TR-I-0011

Interactive Menu System for Speech Workbench

対話型操作環境をワークベンチごとに設定できる
メニューシステムの作成

Katsuteru Maruyama,Takeshi Kawabata
丸山活輝、川端豪

1987.12

概要

音声処理ワークベンチversion.2のメニューシステムについて報告する。多様な音声信号処理の各段階において、各自の要求に応じてカスタマイズされた操作環境を実現することは、音声研究の効率化のために重要である。本報告では筆者が本研究所の音声研究者の意見を取りまとめ設計及び製作した、容易に自分の環境を設計し組み込むことが可能な汎用メニューシステムの使用方法について述べる。

ATR Interpreting Telephony Research Laboratories
ATR 自動翻訳電話研究所

目　次　　ページ

1. 概要

1.1 機能　1

1.2 処理の流れ　1

2. ソフトウェア構成

2.1 概要　2

2.2 ブロック・チャート　3

2.3 ジェネラル・フローチャート　5

2.4 各関数の説明

WB_Menu　7

WB_CloseMenu　7

WB_CreateMenu　7

WB_Erase　8

WB_Exec　8

WB_ExeFunc　9

WB_GetAddo　9

WB_GetGlobal　10

WB_GetPrmf　10

WB_GetRPrmf　11

WB_GetVer　11

WB_History　12

WB_InitGlobal　12

WB_InitItem　13

WB_InitSlice　13

WB_MakeName 13
WB_MakePrmf 14
WB_MakeRPrmf 14
WB_Mdfystring 15
WB_PrmMenu 15
WB_PulldownMenu 16
WB_ReadData 17
WB_ReadFunc 17
WB_ReadHist 18
WB_ReadItem 18
WB_ReadMain 19
WB_ReadSub 19
WB_RequestMenu 20
WB_Reset 20
WB_SetFile 21
WB_SetPrm 21
WB_SetVer 22
WB_SubMenu 23
WB_UpdateHist 24
WB_UpdateItem 24
WB_kill 25
WB_mkfnam 25
WB_GetData 25
WB_string 26
2.5 取り込みファイル 26

3. 入出力ファイル

3.1 メニュー定義ファイル 27

3.2 パラメータ定義ファイル 29

3.3 パラメータ・ファイル 32

3.4 ヒストリー・ファイル 32

3.5 ストリングス・データ・ファイル 32

3.6 データ・ファイル・バージョン管理ファイル 32

3.7 スライス・プログラム・ＰＩＤ・ファイル 33

4. 参照テーブル

4.1 ＰＩＤ・テーブル 33

4.2 パラメータ・ファイル・テーブル 33

4.2 グローバル・ファンクション・テーブル 33

5. 機能説明

5.1 機能プログラムの起動 34

5.2 機能プログラムのパラメータ・セット 35

5.3 ヒストリー処理 36

5.4 画面消去 37

5.5 リセット処理 38

5.6 終了処理 38

6. 制限／注意事項 39

7. 操作説明

7.1 操作概略 40

7.2 起動方法 40

7.3　メニューオペレーションの説明　40

8．エラー・メッセージの説明　45

9．機能プログラムの追加方法　47

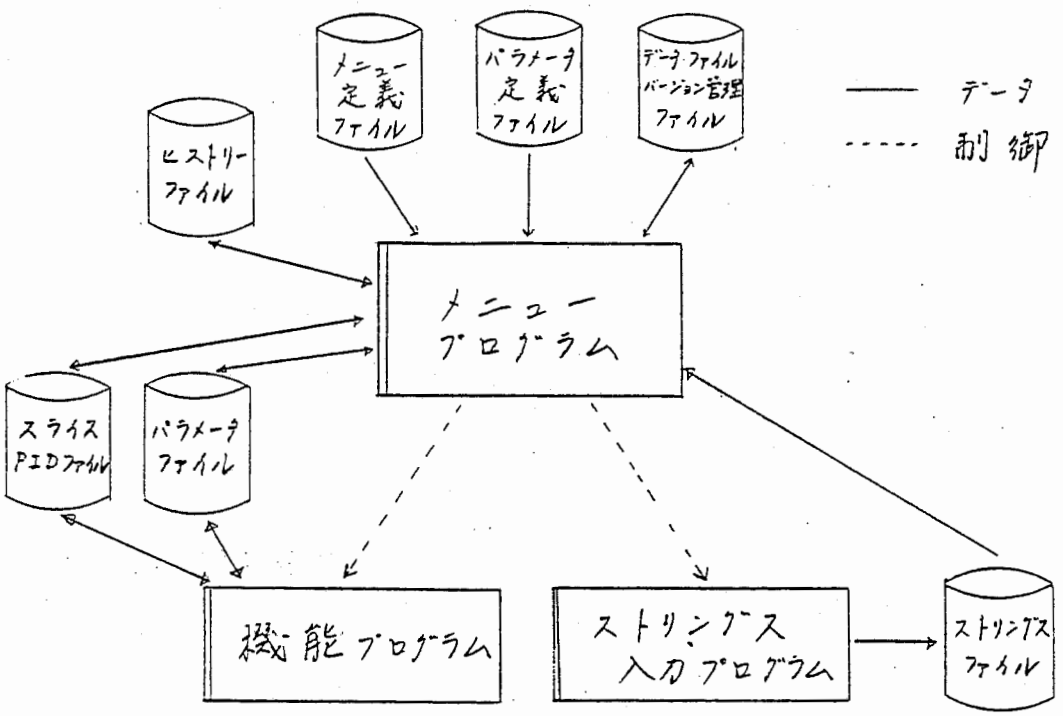
1. 概要

1.1 機能

本メニュー・プログラムは、ある形式に則ったファイルを読み込み、メニュー指示標をディスプレイに出力する。その後、ユーザの指示（マウス ミドル・ボタンのヒット）に従い、下記の処理を行う。

- 機能（ユーザ作成）プログラムを起動する。
- 各機能プログラムのパラメータ値の変更及び、ファイルの設定。
- ヒストリー（機能プログラム実行の履歴及び、再起動）処理。
- 画面の消去。
- リセット（初期状態に戻す）処理。
- 終了処理。

1.2 処理の流れ



機能プログラム及び、ストリングス入力プログラムは、子プロセスとして起動する。

2. ソフトウェア構成

2.1 概要

本ソフトウェアは、メニューとストリングス入力の２つのプログラムより構成されている。

1. メニュー・プログラム

メインを含め、３７モジュールで構成されている。
メニュー指示標の表示、メニュー選択処理は、X（ウィンドウ・システム）ライブラリーを使用している。

2. ストリングス入力プログラム

ＧＫＳのストリングス関数を使用して、文字入力処理を行っている。
以下の関数をコールしている。

(1). ULTRIX Graphical Kernel System （ＧＫＳ）

グラフィック・ソフトウェア関数
（詳細は、ＤＥＣ社 ＧＫＳ マニュアル参照）

(2). ＧＫＳ コネクト関数

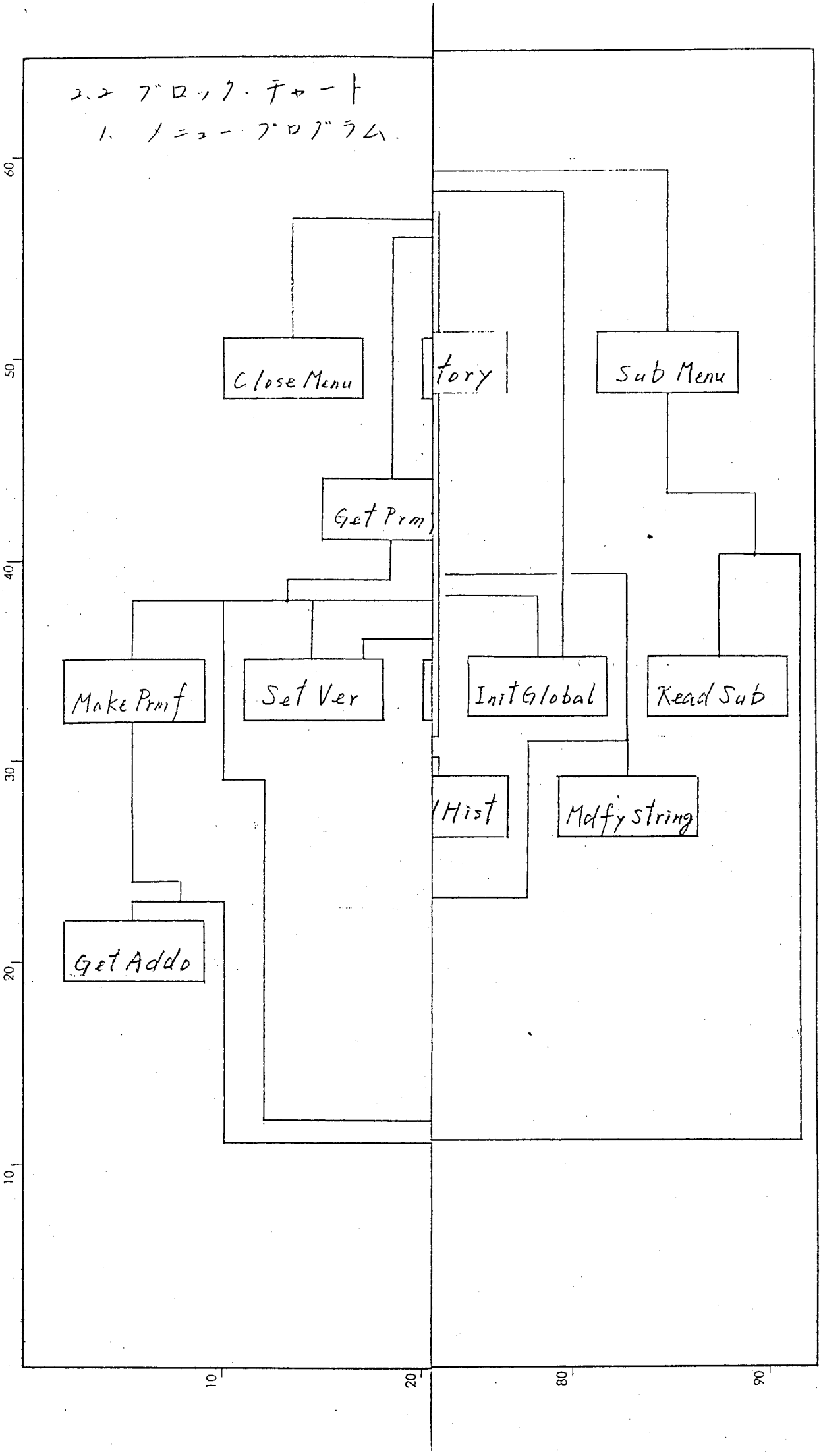
ＧＫＳ を、ユーザ・プログラムから可能な限り call by value 形式のパラメータで、コール出来る様に変更した関数である。
（ＧＫＳ コネクト関数コーリング・シーケンス、パラメーター覧書参照）

(3). ウィンドウ・システム関数

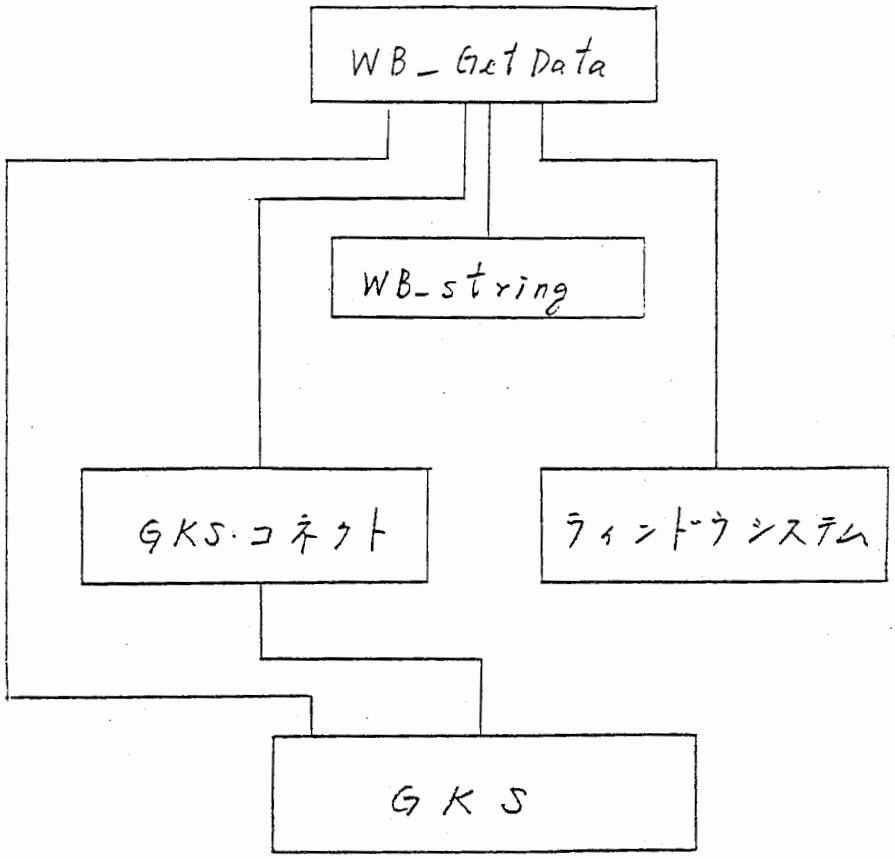
ウィンドウ（四角形の枠）を、ディスプレイ上に位置決めを行うソフトウェアである。
（詳細は、ウィンドウ・システム関数 仕様書 を参照）

2.2 ブロック・チャート

1. メニュー・プログラム.

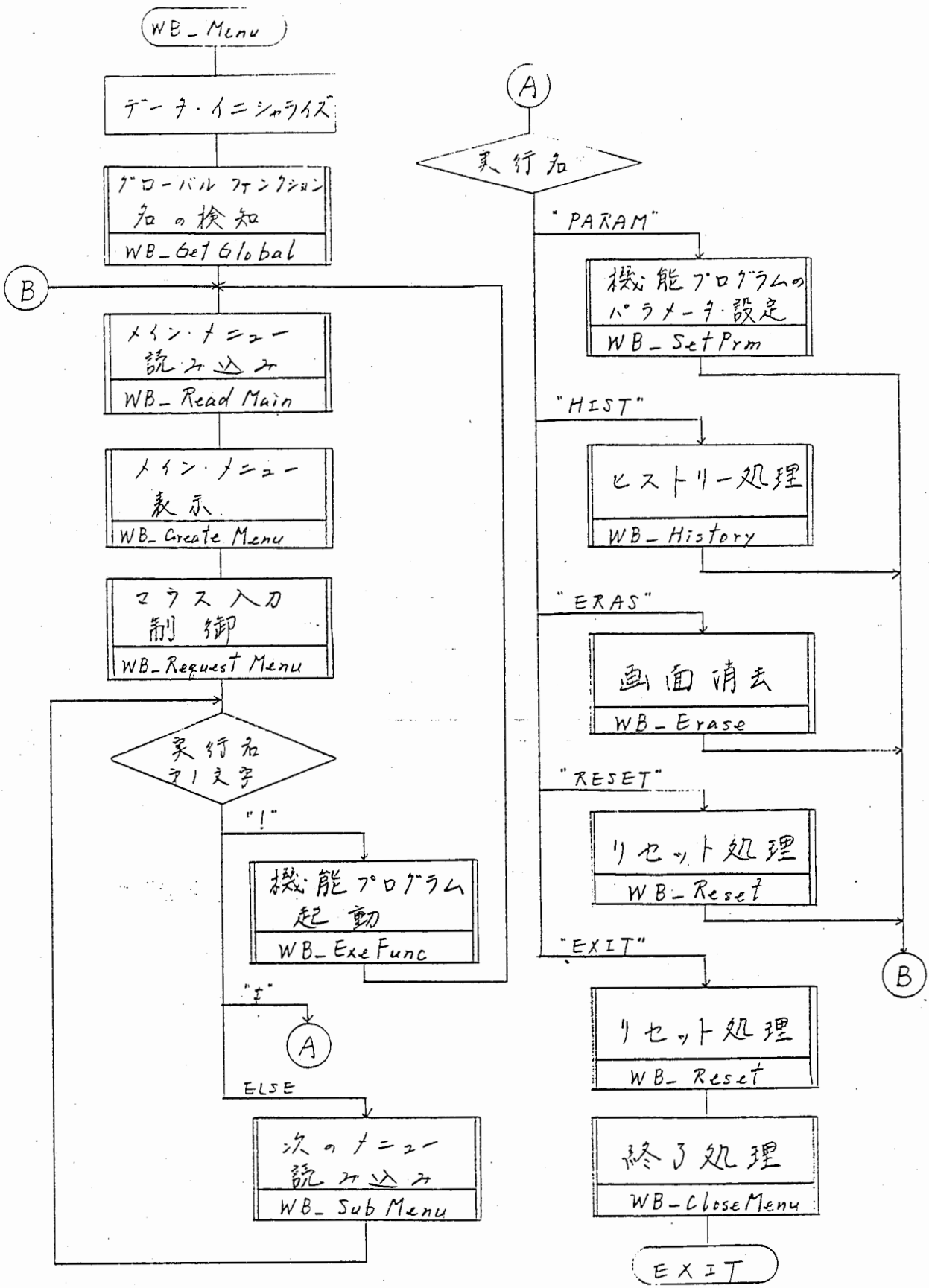


2. ストリングス入力プログラム

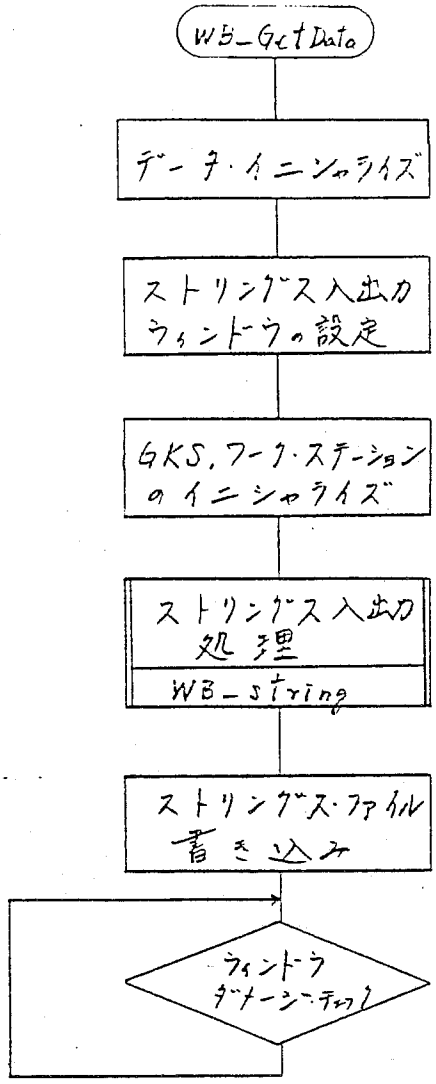


2.3 ジェネラル・フローチャート

1. メニュー・プログラム



2. ストリングス入力プログラム



2.4 各関数の説明

1．メニュー・プログラムの関数

(1). WB_Menu　（メイン・プログラム）

1).機能

・機能、ストリングス入力プログラムを制御（起動／停止）する。
・パラメータ設定、ヒストリー処理及び、リセット処理を制御する。
・各種データの入手、メニュー入出力に関する制御を行う。
・データ及び、テーブル等のイニシャライズ。
・データ・ファイル名を、アーギュメントで与えることが出来る。

(2). WB_CloseMenu

1).機能

・X（ウィンドウ・システム）ライブラリーの使用終了処理。

2).コーリング・シーケンス

```
WB_CloseMenu ( )
```

(3). WB_CreateMenu

1).機能

・メニュー指示標のウィンドウを作成する。そして、作成した
ウィンドウの識別子を返す。

2).コーリング・シーケンス

```
Window WB_CreateMenu ( window, n_main, width )
```

window　：　作成するウィンドウの親ウィンドウ。
(input) Window window

n_main　：　指示標の項目数。
(input) integer value

*width　：　指示標１項目当たりのウィンドウ幅。（ＤＣ）
(output) integer reference

(4). ＷＢ＿Ｅｒａｓｅ

1).機能

・現在ディスプレイに表示している、機能プログラムを全て消去する。
（機能プログラムを Ｋｉｌｌ する）

2).コーリング・シーケンス

WB_Erase (pt)

pt[] : ＰＩＤ・テーブルの配列。
(input/output) PIDTab pt[]

(5). ＷＢ＿Ｅｘｅｃ

1).機能

・指定のパラメータ・ファイルで、実行されていた機能プログラムを検出し、ヒストリー・スイッチ（－ｈ）を付けて起動させる。
・ＰＩＤ・テーブルにプロセスＩＤを格納する。

2).コーリング・シーケンス

WB_Exec (cmd, file, pt, gt)

*cmd : 機能プログラム実行コマンド。
(input) character reference

*file : パラメータ・ファイル名。
(input) character reference

pt[] : ＰＩＤ・テーブルの配列。
(output) PIDTab pt[]

gt[] : グローバル・ファンクション・テーブルの配列。
(input) GblTab gt[]

(6). ＷＢ＿ＥｘｅＦｕｎｃ

1).機能

・指定された機能プログラムを起動させる。そして、ＰＩＤ・テーブルにプロセスＩＤを格納する。
・機能プログラムの停止、ヒストリー・ファイルの更新を制御する。

2).コーリング・シーケンス

WB_ExeFunc (pt, func, cmd, file)

pt[] : ＰＩＤ・テーブルの配列。
(input) PIDTab gt[]

*func : ファンクション名。
(input) character reference

*cmd : 機能プログラムの実行コマンド。
(input) character reference

*file : パラメータ・ファイル名。
(input) character reference

(7). ＷＢ＿ＧｅｔＡｄｄｏ

1).機能

・パラメータ定義ファイルより、指定されたファンクションの最初のデータを検出し、アーギュメントに設定する。

2).コーリング・シーケンス

WB_GetAddo (fp, func, data)

*fp : ファイル構造体のポインタ。
(input) FILE *fp

*func : ファンクション名。
(input) character reference

*data : データ格納エリア。
(output) character reference

(8). ＷＢ＿ＧｅｔＧｌｏｂａｌ

1).機能

・メニュー定義ファイルに、グローバル・ファンクションとして定義されている全てのファンクションを検索し、グローバル・ファンクション・テーブルに登録する。
・グローバル・ファンクションの数を返す。

2).コーリング・シーケンス

```
int WB_GetGlobal ( file, gt )
```

*file　：　メニュー定義ファイル名
(input) character reference

gt[]　：　グローバル・ファンクション・テーブルの配列。
(output) GblTab gt[]

(9). ＷＢ＿ＧｅｔＰｒｍｆ

1).機能

・指定の機能プログラムの最新パラメータ・ファイル名のポインタを返す。
・パラメータ・ファイルの管理（ファイル作成、ファイル名、バージョンＮｏ.）を行う。

2).コーリング・シーケンス

```
char *WB_GetPrmf ( vt, gt, func, file )
```

vt[]　：　パラメータ・ファイル・バージョン管理テーブルの配列。
(input/output) VerTab vt[]

gt[]　：　グローバル・ファンクション・テーブルの配列。
(input) GblTab gt[]

*func　：　ファンクション名。
(input) character reference

*file　：　パラメータ・ファイル名。
(input) character reference

(10). ＷＢ＿ＧｅｔＲＰｒｍｆ

1).機能

・指定機能プログラムの、最新バージョン・パラメータ・ファイル名のポインタを返す。又、バージョンＮｏ.もアーギュメントに設定する。
・パラメータ・ファイルのバージョンＮｏ.、パラメータ定義ファイル作成の管理を行う。

2).コーリング・シーケンス

```
char *WB_GetRPrmf ( vt, func, file, vn )
```

vt[] : パラメータ・ファイル・バージョン管理テーブルの配列。
(input/output) VerTab vt[]

*func : ファンクション名。
(input) character reference

*file : パラメータ・ファイル名。
(input) character reference

*vn : パラメータ・ファイル・バージョンＮｏ.。
(output) integer reference

(11). ＷＢ＿ＧｅｔＶｅｒ

1).機能

・指定パラメータ・ファイルのバージョンＮｏ.をアーギュメントに設定する。
・正常の時０、異常の時－１を返す。

2).コーリング・シーケンス

```
int WB_GetVer ( vt, file, vn )
```

vt[] : パラメータ・ファイル・バージョン管理テーブルの配列。
(input) VerTab vt[]

*file : パラメータ・ファイル名。
(input) character reference

*vn : パラメータ・ファイル・バージョンＮｏ.。
(output) integer reference

(12). ＷＢ＿Ｈｉｓｔｏｒｙ

1).機能

・ヒストリー処理を制御する。
・ヒストリー・ファイルの読み込み、履歴の選択及び、履歴からの再起動の制御を行う。

2).コーリング・シーケンス

```
WB_History ( window, width, wp, pt, gt )
```

window　：　メニュー指示標のウィンドウ識別子。
(input) Window window

width　：　指示標１項目当たりのウィンドウ幅。（ＤＣ）
(input) integer value

wp　：　メニュー指示標選択時のウィンドウ位置。
(input) integer value

pt[]　：　ＰＩＤ・テーブルの配列。
(input/output) PIDTab pt[]

gt[]　：　グローバル・ファンクション・テーブルの配列。
(input) GblTab gt[]

(13). ＷＢ＿ＩｎｉｔＧｌｏｂａｌ

1).機能

・データ・ファイル・バージョン管理ファイルをイニシャライズする。
・正常の時０、異常の時－１を返す。

2).コーリング・シーケンス

```
int WB_InitGlobal ( file )
```

*file　：　データ・ファイル・バージョンＮＯ．管理ファイル名
(input) character reference

(14). ＷＢ＿ＩｎｉｔＩｔｅｍ

1).機能

・パラメータ定義ファイルより、指定されたファンクションの、項目名（”ＣＡＮＣＥＬ”を追加して）と実行名をアーギュメントに設定する。
・項目数を返す。

2).コーリング・シーケンス

```
int WB_InitItem ( fp, func, item, exe )

*fp        :   ファイル構造体のポインタ。
               (input) FILE *fp

*func      :   ファンクション名。
               (input) character reference

item[][]   :   項目名格納エリア。
               (output) character aray

exe[][]    :   実行部格納エリア。
               (output) character aray
```

(15). ＷＢ＿ＩｎｉｔＳｌｉｃｅ

1).機能

・スライス・ＰＩＤ・ファイルをイニシャライズする。

2).コーリング・シーケンス

```
WB_InitSlice ( )
```

(16). ＷＢ＿ＭａｋｅＮａｍｅ

1).機能

・指定されたファンクション名より、パラメータ・ファイル名を作成し、ポインタを返す。

2).コーリング・シーケンス

```
char *WB_MakeName ( func )

*func      :   ファンクション名。
               (input) character reference
```

(17). ＷＢ＿ＭａｋｅＰｒｍｆ

1).機能

・指定ファンクションの、最新バージョン・パラメータ・ファイル名のポインタを返す。又、バージョンＮｏ.もアーギュメントに設定する。
・パラメータ・ファイルを作成する。

2).コーリング・シーケンス

```
char *WB_MakePrmf ( gt, func, file, vn )
```

gt[] : グローバル・ファンクション・テーブルの配列。
(input) gblTab gt[]

*func : ファンクション名。
(input) character reference

*file : パラメータ・ファイル名。
(input) character reference

*vn : パラメータ・ファイル・バージョンＮｏ.。
(output) integer reference

(18). ＷＢ＿ＭａｋｅＲＰｒｍｆ

1).機能

・指定ファンクションの、最新バージョン・パラメータ・定義ファイル名のポインタを返す。又、バージョンＮｏ.もアーギュメントに設定する。
・パラメータ・ファイルより、パラメータ・定義ファイルを作成する。

2).コーリング・シーケンス

```
char *WB_MakeRPrmf ( func, file, vn )
```

*func : ファンクション名。
(input) character reference

*file : パラメータ・ファイル名。
(input) character reference

*vn : パラメータ・ファイル・バージョンＮｏ.。
(output) integer reference

(19). ＷＢ＿ＭｄｆｙＳｔｒｉｎｇ

1).機能

・入力アーギュメントの文字列の頭に ”（ダブルクォテーション）をつけ、出力アーギュメントに設定する。

2).コーリング・シーケンス

```
char *WB_Mdfystring ( in_string, ot_string )
```

*in_string : 文字データ。
(input) character reference

*ot_string : 文字データ。
(output) character reference

(20). ＷＢ＿ＰｒｍＭｅｎｕ

1).機能

・パラメータ・データ設定の制御を行う。
・ストリングス入力プログラムを制御（起動／停止）する。
・パラメタ・セットのステータス（設定：１，未設定：０）を返す。

2).コーリング・シーケンス

```
int WB_PrmMenu ( window, width, wp, func, file )
```

window : メニュー指示標のウィンドウ識別子。
(input) Window window

width : 指示標１項目当たりのウィンドウ幅。（ＤＣ）
(input) integer value

wp : メニュー指示標選択時のウィンドウ位置。
(input) integer value

*func : ファンクション名。
(input) character reference

*file : パラメータ定義ファイル名。
(input) character reference

(21). WB_PulldownMenu

1).機能

・アーギュメントの指定（項目名、出力位置・サイズ）に従って、
プールダウン指示標を表示する。
・ユーザが選択した、項目の番号を返す。

2).コーリング・シーケンス

```
int WB_PulldownMenu ( window, x, y, width, n, item )
```

window : 作成するウィンドウの親ウィンドウ。
(input) Window window

x : 作成するウィンドウのX座標（左端）。（ＤＣ）

y : 作成するウィンドウのY座標（上端）。（ＤＣ）

n : 項目数。
(input) integer value

item[][] : 項目名。
(input) character array

*width : 指示標１項目当たりのウィンドウ幅。（ＤＣ）
(output) integer reference

(22). ＷＢ＿ＲｅａｄＤａｔａ

1).機能

・パラメータ定義ファイルより、指定ファンクションの項目名、データ部をアーギュメントに設定する。
・項目数を返す。

2).コーリング・シーケンス

```
int WB_ReadData ( fp, func, item, data )
```

*fp　　：　ファイル構造体のポインタ。
(input) FILE *fp

*func　　：　ファンクション名。
(input) character reference

item[][]　：　項目名格納エリア。
(output) char item[][]

data[][]　：　データ部格納エリア。
(output) char exe[][]

(23). ＷＢ＿ＲｅａｄＦｕｎｃ

1).機能

・パラメータ定義ファイルより、ファンクション項目部（項目名（”ｎｏｐ”を追加して）、実行部）をアーギュメントに設定する。
・項目数を返す。

2).コーリング・シーケンス

```
int WB_ReadFunc ( fp, item, exe )
```

*fp　　：　ファイル構造体のポインタ。
(input) FILE *fp

item[][]　：　項目名格納エリア。
(output) char item[][]

exe[][]　：　実行部格納エリア。
(output) char exe[][]

(24). ＷＢ＿ＲｅａｄＨｉｓｔ

1).機能

・ヒストリー・ファイルより、履歴の情報（コマンド、パラメータファイル名）、履歴指示標の項目をアーギュメントに設定する。
・履歴指示標の項目数を返す。

2).コーリング・シーケンス

```
int WB_ReadHist ( fp, cmd, file, hist )
```

*fp : ファイル構造体のポインタ。
(input) FILE *fp

cmd[][] : コマンド格納エリア。
(output) character array

file[][] : パラメータ・ファイル名格納エリア。
(output) character array

hist[][] : 履歴指示標の項目名格納エリア。
(output) character array

(25). ＷＢ＿ＲｅａｄＩｔｅｍ

1).機能

・パラメータ定義ファイルより、指定ファンクションの項目名、データ部をアーギュメントに設定する。
・項目数を返す。
※ WB_InitItem 関数は "CANCEL" 項目を付けるが、この関数は付けない。

2).コーリング・シーケンス

```
int WB_ReadItem ( fp, func, item, data )
```

*fp : ファイル構造体のポインタ。
(input) FILE *fp

*func : ファンクション名。
(input) character reference

item[][] : 項目名格納エリア。
(output) char item[][]

data[][] : データ部格納エリア。
(output) char exe[][]

(26). ＷＢ＿ＲｅａｄＭａｉｎ

1).機能

・メニュー定義ファイルより、メイン・メニュー部（項目名、実行部）を読み込む。
・項目数を返す。

2).コーリング・シーケンス

```
int WB_ReadMain ( fp, item, exe )
```

*fp	:	ファイル構造体のポインタ。 (input) FILE *fp
item[][]	:	指示標の項目名格納エリア。 (output) char item[][]
exe[][]	:	実行部格納エリア。 (output) char exe[][]

(27). ＷＢ＿ＲｅａｄＳｕｂ

1).機能

・メニュー定義ファイルより、指定インデックス項目のメニュー情報（項目名、実行部、パラメータ・ファイル名）をアーギュメントに設定する。
・項目数を返す。

2).コーリング・シーケンス

```
int WB_ReadSub ( fp, d_exe, item, exe, file )
```

*fp	:	ファイル構造体のポインタ。 (input) FILE *fp
*d_exe	:	インデックス項目名。 (input) character reference
item[][]	:	項目名格納エリア。 (output) character array
exe[][]	:	実行名格納エリア。 (output) character array
file[][]	:	パラメータ・ファイル名格納エリア。 (output) character array

(28). ＷＢ＿ＲｅｑｕｅｓｔＭｅｎｕ

1).機能

・メニュー指示標を出力する。
・マウス入力（ミドル・ボタンのヒット）要求を出す。
・ユーザが選択した、指示標項目の番号を返す。

2).コーリング・シーケンス

```
int WB_RequestMenu ( n_main, item, width )
```

n_main	：	指示標の項目数。 (input) integer value
item[][]	：	指示標の項目名格納エリア。 (input) char item[][]
width	：	指示標１項目当たりのウィンドウ幅。（ＤＣ） (input) integer value

(29). ＷＢ＿Ｒｅｓｅｔ

1).機能

・本プログラムを初期状態（起動時）に戻す。
・ディスプレ画面、ヒストリー・ファイルを消去する。
・ＰＩＤ・テーブル、パラメータ・ファイル・バージョン管理テーブルのイニシャライズ。

2).コーリング・シーケンス

```
WB_Reset ( pt, gt )
```

pt[]	：	ＰＩＤ・テーブルの配列。 (input/output) PIDTab pt[]
vt[]	：	パラメータ・ファイル・バージョン管理テーブルの配列。 (input/output) VerTab vt[]

(30). ＷＢ＿ＳｅｔＦｉｌｅ

1).機能

・本メニュー・プログラムを起動時に、アーギュメントで指定した、入力データ・ファイル名を該当のパラメータ・ファイルに設定する。

2).コーリング・シーケンス

WB_SetFile (file, in_file, func, gt)

*file : パラメータ・ファイル名。
(input) character reference

*in_file : 入力データ・ファイル名。
(input) character reference

*func : ファンクション名。
(input) character reference

gt[] : グローバル・ファンクション・テーブルの配列。
(input) GblTab gt[]

(31). ＷＢ＿ＳｅｔＰｒｍ

1).機能

・機能プログラムのパラメータ値、ファイル名の設定を制御する。

2).コーリング・シーケンス

WB_SetPrm (window, width, wp, vt)

window : 作成するウィンドウの親ウィンドウ。
(input) Window window

*width : 指示標１項目当たりのウィンドウ幅。（ＤＣ）
(output) integer reference

wp : メニュー指示標選択時のウィンドウ位置。
(input) integer value

vt[] : ファイル・バージョン管理テーブルの配列。
(input/output) VerTab vt[]

(32). ＷＢ＿ＳｅｔＶｅｒ

1).機能

・ファイル・バージョン管理テーブルに指定パラメータ・ファイルのバージョンＮｏ.を設定する。
・正常の時０、異常の時－１を返す。

2).コーリング・シーケンス

```
int WB_SetVer ( vt, file, vn )
```

vt[]　：　パラメータ・ファイル・バージョン管理テーブルの配列。
(input/output) VerTab vt[]

*file　：　パラメータ・ファイル名。
(input) character reference

vn　：　パラメータ・ファイル・バージョンＮｏ.。
(output) integer value

(33). ＷＢ＿ＳｕｂＭｅｎｕ

1).機能

・メニュー定義ファイルより、指定ファンクションのメニュー情報（項目名、実行部、パラメータ・ファイル名）設定の制御を行う。
・プールダウン・メニューの出力／選択の管理をする。
・ユーザが選択した、プールダウン・メニューの項目番号を返す。

2).コーリング・シーケンス

```
int WB_SubMenu ( window, width, wp, func, item, exe, file )
```

window　：　メニュー指示標のウィンドウ識別子。
(input) Window window

width　：　指示標１項目当たりのウィンドウ幅。（ＤＣ）
(input) integer value

wp　：　メニュー指示標選択時のウィンドウ位置。
(input) integer value

*func　：　ファンクション名。
(input) character reference

item[][]　：　項目名格納エリア。
(output) character array

exe[][]　：　実行名格納エリア。
(output) character array

*file　：　パラメータ定義ファイル名。
(output) character reference

(34). WB_UpdateHist

1).機能

・ヒストリー・ファイルに指定の実行コマンド、パラメータ・ファイル名を登録する。
・ヒストリー・ファイルに、書かれている履歴数。又は、異常時に負値を返す。

2).コーリング・シーケンス

```
int WB_UpdateHist ( cmd, file  )
```

*cmd : 機能プログラム実行コマンド。
(input) character reference

*file : パラメータ・ファイル名。
(input) character reference

(35). WB_UpdateItem

1).機能

・パラメータ定義ファイルに、指定ファンクションの項目にデータを設定する。

2).コーリング・シーケンス

```
WB_UpdateItem ( file, func, item, data  )
```

*file : パラメータ定義ファイル名。
(input) character reference

*func : 検索するファンクション名。
(input) character reference

*item : 検索する項目名。
(input) character reference

*data : 設定するデータ。
(input) character array

(36). ＷＢ＿ｋｉｌｌ

1).機能

・指定されたＰＩＤを停止（ｋｉｌｌ）する。そして、アーギュメントに０を設定する。

2).コーリング・シーケンス

```
WB_kill ( pid )

 *pid        :   プロセスＩＤ。
                 (input/output) integer reference
```

(37). ＷＢ＿ｍｋｆｎａｍｅ

1).機能

・文字列の語尾に ”．バージョン番号” を付けた、ファイル名をアーギュメントに設定する。
・ファイル名のポインタを返す。

2).コーリング・シーケンス

```
char *WB_mkfname ( strng, no )

 *strng      :   文字列。
                 (input/output) character reference

 no          :   バージョン番号。
                 (input) integer value
```

２．ストリングス入力プログラムの関数

(1). ＷＢ＿ＧｅｔＤａｔａ　（ストリングス入力プログラム・メイン）

1).機能

・ＧＫＳの初期化、ストリングス入力ウィンドウの設定を制御する。
・ストリングス入力の制御を行う。
・入力ストリングスをファイルに書き込む。

(2). ＷＢ＿ｓｔｒｉｎｇ

1).機能

・現在の文字データの値を出力し、入力文字データをアーギュメントに設定する。

2).コーリング・シーケンス

WB_string (wsid, devno, echo_type, strng, size, status)

wsid	:	ワークステーション識別子。 (input) integer value
devno	:	デバイスＮｏ．。 (input) integer value
echo_type	:	エコー・タイプ。 (input) integer value
*strng	:	文字列データ。 (input/output) character reference
size	:	文字列データのサイズ。（バイト） (input) integer value
*status	:	リターン・ステイタス。 (input) integer reference

2.5 取り込みファイル

本プログラム（関数）中で、プリ・プロセッサ #include によって取り込むファイルに付いて説明する。
尚、システム、ＧＫＳ及び、Ｘ（ウィンドウ・システム）に関するファイルは除く。

(1). ＷＢ＿ｌｉｍｉｔｓ．ｈ

・各データ値及び、バッファ・サイズの最大値を定義している。
・各種テーブル（ＰＩＤ、パラメータ・ファイル・バージョン管理、グローバル・ファンクション）を定義している。

(2). ＷＢ＿ｆｉｌｅｓ．ｈ

・本プログラムで、入出力するファイル名を定義している。

3. 入出力ファイル

3.1 メニュー定義ファイル （ .wbrc ）

本プログラムの起動前に、ユーザ自身が作成するファイルである。
このファイルの記述に従って、メニュー指示標、プールダウン・メニュー指示標を出力／処理する。
メニュー指示標は、メイン・メニュー部の情報、プールダウン指示標はサブ・メニュー部の情報より作成される。

1).構造

・メイン・メニュー部は１つだが、サブ・メニュー部は複数の記述ができる。
・メイン／サブ・メニュー部は、使用用途は異なるが、構造は等しい。
・５つの要素［メニュー名、項目数、項目名、実行部］で構成され、項目名と実行部が、処理単位の対となる。

メニュー名　：　メイン・メニューは、"MAIN"（固定）で示される。
サブ・メニューは "MAIN" 以外で示される。
（英数字とアンダーバーのみで記述する。）

項目数　：　項目名及び、実行部の数。

項目名　：　メニュー／プールダウン指示標の表示部分になる。
（英数字とアンダーバーのみで記述する。）

実行部　：　下記に示す３つの形式がある。

・他のサブ・メニューに移る　：　メニュー名。
・機能プログラムを実行する　：　!実行イメージのファイル名とパラメータ・ファイル名
・本プログラムの機能を実行する　：　$機能名

機能名　下記の５機能がある。
PARAM　：　機能プログラムのパラメータ設定
HIST　：　ヒストリー処理
ERASE　：　画面の消去
RESET　：　リセット処理
EXIT　：　終了処理

2).記述形式

```
メニュー名      項目数[n]
                項目名[1]     実行部[1]
                項目名[2]     実行部[2]
                   ¦             ¦
                   ¦             ¦
                項目名[n]     実行部[n]
```

上記の様に、メニュー名と項目数、項目名と実行部の区切りは、スペース又は、タブ（複数でも可）で、他の要素の区切りはキャリッジ・リターンである。

3).記述例

```
MAIN        4
            MENU        MENU
            PARAM       $PARAM
            HISTORY     $HIST
            ELSE        ELSE
MENU        3
            AD          !ad     .ad
            DA          !da     .da
            GLOBAL      GLOBAL
GLOBAL      3
            WAVE        !WB_wave        .global
            CORREL      !WB_correl      .global
            POWER       !WB_power       .global
ELSE        3
            ERASE       $ERASE
            RESET       $RESET
            EXIT        $EXIT
```

3.2 パラメータ定義ファイル （ .wbrp ）

本プログラムの起動前に、ユーザ自身が作成するファイルである。
このファイルの記述に従って、各機能プログラムのパラメータ設定処理が行われる。

1).構造

・機能プログラム記述部と、パラメータ記述部に分かれる。
・機能プログラム記述部は１つだが、パラメータ記述部は複数の記述ができる。
・機能プログラム／パラメータ記述部とでは、記述内容は異なるが、構造は等しい。
・５つの要素［ファンクション名、項目数、項目名、データ部］で構成され、項目名とデータ部が、処理単位の対となる。

・ファンクション名
　機能プログラム記述部　：　”FUNC”（固定）で示される。

　パラメータ記述部　　：　”FUNC” 以外で示される。
　　機能プログラム記述部のデータ部及び、
　　メニュー定義ファイルのパラメータ・ファイル名項目名と統一を取らねばならない。
　　（英数字とアンダーバーのみで記述する。）

・項目数　　：　項目名及び、データ部の数。
　　（機能プログラム／パラメータ記述部共通）

・項目名　　：　パラメータ設定プールダウン指示標の表示部分となる。
　　（英数字とアンダーバーのみで記述する。）
　機能プログラム記述部　：　機能プログラム名。
　パラメータ記述部　　：　パラメータ項目名。

・データ部
　機能プログラム記述部
　　ファンクション名（パラメータ記述部）を記述する。
　　このファンクション名より、パラメータ・ファイル名を作成する。
　　（メニュー定義ファイル実行部の、パラメータ・ファイル名と統一せねばならない。）

　パラメータ記述部　　データ値を記述する。
　　文字／文字列データ　　：　”文字列
　　数値データ　　：　#数値
　　プールダウン指示標でデータを選択　：　@インデックス項目
　　（この機能は、起動直後のパラメータ設定時にのみ使用できる。）

　　インデックス項目
　　英数字とアンダーバーのみで記述する。

2).記述形式

```
ファンクション名    項目数[n]
                    項目名[1]    データ部[1]
                    項目名[2]    データ部[2]
                      ¦            ¦
                      ¦            ¦
                    項目名[n]    データ部[n]
```

上記の様に、ファンクション名と項目数、項目名とデータ部の区切りは、スペース又は、タブ（複数でも可）で、他の要素の区切りはキャリッジ・リターンである。

3).記述例

```
FUNC        3
            AD          AD
            DA          DA
            GLOBAL      GLOBAL
AD          3
            file        "adfile
            start       #0
            end         #1000
            sampling    @sample
sample      3
            12khz       #12
            20khz       #20
            30khz       #30
DA          3
            file        "dafile
            start       #0
            end         #100
GLOBAL      3
            WAVE        WAVE
            CORREL      CORREL
            POWER       POWER
WAVE        9
            file        "/data/speech/rikei/data
            fp          #0
            intval      #819
            hz          #20
            scale       #1.0
            width       #0
            height      #0
            x           #0
            y           #0
```

```
CORREL     4
           file      "0
           ns        #64
           nw        #256
           window    @anl=wind
POWER      2
           height    #0
           y         #0
anl_wind   3
           Ham       #0
           Han       #1
           Rect      #2
```

※ "GLOBAL"（ファンクション名）の記述部と、その項目名の各パラメータ記述部（"WAVE","CORREL","POWER"）は連続していなければならない。

※ メニュー定義ファイル中の "GLOBAL"（メニュー名）の項目名と、パラメータ定義ファイル中の "GLOBAL"（ファンクション名）のデータ部（ファンクション名）を等しくせねばならない。

3:3 パラメータ・ファイル

本プログラムが、パラメータ定義ファイルより、各機能プログラム別（グローバル・ファンクションは例外）に必要データを格納する。
機能プログラム起動時に、アーギュメントで指定するファイルである。

記述例　30､31頁のパラメータ定義ファイルより、ＡＤファンクションのパラメータ・ファイルを下記に示す。

ファイル名　：　.ad.0

```
AD          3
            file      "adfile
            start     #0
            end       #1000
```

3.4 ヒストリー・ファイル　(.wbhist)

本プログラムから起動された、機能プログラムの実行コマンド（実行イメージ・ファイル名、パラメータ・ファイル名）が記述されている。

[フォーマット]

格納されている実行コマンド数
シーケンスＮｏ．　実行イメージ・ファイル名　パラメータ・ファイル名

[Example]

```
3
1 WB_wave .global.0
2 WB_power .global.0
3 ad .ad.0
```

3.5 ストリングス・データ・ファイル　(.wbprmdat)

ストリングス入力プログラムの要求で、ユーザが入力したストリングスを格納している。

3.6 データ・ファイル・バージョン管理ファイル　(.glbfver)

グローバル・ファンクション・プログラムで作成される、データ・ファイルのバージョン番号を格納するファイルである。

3.7 スライス・ＰＩＤ・ファイル　（.wbslice）

スライス出力プログラム（WB_fftslice,WB_lpcslice)のＰＩＤを格納するファイルである。

4. 参照テーブル

本プログラムが、処理遂行の為に必要とする、情報（データ）を格納（メモリーに）するものである。
WB_limits.h（ヘッダー・ファイル）で定義している。

4.1 ＰＩＤ・テーブル

起動している、機能プログラムの情報（ファンクション名、実行ファイル名、ＰＩＤ）が格納されている。

4.2 パラメータ・ファイル・テーブル

機能プログラムの起動時に、指定されたパラメータ・ファイル名とバージョンＮｏ．が格納されている。

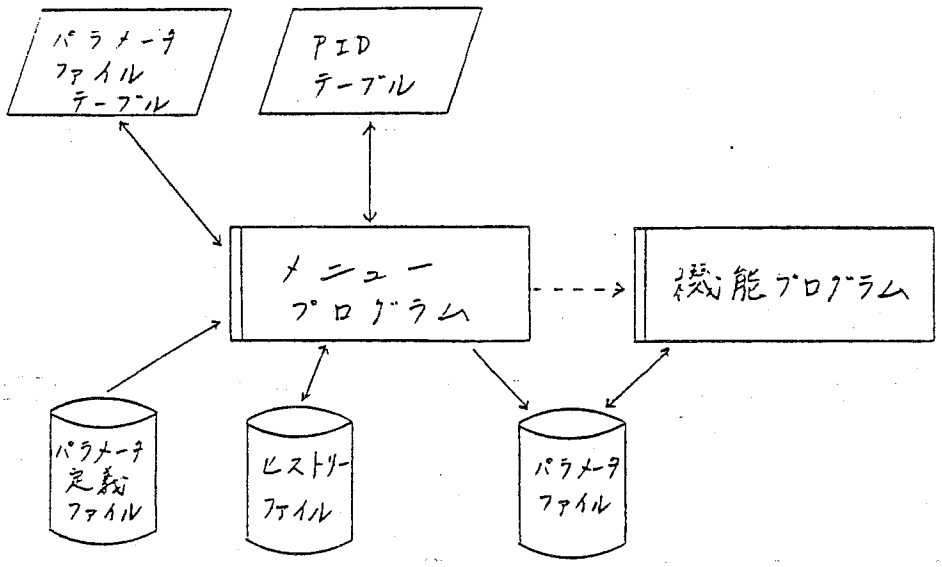
4.3 グローバル・ファンクション・テーブル

メニュー定義ファイルに、グローバル・ファンクションとして、定義されているファンクション名を格納する。

5. 機能説明

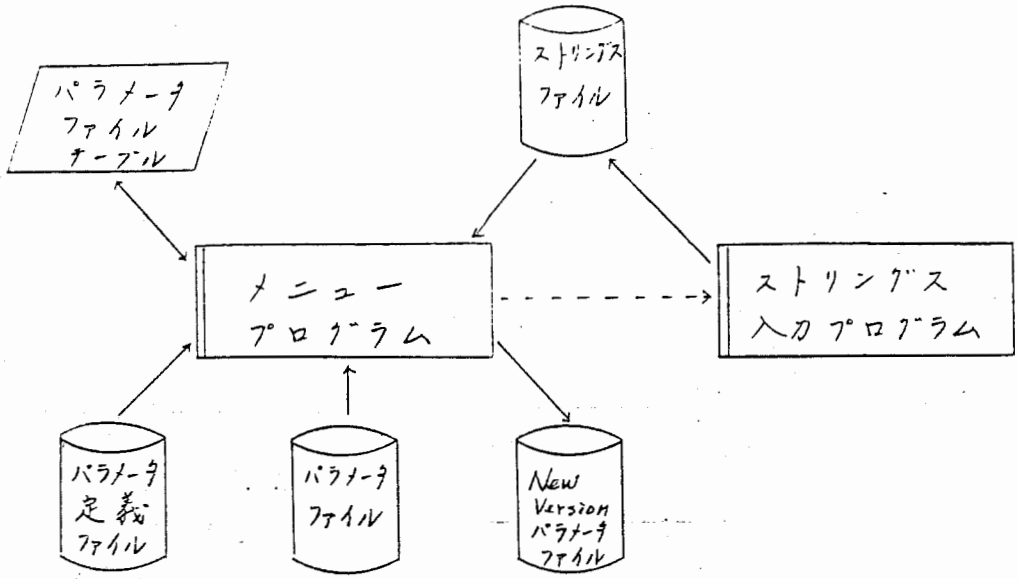
本メニュー・プログラムの全ての機能は、メニュー（プールダウン）指示標を選択することで遂行される。

5.1 機能プログラムの起動



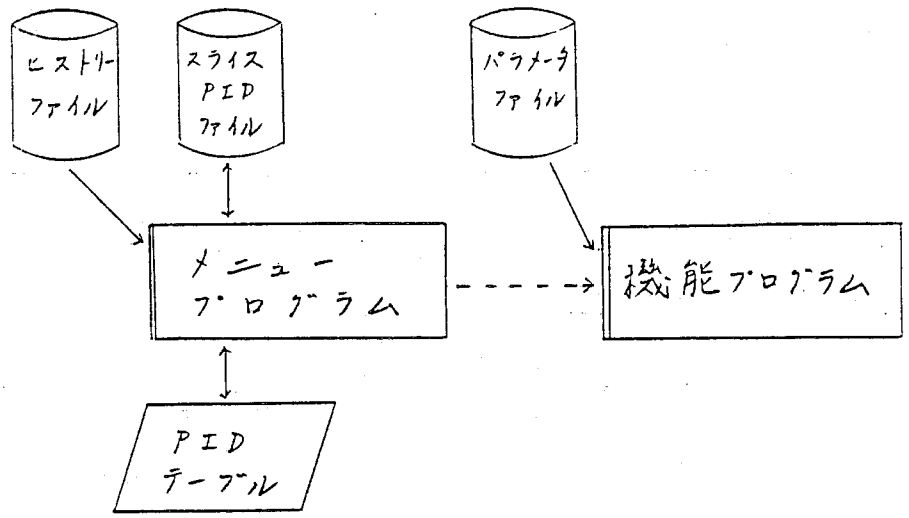
・パラメータ・ファイル・テーブルを参照し、当該のパラメータ・ファイルが無い場合は、パラメータ定義ファイルのデータで、パラメータ・ファイルを作成する。
・ＰＩＤ・テーブルを検索し、同じ機能プログラムが稼働していれば停止（kill）する。
・パラメータ・ファイルをアーギュメントとし、子プロセスとして指定機能プログラムを起動させる。

5.2 機能プログラムのパラメータ・セット



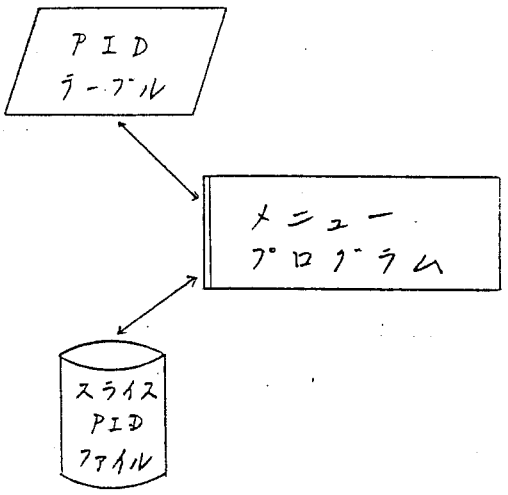
・パラメータ・ファイル・テーブルを参照し、当該のパラメータ・ファイル（パラメータ・ファイルが無い場合は、パラメータ定義ファイルを使用する）のデータをデフォルトとして、ニュー・バージョンのパラメータ・ファイルを作成する。
・ニュー・バージョンのパラメータ・ファイルに対して、ユーザ指定項目のパラメータ値の設定を行う。
・パラメータ値の入力は、ストリングス入力プログラム(WB_GetData)（本プログラムの子プロセス）で、全て文字列として処理される。
・入力された文字列は、ストリングス・ファイル(.wbprmdat)に出力される。又、ストリングス・ファイルのデータを、本メニュー・プログラムに取り込めば、ストリングス・ファイルを消去し、ストリングス入力プログラムを停止する。
・パラメータ設定終了（パラメータ設定指示標から抜ける）後 ”ＣＡＮＣＥＬ”で無ければ、パラメータ・ファイル・テーブルにニュー・バージョン・パラメータ・ファイルを登録する。

5.3　ヒストリー処理



・ヒストリー・ファイルを読み込み、機能プログラム実行の履歴（シーケンスＮｏ．、実行コマンド、パラメータ・ファイル）をプールダウン表示する。
・履歴数が、N_DOWN（WB_limits.hで定義）をオーバーすると、”NEXT”，”BACK”項目を選択して履歴の参照を行う。
・ユーザが選択（マウス・ミドル・ボタンのヒット）した、機能プログラムを子プロセスとし、”－h”、パラメータ・ファイルをアーギュメントとして起動する。
・現在稼働中の、起動プログラムは全て停止（ｋｉｌｌ）する。

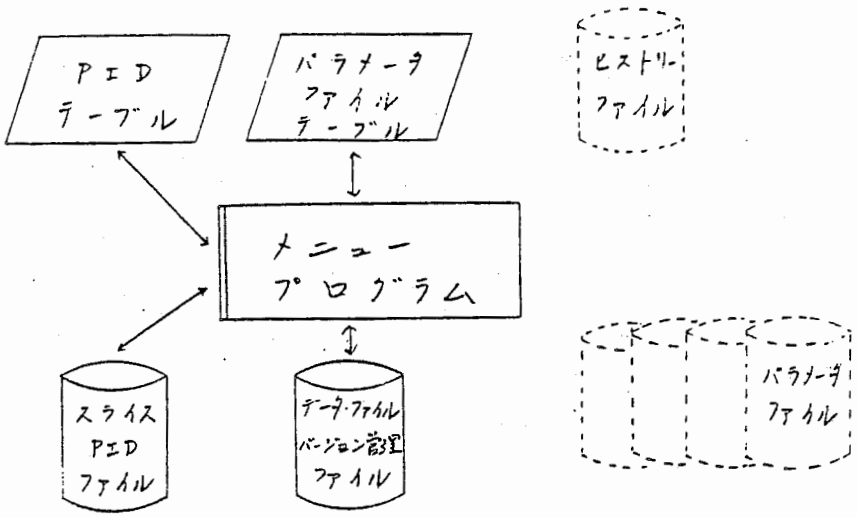
5.4　画面消去



・ＰＩＤ・テーブル、スライス・ＰＩＤ・ファイルを参照し、現在稼働中の機能プログラムを全て停止（ｋｉｌｌ）する。

※ヒストリー・ファイルは、初期化されないので、前述のヒストリー機能で復帰可能である。

5.5 リセット処理



・ＰＩＤ・テーブル、スライス・ＰＩＤ・ファイルを参照し、現在稼働中の機能プログラムを全て停止（ｋｉｌｌ）する。
・パラメータ・ファイル・テーブルより、パラメータ・ファイルを全て消去し、データ・ファイル・バージョン管理ファイルを初期化する。
（グローバルファンクションで作成された、データ・ファイルは消去されない。）
・ヒストリー・ファイルを消去する。

5.6 終了処理

・前述のリセット処理後、プログラムを終了させる。

6. 制限／注意事項

1. リミットがあるデータ（WB_limits.h で定義されている）

・メニュー定義ファイル(.wbrc)、パラメータ定義ファイル(.wbrp)中の項目ファンクション名及び、データのサイズ（文字数）が、MAXDATA より小さくなければならない。

・メニュー定義ファイルの、メイン・メニューの項目数が MAXITEM より小さくなければならない。

・メニュー定義ファイルの、サブ・メニューの項目数及び、パラメータ定義ファイルの機能プログラム／パラメータ記述部の項目数が N_DOWN より小さくなければならない。

・機能プログラム数が、 MAXFUNC より小さくなければならない。

2. パラメータ・ファイルのバージョン・アップ

パラメータ設定処理の、ストリングス入力操作まで行わないと、パラメータ・ファイルのバージョン・アップが行われない。

7. 操作説明

7.1 操作概略

基本的には、全てマウス（Mボタンのヒット）により操作を行うが、
本プログラムの起動時、パラメータ値の入力のみキーボード入力で行う。
又、１つの処理が終了すると、次ぎの処理選択（マウスの入力）を待つ。
（キーボード入力部は、▒▒▒ で表す。　<CR> はキャリッジ・リターン）

7.2　起動方法

本プログラムの起動方法は２種類ある。

(1).入力データ・ファイル名をアーギュメントとして、起動する場合

```
# WB_Menu filename <CR>
```

アーギュメントの filename は、最初に起動された機能プログラムの
パラメータ・ファイルの、ファイル項目（"file"）のデータとする。

(2).プログラム名のみで、起動する場合

```
# WB_Menu <CR>
```

7.3　メニュー・オペレーションの説明

本プログラムが起動すると、メニュー定義ファイル（２８頁の記述例を参考
にする）を読み込み、ワークステーションの画面最上部にメニュー指示標が、
下記の様に表示される。（メイン・メニュー部の項目名が表示される）
任意の指示標枠内にカーソルを移動させ、マウスをヒットする。

MENU	PARAM	HISTORY	ELSE

1. ” MENU ” オペレーション

"MENU" を選択すると、下記のプールダウン・メニューが表示される。
右側に、その枠を選んだ時の、オペレーションを示す。
何れかを選択すれば、プールダウン・メニューは消える。

MENU	ノーオペレーション
AD	ad を起動する。
DA	da を起動する。
GLOBAL	GLOBAL メニューをプールダウン表示

2. ” PARAM ” オペレーション

・PARAM を選択すると、機能プログラムのパラメータ・セット処理を行う。
・パラメータ定義ファイル（30,31頁の記述例を参考にする）を読み込み、
下記のプールダウン・メニューが表示される。
右側に、その枠を選んだ時の、オペレーションを示す。

nop	ノー・オペレーション （プールダウン表示消える）
AD	AD パラメータ項目をプールダウン表示
DA	DA パラメータ項目をプールダウン表示
GLOBAL	GLOBAL ファンクション名をプールダウン表示

ここでは、"AD" を選択したものとして、次頁に示す。

・"AD" を選択する。（AD プログラムのパラメータ設定を行う）

・下記のプールダウン・メニューが表示される。
・右側に、その枠を選んだ時の、オペレーションを示す。
・"AD", "CANCEL" 以外を選択した場合は、パラメータ設定後再度、
下記のプールダウン・メニューが表示される。

AD	ノー・オペレーション及び設定終了の時選択する（プールダウン表示消える）
file	file 項目のデータ設定を行う。
start	start 項目のデータ設定を行う。
end	end 項目のデータ設定を行う。
sampling	sample 項目名をプールダウン表示
CANCEL	今回のパラメータ設定を無効にする。

・"file" を選択する。（AD の file 項目のデータ設定を行う。）

・下記の様にデフォルト値を表示したウィンドウが表れる。

adfile

設定の際には、デフォルト値を消去して入力する。

・"sampling" を選択する。（AD の sample 項目のパラメータ設定を行う）

・下記のプールダウン・メニューが表示される。
・右側に、その枠を選んだ時の、オペレーションを示す。
何れかを選択すれば、プールダウン・メニューは消える。

sample	ノー・オペレーション
manual	マニュアル入力（前頁同様に設定する。）
12khz	AD sampling項目のデータに #12 が設定される
20khz	AD sampling項目のデータに #20 が設定される
30khz	AD sampling項目のデータに #30 が設定される

※尚、この設定が出来るのは、本メニュー・プログラム起動直後のパラメータ設定時のみである。

3. ” HISTORY ” オペレーション

・”HISTORY” を選択すると、ヒストリー処理が実行される。
・ヒストリー・ファイル（32頁の [Example] を参考にする）を読み込み、
下記のプールダウン・メニューが表示される。
右側に、その枠を選んだ時の、オペレーションを示す。
何れかを選択すれば、プールダウン・メニューは消える。

nop	ノー・オペレーション
1.WB_wave	WB_wave プログラムを起動する。
2.WB_power	WB_Power プログラムを起動する。
3.ad	ad プログラムを起動する。

※履歴の項目数が、N_DOWN より多い場合は、”NEXT” ”BACK” 指示標が、
表示される。
”NEXT” ： 現在表示されている、次ぎの履歴項目から最大２２項目
を表示する。
”BACK” ： 現在表示されている、履歴項目より２３項目前から表示
する。

4. ” ELSE ” オペレーション

・下記のプールダウン・メニューが表示される。

・右側に、その枠を選んだ時の、オペレーションを示す。

何れかを選択すれば、プールダウン・メニューは消える。

ELSE	ノー・オペレーション
ERASE	画面消去の処理を行う。
RESET	リセット処理を行う。
EXIT	終了処理を行う。

8. エラー・メッセージ

エラーが起これば、以下のメッセージを出力（本プログラムを起動させたウィンドウ）し、ベル音が２回鳴る。

(1). X（ウィンドウ・システム）で、ディスプレイのオープンが出来なかった。

” Cannot open display”

(2). ファイル・オープン・エラー。

” ファイル名 file (read/write/update) open error”

(3). 機能プログラム数が、MAXFUNC より多い。

” Out of range the number of function”

(4). 文字入力が、正しく出来なかった。

" GKS strings input error"

(5). メイン・メニューの項目数が、MAXITEM より多い。

" MainMenu::item over !"

(6). プールダウン出力の項目数が、N_DOWN より多い。

" PulldownMenu::item over !"

9. 機能プログラム追加方法

9.1 メニュー定義ファイルの記述方法

何れかのサブ・メニュー部に、追加する機能プログラムの項目名と、実行部（！実行イメージのファイル名、パラメータ・ファイル名）を記入する。

追加記入した、サブ・メニュー部の項目数を増やすことに注意。

・以下に例として、28頁の記述例の MENU （サブ・メニュー）に example （実行イメージファイル）を追加する。

```
MENU    4
        AD      !ad     .ad
        DA      !da     .da
        GLOBAL  GLOBAL
        EXAMPLE !example .example
```

9.2 パラメータ定義ファイルの記述方法

機能プログラム記述部に、追加する機能プログラムの項目名と、ファンクション名（パラメータ・ファィル名（最初に"."を付け小文字にする。）になる事に注意）を記入する。

機能プログラム記述部の項目数を増やすことに注意。

可変データをパラメータ記述部の形式に則り追加記入する。

・次頁に例として、30､31頁の記述例に example （実行イメージファイル）、可変データを入力ファイル（indata）、サンプリング周波数（12khz）として追加する。

```
FUNC    4
        AD          AD
        DA          DA
        EXAMPLE     EXAMPLE
        GLOBAL      GLOBAL
EXAMPLE 2
        file        "indata
        sampling    #12
```

9.3 パラメータ・ファイル

前述例の、パラメータ・ファイルは .example.0 と成り以下の内容を持つ。

```
EXAMPLE 2
        file        "indata
        sampling    #12
```

9.4 機能プログラム作成の注意事項

ヒストリーから、機能プログラムを起動する時は、"-h" とパラメータ・ファイル名がアーギュメントになる。