gintgl( mukl )
        kpp  = max0( i, j )
        kppx = min0( i, j )
        nuij = kpp*( kpp - 1 )/2 + kppx
        muij = nuij*( nuij + 1 )/2
        cc0  = cc0 + gintgl( muij )
        nuii = i*( i + 1 )/2
        nujj = j*( j + 1 )/2
        nukk = k*( k + 1 )/2
        null = l*( l + 1 )/2
        lmd  = max0( nujj, nukk )
        lmdx = min0( nujj, nukk )
        mukj = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( null, nuii )
        lmdx = min0( null, nuii )
        muli = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( null, nukk )
        lmdx = min0( null, nukk )
        mukl = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( nujj, nuii )
        lmdx = min0( nujj, nuii )
        muij = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( mukj ) - gintgl( muli )
     $             + gintgl( mukl ) + gintgl( muij )
        ncimat = nomg*( nomg + 1 )/2
        cimat( ncimat ) = cc0
c========( Thu Oct 19 18:11:25 JST 1995 )======
c$$$            cimat( ncimat ) = cimat( ncimat ) + h000
c========( Thu Oct 19 18:11:25 JST 1995 )======

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,j,l,  i,k,j,l,
     #nomg,nomg,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    3 continue
c
```

```
c
      call clockm('cimt44.   ')
c
c
      return
      end
      subroutine cimt4s
c=====================================================================*
c                                          @                          *
c       @@@@      @    @    @   @@@@@ @    @    @@@@                 *
c      @    @     @    @@  @@     @   @    @   @                     *
c      @          @    @ @@ @     @   @    @    @@@@                 *
c      @          @    @    @     @   @@@@@@@       @                *
c      @     @    @    @    @     @        @    @    @                *
c       @@@@      @    @    @     @        @    @@@@                 *
c=====================================================================*
c
c  get ci matrix elements between type IV and single excitaion states
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
      coef0 = - 1.0d0/dsqrt( 2.0d0 )
      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )
c        it = lci3( nomg ) + nocs
c        iu = lci4( nomg ) + nocs

        do 2 ntau = momega(1)+1, momega(2)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
c          write(iout,*) '=== Error in cimt4s.f ===='
c          stop
c zero except for delta_is, etc...
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
 3        continue

          cc0 = 0.0d0
          if( k .eq. it ) then
            kpp  = max0( is, iu )
            kppx = min0( is, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
```

```
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mu   = lmd*( lmd - 1 )/2 + lmdx
          cc0  = cc0 - gintgl( mu )
          kpp  = max0( i, is )
          kppx = min0( i, is )
          nu   = kpp*( kpp - 1 )/2 + kppx
          kpp  = max0( ir, iu )
          kppx = min0( ir, iu )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mu   = lmd*( lmd - 1 )/2 + lmdx
          cc0  = cc0 - gintgl( mu )
        endif
        if( i .eq. ir ) then
          kpp  = max0( k, iu )
          kppx = min0( k, iu )
          nu   = kpp*( kpp - 1 )/2 + kppx
          kpp  = max0( is, it )
          kppx = min0( is, it )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mu   = lmd*( lmd - 1 )/2 + lmdx
          cc0  = cc0 + gintgl( mu )
          kpp  = max0( k, it )
          kppx = min0( k, it )
          nu   = kpp*( kpp - 1 )/2 + kppx
          kpp  = max0( is, iu )
          kppx = min0( is, iu )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mu   = lmd*( lmd - 1 )/2 + lmdx
          cc0  = cc0 + gintgl( mu )
        endif
        ncimat = nomg*( nomg - 1 )/2 + ntau
        cimat( ncimat ) = coef0*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,ir,it,is,iu,
     &nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,')',' -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     &3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt4s.   ')
c
c
      return
      end
      subroutine cimt50
c=====================================================================*
c                                        @@@@@@@   @@@                *
c       @@@@       @    @    @   @@@@@ @         @   @                *
c      @    @      @    @@  @@     @   @        @     @               *
c      @           @    @ @@ @     @   @@@@@@  @     @               *
c      @           @    @    @     @         @ @     @               *
c      @    @      @    @    @     @     @     @  @   @               *
c       @@@@       @    @    @     @    @@@@@    @@@                  *
c=====================================================================*
c
c  get ci matrix elements between type V and Hartree-Fock state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
      root3 = dsqrt( 3.0d0 )
      ntau  = 1
      do 1 nomg = momega(6)+1, momega(7)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
c        j = lci1( nomg )
c        i = lci2( nomg )
c        l = lci3( nomg ) + nocs
c        k = lci4( nomg ) + nocs

        kpp  = max0( i, l )
        kppx = min0( i, l )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( j, k )
        kppx = min0( j, k )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = gintgl( mu )
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( j, l )
        kppx = min0( j, l )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( mu )
        ncimat = nomg*( nomg - 1 )/2 + ntau
        cimat( ncimat ) = root3*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,j,l,nomg,ntau,cimat(ncimat)
 100  format('( 0 -> 0) -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    1 continue
c
c
c     all done. return to caller.
      call clockm('cimt50.   ')
c
c
      return
      end
      subroutine cimt51
c=====================================================================*
c                                        @@@@@@@   @                  *
c       @@@@       @    @    @   @@@@@ @         @@                  *
c      @    @      @    @@  @@     @   @        @ @                  *
c      @           @    @ @@ @     @   @@@@@@     @                  *
c      @           @    @    @     @         @    @                  *
c      @    @      @    @    @     @     @     @    @                *
c       @@@@       @    @    @     @    @@@@@   @@@@@                *
c=====================================================================*
c
c  get ci matrix elements between type V and type I
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c
      root3 = dsqrt( 3.0d0 )
      do 1 nomg = momega(6)+1, momega(7)
c$ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )
c        it = lci3( nomg ) + nocs
c        iu = lci4( nomg ) + nocs
```

```
        do 2 ntau = momega(2)+1, momega(3)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
          cc0 = 0.0d0
c$ohtawara
        i1 = lci1( nomg )
        i2 = lci2( nomg )
        i3 = lci3( nomg ) + nocs
        i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if(ir.eq.i .and. iu.eq.k)  go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if(ir.eq.i .and. iu.eq.k)  go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if(ir.eq.i .and. iu.eq.k)  go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if(ir.eq.i .and. iu.eq.k)  go to 3
c          write(iout,*) '=== Error in cimt51.f ===='
c          stop
c zero except for delta_is, etc...
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
 3        continue

          if( k .eq. iu  .and.  i .eq. ir ) then
            kpp  = max0( it, is )
            kppx = min0( it, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( ir, iu )
            kppx = min0( ir, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = root3*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,i,k,  ir,it,is,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt51.   ')
c
c
      return
      end
      subroutine cimt52
c=====================================================================*
c                                        @@@@@@@  @@@@@               *
c       @@@@      @    @    @   @@@@@ @        @     @               *
c      @    @     @    @@  @@     @   @                @               *
c      @          @    @ @@ @     @   @@@@@@@   @@@@@               *
c      @          @    @    @     @         @ @                      *
c      @    @     @    @    @     @   @     @ @                      *
c       @@@@      @    @    @     @    @@@@@  @@@@@@@                *
c=====================================================================*
c
c  get ci matrix elements between type V and type II
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
```

```
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c
      root2 = dsqrt( 2.0d0 )
      root3 = dsqrt( 3.0d0 )
      coef0 = - 0.5d0*root3
      coef1 = root2 - 1.0d0
      do 1 nomg = momega(6)+1, momega(7)
c$ohtawara
c         ir = lci1( nomg )
c         is = lci2( nomg )
c         it = lci3( nomg ) + nocs
c         iu = lci4( nomg ) + nocs

        do 2 ntau = momega(3)+1, momega(4)
          i = lci1( ntau )
          j = lci2( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if( is.eq.j .and. ir.eq.i .and. iu.eq.k ) go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if( is.eq.j .and. ir.eq.i .and. iu.eq.k ) go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if( is.eq.j .and. ir.eq.i .and. iu.eq.k ) go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if( is.eq.j .and. ir.eq.i .and. iu.eq.k ) go to 3
c zero except for delta_is, etc...
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
 3        continue

          cc0 = 0.0d0
          if( i .eq. ir  .and.  k .eq. iu ) then
            kpp  = max0( iu, j )
            kppx = min0( iu, j )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( it, is )
            kppx = min0( it, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            if( j .eq. is ) then
              kpp  = max0( is, it )
              kppx = min0( is, it )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( is, iu )
              kppx = min0( is, iu )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + coef1*gintgl( mu )
              kpp  = max0( ir, it )
              kppx = min0( ir, it )
```

```
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( ir, iu )
              kppx = min0( ir, iu )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - root2*gintgl( mu )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = coef0*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,j,k,  ir,it,is,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt52.   ')
c
c
      return
      end
      subroutine cimt53
c=====================================================================*
c                                        @@@@@@@  @@@@@               *
c       @@@@      @    @    @   @@@@@ @        @     @               *
c      @    @     @    @@  @@     @   @             @               *
c      @          @    @ @@ @     @   @@@@@@   @@@@@                *
c      @          @    @    @     @         @       @               *
c      @    @     @    @    @     @   @     @ @     @               *
c       @@@@      @    @    @     @    @@@@@   @@@@@                *
c=====================================================================*
c
c  get ci matrix elements between type V and type III
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c
      root2 = dsqrt( 2.0d0 )
      root3 = dsqrt( 3.0d0 )
      coef0 = - 0.5d0*root3
      coef1 = root2 - 1.0d0
      do 1 nomg = momega(6)+1, momega(7)
c$ohtawara
c         ir = lci1( nomg )
c         is = lci2( nomg )
c         it = lci3( nomg ) + nocs
c         iu = lci4( nomg ) + nocs

        do 2 ntau = momega(4)+1, momega(5)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
          l = lci4( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if( it.eq.k .and. ir.eq.i .and. iu.eq.1 ) go to 3
          ir = i1
```



```
          is = i2
          it = i4
          iu = i3
          if( it.eq.k .and. ir.eq.i .and. iu.eq.1 ) go to 3
          ir = i1
          is = i2
          it = i3
          iu = i4
          if( it.eq.k .and. ir.eq.i .and. iu.eq.1 ) go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if( it.eq.k .and. ir.eq.i .and. iu.eq.1 ) go to 3
c zero except for delta_is, etc...
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
 3        continue

          cc0 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
            kpp  = max0( ir, 1 )
            kppx = min0( ir, 1 )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( iu, is )
            kppx = min0( iu, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            if( 1 .eq. iu ) then
              kpp  = max0( ir, iu )
              kppx = min0( ir, iu )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( is, iu )
              kppx = min0( is, iu )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 + coef1*gintgl( mu )
              kpp  = max0( ir, it )
              kppx = min0( ir, it )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( it, is )
              kppx = min0( it, is )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - root2*gintgl( mu )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = coef0*cc0
c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,i,1,  ir,it,is,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt53.   ')
c
c
      return
      end
      subroutine cimt54
c=====================================================================*
c                                        @@@@@@@ @                    *
c       @@@@      @    @    @   @@@@@ @        @    @                *
c      @    @     @    @@  @@     @   @        @    @                *
c      @          @    @ @@ @     @   @@@@@@  @    @                *
c      @          @    @    @     @         @ @@@@@@@                *
c      @    @     @    @    @     @   @     @       @                *
c       @@@@      @    @    @     @    @@@@@       @                *
c=====================================================================*
c
```

```
c  get ci matrix elements between type V and type IV
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c
      coef0 = dsqrt( 1.5d0 )
      do 1 nomg = momega(6)+1, momega(7)
c$ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )
c        it = lci3( nomg ) + nocs
c        iu = lci4( nomg ) + nocs

        do 2 ntau = momega(5)+1, momega(6)
c$ohtawara
          i = lci1( ntau )
          j = lci2( ntau )
          k = lci3( ntau ) + nocs
          l = lci4( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if( ir.eq.i .and. it.eq.k .and. is.eq.j .and. iu.eq.l ) go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if( ir.eq.i .and. it.eq.k .and. is.eq.j .and. iu.eq.l ) go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if( ir.eq.i .and. it.eq.k .and. is.eq.j .and. iu.eq.l ) go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if( ir.eq.i .and. it.eq.k .and. is.eq.j .and. iu.eq.l ) go to 3
c zero except for delta_is, etc...
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
 3        continue

          cc0 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
            kpp  = max0( j, l )
            kppx = min0( j, l )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( iu, is )
            kppx = min0( iu, is )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
            if( j .eq. is ) then
              kpp  = max0( ir, iu )
              kppx = min0( ir, iu )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( ir, l )
              kppx = min0( ir, l )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - gintgl( mu )
            endif
            if( l .eq. iu ) then
              kpp  = max0( is, it )
              kppx = min0( is, it )
              nu   = kpp*( kpp - 1 )/2 + kppx
              kpp  = max0( it, j )
              kppx = min0( it, j )
              nux  = kpp*( kpp - 1 )/2 + kppx
              lmd  = max0( nu, nux )
              lmdx = min0( nu, nux )
              mu   = lmd*( lmd - 1 )/2 + lmdx
              cc0  = cc0 - gintgl( mu )
            endif
            if( j .eq. is  .and.  l .eq. iu ) then
              kpp  = max0( ir, it )
              kppx = min0( ir, it )
              nu   = kpp*( kpp - 1 )/2 + kppx
              mu   = nu*( nu + 1 )/2
              cc0  = cc0 + gintgl( mu )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = coef0*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,j,l,  ir,it,is,iu,
     #nomg,ntau,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt54.   ')
c
c
      return
      end
      subroutine cimt55
c=====================================================================*
c                                    @@@@@@@ @@@@@@@                  *
c       @@@@      @    @    @   @@@@@ @       @                       *
c      @    @     @    @@  @@     @   @       @                       *
c      @          @    @ @@ @     @   @@@@@@  @@@@@@                  *
c      @          @    @    @     @        @      @                   *
c      @    @     @    @    @     @   @    @ @    @                   *
c       @@@@      @    @    @     @    @@@@@   @@@@@                  *
c=====================================================================*
c
c  get ci matrix elements between type V and type V
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c off diagonal part
      do 1 nomg = momega(6)+1, momega(7)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
c        j = lci1( nomg )
c        i = lci2( nomg )
c        l = lci3( nomg ) + nocs
c        k = lci4( nomg ) + nocs

        do 2 ntau = momega(6)+1, nomg-1
c$ohtawara
          ir = lci1( ntau )
          is = lci2( ntau )
```

```
      it = lci3( ntau ) + nocs
      iu = lci4( ntau ) + nocs
c      is = lci1( ntau )
c      ir = lci2( ntau )
c      iu = lci3( ntau ) + nocs
c      it = lci4( ntau ) + nocs

      cc0 = 0.0d0
      if( i .eq. ir  .and.  is .eq. j ) then
        kpp  = max0( k, it )
        kppx = min0( k, it )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( l, iu )
        kppx = min0( l, iu )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 + gintgl( mu )
        kpp  = max0( k, iu )
        kppx = min0( k, iu )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( l, it )
        kppx = min0( l, it )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( mu )
        if( k .eq. it ) then
          nu   = ir*( ir + 1 )/2
          kpp  = max0( iu, l )
          kppx = min0( iu, l )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mu   = lmd*( lmd - 1 )/2 + lmdx
          cc0  = cc0 - gintgl( mu )
          kpp  = max0( ir, iu )
          kppx = min0( ir, iu )
          nu   = kpp*( kpp - 1 )/2 + kppx
          kpp  = max0( ir, l )
          kppx = min0( ir, l )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mu   = lmd*( lmd - 1 )/2 + lmdx
          cc0  = cc0 + 1.5d0*gintgl( mu )
        endif
      endif
      if( k .eq. it  .and.  iu .eq. l ) then
        kpp  = max0( i, ir )
        kppx = min0( i, ir )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( j, is )
        kppx = min0( j, is )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( mu )
        kpp  = max0( j, ir )
        kppx = min0( j, ir )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( i, is )
        kppx = min0( i, is )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( mu )
        if( i .eq. ir ) then
          nu   = it*( it + 1 )/2
          kpp  = max0( is, j )
          kppx = min0( is, j )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mu   = lmd*( lmd - 1 )/2 + lmdx
          cc0  = cc0 - gintgl( mu )
          kpp  = max0( it, is )
          kppx = min0( it, is )
          nu   = kpp*( kpp - 1 )/2 + kppx
          kpp  = max0( it, j )
          kppx = min0( it, j )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mu   = lmd*( lmd - 1 )/2 + lmdx
          cc0  = cc0 + 1.5d0*gintgl( mu )
        endif
      endif
      if( i .eq. ir  .and.  k .eq. it ) then
        kpp  = max0( iu, is )
        kppx = min0( iu, is )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( l, j )
        kppx = min0( l, j )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 + 1.5d0*gintgl( mu )
        kpp  = max0( j, is )
        kppx = min0( j, is )
        nu   = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( l, iu )
        kppx = min0( l, iu )
        nux  = kpp*( kpp - 1 )/2 + kppx
        lmd  = max0( nu, nux )
        lmdx = min0( nu, nux )
        mu   = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( mu )
      endif
      ncimat = nomg*( nomg - 1 )/2 + ntau
      cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,j,l,  ir,it,is,iu,
     #nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
c diagonal part
      do 3 nomg = momega(6)+1, momega(7)
c$ohtawara
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
c        j = lci1( nomg )
c        i = lci2( nomg )
c        l = lci3( nomg ) + nocs
c        k = lci4( nomg ) + nocs

        kpp  = max0( i, k )
        kppx = min0( i, k )
        nuik = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( l, j )
        kppx = min0( l, j )
        nujl = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( k, j )
        kppx = min0( k, j )
        nujk = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( i, l )
        kppx = min0( i, l )
        nuil = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( l, k )
        kppx = min0( l, k )
        nukl = kpp*( kpp - 1 )/2 + kppx
        kpp  = max0( i, j )
        kppx = min0( i, j )
        nuij = kpp*( kpp - 1 )/2 + kppx
        muil = nuil*( nuil + 1 )/2
        mujk = nujk*( nujk + 1 )/2
        muik = nuik*( nuik + 1 )/2
        mujl = nujl*( nujl + 1 )/2
        mukl = nukl*( nukl + 1 )/2
        muij = nuij*( nuij + 1 )/2
        cc0  = ediff( nuik ) + ediff( nujl )
     $       + 1.5d0*( gintgl( muik ) + gintgl( mujl ) )
```

```
     $        - 0.5d0*( gintgl( muil ) + gintgl( mujk ) )
     $        - gintgl( mukl ) - gintgl( muij )
        nuii = i*( i + 1 )/2
        nujj = j*( j + 1 )/2
        nukk = k*( k + 1 )/2
        null = l*( l + 1 )/2
        lmd  = max0( nujj, nukk )
        lmdx = min0( nujj, nukk )
        mukj = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( null, nuii )
        lmdx = min0( null, nuii )
        muli = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( null, nukk )
        lmdx = min0( null, nukk )
        mukl = lmd*( lmd - 1 )/2 + lmdx
        lmd  = max0( nujj, nuii )
        lmdx = min0( nujj, nuii )
        muij = lmd*( lmd - 1 )/2 + lmdx
        cc0  = cc0 - gintgl( mukj ) - gintgl( muli )
     $             + gintgl( mukl ) + gintgl( muij )
        ncimat = nomg*( nomg + 1 )/2
        cimat( ncimat ) = cc0

c$ohtawara
      if(idebug.ge.1) write(iout,110) i,k,j,l,  i,k,j,l,
     #nomg,nomg,cimat(ncimat)
 110  format('(',i2,' ->',i2,',',i2,' ->',i2,')',' -> (',
     #i2,' ->',i2,',',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    3 continue
c========( Wed Oct 25 07:34:11 JST 1995 )======
c$$$      do 3 nomg = momega(6)+1, momega(7)
c$$$        i = lci1( nomg )
c$$$        j = lci2( nomg )
c$$$        k = lci3( nomg ) + nocs
c$$$        l = lci4( nomg ) + nocs
c$$$        kpp  = max0( i, l )
c$$$        kppx = min0( i, l )
c$$$        nuil = kpp*( kpp - 1 )/2 + kppx
c$$$        muil = nuil*( nuil + 1 )/2
c$$$        cc0  = ediff( nuil ) - 0.5d0*gintgl( muil )
c$$$        kpp  = max0( k, j )
c$$$        kppx = min0( k, j )
c$$$        nujk = kpp*( kpp - 1 )/2 + kppx
c$$$        mujk = nujk*( nujk + 1 )/2
c$$$        cc0  = cc0 + ediff( nujk ) - 0.5d0*gintgl( mujk )
c$$$        kpp  = max0( i, k )
c$$$        kppx = min0( i, k )
c$$$        nuik = kpp*( kpp - 1 )/2 + kppx
c$$$        muik = nuik*( nuik + 1 )/2
c$$$        cc0  = cc0 + 1.5d0*gintgl( muik )
c$$$        kpp  = max0( l, j )
c$$$        kppx = min0( l, j )
c$$$        nujl = kpp*( kpp - 1 )/2 + kppx
c$$$        mujl = nujl*( nujl + 1 )/2
c$$$        cc0  = cc0 + 1.5d0*gintgl( mujl )
c$$$        kpp  = max0( l, k )
c$$$        kppx = min0( l, k )
c$$$        nukl = kpp*( kpp - 1 )/2 + kppx
c$$$        mukl = nukl*( nukl + 1 )/2
c$$$        cc0  = cc0 - gintgl( mukl )
c$$$        kpp  = max0( i, j )
c$$$        kppx = min0( i, j )
c$$$        nuij = kpp*( kpp - 1 )/2 + kppx
c$$$        muij = nuij*( nuij + 1 )/2
c$$$        cc0  = cc0 - gintgl( muij )
c$$$        nuii = i*( i + 1 )/2
c$$$        nujj = j*( j + 1 )/2
c$$$        nukk = k*( k + 1 )/2
c$$$        null = l*( l + 1 )/2
c$$$        lmd  = max0( nujj, nukk )
c$$$        lmdx = min0( nujj, nukk )
c$$$        mukj = lmd*( lmd - 1 )/2 + lmdx
c$$$        lmd  = max0( null, nuii )
c$$$        lmdx = min0( null, nuii )
c$$$        muli = lmd*( lmd - 1 )/2 + lmdx
c$$$        lmd  = max0( null, nukk )
c$$$        lmdx = min0( null, nukk )
c$$$        mukl = lmd*( lmd - 1 )/2 + lmdx
c$$$        lmd  = max0( nujj, nuii )
c$$$        lmdx = min0( nujj, nuii )
c$$$        muij = lmd*( lmd - 1 )/2 + lmdx
c$$$        cc0  = cc0 - gintgl( mukj ) - gintgl( muli )
c$$$     $             + gintgl( mukl ) + gintgl( muij )
c$$$        ncimat = nomg*( nomg + 1 )/2
c$$$        cimat( ncimat ) = cc0
c$$$    3 continue
c========( Wed Oct 25 07:34:00 JST 1995 )======
c
c
      call clockm('cimt55.   ')
c
c
      return
      end
      subroutine cimt5s
c=================================================================*
c                                   @@@@@@@                       *
c       @@@@      @    @    @   @@@@@ @         @@@@              *
c      @    @     @   @@  @@     @   @        @                  *
c      @          @   @ @@ @     @   @@@@@@    @@@@              *
c      @          @   @    @     @         @       @              *
c      @    @     @   @    @     @    @    @  @    @              *
c       @@@@      @   @    @     @     @@@@@    @@@@              *
c=================================================================*
c
c  get ci matrix elements between type V and single excitaion states
c
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c
      coef0 = - dsqrt( 1.5d0 )
      do 1 nomg = momega(6)+1, momega(7)
c$ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )
c        it = lci3( nomg ) + nocs
c        iu = lci4( nomg ) + nocs

        do 2 ntau = momega(1)+1, momega(2)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
c          write(6,*) '=== Error in cimt5s.f ===='
c          stop
c zero except for delta_is, etc...
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 0
 3        continue
```

```
          cc0 = 0.0d0
          if( k .eq. it ) then
            kpp  = max0( is, iu )
            kppx = min0( is, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - gintgl( mu )
            kpp  = max0( i, is )
            kppx = min0( i, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( ir, iu )
            kppx = min0( ir, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
          endif
          if( i .eq. ir ) then
            kpp  = max0( k, iu )
            kppx = min0( k, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( is, it )
            kppx = min0( is, it )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 - gintgl( mu )
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            kpp  = max0( is, iu )
            kppx = min0( is, iu )
            nux  = kpp*( kpp - 1 )/2 + kppx
            lmd  = max0( nu, nux )
            lmdx = min0( nu, nux )
            mu   = lmd*( lmd - 1 )/2 + lmdx
            cc0  = cc0 + gintgl( mu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = coef0*cc0

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,ir,it,is,iu,
     &nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,')',' -> (',i2,' ->',i2,',',i2,' ->',i2,')',
     &3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c
      call clockm('cimt5s.   ')
c
c
      return
      end
      subroutine cimtss
c=====================================================================*
c                                                                     *
c       @@@@      @    @    @   @@@@@   @@@@    @@@@                 *
c      @    @     @    @@  @@     @    @        @                    *
c      @          @    @ @@ @     @     @@@@    @@@@                 *
c      @          @    @    @     @         @       @                 *
c      @    @     @    @    @     @    @    @  @    @                 *
c       @@@@      @    @    @     @     @@@@    @@@@                 *
c=====================================================================*
c
c  get ci matrix elements between single excitaion states
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c$ohtawara
      common / io / in, iout, ifile(10)

c
c
      do 1 nomg = momega(1)+1, momega(2)
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        do 2 ntau = momega(1)+1, nomg
          ir = lci1( ntau )
          it = lci3( ntau ) + nocs
          kpp  = max0( i, k )
          kppx = min0( i, k )
          nu   = kpp*( kpp - 1 )/2 + kppx
          nuik = nu
          kpp  = max0( ir, it )
          kppx = min0( ir, it )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mikrt = lmd*( lmd - 1 )/2 + lmdx
          kpp  = max0( i, ir )
          kppx = min0( i, ir )
          nu   = kpp*( kpp - 1 )/2 + kppx
          kpp  = max0( k, it )
          kppx = min0( k, it )
          nux  = kpp*( kpp - 1 )/2 + kppx
          lmd  = max0( nu, nux )
          lmdx = min0( nu, nux )
          mirkt = lmd*( lmd - 1 )/2 + lmdx
          ncimat = nomg*( nomg - 1 )/2 + ntau
          cimat( ncimat ) = 2.0d0*gintgl( mikrt ) - gintgl( mirkt )
          if( i .eq. ir  .and.  k .eq. it ) then
            imo = lmo( i )
            kmo = lmo( k )
            cimat( ncimat ) = cimat( ncimat ) + eig( kmo ) - eig( imo )
            ediff( nuik ) = cimat( ncimat )
c========( Thu Oct 19 18:11:25 JST 1995 )======
c$$$          cimat( ncimat ) = cimat( ncimat ) + h000
c========( Thu Oct 19 18:11:25 JST 1995 )======
          endif

c$ohtawara
      if(idebug.ge.1) write(iout,100) i,k,ir,it,nomg,ntau,cimat(ncimat)
 100  format('(',i2,' ->',i2,')',' -> (',i2,' ->',i2,')',
     #3x,'CI matrix: ','(',i2,',',i2,')=',f10.6)

    2   continue
    1 continue
c
c     all done. return to caller.
      call clockm('cimtss.   ')
c
c
      return
      end
      subroutine civo00( vec,    eig,    locc,   lvac,   mdim,   ncsfs,
     $                   nend   )
c=====================================================================*
c                                                   @@@     @@@       *
c       @@@@      @    @    @   @@@@   @   @   @   @                 *
c      @    @     @    @    @  @    @     @ @     @                  *
c      @          @    @    @  @       @ @     @ @     @             *
c      @          @    @    @  @       @ @     @ @     @             *
c      @    @     @    @  @   @    @  @   @   @   @                 *
c       @@@@      @     @@     @@@@    @@@     @@@                  *
c=====================================================================*
c
c     OUTPUT MODULE FOR EIGENVALUES AND EIGENVECTORS OF CI HAMILTONIAN
c     MATRIX.
c     CAUTION.
c      MAXIMUM MO NO. PRINTED IN THIS ROUTINE IS LESS THAN 100 (I2).
c      IN CASE OF PRINTING MO WHOSE NO. IS GREATER THAN OR EQUAL TO 100,
c      FORMAT OVERFLOW WILL OCCUR.
c     ******************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      COMMON /PCONST/ PCON(20)
      DIMENSION VEC(MDIM,*),EIG(*),LOCC(*),LVAC(*)
 2000 Format(1H ,7X,11(7X,I4))
```

```
 2010 Format(1H ,'Eigenvalue' ,11(1X,F10.6))
 2015 Format(1H ,'HF-state', 2X, 11(1X,F10.6) )
 2020 Format(1H ,I2,'->',I2,4X,11(1X,F10.6))
C     ****************************************************************
C
C
      autoev = pcon( 3 )
c
c
      if(iwdth.eq.1) then
        iw=6
      else
        iw=11
      end if
c
      ilower=1
      iupper=iw
c
   10 continue
      iupper = min0( iupper, nend )
c
      write(iout,2000) (i,i=ilower,iupper)
      write(iout,2010) ( autoev*eig(i),i=ilower,iupper)
      write(iout,2015) ( vec( 1, j ), j = ilower, iupper )
c
      do 20 icibas = 2, ncsfs+1
        i = icibas - 1
        write(iout,2020) locc(i), lvac(i),
     $        ( vec(icibas,j), j=ilower, iupper )
   20 continue
c
      if( iupper .eq. nend ) go to 999
c
      ilower = ilower + iw
      iupper = iupper + iw
c
      go to 10
c
  999 continue
c
c
      return
      end
      subroutine civo01( vec,    eig,    locc,   lvac,   mdim,   ncsfs,
     $                   nend   )
c=================================================================*
c                                        @@@      @               *
c       @@@@      @    @    @   @@@@   @   @    @@               *
c      @    @     @    @    @  @    @ @      @  @ @               *
c      @          @    @    @  @    @ @      @    @               *
c      @          @    @    @  @    @ @      @    @               *
c      @    @     @     @  @   @    @  @   @     @               *
c       @@@@      @      @@     @@@@    @@@    @@@@@               *
c=================================================================*
C
C     OUTPUT MODULE FOR EIGENVALUES AND EIGENVECTORS OF CI HAMILTONIAN
C     MATRIX.
C     CAUTION.
C      MAXIMUM MO NO. PRINTED IN THIS ROUTINE IS LESS THAN 100 (I2).
C      IN CASE OF PRINTING MO WHOSE NO. IS GREATER THAN OR EQUAL TO 100,
C      FORMAT OVERFLOW WILL OCCUR.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      DIMENSION VEC(MDIM,*),EIG(*),LOCC(*),LVAC(*)
 2000 Format(1H ,7X,11(7X,I4))
 2010 Format(1H ,'Eigenvalue' ,11(1X,F10.6))
 2020 Format(1H ,I2,'=>',I2,4X,11(1X,F10.6))
C     ****************************************************************
      if(iwdth.eq.1) then
        iw=6
      else
        iw=11
      end if
c
      ilower=1
      iupper=iw
c
   10 continue
      iupper = min0( iupper, nend )
c
      write(iout,2000) (i,i=ilower,iupper)
c
      write(iout,2010) (eig(i),i=ilower,iupper)
c
      do 20 icibas = ncsfs+2, 2*ncsfs+1
        i = icibas - 1 - ncsfs
        write(iout,2020) locc(i), lvac(i),
     $        ( vec(icibas,j), j=ilower, iupper )
   20 continue
c
      if( iupper .eq. nend ) go to 999
c
      ilower = ilower + iw
      iupper = iupper + iw
c
      go to 10
c
  999 continue
c
c
      return
      end
      subroutine civo02( vec,    eig,    mdim,   nend  )
c=================================================================*
c                                        @@@    @@@@@              *
c       @@@@      @    @    @   @@@@   @   @  @      @             *
c      @    @     @    @    @  @    @ @      @       @             *
c      @          @    @    @  @    @ @      @  @@@@@             *
c      @          @    @    @  @    @ @      @ @                  *
c      @    @     @     @  @   @    @  @   @  @                   *
c       @@@@      @      @@     @@@@    @@@   @@@@@@@              *
c=================================================================*
c
c     output module for eigenvalues and eigenvectors of ci hamiltonian
c     matrix.
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'window.h'

      common/ io     / in, iout, ifile(10)
      common/ prwdth / iwdth
      common/ pconst / pcon(20)

      dimension vec(mdim,*), eig(*)
c
c
      autoev = pcon( 3 )
c
c
c$ohtawara
c check
c      write(iout,*) ' in civo02.f'
c      do n = 1,nend
c         write(iout,300) n,mcio(n),lci1(n),lci2(n),lci3(n),lci4(n)
c 300     format(6i3)
c      enddo

      write(iout,1000)
      do 1 n = 1, nend
        l1 = lci1(n)
        l2 = lci2(n)
        l3 = lci3(n)
        l4 = lci4(n)
        if( mcio( n ) .eq. 0 ) then
          l1 = 0
          l2 = 0
          l3 = 0
          l4 = 0
          write(iout,1001) n, l1, l2, l3, l4
        elseif( mcio( n ) .eq. 1 ) then
          l3 = l3 + nocs
          l1 = lmo( l1 )
          l2 = 0
          l3 = lmo( l3 )
          l4 = 0
          write(iout,1002) n, l1, l2, l3, l4
        elseif( mcio( n ) .ge. 2 ) then
          mtype = mcio( n ) - 1
          l3 = l3 + nocs
          l4 = l4 + nocs
          l1 = lmo( l1 )
          l2 = lmo( l2 )
          l3 = lmo( l3 )
          l4 = lmo( l4 )
```

```
          write(iout,1003) n, 11, 12, 13, 14, mtype
        endif
    1 continue
c
c
      if( iwdth .eq. 1 ) then
        iw = 6
      else
        iw = 11
      endif
c
      ilower = 1
      iupper = iw
      write(iout,1100)

c$ohtawara 1996.2.15
      if( iprint_yes .lt. 0 ) go to 999

   10 continue
c
c
      iupper = min0( iupper, nend )
      write(iout,2000) ( i, i = ilower, iupper )
      write(iout,2010) ( autoev*eig(i), i = ilower, iupper )
      write(iout,2011)
c$$$      write(iout,2015) ( vec( 1, j ),   j = ilower, iupper )
c
      do 20 icibas = 1, nend
        write(iout,2020) icibas,
     $        ( vec(icibas,j), j=ilower, iupper )
   20 continue
c
      if( iupper .eq. nend ) go to 999
c
      ilower = ilower + iw
      iupper = iupper + iw
c
      go to 10
c
  999 continue
c
c
 1000 format(//
     $        1h ,30('=-')/
     $        1h ,'Configuration Interaction information'/
     $        1h ,30('=-')/ )
 1001 format(1h ,i4, ' ( ', i4,',', i4, ' ) --> ( ', i4, ',', i4,
     $                ' ) : no excitation (Hartree-Fock state)')
 1002 format(1h ,i4, ' ( ', i4,',', i4, ' ) --> ( ', i4, ',', i4,
     $                ' ) : single electron excitation')
 1003 format(1h ,i4, ' ( ', i4,',', i4, ' ) --> ( ', i4, ',', i4,
     $                ' ) : type =', i4 )
 1100 format(//
     $        1h ,30('=-')/
     $        1h ,'CI eigen values and states'/
     $        1h ,30('=-')/ )
 2000 Format(1H ,7X,11(7X,I4))
 2010 Format(1H ,'Eigenvalue' ,11(1X,F10.6)/)
 2011 format(1H ,'(eV)')
 2015 Format(1H ,'HF-state', 2X, 11(1X,F10.6) )
 2020 Format(1H ,i6,4X,11(1X,F10.6))

      return
      end
C@(#)=CIVOUT(SUB)
      SUBROUTINE CIVOUT(VEC,EIG,LOCC,LVAC,MDIM,NCSFS)
C
C     OUTPUT MODULE FOR EIGENVALUES AND EIGENVECTORS OF CI HAMILTONIAN
C     MATRIX.
C     CAUTION.
C      MAXIMUM MO NO. PRINTED IN THIS ROUTINE IS LESS THAN 100 (I2).
C      IN CASE OF PRINTING MO WHOSE NO. IS GREATER THAN OR EQUAL TO 100,
C      FORMAT OVERFLOW WILL OCCUR.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      DIMENSION VEC(MDIM,*),EIG(*),LOCC(*),LVAC(*)
 2000 Format(1H ,7X,11(7X,I4))
 2010 Format(1H ,'Eigenvalue' ,11(1X,F10.6))
 2020 Format(1H ,I2,'->',I2,4X,11(1X,F10.6))
C     ****************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=6
      ELSE
        IW=11
      END IF
C
      ILOWER=1
      IUPPER=IW
C
   10 IF(IUPPER.GT.NCSFS) THEN
        IUPPER=NCSFS
      END IF
C
      WRITE(IOUT,2000) (I,I=ILOWER,IUPPER)
      WRITE(IOUT,2010) (EIG(I),I=ILOWER,IUPPER)
C
      DO 20 I=1,NCSFS
        WRITE(IOUT,2020) LOCC(I),LVAC(I),(VEC(I,J),J=ILOWER,IUPPER)
   20 CONTINUE
C
      IF(IUPPER.EQ.NCSFS) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
C
C@(#)=CLOCKM(SUB) ... Sun FORTRAN
      SUBROUTINE CLOCKM(RNAME)
C
C     Timing routine for MOS-F.
C     Note.
C      Use Sun FORTRAN Time Function
C      real function etime(single)
C      single ... real single(2).
C                 single(1) = user time (sec)
C                 single(2) = system time (sec)
C      etime  ... user + system time (sec)
C     ****************************************************************
      CHARACTER*7 RNAME
c$ohtawara
      character*8 systime
      integer*4 imonth,iday,iyear

      COMMON /IO    / IN,IOUT,IFILE(10)
      DIMENSION SINGLE(2)
      DATA TIME0/0.0E+00/
 2000 Format(1H ,'<<<<<< ',A7,' All done. CPU-Time = ',F8.2,' Sec.',
     &          ' Total =',I4.2,'H ',I2.2,'M ',I2.2,'S. >>>>>>')
C     ****************************************************************
      TIME1=ETIME(SINGLE)
C
      STEP=TIME1-TIME0
      ICPU=NINT(TIME1)
      CALL TOHMS(ICPU,IHR,IMIN,ISEC)
C
c$ohtawara
c to monitor by console (fort.6)
c      write(6,2000) rname,step,ihr,imin,isec
      call idate(imonth,iday,iyear)
      call time(systime)
      write(6,100) rname,iyear,imonth,iday,systime
 100  format(a7,' : ',i2,'.',i2,'.',i2,'.',3x,a8)

      WRITE(IOUT,2000) RNAME,STEP,IHR,IMIN,ISEC
      TIME0=TIME1
C
      RETURN
      END
C
C@(#)=CNDOS3(SUB)
      SUBROUTINE CNDOS3(COORD,ZETA,IDUMMY,TOANG,IAN,ZCSP3,NATOMS,IOUT)
C
C     DETERMINE THE EXPONENT FOR CARBON ATOMS.
C     On input:
C      COORD  ... CARTESIAN COORDINATES (UNIT : BOHR).
C      ZETA   ... EXPONENTS OF SLATER TYPE ORBITALS.
C      IDUMMY ... SCRATCH ARRAY FOR PRINTING.
C      TOANG  ... CONVERSION FACTOR FROM BOHR TO ANGSTROMS.
C      IAN    ... ATOMIC NUMBER.
C      ZCSP3  ... EXPONENT FOR SP3 CARBON ATOM (UNIT : BOHR**-1).
C      NATOMS ... NO. OF ATOMS.
```

```
C      IOUT   ... LOGICAL UNIT NUMBER FOR STANDARD OUTPUT.
C     On output:
C      ZETA   ... EXPONENTS FOR SLATER TYPE ORBITALS USED IN CNDO/S3
C                 METHOD.
C
C     CAUTION
C      IN CNDO/S3 METHOD,
C       EXPONENT FOR SP3 CARBON         ... 3.07 ANGSTROMS**-1
C       EXPONENT FOR SP AND SP2 CARBON ... 3.78 ANGSTROMS**-1
C
C     REFERENCE.
C      J.CHEM.PHYS.,63(5),1758(1975)
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION COORD(3,*),ZETA(*),IDUMMY(*),IAN(*)
      DATA RC1/1.20D+00/,RC2/1.60D+00/,RC3/2.00D+00/
 2000 Format(1H ,'CNDO/S3 method ... No. of sp3 carbon atoms =',I5,'.')
 2010 Format(1H ,'Center no. for sp3 carbon atoms ...')
 2020 Format(1H ,11(I5,2X))
C     ****************************************************************
      NCSP3=0
C
      DO 20 I=1,NATOMS
        IF(IAN(I).EQ.6) THEN
          NBONDS=0
          CXI   =COORD(1,I)
          CYI   =COORD(2,I)
          CZI   =COORD(3,I)
          DO 10 J=1,NATOMS
            IF(J.EQ.I) GO TO 10
              RIJ=DSQRT( (CXI-COORD(1,J))*(CXI-COORD(1,J))
     &                  +(CYI-COORD(2,J))*(CYI-COORD(2,J))
     &                  +(CZI-COORD(3,J))*(CZI-COORD(3,J)) )*TOANG
              IF     ( (IAN(J).LE. 2)) THEN
                IF(RIJ.LT.RC1) NBONDS=NBONDS+1
              ELSE IF( (IAN(J).LE.10)) THEN
                IF(RIJ.LT.RC2) NBONDS=NBONDS+1
              ELSE
                IF(RIJ.LE.RC3) NBONDS=NBONDS+1
              END IF
   10     CONTINUE
          IF(NBONDS.GT.3) THEN
            ZETA(I)=ZCSP3
            NCSP3  =NCSP3+1
            IDUMMY(NCSP3)=I
          END IF
        END IF
   20 CONTINUE
C
      WRITE(IOUT,2000) NCSP3
      IF(NCSP3.GT.0) THEN
        WRITE(IOUT,2010)
        IW    =11
        ILOWER=1
        IUPPER=IW
   30   IF(IUPPER.GT.NCSP3) THEN
          IUPPER=NCSP3
        END IF
        WRITE(IOUT,2020) (IDUMMY(I),I=ILOWER,IUPPER)
        IF(IUPPER.EQ.NCSP3) GO TO 40
          ILOWER=ILOWER+IW
          IUPPER=IUPPER+IW
          GO TO 30
      END IF
C
   40 RETURN
      END
C
C@(#)=COEFF(FUN) ... CNDO2/3R(COEFF)
      FUNCTION COEFF(NA,NB,LA,LB,M,KL,LL)
C
C     ==============================================================
C     CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R
C     ==============================================================
C     DOUBLE PRECISION FUNCTION COEFF ... FROM CNDO 2/3R
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER A,B,C,D,P,U,V,BA,BE,BL,BMA,CA,CE,CL,CMA,BMI,UL,VL,DMA,CD
      INTEGER AU
      DIMENSION CO(3,3,3)
      DATA CO/ 8.,0.,-4.,0.,4.,0.,0.,0.,4.,0.,8.,0.,0.,0.,12.,0.,0.,0.,
     1  0.,0.,12.,0.,0.,0.,0.,0.,0. /
C     ****************************************************************
      COEFF = 0.D0
      K   = KL - 1
      L   = LL - 1
      KKLL=K + L
      NNLL=NA + NB - LA - LB
      IF   (KKLL.LT.NNLL) RETURN
      LAM = LA - M + 1
      LBM = LB - M + 1
      DO  210   UL =  1, LAM
      DO  200   VL =  1, LBM
      F   = 0.D0
      U   = UL - 1
      V   = VL - 1
      P   = NA + NB - U - V
      IAA = MOD(L+P-K,2)
      IF    (IAA.NE.0)  GO TO 200
      NAMU= NA - M - U
      NBMV= NB - M - V
      IMA = MIN0(L,U)
      JMA = MIN0(L,V)
      CMA = MIN0(L,NAMU)
      DMA = MIN0(L,NBMV)
      L2  = INT(FLOAT(L)/2.)
      BMA = MIN0(L2,M)
      BMI = INT(FLOAT(L-IMA-JMA-CMA-DMA)/2.)
      BMI = MAX0(0,BMI)
      BA  = BMI + 1
      BE  = BMA + 1
      IF(BE.LT.BA) GO TO 200
      DO 130   BL =  BA , BE
      B   = BL - 1
      IJCD= L - 2*B
      IE  = MIN0(U,IJCD) + 1
      IA  = IJCD -JMA - CMA - DMA
      IA  = MAX0(0,IA) + 1
      IF(IE.LT.IA) GO TO 130
      DO  120   IL =  IA , IE
      I   = IL - 1
      JCD = IJCD - I
      JE  = MIN0(V,JCD) + 1
      JA  = JCD - CMA - DMA
      JA  = MAX0(0,JA) + 1
      IF(JE.LT.JA) GO TO 120
      DO  110   JL =  JA, JE
      J   = JL - 1
      CD  = JCD - J
      CE  = MIN0(NAMU,CD) + 1
      CA  = CD - DMA
      CA  = MAX0(0,CA) + 1
      IF(CE.LT.CA) GOTO 110
      DO  100   CL =  CA , CE
      C   = CL - 1
      D   = CD - C
      AU  =MOD(P-K+I+J-C-D,2)
      IF   (AU.NE.0) GO TO 100
      A   =(P-K+I+J-C-D)/2
      IF   (A.LT.0) GO TO 100
      IF   (A.GT.M) GO TO 100
      F   = F + BINOM(M,A)*BINOM(M,B)*BINOM(U,I)*BINOM(V,J)*BINOM(NAMU,
     1    C)*BINOM(NBMV,D)*(-1.D0)**(A+B+J+D)
  100 CONTINUE
  110 CONTINUE
  120 CONTINUE
  130 CONTINUE
      COEFF = COEFF + CO(LA+1,M+1,U+1)*CO(LB+1,M+1,V+1)*F
  200 CONTINUE
  210 CONTINUE
      RETURN
      END
C
C@(#)=COEFFS(FUN) ... CNDO2/3R(COEFFS)
      FUNCTION COEFFS(NA,NB,K)
C
C     ==============================================================
C     CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R
C     ==============================================================
C     DOUBLE PRECISION FUNCTION COEFFS ... FROM CNDO 2/3R
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
C     ****************************************************************
      COEFFS=0.D0
      L=NA+NB-K
      IE=MIN0(L,NA)+1
```

```
      JE=MIN0(L,NB)
      IA=L-JE+1
      DO 10 IL=IA,IE
      I=IL-1
      J=L-I
   10 COEFFS=COEFFS + BINOM(NA,I)*BINOM(NB,J)*(-1.D0)**J
      RETURN
      END
C
C@(#)=CONCLO(SUB) ... GAUSSIAN80(CONCLO)
      SUBROUTINE CONCLO(D,DEXT,SCFCRT,RMSDP,MBPCK,NBASIS,NTT,JCYC,I4OR3,
     &                  IP0,IP1,IP2,IP3,LOGCON,LOGEXT)
C
C     INVESTIGATE CONVERGENCE ON DENSITY MATRIX AND TRY TO PERFORM
C     EXTRAPOLATION ... FROM GAUSSIAN80.
C
C     NOTE On output.
C      1. RMSDP  ... ROOT MEAN SQUARE OF DIFFERENCE BETWEEN D(I,IP0) AND
C                    D(I,IP1).
C      2. I4OR3  ... 0 : DID NOT PERFORM EXTRAPOLATION IN THIS ROUTINE.
C                    1 : PERFORMED 4-POINT EXTRAPOLATION.
C                    2 : PERFORMED AITKEN'S 3-POINT EXTRAPOLATION.
C      3. LOGCON ... IN CASE DENSITY CONVERGED, LOGCON=.TRUE.
C      4. LOGEXT ... IN CASE EXTRAPOLATION PROCEDURE IS PERFORMED IN
C                    THIS ROUTINE, LOGEXT=.TRUE.
C
C     REFERENCE.
C      R.DAUDEL ET AL, "QUANTUM CHEMISTRY", WILEY, 1983.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL LOGCON,LOGEXT
      DIMENSION D(MBPCK,*),DEXT(*)
      DIMENSION SP(4,4),DP(4)
      DATA ZERO/0.0D+00/,PT99/0.99D+00/,PT995/0.995D+00/,ONE/1.0D+00/,
     &     ONEPT9/1.9D+00/,TWO/2.0D+00/,FOUR/4.0D+00/
C     ***************************************************************
C
C     THE ARRAY V USED BY THE COMMENT STATEMENTS
C     MEAN AS FOLLOWED.
C     V(IP0) ... D(I,IP0)-D(I,IP1) ==> D(I,IP1)
C     V(IP1) ... D(I,IP1)-D(I,IP2) ==> D(I,IP2)          ---------
C     V(IP2) ... D(I,IP2)-D(I,IP3) ==> D(I,IP3)          JCYC.EQ.1
C                                                        ---------
      IF(JCYC.EQ.1) RETURN
C                                                        ---------
C     SP ... SCALAR PRODUCT OF VECTOR, V(IP0).           JCYC.GE.2
C     DP ... NORM OF SP.                                 ---------
      S=ZERO
      DO 10 I=1,NTT
        D(I,IP1)=D(I,IP0)-D(I,IP1)
        S=S+D(I,IP1)*D(I,IP1)
   10 CONTINUE
      SP(IP0,IP0)=S
      DP(IP0)=DSQRT(S)
      RMSDP=DP(IP0)/DFLOAT(NBASIS)
C
C     IF DENSITY CONVERGED, RETURN TO CALLER AND QUIT SCF LOOP.
      IF(RMSDP.LT.SCFCRT) THEN
        LOGCON=.TRUE.
        RETURN
      END IF
C
      IF(JCYC.EQ.2) RETURN
C                                                        ---------
C                                                        JCYC.GE.3
C                                                        ---------
      S=ZERO
      DO 20 I=1,NTT
        S=S+D(I,IP1)*D(I,IP2)
   20 CONTINUE
      SP(IP0,IP1)=S
C
C     PHI IS A ANGLE BETWEEN VECTOR V(IP0) AND V(IP1).
      COSPHI=S/(DP(IP0)*DP(IP1))
C
      IF(JCYC.EQ.3) RETURN
C                                                        ---------
C                                                        JCYC.GE.4
C                                                        ---------
      S=ZERO
      DO 30 I=1,NTT
        S=S+D(I,IP1)*D(I,IP3)
   30 CONTINUE
      SP(IP0,IP2)=S
C
      V0  =DP(IP0)
      V1  =DP(IP1)
      V2  =DP(IP2)
      V0V0=SP(IP0,IP0)
      V1V1=SP(IP1,IP1)
      V2V2=SP(IP2,IP2)
      V0V1=SP(IP0,IP1)
      V0V2=SP(IP0,IP2)
      V1V2=SP(IP1,IP2)
C
C     V(IP2)(PROJECT) = A*V(IP0)+B*V(IP1).
C     V(IP2)(PROJECT) : PROJECTION VECTOR OF V(IP2) TO THE PLANE OF THE
C                       VECTOR V(IP0) AND V(IP1).
      TRM= V0V0*V1V1-V0V1*V0V1
      A  =(V0V2*V1V1-V0V1*V1V2)/TRM
      B  =(V1V2*V0V0-V0V1*V0V2)/TRM
C
C     PSI IS A ANGLE BETWEEN VECTOR V(IP2) AND THE PLANE OF V(IP0) AND
C     V(IP1).
      CP2   =(A*A*V0V0+B*B*V1V1+TWO*A*B*V0V1)/V2V2
      COSPSI=DSQRT(CP2)
C
C     IF COSPSI.LE.0.99 VECTORS V(IP0), V(IP1) AND V(IP2) ARE NOT
C     COPLANAR. RETURN TO CALLER WITHOUT PERFORMING EXTRAPOLATION.
      IF(COSPSI.LE.PT99) RETURN
C
C     VECTORS V(IP0), V(IP1) AND V(IP2) ARE NEARLY COPLANAR.
C     V(IP0)=(1/A)*V(IP2)(PROJECT)+(-B/A)*V(IP1)
      X =ONE/A
      Y =-B/A
C
C     XY : DETERMINANT OF QUADRATIC EQUATION P**2+-Y*P-X=0
      XY=Y*Y+FOUR*X
C
C     1. VECTORS V(IP0),V(IP1) AND V(IP2) ARE NEARLY COPLANAR, AND
C     2. 0 <= XY <= 1.9,
C        THEN PERFORM 4-POINT EXTRAPOLATION.
      IF(XY.GE.ZERO. AND. XY.LE.ONEPT9) THEN
C                                                          ------------
C                                                          4-POINT EXT.
C                                                          ------------
        TRM2=ONE/(A+B-ONE)
        TRM1=(ONE-B)*TRM2
        DO 40 I=1,NTT
          DEXT(I)=D(I,IP0)+D(I,IP1)*TRM1+D(I,IP2)*TRM2
   40   CONTINUE
        LOGEXT=.TRUE.
        I4OR3=1
C
C     IF ABS(COSPHI) > 0.995, VECTORS V(IP0) AND V(IP1) ARE NEARLY
C     COLINEAR.
C     1. VECTORS V(IP0),V(IP1) AND V(IP2) ARE NEARLY COPLANAR, AND
C     2. (XY < 0 .OR. 1.9 <XY), AND
C     3. VECTORS V(IP0) AND V(IP1) ARE NEARLY COLINEAR,
C        THEN PERFORM 3-POINT EXTRAPOLATION.
      ELSE IF(DABS(COSPHI).GT.PT995) THEN
C                                                          ------------
C                                                          3-POINT EXT.
C                                                          ------------
        TRM=V0/(V1*COSPHI-V0)
        DO 50 I=1,NTT
          DEXT(I)=D(I,IP0)+D(I,IP1)*TRM
   50   CONTINUE
        LOGEXT=.TRUE.
        I4OR3=2
      END IF
C
      RETURN
      END
C
C@(#)=CPRINT(SUB)
      SUBROUTINE CPRINT(COORD,IAN,NAT)
C
C     PRINT OUT CARTESIAN COORDINATES. STANDARD ORIENTATION.
C     NOTE.
C      ONE AND TWO ELECTRON INTEGRALS ARE CALCULATED IN THIS COORDINATE
C      SYSTEM. VECTOR AMOUNT SUCH AS DIPOLE MOMENTS IS SUBJECT TO THIS
C      COORDINATE SYSTEM.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO    / IN,IOUT,IFILE(10)
```

```
      DIMENSION COORD(3,*),IAN(*)
 2000 Format(1H ,'Standard geometry ...')
 2010 Format(1H ,79('-'))
 2020 Format(1H ,2X,'Center',7X,'Atomic',21X,'Coordinates (Angstroms)'/
     &       1H ,2X,'number',7X,'number',14X,'x',17X,'y',17X,'z')
 2030 Format(1H ,2X,I4,11X,I2,4X,3(4X,F14.9))
C     ****************************************************************
      WRITE(IOUT,2000)
      WRITE(IOUT,2010)
      WRITE(IOUT,2020)
      WRITE(IOUT,2010)
C
      DO 10 I=1,NAT
        IA=IAN(I)
        X =COORD(1,I)
        Y =COORD(2,I)
        Z =COORD(3,I)
        WRITE(IOUT,2030) I,IA, X,Y,Z
   10 CONTINUE
C
      WRITE(IOUT,2010)
C
      RETURN
      END
c
c@(#)=CSort2(Sub)
      Subroutine CSort2(CArray,IArray,ChrTmp,ISize)
      Implicit Integer (a-z)
c
c     Sorting of Character Array using Baka sorting method.
c     On input:
c      CArray ... Character array to sort.
c      IArray ... Integer array to sort in alphabetical order.
c      ChrTmp ... Temporary variable with the same length as CArray.
c      ISize  ... Size of CArray and IArray.
c     On output:
c      CArray ... Alphabetically sorted CArray.
c      IArray ... Sorted IArray.
c
      Character*(*) CArray,ChrTmp
      Dimension CArray(*),IArray(*)
c
      Do 20 i=1,ISize
        Do 10 j=i,ISize
          If (CArray(j) .lt. CArray(i)) Then
            ChrTmp=CArray(i)
            CArray(i)=CArray(j)
            CArray(j)=ChrTmp
            IntTmp=IArray(i)
            IArray(i)=IArray(j)
            IArray(j)=IntTmp
          EndIf
   10   Continue
   20 Continue
c
      Return
      End
C
C@(#)=CTOZ(SUB) ... MOPAC(XYZINT)
      SUBROUTINE CTOZ(COORD,ZCOORD,DEGREE,NA,NB,NC,NATOMS,PI)
C
C     ==========================================================
C     MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC
C     ==========================================================
C
C     CONVERT CARTESIAN COORDINATES INTO INTERNAL COORDINATES.
C     NOTE.
C      THIS ROUTINE IS THE SAME AS XYZINT OF MOPAC.
C      COORD  ... CARTESIAN COORDINATES.
C      ZCOORD ... INTERNAL COORDINATES.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION COORD(3,*),NA(*),NB(*),NC(*),ZCOORD(3,*)
C     ****************************************************************
      NAI1=0
      NAI2=0
      DO 20 I=1,NATOMS
        NA(I)=2
        NB(I)=3
        NC(I)=4
        IM1=I-1
        IF(IM1.EQ.0) GO TO 20
        SUM=100.D0
        DO 10 J=1,IM1
          R=(COORD(1,I)-COORD(1,J))**2+
     &      (COORD(2,I)-COORD(2,J))**2+
     &      (COORD(3,I)-COORD(3,J))**2
          IF(R.LT.SUM.AND.NA(J).NE.J.AND.NB(J).NE.J) THEN
            SUM=R
            K=J
          END IF
   10   CONTINUE
C
C     ATOM I IS NEAREST TO ATOM K
        NA(I)=K
        IF(I.GT.2) NB(I)=NA(K)
        IF(I.GT.3) NC(I)=NB(K)
   20 CONTINUE
C
      NA(1)=0
      NB(1)=0
      NC(1)=0
      NB(2)=0
      NC(2)=0
      NC(3)=0
C
      CALL XYZGEO(COORD,ZCOORD,DEGREE,NA,NB,NC,NATOMS,PI)
C
      RETURN
      END
C
C@(#)=CTRNS(SUB)
      SUBROUTINE CTRNS(C,IAN,NATOMS,IOUT)
C
C     TRANSLATE THE CENTER OF CHARGE TO THE ORIGIN OF COORDINATE SYSTEM
C     On input:
C      C      ... Cartesian coordinates inputed or determined by
C                 routine ZToC.
C      IAN    ... ATOMIC NUMBER.
C      NATOMS ... NO. OF ATOMS.
C     On output:
C      C      ... CARTESIAN COORDINATES WHOSE CENTER OF CHARGE IS
C                 TRANSLATED TO ORIGIN.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION C(3,*),IAN(*)
      DATA ZERO /0.0D+00/
 2000 Format(1H ,'Translate molecule ... The origin is the center of ',
     &          'nuclear charges.')
C     ****************************************************************
      WRITE(IOUT,2000)
C
      XCENTR=ZERO
      YCENTR=ZERO
      ZCENTR=ZERO
      CSUM  =ZERO
C
C     GET THE POSITION OF THE MOLECULAR CHARGE CENTER.
      DO 10 I=1,NATOMS
        IF     (IAN(I).LE. 2) THEN
          IA=IAN(I)
        ELSE IF(IAN(I).LE.10) THEN
          IA=IAN(I)- 2
        ELSE IF(IAN(I).LE.18) THEN
          IA=IAN(I)-10
        ELSE IF(IAN(I).LE.29) THEN
          IA=IAN(I)-18
        ELSE IF(IAN(I).LE.35) THEN
          IA=IAN(I)-28
        END IF
        ANUMBR=DFLOAT(IA)
        XCENTR=XCENTR+ANUMBR*C(1,I)
        YCENTR=YCENTR+ANUMBR*C(2,I)
        ZCENTR=ZCENTR+ANUMBR*C(3,I)
        CSUM  =CSUM  +ANUMBR
   10 CONTINUE
C
      XCENTR=XCENTR/CSUM
      YCENTR=YCENTR/CSUM
      ZCENTR=ZCENTR/CSUM
C
C     THEN TRANSLATE.
      DO 20 I=1,NATOMS
        C(1,I)=C(1,I)-XCENTR
        C(2,I)=C(2,I)-YCENTR
        C(3,I)=C(3,I)-ZCENTR
```

```
   20 CONTINUE
C
      RETURN
      END
C
C@(#)=CZPRNT(SUB)
      SUBROUTINE CZPRNT(COORD,ZCOORD,IATN,NA,NB,NC,NAT)
C
C     PRINT OUT INPUTED GEOMETRY (CARTESIAN AND INTERNAL)
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*2 ELMNT2,EL
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PTb12 / ELMNT2(99)
      DIMENSION COORD(3,*),ZCOORD(3,*),IATN(*),NA(*),NB(*),NC(*)
 2000 Format(1H ,'Input geometry ...')
 2010 Format(1H ,79('-'))
 2020 Format(1H ,16X,'Cartesian(Ang)',18X,'Internal(Ang and Deg)'/
     &       1H ,'  Cn  At',5X,'x',8X,'y',8X,'z',7X,'Length',3X,'Alpha',
     &            5X,'Beta',4X,'NA   NB   NC')
 2030 Format(1H ,I4,2X,A2,1X,3(F8.4,1X),2X,F7.4,1X,F7.3,1X,F8.3,1X,
     &         3(1X,I4))
C     ***************************************************************
      WRITE(IOUT,2000)
      WRITE(IOUT,2010)
      WRITE(IOUT,2020)
      WRITE(IOUT,2010)
C
      DO 10 I=1,NAT
        EL=ELMNT2(IATN(I))
        X =COORD(1,I)
        Y =COORD(2,I)
        Z =COORD(3,I)
        R =ZCOORD(1,I)
        A =ZCOORD(2,I)
        B =ZCOORD(3,I)
        IA=NA(I)
        IB=NB(I)
        IC=NC(I)
        WRITE(IOUT,2030) I,EL, X,Y,Z, R,A,B, IA,IB,IC
   10 CONTINUE
C
      WRITE(IOUT,2010)
C
      RETURN
      END
C
C@(#)=D1EOPE(SUB)
      SUBROUTINE D1EOPE(TGN,TNN,CIEIG,BETAEO,BVEC,MCICSF,MCIINT,NCSFS)
C
C     GET BETA(I,J,K) FOR EOPE.
c     Oct/92 ... A.M.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*3 SHGSUF,EOSUF
      Logical Anchor
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /BETASF/ SHGSUF(18),EOSUF(18)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PLANCK,SLIGHT,EVTOER,PCON(14)
      Common /Anchor/ Anchor
      DIMENSION TGN(MCICSF,*),TNN(MCIINT,*)
      DIMENSION CIEIG(*),BETAEO(*),BVEC(*)
      DATA HENENM/6.328D+02/
      DATA ZERO/0.0D+00/,PT05/0.05D+00/
      DATA TENP5/1.0D+05/,TENP7/1.0D+07/
      DATA NBETA/18/
 2000 Format(1H ,38('-')/
     &       1H ,'Beta(i,j,k) for EOPE ... Beta(-w;0,w).'/
     &       1H ,38('-'))
 2010 Format(1H ,'Incident light : He-Ne laser (632.8 nm,',F7.4,' eV) ',
     &            '...')
 2020 Format(1H ,2X,'X  =',F12.5,4X,'Y  =',F12.5,4X,'Z  =',F12.5,
     &            2X,'Total=',F12.5)
 2030 Format(1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,1(3X,A3,' =',F12.5))
 2040 Format(1H ,33('=')/
     &       1H ,'Frequency dependent beta for EOPE'/
     &       1H ,33('='))
 2050 Format(1H ,I4,'.  Static ...')
```

```
 2060 Format(1H ,I4,'.',F6.3,' eV = ',F8.2,' nm ...')
 2070 Format(1H ,2X,'X =',F10.3,3X,'Y =',F10.3,3X,'Z =',F10.3,
     &            2X,'Total=',F10.3)
 2080 Format(1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,2(2X,A3,'=',F10.3))
 2090 Format(1H ,I4,'. E(S1)-E(Incident light) =',F6.3,' eV ...')
 2100 Format(1H ,I4,'. E(S1)-E(Incident light) =',F6.3,' eV (Static) '
     &            ,'...')
 9910 Format(1H ,58('X')/
     &       1H ,'TRANSITION ENERGY FOR S1 STATE .LT. ENERGY OF ',
     &           'HE/NE LASER.'/
     &       1H ,58('X'))
 9920 Format(1H ,29('X')/
     &       1H ,'TOO LITTLE TRANSITION ENERGY.'/
     &       1H ,29('X'))
C     ***************************************************************
      IPREO=IOPDIP(14)
C
      TOKYS =(EVTOER*AUTOEV*TENP5)/(PLANCK*SLIGHT)
      TOESU =TOANG**5/(EVTOER*SLIGHT)
C
      HENEAU=TENP7/(HENENM*TOKYS)
      HENEEV=HENEAU*AUTOEV
      IF(CIEIG(1).LT.HENEAU) THEN
        WRITE(IOUT,9910)
        GO TO 10
      END IF
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010) HENEEV
c
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_EOPE',29,
     &                         HeNeEV,BVec,BTotal,
     &                         IFile(10),Junk,.True.)
C
C     MCICSF, MCIINT ... FROM INCLUDE STATEMENT.
C
C     GET BETA FOR NE/NE LASER.
      CALL EOPE(HENEAU,BETAEO,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
C
C     THEN CONVERT UNITS OF BETA FROM (BOHR)**5/CHARGE(AU) TO
C     10**(-30) CM**5/STATCOULOMB.
      CALL CHGUNT(BVEC  ,3    ,TOESU)
      CALL CHGUNT(BETAEO,NBETA,TOESU)
C
      BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
C     THEN PRINT.
      WRITE(IOUT,2020) (BVEC(I),I=1,3),BTOTAL
      WRITE(IOUT,2030) (EOSUF(I),BETAEO(I),I=1,NBETA)
c
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_EOPE',29,
     &                         HeNeEV,BVec,BTotal,
     &                         IFile(10),Junk,.False.)
C
   10 IF(IPREO.EQ.0) RETURN
C
C     GET FREQUENCY DEPENDENT BETA.
      WRITE(IOUT,2040)
C
      ISTEP =0
      WEVMAX=CIEIG(1)*AUTOEV
C
      IF(IPREO.EQ.1) THEN
        WEV  = ZERO
        WSTEP= PT05
      ELSE
        WEV  = WEVMAX-PT05
        WSTEP=-PT05
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_EOPERev',32,
     &                           HeNeEV,BVec,BTotal,
     &                           IFile(10),Junk,.True.)
        GO TO 30
      END IF
C                                                  -----------------
C                                                  SCAN FROM 0 TO S1
C                                                  -----------------
   20 IF(WEV.GT.WEVMAX) GO TO 50
C
        ISTEP=ISTEP+1
        WAU  =WEV/AUTOEV
```

```
C
      IF(ISTEP.EQ.1) THEN
        WRITE(IOUT,2050) ISTEP
      ELSE
        WNM=TENP7/(WAU*TOKYS)
        WRITE(IOUT,2060) ISTEP,WEV,WNM
      END IF
C
      CALL EOPE(WAU,BETAEO,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
      CALL CHGUNT(BVEC  ,3    ,TOESU)
      CALL CHGUNT(BETAEO,NBETA,TOESU)
C
      BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
      WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
      WRITE(IOUT,2080) (EOSUF(J),BETAEO(J),J=1,NBETA)
C
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_EOPE',29,
     &                         WEV,BVec,BTotal,
     &                         IFile(10),Junk,.False.)
C
      WEV=WEV+WSTEP
C
    GO TO 20
C                                                  -----------------
C                                                  SCAN FROM S1 TO 0
C                                                  -----------------
   30 IF(WEV.LE.ZERO) GO TO 40
C
      ISTEP = ISTEP+1
      WAU   = WEV/AUTOEV
      WDELTA=-WSTEP*DFLOAT(ISTEP)
      WRITE(IOUT,2090) ISTEP,WDELTA
C
      CALL EOPE(WAU,BETAEO,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
      CALL CHGUNT(BVEC  ,3    ,TOESU)
      CALL CHGUNT(BETAEO,NBETA,TOESU)
C
      BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
      WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
      WRITE(IOUT,2080) (EOSUF(J),BETAEO(J),J=1,NBETA)
C
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_EOPERev',32,
     &                         WDelta,BVec,BTotal,
     &                         IFile(10),Junk,.False.)
C
      WEV=WEV+WSTEP
C
    GO TO 30
C
   40 ISTEP = ISTEP+1
      WAU   = ZERO
      WDELTA= CIEIG(1)*AUTOEV
      WRITE(IOUT,2100) ISTEP,WDELTA
C
      CALL EOPE(WAU,BETAEO,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
      CALL CHGUNT(BVEC  ,3    ,TOESU)
      CALL CHGUNT(BETAEO,NBETA,TOESU)
C
      BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
      WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
      WRITE(IOUT,2080) (EOSUF(J),BETAEO(J),J=1,NBETA)
C
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_EOPERev',32,
     &                         WDelta,BVec,BTotal,
     &                         IFile(10),Junk,.False.)
C
   50 IF(ISTEP.EQ.0) THEN
        WRITE(IOUT,9920)
        CALL MERROR
      END IF
C
      RETURN
      END
C
C@(#)=D1OR(SUB)
      SUBROUTINE D1OR(TGN,TNN,CIEIG,BETAOR,BVEC,MCICSF,MCIINT,NCSFS)
C
C     GET BETA(I,J,K) FOR OR.
C     Oct/92 ... A.M.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*3 SHGSUF,EOSUF
      Logical Anchor
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /BETASF/ SHGSUF(18),EOSUF(18)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PLANCK,SLIGHT,EVTOER,PCON(14)
      Common /Anchor/ Anchor
      DIMENSION TGN(MCICSF,*),TNN(MCIINT,*)
      DIMENSION CIEIG(*),BETAOR(*),BVEC(*)
      DATA HENENM/6.328D+02/
      DATA ZERO/0.0D+00/,PT05/0.05D+00/
      DATA TENP5/1.0D+05/,TENP7/1.0D+07/
      DATA NBETA/18/
 2000 Format(1H ,36('-')/
     &       1H ,'Beta(i,j,k) for OR ... Beta(0;w,-w).'/
     &       1H ,36('-'))
 2010 Format(1H ,'Incident light : He-Ne laser (632.8 nm,',F7.4,' eV) ' ,
     &           '...')
 2020 Format(1H ,2X,'X  =',F12.5,4X,'Y  =',F12.5,4X,'Z  =',F12.5,
     &           2X,'Total=',F12.5)
 2030 Format(1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,1(3X,A3,' =',F12.5))
 2040 Format(1H ,31('=')/
     &       1H ,'Frequency dependent beta for OR'/
     &       1H ,31('='))
 2050 Format(1H ,I4,'.  Static ...')
 2060 Format(1H ,I4,'.',F6.3,' eV = ',F8.2,' nm ...')
 2070 Format(1H ,2X,'X =',F10.3,3X,'Y =',F10.3,3X,'Z =',F10.3,
     &           2X,'Total=',F10.3)
 2080 Format(1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,2(2X,A3,'=',F10.3))
 2090 Format(1H ,I4,'. E(S1)-E(Incident light) =',F6.3,' eV ...')
 2100 Format(1H ,I4,'. E(S1)-E(Incident light) =',F6.3,' eV (Static) '
     &           ,'...')
 9910 Format(1H ,58('X')/
     &       1H ,'TRANSITION ENERGY FOR S1 STATE .LT. ENERGY OF ',
     &           'HE/NE LASER.'/
     &       1H ,58('X'))
 9920 Format(1H ,29('X')/
     &       1H ,'TOO LITTLE TRANSITION ENERGY.'/
     &       1H ,29('X'))
C     ***************************************************************
      IPROR=IOPDIP(15)
C
      TOKYS =(EVTOER*AUTOEV*TENP5)/(PLANCK*SLIGHT)
      TOESU =TOANG**5/(EVTOER*SLIGHT)
C
      HENEAU=TENP7/(HENENM*TOKYS)
      HENEEV=HENEAU*AUTOEV
      IF(CIEIG(1).LT.HENEAU) THEN
        WRITE(IOUT,9910)
        GO TO 10
      END IF
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010) HENEEV
C
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_OR',27,
     &                         HeNeEV,BVec,BTotal,
     &                         IFile(10),Junk,.True.)
C
C     MCICSF, MCIINT ... FROM INCLUDE STATEMENT.
C
C     GET BETA FOR HENE LASER.
      CALL OR(HENEAU,BETAOR,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
C
C     THEN CONVERT UNITS OF BETA FROM (BOHR)**5/CHARGE(AU) TO
C     10**(-30) CM**5/STATCOULOMB.
      CALL CHGUNT(BVEC  ,3    ,TOESU)
      CALL CHGUNT(BETAOR,NBETA,TOESU)
C
      BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
C     THEN PRINT.
      WRITE(IOUT,2020) (BVEC(I),I=1,3),BTOTAL
      WRITE(IOUT,2030) (SHGSUF(I),BETAOR(I),I=1,NBETA)
C
```

```
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_OR',27,
     &                         HeNeEV,BVec,BTotal,
     &                         IFile(10),Junk,.False.)
C
   10 IF(IPROR.EQ.0) RETURN
C
C     GET FREQUENCY DEPENDENT BETA.
      WRITE(IOUT,2040)
C
      ISTEP=0
      WEVMAX=CIEIG(1)*AUTOEV
C
      IF(IPROR.EQ.1) THEN
        WEV  = ZERO
        WSTEP= PT05
      ELSE
        WEV  = WEVMAX-PT05
        WSTEP=-PT05
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_ORRev',30,
     &                           WDelta,BVec,BTotal,
     &                           IFile(10),Junk,.True.)
        GO TO 30
      END IF
C                                                  -----------------
C                                                  SCAN FROM 0 TO S1
C                                                  -----------------
   20 IF(WEV.GT.WEVMAX) GO TO 50
C
        ISTEP=ISTEP+1
        WAU  =WEV/AUTOEV
C
        IF(ISTEP.EQ.1) THEN
          WRITE(IOUT,2050) ISTEP
        ELSE
          WNM=TENP7/(WAU*TOKYS)
          WRITE(IOUT,2060) ISTEP,WEV,WNM
        END IF
C
        CALL OR(WAU,BETAOR,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
        CALL CHGUNT(BVEC  ,3    ,TOESU)
        CALL CHGUNT(BETAOR,NBETA,TOESU)
C
        BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
        WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
        WRITE(IOUT,2080) (SHGSUF(J),BETAOR(J),J=1,NBETA)
C
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_OR',27,
     &                           WEV,BVec,BTotal,
     &                           IFile(10),Junk,.False.)
C
        WEV=WEV+WSTEP
C
      GO TO 20
C                                                  -----------------
C                                                  SCAN FROM S1 TO 0
C                                                  -----------------
   30 IF(WEV.LE.ZERO) GO TO 40
C
        ISTEP = ISTEP+1
        WAU   = WEV/AUTOEV
        WDELTA=-WSTEP*DFLOAT(ISTEP)
        WRITE(IOUT,2090) ISTEP,WDELTA
C
        CALL OR(WAU,BETAOR,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
        CALL CHGUNT(BVEC  ,3    ,TOESU)
        CALL CHGUNT(BETAOR,NBETA,TOESU)
C
        BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
        WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
        WRITE(IOUT,2080) (SHGSUF(J),BETAOR(J),J=1,NBETA)
C
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_ORRev',30,
     &                           WDelta,BVec,BTotal,
     &                           IFile(10),Junk,.False.)
C
        WEV=WEV+WSTEP
C
      GO TO 30
C
   40   ISTEP = ISTEP+1
        WAU   = ZERO
        WDELTA= CIEIG(1)*AUTOEV
        WRITE(IOUT,2100) ISTEP,WDELTA
C
        CALL OR(WAU,BETAOR,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
        CALL CHGUNT(BVEC  ,3    ,TOESU)
        CALL CHGUNT(BETAOR,NBETA,TOESU)
C
        BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
        WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
        WRITE(IOUT,2080) (SHGSUF(J),BETAOR(J),J=1,NBETA)
C
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_ORRev',30,
     &                           WDelta,BVec,BTotal,
     &                           IFile(10),Junk,.False.)
C
   50 IF(ISTEP.EQ.0) THEN
        WRITE(IOUT,9920)
        CALL MERROR
      END IF
C
      RETURN
      END
C
C@(#)=D1SHG(SUB)
      SUBROUTINE D1SHG(TGN,TNN,CIEIG,BETASH,BVEC,MCICSF,MCIINT,NCSFS)
C
C     GET BETA(I,J,K) FOR SHG.
C     Oct/92 ... A.M.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*3 SHGSUF,EOSUF
      Logical Anchor
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /BETASF/ SHGSUF(18),EOSUF(18)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PLANCK,SLIGHT,EVTOER,PCON(14)
      Common /Anchor/ Anchor
      DIMENSION TGN(MCICSF,*),TNN(MCIINT,*)
      DIMENSION CIEIG(*),BETASH(*),BVEC(*)
      DATA YAGNM/1.0640D+03/
      DATA ZERO/0.0D+00/,PT05/0.05D+00/,PT5/0.5D+00/
      DATA TENP5/1.0D+05/,TENP7/1.0D+07/
      DATA NBETA/18/
 2000 Format(1H ,38('-')/
     &       1H ,'Beta(i,j,k) for SHG ... Beta(-2w;w,w).'/
     &       1H ,38('-'))
 2010 Format(1H ,'Incident light : Nd yag laser (1064 nm,',F7.4,' eV) ',
     &       '...')
 2020 Format(1H ,2X,'X  =',F12.5,4X,'Y  =',F12.5,4X,'Z  =',F12.5,
     &       2X,'Total=',F12.5)
 2030 Format(1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,3(3X,A3,' =',F12.5)/
     &       1H ,1X,A3,' =',F12.5,1(3X,A3,' =',F12.5))
 2040 Format(1H ,32('=')/
     &       1H ,'Frequency dependent beta for SHG'/
     &       1H ,32('='))
 2050 Format(1H ,I4,'.  Static ...')
 2060 Format(1H ,I4,'.',F6.3,' eV = ',F8.2,' nm ...')
 2070 Format(1H ,2X,'X =',F10.3,3X,'Y =',F10.3,3X,'Z =',F10.3,
     &       2X,'Total=',F10.3)
 2080 Format(1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,4(2X,A3,'=',F10.3)/
     &       1H ,1X,A3,'=',F10.3,2(2X,A3,'=',F10.3))
 2090 Format(1H ,I4,'. E(S1)/2-E(Incident light) =',F6.3,' eV ...')
 2100 Format(1H ,I4,'. E(S1)/2-E(Incident light) =',F6.3,' eV (Static) '
     &       ,'...')
 9910 Format(1H ,56('X')/
     &       1H ,'TRANSITION ENERGY FOR S1 STATE .LT. ENERGY OF ',
     &           'YAG LASER.'/
     &       1H ,56('X'))
 9920 Format(1H ,29('X')/
     &       1H ,'TOO LITTLE TRANSITION ENERGY.'/
     &       1H ,29('X'))
C     ****************************************************************
      IPRSHG=IOPDIP(13)
C
      TOKYS =(EVTOER*AUTOEV*TENP5)/(PLANCK*SLIGHT)
      TOESU =TOANG**5/(EVTOER*SLIGHT)
C
```

```
      YAGAU=TENP7/(YAGNM*TOKYS)
      YAGEV=YAGAU*AUTOEV
      IF(CIEIG(1).LT.YAGAU) THEN
        WRITE(IOUT,9910)
        GO TO 10
      END IF
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010) YAGEV
C
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_SHG',28,
     &                         YagEV,BVec,BTotal,
     &                         IFile(10),Junk,.True.)
C
C     MCICSF, MCIINT ... FROM INCLUDE STATEMENT.
C
C     GET BETA FOR YAG LASER.
      CALL SHG(YAGAU,BETASH,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
C
C     THEN CONVERT UNITS OF BETA FROM (BOHR)**5/CHARGE(AU) TO
C     10**(-30) CM**5/STATCOULOMB.
      CALL CHGUNT(BVEC  ,3    ,TOESU)
      CALL CHGUNT(BETASH,NBETA,TOESU)
C
      BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
C     THEN PRINT.
      WRITE(IOUT,2020) (BVEC(I),I=1,3),BTOTAL
      WRITE(IOUT,2030) (SHGSUF(I),BETASH(I),I=1,NBETA)
C
      If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_SHG',28,
     &                         YagEV,BVec,BTotal,
     &                         IFile(10),Junk,.False.)
C
   10 IF(IPRSHG.EQ.0) RETURN
C
C     GET FREQUENCY DEPENDENT BETA.
      WRITE(IOUT,2040)
C
      ISTEP =0
      WEVMAX=CIEIG(1)*AUTOEV*PT5
C
      IF(IPRSHG.EQ.1) THEN
        WEV  = ZERO
        WSTEP= PT05
      ELSE
        WEV  = WEVMAX-PT05
        WSTEP=-PT05
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_SHGRev',31,
     &                           WDelta,BVec,BTotal,
     &                           IFile(10),Junk,.True.)
        GO TO 30
      END IF
C                                                  -------------------
C                                                  SCAN FROM 0 TO S1/2
C                                                  -------------------
   20 IF(WEV.GT.WEVMAX) GO TO 50
C
        ISTEP=ISTEP+1
        WAU  =WEV/AUTOEV
C
        IF(ISTEP.EQ.1) THEN
          WRITE(IOUT,2050) ISTEP
        ELSE
          WNM=TENP7/(WAU*TOKYS)
          WRITE(IOUT,2060) ISTEP,WEV,WNM
        END IF
C
        CALL SHG(WAU,BETASH,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
        CALL CHGUNT(BVEC  ,3    ,TOESU)
        CALL CHGUNT(BETASH,NBETA,TOESU)
C
        BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
        WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
        WRITE(IOUT,2080) (SHGSUF(J),BETASH(J),J=1,NBETA)
C
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_SHG',28,
     &                           WEV,BVec,BTotal,
     &                           IFile(10),Junk,.False.)
C
        WEV=WEV+WSTEP
C
      GO TO 20
C                                                  -------------------
C                                                  SCAN FROM S1/2 TO 0
C                                                  -------------------
   30 IF(WEV.LE.ZERO) GO TO 40
C
        ISTEP = ISTEP+1
        WAU   = WEV/AUTOEV
        WDELTA=-WSTEP*DFLOAT(ISTEP)
        WRITE(IOUT,2090) ISTEP,WDELTA
C
        CALL SHG(WAU,BETASH,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
        CALL CHGUNT(BVEC  ,3    ,TOESU)
        CALL CHGUNT(BETASH,NBETA,TOESU)
C
        BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
        WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
        WRITE(IOUT,2080) (SHGSUF(J),BETASH(J),J=1,NBETA)
C
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_SHGRev',31,
     &                           WDelta,BVec,BTotal,
     &                           IFile(10),Junk,.False.)
C
        WEV=WEV+WSTEP
C
      GO TO 30
C
   40   ISTEP = ISTEP+1
        WAU   = ZERO
        WDELTA= CIEIG(1)*AUTOEV*PT5
        WRITE(IOUT,2100) ISTEP,WDELTA
C
        CALL SHG(WAU,BETASH,BVEC,TGN,TNN,CIEIG,NCSFS,MCICSF,MCIINT)
        CALL CHGUNT(BVEC  ,3    ,TOESU)
        CALL CHGUNT(BETASH,NBETA,TOESU)
C
        BTOTAL=DSQRT( BVEC(1)*BVEC(1)+BVEC(2)*BVEC(2)+BVEC(3)*BVEC(3) )
C
        WRITE(IOUT,2070) (BVEC(J),J=1,3),BTOTAL
        WRITE(IOUT,2080) (SHGSUF(J),BETASH(J),J=1,NBETA)
C
        If (Anchor) Call AncBet ('HP_Hyperpolarizabilities_SHGRev',31,
     &                           WDelta,BVec,BTotal,
     &                           IFile(10),Junk,.False.)
C
   50 IF(ISTEP.EQ.0) THEN
        WRITE(IOUT,9920)
        CALL MERROR
      END IF
C
      RETURN
      END
C
C@(#)=DANG(SUB) ... MOPAC(DANG)
      SUBROUTINE DANG(A1,A2,B1,B2,RCOS,TWOPI)
C
C     ==========================================================
C     MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC
C     ==========================================================
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C*********************************************************************
C
C    DANG  DETERMINES THE ANGLE BETWEEN THE POINTS (A1,A2), (0,0),
C          AND (B1,B2).  THE RESULT IS PUT IN RCOS.
C
C*********************************************************************
C*MOS PI=2.0D0*DASIN(1.0D00)
      ZERO=1.0D-6
C*MOS IF( ABS(A1).LT.ZERO.AND. ABS(A2).LT.ZERO) GO TO 10
C*MOS IF( ABS(B1).LT.ZERO.AND. ABS(B2).LT.ZERO) GO TO 10
C*MOS ANORM=1.0D0/ SQRT(A1**2+A2**2)
C*MOS BNORM=1.0D0/ SQRT(B1**2+B2**2)
      IF( DABS(A1).LT.ZERO.AND. DABS(A2).LT.ZERO) GO TO 10
      IF( DABS(B1).LT.ZERO.AND. DABS(B2).LT.ZERO) GO TO 10
      ANORM=1.0D0/ DSQRT(A1**2+A2**2)
      BNORM=1.0D0/ DSQRT(B1**2+B2**2)
      A1=A1*ANORM
      A2=A2*ANORM
      B1=B1*BNORM
      B2=B2*BNORM
      SINTH=(A1*B2)-(A2*B1)
```

```
      COSTH=A1*B1+A2*B2
      IF(COSTH.GT.1.0D0) COSTH=1.0D0
      IF(COSTH.LT.-1.0D0) COSTH=-1.0D0
C*MOS RCOS= ACOS(COSTH)
C*MOS IF( ABS(RCOS).LT.4.0D-4) GO TO 10
C*MOS IF(SINTH.GT.0.D0) RCOS=6.2831853D0-RCOS
      RCOS= DACOS(COSTH)
      IF( DABS(RCOS).LT.4.0D-4) GO TO 10
      IF(SINTH.GT.0.D0) RCOS=TWOPI-RCOS
      RCOS=-RCOS
      RETURN
   10 RCOS=0.0D0
      RETURN
      END
C
C@(#)=DH2E(SUB)
      SUBROUTINE DH2E(PKLON,COORD,GAB,GAA,NATOMS)
C
C     CALCULATION OF ERIS USING DASGUPTA-HUZINAGA FORMULA
C     ******************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON(17)
      DIMENSION PKLON(*),COORD(3,*),GAB(*),GAA(*)
C     ******************************************************************
      CONST1=TOANG*AUTOEV
      CONST2=CONST1+CONST1
C
      DO 30 IAT=1,NATOMS
        GAI  =GAA(IAT)
        IATM1=IAT-1
C
        IF(IATM1.EQ.0) GO TO 20
C
        MX =IAT*(IAT-1)/2
        CIX=COORD(1,IAT)
        CIY=COORD(2,IAT)
        CIZ=COORD(3,IAT)
        PKI=PKLON(IAT)
        DO 10 JAT=1,IATM1
          MA =MX+JAT
          RX =COORD(1,JAT)-CIX
          RY =COORD(2,JAT)-CIY
          RZ =COORD(3,JAT)-CIZ
          RAU=DSQRT(RX*RX+RY*RY+RZ*RZ)
          RIJ=RAU*TOANG
          PKJ=PKLON(JAT)
          GAJ=GAA(JAT)
C
C     DASGUPTA-HUZINAGA FORMULA. (USING KLONDIKE PARAMETER)
          ALP    =CONST2/(GAI*DEXP(PKI*RAU)+GAJ*DEXP(PKJ*RAU))
          GAB(MA)=CONST1/(RIJ+ALP)
C
   10   CONTINUE
C
C     MOVE 1 CENTER ERIS INTO 'GAB'.
   20   MA=IAT*(IAT+1)/2
        GAB(MA)=GAI
   30 CONTINUE
C
      RETURN
      END
C
C@(#)=DIAG1(SUB)
      SUBROUTINE DIAG1(A,VEC,EIG,N,MDIM,SCRTCH,IOUT)
C
C     DIAGONALIZE REAL*8 SYMMETRIC MATRIX.
C
C     On input:
C      A      ... REAL*8 LOWER TRIANGLE MATRIX WITH LINEAR FORM.
C      N      ... SIZE OF MATRIX, A.
C      MDIM   ... MAXIMUM SIZE OF A (ADJUSTABLE DIMENSION FOR VEC).
C      SCRTCH ... SCRATCH ARRAY FOR DIAGONALIZATION.
C      IOUT   ... LOGICAL UNIT NUMBER FOR STANDARD OUTPUT.
C
C     On output:
C      VEC    ... EIGENVECTORS OF A.
C      EIG    ... EIGENVALUES OF A.
C
C     NOTE.
C      ONE OF 3 ROUTINES IS USED IN THIS ROUTINE.
C      1. DIAGD  ... GAUSSIAN 80 DIAGONALIZATION ROUTINE.
C      2. DSEIG2 ... Fujitsu SCIENTIFIC SUBROUTINE LIBRARY 2 (SSL2)
C                    DIAGONALIZATION ROUTINE (SCALAR ALGORITHM).
C      3. DVSEG2 ... Fujitsu SSL2 ROUTINE (VECTOR ALGORITHM).
C
C     CAUTION.
C      A IS DESTROYED IN DIAGONALIZATION ROUTINE.
C      This routine is machine dependent.
C
C      93/Feb A.M.
C     ******************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      Logical First
      DIMENSION A(*),VEC(MDIM,*),EIG(*),SCRTCH(*)
 2000 Format(1h ,'Matrix diagonalization ... ',
     &           'Using DiagD of Gaussian 80.')
 2010 Format(1h ,'Matrix diagonalization ... ',
     &           'Using SSL2 scalar algorithm.')
 2020 Format(1h ,'Matrix diagonalization ... ',
     &           'Using SSL2 vector algorithm.')
 9910 Format(1H ,47('X')/
     &       1H ,'DIAGONALIZATION FAILED. CONDITION CODE = ',I5,'.'/
     &       1H ,47('X'))
      Data First/.True./
C     ******************************************************************
      ICOND=0
C
C     GAUSSIAN 80.
c$$$#ifndef SSL2
c$$$#ifndef SSL2VP
      If (First) Write(IOut,2000)
      CALL DIAGD(A,VEC,EIG,N,SCRTCH(1),SCRTCH(MDIM+1),MDIM)
c$$$#endif
c$$$#endif
C
C     SSL2. SCALAR ALGORITHM.
c$$$#ifdef SSL2
c$$$      If (First) Write(IOut,2010)
c$$$      CALL DSEIG2(A,N,-N,          EIG,VEC,MDIM,SCRTCH,ICOND)
c$$$c$$$#endif
c$$$C
c$$$C     SSL2. VECTOR ALGORITHM.
c$$$c$$$#ifdef SSL2VP
c$$$      If (First) Write(IOut,2020)
c$$$      CALL DVSEG2(A,N,-N,-1.0D+00,EIG,VEC,MDIM,SCRTCH(1),
c$$$     &            SCRTCH(MDIM*15+1),ICOND)
c$$$c$$$#endif
C
      If (First) First=.False.
C
      IF(ICOND.NE.0) THEN
        WRITE(IOUT,9910) ICOND
        CALL MERROR
      END IF
C
      RETURN
      END
c$$$#ifndef SSL2
c$$$#ifndef SSL2VP
C
C@(#)=DIAGD(SUB) ... GAUSSIAN80(DIAGD)
      SUBROUTINE DIAGD(A,V,D,NBASIS,E,E2,MDIM)
C
C     REAL*8 DIAGONALIZATION ROUTINE FOR A HERMITIAN
C     (SYMMETRIC) MATRIX IN PACKED FORM (PACKED REAL FIRST)
C     A IS THE INPUT MATRIX (PACKED: REAL FIRST)
C     V IS THE EIGENVECTOR MATRIX, AGAIN REAL FIRST
C     D CONTAINS THE EIGENVALUES
C     NBASIS IS THE ACTUAL DIMENSION OF THE PROBLEM
C     E IS A SCRATCH ARRAY OF LENGTH MDIM
C     E2 IS A SCRATCH ARRAY OF THE LENGTH 2*MDIM
C     MDIM IS THE MAX. DIMENSION
C
C     CODED IN AUGUST BY ROLF SEEGER
C     SEE NUMERISCHE MATHEMATIK, 11, 181 (1968)
C     ******************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),V(MDIM,1),D(1),E(1),E2(1)
      DATA ZERO/0.0D0/,ONE/1.0D0/
C     ******************************************************************
      CALL EHOUSD(A,NBASIS,D,E,E2)
C
      DO 20 I=1,NBASIS
        DO 10 J=1,NBASIS
          V(J,I)=ZERO
   10   CONTINUE
```

```
      V(I,I)=ONE
   20 CONTINUE
C
      CALL EQRT2D(D,E,NBASIS,V,MDIM,IER)
      CALL EHOBKD(A,NBASIS,1,NBASIS,V,MDIM)
C
      RETURN
      END
C
C@(#)=DIHED(SUB) ... MOPAC(DIHED)
      SUBROUTINE DIHED(XYZ,I,J,K,L,ANGLE,PI)
C
C     ============================================================
C     MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC
C     ============================================================
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION XYZ(3,*)
      DATA TWO/2.0D+00/
C*****************************************************************
C
C      DIHED CALCULATES THE DIHEDRAL ANGLE BETWEEN ATOMS I, J, K,
C            AND L.  THE CARTESIAN COORDINATES OF THESE ATOMS
C            ARE IN ARRAY XYZ.
C
C     DIHED IS A MODIFIED VERSION OF A SUBROUTINE OF THE SAME NAME
C           WHICH WAS WRITTEN BY DR. W. THEIL IN 1973.
C
C*****************************************************************
C*MOS
      TWOPI=TWO*PI
C*MOS
      XI1=XYZ(1,I)-XYZ(1,K)
      XJ1=XYZ(1,J)-XYZ(1,K)
      XL1=XYZ(1,L)-XYZ(1,K)
      YI1=XYZ(2,I)-XYZ(2,K)
      YJ1=XYZ(2,J)-XYZ(2,K)
      YL1=XYZ(2,L)-XYZ(2,K)
      ZI1=XYZ(3,I)-XYZ(3,K)
      ZJ1=XYZ(3,J)-XYZ(3,K)
      ZL1=XYZ(3,L)-XYZ(3,K)
C      ROTATE AROUND Z AXIS TO PUT KJ ALONG Y AXIS
C*MOS DIST= SQRT(XJ1**2+YJ1**2+ZJ1**2)
      DIST= DSQRT(XJ1**2+YJ1**2+ZJ1**2)
      COSA=ZJ1/DIST
      IF(COSA.GT.1.0D0) COSA=1.0D0
      IF(COSA.LT.-1.0D0) COSA=-1.0D0
      DDD=1.0D0-COSA**2
C##   IF(DDD.LE.0.0) GO TO 10
      IF(DDD.LE.0.0D0) GO TO 10
C*MOS YXDIST=DIST* SQRT(DDD)
      YXDIST=DIST* DSQRT(DDD)
      IF(YXDIST.GT.1.0D-9) GO TO 20
   10 CONTINUE
      XI2=XI1
      XL2=XL1
      YI2=YI1
      YL2=YL1
      COSTH=COSA
      SINTH=0.D0
      GO TO 30
   20 COSPH=YJ1/YXDIST
      SINPH=XJ1/YXDIST
      XI2=XI1*COSPH-YI1*SINPH
      XJ2=XJ1*COSPH-YJ1*SINPH
      XL2=XL1*COSPH-YL1*SINPH
      YI2=XI1*SINPH+YI1*COSPH
      YJ2=XJ1*SINPH+YJ1*COSPH
      YL2=XL1*SINPH+YL1*COSPH
C      ROTATE KJ AROUND THE X AXIS SO KJ LIES ALONG THE Z AXIS
      COSTH=COSA
      SINTH=YJ2/DIST
   30 CONTINUE
      YI3=YI2*COSTH-ZI1*SINTH
      YL3=YL2*COSTH-ZL1*SINTH
      CALL DANG(XL2,YL3,XI2,YI3,ANGLE,TWOPI)
C##   IF (ANGLE .LT. 0.) ANGLE=6.2831853D0+ANGLE
C*MOS IF (ANGLE .LT. 0.D0) ANGLE=6.2831853D0+ANGLE
C*MOS IF (ANGLE .GE. 6.2831853D0 ) ANGLE=0.D0
      IF (ANGLE .LT. 0.D0) ANGLE=TWOPI+ANGLE
      IF (ANGLE .GE. TWOPI ) ANGLE=0.D0
      RETURN
      END
```

```
C
C@(#)=DIP1CN(SUB)
      SUBROUTINE DIP1CN
C
C     GET MOLECULAR INTEGRALS FOR DIPOLE MOMENTS.
C     TDX ... X-DIPOLE MATRIX.
C     TDY ... Y-DIPOLE MATRIX.
C     TDZ ... Z-DIPOLE MATRIX.
C
C     TDX = SUM OF C(IAO,IMO)*C(JAO,JMO)*<IAO/X/JAO> OVER IAO AND JAO.
C     WHERE,
C     C ... MOLECULAR ORBITAL COEFFICIENTS,
C     <IAO/X/JAO> ... X-DIPOLE INTEGRAL OVER BASIS FUNCTIONS IAO AND JAO.
C
C     1 CENTER DIPOLE INTEGRALS ARE EVALUATED ANALYTICALLY.
C     2 CENTER DIPOLE INTEGRALS ARE ZERO (USED BY MULLIKEN
C     APPROXIMATION).
C
C     ARRAY TDX IS PACKED FORM WITH RESPECT TO MOLECULAR ORBITALS IMO
C     AND JMO.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*1 CHR1(3)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /EXPNT / ZETA(MAXAT)
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                IBTOA(MAXBAS)
      COMMON /QNOINF/ NC(MAXIAN,2),LC(9),MC(9)
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                NOCS,NVCS,NCSFS
      COMMON /WORK01/ CIVEC(MCICS2),TDX(MCIPCK),TDY(MCIPCK),TDZ(MCIPCK),
     &                DIP(9,9),DUMMY1(1)
      COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),CIEIG(MCICSF),DUMMY2(1)
      DATA ZERO/0.0D+00/
      DATA CHR1/'X','Y','Z'/
 2000 Format(1H ,32('='))/
     &       1H ,'Molecular integrals for dipole-',A1/
     &       1H ,32('='))
C     ***************************************************************
      IPRD=IOPDIP(11)
      NMOS=NOCS+NVCS
      NPCK=NMOS*(NMOS+1)/2
C
      DO 10 I=1,NPCK
        TDX(I)=ZERO
        TDY(I)=ZERO
        TDZ(I)=ZERO
   10 CONTINUE
C
      DO 20 IAT=1,NATOMS
        RX  =COORD(1,IAT)
        RY  =COORD(2,IAT)
        RZ  =COORD(3,IAT)
        IA  =IAN(IAT)
        IL  =NLBAS(IAT)
        IU  =NUBAS(IAT)
        NI  =NIBAS(IAT)
        ZSP =ZETA(IAT)
        ZD  =ZETA(IAT)
        NCSP=NC(IA,1)
        NCD =NC(IA,2)
C
        CALL GETDX(RX,NI,ZSP,ZD,NCSP,NCD,DIP)
        CALL MOLDIP(DIP,C,LMO,IL,IU,NMOS,NBASIS,TDX)
        CALL GETDY(RY,NI,ZSP,ZD,NCSP,NCD,DIP)
        CALL MOLDIP(DIP,C,LMO,IL,IU,NMOS,NBASIS,TDY)
        CALL GETDZ(RZ,NI,ZSP,ZD,NCSP,NCD,DIP)
        CALL MOLDIP(DIP,C,LMO,IL,IU,NMOS,NBASIS,TDZ)
C
   20 CONTINUE
C
      IF(IPRD.EQ.1) THEN
        WRITE(IOUT,2000) CHR1(1)
        CALL MSMOUT(TDX,NMOS)
        WRITE(IOUT,2000) CHR1(2)
        CALL MSMOUT(TDY,NMOS)
        WRITE(IOUT,2000) CHR1(3)
        CALL MSMOUT(TDZ,NMOS)
```

```
      END IF
c========( Wed Oct  4 15:12:46 JST 1995 )======
c$$$          WRITE(IOUT,2000) CHR1(1)
c$$$          CALL MSMOUT(TDX,NMOS)
c$$$          WRITE(IOUT,2000) CHR1(2)
c$$$          CALL MSMOUT(TDY,NMOS)
c$$$          WRITE(IOUT,2000) CHR1(3)
c$$$          CALL MSMOUT(TDZ,NMOS)
c$$$          write(iout,*)' *** nmos, nocs, nvcs ',nmos, nocs, nvcs
c$$$        ii = 1
c$$$        if( ii .eq. 1 ) stop
c========( Wed Oct  4 15:12:46 JST 1995 )======
C
      RETURN
      END
C
C@(#)=DIPZDO(SUB)
      SUBROUTINE DIPZDO
C
C     GET MOLECULAR INTEGRALS FOR DIPOLE MOMENTS.
C     DIPOLE INTEGRALS ARE EVALUATED BY USING THE ZDO APPROXIMATION.
C     TDX ... X-DIPOLE MATRIX.
C     TDY ... Y-DIPOLE MATRIX.
C     TDZ ... Z-DIPOLE MATRIX.
C
C     TDX = SUM OF C(IAO,IMO)*C(IAO,JMO)*<IAO/X/IAO> OVER IAO.
C     WHERE,
C     C ... MOLECULAR ORBITAL COEFFICIENTS,
C     <IAO/X/IAO> ... X-DIPOLE INTEGRAL OF BASIS FUNCTIONS IAO (EQUAL
C                     TO X COMPONENT OF NUCLEAR COORDINATE OF THE ATOM
C                     TO WHICH IAO BELONGS).
C
C     ARRAY TDX IS PACKED FORM WITH RESPECT TO MOLECULAR ORBITALS IMO
C     AND JMO.
C
C     1 CENTER DIPOLE INTEGRALS ARE EVALUATED BY USING THE ZDO
C     APPROXIMATION IN THIS ROUTINE, I.E., <I/R/J>=DELTA(I,J)*<I/R/I>
C
C     CAUTION.
C      ONLY NET CHARGE CONTRIBUTION WILL BE CONSIDERED IN ESTIMATING
C      DIPOLE MOMENTS IF MOLECULAR INTEGRALS FOR DIPOLE MOMENTS ARE
C      CONSTRUCTED IN THIS ROUTINE.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*1 CHR1(3)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /MOL   / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                NAE,NBE,NE,NBASIS
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                IBTOA(MAXBAS)
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                NOCS,NVCS,NCSFS
      COMMON /WORK01/ CIVEC(MCICS2),TDX(MCIPCK),TDY(MCIPCK),TDZ(MCIPCK),
     &                DUMMY1(1)
      COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),CIEIG(MCICSF),DUMMY2(1)
      DATA ZERO/0.0D+00/
      DATA CHR1/'X','Y','Z'/
 2000 Format(1H ,38('=')/
     &       1H ,'Molecular integrals for dipole-',A1,' (ZDO)'/
     &       1H ,38('='))
C     ****************************************************************
      IPRD=IOPDIP(11)
      NMOS=NOCS+NVCS
      NPCK=NMOS*(NMOS+1)/2
C
      DO 10 I=1,NPCK
        TDX(I)=ZERO
        TDY(I)=ZERO
        TDZ(I)=ZERO
   10 CONTINUE
C
      DO 50 IAT=1,NATOMS
        RX=COORD(1,IAT)
        RY=COORD(2,IAT)
        RZ=COORD(3,IAT)
        IL=NLBAS(IAT)
        IU=NUBAS(IAT)
C
C     DIPOLE INTEGRALS ARE TRANSFORMED INTO MOLECULAR INTEGRALS.
        DO 40 IBAS=IL,IU
          N=0
          DO 30 I=1,NMOS
            IP=(LMO(I)-1)*NBASIS+IBAS
            DO 20 J=1,I
              JP=(LMO(J)-1)*NBASIS+IBAS
              N =N+1
              TDX(N)=TDX(N)+RX*C(IP)*C(JP)
              TDY(N)=TDY(N)+RY*C(IP)*C(JP)
              TDZ(N)=TDZ(N)+RZ*C(IP)*C(JP)
   20       CONTINUE
   30     CONTINUE
   40   CONTINUE
   50 CONTINUE
C
      IF(IPRD.EQ.1) THEN
        WRITE(IOUT,2000) CHR1(1)
        CALL MSMOUT(TDX,NMOS)
        WRITE(IOUT,2000) CHR1(2)
        CALL MSMOUT(TDY,NMOS)
        WRITE(IOUT,2000) CHR1(3)
        CALL MSMOUT(TDZ,NMOS)
      END IF
C
      RETURN
      END
C
C@(#)=DMOUT(SUB)
      SUBROUTINE DMOUT(DM,MDIM,IAN,NATOMS,MINCLM,MAXCLM)
C
C     PRINTING OF MO DENSITIES.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*2 ELMNT2,EL
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      COMMON /PTbl2 / ELMNT2(99)
      DIMENSION DM(MDIM,*),IAN(*)
 2000 Format(1H ,5X,11(8X,I3))
 2010 Format(1H ,I3,2X,A2,1X,11(1X,F10.6))
C     ****************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=6
      ELSE
        IW=11
      END IF
C
      ILOWER=MINCLM
      IUPPER=MINCLM+IW-1
C
   10 IF(IUPPER.GT.MAXCLM) THEN
        IUPPER=MAXCLM
      END IF
C
      WRITE(IOUT,2000) (I,I=ILOWER,IUPPER)
C
      DO 20 I=1,NATOMS
        IA=IAN(I)
        EL=ELMNT2(IA)
        WRITE(IOUT,2010) I,EL,(DM(I,J),J=ILOWER,IUPPER)
   20 CONTINUE
C
      IF(IUPPER.EQ.MAXCLM) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
      SUBROUTINE DUMMY
C
C     MAIN(MOS-F)
      ENTRY POLAR
      ENTRY HYPOL2
c
c     Mo1Int(MOS-F)
      Entry IntCI2
      RETURN
C
      END
      subroutine dump00
c=================================================================*
c                                        @@@     @@@              *
```

```
c      @@@@@   @    @  @    @  @@@@@   @   @   @   @                    *
c      @    @  @    @  @@  @@  @    @  @     @ @     @                    *
c      @    @  @    @  @ @@ @  @    @  @     @ @     @                    *
c      @    @  @    @  @    @  @@@@@  @     @ @     @                    *
c      @    @  @    @  @    @  @        @   @   @   @                    *
c      @@@@@    @@@@   @    @  @         @@@     @@@                     *
c=====================================================================*
c$$$c
c$$$c-------------   dump sdci eigen vectors
c$$$c
c$$$      implicit real*8 (a-h,o-z)
c$$$      include 'MSSIZE'
c$$$      include 'energy.h'
c$$$      include 'sdci00.h'
c$$$      include 'sdci01.h'
c$$$      include 'sdci02.h'
c$$$      include 'window.h'
c$$$      include 'work02.h'
c$$$c
c$$$      common /io    / in,iout,ifile(10)
c$$$      common /lvci  / locc(mcicsf),lvac(mcicsf),
c$$$     &                mocc(mcicsf),mvac(mcicsf)
c$$$      common /work01/ ovov(mciint),oovv(mciint),civec(mcicsf,mcicsf),
c$$$     &                dummy(1)
c$$$c
c$$$      do 100 nomg = momega(1)+1, momega(2)
c$$$        i = lci1( nomg )
c$$$        k = lci3( nomg ) + nocs
c$$$        n = nomg - momega( 1 )
c$$$        locc( n ) = lmo( i )
c$$$        lvac( n ) = lmo( k )
c$$$  100 continue
c$$$c
c$$$      nscibs = momega( 2 ) - momega( 1 )
c$$$      if( isdci0 .eq. 1 ) then
c$$$        nend = ncibas
c$$$      else
c$$$        nend = momega( 3 )
c$$$      endif
c$$$      write( iout, 1000 )
c$$$      call civo00( civec,  cieig,  locc,   lvac,   mcicsf, nscibs,
c$$$     $             nend )
c$$$      write( iout, 1100 )
c$$$      call civo01( civec,  cieig,  locc,   lvac,   mcicsf, nscibs,
c$$$     $             nend )
c$$$c
c$$$c
c$$$ 1000 format(1h ,//
c$$$     $       1h ,37('=-')/
c$$$     &       1h ,'eigenvectors of ci hamiltonian matrix'/
c$$$     $       1h ,'(contribution from Hartree-Fock and',
c$$$     $           ' single excitaion states)'/
c$$$     &       1h ,37('=-'))
c$$$ 1100 format(1h ,//
c$$$     $       1h ,37('=-')/
c$$$     &       1h ,'eigenvectors of ci hamiltonian matrix'/
c$$$     $       1h ,'(contribution from double excitaion state type I)'/
c$$$     &       1h ,37('=-'))
c$$$c
c$$$c
      return
      end
C
C@(#)=EHOBKD(SUB) ... GAUSSIAN80(EHOBKD)
      SUBROUTINE EHOBKD  (A,N,M1,M2,Z,IZ)
C
C     ====================================================
C     GAUSSIAN80 GAUSSIAN80 GAUSSIAN80 GAUSSIAN80 GAUSSIAN80
C     ====================================================
      IMPLICIT REAL*8 (A-H,O-Z)
C
C   FUNCTION            - PERFORM A BACK TRANSFORMATION TO FORM THE
C                           EIGENVECTORS OF THE ORIGINAL SYMMETRIC
C                           MATRIX FROM THE EIGENVECTORS OF THE
C                           TRIDIAGONAL MATRIX.
C   PARAMETERS   A      - THE ARRAY CONTAINS THE DETAILS OF THE HOUSE
C                           HOLDER REDUCTION OF THE ORIGINAL MATRIX A AS
C                           GENERATED BY SUBROUTINE 'EHOUSD'.
C                N      - ORDER OF THE REAL SYMMETRIC MATRIX.
C                M1     - M1 AND M2 ARE TWO INPUT SCALARS SUCH THAT
C                           EIGENVECTORS M1 TO M2 OF THE TRIDIAGONAL
C                           MATRIX A HAVE BEEN FOUND AND NORMALIZED
C                           ACCORDING TO THE EUCLIDEAN NORM.
C                M2     - SEE ABOVE - M1
C                Z      - A TWO DIMENSIONAL ARRAY OF SIZE N X (M2-M1+1)
C                           WHICH CONTAINS EIGENVECTORS M1 TO M2 OF
C                           TRIDIAGONAL MATRIX T, NORMALIZED ACCORDING
C                           TO EUCLIDEAN NORM. INPUT Z CAN BE PRODUCED
C                           BY SUBROUTINE EQRT2D, THE RESULTANT
C                           MATRIX OVERWRITES THE INPUT Z.
C                IZ     - ROW DIMENSION OF Z IN CALLING PROGRAM.
C      IMPLEMENTED BY ROLF SEEGER
C      NUMER. MATH., 11, 181 (1968)
C
C
      DIMENSION          A(1),Z(IZ,1)
      DATA ZERO/0.D0/
      IF (N.EQ.1) GO TO 30
      DO 25 I=2,N
         L = I-1
         IA = I*L/2
         H = A(IA+I)
         IF(H) 1,25,1
C                                 DERIVES EIGENVECTORS M1 TO M2 OF
C                                 THE ORIGINAL MATRIX FROM EIGENVECTORS
C                                 M1 TO M2 OF THE SYMMETRIC
C                                 TRIDIAGONAL MATRIX
    1    DO 20 J = M1,M2
            S=ZERO
            DO 10 K = 1,L
               S = S+A(IA+K)*Z(K,J)
   10       CONTINUE
            S = S/H
            DO 15 K=1,L
               Z(K,J) = Z(K,J)-S*A(IA+K)
   15       CONTINUE
   20    CONTINUE
   25 CONTINUE
   30 RETURN
      END
C
C@(#)=EHOUSD(SUB) ... GAUSSIAN80(EHOUSD)
      SUBROUTINE EHOUSD (A,N,D,E,E2)
C
C     ====================================================
C     GAUSSIAN80 GAUSSIAN80 GAUSSIAN80 GAUSSIAN80 GAUSSIAN80
C     ====================================================
C1EHOUSD
      IMPLICIT REAL*8 (A-H,O-Z)
C
C   FUNCTION            - REDUCE A SYMMETRIC MATRIX A TO SYMMETRIC
C                           TRIDIAGONAL FORM USING HOUSEHOLDER'S
C                           REDUCTION.
C   PARAMETERS   A      - THE GIVEN N * N, REAL SYMMETRIC MATRIX A,
C                           WHERE A IS STORED IN SYMMETRIC STORAGE MODE.
C                           THE INPUT A IS REPLACED BY
C                           THE DETAILS OF THE HOUSEHOLDER
C                           REDUCTION OF A.
C                N      - ORDER OF A AND THE LENGTH OF D,E, AND E2
C                D      - THE OUTPUT ARRAY OF LENGTH N, GIVING THE
C                           DIAGONAL ELEMENTS OF THE TRIDIAGONAL MATRIX.
C                E      - THE OUTPUT ARRAY OF LENGTH N, GIVING THE SUB-
C                           DIAGONAL IN THE LAST (N-1) ELEMENTS, E(1) IS
C                           SET TO EQUAL TO 0.
C                E2     - THE ARRAY CONTAINS E(I)**2.
C      IMPLEMENTED BY ROLF SEEGER
C      NUMER. MATH., 11, 181 (1968)
C
C
C2EHOUSD
      COMMON /IO    / IN,IOUT,IFILE(10)
      DIMENSION          A(1),D(1),E(1),E2(1)
C=    DIMENSION ITOL(2)
C=    EQUIVALENCE (TOL,ITOL(1))
C=    DATA ITOL/'00001B00'X,0/
      DATA ZERO/.0D0/
      EPS=2.**(-37)
C*MOS ETA=2.**(-218)
      ETA=2.0D+00**(-218)
      TOL=ETA/EPS
      EPS1=EPS+1
      IF (EPS1.EQ.EPS) WRITE (IOUT,500) EPS,EPS1
500   Format(' EHOUSD-- EPS IS TO SMALL: EPS,EPS+1= ',2G15.5)
C                                 TOL IS A MACHINE DEPENDENT CONSTANT,
C                                 TOL = ETA/EPS, WHERE ETA IS THE
C                                 SMALLEST POSITIVE NUMBER REPRESENT-
```

```
C                                 ABLE IN THE COMPUTER AND EPS IS THE
C                                 SMALLEST POSITIVE NUMBER FOR WHICH
C                                 1+EPS.NE.1.
      NP1=N+1
      DO 45 II=1,N
         I=NP1-II
         L=I-1
         IZ=I*L/2
         H=ZERO
         IF (L.LE.0) GO TO 11
         DO 10 K=1,L
            F=A(IZ+K)
            D(K)=F
            H=H+F*F
   10    CONTINUE
C                                 IF H IS TOO SMALL FOR ORTHOGONALITY
C                                 TO BE GUARANTEED, THE TRANSFORMATION
C                                 IS SKIPPED
   11    IF (H.GT.TOL) GO TO 15
         E(I)=ZERO
         E2(I)=ZERO
         H=ZERO
         GO TO 40
   15    E2(I)=H
         E(I)=DSQRT(H)
         IF (F.GE.0.) E(I)=-E(I)
         G=E(I)
         H=H-F*G
         D(L)=F-G
         A(IZ+L)=D(L)
         F=ZERO
         IF (L.LE.0) GO TO 26
         DO 25 J=1,L
            G=ZERO
            JK=J*(J-1)/2+1
C                                 FORM ELEMENT OF A X U
            DO 20 K=1,L
               G=G+A(JK)*D(K)
               JK = JK+1
               IF (K.GE.J) JK=JK+K-1
   20       CONTINUE
C                                 FORM ELEMENT OF P
            G = G/H
            E(J) = G
            F=F+G*D(J)
   25    CONTINUE
C                                 FORM K
   26    HH=F/(H+H)
         JK = 0
C                                 FORM REDUCED A
         IF (L.LE.0) GO TO 40
         DO 35 J=1,L
            F = D(J)
            E(J) = E(J)-HH*F
            G = E(J)
            DO 30 K=1,J
               JK=JK+1
               A(JK)=A(JK)-F*E(K)-G*D(K)
   30       CONTINUE
   35    CONTINUE
   40    D(I) = A(IZ+I)
         A(IZ+I)=H
   45 CONTINUE
      RETURN
      END
C
C@(#)=EIGOUT(SUB)
      SUBROUTINE EIGOUT(VEC,EIG,MDIM,IAN,IBTOA,NBASIS,MINCLM,MAXCLM)
C
C     OUTPUT MODULE FOR STO BASED EIGENVALUES AND EIGENVECTORS.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*4 ORBTYP,ORB
      CHARACTER*2 ELMNT,EL
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      COMMON /PTbl2 / ELMNT(99)
      COMMON /ORBTYP/ ORBTYP(9)
      DIMENSION VEC(MDIM,*),EIG(*),IAN(*),IBTOA(*)
 2000 Format(1H ,17X,10(7X,I4))
 2010 Format(1H ,'  Eigenvalues ===>',2X,10(1X,F10.6))
 2020 Format(1H ,I4,1X,I3,2X,A2,2X,A4,2X,10(1X,F10.6))
 2030 Format(1H ,I4,10X,          A4,2X,10(1X,F10.6))
C     ***************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=5
      ELSE
        IW=10
      END IF
C
      ILOWER=MINCLM
      IUPPER=MINCLM+IW-1
C
   10 IF(IUPPER.GT.MAXCLM) THEN
        IUPPER=MAXCLM
      END IF
C
      WRITE(IOUT,2000) (I,I=ILOWER,IUPPER)
      WRITE(IOUT,2010) (EIG(I),I=ILOWER,IUPPER)
C
      IATOLD=0
C
      DO 20 I=1,NBASIS
        IAT =IBTOA(I)
        IF(IAT.NE.IATOLD) THEN
          IATOLD=IAT
          IORB  =1
          EL    =ELMNT(IAN(IAT))
          ORB   =ORBTYP(IORB)
          WRITE(IOUT,2020) I,IAT,EL,ORB,(VEC(I,J),J=ILOWER,IUPPER)
        ELSE
          IORB=IORB+1
          ORB =ORBTYP(IORB)
          WRITE(IOUT,2030) I,ORB,(VEC(I,J),J=ILOWER,IUPPER)
        END IF
C
   20 CONTINUE
C
      IF(IUPPER.EQ.MAXCLM) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
      subroutine elmntf
c=====================================================================*
c                                                                     *
c      @@@@@@  @       @    @  @    @   @@@@@  @@@@@@                *
c      @       @       @@  @@  @@   @     @    @                     *
c      @@@@@   @       @ @@ @  @ @  @     @    @@@@@                 *
c      @       @       @    @  @  @ @     @    @                     *
c      @       @       @    @  @   @@     @    @                     *
c      @@@@@@  @@@@@@  @    @  @    @     @    @                     *
c=====================================================================*
c
c     get fij
c
c     ***************************************************************
      implicit real*8 (a-h,o-z)
c
      include 'MSSIZE'
      include 'basinf.h'
      include 'mol.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
c
      common /io    / in,iout,ifile(10)
      common/ gamma  / gab( mapck ), gaa( maxat )
      data zero/ 0.0d+0 /
c     ***************************************************************
c========( Mon Sep 25 13:42:56 JST 1995 )======
c$$$ 2040 Format(1H ,16('=')/
c$$$     &       1H ,'Fock matrix elements'/
c$$$     &       1H ,16('='))
c$$$      write(iout,2040)
c$$$      call ssmout( fockmt, ian,    ibtoa,  n1bas,  nbasis )
c$$$      call fckout( fockmt, nbasis )
c$$$      ii = 1
c$$$      if( ii .eq. 1 ) stop
c========( Mon Sep 25 13:42:57 JST 1995 )======
c
```

```
c$$$      do 1 nu = 1, numax
c$$$        i  = ici1( nu )
c$$$        j  = ici2( nu )
c$$$        ix = lmo( i )
c$$$        jx = lmo( j )
c$$$        ii = nbasis*( ix - 1 )
c$$$        jj = nbasis*( jx - 1 )
c$$$        fock00 = zero
c$$$        do 2 ir0 = 1, nbasis
c$$$          am = zero
c$$$          do 3 is0 = 1, nbasis
c$$$            irsx = max0( ir0, is0 )
c$$$            irsy = min0( ir0, is0 )
c$$$            irs  = irsx*( irsx - 1 )/2 + irsy
c$$$            am   = am + c( jj + is0 )*fockmt( irs )
c$$$    3     continue
c$$$          fock00 = fock00 + c( ii + ir0 )*am
c$$$    2   continue
c$$$        fckmo1( nu ) = fock00
c$$$    1 continue
c$$$c=======( Mon Sep 25 13:42:57 JST 1995 )======
c$$$ 2040 Format(1H ,16('=')/
c$$$     &        1H ,'Fock matrix elements'/
c$$$     &        1H ,16('='))
c$$$      write(iout,2040)
c$$$c$$$      call fckout( fckmo1, nsdci )
c$$$      nst = nae - nocs + 1
c$$$      ned = nae + nvcs
c$$$      write(iout,'(10f11.6)')( eig(i), i = nst, ned )
c$$$      ii = 1
c$$$      if( ii .eq. 1 ) stop
c$$$c=======( Mon Sep 25 13:42:57 JST 1995 )======
c$$$c
c$$$c
      return
      end
C
C@(#)=ENGOUT(SUB)
      SUBROUTINE ENGOUT(EIG,AUTOEV,IHOMO,ILUMO,NBASIS)
C
C     PRINTING OF ORBITAL ENERGIES.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      DIMENSION EIG(*)
 2000 Format(1H ,' Occupied MOs ...')
 2010 Format(1H ,I4,1X,F9.4,7(2X,I4,1X,F9.4))
 2020 Format(1H ,' Virtual MOs ...')
C     ****************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=5
      ELSE
        IW=8
      END IF
C
      WRITE(IOUT,2000)
C
      ILOWER=1
      IUPPER=IW
C
   10 IF(IUPPER.GT.IHOMO) THEN
        IUPPER=IHOMO
      END IF
C
      WRITE(IOUT,2010) (I,EIG(I)*AUTOEV,I=ILOWER,IUPPER)
C
      IF(IUPPER.EQ.IHOMO) GO TO 20
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
   20 WRITE(IOUT,2020)
C
      ILOWER=ILUMO
      IUPPER=ILUMO+IW-1
C
   30 IF(IUPPER.GT.NBASIS) THEN
        IUPPER=NBASIS
      END IF
C
      WRITE(IOUT,2010) (I,EIG(I)*AUTOEV,I=ILOWER,IUPPER)
C
      IF(IUPPER.EQ.NBASIS) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 30
C
      END
C
C@(#)=EOPE(SUB)
      SUBROUTINE EOPE(W,BETAEO,BVEC,TGN,TNN,CIEIG,NCSFS,MDIM1,MDIM2)
C
C     CALCULATE 1-ST HYPERPOLARIZABILITIES FOR ELECTRO-OPTIC POCKELS
C     EFFECT ... BETA(-W;0,W).
C     UNIT : ATOMIC UNIT.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION BETAEO(*),BVEC(*),TGN(MDIM1,*),TNN(MDIM2,*),CIEIG(*)
      DATA ZERO/0.0D+00/,TWO/2.0D+00/,THREE/3.0D+00/
C     ****************************************************************
      N=0
C
      DO 30 I=1,3
        DO 20 J=1,3
          DO 10 K=I,3
            N=N+1
            BETAEO(N)=BETXYZ(ZERO,W,I,J,K,TGN,TNN,CIEIG,NCSFS,
     &                       MDIM1,MDIM2)
   10     CONTINUE
   20   CONTINUE
   30 CONTINUE
C
C     THEN GET BETA-VEC.
C     DO 50 I=1,3
C       BVEC(I)=ZERO
C       DO 40 J=1,3
C         MA1=(I-1)*9+(J-1)*3+J
C         MA2=(J-1)*9+(J-1)*3+I
C         MA3=(J-1)*9+(I-1)*3+J
C         BVEC(I)=BVEC(I)+BETAEO(MA1)+BETAEO(MA2)+BETAEO(MA3)
C  40   CONTINUE
C       BVEC(I)=BVEC(I)/THREE
C  50 CONTINUE
C
      BVEC(1)=(  BETAEO( 1)*THREE
     &         + BETAEO( 5)*TWO+BETAEO(10)
     &         + BETAEO( 9)*TWO+BETAEO(16) ) / THREE
      BVEC(2)=(  BETAEO( 2)*TWO+BETAEO( 4)
     &         + BETAEO(12)*THREE
     &         + BETAEO(15)*TWO+BETAEO(17) ) / THREE
      BVEC(3)=(  BETAEO( 3)*TWO+BETAEO( 7)
     &         + BETAEO(13)*TWO+BETAEO(14)
     &         + BETAEO(18)*THREE          ) / THREE
C
      RETURN
      END
C
C@(#)=EPRPTY(SUB)
      SUBROUTINE EPRPTY
C
C     PRINTING OF MOLECULAR PROPERTIES FOR EXCITED STATES.
C
C     NOTE.
C      THIS ROUTINE PRINT AS FOLLOWS.
C      1. VERTICAL EXCITATION ENERGY ... UNIT IN ELECTRON VOLT,
C                                                CM-1,
C                                                NM.
C      2. OSCILLATOR STRENGTH.
C      3. MAIN CSFS OF THE EXCITED STATE (SINGLY EXCITED CSFS WHICH HAVE
C         LARGE CI COEFFICIENTS).
c      4. Dipole Moments for the excited states.
c      5. Total atomic charges for the excited states (optional).
c
c     Oct/92 ... A.M.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*7 CHR7(2)
      Integer*4 fmt
      Logical Anchor
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
```

```
      Parameter(MDim2 = MCIMOs * MaxBas)
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                 IBTOA(MAXBAS)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PLANCK,SLIGHT,EVTOER,PCON(14)
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                 NOCS,NVCS,NCSFS
      COMMON /LVCI   / LOCC(MCICSF),LVAC(MCICSF),
     &                 MOCC(MCICSF),MVAC(MCICSF)
      COMMON /MULTCI/ MULTCI
      COMMON /DGRND / DXPT,DYPT,DZPT,DXAP,DYAP,DZAP,DX,DY,DZ
      COMMON /WORK01/ TGNX(MCICSF),TGNY(MCICSF),TGNZ(MCICSF),
     &                 TNNX(MCIINT),TNNY(MCIINT),TNNZ(MCIINT),
     &                 TMOMSQ(MCICSF),VEC(MCICSF),OSC(MCICSF),
     &                 LST(MCICSF),IFROM(10),ITO(10),IPCT(10),
     &                 IOcc(MCIMOs,40),IDUMMY(1)
      COMMON /WORK02/ CIEIG(MCICSF),CIVEC(MCICS2),
     &                 CI2Occ(MCIOcc),CI2Vac(MCIVac),EChrg(MaxBas),
     &                 AtChgE(MaxAt),DUMMY(1)
      Common /AOChrg/ GChrg(MaxBas), AtChgG(MaxAt), CMO(MDim2)
      Common /Anchor/ Anchor
      DATA MCV/5/,MPCT/80/
      DATA ZERO/0.0D+00/,TWO/2.0D+00/,THREE/3.0D+00/
      DATA TENP2/1.0D+02/,TENP5/1.0D+05/,TENP7/1.0D+07/
      DATA CHR7/'singlet','triplet'/
      Data NF/6/
 2000 Format(1H ,'<<<<<< Enter EPrpty >>>>>>')
 2010 Format(1H ,'Electron transition spectra (G->E) for ',A7,' state.',
     &          ' NStates =',I4)
 2020 Format(1H ,79('-'))
 2030 Format(1H , 1X,'State',8X,'Transition energy',9X,'Oscillator',11X,
     &              'Main CSFs'/
     &       1H ,11X,'eV',8X,'cm-1',7X,'nm',7X,'strength',6X,'MO',10X,
     &              'CI coef.')
 2040 Format(1H ,I4,2X,F10.5,1X,F10.2,1X,F9.3,3X,F9.6,2X,I4,'->',I4,2X,
     &          F8.5,1X,'(',I3,'%)')
 2050 Format(1H ,51X,I4,'->',I4,2X,F8.5,1X,'(',I3,'%)')
 2060 Format(1h ,34('='))
     &       1h ,'Atomic charges for excited states.'/
     &       1h ,34('='))
 2070 Format(1h , 8('-')/
     &       1h ,'State', i2,'.'/
     &       1h , 8('-'))
 2080 Format(1h , 9('-')/
     &       1h ,'State', i3,'.'/
     &       1h , 9('-'))
 2090 Format(1h ,10('-')/
     &       1h ,'State', i4,'.'/
     &       1h ,10('-'))
 2100 Format(1h ,11('-')/
     &       1h ,'State', i5,'.'/
     &       1h ,11('-'))
 2110 Format(1h ,12('-')/
     &       1h ,'State', i6,'.'/
     &       1h ,12('-'))
 2120 Format(1h ,'Difference atomic charges ...')
 2130 Format(1h ,'Total atomic charges ...')
 2140 Format(1H ,'Dipole moments (Unit : Debye)')
 2150 Format(1H ,'WARNING : DIPOLE MOMENTS ARE ESTIMATED BY POINT ',
     &          'CHARGES (ZDO).')
 2160 Format(1H ,2X,'State',23X,'Dipole',22X,'Delta D',3X,'Oscillator'/
     &       1H ,16X,'x',10X,'y',10X,'z',8X,'Total',7X,'Total',5X,
     &          'strength')
 2170 Format(1H ,1X,'Ground',2X,4(2X,F9.4))
 2180 Format(1H ,I5,4X,         4(2X,F9.4), 3X,F9.4,3X,F9.6)
C     ****************************************************************
      IZDO  =IOPDIP( 1)
      NSTATE=IOPDIP(16)
      NEChrg=IopDip(17)
C
      TOKYS =(EVTOER*AUTOEV*TENP5)/(PLANCK*SLIGHT)
      AUTODB=SLIGHT*TOANG*EVTOER
      TWOTHR=TWO/THREE
C
      I1OR3 =MULTCI/2+1
      IF(NSTATE.EQ.-1) THEN
        NSTATE=NCSFS
      ELSE
        NSTATE=MIN0(NSTATE,NCSFS)
      END IF
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010) CHR7(I1OR3),NSTATE
      WRITE(IOUT,2020)
      WRITE(IOUT,2030)
      WRITE(IOUT,2020)
C
      MVECS=MIN0(NCSFS,MCV)
C
      DO 60 ISTATE=1,NSTATE
C
C     MOVE I-TH CI VECTORS INTO ARRAY 'VEC'.
        IS=NCSFS*(ISTATE-1)
        DO 10 I=1,NCSFS
          VEC(I)=CIVEC(IS+I)
          LST(I)=I
   10   CONTINUE
C
C     FIND I-TH LARGEST ABSOLUTE VALUE OF CI VECTORS.
        ISUM=0
        DO 30 I=1,MVECS
          VMAX=VEC(I)
          IMAX=LST(I)
          IP1 =I+1
          DO 20 J=IP1,NCSFS
            IF(DABS(VEC(J)).GT.DABS(VMAX)) THEN
              VI    =VMAX
              VMAX  =VEC(J)
              VEC(J)=VI
              ILST  =IMAX
              IMAX  =LST(J)
              LST(J)=ILST
            END IF
   20     CONTINUE
          VEC(I)  =VMAX
          IFROM(I)=LOCC(IMAX)
          ITO(I)  =LVAC(IMAX)
          IPCT(I) =IDNINT(VMAX*VMAX*TENP2)
          ISUM    =ISUM+IPCT(I)
          NVECS   =I
          IF(ISUM.GT.MPCT) GO TO 40
   30   CONTINUE
C
   40   TEAU=CIEIG(ISTATE)
        TEEV=TEAU*AUTOEV
        TEKY=TEAU*TOKYS
        TENM=TENP7/TEKY
C
C     GET OSCILLATOR STRENGTH.
        IF(MULTCI.EQ.1) THEN
          FIJ=TMOMSQ(ISTATE)*TEAU*TWOTHR
          OSC(ISTATE)=FIJ
        ELSE
          FIJ=ZERO
        END IF
C
C     THEN PRINT.
        WRITE(IOUT,2040) ISTATE,TEEV,TEKY,TENM,FIJ,IFROM(1),ITO(1),
     &                   VEC(1),IPCT(1)
        IF(NVECS.GT.1) THEN
          DO 50 I=2,NVECS
            WRITE(IOUT,2050) IFROM(I),ITO(I),VEC(I),IPCT(I)
   50     CONTINUE
        END IF
C
   60 CONTINUE
C
      WRITE(IOUT,2020)
c
c     Computation of total atomic charges for SCI excited states.
      If (NEChrg .NE. 0) Then
c
        Write(iout,2060)
        If (NEChrg .Eq. -1) then
          NEChrg = NCSFs
        Else
          NEChrg = Min0(NEChrg, NCSFs)
        EndIf
c
        Do 100 i = 1, NEChrg
c
          If     (i .LT.    10) then
            Assign 2070 to fmt
```

```
          ElseIf (i .LT.   100) then
            Assign 2080 to fmt
          ElseIf (i .LT.  1000) then
            Assign 2090 to fmt
          ElseIf (i .LT. 10000) then
            Assign 2100 to fmt
          Else
            Assign 2110 to fmt
          EndIf
c
          ma =NCSFs*(i - 1) + 1
          Call CIChrg (CIVec(ma), CMO, CI2Occ, CI2Vac, EChrg, NBasis,
     &                 NOCS, NVCS)
c
          If (Anchor) Call AncOut ('CI_AO_Densities',15,EChrg,1,1,
     &                            NBasis,NF,IFile(10),0,Junk)
c
          Do 80 iAt = 1, NAtoms
            IL = NLBas(iAt)
            IU = NUBas(iAt)
            Chg = Zero
            Do 70 j = IL, IU
              Chg = Chg + EChrg(j)
   70       Continue
            AtChgE(iAt) = - Chg
   80     Continue
          Write(iout,fmt) i
          Write(iout,2120)
          Call ACOut(AtChgE, IAN, NAtoms)
          Do 90 iAt = 1, NAtoms
            AtChgE(iAt) = AtChgE(iAt) + AtChgG(iAt)
   90     Continue
          Write(iout, 2130)
          Call ACOut(AtChgE, IAN, NAtoms)
c
          If (Anchor) Call AncOut ('CI_Charges',10,AtChgE,1,1,NAtoms,
     &                            NF,IFile(10),0,Junk)
c
  100   Continue
      End If
c
      IF(MULTCI.NE.1) GO TO 120
c
c     PRINTING OF DIPOLE MOMENTS FOR SINGLET STATE.
      WRITE(IOUT,2140)
      IF(IZDO.EQ.1) THEN
        WRITE(IOUT,2150)
      END IF
      WRITE(IOUT,2020)
      WRITE(IOUT,2160)
      WRITE(IOUT,2020)
      XG=-AUTODB*DX
      YG=-AUTODB*DY
      ZG=-AUTODB*DZ
      TG= DSQRT(XG*XG+YG*YG+ZG*ZG)
      WRITE(IOUT,2170) XG,YG,ZG,TG
      DO 110 ISTATE=1,NSTATE
        MA = ISTATE*(ISTATE+1)/2
        XE =-AUTODB*TNNX(MA)
        YE =-AUTODB*TNNY(MA)
        ZE =-AUTODB*TNNZ(MA)
        TE = DSQRT(XE*XE+YE*YE+ZE*ZE)
        DD = DSQRT((XG-XE)*(XG-XE)+(YG-YE)*(YG-YE)+(ZG-ZE)*(ZG-ZE))
        FIJ= OSC(ISTATE)
        WRITE(IOUT,2180) ISTATE,XE,YE,ZE,TE,DD,FIJ
  110 CONTINUE
      WRITE(IOUT,2020)
c
  120 If (Anchor) Call AncEP2 (NCSFs,NState,NOCS,NVCS,NOCS+NVCS,
     &                         IFile(10),MultCI,MCIMOS,
     &                         LMO,IOcc,
     &                         CIEig,CIVec,Osc,
     &                         TGNX,TGNY,TGNZ,TMomSq,
     &                         TNNX,TNNY,TNNZ,
     &                         AUToDb)
C
C     ALL DONE. RETURN TO CALLER.
      CALL CLOCKM('EPrpty.')
      RETURN
      END
C
C@(#)=EPSLN(FUN)
      DOUBLE PRECISION FUNCTION EPSLN()
      IMPLICIT REAL*8 (A-H,O-Z)
C
C     GET MACHINE EPSILON ... 1 + E > 1.
      DATA PT5/0.5D+00/,ONE/1.0D+00/
C     ******************************************************************
      EPS=ONE
C
   10 EPS=EPS*PT5
C
      IF( (ONE+EPS) .GT. ONE ) GO TO 10
C
      EPSLN=EPS
C
      RETURN
      END
C
C@(#)=EQRT2D(SUB) ... GAUSSIAN80(EQRT2D)
      SUBROUTINE EQRT2D (D,E,N,Z,IZ,IER)
C
C     ====================================================
C     GAUSSIAN80 GAUSSIAN80 GAUSSIAN80 GAUSSIAN80 GAUSSIAN80
C     ====================================================
      IMPLICIT REAL*8 (A-H,O-Z)
C
C   FUNCTION              - FIND THE EIGENVALUES AND EIGENVECTORS OF A
C                             TRIDIAGONAL MATRIX, T, USING QL
C                             TRANSFORMATIONS.
C   PARAMETERS   D        - THE ARRAY OF LENGTH N CONTAINS THE DIAGONAL
C                             ELEMENTS OF TRIDIAGONAL MATRIX T.
C                             THE EIGENVALUES OF T IN ASCENDING ORDER
C                             OVERWRITE THE INPUT D
C                E        - INPUT ARRAY OF LENGTH N CONTAINS THE
C                             SUB-DIAGONAL ELEMENTS OF T. E(1) IS NOT
C                             USED.
C                N        - ORDER OF TRIDIAGONAL MATRIX T
C                Z        - INPUT = THE N X N IDENTITY MATRIX. REPLACED
C                             BY THE N X N MATRIX GIVING THE NORMALIZED
C                             EIGENVECTORS (COLUMN BY COLUMN) OF T
C                IZ       - ROW DIMENSION OF Z IN CALLING PROGRAM
C                IER      - ERROR PARAMETER
C                           TERMINAL ERROR=128+N
C                             N=1 MEANS 30 SUCCESSIVE ITERATIONS ARE MADE
C                                WITHOUT AN EIGENVALUE BEING LOCATED
C      IMPLEMENTED BY ROLF SEEGER
C      NUMER. MATH., 11, 181 (1968)
C
C
      DIMENSION          D(1),E(1),Z(IZ,1)
      DATA ZERO,ONE/.0D0,1.D0/
C=    DIMENSION IDELP(2)
C=    EQUIVALENCE (RDELP,IDELP(1))
C=    EQUIVALENCE        (RDELP,EPS)
C=    DATA IDELP/'00002580'X,0/
C                                   TOL IS A MACHINE DEPENDENT CONSTANT,
C                                   TOL = ETA/EPS, WHERE ETA IS THE
C                                   SMALLEST POSITIVE NUMBER REPRESENT-
C                                   ABLE IN THE COMPUTER AND EPS IS THE
C                                   SMALLEST POSITIVE NUMBER FOR WHICH
C                                   1+EPS.NE.1.
      EPS=2.**(-37)
C=    ETA=2.**(-218)
C=    TOL=ETA/EPS
C=
      IER  = 0
      IF (N.EQ.1) GO TO 9005
C                                   MOVE THE LAST N-1 ELEMENTS
C                                   OF E INTO THE FIRST N-1 LOCATIONS
      DO 5  I=2,N
         E(I-1) = E(I)
    5 CONTINUE
      E(N)=ZERO
      B=ZERO
      F=ZERO
      DO  60  L=1,N
         J = 0
         H=EPS*(DABS(D(L))+DABS(E(L)))
         IF (B.LT.H) B = H
C                                   LOOK FOR SMALL SUB-DIAGONAL ELEMENT
         DO 10  M=L,N
            K=M
            IF (DABS(E(K)) .GT. B) GO TO 10
            GO TO 15
   10    CONTINUE
```

```
   15    M = K
         IF (M.EQ.L) GO TO 55
   20    IF (J .EQ. 30) GO TO 85
         J = J+1
C                                   FORM SHIFT
         P = (D(L+1) - D(L))/(2.*E(L))
         R=DSQRT(P*P+ONE)
         IF(P.GE.ZERO) H=D(L)-E(L)/(P+R)
         IF(P.LT.ZERO) H=D(L)-E(L)/(P-R)
            DO 25  I=L,N
               D(I) = D(I) - H
   25       CONTINUE
         F = F+H
C                                   QL TRANSFORMATION
         P = D(M)
         C=ONE
         S=ZERO
         MM1 = M-1
         MM1PL = MM1+L
            IF (L.GT.MM1) GO TO 50
            DO 45  II=L,MM1
               I = MM1PL-II
               G = C*E(I)
               H = C*P
               IF(DABS(P).LT.DABS(E(I))) GOTO 30
               C = E(I)/P
               R=DSQRT(C*C+ONE)
               E(I+1) = S*P*R
               S = C/R
               C=ONE/R
               GO TO 35
   30          C = P/E(I)
               R=DSQRT(C*C+ONE)
               E(I+1) = S*E(I)*R
               S=ONE/R
               C = C/R
   35          P = C*D(I)-S*G
               D(I+1) = H+S*(C*G+S*D(I))
C                                   FORM VECTOR
               DO 40  K=1,N
                  H = Z(K,I+1)
                  Z(K,I+1) = S*Z(K,I)+C*H
                  Z(K,I) = C*Z(K,I)-S*H
   40          CONTINUE
   45       CONTINUE
   50    E(L) = S*P
         D(L) = C*P
         IF (DABS(E(L)) .GT.B) GO TO 20
   55    D(L) = D(L) + F
   60 CONTINUE
C                                   ORDER EIGENVALUES AND EIGENVECTORS
      DO  80  I=1,N
         K = I
         P = D(I)
         IP1 = I+1
         IF (IP1.GT.N) GO TO 70
         DO 65  J=IP1,N
            IF (D(J) .GE. P) GO TO 65
            K = J
            P = D(J)
   65    CONTINUE
   70    IF (K.EQ.I) GO TO 80
         D(K) = D(I)
         D(I) = P
            DO 75  J=1,N
               P = Z(J,I)
               Z(J,I) = Z(J,K)
               Z(J,K) = P
   75       CONTINUE
   80 CONTINUE
      GO TO 9005
   85 IER = 1
C9000 CALL LNK1E
      STOP 'ABNORMAL END IN G80 ROUTINE EQRT2D'
 9005 RETURN
      END
C
C@(#)=ERI(SUB)
      SUBROUTINE ERI(IGAMMA,IGOUT)
C
C     EVALUATION OF 2 CENTER ERIS, GAMMA.
C
C     REFERENCES.
C      C.J.J.ROOTHAAN, J.CHEM.PHYS.,19,1445(1951)
C      R.PARISER AND R.G.PARR, J.CHEM.PHYS.,21,767(1953)
C      K.NISHIMOTO AND N.MATAGA, Z.PHYS.CHEM.,13,140(1957)
C      K.Nishimoto, Proceedings of Domestic Molecular Structure
c                   Conference in Fukuoka, 2A07(1990) (in Japanese)
C      K.OHNO, THEORET.CHIM.ACTA(BERL.), 2,219(1964)
C      G.KLOPMAN, J.AMER.CHEM.SOC.,87,3300(1965)
C      A.DASGUPTA AND S.HUZINAGA, THEORET.CHIM.ACTA,35,329(1974)
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*34 CHRGMM(7)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON(17)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /GAMMA / GAB(MAPCK),GAA(MAXAT)
      COMMON /EXPNT / ZETA(MAXAT)
      COMMON /KNEWNM/ PNEWNM
      COMMON /QNOINF/ NC(MAXIAN,2),LC(9),MC(9)
      COMMON /WORK01/ DUMMY1(MBPCK,4),DUMMY2(MAXAT,3),DUMMY3(MAXAT,3),
     &                PKLN(MAXAT),DUMMY4(1)
      DATA ONE/1.0D+00/,ONEPT2/1.2D+00/
      DATA CHRGMM/'Analytically (s-type STO only).   ',
     &            'by Pariser-Parr formula.          ',
     &            'by Nishimoto-Mataga formula.      ',
     &            'by Nishimoto-Mataga-Weiss formula.',
     &            'by Ohno formula.                  ',
     &            'by Ohno-Klopman formula.          ',
     &            'by Dasgupta-Huzinaga formula.     '/
 2000 Format(1H ,'2 Center ERI, Gamma ... Evaluated ',A34)
 2010 Format(1H ,'1/(R+k*A) ... k =',F7.4)
 2020 Format(1H ,'F/(R+k*f*A) ... k =',F7.4,' ; f =',F7.4)
 2030 Format(1H ,15('=')/
     &       1H ,'Gamma(A,B) (eV)'/
     &       1H ,15('='))
C     ***************************************************************
      WRITE(IOUT,2000) CHRGMM(IGAMMA)
C
C     MAXIAN ... FROM INCLUDE STATEMENT.
      IF      (IGAMMA.EQ.1) THEN
        CALL ANL2E(COORD,ZETA,GAB,GAA,IAN,NATOMS,NC,MAXIAN)
      ELSE IF (IGAMMA.EQ.2) THEN
        CALL PP2E(COORD,ZETA,GAB,GAA,IAN,NATOMS,NC,MAXIAN)
      ELSE IF (IGAMMA.EQ.3) THEN
        WRITE(IOUT,2010) PNEWNM
        CALL NM2E(ONE,PNEWNM,COORD,GAB,GAA,NATOMS)
      ELSE IF (IGAMMA.EQ.4) THEN
        WRITE(IOUT,2020) PNEWNM,ONEPT2
        CALL NM2E(ONEPT2,PNEWNM,COORD,GAB,GAA,NATOMS)
      ELSE IF (IGAMMA.EQ.5) THEN
        CALL OHNO2E(COORD,GAB,GAA,NATOMS)
      ELSE IF (IGAMMA.EQ.6) THEN
        CALL OK2E(COORD,GAB,GAA,NATOMS)
      ELSE IF (IGAMMA.EQ.7) THEN
        CALL DH2E(PKLN,COORD,GAB,GAA,NATOMS)
      END IF
C
      IF(IGOUT.EQ.1) THEN
        WRITE(IOUT,2030)
        CALL ASMOUT(GAB,IAN,NATOMS)
      END IF
C
C     FINALLY CONVERT EV TO HARTREE.
      EVTOAU=ONE/AUTOEV
      DO 10 I=1,NATOMS
        GAA(I)=GAA(I)*EVTOAU
   10 CONTINUE
      NTTA=NATOMS*(NATOMS+1)/2
      DO 20 I=1,NTTA
        GAB(I)=GAB(I)*EVTOAU
   20 CONTINUE
C
      RETURN
      END
c
c@(#)=Fld(Fun)
      Integer Function Fld(Int,Clm,NChrs)
      Implicit Integer (a-z)
c
      Fld=Mod(Int,10**(NChrs-Clm+1))/10**(NChrs-Clm)
c
```

```
      Return
      End
      subroutine fckout( a,       nbasis, fctr )
C=====================================================================*
C                                                                     *
C      @@@@@@   @@@@   @    @   @@@@   @    @   @@@@@                 *
C      @       @    @  @    @   @    @  @    @     @                  *
C      @@@@@   @       @@@@    @    @  @    @     @                  *
C      @       @       @  @    @    @  @    @     @                  *
C      @       @    @  @    @   @    @  @    @     @                  *
C      @        @@@@   @    @   @@@@    @@@@      @                  *
C=====================================================================*
C
C     output module for sto based symmetric matrix.
C     print out lower triangle matrix.
C     ******************************************************************
      implicit real*8 (a-h,o-z)
      common /io    / in,iout,ifile(10)
      common /prwdth/ iwdth
      dimension a(*)
 2000 format(1h ,11x,10(7x,i4))
 2020 format(1h ,i4,10x,10f11.6)
C     ******************************************************************
      if( iwdth .eq. 1 ) then
        iw = 5
      else
        iw = 10
      endif
C
      ilower = 1
      iupper = iw
C
   10 continue
C
      iupper = min0( iupper, nbasis )
      write(iout,2000) ( i, i = ilower, iupper )
C
      iatold = 0
C
      do 20 i = ilower, nbasis
        nclms = i - ilower + 1
        if( nclms .gt. iw ) then
          iu = ilower + iw - 1
        else
          iu = i
        endif
        ma = i*( i - 1 )/2
        write(iout,2020) i, ( fctr*a(ma+j), j = ilower, iu )
   20 continue
C
      if( iupper .eq. nbasis ) return
C
      ilower = ilower + iw
      iupper = iupper + iw
C
      go to 10
C
      end
C
C@(#)=GETBP(SUB)
      SUBROUTINE GETBP(BP,METHOD,MAXIAN)
C
C     RESONANCE INTEGRALS.
C     REFERENCES.
C      1. CNDO/2      ... QCPE #382 CNDO/SHIFT/UV.
C      2. CNDO/S      ... H,B-F J.PHYS.CHEM.,79(11),1118(1975)
C                         LI,BE,NA-CL QCPE #333 CNDUV99
C      3. CNDO/S2,S3  ... J.CHEM.PHYS.,63(5),1758(1975)
C                         PHYS.REV.,A20(3),1179(1979)
C      4. INDO/S      ... THEOR.CHIM.ACTA(BERL.),42,223(1976)
C     ******************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION BP(MAXIAN,3)
      DATA ZERO/0.0D+00/
C     ******************************************************************
      DO 10 I=1,3
      DO 10 J=1,MAXIAN
        BP(J,I)=ZERO
   10 CONTINUE
C
      GO TO (20,60,90,90,100),METHOD
C
C     CNDO/2
   20 BP( 1,1)= 9.      D+00
      BP( 3,1)= 9.      D+00
      BP( 4,1)=13.      D+00
      BP( 5,1)=17.      D+00
      BP( 6,1)=21.      D+00
      BP( 7,1)=25.      D+00
      BP( 8,1)=31.      D+00
      BP( 9,1)=39.      D+00
      BP(11,1)= 7.7203 D+00
      BP(12,1)= 9.4471 D+00
      BP(13,1)=11.3011 D+00
      BP(14,1)=13.065  D+00
      BP(15,1)=15.070  D+00
      BP(16,1)=18.150  D+00
      BP(17,1)=22.330  D+00
      BP(21,1)= 2.      D+00
      BP(22,1)= 9.33333D+00
      BP(23,1)=12.      D+00
      BP(24,1)=17.      D+00
      BP(25,1)=22.      D+00
      BP(26,1)=26.      D+00
      BP(27,1)=29.      D+00
      BP(28,1)=32.0     D+00
      BP(29,1)=35.0     D+00
      BP(32,1)=10.      D+00
      BP(33,1)=13.      D+00
      BP(34,1)=16.      D+00
      BP(35,1)=22.      D+00
C
      DO 30 I=3,35
        BP(I,2)=BP(I,1)
   30 CONTINUE
      DO 40 I=11,17
        BP(I,3)=BP(I,1)
   40 CONTINUE
      DO 50 I=32,35
        BP(I,3)=BP(I,1)
   50 CONTINUE
C
      BP(21,3)=15.D+00
      BP(22,3)=24.D+00
      BP(23,3)=21.D+00
      BP(24,3)=23.D+00
      BP(25,3)=25.D+00
      BP(26,3)=27.D+00
      BP(27,3)=28.D+00
      BP(28,3)=29.D+00
      BP(29,3)=30.D+00
C
      RETURN
C
C     CNDO/S
   60 BP( 1,1)=12.0 D+00
      BP( 3,1)= 3.  D+00
      BP( 4,1)= 4.  D+00
      BP( 5,1)= 5.0 D+00
      BP( 6,1)=17.5 D+00
      BP( 7,1)=26.0 D+00
      BP( 8,1)=45.0 D+00
      BP( 9,1)=50.0 D+00
      BP(11,1)= 5.  D+00
      BP(12,1)= 1.  D+00
      BP(13,1)= 1.5 D+00
      BP(14,1)= 5.25D+00
      BP(15,1)= 7.8 D+00
      BP(16,1)=13.5 D+00
      BP(17,1)=15.  D+00
C
      DO 70 I=3,17
        BP(I,2)=BP(I,1)
   70 CONTINUE
      DO 80 I=11,17
        BP(I,3)=BP(I,1)
   80 CONTINUE
C
      RETURN
C
C     CNDO/S2,S3
   90 BP( 1,1)=10.00D+00
      BP( 6,1)=20.00D+00
      BP( 6,2)=17.00D+00
      BP( 7,1)=24.74D+00
      BP( 7,2)=12.71D+00
```

```
      BP( 8,1)=25.00D+00
      BP( 8,2)=20.00D+00
      BP( 9,1)=39.33D+00
      BP( 9,2)=18.80D+00
      BP(17,1)=22.52D+00
      BP(17,2)=12.56D+00
      BP(35,1)=21.13D+00
      BP(35,2)=11.23D+00
C
      RETURN
C
C     INDO/S
  100 BP( 1,1)= 9.00D+00
      BP( 6,1)=17.00D+00
      BP( 7,1)=26.00D+00
      BP( 8,1)=34.00D+00
C
      DO 110 I=3,8
        BP(I,2)=BP(I,1)
  110 CONTINUE
C
      RETURN
C
      END
c$$$#endif
c$$$#endif
C
C@(#)=GETD(SUB)
      SUBROUTINE GETD(C,D,OCC,MDIM,NBASIS,NOCC)
C
C     GET DENSITY MATRIX FROM MOLECULAR ORBITAL COEFFICIENTS.
C      C ... MOLECULAR ORBITAL COEFFICIENTS (On input).
C      D ... DENSITY MATRIX (On output).
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION C(MDIM,*),D(*)
      DATA ZERO/0.0D+00/
C     ****************************************************************
c=>   92/Sep A.M.
      ma = 0
c<=
      DO 30 I=1,NBASIS
c=>   92/Sep A.M.
c       MX=I*(I-1)/2
        DO 20 J=1,I
c         MA=MX+J
          ma = ma + 1
          DIJ=ZERO
          DO 10 IMO=1,NOCC
c           DIJ=DIJ+OCC*C(I,IMO)*C(J,IMO)
            DIJ=DIJ+C(I,IMO)*C(J,IMO)
c<=
   10     CONTINUE
          D(MA)=DIJ*Occ
   20   CONTINUE
   30 CONTINUE
C
      RETURN
      END
C
C@(#)=GETDX(SUB)
      SUBROUTINE GETDX(RX,NI,ZSP,ZD,NCSP,NCD,DXINT)
C
C     GET 1 CENTER DIPOLE-X INTEGRALS OVER SLATER-TYPE FUNCTIONS.
C      RX    ... X COMPONENT OF CARTESIAN COORDINATE FOR I-TH ATOM.
C      NI    ... NO. OF BASIS FUNCTIONS FOR I-TH ATOM.
C      ZSP   ... SLATER EXPONENT OF S-P FUNCTIONS FOR I-TH ATOM.
C      ZD    ... SLATER EXPONENT OF D FUNCTION FOR I-TH ATOM.
C      NCSP  ... PRINCIPLE QUANTUM NO. OF S-P FUNCTION FOR I-TH ATOM.
C      NCD   ... PRINCIPLE QUANTUM NO. OF D FUNCTION FOR I-TH ATOM.
C     On output:
C      DXINT ...  X COMPONENT OF 1 CENTER DIPOLE INTEGRALS OVER SLATER
C                 TYPE FUNCTIONS FOR I-TH ATOM.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DXINT(9,9)
      DATA ZERO/0.0D+00/,THREE/3.0D+00/
C     ****************************************************************
      ROOT3=DSQRT(THREE)
C
      DO 20 I=1,NI
        DO 10 J=1,NI
          DXINT(J,I)=ZERO
   10   CONTINUE
   20 CONTINUE
C
C     CALCULATE DIPOLE INTEGRALS FOR X COMPONENT.
      DO 30 I=1,NI
        DXINT(I,I)=RX
 30   CONTINUE
C
      IF(NI.GE.4) THEN
        DXINT(1,2)=SPMIX(NCSP,ZSP,ZSP)
        DXINT(2,1)=DXINT(1,2)
        IF(NI.EQ.9) THEN
          CD=PDMIX(NCSP,NCD,ZSP,ZD)
          DXINT(2,8)=CD
          DXINT(3,9)=CD
          DXINT(4,6)=CD
          DXINT(2,5)=-CD/ROOT3
          DXINT(8,2)=DXINT(2,8)
          DXINT(9,3)=DXINT(3,9)
          DXINT(6,4)=DXINT(4,6)
          DXINT(5,2)=DXINT(2,5)
        END IF
      END IF
C
      RETURN
      END
C
C@(#)=GETDY(SUB)
      SUBROUTINE GETDY(RY,NI,ZSP,ZD,NCSP,NCD,DYINT)
C
C     GET 1 CENTER DIPOLE-Y INTEGRALS OVER SLATER-TYPE FUNCTIONS.
C      RY    ... Y COMPONENT OF CARTESIAN COORDINATE FOR I-TH ATOM.
C      NI    ... NO. OF BASIS FUNCTIONS FOR I-TH ATOM.
C      ZSP   ... SLATER EXPONENT OF S-P FUNCTIONS FOR I-TH ATOM.
C      ZD    ... SLATER EXPONENT OF D FUNCTION FOR I-TH ATOM.
C      NCSP  ... PRINCIPLE QUANTUM NO. OF S-P FUNCTION FOR I-TH ATOM.
C      NCD   ... PRINCIPLE QUANTUM NO. OF D FUNCTION FOR I-TH ATOM.
C     On output:
C      DYINT ...  Y COMPONENT OF 1 CENTER DIPOLE INTEGRALS OVER SLATER
C                 TYPE FUNCTIONS FOR I-TH ATOM.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DYINT(9,9)
      DATA ZERO/0.0D+00/,THREE/3.0D+00/
C     ****************************************************************
      ROOT3=DSQRT(THREE)
C
      DO 20 I=1,NI
        DO 10 J=1,NI
          DYINT(J,I)=ZERO
   10   CONTINUE
   20 CONTINUE
C
C     CALCULATE DIPOLE INTEGRALS FOR Y COMPONENT.
      DO 30 I=1,NI
        DYINT(I,I)=RY
 30   CONTINUE
C
      IF(NI.GE.4) THEN
        DYINT(1,3)=SPMIX(NCSP,ZSP,ZSP)
        DYINT(3,1)=DYINT(1,3)
        IF(NI.EQ.9) THEN
          CD=PDMIX(NCSP,NCD,ZSP,ZD)
          DYINT(2,9)=CD
          DYINT(4,7)=CD
          DYINT(3,8)=-CD
          DYINT(3,5)=-CD/ROOT3
          DYINT(9,2)=DYINT(2,9)
          DYINT(7,4)=DYINT(4,7)
          DYINT(8,3)=DYINT(3,8)
          DYINT(5,3)=DYINT(3,5)
        END IF
      END IF
C
      RETURN
      END
C
C@(#)=GETDZ(SUB)
      SUBROUTINE GETDZ(RZ,NI,ZSP,ZD,NCSP,NCD,DZINT)
C
C     GET 1 CENTER DIPOLE-Z INTEGRALS OVER SLATER-TYPE FUNCTIONS.
C     On input:
C      RZ    ... Z COMPONENT OF CARTESIAN COORDINATE FOR I-TH ATOM.
```

```
C      NI     ... NO. OF BASIS FUNCTIONS FOR I-TH ATOM.
C      ZSP    ... SLATER EXPONENT OF S-P FUNCTIONS FOR I-TH ATOM.
C      ZD     ... SLATER EXPONENT OF D FUNCTION FOR I-TH ATOM.
C      NCSP   ... PRINCIPLE QUANTUM NO. OF S-P FUNCTION FOR I-TH ATOM.
C      NCD    ... PRINCIPLE QUANTUM NO. OF D FUNCTION FOR I-TH ATOM.
C     On output:
C      DZINT ...  Z COMPONENT OF 1 CENTER DIPOLE INTEGRALS OVER SLATER
C                 TYPE FUNCTIONS FOR I-TH ATOM.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DZINT(9,9)
      DATA ZERO/0.0D+00/,TWO/2.0D+00/,THREE/3.0D+00/
C     ****************************************************************
      ROOT3=DSQRT(THREE)
C
      DO 20 I=1,NI
        DO 10 J=1,NI
          DZINT(J,I)=ZERO
   10   CONTINUE
   20 CONTINUE
C
C     CALCULATE DIPOLE INTEGRALS FOR Z COMPONENT.
      DO 30 I=1,NI
        DZINT(I,I)=RZ
 30   CONTINUE
C
      IF(NI.GE.4) THEN
        DZINT(1,4)=SPMIX(NCSP,ZSP,ZSP)
        DZINT(4,1)=DZINT(1,4)
        IF(NI.EQ.9) THEN
          CD=PDMIX(NCSP,NCD,ZSP,ZD)
          DZINT(2,6)=CD
          DZINT(3,7)=CD
          DZINT(4,5)=CD*TWO/ROOT3
          DZINT(6,2)=DZINT(2,6)
          DZINT(7,3)=DZINT(3,7)
          DZINT(5,4)=DZINT(4,5)
        END IF
      END IF
C
      RETURN
      END
C
C@(#)=GETEP(SUB)
      SUBROUTINE GETEP(EP,METHOD,MAXIAN)
C
C     MEAN VALUES OF SUM OF IONIZATION POTENTIAL AND ELECTRON AFFINITY
C     FOR CNDO/2 AND CNDO/S, AND IONIZATION POTENTIAL FOR CNDO/S2,S3,
C     AND INDO/S
C     REFERENCES.
C      1. CNDO/2      ... QCPE #382 CNDO/SHIFT/UV
C      2. CNDO/S      ... H-F J.PHYS.CHEM.,79(11),1118(1975)
C                         NA-CL QCPE #333 CNDUV99
C      3. CNDO/S2,S3  ... J.AMER.CHEM.SOC.,94(15),5296(1972)
C                         J.CHEM.PHYS.,63(5),1758(1975)
C      4. INDO/S      ... THEOR.CHIM.ACTA(BERL.),53,21(1979)
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION EP(MAXIAN,3)
      DATA ZERO/0.0D+00/
C     ****************************************************************
      DO 10 I=1,3
      DO 10 J=1,MAXIAN
        EP(J,I)=ZERO
   10 CONTINUE
C
      GO TO (20,30,40,40,50),METHOD
C
C     CNDO/2 ... (I+A)/2
   20 EP( 1,1)= 7.1761 D+00
      EP( 3,1)= 3.1055 D+00
      EP( 3,2)= 1.258  D+00
      EP( 4,1)= 5.94557D+00
      EP( 4,2)= 2.563  D+00
      EP( 5,1)= 9.59407D+00
      EP( 5,2)= 4.001  D+00
      EP( 6,1)=14.051  D+00
      EP( 6,2)= 5.572  D+00
      EP( 7,1)=19.31637D+00
      EP( 7,2)= 7.275  D+00
      EP( 8,1)=25.39017D+00
      EP( 8,2)= 9.111  D+00
      EP( 9,1)=32.2724 D+00
      EP( 9,2)=11.08   D+00
      EP(11,1)= 2.804  D+00
      EP(11,2)= 1.302  D+00
      EP(11,3)= 0.150  D+00
      EP(12,1)= 5.1254 D+00
      EP(12,2)= 2.0516 D+00
      EP(12,3)= 0.16195D+00
      EP(13,1)= 7.7706 D+00
      EP(13,2)= 2.9951 D+00
      EP(13,3)= 0.22425D+00
      EP(14,1)=10.0327 D+00
      EP(14,2)= 4.1325 D+00
      EP(14,3)= 0.337  D+00
      EP(15,1)=14.0327 D+00
      EP(15,2)= 5.4638 D+00
      EP(15,3)= 0.500  D+00
      EP(16,1)=17.6496 D+00
      EP(16,2)= 6.989  D+00
      EP(16,3)= 0.71325D+00
      EP(17,1)=21.5906 D+00
      EP(17,2)= 8.7081 D+00
      EP(17,3)= 0.97695D+00
      EP(21,1)= 3.657  D+00
      EP(21,2)= 0.558  D+00
      EP(21,3)= 3.793  D+00
      EP(22,1)= 3.770  D+00
      EP(22,2)= 0.690  D+00
      EP(22,3)= 4.140  D+00
      EP(23,1)= 3.822  D+00
      EP(23,2)= 0.777  D+00
      EP(23,3)= 4.475  D+00
      EP(24,1)= 3.909  D+00
      EP(24,2)= 0.876  D+00
      EP(24,3)= 4.822  D+00
      EP(25,1)= 3.983  D+00
      EP(25,2)= 0.975  D+00
      EP(25,3)= 5.157  D+00
      EP(26,1)= 4.120  D+00
      EP(26,2)= 1.062  D+00
      EP(26,3)= 5.504  D+00
      EP(27,1)= 4.170  D+00
      EP(27,2)= 1.160  D+00
      EP(27,3)= 5.839  D+00
      EP(28,1)= 4.306  D+00
      EP(28,2)= 1.260  D+00
      EP(28,3)= 6.182  D+00
      EP(29,1)= 4.567  D+00
      EP(29,2)= 1.347  D+00
      EP(29,3)= 6.520  D+00
      EP(32,1)=11.435  D+00
      EP(32,2)= 4.08   D+00
      EP(33,1)=13.335  D+00
      EP(33,2)= 5.345  D+00
      EP(34,1)=16.315  D+00
      EP(34,2)= 7.1    D+00
      EP(35,1)=19.63   D+00
      EP(35,2)= 8.4    D+00
C
      RETURN
C
C     CNDO/S ... (I+A)/2
   30 EP( 1,1)= 7.175D+00
      EP( 3,1)= 3.105D+00
      EP( 3,2)= 2.05 D+00
      EP( 4,1)= 6.55 D+00
      EP( 4,2)= 3.435D+00
      EP( 5,1)=10.305D+00
      EP( 5,2)= 4.37 D+00
      EP( 6,1)=14.96 D+00
      EP( 6,2)= 5.805D+00
      EP( 7,1)=20.485D+00
      EP( 7,2)= 8.48 D+00
      EP( 8,1)=27.255D+00
      EP( 8,2)=10.965D+00
      EP( 9,1)=28.48 D+00
      EP( 9,2)=12.18 D+00
      EP(11,1)= 2.805D+00
      EP(11,2)= 1.565D+00
      EP(12,1)= 5.875D+00
      EP(12,2)= 2.29 D+00
      EP(13,1)= 8.595D+00
      EP(13,2)= 3.92 D+00
      EP(14,1)=12.125D+00
```

```
      EP(14,2)= 6.005D+00
      EP(15,1)=14.34 D+00
      EP(15,2)= 7.235D+00
      EP(16,1)=15.81 D+00
      EP(16,2)= 8.41 D+00
      EP(17,1)=17.5  D+00
      EP(17,2)= 9.38 D+00
C
      RETURN
C
C     CNDO/S2,S3 ... I
   40 EP( 1,1)=13.595D+00
      EP( 6,1)=21.34 D+00
      EP( 6,2)=11.54 D+00
      EP( 7,1)=27.51 D+00
      EP( 7,2)=14.34 D+00
      EP( 8,1)=35.50 D+00
      EP( 8,2)=17.91 D+00
      EP( 9,1)=43.70 D+00
      EP( 9,2)=20.89 D+00
      EP(17,1)=25.02 D+00
      EP(17,2)=13.96 D+00
      EP(35,1)=23.48 D+00
      EP(35,2)=12.47 D+00
C
      RETURN
C
C     INDO/S ... I
   50 EP( 1,1)=13.06D+00
      EP( 3,1)= 5.39D+00
      EP( 3,2)= 3.54D+00
      EP( 4,1)= 9.32D+00
      EP( 4,2)= 5.96D+00
      EP( 5,1)=14.05D+00
      EP( 5,2)= 8.30D+00
      EP( 6,1)=19.84D+00
      EP( 6,2)=10.93D+00
      EP( 7,1)=25.69D+00
      EP( 7,2)=14.05D+00
      EP( 8,1)=32.90D+00
      EP( 8,2)=17.28D+00
      EP( 9,1)=39.39D+00
      EP( 9,2)=20.86D+00
      EP(11,1)= 5.14D+00
      EP(11,2)= 3.04D+00
      EP(11,3)= 1.52D+00
      EP(12,1)= 7.64D+00
      EP(12,2)= 4.52D+00
      EP(12,3)= 1.74D+00
      EP(13,1)=11.33D+00
      EP(13,2)= 5.98D+00
      EP(13,3)= 1.96D+00
      EP(14,1)=15.13D+00
      EP(14,2)= 7.67D+00
      EP(14,3)= 2.05D+00
      EP(15,1)=18.66D+00
      EP(15,2)=10.78D+00
      EP(15,3)= 3.50D+00
      EP(16,1)=21.11D+00
      EP(16,2)=12.39D+00
      EP(16,3)= 4.11D+00
      EP(17,1)=25.23D+00
      EP(17,2)=15.03D+00
      EP(17,3)= 6.00D+00
      EP(19,1)= 4.34D+00
      EP(19,2)= 2.73D+00
      EP(19,3)= 1.67D+00
      EP(20,1)= 6.03D+00
      EP(20,2)= 3.96D+00
      EP(20,3)= ZERO
      EP(21,1)= 6.72D+00
      EP(21,2)= 4.20D+00
      EP(21,3)= 8.16D+00
      EP(22,1)= 7.28D+00
      EP(22,2)= 4.48D+00
      EP(22,3)= 9.07D+00
      EP(23,1)= 7.73D+00
      EP(23,2)= 4.77D+00
      EP(23,3)= 9.89D+00
      EP(24,1)= 8.07D+00
      EP(24,2)= 5.04D+00
      EP(24,3)=10.66D+00
      EP(25,1)= 8.35D+00
      EP(25,2)= 5.27D+00
      EP(25,3)=11.45D+00
      EP(26,1)= 7.27D+00
      EP(26,2)= 4.08D+00
      EP(26,3)= 8.53D+00
      EP(27,1)= 7.38D+00
      EP(27,2)= 4.02D+00
      EP(27,3)= 9.10D+00
      EP(28,1)= 7.51D+00
      EP(28,2)= 3.93D+00
      EP(28,3)= 9.79D+00
      EP(29,1)= 7.69D+00
      EP(29,2)= 3.84D+00
      EP(29,3)=10.67D+00
      EP(30,1)= 3.54D+00
      EP(30,2)= 4.77D+00
      EP(30,3)=17.57D+00
C
      RETURN
C
      END
C
C@(#)=GETF(SUB)
      SUBROUTINE GETF(D,HCORE,F,IAN,NATOMS,NBASIS,NTT,METHOD)
C
C     GET FOCK MATRIX FROM DENSITY MATRIX AND CORE HAMILTONIAN MATRIX.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /GAMMA / GAB(MAPCK),GAA(MAXAT)
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                IBTOA(MAXBAS)
      COMMON /SLACON/ G1SP(30),F2PP(30),G2SD(11:30),G1PD(11:30),
     &                F2PD(11:30),G3PD(11:30),F2DD(11:30),F4DD(11:30),
     &                MAXSCP
      DIMENSION D(*),HCORE(*),IAN(*),F(*)
      DATA PT5/0.5D+00/,SIX/6.0D+00/,SEVEN/7.0D+00/,F11/1.1D+01/,
     &     F50/5.0D+01/
C     ****************************************************************
C
C     MOVE CORE HAMILTONIAN TO FOCK.
      DO 10 I=1,NTT
        F(I)=HCORE(I)
   10 CONTINUE
C
C     FORM FOCK MATRIX FOR CNDO APPROXIMATION.
      DO 50 I=1,NBASIS
C
C     DIAGONAL PART.
        MX =I*(I-1)/2
        IAT=IBTOA(I)
        MY =IAT*(IAT-1)/2
        MA =MX+I
        FII=-PT5*D(MA)*GAA(IAT)
        DO 20 J=1,I
          JAT=IBTOA(J)
          MB =J*(J+1)/2
          IAB=MY+JAT
          FII=FII+D(MB)*GAB(IAB)
   20   CONTINUE
        FII=FII-D(MB)*GAB(IAB)
        DO 30 J=I,NBASIS
          JAT=IBTOA(J)
          MB =J*(J+1)/2
          IAB=JAT*(JAT-1)/2+IAT
          FII=FII+D(MB)*GAB(IAB)
   30   CONTINUE
C
        F(MA)=F(MA)+FII
C
        IM1=I-1
        IF(IM1.EQ.0) GO TO 50
C
C     OFF DIAGONAL PART.
        DO 40 J=1,IM1
          JAT  =IBTOA(J)
          MA   =MX+J
          IAB  =MY+JAT
          F(MA)=F(MA)-PT5*D(MA)*GAB(IAB)
   40   CONTINUE
C
   50 CONTINUE
```

```
C
      IF(METHOD.NE.5) RETURN
C
C     INDO CORRECTION FOR S AND P ORBITALS.
      DO 90 IAT=1,NATOMS
        IA=IAN(IAT)
        IF(IA.LE.2) GO TO 90
C
C     DIAGONAL PART.
        IL  =NLBAS(IAT)
        IU  =NUBAS(IAT)
        MA  = IL   *(IL+1)/2
        IX  =(IL+1)*(IL+2)/2
        IY  =(IL+2)*(IL+3)/2
        IZ  =(IL+3)*(IL+4)/2
        DSS =D(MA)
        DPP =D(IX)+D(IY)+D(IZ)
        F2IA=F2PP(IA)
        G1IA=G1SP(IA)
        TRM1=-(DSS*G1IA/SIX+SEVEN*DPP*F2IA/F50)
        TRM2=F11*F2IA/F50
C
        F(MA)=F(MA)-G1IA*DPP/SIX
        F(IX)=F(IX)+TRM1+TRM2*D(IX)
        F(IY)=F(IY)+TRM1+TRM2*D(IY)
        F(IZ)=F(IZ)+TRM1+TRM2*D(IZ)
C
C     OFF DIAGONAL PART ON THE SAME ATOM.
        ILP1=IL+1
        TRM =PT5*G1IA
        DO 60 I=ILP1,IU
          MA=I*(I-1)/2+IL
          F(MA)=F(MA)+TRM*D(MA)
   60   CONTINUE
C
        ILP2=IL+2
        TRM =TRM2
        DO 80 I=ILP2,IU
          MX =I*(I-1)/2
          IM1=I-1
          DO 70 J=ILP1,IM1
            MA=MX+J
            F(MA)=F(MA)+TRM*D(MA)
   70     CONTINUE
   80   CONTINUE
C
   90 CONTINUE
C
      RETURN
      END
C
C@(#)=GETFCT(SUB)
      SUBROUTINE GETFCT
C
C     CALCULATION OF FACTORIAL.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /FCTRL / FACT(0:50)
      DATA ONE/1.0D+00/,NFACT/50/
C     ****************************************************************
      FACT(0)=ONE
      DO 10 I=1,NFACT
        FACT(I)=FACT(I-1)*DFLOAT(I)
   10 CONTINUE
C
      RETURN
      END
C
C@(#)=GETGP(SUB)
      SUBROUTINE GETGP(GP,METHOD,IGAMMA,MULTCI,MAXIAN)
C
C     GET EMPIRICAL VALUES OF ONE CENTER ERIS'.
C     REFERENCES.
C      1. CNDO/S      ... H-F  J.PHYS.CHEM.,79(11),1118(1975)
C                         NA-CL  QCPE #333 CNDUV99
C      2. CNDO/S2,S3  ... J.CHEM.PHYS.,63(5),1758(1975)
C                         PHYS.REV.,A20(3),1179(1979)
C      3. INDO/S      ... THEOR.CHIM.ACTA(BERL.),42,223(1976)
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION GP(MAXIAN)
      DATA ZERO/0.0D+00/
C     ****************************************************************
      DO 10 I=1,MAXIAN
        GP(I)=ZERO
   10 CONTINUE
C
      GO TO (20,30,40,40,50),METHOD
C
C     IN CASE OF CNDO/2 METHOD, 1 CENTER ERI IS CALCULATED ANALYTICALLY.
   20 IF(IGAMMA.NE.1) THEN
        GO TO 30
      ELSE
        RETURN
      END IF
C
C     CNDO/S ... PARISER-PARR FORMULA (I-A).
   30 GP( 1)=12.85D+00
      GP( 3)= 2.98D+00
      GP( 4)= 5.35D+00
      GP( 5)= 8.10D+00
      GP( 6)=10.93D+00
      GP( 7)=11.88D+00
      GP( 8)=15.13D+00
      GP( 9)=17.36D+00
      GP(11)= 2.95D+00
      GP(12)= 4.46D+00
      GP(13)= 5.1D+00
      GP(14)= 6.37D+00
      GP(15)= 9.31D+00
      GP(16)=10.01D+00
      GP(17)=11.3D+00
C
      RETURN
C
C     CNDO/S2,S3 ... F0 OF DEWER'S MINDO/2 METHOD.
C     Caution. These values are partly different from the values
C              calculated by the method of Dewer's paper.
   40 GP( 1)=12.85D+00
      GP( 6)=10.63D+00
      GP( 7)=11.66D+00
      GP( 8)=13.10D+00
      GP( 9)=15.18D+00
      GP(17)=10.92D+00
      GP(35)= 8.82D+00
C
      RETURN
C
C     INDO/S ... PARISER-PARR FORMULA (I-A)
   50 GO TO (60,60,70),MULTCI
C
C     SINGLET STATE.
   60 GP( 1)=12.85D+00
      GP( 6)=11.11D+00
      GP( 7)=12.01D+00
      GP( 8)=13.00D+00
C
      RETURN
C
C     TRIPLET STATE.
   70 GP( 1)=12.85
      GP( 6)=10.50
      GP( 7)=12.47
      GP( 8)=13.00
C
      RETURN
C
      END
      double precision function geth0( d,      hcore,  eig,    nbasis,
     $                                 ntt    )
c=====================================================================*
c                                     @@@                             *
c      @@@@   @@@@@@   @@@@@  @    @  @   @                          *
c     @    @  @          @    @    @ @     @                          *
c     @       @@@@@      @    @@@@@@ @     @                          *
c     @  @@@  @          @    @    @ @     @                          *
c     @    @  @          @    @    @  @   @                           *
c      @@@@   @@@@@@     @    @    @   @@@                            *
c=====================================================================*
c
c     get scf electronic energy.
c     ****************************************************************
      implicit real*8 (a-h,o-z)
      dimension d(*), hcore(*), eig(*)
      data zero/ 0.0d+00 /, pt5/ 0.5d+00 /
c     ****************************************************************
```

```
      e = zero
      do 1 i = 1, nbasis
        e = e + eig( i )
    1 continue
      do 2 ir = 1, nbasis
        do 3 is = 1, nbasis
          kpp  = max0( is, ir )
          kppx = min0( is, ir )
          nu = kpp*( kpp - 1 )/2 + kppx
          e = e + pt5*d( nu )*hcore( nu )
    3   continue
    2 continue
c
      geth0 = e
c
      return
      end
C
C@(#)=GETPK(SUB)
      SUBROUTINE GETPK(PK,IGAMMA,MAXIAN)
C
C     HUZINAGA'S KLONDIKE PARAMETERS.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION PK(MAXIAN)
      DATA PT4/0.4D+00/
C     ***************************************************************
      IF(IGAMMA.NE.7) RETURN
C
      DO 10 I=1,MAXIAN
        PK(I)=PT4
   10 CONTINUE
C
      RETURN
      END
C
C@(#)=GETSCP
      SUBROUTINE GETSCP
C
C     SLATER-CONDON PARAMETERS.
C     REFERENCE.
C      THEOR.CHIM.ACTA(BERL.),53,21(1979)
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PLANCK,SLIGHT,EVTOER,PCON(14)
      COMMON /SLACON/ G1SP(30),F2PP(30),G2SD(11:30),G1PD(11:30),
     &                F2PD(11:30),G3PD(11:30),F2DD(11:30),F4DD(11:30),
     &                MAXSCP
      DATA ZERO/0.0D+00/,TENP5/1.0D+05/,MDIM/30/
C     ***************************************************************
      MAXSCP=MDIM
C
C     UNIT IN CM-1.
      G1SP( 1)=  ZERO
      G1SP( 2)=  ZERO
      G1SP( 3)= 20194D+00
      G1SP( 4)= 30876D+00
      G1SP( 5)= 43566D+00
      G1SP( 6)= 55635D+00
      G1SP( 7)= 72255D+00
      G1SP( 8)= 95298D+00
      G1SP( 9)=116828D+00
      G1SP(10)=  ZERO
      G1SP(11)= 13450D+00
      G1SP(12)= 19974D+00
      G1SP(13)= 27093D+00
      G1SP(14)= 38814D+00
      G1SP(15)=  8541D+00
      G1SP(16)= 24807D+00
      G1SP(17)= 71000D+00
      G1SP(18)=  ZERO
      G1SP(19)=  8986D+00
      G1SP(20)= 12600D+00
      G1SP(21)= 12100D+00
      G1SP(22)= 13100D+00
      G1SP(23)= 15100D+00
      G1SP(24)= 14400D+00
      G1SP(25)= 18900D+00
      G1SP(26)= 16300D+00
      G1SP(27)= 22700D+00
      G1SP(28)= 19400D+00
      G1SP(29)= 20700D+00
      G1SP(30)= 20400D+00
C
      F2PP( 1)=  ZERO
      F2PP( 2)=  ZERO
      F2PP( 3)= 10944D+00
      F2PP( 4)= 21425D+00
      F2PP( 5)= 28075D+00
      F2PP( 6)= 36375D+00
      F2PP( 7)= 52100D+00
      F2PP( 8)= 55675D+00
      F2PP( 9)= 69310D+00
      F2PP(10)=  ZERO
      F2PP(11)=  6000D+00
      F2PP(12)= 26400D+00
      F2PP(13)= 12925D+00
      F2PP(14)= 18250D+00
      F2PP(15)= 23775D+00
      F2PP(16)= 36600D+00
      F2PP(17)= 52000D+00
      F2PP(18)=  ZERO
      F2PP(19)=  4000D+00
      F2PP(20)=  2325D+00
      F2PP(21)=  5000D+00
      F2PP(22)=  5500D+00
      F2PP(23)=  6000D+00
      F2PP(24)=  6500D+00
      F2PP(25)=  7000D+00
      F2PP(26)=  7500D+00
      F2PP(27)=  8000D+00
      F2PP(28)=  8500D+00
      F2PP(29)=  9000D+00
      F2PP(30)=  9500D+00
C
      G2SD(11)=  3042D+00
      G2SD(12)=  3885D+00
      G2SD(13)=  1426D+00
      G2SD(14)= 14496D+00
      G2SD(15)= 19081D+00
      G2SD(16)= 25972D+00
      G2SD(17)= 17131D+00
      G2SD(18)=  ZERO
      G2SD(19)=  ZERO
      G2SD(20)=  3730D+00
      G2SD(21)=  5870D+00
      G2SD(22)=  6200D+00
      G2SD(23)=  6240D+00
      G2SD(24)=  5220D+00
      G2SD(25)=  6110D+00
      G2SD(26)=  6640D+00
      G2SD(27)=  6340D+00
      G2SD(28)=  6700D+00
      G2SD(29)=  4460D+00
      G2SD(30)=  ZERO
C
      G1PD(11)=  9224D+00
      G1PD(12)=  3070D+00
      G1PD(13)=  1896D+00
      G1PD(14)=  2506D+00
      G1PD(15)= 25334D+00
      G1PD(16)= 34486D+00
      G1PD(17)=  2274D+00
      G1PD(18)=  ZERO
      G1PD(19)=  ZERO
      G1PD(20)=  5890D+00
      G1PD(21)=  5650D+00
      G1PD(22)=  7320D+00
      G1PD(23)=  5180D+00
      G1PD(24)=  5580D+00
      G1PD(25)=  1240D+00
      G1PD(26)=  2450D+00
      G1PD(27)=  3170D+00
      G1PD(28)=  3010D+00
      G1PD(29)=  5620D+00
      G1PD(30)=  ZERO
C
      F2PD(11)= 11000D+00
      F2PD(12)=  4416D+00
      F2PD(13)=  3461D+00
      F2PD(14)= 19675D+00
      F2PD(15)= 23176D+00
      F2PD(16)= 29173D+00
      F2PD(17)=  5102D+00
      F2PD(18)=  ZERO
      F2PD(19)=  ZERO
```

```
      F2PD(20)=  4480D+00
      F2PD(21)= 11000D+00
      F2PD(22)= 13700D+00
      F2PD(23)= 11200D+00
      F2PD(24)= 11400D+00
      F2PD(25)=  8010D+00
      F2PD(26)=  5020D+00
      F2PD(27)=  6290D+00
      F2PD(28)=  6050D+00
      F2PD(29)= 10700D+00
      F2PD(30)=  ZERO
C
      G3PD(11)=  5506D+00
      G3PD(12)=  1833D+00
      G3PD(13)=  1130D+00
      G3PD(14)= 11828D+00
      G3PD(15)= 15124D+00
      G3PD(16)= 20587D+00
      G3PD(17)=  1358D+00
      G3PD(18)=  ZERO
      G3PD(19)=  ZERO
      G3PD(20)=  ZERO
      G3PD(21)=  2210D+00
      G3PD(22)= 10300D+00
      G3PD(23)=  1710D+00
      G3PD(24)=   297D+00
      G3PD(25)=  4970D+00
      G3PD(26)=  3520D+00
      G3PD(27)=  2260D+00
      G3PD(28)=  3250D+00
      G3PD(29)=  6930D+00
      G3PD(30)=  ZERO
C
      F2DD(11)=  7477D+00
      F2DD(12)=  7477D+00
      F2DD(13)=  7477D+00
      F2DD(14)= 19438D+00
      F2DD(15)= 23925D+00
      F2DD(16)= 28411D+00
      F2DD(17)= 12860D+00
      F2DD(18)=  ZERO
      F2DD(19)=  ZERO
      F2DD(20)= 18900D+00
      F2DD(21)= 29500D+00
      F2DD(22)= 44900D+00
      F2DD(23)= 50800D+00
      F2DD(24)= 63500D+00
      F2DD(25)= 66000D+00
      F2DD(26)= 61000D+00
      F2DD(27)= 64500D+00
      F2DD(28)= 79800D+00
      F2DD(29)= 85980D+00
      F2DD(30)=  ZERO
C
      F4DD(11)=  4876D+00
      F4DD(12)=  4876D+00
      F4DD(13)=  4876D+00
      F4DD(14)= 12677D+00
      F4DD(15)= 15603D+00
      F4DD(16)= 18529D+00
      F4DD(17)=  8387D+00
      F4DD(18)=  ZERO
      F4DD(19)=  ZERO
      F4DD(20)=  9500D+00
      F4DD(21)= 14600D+00
      F4DD(22)= 29700D+00
      F4DD(23)= 35400D+00
      F4DD(24)= 36800D+00
      F4DD(25)= 37900D+00
      F4DD(26)= 38400D+00
      F4DD(27)= 48100D+00
      F4DD(28)= 53300D+00
      F4DD(29)= 57965D+00
      F4DD(30)=  ZERO
C
C     THEN CONVERT INTO HARTREE.
      CTOAU=(PLANCK*SLIGHT)/(EVTOER*AUTOEV*TENP5)
      DO 10 I=1,MAXSCP
        G1SP(I)=G1SP(I)*CTOAU
        F2PP(I)=F2PP(I)*CTOAU
   10 CONTINUE
      DO 20 I=11,MAXSCP
        G2SD(I)=G2SD(I)*CTOAU
        G1PD(I)=G1PD(I)*CTOAU
        F2PD(I)=F2PD(I)*CTOAU
        G3PD(I)=G3PD(I)*CTOAU
        F2DD(I)=F2DD(I)*CTOAU
        F4DD(I)=F4DD(I)*CTOAU
   20 CONTINUE
C
      RETURN
      END
C
C@(#)=GETVAL(SUB)
      SUBROUTINE GETVAL(CHRSTR,LENCHR,ISTART)
C
C     GET NUMERICAL VALUE FROM CHARACTER STRINGS, CHRSTR, AND RETURN IT
C     BY LABELED COMMON /COMVAL/.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /COMVAL/ VAL,IVAL,IEND
      CHARACTER*1 CHRSTR(*)
      CHARACTER*1 CZERO,CNINE,CPLUS,CMINUS,CPOINT,IBL1
      DATA ZERO/0.0D+00/,ONE/1.0D+00/,TEN/1.0D+01/
      DATA CZERO/'0'/,CNINE/'9'/,CPLUS/'+'/,CMINUS/'-'/,CPOINT/'.'/
      DATA IBL1/' '/
 9910 Format(1H ,56('X')/
     &       1H ,'VALUE WAS OUT OF RANGE IN GETVAL. CHECK YOUR INPUT ',
     &           'DATA.'/
     &       1H ,56('X'))
C     ***************************************************************
      FLTN=ZERO
      PTN =ZERO
      ORD =ONE
      PRTY=ONE
      IDIG=0
      MDIG=17
C
      IZERO =ICHAR(CZERO)
      NINE  =ICHAR(CNINE)
      IPLUS =ICHAR(CPLUS)
      MINUS =ICHAR(CMINUS)
      IPOINT=ICHAR(CPOINT)
      IBLANK=ICHAR(IBL1)
C
      LENM1=LENCHR-1
      DO 10 I=ISTART,LENM1
        IA=ICHAR(CHRSTR(I))
        IB=ICHAR(CHRSTR(I+1))
        IF( (IA.GE.IZERO.AND.IA.LE.NINE) .OR. (IA.EQ.IPOINT)  ) GO TO 20
        IF( (IA.EQ.IPLUS.OR.IA.EQ.MINUS) .AND.
     &      ((IB.GE.IZERO.AND.IB.LE.NINE) .OR. (IB.EQ.IPOINT)) ) GO TO 20
   10 CONTINUE
C
      VAL =ZERO
      IVAL=0
      IEND=ISTART
C
      RETURN
C
   20 JSTART=I
      DO 30 I=JSTART,LENCHR
        IA=ICHAR(CHRSTR(I))
        IF(IA.GE.IZERO.AND.IA.LE.NINE) THEN
          IDIG=IDIG+1
          IF(IDIG.GT.9) THEN
            WRITE(IOUT,9910)
            CALL MERROR
          END IF
          FLTN=FLTN*TEN+DFLOAT(IA-IZERO)
        ELSE IF(IA.EQ.IPLUS.AND.I.EQ.JSTART) THEN
          PRTY=+ONE
        ELSE IF(IA.EQ.MINUS.AND.I.EQ.JSTART) THEN
          PRTY=-ONE
        ELSE IF(IA.EQ.IPOINT) THEN
          GO TO 40
        ELSE
          GO TO 60
        END IF
   30 CONTINUE
C
   40 IDIG=0
C
      JSTART=I+1
      DO 50 I=JSTART,LENCHR
```

```
        IA=ICHAR(CHRSTR(I))
        IF(IA.GE.IZERO.AND.IA.LE.NINE) THEN
          IDIG=IDIG+1
          IF(IDIG.GT.MDIG) THEN
            WRITE(IOUT,9910)
            CALL MERROR
          END IF
          PTN=PTN*TEN+DFLOAT(IA-IZERO)
          ORD=ORD/TEN
        ELSE
          GO TO 60
        END IF
   50 CONTINUE
C
   60 IEND=I-1
      DO 70 I=ISTART,IEND
        CHRSTR(I)=IBL1
   70 CONTINUE
C
      VAL =PRTY*(FLTN+PTN*ORD)
      IVAL=IDNINT(PRTY*FLTN)
C
      RETURN
      END
C
C@(#)=GETZP(SUB)
      SUBROUTINE GETZP(ZP,ZCSP3,METHOD,MAXIAN)
C
C     GET EXPONENTS OF SLATER-TYPE ORBITALS.
C     REFERENCES.
C      1. CNDO/2, CNDO/S, AND INDO/S ... QCPE #261 CNDO 2/3R
C      2. CNDO/S2,S3                 ... J.CHEM.PHYS.,63(5),1758(1975)
C                                        PHYS.REV.,A20(3),1179(1979)
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /PCONST/ PI,TOANG,PCON(18)
      DIMENSION ZP(MAXIAN)
      DATA ZERO/0.0D+00/,PT325/0.325D+00/,PT65/0.65D+00/
      DATA F4PT95/4.95D+00/,THREE/3.0D+00/
C     ****************************************************************
      DO 10 I=1,MAXIAN
        ZP(I)=ZERO
   10 CONTINUE
C
      GO TO (20,20,50,60,20),METHOD
C
C     CNDO/2, CNDO/S, AND INDO/S ... SLATER RULE
   20 ZP( 1)=1.2  D+00
      ZP( 2)=1.7  D+00
      DO 30 I=3,10
        ZP(I)=PT325*DFLOAT(I-1)
   30 CONTINUE
      DO 40 I=11,18
        ZP(I)=(PT65*DFLOAT(I)-F4PT95)/THREE
   40 CONTINUE
      ZP(31)=1.351D+00
      ZP(32)=1.527D+00
      ZP(33)=1.702D+00
      ZP(34)=1.878D+00
      ZP(35)=2.054D+00
C
      RETURN
C
C     CNDO/S2 WITH CONVERSION OF UNIT. ANGSTROM**(-1) => BOHR**(-1)
   50 ZP( 1)=2.30D+00*TOANG
      ZP( 6)=3.78D+00*TOANG
      ZP( 7)=3.03D+00*TOANG
      ZP( 8)=4.10D+00*TOANG
      ZP( 9)=4.50D+00*TOANG
C
      RETURN
C
C     CNDO/S3 WITH CONVERSION OF UNIT. ANGSTROM**(-1) => BOHR**(-1)
   60 ZP( 1)=2.30D+00*TOANG
      ZP( 6)=3.78D+00*TOANG
      ZCSP3 =3.07D+00*TOANG
      ZP( 7)=3.03D+00*TOANG
      ZP( 8)=4.10D+00*TOANG
      ZP( 9)=4.50D+00*TOANG
C
      RETURN
C
      END
```



```
C
C@(#)=GPRPTY(SUB)
      SUBROUTINE GPRPTY
C
C     OUTPUT MODULE FOR GROUND STATE RESTRICTED CLOSED SHELL SCF RESULTS.
C      1 ... ENERGIES - ELECTRONIC ENERGY.
C                     - NUCLEAR REPULSION ENERGY.
C                     - TOTAL ENERGY
C                     - ORBITAL ENERGYS (OPTIONAL. WITH MO COEFFICIENTS)
C      2 ... MOLECULAR ORBITAL COEFFICIENTS (OPTIONAL).
C      3 ... ATOMIC DENSITIES FOR CANONICAL MOS (NORMALIZED TO UNITY).
C      4 ... TOTAL ATOMIC CHARGES.
C      5 ... DIPOLE MOMENT FOR GROUND STATE.
C
c     Oct/92 ... A.M.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*17 CHR17(3)
      Logical Anchor
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /IOPGPR/ IOPGPR(18)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON1(1),SLIGHT,EVTOER,EVTOKC,
     &                PCON2(13)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /EXPNT / ZETA(MAXAT)
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                IBTOA(MAXBAS)
      COMMON /ENERGY/ ENERGY
      COMMON /QNOINF/ NC(MAXIAN,2),LC(9),MC(9)
      COMMON /VELEC / ZV(MAXAT)
      COMMON /DGRND / DXPT,DYPT,DZPT,DXAP,DYAP,DZAP,DX,DY,DZ
      Common /AOChrg/ GChrg(MaxBas), AtmChg(MaxAt)
      Common /Anchor/ Anchor
      COMMON /WORK01/ F(MBPCK),HCORE(MBPCK),D(MBPCK),Dummy1(MAXAT),
     &                DM(MAXAT,MAXBAS),
     &                DXINT(9,9),DYINT(9,9),DZINT(9,9),DUMMY2(1)
      COMMON /WORK02/ C(MAXBAS,MAXBAS),EIG(MAXBAS),DUMMY3(1)
      DATA ZERO/0.0D+00/,TWO/2.0D+00/
      DATA CHR17/'Total            ',
     &           'Electronic       ',
     &           'Nuclear repulsion'/
 2000 Format(1H ,'<<<<<< Enter GPrpty >>>>>>')
 2010 Format(1H ,'Energy table for ground state')
 2020 Format(1H ,79('-'))
 2030 Format(1H ,30X,'Hartree',10X,'Electron volt',6X,'kcal/mol')
 2040 Format(1H ,1X,A17,4X,F18.9,2X,F18.9,2X,D16.9)
 2050 Format(1H ,'Highest Occupied  Molecular Orbital (HOMO) ...',I4/
     &       1H ,'Lowest Unoccupied Molecular Orbital (LUMO) ...',I4)
 2060 Format(1H ,'Canonical orbital energies (Unit : eV)')
 2070 Format(1H ,30('=')/
     &       1H ,'Molecular orbital coefficients'/
     &       1H ,30('='))
 2080 Format(1H ,25('=')/
     &       1H ,'MO densities (Norm=unity)'/
     &       1H ,25('='))
 2090 Format(1H ,14('=')/
     &       1H ,'Density matrix'/
     &       1H ,14('='))
 2100 Format(1H ,'Total atomic charges ...')
 2110 Format(1H ,'Dipole moment (Unit : Debye) ...'/
     &       1H ,32X,'x',11X,'y',11X,'z',9X,'Total')
 2120 Format(1H ,2X,'Total atomic charges',2X,4(3X,F9.4))
 2130 Format(1H ,2X,'sp atomic polarization', 4(3X,F9.4))
 2140 Format(1H ,2X,'pd atomic polarization', 4(3X,F9.4))
 2150 Format(1H ,2X,'Total',17X,              4(3X,F9.4))
C     ****************************************************************
C
      MOCOEF=IOPGPR(11)
      MODENS=IOPGPR(12)
      IDNS  =IOPGPR(13)
C
      WRITE(IOUT,2000)
C========( Wed Oct 18 14:38:02 JST 1995 )======
c$$$      c( 2,  10 ) = -0.4287
c$$$      c( 6,  10 ) = -0.5623
c$$$      c( 10, 10 ) = -0.5623
c$$$      c( 14, 10 ) = -0.4287
c$$$      c( 2,  11 ) = -0.5717
c$$$      c( 6,  11 ) = -0.4161
c$$$      c( 10, 11 ) =  0.4161
```

```
c$$$       c( 14, 11 ) =  0.5717
c$$$       c( 2,  12 ) =  0.5623
c$$$       c( 6,  12 ) = -0.4287
c$$$       c( 10, 12 ) = -0.4287
c$$$       c( 14, 12 ) =  0.5623
c$$$       c( 2,  13 ) = -0.4161
c$$$       c( 6,  13 ) =  0.5717
c$$$       c( 10, 13 ) = -0.5717
c$$$       c( 14, 13 ) =  0.4161
c========( Wed Oct 18 14:38:02 JST 1995 )======
C
      EEAU=ENERGY
      EEEV=EEAU*AUTOEV
      EEKC=EEEV*EVTOKC
C
C     ESTIMATION OF NUCLEAR REPULSION ENERGY.
      ENAU=ZERO
      DO 20 IAT=2,NATOMS
        CXI  =COORD(1,IAT)
        CYI  =COORD(2,IAT)
        CZI  =COORD(3,IAT)
        ZVI  =ZV(IAT)
        IATM1=IAT-1
        DO 10 JAT=1,IATM1
          CX  =COORD(1,JAT)-CXI
          CY  =COORD(2,JAT)-CYI
          CZ  =COORD(3,JAT)-CZI
          RIJ =DSQRT(CX*CX+CY*CY+CZ*CZ)
          ENAU=ENAU+ZV(JAT)*ZVI/RIJ
   10   CONTINUE
   20 CONTINUE
C
      ENEV=ENAU*AUTOEV
      ENKC=ENEV*EVTOKC
C
      ETAU=EEAU+ENAU
      ETEV=EEEV+ENEV
      ETKC=EEKC+ENKC
C
      WRITE(IOUT,2010)
      WRITE(IOUT,2020)
      WRITE(IOUT,2030)
      WRITE(IOUT,2020)
      WRITE(IOUT,2040) CHR17(1),ETAU,ETEV,ETKC
      WRITE(IOUT,2040) CHR17(2),EEAU,EEEV,EEKC
      WRITE(IOUT,2040) CHR17(3),ENAU,ENEV,ENKC
      WRITE(IOUT,2020)
C
      IHOMO=NAE
      ILUMO=NAE+1
      WRITE(IOUT,2050) IHOMO,ILUMO
C
C     PRINTING OF ORBITAL ENERGIES.
      WRITE(IOUT,2060)
      CALL ENGOUT(EIG,AUTOEV,IHOMO,ILUMO,NBASIS)
C
c========( Tue Oct 17 13:43:13 JST 1995 )======
c$$$       MOCOEF = IHOMO
c========( Tue Oct 17 13:43:13 JST 1995 )======
      IF(MOCOEF.EQ.0) GO TO 30
C
C     PRINTING OF MOLECULAR ORBITAL COEFFICIENTS.
      IF(MOCOEF.LT.0) THEN
        MINCLM=1
        MAXCLM=NBASIS
      ELSE
        MINCLM=IHOMO-(MOCOEF-1)
        MAXCLM=ILUMO+(MOCOEF-1)
        IF(MINCLM.LT.1) THEN
          MINCLM=1
        END IF
        IF(MAXCLM.GT.NBASIS) THEN
          MAXCLM=NBASIS
        END IF
      END IF
      WRITE(IOUT,2070)
      CALL EIGOUT(C,EIG,MAXBAS,IAN,IBTOA,NBASIS,MINCLM,MAXCLM)
c========( Tue Oct 17 13:43:36 JST 1995 )======
c$$$       ii = 1
c$$$       if( ii .eq. 1 ) stop
c========( Tue Oct 17 13:43:37 JST 1995 )======
C
   30 IF(MODENS.EQ.0) GO TO 70
C
C     PRINTING OF MO DENSITY (NORMALIZED TO UNITY).
      IF(MODENS.LT.0) THEN
        MINCLM=1
        MAXCLM=NBASIS
      ELSE
        MINCLM=IHOMO-(MODENS-1)
        MAXCLM=ILUMO+(MODENS-1)
        IF(MINCLM.LT.1) THEN
          MINCLM=1
        END IF
        IF(MAXCLM.GT.NBASIS) THEN
          MAXCLM=NBASIS
        END IF
      END IF
C
      DO 60 IMO=MINCLM,MAXCLM
        DO 50 IAT=1,NATOMS
          IL=NLBAS(IAT)
          IU=NUBAS(IAT)
          CC=ZERO
          DO 40 I=IL,IU
            CC=CC+C(I,IMO)*C(I,IMO)
   40     CONTINUE
          DM(IAT,IMO)=CC
   50   CONTINUE
   60 CONTINUE
C
      WRITE(IOUT,2080)
      CALL DMOUT(DM,MAXAT,IAN,NATOMS,MINCLM,MAXCLM)
C
C     PRINTING OF DENSITY MATRIX.
   70 IF(IDNS.EQ.1) THEN
        WRITE(IOUT,2090)
        CALL SSMOUT(D,IAN,IBTOA,NLBAS,NBASIS)
      END IF
C
C     PRINTING OF TOTAL ATOMIC CHARGES.
      DO 90 IAT=1,NATOMS
        IL=NLBAS(IAT)
        IU=NUBAS(IAT)
        CHG=ZERO
        DO 80 I=IL,IU
          MA=I*(I+1)/2
          CHG=CHG+D(MA)
          GChrg(i) = D(ma)
   80   CONTINUE
        ATMCHG(IAT)=ZV(IAT)-CHG
   90 CONTINUE
      WRITE(IOUT,2100)
      CALL ACOUT(ATMCHG,IAN,NATOMS)
C
C     GET DIPOLE MOMENT FOR GROUND STATE.
      DXPT=ZERO
      DYPT=ZERO
      DZPT=ZERO
      DXSP=ZERO
      DYSP=ZERO
      DZSP=ZERO
      DXPD=ZERO
      DYPD=ZERO
      DZPD=ZERO
C
      IND =0
C
      DO 140 IAT=1,NATOMS
C
        RX=COORD(1,IAT)
        RY=COORD(2,IAT)
        RZ=COORD(3,IAT)
C
C     POINT CHARGE PART.
        DXPT=DXPT+ATMCHG(IAT)*RX
        DYPT=DYPT+ATMCHG(IAT)*RY
        DZPT=DZPT+ATMCHG(IAT)*RZ
C
        NI  =NIBAS(IAT)
        IA  =IAN(IAT)
        ZSP =ZETA(IAT)
        ZD  =ZETA(IAT)
        NCSP=NC(IA,1)
        NCD =NC(IA,2)
C
```

```
      IF(NI.EQ.1) GO TO 130
C
C     GET 1 CENTER DIPOLE INTEGRALS OVER SLATER-TYPE ORBITALS.
        CALL GETDX(RX,NI,ZSP,ZD,NCSP,NCD,DXINT)
        CALL GETDY(RY,NI,ZSP,ZD,NCSP,NCD,DYINT)
        CALL GETDZ(RZ,NI,ZSP,ZD,NCSP,NCD,DZINT)
C
C     S-P PART.  IBAS=S; JBAS=PX,PZ
        IBAS=1
        DO 100 JBAS=2,4
          MA  =(JBAS+IND)*((JBAS+IND)-1)/2+(IBAS+IND)
          DXSP=DXSP+D(MA)*DXINT(JBAS,IBAS)
          DYSP=DYSP+D(MA)*DYINT(JBAS,IBAS)
          DZSP=DZSP+D(MA)*DZINT(JBAS,IBAS)
  100   CONTINUE
C
      IF(NI.EQ.4) GO TO 130
C
C     P-D PART.  IBAS=PX,PZ; JBAS=DZ2,DXY
        DO 120 IBAS=2,4
          DO 110 JBAS=5,9
            MA  =(JBAS+IND)*((JBAS+IND)-1)/2+(IBAS+IND)
            DXPD=DXPD+D(MA)*DXINT(JBAS,IBAS)
            DYPD=DYPD+D(MA)*DYINT(JBAS,IBAS)
            DZPD=DZPD+D(MA)*DZINT(JBAS,IBAS)
  110     CONTINUE
  120   CONTINUE
C
  130   IND=IND+NI
C
  140 CONTINUE
C
      DXSP=-DXSP*TWO
      DYSP=-DYSP*TWO
      DZSP=-DZSP*TWO
      DXPD=-DXPD*TWO
      DYPD=-DYPD*TWO
      DZPD=-DZPD*TWO
      DXAP= DXSP+DXPD
      DYAP= DYSP+DYPD
      DZAP= DZSP+DZPD
      DTPT= DSQRT(DXPT**2+DYPT**2+DZPT**2)
      DTSP= DSQRT(DXSP**2+DYSP**2+DZSP**2)
      DTPD= DSQRT(DXPD**2+DYPD**2+DZPD**2)
      DXT = DXPT+DXSP+DXPD
      DYT = DYPT+DYSP+DYPD
      DZT = DZPT+DZSP+DZPD
      DTOT=DSQRT(DXT**2+DYT**2+DZT**2)
      AUTODB=SLIGHT*TOANG*EVTOER
C
      WRITE(IOUT,2110)
      WRITE(IOUT,2120) DXPT*AUTODB,DYPT*AUTODB,DZPT*AUTODB,DTPT*AUTODB
      WRITE(IOUT,2130) DXSP*AUTODB,DYSP*AUTODB,DZSP*AUTODB,DTSP*AUTODB
      WRITE(IOUT,2140) DXPD*AUTODB,DYPD*AUTODB,DZPD*AUTODB,DTPD*AUTODB
      WRITE(IOUT,2150) DXT *AUTODB,DYT *AUTODB,DZT *AUTODB,DTOT*AUTODB
c
      If (Anchor) Call AncGP(NAtoms,NBasis,MaxBas,IFile(10),
     &                 DXT*AUTODB,DYT*AUTODB,DZT*AUTODB,DTOT*AUTODB,
     &                 AtmChg,Eig,C,GChrg)

C
C     ALL DONE. RETURN TO CALLER.
      CALL CLOCKM('GPrpty.')
      RETURN
      END
C
C@(#)=GUESS(SUB)
      SUBROUTINE GUESS
C
C     INITIAL GUESS CALCULATION BY HUCKEL TYPE METHOD.
C     93/Jun. A.M.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cf    INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO     / IN,IOUT,IPILE(10)
      COMMON /METHOD/ METHOD
      COMMON /IOPGES/ IOPGES(18)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /GAMMA / GAB(MAPCK),GAA(MAXAT)
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                 IBTOA(MAXBAS)
      COMMON /QNOINF/ NC(MAXIAN,2),LC(9),MC(9)
      COMMON /SLACON/ G1SP(30),F2PP(30),G2SD(11:30),G1PD(11:30),
     &                 F2PD(11:30),G3PD(11:30),F2DD(11:30),F4DD(11:30),
     &                 MAXSCP
      COMMON /VELEC / ZV(MAXAT)
      COMMON /WORK01/ S(MBPCK),HCORE(MBPCK),D(MBPCK),H(MBPCK),
     &                 EN(MAXAT,3),BETA(MAXAT,3),DUMMY(1)
      COMMON /WORK02/ HVEC(MAXBAS,MAXBAS),HEIG(MAXBAS),
     &                 SCRTCH(MAXBAS,19)
      DATA ZERO/0.0D+00/,PT5/0.5D+00/,PT75/0.75D+00/,ONE/1.0D+00/,
     &     TWO/2.0D+00/,SIX/6.0D+00/,F25/25.0D+00/
 2000 Format(1H ,'<<<<<< Enter Guess >>>>>>')
 2010 Format(1H ,'Initial Huckel guess.')
 2020 Format(1H ,25('=')/
     &       1H ,'HUCKEL HAMILTONIAN MATRIX'/
     &       1H ,25('='))
 2030 Format(1H ,13('=')/
     &       1H ,'INITIAL GUESS'/
     &       1H ,13('='))
 2040 Format(1H ,16('=')/
     &       1H ,'CORE HAMILTONIAN'/
     &       1H ,16('='))
C     ***************************************************************
      IPRGES=IOPGES(11)
      IPRHCR=IOPGES(12)
      NTT   =NBASIS*(NBASIS+1)/2
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010)
C
C     FORM EXTENDED HUCKEL-TYPE HAMILTONIAN MATRIX ... TWO STEP.
C     1 ... DIAGONAL PART AND OFF DIAGONAL PART ON THE SAME ATOM.
      DO 30 IAT=1,NATOMS
        IL=NLBAS(IAT)
        IU=NUBAS(IAT)
        IX=0
        DO 20 I=IL,IU
          IX=IX+1
          LI=LC(IX)+1
          MX=I*(I-1)/2
          MA=MX+I
          HCORE(MA)=-EN(IAT,LI)
          IM1=I-1
          IF(IM1.LT.IL) GO TO 20
          DO 10 J=IL,IM1
            MB=MX+J
            HCORE(MB)=ZERO
   10     CONTINUE
   20   CONTINUE
   30 CONTINUE
C
C     2 ... OFF DIAGONAL PART ON THE DIFFERENT ATOMS.
      DO 70 IAT=2,NATOMS
        IA   =IAN(IAT)
        IL   =NLBAS(IAT)
        IU   =NUBAS(IAT)
        IATM1=IAT-1
        IX   =0
        DO 60 I=IL,IU
          MX=I*(I-1)/2
          IX=IX+1
          LI=LC(IX)+1
          BI=BETA(IAT,LI)
          DO 50 JAT=1,IATM1
            JA=IAN(JAT)
            JL=NLBAS(JAT)
            JU=NUBAS(JAT)
            JX=0
            IF(METHOD.EQ.1 .AND. (IA.GT.9 .OR. JA.GT.9)) THEN
              CON=PT75
            ELSE
              CON=ONE
            END IF
            DO 40 J=JL,JU
              JX=JX+1
              LJ=LC(JX)+1
              BJ=BETA(JAT,LJ)
              MA=MX+J
              HCORE(MA)=CON*S(MA)*(BI+BJ)*PT5
   40       CONTINUE
   50     CONTINUE
   60   CONTINUE
   70 CONTINUE
```

```
C
C     THEN MOVE HCORE TO H (DESTROYED BY ROUTINE DIAGD).
      DO 80 I=1,NTT
        H(I)=HCORE(I)
   80 CONTINUE
C
      IF(IPRGES.EQ.1) THEN
        WRITE(IOUT,2020)
        CALL SSMOUT(H,IAN,IBTOA,NLBAS,NBASIS)
      END IF
C
C     DIAGONALIZE H.
      CALL DIAG1(H,HVEC,HEIG,NBASIS,MAXBAS,SCRTCH,IOUT)
C
C     NOW ARRAYS CONTAIN AS FOLLOWS.
C     HCORE ... EXTENDED HUCKEL MATRIX ELEMENTS.
C     HVEC  ... EIGENVECTORS OF HCORE.
C     HEIG  ... EIGENVALUES OF HCORE.
C     H     ... DESTROYED BY ROUTINE DIAGD.
C
      IF(IPRGES.EQ.1) THEN
        WRITE(IOUT,2030)
        CALL EIGOUT(HVEC,HEIG,MAXBAS,IAN,IBTOA,NBASIS,1,NBASIS)
      END IF
C
C     FORM CORE HAMILTONIAN MATRIX.
      IF(METHOD.LE.2) THEN
        CON=PT5
      ELSE
        CON=ONE
      END IF
C
      DO 120 IAT=1,NATOMS
        ZD=ZERO
        MX=IAT*(IAT-1)/2
        DO 90 JAT=1,IAT
          MA=MX+JAT
          ZD=ZD+ZV(JAT)*GAB(MA)
   90   CONTINUE
C
        ZD  =ZD-ZV(IAT)*GAB(MA)
C
        DO 100 JAT=IAT,NATOMS
          MA=JAT*(JAT-1)/2+IAT
          ZD=ZD+ZV(JAT)*GAB(MA)
  100   CONTINUE
C
        GTERM=GAA(IAT)*CON-ZD
        IL   =NLBAS(IAT)
        IU   =NUBAS(IAT)
        DO 110 I=IL,IU
          MA=I*(I+1)/2
          HCORE(MA)=HCORE(MA)+GTERM
  110   CONTINUE
  120 CONTINUE
C
      IF(METHOD.NE.5) GO TO 140
C
C     INDO/S CORRECTION ... ADD 1 CENTER ERIS BY USING SLATER-CONDON
C                           PARAMETER.
C      ... Haven't debugged yet.
C     ONLY S AND P BASIS FUNCTIONS ARE EMPLOYED IN CURRENT VERSION OF
C     MOS-F.
      DO 130 IAT=1,NATOMS
        IA=IAN(IAT)
        IF(IA.LE.2) GO TO 130
C
        SL=DMIN1(TWO,ZV(IAT))
        PM=DMAX1(ZERO,ZV(IAT)-TWO)
C
        IL=NLBAS(IAT)
        MA= IL   *(IL+1)/2
        IX=(IL+1)*(IL+2)/2
        IY=(IL+2)*(IL+3)/2
        IZ=(IL+3)*(IL+4)/2
C
        HCORE(MA)=HCORE(MA)+PM*G1SP(IA)/SIX
        HP       =SL*G1SP(IA)/SIX+TWO*(PM-ONE)*F2PP(IA)/F25
        HCORE(IX)=HCORE(IX)+HP
        HCORE(IY)=HCORE(IY)+HP
        HCORE(IZ)=HCORE(IZ)+HP
C
  130 CONTINUE
C
  140 IF(IPRHCR.EQ.1) THEN
        WRITE(IOUT,2040)
        CALL SSMOUT(HCORE,IAN,IBTOA,NLBAS,NBASIS)
      END IF
C
C     ALL DONE. RETURN TO CALLER.
      CALL CLOCKM('Guess. ')
      RETURN
      END
C
C@(#)=HARMTR(SUB) ... CNDO2/3R(HARMTR)
      SUBROUTINE HARMTR(E,MAXL,T)
C
C     ==============================================================
C     CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R
C     ==============================================================
C     ******************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION T(9,9),E(3)
C     ******************************************************************
      COST = E(3)
      IF((1.D0-COST**2).GT.0.0000000001) GO TO 20
   10 SINT = 0.D0
      GO TO 30
   20 SINT=DSQRT(1.D0-COST**2)
   30 CONTINUE
      IF(SINT.GT.0.000001  )  GO TO 50
   40 COSP = 1.D0
      SINP = 0.D0
      GO TO 70
   50 COSP = E(1)/SINT
   60 SINP = E(2)/SINT
   70 CONTINUE
      DO 80 I=1,9
      DO 80 J=1,9
   80 T(I,J) = 0.D0
      T(1,1) =1.D0
      IF (MAXL.GT.1) GO TO 100
   90 IF (MAXL.GT.0) GO TO 110
      GO TO 120
  100 COS2T = COST**2-SINT**2
      SIN2T = 2.D0*SINT*COST
      COS2P = COSP**2-SINP**2
      SIN2P = 2.D0*SINP*COSP
C     TRANSFORMATION MATRIX ELEMENTS FOR D FUNCTIONS
      SQRT3=DSQRT(3.D0)
      T(5,5) = (3.D0*COST**2-1.D0)/2.D0
      T(5,6) = -SQRT3   *SIN2T/2.D0
      T(5,8) = SQRT3   *SINT**2/2.D0
      T(6,5) = SQRT3   *SIN2T*COSP/2.D0
      T(6,6) = COS2T*COSP
      T(6,7) = -COST*SINP
      T(6,8) =-T(6,5)/SQRT3
      T(6,9) = SINT*SINP
      T(7,5) = SQRT3   *SIN2T*SINP/2.D0
      T(7,6) = COS2T*SINP
      T(7,7) = COST*COSP
      T(7,8) = -T(7,5)/SQRT3
      T(7,9) = -SINT*COSP
      T(8,5) = SQRT3   *SINT**2*COS2P/2.D0
      T(8,6) = SIN2T*COS2P/2.D0
      T(8,7) = -SINT*SIN2P
      T(8,8) = (1.D0+COST**2)*COS2P/2.D0
      T(8,9) = -COST*SIN2P
      T(9,5) = SQRT3   *SINT**2*SIN2P/2.D0
      T(9,6) = SIN2T*SIN2P/2.D0
      T(9,7) = SINT*COS2P
      T(9,8) = (1.D0+COST**2)*SIN2P/2.D0
      T(9,9) = COST*COS2P
  110 CONTINUE
C     TRANSFORMATION MATRIX ELEMENTS FOR P FUNCTIONS
      T(2,2) = COST*COSP
      T(2,3) = -SINP
      T(2,4) = SINT*COSP
      T(3,2) = COST*SINP
      T(3,3) = COSP
      T(3,4) = SINT*SINP
      T(4,2) = -SINT
      T(4,4) = COST
  120 CONTINUE
      RETURN
      END
```

```
C
C@(#)=HYPOL1(SUB)
      SUBROUTINE HYPOL1
C
C     EVALUATION OF 1-ST HYPERPOLARIZABILITY TENSORS FOR
C      1. SECOND HARMONIC GENERATION (SHG),
C      2. ELECTRO-OPTIC POCKELS EFFECT (EOPE),
C      3. OPTICAL RECTIFICATION (OR).
C
C     REFERENCES.
C      J.A.MORRELL AND A.C.ALBRECHT, CHEM.PHYS.LETT.,64,46(1979)
C      S.P.KARNA,P.N.PRASAD ,AND M.DUPUIS, J.CHEM.PHYS.,94,1171(1991)
C      J.E.RICE ET AL, J.CHEM.PHYS.,93,8828(1990)
C      J.F.WARD, REV.MOD.PHYS.,37,1(1965)
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /MULTCI/ MULTCI
      COMMON /DGRND / DXPT,DYPT,DZPT,DXAP,DYAP,DZAP,DX,DY,DZ
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                NOCS,NVCS,NCSFS
      COMMON /WORK01/ TGN(MCICSF,3),TNN(MCIINT,3),DUMMY1(1)
      COMMON /WORK02/ CIEIG(MCICSF),BETA(18),BVEC(18),DUMMY2(1)
 2000 Format(1H ,'<<<<<< Enter Hypo11 >>>>>>')
 2010 Format(1H ,'Frequency dependent 1-st ',
     &           'hyperpolarizability tensors, Beta(i,j,k).'/
     &       1H ,'Unit for beta ... *10**(-30) esu '
     &           '(cm**5/statcoulomb).')
 2020 Format(1H ,'This routine skipped because triplet CI calculation ',
     &           'was performed.')
C     ****************************************************************
      WRITE(IOUT,2000)
      WRITE(IOUT,2010)
C
      IF(MULTCI.EQ.3) THEN
        WRITE(IOUT,2020)
        GO TO 20
      END IF
C
C     NOW, COMMON /WORK01/ TGN(MCICSF,3),TNN(MCIINT,3)
C          COMMON /WORK02/ CIEIG(MCICSF)
C
C     SUBTRACT DIPOLE.
      DO 10 I=1,NCSFS
        MA=I*(I+1)/2
        TNN(MA,1)=TNN(MA,1)-DX
        TNN(MA,2)=TNN(MA,2)-DY
        TNN(MA,3)=TNN(MA,3)-DZ
   10 CONTINUE
C
C     THEN, GET 1-ST HYPERPOLARIZABILITY TENSORS, BETA(I,J,K).
      CALL D1SHG  (TGN,TNN,CIEIG,BETA,BVEC,MCICSF,MCIINT,NCSFS)
      CALL D1EOPE (TGN,TNN,CIEIG,BETA,BVEC,MCICSF,MCIINT,NCSFS)
      CALL D1OR   (TGN,TNN,CIEIG,BETA,BVEC,MCICSF,MCIINT,NCSFS)
C
C     ALL DONE. RETURN TO CALLER.
   20 CALL CLOCKM('Hypo11.')
      RETURN
      END
c
c@(#)=IClear(Sub)
      Subroutine IClear(IArray,ISize)
      Implicit Integer (a-z)
c
c     Clear IArray.
c     On input:
c      IArray ... Integer array to clear.
c      ISize  ... Size of IArray.
c     On output.
c      IArray ... Zero-cleared integer array.
c
      Dimension IArray(*)
c
      Do 10 i=1,ISize
        IArray(i)=0
   10 Continue
c
      Return
      End
C
C@(#)=INTCHR(FUN)
      INTEGER FUNCTION INTCHR(CARD,IB,IERR)
C
C     GET INTEGER TYPE DATA FROM DATA BLOCK IN THE 'CARD'.
C
C     On input:
C      CARD ... CARD CONTAINS DATA BLOCK.
C      IB   ... CHECK DATA BLOCK THAT START FROM IB-TH COLUMN.
C     On output:
C      INTCHR ... INTEGER TYPE DATA.
C
C     ****************************************************************
      CHARACTER*80 CARD
      CHARACTER*1  NUMBER(0:9),PLUS,MINUS,SPACE,CHR1
      DATA NUMBER/'0','1','2','3','4','5','6','7','8','9'/
      DATA PLUS/'+'/,MINUS/'-'/,SPACE/' '/
      DATA MAXCLM/80/
C     ****************************************************************
      IERR  =0
      NBITS =32
      INTMAX=(2**(NBITS-1)-1)/10
      IFLG=0
      IS  =IB
      CHR1=CARD(IS:IS)
      IF(CHR1.EQ.PLUS) THEN
        IS=IS+1
      END IF
      IF(CHR1.EQ.MINUS) THEN
        IFLG=1
        IS=IS+1
      END IF
      ISUM=0
      DO 30 I=IS,MAXCLM
        CHR1=CARD(I:I)
        IF(CHR1.EQ.SPACE) THEN
          GO TO 40
        END IF
        IF(ISUM.GT.INTMAX) THEN
          IERR=1
          RETURN
        END IF
        DO 10 J=0,9
          IF(CHR1.EQ.NUMBER(J)) THEN
            ISUM=ISUM*10+J
            GO TO 20
          END IF
   10   CONTINUE
        CALL MERROR
   20   IF(ISUM.LT.0) THEN
          IERR=1
          RETURN
        END IF
   30 CONTINUE
   40 IF(IFLG.EQ.1) THEN
        ISUM=-ISUM
      END IF
C
      INTCHR=ISUM
C
      RETURN
      END
      subroutine intci0
c=================================================================*
c                                                                 *
c        @    @    @   @@@@@   @@@@      @    @   @              *
c        @    @@   @     @    @    @     @   @     @              *
c        @    @ @  @     @    @          @   @     @              *
c        @    @  @ @     @    @          @   @     @              *
c        @    @   @@     @    @    @     @    @   @               *
c        @    @    @     @     @@@@      @     @@@                *
c=================================================================*
c
c     cndo type integral transformation.
c
c     ****************************************************************
      implicit real*8 (a-h,o-z)
c
      include 'MSSIZE'
      include 'basinf.h'
      include 'mol.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
```

```
      include 'work02.h'
c
      parameter( mdim2 = maxian*65 + 1 )
      common /io     / in,iout,ifile(10)
      common/ gamma  / gab( mapck ), gaa( maxat )
      data zero/ 0.0d+0 /
c     ****************************************************************

c---  get list vector nu(i,j)
      nu = 0
      do 1 i = 1, nsdci
        do 2 j = 1, i
          nu = nu + 1
          ici1( nu ) = i
          ici2( nu ) = j
    2   continue
    1 continue
      numax = nu

c
c---  get general molecular integrals
      mu = 0
      do 3 nu = 1, numax
        i  = ici1( nu )
        j  = ici2( nu )
        ix = lmo( i )
        jx = lmo( j )
        ii = nbasis*( ix - 1 )
        jj = nbasis*( jx - 1 )
        do 4 nux = 1, nu
          mu = mu + 1
          k  = ici1( nux )
          l  = ici2( nux )
          kx = lmo( k )
          lx = lmo( l )
          kk = nbasis*( kx - 1 )
          ll = nbasis*( lx - 1 )
          gint00 = zero
          do 5 ir0 = 1, nbasis
            irat = ibtoa( ir0 )
            am = zero
            do 6 it0 = 1, nbasis
              itat = ibtoa( it0 )
              irtx = max0( irat, itat )
              irty = min0( irat, itat )
              irt  = irtx*( irtx - 1 )/2 + irty
              am   = am + c( kk + it0 )*c( ll + it0 )*gab( irt )
    6       continue
            gint00 = gint00 + c( ii + ir0 )*c( jj + ir0 )*am
    5     continue
          gintgl( mu ) = gint00
    4   continue
    3 continue
      mumax = mu
c
c
      return
      end
C
C@(#)=INTCI1(SUB)
      SUBROUTINE INTCI1
C
C     CNDO TYPE INTEGRAL TRANSFORMATION.
C
C     TWO TYPES OF MOLECULAR INTEGRALS ARE NEEDED IN SCI CALCULATION.
C     TYPE 1 ... (IA/JB) OVOV.
C     TYPE 2 ... (IJ/AB) OOVV.
C     I,J ... OCCUPIED MO (O).
C     A,B ... VACANT MO (V).
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER*4 A,B,P,Q,AA,BB,BMAX
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      PARAMETER(MDIM1=MAXBAS*(MCIOCC+1)+1)
      PARAMETER(MDIM2=MAXIAN*65+1)
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /IOPCI  / IOPCI(18)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                 IBTOA(MAXBAS)
      COMMON /GAMMA / GAB(MAPCK),GAA(MAXAT)
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                 NOCS,NVCS,NCSFS
      COMMON /WORK01/ OVOV(MCIINT),OOVV(MCIINT),AQ(MAXBAS),
     &                 AQJ(MAXBAS,MCIOCC),DUMMY(MDIM1)
      COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),IDUMMY(MDIM2)
      DATA ZERO/0.0D+00/
C     ****************************************************************
      MULTCI=IOPCI( 1)
      IF(MULTCI.EQ.3) GO TO 80
C
C     P AND Q (INTEGER*4) REPRESENT BASIS FUNCTIONS IN THE DO LOOPS
C     CODED BELOW.
C                                                          ------------
C                                                          (IA/JB) OVOV
C                                                          ------------
      N=0
      DO 70 I=1,NOCS
        II=NBASIS*(IOMO(I)-1)
        DO 60 A=1,NVCS
          AA=NBASIS*(IVMO(A)-1)
C
C     GET A(Q,I,A) (SUM OVER P).
          DO 20 Q=1,NBASIS
            IQAT=IBTOA(Q)
            AM=ZERO
            DO 10 P=1,NBASIS
              IPAT=IBTOA(P)
              IF(IPAT.GT.IQAT) THEN
                IPQ=IPAT*(IPAT-1)/2+IQAT
              ELSE
                IPQ=IQAT*(IQAT-1)/2+IPAT
              END IF
              AM=AM+GAB(IPQ)*C(II+P)*C(AA+P)
   10       CONTINUE
            AQ(Q)=AM
   20     CONTINUE
C
C     GET (IA/JB) (SUM OVER Q).
          DO 50 J=1,I
            JJ=NBASIS*(IOMO(J)-1)
C
            IF(J.EQ.I) THEN
              BMAX=A
            ELSE
              BMAX=NVCS
            END IF
C
            DO 40 B=1,BMAX
              BB=NBASIS*(IVMO(B)-1)
              N =N+1
              AM=ZERO
              DO 30 Q=1,NBASIS
                AM=AM+AQ(Q)*C(JJ+Q)*C(BB+Q)
   30         CONTINUE
              OVOV(N)=AM
   40       CONTINUE
   50     CONTINUE
   60   CONTINUE
   70 CONTINUE
C                                                          ------------
C                                                          (IJ/AB) OOVV
C                                                          ------------
   80 N=0
      DO 160 I=1,NOCS
        II=NBASIS*(IOMO(I)-1)
        DO 110 J=1,I
          JJ=NBASIS*(IOMO(J)-1)
C
C     GET A(Q,J,I) (SUM OVER P).
          DO 100 Q=1,NBASIS
            IQAT=IBTOA(Q)
            AM  =ZERO
            DO 90 P=1,NBASIS
              IPAT=IBTOA(P)
              IF(IPAT.GT.IQAT) THEN
                IPQ=IPAT*(IPAT-1)/2+IQAT
              ELSE
                IPQ=IQAT*(IQAT-1)/2+IPAT
              END IF
              AM=AM+GAB(IPQ)*C(II+P)*C(JJ+P)
   90       CONTINUE
            AQJ(Q,J)=AM
  100     CONTINUE
```

```
  110   CONTINUE
C
C     GET (IJ/AB) (SUM OVER Q).
        DO 150 A=1,NVCS
          AA=NBASIS*(IVMO(A)-1)
          DO 140 J=1,I
C
            IF(J.EQ.I) THEN
              BMAX=A
            ELSE
              BMAX=NVCS
            END IF
C
            DO 130 B=1,BMAX
              BB=NBASIS*(IVMO(B)-1)
              N =N+1
              AM=ZERO
              DO 120 Q=1,NBASIS
                AM=AM+AQJ(Q,J)*C(AA+Q)*C(BB+Q)
  120         CONTINUE
              OOVV(N)=AM
  130       CONTINUE
  140     CONTINUE
  150   CONTINUE
  160 CONTINUE
C
      RETURN
      END
C
C@(#)=IPOS(SUB)
      SUBROUTINE IPOS(CHR1,LENCHR,IP)
C
C     ************************************************************
      CHARACTER*800 OPTCRD,Crd2
      CHARACTER*80  ICARD,iCrd2
      CHARACTER*1   CHR1(*),IBL1
      COMMON /CARD1 / OPTCRD,ICARD,Crd2,iCrd2
      COMMON /CARD2 / MAXCRD,LENOCR,LENICR,NCARDS,ISMIN,IEMAX
      DATA IBL1/' '/
C     ************************************************************
      DO 10 I=1,LENCHR
        ICARD(I:I)=CHR1(I)
   10 CONTINUE
C
      IP=INDEX(OPTCRD,ICARD(1:LENCHR))
C
      IF(IP.NE.0) THEN
        IPEND=IP+LENCHR-1
        DO 20 I=IP,IPEND
          OPTCRD(I:I)=IBL1
   20   CONTINUE
      END IF
C
      RETURN
      END
C
C@(#)=IREAD(SUB)
      SUBROUTINE IREAD(IVAL,NWRDS,MAXDIM)
C
C     READ INTEGER TYPE DATA.
C
C     On input:
C      MAXDIM ... MAXIMUM SIZE OF ARRAY IVAL.
C     On output:
C      IVAL(I) ... I-TH INTEGER TYPE DATA.
C      NWRDS ... NO OF DATA.
C
C     NOTES.
C      1. THIS ROUTINE IS TERMINATED BY A BLANK CARD OR EOF.
C      2. FREE FORMAT WITH COMMA OR SPACE DELIMITER.
C      3. LENGTH OF ONE WORD IS 4 BYTES.
C      4. LENGTH OF ONE RECORD IS 80 BYTES.
C     ************************************************************
      CHARACTER*80 CARD
      INTEGER*4 CHKTYP
      COMMON /IO    /IN,IOUT,IFILE(10)
      DIMENSION IVAL(*)
      DIMENSION IBLK(41)
 9910 Format(1H ,36('X')/
     &       1H ,'CARD CONTAINS NON-INTEGER TYPE DATA.'/
     &       1H ,36('X'))
 9920 Format(1H ,45('X')/
     &       1H ,'NO. OF DATA EXCEED MAXIMUM SIZE. MDIM =',I6/
     &       1H ,45('X'))
 9930 Format(1H ,40('X')/
     &       1H ,'CARD CONTAINS TOO LARGE INTEGER TYPE NO.'/
     &       1H ,40('X'))
 9940 Format(1H ,28('X')/
     &       1H ,'CARD CONTAINS NO DATA BLOCK.'
     &       1H ,28('X'))
C     ************************************************************
      N=0
C
   10 CALL RDCARD(CARD,NBLKS,IBLK)
      IF(NBLKS.EQ.0) THEN
        GO TO 30
      END IF
      DO 20 I=1,NBLKS
        IB =IBLK(I)
        IRC=CHKTYP(CARD,IB)
        IF(IRC.NE.1) THEN
          WRITE(IOUT,9910)
          CALL MERROR
        END IF
        N=N+1
        IF(N.GT.MAXDIM) THEN
          WRITE(IOUT,9920) MAXDIM
          CALL MERROR
        END IF
        IVAL(N)=INTCHR(CARD,IB,IERR)
        IF(IERR.EQ.1) THEN
          WRITE(IOUT,9930)
          CALL MERROR
        END IF
   20 CONTINUE
      GO TO 10
   30 IF(N.EQ.0) THEN
        WRITE(IOUT,9940)
        CALL MERROR
      END IF
      NWRDS=N
C
      RETURN
      END
C
c@(#)=IToC(Sub)
      Subroutine IToC(Int,Chr)
      Implicit Integer (a-z)
c
c     Convert integer to character.
c     On input:
c      Int    ... Integer variable.
c     On output:
c      Chr    ... Character variable corresponding to Int.
c
      Character*(*) Chr
      Character*1 iB1,CNumbr(0:9),Minus
      Save iB1,Minus
      Data iB1/1h /,Minus/1h-/
      Save CNumbr
      Data CNumbr/1h0,1h1,1h2,1h3,1h4,1h5,1h6,1h7,1h8,1h9/
c
      Chr=iB1
c
      LenChr=Len(Chr)
c
      Do 10 i=1,LenChr
        If ((Int/10**i) .eq. 0) Then
          NFigs=i
          GoTo 20
        EndIf
   10 Continue
c
      Stop 'Abort in IToC. Incredible number.'
c
   20 n=0
      If(Int .lt. 0) Then
        Chr(1:1)=Minus
        Int=IAbs(Int)
        n=1
      EndIf
c
      Do 30 i=1,NFigs
        j=Fld(Int,i,NFigs)
        n=n+1
        Chr(n:n)=CNumbr(j)
```

```
   30 Continue
c
      Return
      End
C
C@(#)=JCTRL(SUB)
      SUBROUTINE JCTRL(IRDEND,IfCI,IfBeta)
C
C     SET COMPUTATIONAL PROCEDURE AND OPTIONS.
C     THIS PROGRAM IS COMPOSED OF 3 SUBPROGRAMS LISTED BELOW.
C      1. RDOPT  ... READ OPTION CARDS.
C      2. OPTSET ... SETTING OF OPTIONS.
C      3. PRTOPT ... PRINTING OF OPTIONS.
C     ****************************************************************
      Logical IfCI,IfBeta
      COMMON /IO    / IN,IOUT,IFILE(10)
 2000 Format(1H ,'<<<<<< Enter JCtrl >>>>>>')
C     ****************************************************************
      IRDEND=0
      IfCI = .True.
      IfBeta = .True.
C
      WRITE(IOUT,2000)
C
      CALL RDOPT(IRDEND)
C
      IF(IRDEND.EQ.1) THEN
        GO TO 10
      END IF
C
      CALL OPTSET(IPRIOP,IfCI,IfBeta)
      CALL PRTOPT(IPRIOP,IfCI)
C
C     ALL DONE. RETURN TO CALLER.
   10 CALL CLOCKM('JCtrl. ')
      RETURN
      END
      subroutine 11cn00
c=====================================================================*
c                 @                       @@@     @@@                 *
c      @         @@      @@@@   @    @  @   @   @   @                *
c      @        @ @     @    @  @@   @ @     @ @     @                *
c      @          @     @       @ @  @ @     @ @     @                *
c      @          @     @       @  @ @ @     @ @     @                *
c      @          @     @    @  @   @@  @   @   @   @                *
c      @@@@@@  @@@@@    @@@@   @    @   @@@     @@@                  *
c=====================================================================*
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'basinf.h'
      include 'mol.h'
      include 'tmomnt.h'
      include 'window.h'
      include 'work02.h'
c
      common/ velec  / zv( maxat )
      common/ io     / in,     iout,   ifile( 10 )
      common/ expnt  / zeta( maxat )
      common/ qnoinf / nc(maxian,2), lc(9), mc(9)
c
      dimension dip( 9, 9 )
c
c----  contribution from nuclears
c
      tlx = 0.0d0
      tly = 0.0d0
      tlz = 0.0d0
c
      do 1 iat = 1, natoms
        zvi = zv( iat )
        cxi = zvi*coord( 1, iat )
        cyi = zvi*coord( 2, iat )
        czi = zvi*coord( 3, iat )
        tlx = tlx + cxi
        tly = tly + cyi
        tlz = tlz + czi
    1 continue
c
c----  contribution from occupied mo's
c
      do 2 ip0 = 1, nae
        do 3 iat = 1, natoms
          rx = coord( 1, iat )
          ry = coord( 2, iat )
          rz = coord( 3, iat )
          ia = ian( iat )
          ncsp = nc( ia, 1 )
          ncd  = nc( ia, 2 )
          il = nlbas( iat )
          iu = nubas( iat )
          ni = nibas( iat )
          zsp = zeta( iat )
          zd  = zeta( iat )
          call getdx( rx,     ni,     zsp,    zd,     ncsp,   ncd,
     $                dip     )
          do 4 k = 1, ni
            it0 = il + k - 1
            ipt = ( ip0 - 1 )*nbasis + it0
            do 5 kp = 1, ni
              iu0 = il + kp - 1
              dtu = 2.0d0*dip( k, kp )
              ipu = ( ip0 - 1 )*nbasis + iu0
              tlx = tlx + dtu*c( ipt )*c( ipu )
    5       continue
    4     continue
          call getdy( ry,     ni,     zsp,    zd,     ncsp,   ncd,
     $                dip     )
          do 6 k = 1, ni
            it0 = il + k - 1
            ipt = ( ip0 - 1 )*nbasis + it0
            do 7 kp = 1, ni
              iu0 = il + kp - 1
              dtu = 2.0d0*dip( k, kp )
              ipu = ( ip0 - 1 )*nbasis + iu0
              tly = tly + dtu*c( ipt )*c( ipu )
    7       continue
    6     continue
          call getdz( rz,     ni,     zsp,    zd,     ncsp,   ncd,
     $                dip     )
          do 8 k = 1, ni
            it0 = il + k - 1
            ipt = ( ip0 - 1 )*nbasis + it0
            do 9 kp = 1, ni
              iu0 = il + kp - 1
              dtu = 2.0d0*dip( k, kp )
              ipu = ( ip0 - 1 )*nbasis + iu0
              tlz = tlz + dtu*c( ipt )*c( ipu )
    9       continue
    8     continue
    3   continue
    2 continue
c
c
      call clockm('11cn00.')
c
c
      return
      end
      subroutine 1zdo00
c=====================================================================*
c                                         @@@     @@@                 *
c      @       @@@@@@  @@@@@    @@@@   @   @   @   @                 *
c      @           @   @    @  @    @  @ @     @ @     @               *
c      @          @    @    @  @    @  @ @     @ @     @               *
c      @         @     @    @  @    @  @ @     @ @     @               *
c      @        @      @    @  @    @  @ @     @ @     @               *
c      @       @       @    @  @    @  @   @   @   @                 *
c      @@@@@@  @@@@@@  @@@@@    @@@@    @@@     @@@                   *
c=====================================================================*
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'basinf.h'
      include 'mol.h'
      include 'tmomnt.h'
      include 'window.h'
      include 'work02.h'
c
      common/ velec / zv( maxat )
      common/ io    / in,     iout,   ifile( 10 )
c
c----  contribution from nuclears
c
      tlx = 0.0d0
      tly = 0.0d0
      tlz = 0.0d0
c
      do 1 iat = 1, natoms
```

```
        zvi = zv( iat )
        cxi = zvi*coord( 1, iat )
        cyi = zvi*coord( 2, iat )
        czi = zvi*coord( 3, iat )
        tlx = tlx + cxi
        tly = tly + cyi
        tlz = tlz + czi
    1 continue
c
c----  contribution from occupied mo's
c
      do 2 ip0 = 1, nae
        do 3 it0 = 1, nbasis
          iat = ibtoa( it0 )
          cxi = 2.0d0*coord( 1, iat )
          cyi = 2.0d0*coord( 2, iat )
          czi = 2.0d0*coord( 3, iat )
          ipt = ( ip0 - 1 )*nbasis + it0
          tlx = tlx + cxi*c( ipt )*c( ipt )
          tly = tly + cyi*c( ipt )*c( ipt )
          tlz = tlz + czi*c( ipt )*c( ipt )
    3   continue
    2 continue
c
c
      call clockm('lzdo00.')
c
c
      return
      end
C
C@(#)=MERROR(SUB)
      SUBROUTINE MERROR
C
C     MOS-F DETECT ERROR. TERMINATE JOB.
C     ****************************************************************
      COMMON /IO     / IN,IOUT,IFILE(10)
      DATA I0/0/,I1/1/
 9000 Format(1H ,44('#')/
     &       1H ,'# MOS-F DETECT ERROR. JOB STOPS ',
     &                'EXECUTING. #'/
     &       1H ,44('#'))
C     ****************************************************************
      WRITE(IOUT,9000)
C
C     TRACE BACK ... ONLY AVAILABLE FOR MAINFRAME COMPUTER.
      I=I0
CSUN  I=I1
      I=I0+I1/I
C
      STOP
      END
C
C@(#)=MINPUT(SUB)
      SUBROUTINE MINPUT(JRDEND)
C
C     INPUT OPTIONS AND MOLECULAR INFORMATIONS.
C     THIS PROGRAM IS COMPOSED OF 4 SUBPROGRAMS LISTED BELOW.
C      1. RDTITL ... READ TITLE CARDS.
C      2. RDGEOM ... GET GEOMETRICAL INFORMATION AND PRINT IT OUT.
C      3. PRMSET ... SETTING OF ATOMIC PARAMETERS.
C      4. PCNOE  ... PCNOE CALCULATION.
C     ****************************************************************
      COMMON /IO     / IN,IOUT,IFILE(10)
 2000 Format(1H ,'<<<<<< Enter MInput >>>>>>')
C     ****************************************************************
      JRDEND=0
C
      WRITE(IOUT,2000)
C
      CALL RDTITL(JRDEND)
C
      IF(JRDEND.GE.1) THEN
        GO TO 10
      END IF
C
      CALL RDGEOM
      CALL PRMSET
      CALL PCNOE
C
C     ALL DONE. RETURN TO CALLER.
   10 CALL CLOCKM('MInput.')
      RETURN
      END
      subroutine mkcibs
c=====================================================================*
c                                                                     *
c      @    @  @    @   @@@@      @    @@@@@    @@@@                 *
c      @@  @@  @   @   @    @     @    @      @  @                   *
c      @ @@ @  @@@@    @          @    @@@@@    @@@@                 *
c      @    @  @  @    @          @    @    @       @                *
c      @    @  @   @   @    @     @    @    @  @    @                *
c      @    @  @    @   @@@@      @    @@@@@    @@@@                 *
c=====================================================================*
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'window.h'
c$ohtawara
      common / io / in, iout, ifile(10)

      nsdci = nocs + nvcs

c---- hartree-fock solution
      nomg = 1
      mcio( nomg ) = 0
      lci1( nomg ) = 0
      lci2( nomg ) = 0
      lci3( nomg ) = 0
      lci4( nomg ) = 0
      momega( 1 ) = nomg

c---- single excitation
      do 1 ig = 1, nocs
         do 2 ih = 1, nvcs
            nomg = nomg + 1
            if( nomg .gt. msdcib ) go to 9999
            mcio( nomg ) = 1
            lci1( nomg ) = ig
            lci2( nomg ) = 0
            lci3( nomg ) = ih
            lci4( nomg ) = 0
 2       continue
 1    continue
      momega( 2 ) = nomg
      if( isdci0 .eq. 0 ) go to 9000

c---- double excitation ( type I )
      do 3 ig = 1, nocs
         do 4 ih = 1, nvcs
            nomg = nomg + 1
            if( nomg .gt. msdcib ) go to 9999
            mcio( nomg ) = 2
            lci1( nomg ) = ig
            lci2( nomg ) = ig
            lci3( nomg ) = ih
            lci4( nomg ) = ih
 4       continue
 3    continue
      momega( 3 ) = nomg
      if( isdci0 .eq. 1 ) go to 9000
      if( isdci0 .eq. 2 ) go to 1000

c---- double excitation ( type II )
      do 5 ig = 2, nocs
         do 6 igx = 1, ig-1
            do 7 ih = 1, nvcs
               nomg = nomg + 1
               if( nomg .gt. msdcib ) go to 9999
               mcio( nomg ) = 3
c$ohtawara
               lci1( nomg ) = igx
               lci2( nomg ) = ig
c               lci1( nomg ) = ig
c               lci2( nomg ) = igx

               lci3( nomg ) = ih
               lci4( nomg ) = ih
 7          continue
 6       continue
 5    continue
      momega( 4 ) = nomg
 1000 continue

c---- double excitation ( type III )
```

```
      do 8 ig = 1, nocs
         do 9 ih = 2, nvcs
            do 10 ihx = 1, ih-1
               nomg = nomg + 1
               if( nomg .gt. msdcib ) go to 9999
               mcio( nomg ) = 4
               lci1( nomg ) = ig
               lci2( nomg ) = ig
c$ohtawara
               lci3( nomg ) = ihx
               lci4( nomg ) = ih
c               lci3( nomg ) = ih
c               lci4( nomg ) = ihx

 10         continue
 9       continue
 8    continue
      momega( 5 ) = nomg
      if( isdci0 .eq. 2 ) go to 9000

c---- double excitation ( type IV )
      do 11 ig = 2, nocs
         do 12 igx = 1, ig-1
            do 13 ih = 1, nvcs
               do 14 ihx = 1, ih-1
                  if( ih .ne. ihx ) then
                     nomg = nomg + 1
                     if( nomg .gt. msdcib ) go to 9999
                     mcio( nomg ) = 5
c$ohtawara
                     lci1( nomg ) = igx
                     lci2( nomg ) = ig
                     lci3( nomg ) = ihx
                     lci4( nomg ) = ih
c                     lci1( nomg ) = ig
c                     lci2( nomg ) = igx
c                     lci3( nomg ) = ih
c                     lci4( nomg ) = ihx

                  endif
 14             continue
 13          continue
 12       continue
 11    continue
      momega( 6 ) = nomg

c---- double excitation ( type V )
      do 15 ig = 2, nocs
         do 16 igx = 1, ig-1
            do 17 ih = 1, nvcs
               do 18 ihx = 1, ih-1
                  if( ih .ne. ihx ) then
                     nomg = nomg + 1
                     if( nomg .gt. msdcib ) go to 9999
                     mcio( nomg ) = 6
c$ohtawara
                     lci1( nomg ) = igx
                     lci2( nomg ) = ig
                     lci3( nomg ) = ihx
                     lci4( nomg ) = ih
c                     lci1( nomg ) = ig
c                     lci2( nomg ) = igx
c                     lci3( nomg ) = ih
c                     lci4( nomg ) = ihx

                  endif
 18             continue
 17          continue
 16       continue
 15    continue
      momega( 7 ) = nomg
      go to 9000

 9999 continue
      write(iout,*)' **** number of CI bases exceeding msdcib'
      stop

 9000 continue
      ncibas = nomg
      ncimat = ncibas*( ncibas + 1 )/2

cc$ohtawara
cc sort CI configuration agree with ZINDO
```

```
cc       do i=1,ncibas
cc          tmp(lci1(i),lci2(i),lci3(i),lci4(i))=1.0
cc          write(iout,100) i,lci1(i),lci2(i),lci3(i),lci4(i)
cc       enddo
cc 100   format(i3,2x,4i3)
cc
cc       j = 5
cc       do i1 = 1,2
cc        do i2 = 1,2
cc         do i3 = 1,2
cc          do i4 = 1,2
ccc                 write(iout,110) i1,i2,i3,i4
cc                if (tmp(i1,i2,i3,i4) .eq. 1.0) then
cc                   j=j+1
cc                   lci1(j) = i1
cc                   lci2(j) = i2
cc                   lci3(j) = i3
cc                   lci4(j) = i4
cc       if(i1.eq.i3 .and. i2.eq.i4 .and. i1.ne.i2 .and. i1.eq.1.0) then
cc          j=j+1
cc          lci1(j) = i1
cc          lci2(j) = i2
cc          lci3(j) = i3
cc          lci4(j) = i4
cc       endif
cc               endif
cc             enddo
cc           enddo
cc         enddo
cc       enddo
cc       jmax = j
cc       do j=1,jmax
cc          write(iout,100) j,lci1(j),lci2(j),lci3(j),lci4(j)
cc       enddo
cc 110   format(4i3)

      return
      end
C
C@(#)=MOINFO(SUB)
      SUBROUTINE MOINFO
C
C     GET WINDOW INFORMATION FOR INTEGRAL TRANSFORMATION.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      PARAMETER(MAXDIM=MAXIAN*65+1)
      COMMON /IOPCI / IOPCI(18)
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /MOL   / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                NAE,NBE,NE,NBASIS
c========( Fri Sep 15 14:03:28 JST 1995 )======
c$$$      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
c$$$     &                NOCS,NVCS,NCSFS
      include 'window.h'
c========( Fri Sep 15 14:03:33 JST 1995 )======
      COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),IDUMMY(MAXDIM)
      DIMENSION IVAL(MCIMOS)
 2000 Format(1H ,'Window information'/
     &       1H ,'Active MO range ...')
 2010 Format(1H ,'Occupied : ',I4,' TO ',I4,
     &         ',  Virtual : ',I4,' TO ',I4)
 2020 Format(1H ,'Occupied : ',I4,' MOs')
 2030 Format(1H ,I4,15(1X,I4))
 2040 Format(1H ,'Virtual  : ',I4,' MOs')
 9910 Format(1H ,50('X')/
     &       1H ,'SCI CALCULATION CANCELLED BECAUSE OF NCSFS .LT. 1.'/
     &       1H ,50('X'))
 9920 Format(1H ,49('X')/
     &       1H ,'NO. OF ACTIVE MOS EXCEED MAXIMUM SIZE. MDIM =',I4/
     &       1H ,49('X'))
 9930 Format(1H ,40('X')/
     &       1H ,I4,' MO NUMBER OUT OF RANGE ... IGNORED.'/
     &       1H ,40('X'))
 9940 Format(1H ,49('X')/
     &       1H ,'SAME MO NUMBER EXIST IN INPUT STREAM ... IGNORED.'/
     &       1H ,49('X'))
C     ****************************************************************
      NOCI=IOPCI( 2)
      NVCI=IOPCI( 3)

      WRITE(IOUT,2000)
```

```
      IHOMO=NAE
      MOMO =IHOMO
      MVMO =NBASIS-IHOMO
C                                                       ---------------
C                                                       STANDARD WINDOW
C                                                       ---------------
      IF(NOCI.NE.0) THEN
        NOCS=NOCI
        NVCS=NVCI
        IF(NOCS.GT.MOMO) THEN
          NOCS=MOMO
        END IF
        IF(NVCS.GT.MVMO) THEN
          NVCS=MVMO
        END IF
        NCSFS=NOCS*NVCS
        IF(NCSFS.LT.1) THEN
          WRITE(IOUT,9910)
          CALL MERROR
        END IF
        N=IHOMO-NOCS+1
        DO 10 I=1,NOCS
          IOMO(I)=N
          LMO(I) =N
          N=N+1
   10   CONTINUE
        N=IHOMO+1
        DO 20 I=1,NVCS
          IVMO(I)=N
          LMO(I+NOCS)=N
          N=N+1
   20   CONTINUE
C
C     PRINTING OF WINDOW INFORMATION.
        WRITE(IOUT,2010) IOMO(1),IOMO(NOCS),IVMO(1),IVMO(NVCS)
C
        RETURN
C
      END IF
C                                                       ---------------
C                                                       OPTIONAL WINDOW
C     READ ACTIVE MOS FROM CARD.                        ---------------
      CALL IREAD(IVAL,NWRDS,MCIMOS)
C
      NO   =0
      NV   =0
      NERRS=0
      DO 30 I=1,NWRDS
        IMO=IVAL(I)
        IF( (IMO.LE.IHOMO) .AND. (IMO.GT.0) ) THEN
          NO=NO+1
          IF(NO.GT.MCIOCC) THEN
            WRITE(IOUT,9920) MCIOCC
            CALL MERROR
          END IF
          IOMO(NO)=IMO
        ELSE IF( (IMO.GT.IHOMO) .AND. (IMO.LE.NBASIS) ) THEN
          NV=NV+1
          IF(NV.GT.MCIVAC) THEN
            WRITE(IOUT,9920) MCIVAC
            CALL MERROR
          END IF
          IVMO(NV)=IMO
        ELSE
          NERRS=NERRS+1
        END IF
   30 CONTINUE
C
      IF(NERRS.GT.0) THEN
        WRITE(IOUT,9930)
      END IF
C
C     BEFORE SORTING, CHECK THE EXISTENCE OF THE SAME MO NUMBER.
      NM1=NO-1
      N  =0
      IF(NM1.NE.0) THEN
        DO 50 I=1,NM1
          IMO=IOMO(I)
          IP1=I+1
          DO 40 J=IP1,NO
            IF(IOMO(J).EQ.IMO) GO TO 50
   40     CONTINUE
          N=N+1
          IOMO(N)=IOMO(I)
   50   CONTINUE
        N=N+1
        IOMO(N)=IOMO(NO)
        NOCS=N
      ELSE
        NOCS=NO
      END IF
C
      NM1=NV-1
      N  =0
      IF(NM1.NE.0) THEN
        DO 70 I=1,NM1
          IMO=IVMO(I)
          IP1=I+1
          DO 60 J=IP1,NV
            IF(IVMO(J).EQ.IMO) GO TO 70
   60     CONTINUE
          N=N+1
          IVMO(N)=IVMO(I)
   70   CONTINUE
        N=N+1
        IVMO(N)=IVMO(NV)
        NVCS=N
      ELSE
        NVCS=NV
      END IF
C
      NCSFS=NOCS*NVCS
      IF(NCSFS.LT.1) THEN
        WRITE(IOUT,9910)
        CALL MERROR
      END IF
C
C     THEN SORTING.
      NM1=NOCS-1
      IF(NM1.NE.0) THEN
        DO 90 I=1,NM1
          IMO=IOMO(I)
          IP1=I+1
          DO 80 J=IP1,NOCS
            IF(IOMO(J).LT.IMO) THEN
              JMO=IOMO(J)
              IOMO(J)=IMO
              IMO=JMO
            END IF
   80     CONTINUE
          IOMO(I)=IMO
   90   CONTINUE
      END IF
C
      NM1=NVCS-1
      IF(NM1.NE.0) THEN
        DO 110 I=1,NM1
          IMO=IVMO(I)
          IP1=I+1
          DO 100 J=IP1,NVCS
            IF(IVMO(J).LT.IMO) THEN
              JMO=IVMO(J)
              IVMO(J)=IMO
              IMO=JMO
            END IF
  100     CONTINUE
          IVMO(I)=IMO
  110   CONTINUE
      END IF
C
      IF( (NOCS.LT.NO) .OR. (NVCS.LT.NV) ) THEN
        WRITE(IOUT,9940)
      END IF
C
      DO 120 I=1,NOCS
        LMO(I)=IOMO(I)
  120 CONTINUE
      DO 130 I=1,NVCS
        LMO(I+NOCS)=IVMO(I)
  130 CONTINUE
C
C     PRINTING OF WINDOW INFORMATION.
      WRITE(IOUT,2020) NOCS
      WRITE(IOUT,2030) (IOMO(I),I=1,NOCS)
```

```
      WRITE(IOUT,2040) NVCS
      WRITE(IOUT,2030) (IVMO(I),I=1,NVCS)
C
      RETURN
      END
C
C@(#)=INTCI2(SUB)
c     SUBROUTINE INTCI2
C
C     INDO TYPE INTEGRAL TRANSFORMATION.
c      ... Haven't debugged yet.
C
C     TWO TYPES OF MOLECULAR INTEGRALS ARE NEEDED IN SCI CALCULATION.
C     TYPE 1 ... (IA/JB) OVOV.
C     TYPE 2 ... (IJ/AB) OOVV.
C     I,J ... OCCUPIED MO (O).
C     A,B ... VACANT MO (V).
C     ****************************************************************
c     IMPLICIT REAL*8 (A-H,O-Z)
c     INTEGER*4 A,B,P,Q,AA,BB,BMAX
cfj   INCLUDE (MSSIZE)
c     INCLUDE 'MSSIZE'
c     PARAMETER(MAXI16=MAXIAN*16)
c     COMMON /IO     / IN,IOUT,IFILE(10)
c     COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
c    &                 NAE,NBE,NE,NBASIS
c     COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
c    &                 IBTOA(MAXBAS)
c     COMMON /GAMMA / GAB(MAPCK),GAA(MAXAT)
c     COMMON /SLACON/ G1SP(30),F2PP(30),G2SD(11:30),G1PD(11:30),
c    &                 F2PD(11:30),G3PD(11:30),F2DD(11:30),F4DD(11:30),
c    &                 MAXSCP
c     COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
c    &                 NOCS,NVCS,NCSFS
c     COMMON /WORK01/ OVOV(MCIINT),OOVV(MCIINT),AQ(MAXBAS),BQ(MAXBAS),
c    &                 AQJ(MAXBAS,MCIOCC),BQJ(MAXBAS,MCIOCC),DUMMY1(1)
c     COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),COULMB(MAXI16),
c    &                 EXCHNG(MAXI16),IFLG(MAXIAN),IDUMMY(1)
c     DATA ZERO/0.0D+00/,ONE/1.0D+00/,TWO/2.0D+00/,THREE/3.0D+00/,
c    &     FOUR/4.0D+00/,F25/2.5D+01/
c     DATA N1ERIS/16/,NBMAX/4/
C     ****************************************************************
c     ONETHR=ONE  /THREE
c     F2P25 =TWO  /F25
c     F3P25 =THREE/F25
c     F4P25 =FOUR /F25
C
C     GET 1 CENTER ERIS.
c     DO 10 I=1,MAXIAN
c       IFLG(I)=0
c 10  CONTINUE
C
c     DO 20 IAT=1,NATOMS
c       NI=NIBAS(IAT)
c       IA=IAN(IAT)
C
c       IF(NI.GT.NBMAX) THEN
c         STOP
c       END IF
c       IF(IA.LT.3)      GO TO 20
c       IF(IFLG(I).EQ.1) GO TO 20
C
c       IFLG(IA)=1
C
c       IATX= N1ERIS*(IAT-1)
c       PPPP= F2PP(IA)*F4P25
c       PPQQ=-F2PP(IA)*F2P25
c       SPSP= G1SP(IA)*ONETHR
c       PQPQ= F2PP(IA)*F3P25
C
c       COULMB(IATX+ 6)=PPPP
c       COULMB(IATX+11)=PPPP
c       COULMB(IATX+16)=PPPP
C
c       COULMB(IATX+ 7)=PPQQ
c       COULMB(IATX+ 8)=PPQQ
c       COULMB(IATX+10)=PPQQ
c       COULMB(IATX+12)=PPQQ
c       COULMB(IATX+14)=PPQQ
c       COULMB(IATX+15)=PPQQ
C
c       COULMB(IATX+ 1)=ZERO
c       COULMB(IATX+ 2)=ZERO
c       COULMB(IATX+ 3)=ZERO
c       COULMB(IATX+ 4)=ZERO
c       COULMB(IATX+ 5)=ZERO
c       COULMB(IATX+ 9)=ZERO
c       COULMB(IATX+13)=ZERO
C
c       EXCHNG(IATX+ 2)=SPSP
c       EXCHNG(IATX+ 3)=SPSP
c       EXCHNG(IATX+ 4)=SPSP
c       EXCHNG(IATX+ 5)=SPSP
c       EXCHNG(IATX+ 9)=SPSP
c       EXCHNG(IATX+13)=SPSP
C
c       EXCHNG(IATX+ 7)=PQPQ
c       EXCHNG(IATX+ 8)=PQPQ
c       EXCHNG(IATX+10)=PQPQ
c       EXCHNG(IATX+12)=PQPQ
c       EXCHNG(IATX+14)=PQPQ
c       EXCHNG(IATX+15)=PQPQ
C
c       EXCHNG(IATX+ 1)=ZERO
c       EXCHNG(IATX+ 6)=ZERO
c       EXCHNG(IATX+11)=ZERO
c       EXCHNG(IATX+16)=ZERO
C
c 20 CONTINUE
C
C     P AND Q (INTEGER*4) REPRESENT BASIS FUNCTIONS IN THE DO LOOPS
C     CODED BELOW.
c     N=0
c     DO 120 I=1,NOCS
c       II=NBASIS*(IOMO(I)-1)
c       DO 110 A=1,NVCS
c         AA=NBASIS*(IVMO(A)-1)
C
C     SUM OVER P.
c         DO 40 Q=1,NBASIS
c           IQAT=IBTOA(Q)
c           AM=ZERO
c           DO 30 P=1,NBASIS
c             IPAT=IBTOA(P)
c             IF(IPAT.GT.IQAT) THEN
c               IPQ=IPAT*(IPAT-1)/2+IQAT
c             ELSE
c               IPQ=IQAT*(IQAT-1)/2+IPAT
c             END IF
c             AM=AM+GAB(IPQ)*C(II+P)*C(AA+P)
c 30        CONTINUE
c           AQ(Q)=AM
c 40      CONTINUE
C
c         DO 70 IAT=1,NATOMS
c           IA=IAN(IAT)
c           IF(IA.LT.3) GO TO 70
c           IL=NLBAS(IAT)
c           IU=NUBAS(IAT)
c           NN=N1ERIS*(IA-1)
c           DO 60 Q=IL,IU
c             AM=ZERO
c             BM=ZERO
c             DO 50 P=IL,IU
c               NN=NN+1
c               AM=AM+COULMB(NN)*C(II+P)*C(AA+P)
c               BM=BM+EXCHNG(NN)*C(II+P)*C(AA+P)
c 50          CONTINUE
c             AQ(Q)=AQ(Q)+AM
c             BQ(Q)=BM
c 60        CONTINUE
c 70      CONTINUE
C
C     SUM OVER Q.
c         DO 100 J=1,I
c           JJ=NBASIS*(IOMO(J)-1)
C
c           IF(J.EQ.I) THEN
c             BMAX=A
c           ELSE
c             BMAX=NVCS
c           END IF
C
c           DO 90 B=1,BMAX
c             BB=NBASIS*(IVMO(B)-1)
c             N =N+1
```

```
c             AM=ZERO
c             BM=ZERO
c             DO 80 Q=1,NBASIS
c               AM=AM+AQ(Q)*C(JJ+Q)*C(BB+Q)
c               BM=BM+BQ(Q)*C(JJ+Q)*C(BB+Q)
c  80         CONTINUE
c             OVOV(N)=AM
c             OOVV(N)=BM
c  90       CONTINUE
c 100     CONTINUE
c 110   CONTINUE
c 120 CONTINUE
C
c     N=0
c     DO 230 I=1,NOCS
c       II=NBASIS*(IOMO(I)-1)
c       DO 150 J=1,I
c         JJ=NBASIS*(IOMO(J)-1)
C
C     SUM OVER P.
c         DO 140 Q=1,NBASIS
c           IQAT=IBTOA(Q)
c           AM  =ZERO
c           DO 130 P=1,NBASIS
c             IPAT=IBTOA(P)
c             IF(IPAT.GT.IQAT) THEN
c               IPQ=IPAT*(IPAT-1)/2+IQAT
c             ELSE
c               IPQ=IQAT*(IQAT-1)/2+IPAT
c             END IF
c             AM=AM+GAB(IPQ)*C(II+P)*C(JJ+P)
c 130       CONTINUE
c           AQJ(Q,J)=AM
c 140     CONTINUE
c 150   CONTINUE
C
c         DO 180 IAT=1,NATOMS
c           IA=IAN(IAT)
c           IF(IA.LT.3) GO TO 180
c           IL=NLBAS(IAT)
c           IU=NUBAS(IAT)
c           NN=N1ERIS*(IA-1)
c           DO 170 Q=IL,IU
c             AM=ZERO
c             BM=ZERO
c             DO 160 P=IL,IU
c               NN=NN+1
c               AM=AM+COULMB(NN)*C(II+P)*C(AA+P)
c               BM=BM+EXCHNG(NN)*C(II+P)*C(AA+P)
c 160         CONTINUE
c             AQJ(Q,J)=AQJ(Q,J)+AM
c             BQJ(Q,J)=BM
c 170       CONTINUE
c 180     CONTINUE
C
C
C     SUM OVER Q.
c       DO 220 A=1,NVCS
c         AA=NBASIS*(IVMO(A)-1)
c         DO 210 J=1,I
C
c           IF(J.EQ.I) THEN
c             BMAX=A
c           ELSE
c             BMAX=NVCS
c           END IF
C
c           DO 200 B=1,BMAX
c             BB=NBASIS*(IVMO(B)-1)
c             N =N+1
c             AM=ZERO
c             BM=ZERO
c             DO 190 Q=1,NBASIS
c               AM=AM+AQJ(Q,J)*C(AA+Q)*C(BB+Q)
c 190         CONTINUE
c             OOVV(N)=OOVV(N)+AM
c             OVOV(N)=OVOV(N)+BM
c 200       CONTINUE
c 210     CONTINUE
c 220   CONTINUE
c 230 CONTINUE
C
c     RETURN
c     END
C
C@(#)=MOIOUT(SUB)
      SUBROUTINE MOIOUT
C
C     OUTPUT MODULE FOR MOLECULAR INTEGRALS, (IA/JB) AND (IJ/AB).
C     CAUTION.
C      MAXIMUM MO NO. PRINTED IN THIS ROUTINE IS LESS THAN 100.
C      IN CASE OF PRINTING MO WHOSE NO. IS GREATER THAN OR EQUAL TO 100,
C      FORMAT OVERFLOW WILL OCCUR.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER*4 A,B,AA,BB,BMAX
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      PARAMETER(MAXDIM=MAXBAS*(MCIOCC+1)*2+1)
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                NOCS,NVCS,NCSFS
      COMMON /WORK01/ OVOV(MCIINT),OOVV(MCIINT),DUMMY(MAXDIM)
      DIMENSION BUF(5),II(5),AA(5),JJ(5),BB(5)
 2000 Format(1H ,19('=')/
     &       1H ,'Molecular integrals'/
     &       1H ,19('='))
 2010 Format(1H ,'Type-1 ... (i,a,j,b)')
 2020 Format(1H ,5(:,'(',I2,',',I2,',',I2,',',I2,')=',F10.6,2X))
 2030 Format(1H ,'Type-2 ... (i,j,a,b)')
C     ***************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=3
      ELSE
        IW=5
      END IF
C
      WRITE(IOUT,2000)
C
C     PRINTING OF (IA/JB).
      WRITE(IOUT,2010)
      N=0
      DO 40 I=1,NOCS
        DO 30 A=1,NVCS
          DO 20 J=1,I
            IF(J.EQ.I) THEN
              BMAX=A
            ELSE
              BMAX=NVCS
            END IF
            DO 10 B=1,BMAX
              IX=MOD(N,IW)+1
              N =N+1
              BUF(IX)=OVOV(N)
              II(IX) =IOMO(I)
              AA(IX) =IVMO(A)
              JJ(IX) =IOMO(J)
              BB(IX) =IVMO(B)
              IF(IX.EQ.IW) THEN
                WRITE(IOUT,2020) (II(K),AA(K),JJ(K),BB(K),BUF(K),K=1,IW)
              END IF
   10       CONTINUE
   20     CONTINUE
   30   CONTINUE
   40 CONTINUE
C
      IF(IX.NE.IW) THEN
        WRITE(IOUT,2020) (II(K),AA(K),JJ(K),BB(K),BUF(K),K=1,IX)
      END IF
C
C     PRINTING OF (IJ/AB).
      WRITE(IOUT,2030)
      N=0
      DO 80 I=1,NOCS
        DO 70 A=1,NVCS
          DO 60 J=1,I
            IF(J.EQ.I) THEN
              BMAX=A
            ELSE
              BMAX=NVCS
            END IF
            DO 50 B=1,BMAX
              IX=MOD(N,IW)+1
              N =N+1
              BUF(IX)=OOVV(N)
```

```
              II(IX) =IOMO(I)
              JJ(IX) =IOMO(J)
              AA(IX) =IVMO(A)
              BB(IX) =IVMO(B)
              IF(IX.EQ.IW) THEN
                WRITE(IOUT,2020) (II(K),JJ(K),AA(K),BB(K),BUF(K),K=1,IW)
              END IF
   50       CONTINUE
   60     CONTINUE
   70   CONTINUE
   80 CONTINUE
C
      IF(IX.NE.IW) THEN
        WRITE(IOUT,2020) (II(K),JJ(K),AA(K),BB(K),BUF(K),K=1,IX)
      END IF
C
      RETURN
      END
C
C@(#)=MOLDIP(SUB)
      SUBROUTINE MOLDIP(DIP,C,LMO,IL,IU,NMOS,NBASIS,TD)
C
C     GET MOLECULAR INTEGRALS FOR DIPOLE MOMENTS.
C     On input:
C      DIP    ... DIPOLE INTEGRALS OVER SLATER TYPE FUNCTIONS.
C      C      ... MOLECULAR ORBITAL COEFFICIENTS.
C      LMO    ... LIST VECTOR FOR ACTIVE MOS.
C      IL     ... LOWER LOCATION NO. OF BASIS FUNCTIONS BELONGING TO
C                 I-TH ATOM.
C      IU     ... UPPER LOCATION NO. OF BASIS FUNCTIONS BELONGING TO
C                 I-TH ATOM.
C      NMOS   ... NO. OF ACTIVE MOS.
C      NBASIS ... NO. OF BASIS FUNCTIONS.
C     On output:
C      TD     ... TD(IMO,JMO) =
C                 SUM OF C(IAO,IMO)*C(JAO,JMO)*<IAO/R/JAO> OVER
C                 IAO AND JAO (BOTH BELONGING TO I-TH ATOM).
C                 WHERE,
C                 C            ... MOLECULAR ORBITAL COEFFICIENTS,
C                 <IAO/R/JAO> ... DIPOLE INTEGRAL OVER BASIS FUNCTIONS
C                                 IAO AND JAO.
C                 ACTUALLY ARRAY TDX IS PACKED FORM WITH RESPECT TO
C                 MOLECULAR ORBITALS IMO AND JMO.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DIP(9,9),C(*),TD(*),LMO(*)
      DATA ZERO/0.0D+00/
C     ****************************************************************
      IX=0
      DO 40 IBAS=IL,IU
        IX=IX+1
        JX=0
        DO 30 JBAS=IL,IU
          JX=JX+1
          DIJ=DIP(IX,JX)
          IF(DIJ.EQ.ZERO) GO TO 30
          N=0
          DO 20 I=1,NMOS
            IP=(LMO(I)-1)*NBASIS+IBAS
            DO 10 J=1,I
              N =N+1
              JP=(LMO(J)-1)*NBASIS+JBAS
              TD(N)=TD(N)+DIJ*C(IP)*C(JP)
   10       CONTINUE
   20     CONTINUE
   30   CONTINUE
   40 CONTINUE
C
      RETURN
      END
      subroutine molint
C=====================================================================*
C                                                                     *
C      @    @   @@@@   @          @    @    @   @@@@@                *
C      @@  @@  @    @  @          @    @@   @       @                *
C      @ @@ @  @    @  @          @    @ @  @       @                *
C      @    @  @    @  @          @    @  @ @       @                *
C      @    @  @    @  @          @    @   @@       @                *
C      @    @   @@@@   @@@@@@     @    @    @       @                *
C=====================================================================*
      implicit real*8 (a-h,o-z)
c
      include 'MSSIZE'
      include 'basinf.h'
      include 'mol.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
C
C     INTEGRAL TRANSFORMATION ... PARTIAL SUMMATION METHOD.
C                                 (FOUR-INDEX TRANSFORMATION METHOD)
C      (PP/QQ), (PQ/PQ) -> (IA/JB), (IJ/AB)
C       P, Q ... INDICES FOR BASIS FUNCTIONS.
C       I, J ... INDICES FOR OCCUPIED MOS.
C       A, B ... INDICES FOR VIRTUAL MOS.
C
C     NOTES.
C      1. THIS ROUTINE FORMS MOLECULAR INTEGRALS WHOSE TYPES ARE
C         (IA/JB) OR (IJ/AB).
C      2. ROUTINE INTCI1 PERFORM CNDO TYPE INTEGRAL TRANSFORMATION.
C      3. ROUTINE INTCI2 PERFORM INDO TYPE INTEGRAL TRANSFORMATION.
c            ... not available at present.
C      4. IN POST-SCF PROCEDURE, ARRAY C (MO COEFFICIENTS) IS PACKED TO
C         ONE DIMENSIONAL ARRAY.
C
C     REFERENCES.
C      1. H.F.SCHAEFER, ED., "METHODS OF ELECTRONIC STRUCTURE THEORY"
C         (MODERN THEORETICAL CHEMISTRY, VOL.3), PLENUM, 1977.
C      2. R.DAUDEL ET AL, "QUANTUM CHEMISTRY", WILEY, 1983.
c
c     93/Jun. A.M.
C     ****************************************************************
      COMMON /METHOD/ METHOD
      COMMON /IOPCI / IOPCI(18)
      COMMON /IO    / IN,IOUT,IFILE(10)
 2000 Format(1H ,'<<<<<< Enter MolInt >>>>>>')
C     ****************************************************************
      moint=iopci(11)
c
      write(iout,2000)
c
c     pack mo coefficients.
      call packmo
c
c     get window information for integral transformation.
      call moinfo
c
c     then perform integral transformation.
      if( isdci0 .eq. 0 ) then
        if( method .ne. 5 ) then
          call intci1
        else
          call intci2
        endif
        if( moint .eq. 1 ) then
          call moiout
        endif
      else
        nsdci = nocs + nvcs
        call mkcibs
        call intci0
c$$$        call elmntf
      endif
c
c     all done. return to caller.
      call clockm('molint.')
c
c
      return
      end
C
C@(#)=MSMOUT(SUB)
      SUBROUTINE MSMOUT(A,NBASIS)
C
C     OUTPUT MODULE FOR MOLECULAR ORBITAL BASED SYMMETRIC MATRIX.
C     PRINT OUT LOWER TRIANGLE MATRIX.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      DIMENSION A(*)
 2000 Format(1H ,5X,11(8X,I3))
 2010 Format(1H ,I4,4X,11(1X,F10.6))
C     ****************************************************************
      IF(IWDTH.EQ.1) THEN
```

```
        IW=6
      ELSE
        IW=11
      END IF
C
      ILOWER=1
      IUPPER=IW
C
   10 IF(IUPPER.GT.NBASIS) THEN
        IUPPER=NBASIS
      END IF
C
      WRITE(IOUT,2000) (I,I=ILOWER,IUPPER)
C
      DO 20 I=ILOWER,NBASIS
        NCLMS=I-ILOWER+1
        IF(NCLMS.GT.IW) THEN
          IU=ILOWER+IW-1
        ELSE
          IU=I
        END IF
C
        MA=I*(I-1)/2
C
        WRITE(IOUT,2010) I,(A(MA+J),J=ILOWER,IU)
C
   20 CONTINUE
C
      IF(IUPPER.EQ.NBASIS) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
C
C@(#)=NM2E(SUB)
      SUBROUTINE NM2E(WEISS,PNEWNM,COORD,GAB,GAA,NATOMS)
C
C     CALCULATION OF ERIS USING NISHIMOTO-MATAGA FORMULA OR
C     NISHIMOTO-MATAGA-WEISS FORMULA.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON(17)
      DIMENSION COORD(3,*),GAB(*),GAA(*)
C     ****************************************************************
      CONST1=TOANG*AUTOEV*WEISS
      CONST2=(CONST1+CONST1)*PNEWNM
C
      DO 30 IAT=1,NATOMS
        GAI  =GAA(IAT)
        IATM1=IAT-1
C
        IF(IATM1.EQ.0) GO TO 20
C
        MX =IAT*(IAT-1)/2
        CIX=COORD(1,IAT)
        CIY=COORD(2,IAT)
        CIZ=COORD(3,IAT)
        DO 10 JAT=1,IATM1
          MA =MX+JAT
          RX =COORD(1,JAT)-CIX
          RY =COORD(2,JAT)-CIY
          RZ =COORD(3,JAT)-CIZ
          RIJ=DSQRT(RX*RX+RY*RY+RZ*RZ)*TOANG
          GAJ=GAA(JAT)
C
C     NISHIMOTO-MATAGA FORMULA. (USING WEISS MODIFICATION)
          ALP    =CONST2/(GAI+GAJ)
          GAB(MA)=CONST1/(RIJ+ALP)
C
   10   CONTINUE
C
C     MOVE 1 CENTER ERIS INTO 'GAB'.
   20   MA=IAT*(IAT+1)/2
        GAB(MA)=GAI
   30 CONTINUE
C
      RETURN
      END
C
C@(#)=OHNO2E(SUB)
      SUBROUTINE OHNO2E(COORD,GAB,GAA,NATOMS)
C
C     CALCULATION OF ERIS USING OHNO FORMULA.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON(17)
      DIMENSION COORD(3,*),GAB(*),GAA(*)
C     ****************************************************************
      CONST1=TOANG*AUTOEV
      CONST2=CONST1+CONST1
C
      DO 30 IAT=1,NATOMS
        GAI  =GAA(IAT)
        IATM1=IAT-1
C
        IF(IATM1.EQ.0) GO TO 20
C
        MX =IAT*(IAT-1)/2
        CIX=COORD(1,IAT)
        CIY=COORD(2,IAT)
        CIZ=COORD(3,IAT)
        DO 10 JAT=1,IATM1
          MA =MX+JAT
          RX =COORD(1,JAT)-CIX
          RY =COORD(2,JAT)-CIY
          RZ =COORD(3,JAT)-CIZ
          RIJ=DSQRT(RX*RX+RY*RY+RZ*RZ)*TOANG
          GAJ=GAA(JAT)
C
C     OHNO FORMULA.
          ALP    =CONST2/(GAI+GAJ)
          GAB(MA)=CONST1/DSQRT(RIJ*RIJ+ALP*ALP)
C
   10   CONTINUE
C
C     MOVE 1 CENTER ERIS INTO 'GAB'.
   20   MA=IAT*(IAT+1)/2
        GAB(MA)=GAI
   30 CONTINUE
C
      RETURN
      END
C
C@(#)=OK2E(SUB)
      SUBROUTINE OK2E(COORD,GAB,GAA,NATOMS)
C
C     CALCULATION OF ERIS USING OHNO-KLOPMAN FORMULA.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON(17)
      DIMENSION COORD(3,*),GAB(*),GAA(*)
      DATA TWO/2.0D+00/
C     ****************************************************************
      CONST1=TOANG*AUTOEV
      CONST2=CONST1/TWO
C
      DO 30 IAT=1,NATOMS
        GAI  =GAA(IAT)
        IATM1=IAT-1
C
        IF(IATM1.EQ.0) GO TO 20
C
        MX =IAT*(IAT-1)/2
        CIX=COORD(1,IAT)
        CIY=COORD(2,IAT)
        CIZ=COORD(3,IAT)
        DO 10 JAT=1,IATM1
          MA =MX+JAT
          RX =COORD(1,JAT)-CIX
          RY =COORD(2,JAT)-CIY
          RZ =COORD(3,JAT)-CIZ
          RIJ=DSQRT(RX*RX+RY*RY+RZ*RZ)*TOANG
          GAJ=GAA(JAT)
C
C     OHNO-KLOPMAN FORMULA.
          ALP    =CONST2*(GAI+GAJ)/(GAI*GAJ)
          GAB(MA)=CONST1/DSQRT(RIJ*RIJ+ALP*ALP)
C
   10   CONTINUE
C
C     MOVE 1 CENTER ERIS INTO 'GAB'.
   20   MA=IAT*(IAT+1)/2
        GAB(MA)=GAI
```

```
   30 CONTINUE
C
      RETURN
      END
      subroutine openf( lunum5 )
C=====================================================================*
C                                                                     *
C       @@@@   @@@@@   @@@@@@  @    @  @@@@@@                        *
C      @    @  @    @  @       @@   @  @                              *
C      @    @  @    @  @@@@@   @ @  @  @@@@@                          *
C      @    @  @@@@@   @       @  @ @  @                              *
C      @    @  @       @       @   @@  @                              *
C       @@@@   @       @@@@@@  @    @  @                              *
C=====================================================================*
c$$$      include 'mimd00.idata'
      character*30 file1,  file2,  file3,  file4,  file5
      character fstat*7, date1*50
c$ohtawara
      common / io / in, iout, ifile(10)

      fstat = 'unknown'
c
c
c$$$      lunum6 = 6
c$$$      lunum7 = 7
c$$$      lunum8 = 8
c$$$      lunum9 = 9
c$$$      lunuma = 10
c
c$$$      call fdate( date1 )
c$$$      write(iout, '(1h )' )
c$$$      write(iout, '(1h ,a50)' ) date1
c$$$      write(iout, '(1h )' )
c

      open( 20, file='fnames', form='formatted', status='old' )
      rewind 20
      read(20,'(a30)') file1
c$$$      read(20,'(a30)') file2
c$$$      read(20,'(a30)') file3
c$$$      read(20,'(a30)') file4
c$$$      read(20,'(a30)') file5
      close( 20 )
c
coht      write(iout, '(//)' )
coht      write(iout, 10 )
coht      write(iout, '(/)' )
coht      write(iout, '(1h ,2a30)' ) '#5  input data -------------> ', file1
c$$$      write(iout, '(1h ,2a30)' ) '#7  cell attributes --------> ', file2
c$$$      write(iout, '(1h ,2a30)' ) '#8  physical quantities ----> ', file3
c$$$      write(iout, '(1h ,2a30)' ) '#7  calc. parameters -------> ', file4
c$$$      write(iout, '(1h ,2a30)' ) '#10 statistical quantities -> ', file5
coht      write(iout, '(//)' )
coht  10  format(' >>>>> file names <<<<<')
c
c
c
c

      open( lunum5, file=file1, form='formatted', status='old' )
c$$$      open( lunum7, file=file2, form='formatted', status=fstat )
c$$$      open( lunum8, file=file3, form='formatted', status=fstat )
c$$$      open( lunum9, file=file4, form='formatted', status=fstat )
c$$$      open( lunuma, file=file5, form='formatted', status=fstat )
c
c
c

      return
      end
      subroutine closef( lunum5 )
C=====================================================================*
C                                                                     *
C       @@@@   @        @@@@    @@@@   @@@@@@  @@@@@@                 *
C      @    @  @       @    @  @    @  @       @                      *
C      @       @       @    @  @@@@   @@@@@   @@@@@                  *
C      @       @       @    @       @  @       @                      *
C      @    @  @       @    @  @    @  @       @                      *
C       @@@@   @@@@@@   @@@@    @@@@   @@@@@@  @                      *
C=====================================================================*
c$$$      include 'mimd00.idata'
c
      character date1*50
c$$$      call fdate( date1 )
c$$$      write(iout, '(1h )' )
c$$$      write(iout, '(1h ,a50)' ) date1
c
********************************************* fir s1
c      close( 6 )
********************************************* fir s1
      close( lunum5 )
c$$$      close( lunum7 )
c$$$      close( lunum8 )
c$$$      close( lunum9 )
c$$$      close( lunuma )
c
      return
      end
      subroutine clock0
C=====================================================================*
C                                                                     *
C       @@@@   @        @@@@    @@@@   @    @  @   @                 *
C      @    @  @       @    @  @    @  @    @  @    @                 *
C      @       @       @    @  @       @@@@   @    @                 *
C      @       @       @    @  @       @  @   @    @                 *
C      @    @  @       @    @  @    @  @   @   @   @                 *
C       @@@@   @@@@@@   @@@@    @@@@   @    @   @@@                  *
C=====================================================================*
c
c------------ for unix
c
      character date1*50
      dimension time( 2 )
      elt   = etime( time )
      xtime = time( 1 )
c$$$      call fdate( date1 )
c
c-------------<< for cray xms >>
c
c      call second( xtime )
c
c------------<< for supertech s1 >>
c
c      call timeused( icpu, it1, it2 )
c      xtime = icpu
c      xtime = xtime*1.0d-6
c
c
      write(iout, '(1h )' )
c$$$      write(iout, '(1h ,a50)' ) date1
      ithr = int( xtime/3.6d3 )
      itmn = int( ( xtime - ithr*3.6d3 )/6.0d1 )
      tsec = xtime - ithr*3.6d3 - itmn*6.0d1
      htime = xtime/3.6d3
      write(iout,100) ithr, itmn, tsec, htime
c
  100 format(1h ,'used cpu time : ',i4,' hours ',i3,' minutes',
     i     1pd9.2,' seconds (',1pd12.5,' hours )' )
  200 format(1h ,' >>> cpu time ',1pg12.5 )
c
c
      return
      end
C
C@(#)=OptSet(SUB)
      Subroutine OptSet(IPrIop,IfCI,IfBeta)
C
C     Jul/93 A.M.
C
C     Setting of options in MOS-F.
C     Note.
C      All variables in labeled common /IOPMOL/, /IOPINT/, /IOPGES/,
C      /IOPSCF/, /IOPGPR/, /IOPCI /, /IOPDIP/, /METHOD/, /PRWDTH/ and
C      /FACTOR/ are defined here. Variables ICHARG and MULTIP in
C      /MOL   / and SCFCRT in /CRITRN/ are also defined.
C      Keyword and options are listed below.
C
C     ***************************************************************
C     Keyword on option cards
C     ***************************************************************
C
C=Method
C
C     1. 'CNDO/2'  ... CNDO/2 method (by Pople).
C     2. 'CNDO/S'  ... CNDO/S method (by Jaffe).
C     3. 'CNDO/S2' ... CNDO/S2 method (by Duke).
C     4. 'CNDO/S3' ... CNDO/S3 method (by Duke).
C     5. 'INDO/S'  ... INDO/S method (by Zerner). ... not available at
```

```
C                                                   present.
C         Default  ... CNDO/S method.
C
C=Charge
C     'CHARGE=N' ... Molecular charge.
C      Default   ... N=0 (neutral molecule).
C                       *** Caution ***
C                       Only closed shell Hartree-Fock calculation
C                       is available (spin multiplicity is fixed to  1).
C                       ***************
C
C=IWdth
C
C     1. 'TSS'    ... Maximum size of column in printing is 80.
C     2. 'L/P'    ... Maximum size of column in printing is 133.
C         Default ... 'TSS'
C
C=PrtIOp
C
C     'PRTIOP' ... Print option table.
C      Default ... Do not print option table.
C
C=SCFCrt
C
C     'SCFCRT=X.XXXXXXXXX' ... SCF criterion (from 1.0d-02 to 1.0d-09)
C      Default ... 'SCFCRT=0.00003' (3.0d-05)
C
C=FS,FP,FD
C
C     'FS=X.XXXXX' ... Scale factor of p-sigma type overlap integrals.
C      Default ... INDO/S : 'FS=1.267'
C                  Others : 'FS=1.0'
C
C     'FP=X.XXXXX' ... Scale factor of p-pi type overlap integrals.
C      Default ... INDO/S, singlet : 'FP=0.585'
C                  INDO/S, triplet : 'FP=0.680'
C                  CNDO/S          : 'FP=0.585'
C                  Others          : 'FP=1.0'
C
C     'FD=X.XXXXX' ... Scale factor of overlap integrals involving d
C                      orbitals.
C      Default ... CNDO/S : 'FD=0.3'
C                  Others : 'FD=1.0'
C
C=K-NM
C
C     'K-NM=XX.XXXX' ... K-factor for new Nishimoto-Mataga (NM) gamma.
C      Default ... 'K-NM=1.0' (conventional Nishimoto-Mataga formula).
C                       See below (2 center ERIs of IOpInt).
C
c=NoCI
c
c     'NoCI' ... Do not carry out CI calculation. This implies 'NoBeta'.
c      Default ... Carry out.
c
c=NoBeta
c
c     'NoBeta' ... Do not carry out the computation of
c                  first hyperpolarizability.
c      Default ... Carry out.
c
c=Anchor
c
c     'Anchor' ... Output for Anchor II. Subroutines 'AncMol', 'AncBas',
c                  'AncGP','AncEP', and so on are called by 'Anchor'.
c                  If you use other post-process graphic packages, you
c                  need to hack these routines.
c      Default ... Do not output.
C
C=Pi
C
C     'PI=X.XXXXXXXXXX' ... The pi (default ... see routine setcom)
C
C=ToAng
C
C     'TOANG=X.XXXXXXXXXX' ... Bohr to Angstrom (default ... SetCom)
C
C=AUToEV
C
C     'AUTOEV=XX.XXXXXXXXX' ... Hartree to eV (default ... SeTCom)
C
C=Planck
C
C     'PLANCK=X.XXXXXXXXXX' ... Planck's constant (default ... SeTCom)
C
C=SLight
C
C     'SLIGHT=X.XXXXXXXXXX' ... Speed of light (default ... SetCom)
C
C=EVToER
C
C     'EVTOER=X.XXXXXXXXXX' ... EV to erg (default ... SetCom)
C
C=EVToKC
C
C     'EVTOKC=XX.XXXXXXXXX' ... EV to kcal/mol (default ... SetCom)
C
C=IOpMol
C
C     'COORD='      ... Coordinate input.
C      1. 'COORD=INTER' ... Input internal coordinates.
C      2. 'COORD=CART'  ... Input Cartesian coordinates.
C          Default ... 'COORD=INTER'
C
C     'SYMM' ... Symmetry control for molecule is turned on.
C                    ... not available at present.
C
C     'PARM=' ... Atomic parameter input.
C      1. 'PARM=TYPE' ... Input parameters with respect to atomic type.
C      2. 'PARM=ATOM' ... Input parameters with respect to each atom.
C
C     'PCNOE' ... Do PCNOE calculation.
C
C     'ATDIST' ... Print distance matrix.
C
C     'PARMOUT=' ... Printing of atomic parameters.
C      1. 'PARMOUT=TYPE' ... Print parameters with respect to atomic
C                            type.
C      2. 'PARMOUT=ATOM' ... Print parameters with respect to each atom.
C      3. 'PARMOUT=ALL'  ... Print parameters whose atomic numbers are
C                            from 1 to 99.
C
C     'SLACON' ... Print Slater-Condon parameters.
C
C=IOpInt
C
C     'G='     ... 2 center ERIs, gamma(i,j).
C      1. 'G=A'   ... Analytic formula.
C      2. 'G=PP'  ... Pariser-Parr formula.
C      3. 'G=NM'  ... Nishimoto-Mataga formula (include new-NM formula,
C                     see above (K-NM)).
C      4. 'G=NMW' ... Nishimoto-Mataga-Weiss formula (see above (K-NM)).
C      5. 'G=O'   ... Ohno formula.
C      6. 'G=OK'  ... Ohno-Klopman formula.
C      7. 'G=DH'  ... DasGupta-Huzinaga formula.
C          Default ... CNDO/2          : 'G=A'
C                      CNDO/S,singlet  : 'G=NM'
C                      CNDO/S,triplet  : 'G=PP'
C                      CNDO/S2,S3      : 'G=NM'
C                      INDO/S,singlet  : 'G=NMW'
C                      INDO/S,triplet  : 'G=PP'
C
C     'DON2' ... Use d orbitals on second row atoms.
C
C     'DON3' ... Use d orbitals on third row atoms except transition
C                metals (always use d orbitals on transition metals).
C
C     'SOUT' ... Print scaled overlap integrals.
C
C     'GOUT' ... Print ERIs, gamma(i,j).
C
C=IOpGes
C
C     'GUESS' ... Print result of initial guess calculation.
C
C     'HCORE' ... Print core hamiltonian matrix.
C
C=IOpSCF
C
C     'SCFCYC=N' ... Maximum no. of SCF iteration is N. (N=1,999)
C      Default ... 'SCFCYC=100'
C
C     'FOCK' ... Print Fock matrix.
C
C=IOpGPr
C
```

```
C     'MOCOEF=N'   ... Print the N-highest occupied and the N-lowest
C                      unoccupied MOs. If N=-1, print all MOs.
C     'MOCOEF=ALL' ... Print all MOs.
C     'MOCOEF'     ... N=5.
C
C     'MODENS=N'   ... Print the N-highest occupied and the N-lowest
C                      unoccupied MO densities. If N=-1, print all MO
C                      densities.
C     'MODENS=ALL' ... Print all MO densities.
C     'MODENS'     ... N=5.
C
C     'DENS' ... Print density matrix.
C
C=IOpCI (  )
C
C     1. 'SINGLET' ... Singlet CI calculation.
C     2. 'TRIPLET' ... Triplet CI calculation.
C         Default  ... 'SINGLET'
C
C     1. 'CI(M N)' ... All possible singly excited CSFs formed from the
C                      M-highest occupied and the N-lowest unoccupied
C                      MOs are used in the SCI calculation.
C                      Thus the number of singly excited CSFs is M*N.
C     2. 'CI'      ... M=5,N=5.
C
C     3. 'CI=READ' ... Read active mo numbers from cards (useful to form
C                      CSFs by using discontinuous MOs).
C
C     'MOINT' ... Print molecular integrals.
C
C     'CIMAT' ... Print ci hamiltonian matrix.
C
C     'CIVEC' ... Printing of CI vector.
C
C=IOpDip(  )
C
C     'DIPOLE=ZDO' ... Use the zero differential overlap (ZDO)
C                      approximation throughout in evaluating 1 center
C                      dipole integrals, <i/r/j>.
C                      i.e., <i/r/j>=detla(i,j)*<i/r/i>.
C                      *** Caution ***
C                      Only net charge contribution will be considered
C                      in estimating dipole moments if the ZDO
C                      approximation is used.
C                      ***************
C      Default ...     Relax the ZDO approximation only in case of
C                      evaluating 1 center dipole integrals.
C
C     'DIPOLE' ... Print MO-based dipole matrix.
C
C     1. 'TMOM' ... Print transition moments.
C
C     2. 'TMOM=ALL' ... Print transition moments including those
C                       between excited states.
C
C     'SHG'       ... Print frequency-dependent 1-st
C                     hyperpolarizabilities, beta(i,j,k), for second
C                     harmonic generation (SHG).
C     'SHG=REV'   ... Print 1-st hyperpolarizabilities for SHG vs. the
C                     difference between the transition energy from
C                     ground state to S1 state and the energy of incident
C                     light.
C
C     'EO'        ... Print frequency dependent 1-st
C                     hyperpolarizabilities, beta(i,j,k), for
C                     electro-optic Pockels effect (EOPE).
C     'EO=REV'    ... Print 1-st hyperpolarizabilities for EOPE vs. the
C     'EOPE=REV'      difference between the transition energy from
C                     ground state to S1 state and the energy of incident
C                     light.
C
C     'OR'        ... Print frequency dependent 1-st
C                     hyperpolarizabilities, beta(i,j,k), for optical
C                     rectification (OR).
C     'OR=REV'    ... Print 1-st hyperpolarizabilities for OR vs. the
C                     difference between the transition energy from
C                     ground state to s1 state and the energy of incident
C                     light.
C
C     'NSTATES=N' ... Print transition energies and the dipole moments
C                     for the N-lowest excited states. If N=-1, print
C                     them for all the calculated excited states.
C     'NSTATES=ALL' ... Print transition energies and the dipole moments
C                     for all the calculated excited states.
C      Default ... NSTATES=10
c
c     'ECharge' ... Compute and print total atomic charges for the SCI
c                   excited states.
c     'ECharge=n' ... Compute and print total atomic charges for the n
c                     lowest SCI excited states.
c     'ECharge=All' ... Compute and print total atomic charges for all
c                     the SCI excited states.
c      Default ... n=nStates
C
C     ****************************************************************
C     OPTIONS  IOP(  )
C     ****************************************************************
C
C=IOpMol(  )
C
C     Mol( 1) ... Coordinate input.
C     1 ... Input internal coordinates.
C     2 ... Input Cartesian coordinates.
C
C     Mol( 2) ... Symmetry control.
C     0 ... Symmetry is turned off. Default.
C     1 ... Symmetry is turned on ... not available at present.
C
C     Mol( 3) ... Atomic parameter input (beta, gamma, ionization
C                 potential, Slater exponent, and so on).
C     0 ... Use default parameters. Default.
C     1 ... Input parameters with respect to atomic type.
C     2 ... Input parameters with respect to each atom.
C
C     Mol( 4) ... PCNOE calculation.
C     0 ... Do not do PCNOE calculation. Default.
C     1 ... Do PCNOE calculation.
C
C     Mol( 5) through Mol(10) ... not used at present.
C
C     Mol(11) ... Printing of distance matrix.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     Mol(12) ... Printing of atomic parameters.
C     0 ... Do not print.
C     1 ... Print parameters with respect to atomic type.
C     2 ... Print parameters with respect to each atom.
C     3 ... Print parameters whose atomic numbers are from 1 to 99.
C
C     MOL(13) ... Printing of Slater-Condon parameters.
C     0 ... Do not print. Default
C     1 ... Print.
C
C     Mol(14) through Mol(18) ... not used at present.
C
C=IOpInt(  )
C
C     Int( 1) ... Resonance integrals, beta(i,j) (This option does
C                 not depend on any keywords).
C     1 ... Use values for the CNDO/2 method.
C     2 ... Use values for the CNDO/S method.
C     3 ... Use values for the CNDO/S2 method.
C     4 ... Use values for the CNDO/S3 method.
C     5 ... Use values for the INDO/S method.
C
C     Int( 2) ... 2 center eris, gamma(i,j).
C     1 ... Analytic formula.
C     2 ... Pariser-Parr formula.
C     3 ... Nishimoto-Mataga formula (include new NM formula. see
C           above (k-nm)),
C     4 ... Nishimoto-Mataga-Weiss formula (see above (k-nm)).
C     5 ... Ohno formula.
C     6 ... Ohno-Klopman Formula.
C     7 ... DasGupta-Huzinaga formula.
C
C     Int( 3) ... Use of d orbitals on second row atoms.
C     0 ... Do not use. Default.
C     1 ... Use.
C
C     Int( 4) ... Use of d orbitals on third row atoms except transition
C                 metals (always use d orbitals on transition metals).
C     0 ... Do not use. Default.
C     1 ... Use.
C
C     Int( 5) through Int(10) ... not used at present.
```

```
C
C     Int(11) ... Printing of scaled overlap integrals.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     Int(12) ... Printing of ERIs, gamma(i,j).
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     Int(13) through Int(18) ... not used at present.
C
C=IOpGes(  )
C
C     Ges( 1) through Ges(10) ... not used at present.
C
C     Ges(11) ... Printing of initial guess.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     Ges(12) ... Printing of core hamiltonian matrix.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     Ges(13) through Ges(18) ... not used at present.
C
C=IOpSCF(  )
C
C     SCF( 1) ... Maximum no. of SCF iteration.
C     N ... Maximum no. of iteration is N. (N=1,999)
C
C     SCF( 2) through SCF(10) ... not used at present.
C
C     SCF(11) ... Printing of Fock matrix.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     SCF(12) through SCF(18) ... Not used at present.
C
C=IOpGPr(  )
C
C     GPr( 1) through GPr(10) ... not used at present.
C
C     GPr(11) ... Printing of MO coefficients.
C     0 ... Do not print. Default.
C     N ... Print the N-highest occupied and the N-lowest unoccupied
c           MOs.
C    -1 ... Print all MOs.
C
C     GPr(12) ... Printing of MO densities.
C     0 ... Do not print. Default.
C     N ... Print the N-highest occupied and the N-lowest unoccupied
c           MO densities.
C    -1 ... Print all MO densities.
C
C     GPr(13) ... Printing of density matrix.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     GPr(14) through GPr(18) ... not used at present.
C
C=IOpCI (  )
C
C     CI ( 1) ... Spin multiplicity in the CI calculation.
C     1 ... Singlet.
C     3 ... Triplet.
C
C     CI ( 2) ... Occupied MO range in the CI calculation.
C     0 ... Read active MOs from cards.
C     N ... Consider the N-highest occupied MOs.
C
C     CI ( 3) ... Unoccupied MO range in the CI calculation.
C     0 ... Read active MOs from cards.
C     N ... Consider the N-lowest unoccupied MOs.
C
C     CI ( 4) through CI (10) ... not used at present.
C
C     CI (11) ... Printing of molecular integrals.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     CI (12) ... Printing of CI hamiltonian matrix.
C     0 ... Do not print. Default.
C     1 ... Print.
```



```
C
C     CI (13) ... Printing of CI vector.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C=IOpDip(  )
C
C     Dip( 1) ... Handling of 1 center dipole integrals, <i/r/j>.
C     0 ... Evaluate analytically.
C     1 ... Use the ZDO approximation, i.e., <i/r/j>=delta(i,j)<i/r/i>.
C           *** Caution ***
C           Only net charge contribution will be considered in
C           estimating dipole moments if the ZDO approximation is used.
C           ***************
C
C     Dip( 2) through Dip(10) ... not used at present.
C
C     Dip(11) ... Printing of MO-based dipole matrix.
C     0 ... Do not print. Default.
C     1 ... Print.
C
C     Dip(12) ... Printing of matrix elements of <g/-er/e> and <e/r/e'>.
C                 (g   :wave function of ground state,
C                  e,e':wave functions of excited states)
C     0 ... Do not print. Default.
C     1 ... Print <g/-er/e>.
C     2 ... Print <g/-er/e> and <e/r/e'>.
C
C     Dip(13) ... Printing of frequency-dependent 1-st hyperpolariz-
C                 abilities, beta(i,j,k), for second harmonic
c                 generation (SHG).
C     0 ... Do not print. Default.
C     1 ... Print.
C     2 ... Print 1-st hyperpolarizabilities for SHG vs. the difference
C           between the transition energy from ground state to S1 state
C           and the energy of incident light.
C
C     Dip(14) ... Printing of frequency-dependent 1-st hyperpolariz-
C                 abilities, beta(i,j,k), for electro-optic Pockels
c                 effect (EOPE).
C     0 ... Do not print. Default.
C     1 ... Print.
C     2 ... Print 1-st hyperpolarizabilities for EOPE vs. the difference
C           between the transition energy from ground state to S1 state
C           and the energy of incident light.
C
C     Dip(15) ... Printing of frequency-dependent 1-st hyperpolariz-
C                 ability, beta(i,j,k), for optical rectification (OR).
C     0 ... Do not print. Default.
C     1 ... Print.
C     2 ... Print 1-st hyperpolarizabilities for or vs. the difference
C           between the transition energy from ground state to s1 state
C           and the energy of incident light.
C
C     Dip(16) ... Printing of transition energies and the dipole
C                 moments for the excited states.
C     N ... Print them for the N-lowest excited states. Default. N=10.
C    -1 ... Print them for all the calculated excited states.
C
c     Dip(17) ... Printing of total atomic charges for the SCI-excited
c                 states.
c     0 ... Do not print.
c     N ... Compute and print total atomic charges for the N-lowest
c           SCI excited states.
c    -1 ... Compute and print total atomic charges for all the SCI-
c           excited states.
C     Dip(18) ... not used at present.
C
c     Oct/92 ... A.M.
c     93/Mar ... A.M.
c     Jul/93 ... A.M.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      Character*256 FilNam
      CHARACTER*800 OPTCRD,Crd2
      CHARACTER*80  ICARD,iCrd2
      Character*8   ModelH
      CHARACTER*1   IBL1,COMMA,MINUS,RPAREN
      Logical IfCI,IfBeta
      Logical Anchor
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO    / IN,IOUT,IFILE(10)
```

```
      COMMON /CARD1 / OPTCRD,ICARD,Crd2,iCrd2
      COMMON /CARD2 / MAXCRD,LENOCR,LENICR,NCARDS,ISMIN,IEMAX
      COMMON /MOL   / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                NAE,NBE,NE,NBASIS
      COMMON /METHOD/ METHOD
      Common /CMethd/ ModelH
      COMMON /IOPMOL/ IOPMOL(18)
      COMMON /IOPINT/ IOPINT(18)
      COMMON /IOPGES/ IOPGES(18)
      COMMON /IOPSCF/ IOPSCF(18)
      COMMON /IOPGPR/ IOPGPR(18)
      COMMON /IOPCI / IOPCI (18)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /PCONST/ PCON(20)
      COMMON /PRWDTH/ IWDTH
      COMMON /CRITRN/ SCFCRT,CRTDUM(9)
      COMMON /FACTOR/ FS,FP,FD
      COMMON /KNEWNM/ PNEWNM
      COMMON /COMVAL/ VAL,IVAL,IEND
      Common /Anchor/ Anchor
      DIMENSION IPRM(10)
      DATA ZERO/0.0D+00/,TENP2/1.0D+02/
      DATA ONE/1.0D+00/,TENM9/1.0D-09/,TENM2/1.0D-02/,TOLSCF/3.0D-05/
      DATA MODEF/5/
      DATA IBL1/' '/,COMMA/','/,MINUS/'-'/,RPAREN/')'/
 9910 Format(1H ,75('X')/
     &       1H ,A7,' ILLEGAL CHARACTER WAS FOUND IN OPTION CARD.',
     &              ' CHECK YOUR INPUT DATA.'/
     &       1H ,75('X'))
 9920 Format(1H ,'WARNING! ',A7,' RIGHT PARENTHESIS IS INSUFFICIENT.')
 9930 Format(1H ,'WARNING! ',A7,' VALUE IS OUT OF RANGE. DEFAULT ',
     &           'VALUE IS ASSUMED.')
 9940 Format(1H ,'WARNING! OPTION CARD CONTAINS UNRECOGNIZABLE ',
     &           'OR UNNECESSARY KEYWORDS.')
 9950 Format(1H ,80A1)
C     ************************************************************
C     CLEAR IOP.
      IOPMAX=18
      DO 10 I=1,IOPMAX
        IOPMOL(I)=0
        IOPINT(I)=0
        IOPGES(I)=0
        IOPSCF(I)=0
        IOPGPR(I)=0
        IOPCI (I)=0
        IOPDIP(I)=0
   10 CONTINUE
C
      NLTTRS=NCARDS*(LENICR+1)
      DO 20 I=1,NLTTRS
        IF(OPTCRD(I:I).EQ.COMMA) THEN
          OPTCRD(I:I)=IBL1
        END IF
   20 CONTINUE
c
c     CI computation performed?
      Call SetFlg(' NOCI ', 6,IPrm(1))
      If (IPrm(1) .Eq. 1) IfCI = .False.
      If (Iprm(1) .Eq. 1) IfBeta = .False.
c
c     First hyperpolarizability computation performed?
      Call SetFlg(' NOBETA ', 8,IPrm(1))
      If (IPrm(1) .Eq. 1) IfBeta = .False.
c
c     Output for Anchor II.
      Call SetFlg(' ANCHOR ', 8,IPrm(1))
      If (IPrm(1) .Eq. 0) Anchor = .False.
      If (IPrm(1) .Eq. 1) Anchor = .True.
C
C     NET CHARGE OF MOLECULE.
      CALL IPOS(' CHARGE=',8,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        ICHARG=IVAL
      END IF
C
C     SPIN MULTIPLICITY IN GROUND STATE IS FIXED AT SINGLET IN CURRENT
C     VERSION OF MOS-F.
      MULTIP=1
C
C     METHOD.
      CALL IPOS(' CNDO/2 ' ,8,IPRM(1))
      CALL IPOS(' CNDO/S ' ,8,IPRM(2))
      CALL IPOS(' CNDO/S2 ',9,IPRM(3))
      CALL IPOS(' CNDO/S3 ',9,IPRM(4))
      CALL IPOS(' INDO/S ' ,8,IPRM(5))
      CALL CHKKEY(IPRM,5,METHOD)
      IF(METHOD.EQ.0) THEN
        METHOD=2
      END IF
c========( Mon Aug 28 16:33:44 JST 1995 )======
c$$$      If (Method .eq. 1) ModelH = 8hCNDO/2
c$$$      If (Method .eq. 2) ModelH = 8hCNDO/S
c$$$      If (Method .eq. 3) ModelH = 8hCNDO/S2
c$$$      If (Method .eq. 4) ModelH = 8hCNDO/S3
c$$$      If (Method .eq. 5) ModelH = 8hINDO/S
      If (Method .eq. 1) ModelH = 'CNDO/2  '
      If (Method .eq. 2) ModelH = 'CNDO/S  '
      If (Method .eq. 3) ModelH = 'CNDO/S2 '
      If (Method .eq. 4) ModelH = 'CNDO/S3 '
      If (Method .eq. 5) ModelH = 'INDO/S  '
c========( Mon Aug 28 16:33:48 JST 1995 )======
C#
      IF(METHOD.EQ.5) THEN
        STOP 'INDO/S CALCULATION IS NOT AVAILABLE.'
      END IF
C
C     SPIN MULTIPLICITY IN CI CALCULATION.
      CALL IPOS(' SINGLET ',9,IPRM(1))
      CALL IPOS(' TRIPLET ',9,IPRM(2))
      CALL CHKKEY(IPRM,2,MULTCI)
      IF(MULTCI.EQ.2) THEN
        MULTCI=3
      ELSE
        MULTCI=1
      END IF
C
C     BETA.
      IBETA=METHOD
C
C     GAMMA.
      CALL IPOS(' G=A '  ,5,IPRM(1))
      CALL IPOS(' G=PP ' ,6,IPRM(2))
      CALL IPOS(' G=NM ' ,6,IPRM(3))
      CALL IPOS(' G=NMW ',7,IPRM(4))
      CALL IPOS(' G=O '  ,5,IPRM(5))
      CALL IPOS(' G=OK ' ,6,IPRM(6))
      CALL IPOS(' G=DH ' ,6,IPRM(7))
      CALL CHKKEY(IPRM,7,IGAMMA)
      IF(IGAMMA.EQ.0) THEN
        IF     (METHOD.EQ.1) THEN
          IGAMMA=1
        ELSE IF(METHOD.EQ.2) THEN
          IF(MULTCI.EQ.1) THEN
            IGAMMA=3
          ELSE
            IGAMMA=2
          END IF
        ELSE IF(METHOD.EQ.3) THEN
          IGAMMA=3
        ELSE IF(METHOD.EQ.4) THEN
          IGAMMA=3
        ELSE IF(METHOD.EQ.5) THEN
          IF(MULTCI.EQ.1) THEN
            IGAMMA=4
          ELSE
            IGAMMA=2
          END IF
        END IF
      END IF
C
C     DETERMINE IF D ORBITALS ARE USED ON SECOND OR THIRD ROW ATOMS.
      CALL SETFLG(' DON2 ',6,IFDON2)
      CALL SETFLG(' DON3 ',6,IFDON3)
C
C     INPUT OPTION OF ATOMIC PARAMETERS.
      CALL IPOS(' PARM=TYPE ',11,IPRM(1))
      CALL IPOS(' PARM=ATOM ',11,IPRM(2))
      CALL CHKKEY(IPRM,2,IRDPRM)
C
C     PCNOE CALCULATION.
      CALL SETFLG(' PCNOE ',7,IPCNOE)
C
C     COORDINATE INPUT.
      CALL IPOS(' COORD=INTER ',13,IPRM(1))
      CALL IPOS(' COORD=CART ' ,12,IPRM(2))
```

```
      CALL CHKKEY(IPRM,2,ICOORD)
      IF(ICOORD.EQ.0) THEN
        ICOORD=1
      END IF
C
C     SYMMETRY.
      CALL SETFLG(' SYMM ',6,ISYMM)
C
C     MAXIMUM NO. OF SCF ITERATION.
      CALL IPOS(' SCFCYC=',8,IPRM(1))
      IF(IPRM(1).EQ.0) THEN
        ITMAX=100
      ELSE
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        ITMAX=IVAL
      END IF
      IF((ITMAX.LT.1) .OR. (ITMAX.GT.999)) THEN
        ITMAX=100
      END IF
C
C     RANGE OF MO IN CI CALCULATION.
      NOCI=0
      NVCI=0
      LENCHR=4
      CALL IPOS(' CI '      ,LENCHR,IPRM(1))
      CALL IPOS(' CI('      ,LENCHR,IPRM(2))
      CALL IPOS(' CI=READ ',      9,IPRM(3))
      CALL CHKKEY(IPRM,3,ID)
      IF(ID.EQ.2) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(2))
        NOCI=IVAL
        IF(IEND.EQ.IPRM(2)) THEN
          IX=IPRM(2)+LENCHR
        ELSE
          IX=IEND+1
        END IF
        IF(OPTCRD(IX:IX).EQ.RPAREN) THEN
          NVCI=NOCI
          OPTCRD(IX:IX)=IBL1
        ELSE IF(OPTCRD(IX:IX).NE.IBL1) THEN
          WRITE(IOUT,9910) 'CI.     '
          CALL MERROR
        ELSE
          IP2=IEND+2
          CALL GETVAL(OPTCRD,LENOCR,IP2)
          NVCI=IVAL
          IP1=IEND+1
          IF(OPTCRD(IP1:IP1).EQ.RPAREN) THEN
            OPTCRD(IP1:IP1)=IBL1
          ELSE
            WRITE(IOUT,9920) 'CI.'
          END IF
        END IF
      END IF
      IF(ID.NE.3) THEN
        IF(NOCI.EQ.0) THEN
          NOCI=MIN0(MODEF,MCIOCC)
          NVCI=MIN0(MODEF,MCIVAC)
        ELSE IF(NVCI.EQ.0) THEN
          NVCI=MIN0(NOCI, MCIVAC)
        END IF
      END IF
      IF((NOCI.LT.0.OR.NOCI.GT.MCIOCC)   .OR.
     &   (NVCI.LT.0.OR.NVCI.GT.MCIVAC))THEN
        WRITE(IOUT,9930) 'CI.'
        NOCI=MIN0(MODEF,MCIOCC)
        NVCI=MIN0(MODEF,MCIVAC)
      END IF
C
C     PRINT OPTION OF MO COEFFICIENTS.
      CALL IPOS(' MOCOEF ',      8,IPRM(1))
      CALL IPOS(' MOCOEF=',      8,IPRM(2))
      CALL CHKKEY(IPRM,2,MOCOEF)
      IF(MOCOEF.EQ.1) THEN
        MOCOEF=MODEF
      ELSE IF(MOCOEF.EQ.2) THEN
        IS=IPRM(2)+8
        IE=IS+3
        IF(OPTCRD(IS:IE).EQ.'ALL ') THEN
          IVAL=-1
          OPTCRD(IS:IE)=IBL1
        ELSE
          CALL GETVAL(OPTCRD,LENOCR,IPRM(2))
          IF(IVAL.LT.-1) THEN
            WRITE(IOUT,9930) 'MOCOEF.'
            IVAL=MODEF
          ELSE IF(IVAL*2.GT.MAXBAS) THEN
            IVAL=-1
          END IF
        END IF
        MOCOEF=IVAL
      END IF
C
C     MODENS.
      CALL IPOS(' MODENS ',      8,IPRM(1))
      CALL IPOS(' MODENS=',      8,IPRM(2))
      CALL CHKKEY(IPRM,2,MODENS)
      IF(MODENS.EQ.1) THEN
        MODENS=MODEF
      ELSE IF(MODENS.EQ.2) THEN
        IS=IPRM(2)+8
        IE=IS+3
        IF(OPTCRD(IS:IE).EQ.'ALL ') THEN
          IVAL=-1
          OPTCRD(IS:IE)=IBL1
        ELSE
          CALL GETVAL(OPTCRD,LENOCR,IPRM(2))
          IF(IVAL.LT.-1) THEN
            WRITE(IOUT,9930) 'MODENS.'
            IVAL=MODEF
          ELSE IF(IVAL*2.GT.MAXBAS) THEN
            IVAL=-1
          END IF
        END IF
        MODENS=IVAL
      END IF
C
C     PRINTING OF THE TRANSITION ENERGIES AND DIPOLE MOMENTS FOR THE
C     EXCITED STATES.
      CALL IPOS(' NSTATES=',      9,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        IS=IPRM(1)+9
        IE=IS+3
        IF(OPTCRD(IS:IE).EQ.'ALL ') THEN
          IVAL=-1
          OPTCRD(IS:IE)=IBL1
        ELSE
          CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
          IF(IVAL.LT.-1 .OR. IVAL.EQ.0) THEN
            WRITE(IOUT,9930) 'NSTATES.'
            IVAL=10
          END IF
        END IF
        NSTATE=IVAL
      ELSE
        NSTATE=10
      END IF
C
C     Printing of total atomic charges for the SCI excited states.
      CALL IPOS(' ECHARGE ',      9,IPRM(1))
      CALL IPOS(' ECHARGE=',      9,IPRM(2))
      CALL CHKKEY(IPRM,2,NEChrg)
      If (NEChrg .Eq. 1) Then
        NEChrg = NState
      Else If ( NEChrg .Eq. 2) Then
        IS=IPRM(2)+9
        IE=IS+3
        IF(OPTCRD(IS:IE).EQ.'ALL ') THEN
          IVAL=-1
          OPTCRD(IS:IE)=IBL1
        ELSE
          CALL GETVAL(OPTCRD,LENOCR,IPRM(2))
          IF(IVAL.LT.-1) THEN
            WRITE(IOUT,9930) 'ECharge.'
            IVAL=NState
          END IF
        END IF
        NEChrg=IVAL
      END IF
C
C     TSS OR L/P
      CALL IPOS(' TSS ',5,IPRM(1))
      CALL IPOS(' L/P ',5,IPRM(2))
      CALL CHKKEY(IPRM,2,IWDTH)
      IF(IWDTH.EQ.0) THEN
        IWDTH=1
```

```
      END IF
C
C     PRINTING OF ATOMIC PARAMETERS.
      CALL IPOS(' PARMOUT=TYPE ',14,IPRM(1))
      CALL IPOS(' PARMOUT=ATOM ',14,IPRM(2))
      CALL IPOS(' PARMOUT=ALL ' ,13,IPRM(3))
      CALL CHKKEY(IPRM,3,IPRPRM)
C
C     PRINTING OF TRANSITION MOMENTS.
      CALL IPOS(' TMOM '     , 6,IPRM(1))
      CALL IPOS(' TMOM=ALL ',10,IPRM(2))
      CALL CHKKEY(IPRM,2,IPRTM)
C
C     SETTING OF THE OTHER PRINT OPTIONS.
      CALL SETFLG(' PRTIOP ' ,8,IPRIOP)
      CALL SETFLG(' ATDIST ' ,8,IATDIS)
      CALL SETFLG(' SLACON ' ,8,IPRSCP)
      CALL SETFLG(' SOUT '   ,6,IOVOUT)
      CALL SETFLG(' GOUT '   ,6,IPRGMM)
      CALL SETFLG(' GUESS '  ,7,IPRGES)
      CALL SETFLG(' HCORE '  ,7,IPRHCR)
      CALL SETFLG(' FOCK '   ,6,IPRFCK)
      CALL SETFLG(' DENS '   ,6,IDNS  )
      CALL SETFLG(' MOINT '  ,7,MOINT )
      CALL SETFLG(' CIMAT '  ,7,ICIMAT)
      CALL SETFLG(' CIVEC '  ,7,ICIVEC)
      CALL SETFLG(' DIPOLE ' ,8,IPRD  )
C
C     FREQUENCY DEPENDENT BETA.
      CALL IPOS(' SHG '         ,5,IPRM(1))
      CALL IPOS(' SHG=REV '     ,9,IPRM(2))
      CALL CHKKEY(IPRM,2,IPRSHG)
C
      CALL IPOS(' EO '        ,4,IPRM(1))
      CALL IPOS(' EO=REV '    ,8,IPRM(2))
      CALL IPOS(' EOPE '      ,6,IPRM(3))
      CALL IPOS(' EOPE=REV ',10,IPRM(4))
      CALL CHKKEY(IPRM,4,IPREO)
      IF(IPREO.GT.2) THEN
        IPREO=IPREO-2
      END IF
C
      CALL IPOS(' OR '      ,4,IPRM(1))
      CALL IPOS(' OR=REV ' ,8,IPRM(2))
      CALL CHKKEY(IPRM,2,IPROR)
C
C     HANDLING OF 1 CENTER DIPOLE INTEGRALS IN EVALUATING DIPOLE MOMENTS
C     AND TRANSITION MOMENTS.
      CALL SETFLG(' DIPOLE=ZDO ',12,IZDO)
C
C     SCF CRITERION.
      CALL IPOS(' SCFCRT=',8,IPRM(1))
      IF(IPRM(1).EQ.0) THEN
        SCFCRT=TOLSCF
      ELSE
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        IF(VAL.GT.TENM2.OR.VAL.LT.TENM9) THEN
          WRITE(IOUT,9930) 'SCFCRT.'
          VAL=TOLSCF
        END IF
        SCFCRT=VAL
      END IF
C
C     SCALE FACTOR FOR BETA(I,J) : FS=P-SIGMA, FP=P-PI, FD=D ORBITAL.
      FS=ONE
      CALL IPOS(' FS=',4,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        FS=VAL
      ELSE IF(METHOD.EQ.5) THEN
        FS=1.267D+00
      END IF
C
      FP=ONE
      CALL IPOS(' FP=',4,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        FP=VAL
      ELSE IF(METHOD.EQ.2) THEN
        FP=0.585D+00
      ELSE IF(METHOD.EQ.5) THEN
        IF(MULTCI.EQ.1) THEN
          FP=0.585D+00
        ELSE
          FP=0.680D+00
        END IF
      END IF
C
      FD=ONE
      CALL IPOS(' FD=',4,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        FD=VAL
      ELSE IF(METHOD.EQ.2) THEN
        FD=0.300D+00
      END IF
C
C     K FACTOR FOR NEW NISHIMOTO-MATAGA GAMMA.
      CALL IPOS(' K-NM=',6,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        PNEWNM=VAL
      ELSE
        PNEWNM=ONE
      END IF
      IF(PNEWNM.LE.ZERO .OR. PNEWNM.GE.TENP2) THEN
        WRITE(IOUT,9930) 'K-NM.'
        PNEWNM=ONE
      END IF
C
C     CHANGE PHYSICAL CONSTANT.
      CALL IPOS(' PI=',4,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        PCON(1)=VAL
      END IF
C
      CALL IPOS(' TOANG=',7,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        PCON(2)=VAL
      END IF
C
      CALL IPOS(' AUTOEV=',8,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        PCON(3)=VAL
      END IF
C
      CALL IPOS(' PLANCK=',8,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        PCON(4)=VAL
      END IF
C
      CALL IPOS(' SLIGHT=',8,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        PCON(5)=VAL
      END IF
C
      CALL IPOS(' EVTOER=',8,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        PCON(6)=VAL
      END IF
C
      CALL IPOS(' EVTOKC=',8,IPRM(1))
      IF(IPRM(1).NE.0) THEN
        CALL GETVAL(OPTCRD,LENOCR,IPRM(1))
        PCON(7)=VAL
      END IF
C
C     OPTCRD IS NOW BLANK.
      IF(OPTCRD.NE.IBL1) THEN
        WRITE(IOUT,9940)
        WRITE(IOUT,9950) (MINUS,IJKL=ISMIN,IEMAX)
        DO 30 I=1,NCARDS
          IS=(I-1)*(LENICR+1)+2
          IE=IS+LENICR-1
          ICARD=OPTCRD(IS:IE)
          WRITE(IOUT,9950) (ICARD(J:J),J=ISMIN,IEMAX)
   30   CONTINUE
        WRITE(IOUT,9950) (MINUS,IJKL=ISMIN,IEMAX)
      END IF
C
```

```
C     SET SPECIAL CONDITIONS.
      IF(IRDPRM.GT.IPRPRM) THEN
        IPRPRM=IRDPRM
      END IF
c
      If (Anchor) then
        Call GetEnv('File10',FilNam)
        Open (Unit = IFile(10),
     &        File = FilNam,
     &        Status = 'UnKnown')
      EndIf
C
C     THEN REMOVE VALUES TO IOP.
      IOPMOL( 1)=ICOORD
      IOPMOL( 2)=ISYMM
      IOPMOL( 3)=IRDPRM
      IOPMOL( 4)=IPCNOE
      IOPMOL(11)=IATDIS
      IOPMOL(12)=IPRPRM
      IOPMOL(13)=IPRSCP
C
      IOPINT( 1)=IBETA
      IOPINT( 2)=IGAMMA
      IOPINT( 3)=IFDON2
      IOPINT( 4)=IFDON3
      IOPINT(11)=IOVOUT
      IOPINT(12)=IPRGMM
C
      IOPGES(11)=IPRGES
      IOPGES(12)=IPRHCR
C
      IOPSCF( 1)=ITMAX
      IOPSCF(11)=IPRFCK
C
      IOPGPR(11)=MOCOEF
      IOPGPR(12)=MODENS
      IOPGPR(13)=IDNS
C
      IOPCI ( 1)=MULTCI
      IOPCI ( 2)=NOCI
      IOPCI ( 3)=NVCI
      IOPCI (11)=MOINT
      IOPCI (12)=ICIMAT
      IOPCI (13)=ICIVEC
C
      IOPDIP( 1)=IZDO
      IOPDIP(11)=IPRD
      IOPDIP(12)=IPRTM
      IOPDIP(13)=IPRSHG
      IOPDIP(14)=IPREO
      IOPDIP(15)=IPROR
      IOPDIP(16)=NSTATE
      IopDip(17) = NEChrg
C
      RETURN
      END
C
C@(#)=OR(SUB)
      SUBROUTINE OR(W,BETAOR,BVEC,TGN,TNN,CIEIG,NCSFS,MDIM1,MDIM2)
C
C     CALCULATE 1-ST HYPERPOLARIZABILITIES FOR OPTICAL RECTIFICATION
C     (OR) ... BETA(0;W,-W).
C     UNIT : ATOMIC UNIT.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION BETAOR(*),BVEC(*),TGN(MDIM1,*),TNN(MDIM2,*),CIEIG(*)
      DATA TWO/2.0D+00/,THREE/3.0D+00/
C     ***************************************************************
      N=0
      DO 30 I=1,3
        DO 20 J=1,3
          DO 10 K=J,3
            N=N+1
            BETAOR(N)=BETXYZ(W,-W,I,J,K,TGN,TNN,CIEIG,NCSFS,MDIM1,MDIM2)
   10     CONTINUE
   20   CONTINUE
   30 CONTINUE
C
C     THEN GET BETA-VEC.
C     DO 50 I=1,3
C       BVEC(I)=ZERO
C       NNI=(I-1)*6
C       DO 40 J=1,3
C         NNJ=(J-1)*6
C         II =MIN0(I,J)
C         JJ =MAX0(I,J)
C         MA1=NNI+(J-1) *(8-J) /2+1
C         MA2=NNJ+(II-1)*(8-II)/2+JJ-II+1
C         BVEC(I)=BVEC(I)+BETAOR(MA1)+TWO*BETAOR(MA2)
C  40   CONTINUE
C       BVEC(I)=BVEC(I)/THREE
C  50 CONTINUE
C
      BVEC(1)=(  BETAOR( 1)*THREE
     &         + BETAOR( 8)*TWO+BETAOR( 4)
     &         + BETAOR(15)*TWO+BETAOR( 6) ) / THREE
      BVEC(2)=(  BETAOR( 2)*TWO+BETAOR( 7)
     &         + BETAOR(10)*THREE
     &         + BETAOR(17)*TWO+BETAOR(12) ) / THREE
      BVEC(3)=(  BETAOR( 3)*TWO+BETAOR(13)
     &         + BETAOR(11)*TWO+BETAOR(16)
     &         + BETAOR(18)*THREE          ) / THREE
C
      RETURN
      END
C
C@(#)=OVLP(SUB) ... CNDO2/3R
      SUBROUTINE OVLP(IOVOUT)
C
C     CALCULATION OF OVERLAP INTEGRALS FOR SLATER-TYPE ORBITALS.
c     93/Jan A.M.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PCONST/ PI,TOANG,PCON(18)
      COMMON /MOL   / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                NAE,NBE,NE,NBASIS
      COMMON /FACTOR/ FS,FP,FD
      COMMON /EXPNT / ZETA(MAXAT)
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                IBTOA(MAXBAS)
      COMMON /QNOINF/ NC(MAXIAN,2),LC(9),MC(9)
      COMMON /FCTRL / FACT(0:50)
      COMMON /WORK01/ S(MBPCK),DUMMY1(MBPCK,3),DUMMY2(MAXAT,7),
     &                DUMMY3(1)
      DIMENSION SAB(9,9),TMAT(9,9),DUM(9,9),CI(3),CJ(3),E(3),SCAL(3)
      EQUIVALENCE (FS,SCAL(1))
      DATA ZERO/0.0D+00/,PT5/0.5D+00/,ONE/1.0D+00/
      Data TenP2/1.0d+02/
 2000 Format(1H ,14('=')/
     &       1H ,'Overlap matrix'/
     &       1H ,14('='))
 9910 Format(1H ,64('X')/
     &       1H ,'TOO SHORT DISTANCE BETWEEN ATOMS ',I3,' AND ',I3,'. ',
     &          'R=',1PD10.3,' BOHR.'/
     &       1H ,64('X'))
C     ***************************************************************
      RTHRSH=PT5/TOANG
c========( Thu Sep  7 09:55:33 JST 1995 )======
c$$$              write(iout,*)' ma, iat, jat, i, j'
c========( Thu Sep  7 09:55:33 JST 1995 )======
C
      DO 140 IAT=1,NATOMS
        NI=NIBAS(IAT)
        IL=NLBAS(IAT)
        IATM1=IAT-1
        IF(IATM1.EQ.0) GO TO 110
C
        IA=IAN(IAT)
        CI(1)=COORD(1,IAT)
        CI(2)=COORD(2,IAT)
        CI(3)=COORD(3,IAT)
C
C     TWO CENTER TERMS.
        DO 100 JAT=1,IATM1
          NJ=NIBAS(JAT)
          JL=NLBAS(JAT)
          DO 10 I=1,NI
          DO 10 J=1,NJ
            SAB(I,J)=ZERO
   10     CONTINUE
C
          JA=IAN(JAT)
          CJ(1)=COORD(1,JAT)
```

```
      CJ(2)=COORD(2,JAT)
      CJ(3)=COORD(3,JAT)
C
      CALL RELVEC(CI,CJ,R,E)
C
      IF(R.LT.RTHRSH) THEN
        WRITE(IOUT,9910) IAT,JAT,R
        STOP
      END IF
c=>   93/Jan A.M.
      If (R .gt. TenP2) GoTo 75
c<=
C
      ZRI=ZETA(IAT)*R
      ZRJ=ZETA(JAT)*R
C
      DO 30 I=1,NI
        LCI=LC(I)
        MCI=MC(I)
        II =LCI/2+1
        NCI=NC(IA,II)
        DO 20 J=1,NJ
          LCJ=LC(J)
          MCJ=MC(J)
          JJ =LCJ/2+1
          NCJ=NC(JA,JJ)
          IF(MCJ.NE.MCI) GO TO 20
          IF(MCI.LT.0) THEN
            SAB(I,J)=SAB(I-1,J-1)
            GO TO 20
          END IF
C
          SX   =(ZRI**(NCI+NCI+1)) * (ZRJ**(NCJ+NCJ+1))
          SX   =SX/(FACT(NCI+NCI)*FACT(NCJ+NCJ))
          SIAJA=DSQRT(SX)
          PHR  =(-ONE)**(LCJ+MCJ)
          SIJ  =SIAJA*PHR*SS(NCI,LCI,MCI,NCJ,LCJ,ZRI,ZRJ)
C
C     SCALING OF OVERLAP INTEGRALS.
          IF((LCI.EQ.2) .OR. (LCJ.EQ.2)) THEN
            SIJ=SIJ*FD
          ELSE IF((LCI.EQ.1) .AND. (LCJ.EQ.1)) THEN
            SIJ=SIJ*SCAL(MCI+1)
          END IF
C
          SAB(I,J)=SIJ
C
   20       CONTINUE
   30     CONTINUE
C
      LCMAX=MAX0(LC(NI),LC(NJ))
C
      CALL HARMTR(E,LCMAX,TMAT)
C
C     TRANSFORMATION OF SAB WITH RESPECT TO LOCAL COORDINATE INTO THAT
C     WITH RESPECT TO CARTESIAN COORDINATE.
      DO 50 I=1,NI
      DO 50 J=1,NJ
        T=ZERO
        DO 40 K=1,NJ
          T=T+SAB(I,K)*TMAT(J,K)
   40   CONTINUE
        DUM(I,J)=T
   50 CONTINUE
C
      DO 70 I=1,NI
      DO 70 J=1,NJ
        T=ZERO
        DO 60 K=1,NI
          T=T+TMAT(I,K)*DUM(K,J)
   60   CONTINUE
        SAB(I,J)=T
   70 CONTINUE
C
C     MOVE SAB INTO S.
c=>   93/Jan A.M.
c     DO 90 I=1,NI
   75 DO 90 I=1,NI
        IX=IL+I-1
        MX=IX*(IX-1)/2
        DO 80 J=1,NJ
          JX=JL+J-1
          MA=MX+JX
```

```
          S(MA)=SAB(I,J)
c========( Thu Sep  7 09:55:33 JST 1995 )======
c$$$            write(iout,*) ma, iat, jat, i, j
c========( Thu Sep  7 09:55:33 JST 1995 )======
   80     CONTINUE
   90   CONTINUE
C
  100 CONTINUE
C
C     ONE CENTER TERMS.
  110 DO 130 I=1,NI
        IX=IL+I-1
        MX=IX*(IX-1)/2
        DO 120 J=1,I
          JX=IL+J-1
          MA=MX+JX
          S(MA)=ZERO
c========( Thu Sep  7 09:55:33 JST 1995 )======
c$$$            write(iout,*) ma, iat, jat, i, j
c========( Thu Sep  7 09:55:33 JST 1995 )======
  120   CONTINUE
        S(MA)=ONE
c========( Thu Sep  7 09:55:33 JST 1995 )======
c$$$            write(iout,*) ma, iat, jat, i, j
c========( Thu Sep  7 09:55:33 JST 1995 )======
  130 CONTINUE
  140 CONTINUE
C
c========( Thu Sep  7 09:55:01 JST 1995 )======
c$$$    ii = 1
c$$$    if( ii .eq. 1 ) stop
c========( Thu Sep  7 09:55:02 JST 1995 )======
C     PRINT OUT SCALED OVERLAP MATRIX.
      IF(IOVOUT.EQ.1) THEN
        WRITE(IOUT,2000)
        CALL SSMOUT(S,IAN,IBTOA,NLBAS,NBASIS)
      END IF
C
      RETURN
      END
      subroutine packci
c=====================================================================*
c                                                                     *
c      @@@@@     @@     @@@@   @    @   @@@@      @                  *
c      @    @   @ @    @    @  @   @   @    @     @                  *
c      @    @  @   @   @       @@@@    @          @                  *
c      @@@@@   @@@@@@  @        @  @   @          @                  *
c      @       @    @  @    @   @   @   @    @     @                  *
c      @       @    @   @@@@    @    @   @@@@      @                  *
c=====================================================================*
c
c     move eigenvectors of ci hamiltonian matrix and pack them.
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci00.h'
      parameter( mciin2 = mciint + mciint )
      common/ work01 / dummy1(mciin2), civec(mcicsf,mcicsf), dummy2(1)
c
c
      n = 0
      do 1 i = 1, ncibas
        do 2 j =1, ncibas
          n = n + 1
          dummy1( n ) = civec( j, i )
    2   continue
    1 continue
c
      return
      end
C
C@(#)=PACKCV(SUB)
      SUBROUTINE PACKCV
C
C     MOVE EIGENVECTORS OF CI HAMILTONIAN MATRIX AND PACK THEM.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      PARAMETER(MCIIN2=MCIINT+MCIINT)
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                NOCS,NVCS,NCSFS
      COMMON /WORK01/ DUMMY1(MCIIN2),CIVEC(MCICSF,MCICSF),DUMMY2(1)
```

```
C     ***************************************************************
      N=0
      DO 20 I=1,NCSFS
        DO 10 J=1,NCSFS
          N=N+1
          DUMMY1(N)=CIVEC(J,I)
   10   CONTINUE
   20 CONTINUE
C
      RETURN
      END
C
C@(#)=PACKMO(SUB)
      SUBROUTINE PACKMO
C
C     PACKING OF ARRAY C (MOLECULAR ORBITAL COEFFICIENTS).
C
C     NOTE.
C      IN POST-SCF PROCEDURE, PACKED 'C' IS USED.
C ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      PARAMETER(MAXDIM=MAXIAN*65+1)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),IDUMMY(MAXDIM)
C ***************************************************************
      DO 20 J=2,NBASIS
        IX=(J-1)*NBASIS
        JX=(J-1)*MAXBAS
        DO 10 I=1,NBASIS
          IP=IX+I
          JP=JX+I
          C(IP)=C(JP)
   10   CONTINUE
   20 CONTINUE
C
      RETURN
      END
C
C@(#)=PCNOE(SUB)
      SUBROUTINE PCNOE
C
C     ***************************************************************
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /IOPMOL/ IOPMOL(18)
 2000 Format(1H ,54('#')/
     &       1H ,'PCNOE ... Pseudo complete neglect of everything. ',
     &           'Stop!'/
     &       1H ,54('#'))
C     ***************************************************************
      IPCNOE=IOPMOL( 4)
      IF(IPCNOE.EQ.0) THEN
        RETURN
      END IF
      WRITE(IOUT,2000)
      STOP
      END
C
C@(#)=PDMIX(FUN)
      DOUBLE PRECISION FUNCTION PDMIX(NCP,NCD,ZP,ZD)
C
C     GET 1 CENTER P-D MIXING TERM, <P/R/D>.
C     PDMIX IS INDEPENDENT OF MAGNETIC QUANTUM NOS. OF BASIS FUNCTIONS.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /FCTRL / FACT(0:50)
      DATA FIVE/5.0D+00/
C     ***************************************************************
      ROOT5=DSQRT(FIVE)
      NC2=NCP+NCD
      TRM1=FACT(NC2+1) *DFLOAT(2**(NC2+1)) *(ZP**NCP) *(ZD**NCD)*
     &     DSQRT(ZP*ZD)
      TRM2=ROOT5 *((ZP+ZD)**(NC2+2))*DSQRT(FACT(NCP+NCP)*FACT(NCD+NCD))
C
      PDMIX=TRM1/TRM2
C
      RETURN
      END
C
C@(#)=PP2E(SUB)
      SUBROUTINE PP2E(COORD,ZETA,GAB,GAA,IAN,NATOMS,NC,MDIM)
C
C     CALCULATION OF ERIS USING PARISER-PARR FORMULA.
C     93/Jun. A.M. ... Compiler problem.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON(17)
      DIMENSION COORD(3,*),ZETA(*),GAB(*),GAA(*),IAN(*),NC(MDIM,*)
      DATA CONPP/8.687D+00/,TWOPT8/2.8D+00/,F3PT7/3.7D+00/
      DATA ONE/1.0D+00/,TWO/2.0D+00/
C     ***************************************************************
      CONST=TOANG*AUTOEV
      SQ28 =TWOPT8*TWOPT8
      SQ37 =F3PT7*F3PT7
      DET  =SQ37*TWOPT8-SQ28*F3PT7
C
      GAB(1)=GAA(1)
C
      If(NAtoms .eq. 1) Then
        Return
      EndIf
C
C     DO 30 IAT=1,NATOMS
      DO 30 IAT=2,NATOMS
        GAI  =GAA(IAT)
        IATM1=IAT-1
C
C       IF(IATM1.EQ.0) GO TO 20
C
        IA =IAN(IAT)
        NCI=NC(IA,1)
        MX =IAT*(IAT-1)/2
        CIX=COORD(1,IAT)
        CIY=COORD(2,IAT)
        CIZ=COORD(3,IAT)
        RA =CONPP*TOANG/(ZETA(IAT)*DFLOAT(NCI))
        DO 10 JAT=1,IATM1
          JA =IAN(JAT)
          NCJ=NC(JA,1)
          MA =MX+JAT
          RX =COORD(1,JAT)-CIX
          RY =COORD(2,JAT)-CIY
          RZ =COORD(3,JAT)-CIZ
          RIJ=DSQRT(RX*RX+RY*RY+RZ*RZ)*TOANG
          GAJ=GAA(JAT)
          RB =CONPP*TOANG/(ZETA(JAT)*DFLOAT(NCJ))
          SRP=(RA+RB)*(RA+RB)
          SRM=(RA-RB)*(RA-RB)
C
C     PARISER-PARR FORMULA.
          IF(RIJ.GT.TWOPT8) THEN
            RIJ2 =RIJ+RIJ
            SRIJ2=RIJ2*RIJ2
            GAB(MA)=CONST*( (ONE/DSQRT(ONE+SRM/SRIJ2))
     &                    +(ONE/DSQRT(ONE+SRP/SRIJ2)) )/RIJ2
          ELSE
            RIJ2 =TWOPT8+TWOPT8
            SRIJ2=RIJ2*RIJ2
            GAB28=CONST*( (ONE/DSQRT(ONE+SRM/SRIJ2))
     &                   +(ONE/DSQRT(ONE+SRP/SRIJ2)) )/RIJ2
            RIJ2 =F3PT7+F3PT7
            SRIJ2=RIJ2*RIJ2
            GAB37=CONST*( (ONE/DSQRT(ONE+SRM/SRIJ2))
     &                   +(ONE/DSQRT(ONE+SRP/SRIJ2)) )/RIJ2
            GAV  =(GAI+GAJ)/TWO
            X28  =GAV-GAB28
            Y37  =GAV-GAB37
            A    =(SQ37*X28-SQ28*Y37)/DET
            B    =(-F3PT7*X28+TWOPT8*Y37)/DET
            GAB(MA)=GAV-(A+B*RIJ)*RIJ
          END IF
C
   10   CONTINUE
C
C     MOVE 1 CENTER ERIS INTO 'GAB'.
C  20   MA=IAT*(IAT+1)/2
        MA=IAT*(IAT+1)/2
        GAB(MA)=GAI
   30 CONTINUE
C
      RETURN
      END
C
C@(#)=PRECAL(SUB)
```

```
      SUBROUTINE PRECAL
C
C     ****************************************************************
      CALL SETCOM
      CALL PRTITL
      RETURN
      END
C
C@(#)=PRMSET(SUB)
      SUBROUTINE PRMSET
C
C     SETTING OF ATOMIC PARAMETERS LISTED BELOW.
C     1. EN        ... I OR (I+A)/2
C     2. BETA      ... RESONANCE INTEGRALS.
C     3. ZETA      ... EXPONENTS OF SLATER-TYPE ORBITALS.
C     4. GAA       ... 1 CENTER ERIS'(EXCEPT CNDO/2 METHOD).
C     5. PKLN      ... KLONDIKE PARAMETERS BY HUZINAGA.
C     6. /SLACON/ ... SLATER-CONDON PARAMETERS (INDO/S METHOD).
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /METHOD/ METHOD
      COMMON /IOPMOL/ IOPMOL(18)
      COMMON /IOPINT/ IOPINT(18)
      COMMON /IOPCI / IOPCI (18)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /PCONST/ PI,TOANG,AUTOEV,PCON(17)
      COMMON /EXPNT / ZETA(MAXAT)
      COMMON /GAMMA / GAB(MAPCK),GAA(MAXAT)
      COMMON /WORK01/ DUMMY(MBPCK,4),EN(MAXAT,3),BETA(MAXAT,3),
     &                PKLN(MAXAT),EP(MAXIAN,3),BP(MAXIAN,3),ZP(MAXIAN),
     &                GP(MAXIAN),PK(MAXIAN),IFLG(MAXAT),IDUMMY(1)
      DATA ZERO/0.0D+00/,ONE/1.0D+00/
 9910 Format(1H ,31('X')/
     &       1H ,'INSUFFICIENT ATOMIC PARAMETERS.'/
     &       1H ,31('X'))
C     ****************************************************************
      IRDPRM=IOPMOL( 3)
      IPRPRM=IOPMOL(12)
      IPRSCP=IOPMOL(13)
      IGAMMA=IOPINT( 2)
      MULTCI=IOPCI ( 1)
C
      CALL GETEP(EP,METHOD,MAXIAN)
      CALL GETBP(BP,METHOD,MAXIAN)
      CALL GETZP(ZP,ZCSP3,METHOD,MAXIAN)
      CALL GETGP(GP,METHOD,IGAMMA,MULTCI,MAXIAN)
      CALL GETPK(PK,IGAMMA,MAXIAN)
C
      IF((METHOD.EQ.5) .OR. (IPRSCP.EQ.1)) THEN
        CALL GETSCP
      END IF
C
      CALL RWPRM1(EP,BP,ZP,GP,PK,IAN,IFLG,NATOMS,IRDPRM,IPRPRM,METHOD,
     &            IGAMMA,MAXIAN)
C
      DO 10 I=1,NATOMS
        IA=IAN(I)
        EN(I,1)  =EP(IA,1)
        EN(I,2)  =EP(IA,2)
        EN(I,3)  =EP(IA,3)
        BETA(I,1)=BP(IA,1)
        BETA(I,2)=BP(IA,2)
        BETA(I,3)=BP(IA,3)
        ZETA(I)  =ZP(IA)
        GAA(I)   =GP(IA)
        PKLN(I)  =PK(IA)
   10 CONTINUE
C
      IF(METHOD.EQ.4) THEN
        CALL CNDOS3(COORD,ZETA,DUMMY,TOANG,IAN,ZCSP3,NATOMS,IOUT)
      END IF
C
      CALL RWPRM2(EN,BETA,ZETA,GAA,PKLN,IAN,IFLG,NATOMS,IRDPRM,IPRPRM,
     &            METHOD,IGAMMA,MAXAT)
C
      DO 20 I=1,NATOMS
        ERRFLG=EN(I,1)*BETA(I,1)*ZETA(I)
        IF(IGAMMA.NE.1) THEN
          ERRFLG=ERRFLG*GAA(I)
        END IF
        IF(IGAMMA.EQ.7) THEN
          ERRFLG=ERRFLG*PKLN(I)
        END IF
        IF(ERRFLG.EQ.ZERO) THEN
          WRITE(IOUT,9910)
          IPRPRM=2
          IRDPRM=0
          CALL RWPRM2(EN,BETA,ZETA,GAA,PKLN,IAN,IFLG,NATOMS,IRDPRM,
     &                IPRPRM,METHOD,IGAMMA,MAXAT)
          CALL MERROR
        END IF
   20 CONTINUE
C
      IF(IPRSCP.EQ.1) THEN
        CALL SCPOUT
      END IF
C
C     FINALLY CONVERT EV TO HARTREE (-BETA TO BETA).
      EVTOAU=ONE/AUTOEV
      DO 40 I=1,3
        DO 30 J=1,NATOMS
          EN(J,I)  = EN(J,I)  *EVTOAU
          BETA(J,I)=-BETA(J,I)*EVTOAU
   30   CONTINUE
   40 CONTINUE
C
      RETURN
      END
c#ifdef sun
C
C@(#)=PRTITL(SUB) ... Sun FORTRAN
      SUBROUTINE PRTITL
C
C     USE SUN FORTRAN TIME FUNCTIONS.
C      1 ... SUBROUTINE FDATE(CHR24) CHR24
C      2 ... Integer Function Time
C             No arguments required.
C      3 ... Subroutine ITime ... obsolete
C                SUBROUTINE ITIME(IARRAY) INTEGER IARRAY(3)
C                 IARRAY(1) ... HOUR.
C                 IARRAY(2) ... MINUTE.
C                 IARRAY(3) ... SECOND.
C     93/Mar A.M.
C     93/Apr A.M.
C     ****************************************************************
      Implicit Integer (a-z)
      Character*60 Card
      CHARACTER*24 DAYTIM
      Character*10 SysNam, NodNam, RelsNo, MacNam
      Character*1 Aster
      Save Aster
      Data Aster /1h*/
      Save MaxClm, MaxMA
      Data MaxClm /79/, MaxMA /58/
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /ELAPSE/ ISTART,IARRAY(3)
 2000 Format(1H1/
     &       1H ,'Just entered MOS-F system at ',A24,' ...')
 2010 Format(1h ,79A1)
 2020 Format(1H ,A/
     &       1h ,22x,'By A. Matsuura. Fujitsu Labs Ltd.')
C     ****************************************************************
c$$$      CALL FDATE(DAYTIM)
C     CALL ITIME(IARRAY)
C     CALL TOSEC(IARRAY,ISTART)
coht      IStart=time()
c
c     Cuts ... C-language function.
coht      Call Cuts (SysNam, NodNam, RelsNo, MacNam)
c
      Card = '   MOS-F (V01L05) ... '//
     &       SysNam(1:StrLen(SysNam))//' Release '//
     &       RelsNo(1:StrLen(RelsNo))//' '//
     &       MacNam(1:StrLen(MacNam))//'/'//
     &       NodNam(1:StrLen(NodNam))//'.'//Char(0)
c
coht      LenCrd = StrLen(Card)
coht      NAster = max0 (MaxMA, (min0 (MaxClm, (LenCrd + 3))) )
c
coht      WRITE(IOUT,2000) DAYTIM
coht      Write(IOut,2010) (Aster, ijkl = 1, NAster)
coht      Write(IOut,2020) Card(1:LenCrd)
```

```
coht      Write(IOut,2010) (Aster, ijkl = 1, NAster)
C
      RETURN
      END
C
C@(#)=PRTOPT(SUB)
      SUBROUTINE PRTOPT(IPRIOP,IfCI)
C
C     PRINTING OF LABELED COMMON /IOPXXX/.
C     ***************************************************************
      CHARACTER*19 CHR19(5)
      Logical IfCI
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /METHOD/ METHOD
      COMMON /IOPMOL/ IOPMOL(18)
      COMMON /IOPINT/ IOPINT(18)
      COMMON /IOPGES/ IOPGES(18)
      COMMON /IOPSCF/ IOPSCF(18)
      COMMON /IOPGPR/ IOPGPR(18)
      COMMON /IOPCI / IOPCI (18)
      COMMON /IOPDIP/ IOPDIP(18)
      DATA CHR19/'CNDO/2 calculation ','CNDO/S calculation ',
     &           'CNDO/S2 calculation','CNDO/S3 calculation',
     &           'INDO/S calculation '/
 2000 Format(1H ,A19)
 2010 Format(1H ,79('-'))
 2020 Format(1H ,' IOp :',18I4)
 2030 Format(1H ,' Mol :',18(1X,I3))
 2040 Format(1H ,' Int :',18(1X,I3))
 2050 Format(1H ,' Ges :',18(1X,I3))
 2060 Format(1H ,' SCF :',18(1X,I3))
 2070 Format(1H ,' GPr :',18(1X,I3))
 2080 Format(1H ,' CI  :',18(1X,I3))
 2090 Format(1H ,' Dip :',18(1X,I3))
C     ***************************************************************
      IOPMAX=18
C
      WRITE(IOUT,2000)  CHR19(METHOD)
C
      IF(IPRIOP.EQ.0) RETURN
C
      WRITE(IOUT,2010)
      WRITE(IOUT,2020) (I,I=1,IOPMAX)
      WRITE(IOUT,2010)
      WRITE(IOUT,2030) (IOPMOL(I),I=1,IOPMAX)
      WRITE(IOUT,2040) (IOPINT(I),I=1,IOPMAX)
      WRITE(IOUT,2050) (IOPGES(I),I=1,IOPMAX)
      WRITE(IOUT,2060) (IOPSCF(I),I=1,IOPMAX)
      WRITE(IOUT,2070) (IOPGPR(I),I=1,IOPMAX)
      If (IfCI) WRITE(IOUT,2080) (IOPCI (I),I=1,IOPMAX)
      If (IfCI) WRITE(IOUT,2090) (IOPDIP(I),I=1,IOPMAX)
      WRITE(IOUT,2010)
C
      RETURN
      END
c$$$#endif
c$$$#ifdef sun
C
C@(#)=PSTCAL(SUB) ... Sun FORTRAN
      SUBROUTINE PSTCAL
C
C     Job is terminated in this routine.
C     Use Sun FORTRAN time functions.
C     93/Mar A.M.
C     ***************************************************************
      CHARACTER*24 DAYTIM
      Integer Time
      Logical Anchor
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /ELAPSE/ ISTART,IARRAY(3)
      Common /Anchor/ Anchor
      DIMENSION SINGLE(2)
C2000 Format(1H ,63('*'))
 2010 Format(1H ,'   >>> MOS-F ... Normal end at ',A24,'.')
 2020 Format(1H ,'        CPU Time       ...',
     &       I4.2,' Hr ',I2.2,' Min ',I2.2,' Sec.')
 2030 Format(1H ,'        Elapsed Time  ...',
     &       I4.2,' Hr ',I2.2,' Min ',I2.2,' Sec.')
 2040 Format(1H ,'Job terminated normally in routine PstCal.')
C     ***************************************************************
C     WRITE(IOUT,2000)
C
c$$$      CALL FDATE(DAYTIM)
```

```
coht      WRITE(IOUT,2010) DAYTIM
C
coht      TIME1=ETIME(SINGLE)
coht      ICPU =NINT(TIME1)
coht      CALL TOHMS(ICPU,IHR,IMIN,ISEC)
coht      WRITE(IOUT,2020) IHR,IMIN,ISEC
C
C     CALL ITIME(IARRAY)
C     CALL TOSEC(IARRAY,IEND)
coht      IEnd=Time()
coht      IELAPS=IEND-ISTART
coht      CALL TOHMS(IELAPS,IHR,IMIN,ISEC)
coht      WRITE(IOUT,2030) IHR,IMIN,ISEC
C
C     WRITE(IOUT,2000)
coht      WRITE(IOUT,2040)
C
coht      If (Anchor) Then
coht        IUnit=IFile(10)
coht        Write(IUnit,*) ' '
coht        Close(Unit=IUnit)
coht      EndIf
C
      STOP
      END
C
C@(#)=RCOORD(SUB)
      SUBROUTINE RCOORD(C,IATN,NA,NB,NC,NATOMS,NATDUM,MAXDIM,KEY)
C
C     READ COORDINATES (INTERNAL OR CARTESIAN COORDINATES).
C     NOTE.
C      THIS ROUTINE IS THE SAME AS GETGEO OF MOPAC.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*80 LINE,STRING
      CHARACTER*2  ELMNT,ELE
      Character*2  DmyAtm
      CHARACTER*1  SPACE,NINE,ZERO,COMMA
      LOGICAL      LEADSP
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PERTBL/ ELMNT(99)
      DIMENSION C(3,*),IATN(*),NA(*),NB(*),NC(*)
      DIMENSION ISTART(40)
      DATA LENCHR /80/
      Data DmyAtm /'X '/
      DATA COMMA,SPACE,NINE,ZERO /',',' ','9','0'/
 1000 Format(A80)
 9910 Format(1H ,46('X')/
     &       1H ,'NO. OF ATOMS .GT. ',I4,'. CHECK YOUR INPUT DATA.'/
     &       1H ,46('X'))
 9920 Format(1H ,45('X')/
     &       1H ,'ILLEGAL ATOMIC NUMBER. CHECK YOUR INPUT DATA.'/
     &       1H ,45('X'))
 9930 Format(1H ,28('X')/
     &       1H ,'UNRECOGNIZED ELEMENT NAME:',A2/
     &       1H ,28('X'))
 9940 Format(1H ,43('X')/
     &       1H ,'NO. OF ATOMS .LT. 2. CHECK YOUR INPUT DATA.'/
     &       1H ,43('X'))
 9950 Format(1H ,39('X')/
     &       1H ,'EOF IS DETECTED. CHECK YOUR INPUT DATA.'/
     &       1H ,39('X'))
C     ***************************************************************
      NATOMS=0
      NATDUM=0
C
   10 READ(IN,1000,END=80) LINE
c=>   92/Aug A.M.
      Call ToUpr(Line, LenChr)
c<=
      IF(LINE.EQ.SPACE) GO TO 70
      IF(NATDUM.GE.MAXDIM) THEN
        WRITE(IOUT,9910) MAXDIM
        CALL MERROR
      END IF
C
      NATDUM=NATDUM+1
C
      DO 20 I=1,LENCHR
        IF((LINE(I:I).LT.SPACE) .OR. (LINE(I:I).EQ.COMMA)) THEN
          LINE(I:I)=SPACE
        END IF
   20 CONTINUE
```

```
C
C   INITIALIZE ISTART TO INTERPRET BLANKS AS ZERO'S
      DO 30 I=1,10
        ISTART(I)=LENCHR
   30 CONTINUE
C
C     Find initial digit of all numbers, check for leading spaces
C     followed by a character and store in iStart
      LEADSP=.TRUE.
      N=0
      DO 40 I=1,LENCHR
        IF(LEADSP.AND.LINE(I:I).NE.SPACE) THEN
          N=N+1
          ISTART(N)=I
        END IF
        LEADSP=(LINE(I:I).EQ.SPACE)
   40 CONTINUE
C
C     GET ATOMIC NUMBER.
      STRING=LINE(ISTART(1):ISTART(2)-1)
      IF((STRING(1:1) .GE. ZERO) .AND. (STRING(1:1) .LE. NINE)) THEN
        LABEL=READA(STRING,1)
        IF(LABEL.LT.0.OR.LABEL.GT.99) THEN
          WRITE(IOUT,9920)
          CALL MERROR
        ELSE IF(LABEL.EQ.0) THEN
          NATDUM=NATDUM-1
          GO TO 70
        END IF
      ELSE
        ELE=STRING(1:2)
        DO 50 I=1,99
          IF(ELE.EQ.ELMNT(I)) THEN
            LABEL=I
            GO TO 60
          END IF
   50   CONTINUE
c
c     Dummy atoms for Gaussian system.
        If (Ele .Eq. DmyAtm) Then
          Label = 99
          Goto 60
        EndIf
c
        WRITE(IOUT,9930) ELE
        CALL MERROR
      END IF
C
C     ALL O.K.
   60 IF(LABEL.NE.99) THEN
        NATOMS=NATOMS+1
      END IF
C
C     KEY=1 ... Internal coordinates.
C         2 ... CARTESIAN COORDINATES.
      IF(KEY.EQ.1) THEN
        IATN(NATDUM)=LABEL
        C(1,NATDUM) =READA(LINE,ISTART(2))
        C(2,NATDUM) =READA(LINE,ISTART(4))
        C(3,NATDUM) =READA(LINE,ISTART(6))
        NA(NATDUM)  =READA(LINE,ISTART(8))
        NB(NATDUM)  =READA(LINE,ISTART(9))
        NC(NATDUM)  =READA(LINE,ISTART(10))
      ELSE
        IATN(NATDUM)=LABEL
        C(1,NATDUM) =READA(LINE,ISTART(2))
        C(2,NATDUM) =READA(LINE,ISTART(3))
        C(3,NATDUM) =READA(LINE,ISTART(4))
      END IF
C
      GO TO 10
C
   70 IF(NATOMS.LT.2) THEN
        WRITE(IOUT,9940)
        CALL MERROR
      END IF
C
      NA(2)=1
      IF(NA(3).EQ.0) THEN
        NB(3)=1
        NA(3)=2
      END IF
C
      RETURN
C
   80 WRITE(IOUT,9950)
      CALL MERROR
C
      END
C
C@(#)=RDCARD(SUB)
      SUBROUTINE RDCARD(CARD,NBLKS,IBLK)
C
C     READ A CARD AND INVESTIGATE NO OF DATA BLOCKS.
C
C     ON READ:
C      CARD ... CHARACTER*80
C     On output:
C      NBLKS ... NO OF DATA BLOCKS IN THE 'CARD'.
C      IBLK(I) ... FIRST COLUMN OF I-TH DATA BLOCK IN THE 'CARD'.
C     ************************************************************
      CHARACTER*80 CARD
      CHARACTER*1  CHR1,SPACE,COMMA
      COMMON /IO    / IN,IOUT,IFILE(10)
      DIMENSION IBLK(*)
      DATA MAXCLM/80/
      DATA SPACE/' '/,COMMA/','/
 1000 Format(A80)
C     ************************************************************
      NB=0
      NC=0
      READ(IN,1000,END=20) CARD
c=>   92/Aug A.M.
      Call ToUpr(Card, MaxClm)
c<=
      DO 10 I=1,MAXCLM
        CHR1=CARD(I:I)
        IF( (CHR1.EQ.SPACE) .OR. (CHR1.EQ.COMMA) ) THEN
          NC=0
        ELSE
          NC=NC+1
        END IF
        IF(NC.EQ.1) THEN
          NB=NB+1
          IBLK(NB)=I
        END IF
   10 CONTINUE
   20 NBLKS=NB
C
      RETURN
      END
C
C@(#)=RDGEOM(SUB)
      SUBROUTINE RDGEOM
C
C     READ GEOMETRY AND CONVERT INTO CARTESIAN OR INTERNAL COORDINATES.
C     NOTE.
C      1. UNIT OF GEOMETRY : ANGSTROM AND DEGREE.
C      2. BOTH INTERNAL AND CARTESIAN INPUT ARE AVAILABLE.
C      3. IN CASE OF INTERNAL COORDINATE INPUT, INPUT-FORMAT IS THE SAME
C         AS MOPACS'. BUT IN THIS PROGRAM, CONTROL FLAGS FOR GEOMETRY
C         OPTIMIZATION AND REACTION COORDINATE ARE MEANINGLESS.
C      4. VARIABLES.
C         COORD  ... CARTESIAN COORDINATES.
C         ZCoord ... Internal coordinate elements.
C         IAN    ... ARRAY OF ATOMIC NUMBERS.
C         IATN   ... ARRAY OF ATOMIC NUMBERS INCLUDING DUMMY ATOMS
C         NATOMS ... NO. OF ATOMS.
C         NATDUM ... NO. OF ATOMS INCLUDING DUMMY ATOMS.
C      5. DUMMY ATOM IS 'XX', 'X ', OR 99 (Case are ignored).
c     Oct/92 ... A.M.
c     93/Jan ... A.M.
c     93/Mar ... A.M.
C     ************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      Character*256 ChmFrm
      Character*8 Model1H
      Character*2 Elmnt,Elmnt2
      Logical Anchor
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /IOPMOL/ IOPMOL(18)
      COMMON /MOL   / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                NAE,NBE,NE,NBASIS
      COMMON /ZMAT  / ZCOORD(3,MAXAT),IATN(MAXAT),NA(MAXAT),NB(MAXAT),
```

```
     &                NC(MAXAT),NATDUM
      Common /CMethd/ ModelH
      COMMON /PCONST/ PI,TOANG,PCON(18)
      Common /PerTbl/ Elmnt(99)
      Common /PTbl2 / Elmnt2(99)
      Common /Mass  / AtMass(99)
      Common /Anchor/ Anchor
      DATA ONE/1.0D+00/,F180/180.0D+00/
C     ****************************************************************
C     MAXAT : FROM INCLUDE(MSSIZE)
      MAXDIM=MAXAT
C
      ICOORD=IOPMOL( 1)
      ISYMM =IOPMOL( 2)
      IATDIS=IOPMOL(11)
C
C     READ GEOMETRY.
      IF(ICOORD.EQ.1) THEN
        TORAD=PI/F180
        CALL RCOORD(ZCOORD,IATN,NA,NB,NC,NATOMS,NATDUM,MAXDIM,ICOORD)
        CALL ZTOC (ZCOORD,COORD,TORAD,NA,NB,NC,NATDUM)
      ELSE
        TODEG=F180/PI
        CALL RCOORD(COORD,IATN,NA,NB,NC,NATOMS,NATDUM,MAXDIM,ICOORD)
        CALL CTOZ(COORD,ZCOORD,TODEG,NA,NB,NC,NATDUM,PI)
      END IF
C
C     PRINT INPUT ORIENTATION.
      CALL CZPRNT(COORD,ZCOORD,IATN,NA,NB,NC,NATDUM)
C
C     THEN, REDUCE DUMMY ATOMS, IF ANY.
      N=0
      DO 10 I=1,NATDUM
        IF(IATN(I).NE.99) THEN
          N=N+1
          IAN(N)=IATN(I)
          COORD(1,N)=COORD(1,I)
          COORD(2,N)=COORD(2,I)
          COORD(3,N)=COORD(3,I)
        END IF
   10 CONTINUE
C
C     TRANSLATE MOLECULE.
C     THE ORIGIN OF CARTESIAN COORDINATE SYSTEM : THE CENTER OF VALENCE
C                                                 CHARGE.
      CALL CTRNS(COORD,IAN,NATOMS,IOUT)
C
C     GET POINT GROUP.
      IF(ISYMM.EQ.1) THEN
        CALL SYMM
      END IF
C
C     PRINT STANDARD ORIENTATION.
      CALL CPRINT(COORD,IAN,NATOMS)
c
c     Print chemical formula and molecular weight.
      Call CForm(IOut,IAN,NAtoms,Atmass,Elmnt2,ChmFrm,NChrs,WMol)
c
      If (Anchor) Call AncMol (NAtDum,NAtoms,IFile(10),
     &                         ChmFrm,NChrs,WMol,
     &                         ModelH,
     &                         IAtN,NA,NB,NC,IAN,
     &                         ZCoord,Coord,
     &                         Elmnt)
C
C     PRINT INTERATOMIC DISTANCES.
      IF(IATDIS.EQ.1) THEN
        CALL ATDIST(COORD,IAN,NATOMS)
      END IF
C
C     CONVERT ANGSTROM TO BOHR.
      TOBOHR=ONE/TOANG
      DO 20 I=1,NATOMS
      DO 20 J=1,3
        COORD(J,I)=COORD(J,I)*TOBOHR
   20 CONTINUE
C
      RETURN
      END
      subroutine rdopt( irdend )
C=======================================================================*
C                                                                       *
C      @@@@@   @@@@@    @@@@   @@@@@    @@@@@                           *
```

```
c      @    @  @    @  @    @  @    @      @                            *
c      @    @  @    @  @    @  @    @      @                            *
c      @@@@@   @    @  @    @  @@@@@       @                            *
c      @   @   @    @  @    @  @           @                            *
c      @    @  @@@@@    @@@@   @           @                            *
C=======================================================================*
C     Read and print keywords.
C     Note.
C      1. Columns from 1 to 80 are available in an option card.
C      2. No. of option cards is within the value of maxcrd(maxncr).
C      3. This routine is terminated by a blank card.
C      4. Keywords are separated by a space or a comma.
c     93/Jun. A.M.
C     ****************************************************************
      implicit real*8 (a-h,o-z)
c
      include 'MSSIZE'
      include 'sdci00.h'
c
      Character*800 OPTCRD, Crd2
      Character*80  ICARD, iCrd2
      CHARACTER*1   IBL1,MINUS
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /CARD1 / OPTCRD,ICARD,Crd2,iCrd2
      COMMON /CARD2 / MAXCRD,LENOCR,LENICR,NCARDS,ISMIN,IEMAX
      DATA NEXEC/0/,LOCARD/800/,LICARD/80/,MAXNCR/9/
      DATA IBL1/' '/,MINUS/'-'/
 1000 Format(A80)
 2000 Format(1H ,80A1)
 9910 Format(1H ,43('X')/
     &       1H ,'THERE WAS NO CARD TO READ IN ROUTINE RDOPT.'/
     &       1H ,43('X'))
 9920 Format(1H ,43('X')/
     &       1H ,'TOO MANY OPTION CARDS. NO. OF CARDS .LE.',I2,'.'/
     &       1H ,43('X'))
C     ****************************************************************

      namelist/ sdci / isdci0

      MAXCRD=MAXNCR
      LENOCR=LOCARD
      LENICR=LICARD

      OPTCRD=IBL1
      ISMIN =LENICR
      IEMAX =1
c========( Fri Sep  8 10:27:03 JST 1995 )======
      rewind in
      read( in, sdci )
      write( iout, sdci )

      N=0
   10 READ(IN,1000,END=80) ICARD
c=>   92/Aug, 93/Jun A.M.
      ICrd2=ICard
      Call ToUpr(ICard, LenICr)
c<=
      IF(ICARD.EQ.IBL1) THEN
        IF(N.EQ.0) THEN
          IF(NEXEC.EQ.0) THEN
            WRITE(IOUT,9910)
            CALL MERROR
          ELSE
            IRDEND=1
            RETURN
          END IF
        ELSE
          GO TO 60
        END IF
      ELSE IF(N.EQ.MAXCRD) THEN
        WRITE(IOUT,9920) MAXCRD
        CALL MERROR
      END IF
C
      N=N+1
C
      DO 20 I=1,LENICR
        IF(ICARD(I:I).NE.IBL1) GO TO 30
   20 CONTINUE
   30 ISTART=I
      DO 40 I=LENICR,1,-1
        IF(ICARD(I:I).NE.IBL1) GO TO 50
   40 CONTINUE
```

```
   50 IEND=I
C
      IF(ISTART.LT.ISMIN) THEN
        ISMIN=ISTART
      END IF
      IF(IEND  .GT.IEMAX) THEN
        IEMAX=IEND
      END IF
C
      IS=(N-1)*(LENICR+1)+2
      IE=IS+LENICR-1
      OPTCRD(IS:IE)=ICARD
      Crd2(IS:IE)  =ICrd2
C
      GO TO 10
C
   60 NCARDS=N
C
      WRITE(IOUT,2000) (MINUS,IJKL=ISMIN,IEMAX)
      DO 70 I=1,NCARDS
        IS=(I-1)*(LENICR+1)+2
        IE=IS+LENICR-1
        ICrd2=Crd2(IS:IE)
        WRITE(IOUT,2000) (ICrd2(J:J),J=ISMIN,IEMAX)
   70 CONTINUE
      WRITE(IOUT,2000) (MINUS,IJKL=ISMIN,IEMAX)
C
      NEXEC=1
C
      RETURN
C
   80 IF(NEXEC.EQ.0) THEN
        WRITE(IOUT,9910)
        CALL MERROR
      ELSE
        IRDEND=1
        RETURN
      END IF
C
      END
C
C@(#)=RDTITL(SUB)
      SUBROUTINE RDTITL(JRDEND)
C
C     READ TITLE AND PRINT IT OUT.
C     NOTE.
C      1. Columns from 1 to 80 are available in title card.
C      2. NO. OF TITLE CARDS IS WITHIN THE VALUE OF MAXCRD. SEE ROUTINE,
C         RDOPT.
C      3. THIS ROUTINE IS TERMINATED BY A BLANK CARD.
c     93/Jun A.M.
C     ***************************************************************
c$ohtawara
      character*80 title

      CHARACTER*800 CARDS,Crd2
      CHARACTER*80  ICARD,iCrd2
      CHARACTER*1   IBL1,MINUS
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /CARD1 / CARDS,ICARD,Crd2,iCrd2
      COMMON /CARD2 / MAXCRD,LENOCR,LENICR,IDUMMY(2)
c$ohtawara
      common /my_title/ title

      DATA NEXEC/0/
      DATA IBL1/' '/,MINUS/'-'/
 1000 Format(A80)
 2000 Format(1H ,80A1)
 9910 Format(1H ,42('X')/
     &       1H ,'TOO MANY TITLE CARDS. NO. OF CARDS .LE.',I2,'.'/
     &       1H ,42('X'))
 9920 Format(1H ,44('X')/
     &       1H ,'THERE WAS NO CARD TO READ IN ROUTINE RDTITL.'/
     &       1H ,44('X'))
C     ***************************************************************
      ISMIN=LENICR
      IEMAX=1
C
      N=0
   10 READ(IN,1000,END=70) ICARD
      IF(ICARD.NE.IBL1) THEN
        N=N+1
        IF(N.GT.MAXCRD) THEN
          WRITE(IOUT,9910) MAXCRD
          CALL MERROR
        END IF
        IS=(N-1)*LENICR+1
        IE=N*LENICR
        CARDS(IS:IE)=ICARD
        DO 20 I=1,LENICR
          IF(ICARD(I:I).NE.IBL1) GO TO 30
   20   CONTINUE
   30   ISTART=I
        DO 40 I=LENICR,1,-1
          IF(ICARD(I:I).NE.IBL1) GO TO 50
   40   CONTINUE
   50   IEND=I
        IF(ISTART.LT.ISMIN) THEN
          ISMIN=ISTART
        END IF
        IF(IEND  .GT.IEMAX) THEN
          IEMAX=IEND
        END IF
        GO TO 10
      ELSE IF(N.EQ.0.AND.NEXEC.EQ.0) THEN
        WRITE(IOUT,9920)
        CALL MERROR
      ELSE IF(N.EQ.0) THEN
        NEXEC =0
        JRDEND=1
        RETURN
      END IF
C
      WRITE(IOUT,2000) (MINUS,IJKL=ISMIN,IEMAX)
      DO 60 I=1,N
        IS=(I-1)*LENICR+1
        IE=I*LENICR
        ICARD=CARDS(IS:IE)
        WRITE(IOUT,2000) (ICARD(J:J),J=ISMIN,IEMAX)
   60 CONTINUE
      WRITE(IOUT,2000) (MINUS,IJKL=ISMIN,IEMAX)

c$ohtawara
      title=icard

      NEXEC=NEXEC+1
C
      RETURN
C
   70 IF(NEXEC.EQ.0) THEN
        WRITE(IOUT,9920)
        CALL MERROR
      ELSE
        JRDEND=2
        RETURN
      END IF
C
      END
C
C@(#)=READA(FUN) ... MOPAC(READA)
      DOUBLE PRECISION FUNCTION READA(A,ISTART)
C
C     ==========================================================
C     MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC
C     ==========================================================
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER*1 A(80)
      NINE=ICHAR('9')
      IZERO=ICHAR('0')
      MINUS=ICHAR('-')
      IDOT=ICHAR('.')
C*MOS
      MDIG=17
C*MOS
      IDIG=0
      C1=0
      C2=0
      ONE=1.D0
      X = 1.D0
      DO 10 J=ISTART,79
         N=ICHAR(A(J))
         M=ICHAR(A(J+1))
         IF(N.LE.NINE.AND.N.GE.IZERO .OR.N.EQ.IDOT)GOTO 2
   10
         IF(N.EQ.MINUS.AND.(M.LE.NINE.AND.M.GE.IZERO
```

```
     1 .OR. M.EQ.IDOT)) GOTO 20
   10 CONTINUE
      READA=0.D0
      RETURN
   20 CONTINUE
      DO 30 I=J,80
         N=ICHAR(A(I))
         IF(N.LE.NINE.AND.N.GE.IZERO) THEN
            IDIG=IDIG+1
C*MOS       IF (IDIG.GT.10) GOTO 60
            IF (IDIG.GT.MDIG) GOTO 60
            C1=C1*10+N-IZERO
         ELSEIF(N.EQ.MINUS.AND.I.EQ.J) THEN
            ONE=-1.D0
         ELSEIF(N.EQ.IDOT) THEN
            GOTO 40
         ELSE
            GOTO 60
         ENDIF
   30 CONTINUE
   40 CONTINUE
      IDIG=0
      DO 50 II=I+1,80
         N=ICHAR(A(II))
         IF(N.LE.NINE.AND.N.GE.IZERO) THEN
            IDIG=IDIG+1
C*MOS       IF (IDIG.GT.10) GOTO 60
            IF (IDIG.GT.MDIG) GOTO 60
            C2=C2*10+N-IZERO
            X = X /10
         ELSEIF(N.EQ.MINUS.AND.II.EQ.I) THEN
            X=-X
         ELSE
            GOTO 60
         ENDIF
   50 CONTINUE
C
C PUT THE PIECES TOGETHER
C
   60 CONTINUE
      READA= ONE * ( C1 + C2 * X)
      RETURN
      END
C
C@(#)=RELVEC(SUB) ... CNDO2/3R(RELVEC)
      SUBROUTINE RELVEC(C1,C2,R,E)
C
C     ==========================================================
C     CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R
C     ==========================================================
C     ************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION E(3),C1(3),C2(3)
C     ************************************************************
      X = 0.D0
      DO 10 I=1,3
      E(I) = C2(I)-C1(I)
      X = X+E(I)**2
   10 CONTINUE
      R=DSQRT(X)
      DO 40 I=1,3
      IF (R.GT..000001D0) GO TO 30
   20 GO TO 40
   30 E(I) =E(I)/R
   40 CONTINUE
      RETURN
      END
      subroutine rhfclo
C=================================================================*
C                                                                 *
C      @@@@@   @    @  @@@@@@   @@@@   @        @@@@             *
C      @    @  @    @  @        @    @  @        @    @           *
C      @    @  @@@@@@  @@@@@   @       @        @    @           *
C      @@@@@   @    @  @        @       @        @    @           *
C      @   @   @    @  @        @    @  @        @    @           *
C      @    @  @    @  @         @@@@   @@@@@@   @@@@             *
C=================================================================*
C
C     SOLVE RESTRICTED CLOSED SHELL HARTREE-POCK EQUATION BY USING THE
C     ITERATIVE PROCEDURE.
C
C     SCF CRITERION IN THIS PROGRAM IS DENSITY MATRIX CONVERGENCE WITH
C     MAGNITUDE LESS THAN 'SCFCRT'.
C
C     CAUTION.
C      1. DENSITY MATRIX POINTERS (IP0-IP3) ARE USED IN THIS PROGRAM.
C         DENSITY MATRIX ... D(MBPCK,4)
C         IP0 ...  I-TH    CYCLE. D(MBPCK,IP0):CURRENT DENSITY MATRIX
C         IP1 ... (I-1)-TH CYCLE.
C         IP2 ... (I-2)-TH CYCLE.
C         IP3 ... (I-3)-TH CYCLE.
C      2. THE ARRAYS OF PREVIOUS THREE DENSITY MATRICES CHANGE INTO
C         DIFFERENCE BETWEEN DENSITY MATRICES (I & I-1, I-1 & I-2,
C         AND I-2 & I-3) IN ROUTINE 'CONCLO'. SEE 'CONCLO'.
C     ************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci02.h'
c
      CHARACTER*13 CHR13
      LOGICAL LOGCON,LOGEXT
cfj   INCLUDE (MSSIZE)
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /METHOD/ METHOD
      COMMON /IOPSCF/ IOPSCF(18)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /CRITRN/ SCFCRT,CRTDUM(9)
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                IBTOA(MAXBAS)
      COMMON /ENERGY/ ENERGY
      COMMON /WORK01/ F(MBPCK),HCORE(MBPCK),D(MBPCK,4),DUMMY(1)
      COMMON /WORK02/ C(MAXBAS,MAXBAS),EIG(MAXBAS),SCRTCH(MAXBAS,19)
      DIMENSION DEXT(MBPCK)
      DIMENSION CHR13(2)
      EQUIVALENCE(C(1,1),DEXT(1))
      DATA ZERO/0.0D+00/,TWO/2.0D+00/
      DATA CHR13/'Spiral (4-pt)','Aitken (3-pt)'/
 2000 Format(1H ,'<<<<<< Enter RHFClo >>>>>>')
 2010 Format(1H ,'RHF closed shell SCF ... Maximum cycles =',I5/
     &       1H ,'SCF criterion ... Convergence on density matrix =',
     &       1PD11.4)
 2020 Format(1H ,79('-'))
 2030 Format(1H , 1X,'Iteration',4X,'Electronic energy',11X,
     &                'Convergence',12X,'Extrapolation'/
     &       1H ,18X,'(Hartree)',11X,'Energy',7X,'Density')
 2040 Format(1H ,I6,5X,F18.9)
 2050 Format(1H ,I6,5X,F18.9,6X,1PD11.4,2X,1PD11.4)
 2060 Format(1H ,  11X,F18.9,2X,'...   Non-variational value',7X,A13)
 2070 Format(1H ,'Density converged to',1PD12.5,2X,'... SCF criterion ',
     &                'satisfied.')
 2080 Format(1H ,11('=')/
     &       1H ,'Fock matrix'/
     &       1H ,11('='))
 9910 Format(1H ,'WARNING! DENSITY CONVERGED BUT ENERGY NOT LOWEST.')
 9920 Format(1H ,40('X')/
     &       1H ,'SCF CRITERION NOT MET WITHIN ',I3,' CYCLES.'/
     &       1H ,40('X'))
 9930 Format(1H ,40('X')/
     &       1H ,'DENSITY CONVERGED BUT ENERGY NOT LOWEST.'/
     &       1H ,40('X'))
C     ************************************************************
      ITMAX =IOPSCF( 1)
      IPRFCK=IOPSCF(11)
C
      ICYC  =0
      JCYC  =0
      LOGCON=.FALSE.
      LOGEXT=.FALSE.
      NTT   =NBASIS*(NBASIS+1)/2
C
      EMIN  =ZERO
C
C     ARRAY HCORE ALWAYS CONTAINS CORE HAMILTONIAN IN THIS ROUTINE.
C     AND NOW,
C     ARRAY C CONTAINS EIGENVECTORS OF HUCKEL HAMILTONIAN MATRIX
C     OBTAINED FROM INITIAL GUESS CALCULATION.
      WRITE(IOUT,2000)
      WRITE(IOUT,2010) ITMAX,SCFCRT
      WRITE(IOUT,2020)
      WRITE(IOUT,2030)
      WRITE(IOUT,2020)
C                                                   --------------
C                                                   SCF LOOP START
C                                                   --------------
```

```
C     ICYC ... NO. OF SCF ITERATIONS.
C     JCYC ... NO. OF SCF ITERATIONS BEFORE EXTRAPOLATION. JCYC IS
C              CLEARED AFTER PERFORMING EXTRAPOLATION.
   10 ICYC=ICYC+1
      IF(ICYC.GT.ITMAX) GO TO 9000
      JCYC=JCYC+1
C
C     DEFINE DENSITY MATRIX POINTERS.
      IP0=MOD(ICYC-1,4)+1
      IP1=MOD(ICYC-2,4)+1
      IP2=MOD(ICYC-3,4)+1
      IP3=MOD(ICYC-4,4)+1
C
C     GET DENSITY MATRIX.
      CALL GETD(C,D(1,IP0),TWO,MAXBAS,NBASIS,NAE)
C
C     ESTIMATE CONVERGENCE ON DENSITY MATRIX AND TRY TO GET EXTRAPOLATED
C     DENSITY MATRIX. PREVIOUS DENSITY MATRIX (D(1,IP1)) IS DESTROYED
C     HERE. ==>  D(I,IP1)=D(I,IP0)-D(I,IP1)
      CALL CONCLO(D,DEXT,SCFCRT,DCON,MBPCK,NBASIS,NTT,JCYC,I4OR3,
     &            IP0,IP1,IP2,IP3,LOGCON,LOGEXT)
C
C     GET FOCK MATRIX
      CALL GETF(D(1,IP0),HCORE,F,IAN,NATOMS,NBASIS,NTT,METHOD)
C
C     NOW,
C     ARRAY D(I,IP0) CONTAINS CURRENT DENSITY MATRIX.
C     ARRAY F CONTAINS CURRENT FOCK MATRIX.
C
C     GET ELECTRONIC ENERGY FOR CURRENT DENSITY MATRIX.
      ENERGY=SCFENG(D(1,IP0),HCORE,F,NBASIS,NTT)
C
C     PRINTING OF SCF INFORMATION.
C     CAUTION.
C      ELECTRONIC ENERGY OBTAINED FROM EXTRAPOLATED DENSITY MATRIX IS
C      NOT BASED ON VARIATIONAL PRINCIPLE.
      IF     (ICYC.EQ.1 .OR. JCYC.EQ.1) THEN
        WRITE(IOUT,2040) ICYC,ENERGY
        EOLD=ENERGY
      ELSE
        ECON=EOLD-ENERGY
        WRITE(IOUT,2050) ICYC,ENERGY,ECON,DCON
        IF(ENERGY.LT.EMIN) THEN
          EMIN=ENERGY
        END IF
        EOLD=ENERGY
      END IF
C
C     IF EXTRAPOLATION IS PERFORMED, GET ELECTRONIC ENERGY FOR
C     EXTRAPOLATED DENSITY MATRIX.
C     CAUTION.
C      ELECTRONIC ENERGY OBTAINED FROM EXTRAPOLATED DENSITY MATRIX IS
C      NOT BASED ON VARIATIONAL PRINCIPLE.
      IF(LOGEXT) THEN
        CALL GETF(DEXT,HCORE,F,IAN,NATOMS,NBASIS,NTT,METHOD)
        EEXT=SCFENG(DEXT,HCORE,F,NBASIS,NTT)
        WRITE(IOUT,2060) EEXT,CHR13(I4OR3)
        LOGEXT=.FALSE.
        JCYC  =0
      END IF
C
C     IF SCF CRITERIA MET, EXIT SCF LOOP.
      IF(LOGCON) GO TO 20
C
c========( Mon Sep 25 13:24:27 JST 1995 )======
c     save fock matrix elements
      nfckm = nbasis*( nbasis + 1 )/2
      do 1000 nfck = 1, nfckm
        fockmt( nfck ) = f( nfck )
 1000 continue
c
C     IF DENSITY NOT CONVERGED, DIAGONALIZE FOCK MATRIX.
      CALL DIAG1(F,C,EIG,NBASIS,MAXBAS,SCRTCH,IOUT)
C
C     NOW,
C     ARRAY C CONTAINS MOLECULAR ORBITAL COEFFICIENTS TO GET DENSITY
C     MATRIX OF NEXT CYCLE.
      GO TO 10
C                                                  ------------
C                                                  SCF LOOP END
C     EPSLN() ... MACHINE EPSILON.                 ------------
   20 IF     ( (ENERGY-EMIN) .GT. SCFCRT  ) THEN
        GO TO 9010
      ELSE IF( (ENERGY-EMIN) .GT. EPSLN() ) THEN
        WRITE(IOUT,9910)
      END IF
C
C     DENSITY CONVERGED WITHIN ITMAX.
      WRITE(IOUT,2020)
      WRITE(IOUT,2070) DCON
c========( Thu Oct 19 17:35:20 JST 1995 )======
      h000 = geth0( d(1,ip0), hcore, eig, nbasis, ntt )
c$$$       ENERGY = h000
      write(iout,*)' &&&& ENERGY, h000 = ',ENERGY, h000
c========( Thu Oct 19 17:35:20 JST 1995 )======
C
C     MOVE CURRENT DENSITY MATRIX, D(I,IP0) TO D(I,1).
      IF(IP0.NE.1) THEN
        DO 30 I=1,NTT
          D(I,1)=D(I,IP0)
   30   CONTINUE
      END IF
C
C     NOW,
C     ARRAY D(I,1) CONTAINS DENSITY MATRIX.
C     ARRAY C CONTAINS MOLECULAR ORBITAL COEFFICIENTS.
C     ARRAY F CONTAINS FOCK MATRIX.
C     ARRAY EIG CONTAINS ORBITAL ENERGIES.
C     CAUTION.
C      ARRAYS D(I,1),C, AND EIG ARE USED IN THE FOLLOWING ROUTINES.
C
C     PRINTING OF FINAL FOCK MATRIX.
      IF(IPRFCK.EQ.1) THEN
        WRITE(IOUT,2080)
        CALL SSMOUT(F,IAN,IBTOA,NLBAS,NBASIS)
      END IF
C
C     ALL DONE. RETURN TO CALLER.
      CALL CLOCKM('RHFClo.')
      RETURN
C                                                  --------------
C                                                  ERROR OCCURRED
C     NO. OF ITERATION EXCEED ITMAX.               --------------
 9000 WRITE(IOUT,9920) ITMAX
      CALL MERROR
 9010 WRITE(IOUT,9930)
      CALL MERROR
C
      END
C
C@(#)=RWPRM1(SUB)
      SUBROUTINE RWPRM1(EP,BP,ZP,GP,PK,IAN,IFLG,NATOMS,IRDPRM,IPRPRM,
     &                  METHOD,IGAMMA,MAXIAN)
C
C     READ ATOMIC PARAMETERS AND PRINT THEM OUT.
C     93/Apr. A.M.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*12 CHR12(2)
      CHARACTER*2  CHR2,ELMNT,Elmnt2,EL
      CHARACTER*1  CHR1(2),IBL1
      LOGICAL LOGRD,LOGPR1,LOGPR2
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /PERTBL/ ELMNT(99)
      Common /PTbl2 / Elmnt2(99)
      DIMENSION EP(MAXIAN,3),BP(MAXIAN,3),ZP(*),GP(*),PK(*),IAN(*)
      DIMENSION IFLG(*)
      DIMENSION E(3),B(3)
      EQUIVALENCE (CHR2,CHR1(1))
      DATA CHR12/'(I+A)/2 (eV)','  I (eV)  '/
      DATA ZERO/0.0D+00/
      DATA IBL1/' '/
 1000 Format(A2,6X,9F8.0)
 2000 Format(1H ,17('=')/
     &       1H ,'Atomic parameters'/
     &       1H ,17('='))
 2010 Format(1H ,75('-'))
 2020 Format(1H ,' At',8X,A12,11X,'-Beta0 (eV)',8X,'Gamma',4X,'Zeta',3X,
     &           'Kappa'/
     &       1H ,9X,'s',6X,'p',6X,'d',8X,'s',6X,'p',6X,'d',6X,'(eV)',
     &           6X,'(Bohr**-1)')
 2030 Format(1H ,1X,A2,3X,3(F6.3,1X),2X,3(F6.3,1X),2X,F6.3,2(2X,F6.3))
 2040 Format(1H ,'Huzinaga''s Klondike parameters, kappa, are not ',
     &           'used here.')
 9910 Format(1H ,30('X')/
     &       1H ,'UNRECOGNIZABLE ELEMENT NAME:',A2/
```

```
     &       1H ,30('X'))
 9920 Format(1H ,65('X')/
     &       1H ,'WARNING! THE SAME ELEMENT AS FORMER INPUT LIST WAS ',
     &           'APPEARED AGAIN.'/
     &       1H ,65('X'))
 9930 Format(1H ,41('X')/
     &       1H ,'WARNING! THERE WAS NO POSITIVE PARAMETER.'/
     &       1H ,41('X'))
 9940 Format(1H ,53('X')/
     &       1H ,'WARNING! THERE WAS NO CARD TO READ IN ROUTINE RWPRM1.'
     &      /1H ,53('X'))
 9950 Format(1H ,52('X')/
     &       1H ,'WARNING! EXPONENT FOR SP3 CARBON ATOMS NOT REPLACED.'/
     &       1H ,52('X'))
 9960 Format(1H ,59('X')/
     &       1H ,'WARNING! EXPONENT FOR SP3 CARBON ATOMS IS NOT ',
     &           'LISTED BELOW.'/
     &       1H ,59('X'))
 9970 Format(1H ,39('X')/
     &       1H ,'EOF IS DETECTED. CHECK YOUR INPUT DATA.'/
     &       1H ,39('X'))
C     ***************************************************************
      LOGRD =IRDPRM.EQ.1
      LOGPR1=IPRPRM.EQ.1
      LOGPR2=IPRPRM.EQ.3
C
      IF((.NOT.LOGRD) .AND. (.NOT.LOGPR1) .AND. (.NOT.LOGPR2)) RETURN
C
C     SET FLAG.
      DO 10 I=1,MAXIAN
        IFLG(I)=0
   10 CONTINUE
      DO 20 I=1,NATOMS
        IA=IAN(I)
        IFLG(IA)=1
   20 CONTINUE
C
      IF(.NOT.LOGRD) GO TO 80
C
C     READ ATOMIC PARAMETERS.
      JFLG=0
   30 READ(IN,1000,END=110) CHR2,(E(I),I=1,3),(B(I),I=1,3),G,Z,P
c
      Len2=Len(Chr2)
      Call ToUpr (Chr2, Len2)
c
      IF(CHR2.EQ.IBL1) GO TO 70
C
      JFLG=1
C
      IF(CHR1(1).EQ.IBL1) THEN
        CHR1(1)=CHR1(2)
        CHR1(2)=IBL1
      END IF
      DO 40 I=1,MAXIAN
        EL=ELMNT(I)
        IF(CHR2.EQ.EL) GO TO 50
   40 CONTINUE
      WRITE(IOUT,9910) CHR2
      CALL MERROR
C
   50 IA=I
      IF(IFLG(IA).EQ.-1) THEN
        WRITE(IOUT,9920)
        GO TO 30
      END IF
C
      IFLG(IA)=-1
C
      KFLG=0
      DO 60 I=1,3
        IF(E(I).GT.ZERO) THEN
          EP(IA,I)=E(I)
          KFLG=1
        END IF
        IF(B(I).GT.ZERO) THEN
          BP(IA,I)=B(I)
          KFLG=1
        END IF
   60 CONTINUE
      IF(G.GT.ZERO) THEN
        GP(IA)=G
        KFLG=1
      END IF
      IF(Z.GT.ZERO) THEN
        ZP(IA)=Z
        KFLG=1
      END IF
      IF((P.GT.ZERO) .AND. (IGAMMA.EQ.7)) THEN
        PK(IA)=P
        KFLG=1
      END IF
C
      IF(KFLG.EQ.0) THEN
        WRITE(IOUT,9930)
      END IF
C
      GO TO 30
C
   70 IF(JFLG.EQ.0) THEN
        WRITE(IOUT,9940)
      END IF
      IF( (METHOD.EQ.4) .AND. (IFLG(6).EQ.-1) ) THEN
        WRITE(IOUT,9950)
      END IF
C
   80 IF((.NOT.LOGPR1) .AND. (.NOT.LOGPR2)) RETURN
C
C     PRINT OUT ATOMIC PARAMETERS.
      IF(METHOD.EQ.4) THEN
        IF(IFLG(6).NE.0) THEN
          WRITE(IOUT,9960)
        ELSE IF(LOGPR2) THEN
          WRITE(IOUT,9960)
        END IF
      END IF
C
      IF(METHOD.LE.2) THEN
        MA=1
      ELSE
        MA=2
      END IF
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010)
      WRITE(IOUT,2020) CHR12(MA)
      WRITE(IOUT,2010)
C
      IF(LOGPR1) THEN
        DO 90 I=1,MAXIAN
          IF(IFLG(I).NE.0) THEN
            WRITE(IOUT,2030) Elmnt2(I),(EP(I,J),J=1,3),(BP(I,J),J=1,3),
     &                       GP(I),ZP(I),PK(I)
          END IF
   90   CONTINUE
      ELSE
        DO 100 I=1,MAXIAN
          WRITE(IOUT,2030)   Elmnt2(I),(EP(I,J),J=1,3),(BP(I,J),J=1,3),
     &                       GP(I),ZP(I),PK(I)
  100   CONTINUE
      END IF
C
      WRITE(IOUT,2010)
      IF(IGAMMA.NE.7) THEN
        WRITE(IOUT,2040)
      END IF
C
      RETURN
C
  110 WRITE(IOUT,9970)
      CALL MERROR
C
      END
C
C@(#)=RWPRM2(SUB)
      SUBROUTINE RWPRM2(EN,BETA,ZETA,GAA,PKLN,IAN,IFLG,NATOMS,IRDPRM,
     &                  IPRPRM,METHOD,IGAMMA,MAXAT)
C
C     READ ATOMIC PARAMETERS AND PRINT THEM OUT.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*12 CHR12(2)
      CHARACTER*2  Elmnt2
      COMMON /IO    / IN,IOUT,IFILE(10)
      Common /PTb12 / Elmnt2(99)
      DIMENSION EN(MAXAT,3),BETA(MAXAT,3),ZETA(MAXAT),GAA(MAXAT)
```

```
      DIMENSION PKLN(MAXAT),IAN(*),IFLG(*)
      DIMENSION E(3),B(3)
      DATA CHR12/'(I+A)/2 (eV)','  I (eV)  '/
      DATA ZERO/0.0D+00/
 1000 Format(I4,4X,9F8.0)
 2000 Format(1H ,17('=')/
     &       1H ,'Atomic parameters'/
     &       1H ,17('='))
 2010 Format(1H ,79('-'))
 2020 Format(1H ,' CN  At',8X,A12,11X,'-Beta0 (EV)',8X,'Gamma',4X,
     &          'Zeta    Kappa'/
     &       1H ,13X,'s',6X,'p',6X,'d',8X,'s',6X,'p',6X,'d',6X,'(eV)',
     &          6X,'(Bohr**-1)')
 2030 Format(1H ,I3,2X,A2,3X,3(F6.3,1X),2X,3(F6.3,1X),2X,F6.3,
     &          2(2X,F6.3))
 2040 Format(1H ,'Huzinaga''s Klondike parameters, kappa, are not ',
     &          'used here.')
 9910 Format(1H ,36('X')/
     &       1H ,'CENTER NO. OF ATOM WAS OUT OF RANGE.'/
     &       1H ,36('X'))
 9920 Format(1H ,69('X')/
     &       1H ,'WARNING! THE SAME CENTER NO. AS FORMER INPUT LIST ',
     &          'WAS APPEARED AGAIN.'/
     &       1H ,69('X'))
 9930 Format(1H ,41('X')/
     &       1H ,'WARNING! THERE WAS NO POSITIVE PARAMETER.'/
     &       1H ,41('X'))
 9940 Format(1H ,53('X')/
     &       1H ,'WARNING! THERE WAS NO CARD TO READ IN ROUTINE RWPRM1.'
     &      /1H ,53('X'))
 9950 Format(1H ,39('X')/
     &       1H ,'EOF IS DETECTED. CHECK YOUR INPUT DATA.'/
     &       1H ,39('X'))
C     ***************************************************************
      IF(IRDPRM.NE.2) GO TO 50
C
C     READ ATOMIC PARAMETERS.
      N=0
   10 READ(IN,1000,END=70) ICN,(E(I),I=1,3),(B(I),I=1,3),G,Z,P
      IF(ICN.EQ.0) THEN
        GO TO 40
      ELSE IF((ICN.LT.0) .OR. (ICN.GT.NATOMS)) THEN
        WRITE(IOUT,9910)
        CALL MERROR
      END IF
C
      N=N+1
C
      IF(N.NE.0) THEN
        NM1=N-1
        DO 20 I=1,NM1
          IF(ICN.EQ.IFLG(I)) THEN
            WRITE(IOUT,9920)
            GO TO 10
          END IF
   20   CONTINUE
      END IF
C
      IFLG(N)=ICN
C
      KFLG=0
      DO 30 I=1,3
        IF(E(I).GT.ZERO) THEN
          EN(ICN,I)=E(I)
          KFLG=1
        END IF
        IF(B(I).GT.ZERO) THEN
          BETA(ICN,I)=B(I)
          KFLG=1
        END IF
   30 CONTINUE
      IF(G.GT.ZERO) THEN
        GAA(ICN)=G
        KFLG=1
      END IF
      IF(Z.GT.ZERO) THEN
        ZETA(ICN)=Z
        KFLG=1
      END IF
      IF((P.GT.ZERO) .AND. (IGAMMA.EQ.7)) THEN
        PKLN(ICN)=P
        KFLG=1
      END IF
C
      IF(KFLG.EQ.0) THEN
        WRITE(IOUT,9930)
      END IF
C
      GO TO 10
C
   40 IF(N.EQ.0) THEN
        WRITE(IOUT,9940)
      END IF
C
   50 IF(IPRPRM.NE.2) RETURN
C
C     PRINT OUT ATOMIC PARAMETERS.
      IF(METHOD.LE.2) THEN
        MA=1
      ELSE
        MA=2
      END IF
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010)
      WRITE(IOUT,2020) CHR12(MA)
      WRITE(IOUT,2010)
      DO 60 I=1,NATOMS
        IA=IAN(I)
        WRITE(IOUT,2030) I,Elmnt2(IA),(EN(I,J),J=1,3),(BETA(I,J),J=1,3),
     &                   GAA(I),ZETA(I),PKLN(I)
   60 CONTINUE
      WRITE(IOUT,2010)
      IF(IGAMMA.NE.7) THEN
        WRITE(IOUT,2040)
      END IF
C
      RETURN
C
   70 WRITE(IOUT,9950)
      CALL MERROR
C
      END
C
C@(#)=SCFENG(FUN)
      DOUBLE PRECISION FUNCTION SCFENG(D,HCORE,F,NBASIS,NTT)
C
C     GET SCF ELECTRONIC ENERGY.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION D(*),HCORE(*),F(*)
      DATA ZERO/0.0D+00/,PT5/0.5D+00/
C     ***************************************************************
      E=ZERO
      DO 10 I=1,NTT
        E=E+D(I)*(HCORE(I)+F(I))
   10 CONTINUE
      DO 20 I=1,NBASIS
        MA=I*(I+1)/2
        E=E-PT5*D(MA)*(HCORE(MA)+F(MA))
   20 CONTINUE
C
      SCFENG=E
C
      RETURN
      END
C
C@(#)=SCI(SUB)
      SUBROUTINE SCI
C
C     SINGLE EXCITATION CI CALCULATION FOR SINGLET AND TRIPLET STATES.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*7 CHR7(2)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /IOPCI / IOPCI(18)
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                NOCS,NVCS,NCSFS
      COMMON /LVCI  / LOCC(MCICSF),LVAC(MCICSF),
     &                MOCC(MCICSF),MVAC(MCICSF)
      COMMON /MULTCI/ MULTCI
      COMMON /WORK01/ OVOV(MCIINT),OOVV(MCIINT),CIVEC(MCICSF,MCICSF),
     &                DUMMY(1)
      COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),CIEIG(MCICSF),
```

```
     &                 SCRTCH(MCICSF,19)
      DIMENSION CIMAT(MCIINT)
      EQUIVALENCE(CIMAT(1),OVOV(1))
      DATA TWO/2.0D+00/
      DATA CHR7/'singlet','triplet'/
 2000 Format(1H ,'<<<<<< Enter SCI >>>>>>')
 2010 Format(1H ,'Singly excited CI calculation for ',A7,' state.'/
     &       1H ,'No. of spin adapted CSFs =',I5)
 2020 Format(1H ,21('=')/
     &       1H ,'CI Hamiltonian matrix'/
     &       1H ,21('='))
 2030 Format(1H ,37('=')/
     &       1H ,'Eigenvectors of CI Hamiltonian matrix'/
     &       1H ,37('='))
C     ***************************************************************
      MULTCI=IOPCI( 1)
      ICIMAT=IOPCI(12)
      ICIVEC=IOPCI(13)
C
      NELMS=NCSFS*(NCSFS+1)/2
C
      I1OR3=MULTCI/2+1
      WRITE(IOUT,2000)
      WRITE(IOUT,2010) CHR7(I1OR3),NCSFS
C
C     GET LIST VECTORS FOR CSFS.
      N=0
      DO 20 I=1,NOCS
        IMO=IOMO(I)
        DO 10 J=1,NVCS
          JMO=IVMO(J)
          N=N+1
          LOCC(N)=IMO
          LVAC(N)=JMO
          MOCC(N)=I
          MVAC(N)=J+NOCS
   10   CONTINUE
   20 CONTINUE
C
C     GET CI HAMILTONIAN MATRIX ELEMENTS FOR SINGLY EXCITED CSFS.
      IF(MULTCI.EQ.1) THEN
        DO 30 I=1,NELMS
          CIMAT(I)=TWO*OVOV(I)-OOVV(I)
   30   CONTINUE
      ELSE
        DO 40 I=1,NELMS
          CIMAT(I)=-OOVV(I)
   40   CONTINUE
      END IF
C
C     DIAGONAL CORRECTION.
      DO 50 I=1,NCSFS
        IMO=LOCC(I)
        JMO=LVAC(I)
        EIJ=EIG(JMO)-EIG(IMO)
        MA =I*(I+1)/2
        CIMAT(MA)=CIMAT(MA)+EIJ
   50 CONTINUE
C
      IF(ICIMAT.EQ.1) THEN
        WRITE(IOUT,2020)
        CALL CIMOUT(CIMAT,LOCC,LVAC,NCSFS)
      END IF
c========( Wed Oct 18 15:25:39 JST 1995 )======
        WRITE(IOUT,2020)
        CALL CIMOUT(CIMAT,LOCC,LVAC,NCSFS)
c========( Wed Oct 18 15:25:39 JST 1995 )======
C
C     THEN DIAGONALIZE.
      CALL DIAG1(CIMAT,CIVEC,CIEIG,NCSFS,MCICSF,SCRTCH,IOUT)
C
c========( tue sep 12 13:06:57 jst 1995 )======
      write( iout, 2030 )
      CALL CIVOUT(CIVEC,CIEIG,LOCC,LVAC,MCICSF,NCSFS)
c$$$      ii = 1
c$$$      if( ii .eq. 1 ) stop
c========( tue sep 12 13:06:57 jst 1995 )======
      IF(ICIVEC.EQ.1) THEN
        WRITE(IOUT,2030)
        CALL CIVOUT(CIVEC,CIEIG,LOCC,LVAC,MCICSF,NCSFS)
      END IF
C
C     ALL DONE. RETURN TO CALLER.
      CALL CLOCKM('SCI.   ')
      RETURN
      END
C
C@(#)=SCPOUT(SUB)
      SUBROUTINE SCPOUT
C
C     PRINTING OF SLATER-CONDON PARAMETERS.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*2 ELMNT,EL
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /PERTBL/ ELMNT(99)
      COMMON /PCONST/ PI,TOANG,AUTOEV,PLANCK,SLIGHT,EVTOER,PCON(14)
      COMMON /SLACON/ G1SP(30),F2PP(30),G2SD(11:30),G1PD(11:30),
     &                F2PD(11:30),G3PD(11:30),F2DD(11:30),F4DD(11:30),
     &                MAXSCP
      DIMENSION SCP(8)
      DATA ZERO/0.0D+00/,TENP5/1.0D+05/
      DATA IFLG/0/
 2000 Format(1H ,24('=')/
     &       1H ,'Slater-Condon parameters'/
     &       1H ,24('='))
 2010 Format(1H ,'Unit : cm-1.')
 2020 Format(1H ,79('-'))
 2030 Format(1H ,9X,'G1(S,P)  F2(P,P)  G2(S,D)  G1(P,D)  ',
     &           'F2(P,D)  G3(P,D)  F2(D,D)  F4(D,D)')
 2040 Format(1H ,I2,2X,A2,1X,8(1X,F8.1))
 2050 Format(1H ,'Unit : electron volt. 1 cm-1 =',D16.9,' eV in this ',
     &           'program.')
 2060 Format(1H ,I2,2X,A2,1X,8(1X,F8.5))
 2070 Format(1H ,'ROUTINE SCPOUT SKIPPED BECAUSE ALREADY PRINTED SCP.')
C     ***************************************************************
      IF(IFLG.EQ.1) THEN
        WRITE(IOUT,2070)
        RETURN
      END IF
C
      DO 10 I=3,8
        SCP(I)=ZERO
   10 CONTINUE
C
      TOKYS=(EVTOER*AUTOEV*TENP5)/(PLANCK*SLIGHT)
      CTOEV=AUTOEV/TOKYS
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010)
      WRITE(IOUT,2020)
      WRITE(IOUT,2030)
      WRITE(IOUT,2020)
      DO 20 I=3,10
        EL=ELMNT(I)
        SCP(1)=G1SP(I)*TOKYS
        SCP(2)=F2PP(I)*TOKYS
        WRITE(IOUT,2040) I,EL,(SCP(J),J=1,8)
   20 CONTINUE
      DO 30 I=11,MAXSCP
        EL=ELMNT(I)
        SCP(1)=G1SP(I)*TOKYS
        SCP(2)=F2PP(I)*TOKYS
        SCP(3)=G2SD(I)*TOKYS
        SCP(4)=G1PD(I)*TOKYS
        SCP(5)=F2PD(I)*TOKYS
        SCP(6)=G3PD(I)*TOKYS
        SCP(7)=F2DD(I)*TOKYS
        SCP(8)=F4DD(I)*TOKYS
        WRITE(IOUT,2040) I,EL,(SCP(J),J=1,8)
   30 CONTINUE
      WRITE(IOUT,2020)
C
      DO 40 I=3,8
        SCP(I)=ZERO
   40 CONTINUE
C
      WRITE(IOUT,2050) CTOEV
      WRITE(IOUT,2020)
      DO 50 I=3,10
        EL=ELMNT(I)
        SCP(1)=G1SP(I)*AUTOEV
        SCP(2)=F2PP(I)*AUTOEV
        WRITE(IOUT,2060) I,EL,(SCP(J),J=1,8)
   50 CONTINUE
      DO 60 I=11,MAXSCP
```

```
        EL=ELMNT(I)
        SCP(1)=G1SP(I)*AUTOEV
        SCP(2)=F2PP(I)*AUTOEV
        SCP(3)=G2SD(I)*AUTOEV
        SCP(4)=G1PD(I)*AUTOEV
        SCP(5)=F2PD(I)*AUTOEV
        SCP(6)=G3PD(I)*AUTOEV
        SCP(7)=F2DD(I)*AUTOEV
        SCP(8)=F4DD(I)*AUTOEV
        WRITE(IOUT,2060) I,EL,(SCP(J),J=1,8)
   60 CONTINUE
      WRITE(IOUT,2020)
C
      IFLG=1
C
      RETURN
      END
      subroutine sdci1
C*******************************************************************
C                                                                   *
C       @@@@   @@@@@    @@@@      @     @@                         *
C      @       @    @  @    @     @    @ @                         *
C       @@@@   @    @  @          @      @                          *
C           @  @    @  @          @      @                          *
C      @    @  @    @  @    @     @      @                          *
C       @@@@   @@@@@    @@@@      @    @@@@@                        *
C*******************************************************************
C
C  ci calculation for singlet states with single and double excitation
C        ncibas : number of total ci bases
C     ******************************************************************
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'
C
      common /io    / in,iout,ifile(10)
      common /iopci / iopci(18)
      common /lvci  / locc(mcicsf),lvac(mcicsf),
     &                mocc(mcicsf),mvac(mcicsf)
      common /multci/ multci
      common /work01/ ovov(mciint),oovv(mciint),civec(mcicsf,mcicsf),
     &                dummy(1)

c$ohtawara for test
c       common /my_work1/ number
c       dimension zindo(15,15)


      data two/2.0d+00/
      character*7 chr7(2)
      data chr7/'singlet','triplet'/
 2000 format(1h ,'<<<<<< enter sdci1 >>>>>>')
 2010 format(1h ,'sdci calculation for ',a7,' state.'/
     &       1h ,'no. of spin adapted sdci bases =',i5)
c 2020 format(1h ,21('=')/
c     &       1h ,'ci hamiltonian matrix'/
c     &       1h ,21('='))
c 2030 format(1h ,37('=')/
c     &       1h ,'eigenvectors of ci hamiltonian matrix'/
c     &       1h ,37('='))
c     ******************************************************************
c
      write(iout,2000)
c
c
      multci = iopci( 1  )
      icimat = iopci( 12 )
      icivec = iopci( 13 )
      ilor3  = multci/2 + 1
      write(iout,2010) chr7(ilor3), ncibas

c cut for save memory ***************************************************
c$ohtawara test output integral(ij|kl)
      if (idebug.eq.1) call intout
c cut for save memory ***************************************************

      cimat( 1 ) = 0.0d0
      ntau = 1
```

```
      do 1 nomg = momega(1)+1, momega(2)
        ncimat = nomg*( nomg - 1 )/2 + ntau
        cimat( ncimat ) = 0.0d0
    1 continue
      call cimtss

c      write(iout,'('' *** s ***'')')
c      call fckout( cimat, ncibas, 1.0d0 )

c========( Wed Sep 27 09:21:49 JST 1995 )======
c$$$      ii = 1
c$$$      if( ii .eq. 1 ) go to 9000
c========( Wed Sep 27 09:21:49 JST 1995 )======
      call cimt10
      call cimt1s
      call cimt11

c      write(iout,'('' *** 1 ***'')')
c      call fckout( cimat, ncibas, 1.0d0 )

c
      if( isdci0 .eq. 1 ) go to 9000
c
      if( isdci0 .eq. 2 ) then
        momgs = momega( 3 ) + 1
        momge = momega( 5 )
        call cimt30( momgs,  momge )
        mtaus = momega( 1 ) + 1
        mtaue = momega( 2 )
        call cimt3s( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 2 ) + 1
        mtaue = momega( 3 )
        call cimt31( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 3 ) + 1
        call cimt33( momgs,  momge,  mtaus )
      elseif( isdci0 .eq. 3 ) then

        call cimt20
        call cimt2s
        call cimt21
        call cimt22

c      write(iout,'('' *** 2 ***'')')
c      call fckout( cimat, ncibas, 1.0d0 )

        momgs = momega( 4 ) + 1
        momge = momega( 5 )
        call cimt30( momgs,  momge )
        mtaus = momega( 1 ) + 1
        mtaue = momega( 2 )
        call cimt3s( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 2 ) + 1
        mtaue = momega( 3 )
        call cimt31( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 3 ) + 1
        mtaue = momega( 4 )
        call cimt32( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 4 ) + 1
        call cimt33( momgs,  momge,  mtaus  )

c      write(iout,'('' *** 3 ***'')')
c      call fckout( cimat, ncibas, 1.0d0 )

        call cimt40
        call cimt4u
        call cimt41
        call cimt42
        call cimt43
        call cimt44

c      write(iout,'('' *** 4 ***'')')
c      call fckout( cimat, ncibas, 1.0d0 )

c========( Wed Sep 27 09:21:49 JST 1995 )======
c$$$      ii = 1
c$$$      if( ii .eq. 1 ) go to 9000
c========( Wed Sep 27 09:21:49 JST 1995 )======
        call cimt50
        call cimt5s
        call cimt51
        call cimt52
        call cimt53
        call cimt54
```

```
        call cimt55

c       write(iout,'(('' *** 5 ***''))')
c       call fckout( cimat, ncibas, 1.0d0 )


      else
        write(iout,*)' **** bad choice of ci bases'
        stop
      endif
c
 9000 continue

c$ ------ TEST TEST TEST ----- Wed Jan 24 17:35:42 JST 1996 *****************
c$ ------ TEST    READ ZINDO RESULTS
c$ohtawara test read ci matrix of zindo results
c       cut
c$ ------ TEST    READ ZINDO RESULTS
c$ ------ TEST TEST TEST ----- Wed Jan 24 17:35:42 JST 1996 *****************


c$ohtawara
c for small print out
      if (iprint_yes .lt. 0)  go to 999

      write(iout,'(('' *** CI Hamiltonian matrix elements ***''))')
      call fckout( cimat, ncibas, 1.0d0 )

 999  continue

c----     then diagonalize.
      call diagl( cimat,  civec,  cieig,  ncibas, mcicsf, scrtch, iout )

      call cidump

c     all done. return to caller.
      call clockm('sdcil.   ')


      return
      end
c
c0(#)=SETCOM(SUB)
      SUBROUTINE SETCOM
c
c     PRESET SOME LABELED COMMON.
c     NOTE.
c      THIS PROGRAM DOES NOT USE BLOCK DATA STATEMENT AT ALL.
c     Oct/92 ... A.M.
c     93/Jan ... A.M.
c     93/Feb ... A.M.
c     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*4 ORB(9),ORBTYP
      CHARACTER*3 SHG(18),EO(18),SHGSUF,EOSUF
      CHARACTER*2 EL(99),ELMNT,El2(99),Elmnt2
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /ORBTYP/ ORBTYP(9)
      COMMON /PERTBL/ ELMNT(99)
      COMMON /PTBL2 / ELMNT2(99)
      Common /Mass  / AtMass(99)
      COMMON /BETASF/ SHGSUF(18),EOSUF(18)
      COMMON /PCONST/ PCON(20)
      Dimension AW(99)
      DATA ORB /'   s','  px','  py','  pz',' dz2',' dxy',' dyz',
     &          'dx-y',' dxy'/
      DATA EL
     &    /'H ','HE',
     &     'LI','BE','B ','C ','N ','O ','F ','NE',
     &     'NA','MG','AL','SI','P ','S ','CL','AR',
     &     'K ','CA','SC','TI','V ','CR','MN','FE','CO','NI','CU',
     &     'ZN','GA','GE','AS','SE','BR','KR',
     &     'RB','SR','Y ','ZR','NB','MO','TC','RU','RH','PD','AG',
     &     'CD','IN','SN','SB','TE','I ','XE',
     &     'CS','BA','LA','CE','PR','ND','PM','SM','EU','GD','TB','DY',
     &     'HO','ER','TM','YB','LU','HF','TA','W ','RE','OS','IR','PT',
     &     'AU','HG','TL','PB','BI','PO','AT','RN',
     &     'FR','RA','AC','TH','PA','U ','NP','PU','AM','CM','BK','CF',
     &     'XX'/
      DATA EL2
     &    /'H ','He',
     &     'Li','Be','B ','C ','N ','O ','F ','Ne',
     &     'Na','Mg','Al','Si','P ','S ','Cl','Ar',
     &     'K ','Ca','Sc','Ti','V ','Cr','Mn','Fe','Co','Ni','Cu',
     &     'Zn','Ga','Ge','As','Se','Br','Kr',
     &     'Rb','Sr','Y ','Zr','Nb','Mo','Tc','Ru','Rh','Pd','Ag',
     &     'Cd','In','Sn','Sb','Te','I ','Xe',
     &     'Cs','Ba','La','Ce','Pr','Nd','Pm','Sm','Eu','Gd','Tb','Dy',
     &     'Ho','Er','Tm','Yb','Lu','Hf','Ta','W ','Re','Os','Ir','Pt',
     &     'Au','Hg','Tl','Pb','Bi','Po','At','Rn',
     &     'Fr','Ra','Ac','Th','Pa','U ','Np','Pu','Am','Cm','Bk','Cf',
     &     'XX'/
      Data (AW(i),i=1,40)/
     &      1.00794  d0,   4.002602 d0,   6.941    d0,   9.012182 d0,
     &      10.811   d0,  12.011    d0,  14.00674  d0,  15.9994   d0,
     &      18.9984032d0, 20.1797   d0,  22.989768 d0,  24.3050   d0,
     &      26.981539 d0, 28.0855   d0,  30.973762 d0,  32.066    d0,
     &      35.4527   d0, 39.948    d0,  39.0983   d0,  40.078    d0,
     &      44.955910 d0, 47.88     d0,  50.9415   d0,  51.9961   d0,
     &      54.93805  d0, 55.847    d0,  58.93320  d0,  58.6934   d0,
     &      63.546    d0, 65.39     d0,  69.723    d0,  72.61     d0,
     &      74.92159  d0, 78.96     d0,  79.904    d0,  83.80     d0,
     &      85.4678   d0, 87.62     d0,  88.90585  d0,  91.224    d0/
      Data (AW(i),i=41,99)/
     &      92.90638  d0,  95.94    d0,  -1.0      d0, 101.07     d0,
     &     102.90550  d0, 106.42    d0, 107.8682   d0, 112.411    d0,
     &     114.82     d0, 118.710   d0, 121.757    d0, 127.60     d0,
     &     126.90447  d0, 131.29    d0, 132.90543  d0, 137.327    d0,
     &     138.9055   d0, 140.115   d0, 140.90765  d0, 144.24     d0,
     &      -1.0      d0, 150.36    d0, 151.965    d0, 157.25     d0,
     &     158.92534  d0, 162.50    d0, 164.93032  d0, 167.26     d0,
     &     168.93421  d0, 173.04    d0, 174.967    d0, 178.49     d0,
     &     180.9479   d0, 183.85    d0, 186.207    d0, 190.2      d0,
     &     192.22     d0, 195.08    d0, 196.96654  d0, 200.59     d0,
     &     204.3833   d0, 207.2     d0, 208.98037  d0,
     &                          6*-1.0d0,
     &                     232.0381   d0, 231.03588  d0, 238.0289   d0,
     &                          7*-1.0d0 /
      DATA SHG /'XXX','XXY','XXZ','XYY','XYZ','XZZ',
     &          'YXX','YXY','YXZ','YYY','YYZ','YZZ',
     &          'ZXX','ZXY','ZXZ','ZYY','ZYZ','ZZZ'/
      DATA EO  /'XXX','XXY','XXZ','XYX','XYY','XYZ','XZX','XZY','XZZ',
     &          'YXY','YXZ','YYY','YYZ','YZY','YZZ','ZXZ','ZYZ','ZZZ'/
c
c     THE PI
      DATA PI     /3.1415926535897932D+00/
c
c     FROM BOHR TO ANGSTROM
      DATA TOANG  /0.5291670000000000D+00/
c
c     FROM HARTREE TO EV.
      DATA AUTOEV /2.7210000000000000D+01/
c
c     PLANCK'S CONSTANT
      DATA PLANCK /6.6261760000000000D+00/
c
c     SPEED OF LIGHT.
      DATA SLIGHT /2.9979245800000000D+00/
c
c     FROM EV TO ERG.
      DATA EVTOER /1.6021892000000000D+00/
c
c     FROM EV TO KCAL/MOL
      DATA EVTOKC /2.3061000000000000D+01/
c     ****************************************************************
c
c     Common /IO    / ... Fortran logical unit.
c
c      In ... Standard input.
c      Iout ... Standard output.
c      IFile(1) through IFile(9) ... Not used at present.
c      IFile(10) ... Output for post-process graphic package.
c
      IN=5
c$ohtawara 1996.2.17
c      IOUT=6
      iout = 16

c========={ Thu Sep 14 09:19:58 JST 1995 }======
      call openf( in )
c========={ Thu Sep 14 09:19:58 JST 1995 }======
      DO 10 I=1,10
        IFILE(I)=I+10
   10 CONTINUE
c
      PCON(1)=PI
```

```
      PCON(2)=TOANG
      PCON(3)=AUTOEV
      PCON(4)=PLANCK
      PCON(5)=SLIGHT
      PCON(6)=EVTOER
      PCON(7)=EVTOKC
C
      DO 20 I=1,9
        ORBTYP(I)=ORB(I)
   20 CONTINUE
C
      DO 30 I=1,99
        ELMNT(I)=EL(I)
        ELMNT2(I)=EL2(I)
   30 CONTINUE
C
      DO 40 I=1,18
        SHGSUF(I)=SHG(I)
        EOSUF(I) =EO(I)
   40 CONTINUE
C
C     Atomic weight
      Do 50 i=1,99
        AtMass(i)=AW(i)
   50 Continue
      RETURN
      END
C
C@(#)=SETFLG(SUB)
      SUBROUTINE SETFLG(CHR1,LENCHR,IFLG)
C
C     SETTING OF IFLG.
C     NOTE.
C      IFLG=0 ... NO OR DO NOT.
C          =1 ... YES OR DO.
C     ****************************************************************
      CHARACTER*1 CHR1(*)
C     ****************************************************************
      CALL IPOS(CHR1,LENCHR,IFLG)
C
      IF(IFLG.NE.0) THEN
        IFLG=1
      END IF
C
      RETURN
      END
C
C@(#)=SETTM(SUB)
      SUBROUTINE SETTM
C
C     MOVE DATA ON LABELED COMMON /WORK01/ AND /WORK02/.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      PARAMETER(MAXDIM=MCICSF+MCICS2+MCIPCK*3)
      Parameter(MDim2 = MCIMOs * MaxBas)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                 NOCS,NVCS,NCSFS
      COMMON /WORK01/ CIVEC(MCICS2),TDX(MCIPCK),TDY(MCIPCK),TDZ(MCIPCK),
     &                 DUMMY1(1)
      COMMON /WORK02/ C(MAXDIM),DUMMY2(1)
      Common /AOChrg/ GChrg(MaxBas), AtmChg(MaxAt), CMO(MDim2)
C     ****************************************************************
C
C     NOW, COMMON /WORK01/ CIVEC(MCICSF**2),
C                          TDX(MCIPCK),TDY(MCIPCK),TDZ(MCIPCK)
C          COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),CIEIG(MCICSF)
c          Common /AOChrg/ GChrg(MaxBas), AtmChg(MaxAt)
c
c     First move C to CMO.
      n = 0
      DO 20 i = 1, NOCS + NVCS
        Ind = (LMO(i) - 1)*NBasis
        Do 10 j = 1, NBasis
          n = n + 1
          CMO(n) = C(Ind + j)
   10   Continue
   20 Continue
C
C     MOVE CIEIG.
      IND=MAXB2+MAXBAS
      DO 30 I=1,NCSFS
        C(I)=C(I+IND)
   30 CONTINUE
C
C     MOVE CIVEC.
      IND=MCICSF
      NT =NCSFS*NCSFS
      DO 40 I=1,NT
        C(I+IND)=CIVEC(I)
   40 CONTINUE
C
C     MOVE TDX, TDY, AND TDZ.
      IND1=IND +MCICS2
      IND2=IND1+MCIPCK
      IND3=IND2+MCIPCK
      NMOS=NOCS+NVCS
      NT  =NMOS*(NMOS+1)/2
      DO 50 I=1,NT
        C(I+IND1)=TDX(I)
        C(I+IND2)=TDY(I)
        C(I+IND3)=TDZ(I)
   50 CONTINUE
C
C     NOW, COMMON /WORK01/ DUMMY
C          COMMON /WORK02/ CIEIG(MCICSF),CIVEC(MCICSF**2),
C                          TDX(MCIPCK),TDY(MCIPCK),TDZ(MCIPCK)
c          Common /AOChrg/ GChrg(MaxBas), AtmChg(MaxAt),
c                          CMO(MCIMOs*MaxBas)
C
      RETURN
      END
C
C@(#)=SGINFO(SUB)
      SUBROUTINE SGINFO(IFDON2,IFDON3)
C
C     GET INFORMATIONS FOR ATOMIC INTEGRAL EVALUATION.
c     Oct/92 ... A.M.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      Logical Anchor
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /BASINF/ NLBAS(MAXAT),NUBAS(MAXAT),NIBAS(MAXAT),
     &                 IBTOA(MAXBAS)
      COMMON /QNOINF/ NC(MAXIAN,2),LC(9),MC(9)
      COMMON /FACTOR/ FS,FP,FD
      COMMON /VELEC / ZV(MAXAT)
      Common /Expnt / Zeta(MaxAt),Dmy(MaxBas)
      Common /Anchor/ Anchor
      DIMENSION I4OR9(0:1)
      DATA I4OR9/4,9/
 2000 Format(1H ,'D orbitals are used on the second row atoms.')
 2010 Format(1H ,'D orbitals are used on the third row atoms.')
 2020 Format(1H ,I5,' Basis functions.',I10,' Valence electrons.'/
     &       1H ,I5,' Alpha electrons.',I10,' Beta electrons.')
 2030 Format(1H ,'Scaling factors of overlap integrals ...'/
     &       1H ,'p-sigma =',F8.5,'  p-pi =',F8.5,'  d = ',F8.5)
 9910 Format(1H ,50('X')/
     &       1H ,'PROGRAM STOPS EXECUTING BECAUSE ATOMIC NO .GT. 35'/
     &       1H ,50('X'))
 9920 Format(1H ,58('X')/
     &       1H ,'PROGRAM STOPS EXECUTING BECAUSE NO OF BASIS SET .GT.',
     &           I5,'.'/
     &       1H ,58('X'))
 9930 Format(1H ,57('X')/
     &       1H ,'PROGRAM STOPS EXECUTING BECAUSE OF ODD VALENCE ',
     &           'ELECTRONS.'/
     &       1H ,57('X'))
C     ****************************************************************
      MDIM=MAXIAN
      DO 10 I=1,MDIM
        NC(I,1)=0
        NC(I,2)=0
   10 CONTINUE
C
C     FILL COMMON /BASINF/ AND SOME VARIABLES OF /MOL   /.
      NBAS=0
      NVE =0
      DO 30 I=1,NATOMS
```

```
      NLBAS(I)=NBAS+1
C
      IA=IAN(I)
      IF      (IA.LE. 2) THEN
        NBAS=NBAS+1
        IVE =IA
      ELSE IF (IA.LE.10) THEN
        NBAS=NBAS+4
        IVE =IA-2
      ELSE IF (IA.LE.18) THEN
        NBAS=NBAS+I4OR9(IFDON2)
        IVE =IA-10
      ELSE IF (IA.LE.29) THEN
        NBAS=NBAS+9
        IVE =IA-18
      ELSE IF (IA.LE.35) THEN
        NBAS=NBAS+I4OR9(IFDON3)
        IVE =IA-28
      ELSE
        WRITE(IOUT,9910)
        CALL MERROR
      END IF
C
      NIBAS(I)=NBAS-NLBAS(I)+1
      DO 20 J=NLBAS(I),NBAS
        IBTOA(J)=I
   20 CONTINUE
      NUBAS(I)=NBAS
C
      NVE   =NVE+IVE
      ZV(I) =DFLOAT(IVE)
   30 CONTINUE
C
     NBASIS=NBAS
C
     IF(NBASIS.GT.MAXBAS) THEN
       WRITE(IOUT,9920) MAXBAS
       CALL MERROR
     END IF
C
     NE    =NVE-ICHARG
     NBE   =NE/2
     NAE   =NE-NBE
C
     IF(NAE.NE.NBE) THEN
       WRITE(IOUT,9930)
       CALL MERROR
     END IF
C
C    FILL COMMON /QNOINF/.
C    NC(IAN,1) ... PRINCIPLE QUANTUM NUMBER FOR SP SHELL.
C    NC(IAN,2) ... PRINCIPLE QUANTUM NUMBER FOR D SHELL.
     NC(1,1)=1
     NC(2,1)=1
     DO 40 I=3,10
       NC(I,1)=2
   40 CONTINUE
     DO 50 I=11,18
       NC(I,1)=3
       NC(I,2)=3
   50 CONTINUE
     DO 60 I=19,29
       NC(I,1)=4
       NC(I,2)=3
   60 CONTINUE
     DO 70 I=30,35
       NC(I,1)=4
       NC(I,2)=4
   70 CONTINUE
C
C    AZIMUTHAL QUANTUM NUMBER.
     LC(1)= 0
     LC(2)= 1
     LC(3)= 1
     LC(4)= 1
     LC(5)= 2
     LC(6)= 2
     LC(7)= 2
     LC(8)= 2
     LC(9)= 2
C
C    MAGNETIC QUANTUM NUMBER.
     MC(1)= 0
     MC(2)= 1
     MC(3)=-1
     MC(4)= 0
     MC(5)= 0
     MC(6)= 1
     MC(7)=-1
     MC(8)= 2
     MC(9)=-2
C
C    GET FACTORIAL.
     CALL GETFCT
C
     IF(IFDON2.EQ.1) THEN
       WRITE(IOUT,2000)
     END IF
     IF(IFDON3.EQ.1) THEN
       WRITE(IOUT,2010)
     END IF
C
     WRITE(IOUT,2020) NBASIS,NE,NAE,NBE
     WRITE(IOUT,2030) FS,FP,FD
C
     If (Anchor) Call AncBas(Zeta,Dmy,NAtoms,NBasis,NLBas,NUBas,
    &                        IFile(10))
c
     RETURN
     END
C
C@(#)=SGINT(SUB)
     SUBROUTINE SGINT
C
C    EVALUATION OF OVERLAP AND ELECTRON REPULSION INTEGRALS (ERIS).
C    THIS ROUTINE IS COMPOSED OF 3 SUBPROGRAMS LISTED BELOW.
C     1. SGINFO ... GET INFORMATIONS FOR INTEGRAL EVALUATION.
C     2. OVLP   ... CALCULATION OF SCALED OVERLAP INTEGRALS.
C                   SCALING FACTOR (SEE ROUTINE RDOPT).
C                   CNDO/S : P-PI = 0.585  OTHERS = 1.00
C                   INDO/S : P-PI = 0.585 (SINGLET), 0.680 (TRIPLET)
C                            P-SIGMA = 1.267  OTHERS = 1.00
C                   OTHERS : NOT SCALED (ALL THE FACTORS ARE UNITY).
C     3. ERI    ... EVALUATION OF 2 CENTER ERIS.
C                   7 DIFFERENT FORMULA FOR EVALUATING 2 CENTER ERIS
C                   ARE AVAILABLE IN ROUTINE 'ERI'.
C                     ROUTINE
C                   a. ANL2E  : ANALYTIC CALCULATION OVER S FUNCTIONS.
C                   b. PP2E   : PARISER-PARR FORMULA.
C                   c. NM2E   : NISHIMOTO-MATAGA(-WEISS) FORMULA.
C                   d. OHNO2E : OHNO FORMULA.
C                   e. OK2E   : OHNO-KLOPMAN FORMULA.
C                   f. DH2E   : DASGUPTA-HUZINAGA FORMULA.
C
C    REFERENCES.
C     C.J.J.ROOTHAAN, J.CHEM.PHYS.,19,1445(1951)
C     R.PARISER AND R.G.PARR, J.CHEM.PHYS.,21,767(1953)
C     K.NISHIMOTO AND N.MATAGA, Z.PHYS.CHEM.,13,140(1957)
C     K.Nishimoto, Proceedings of Domestic Molecular Structure
c                  Conference in Fukuoka, 2A07(1990) (in Japanese)
C     K.OHNO, THEORET.CHIM.ACTA(BERL.), 2,219(1964)
C     G.KLOPMAN, J.AMER.CHEM.SOC.,87,3300(1965)
C     A.DASGUPTA AND S.HUZINAGA, THEORET.CHIM.ACTA,35,329(1974)
C    ***************************************************************
     COMMON /IO    / IN,IOUT,IFILE(10)
     COMMON /IOPINT/ IOPINT(18)
 2000 Format(1H ,'<<<<<< Enter SGInt >>>>>>')
C    ***************************************************************
     IGAMMA=IOPINT( 2)
     IFDON2=IOPINT( 3)
     IFDON3=IOPINT( 4)
     IOVOUT=IOPINT(11)
     IGOUT =IOPINT(12)
C
     WRITE(IOUT,2000)
C
     CALL SGINFO(IFDON2,IFDON3)
     CALL OVLP  (IOVOUT)
     CALL ERI   (IGAMMA,IGOUT)
C
C    ALL DONE. RETURN TO CALLER.
     CALL CLOCKM('SGInt. ')
C
     RETURN
     END
C
```

```
C@(#)=SHG(SUB)
      SUBROUTINE SHG(W,BETASH,BVEC,TGN,TNN,CIEIG,NCSFS,MDIM1,MDIM2)
C
C     CALCULATE 1-ST HYPERPOLARIZABILITIES FOR SECOND HARMONIC
C     GENERATION (SHG) ... BETA(-2W;W,W).
C     UNIT : ATOMIC UNIT.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION BETASH(*),BVEC(*),TGN(MDIM1,*),TNN(MDIM2,*),CIEIG(*)
      DATA TWO/2.0D+00/,THREE/3.0D+00/
C     ***************************************************************
      N=0
      DO 30 I=1,3
        DO 20 J=1,3
          DO 10 K=J,3
            N=N+1
            BETASH(N)=BETXYZ(W,W,I,J,K,TGN,TNN,CIEIG,NCSFS,MDIM1,MDIM2)
   10     CONTINUE
   20   CONTINUE
   30 CONTINUE
C
C     THEN GET BETA-VEC.
C     DO 50 I=1,3
C       BVEC(I)=ZERO
C       NNI=(I-1)*6
C       DO 40 J=1,3
C         NNJ=(J-1)*6
C         II =MIN0(I,J)
C         JJ =MAX0(I,J)
C         MA1=NNI+(J-1) *(8-J) /2+1
C         MA2=NNJ+(II-1)*(8-II)/2+JJ-II+1
C         BVEC(I)=BVEC(I)+BETASH(MA1)+TWO*BETASH(MA2)
C  40   CONTINUE
C       BVEC(I)=BVEC(I)/THREE
C  50 CONTINUE
C
      BVEC(1)=(  BETASH( 1)*THREE
     &         + BETASH( 8)*TWO+BETASH( 4)
     &         + BETASH(15)*TWO+BETASH( 6) ) / THREE
      BVEC(2)=(  BETASH( 2)*TWO+BETASH( 7)
     &         + BETASH(10)*THREE
     &         + BETASH(17)*TWO+BETASH(12) ) / THREE
      BVEC(3)=(  BETASH( 3)*TWO+BETASH(13)
     &         + BETASH(11)*TWO+BETASH(16)
     &         + BETASH(18)*THREE          ) / THREE
C
      RETURN
      END
C
C@(#)=SPMIX(FUN)
      DOUBLE PRECISION FUNCTION SPMIX(NC,ZS,ZP)
C
C     GET 1 CENTER S-P MIXING TERM, <S/R/P>.
C     PDMIX IS INDEPENDENT OF MAGNETIC QUANTUM NOS. OF BASIS FUNCTIONS.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /FCTRL / FACT(0:50)
      DATA THREE/3.0D+00/
C     ***************************************************************
      ROOT3=DSQRT(THREE)
      NC2  =NC+NC
      ZZ   =ZS*ZP
      TRM1 =DFLOAT((NC2+1) *(2**(NC2+1))) *(ZZ**NC) *DSQRT(ZZ)
      TRM2 =ROOT3 *((ZS+ZP)**(NC2+2))
C
      SPMIX=TRM1/TRM2
C
      RETURN
      END
C
C@(#)=SS(FUN) ... CNDO2/3R(SS)
      FUNCTION SS(NN1,LL1,MM,NN2,LL2,ALPHA,BETA)
C
C     ============================================================
C     CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R CNDO2/3R
C     ============================================================
C     DOUBLE PRECISION FUNCTION SS ... FROM CNDO 2/3R
C     PROCEDURE FOR CALCULATING REDUCED OVERLAP INTEGRALS
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER ULIM,UL2
      COMMON /AUXINT/ A(17),B(17)
      COMMON /FCTRL / FACT(0:50)
C     ***************************************************************
      N1=NN1
      L1=LL1
      M=MM
      N2=NN2
      L2=LL2
      P =(ALPHA + BETA)/2.D0
      PT=(ALPHA - BETA)/2.D0
      X = 0.D0
      M=IABS(M)
C     REVERSE QUANTUM NUMBERS IF NECESSARY
      IF((L2.LT.L1).OR.((L2.EQ.L1).AND.(N2.LT.N1))) GO TO 20
   10 GO TO 30
   20 K = N1
      N1= N2
      N2= K
      K= L1
      L1= L2
      L2= K
      PT=-PT
   30 CONTINUE
      K = MOD((N1+N2-L1-L2),2)
C     FIND A AND B INTEGRALS
      CALL AINTGS(P,N1+N2)
      CALL BINTGS(PT,N1+N2)
C     BEGIN SECTION USED FOR OVERLAP INTEGRALS INVOLVING S FUNCTIONS
      LLIM=1
      ULIM=N1+N2+1
      IF((L1.GT.0).OR.(L2.GT.0)) GO TO 60
      DO 50 I1=LLIM,ULIM
      NNI1=N1+N2-I1+2
   50 X=X+COEFFS(N1,N2,I1-1)*A(I1)*B(NNI1)*0.5D0
      SS=X
      GO TO 80
C     BEGIN SECTION USED FOR OVERLAPS INVOLVING NON-S FUNCTIONS
   60 DO 70 I1=LLIM,ULIM
      UL2 = ULIM/2 + MOD(ULIM,2)
      DO 70 J1=LLIM,UL2
      IIII= 2*J1 + MOD(K+I1+1,2)-1
   70 X = X + COEFF(N1,N2,L1,L2,M,I1,IIII)*A(I1)*B(IIII)
      SS = X*(FACT(M+1)/8.D0)**2*DSQRT(DFLOAT(2*L1+1)*FACT(L1-M)*
     1 DFLOAT(2*L2+1)*FACT(L2-M)/(4.D0*FACT(L1+M)*FACT(L2+M)))
   80 CONTINUE
      RETURN
      END
C
C@(#)=SSMOUT(SUB)
      SUBROUTINE SSMOUT(A,IAN,IBTOA,NLBAS,NBASIS)
C
C     OUTPUT MODULE FOR STO BASED SYMMETRIC MATRIX.
C     PRINT OUT LOWER TRIANGLE MATRIX.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*4 ORBTYP,ORB
      CHARACTER*2 ELMNT,EL
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /PRWDTH/ IWDTH
      COMMON /PTb12 / ELMNT(99)
      COMMON /ORBTYP/ ORBTYP(9)
      DIMENSION A(*),IAN(*),IBTOA(*),NLBAS(*)
 2000 Format(1H ,17X,10(7X,I4))
 2010 Format(1H ,I4,1X,I3,2X,A2,2X,A4,2X,10F11.6)
 2020 Format(1H ,I4,10X,          A4,2X,10F11.6)
C     ***************************************************************
      IF(IWDTH.EQ.1) THEN
        IW=5
      ELSE
        IW=10
      END IF
C
      ILOWER=1
      IUPPER=IW
C
   10 IF(IUPPER.GT.NBASIS) THEN
        IUPPER=NBASIS
      END IF
C
      WRITE(IOUT,2000) (I,I=ILOWER,IUPPER)
C
      IATOLD=0
C
      DO 20 I=ILOWER,NBASIS
        NCLMS=I-ILOWER+1
```

```
      IF(NCLMS.GT.IW) THEN
        IU=ILOWER+IW-1
      ELSE
        IU=I
      END IF
C
      MA=I*(I-1)/2
C
      IAT =IBTOA(I)
      IF(IAT.NE.IATOLD) THEN
        IATOLD=IAT
        NL     =NLBAS(IAT)
        IORB   =I-NL+1
        EL     =ELMNT(IAN(IAT))
        ORB    =ORBTYP(IORB)
        WRITE(IOUT,2010) I,IAT,EL,ORB,(A(MA+J),J=ILOWER,IU)
      ELSE
        IORB=IORB+1
        ORB =ORBTYP(IORB)
        WRITE(IOUT,2020) I,ORB,(A(MA+J),J=ILOWER,IU)
      END IF
C
   20 CONTINUE
C
      IF(IUPPER.EQ.NBASIS) RETURN
C
      ILOWER=ILOWER+IW
      IUPPER=IUPPER+IW
C
      GO TO 10
C
      END
c
c@(#)=StrLen(Fun)
      Integer Function StrLen (CStr)
c
c     Get length of C string.
c
c     On input:
c      CStr   ... A C string.
c     On output:
c      StrLen   ... Length of CStr.
c
c     ****************************************************************
      Implicit Integer (A-Z)
      Character*(*) CStr
c     ****************************************************************
c
      StrLen = 0
c
c     First, get length of FORTRAN string.
      LenStr = Len (CStr)
c
      if (LenStr .eq. 0) Return
c
c     Then search the NULL character. Not count the NULL character.
      Do 10 i = 1, LenStr
        if (CStr(i:i) .eq. Char(0)) Then
          StrLen = i - 1
          Return
        EndIf
   10 Continue
c
c     Not a C string.
      Stop 'Abort in StrLen. Not a C string.'
c
      End
C
C@(#)=SYMM(SUB) ... DUMMY ROUTINE NOW
      SUBROUTINE SYMM
C
C     THIS IS A DUMMY ROUTINE IN CURRENT VERSION OF MOS-F.
C     ****************************************************************
      COMMON /IO    / IN,IOUT,IFILE(10)
 8000 Format(1H ,'ROUTINE SYMM SKIPPED IN CURRENT VERSION OF MOS-F.')
C     ****************************************************************
      WRITE(IOUT,8000)
      RETURN
      END
c$$$#endif
c$$$#ifdef FUJITSU
C
c$$$C@(#)=PRTITL(SUB) ... Fujitsu FORTRAN
c$$$      SUBROUTINE PRTITL
c$$$C
c$$$C     USE Fujitsu SYSTEM ROUTINES.
c$$$C      1 ... SUBROUTINE DATE(CHR8) CHR8:YY-MM-DD
c$$$C      2 ... SUBROUTINE TIME(INT4) INT4:TIME(MSEC)
c$$$C     93/Mar A.M.
c$$$C     93/Apr A.M.
c$$$C     ****************************************************************
c$$$      Implicit Integer (a-z)
c$$$      Character*256 Card
c$$$      Character*80 SysNam, NodNam, RelsNo, MacNam
c$$$      CHARACTER*8 TODAY
c$$$      Character*1 Aster
c$$$      Save Aster
c$$$      Data Aster /1h*/
c$$$      Save MaxClm, MaxMA
c$$$      Data MaxClm /79/, MaxMA /58/
c$$$      COMMON /IO    / IN,IOUT,IFILE(10)
c$$$      COMMON /ELAPSE/ ISTART,IARRAY(3)
c$$$ 2000 FORMAT(1H1/
c$$$     &       1H ,'Just entered MOS-F system at ',I2.2,':',I2.2,':',I2.2,
c$$$     &            ' on ',A8,' ...')
c$$$ 2010 Format(1h ,79A1)
c$$$ 2020 Format(1H ,A/
c$$$     &       1h ,22x,'By A. Matsuura. Fujitsu Labs Ltd.')
c$$$C     ****************************************************************
c$$$      CALL DATE(TODAY)
c$$$      CALL TIME(Now)
c$$$C
c$$$      Now=Now/1000
c$$$      CALL TOHMS(Now,IHR,IMIN,ISEC)
c$$$C
c$$$      IStart=TimeF()
c$$$c
c$$$c     Cuts ... C-language function.
c$$$      Call Cuts (SysNam, NodNam, RelsNo, MacNam)
c$$$c
c$$$      Card = '   MOS-F (V01L05) ... '//
c$$$     &       SysNam(1:StrLen(SysNam))//' Release '//
c$$$     &       RelsNo(1:StrLen(RelsNo))//' '//
c$$$     &       MacNam(1:StrLen(MacNam))//'/'//
c$$$     &       NodNam(1:StrLen(NodNam))//'.'//Char(0)
c$$$c
c$$$      LenCrd = StrLen(Card)
c$$$      NAster = max0 (MaxMA, (min0 (MaxClm, (LenCrd + 3))) )
c$$$c
c$$$      WRITE(IOUT,2000) IHR,IMIN,ISEC,TODAY
c$$$      Write(IOut,2010) (Aster, ijkl = 1, NAster)
c$$$      Write(IOut,2020) Card(1:LenCrd)
c$$$      Write(IOut,2010) (Aster, ijkl = 1, NAster)
c$$$C
c$$$      RETURN
c$$$      END
C
c$$$C@(#)=CLOCKM(SUB) ... Fujitsu FORTRAN
c$$$      SUBROUTINE CLOCKM(RNAME)
c$$$C
c$$$C     TIMING ROUTINE FOR MOS-F.
c$$$C     NOTE.
c$$$C      USE Fujitsu SYSTEM ROUTINE.
c$$$C      SUBROUTINE CLOCK(C,I,J)
c$$$C      C ... CPU-TIME.
c$$$C      I ... 0 UNIT = SEC.
c$$$C            1 UNIT = MSEC.
c$$$C      J ... 1 C = REAL*4
c$$$C            2 C = REAL*8
c$$$c      Subroutine ClockV(V,C,i,j)
c$$$c      V ... VU-time.
c$$$c      C ... CPU-time.
c$$$c      i ... 0 Unit = sec.
c$$$c            1 Unit = msec.
c$$$c      j ... 1 V,C = Real*4
c$$$c            2 V,C = Real*8
c$$$C     ****************************************************************
c$$$      CHARACTER*7 RNAME
c$$$      COMMON /IO    / IN,IOUT,IFILE(10)
c$$$      Data VU0/0.0E+00/
c$$$      DATA TIME0/0.0E+00/
c$$$c2000 FORMAT(1H ,'<<<<<< ',A7,' All done. CPU-time = ',F8.2,' Sec. ',
c$$$c    &            'Total =',I4.2,'H ',I2.2,'M ',I2.2,'S. >>>>>>')
c$$$ 2000 FORMAT(1H ,'<<<<<< ',A7,' All done. CPU =',F8.2,' s. ',
c$$$     &            'VU =',F8.2,' s. V-Rate =',F5.1,' >>>>>>')
c$$$C     ****************************************************************
```

```
c$$$c      CALL CLOCK(TIME1,0,1)
c$$$       Call ClockV(VU1,Time1,0,1)
c$$$       STEP=TIME1-TIME0
c$$$       VStep=VU1-VU0
c$$$       VRate=(VStep/Step)*100.0E0
c$$$c      ICPU=NINT(TIME1)
c$$$c      CALL TOHMS(ICPU,IHR,IMIN,ISEC)
c$$$c      WRITE(IOUT,2000) RNAME,STEP,IHR,IMIN,ISEC
c$$$       WRITE(IOUT,2000) RNAME,STEP,VStep,VRate
c$$$C
c$$$       TIME0=TIME1
c$$$       VU0  =VU1
c$$$C
c$$$       RETURN
c$$$       END
C
C@(#)=TimeF(Fun) ... Fujitsu FORTRAN
      Integer Function TimeF()
C
C     Returns the value of time since 00:00:00 JST 1992-01-01.
C      Unit : second
C      No arguments required.
C     USE Fujitsu System routines.
C      1 ... Subroutine Date(Chr8) Chr8:YY-MM-DD
C      2 ... Subroutine Time(Int4) Int4:Time(msec)
C     Note.
C      A.D. 2000 ... Leap year (because mod(2000,400) = 0).
C      A.D. 2100 ... This program not available.
C     ****************************************************************
      Implicit Integer (a-z)
      Character*8 Today
      Dimension MDay(12)
      Save Syear,MDay
      Data Syear/92/
      Data MDay/31,28,31,30,31,30,31,31,30,31,30,31/
C     ****************************************************************
c$$$       Call Date(Today)
coht       Call Time(Now)
coht       Now=Now/1000
C
coht       Day=0
C
C     First is year.
coht       Read(Today(1:2),'(i2)') TYear
C
C     New century.
coht       If (TYear .lt. SYear) TYear=TYear+100
coht       Year=TYear-SYear
C
C     1992 ... Leap year
coht       Do 10 i=0,Year-1
coht         Day=Day+365
coht         If (mod(i,4) .eq. 0) Day=Day+1
coht    10 Continue
C
C     Second is month.
coht       Read(Today(4:5),'(i2)') TMonth
coht       Do 20 i=1,TMonth-1
coht         Day=Day+MDay(i)
coht    20 Continue
coht       If (mod(TYear,4) .eq. 0 .and. Tmonth .gt. 2) Day=Day+1
C
C
C     Then day ...
coht       Read(Today(7:8),'(i2)') SDay
coht       Day=Day+SDay-1
C
coht       TimeF=Day*24*3600+Now
C
      Return
      End
      subroutine tmb10
c*====================================================================*
c                                  @       @@@                        *
c       @@@@@  @     @  @@@@@     @@      @   @                      *
c         @    @@  @@  @    @  @ @    @     @                        *
c         @    @ @@ @  @@@@@     @    @     @                        *
c         @    @    @  @    @     @    @     @                        *
c         @    @    @  @    @     @    @   @                          *
c         @    @    @  @@@@@   @@@@@    @@@                           *
c*====================================================================*
c
c  get trans. moment elements between double excitaion state type I
c  and HF state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $      tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $      dummy1(1)
c
c
      ntau = 1
      do 1 nomg = momega(2)+1, momega(3)
        ncimat = nomg*( nomg - 1 )/2 + ntau
        tmbx( ncimat ) = 0.0d0
        tmby( ncimat ) = 0.0d0
        tmbz( ncimat ) = 0.0d0
    1 continue
c
c
      return
      end
      subroutine tmb11
c*====================================================================*
c                                  @       @                          *
c       @@@@@  @     @  @@@@@     @@      @@                         *
c         @    @@  @@  @    @  @ @    @ @                            *
c         @    @ @@ @  @@@@@     @       @                            *
c         @    @    @  @    @     @       @                            *
c         @    @    @  @    @     @       @                            *
c         @    @    @  @@@@@   @@@@@   @@@@@                          *
c*====================================================================*
c
c  get trans. moment elements between type I
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $      tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $      dummy1(1)
c
c
      do 1 nomg = momega(2)+1, momega(3)
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        do 2 ntau = momega(2)+1, nomg
          ir = lci1( ntau )
          it = lci3( ntau ) + nocs
          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
            nut = it*( it + 1 )/2
            nur = ir*( ir + 1 )/2
            x00 = x00 + t1x + 2.0d0*tdx( nut ) - 2.0d0*tdx( nur )
            y00 = y00 + t1y + 2.0d0*tdy( nut ) - 2.0d0*tdy( nur )
            z00 = z00 + t1z + 2.0d0*tdz( nut ) - 2.0d0*tdz( nur )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb1s
c*====================================================================*
c                                  @                                  *
c       @@@@@  @     @  @@@@@    @@      @@@@                        *
c         @    @@  @@  @    @  @ @     @                             *
c         @    @ @@ @  @@@@@     @      @@@@                         *
c         @    @    @  @    @     @    @     @                        *
c         @    @    @  @    @     @    @     @                        *
c         @    @    @  @@@@@   @@@@@    @@@@                          *
```

```
c=====================================================================*
c
c  get trans. moment elements between type I and
c  single excitation state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $      tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $      dummy1(1)
c
c
      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momega(2)+1, momega(3)
        ir = lci1( nomg )
        it = lci3( nomg ) + nocs
        do 2 ntau = momega(1)+1, momega(2)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
            kpp  = max0( ir, it )
            kppx = min0( ir, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + root2*tdx( nu )
            y00 = y00 + root2*tdy( nu )
            z00 = z00 + root2*tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb20
c=====================================================================*
c                                 @@@@@    @@@                        *
c       @@@@@  @     @  @@@@@  @      @  @   @                       *
c         @    @@  @@  @    @        @ @     @                       *
c         @    @ @@ @  @@@@@   @@@@@  @      @                       *
c         @    @    @  @    @ @        @     @                       *
c         @    @    @  @    @ @         @   @                        *
c         @    @    @  @@@@@  @@@@@@@   @@@                          *
c=====================================================================*
c
c  get trans. moment elements between double excitaion state type II
c  and HF state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $      tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $      dummy1(1)
c
c
      ntau = 1
      do 1 nomg = momega(3)+1, momega(4)
        ncimat = nomg*( nomg - 1 )/2 + ntau
        tmbx( ncimat ) = 0.0d0
        tmby( ncimat ) = 0.0d0
        tmbz( ncimat ) = 0.0d0
    1 continue
c
c
      return
      end
      subroutine tmb21
c=====================================================================*
c                                 @@@@@     @                         *
```

```
c       @@@@@  @     @  @@@@@  @      @   @@                         *
c         @    @@  @@  @    @        @  @ @                          *
c         @    @ @@ @  @@@@@   @@@@@     @                           *
c         @    @    @  @    @ @         @                            *
c         @    @    @  @    @ @         @                            *
c         @    @    @  @@@@@  @@@@@@@  @@@@@                         *
c=====================================================================*
c
c  get trans. moment elements between type II and type I
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $      tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $      dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 21 ]'

      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momega(3)+1, momega(4)
c$ohtawara
c         ir = lci1( nomg )
c         is = lci2( nomg )
c         it = lci3( nomg ) + nocs

        do 2 ntau = momega(2)+1, momega(3)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          ir = i1
          is = i2
          it = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
c zero except for (delta_ir.eq.1).and.(delta_kt.eq.1)
c          ncimat = nomg*( nomg - 1 )/2 + ntau
c          tmbx( ncimat ) = 0.0d0
c          tmby( ncimat ) = 0.0d0
c          tmbz( ncimat ) = 0.0d0
 3        continue
c         write(iout,10) i,k,i,k, ir,it,is,it
c 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
c     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k,i,k, ir,it,is,it
 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( ir, is )
            kppx = min0( ir, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + root2*tdx( nu )
            y00 = y00 + root2*tdy( nu )
            z00 = z00 + root2*tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
```

```
      return
      end
      subroutine tmb22
c=====================================================================*
c                                  @@@@@   @@@@@                      *
c       @@@@@  @    @  @@@@@  @     @ @     @                        *
c         @    @@  @@  @    @        @       @                        *
c         @    @ @@ @  @@@@@   @@@@@   @@@@@                          *
c         @    @    @  @    @ @       @                                *
c         @    @    @  @    @ @       @                                *
c         @    @    @  @@@@@  @@@@@@@ @@@@@@@                          *
c=====================================================================*
c
c  get trans. moment elements between type II
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 22 ]'

      do 1 nomg = momega(3)+1, momega(4)
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        do 2 ntau = momega(3)+1, nomg
          ir = lci1( ntau )
          is = lci2( ntau )
          it = lci3( ntau ) + nocs
c$ohtawara
c         write(iout,10) i,k,j,k, ir,it,is,it
c 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
c     &       '(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k,j,k, ir,it,is,it
 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &       '(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( j, is )
            kppx = min0( j, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 - tdx( nu )
            y00 = y00 - tdy( nu )
            z00 = z00 - tdz( nu )
            if( j .eq. is ) then
              nut = it*( it + 1 )/2
              nur = ir*( ir + 1 )/2
              x00 = x00 + t1x + 2.0d0*tdx( nut ) - tdx( nur )
              y00 = y00 + t1y + 2.0d0*tdy( nut ) - tdy( nur )
              z00 = z00 + t1z + 2.0d0*tdz( nut ) - tdz( nur )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb2s
c=====================================================================*
c                                  @@@@@                              *
c       @@@@@  @    @  @@@@@  @     @   @@@@                         *
c         @    @@  @@  @    @        @  @                             *
```

```
c         @    @ @@ @  @@@@@   @@@@@    @@@@                         *
c         @    @    @  @    @ @             @                         *
c         @    @    @  @    @ @         @    @                        *
c         @    @    @  @@@@@  @@@@@@   @@@@                           *
c=====================================================================*
c
c  get trans. moment elements between type II and
c  single excitation state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 2s ]'

      do 1 nomg = momega(3)+1, momega(4)
c$ ohtawara
c         ir = lci1( nomg )
c         is = lci2( nomg )

        it = lci3( nomg ) + nocs
        do 2 ntau = momega(1)+1, momega(2)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ ohtawara
          ir = lci1( nomg )
          is = lci2( nomg )
          i1 = ir
          i2 = is
          if(is.eq.i) then
            ir = i2
            is = i1
        endif
c         write(iout,10) i,k, ir,it,is,it
c 10      format('(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k, ir,it,is,it
 10      format('(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( is, it )
            kppx = min0( is, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 - tdx( nu )
            y00 = y00 - tdy( nu )
            z00 = z00 - tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb30( momgs,  momge  )
c=====================================================================*
c                                  @@@@@    @@@                       *
c       @@@@@  @    @  @@@@@  @     @  @   @                         *
c         @    @@  @@  @    @        @ @     @                        *
c         @    @ @@ @  @@@@@   @@@@@  @     @                        *
c         @    @    @  @    @ @       @ @     @                        *
c         @    @    @  @    @ @     @  @   @                           *
c         @    @    @  @@@@@   @@@@@    @@@                           *
c=====================================================================*
c
```

```
c  get trans. moment elements between double excitaion state type III
c  and HF state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      ntau = 1
      do 1 nomg = momgs, momge
        ncimat = nomg*( nomg - 1 )/2 + ntau
        tmbx( ncimat ) = 0.0d0
        tmby( ncimat ) = 0.0d0
        tmbz( ncimat ) = 0.0d0
    1 continue
c
c
      return
      end
      subroutine tmb31( momgs,  momge,  mtaus,  mtaue  )
c=====================================================================*
c                                 @@@@@     @                         *
c       @@@@@  @    @  @@@@@  @      @   @@                         *
c         @    @@  @@  @    @        @  @ @                          *
c         @    @ @@ @  @@@@@   @@@@@     @                           *
c         @    @    @  @    @        @    @                           *
c         @    @    @  @    @ @      @    @                           *
c         @    @    @  @@@@@   @@@@@   @@@@@                          *
c=====================================================================*
c
c  get trans. moment elements between type III and type I
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)

c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 31 ]'

      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momgs, momge
c$ohtawara
c         ir = lci1( nomg )
c         it = lci3( nomg ) + nocs
c         iu = lci4( nomg ) + nocs

        do 2 ntau = mtaus, mtaue
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          ir = i1

          i2 = lci3( nomg ) + nocs
          i3 = lci4( nomg ) + nocs

          it = i2
          iu = i3
c$$$ my bug fixed. 1996.2.6
c$$$          if(ir.eq.i .and. iu.eq.k)  go to 3
          if(ir.eq.i .and. iu.eq.k)  then
             it = i3
             iu = i2
          endif
c$$$          if(ir.eq.i .and. iu.eq.k)  go to 3

c zero except for (delta_ir.eq.1).and.(delta_ku.eq.1)
c          ncimat = nomg*( nomg - 1 )/2 + ntau
c          tmbx( ncimat ) = 0.0d0
c          tmby( ncimat ) = 0.0d0
c          tmbz( ncimat ) = 0.0d0

c$$$ 3        continue

c       write(iout,10) i,k,i,k, ir,it,ir,iu
c 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
c     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k,i,k, ir,it,ir,iu
 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( it, iu )
            kppx = min0( it, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + root2*tdx( nu )
            y00 = y00 + root2*tdy( nu )
            z00 = z00 + root2*tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb32( momgs,  momge,  mtaus,  mtaue  )
c=====================================================================*
c                                 @@@@@   @@@@@                       *
c       @@@@@  @    @  @@@@@  @      @ @      @                      *
c         @    @@  @@  @    @        @        @                      *
c         @    @ @@ @  @@@@@   @@@@@   @@@@@                          *
c         @    @    @  @    @        @ @                              *
c         @    @    @  @    @ @      @ @                              *
c         @    @    @  @@@@@   @@@@@  @@@@@@@                         *
c=====================================================================*
c
c  get trans. moment elements between type III and type II
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      do 1 nomg = momgs, momge
        do 2 ntau = mtaus, mtaue
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = 0.0d0
          tmby( ncimat ) = 0.0d0
          tmbz( ncimat ) = 0.0d0
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb33( momgs,  momge,  mtaus  )
c=====================================================================*
c                                 @@@@@   @@@@@                       *
c       @@@@@  @    @  @@@@@  @      @ @      @                      *
c         @    @@  @@  @    @        @        @                      *
c         @    @ @@ @  @@@@@   @@@@@   @@@@@                          *
c         @    @    @  @    @        @        @                      *
c         @    @    @  @    @ @      @ @      @                      *
c         @    @    @  @@@@@   @@@@@   @@@@@                          *
```

```
c=====================================================================*
c
c  get trans. moment elements between type III
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $      tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $      dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 33 ]'

      do 1 nomg = momgs, momge
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
        do 2 ntau = mtaus, nomg
          ir = lci1( ntau )
          it = lci3( ntau ) + nocs
          iu = lci4( ntau ) + nocs
c$ohtawara
c         write(iout,10) i,k,i,l, ir,it,ir,iu
c 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
c     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k,i,l, ir,it,ir,iu
 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( l, iu )
            kppx = min0( l, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + tdx( nu )
            y00 = y00 + tdy( nu )
            z00 = z00 + tdz( nu )
            if( l .eq. iu ) then
              nut = it*( it + 1 )/2
              nur = ir*( ir + 1 )/2
              x00 = x00 + tlx + tdx( nut ) - 2.0d0*tdx( nur )
              y00 = y00 + tly + tdy( nut ) - 2.0d0*tdy( nur )
              z00 = z00 + tlz + tdz( nut ) - 2.0d0*tdz( nur )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb3s( momgs,  momge,  mtaus,  mtaue  )
c=====================================================================*
c                                 @@@@@                               *
c       @@@@@  @    @  @@@@@  @     @   @@@@                         *
c         @    @@  @@  @    @        @  @                             *
c         @    @ @@ @  @@@@@   @@@@@    @@@@                         *
c         @    @    @  @    @        @       @                        *
c         @    @    @  @    @ @      @  @    @                        *
c         @    @    @  @@@@@   @@@@@    @@@@                         *
c=====================================================================*
c
c  get trans. moment elements between type III and
c  single excitation state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $      tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $      dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 3s ]'

      do 1 nomg = momgs, momge
        ir = lci1( nomg )
c$ohtawara
c       it = lci3( nomg ) + nocs
c       iu = lci4( nomg ) + nocs

        do 2 ntau = mtaus, mtaue
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          it = lci3( nomg ) + nocs
          iu = lci4( nomg ) + nocs
          i1 = it
          i2 = iu
c$$$ my bug  1996.2.6
c$$$      if(iu.eq.k) then
          if(iu.ne.k) then
             it = i2
             iu = i1
          endif
c         write(iout,10) i,k, ir,it,ir,iu
c 10      format('(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
c$ohtawara bug in equation(4.8)
c          if( i .eq. ir  .and.  k .eq. it ) then
          if( i .eq. ir  .and.  k .eq. iu ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k, ir,it,ir,iu
 10      format('(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( ir, it )
            kppx = min0( ir, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + tdx( nu )
            y00 = y00 + tdy( nu )
            z00 = z00 + tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb40
c=====================================================================*
c                                 @         @@@                       *
c       @@@@@  @    @  @@@@@  @    @   @   @                         *
c         @    @@  @@  @    @ @    @  @ @     @                       *
c         @    @ @@ @  @@@@@  @    @  @ @     @                       *
c         @    @    @  @    @ @@@@@@@ @     @                        *
c         @    @    @  @    @       @   @   @                         *
c         @    @    @  @@@@@       @    @@@                          *
c=====================================================================*
c
c  get trans. moment elements between double excitaion state type IV
c  and HF state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
```

```
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      ntau = 1
      do 1 nomg = momega(5)+1, momega(6)
        ncimat = nomg*( nomg - 1 )/2 + ntau
        tmbx( ncimat ) = 0.0d0
        tmby( ncimat ) = 0.0d0
        tmbz( ncimat ) = 0.0d0
    1 continue
c
c
      return
      end
      subroutine tmb41
c=====================================================================*
c                                @          @                         *
c       @@@@@  @    @  @@@@@  @    @    @@                          *
c         @    @@  @@  @    @ @    @   @ @                           *
c         @    @ @@ @  @@@@@  @    @     @                           *
c         @    @    @  @    @ @@@@@@@    @                           *
c         @    @    @  @    @         @    @                           *
c         @    @    @  @@@@@         @   @@@@@                        *
c=====================================================================*
c
c  get trans. moment elements between type IV and type I
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      do 1 nomg = momega(5)+1, momega(6)
        do 2 ntau = momega(2)+1, momega(3)
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = 0.0d0
          tmby( ncimat ) = 0.0d0
          tmbz( ncimat ) = 0.0d0
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb42
c=====================================================================*
c                                @         @@@@@                      *
c       @@@@@  @    @  @@@@@  @    @  @      @                      *
c         @    @@  @@  @    @ @    @  @      @                      *
c         @    @ @@ @  @@@@@  @    @   @@@@@                         *
c         @    @    @  @    @ @@@@@@@ @                               *
c         @    @    @  @    @       @  @                               *
c         @    @    @  @@@@@        @  @@@@@@@                        *
c=====================================================================*
c
c  get trans. moment elements between type IV and type II
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 42 ]'
      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
c         ir = lci1( nomg )
c         is = lci2( nomg )
c         it = lci3( nomg ) + nocs
c         iu = lci4( nomg ) + nocs

        do 2 ntau = momega(3)+1, momega(4)
          i = lci1( ntau )
          j = lci2( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
c$$$ my bug fixed 1996.2.6
c$$$          if( is.eq.i .and. ir.eq.j .and. iu.eq.k ) go to 3
          if( ir.ne.i ) go to 5
          if( it.ne.k ) then
             ir = i1
             is = i2
             it = i4
             iu = i3
          endif
          go to 3
c$$$ my bug fixed 1996.2.6
c$$$          if( is.eq.i .and. ir.eq.j .and. iu.eq.k ) go to 3
 5        ir = i2
          is = i1
          it = i3
          iu = i4
c$$$ my bug fixed 1996.2.6
c$$$          if( is.eq.i .and. ir.eq.j .and. iu.eq.k ) go to 3
          if( it.ne.k ) then
             ir = i2
             is = i1
             it = i4
             iu = i3
          endif
c$$$ my bug fixed 1996.2.6
c$$$          if( is.eq.i .and. ir.eq.j .and. iu.eq.k ) go to 3
c zero except for delta_is, etc...
c          ncimat = nomg*( nomg - 1 )/2 + ntau
c          tmbx( ncimat ) = 0.0d0
c          tmby( ncimat ) = 0.0d0
c          tmbz( ncimat ) = 0.0d0
c$$ my bug fixed 1996.2.6

 3        continue

c         write(iout,10) i,k,j,k, ir,it,is,iu
c 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
c     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  j .eq. is  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k,j,k, ir,it,is,iu
 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( it, iu )
            kppx = min0( it, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + root2*tdx( nu )
            y00 = y00 + root2*tdy( nu )
            z00 = z00 + root2*tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
```

```
    1 continue
c
c
      return
      end
      subroutine tmb43
c=====================================================================*
c                                @        @@@@@                        *
c       @@@@@  @    @  @@@@@  @    @  @     @                        *
c         @    @@  @@  @    @ @    @        @                        *
c         @    @ @@ @  @@@@@  @    @   @@@@@                         *
c         @    @    @  @    @ @@@@@@@       @                        *
c         @    @    @  @    @       @  @     @                        *
c         @    @    @  @@@@@       @   @@@@@                         *
c=====================================================================*
c
c  get trans. moment elements between type IV and type III
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 43 ]'

      root2 = dsqrt( 2.0d0 )
      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )
c        it = lci3( nomg ) + nocs
c        iu = lci4( nomg ) + nocs

        do 2 ntau = momega(4)+1, momega(5)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
          l = lci4( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if( it.eq.k .and. ir.eq.i .and. iu.eq.l ) go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if( it.eq.k .and. ir.eq.i .and. iu.eq.l ) go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if( it.eq.k .and. ir.eq.i .and. iu.eq.l ) go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if( it.eq.k .and. ir.eq.i .and. iu.eq.l ) go to 3
c zero except for delta_is, etc...
c          ncimat = nomg*( nomg - 1 )/2 + ntau
c          tmbx( ncimat ) = 0.0d0
c          tmby( ncimat ) = 0.0d0
c          tmbz( ncimat ) = 0.0d0
 3        continue
c         write(iout,10) i,k,i,l, ir,it,is,iu
c 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
c     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  l .eq. iu  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k,i,l, ir,it,is,iu
 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( ir, is )
            kppx = min0( ir, is )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + root2*tdx( nu )
            y00 = y00 + root2*tdy( nu )
            z00 = z00 + root2*tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb44
c=====================================================================*
c                                @        @                            *
c       @@@@@  @    @  @@@@@  @    @  @    @                         *
c         @    @@  @@  @    @ @    @  @    @                         *
c         @    @ @@ @  @@@@@  @    @  @    @                         *
c         @    @    @  @    @ @@@@@@@ @@@@@@@                         *
c         @    @    @  @    @       @       @                         *
c         @    @    @  @@@@@       @       @                         *
c=====================================================================*
c
c  get trans. moment elements between type IV
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 44 ]'

      do 1 nomg = momega(5)+1, momega(6)
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
        do 2 ntau = momega(5)+1, nomg
          ir = lci1( ntau )
          is = lci2( ntau )
          it = lci3( ntau ) + nocs
          iu = lci4( ntau ) + nocs
c$ohtawara
c       write(iout,10) i,k,j,k, ir,it,is,it
c 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
c     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k,j,k, ir,it,is,it
 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

c$ohtawara bug in formula (4.25)
c            if( j .eq. is ) then
c              nut = it*( it + 1 )/2
```

```
c              nur = ir*( ir + 1 )/2
c              x00 = x00 + tlx + tdx( nut ) - tdx( nur )
c              y00 = y00 + tly + tdy( nut ) - tdy( nur )
c              z00 = z00 + tlz + tdz( nut ) - tdz( nur )
c              kpp  = max0( 1, iu )
c              kppx = min0( 1, iu )
c              nu   = kpp*( kpp - 1 )/2 + kppx
c              x00 = x00 + tdx( nu )
c              y00 = y00 + tdy( nu )
c              z00 = z00 + tdz( nu )
c            endif
            if( j .eq. is  .and.  1 .eq. iu ) then
              nut = it*( it + 1 )/2
              nur = ir*( ir + 1 )/2
              x00 = x00 + tlx + tdx( nut ) - tdx( nur )
              y00 = y00 + tly + tdy( nut ) - tdy( nur )
              z00 = z00 + tlz + tdz( nut ) - tdz( nur )
            endif
            if( j .eq. is ) then
              kpp  = max0( 1, iu )
              kppx = min0( 1, iu )
              nu   = kpp*( kpp - 1 )/2 + kppx
              x00 = x00 + tdx( nu )
              y00 = y00 + tdy( nu )
              z00 = z00 + tdz( nu )
            endif
c$ohtawara --- end of bug fix ---

            if( 1 .eq. iu ) then
              kpp  = max0( j, is )
              kppx = min0( j, is )
              nu   = kpp*( kpp - 1 )/2 + kppx
              x00 = x00 - tdx( nu )
              y00 = y00 - tdy( nu )
              z00 = z00 - tdz( nu )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb4s
c=====================================================================*
c                                  @                                  *
c       @@@@@  @    @  @@@@@  @    @    @@@@                        *
c         @    @@  @@  @    @ @    @   @                             *
c         @    @ @@ @  @@@@@  @    @    @@@@                        *
c         @    @    @  @    @ @@@@@@@       @                        *
c         @    @    @  @    @       @   @    @                        *
c         @    @    @  @@@@@       @    @@@@                        *
c=====================================================================*
c
c  get trans. moment elements between type IV and
c  single excitation state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummyl(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 4s ]'

      coeff0 = -1.0d0/dsqrt( 2.0d0 )
      do 1 nomg = momega(5)+1, momega(6)
c$ohtawara
c        ir = lci1( nomg )
c        is = lci2( nomg )
c        it = lci3( nomg ) + nocs
c        iu = lci4( nomg ) + nocs

        do 2 ntau = momega(1)+1, momega(2)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs

          ir = i1
          is = i2
          it = i3
          iu = i4
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
c zero except for delta_is, etc...
c          ncimat = nomg*( nomg - 1 )/2 + ntau
c          tmbx( ncimat ) = 0.0d0
c          tmby( ncimat ) = 0.0d0
c          tmbz( ncimat ) = 0.0d0
 3        continue
c         write(iout,10) i,k, ir,it,is,iu
c 10      format('(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k, ir,it,is,iu
 10      format('(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( is, iu )
            kppx = min0( is, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + coeff0*tdx( nu )
            y00 = y00 + coeff0*tdy( nu )
            z00 = z00 + coeff0*tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb50
c=====================================================================*
c                                  @@@@@@@   @@@                      *
c       @@@@@  @    @  @@@@@  @         @   @                        *
c         @    @@  @@  @    @ @        @     @                        *
c         @    @ @@ @  @@@@@  @@@@@@  @     @                        *
c         @    @    @  @    @        @ @     @                        *
c         @    @    @  @    @ @      @  @   @                        *
c         @    @    @  @@@@@   @@@@@    @@@                           *
c=====================================================================*
c
c  get trans. moment elements between double excitaion state type V
c  and HF state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
```

```
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      ntau = 1
      do 1 nomg = momega(6)+1, momega(7)
        ncimat = nomg*( nomg - 1 )/2 + ntau
        tmbx( ncimat ) = 0.0d0
        tmby( ncimat ) = 0.0d0
        tmbz( ncimat ) = 0.0d0
    1 continue
c
c
      return
      end
      subroutine tmb51
c=====================================================================*
c                                @@@@@@@    @                         *
c       @@@@@  @    @  @@@@@  @          @@                         *
c         @    @@  @@  @    @ @         @ @                         *
c         @    @ @@ @  @@@@@  @@@@@@     @                          *
c         @    @    @  @    @        @    @                          *
c         @    @    @  @    @ @      @    @                          *
c         @    @    @  @@@@@   @@@@@   @@@@@                         *
c=====================================================================*
c
c  get trans. moment elements between type V and type I
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      do 1 nomg = momega(6)+1, momega(7)
        do 2 ntau = momega(2)+1, momega(3)
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = 0.0d0
          tmby( ncimat ) = 0.0d0
          tmbz( ncimat ) = 0.0d0
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb52
c=====================================================================*
c                                @@@@@@@  @@@@@                      *
c       @@@@@  @    @  @@@@@  @        @     @                      *
c         @    @@  @@  @    @ @                @                     *
c         @    @ @@ @  @@@@@  @@@@@@   @@@@@                         *
c         @    @    @  @    @        @ @                             *
c         @    @    @  @    @ @      @ @                             *
c         @    @    @  @@@@@   @@@@@  @@@@@@@                        *
c=====================================================================*
c
c  get trans. moment elements between type V and type II
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      do 1 nomg = momega(6)+1, momega(7)
        do 2 ntau = momega(3)+1, momega(4)
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = 0.0d0
          tmby( ncimat ) = 0.0d0
          tmbz( ncimat ) = 0.0d0
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb53
c=====================================================================*
c                                @@@@@@@  @@@@@                      *
c       @@@@@  @    @  @@@@@  @        @     @                      *
c         @    @@  @@  @    @ @                @                     *
c         @    @ @@ @  @@@@@  @@@@@@   @@@@@                         *
c         @    @    @  @    @        @       @                       *
c         @    @    @  @    @ @      @ @     @                       *
c         @    @    @  @@@@@   @@@@@   @@@@@                         *
c=====================================================================*
c
c  get trans. moment elements between type V and type III
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      do 1 nomg = momega(6)+1, momega(7)
        do 2 ntau = momega(4)+1, momega(5)
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = 0.0d0
          tmby( ncimat ) = 0.0d0
          tmbz( ncimat ) = 0.0d0
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb54
c=====================================================================*
c                                @@@@@@@ @                           *
c       @@@@@  @    @  @@@@@  @        @     @                      *
c         @    @@  @@  @    @ @        @     @                      *
c         @    @ @@ @  @@@@@  @@@@@@   @     @                      *
c         @    @    @  @    @        @ @@@@@@@                       *
c         @    @    @  @    @ @      @       @                       *
c         @    @    @  @@@@@   @@@@@        @                       *
c=====================================================================*
c
c  get trans. moment elements between type V and type IV
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      do 1 nomg = momega(6)+1, momega(7)
        do 2 ntau = momega(5)+1, momega(6)
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = 0.0d0
          tmby( ncimat ) = 0.0d0
          tmbz( ncimat ) = 0.0d0
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb55
c=====================================================================*
c                                @@@@@@@ @@@@@@@                     *
```

```
c       @@@@@  @    @  @@@@@  @       @                                 *
c         @    @@  @@  @    @ @       @                                 *
c         @    @ @@ @  @@@@@  @@@@@@  @@@@@@                            *
c         @    @    @  @    @       @       @                            *
c         @    @    @  @    @ @     @ @     @                            *
c         @    @    @  @@@@@   @@@@@   @@@@@                            *
c=====================================================================*
c
c  get trans. moment elements between type V
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 55 ]'

      do 1 nomg = momega(6)+1, momega(7)
        i = lci1( nomg )
        j = lci2( nomg )
        k = lci3( nomg ) + nocs
        l = lci4( nomg ) + nocs
        do 2 ntau = momega(6)+1, nomg
          ir = lci1( ntau )
          is = lci2( ntau )
          it = lci3( ntau ) + nocs
          iu = lci4( ntau ) + nocs
c       write(iout,10) i,k,j,k, ir,it,is,it
c 10    format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
c     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k,j,k, ir,it,is,it
 10   format('(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')

c$ohtawara bug in formula (4.25)
c            if( j .eq. is ) then
c              nut = it*( it + 1 )/2
c              nur = ir*( ir + 1 )/2
c              x00 = x00 + t1x + tdx( nut ) - tdx( nur )
c              y00 = y00 + t1y + tdy( nut ) - tdy( nur )
c              z00 = z00 + t1z + tdz( nut ) - tdz( nur )
c              kpp  = max0( l, iu )
c              kppx = min0( l, iu )
c              nu   = kpp*( kpp - 1 )/2 + kppx
c              x00 = x00 + tdx( nu )
c              y00 = y00 + tdy( nu )
c              z00 = z00 + tdz( nu )
c            endif
            if( j .eq. is  .and.  l .eq. iu ) then
              nut = it*( it + 1 )/2
              nur = ir*( ir + 1 )/2
              x00 = x00 + t1x + tdx( nut ) - tdx( nur )
              y00 = y00 + t1y + tdy( nut ) - tdy( nur )
              z00 = z00 + t1z + tdz( nut ) - tdz( nur )
            endif
            if( j .eq. is ) then
              kpp  = max0( l, iu )
              kppx = min0( l, iu )
              nu   = kpp*( kpp - 1 )/2 + kppx
              x00 = x00 + tdx( nu )
              y00 = y00 + tdy( nu )
              z00 = z00 + tdz( nu )
            endif
c$ohtawara --- end of bug fix ---

            if( l .eq. iu ) then
              kpp  = max0( j, is )
              kppx = min0( j, is )
              nu   = kpp*( kpp - 1 )/2 + kppx
              x00 = x00 - tdx( nu )
              y00 = y00 - tdy( nu )
              z00 = z00 - tdz( nu )
            endif
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmb5s
c=====================================================================*
c                                 @@@@@@@                             *
c       @@@@@  @    @  @@@@@  @         @@@@                        *
c         @    @@  @@  @    @ @       @                               *
c         @    @ @@ @  @@@@@  @@@@@@    @@@@                          *
c         @    @    @  @    @       @       @                          *
c         @    @    @  @    @ @     @  @    @                          *
c         @    @    @  @@@@@   @@@@@    @@@@                          *
c=====================================================================*
c
c  get trans. moment elements between type V and
c  single excitation state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,*) '[ 5s ]'

      coeff0 = -dsqrt( 1.5d0 )
      do 1 nomg = momega(6)+1, momega(7)
c$ohtawara
c         ir = lci1( nomg )
c         is = lci2( nomg )
c         it = lci3( nomg ) + nocs
c         iu = lci4( nomg ) + nocs

        do 2 ntau = momega(1)+1, momega(2)
          i = lci1( ntau )
          k = lci3( ntau ) + nocs
c$ohtawara
          i1 = lci1( nomg )
          i2 = lci2( nomg )
          i3 = lci3( nomg ) + nocs
          i4 = lci4( nomg ) + nocs
          ir = i1
          is = i2
          it = i3
          iu = i4
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i1
          is = i2
          it = i4
          iu = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i3
          iu = i4
          if(ir.eq.i .and. it.eq.k)  go to 3
          ir = i2
          is = i1
          it = i4
          iu = i3
          if(ir.eq.i .and. it.eq.k)  go to 3
```

```
c zero except for delta_is, etc...
c          ncimat = nomg*( nomg - 1 )/2 + ntau
c          tmbx( ncimat ) = 0.0d0
c          tmby( ncimat ) = 0.0d0
c          tmbz( ncimat ) = 0.0d0
 3        continue
c         write(iout,10) i,k, ir,it,is,iu
c 10      format('(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')

          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( i .eq. ir  .and.  k .eq. it ) then
c$ohtawara
c for small print out (given by sdci01.h)
      if (iprint_yes .ge. 0) write(iout,10) i,k, ir,it,is,iu
 10      format('(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')

            kpp  = max0( is, iu )
            kppx = min0( is, iu )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + coeff0*tdx( nu )
            y00 = y00 + coeff0*tdy( nu )
            z00 = z00 + coeff0*tdz( nu )
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmbs0
c=====================================================================*
c                                        @@@                          *
c       @@@@@  @    @  @@@@@    @@@@   @   @                         *
c         @    @@  @@  @    @  @       @     @                       *
c         @    @ @@ @  @@@@@    @@@@  @     @                       *
c         @    @    @  @    @       @ @     @                       *
c         @    @    @  @    @  @    @  @   @                         *
c         @    @    @  @@@@@    @@@@    @@@                           *
c=====================================================================*
c
c  get trans. moment elements between single excitaion state
c  and HF state
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      root2 = dsqrt( 2.0d0 )
      ntau = 1
      do 1 nomg = momega(1)+1, momega(2)
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        kpp  = max0( i, k )
        kppx = min0( i, k )
        nu   = kpp*( kpp - 1 )/2 + kppx
        ncimat = nomg*( nomg - 1 )/2 + ntau
        tmbx( ncimat ) = root2*tdx( nu )
        tmby( ncimat ) = root2*tdy( nu )
        tmbz( ncimat ) = root2*tdz( nu )
    1 continue
c
c     all done. return to caller.
      call clockm('cimtss.   ')
c
c
      return
      end
      subroutine tmbss
c=====================================================================*
c                                                                     *
c       @@@@@  @    @  @@@@@    @@@@    @@@@                         *
c         @    @@  @@  @    @  @    @       @                        *
c         @    @ @@ @  @@@@@    @@@@    @@@@                         *
c         @    @    @  @    @       @       @                        *
c         @    @    @  @    @  @    @  @    @                        *
c         @    @    @  @@@@@    @@@@    @@@@                         *
c=====================================================================*
c
c  get trans. moment elements between single excitaion states
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
c
      common/ work01 / civec( mcics2 ),
     $     tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $     dummy1(1)
c
c
      do 1 nomg = momega(1)+1, momega(2)
        i = lci1( nomg )
        k = lci3( nomg ) + nocs
        do 2 ntau = momega(1)+1, nomg
          ir = lci1( ntau )
          it = lci3( ntau ) + nocs
          x00 = 0.0d0
          y00 = 0.0d0
          z00 = 0.0d0
          if( k .eq. it ) then
            kpp  = max0( i, ir )
            kppx = min0( i, ir )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 - tdx( nu )
            y00 = y00 - tdy( nu )
            z00 = z00 - tdz( nu )
          endif
          if( i .eq. ir ) then
            kpp  = max0( k, it )
            kppx = min0( k, it )
            nu   = kpp*( kpp - 1 )/2 + kppx
            x00 = x00 + tdx( nu )
            y00 = y00 + tdy( nu )
            z00 = z00 + tdz( nu )
          endif
          if( i .eq. ir  .and.  k .eq. it ) then
            x00 = x00 + t1x
            y00 = y00 + t1y
            z00 = z00 + t1z
          endif
          ncimat = nomg*( nomg - 1 )/2 + ntau
          tmbx( ncimat ) = x00
          tmby( ncimat ) = y00
          tmbz( ncimat ) = z00
    2   continue
    1 continue
c
c
      return
      end
      subroutine tmdump
c=====================================================================*
c                                                                     *
c       @@@@@  @    @  @@@@@   @    @  @    @  @@@@@                 *
c         @    @@  @@  @    @  @    @  @@  @@  @    @                 *
c         @    @ @@ @  @    @  @    @  @ @@ @  @    @                 *
c         @    @    @  @    @  @    @  @    @  @@@@@                 *
c         @    @    @  @    @  @    @  @    @  @                      *
c         @    @    @  @@@@@    @@@@   @    @  @                      *
c=====================================================================*
c$$$c
c$$$c-------------   dump sdci eigen vectors
c$$$c
c$$$      implicit real*8 (a-h,o-z)
c$$$      include 'MSSIZE'
c$$$      include 'energy.h'
c$$$      include 'sdci00.h'
c$$$      include 'sdci01.h'
c$$$      include 'sdci02.h'
c$$$      include 'window.h'
c$$$      include 'work02.h'
c$$$c
```

```
c$$$      common /io    / in,iout,ifile(10)
c$$$      common /lvci  / locc(mcicsf),lvac(mcicsf),
c$$$     &                mocc(mcicsf),mvac(mcicsf)
c$$$      common /work01/ ovov(mciint),oovv(mciint),civec(mcicsf,mcicsf),
c$$$     &                dummy(1)
c$$$c
c$$$      do 100 nomg = momega(1)+1, momega(2)
c$$$        i = lci1( nomg )
c$$$        k = lci3( nomg ) + nocs
c$$$        n = nomg - momega( 1 )
c$$$        locc( n ) = lmo( i )
c$$$        lvac( n ) = lmo( k )
c$$$  100 continue
c$$$c
c$$$      nscibs = momega( 2 ) - momega( 1 )
c$$$      if( isdci0 .eq. 1 ) then
c$$$        nend = ncibas
c$$$      else
c$$$        nend = momega( 3 )
c$$$      endif
c$$$      write( iout, 1000 )
c$$$      call civo00( civec,  cieig,  locc,   lvac,   mcicsf, nscibs,
c$$$     $             nend )
c$$$      write( iout, 1100 )
c$$$      call civo01( civec,  cieig,  locc,   lvac,   mcicsf, nscibs,
c$$$     $             nend )
c$$$c
c$$$c
c$$$ 1000 format(1h ,//
c$$$     $       1h ,37('=-')/
c$$$     &       1h ,'eigenvectors of ci hamiltonian matrix'/
c$$$     $       1h ,'(contribution from Hartree-Fock and',
c$$$     $          ' single excitaion states)'/
c$$$     &       1h ,37('=-'))
c$$$ 1100 format(1h ,//
c$$$     $       1h ,37('=-')/
c$$$     &       1h ,'eigenvectors of ci hamiltonian matrix'/
c$$$     $       1h ,'(contribution from double excitaion state type I)'/
c$$$     &       1h ,37('=-'))
c$$$c
c$$$c
      return
      end
C
C@(#)=TMOM(SUB)
      SUBROUTINE TMOM
C
C     GET TRANSITION MOMENTS USING SCI WAVEFUNCTIONS.
C     ***************************************************************
      COMMON /IO    / IN,IOUT,IFILE(10)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /MULTCI/ MULTCI
 2000 Format(1H ,'<<<<<< Enter TMom >>>>>>')
 2010 Format(1H ,'Computation of electronic transition moments.')
 2020 Format(1H ,'Transition mements not calculated because triplet ',
     &           'CI calculation was performed.')
 2030 Format(1H ,'ZDO approximation is used in estimating 1 center ',
     &           'dipole integrals.')
C     ***************************************************************
      IZDO=IOPDIP( 1)
C
      WRITE(IOUT,2000)
      WRITE(IOUT,2010)
C
C     NOW, COMMON /WORK01/ DUMMY1(MCIINT*2),CIVEC(MCICSF,MCICSF),
C                          DUMMY2(1)
C     CIVEC ... EIGENVECTORS OF CI HAMILTONIAN MATRIX.
C
C     MOVE EIGENVECTORS OF CI HAMILTONIAN MATRIX AND PACK THEM.
      CALL PACKCV
C
C     NOW, COMMON /WORK01/ CIVEC(MCICSF**2),
C                          DUMMY1(1)
C     CIVEC ... EIGENVECTORS OF CI Hamiltonian MATRIX (PACKED FORM).
C
      IF(MULTCI.EQ.3) THEN
        WRITE(IOUT,2020)
        GO TO 10
      END IF
C
C     GET MOLECULAR INTEGRALS FOR DIPOLE MOMENTS, TDX, TDY AND TDZ.
      IF(IZDO.EQ.1) THEN
        WRITE(IOUT,2030)
        CALL DIPZDO
      ELSE
        CALL DIP1CN
      END IF
C
C     NOW, COMMON /WORK01/ CIVEC(MCICSF**2),
C                          TDX(MCIPCK),TDY(MCIPCK),TDZ(MCIPCK)
C          COMMON /WORK02/ C(MAXB2),EIG(MAXBAS),CIEIG(MCICSF)
C
C     CIVEC ... EIGENVECTORS OF CI Hamiltonian MATRIX (PACKED FORM).
C     TDX, TDY AND TDZ ... MOLECULAR INTEGRALS FOR DIPOLE MOMENTS.
C     C ... MOLECULAR ORBITAL COEFFICIENTS.
C     EIG ... ORBITAL ENERGIES.
C     CIEIG ... EIGENVALUES OF CI Hamiltonian MATRIX.
C
C     MOVE DATA ON COMMON /WORK01/ AND /WORK02/.
   10 CALL SETTM
C
C     NOW, COMMON /WORK01/ DUMMY
C          COMMON /WORK02/ CIEIG(MCICSF),CIVEC(MCICSF**2),
C                          TDX(MBPCK),TDY(MBPCK),TDZ(MBPCK)
C     CIVEC ... EIGENVECTORS OF CI Hamiltonian MATRIX (PACKED FORM).
C     CIEIG ... EIGENVALUES OF CI Hamiltonian MATRIX.
C     TDX, TDY AND TDZ ... MOLECULAR INTEGRALS FOR DIPOLE MOMENTS.
C
C     THEN GET TRANSITION MOMENTS USING SCI WAVEFUNCTIONS.
      IF(MULTCI.EQ.1) THEN
        CALL TMOMCI
      END IF
C
C     ALL DONE. RETURN TO CALLER.
      CALL CLOCKM('TMom.  ')
      RETURN
      END
      subroutine tmsdbs
c=====================================================================*
c                                                                     *
c       @@@@@  @    @   @@@@   @@@@@   @@@@@    @@@@                 *
c         @    @@  @@  @       @    @  @    @  @                    *
c         @    @ @@ @   @@@@   @    @  @@@@@    @@@@                 *
c         @    @    @       @  @    @  @    @       @                *
c         @    @    @  @    @  @    @  @    @  @    @                *
c         @    @    @   @@@@   @@@@@   @@@@@    @@@@                 *
c=====================================================================*
c
c   transition moment element calculation with respect to sdci bases
c
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      include 'tmomnt.h'
      include 'sdci00.h'
      include 'sdci01.h'
      common/ io    / in,iout,ifile(10)
c
c
      write(iout,2000)
c
c
      tmbx( 1 ) = t1x
      tmby( 1 ) = t1y
      tmbz( 1 ) = t1z

c cut for save memory ***************************************************
c$ohtawara test output trantition memonet element
c      write(iout,*) '=== tmsdbs.f ==== idebug: ',idebug
c      if (idebug.eq.-1) call trout
c cut for save memory ***************************************************

c
      call tmbs0
      call tmbss
c========( Mon Oct 16 11:48:50 JST 1995 )======
c$$$      ii = 1
c$$$      if( ii .eq. 1 ) go to 9000
c========( Mon Oct 16 11:48:50 JST 1995 )======
c
      call tmbl0
      call tmbls
      call tmbl1
c
      if( isdci0 .eq. 1 ) go to 9000
c
      if( isdci0 .eq. 2 ) then
```

```
        momgs = momega( 3 ) + 1
        momge = momega( 5 )
        call tmb30( momgs,  momge )
        mtaus = momega( 1 ) + 1
        mtaue = momega( 2 )
        call tmb3s( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 2 ) + 1
        mtaue = momega( 3 )
        call tmb31( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 3 ) + 1
        call tmb33( momgs,  momge,  mtaus )
      elseif( isdci0 .eq. 3 ) then
        call tmb20
        call tmb2s
        call tmb21
        call tmb22
        momgs = momega( 4 ) + 1
        momge = momega( 5 )
        call tmb30( momgs,  momge )
        mtaus = momega( 1 ) + 1
        mtaue = momega( 2 )
        call tmb3s( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 2 ) + 1
        mtaue = momega( 3 )
        call tmb31( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 3 ) + 1
        mtaue = momega( 4 )
        call tmb32( momgs,  momge,  mtaus,  mtaue )
        mtaus = momega( 4 ) + 1
        call tmb33( momgs,  momge,  mtaus  )
        call tmb40
        call tmb4s
        call tmb41
        call tmb42
        call tmb43
        call tmb44
        call tmb50
        call tmb5s
        call tmb51
        call tmb52
        call tmb53
        call tmb54
        call tmb55
      else
        write(iout,'(('' **** bad choice of ci bases'')')
        stop
      endif
c
 9000 continue
c
c
      call tmdump
c
c     all done. return to caller.
      call clockm('tmsdbs ')
c
 2000 format(1h ,'<<<<<< enter tmsdbs >>>>>>')
c
c
      return
      end
      subroutine tmsdst
c=====================================================================*
c                                                                     *
c       @@@@@  @    @   @@@@   @@@@@    @@@@    @@@@@                 *
c         @    @@  @@  @       @    @  @           @                 *
c         @    @ @@ @   @@@@   @    @   @@@@       @                 *
c         @    @    @       @  @    @       @      @                 *
c         @    @    @  @    @  @    @  @    @      @                 *
c         @    @    @   @@@@   @@@@@    @@@@       @                 *
c=====================================================================*
c
c  transform trans. moment elements to sdci state representation
c
      implicit real*8 (a-h,o-z)
c$ohtawara
      character*8 systime

      include 'MSSIZE'
      include 'sdci00.h'
      include 'tmomnt.h'

      common /pconst/ pi,toang,autoev,planck,slight,evtoer,pcon(14)
```

```
      common/ work01 / civec( mcics2 ),
     $      tdx( mcipck ), tdy( mcipck ), tdz( mcipck ),
     $      dummy1(1)
c$ohtawara
      common / io / in, iout, ifile(10)

c cut for low mem ******************************************************
c========== Tue Jan 23 17:30:46 JST 1996 =========
c$ohtawara
c  checking routine
c$ohtawara for my trmom calc.
c      common /mywork01/ aci(mcicsf,mcicsf)
c      dimension atx(msdcib,msdcib),aty(msdcib,msdcib),atz(msdcib,msdcib)
c      dimension trmx(msdcib,msdcib),trmy(msdcib,msdcib),trmz(msdcib,msdcib)
c cut for low mem ******************************************************

c$ohtawara
c jump my routine
      go to 999

c cut for low mem ******************************************************
c
c      do m1 = 1,ncibas
c         do m2 = 1,ncibas
c            m3  = max0( m1, m2 )
c            m4  = min0( m1, m2 )
c            m5  = m3*( m3 - 1 )/2 + m4
c            atx(m1,m2) = tmbx( m5 )
c            aty(m1,m2) = tmby( m5 )
c            atz(m1,m2) = tmbz( m5 )
c 200        format(2i3,2x,f9.4,2x,3f9.4)
c         enddo
c      enddo
c
c for small print out (given by sdci00.h)
c      if (iprint_yes0 .lt. 0)  go to 300
c      write(iout,*) 'atx,aty,atz'
c      write(iout,*) '*** x ***'
c      call matout(ncibas,atx)
c      write(iout,*) '*** y ***'
c      call matout(ncibas,aty)
c      write(iout,*) '*** z ***'
c      call matout(ncibas,atz)
c 300  continue
c
cc      write(iout,*) 'autodb = ',autodb
c      autodb = slight*toang*evtoer
c
c$ time
c      call time(systime)
c      write(6,*) 'my routine start  ',systime
c
c      do m1 = 1,ncibas
c         do m2 = 1,ncibas
c            trmx(m1,m2) = 0.0d0
c            trmy(m1,m2) = 0.0d0
c            trmz(m1,m2) = 0.0d0
c            do m3 = 1,ncibas
c               do m4 = 1,ncibas
c                  partx = aci(m1,m3)*aci(m2,m4)*atx(m3,m4)*autodb
c                  party = aci(m1,m3)*aci(m2,m4)*aty(m3,m4)*autodb
c                  partz = aci(m1,m3)*aci(m2,m4)*atz(m3,m4)*autodb
c                  trmx(m1,m2) = trmx(m1,m2) + partx
c                  trmy(m1,m2) = trmy(m1,m2) + party
c                  trmz(m1,m2) = trmz(m1,m2) + partz
c               enddo
c            enddo
c         enddo
c
cc monitor large loop
c         if( mod(m1,10) .eq. 0 ) then
c            call time(systime)
c            write(6,100) ncibas,m1,systime
c 100        format(2i6,' : ',a8)
c         endif
c
c      enddo
c
cc$ time
cc      call time(systime)
cc      write(6,*) 'my routine finish ',systime
c
cc for small print out (given by sdci00.h)
```

```
c       if (iprint_yes0 .lt. 0)  go to 400
c       write(iout,*) 'my trmom'
c       write(iout,*) '*** X ***'
c       call matout(ncibas,trmx)
c       write(iout,*) '*** Y ***'
c       call matout(ncibas,trmy)
c       write(iout,*) '*** Z ***'
c       call matout(ncibas,trmz)
c 400   continue
c
cc       do m1 = 1,ncibas
cc        do m2 = 1,ncibas
cc         write(iout,210) m1,m2,trmx(m1,m2),trmy(m1,m2),trmz(m1,m2)
cc 210     format(2i3,2x,3f10.4)
cc        enddo
cc       enddo
c
c  end of checking routine
c========== Tue Jan 23 17:30:46 JST 1996 ==========
c cut for low mem end ***********************************************************

c$ohtawara
 999  continue

c$ohtawara
c next loop : large cost

c$ time
c       call time(systime)
c       write(6,*) 'org routine start  ',systime

c original routine  (1.5* slow ,but 0.5*memory)
      do 1 lamd1 = 1, ncibas
        do 2 lamd2 = 1, lamd1
          tx00 = 0.0d0
          ty00 = 0.0d0
          tz00 = 0.0d0
          do 3 nomg = 1, ncibas
            llomg = ( lamd1 - 1 )*ncibas + nomg
            tx01 = 0.0d0
            ty01 = 0.0d0
            tz01 = 0.0d0
            do 4 ntau = 1, ncibas
              l2tau = ( lamd2 - 1 )*ncibas + ntau
              not1  = max0( nomg, ntau )
              not2  = min0( nomg, ntau )
              n     = not1*( not1 - 1 )/2 + not2
              tx01 = tx01 + civec( l2tau )*tmbx( n )
              ty01 = ty01 + civec( l2tau )*tmby( n )
              tz01 = tz01 + civec( l2tau )*tmbz( n )
     4      continue
            tx00 = tx00 + civec( llomg )*tx01
            ty00 = ty00 + civec( llomg )*ty01
            tz00 = tz00 + civec( llomg )*tz01
     3    continue
          1000 = lamd1*( lamd1 - 1 )/2 + lamd2
          tmsx( 1000 ) = tx00
          tmsy( 1000 ) = ty00
          tmsz( 1000 ) = tz00
     2  continue

c$ohtawara monitor large loop
        if( mod(lamd1,100) .eq. 0 ) then
          call time(systime)
          write(6,110) ncibas,lamd1,systime
 110      format(2i6,' : ',a8)
        endif

     1 continue

c$ time
c       call time(systime)
c       write(6,*) 'org routine finish ',systime

c$ cut by ohtawara
c       do 5 lamd1 = 1, ncibas
c         do 6 lamd2 = 1, lamd1
c           1000 = lamd1*( lamd1 - 1 )/2 + lamd2
c           tx00 = tmsx( 1000 )
c           ty00 = tmsy( 1000 )
c           tz00 = tmsz( 1000 )
c           tmbx( 1000 ) = tx00*tx00 + ty00*ty00 + tz00*tz00
c     6   continue
c     5 continue

 990  continue

c========( Fri Oct 13 12:40:52 JST 1995 )======
c$$$       write(iout,2000)
c$$$       write(iout,2010)
c$$$       write(iout,2020)
c$$$       write(iout,2010)
c$$$       lamd2 = 1
c$$$       do 191 lamd1 = 1, ncibas
c$$$c$$$          total=dsqrt(tmomsq(i))
c$$$         i  = lamd1*( lamd1 - 1 )/2 + lamd2
c$$$         xx = -tmsx( i )
c$$$         yy = -tmsy( i )
c$$$         zz = -tmsz( i )
c$$$         write(iout,2030) lamd1, xx, yy, zz
c$$$c$$$          write(iout,2030) i, xx, yy, zz, total
c$$$  191 continue
c$$$       write(iout,2010)
c$$$ 2000 Format(1H ,28('=')/
c$$$      &        1H ,'<G|-er|E(SCI)> (Atomic unit)'/
c$$$      &        1H ,28('='))
c$$$ 2010 Format(1H ,79('-'))
c$$$ 2020 Format(1H ,3X,'State',13X,'X',15X,'Y',15X,'Z',13X,'Total')
c$$$ 2030 Format(1H ,Iiout,6X,4(2X,F14.9))
c========( Fri Oct 13 12:40:52 JST 1995 )======


      autodb = slight*toang*evtoer

c$ohtawara
c for small print out (given by sdci00.h)
c       write(6,*) iprint_yes0

      if (iprint_yes0 .le. 0)  go to 998

      write(iout,'('' ** number of CI bases : '',i6/)') ncibas
      write(iout,'('' ** transition moments '')')
      write(iout,'('' *** Mx (Debye) ***'')')
      call fckout( tmsx, ncibas, autodb )
      write(iout,*)' '
      write(iout,'('' *** My (Debye) ***'')')
      call fckout( tmsy, ncibas, autodb )
      write(iout,*)' '
      write(iout,'('' *** Mz (Debye) ***'')')
      call fckout( tmsz, ncibas, autodb )

 998  continue

c$ cut by ohtawara
c       write(iout,*)' '
c       write(iout,'('' &&& Mx**2 + My**2 + Mz**2 (Debye**2) &&&'')')
c       call fckout( tmbx, ncibas, autodb*autodb )

c$$$       ii = 1
c$$$       if( ii .eq. 1 ) stop
c$$$       call dump00


c$ohtawara output fort.10 for gamma calc.
c out put fort.10 for old style(g93)
c       call outfile

c out put fort.20 for new style
      call outfile20

c out put fort.21 (unformatted file)
c       call outfile21


      return
      end

c=====================================================================
c@ohtawara        Mon Feb  5 14:00:25 JST 1996
      subroutine matout(num,x)
      implicit real*8 (a-h,o-z)
      include 'MSSIZE'
      dimension x(msdcib,msdcib)

c number of column
c       icol = 11
```

```
      icol = 6

      minc= 1
      maxc= num
      ilow = minc
      iup  = minc + icol-1
   10 if(iup.gt.maxc) then
         iup = maxc
      endif
      write(iout,100) (i,i=ilow,iup)
      do i=1,num
        write(iout,110) i,(x(i,j),j=ilow,iup)
      enddo
 100  format(1h ,11(8x,i3))
 110  format(1h ,i3,11(1x,f10.6))
      if(iup.eq.maxc) return
      ilow = ilow + icol
      iup  = iup  + icol
      go to 10

      end

c==========================================================================
c@ohtawara  Tue Feb  6 17:57:47 JST 1996
c out put energy of CI states and transition moments between CI states
      subroutine outfile
      implicit real*8 (a-h,o-z)
      character*80 title
      include 'MSSIZE'
      include 'sdci00.h'
      include 'tmomnt.h'
      include 'work02.h'
      common /pconst/ pi,toang,autoev,planck,slight,evtoer,pcon(14)
      common /my_title/ title
      dimension aeig(mcicsf)

      open(10,file='fort.10',form='formatted',status='unknown')

      autodb = slight*toang*evtoer
      aeig(1) = 0.0d0
      do i=2,ncibas
         aeig(i) = ( cieig(i)-cieig(1) ) * autoev
      enddo

      write(10,*) title
      write(10,100) ncibas
      do i=1,ncibas
         write(10,110) i,aeig(i)
      enddo

      do i=1,ncibas
         do j=1,ncibas
            n1  = max0( i, j )
            n2  = min0( i, j )
            n   = n1 *( n1 - 1 )/2 + n2
            write(10,120) i,j,tmsx(n)*autodb,tmsy(n)*autodb,tmsz(n)*autodb
         enddo
      enddo

 100  format(5x,i5)
 110  format(i5,5x,f10.5)
 120  format(2i5,5x,3f10.5)

      return
      end

c==========================================================================
c@ohtawara  Tue Feb  6 17:57:47 JST 1996
c out put energy of CI states and transition moments between CI states
      subroutine outfile20
      implicit real*8 (a-h,o-z)
      character*80 title
      include 'MSSIZE'
      include 'sdci00.h'
      include 'tmomnt.h'
      include 'work02.h'
      common /pconst/ pi,toang,autoev,planck,slight,evtoer,pcon(14)
      common /my_title/ title
      dimension aeig(mcicsf)
c      dimension xmom(mcicsf,mcicsf),ymom(mcicsf,mcicsf),zmom(mcicsf,mcicsf)

      open(20,file='fort.20',form='formatted',status='unknown')
```



```
      autodb = slight*toang*evtoer
      aeig(1) = 0.0d0
      do i=2,ncibas
         aeig(i) = ( cieig(i)-cieig(1) ) * autoev
      enddo

c      do i=1,ncibas
c         do j=1,ncibas
c            n1  = max0( i, j )
c            n2  = min0( i, j )
c            n   = n1 *( n1 - 1 )/2 + n2
c            xmom(i,j) = tmsx(n)*autodb
c            ymom(i,j) = tmsy(n)*autodb
c            zmom(i,j) = tmsz(n)*autodb
c         enddo
c      enddo

      write(20,*) title
      write(20,90) ncibas
 90   format(5x,i5)
      do i=1,ncibas
         write(20,110) i,aeig(i)
      enddo
 110  format(1h ,i3,11(1x,f10.6))

      iw   = 10
      ilow = 1
      iup  = iw
 1010 continue
      iup  = min0( iup, ncibas )
      write(20,2000) ( i,i=ilow,iup )
      iatold = 0
      do i = ilow,ncibas
        nclms = i - ilow + 1
        if( nclms .gt. iw ) then
          iu = ilow + iw - 1
        else
          iu = i
        endif
        ma = i*( i - 1 )/2
        write(20,2010) i, ( autodb*tmsx(ma+j), j = ilow, iu )
      enddo
      if( iup .eq. ncibas ) go to 500
      ilow = ilow + iw
      iup  = iup  + iw
      go to 1010

 2000 format(1h ,4x,10(7x,i4))
 2010 format(1h ,i4,3x,10f11.6)

 500  continue
      ilow = 1
      iup  = iw
 1510 continue
      iup  = min0( iup, ncibas )
      write(20,2000) ( i,i=ilow,iup )
      iatold = 0
      do i = ilow,ncibas
        nclms = i - ilow + 1
        if( nclms .gt. iw ) then
          iu = ilow + iw - 1
        else
          iu = i
        endif
        ma = i*( i - 1 )/2
        write(20,2010) i, ( autodb*tmsy(ma+j), j = ilow, iu )
      enddo
      if( iup .eq. ncibas ) go to 510
      ilow = ilow + iw
      iup  = iup  + iw
      go to 1510

 510  continue
      ilow = 1
      iup  = iw
 1610 continue
      iup  = min0( iup, ncibas )
      write(20,2000) ( i,i=ilow,iup )
      iatold = 0
      do i = ilow,ncibas
        nclms = i - ilow + 1
        if( nclms .gt. iw ) then
          iu = ilow + iw - 1
```

```
        else
          iu = i
        endif
        ma = i*( i - 1 )/2
        write(20,2010) i, ( autodb*tmsz(ma+j), j = ilow, iu )
      enddo
      if( iup .eq. ncibas ) go to 520
      ilow = ilow + iw
      iup  = iup  + iw
      go to 1610

 520  return
      end


c number of column
c       icol = 11
c       icol = 6
c       num = ncibas
c       minc= 1
c       maxc= num
c
c       ilow = minc
c       iup  = minc + icol-1
c    10 if(iup.gt.maxc) then
c          iup = maxc
c       endif
c       write(11,100) (i,i=ilow,iup)
c       do i=1,num
c         write(11,110) i,(xmom(i,j),j=ilow,iup)
c       enddo
c 100   format(1h ,11(8x,i3))
c 110   format(1h ,i3,11(1x,f10.6))
c       if(iup.eq.maxc) go to 510
c       ilow = ilow + icol
c       iup  = iup  + icol
c       go to 10
c
c 510   continue
c       ilow = minc
c       iup  = minc + icol-1
c    20 if(iup.gt.maxc) then
c          iup = maxc
c       endif
c       write(11,100) (i,i=ilow,iup)
c       do i=1,num
c         write(11,110) i,(ymom(i,j),j=ilow,iup)
c       enddo
c       if(iup.eq.maxc) go to 520
c       ilow = ilow + icol
c       iup  = iup  + icol
c       go to 20
c
c 520   continue
c       ilow = minc
c       iup  = minc + icol-1
c    30 if(iup.gt.maxc) then
c          iup = maxc
c       endif
c       write(11,100) (i,i=ilow,iup)
c       do i=1,num
c         write(11,110) i,(zmom(i,j),j=ilow,iup)
c       enddo
c       if(iup.eq.maxc) go to 530
c       ilow = ilow + icol
c       iup  = iup  + icol
c       go to 30
c
c
c 530   return
c       end
c=====================================================================
c       subroutine outfile21
c       implicit real*8 (a-h,o-z)
c       character*80 title
c       include 'MSSIZE'
c       include 'sdci00.h'
c       include 'tmomnt.h'
c       include 'work02.h'
c       common /pconst/ pi,toang,autoev,planck,slight,evtoer,pcon(14)
c       common /my_title/ title
c       dimension aeig(mcicsf)
c
```

```
c       open(21,file='fort.21',form='unformatted',status='unknown')
c
c       autodb = slight*toang*evtoer
c       aeig(1) = 0.0d0
c
c       do i=2,ncibas
c          aeig(i) = ( cieig(i)-cieig(1) ) * autoev
c       enddo
c
c       write(21) title
c       write(21) ncibas
c
c       do i=1,ncibas
c          write(21) aeig(i)
c       enddo
c
c       do i = 1,ncibas
c          do j = 1,ncibas
c             m1 = max0( i, j )
c             m2 = min0( i, j )
c             m  = m1 * ( m1 - 1 ) / 2 + m2
c             write(21) autodb*tmsx( m ), autodb*tmsy( m ), autodb*tmsz( m )
c          enddo
c       enddo
c
c       return
c       end
c
C@(#)=TMOMCI(SUB)
      SUBROUTINE TMOMCI
C
C     GET DIPOLE AND TRANSITION MOMENTS BY USING SCI WAVEFUNCTIONS.
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER*4 A,B,R,S
      CHARACTER*1 CHR1(3)
cfj   INCLUDE (MSSIZE)
      INCLUDE 'MSSIZE'
      COMMON /IO     / IN,IOUT,IFILE(10)
      COMMON /IOPDIP/ IOPDIP(18)
      COMMON /MOL    / COORD(3,MAXAT),IAN(MAXAT),NATOMS,ICHARG,MULTIP,
     &                 NAE,NBE,NE,NBASIS
      COMMON /WINDOW/ IOMO(MCIOCC),IVMO(MCIVAC),LMO(MCIMOS),
     &                 NOCS,NVCS,NCSFS
      COMMON /LVCI   / LOCC(MCICSF),LVAC(MCICSF),
     &                 MOCC(MCICSF),MVAC(MCICSF)
      COMMON /DGRND / DXPT,DYPT,DZPT,DXAP,DYAP,DZAP,DX,DY,DZ
      COMMON /WORK01/ TGNX(MCICSF),TGNY(MCICSF),TGNZ(MCICSF),
     &                 TNNX(MCIINT),TNNY(MCIINT),TNNZ(MCIINT),
     &                 TMOMSQ(MCICSF),
     &                 XTMP(MCICSF),YTMP(MCICSF),ZTMP(MCICSF),DUMMY1(1)
      COMMON /WORK02/ CIEIG(MCICSF),CIVEC(MCICS2),
     &                 TDX(MCIPCK),TDY(MCIPCK),TDZ(MCIPCK),DUMMY2(1)
      DATA ZERO /0.0D+00/,TWO/2.0D+00/
      DATA CHR1/'X','Y','Z'/
 2000 Format(1H ,28('='))
     &       1H ,'<G/-er/E(SCI)> (Atomic unit)'/
     &       1H ,28('='))
 2010 Format(1H ,79('-'))
 2020 Format(1H ,3X,'State',13X,'X',15X,'Y',15X,'Z',13X,'Total')
 2030 Format(1H ,I6,6X,4(2X,F14.9))
 2040 Format(1H ,25('='))
     &       1H ,'<E(SCI)/',A1,'/E''(SCI)> (Bohr)'/
     &       1H ,25('='))
C     ****************************************************************
      IZDO =IOPDIP( 1)
      IPRTM=IOPDIP(12)
      NOCC =NAE
      ROOT2=DSQRT(TWO)
C
C     GET TRANSITION MOMENTS (G->E)
C     TRANSITION MOMENT X = -TGNX(=-<G/R/E>) ; Y = -TGNY ; Z = -TGNZ.
      DO 20 I=1,NCSFS
        IP=NCSFS*(I-1)
        TX=ZERO
        TY=ZERO
        TZ=ZERO
        DO 10 J=1,NCSFS
          IPJ=IP+J
          A  =MOCC(J)
          R  =MVAC(J)
          MA =R*(R-1)/2+A
          TX =TX+CIVEC(IPJ)*TDX(MA)
```

```
          TY =TY+CIVEC(IPJ)*TDY(MA)
          TZ =TZ+CIVEC(IPJ)*TDZ(MA)
   10   CONTINUE
        TGNX(I)   =TX*ROOT2
        TGNY(I)   =TY*ROOT2
        TGNZ(I)   =TZ*ROOT2
        TMOMSQ(I)=TGNX(I)*TGNX(I)+TGNY(I)*TGNY(I)+TGNZ(I)*TGNZ(I)
   20 CONTINUE
C
C     GET TRANSITION MOMENTS (E->E).
C     TRANSITION MOMENT X = -TNNX(=-<E/R/E>) ; Y = -TNNY ; Z = -TNNZ.
      NPCK=NCSFS*(NCSFS+1)/2
      DO 30 I=1,NPCK
        TNNX(I)=ZERO
        TNNY(I)=ZERO
        TNNZ(I)=ZERO
   30 CONTINUE
C
C     GET DIPOLE MOMENT FOR GROUND STATE USING FOR THE CALCULATION OF
C     HYPERPOLARIZABILITIES.
C     DIPOLE X = -DX, Y = -DY, Z = -DZ
      IF(IZDO.EQ.1) THEN
        DX=-DXPT
        DY=-DYPT
        DZ=-DZPT
      ELSE
        DX=-(DXPT+DXAP)
        DY=-(DYPT+DYAP)
        DZ=-(DZPT+DZAP)
      END IF
C
C     THEN CALCULATE ... 3 STEPS.
      DO 100 A=1,NOCS
        ITMP=NVCS*(A-1)
        DO 90 R=1,NVCS
C
          DO 40 J=1,NCSFS
            XTMP(J)=ZERO
            YTMP(J)=ZERO
            ZTMP(J)=ZERO
   40     CONTINUE
C
          DO 60 S=1,NVCS
            IF(R.GT.S) THEN
              MA=(R+NOCS)*(R-1+NOCS)/2+S+NOCS
            ELSE
              MA=(S+NOCS)*(S-1+NOCS)/2+R+NOCS
            END IF
            XM=TDX(MA)
            YM=TDY(MA)
            ZM=TDZ(MA)
            JP=ITMP+S
            DO 50 J=1,NCSFS
              JC=NCSFS*(J-1)+JP
              XTMP(J)=XTMP(J)+CIVEC(JC)*XM
              YTMP(J)=YTMP(J)+CIVEC(JC)*YM
              ZTMP(J)=ZTMP(J)+CIVEC(JC)*ZM
   50       CONTINUE
   60     CONTINUE
C
          N =0
          IP=ITMP+R
          DO 80 I=1,NCSFS
            IC=NCSFS*(I-1)+IP
            CC=CIVEC(IC)
            DO 70 J=1,I
              N=N+1
              TNNX(N)=TNNX(N)+XTMP(J)*CC
              TNNY(N)=TNNY(N)+YTMP(J)*CC
              TNNZ(N)=TNNZ(N)+ZTMP(J)*CC
   70       CONTINUE
   80     CONTINUE
C
   90   CONTINUE
  100 CONTINUE
C
      DO 170 R=1,NVCS
        DO 160 A=1,NOCS
C
          DO 110 J=1,NCSFS
            XTMP(J)=ZERO
            YTMP(J)=ZERO
            ZTMP(J)=ZERO
  110     CONTINUE
C
          DO 130 B=1,NOCS
            IF(A.GT.B) THEN
              MA=A*(A-1)/2+B
            ELSE
              MA=B*(B-1)/2+A
            END IF
            XM=TDX(MA)
            YM=TDY(MA)
            ZM=TDZ(MA)
            JP=NVCS*(B-1)+R
            DO 120 J=1,NCSFS
              JC=NCSFS*(J-1)+JP
              XTMP(J)=XTMP(J)+CIVEC(JC)*XM
              YTMP(J)=YTMP(J)+CIVEC(JC)*YM
              ZTMP(J)=ZTMP(J)+CIVEC(JC)*ZM
  120       CONTINUE
  130     CONTINUE
C
          N =0
          IP=NVCS*(A-1)+R
          DO 150 I=1,NCSFS
            IC=NCSFS*(I-1)+IP
            CC=CIVEC(IC)
            DO 140 J=1,I
              N=N+1
              TNNX(N)=TNNX(N)-XTMP(J)*CC
              TNNY(N)=TNNY(N)-YTMP(J)*CC
              TNNZ(N)=TNNZ(N)-ZTMP(J)*CC
  140       CONTINUE
  150     CONTINUE
C
  160   CONTINUE
  170 CONTINUE
C
      DO 180 I=1,NCSFS
        MA=I*(I+1)/2
        TNNX(MA)=TNNX(MA)+DX
        TNNY(MA)=TNNY(MA)+DY
        TNNZ(MA)=TNNZ(MA)+DZ
  180 CONTINUE
C
      IF(IPRTM.EQ.1 .OR. IPRTM.EQ.2) THEN
        WRITE(IOUT,2000)
        WRITE(IOUT,2010)
        WRITE(IOUT,2020)
        WRITE(IOUT,2010)
        DO 190 I=1,NCSFS
          TOTAL=DSQRT(TMOMSQ(I))
          XX=-TGNX(I)
          YY=-TGNY(I)
          ZZ=-TGNZ(I)
          WRITE(IOUT,2030) I,XX,YY,ZZ,TOTAL
  190   CONTINUE
        WRITE(IOUT,2010)
      END IF
      IF(IPRTM.EQ.2) THEN
        WRITE(IOUT,2040) CHR1(1)
        CALL MSMOUT(TNNX,NCSFS)
        WRITE(IOUT,2040) CHR1(2)
        CALL MSMOUT(TNNY,NCSFS)
        WRITE(IOUT,2040) CHR1(3)
        CALL MSMOUT(TNNZ,NCSFS)
      END IF
c========( Mon Oct 16 11:23:54 JST 1995 )======
        WRITE(IOUT,2000)
        WRITE(IOUT,2010)
        WRITE(IOUT,2020)
        WRITE(IOUT,2010)
        DO 191 I=1,NCSFS
          TOTAL=DSQRT(TMOMSQ(I))
          XX=-TGNX(I)
          YY=-TGNY(I)
          ZZ=-TGNZ(I)
          WRITE(IOUT,2030) I,XX,YY,ZZ,TOTAL
  191   CONTINUE
        WRITE(IOUT,2010)
        WRITE(IOUT,2040) CHR1(1)
        CALL MSMOUT(TNNX,NCSFS)
        WRITE(IOUT,2040) CHR1(2)
        CALL MSMOUT(TNNY,NCSFS)
        WRITE(IOUT,2040) CHR1(3)
```

```
        CALL MSMOUT(TNNZ,NCSFS)
c$$$      ii = 1
c$$$      if( ii .eq. 1 ) stop
c========( Mon Oct 16 11:23:54 JST 1995 )======
C
      RETURN
      END
      subroutine tmsdci
c=====================================================================*
c                                                                     *
c       @@@@@  @    @   @@@@   @@@@@    @@@@      @                  *
c         @    @@  @@  @       @    @  @    @     @                  *
c         @    @ @@ @   @@@@   @    @  @          @                  *
c         @    @    @       @  @    @  @          @                  *
c         @    @    @  @    @  @    @  @    @     @                  *
c         @    @    @   @@@@   @@@@@    @@@@      @                  *
c=====================================================================*
c
c     get transition moments using sdci wavefunctions.
c
      implicit real*8 (a-h,o-z)
c========( Thu Oct 12 15:46:13 JST 1995 )======
      include 'MSSIZE'
      include 'basinf.h'
      include 'mol.h'
      include 'tmomnt.h'
c========( Thu Oct 12 15:46:13 JST 1995 )======
      include 'dgrnd.h'
c
      common /io    / in,iout,ifile(10)
      common /iopdip/ iopdip(18)
      common /multci/ multci
c
c
      izdo = iopdip( 1 )
c
      write(iout,2000)
      write(iout,2010)
c
c     now, common /work01/ dummy1(mciint*2),civec(mcicsf,mcicsf),
c                          dummy2(1)
c     civec ... eigenvectors of ci hamiltonian matrix.
c
c     move eigenvectors of ci hamiltonian matrix and pack them.
      call packci
c
c     now, common /work01/ civec(mcicsf**2),
c                          dummy1(1)
c     civec ... eigenvectors of ci hamiltonian matrix (packed form).
c
      if( multci .eq. 3 ) then
        write(iout,2020)
        go to 9999
      end if
c
c     get molecular integrals for dipole moments, tdx, tdy and tdz.
      if( izdo .eq. 1 ) then
        write(iout,2030)
        call dipzdo
      else
        call diplcn
      endif
c
c     now, common /work01/ civec(mcicsf**2),
c                          tdx(mcipck),tdy(mcipck),tdz(mcipck)
c          common /work02/ c(maxb2),eig(maxbas),cieig(mcicsf)
c
c     civec ... eigenvectors of ci hamiltonian matrix (packed form).
c     tdx, tdy and tdz ... molecular integrals for dipole moments.
c     c ... molecular orbital coefficients.
c     eig ... orbital energies.
c     cieig ... eigenvalues of ci hamiltonian matrix.
c
c
c$$$  1. get lx, ly & lz.                        (lzdo00/llcn00)
c$$$  2. get transition moments using sdci bases.      (tmsdbs)
c$$$  3. transition moments using sdci wavefunctions. (tmsdst)
c$$$      if( izdo .eq. 1 ) then
c$$$        call lzdo00
c$$$      else
c$$$        call llcn00
c$$$      endif
      if( izdo .eq. 1 ) then
        dx = -dxpt
        dy = -dypt
        dz = -dzpt
      else
        dx = -( dxpt + dxap )
        dy = -( dypt + dyap )
        dz = -( dzpt + dzap )
      end if
      tlx = dx
      tly = dy
      tlz = dz
c
      call tmsdbs
      call tmsdst
c
c
 9999 continue
c     all done. return to caller.
      call clockm('tmsdci.  ')
c
 2000 format(1h ,'<<<<<< enter tmsdci >>>>>>')
 2010 format(1h ,'computation of electronic transition moments.')
 2020 format(1h ,'transition mements not calculated because triplet ',
     &           'ci calculation was performed.')
 2030 format(1h ,'zdo approximation is used in estimating 1 center ',
     &           'dipole integrals.')
c
c
      return
      end
c$$$#endif
c$$$#ifdef FUJITSU
C
c$$$C@(#)=PSTCAL(SUB) ... Fujitsu FORTRAN
c$$$      SUBROUTINE PSTCAL
c$$$C
c$$$C     JOB IS TERMINATED IN THIS ROUTINE.
c$$$C     USE Fujitsu SYSTEM ROUTINE.
c$$$C     93/Mar A.M.
c$$$C     ***************************************************************
c$$$      CHARACTER*8 TODAY
c$$$      Integer TimeF
c$$$      Logical Anchor
c$$$      COMMON /IO    / IN,IOUT,IFILE(10)
c$$$      COMMON /ELAPSE/ ISTART,IARRAY(3)
c$$$      Common /Anchor/ Anchor
c$$$C2000 FORMAT(1H ,59('*'))
c$$$ 2010 FORMAT(1H ,'   >>> MOS-F ... Normal end at ',I2.2,':',I2.2,':',
c$$$     &       I2.2,' on ',A8,'.')
c$$$ 2020 FORMAT(1H ,'        CPU time       ...',
c$$$     &       I4.2,' Hr ',I2.2,' Min ',I2.2,' Sec.')
c$$$ 2025 Format(1h ,'        VU-Time        ...',
c$$$     &       I4.2,' Hr ',I2.2,' Min ',I2.2,' Sec.')
c$$$ 2030 FORMAT(1H ,'        Elapsed time  ...',
c$$$     &       I4.2,' Hr ',I2.2,' Min ',I2.2,' Sec.')
c$$$ 2035 Format(1h ,'        VU/CPU        ... ',F7.2,' %.')
c$$$ 2036 Format(1h ,'        Elapsed/CPU   ... ',F7.2)
c$$$ 2040 FORMAT(1H ,'Job terminated normally in routine PstCal.')
c$$$C     ***************************************************************m
c$$$C     WRITE(IOUT,2000)
c$$$C
c$$$      CALL DATE(TODAY)
c$$$      CALL TIME(IEND)
c$$$      IEND=IEND/1000
c$$$      CALL TOHMS(IEND,IHR,IMIN,ISEC)
c$$$      WRITE(IOUT,2010) IHR,IMIN,ISEC,TODAY
c$$$C
c$$$c     CALL CLOCK(TIME1,0,1)
c$$$      Call ClockV(VU1,Time1,0,1)
c$$$      ICPU=NINT(TIME1)
c$$$      CALL TOHMS(ICPU,IHR,IMIN,ISEC)
c$$$      WRITE(IOUT,2020) IHR,IMIN,ISEC
c$$$c
c$$$      ICPU=NINT(VU1)
c$$$      CALL TOHMS(ICPU,IHR,IMIN,ISEC)
c$$$      WRITE(IOUT,2025) IHR,IMIN,ISEC
c$$$      VRate=(VU1/Time1)*100.E0
c$$$C
c$$$C     IELAPS=IEND-ISTART
c$$$      IElaps=TimeF()-IStart
c$$$      CALL TOHMS(IELAPS,IHR,IMIN,ISEC)
c$$$      WRITE(IOUT,2030) IHR,IMIN,ISEC
c$$$c
```

```
c$$$      EPC=Float(IElaps)/Time1
c$$$      Write(IOut,2035) VRate
c$$$      Write(IOut,2036) EPC
c$$$C
c$$$C     WRITE(IOUT,2000)
c$$$      WRITE(IOUT,2040)
c$$$C
c$$$      If (Anchor) Then
c$$$        IUnit=IFile(10)
c$$$        Write(IUnit,*) ' '
c$$$        Close(Unit=IUnit)
c$$$      EndIf
c$$$C
c$$$      STOP
c$$$      END
c$$$#endif
C
C@(#)=TOHMS(SUB)
      SUBROUTINE TOHMS(NSEC,IHR,IMIN,ISEC)
C
C     ***************************************************************
      IHR =NSEC /3600
      IMIN=NSEC /60  -IHR *60
      ISEC=NSEC      -IMIN*60-IHR*3600
C
      RETURN
      END
C
C@(#)=TOSEC(SUB)
      SUBROUTINE TOSEC(IARRAY,NSEC)
C
C     ***************************************************************
      DIMENSION IARRAY(*)
C     ***************************************************************
      NSEC=IARRAY(1)*3600+IARRAY(2)*60+IARRAY(3)
C
      RETURN
      END
c=>   92/Aug A.M.   ...   New Routine is appended.
C
c@(#)=ToUpr(Sub)
      Subroutine ToUpr(String, LenStr)
c
c     Convert lower case into uper case.
c     On input:
c      String ... String containing lower cases.
c      LenStr ... Length of string.
c     On output:
c      String ... String not containing lower case.
c
c     ***********************************************************************
      Implicit Integer (a-z)
      Character*(*) String
      Character*1 LwCase(26), UpCase(26)
c
      Data LwCase /
     &            1ha, 1hb, 1hc, 1hd, 1he, 1hf, 1hg,
     &            1hh, 1hi, 1hj, 1hk, 1hl, 1hm, 1hn,
     &            1ho, 1hp, 1hq, 1hr, 1hs, 1ht, 1hu,
     &            1hv, 1hw, 1hx, 1hy, 1hz/
      Data UpCase /
     &            1hA, 1hB, 1hC, 1hD, 1hE, 1hF, 1hG,
     &            1hH, 1hI, 1hJ, 1hK, 1hL, 1hM, 1hN,
     &            1hO, 1hP, 1hQ, 1hR, 1hS, 1hT, 1hU,
     &            1hV, 1hW, 1hX, 1hY, 1hZ/
      Data NAlp /26/
c     ***********************************************************************
      If (LenStr .LE. 0) Stop 'LenStr .LE. 0. Abort in ToUpr'
c
      Do 20 i = 1, LenStr
        Do 10 j = 1, NAlp
          If (String(i:i) .Eq. LwCase(j)) Then
            String(i:i) =  UpCase(j)
            Go To 20
          End If
   10   Continue
   20 Continue
c
      Return
      End
C
C@(#)=XYZGEO(SUB) ... MOPAC(XYZGEO)
      SUBROUTINE XYZGEO(COORD,ZCOORD,DEGREE,NA,NB,NC,NATOMS,PI)
C
C     ==========================================================
C     MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC
C     ==========================================================
C
C     XYZGEO CONVERTS COORDINATES FROM CARTESIAN TO INTERNAL.
C     NOTE.
C      THIS ROUTINE IS THE SAME AS XYZGEO OF MOPAC.
C     ***************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION COORD(3,*),ZCOORD(3,*),NA(*),NB(*),NC(*)
      DATA FIVE/5.0D+00/,F15/1.5D+01/,F180/1.8D+02/
C     ***************************************************************
      DO 30 I=2,NATOMS
        J=NA(I)
        K=NB(I)
        L=NC(I)
        IF(I.LT.3) GO TO 30
        II=I
        CALL BANGLE(COORD,II,J,K,ZCOORD(2,I))
        ZCOORD(2,I)=ZCOORD(2,I)*DEGREE
        IF(I.LT.4) GO TO 30
C
C     MAKE SURE DIHEDRAL IS MEANINGFUL
        CALL BANGLE(COORD,J,K,L,ANGL)
C
C     TOL = 0.2617994 RAD (15 DEGREES)
C*MOS   TOL=0.2617994D0
C*MOS   IF(ANGL.GT.3.1415926D0-TOL.OR.ANGL.LT.TOL) THEN
        TOL=PI*F15/F180
        IF(ANGL.GT.(PI-TOL) .OR. ANGL.LT.TOL) THEN
C
C     ANGLE IS UNSATISFACTORY, LET'S SEARCH FOR ANOTHER ATOM FOR
C     DEFINING THE DIHEDRAL.
   10     SUM=100.D0
          DO 20 I1=1,II-1
            R=(COORD(1,I1)-COORD(1,K))**2+
     &        (COORD(2,I1)-COORD(2,K))**2+
     &        (COORD(3,I1)-COORD(3,K))**2
            IF(R.LT.SUM.AND.I1.NE.J.AND.I1.NE.K) THEN
              CALL BANGLE(COORD,J,K,I1,ANGL)
C*MOS         IF(ANGL.LT.3.1415926D0-TOL.OR.ANGL.GT.TOL) THEN
              IF(ANGL.LT.(PI-TOL) .OR. ANGL.GT.TOL) THEN
                SUM=R
                L=I1
                NC(II)=L
              END IF
            END IF
   20     CONTINUE
          IF(SUM.GT.99.D0.AND.TOL.GT.0.1D0) THEN
C
C     ANYTHING WITHIN 5 DEGREES?
C*MOS         TOL=0.087266D0
              TOL=PI*FIVE/F180
              GO TO 10
          END IF
        END IF
        CALL DIHED(COORD,II,J,K,L,ZCOORD(3,I),PI)
        ZCOORD(3,I)=ZCOORD(3,I)*DEGREE
   30 ZCOORD(1,I)=DSQRT((COORD(1,I)-COORD(1,J))**2+
     &                  (COORD(2,I)-COORD(2,J))**2+
     &                  (COORD(3,I)-COORD(3,J))**2)
C
      ZCOORD(1,1)=0.D0
      ZCOORD(2,1)=0.D0
      ZCOORD(3,1)=0.D0
      ZCOORD(2,2)=0.D0
      ZCOORD(3,2)=0.D0
      ZCOORD(3,3)=0.D0
C
      RETURN
      END
C
C@(#)=ZTOC(SUB) ... MOPAC(GMETRY)
      SUBROUTINE ZTOC(ZCOORD,COORD,TORAD,NA,NB,NC,NATDUM)
C
C     ==========================================================
C     MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC MOPAC
C     ==========================================================
C
C     CONVERT INTERNAL COORDINATES INTO CARTESIAN COORDINATES.
C     NOTE.
C      THIS ROUTINE IS THE SAME AS GMETRY OF MOPAC.
```

```
C     ****************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /IO    / IN,IOUT,IFILE(10)
      DIMENSION COORD(3,*),ZCOORD(3,*),NA(*),NB(*),NC(*)
 9910 Format(1H ,75('X')/
     &       1H ,'CARTESIAN COORDINATE OF ATOM ',I3,' COULD NOT BE ',
     &           'DETERMINED SINCE ASSOCIATED'/
     &       1H ,'ATOMS ',I3,',',I3,' AND ',I3,' WERE ALMOST LINEAR ',
     &           'BUT NOT PERFECT LINEAR. CHECK YOUR'/
     &       1H ,'INPUT DATA.'/
     &       1H ,75('X'))
C     ****************************************************************
      COORD(1,1)=0.0D00
      COORD(2,1)=0.0D00
      COORD(3,1)=0.0D00
      COORD(1,2)=ZCOORD(1,2)
      COORD(2,2)=0.0D00
      COORD(3,2)=0.0D00
C
      IF(NATDUM.EQ.2) THEN
        RETURN
      END IF
C
      CCOS=DCOS(ZCOORD(2,3)*TORAD)
      IF(NA(3).EQ.1)THEN
        COORD(1,3)=COORD(1,1)+ZCOORD(1,3)*CCOS
      ELSE
        COORD(1,3)=COORD(1,2)-ZCOORD(1,3)*CCOS
      END IF
      COORD(2,3)=ZCOORD(1,3)*DSIN(ZCOORD(2,3)*TORAD)
      COORD(3,3)=0.0D00
C
      IF(NATDUM.EQ.3) THEN
        RETURN
      END IF
C
      DO 100 I=4,NATDUM
        COSA=DCOS(ZCOORD(2,I)*TORAD)
        MB=NB(I)
        MC=NA(I)
        XB=COORD(1,MB)-COORD(1,MC)
        YB=COORD(2,MB)-COORD(2,MC)
        ZB=COORD(3,MB)-COORD(3,MC)
        RBC=1.0D00/DSQRT(XB*XB+YB*YB+ZB*ZB)
        IF (DABS(COSA).LT.0.99999999991D00) GO TO 40
C
C     ATOMS MC, MB, AND (I) ARE COLLINEAR
        RBC=ZCOORD(1,I)*RBC*COSA
        COORD(1,I)=COORD(1,MC)+XB*RBC
        COORD(2,I)=COORD(2,MC)+YB*RBC
        COORD(3,I)=COORD(3,MC)+ZB*RBC
        GO TO 100
C
C     THE ATOMS ARE NOT COLLINEAR
   40   MA=NC(I)
        XA=COORD(1,MA)-COORD(1,MC)
        YA=COORD(2,MA)-COORD(2,MC)
        ZA=COORD(3,MA)-COORD(3,MC)
C
C     ROTATE ABOUT THE Z-AXIS TO MAKE YB=0, AND XB POSITIVE.  IF XYB IS
C     TOO SMALL, FIRST ROTATE THE Y-AXIS BY 90 DEGREES.
        XYB=DSQRT(XB*XB+YB*YB)
        K=-1
        IF (XYB.GT.0.1D00) GO TO 50
        XPA=ZA
        ZA=-XA
        XA=XPA
        XPB=ZB
        ZB=-XB
        XB=XPB
        XYB=DSQRT(XB*XB+YB*YB)
        K=+1
C
C     ROTATE ABOUT THE Y-AXIS TO MAKE ZB VANISH
   50   COSTH=XB/XYB
        SINTH=YB/XYB
        XPA=XA*COSTH+YA*SINTH
        YPA=YA*COSTH-XA*SINTH
        SINPH=ZB*RBC
        COSPH=DSQRT(ABS(1.D00-SINPH*SINPH))
        XQA=XPA*COSPH+ZA*SINPH
        ZQA=ZA*COSPH-XPA*SINPH
C
C     ROTATE ABOUT THE X-AXIS TO MAKE ZA=0, AND YA POSITIVE.
        YZA=DSQRT(YPA**2+ZQA**2)
        IF(YZA.LT.2.D-2 ) THEN
          IF(YZA.LT.1.D-4) GO TO 70
          WRITE(IOUT,9910) I,MC,MB,MA
          CALL MERROR
        END IF
        COSKH=YPA/YZA
        SINKH=ZQA/YZA
        GO TO 80
   70   CONTINUE
C
C     ANGLE TOO SMALL TO BE IMPORTANT
        COSKH=1.D0
        SINKH=0.D0
   80   CONTINUE
C
C     COORDINATES :-   A=(XQA,YZA,0),   B=(RBC,0,0),  C=(0,0,0)
C     NONE ARE NEGATIVE.
C     THE COORDINATES OF I ARE EVALUATED IN THE NEW FRAME.
        SINA=DSIN(ZCOORD(2,I)*TORAD)
        SIND=-DSIN(ZCOORD(3,I)*TORAD)
        COSD=DCOS(ZCOORD(3,I)*TORAD)
        XD=ZCOORD(1,I)*COSA
        YD=ZCOORD(1,I)*SINA*COSD
        ZD=ZCOORD(1,I)*SINA*SIND
C
C     TRANSFORM THE COORDINATES BACK TO THE ORIGINAL SYSTEM.
        YPD=YD*COSKH-ZD*SINKH
        ZPD=ZD*COSKH+YD*SINKH
        XPD=XD*COSPH-ZPD*SINPH
        ZQD=ZPD*COSPH+XD*SINPH
        XQD=XPD*COSTH-YPD*SINTH
        YQD=YPD*COSTH+XPD*SINTH
        IF (K.LT.1) GO TO 90
        XRD=-ZQD
        ZQD=XQD
        XQD=XRD
   90   COORD(1,I)=XQD+COORD(1,MC)
        COORD(2,I)=YQD+COORD(2,MC)
        COORD(3,I)=ZQD+COORD(3,MC)
  100 CONTINUE
C
      RETURN
      END
/*
 *@(#)=cuts_(c function)
 *
 *      Get current OS name, node name, and so on.
 *
 *      On input:
 *       None.
 *      On output:
 *       system_name  ... current OS name.
 *       node_ name   ... current node name.
 *       release_no   ... current release number of the OS.
 *       machine_name ... current machine name.
 *
 */

#include <stdio.h>
#include <string.h>
#include <sys/utsname.h>

void cuts_ (system_name, node_name, release_no, machine_name)
char *system_name, *node_name, *release_no, *machine_name;
{

        struct utsname name;

        if (uname(&name) != -1) {
                strcpy (system_name, name.sysname);
                strcpy (node_name, name.nodename);
                strcpy (release_no, name.release);
                strcpy (machine_name, name.machine);
        }
        else {
                printf (" Abort in cuts_. Return code of uname = -1\n");
                exit (0);
        }

}
      subroutine trout
```

```
c=====================================================================*
c TROUT by K. Ohtawara
c last update: Mon Jan 22 17:18:41 JST 1996
c subroutines
c   chk2s  chk21  chk22
c   chk3s  chk31          chk33
c   chk4s          chk42  chk43  chk44
c   chk5s                                chk55
c
c  transition moment check program
c  interaction w/ singly excited states
c   +s0 +ss
c  interaction w/ doubly excited states
c   @10 +1s +11
c   @20  2s  21  22
c   @30 *3s  31 @32  33
c   @40  4s @41  42  43 *44
c   @50  5s @51 @52 @53 @54 *55
c   @ : = zero
c   + : OK(not change about i,j,k,l,r,s,t,u)
c   = : OK by debug
c   * : have bug
c=====================================================================*
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)
      common / io / in, iout, ifile(10)

      dimension ax(10,10),ay(10,10),az(10,10)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      if(idebug.ne.-1) go to 999
      do i = 1,4
         do j = 1,4
            kp  = max0( i,j )
            kpx = min0( i,j )
            mu  = kp*( kp - 1 )/2 + kpx
            write(iout,100) i,j,mu,tdx(mu),tdy(mu),tdz(mu)
 100        format(' m_ij',2i3,i5,3f10.5)
            ax(i,j) = tdx( mu )
            ay(i,j) = tdy( mu )
            az(i,j) = tdz( mu )
         enddo
      enddo

c=====================================================================
c     Control the checking program
c      -> Uncomment call statements you want to check
c=====================================================================

      call chk2s3s(2)
      call chk2s3s(3)
      call chk4s5s(4)
      call chk4s5s(5)
      call chk2131(21)
      call chk2131(31)
      call chk2233(22)
      call chk2233(33)
      call chk4243(42)
      call chk4243(43)
      call chk4455(44)
      call chk4455(55)

 999  return
      end
c=====================================================================
      subroutine chk2s3s(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)
      dimension ax(10,10),ay(10,10),az(10,10)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,200) nums
 200  format('========== checking ',i1,'s ==========')
       j = 0
       l = 0
      if(nums.eq.2) iu = 0
      if(nums.eq.3) is = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      continue
         if(nums.eq.2) then
            ir = 1
            is = 2
            it = 3
            call cp2s
         else
            ir = 1
            it = 3
            iu = 4
            call cp3s
         endif
         if(nums.eq.2) then
            ir = 2
            is = 1
            it = 3
            call cp2s
         else
            ir = 1
            it = 4
            iu = 3
            call cp3s
         endif
         if(nums.eq.2) then
            ir = 1
            is = 2
            it = 4
            call cp2s
         else
            ir = 2
            it = 3
            iu = 4
            call cp3s
         endif
         if(nums.eq.2) then
            ir = 2
            is = 1
            it = 4
            call cp2s
         else
            ir = 2
            it = 4
            iu = 3
            call cp3s
         endif
      enddo
      return
      end
c-------
      subroutine chk4s5s(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)
```

```
      dimension ax(10,10),ay(10,10),az(10,10)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,200) nums
 200  format('========== checking ',i1,'s ==========')
       j = 0
       l = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      continue
            ir = 1
            is = 2
            it = 3
            iu = 4
            call cp4s5s(nums)
            ir = 2
            is = 1
            it = 4
            iu = 3
            call cp4s5s(nums)
            ir = 1
            is = 2
            it = 4
            iu = 3
            call cp4s5s(nums)
            ir = 2
            is = 1
            it = 3
            iu = 4
            call cp4s5s(nums)
      enddo
      return
      end
c-------
      subroutine chk2131(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,200) nums
 200  format('========== checking ',i2,' ==========')
      if(nums.eq.21) iu = 0
      if(nums.eq.31) is = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      continue
         j = i
         l = k
         if(nums.eq.21) then
            ir = 1
            is = 2
            it = 3
            call cp2131(nums)
         else
            ir = 1
            it = 3
            iu = 4
            call cp2131(nums)
         endif
         if(nums.eq.21) then
            ir = 2
            is = 1
            it = 3
            call cp2131(nums)
         else
            ir = 1
            it = 4
            iu = 3
            call cp2131(nums)
         endif
         if(nums.eq.21) then
            ir = 1
            is = 2
            it = 4
            call cp2131(nums)
         else
            ir = 2
            it = 3
            iu = 4
            call cp2131(nums)
         endif
         if(nums.eq.21) then
            ir = 2
            is = 1
            it = 4
            call cp2131(nums)
         else
            ir = 2
            it = 4
            iu = 3
            call cp2131(nums)
         endif
      enddo
      return
      end
c-------
      subroutine chk2233(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,200) nums
 200  format('========== checking ',i2,' ==========')

      if(nums.eq.33) go to 100

      do n = 1,4
         l = 0
         go to (1,2,3,4),n
 1       i = 1
         j = 2
         k = 3
         go to 99
 2       i = 2
         j = 1
         k = 3
         go to 99
 3       i = 1
         j = 2
         k = 4
         go to 99
```

```
4       i = 2
        j = 1
        k = 4
99      continue
        ir = 1
        is = 2
        it = 3
        call cp2233(nums)
        ir = 2
        is = 1
        it = 3
        call cp2233(nums)
        ir = 1
        is = 2
        it = 4
        call cp2233(nums)
        ir = 2
        is = 1
        it = 4
        call cp2233(nums)
     enddo
     go to 999

100  j = 0
     do n = 5,8
     go to (5,6,7,8),n
5       i = 1
        k = 3
        l = 4
        go to 9
6       i = 1
        k = 4
        l = 3
        go to 9
7       i = 2
        k = 3
        l = 4
        go to 9
8       i = 2
        k = 4
        l = 3
9      continue
           ir = 1
           it = 3
           iu = 4
           call cp2233(nums)
           ir = 1
           it = 4
           iu = 3
           call cp2233(nums)
           ir = 2
           it = 3
           iu = 4
           call cp2233(nums)
           ir = 2
           it = 4
           iu = 3
           call cp2233(nums)
        enddo
999  return
     end
c-------
     subroutine chk4243(nums)
     implicit real*8 (a-h,o-z)
     character*7 chr7(2)
     include 'MSSIZE'
     include 'sdci00.h'
     include 'sdci01.h'
     include 'tmomnt.h'
     include 'window.h'
     common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
    $dummy1(1)

     dimension ax(10,10),ay(10,10),az(10,10)
     common /gmom/ ax,ay,az
     common /sufx/ i,j,k,l,ir,is,it,iu

     write(iout,200) nums
200  format('========== checking ',i2,' ==========')

     if(nums.eq.42) go to 9
     if(nums.eq.43) go to 19
9       l = 0
     do n = 1,4
        go to (1,2,3,4),n
1       i = 1
        j = 2
        k = 3
        go to 99
2       i = 2
        j = 1
        k = 3
        go to 99
3       i = 1
        j = 2
        k = 4
        go to 99
4       i = 2
        j = 1
        k = 4
99      continue

        ir = 1
        is = 2
        it = 3
        iu = 4
        call cp4243(nums)
        ir = 2
        is = 1
        it = 4
        iu = 3
        call cp4243(nums)
        ir = 1
        is = 2
        it = 4
        iu = 3
        call cp4243(nums)
        ir = 2
        is = 1
        it = 3
        iu = 4
        call cp4243(nums)
     enddo
     go to 999
19      j = 0
     do n = 1,4
        go to (11,12,13,14),n
11      i = 1
        k = 3
        l = 4
        go to 199
12      i = 1
        k = 4
        l = 3
        go to 199
13      i = 2
        k = 3
        l = 4
        go to 199
14      i = 2
        k = 4
        l = 3
199     continue
        ir = 1
        is = 2
        it = 3
        iu = 4
        call cp4243(nums)
        ir = 2
        is = 1
        it = 4
        iu = 3
        call cp4243(nums)
        ir = 1
        is = 2
        it = 4
        iu = 3
        call cp4243(nums)
        ir = 2
        is = 1
        it = 3
        iu = 4
        call cp4243(nums)
     enddo
999  return
     end
```

```
c-------
      subroutine chk4455(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,200) nums
 200  format('========== checking ',i2,' ==========')
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         j = 2
         k = 3
         l = 4
         go to 99
 2       i = 2
         j = 1
         k = 4
         l = 3
         go to 99
 3       i = 1
         j = 2
         k = 4
         l = 3
         go to 99
 4       i = 2
         j = 1
         k = 3
         l = 4
         go to 99
 99      continue
         ir = 1
         is = 2
         it = 3
         iu = 4
         call cp4455(nums)
         ir = 2
         is = 1
         it = 4
         iu = 3
         call cp4455(nums)
         ir = 1
         is = 2
         it = 4
         iu = 3
         call cp4455(nums)
         ir = 2
         is = 1
         it = 3
         iu = 4
         call cp4455(nums)
      enddo
      return
      end
c-----------------------------------------------------------------------
      subroutine cp2s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      dimension iob1(4),iob2(4)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      iob1(1) = 2
      iob1(2) = 2
      iob1(3) = 0
      iob1(4) = 0
      iob2(1) = 2
      iob2(2) = 2
      iob2(3) = 0
      iob2(4) = 0
      ibin=0
      write(iout,10) i,k, ir,it,is,it
      iob1(ir) = 1
      iob1(is) = 1
      iob1(it) = 2
      write(iout,12) (iob1(kk),kk=1,4)
      iob2(i) = 1
      iob2(k) = 1
      write(iout,12) (iob2(kk),kk=1,4)
      do kk=1,4
         ibin = abs( iob1(kk) - iob2(kk) ) + ibin
      enddo
      write(iout,14) ibin
 10   format(/'(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')
 12   format(4i3)
 14   format('='i3)

      d_ri = 1.0d0* ( i-ir ) + 1.0d0
      d_tk = 1.0d0* ( k-it ) + 1.0d0
      d_si = 1.0d0* ( i-is ) + 1.0d0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_si.ne.1.0) d_si = 0.0
      if(d_tk.eq.1.0 .and. (d_ri.eq.1.0 .or. d_si.eq.1.0) ) then
         if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '********** bingo '
         dex= -1.0d0 * d_ri * d_tk * ax(is,it)
         dey= -1.0d0 * d_ri * d_tk * ay(is,it)
         dez= -1.0d0 * d_ri * d_tk * az(is,it)
      else
         go to 99
      endif
      write(iout,210) dex,dey,dez
 210  format(3f10.5)
c      write(iout,220) d_ri,d_tk,ax(is,it),ay(is,it),az(is,it)
c 220  format(2f5.1,2x,3f10.5)
 99   return
      end
c-----------------------------------------------------------------------
      subroutine cp3s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      dimension iob1(4),iob2(4)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      iob1(1) = 2
      iob1(2) = 2
      iob1(3) = 0
      iob1(4) = 0
      iob2(1) = 2
      iob2(2) = 2
      iob2(3) = 0
      iob2(4) = 0
      ibin=0
      write(iout,10) i,k, ir,it,ir,iu
      iob1(ir) = 0
      iob1(it) = 1
      iob1(iu) = 1
      write(iout,12) (iob1(kk),kk=1,4)
      iob2(i) = 1
      iob2(k) = 1
      write(iout,12) (iob2(kk),kk=1,4)
      do kk=1,4
         ibin = abs( iob1(kk) - iob2(kk) ) + ibin
      enddo
      write(iout,14) ibin
 10   format(/'(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')
 12   format(4i3)
```

```
 14   format('='i3)

      d_ri = 1.0d0* ( i-ir ) + 1.0d0
      d_tk = 1.0d0* ( k-it ) + 1.0d0
      d_uk = 1.0d0* ( k-iu ) + 1.0d0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_uk.ne.1.0) d_uk = 0.0
      if(d_ri.eq.1.0 .and. (d_tk.eq.1.0 .or. d_uk.eq.1.0) ) then
         if(d_ri.eq.1.0 .and. d_uk.eq.1.0) write(iout,*) '********** bingo '
         dex= 1.0d0 * d_ri * d_uk * ax(ir,it)
         dey= 1.0d0 * d_ri * d_uk * ay(ir,it)
         dez= 1.0d0 * d_ri * d_uk * az(ir,it)
      else
         go to 99
      endif
      write(iout,210) dex,dey,dez
 210  format(3f10.5)
c      write(iout,220) d_ri,d_tk,ax(ir,it),ay(ir,it),az(ir,it)
c 220  format(2f5.1,2x,3f10.5)
 99   return
      end
c-----------------------------------------------------------------------
      subroutine cp4s5s(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      dimension iob1(4),iob2(4)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      iob1(1) = 2
      iob1(2) = 2
      iob1(3) = 0
      iob1(4) = 0
      iob2(1) = 2
      iob2(2) = 2
      iob2(3) = 0
      iob2(4) = 0
      ibin=0
      write(iout,10) i,k, ir,it,is,iu
      iob1(ir) = 1
      iob1(is) = 1
      iob1(it) = 1
      iob1(iu) = 1
      write(iout,12) (iob1(kk),kk=1,4)
      iob2(i) = 1
      iob2(k) = 1
      write(iout,12) (iob2(kk),kk=1,4)
      do kk=1,4
         ibin = abs( iob1(kk) - iob2(kk) ) + ibin
      enddo
      write(iout,14) ibin
 10   format(/'(',i1,'-',i1,')',3x,'(',i1,'-',i1,2x,i1,'-',i1,')')
 12   format(4i3)
 14   format('='i3)

      d_ri = 1.0d0* ( i-ir ) + 1.0d0
      d_tk = 1.0d0* ( k-it ) + 1.0d0
      d_si = 1.0d0* ( i-is ) + 1.0d0

      if(d_ri.ne.1.0) d_ri = 0.0d0
      if(d_tk.ne.1.0) d_tk = 0.0d0
      if(d_si.ne.1.0) d_si = 0.0d0

      if(d_tk.eq.1.0 .and. (d_ri.eq.1.0 .or. d_si.eq.1.0) ) then
         if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '********** bingo '
         if(nums.eq.4) then
            dex= -1.0d0 * dsqrt(0.5d0) * d_ri * d_tk * ax(is,iu)
            dey= -1.0d0 * dsqrt(0.5d0) * d_ri * d_tk * ay(is,iu)
            dez= -1.0d0 * dsqrt(0.5d0) * d_ri * d_tk * az(is,iu)
         else if(nums.eq.5) then
            dex= -1.0d0 * dsqrt(1.5d0) * d_ri * d_tk * ax(is,iu)
            dey= -1.0d0 * dsqrt(1.5d0) * d_ri * d_tk * ay(is,iu)
            dez= -1.0d0 * dsqrt(1.5d0) * d_ri * d_tk * az(is,iu)
         endif
      else
         go to 99
      endif
      write(iout,210) dex,dey,dez
      if(nums.eq.4) then
         write(iout,220) ax(is,iu),ay(is,iu),az(is,iu),d_ri,d_tk
      else if(nums.eq.5) then
         write(iout,220) ax(is,iu),ay(is,iu),az(is,iu),d_ri,d_tk
      endif
 210  format(3f10.5)
 220  format(3f10.5,2x,2f5.1)
 99   return
      end
c-----------------------------------------------------------------------
      subroutine cp2131(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      dimension iob1(4),iob2(4)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      iob1(1) = 2
      iob1(2) = 2
      iob1(3) = 0
      iob1(4) = 0
      iob2(1) = 2
      iob2(2) = 2
      iob2(3) = 0
      iob2(4) = 0
      ibin=0
      if(nums.eq.21) then
         write(iout,10) i,k,i,k, ir,it,is,it
         iob1(ir) = 1
         iob1(is) = 1
         iob1(it) = 2
      else if(nums.eq.31) then
         write(iout,10) i,k,i,k, ir,it,ir,iu
         iob1(ir) = 0
         iob1(it) = 1
         iob1(iu) = 1
      endif
      write(iout,12) (iob1(kk),kk=1,4)
      iob2(i) = 0
      iob2(k) = 2
      write(iout,12) (iob2(kk),kk=1,4)
      do kk=1,4
         ibin = abs( iob1(kk) - iob2(kk) ) + ibin
      enddo
      write(iout,14) ibin
 10   format(/'(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')
 12   format(4i3)
 14   format('='i3)

      d_ri = 1.0d0* ( i-ir ) + 1.0d0
      d_tk = 1.0d0* ( k-it ) + 1.0d0
      d_si = 1.0d0* ( i-is ) + 1.0d0
      d_uk = 1.0d0* ( k-iu ) + 1.0d0

      if(d_ri.ne.1.0) d_ri = 0.0d0
      if(d_tk.ne.1.0) d_tk = 0.0d0
      if(d_si.ne.1.0) d_si = 0.0d0
      if(d_uk.ne.1.0) d_uk = 0.0d0

      if(nums.eq.21) then
         if(d_tk.eq.1.0 .and. (d_ri.eq.1.0 .or. d_si.eq.1.0) ) then
            if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '********** bingo '
            dex= dsqrt(0.5d0) * d_ri * d_tk * ax(ir,is)
            dey= dsqrt(0.5d0) * d_ri * d_tk * ay(ir,is)
            dez= dsqrt(0.5d0) * d_ri * d_tk * az(ir,is)
         endif
         else if(nums.eq.31) then
         if(d_ri.eq.1.0 .and. (d_tk.eq.1.0 .or. d_uk.eq.1.0) ) then
```

```
            if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '********** bingo '
            dex= dsqrt(1.5d0) * d_ri * d_tk * ax(it,iu)
            dey= dsqrt(1.5d0) * d_ri * d_tk * ay(it,iu)
            dez= dsqrt(1.5d0) * d_ri * d_tk * az(it,iu)
         endif
      else
         go to 99
      endif
      write(iout,210) dex,dey,dez
      if(nums.eq.21) then
         write(iout,220) ax(ir,is),ay(ir,is),az(ir,is),d_ri,d_tk
      else if(nums.eq.31) then
         write(iout,220) ax(it,iu),ay(it,iu),az(it,iu),d_ri,d_tk
      endif
 210  format(3f10.5)
 220  format(3f10.5,2x,2f5.1)
 99   return
      end
c-----------------------------------------------------------------------
      subroutine cp2233(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      dimension iob1(4),iob2(4)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      iob1(1) = 2
      iob1(2) = 2
      iob1(3) = 0
      iob1(4) = 0
      iob2(1) = 2
      iob2(2) = 2
      iob2(3) = 0
      iob2(4) = 0
      ibin=0
      if(nums.eq.22) then
         write(iout,10) i,k,j,k, ir,it,is,it
         iob1(ir) = 1
         iob1(is) = 1
         iob1(it) = 2
         iob2(i)  = 1
         iob2(j)  = 1
         iob2(k)  = 2
      else if(nums.eq.33) then
         write(iout,10) i,k,i,l, ir,it,ir,iu
         iob1(ir) = 0
         iob1(it) = 1
         iob1(iu) = 1
         iob2(i)  = 0
         iob2(k)  = 1
         iob2(l)  = 1
      endif
      write(iout,12) (iob1(kk),kk=1,4)
      write(iout,12) (iob2(kk),kk=1,4)
      do kk=1,4
         ibin = abs( iob1(kk) - iob2(kk) ) + ibin
      enddo
      write(iout,14) ibin
 10   format(/'(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &        '(',i1,'-',i1,2x,i1,'-',i1,')')
 12   format(4i3)
 14   format('='i3)

      d_ri = 1.0d0* ( i-ir ) + 1.0d0
      d_tk = 1.0d0* ( k-it ) + 1.0d0
      d_sj = 1.0d0* ( j-is ) + 1.0d0
      d_ul = 1.0d0* ( l-iu ) + 1.0d0

      d_si = 1.0d0* ( i-is ) + 1.0d0
      d_uk = 1.0d0* ( k-iu ) + 1.0d0

      if(d_ri.ne.1.0) d_ri = 0.0d0
      if(d_tk.ne.1.0) d_tk = 0.0d0
      if(d_sj.ne.1.0) d_sj = 0.0d0
      if(d_ul.ne.1.0) d_ul = 0.0d0

      if(d_si.ne.1.0) d_si = 0.0d0
      if(d_uk.ne.1.0) d_uk = 0.0d0
c###
      if(nums.eq.22) then
c         if(d_tk.eq.1.0 .and. (d_ri.eq.1.0 .or. d_si.eq.1.0) ) then
            if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '********** bingo '
            dex= d_ri*d_tk*( d_sj*(tlx + 2.0d0*ax(it,it)-ax(ir,ir)-ax(is,j)))
            dey= d_ri*d_tk*( d_sj*(tly + 2.0d0*ay(it,it)-ay(ir,ir)-ay(is,j)))
            dez= d_ri*d_tk*( d_sj*(tlz + 2.0d0*az(it,it)-az(ir,ir)-az(is,j)))
c         endif
         else if(nums.eq.33) then
c         if(d_ri.eq.1.0 .and. (d_tk.eq.1.0 .or. d_uk.eq.1.0) ) then
            if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '********** bingo '
            dex= d_ri*d_tk*( d_ul*(tlx + ax(it,it)-2.0d0*ax(ir,ir)+ax(l,iu)))
            dey= d_ri*d_tk*( d_ul*(tly + ay(it,it)-2.0d0*ay(ir,ir)+ay(l,iu)))
            dez= d_ri*d_tk*( d_ul*(tlz + az(it,it)-2.0d0*az(ir,ir)+az(l,iu)))
c         endif
      else
         go to 99
      endif
      write(iout,210) dex,dey,dez
      if(nums.eq.22) then
         write(iout,220) ax(it,it),ay(it,it),az(it,it),
     &                   ax(ir,ir),ay(ir,ir),az(ir,ir),
     &                   ax(is,j ),ay(is,j ),az(is,j ),
     &                   d_ri,d_tk,d_sj
      else if(nums.eq.33) then
         write(iout,220) ax(it,it),ay(it,it),az(it,it),
     &                   ax(ir,ir),ay(ir,ir),az(ir,ir),
     &                   ax(l ,iu),ay(l ,iu),az(l ,iu),
     &                   d_ri,d_tk,d_sj
      endif
 210  format(3f10.5)
 220  format(3(3f10.5/),2x,3f5.1)
 99   return
      end
c-----------------------------------------------------------------------
      subroutine cp4243(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      dimension iob1(4),iob2(4)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

      iob1(1) = 2
      iob1(2) = 2
      iob1(3) = 0
      iob1(4) = 0
      iob2(1) = 2
      iob2(2) = 2
      iob2(3) = 0
      iob2(4) = 0
      ibin=0
      iob1(ir) = 1
      iob1(is) = 1
      iob1(it) = 1
      iob1(iu) = 1
      if(nums.eq.42) then
         write(iout,10) i,k,j,k, ir,it,is,iu
         iob2(i) = 1
         iob2(j) = 1
         iob2(k) = 2
      endif
      if(nums.eq.43) then
         write(iout,10) i,k,i,l, ir,it,is,iu
         iob2(i) = 0
         iob2(k) = 1
         iob2(l) = 1
      endif
      write(iout,12) (iob1(kk),kk=1,4)
      write(iout,12) (iob2(kk),kk=1,4)
      do kk=1,4
```

```
         ibin = abs( iob1(kk) - iob2(kk) ) + ibin
      enddo
      write(iout,14) ibin
 10   format(/'(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &         '(',i1,'-',i1,2x,i1,'-',i1,')')
 12   format(4i3)
 14   format('='i3)

      d_ri = 1.0d0* ( i-ir ) + 1.0d0
      d_sj = 1.0d0* ( j-is ) + 1.0d0
      d_tk = 1.0d0* ( k-it ) + 1.0d0
      d_ul = 1.0d0* ( l-iu ) + 1.0d0

      if(d_ri.ne.1.0) d_ri = 0.0d0
      if(d_sj.ne.1.0) d_sj = 0.0d0
      if(d_tk.ne.1.0) d_tk = 0.0d0
      if(d_ul.ne.1.0) d_ul = 0.0d0

      if(nums.eq.42) then
         if(d_ri.eq.1.0 .and. d_sj.eq.1.0 .and. d_tk.eq.1.0)
     &      write(iout,*) '********** bingo '
            dex= dsqrt(2.0d0) * d_ri * d_sj * d_tk * ax(it,iu)
            dey= dsqrt(2.0d0) * d_ri * d_sj * d_tk * ay(it,iu)
            dez= dsqrt(2.0d0) * d_ri * d_sj * d_tk * az(it,iu)
         else if(nums.eq.43) then
            if(d_ri.eq.1.0 .and. d_tk.eq.1.0 .and. d_ul.eq.1.0)
     &      write(iout,*) '********** bingo '
            dex= dsqrt(2.0d0) * d_ri * d_tk * d_ul * ax(it,iu)
            dey= dsqrt(2.0d0) * d_ri * d_tk * d_ul * ay(it,iu)
            dez= dsqrt(2.0d0) * d_ri * d_tk * d_ul * az(it,iu)
      else
         go to 99
      endif
      write(iout,210) dex,dey,dez
      if(nums.eq.42) then
         write(iout,220) ax(it,iu),ay(it,iu),az(it,iu),d_ri,d_sj,d_tk
      else if(nums.eq.43) then
         write(iout,220) ax(it,iu),ay(it,iu),az(it,iu),d_ri,d_tk,d_ul
      endif
 210  format(3f10.5)
 220  format(3f10.5,2x,3f5.1)
 99   return
      end
c-----------------------------------------------------------------------
      subroutine cp4455(nums)
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'tmomnt.h'
      include 'window.h'
      common/work01/civec(mcics2),tdx(mcipck),tdy(mcipck),tdz(mcipck),
     $dummy1(1)

      dimension ax(10,10),ay(10,10),az(10,10)
      dimension iob1(4),iob2(4)
      common /gmom/ ax,ay,az
      common /sufx/ i,j,k,l,ir,is,it,iu

c      iob1(1) = 2
c      iob1(2) = 2
c      iob1(3) = 0
c      iob1(4) = 0
c      iob2(1) = 2
c      iob2(2) = 2
c      iob2(3) = 0
c      iob2(4) = 0
      ibin=0
      write(iout,10) i,k,j,k, ir,it,is,it
      iob1(ir) = 1
      iob1(is) = 1
      iob1(it) = 1
      iob1(iu) = 1
      iob2(i)  = 1
      iob2(j)  = 1
      iob2(k)  = 1
      iob2(l)  = 1
      write(iout,12) (iob1(kk),kk=1,4)
      write(iout,12) (iob2(kk),kk=1,4)
      do kk=1,4
         ibin = abs( iob1(kk) - iob2(kk) ) + ibin
      enddo
      write(iout,14) ibin
 10   format(/'(',i1,'-',i1,2x,i1,'-',i1,')',3x,
     &         '(',i1,'-',i1,2x,i1,'-',i1,')')
 12   format(4i3)
 14   format('='i3)

      d_ri = 1.0d0* ( i-ir ) + 1.0d0
      d_tk = 1.0d0* ( k-it ) + 1.0d0
      d_sj = 1.0d0* ( j-is ) + 1.0d0
      d_ul = 1.0d0* ( l-iu ) + 1.0d0

      if(d_ri.ne.1.0) d_ri = 0.0d0
      if(d_tk.ne.1.0) d_tk = 0.0d0
      if(d_sj.ne.1.0) d_sj = 0.0d0
      if(d_ul.ne.1.0) d_ul = 0.0d0

c###
      if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '********** bingo '
      dex= d_ri*d_tk*(d_sj*d_ul*(tlx+ax(it,it)-ax(ir,ir))+
     &     d_sj*ax(iu,l)-d_ul*ax(is,j))
      dey= d_ri*d_tk*(d_sj*d_ul*(tlx+ay(it,it)-ay(ir,ir))+
     &     d_sj*ay(iu,l)-d_ul*ay(is,j))
      dez= d_ri*d_tk*(d_sj*d_ul*(tlx+az(it,it)-az(ir,ir))+
     &     d_sj*az(iu,l)-d_ul*az(is,j))

      write(iout,210) dex,dey,dez
      write(iout,220) ax(it,it),ay(it,it),az(it,it),
     &                ax(ir,ir),ay(ir,ir),az(ir,ir),
     &                ax(iu,l ),ay(iu,l ),az(iu,l ),
     &                ax(is,j ),ay(is,j ),az(is,j ),
     &                d_ri,d_tk,d_sj,d_ul
 210  format(3f10.5)
 220  format(4(3f10.5/),2x,4f5.1)
      return
      end
c-----------------------------------------------------------------------

      subroutine intout
c=====================================================================*
c         INTOUT by K. Ohtawara
c subroutines
c   check2s  check21
c   check3s  check31  check32
c   check4s  check41  check42
c   check5s  check51  check52
c
c  CI matrix check program
c   +10 +1s +11
c   +20 -2s =21 +22
c   +30 -3s =31 =32 +33
c   +40 =4s =41 =42 =43 +44
c   +50 =5s =51 =52 -53 -54 +55
c   + : OK before debug
c   = : OK by debug
c   - : debugging now
c     : not yet
c
c=====================================================================*
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common / io / in, iout, ifile(10)

      if(idebug.ne.2) go to 999
      do i = 1,4
         do j = 1,4
            kpp  = max0( i,j )
            kppx = min0( i,j )
            nu   = kpp*( kpp - 1 )/2 + kppx
            do k= 1,4
               do l= 1,4
                  kp   = max0( k,l )
                  kpx  = min0( k,l )
                  nux  = kp *( kp  - 1 )/2 + kpx
```

```
                iq   = max0(nu,nux)
                iqx  = min0(nu,nux)
                mu   = iq*( iq - 1 )/2 + iqx
c                write(iout,100) i,j,k,l,mu,gintgl( mu )
c 100            format('( ij | kl ) ',4i3,i5,f10.5)
                a(i,j,k,l) = gintgl( mu )
             enddo
          enddo
       enddo
    enddo


c*********************************************************************
c=====================================================================
c     Control the checking program
c      -> Uncomment call statements you want to check
c=====================================================================
c*********************************************************************

c      call check2s
c      call check3s

c zero except for BINGO (BINGO means kroneker delta)
cOK      call check4s
cOK      call check5s

c zero except for BINGO
cOK      call check21
cOK      call check31

c zero except for BINGO
cOK      call check41
cOK      call check51

c zero except for BINGO
cOK      call check32

c zero except for BINGO
cOK      call check42
cOK      call check52

c zero except for BINGO
cOK      call check43
cOK      call check53
cOK      call check54

 999  return
      end
c=====================================================================
      subroutine check2s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 2s'
       j = 0
       l = 0
      iu = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      ir = 1
         is = 2
         it = 3
         call comp2s
         ir = 1
         is = 2
         it = 4
         call comp2s
         ir = 2
         is = 1
         it = 3
         call comp2s
         ir = 2
         is = 1
         it = 4
         call comp2s
      enddo
      return
      end

c=====================================================================
      subroutine check3s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 3s'
       j = 0
       l = 0
      is = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      ir = 1
         it = 3
         iu = 4
         call comp3s
         ir = 1
         it = 4
         iu = 3
         call comp3s
         ir = 2
         it = 3
         iu = 4
         call comp3s
         ir = 2
         it = 4
         iu = 3
         call comp3s
      enddo
      return
      end
c=====================================================================
      subroutine check4s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
```

```
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 4s'
       j = 0
       l = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      ir = 1
         is = 2
         it = 3
         iu = 4
         call comp4s
         ir = 1
         is = 2
         it = 4
         iu = 3
         call comp4s
         ir = 2
         is = 1
         it = 3
         iu = 4
         call comp4s
         ir = 2
         is = 1
         it = 4
         iu = 3
         call comp4s
      enddo
      return
      end

c=====================================================================
      subroutine check5s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 5s'
       j = 0
       l = 0
       do n=1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      ir = 1
         is = 2
         it = 3
         iu = 4
         call comp5s
         ir = 1
         is = 2
         it = 4
         iu = 3
         call comp5s
         ir = 2
         is = 1
         it = 3
         iu = 4
         call comp5s
         ir = 2
         is = 1
         it = 4
         iu = 3
         call comp5s
      enddo
      return
      end
c=====================================================================
      subroutine check21
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 21'
      j = 0
      l = 0
      iu = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      ir = 1
         is = 2
         it = 3
         call comp21
         ir = 1
         is = 2
         it = 4
         call comp21
         ir = 2
         is = 1
         it = 3
         call comp21
         ir = 2
         is = 1
         it = 4
         call comp21
      enddo
      return
      end
c=====================================================================
      subroutine check31
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu
```

```
      write(iout,*) '*** CHECKING 31'
      j = 0
      l = 0
      is = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         go to 99
 2       i = 1
         k = 4
         go to 99
 3       i = 2
         k = 3
         go to 99
 4       i = 2
         k = 4
 99      ir = 1
         it = 3
         iu = 4
         call comp31
         ir = 1
         it = 4
         iu = 3
         call comp31
         ir = 2
         it = 3
         iu = 4
         call comp31
         ir = 2
         it = 4
         iu = 3
         call comp31
      enddo
      return
      end

c=====================================================================
      subroutine check41
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 41'
       j = 0
       l = 0
       do n = 1,4
          go to (1,2,3,4),n
 1        i = 1
          k = 3
          go to 99
 2        i = 1
          k = 4
          go to 99
 3        i = 2
          k = 3
          go to 99
 4        i = 2
          k = 4
 99       ir = 1
          is = 2
          it = 3
          iu = 4
          call comp41
          ir = 1
          is = 2
          it = 4
          iu = 3
          call comp41
          ir = 2
          is = 1
          it = 3
          iu = 4
          call comp41
          ir = 2
          is = 1
          it = 4
          iu = 3
          call comp41
       enddo
      return
      end
c=====================================================================
      subroutine check32
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 32'
       l = 0
       is= 0
       do n = 1,4
          go to (1,2,3,4),n
 1        i = 1
          j = 2
          k = 3
          go to 99
 2        i = 2
          j = 1
          k = 3
          go to 99
 3        i = 1
          j = 2
          k = 4
          go to 99
 4        i = 2
          j = 1
          k = 4
 99       ir = 1
          it = 3
          iu = 4
          call comp32
          ir = 1
          it = 4
          iu = 3
          call comp32
          ir = 2
          it = 3
          iu = 4
          call comp32
          ir = 2
          it = 4
          iu = 3
          call comp32
       enddo
       return
       end
c=====================================================================
      subroutine check42
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 42'
       l = 0
       do n = 1,4
```

```
         go to (1,2,3,4),n
1        i = 1
         j = 2
         k = 3
         go to 99
2        i = 1
         j = 2
         k = 4
         go to 99
3        i = 2
         j = 1
         k = 3
         go to 99
4        i = 2
         j = 1
         k = 4
99       ir = 1
         is = 2
         it = 3
         iu = 4
         call comp42
         ir = 1
         is = 2
         it = 4
         iu = 3
         call comp42
         ir = 2
         is = 1
         it = 3
         iu = 4
         call comp42
         ir = 2
         is = 1
         it = 4
         iu = 3
         call comp42
      enddo
      return
      end
c=====================================================================
      subroutine check51
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 51'
      j = 0
      l = 0
      do n = 1,4
         go to (1,2,3,4),n
1        i = 1
         k = 3
         go to 99
2        i = 1
         k = 4
         go to 99
3        i = 2
         k = 3
         go to 99
4        i = 2
         k = 4
99       ir = 1
         is = 2
         it = 3
         iu = 4
         call comp51
         ir = 1
         is = 2
         it = 4
         iu = 3
         call comp51
         ir = 2
         is = 1
         it = 3
         iu = 4
         call comp51
         ir = 2
         is = 1
         it = 4
         iu = 3
         call comp51
      enddo
      return
      end
c=====================================================================
      subroutine check52
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 52'
      l = 0
      do n = 1,4
         go to (1,2,3,4),n
1        i = 1
         j = 2
         k = 3
         go to 99
2        i = 1
         j = 2
         k = 4
         go to 99
3        i = 2
         j = 1
         k = 3
         go to 99
4        i = 2
         j = 1
         k = 4
99       ir = 1
         is = 2
         it = 3
         iu = 4
         call comp52
         ir = 1
         is = 2
         it = 4
         iu = 3
         call comp52
         ir = 2
         is = 1
         it = 3
         iu = 4
         call comp52
         ir = 2
         is = 1
         it = 4
         iu = 3
         call comp52
      enddo
      return
      end
c=====================================================================
      subroutine check43
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
```

```
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 43'
      j = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         l = 4
         go to 99
 2       i = 1
         k = 4
         l = 3
         go to 99
 3       i = 2
         k = 3
         l = 4
         go to 99
 4       i = 2
         k = 4
         l = 3
 99      ir = 1
         is = 2
         it = 3
         iu = 4
         call comp43
         ir = 1
         is = 2
         it = 4
         iu = 3
         call comp43
         ir = 2
         is = 1
         it = 3
         iu = 4
         call comp43
         ir = 2
         is = 1
         it = 4
         iu = 3
         call comp43
      enddo
      return
      end
c=====================================================================
      subroutine check53
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 53'
      j = 0
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         k = 3
         l = 4
         go to 99
 2       i = 1
         k = 4
         l = 3
         go to 99
 3       i = 2
         k = 3
         l = 4
         go to 99
 4       i = 2
         k = 4
         l = 3
 99      ir = 1
         is = 2
         it = 3
         iu = 4
         call comp53
         ir = 1
         is = 2
         it = 4
         iu = 3
         call comp53
         ir = 2
         is = 1
         it = 3
         iu = 4
         call comp53
         ir = 2
         is = 1
         it = 4
         iu = 3
         call comp53
      enddo
      return
      end
c=====================================================================
      subroutine check54
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      write(iout,*) '*** CHECKING 54'
      do n = 1,4
         go to (1,2,3,4),n
 1       i = 1
         j = 2
         k = 3
         l = 4
         go to 99
 2       i = 1
         j = 2
         k = 4
         l = 3
         go to 99
 3       i = 2
         j = 1
         k = 3
         l = 4
         go to 99
 4       i = 2
         j = 1
         k = 4
         l = 3
 99      ir = 1
         is = 2
         it = 3
         iu = 4
         call comp54
         ir = 1
         is = 2
         it = 4
         iu = 3
         call comp54
         ir = 2
         is = 1
         it = 3
         iu = 4
         call comp54
         ir = 2
         is = 1
         it = 4
         iu = 3
         call comp54
      enddo
      return
      end

c*********************************************************************
c=====================================================================
```

```
C==============================================================================
C******************************************************************************
      subroutine comp2s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_kt = 1.0* ( k-it ) + 1.0
      d_ri = 1.0* ( i-ir ) + 1.0
      d_tk = 1.0* ( it-k ) + 1.0
      if(d_kt.ne.1.0) d_kt = 0.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_kt.eq.1.0 .and. d_ri.eq.1.0 .and. d_tk.eq.1.0)
     & write(iout,*) '              *** BINGO ***'

      debug = -1.0*(   d_ri*a(is,it,k,it)
     &         -d_tk*( a(is,it,ir,i)+a(ir,it,is,i) )  )

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_kt,d_ri,d_tk
      write(iout,220)   is,it,k,it,   is,it,ir,i,   ir,it,is,i
      write(iout,230) a(is,it,k,it),a(is,it,ir,i),a(ir,it,is,i)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta kt, ir, tk : ',3f5.1)
 220  format(2x,3(4i2,2x))
 230  format(3f10.5/)
      return
      end
c---
      subroutine comp3s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( i-ir ) + 1.0
      d_tk = 1.0* ( it-k ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_ri.eq.1.0 .and. d_tk.eq.1.0)
     & write(iout,*) '              *** BINGO ***'

      debug = d_ri*( a(ir,it,k,iu)+a(k,it,ir,iu) )
     &         -d_tk* a(ir,i,ir,iu)

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_tk
      write(iout,220)   ir,it,k,iu,   k,it,ir,iu ,  ir,i,ir,iu
      write(iout,230) a(ir,it,k,iu),a(k,it,ir,iu),a(ir,i,ir,iu)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta ir, tk : ',2f5.1)
 220  format(2x,3(4i2,2x))
 230  format(3f10.5/)
      return
      end
c---
      subroutine comp4s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( i-ir ) + 1.0
      d_tk = 1.0* ( it-k ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '*** BINGO ***'

      debug = -sqrt(0.5)*( -d_tk*( a(ir,i,is,iu)+a(is,i,ir,iu) )+
     &         d_ri*( a(is,it,k,iu)+a(k,it,is,iu) )  )

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_tk
      write(iout,220)   ir,i,is,iu,  is,i,ir,iu,
     &                  is,it,k,iu,  k,it,is,iu
      write(iout,230) a(ir,i,is,iu),a(is,i,ir,iu),
     &                a(is,it,k,iu),a(k,it,is,iu)

 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta kt, ir, tk : ',2f5.1)
 220  format(2x,4(4i2,2x))
 230  format(4f10.5/)
      return
      end
c---
      subroutine comp5s
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( i-ir ) + 1.0
      d_tk = 1.0* ( it-k ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_ri.eq.1.0 .and. d_tk.eq.1.0) write(iout,*) '*** BINGO ***'

      debug = -sqrt(1.5)*( -d_tk*( a(ir,i,is,iu)-a(is,i,ir,iu) )
     &         -d_ri*( a(is,it,k,iu)-a(k,it,is,iu) )  )

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_tk
      write(iout,220)   ir,i,is,iu,  is,i,ir,iu,
     &                  is,it,k,iu,  k,it,is,iu
      write(iout,230) a(ir,i,is,iu),a(is,i,ir,iu),
     &                a(is,it,k,iu),a(k,it,is,iu)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta kt, ir, tk : ',2f5.1)
 220  format(2x,4(4i2,2x))
 230  format(4f10.5/)
      return
      end
c---
      subroutine comp21
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
```

```
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( i-ir ) + 1.0
      d_kt = 1.0* ( k-it ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_kt.ne.1.0) d_kt = 0.0
      if(d_ri.eq.1.0 .and. d_kt.eq.1.0)
     & write(iout,*) '              *** BINGO ***'

      debug = sqrt(2.0)*d_kt*(d_ri*( 2.0*a(it,it,ir,is)-a(it,is,it,ir))
     &        -a(i,ir,i,is) )

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_kt,d_ri
      write(iout,220)  it,it,ir,is ,it,is,it,ir ,
     &                 i,ir,i,is,   i,ir,i,is
      write(iout,230) a(it,it,ir,is),a(it,is,it,ir),
     &                a(i,ir,i,is),a(i,ir,i,is)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta kt, ri : ',2f5.1)
 220  format(2x,4(4i2,2x))
 230  format(4f10.5/)
      return
      end
c---
      subroutine comp31
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( i-ir ) + 1.0
      d_ku = 1.0* ( k-iu ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_ku.ne.1.0) d_ku = 0.0
      if(d_ri.eq.1.0 .and. d_ku.eq.1.0)
     & write(iout,*) '              *** BINGO ***'

      debug = sqrt(2.0)*d_ri*(d_ku*(-2.0*a(ir,ir,it,iu)+a(ir,iu,ir,it))
     &         +a(k,it,k,iu))

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_ku
      write(iout,220)   ir,ir,it,iu ,  ir,iu,ir,it,   k,it,k,iu
      write(iout,230) a(ir,ir,it,iu),a(ir,iu,ir,it),a(k,it,k,iu)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta ri, ku : ',2f5.1)
 220  format(2x,3(4i2,2x))
 230  format(3f10.5/)
      return
      end
c---
      subroutine comp41
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( i-ir ) + 1.0
      d_ku = 1.0* ( k-iu ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_ku.ne.1.0) d_ku = 0.0
      if(d_ri.eq.1.0 .and. d_ku.eq.1.0)
     & write(iout,*) '              *** BINGO ***'

      debug = d_ri*d_ku*( 2*a(iu,it,is,ir)-a(iu,ir,is,it) )

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_ku
      write(iout,220)   iu,it,is,ir,   iu,ir,is,it
      write(iout,230) a(iu,it,is,ir),a(iu,ir,is,it)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta ri, ku     : ',2f5.1)
 220  format(2x,2(4i2,2x))
 230  format(2f10.5/)
      return
      end
c---
      subroutine comp51
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( i-ir ) + 1.0
      d_ku = 1.0* ( k-iu ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_ku.ne.1.0) d_ku = 0.0
      if(d_ri.eq.1.0 .and. d_ku.eq.1.0)
     & write(iout,*) '              *** BINGO ***'

      debug = sqrt(3.0)*d_ri*d_ku*a(iu,ir,is,it)

      write(iout,190) i,j,k,l,  ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_ku
      write(iout,220)   iu,ir,is,it
      write(iout,230) a(iu,ir,is,it)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta ri, ku     : ',f5.1)
 220  format(2x,4i2)
 230  format(f10.5/)
      return
      end
c---
      subroutine comp32
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( ir-i ) + 1.0
      d_rj = 1.0* ( ir-j ) + 1.0
      d_uk = 1.0* ( iu-k ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_rj.ne.1.0) d_rj = 0.0
      if(d_uk.ne.1.0) d_uk = 0.0
```

```
      debug = d_uk*(d_ri*(2*a(ir,j,iu,it)-a(ir,it,iu,j))+
     &            d_rj*(2*a(ir,i,iu,it)-a(ir,it,iu,i)))

      if(d_uk.eq.1.0 .and. d_ri.eq.1.0)
     & write(iout,*) '                *** BINGO ***'
      if(d_uk.eq.1.0 .and. d_rj.eq.1.0)
     & write(iout,*) '                *** BINGO ***'

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_uk,d_ri,d_rj
      write(iout,220)   ir,j,iu,it,   ir,it,iu,j,
     &                  ir,i,iu,it,   ir,it,iu,i
      write(iout,230) a(ir,j,iu,it),a(ir,it,iu,j),
     &               a(ir,i,iu,it),a(ir,it,iu,i)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta uk, ir, rj : ',3f5.1)
 220  format(2x,4(4i2,2x))
 230  format(4f10.5/)
      return
      end
c---
      subroutine comp42
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_si = 1.0* ( is-i ) + 1.0
      d_rj = 1.0* ( ir-j ) + 1.0
      d_uk = 1.0* ( iu-k ) + 1.0
      if(d_si.ne.1.0) d_si = 0.0
      if(d_rj.ne.1.0) d_rj = 0.0
      if(d_uk.ne.1.0) d_uk = 0.0
      if(d_si.eq.1.0 .and. d_rj.eq.1.0 .and. d_uk.eq.1.0)
     & write(iout,*) '                *** BINGO ***'

      debug = sqrt(2.0)*d_si* ( d_rj*( a(k,it,k,iu)+
     &         d_uk*( -a(is,is,iu,it)+0.5*a(is,it,is,iu) ) ) +
     &         d_uk*( 0.5*a(ir,it,j,iu)-a(ir,j,iu,it) ) )

      write(iout,190) i,j,k,l,ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_si,d_rj,d_uk
      write(iout,220)   k,it,k,iu,   is,is,iu,it ,  is,it,is,iu,
     &               ir,it,j,iu,  ir,j,iu,it
      write(iout,230) a(k,it,k,iu),a(is,is,iu,it),a(is,it,is,iu),
     &             a(ir,it,j,iu),a(ir,j,iu,it)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta kt, ir, tk : ',3f5.1)
 220  format(2x,5(4i2,2x))
 230  format(5f10.5/)
      return
      end
c---
      subroutine comp52
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( ir-i ) + 1.0
      d_uk = 1.0* ( iu-k ) + 1.0
      d_sj = 1.0* ( is-j ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_uk.ne.1.0) d_uk = 0.0
      if(d_sj.ne.1.0) d_sj = 0.0
      if(d_ri.eq.1.0 .and. d_uk.eq.1.0 .and. d_sj.eq.1.0)
     & write(iout,*) '                *** BINGO ***'

      debug = -sqrt(3.0)/2.0*d_ri*d_uk*( a(is,it,iu,j)+
     &          d_sj*( (sqrt(2.0)-1.0)*a(is,it,is,iu)-
     &          sqrt(2.0)*a(ir,it,ir,iu) ) )

      write(iout,190) i,j,k,l,  ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_uk,d_sj
      write(iout,220)   is,it,iu,j ,  is,it,is,iu ,  ir,it,ir,iu
      write(iout,230) a(is,it,iu,j),a(is,it,is,iu),a(ir,it,ir,iu)
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta kt, ir, tk : ',3f5.1)
 220  format(2x,3(4i2,2x))
 230  format(3f10.5/)
      return
      end
c---
      subroutine comp43
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( ir-i ) + 1.0
      d_tk = 1.0* ( it-k ) + 1.0
      d_ul = 1.0* ( iu-l ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_ul.ne.1.0) d_ul = 0.0
      if(d_ri.eq.1.0 .and. d_tk.eq.1.0 .and. d_ul.eq.1.0)
     & write(iout,*) '                *** BINGO ***'

      debug = sqrt(2.0)*d_tk*( d_ri*( a(ir,is,l,iu) -
     &         0.5*a(is,iu,l,ir)+d_ul*(a(it,it,ir,is)-
     &         0.5*a(it,is,it,ir)) ) - d_ul*a(ir,i,is,i) )

      write(iout,190) i,j,k,l,  ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_tk,d_ul
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta ri, tk, u; : ',3f5.1)
      return
      end
c---
      subroutine comp53
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( ir-i ) + 1.0
      d_tk = 1.0* ( it-k ) + 1.0
      d_ul = 1.0* ( iu-l ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_ul.ne.1.0) d_ul = 0.0
      if(d_ri.eq.1.0 .and. d_tk.eq.1.0 .and. d_ul.eq.1.0)
```

```
     & write(iout,*) '              *** BINGO ***'

      debug = -sqrt(3.0)/2.0*d_ri*d_tk*( a(is,iu,l,ir)+
     &          d_ul*( (sqrt(2.0)-1.0)*a(is,iu,ir,iu)-
     &          sqrt(2.0)*a(it,ir,it,is) ) )

      write(iout,190) i,j,k,l,  ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_tk,d_ul
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta kt, ir, tk : ',3f5.1)
      return
      end
c---
      subroutine comp54
      implicit real*8 (a-h,o-z)
      character*7 chr7(2)
      include 'MSSIZE'
      include 'energy.h'
      include 'sdci00.h'
      include 'sdci01.h'
      include 'sdci02.h'
      include 'window.h'
      include 'work02.h'

      dimension a(10,10,10,10)
      common /gint/ a
      common /sufx/ i,j,k,l,ir,is,it,iu

      d_ri = 1.0* ( ir-i ) + 1.0
      d_tk = 1.0* ( it-k ) + 1.0
      d_sj = 1.0* ( is-j ) + 1.0
      d_ul = 1.0* ( iu-l ) + 1.0
      if(d_ri.ne.1.0) d_ri = 0.0
      if(d_tk.ne.1.0) d_tk = 0.0
      if(d_sj.ne.1.0) d_ul = 0.0
      if(d_ul.ne.1.0) d_ul = 0.0
      if(d_ri.eq.1.0 .and. d_tk.eq.1.0 .and. d_sj.eq.1.0 .and. d_ul.eq.1.0)
     & write(iout,*) '              *** BINGO ***'

      debug = sqrt(1.5)*d_ri*d_tk*( a(is,iu,l,j)-
     &          d_sj*a(ir,iu,ir,l) - d_ul*a(is,it,it,j) +
     &          d_sj*d_ul*a(ir,it,ir,it) )

      write(iout,190) i,j,k,l,  ir,is,it,iu
      write(iout,200) debug
      write(iout,210) d_ri,d_tk,d_ul
 190  format('(i,j,k,l),(r,s,t,u): ',4i2,5x,4i2)
 200  format('  ->   ',f10.6)
 210  format('delta ri, tk, ul : ',3f5.1)
      return
      end
c=====================================================================
```

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 |
| | | | mosf |

機能

メインプログラム

処理

1. 定数の設定 (precal)
2. 入力データの読み込み (jctrl)
3. 分子を構成する原子の属性の設定 (minput)
4. 原子積分の評価 (sgint)
5. Hueckel法によってSCF計算の反復計算の初期MOを求める。(guess)
6. SCF計算の反復計算によりHartree-Fock-Roothaan方程式を解き、MOを求める。(rhfclo)
7. 6.で求まったMOによるHartree-Fockの基底状態での各種物理量の計算 (gprpty)
8. フラッグifciが真ならば、以下の様にして、CI法の固有状態を求める。
9. CI法の基底の定義と分子積分の評価 (molint)
10. isdci0 = 0の場合
    1) オリジナルのSCIにより、CI法の固有状態を求める。(sci)
    2) 遷移モーメントを求める。(tmom)
    3) CI法の固有状態に関連するその他の物理量の計算(eprpty, polar, hypol1, hyplo2)
11. isdci0 ≧ 1の場合
    1) 新規サブルーチンでSDCIを実行して、CI法の固有状態を求める。(sdci1)
    2) 遷移モーメントを求める。(tmsdci)



| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 |
| | | | eri |

機能

理論仕様書(2.6),(2,7)の二中心積分$\Gamma_a$,$\Gamma_{ab}$を評価する。

処理

理論仕様書(A3.4), 図 A3.1の方法で$\Gamma_{ab}$を配列要素gab(n)に格納する。また、原子iの$\Gamma_a$は配列要素gaa(i)に格納する。ただし、積分の評価方法は、フラッグigammaの値によって以下のように指定される。

igamma = 1 解析的
2 Pariser-Parrの方法
3 西本-又賀の方法
4 西本-又賀-Weissの方法
5 大野の方法
6 大野-Klopmanの方法
7 Dasgupta-藤永の方法

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 |
| | | | gprpty |

**機能**

Hartree-Fockの基底状態における物性値の計算

**処理**

1. イオン系のエネルギーと全エネルギーの評価
2. 指定されたMOの各原子ごとの密度分布dm( i, j )の評価
   dm( i, j ) : 原子"i"におけるMO"j"の密度
3. 原子ごとの有効荷数atmchg(i)の評価
4. Hartree-Fockの基底状態における双極子モーメントを計算する。
   dxsp, dysp, dzsp : s-p間の要素からの寄与
   dxpd, dypd, dzpd : p-d間の要素からの寄与
   dxap, dyap, dzap : 全電子系からの寄与
   dxpt, dypt, dzpt :イオン系からの寄与
   dxt, dyt, dzt :全電子系とイオン系からの寄与

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 |
| | | | guess |

**機能**

SCF反復計算のための、初期MOを計算する。

**処理**

1. 対角成分が$-I_r$($I_r$:軌道rのイオン化エネルギー)で、非対角成分が理論仕様書(2.10)のコア積分を求め、(A3.1)と図 A3.1の方法で配列要素hcore (n)に格納する。
2. hcore(n)をh(n)にコピーする。
3. hの対角化を行い、固有ベクトルを試行のMOとする。(diag1)
4. コア積分の対角成分を理論仕様書(2.8)とし、hcoreに格納する。

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 intci0 |

機能

分子積分を求める。(新規サブルーチン)

処理

理論仕様書(A4.2)の分子積分を評価し、(A3.6)の方法で、1次元配列要素gintgl($\mu$)($1 \leqq \mu \leqq \mu_{max}$)に格納する。

ここで、以下の条件が存在する。

$1 \leqq i \leqq n_{sdci}$, $1 \leqq j \leqq i$, $1 \leqq k \leqq n_{sdci}$, $1 \leqq \ell \leqq k$, ($n_{sdci} = n_{ocs} + n_{vcs}$)

$1 \leqq \nu(i,j) \leqq n_{sdci}(n_{sdci}+1)/2$, $1 \leqq \nu(k,\ell) \leqq \nu(i,j)$

主要な変数の内容は以下のとおりである。

ici1($\nu$), ici2($\nu$) : $i_\nu$, $j_\nu$ または、$k_\nu$, $\ell_\nu$

gintgl($\mu$) : $I_\mu = [q_i q_j{}^i q_k q_\ell]$

c(i) : 1次元配列化されたHartree-Fock方程式の固有ベクトル

gab : $\Gamma_{ab}$

lmo(i) : $q_i$

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 intci1 |

機能

分子積分を求める。(オリジナルのサブルーチン)

処理

理論仕様書の(A4.2)を以下の場合につて、評価する。

1. [i'a'|j'b']を求め、配列要素ovov(n)に格納する。
2. [i'j'|a'b']を求め、配列要素oovv(n)に格納する。

ここで、以下の条件が存在する。

$1 \leqq i \leqq n_{ocs}$, $1 \leqq j \leqq i$, $1 \leqq a \leqq n_{vcs}$, $1 \leqq b \leqq b_{max}$, $b_{max} = a\ (i=j)$, $= n_{vcs}\ (i \neq j)$

また、理論仕様書の(A4.2)のr, tは、サブルーチン内でq, pとされている。

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 mkcibs |

機能

SDCIの基底系$\Phi_\omega$を定義する。(新規サブルーチン)

処理

理論仕様書のA.2の$\Phi_\omega$の定義に必要な以下の属性データの設定を行う。

momega(i) : $\Omega_i$ ($1 \leqq i \leqq 7$)

mcio($\omega$) : $\Phi_\omega$の型

= 1 Hartree-Fockの基底状態 (今後0で表わす。)
2 1電子励起状態 (s)
3 2電子励起状態 type 1 (1)
4 2電子励起状態 type 2 (2)
5 2電子励起状態 type 3 (3)
6 2電子励起状態 type 4 (4)
7 2電子励起状態 type 5 (5)

lci1($\omega$), lci2($\omega$) : $\Phi_\omega$の遷移元の番号

lci3($\omega$), lci4($\omega$) : $\Phi_\omega$の遷移先の番号

$\Phi_\omega$の各型に対してlci1, lci2, lci3, lci4は以下の値を持つ。

| 型 | lci1等の値 |
|---|---|
| 0 | lci1=lci2=lci3=lci4=0 |
| s | $1 \leqq$ lci1 $\leqq n_{ocs}$, lci2=0, $1 \leqq$ lci3 $\leqq n_{vcs}$, lci4=0 |
| 1 | $1 \leqq$ lci1 $\leqq n_{ocs}$, lci2=lci1, $1 \leqq$ lci3 $\leqq n_{vcs}$, lci4=lci3 |
| 2 | $2 \leqq$ lci1 $\leqq n_{ocs}$, $1 \leqq$ lci2 $\leqq$ lci1-1, $1 \leqq$ lci3 $\leqq n_{vcs}$, lci4=lci3 |
| 3 | $1 \leqq$ lci1 $\leqq n_{ocs}$, lci2=lci1, $2 \leqq$ lci3 $\leqq n_{vcs}$, $1 \leqq$ lci4 $\leqq$ lci3-1 |
| 4 | $2 \leqq$ lci1 $\leqq n_{ocs}$, $1 \leqq$ lci2 $\leqq$ lci1-1, $2 \leqq$ lci3 $\leqq n_{vcs}$, $1 \leqq$ lci4 $\leqq$ lci3-1 |
| 5 | $2 \leqq$ lci1 $\leqq n_{ocs}$, $1 \leqq$ lci2 $\leqq$ lci1-1, $2 \leqq$ lci3 $\leqq n_{vcs}$, $1 \leqq$ lci4 $\leqq$ lci3-1 |



| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 moinfo |

機能

遷移に関連するMOの指定

処理

以下のデータの設定を行う。

nocs=iopci(2) : 遷移元の被占軌道のMOの総数 ($n_{ocs}$)

nvcs=iopci(3) : 遷移先の空軌道のMOの総数 ($n_{vcs}$)

ihomo : HOMOの軌道通し番号

iomo(i) : i番目に指定された遷移元の被占軌道のMOの通し番号($1 \leqq i \leqq n_{ocs}$)

ivmo(i) : i番目に指定された遷移先の空軌道のMOの通し番号($1 \leqq i \leqq n_{vcs}$)

lmo(j) : j番目に指定された遷移に関連するMOの通し番号($1 \leqq j \leqq n_{ocs}+n_{vcs}$)

1. iopci(2)≠0の場合

遷移元の被占軌道のMOの通し番号nは、次の範囲内にある。

ihomo - $n_{ocs}$+1 $\leqq$ n $\leqq$ ihomo

この範囲内のMOに対してあらたに割り当てられた番号をiとした場合の上記nとの対応をn=iomo(i)=lmo(j)とする。($1 \leqq i \leqq n_{ocs}$, j=i)

遷移先の空軌道のMOの通し番号nは、次の範囲内にある。

ihomo +1 $\leqq$ n $\leqq$ ihomo + $n_{vcs}$

この範囲内のMOに対してあらたに割り当てられた番号をiとした場合の上記nとの対応をn=ivmo(i)=lmo(j)とする。($1 \leqq i \leqq n_{vcs}$, j=i+$n_{ocs}$)

2. iopci(2)=0の場合

nwrds : 遷移に関するMOの総数の指定値 (入力)

ival(j) : 指定した遷移の番号をj($1 \leqq j \leqq$ nwrds)とした場合のMOの通し番号 (入力)

ival(j)の値に対するiomo, ivmoの設定を以下の様に行う。

a) ival(j)$\leqq$ihomoの場合

ival(j)を遷移元の被占軌道のMOの通し番号nとし、n=iomo(i)=ival(j)とする。この場合のあらたに割り当てられた番号をiとする。

b) ihomo<ival(j)の場合

ival(j)を遷移先の空軌道のMOの通し番号nとし、n=ivmo(i)=ival(j)とする。この場合のあらたに割り当てられた番号をiとする。

c) iomo, ivmo の重複のチェックを行い、iomo, ivmoを再定義する。iomo, ivmoのそれぞれの要素の総数を$n_{ocs}$, $n_{vcs}$とする。lmo(j) ($1 \leqq j \leqq n_{ocs}+n_{vcs}$)を設定する。

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 molint |

機能

CI法の基底の定義と分子積分の評価

処理

1. Hartree-Fock方程式の固有ベクトルの2次元配列cの1次元化 (packmo)
2. 遷移に関連するMOの指定 (moinfo)
3. isdci0 = 0の場合
   オリジナルのサブルーチンにより、分子積分を求める。(intci1)
4. isdci0 ≧ 1の場合
   1) 新規サブルーチンでSDCIの基底系 $\Phi_{\omega}$ を定義する。(mkcibs)
   2) 新規サブルーチンで分子積分を求める。(intci0)

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 ovlp |

機能

理論仕様書(1.13)の重なり積分 $S_{ij}$ を解析的に評価する。

処理

理論仕様書(A3.1), (A3.2), 図 A3.1の方法で $S_{ij}$ を配列要素s(n)に格納する。ただし、AOiの局在する原子とAOjの局在する原子の距離が100Åより大きければ、$S_{ij}$ をゼロとする。

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 |
| | | | rhfclo |

機能

SCF反復計算によりHartree-Fock方程式の固有状態となるMOを計算する。

処理

1. 結合次数$P_{tu}=2\sum_{1\leq j\leq n}C_{jt}C_{ju}$を求め、(A3.1)と図 A3.1の方法で配列要素 d (n,ip0)に格納する。ここで、ip0は現反復計算をさすポインターである。(getd)
2. 現反復計算と前反復計算の結合次数の差のベクトルのノルムを計算し、反復計算の誤差とする。この誤差が収束許容誤差より小さければ、論理変数logconを真とする。(conclo)
3. 理論仕様書の(2.11),(2,13)のFock行列要素を求め、(A3.1)と図 A3.1の方法で配列要素f(n)に格納する。(getf)
4. Hartree-Fockの基底状態における電子系のエネルギー期待値を計算する。(scfeng)
5. 論理変数logconが真ならば、SCF反復ループをぬける。
6. Fock行列の対角化を行い、固有ベクトルを現反復計算のMOとする。(diag1)
7. 1.に戻る。

以上の処理により、MOと固有値は、以下の様に格納される。
c(i,j) : MO "j" の第i成分　　(あとでサブルーチンpackmoで1次元化される。)
eig(j) : MO "j" のエネルギー順位

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 |
| | | | sci |

機能

CI法の固有状態を求める。(オリジナルのSCI)

処理

1. 理論仕様書(4.10)の行列要素$H_{\omega\tau}$を求め、(A3.7)の方法で1次元配列要素cimat(i)に格納する。Hartree-Fockの基底状態と1電子励起状態が混合しないので、(4.10)の固有値問題は、1電子励起状態の基底系のなす部分空間内の固有値問題となる。よって、(4.11)の$H_{\omega\tau}$は、(A2.10)と(A2.22)のみ考慮すればよい。
2. 理論仕様書(4.10)の固有値問題を解き、SCIの固有状態と固有値を求める。(diag1)

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 sdci1 |

機能

CI法の固有状態を求める。(新規のSDCI)

処理

1. 理論仕様書(4.10)の行列要素$H_{\omega\tau}$を求め、(A3.7)の方法で1次元配列要素cimat(i)に格納する†。Hartree-Fockの基底状態と2電子励起状態が混合するので、(4.10)の固有値問題は、Hartree-Fockの基底状態と励起状態を合わせた基底系に関する問題となる。
2. 理論仕様書(4.10)の固有値問題を解き、SDCIの固有状態と固有値を求める。(diag1)

†基底$\Phi_\omega$は、mkcibsのモジュール説明に示した型に分割される。それにしたがって行列要素$H_{\omega\tau}$もブロック化される。このブロックを$[H]_{AB}$(A,B=0,s,1,2,3,4,5)と表わす。例えば、$[H]_{2s}$は、$\omega$と$\tau$が以下の範囲の行列要素$H_{\omega\tau}$からなるブロックである。

$\Omega_3+1 \leqq \omega \leqq \Omega_4,\ \Omega_1+1 \leqq \tau \leqq \Omega_2$

理論仕様書のA.2で与えられた行列要素$H_{\omega\tau}$は、ブロックごとにサブルーチンcimtss, cimt10, ....., cimt55で評価される。例えば、$[H]_{42}$は、cimt42で評価される。なお、フラッグisdci0の値によって、全体行列は以下のブロックから構成される。

a) isdci0 = 1 (case1)

0.0

$[H]_{s0}[H]_{ss}$

$[H]_{10}[H]_{1s}[H]_{11}$

b) isdci0 = 2 (case2)

0.0

$[H]_{s0}[H]_{ss}$

$[H]_{10}[H]_{1s}[H]_{11}$

$[H]_{30}[H]_{3s}[H]_{31}[H]_{33}$

c) isdci0 = 3 (case3)

0.0

$[H]_{s0}[H]_{ss}$

$[H]_{10}[H]_{1s}[H]_{11}$

$[H]_{20}[H]_{2s}[H]_{21}[H]_{22}$

$[H]_{30}[H]_{3s}[H]_{31}[H]_{32}[H]_{33}$

$[H]_{40}[H]_{4s}[H]_{41}[H]_{42}[H]_{43}[H]_{44}$

$[H]_{50}[H]_{5s}[H]_{51}[H]_{52}[H]_{53}[H]_{54}[H]_{55}$

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 setcom |

機能

定数の設定

処理

定数の設定

pcon(1)=pi(円周率), pcon(2)=toang(BohrからÅへの変換の係数),
pcon(3)=autoev(HartreeからeVへの変換の係数), pcon(4)=planck(Planckt定数),
pcon(5)=slight(光速), pcon(6)=evtoer(eVからergへの変換の係数),
pcon(7)=evtokc(eVからKcalへの変換の係数),
orbtyp(i): 軌道の型((A5.32)のk),
elmnt(i), elmnt2(i):元素記号,
atmass(i): 原子の質量数

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 sginfo |

機能

AOの定義

処理

以下のデータを設定する。

zv(i):原子iの価電子数

nlbas(i):原子iに局在するAOの通し番号の最小値

nubas(i):原子iに局在するAOの通し番号の最大値

nibas(i):原子iに局在するAOの総数

ibtoa(j):通し番号jのAOの属す原子の通し番号

nbasis : AOの総数

ne : 価電子の総数

nae : 上向きスピンの価電子数または、被占軌道の総数

nbe : 下向きスピンの価電子数

nc(i,1) : 原子番号iの原子の価電子のs軌道, p軌道の主量子数 n

nc(i,2) : 原子番号iの原子の価電子のd軌道の主量子数 n

lc(k) : k番の軌道の型の方位量子数 l

mc(k) : k番の軌道の型の磁気量子数 m

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 sgint |

機能

原子積分の評価

処理

1. AOの定義 (sginfo)
2. 重なり積分$S_{rs}$の評価 (ovlp)
3. Coulomn積分$\Gamma_a$,$\Gamma_{ab}$の評価 (eri)

| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 |
| | | | tmsdbs |

機能

理論仕様書(A5.4)の$M_{\omega\tau}$を(A5.6), (A5.7), .... , (A5.29)で評価する。

処理

理論仕様書の(A5.4)で与えられた$M_{\omega\tau}$は、行列要素$H_{\omega\tau}$と同様にブロックごとにサブルーチンtmbss, tmb10, ....., tmb55で評価される。ブロックを$[M]_{AB}$と表わす。例えば、$[M]_{42}$は、tmb42で評価される。なお、フラッグisdci0の値によって、全体の遷移モーメントは、以下のブロックから構成される。

a) isdci0 = 1 (case1)

$L$

$[M]_{s0}[M]_{ss}$

$[M]_{10}[M]_{1s}[M]_{11}$

b) isdci0 = 2 (case2)

$L$

$[M]_{s0}[M]_{ss}$

$[M]_{10}[M]_{1s}[M]_{11}$

$[M]_{30}[M]_{3s}[M]_{31}\,[M]_{33}$

c) isdci0 = 3 (case3)

$L$

$[M]_{s0}[M]_{ss}$

$[M]_{10}[M]_{1s}[M]_{11}$

$[M]_{20}[M]_{2s}[M]_{21}\,[M]_{22}$

$[M]_{30}[M]_{3s}[M]_{31}\,[M]_{32}[M]_{33}$

$[M]_{40}[M]_{4s}[M]_{41}\,[M]_{42}[M]_{43}[M]_{44}$

$[M]_{50}[M]_{5s}[M]_{51}\,[M]_{52}[M]_{53}[M]_{54}[M]_{55}$



| モジュール説明 | Ver. | Date | page |
|---|---|---|---|
| | | | モジュール名 |
| | | | tmsdci |

機能

遷移モーメントのCI法の固有状態間の要素を求める。(新規)

処理

1. 理論仕様書(4.10)のSDCIの固有ベクトルの格納部分(civec)を、以下の様に移動する。その際、1次元配列に移動する。(packci)

```
common /work01/ dummy1(mciint*2),civec(mcicsf,mcicsf),
                dummy2(1)
                    ↓
common /work01/ civec(mcicsf**2), dummy1(1)
```

2. iopdip(1) = 1 ならばZDO近似で理論仕様書(A5.30)を評価する。(dipzdo)
   iopdip(1) ≠ 1 ならば解析的に理論仕様書(A5.31)を評価する。(dip1cn)
3. 理論仕様書(A5.6)の空間的各成分をtlx, tly, tlz に代入する。(現在はサブルーチンgprptyで計算された値を代入しているが、新規サブルーチンℓzdo00または、ℓ1cn00でも計算できる。)
4. 理論仕様書(A5.4)の$M_{\omega\tau}$を(A5.6), (A5.7), .... , (A5.29)で評価する。(tmsdbs)
5. 理論仕様書(A5.3)の $\tilde{M}_{\lambda\lambda'}$ を評価する。(tmsdst)

| モジュール説明 | Ver. | Date | | | page |
|---|---|---|---|---|---|
| | | | | | モジュール名 |
| | | | | | tmsdst |

機能

理論仕様書(A5.3)の $\tilde{M}_{\lambda\lambda'}$ を評価する。

処理

理論仕様書(A5.3)を評価し、(A3.8)の方法で1次元配列 tmsx, tmsy, tmszに格納する。

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 basinf |

内容

原子軌道(AO)の情報

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| nlbas | i*4 | maxat | |
| nubas | i*4 | maxat | nlbas(i):<br>原子iに局在するAOの通し番号の最小値 |
| nibas | i*4 | maxat | nubas(i):<br>原子iに局在するAOの通し番号の最大値 |
| ibtoa | i*4 | maxbas | nibas(i):<br>原子iに局在するAOの総数<br>ibtoa(j):<br>通し番号jのAOの属す原子の通し番号<br><br>例：<br>nを原子iのAOとすると、以下の関係が成立する。<br>nlbas(i)≦n≦nubas(i),<br>nibas(i) = nubas(i) - nlbas(i) + 1<br>i = ibtoa(n) |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 energy |

内容

Hartree-Fockの基底状態における電子系のエネルギー

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| energy | i*4 | | $E_0=\langle\Psi_0\vert H\vert\Psi_0\rangle$ |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 mol |

内容

分子の構成に関連する情報

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| coord | r*8 | 3,maxat | 原子核の位置ベクトルの成分 |
| ian | i*4 | maxat | ian(i) : 原子iの原子番号 |
| natoms | i*4 | | 分子を構成する原子の総数 |
| icharg | i*4 | | 分子の荷数 |
| multip | i*4 | | |
| nae | i*4 | | 上向きスピンの価電子数または、被占軌道の総数 |
| nbe | i*4 | | 下向きスピンの価電子数 |
| ne | i*4 | | 価電子の総数 |
| nbasis | i*4 | | AOの総数 |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 qnoinf |

内容

原子軌道(AO)の情報

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| nc | i*4 | maxian, 2 | nc(i,1) : 原子番号iの原子の価電子のs軌道, p軌道の主量子数 n<br>nc(i,2) : 原子番号iの原子の価電子のd軌道の主量子数 n |
| lc | i*4 | 9 | lc(k) : k番の軌道の型の方位量子数 l |
| mc | i*4 | 9 | mc(k) : k番の軌道の型の磁気量子数 m |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 sdci00 |

内容

新規SDCI用のデータ

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| cimat | r*8 | msdcim | 理論仕様書(4.11), A.2のSDCIのHamiltonianの行列要素 $<\Phi_{\omega}\|H\|\Phi_{\tau}>$ 理論仕様書(A3.7)の方法により格納 |
| h000 | | | 未使用 |
| ncibas | i*4 | | SDCIの基底の総数 |
| ncimat | i*4 | | 行列要素 $<\Phi_{\omega}\|H\|\Phi_{\tau}>$の総数 =ncibas( ncibas + 1 )/2 |
| isdci0 | i*4 | | = 0 オリジナルのmosfのSCIを実行する。<br>1 type 1を含むSDCIを実行する(case1)<br>2 type 1とtype 3を含むSDCIを実行する。(case2)<br>3 全二電子励起状態のタイプ (type 1, type 2, type 3, type 4, type 5) を含むSDCIを実行する。(case3) |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 sdci01 |

内容

新規SDCI用のデータ

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| mcio | i*4 | msdcib | モジュール説明の"mkcibs"参照 |
| momega | i*4 | 7 | |
| lci1 | i*4 | msdcib | |
| lci2 | i*4 | msdcib | |
| lci3 | i*4 | msdcib | |
| lci4 | i*4 | msdcib | |
| ici1 | i*4 | mcipck | 理論仕様書(A3.6)の $\nu$ (i,j)の i |
| ici2 | i*4 | mcipck | 理論仕様書(A3.6)の $\nu$ (i,j)の j |
| nsdci | i*4 | | 遷移に関連するMOの総数 $n_{sdci} = n_{ocs} + n_{vcs}$ |
| numax | i*4 | | 理論仕様書(A3.6)の $\nu$ の総数 $\nu_{max}=n_{sdci}(n_{sdci}+1)/2$ |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 sdci02 |

内容

新規SDCI用のデータ

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| fockmt | r*8 | mbpck | 理論仕様書(2.11), (2.13)の$F_{rs}$ |
| ediff | r*8 | mcipck | 理論仕様書(A2.10)の$E_{i->k}$ |
| gintgl | r*8 | mumax0 | 理論仕様書(A3.6)の分子積分 |
| mumax | i*4 | | gintglの要素の総数 $\mu_{max}$ |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 tmomnt |

内容

遷移モーメント

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| tlx | r*8 | | 双極子モーメントのx成分 |
| tly | r*8 | | 双極子モーメントのy成分 |
| tlz | r*8 | | 双極子モーメントのx成分 |
| tmbx | r*8 | msdcim | 理論仕様書の(A5.4)のx成分 |
| tmby | r*8 | msdcim | 理論仕様書の(A5.4)のy成分 |
| tmbz | r*8 | msdcim | 理論仕様書の(A5.4)のz成分 |
| tmsx | r*8 | msdcim | 理論仕様書の(A5.3)のx成分 |
| tmsy | r*8 | msdcim | 理論仕様書の(A5.3)のy成分 |
| tmsz | r*8 | msdcim | 理論仕様書の(A5.3)のz成分 |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 velec |

内容

価電子の情報

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| zv | r*8 | maxat | zv(i) : 原子iの価電子数 |

## コモン 変数説明

| Ver. | Date | page |
|---|---|---|
| | | ラベル名 window |

内容

遷移に関連するMOの情報

| 変数名 | 型 | 寸法 | 意味 |
|---|---|---|---|
| nocs | i*4 | | 遷移元の被占軌道のMOの総数 ($n_{ocs}$) |
| nvcs | i*4 | | 遷移先の空軌道のMOの総数 ($n_{vcs}$) |
| iomo | i*4 | | iomo(i) : i番目に指定された遷移元の被占軌道のMOの通し番号($1 \leq i \leq n_{ocs}$) |
| ivmo | i*4 | | ivmo(i) : i番目に指定された遷移先の空軌道のMOの通し番号($1 \leq i \leq n_{vcs}$) |
| lmo | i*4 | | lmo(j) : j番目に指定された遷移に関連するMOの通し番号($1 \leq j \leq n_{ocs}+n_{vcs}$) |
| ncsfs | i*4 | | 1電子励起状態の総数 |

入力ファイル：aaa.dat

```
$sdci
  isdci0 = 3,
$end
ci=read
cndo/s coord=cart nobeta g=ok scfcrt=0.000001

butadiene 2*2

C     -1.823300     0.118400     0.000000
C     -0.601600    -0.410300     0.000000
C      0.601600     0.410300     0.000000
C      1.823300    -0.118400     0.000000
H      2.722300     0.490100     0.000000
H      2.009100    -1.189100     0.000000
H     -2.722300    -0.490100     0.000000
H     -2.009100     1.189100     0.000000
H      0.457900     1.498200     0.000000
H     -0.457900    -1.498200     0.000000

10 11    12 13
```

} 全ての励起タイプを含めたSDCIを指定

→ CIを発生させる軌道の番号を最後に読み込む

→ CNDO/S法で，2中心電子間反発積分をOhno-Klopmanの式で計算

コメント行

原子の元素記号，X，Y，Z 座標（Å単位）

→ CI配置を発生させる軌道を指定

```
#-----------------------------------------------------------------------
#   CNDO/S-SDCI  < 5*5 CI                  Execute by K.Ohtawara
#   ATR Optical and Radio Communication Research Labs.

if ($#argv == 0) then
 echo "Usage: run < file_name >"
 exit 1
endif
if (!(-e $1:r.dat)) then
 echo "------------   ***.dat  file not found !!! -----------"
 exit 0
endif
echo "=== CNDO/S-SDCI started.  Calculating ...."

echo "----- Data file name -----">$$
ls -l $1:r.dat >>$$
echo "----- Started Time -----">>$$
date >>$$
date
cp $1:r.dat fort.5
./mosf66
echo "----- Ended time -----">>$$
date >>$$
date
head -3 runmos >> $$

if(-e fort.16) then
  cat $$ >>fort.16
  mv fort.16 $1:r.log
endif
if(-e fort.20) then
  mv fort.20 $1:r.tr
endif

rm fort.*
rm $$

ls -l $1:r.*
```

実行用シェルスクリプト

(runmos)

1) データファイル名 aaa.datに対し
runmos aaa で実行

2) 出力ファイルは 2つ生成する

aaa.log 分子軌道など

aaa.tr 遷移モーメント行列

aaa.trは、続くγ計算の入力データとして使用する。

出力ファイル：aaa.log

```
<<<<<< Enter JCtrl >>>>>>
$SDCI
ISDCI0   =          3
$END
-------------------------------------------------------
ci=read
cndo/s coord=cart mocoef=all nobeta g=ok scfcrt=0.000001
-------------------------------------------------------
CNDO/S calculation
<<<<<< JCtrl.  All done. CPU-Time =     0.39 Sec. Total =  00H 00M 00S. >>>>>>
<<<<<< Enter MInput >>>>>>
-------------
butadiene 2*2
-------------
Input geometry ...
-----------------------------------------------------------------------------
                Cartesian(Ang)                    Internal(Ang and Deg)
 Cn  At     x        y        z        Length   Alpha     Beta     NA   NB   NC
-----------------------------------------------------------------------------
  1  C   -1.8233   0.1184   0.0000    0.0000    0.000     0.000     0    0    0
  2  C   -0.6016  -0.4103   0.0000    1.3312    0.000     0.000     1    0    0
  3  C    0.6016   0.4103   0.0000    1.4564  122.305     0.000     2    1    0
  4  C    1.8233  -0.1184   0.0000    1.3312  122.305   180.000     3    2    1
  5  H    2.7223   0.4901   0.0000    1.0856  122.506   180.000     4    3    2
  6  H    2.0091  -1.1891   0.0000    1.0867  123.245     0.000     4    3    2
  7  H   -2.7223  -0.4901   0.0000    1.0856  122.506   180.000     1    2    3
  8  H   -2.0091   1.1891   0.0000    1.0867  123.245     0.000     1    2    3
  9  H    0.4579   1.4982   0.0000    1.0973  116.770     0.000     3    2    1
 10  H   -0.4579  -1.4982   0.0000    1.0973  120.926   180.000     2    1    3
-----------------------------------------------------------------------------
Translate molecule ... The origin is the center of nuclear charges.
Standard geometry ...
-----------------------------------------------------------------------------
 Center      Atomic                    Coordinates (Angstroms)
 number      number              x                y                z
-----------------------------------------------------------------------------
    1           6          -1.823300000      0.118400000      0.000000000
    2           6          -0.601600000     -0.410300000      0.000000000
    3           6           0.601600000      0.410300000      0.000000000
    4           6           1.823300000     -0.118400000      0.000000000
    5           1           2.722300000      0.490100000      0.000000000
    6           1           2.009100000     -1.189100000      0.000000000
    7           1          -2.722300000     -0.490100000      0.000000000
    8           1          -2.009100000      1.189100000      0.000000000
    9           1           0.457900000      1.498200000      0.000000000
   10           1          -0.457900000     -1.498200000      0.000000000
-----------------------------------------------------------------------------
Chemical formula ... C4H6
Molecular weight ...   54.0916
<<<<<< MInput. All done. CPU-Time =     0.10 Sec. Total =  00H 00M 00S. >>>>>>
<<<<<< Enter SGInt >>>>>>
   22 Basis functions.        22 Valence electrons.
   11 Alpha electrons.        11 Beta electrons.
Scaling factors of overlap integrals ...
p-sigma = 1.00000  p-pi = 0.58500  d =  0.30000
2 Center ERI, Gamma ... Evaluated by Ohno-Klopman formula.
<<<<<< SGInt.  All done. CPU-Time =     0.05 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< Enter Guess >>>>>>
Initial Huckel guess.
Matrix diagonalization ... Using DiagD of Gaussian 80.
<<<<<< Guess.  All done. CPU-Time =     0.05 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< Enter RHFClo >>>>>>
RHF closed shell SCF ... Maximum cycles =  100
SCF criterion ... Convergence on density matrix = 1.0000D-06
-----------------------------------------------------------------------------
 Iteration    Electronic energy          Convergence              Extrapolation
                (Hartree)            Energy       Density
-----------------------------------------------------------------------------
     1          -72.560580167
     2          -72.563882522       3.3024D-03   4.3482D-03
     3          -72.564346264       4.6374D-04   1.6243D-03
     4          -72.564426445       8.0181D-05   6.9429D-04
     5          -72.564441241       1.4796D-05   2.9909D-04
     6          -72.564444106       2.8651D-06   1.3385D-04
                -72.564403704  ...   Non-variational value        Spiral (4-pt)
     7          -72.564444770
     8          -72.564444840       6.9999D-08   2.0792D-05
     9          -72.564444860       2.0492D-08   1.2764D-05
    10          -72.564444867       6.4211D-09   6.9301D-06
    11          -72.564444869       2.0353D-09   3.9482D-06
                -72.564444874  ...   Non-variational value        Aitken (3-pt)
    12          -72.564444870
    13          -72.564444870       1.4211D-13   3.4003D-08
-----------------------------------------------------------------------------
Density converged to 3.40029D-08  ... SCF criterion satisfied.
 &&&& ENERGY, h000 =    -72.5644448695406        -70.0095167925614
<<<<<< RHFClo. All done. CPU-Time =     0.43 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< Enter GPrpty >>>>>>
Energy table for ground state
-----------------------------------------------------------------------------
                                Hartree          Electron volt       kcal/mol
-----------------------------------------------------------------------------
 Total                      -16.206021502      -440.965845077  -0.101691134D+05
 Electronic                 -72.564444870     -1974.478544900  -0.455334497D+05
 Nuclear repulsion           56.358423367      1533.512699824   0.353643364D+05
-----------------------------------------------------------------------------
Highest Occupied  Molecular Orbital (HOMO) ...  11
Lowest Unoccupied Molecular Orbital (LUMO) ...  12
Canonical orbital energies (Unit : eV)
 Occupied MOs ...
   1  -40.1633     2  -34.3506     3  -26.7862     4  -25.6529     5  -20.3988
   6  -19.8749     7  -16.1210     8  -15.3511     9  -14.2050    10  -14.0238
  11  -10.5728
 Virtual MOs ...
  12    0.1522    13    2.3137    14    4.4848    15    5.3672    16    6.0781
  17    6.7752    18    7.7234    19    7.9277    20    8.5347    21    9.6267
  22   10.5359
==============================
Molecular orbital coefficients
==============================
                                  1          2          3          4          5
 Eigenvalues ===>            -1.476049  -1.262426  -0.984424  -0.942775  -0.749681
  1   1  C      s            -0.348977   0.491358   0.376721  -0.120711  -0.047386
  2             px            -0.144542   0.060248   0.244271   0.057417  -0.027419
  3             py             0.027903  -0.046406   0.007280  -0.260225  -0.428288
  4             pz             0.000000   0.000000   0.000000   0.000000   0.000000
  5   2  C      s            -0.518603   0.273964  -0.241999   0.280872   0.052979
  6             px            -0.042495  -0.270573  -0.278417   0.013395  -0.210594
  7             py            -0.089436   0.037764  -0.034983  -0.355574  -0.282100
  8             pz             0.000000   0.000000   0.000000   0.000000   0.000000
  9   3  C      s            -0.518603  -0.273964  -0.241999  -0.280872   0.052979
 10             px             0.042495  -0.270573   0.278417   0.013395   0.210594
 11             py             0.089436   0.037764   0.034983  -0.355574   0.282100
 12             pz             0.000000   0.000000   0.000000   0.000000   0.000000
 13   4  C      s            -0.348977  -0.491358   0.376721   0.120711  -0.047386
 14             px             0.144542   0.060248   0.244271   0.057417   0.027419
 15             py            -0.027903  -0.046406  -0.007280  -0.260225   0.428288
 16             pz             0.000000   0.000000   0.000000   0.000000   0.000000
 17   5  H      s            -0.116054  -0.227616   0.316211  -0.004218   0.176567
 18   6  H      s            -0.145049  -0.208529   0.218742   0.245237  -0.335012
 19   7  H      s            -0.116054   0.227616   0.316211   0.004218   0.176567
 20   8  H      s            -0.145049   0.208529   0.218742  -0.245237  -0.335012
 21   9  H      s            -0.208046  -0.088309  -0.114978  -0.385688   0.208390
 22  10  H      s            -0.208046   0.088309  -0.114978   0.385688   0.208390
                                  6          7          8          9         10
 Eigenvalues ===>            -0.730425  -0.592466  -0.564172  -0.522052  -0.515392
  1   1  C      s            -0.074994  -0.007024  -0.100051  -0.032511   0.000000
  2             px             0.488065   0.017890  -0.327176  -0.259007   0.000000
  3             py             0.023805   0.426938   0.172639  -0.194188   0.000000
  4             pz             0.000000   0.000000   0.000000   0.000000  -0.429580
  5   2  C      s             0.168269   0.079107  -0.006021   0.070176   0.000000
  6             px            -0.226887   0.127072   0.310245   0.365923   0.000000
  7             py             0.181476  -0.225751  -0.324765   0.301707   0.000000
  8             pz             0.000000   0.000000   0.000000   0.000000  -0.561659
  9   3  C      s            -0.168269  -0.079107  -0.006021   0.070176   0.000000
 10             px            -0.226887   0.127072  -0.310245  -0.365923   0.000000
```

SCF計算

scfcrtより小さくなり収束

全エネルギー

各軌道のエネルギー

入力座標を分子のchargeの中心に移動

分子軌道

x-y平面にある分子に対して、2Pz軌道の係数の大きい、10,11,12,13が π軌道と分かる

```
11              py     0.181476  -0.225751   0.324765  -0.301707   0.000000
12              pz     0.000000   0.000000   0.000000   0.000000  -0.561659
13    4  C      s     0.074994   0.007024  -0.100051  -0.032511   0.000000
14              px     0.488065   0.017890   0.327176   0.259007   0.000000
15              py     0.023805   0.426938  -0.172639   0.194188   0.000000
16              pz     0.000000   0.000000   0.000000   0.000000  -0.429580
17    5  H      s     0.361526   0.233827   0.105061   0.296248   0.000000
18    6  H      s     0.080974  -0.359928   0.162928  -0.175714   0.000000
19    7  H      s    -0.361526  -0.233827   0.105061   0.296248   0.000000
20    8  H      s    -0.080974   0.359928   0.162928  -0.175714   0.000000
21    9  H      s     0.074952  -0.244464   0.337340  -0.213683   0.000000
22   10  H      s    -0.074952   0.244464   0.337340  -0.213683   0.000000
                        11         12         13         14         15
Eigenvalues ===>    -0.388564   0.005594   0.085031   0.164822   0.197250
 1    1  C      s     0.000000   0.000000   0.000000  -0.339515   0.433201
 2              px     0.000000   0.000000   0.000000  -0.057330  -0.038213
 3              py     0.000000   0.000000   0.000000   0.022071   0.041121
 4              pz    -0.570835  -0.561659  -0.417309   0.000000   0.000000
 5    2  C      s     0.000000   0.000000   0.000000   0.470132  -0.236018
 6              px     0.000000   0.000000   0.000000   0.070558   0.185852
 7              py     0.000000   0.000000   0.000000   0.035303   0.067628
 8              pz    -0.417309   0.429580   0.570835   0.000000   0.000000
 9    3  C      s     0.000000   0.000000   0.000000  -0.470132  -0.236018
10              px     0.000000   0.000000   0.000000   0.070558  -0.185852
11              py     0.000000   0.000000   0.000000   0.035303  -0.067628
12              pz     0.417309   0.429580  -0.570835   0.000000   0.000000
13    4  C      s     0.000000   0.000000   0.000000   0.339515   0.433201
14              px     0.000000   0.000000   0.000000  -0.057330   0.038213
15              py     0.000000   0.000000   0.000000   0.022071  -0.041121
16              pz     0.570835  -0.561659   0.417309   0.000000   0.000000
17    5  H      s     0.000000   0.000000   0.000000  -0.203083  -0.254115
18    6  H      s     0.000000   0.000000   0.000000  -0.185594  -0.331572
19    7  H      s     0.000000   0.000000   0.000000   0.203083  -0.254115
20    8  H      s     0.000000   0.000000   0.000000   0.185594  -0.331572
21    9  H      s     0.000000   0.000000   0.000000   0.279318   0.199630
22   10  H      s     0.000000   0.000000   0.000000  -0.279318   0.199630
                        16         17         18         19         20
Eigenvalues ===>     0.223377   0.248996   0.283845   0.291352   0.313659
 1    1  C      s    -0.321440  -0.144248  -0.008856   0.000616  -0.007022
 2              px     0.197401   0.228259  -0.112640   0.170020  -0.204531
 3              py    -0.067112  -0.011653  -0.444547   0.461178  -0.134825
 4              pz     0.000000   0.000000   0.000000   0.000000   0.000000
 5    2  C      s    -0.246001  -0.268332  -0.075272   0.057352   0.148466
 6              px    -0.222683  -0.017741  -0.078333  -0.035609   0.193814
 7              py     0.107539   0.342821  -0.166061  -0.116666   0.497813
 8              pz     0.000000   0.000000   0.000000   0.000000   0.000000
 9    3  C      s     0.246001  -0.268332   0.075272   0.057352  -0.148466
10              px    -0.222683   0.017741  -0.078333   0.035609   0.193814
11              py     0.107539  -0.342821  -0.166061   0.116666   0.497813
12              pz     0.000000   0.000000   0.000000   0.000000   0.000000
13    4  C      s     0.321440  -0.144248   0.008856   0.000616   0.007022
14              px     0.197401  -0.228259  -0.112640  -0.170020  -0.204531
15              py    -0.067112   0.011653  -0.444547  -0.461178  -0.134825
16              pz     0.000000   0.000000   0.000000   0.000000   0.000000
17    5  H      s    -0.291300   0.255129   0.302760   0.331968   0.165148
18    6  H      s    -0.265277   0.083799  -0.372212  -0.350822  -0.056921
19    7  H      s     0.291300   0.255129  -0.302760   0.331968  -0.165148
20    8  H      s     0.265277   0.083799   0.372212  -0.350822   0.056921
21    9  H      s    -0.276240   0.406200   0.141520  -0.083428  -0.319377
22   10  H      s     0.276240   0.406200  -0.141520  -0.083428   0.319377
                        21         22
Eigenvalues ===>     0.353793   0.387208
 1    1  C      s     0.120469   0.140002
 2              px     0.402604   0.397204
 3              py    -0.183611  -0.162777
 4              pz     0.000000   0.000000
 5    2  C      s    -0.183642  -0.047263
 6              px     0.331744   0.510607
 7              py    -0.281046  -0.006254
 8              pz     0.000000   0.000000
 9    3  C      s    -0.183642   0.047263
10              px    -0.331744   0.510607
11              py     0.281046  -0.006254
```



各原子上のcharge

```
12              pz     0.000000   0.000000
13    4  C      s     0.120469  -0.140002
14              px    -0.402604   0.397204
15              py     0.183611  -0.162777
16              pz     0.000000   0.000000
17    5  H      s     0.129219  -0.133269
18    6  H      s     0.146359  -0.116921
19    7  H      s     0.129219   0.133269
20    8  H      s     0.146359   0.116921
21    9  H      s    -0.169708   0.041335
22   10  H      s    -0.169708  -0.041335
Total atomic charges ...
  1  C   -0.10894     2  C   -0.01060     3  C   -0.01060     4  C   -0.10894
  5  H    0.03872     6  H    0.04346     7  H    0.03872     8  H    0.04346
  9  H    0.03735    10  H    0.03735
Dipole moment (Unit : Debye) ...
                                  x           y           z          Total
  Total atomic charges        0.0000      0.0000      0.0000      0.0000
  sp atomic polarization      0.0000      0.0000      0.0000      0.0000
  pd atomic polarization      0.0000      0.0000      0.0000      0.0000
  Total                       0.0000      0.0000      0.0000      0.0000
<<<<<< GPrpty. All done. CPU-Time =     0.06 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< Enter MolInt >>>>>>
Window information
Active MO range ...
Occupied :    2 MOs
  10   11
Virtual  :    2 MOs
  12   13
```

CI配置発生に使う軌道

SDCI計算

```
<<<<<< molint. All done. CPU-Time =     0.04 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< enter sdci1 >>>>>>
sdci calculation for singlet state.
no. of spin adapted sdci bases =   15
<<<<<< cimtss. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt10. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt1s. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt11. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt20. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt2s. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt21. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt22. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt30. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt3s. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt31. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt32. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt33. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt40. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt4s. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt41. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt42. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt43. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt44. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt50. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt5s. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt51. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt52. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt53. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt54. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
<<<<<< cimt55. All done. CPU-Time =     0.00 Sec. Total =  00H 00M 01S. >>>>>>
```

発生したCI配置とそのタイプ

```
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
Configuration Interaction information
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

  1 (    0,    0 ) --> (    0,    0 ) : no excitation (Hartree-Fock state)
  2 (   10,    0 ) --> (   12,    0 ) : single electron excitation
  3 (   10,    0 ) --> (   13,    0 ) : single electron excitation
  4 (   11,    0 ) --> (   12,    0 ) : single electron excitation
  5 (   11,    0 ) --> (   13,    0 ) : single electron excitation
  6 (   10,   10 ) --> (   12,   12 ) : type =   1
  7 (   10,   10 ) --> (   13,   13 ) : type =   1
  8 (   11,   11 ) --> (   12,   12 ) : type =   1
```

```
    9 (    11,   11 ) --> (    13,   13 ) : type =    1
   10 (    10,   11 ) --> (    12,   12 ) : type =    2
   11 (    10,   11 ) --> (    13,   13 ) : type =    2
   12 (    10,   10 ) --> (    12,   13 ) : type =    3
   13 (    11,   11 ) --> (    12,   13 ) : type =    3
   14 (    10,   11 ) --> (    12,   13 ) : type =    4
   15 (    10,   11 ) --> (    12,   13 ) : type =    5


 =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
 CI eigen values and states
 =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

 <<<<<< sdcil.  All done. CPU-Time =     0.02 Sec. Total =  00H 00M 01S. >>>>>>
 <<<<<< enter tmsdci >>>>>>
 computation of electronic transition moments.
 <<<<<< enter tmsdbs >>>>>>
 <<<<<< cimtss. All done. CPU-Time =     0.01 Sec. Total =  00H 00M 01S. >>>>>>
 <<<<<< tmsdbs  All done. CPU-Time =     0.01 Sec. Total =  00H 00M 01S. >>>>>>
 <<<<<< tmsdci. All done. CPU-Time =     0.09 Sec. Total =  00H 00M 01S. >>>>>>
----- Data file name -----
-rw-r--r--  1 ohtawara      560 Feb 29 21:21 but2s.dat
----- Started Time -----
Thu Feb 29 21:21:34 JST 1996
----- Ended time -----
Thu Feb 29 21:21:36 JST 1996
#----------------------------------------------------------------------
#   CNDO/S-SDCI  < 5*5 CI                Execute by K.Ohtawara
#   ATR Optical and Radio Communication Research Labs.
```

CIの固有値,固有ベクトルは.
出力できるが,ファイルが大きくなるので
cutした.

## D2. 2 次超分極率計算プログラム

2次分極率計算
ソースプログラム

```
C1996.2.20 Version (recalc version)  K.Ohtawara
C
C94. 4.21 -> add material param routine
C94. 7. 7 -> modify (g,m,n,v) m,n,v not include ground state
C            also add. eq. for minus contribution (g-m-g-n-g)
C94. 7.29 -> [a_bar]mn = <m|a|n> - <g|a|g>
C94. 8. 8 -> add material param routine for asymmetric system
C94. 8.22 -> add material param routine for 2,3,4 paths model
C94. 8.23 -> add material param routine for 5 paths model
C94. 9.30 -> modify [a_bar]mn = <m|a|n> - <g|a|g>(delta_mn)
C94. 9.30 -> calc. only xxxx tensor (fast version)
C94.11.21 -> print 30<
C94.11.25 -> contributions are arranged in order of m* (g93.f)
C96. 2.13 -> make fast ,save memory
C96. 2.19 -> save memory: recalc and cut ta2( , , )
C96. 2.20 -> bug fixed (when g(i,i,i,i).eq.0 bus error) amaxg
C96. 3. 6 -> recalc(3rd) when mcount .gt. nbase2 (memory overflow)
C96. 3. 6 -> last modified
c-----------------------------------------------------------------------
C     Program g100.f
C   Calculate Second-order Molecular Hyperpolarizabikity
C      gamma ijkl (-3w;w,w,w)
C            K.Ohtawara
C            ATR Optical and Radio Communication Research Labs.
C  calc gamma ijkl (-3w;w,w,w) using Orr's formula
C
C --- input file is fort.5 (***.tr)
c-----------------------------------------------------------------------
      program main
      implicit double precision (a-h,o-z)
      integer g,m,n,v
      character ident*80,systime*8
c*****************************************************************
c choose size of nbase depend on number of CIc
c       CI      nbase  memory(bss)
c      2 * 2      15
c      3 * 3      55
c      4 * 4     153
c      5 * 5     351
c      6 * 6     703
c      7 * 7    1275
c      8 * 8    2145
c      9 * 9    3403
c     10 *10    5151
c      parameter( nbase  = 360 )
c      parameter( nbase  = 710 )
      parameter( nbase  = 2150 )
c*****************************************************************
c     nbase3 = ( n_tmom * (n_tmom-1) ) / 2 +1  : size of tms
      parameter( nbase2 = nbase*100 )
      parameter( nbase3 = int(nbase*nbase/2) )

      dimension eng(0:nbase),tm(0:nbase,0:nbase,3),gamma(3,3,3,3),mst(3)
      dimension partm(0:nbase),partg(0:nbase,6),pg(6)
      dimension rra(nbase),rrb(nbase),jra(nbase),jrb(nbase)
      dimension work1(nbase2),work2(nbase2),iperm(nbase2),ip(nbase2)
      dimension ind1(nbase2),ind2(nbase2),ind3(nbase2)
      dimension tms(nbase3)

      common / io / iout
c-----------------------------------------------------------------------
      character ch(4),ten(3)*1,tensor(21)*4
      data ten/'x','y','z'/
      data tensor/'xxxx','yyyy','zzzz',
     #'xxyy','xxzz','yyxx','yyzz','zzxx','zzyy',
     #'xyxy','xzxz','yxyx','yzyz','zxzx','zyzy',
     #'xyyx','xzzx','yxxy','yzzy','zxxz','zyyz'/

      open(5,file='fort.5',form='formatted',status='old')
c      open(6,file='fort.6',form='formatted',status='unknown')
      iout = 16
      open(16,file='fort.16',form='formatted',status='unknown')

      open(7,file='fort.7',form='formatted',status='unknown')
      open(8,file='fort.8',form='formatted',status='unknown')
      open(9,file='fort.9',form='formatted',status='unknown')
      open(12,file='fort.12',form='formatted',status='unknown')
      open(13,file='fort.13',form='formatted',status='unknown')
c-----------------------------------------------------------------------
c read routine
      read(5,10)   ident
      read(5,*)    nmat
      write(6,*)   ident
      write(6,*)   nmat
 10   format(a80)
      write(7,*)  ident
      write(8,*)  ident
      write(9,*)  ident

      do i = 1, nmat
         read(5,*) idum,eng(i-1)
      enddo
      do i = 0,nmat
       do j = 0,nmat
        do k = 1,3
         tm(i,j,k)=0.D0
        enddo
       enddo
      enddo

      ncibas = nmat

      do ixyz = 1,3
         iw   = 10
         ilow = 1
         iup  = iw
 510     continue
         iup  = min0( iup, ncibas )
         read(5,*) ( idum,i=ilow,iup )
         iatold = 0
         do i = ilow,ncibas
            nclms = i - ilow + 1
            if( nclms .gt. iw ) then
               iu = ilow + iw - 1
            else
               iu = i
            endif
            ma = i*( i - 1 )/2
            read(5,*) idum,( tms(ma+j), j = ilow, iu )
         enddo
         if( iup .eq. ncibas ) go to 500
         ilow = ilow + iw
         iup  = iup  + iw
         go to 510
 500     continue

         do i = 1,ncibas
            do j = 1,ncibas
               n1  = max0( i, j )
               n2  = min0( i, j )
               n   = n1 *( n1 - 1 )/2 + n2
               tm(i-1,j-1,ixyz) = tms(n)
c tr_mom (in cgs: *1e-18)
c          tm(i-1,j-1,ixyz) = tm(i-1,j-1,ixyz) * 1.0d-18
               tm(i-1,j-1,ixyz) = tm(i-1,j-1,ixyz)
            enddo
         enddo
      enddo
c-----------------------------------------------------------------------
c      elec    = 4.803d-10,     e4=elec*elec*elec*elec
c      evtoerg = 1.601864d-12,  plank   = 1.054d-27

      evtoerg = 1.601864d0
      plank   = 1.054d0
      plank3  = plank * plank * plank
      e4      = 1.d0
      co      = e4 / plank3
      eng(0)  = 0.d0
      evtoergplank = evtoerg / plank

c eV -> erg ( *1.6d-12 )  -> freq (w) ( /plank )
c w:freq, e:Energy in eV, eng:Energy in erg

      do i=1,3
         gamma(i,i,i,i) = 0.d0
      enddo

c --- wave length of incident light [ nm ]
      ramda = 1900.d0
      omg0  = 0.d0
      omg1  = 1239.d0 / ramda * evtoerg / plank
      g = 0

c-----------------------------------------------------------------------
c Loop: omg = 0.65 eV, 0 eV
```

```
      do 5000 iomg = 1,2

         amaxg  = 0.d0
         mt = 0

         if(iomg.EQ.1) omg = omg1
         if(iomg.EQ.2) omg = omg0
         write(iout,400)
         write(iout,*) ident
         write(iout,401) omg * plank / evtoerg, nmat
         write(12,400)
         write(12,*) ident
         write(12,401) omg * plank / evtoerg, nmat
         write(13,400)
         write(13,*) ident
         write(13,401) omg * plank / evtoerg, nmat
 400  format('-----------------------------------------------------------------')
 401  format('w = ',f5.2,2x,'eV',5x,'CI = ',i5)

c-----------------------------------------------------------------------------
c calc selective tensor components listed as data /tensor/
c      iten = 1
c      do 3000 iten = 1,21

      do 3000 iten = 1,3

c save memory for sort
c recalc for define threshold
c  1st : find threshold, 2nd : cut below threshold
c  3rd : recalc when overflow
c
c set threshold level for sorting
      digit = 10d-5
      iover = 0

      do 3100 irec = 1,3
      if( irec .eq. 3  .and.  iover .eq. 0 ) go to 3100

      call time(systime)
      write(6,*) tensor(iten),' ',irec,'   ',systime

         do iloop = 1,4
            ch(iloop) = tensor(iten)(iloop:iloop)
         enddo

c ichar('x')=120,ichar('y')=121,ichar('z')=122 : x->1,y->2,z->3
         i = ichar(ch(1))-119
         j = ichar(ch(2))-119
         k = ichar(ch(3))-119
         l = ichar(ch(4))-119

         total1 = 0.d0
         total2 = 0.d0

      if( irec. ge. 2 ) go to 5900

c clear work for previous tensor before
c store work for next tensor at 2nd time
      do mclr = 1,mcount
         work1( mclr ) = 0.d0
         iperm( mclr ) = 0
         ind1( mclr ) = 0
         ind2( mclr ) = 0
         ind3( mclr ) = 0
      enddo
      mcount = 0

      do jmst = 0,nmat-1
         partm( jmst ) = 0.d0
         do jtype = 1,6
            partg( jmst,jtype ) = 0.d0
            pg( jtype ) = 0.d0
         enddo
      enddo

 5900    continue

c-----------------------------------------------------------------------------
         do 1000 v = 1,nmat-1
            do 1001 n =1 ,nmat-1
               do 1002 m = 1,nmat-1
                  wm   = eng(m) * evtoergplank
                  wn   = eng(n) * evtoergplank
                  wv   = eng(v) * evtoergplank
                  ggimn = 0.0d0
                  ggjmn = 0.0d0
                  ggkmn = 0.0d0
                  gglmn = 0.0d0
                  gginv = 0.0d0
                  ggjnv = 0.0d0
                  ggknv = 0.0d0
                  gglnv = 0.0d0

                  if( m .eq. n ) then
                     ggimn = tm(g,g,i)
                     ggjmn = tm(g,g,j)
                     ggkmn = tm(g,g,k)
                     gglmn = tm(g,g,l)
                  endif
                  if( n .eq. v ) then
                     gginv = tm(g,g,i)
                     ggjnv = tm(g,g,j)
                     ggknv = tm(g,g,k)
                     gglnv = tm(g,g,l)
                  endif

c-----------------------------------------------------------------------------
c first term ( sigma_mnv( Upper/Lower ) )
      up1  = tm(g,m,i)*(tm(m,n,j)-ggjmn)*(tm(n,v,k)-ggknv)*tm(v,g,l)
      up2  = tm(g,m,i)*(tm(m,n,j)-ggjmn)*(tm(n,v,l)-gglnv)*tm(v,g,k)
      up3  = tm(g,m,i)*(tm(m,n,k)-ggkmn)*(tm(n,v,j)-ggjnv)*tm(v,g,l)
      up4  = tm(g,m,i)*(tm(m,n,k)-ggkmn)*(tm(n,v,l)-gglnv)*tm(v,g,j)
      up5  = tm(g,m,i)*(tm(m,n,l)-gglmn)*(tm(n,v,j)-ggjnv)*tm(v,g,k)
      up6  = tm(g,m,i)*(tm(m,n,l)-gglmn)*(tm(n,v,k)-ggknv)*tm(v,g,j)
      ow1  =  (wm  -3*omg)*(wn  -2*omg)*(wv  -omg)
      t1   = up1  / ow1
      t2   = up2  / ow1
      t3   = up3  / ow1
      t4   = up4  / ow1
      t5   = up5  / ow1
      t6   = up6  / ow1

      up7  = tm(g,m,j)*(tm(m,n,i)-ggimn)*(tm(n,v,k)-ggknv)*tm(v,g,l)
      up8  = tm(g,m,j)*(tm(m,n,i)-ggimn)*(tm(n,v,l)-gglnv)*tm(v,g,k)
      up9  = tm(g,m,k)*(tm(m,n,i)-ggimn)*(tm(n,v,j)-ggjnv)*tm(v,g,l)
      up10 = tm(g,m,k)*(tm(m,n,i)-ggimn)*(tm(n,v,l)-gglnv)*tm(v,g,j)
      up11 = tm(g,m,l)*(tm(m,n,i)-ggimn)*(tm(n,v,j)-ggjnv)*tm(v,g,k)
      up12 = tm(g,m,l)*(tm(m,n,i)-ggimn)*(tm(n,v,k)-ggknv)*tm(v,g,j)
      ow2  =  (wm  +omg)*(wn  -2*omg)*(wv  -omg)
      t7   = up7  / ow2
      t8   = up8  / ow2
      t9   = up9  / ow2
      t10  = up10 / ow2
      t11  = up11 / ow2
      t12  = up12 / ow2

      up13 = tm(g,m,j)*(tm(m,n,k)-ggkmn)*(tm(n,v,i)-gginv)*tm(v,g,l)
      up14 = tm(g,m,j)*(tm(m,n,l)-gglmn)*(tm(n,v,i)-gginv)*tm(v,g,k)
      up15 = tm(g,m,k)*(tm(m,n,j)-ggjmn)*(tm(n,v,i)-gginv)*tm(v,g,l)
      up16 = tm(g,m,k)*(tm(m,n,l)-gglmn)*(tm(n,v,i)-gginv)*tm(v,g,j)
      up17 = tm(g,m,l)*(tm(m,n,j)-ggjmn)*(tm(n,v,i)-gginv)*tm(v,g,k)
      up18 = tm(g,m,l)*(tm(m,n,k)-ggkmn)*(tm(n,v,i)-gginv)*tm(v,g,j)
      ow3  =  (wm  +omg)*(wn  +2*omg)*(wv  -omg)
      t13  = up13 / ow3
      t14  = up14 / ow3
      t15  = up15 / ow3
      t16  = up16 / ow3
      t17  = up17 / ow3
      t18  = up18 / ow3

      up19 = tm(g,m,j)*(tm(m,n,k)-ggkmn)*(tm(n,v,l)-gglnv)*tm(v,g,i)
      up20 = tm(g,m,j)*(tm(m,n,l)-gglmn)*(tm(n,v,k)-ggknv)*tm(v,g,i)
      up21 = tm(g,m,k)*(tm(m,n,j)-ggjmn)*(tm(n,v,l)-gglnv)*tm(v,g,i)
      up22 = tm(g,m,k)*(tm(m,n,l)-gglmn)*(tm(n,v,j)-ggjnv)*tm(v,g,i)
      up23 = tm(g,m,l)*(tm(m,n,j)-ggjmn)*(tm(n,v,k)-ggknv)*tm(v,g,i)
      up24 = tm(g,m,l)*(tm(m,n,k)-ggkmn)*(tm(n,v,j)-ggjnv)*tm(v,g,i)
      ow4  =  (wm  +omg)*(wn  +2*omg)*(wv  +3*omg)
      t19  = up19 / ow4
      t20  = up20 / ow4
      t21  = up21 / ow4
      t22  = up22 / ow4
      t23  = up23 / ow4
      t24  = up24 / ow4

      tal1=(t1+t2+t3+t4+t5+t6+t7+t8+t9+t10+t11+t12+
     #t13+t14+t15+t16+t17+t18+t19+t20+t21+t22+t23+t24)
```

```
      total1 = total1 + ta1
      ta2 = (1/(plank3)) * ta1 /24.D0

c set threshold for sort (save memory)
      if( irec .eq. 1 ) go to 6500
      ajudge  = dabs( gamma(i,j,k,l) * digit )
      ta2now  = dabs( ta2 )
      if( ta2now .gt. ajudge ) then
         mcount = mcount + 1
c avoid over flow
         if( mcount .ge. nbase2 ) then
            digit = 10d-2
            mcount = 0
            iover = 1
            go to 3100
         else
            work1( mcount ) = ta2
            iperm( mcount ) = mcount
            ind1( mcount ) = m
            ind2( mcount ) = n
            ind3( mcount ) = v
         endif
      endif

 6500    continue

c-----------------------------------------------------------------------
c classify the type of transition for the 1st term
c m* define as 1st state of transition

c calc only irec=1
      if( irec .ge. 2 ) go to 6000

      if( m .eq. v ) then
         jtype = 1
         if( m .eq. n ) then
            jtype = 3
         endif
      else
         if( m .eq. n ) then
            jtype = 5
         else
            if( n .eq. v ) then
               jtype = 5
            endif
         endif
      endif
      if( m .ne. v .and. m .ne. n .and. n .ne. v ) jtype = 6
      jmst  = m

      partm( jmst ) = partm( jmst ) + ta2
      partg( jmst,jtype ) = partg( jmst,jtype ) + ta2
      pg( jtype ) = pg( jtype ) + ta2

 6000 continue

c-----------------------------------------------------------------------
c second term ( -sigma_mn( Upper/Lower ) )

      if( v .eq. 1 ) then

         aup1  = tm(g,n,i)*tm(n,g,j)*tm(g,m,k)*tm(m,g,l)
         aup2  = tm(g,n,i)*tm(n,g,j)*tm(g,m,l)*tm(m,g,k)
         aup3  = tm(g,n,i)*tm(n,g,k)*tm(g,m,j)*tm(m,g,l)
         aup4  = tm(g,n,i)*tm(n,g,k)*tm(g,m,l)*tm(m,g,j)
         aup5  = tm(g,n,i)*tm(n,g,l)*tm(g,m,j)*tm(m,g,k)
         aup6  = tm(g,n,i)*tm(n,g,l)*tm(g,m,k)*tm(m,g,j)
         aow1  =  (wn  -3*omg)*(wn  -omg)*(wm  -omg)
         at1   = aup1  / aow1
         at2   = aup2  / aow1
         at3   = aup3  / aow1
         at4   = aup4  / aow1
         at5   = aup5  / aow1
         at6   = aup6  / aow1

         aup7  = tm(g,n,j)*tm(n,g,i)*tm(g,m,k)*tm(m,g,l)
         aup8  = tm(g,n,j)*tm(n,g,i)*tm(g,m,l)*tm(m,g,k)
         aup9  = tm(g,n,k)*tm(n,g,i)*tm(g,m,j)*tm(m,g,l)
         aup10 = tm(g,n,k)*tm(n,g,i)*tm(g,m,l)*tm(m,g,j)
         aup11 = tm(g,n,l)*tm(n,g,i)*tm(g,m,j)*tm(m,g,k)
         aup12 = tm(g,n,l)*tm(n,g,i)*tm(g,m,k)*tm(m,g,j)
         aow2  =  (wn  -omg)*(wm  +omg)*(wm  -omg)
         at7   = aup7  / aow2
         at8   = aup8  / aow2
         at9   = aup9  / aow2
         at10  = aup10 / aow2
         at11  = aup11 / aow2
         at12  = aup12 / aow2

         aup13 = tm(g,n,j)*tm(n,g,k)*tm(g,m,i)*tm(m,g,l)
         aup14 = tm(g,n,j)*tm(n,g,l)*tm(g,m,i)*tm(m,g,k)
         aup15 = tm(g,n,k)*tm(n,g,j)*tm(g,m,i)*tm(m,g,l)
         aup16 = tm(g,n,k)*tm(n,g,l)*tm(g,m,i)*tm(m,g,j)
         aup17 = tm(g,n,l)*tm(n,g,j)*tm(g,m,i)*tm(m,g,k)
         aup18 = tm(g,n,l)*tm(n,g,k)*tm(g,m,i)*tm(m,g,j)
         aow3  =  (wn  +3*omg)*(wn  +omg)*(wm  +omg)
         at13  = aup13 / aow3
         at14  = aup14 / aow3
         at15  = aup15 / aow3
         at16  = aup16 / aow3
         at17  = aup17 / aow3
         at18  = aup18 / aow3

         aup19 = tm(g,n,j)*tm(n,g,k)*tm(g,m,l)*tm(m,g,i)
         aup20 = tm(g,n,j)*tm(n,g,l)*tm(g,m,k)*tm(m,g,i)
         aup21 = tm(g,n,k)*tm(n,g,j)*tm(g,m,l)*tm(m,g,i)
         aup22 = tm(g,n,k)*tm(n,g,l)*tm(g,m,j)*tm(m,g,i)
         aup23 = tm(g,n,l)*tm(n,g,j)*tm(g,m,k)*tm(m,g,i)
         aup24 = tm(g,n,l)*tm(n,g,k)*tm(g,m,j)*tm(m,g,i)
         aow4  =  (wn  +omg)*(wm  -omg)*(wm  +omg)
         at19  = aup19 / aow4
         at20  = aup20 / aow4
         at21  = aup21 / aow4
         at22  = aup22 / aow4
         at23  = aup23 / aow4
         at24  = aup24 / aow4

         ata11= -1.0* (at1+at2+at3+at4+at5+at6+at7+at8+at9+at10+
     &        at11+at12+at13+at14+at15+at16+at17+at18+at19+at20+at21+
     &        at22+at23+at24)
         total2 = total2 + ata11

c set threshold for sort (save memory)
      if( irec .eq. 1 ) go to 6600
      ajudge  = dabs( gamma(i,j,k,l) * digit )
      ta2now  = dabs( ta2 )
      if( ta2now .gt. ajudge ) then
         mcount = mcount + 1
c avoid over flow
         if( mcount .ge. nbase2 ) then
            digit = 10d-2
            mcount = 0
            iover = 1
            go to 3100
         else
            work1( mcount ) = ta2
            iperm( mcount ) = mcount
            ind1( mcount ) = m
            ind2( mcount ) = 0
            ind3( mcount ) = n
         endif
      endif
 6600 continue

c-----------------------------------------------------------------------
c classify the type of transition for the 2nd term

c calc only irec=1
         if( irec .ge. 2 ) go to 6100

         if( m .eq. n ) then
            jtype = 2
         else
            jtype = 4
         endif
         jmst  = m

         partm( jmst ) = partm( jmst ) + ta2
         partg( jmst,jtype ) = partg( jmst,jtype ) + ta2
         pg( jtype ) = pg( jtype ) + ta2

 6100    continue

      endif
c end of 2nd term
```

```
c-----------------------------------------------------------------------------

 1002       continue
 1001    continue
 1000 continue

      gamma(i,j,k,l) = (1/(plank3)) * (1.0d0/24.d0) * ( total1 + total2 )

c-----------------------------------------------------------------------------
c contribution analysis : write contribution of individual m* and type
c sort matrix using imsl library
c sort part gamma ascendingly in individual m*

c calc only irec=1
      if( irec .ge. 2 ) go to 6200

      itop = 5
      do jmst = 1,nmat
         rra(jmst) = partm(jmst)
         jra(jmst) = jmst
      enddo

c call imsl math library
      call DSVRBP(nmat,rra,rrb,jra)

      do nn = 1,nmat
         rra(nn) = rrb(nmat-nn+1)
         jrb(nn) = jra(nmat-nn+1)
      enddo

c write sort results
      write(13,710) ten(i),ten(j),ten(k),ten(l),gamma(i,j,k,l)*1.d-36
      if( gamma(i,j,k,l). eq. 0.d0 ) go to 790

c contribution from individual types -------------------------
      write(13,700)
      write(13,702)
      wsum = 0.d0
      do n1 = 1,6
         wsum = wsum +  abs(pg(n1))/abs(gamma(i,j,k,l)) * 100.d0
      enddo

      do n1 = 1,6
         print1 = pg(n1)*1.d-36
         print2 = pg(n1)/gamma(i,j,k,l)*100.d0
         print3 = abs( print2 ) / wsum *100.d0
         write(13,704) n1, print1, print2, print3
      enddo
 700  format(/'< contribution from individual transition types >')
 702  format('                (part gamma)        (%)     weight(%)')
 704  format(4x,'type ',i1,3x,e13.5e3,2f10.1)

c recognize m* state along maximum tensor ( when w=0 )
      if( abs(gamma(iten,iten,iten,iten)) .ge. amaxg ) then
         mt = iten
         amaxg = abs( gamma(iten,iten,iten,iten) )
      endif

c 1996.2.20bug(bus error) : when  gamma(i,i,i,i) = 0.0
c       if( iten .eq. 3 ) then
c          amaxg = dmax1( gamma(1,1,1,1),gamma(2,2,2,2),gamma(3,3,3,3) )
c          do n1 = 1,3
c             if( amaxg .eq. gamma(n1,n1,n1,n1) ) mt = n1
c          enddo
c       endif

      if( iomg .eq. 2 ) then
         if( tensor(iten) .eq. 'xxxx' ) mst(iten) = jrb(1)
         if( tensor(iten) .eq. 'yyyy' ) mst(iten) = jrb(1)
         if( tensor(iten) .eq. 'zzzz' ) mst(iten) = jrb(1)
      endif

c contribution from individual m*    --------------------------
      write(13,715) itop
      write(13,720)
      do nn = 1,itop
         print1 = rra(nn)*1.d-36
         print2 = rra(nn)/gamma(i,j,k,l)*100.d0
         write(13,730) nn,jrb(nn),print1,print2

         wsum = 0.d0
         do n1 = 1,6
            wsum = wsum +  abs( partg(jrb(nn),n1)/rra(nn) ) *100.d0
         enddo
         do jtype = 1,6
            print1 = partg(jrb(nn),jtype)*1.d-36
            print2 = partg(jrb(nn),jtype)/rra(nn)*100.d0
            print3 = abs( print2 ) / wsum *100.d0
            write(13,731) jtype, print1, print2, print3
         enddo
      enddo

 790  continue

 710  format(/4a1,3x,e12.5e3,5x)
 715  format(/'< contribution from individual m* states : top ',i2,' >')
 720  format('(num) (m*)    (part gamma)        (%)     weight(%)')
 730  format(/i3,i5,5x,e13.5e3,f10.1/)
 731  format(4x,'type ',i1,3x,e13.5e3,2f10.1)

 6200 continue

c-----------------------------------------------------------------------------
c new sort routine 96.2.27  -> cut under (gamma*digit)
c calc only irec=2,3
      if( irec .eq. 1 ) go to 900

      numofsort = 50

      if ( gamma(i,j,k,l) .eq. 0.d0) then
         write(iout,800) ten(i),ten(j),ten(k),ten(l),gamma(i,j,k,l)
         write(12,1800) ten(i),ten(j),ten(k),ten(l),gamma(i,j,k,l)**1.d-36
         go to 900
      endif

      write(iout,810) ten(i),ten(j),ten(k),ten(l),gamma(i,j,k,l),mcount
      write(12,1810) ten(i),ten(j),ten(k),ten(l),
     &                gamma(i,j,k,l)*1.d-36,mcount

c call imsl math library
      call DSVRBP(mcount,work1,work2,iperm)

c invert
      do n1 = 1, mcount
         work1( n1 ) = work2( mcount - n1 + 1)
         ip( n1 )    = iperm( mcount - n1 + 1)
      enddo

      if( mcount .lt. numofsort) write(iout,830) mcount
      aintg = 0.d0
      do n1 = 1, numofsort
         ix = ip( n1 )
         aintg = aintg + work1( n1 )
         apart = aintg / gamma(i,j,k,l) * 100.0D0
         write(iout,820) n1,ind1(ix),ind2(ix),ind3(ix),work1(n1),aintg,apart
         write(12,1820) n1,ind1(ix),ind2(ix),ind3(ix),
     &                  work1(n1)*1.d-36,aintg*1.d-36,apart
c output for graph  x:fort.7, y:fort.8, z:fort.9
         write(iten+6,1900) n1,work1(n1)*1.d-36,aintg*1.d-36
      enddo

 800  format(/4a1,5x,'[ 10^-36 esu ]',f12.6,4x,'count=     0',)
 810  format(/4a1,5x,'[ 10^-36 esu ]',f12.6,4x,'count='i6,)
 820  format(i5,6x,3i4,2f12.6,f10.1)
 830  format('*** caution *** : unreliable value below ',i6)

 1800 format(4a1,3x,e12.5e3,5x,'count=     0',)
 1810 format(4a1,3x,e12.5e3,5x,'count=',i6)
 1820 format(i5,6x,3i4,2e16.5,f10.1)
 1900 format(i5,2e16.5)


 900  continue
c end of sort routine
c-----------------------------------------------------------------------------

 3100 continue
 3000 continue
c end of large loop (iten)

c-----------------------------------------------------------------------------
c  calc net gamma
      ga = 0.d0
      do 4000 i=1,3
         do 4000 j=1,3
            if(i.eq.j) go to 4000
```

```
          ga = ga + gamma(i,i,j,j) + gamma(i,j,i,j) + gamma(i,j,j,i)
 4000 continue

      gadiag = gamma(1,1,1,1) + gamma(2,2,2,2) + gamma(3,3,3,3)
      ga     = ga / 3.d0 + gadiag
      ga     = ga / 5.d0
      chi    = ga * 6.02d23

      write(iout,1100) ga  * 1.d-36
      write(iout,1110) chi * 1.d-36
      write(iout,1120)
 1100  format(/'gamma       [esu]  ',e12.5)
 1110  format('chi3(rough) [esu]  ',e12.5)
 1120  format(/)

 5000 continue

c end of Large Large Loop (omg=0.65,0)

c-----------------------------------------------------------------------
c calc material parameters about maximum tensor component
c tm: tr_mom in Debye, eng:energy in eV

c mstar is m* along largest tensor ( w=0 )
      mstar = mst( mt )
      suma = 0.d0
      sumb = 0.d0
      sumc = 0.d0
      do n1 = 1,nmat-1
         suma = suma +  tm(n1,g,mt)
         sumb = sumb +  tm(mstar,n1,mt) - tm(g,g,mt)
         sumc = sumc +  eng(n1)
      enddo

      write(iout,410)
      write(iout,*)   ident
      write(iout,412) tensor(mt)
      write(iout,413) mstar

      write(iout,414) tensor(mt),gamma(mt,mt,mt,mt)*1.e-36
      write(iout,415) ga*1.e-36
      write(iout,420) tm(mstar,g,mt),tm(mstar,mstar,mt)-tm(g,g,mt),tm(g,g,mt)
      write(iout,422) eng(mstar)

 410  format('  <<< material parameters ( w = 0 ) >>>')
 412  format('   tensor          : ',4x,a4 )
 413  format('   m* state        : ',i5)
 414  format('   g_',a4,'(0;0,0,0) : ',e14.5,' esu  ')
 415  format('   g(0;0,0,0)      : ',e14.5,' esu')
 420  format('   u_mg            : ',f10.5,' Debye',/,
     &       '   u_mm-           : ',f10.5,' Debye',/,
     &       '   u_gg            : ',f10.5,' Debye')
 422  format('   E_mg            : ',f10.5,' eV ')

c-----------------------------------------------------------------------
c parameters(g_dag,psi,x) for individual types (5types model)

      s1 = 0.d0
      s2 = 0.d0
      s3 = 0.d0
      s4 = 0.d0
      s5 = 0.d0
      s6 = 0.d0

      umm = tm(mstar,mstar,mt) - tm(g,g,mt)
      ugg = tm(g,g,mt)
      m = mstar
      ergm = eng(m) * evtoerg

      do i = 1,nmat-1
       if( i .ne. mstar ) then
         ergi = eng(i) * evtoerg
         s1 = s1 + ergm/ergi * tm(m,i,mt)**2
         s2 = s2 + umm**2 * tm(m,i,mt)**2 / (ergm**2 * ergi )
         s4 = s4 + tm(m,i,mt)**2 / ergi
         s5 = s5 + tm(i,g,mt)**2 / ergi
         s6 = s6 + tm(m,i,mt)*tm(i,g,mt) / ergi
       endif
      enddo

      s10  = s1 + umm**2
      dag5 = 1.d0/(4.d0*ergm**3)* s10**2 + (1/4)*s2
      x5   = dsqrt(2.d0*tm(m,g,mt)**2 / s10)
```



```
      g5 = tm(m,g,mt)**2/ergm**2 * s4 - tm(m,g,mt)**4/ergm**3 +
     &     tm(m,g,mt)**2 * umm**2/ergm**3 - tm(m,g,mt)**2/ergm**2*s5 +
     &     ( tm(m,g,mt)*umm/ergm**2 ) * s6

      phi5 = g5 / dag5

      write(iout,450) g5,dag5,phi5
 450  format('   g_5types        : ',e14.5,' esu',/,
     &       '   g_dag           : ',e14.5,' esu',/,
     &       '   Phi             : ',f10.5)

      call clock('all    ')

      stop
      end

c=====================================================================
      subroutine clock(rname)

c single: real single(2)
c    single(1) = user time (sec)
c    single(2) = system time (sec)
c etime : user + system time (sec)

      character*7 rname
      dimension single(2)
      common / io / iout
      data time0 / 0.0e+0 /

c 100  format(1h ,'=== ',a6,' done ===',' CPU time = ',f8.2,' sec',
c     &          ' Total =',i4.2,'hr ',i2.2,'min ',i2.2,'sec')
 100  format(/ ,'*** ',' CPU time = ',f8.2,' sec',' : ',
     &          ' Total =',i4.2,'hr ',i2.2,'min ',i2.2,'sec','  *** '/)

      time1 = etime( single )
      step  = time1 - time0
      icpu  = nint( time1 )

      ihr   = icpu / 3600
      imin  = icpu /   60  - ihr * 60
      isec  = icpu - imin * 60 - ihr * 3600
c      write(iout,100) rname,step,ihr,imin,isec
      write(iout,100) step,ihr,imin,isec
      write(6,100) step,ihr,imin,isec
      time0 = time1

      return
      end
c=====================================================================
```

入力ファイル： aaa.tr

butadiene 2*2 → コメント

15 → 状態数（CI数）

| | |
|---|---|
| 1 | 0.000000 |
| 2 | 6.104268 |
| 3 | 6.376558 |
| 4 | 8.198333 |
| 5 | 10.031304 |
| 6 | 10.079501 |
| 7 | 10.950825 |
| 8 | 11.265032 |
| 9 | 14.408706 |
| 10 | 14.690229 |
| 11 | 15.787569 |
| 12 | 16.189716 |
| 13 | 16.878268 |
| 14 | 24.776737 |
| 15 | 25.070125 |

励起状態のエネルギー（基底状態を1とする）

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.000000 | | | | | | | | | |
| 2 | 5.439123 | 0.000000 | | | | | | | | |
| 3 | 0.000000 | 0.896436 | 0.000000 | | | | | | | |
| 4 | 0.000000 | 1.879110 | 0.000000 | 0.000000 | | | | | | |
| 5 | -0.214871 | 0.000000 | 5.985123 | -2.744225 | 0.000000 | | | | | |
| 6 | 0.000000 | 8.801798 | 0.000000 | 0.000000 | 1.919589 | 0.000000 | | | | |
| 7 | -0.087060 | 0.000000 | 3.445815 | 4.546592 | 0.000000 | 0.538894 | 0.000000 | | | |
| 8 | 0.000000 | -1.628350 | 0.000000 | 0.000000 | 5.416789 | 0.000000 | 0.649500 | 0.000000 | | |
| 9 | 0.163292 | 0.000000 | -0.645425 | -1.094949 | 0.000000 | -8.982208 | 0.000000 | -1.672115 | 0.000000 | |
| 10 | 0.000000 | -1.401435 | 0.000000 | 0.000000 | 0.921908 | 0.000000 | 4.413028 | 0.000000 | -1.197040 | 0.000000 |
| 11 | 0.000000 | -1.431051 | 0.000000 | 0.000000 | 1.201809 | 0.000000 | -3.239655 | 0.000000 | -0.837901 | 0.000000 |
| 12 | -0.089040 | 0.000000 | 0.555974 | 0.808713 | 0.000000 | 0.459942 | 0.000000 | -2.245332 | 0.000000 | 6.920891 |
| 13 | 0.000000 | 0.650157 | 0.000000 | 0.000000 | 0.010287 | 0.000000 | 1.162987 | 0.000000 | -10.760037 | 0.000000 |
| 14 | 0.031248 | 0.000000 | 0.201330 | -0.242212 | 0.000000 | -0.419524 | 0.000000 | 1.325692 | 0.000000 | 0.757346 |
| 15 | 0.000000 | 0.203568 | 0.000000 | 0.000000 | -0.184697 | 0.000000 | -0.422659 | 0.000000 | 2.230585 | 0.000000 |

| | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|
| 11 | 0.000000 | | | | |
| 12 | -6.237098 | 0.000000 | | | |
| 13 | 0.000000 | -0.220539 | 0.000000 | | |
| 14 | 2.107402 | 0.000000 | -4.017798 | 0.000000 | |
| 15 | 0.000000 | -1.154273 | 0.000000 | 21.616033 | 0.000000 |

x方向の遷移モーメント行列

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.000000 | | | | | | | | | |
| 2 | -1.209630 | 0.000000 | | | | | | | | |
| 3 | 0.000000 | 1.949224 | 0.000000 | | | | | | | |
| 4 | 0.000000 | -0.831718 | 0.000000 | 0.000000 | | | | | | |
| 5 | 1.590110 | 0.000000 | 1.025008 | 0.347327 | 0.000000 | | | | | |
| 6 | 0.000000 | -1.015776 | 0.000000 | 0.000000 | 1.237584 | 0.000000 | | | | |
| 7 | 0.432898 | 0.000000 | -0.319644 | -1.106741 | 0.000000 | 0.627695 | 0.000000 | | | |
| 8 | 0.000000 | 1.900072 | 0.000000 | 0.000000 | -0.693300 | 0.000000 | -0.517165 | 0.000000 | | |
| 9 | -0.227276 | 0.000000 | 1.279214 | -0.420344 | 0.000000 | 0.338833 | 0.000000 | 0.309958 | 0.000000 | |
| 10 | 0.000000 | 0.784086 | 0.000000 | 0.000000 | -1.374323 | 0.000000 | 0.293815 | 0.000000 | -0.104182 | 0.000000 |
| 11 | 0.000000 | 0.320136 | 0.000000 | 0.000000 | -0.595573 | 0.000000 | -0.632596 | 0.000000 | -0.328851 | 0.000000 |
| 12 | 0.001035 | 0.000000 | 0.276495 | 1.204249 | 0.000000 | -0.348615 | 0.000000 | -0.208104 | 0.000000 | 0.766795 |
| 13 | 0.000000 | 0.096001 | 0.000000 | 0.000000 | -1.910183 | 0.000000 | -0.552571 | 0.000000 | -0.374162 | 0.000000 |
| 14 | -0.036333 | 0.000000 | -0.354688 | 0.077969 | 0.000000 | -0.306468 | 0.000000 | 0.109379 | 0.000000 | 0.207416 |
| 15 | 0.000000 | -0.406522 | 0.000000 | 0.000000 | -0.365751 | 0.000000 | -0.143458 | 0.000000 | 0.276676 | 0.000000 |

| | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|
| 11 | 0.000000 | | | | |
| 12 | -0.671134 | 0.000000 | | | |
| 13 | 0.000000 | -0.011533 | 0.000000 | | |
| 14 | 0.359790 | 0.000000 | -0.391994 | 0.000000 | |
| 15 | 0.000000 | -0.121766 | 0.000000 | 2.675474 | 0.000000 |

y方向の遷移モーメント行列

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.000000 | | | | | | | | | |
| 2 | 0.000000 | 0.000000 | | | | | | | | |
| 3 | 0.000000 | 0.000000 | 0.000000 | | | | | | | |
| 4 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | | | | | | |
| 5 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | | | | | |
| 6 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | | | | |
| 7 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | | | |
| 8 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | | |
| 9 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | |
| 10 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 11 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 12 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 13 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 14 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 15 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |

| | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|
| 11 | 0.000000 | | | | |
| 12 | 0.000000 | 0.000000 | | | |
| 13 | 0.000000 | 0.000000 | 0.000000 | | |
| 14 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | |
| 15 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |

z方向の遷移モーメント行列

実行用シェルスクリプト
(run)

```
#-----------------------------------------------------------------------
#   g100.f  Gamma ijkl (-3w;w,w,w)      Execute by K.Ohtawara
#   ATR Optical and Radio Communication Research Labs.

if ($#argv == 0) then
 echo "Usage: run < file_name >"
 exit 1
endif
if (!(-e $1:r.tr)) then
 echo "------------   ***.tr  file not found !!! -----------"
 exit 0
endif

echo "=== Gamma started.  Calculating ...."

echo "***  Data file :">$$
ls -l $1:r.tr >>$$
echo "***  Started time :">>$$
date >>$$
date
cp $1:r.tr fort.5
./g100
echo "***  Ended time :">>$$
date >>$$
date
head -3 rung >> $$

if(-e fort.16) then
  cat $$ >>fort.16
  mv fort.16 $1:r.g
endif

if(-e fort.12) then
  cat $$ >>fort.12
  mv fort.12 $1:r.g_12
endif

if(-e fort.13) then
  cat $$ >>fort.13
  mv fort.13 $1:r.g_type
endif

if(-e fort.7) then
  mv fort.7 $1:r.x
endif
if(-e fort.8) then
  mv fort.8 $1:r.y
endif
if(-e fort.9) then
  mv fort.9 $1:r.z
endif

#if(-e fort.11) then
#  mv fort.11 $1:r.11
#endif

rm fort.*
rm $$

ls -l $1:r.tr
ls -l $1:r.g*
```

1) データファイル名 aaa.tr に対し、
run aaa で実行

2) 出力ファイルは6つ生成する

aaa.g, aaa.g_12 } 2次分極率(γ)
aaa.g_type 遷移タイプ解析
aaa.x, aaa.y, aaa.z } γの各方向成分 (プロット用データ)

出力ファイル： aaa.g

2次分極率 γ の各テンソル成分と γ に寄与度の高い順に並べた，各遷移パスごとの γ 値

```
   butadiene 2*2
w =  0.65  eV      CI =     15

xxxx      [ 10^-36 esu ]     0.811917     count=     78
    1          1   5   1     1.690946     1.690946      208.3
    2          1   0   1    -1.055333     0.635613       78.3
    3          1   3   1     0.096199     0.731812       90.1
    4          1   7   1     0.051457     0.783269       96.5
    5          1   2   1     0.028911     0.812180      100.0
    6          1   9   1     0.028896     0.841076      103.6
    7          1  10   1     0.027969     0.869045      107.0
    8          1   5   8    -0.021195     0.847850      104.4
    9          8   5   1    -0.020755     0.827095      101.9
   10          1   5   4    -0.008645     0.818450      100.8

yyyy      [ 10^-36 esu ]     0.010706     count=    150
    1          1   2   1     0.006761     0.006761       63.1
    2          1   7   1     0.003465     0.010226       95.5

gamma        [esu]    0.16452E-36
chi3(rough)  [esu]    0.99044E-13

-------------------------------------------------------------
   butadiene 2*2
w =  0.00  eV      CI =     15

xxxx      [ 10^-36 esu ]     0.700403     count=     78
    1          1   5   1     1.484627     1.484627      212.0
    2          1   0   1    -0.936134     0.548494       78.3
```

入射光エネルギー

γxxxx

各遷移パスごとの γ 値

γ 値の積分

各 γ 値のテンソル成分に対する割合(%)の積分

遷移パス … 151 は g-1-5-1-g を表す

→ ガスモデル (isotropic) による γ

```
  <<< material parameters ( w = 0 ) >>>
   butadiene 2*2
   tensor            :     xxxx
   m* state          :     1
   g_xxxx(0;0,0,0)   :    0.70040E-36 esu
   g(0;0,0,0)        :    0.14196E-36 esu
   u_mg              :    5.43912 Debye
   u_mm-             :    0.00000 Debye
   u_gg              :    0.00000 Debye
   E_mg              :    6.10427 eV
   g_5types          :    0.75605E+00 esu
   g_dag             :    0.76605E+00 esu
   Phi               :    0.98694

***  CPU time =     6.92 sec :  Total =  00hr 00min 07sec  ***
```

物質パラメータ値

5type による γ

$\gamma^{\dagger}$

$\Psi$

出力ファイル: aaa.g

```
-----------------------------------------------------------------
   butadiene 2*2
w =  0.65  eV     CI =    15

xxxx     [ 10^-36 esu ]    0.811917    count=    78
    1        1   5   1    1.690946    1.690946     208.3
    2        1   0   1   -1.055333    0.635613      78.3
    3        1   3   1    0.096199    0.731812      90.1
    4        1   7   1    0.051457    0.783269      96.5
    5        1   2   1    0.028911    0.812180     100.0
    6        1   9   1    0.028896    0.841076     103.6
    7        1  10   1    0.027969    0.869045     107.0
    8        1   5   8   -0.021195    0.847850     104.4
    9        8   5   1   -0.020755    0.827095     101.9
   10        1   5   4   -0.008645    0.818450     100.8
   11        4   5   1   -0.008495    0.809955      99.8
   12        1  12   1    0.005389    0.815344     100.4
   13        1   2   4   -0.004516    0.810828      99.9
   14        4   2   1   -0.004428    0.806400      99.3
   15        1   7   4    0.004014    0.810415      99.8
   16        4   7   1    0.003946    0.814361     100.3
   17        1   3   4    0.003291    0.817651     100.7
   18        4   3   1    0.003231    0.820882     101.1
   19        1   3   6   -0.002018    0.818864     100.9
   20        6   3   1   -0.001978    0.816886     100.6
   21        1  12   8   -0.001098    0.815788     100.5
   22        8  12   1   -0.001077    0.814711     100.3
   23        1   2   6   -0.000962    0.813749     100.2
   24        6   2   1   -0.000942    0.812807     100.1
   25        1   5   6   -0.000898    0.811909     100.0
   26        6   5   1   -0.000882    0.811028      99.9
   27        1   9  11    0.000850    0.811878     100.0
   28       11   9   1    0.000834    0.812711     100.1
   29        4   0   1   -0.000809    0.811902     100.0
   30        1   9   6    0.000790    0.812693     100.1
   31        6   9   1    0.000777    0.813470     100.2
   32        1   0   4   -0.000749    0.812721     100.1
   33        1  10  11   -0.000727    0.811994     100.0
   34       11  10   1   -0.000712    0.811282      99.9
   35        4   2   4    0.000693    0.811974     100.0
   36        1   3   8   -0.000688    0.811287      99.9
   37        8   3   1   -0.000673    0.810614      99.8
   38        1   7   8    0.000649    0.811264      99.9
   39        8   7   1    0.000636    0.811900     100.0
   40        1  10   4    0.000551    0.812451     100.1
   41        1  10   6   -0.000550    0.811901     100.0
   42        4  10   1    0.000543    0.812444     100.1
   43        6  10   1   -0.000541    0.811903     100.0
   44        1   5  11   -0.000526    0.811377      99.9
   45       11   5   1   -0.000514    0.810863      99.9
   46        1   9   4    0.000446    0.811309      99.9
   47        4   9   1    0.000439    0.811748     100.0
   48        1   7  11   -0.000422    0.811326      99.9
   49       11   7   1   -0.000414    0.810912      99.9
   50        1  14   1    0.000353    0.811265      99.9

yyyy     [ 10^-36 esu ]    0.010706    count=   150
    1        1   2   1    0.006761    0.006761      63.1
    2        1   7   1    0.003465    0.010226      95.5
    3        1   2   4   -0.002768    0.007458      69.7
    4        4   2   1   -0.002714    0.004745      44.3
    5        1   0   1   -0.002582    0.002163      20.2
    6        4   0   1   -0.002192   -0.000029      -0.3
    7        1   0   4   -0.002028   -0.002057     -19.2
    8        4   0   4   -0.001609   -0.003665     -34.2
    9        4  12   4    0.001384   -0.002281     -21.3
   10        1   5   1    0.001114   -0.001167     -10.9
   11        4   2   4    0.001113   -0.000054      -0.5
   12        1   5   4    0.001059    0.001004       9.4
   13        4   5   1    0.001040    0.002044      19.1
   14        4   5   4    0.000990    0.003035      28.3
   15        1   7   4    0.000987    0.004021      37.6
   16        4   7   1    0.000970    0.004991      46.6
   17        1   3   1    0.000932    0.005923      55.3
   18        4   9   4    0.000826    0.006750      63.0
   19        1   9   4    0.000612    0.007362      68.8
   20        4   9   1    0.000602    0.007964      74.4
   21        1   9   1    0.000447    0.008412      78.6
   22        1   2   8    0.000340    0.008752      81.7
   23        8   2   1    0.000332    0.009084      84.8
   24        1   3   4    0.000303    0.009387      87.7
   25        4   3   1    0.000298    0.009685      90.5
   26        4   7   4    0.000277    0.009962      93.0
   27        1   3   6   -0.000240    0.009721      90.8
   28        6   3   1   -0.000236    0.009486      88.6
   29        1   2   6    0.000215    0.009700      90.6
   30        6   2   1    0.000210    0.009910      92.6
   31        1   7   6    0.000183    0.010093      94.3
   32        6   7   1    0.000180    0.010273      96.0
   33        6   0   1   -0.000144    0.010129      94.6
   34        4  10   4    0.000144    0.010273      96.0
   35        4   2   8   -0.000137    0.010136      94.7
   36        8   2   4   -0.000136    0.010000      93.4
   37        1   5   6    0.000134    0.010133      94.6
   38        1   0   6   -0.000132    0.010001      93.4
   39        6   5   1    0.000131    0.010132      94.6
   40        4   5   6    0.000125    0.010257      95.8
   41        6   5   4    0.000125    0.010382      97.0
   42        6   0   4   -0.000105    0.010277      96.0
   43        4   0   6   -0.000104    0.010173      95.0
   44        1  10   4    0.000101    0.010274      96.0
   45        4  12   6    0.000100    0.010373      96.9
   46        6  12   4    0.000100    0.010473      97.8
   47        4  10   1    0.000099    0.010572      98.7
   48        4   3   4    0.000097    0.010669      99.7
   49        1  12   4    0.000090    0.010759     100.5
   50        4  12   1    0.000089    0.010848     101.3

zzzz     [ 10^-36 esu ]    0.000000    count=     0

gamma        [esu]    0.16452E-36
chi3(rough)  [esu]    0.99044E-13

-----------------------------------------------------------------
   butadiene 2*2
w =  0.00  eV     CI =    15

xxxx     [ 10^-36 esu ]    0.700403    count=    78
    1        1   5   1    1.484627    1.484627     212.0
    2        1   0   1   -0.936134    0.548494      78.3
    3        1   3   1    0.083194    0.631688      90.2
    4        1   7   1    0.045465    0.677153      96.7
    5        1   9   1    0.025824    0.702977     100.4
    6        1  10   1    0.025056    0.728033     103.9
    7        1   2   1    0.024343    0.752375     107.4
    8        1   5   8   -0.019270    0.733106     104.7
    9        8   5   1   -0.019270    0.713836     101.9
   10        4   5   1   -0.007784    0.706053     100.8
   11        1   5   4   -0.007784    0.698269      99.7
   12        1  12   1    0.004838    0.703107     100.4
   13        1   2   4   -0.003907    0.699200      99.8
   14        4   2   1   -0.003907    0.695293      99.3
   15        4   7   1    0.003636    0.698928      99.8
   16        1   7   4    0.003636    0.702564     100.3
   17        1   3   4    0.002921    0.705485     100.7
   18        4   3   1    0.002921    0.708405     101.1
   19        6   3   1   -0.001796    0.706609     100.9
   20        1   3   6   -0.001796    0.704813     100.6
   21        8  12   1   -0.001018    0.703795     100.5
   22        1  12   8   -0.001018    0.702777     100.3
   23        6   2   1   -0.000835    0.701942     100.2
   24        1   2   6   -0.000835    0.701107     100.1
   25        1   5   6   -0.000811    0.700296     100.0
   26        6   5   1   -0.000811    0.699485      99.9
   27       11   9   1    0.000787    0.700272     100.0
   28        1   9  11    0.000787    0.701060     100.1
   29        6   9   1    0.000726    0.701785     100.2
   30        1   9   6    0.000726    0.702511     100.3
   31        4   0   1   -0.000715    0.701796     100.2
   32        1   0   4   -0.000715    0.701081     100.1
   33       11  10   1   -0.000674    0.700407     100.0
   34        1  10  11   -0.000674    0.699733      99.9
   35        4   2   4    0.000627    0.700360     100.0
   36        8   3   1   -0.000617    0.699743      99.9
   37        1   3   8   -0.000617    0.699126      99.8
   38        8   7   1    0.000594    0.699720      99.9
   39        1   7   8    0.000594    0.700314     100.0
   40        1  10   6   -0.000506    0.699808      99.9
   41        6  10   1   -0.000506    0.699302      99.8
   42        1  10   4    0.000506    0.699808      99.9
   43        4  10   1    0.000506    0.700314     100.0
   44        1   5  11   -0.000479    0.699835      99.9
   45       11   5   1   -0.000479    0.699356      99.9
   46        1   9   4    0.000408    0.699764      99.9
   47        4   9   1    0.000408    0.700173     100.0
```

```
   48          1   7  11   -0.000387    0.699786      99.9
   49         11   7   1   -0.000387    0.699399      99.9
   50          1  14   1    0.000319    0.699718      99.9

yyyy     [ 10^-36 esu ]    0.009379    count=   150
    1          1   2   1    0.005692    0.005692      60.7
    2          1   7   1    0.003062    0.008754      93.3
    3          4   2   1   -0.002394    0.006360      67.8
    4          1   2   4   -0.002394    0.003965      42.3
    5          1   0   1   -0.002290    0.001675      17.9
    6          1   0   4   -0.001937   -0.000261      -2.8
    7          4   0   1   -0.001937   -0.002198     -23.4
    8          4   0   4   -0.001541   -0.003739     -39.9
    9          4  12   4    0.001322   -0.002417     -25.8
   10          4   2   4    0.001007   -0.001410     -15.0
   11          1   5   1    0.000978   -0.000432      -4.6
   12          4   5   1    0.000953    0.000521       5.6
   13          1   5   4    0.000953    0.001474      15.7
   14          4   5   4    0.000929    0.002403      25.6
   15          1   7   4    0.000894    0.003297      35.1
   16          4   7   1    0.000894    0.004190      44.7
   17          1   3   1    0.000806    0.004996      53.3
   18          4   9   4    0.000786    0.005782      61.7
   19          4   9   1    0.000561    0.006343      67.6
   20          1   9   4    0.000561    0.006903      73.6
   21          1   9   1    0.000400    0.007303      77.9
   22          8   2   1    0.000297    0.007601      81.0
   23          1   2   8    0.000297    0.007898      84.2
   24          4   3   1    0.000269    0.008167      87.1
   25          1   3   4    0.000269    0.008437      89.9
   26          4   7   4    0.000261    0.008697      92.7
   27          6   3   1   -0.000214    0.008483      90.4
   28          1   3   6   -0.000214    0.008269      88.2
   29          6   2   1    0.000186    0.008456      90.2
   30          1   2   6    0.000186    0.008642      92.1
   31          6   7   1    0.000166    0.008808      93.9
   32          1   7   6    0.000166    0.008974      95.7
   33          4  10   4    0.000137    0.009112      97.1
   34          1   0   6   -0.000127    0.008984      95.8
   35          6   0   1   -0.000127    0.008857      94.4
   36          4   2   8   -0.000125    0.008732      93.1
   37          8   2   4   -0.000125    0.008607      91.8
   38          6   5   1    0.000121    0.008727      93.1
   39          1   5   6    0.000121    0.008848      94.3
   40          6   5   4    0.000117    0.008966      95.6
   41          4   5   6    0.000117    0.009083      96.8
   42          4   0   6   -0.000100    0.008983      95.8
   43          6   0   4   -0.000100    0.008883      94.7
   44          6  12   4    0.000095    0.008978      95.7
   45          4  12   6    0.000095    0.009073      96.7
   46          4  10   1    0.000092    0.009166      97.7
   47          1  10   4    0.000092    0.009258      98.7
   48          4   3   4    0.000090    0.009348      99.7
   49          1  12   4    0.000083    0.009431     100.6
   50          4  12   1    0.000083    0.009514     101.4

zzzz     [ 10^-36 esu ]    0.000000    count=     0

gamma        [esu]   0.14196E-36
chi3(rough) [esu]   0.85458E-13


  <<< material parameters ( w = 0 ) >>>
   butadiene 2*2
   tensor            :     xxxx
   m* state          :     1
   g_xxxx(0;0,0,0)   :    0.70040E-36 esu
   g(0;0,0,0)        :    0.14196E-36 esu
   u_mg              :    5.43912 Debye
   u_mn-             :    0.00000 Debye
   u_gg              :    0.00000 Debye
   E_mg              :    6.10427 eV
   g_5types          :    0.75605E+00 esu
   g_dag             :    0.76605E+00 esu
   Phi               :    0.98694

***  CPU time =     6.92 sec :  Total =  00hr 00min 07sec  ***

***  Data file :
-rw-rw-r--  1 ohtawara     5364 Feb 20 22:48 but2s.tr
***  Started time :
Tue Feb 20 23:34:04 JST 1996
```

```
***  Ended time :
Tue Feb 20 23:34:12 JST 1996
#-----------------------------------------------------------------------
#   g100.f  Gamma ijkl (-3w;w,w,w)      Execute by K.Ohtawara
#   ATR Optical and Radio Communication Research Labs.
```

出力ファイル：aaa.g_type

```
-----------------------------------------------------------
   butadiene 2*2
w =  0.65  eV     CI =    15

xxxx   0.81192E-036

< contribution from individual transition types >
              (part gamma)        (%)    weight(%)
    type 1    0.19320E-035     238.0       63.3
    type 2   -0.10553E-035    -130.0       34.6
    type 3    0.00000E+000       0.0        0.0
    type 4   -0.24869E-038      -0.3        0.1
    type 5    0.00000E+000       0.0        0.0
    type 6   -0.62226E-037      -7.7        2.0

< contribution from individual m* states : top  5 >
(num) (m*)    (part gamma)        (%)    weight(%)

  1    1      0.84177E-036     103.7

    type 1    0.19301E-035     229.3       63.9
    type 2   -0.10553E-035    -125.4       35.0
    type 3    0.00000E+000       0.0        0.0
    type 4   -0.11899E-038      -0.1        0.0
    type 5    0.00000E+000       0.0        0.0
    type 6   -0.31822E-037      -3.8        1.1

  2    8     -0.21095E-037      -2.6

    type 1    0.50744E-039      -2.4        2.3
    type 2   -0.59031E-043       0.0        0.0
    type 3    0.00000E+000       0.0        0.0
    type 4   -0.28969E-039       1.4        1.3
    type 5    0.00000E+000       0.0        0.0
    type 6   -0.21312E-037     101.0       96.4

  3    4     -0.41428E-038      -0.5

    type 1    0.11730E-038     -28.3       18.1
    type 2   -0.53634E-042       0.0        0.0


yyyy   0.10706E-037
```

各タイプごとのγ値

$m^*$状態を経由するパスのγへの寄与の大きな$m^*$順ごとについての各タイプごとのγ値

出力ファイル：aaa.g_type

```
-----------------------------------------------------------------
   butadiene 2*2
w =  0.65  eV     CI =    15

xxxx   0.81192E-036

< contribution from individual transition types >
                (part gamma)        (%)     weight(%)
    type 1     0.19320E-035      238.0       63.3
    type 2    -0.10553E-035     -130.0       34.6
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.24869E-038       -0.3        0.1
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.62226E-037       -7.7        2.0

< contribution from individual m* states : top  5 >
(num) (m*)      (part gamma)        (%)     weight(%)

  1    1        0.84177E-036      103.7

    type 1     0.19301E-035      229.3       63.9
    type 2    -0.10553E-035     -125.4       35.0
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.11899E-038       -0.1        0.0
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.31822E-037       -3.8        1.1

  2    8       -0.21095E-037       -2.6

    type 1     0.50744E-039       -2.4        2.3
    type 2    -0.59031E-043        0.0        0.0
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.28969E-039        1.4        1.3
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.21312E-037      101.0       96.4

  3    4       -0.41428E-038       -0.5

    type 1     0.11730E-038      -28.3       18.1
    type 2    -0.53634E-042        0.0        0.0
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.80953E-039       19.5       12.5
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.45058E-038      108.8       69.4

  4    6       -0.32739E-038       -0.4

    type 1     0.10741E-039       -3.3        3.1
    type 2    -0.11033E-043        0.0        0.0
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.11807E-039        3.6        3.4
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.32632E-038       99.7       93.5

  5   11       -0.11229E-038       -0.1

    type 1     0.48219E-040       -4.3        4.0
    type 2    -0.36630E-044        0.0        0.0
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.74287E-040        6.6        6.1
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.10969E-038       97.7       90.0

yyyy   0.10706E-037

< contribution from individual transition types >
                (part gamma)        (%)     weight(%)
    type 1     0.17863E-037      166.8       62.2
    type 2    -0.41971E-038      -39.2       14.6
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.48001E-038      -44.8       16.7
    type 5     0.00000E+000        0.0        0.0
    type 6     0.18408E-038       17.2        6.4

< contribution from individual m* states : top  5 >
(num) (m*)      (part gamma)        (%)     weight(%)

  1    1        0.90866E-038       84.9

    type 1     0.12864E-037      141.6       69.1
    type 2    -0.25816E-038      -28.4       13.9
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.21859E-038      -24.1       11.7
    type 5     0.00000E+000        0.0        0.0
    type 6     0.99010E-039       10.9        5.3

  2    4        0.12178E-038       11.4

    type 1     0.48652E-038      399.5       53.7
    type 2    -0.16086E-038     -132.1       17.7
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.23150E-038     -190.1       25.5
    type 5     0.00000E+000        0.0        0.0
    type 6     0.27617E-039       22.7        3.0

  3    6        0.24405E-039        2.3

    type 1     0.11272E-039       46.2       14.9
    type 2    -0.67446E-041       -2.8        0.9
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.25013E-039     -102.5       33.0
    type 5     0.00000E+000        0.0        0.0
    type 6     0.38820E-039      159.1       51.2

  4    8        0.15805E-039        1.5

    type 1     0.20567E-040       13.0        8.1
    type 2    -0.22153E-042       -0.1        0.1
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.48400E-040      -30.6       19.0
    type 5     0.00000E+000        0.0        0.0
    type 6     0.18611E-039      117.8       72.9

  5   13       -0.20318E-042        0.0

    type 1     0.17861E-042      -87.9       17.2
    type 2    -0.28068E-046        0.0        0.0
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.62113E-042      305.7       59.8
    type 5     0.00000E+000        0.0        0.0
    type 6     0.23936E-042     -117.8       23.0

zzzz   0.00000E+000
-----------------------------------------------------------------
   butadiene 2*2
w =  0.00  eV     CI =    15

xxxx   0.70040E-036

< contribution from individual transition types >
                (part gamma)        (%)     weight(%)
    type 1     0.16954E-035      242.1       63.0
    type 2    -0.93613E-036     -133.7       34.8
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.22875E-038       -0.3        0.1
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.56564E-037       -8.1        2.1

< contribution from individual m* states : top  5 >
(num) (m*)      (part gamma)        (%)     weight(%)

  1    1        0.72775E-036      103.9

    type 1     0.16937E-035      232.7       63.7
    type 2    -0.93613E-036     -128.6       35.2
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.11434E-038       -0.2        0.0
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.28640E-037       -3.9        1.1

  2    8       -0.19538E-037       -2.8

    type 1     0.48945E-039       -2.5        2.4
    type 2    -0.57824E-043        0.0        0.0
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.25466E-039        1.3        1.2
    type 5     0.00000E+000        0.0        0.0
    type 6    -0.19773E-037      101.2       96.4

  3    4       -0.36187E-038       -0.5

    type 1     0.10780E-038      -29.8       18.7
    type 2    -0.51376E-042        0.0        0.0
    type 3     0.00000E+000        0.0        0.0
    type 4    -0.71530E-039       19.8       12.4
    type 5     0.00000E+000        0.0        0.0
```

```
    type 6   -0.39809E-038     110.0      68.9

 4    6      -0.29468E-038      -0.4

    type 1    0.10037E-039      -3.4       3.2
    type 2   -0.10643E-043       0.0       0.0
    type 3    0.00000E+000       0.0       0.0
    type 4   -0.10421E-039       3.5       3.3
    type 5    0.00000E+000       0.0       0.0
    type 6   -0.29429E-038      99.9      93.5

 5   11      -0.10339E-038      -0.1

    type 1    0.46931E-040      -4.5       4.2
    type 2   -0.36037E-044       0.0       0.0
    type 3    0.00000E+000       0.0       0.0
    type 4   -0.65193E-040       6.3       5.8
    type 5    0.00000E+000       0.0       0.0
    type 6   -0.10156E-038      98.2      90.1

yyyy   0.93792E-038

< contribution from individual transition types >
              (part gamma)        (%)    weight(%)
    type 1    0.15758E-037     168.0      60.9
    type 2   -0.38376E-038     -40.9      14.8
    type 3    0.00000E+000       0.0       0.0
    type 4   -0.44179E-038     -47.1      17.1
    type 5    0.00000E+000       0.0       0.0
    type 6    0.18767E-038      20.0       7.2

< contribution from individual m* states : top  5 >
(num) (m*)    (part gamma)        (%)    weight(%)

  1    1      0.76795E-038      81.9

    type 1    0.11068E-037     144.1      67.3
    type 2   -0.22900E-038     -29.8      13.9
    type 3    0.00000E+000       0.0       0.0
    type 4   -0.20887E-038     -27.2      12.7
    type 5    0.00000E+000       0.0       0.0
    type 6    0.98990E-039      12.9       6.0

  2    4      0.13317E-038      14.2

    type 1    0.45644E-038     342.8      53.5
    type 2   -0.15408E-038    -115.7      18.1
    type 3    0.00000E+000       0.0       0.0
    type 4   -0.20557E-038    -154.4      24.1
    type 5    0.00000E+000       0.0       0.0
    type 6    0.36377E-039      27.3       4.3

  3    6      0.23206E-039       2.5

    type 1    0.10584E-039      45.6      15.1
    type 2   -0.65062E-041      -2.8       0.9
    type 3    0.00000E+000       0.0       0.0
    type 4   -0.22875E-039     -98.6      32.6
    type 5    0.00000E+000       0.0       0.0
    type 6    0.36147E-039     155.8      51.4

  4    8      0.13581E-039       1.4

    type 1    0.19190E-040      14.1       8.5
    type 2   -0.21700E-042      -0.2       0.1
    type 3    0.00000E+000       0.0       0.0
    type 4   -0.44167E-040     -32.5      19.7
    type 5    0.00000E+000       0.0       0.0
    type 6    0.16100E-039     118.6      71.7

  5   13      0.17805E-042       0.0

    type 1    0.17610E-042      98.9      13.5
    type 2   -0.27874E-046      -0.0       0.0
    type 3    0.00000E+000       0.0       0.0
    type 4   -0.56314E-042    -316.3      43.2
    type 5    0.00000E+000       0.0       0.0
    type 6    0.56512E-042     317.4      43.3

zzzz   0.00000E+000
***  Data file :
-rw-rw-r--   1 ohtawara     5364 Feb 20 22:48 but2s.tr
***  Started time :
Tue Feb 20 23:34:04 JST 1996
***  Ended time :
Tue Feb 20 23:34:12 JST 1996
#-----------------------------------------------------------------------
#   g100.f  Gamma ijkl (-3w;w,w,w)       Execute by K.Ohtawara
#   ATR Optical and Radio Communication Research Labs.
```

## D3. 必要メモリ量と実行時間

各プログラムの実行時間は、$n$- オクタテトラエン: $C_8H_{10}$に対するConvex C 240(理論性能：4CPU当り0.2GFLOPS, 主メモリ512MB)での1CPU換算のCPU時間を示す。

1) CNDO/S-SDCI プログラム

| CI 形式 [a)] | CI 配置数 | 実行時メモリ量 [b)] | 実行時間 |
|---|---|---|---|
| 5 × 5 SDCI | 351 | 25 MB | 1 時間 26 分 |
| 6 × 6 SDCI | 703 | 59 MB | 49 時間 34 分 |
| 8 × 8 SDCI | 2145 | 422 MB | |
| 9 × 9 SDCI | 3403 | 1.04 GB | |
| 10 × 10 SDCI | 5151 | 1.72 GB | |

a) 11 × 11 以上の SDCI は、必要メモリ量が 2GB を越えるため、object code を link できない。
b) 最大原子個数を 50 個とした場合。

2) 2 次超分極率計算プログラム

| CI 形式 | CI 配置数 | 実行時メモリ量 | 実行時間 |
|---|---|---|---|
| 5 × 5 SDCI | 351 | 382 MB[a)] | 2 時間 30 分 |
| 6 × 6 SDCI | 703 | 17 MB[b)] | 42 時間 35 分 |
| 8 × 8 SDCI | 2145 | 138 MB | |
| 9 × 9 SDCI | 3403 | 399 MB | |
| 10 × 10 SDCI | 5151 | 777 MB | |

a) 5 × 5 以下のための 2 次超分極率計算プログラムは、各遷移パスごとの超分極率を絶対値順に並べるための sort routine で、配列を大きく取っているために、必要メモリ量が多い。
b) 5 × 5 以上のための 2 次超分極率計算プログラムは、メモリ量の節約をしているが、約 2 倍の計算時間が必要となる。