PcWaveForm アプリケーションノート

PcWaveForm スクリプト(Archi_1)簡易マニュアル

概要

本アプリケーションノートは BASIC、C 言語などの他言語の経験者向けの簡易マニュアルです。下記内容を記述しています。

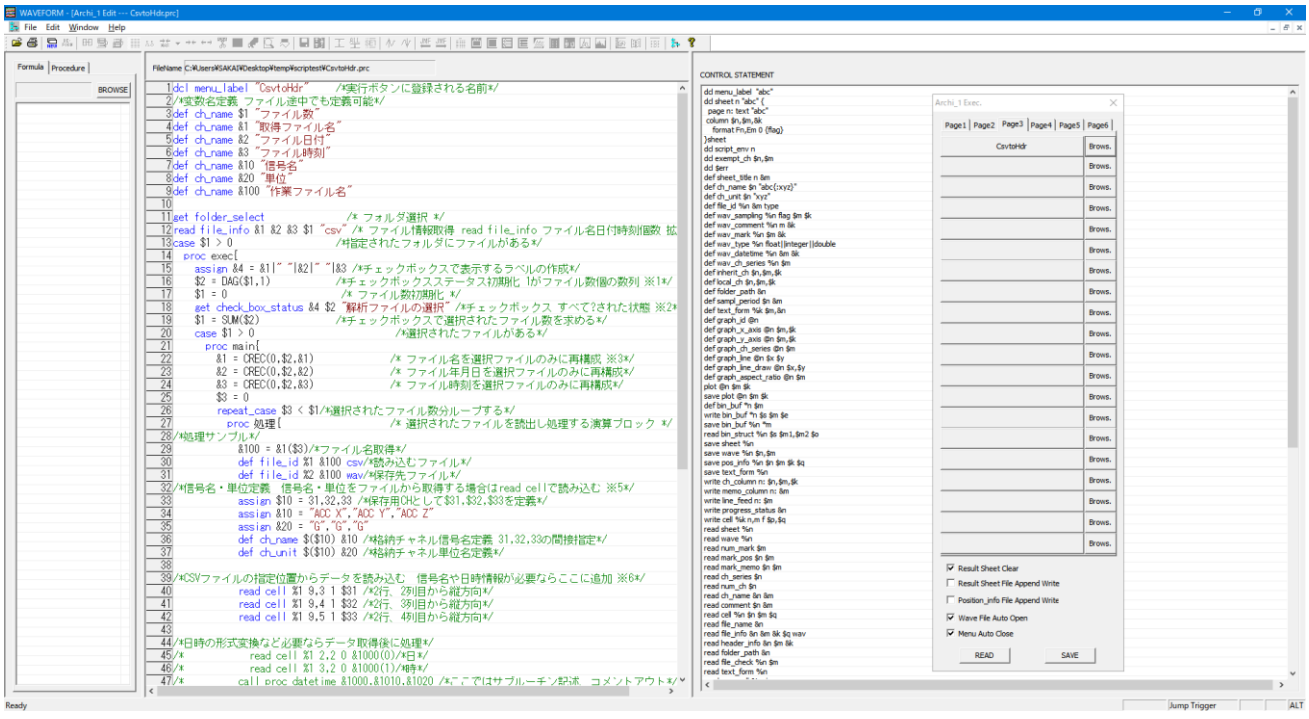
①スクリプトファイルの形式と実行方法
②文法について
　基本的な変数取り扱いと、c 言語などと扱いが異なる注意点についてまとめてあります。
③ファイルからデータを読み込む・hdr/dat で保存する。
　CSV ファイルからデータを取得する方法を例として記述しています。
　スクリプトで扱ったデータを hdr/dat ファイルとして保存する方法について記述しています。
④スクリプト例
　1.CSV ファイルを hdr/dat ファイルで保存する
　　テキストファイルからのデータ取得例です。
　フォルダ選択、ファイル選択、選択ファイル毎の演算・結果保存のループ処理を例示しています。
　2.hdr/dat ファイルの情報取得・条件選択を行う
　　hdr/dat ファイルからの情報取得例です。
　　CH 情報取得、ドロップダウンリスト、結果リスト表示を例示しています。

記述された文法・構文詳細については、「PL-U4101C1 PcWaveForm Archi_1 Script 言語仕様編」を、関数については「PL-U4101C1_PcWaveForm_取扱説明書_演算関数仕様編」を参照ください。

①スクリプトのファイル形式と実行方法

拡張子 prc のテキストファイルです。記述順に 1 行ずつ実行されます（インタプリンタ）。
PcWaveForm のエディタ機能または任意のテキストエディタで作成します。エディタ機能は Option メニューの Archi_1 Editor、または登録したスクリプト（後述）のラベル上で右クリックして表示されるメニューから Edit を選択して起動します。

エディタ画面



実行手順

手順 1 スクリプトの作成

```
dcl menu_label "hello"
disp message "hello,world"
```

作成後、任意のファイル名.prc で保存します。

例文内容

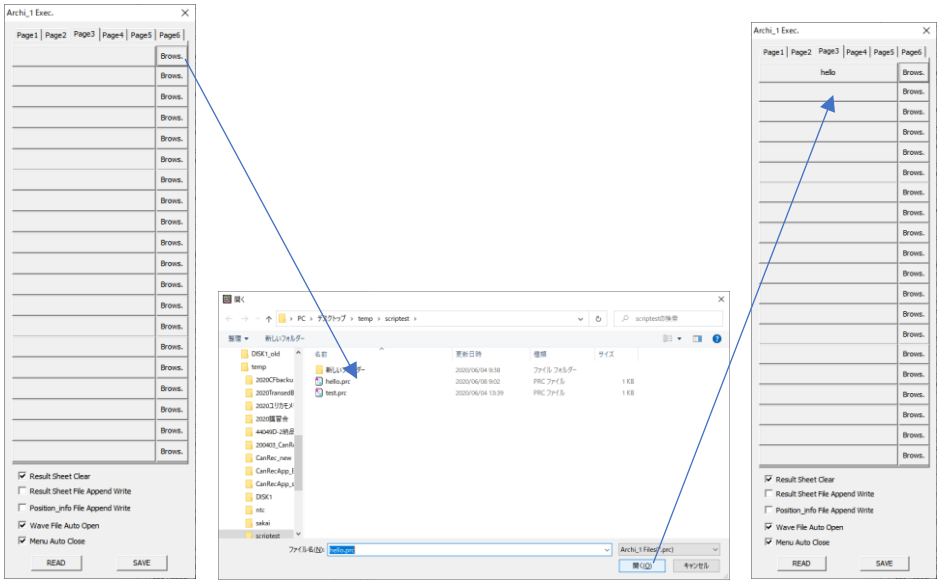
1 行目 dcl menu_label は、スクリプト登録時の名前を指定する構文です。

2 行目 disp message は、文字列や変数内容をダイアログに表示する構文です。

手順 2 スクリプトを登録

Option メニューの Archi_1 Execute または下図のアイコンを選択します。

メニューが表示されるので Brows.ボタンを選択→ファイル選択→登録となります。



誤って登録した場合、登録したラベル上で右クリックして表示されるメニューで Delete を選択します。

手順 3 スクリプトの実行

登録したボタンを左クリックでスクリプトが実行されます。

②文法について

変数の扱い

#xx　　：先頭に#が付いた変数は、解析対象ファイルの CHxx のデータ列（数列）です。
　表示波形を範囲選択した状態でスクリプトを実行した場合
　　選択範囲のデータが#xx に格納されます。
　hdr/dat ファイルを読み込んだ場合
　　ファイル全体が#xx に格納されます。
$xx　　：先頭に$が付いた変数は、数列です。
&xx　　：先頭に&が付いた変数は、文字列です。各要素の内容は""で囲んで扱います。
数値は double 精度で扱われます。
変数の各要素は$xx(n)で指定します。要素の index は 0 から始まります。
$xx と記述した場合は、数列全体を扱います。
xx 部分は直値での記述のほか、間接指定や連続指定が可能です。

間接指定

構文、関数によっては、#X,$X,&X は$($x),&($x)と間接指定記述が可能です。
例）$1 = 10 の時、 #($1)と記述すると#10 を意味します。

連続指定

$[開始数値,個数]と記述すると、変数の連続指定が可能です。
例）$[100,10]と記述すると$100,$101,・・・,$109 を連続指定が可能です。

間接指定、連続指定の実例を本書末尾のスクリプト例に記述しています。

変数名・単位定義

変数は名前、単位指定が可能です。hdr/dat に保存する場合、変数に定義した名前、単位が信号名、単位として扱われます。
記述例）

```
def ch_name &1 “汎用文字列”
def ch_name $1 “変数名”
def ch_unit $1 “変数単位”
assign $1 “変数名:単位” = 0,1,2,
```

コメント

/*と*/に囲まれた範囲がコメントです。//は使えません
例）

```
/*これはコメント*/

//これはエラー

/*$1 = $2+$3/*行頭に/*があるので全体がコメント*/

/*
複数行にまたがってもコメントとして扱う
*/
```

演算・代入

代入演算子 = の左辺に結果格納 CH、右辺に代入値・計算式を記述します
記述時の注意
①左辺 = 右辺の=前後に半角スペースを記述します。
②右辺の式記述において、四則演算記号の前後はスペースなしです。

③演算は数列全体を一気に計算します
④代入演算子は=のみです。加算代入演算子 += 等はありません。
例）四則演算
$1 = #1+#2
上記の計算式で
#1 = 0,1,2,3
#2 = 1,2,3,4
だった場合
$1 = 1,3,5,7 となります。ループ文を使って各要素を 1 つずつ計算する必要はありません。

例）関数を使用 関数の詳細は
$10 = LEN($1) /*$1 の要素数を返す*/
$1 = 1,3,5,7 だった場合、$10 は 4 となります。

初期値を与える場合など、計算式を伴わない場合、assign 文で定義します。
例）
assign $1 ”chname:unit” = 100<0>　/*データ数 100 の 0 データを作成*、CH 名:単位の指定も可能/
assign $10 = 1,2,3
assign $($10) = 0,1,2,3/*$10=1,2,3 なので$1,$2,$3 の中身が 0,1,2,3 となります*/
assign &1 = “aaa”,”bbb”,”ccc”
assign &1(1) = “ddd”

条件分岐文

case 文を使用します。if 文に相当します。else はありません。
case に続いて 条件式 を記述し、成立したときのみ処理する内容を proc XXX{ ～ }XXX で囲まれた範囲に記述します。proc は固定、XXX は任意に指定可能ですが、始まりを示す{と終了を示す}の後に必ず記述が必要です。

記述例
case $1(0) < 6
proc abc{
/* ここに処理を記述 */
}abc

繰り返し文

repeat_case 文を使用します while 文に相当します。
case に続いて 条件式 を記述し、成立したときのみ処理する内容を proc XXX{ ～ }XXX で囲まれた範囲に記述します。proc は固定、XXX は任意に指定可能ですが、始まりを示す{と終了を示す}の後に必ず記述が必要です。

記述例
$1 = 0
repeat_case $1 < 6
proc loop{
　/*ここに処理を記述*/
　$1 = $1+1/*ループ用変数をインクリメント*/
　}loop

外部処理ブロック（サブルーチン、関数）

メイン処理関数/サブルーチンに相当します。処理ブロックはメイン部終了を示す end 以下に記述します。
処理ブロック内のローカル変数、メイン処理からの引継ぎ変数の指定が可能です。

呼び出し文法： call proc 演算処理ブロック名 引数チャネル列

```
  call proc date &1000,&1010,&1020
```
処理ブロック文法： proc 処理ブロック名{ }処理ブロック名

例）日時の表記を hdr ファイルに必要な形式に直す。
```
proc datetime{
  /*-------------------------------------------------------------
  datetime 引数内容
  &1        : date_data in（YYYY/MM/DD,hh:mm:ss）
  &100      : date_data out（MM-DD-YYYY）
  &101      : time_data out（hh:mm:ss）
  -------------------------------------------------------------*/
  def inherit_ch &1,&100,&101   /*呼び出し元から引き継ぐ変数・戻り値*/
  def local_ch &[200,3]         /*ローカル変数、メイン部に影響しない*/
  /*日付並べ替え*/
  &200 = CEXT(1,4,&1(0))
  &201 = CEXT(6,2,&1(0))
  &202 = CEXT(9,2,&1(0))
  assign &100 = &201|"-"|&202|"-"|&200 /* MM-DD-YYYY */
  /*時刻並べ替え*/
  assign &101 = &1(1)   /*時刻は変更必要なし*/
}datetime
```

inherit_ch で指定しした&1,&100 の内容は、呼び出し元の変数を引き継ぎ、また値を変更した場合は呼び出し元の値も変わります。
local_ch で指定した&200~202 はデータを変更しても呼び出し元に影響がありません。

③ファイルからデータを読み込む・hdr/dat で保存する。

取り扱うファイルを定義する

ファイルを扱う場合、Script で使用する読み出しファイルまたは書き込みファイルのファイル番号を定義します。

文法： def file_id %ファイル番号 ファイル名 取扱属性

例）スクリプトファイルと同じフォルダに存在する a.csv を扱う。
```
&100 = "a"
def file_id %1 &100 csv /*&100 で指定された名前の csv ファイルを対象とする*/
def file_id %2 &100 wav /*&100 で指定された名前の hdr/dat ファイルを対象とする*/
```
取扱属性で wav を指定した場合、hdr/dat がペアで扱われます。個別に指定する必要はありません。

指定したテキストファイルからデータを読み込む

文法：

read cell %ファイル番号 読み出し開始行, 列番号 方向フラグ 格納先チャネル列 セル数 読み出し先フラグ

ファイル番号　：あらかじめ def file_id 文で指定した番号です。
読み出し開始行, 列番号　：セルの左上を 1,1 として指定します。
方向フラグ　：データの読み込み方向を指定します。0 横方向、1 縦方向にデータを取得します。
格納先チャネル列　：格納先の変数を指定します。”,”で区切りで複数の変数を指定可能です。連続指定、間接指定も可能です。
格納先変数の index を指定した場合、指定セルの 1 データを指定 index に格納します。
格納先変数を複数個指定した場合、変数ごとに方向フラグで指定していない方向へ 1 セル分位置を開始位置として、データを格納します。
セル数　：読み込むセル数を指定します。省略した場合、存在するセルを最後まで読み込みます。
読み出し先フラグ　：読み込み対象がファイルの場合 0、テキストフォーム※の場合 1 をしてします。
セル数、読み出し先フラグは省略可

※テキストフォームについては別途取扱説明書を参照してください。
記述例）
read cell %1 2,2 1 &100(0) /*格納先チャネルの index を明示すると 1 データだけ取得*/
read cell %1 2,2 0 &10 /*&10 に 2,2 を先頭に横方向のデータを取得*/
read cell %1 3,2 1 $31,$32 /*$31 に 3,2 を先頭に縦方向のデータを取得、$32 に 4,2 を先頭に縦方向のデータを取得*/

CSV ファイルと読み込み記述例

FileName	3to7.hdr							
Date	2005/10/22							
Time	18:06:01							
sampling	1000							
ch num	4							
		Ch1		Ch2		Ch3	Ch4	
		X Axis Acc		Y Axis Acc		Z Axis Acc	Rotational Speed	
Index	sec	G		G		G	rpm	
0	0	$31	-1.206	$32	-0.08	$33 0.769	$34 3002	
1	0		-1.149		-0.017	0.819	2986	
2	0		-1.154		0.146	0.759	2971	
3	0.001		-1.169		0.23	0.569	2975	
4	0.001		-1.245		0.179	0.628	2986	
5	0.001		-1.284		0.226	0.69	3004	
6	0.001		-1.271		0.229	0.684	2995	
7	0.001		-1.26		0.241	0.813	2986	
8	0.002		-1.212		0.44	0.824	2978	
9	0.002		-1.167		0.55	0.86	2975	
10	0.002		-1.148		0.496	0.958	2971	
11	0.002		-1.113		0.41	0.966	2962	
12	0.002		-1.058		0.292	1.059	2953	

read cell %1 2,2 0 &100(0) /*年月日2行2列目から1データ読み出し*/
read cell %1 3,2 0 &100(1) /*時刻3行2列目から1データ読み出し*/
read cell %1 4,2 0 $1000(0) /*サンプリングレート*/
read cell %1 5,2 0 $1001(0) /*CH数*/
read cell %1 7,3 0 &110 /*CH名 */
read cell %1 8,3 0 &120 /*単位*/
/*CH1～4データ 9行3列目から縦方向に読み出し*/
read cell %1 9,3 1 $31,$32,$33,$34

hdr/dat ファイルを保存する

サンプリングレート（省略不可）、データ形式（省略時は integer）、ファイル日時（省略時はファイル作成日時）を設定し、保存します。
あらかじめ def File_id で取り扱いファイルを定義する必要があります。

```
def wav_sampling %2 0 100 "sec" 0    /*フラグ（周期 1/周波数 0） サンプリング値 単位 X_OFFSET*/
def wav_type %2 float                /*integer/float/double*/
def wav_datetime %2 &1010 &1020   /*MM-DD-YYYY hh:mm:ss*/
save wave %2 $31,$32
```

④スクリプト例

1.CSV ファイルを hdr/dat ファイルで保存する

フォルダ指定、ファイル選択を行い、チェックボックスで選択した csv を hdr/dat ファイル保存します。
CSV のフォーマットは③の記述例を想定しています。

```
dcl menu_label "CsvtoHdr"       /*実行ボタンに登録される名前*/
/*変数名定義 ファイル途中でも定義可能*/
def ch_name $1 ”ファイル数”
def ch_name &1 ”取得ファイル名”
def ch_name &2 ”ファイル日付”
def ch_name &3 ”ファイル時刻”
def ch_name &10 ”信号名”
def ch_name &20 ”単位”
def ch_name &100 ”作業ファイル名”

get folder_select                /* フォルダ選択 */
read file_info &1 &2 &3 $1 "csv" /* ファイル情報取得 read file_info ファイル名日付時刻個数 拡張子*/
```

```
case $1 > 0                          /*指定されたフォルダにファイルがある*/
  proc exec{
    assign &4 = &1|" "|&2|" "|&3 /*チェックボックスで表示するラベルの作成*/
    $2 = DAG($1,1)                /*チェックボックスステータス初期化 1 がファイル数個の数列 ※1*/
    $1 = 0                         /*  ファイル数初期化 */
    get check_box_status &4 $2 "解析ファイルの選択" /*チェックボックス すべて☑された状態 ※2*/
    $1 = SUM($2)                  /*チェックボックスで選択されたファイル数を求める*/
    case $1 > 0                                  /*選択されたファイルがある*/
      proc main{
        &1 = CREC(0,$2,&1)                /*  ファイル名を選択ファイルのみに再構成 ※3*/
        &2 = CREC(0,$2,&2)                /*  ファイル年月日を選択ファイルのみに再構成*/
        &3 = CREC(0,$2,&3)                /*  ファイル時刻を選択ファイルのみに再構成*/
        $3 = 0
        repeat_case $3 < $1/*選択されたファイル数分ループする*/
          proc 処理{                      /*  選択されたファイルを読出し処理する演算ブロック */
/*処理サンプル*/
            &100 = &1($3)/*ファイル名取得*/
            def file_id %1 &100 csv/*読み込むファイル*/
            def file_id %2 &100 wav/*保存先ファイル*/
/*信号名・単位定義　信号名・単位をファイルから取得する場合は read cell で読み込む ※5*/
            assign $10 = 31,32,33 /*保存用 CH として$31,$32,$33 を定義*/
            assign &10 = "ACC X","ACC Y","ACC Z"
            assign &20 = "G","G","G"
            def ch_name $($10) &10 /*格納チャネル信号名定義 31,32,33 の間接指定*/
            def ch_unit $($10) &20 /*格納チャネル単位名定義*/

/*CSV ファイルの指定位置からデータを読み込む　※6*/
            read cell %1 9,3 1 $31 /*2 行、2 列目から縦方向*/
            read cell %1 9,4 1 $32 /*2 行、3 列目から縦方向*/
            read cell %1 9,5 1 $33 /*2 行、4 列目から縦方向*/

/*日時の形式変換など必要ならデータ取得後に処理*/
/*            read cell %1 2,2 0 &1000(0)/*日*/
/*            read cell %1 3,2 0 &1000(1)/*時*/
/*          call proc datetime &1000,&1010,&1020 /*日付形式変更サブルーチン、コメントアウト*/

/*データ取得後の処理を記述*/
/*処理例 ローパスフィルタ ここではコメントアウトしています*/
/* def sampl_period 0.001 "sec"       /*サンプリング定義 ※4*/
/* $31 = LPR(100,LPF(100,$31))  /*100HzLPF*/
/*処理例 ここまで*/

/*保存する hdr/dat の情報作成　本例ではサンプリングレートとデータタイプのみ指定*/
            def wav_sampling %2 0 1000 "sec" 0
            def wav_type %2 float
/*            def wav_datetime %2 &1010 &1020 /*日付時間を指定する場合記述*/
/*保存*/
            save wave %2 $[31,3]/*$31,$32,$33 の連続指定*/
            $3 = $3+1/*ファイルインデックスをインクリメントして次のループへ*/
          }処理
      }main
```

```
  }exec
end
/*  サブルーチンは end 以降に記述/**/
/*proc datetime{
/*変換処理  サンプルの形式なら外部処理ブロック例を記述で OK*/
/*}datetime
/**/
```

※処理内容が分かりにくい部分および記述例補足
例）フォルダにファイルが a.csv, b.csv, c.csv と 3 つあった場合。
※1 DAG でデータ数がファイル個数(この例では 3)の 1 の数列を作成、$2=1,1,1 となる

※2 チェックボックスで a,c を選んだ場合、$2=1,0,1 となる

※3 ファイル名は&1 に a,b,c が格納されている。CREC で文字列配列のうち、数列$2=1,0,1 の 1 のデータ番号のみの文字列配列に再構成。&1 は a,c となる。

※4 CSV ファイルからデータを取得してもサンプリングレートが分かりません。サンプリングレートが分からないとフィルタ処理はできないため、定義が必要です。hdr/dat を読み込んだ場合は自動で取得しますので定義は不要です。

※5 変換する CSV ファイルの信号名・単位が固定の場合、本例のように固定値記述で問題ありません。単位が任意に変更される可能性がある場合、下記のようにファイルから読み込みます。

```
read cell %1 7,3 0 &10 /*CH 名読み込み*/
```

※6 本例では 1CH ずつ記述していますが、下記のような記述も可能です。

```
$100 = 2    /*行*/
$101 = 2    /*列*/
$102 = 3    /*CH 数*/
read cell %1 $100,$101 1 $[31,$102] /*$31 から CH 数分読み込み、ここでは 3ch なので$31,$32,$33*/
```

2.hdr/dat ファイルの情報取得・表示

ドロップダウンリストで最大値/最小値の位置を検索したい CH を選択し、選択した CH の最大/最小値の位置（data index）を検索、その位置でのすべての CH の値を表示します

```
dcl menu_label "最大最少位置データ確認"
dcl sheet 1 { /*  結果シート宣言の開始とページ数設定   ※1*/
   page 1: "収録ファイル" /*  ページ宣言の開始とページ表題の設定*/
      column &4,&5,&1,$4,$8,&9,$5,&6,&7,$60,$61,$10,$62,$12/*,$12,$13 /*書き込みチャネル宣言*/
      format A,A,A,F0,F2,A,F0,A,A,F0,F0,F3,F0,F3 2 /* Format  宣言*/
}sheet /*  結果シート宣言の終了*/

def ch_name &1 "file_name"
def ch_name &2 "生成年月日"
def ch_name &3 "生成時刻"
def ch_name &4 "収録開始年月日"
def ch_name &5 "収録開始時刻"
def ch_name &6 "信号名"
def ch_name &7 "単位"
def ch_name &8 "folder_name"
def ch_name &9 "収録データ数"
def ch_name &11 "条件"
```

```
def ch_name &20 "Temp1"
def ch_name $1 "ファイル数"
def ch_name $2 "check_status"
def ch_name $4 "収録チャネル数"
def ch_name $5 "収録チャネル番号"
def ch_name $6 "file_counter"
def ch_name $8 "サンプリング周波数:Hz"
def ch_name $9 "temp"
def ch_name $60 "検出対象 CH No."
def ch_name $61 "最大値検出 Index"
def ch_name $62 "最小値検出 Index"
def ch_name $100 "ch_counter"

$1 = 0 /* ファイル数初期化 */
assign &11 = "最小値","最大値"

get folder_select /* フォルダ選択 */
read file_info &1 &2 &3 $1 .hdr /* ファイル情報取得 */
assign $20 "chsel:" = 0 /*CH 選択フラグ 最初のファイルのみ or ファイル毎*/
case $1 = 0
  proc end{
    disp message "対象のファイルがありません。"
  }end
case $1 > 0
  proc main{
    read folder_path &8 /* カレントフォルダ名取得 */
    def sheet_title 0: &8 /* 結果シート・タイトル定義 */
    assign &20 = &1|" "|&2|" "|&3
    $2 = DAG($1,1) /* check box status 初期化 */
    $1 = 0 /* ファイル数初期化 */
    get check_box_status &20 $2 "解析ファイルの選択"
    $1 = SUM($2)

    case $1 > 0
      proc main2{
        &1 = CREC(0,$2,&1) /* ファイル名を選択ファイルのみに再構成*/
        &2 = CREC(0,$2,&2) /* ファイル年月日を選択ファイルのみに再構成*/
        &4 = CREC(0,$2,&3) /* ファイル時刻を選択ファイルのみに再構成*/
        $6 = 0 /* file counter 初期化*/

        repeat_case $6 < $1
          proc file_list {
            def file_id %1 &1($6) wav
            read wave %1 /*hdr/dat ファイルからデータを読み込む*/
            &4 = CSDT() /* 収録開始年月日取得 */
            &5 = CSTM() /* 収録開始時刻取得 */

            write ch_column 1: &1($6),&4,&5 /* ファイル名、開始年月日、時刻書き込み*/
            $4 = NCH()      /* 収録チャネル数取得*/
            $5 = CHS()      /* 収録チャネル番号取得*/
            &6 = CHNM(0) /* 収録チャネル信号名取得*/
```

```
        &7 = CUNT(0) /*  収録チャネル単位取得*/
        case $20 = 0 /*CH 選択の管理フラグを確認*/
          proc setting{
            assign &66 "チャネル選択:" = "CH"|$5(F0)|" "|&6
            assign &10 "チャネル選択:" = &66(0)     /*ドロップダウンリストの初期値設定*/
            /*      assign &12 "解析対象:" = &11(0) /*最大値 or 最小値を検索対象*/
            get value &66:&10 /*チャネル選択&閾値設定  ※2 */
            $20 = 1                /*処理済みフラグを立てる。立てない場合はファイル毎に選択*/
            &60 = CSEP(3,&10) /*文字列のスペースを区切り文字として文字列配列にする*/
            $60 = CTN(&60(0)) /*文字列配列の CH 番号を文字列→数値に変換*/
          }setting

        /*case &12 = "最大値" /*最大値のみ検索したい場合*/
        proc max{
          def ch_name $10 "最大 Index の値"
          $61 = MXP(#($60))
        }max
        /*case &12 = "最小値" /*最小値のみ検索したい場合*/
        proc min{
          def ch_name $12 "最小 Index の値"
          $62 = MNP(#($60))
        }min

        $21 = 0
        $10 = DAG($4,0)
        $12 = DAG($4,0)
        repeat_case $21 < $4   /*CH 数分ループ*/
          proc getval{
            $11 = $5($21)
            $10($21) = PTV($61,#($11))
            $12($21) = PTV($62,#($11))
            $21 = $21+1
          }getval

         /*  チャネル数、番号、信号名、単位など書き込み*/
          write ch_column 1: $4,$5,&6,&7,$60,$61,$10,$62,$12

        $8 = 1/PRD() /*  サンプリング周波数取得  */
        read header_info &9 $9 "NUM_SAMPS" /*  収録データ数取得  */
        write ch_column 1: $8,&9 /*  データ数、サンプリング周波数書き込み  */
        write line_feed 1: 1 /*  書き込み行揃え  */
        $6 = $6+1 /* file counter UP */
      }file_list
    }main2
  }main
end
```

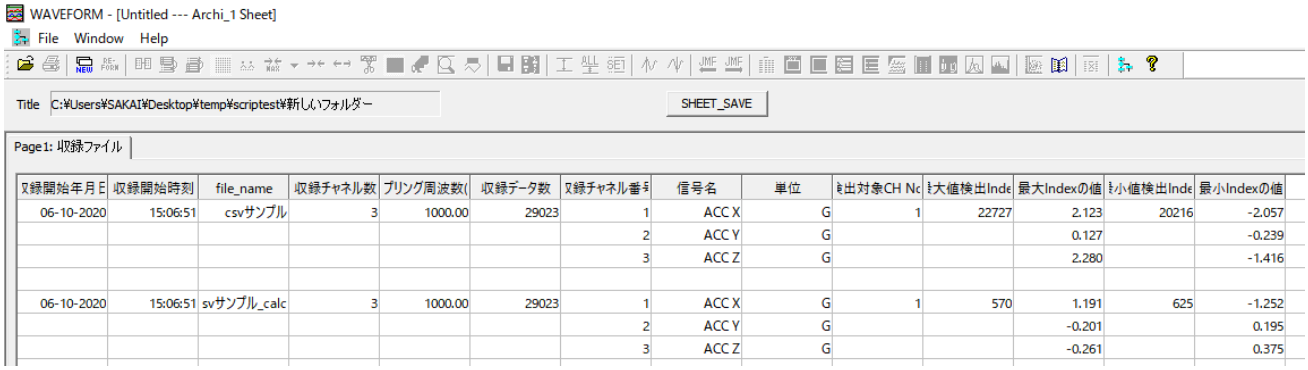
※処理内容が分かりにくい部分および記述例補足

※1

結果リスト表示の設定です。結果表示例を示します。

dcl sheet で宣言した変数を、write ch_column でシートに書き込みます。宣言した変数を一度の write ch_column で書き込む必要はな

く、本例のように変数ごとに任意のタイミングで書き込み可能です。

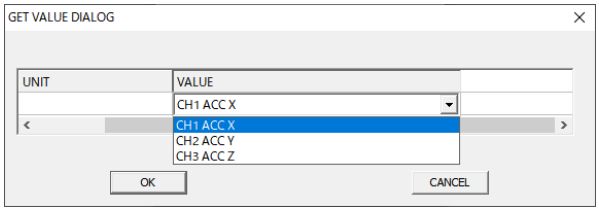


収録開始年月日	収録開始時刻	file_name	収録チャネル数	プリング周波数(	収録データ数	収録チャネル番号	信号名	単位	出対象CH No	大値検出Inde	最大Indexの値	小値検出Inde	最小Indexの値
06-10-2020	15:06:51	csvサンプル	3	1000.00	29023	1	ACC X	G	1	22727	2.123	20216	-2.057
						2	ACC Y	G			0.127		-0.239
						3	ACC Z	G			2.280		-1.416
06-10-2020	15:06:51	svサンプル_calc	3	1000.00	29023	1	ACC X	G	1	570	1.191	625	-1.252
						2	ACC Y	G			-0.201		0.195
						3	ACC Z	G			-0.261		0.375

※2

&66 の文字列配列をドロップダウンリストとして表示、選択された文字列を&10 に格納します。

最大値/最小値を選択する場合、&11:&12 を追加。



改定履歴

2020/7/16	Ver.1.02	例文の説明を追加
2020/7/02	Ver.1.01	改ページ修正
2020/6/18	Ver.1.00	初版