

PCW(H), PCWHD Version 4 メニュー説明

ファイル・メニュー



New[新規]



新規ファイルの作成

新規プロジェクトを作成します。プロジェクトは手動又は、ウィザードを使って作成することが出来ます。手動で作成した場合は .PJT ファイルのみが作成されます。他の C メイン・ファイルは指定しなければいけません。ウィザードではユーザーがプロジェクトのパラメータを指定し、.C、.HT と .PJT ファイルが作成されます。



Source File[ソース・ファイル]

新しいソース・ファイル(.c 又は、.h ファイル)を作成



Project Wizard[プロジェクト・ウィザード]

ウィザードはユーザーがプロジェクトのパラメータを指定することで完全な.C、.H と.PJT ファイルが作成されます。



Project Manual[プロジェクト・ウィザード]

新しいプロジェクトを手動で作成します。基本プロジェクト情報を持つために .PJT ファイルのみが作成されます。存在する .C メイン・ファイルは指定されるか、又は、空の 1 つが作成されます。



RTF File[RTF ファイル]

新しい RTF ファイルを作るための RTF エディター・プログラムが開かれます。



Flow Chart[フローチャート]

新しいフローチャートを作るために早くて簡単なチャートのための PCW に含まれているフローチャート・エディター・プログラムが開かれます。



Open[開く]

エディターにファイルをオープンします。他のファイルがオープンされていないときは、このオープンされたファイルに対してプロジェクト名が設定されます。Ctrl-O がショートカットです。



Any File[全てのファイル]

全てのタイプのファイルがエディターに開かれます。



Source File[ソース・ファイル]

ソース・ファイル(.c 又は、.h)をエディターに開きます。



Project[プロジェクト]

プロジェクト・ファイル(.PJT)をエディタに開きます。



Output File[出力ファイル]

出力ファイル(.hex, .err, .lst, .sym, .cod, .cof, .tre, .sta)をエディターに開きます。 .

RTF File[RTF ファイル]



RTF ファイルをを RTF エディター・プログラムに開きます。

Flow Chart[フローチャート]



フローチャート・ファイルをフローチャート・エディター・プログラムに開きます。

As hex file[ヘキサ・ファイルとして]



ヘキサ・エディターにファイルを開く

As text file[テキスト・ファイルとして]



テキスト・ファイルとしてファイルを開く

Open All Files[すべてのファイルを開く]



最後にコンパイルされたプロジェクト、又は、現在開かれているプロジェクト全てのファイルがオープンされます。このためには全てのインクルード・ファイルがわかるようにプログラムがコンパイルされていないといけないです。

Close[閉じる]



編集のために開かれているファイルを閉じます。PCW では1つのファイルがオープンされている場合、他のプログラムを編集のために開くことは出来ません。

Close All[全てのファイルを閉じる]



エディタに現在開かれている全てのファイルを閉じます。

Save[保存]



編集のために現在選択されているファイルが保存されます。

Save As[名前を付けて保存]



現在選択されているファイルを保存するためにファイル名を付けて保存します。

Save All[すべてを保存]



すべてのオープンされたファイルが保存されます。

Exit[終了]



IDE(PCW)を終了

Encrypt[暗号化]



暗号化されたインクルード・ファイル (拡張子.encrypted) を作成します。

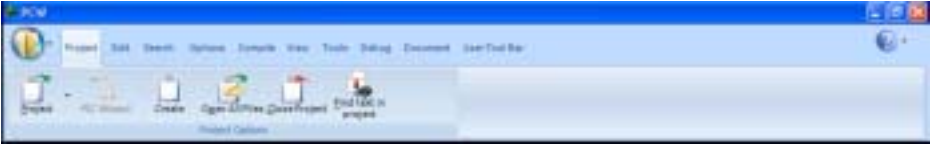
標準のコンパイラの#include コマンドにこの拡張子.encrypted を持つファイルを指定することができます。暗号化されたファイルは、読み込時に復号されます。これにより、ソースコードを公開しないでインクルード・ファイルを配布することが出来ます。


Print[印刷]



現在選択されているファイルを印刷

プロジェクト・メニュー



Project[プロジェクト]  指定された .PJT ファイルとメイン・ソースがロードされます。矢印上でクリックしますと最近使用されたファイルをリストにし、そのファイルを選択してオープンできます。

PIC Wizard[PIC ウィザード]

このコマンドはブランク形式になっており、RS232 I/O と I2C の特性、タイマー・オプション、使用するインターラプト(割り込み)、A/D オプション、必要なドライバーやピン名等をフォームに指定することで新しいプロジェクトを作成することが出来ます。

ドライバーが選択された時、ツールが必要なピンを選択し、ピンが結合されます。ユーザは最終的なピン選択を編集することが出来ます。全ての選択行われますと、#define, #includes とプロジェクトに必要な初期コマンドで初期.C, .H と.PJT ファイルが作成されます。これは新しいプロジェクトを始める早い方法です。1度ファイルが作成されますと、メニューに戻ってさらに変更することは出来ません。

Manual Create[手動作成]

新しいプロジェクトを手動で作成します。 .pjt ファイルのみが基本的なプロジェクト情報を保持するために作成されます。存在する .C メイン・ファイルは指定されるか、又は、空の1つが作成されます。手動ユティリティは下記のオプションを持っています。

ターゲット：ドロップ・ダウン・メニューからターゲットを選択

ファイル：プロジェクトに追加したいファイルを選択することが出来ます。"+Add"上でクリックすることで、新しいソース・ファイルを追加することが出来ます。ファイルの削除はそのファイルを選択し、"-Remove"をクリックすることで削除することが出来ます。

インクルード：プロジェクトのインクルードされたディレクトリーを選択することが出来ます。

デフォルトは：

C:%Program Files%PICC%Devices

C:%Program Files%PICC%Drivers

出力ファイル：

希望するボックスの上にチェックをすることでコンパイラによる出力ファイルを指定することが出来ます。例えば、リスト・ファイルでオペコードを見たいときは、"With opcodes"ラジオ・ボタンをクリックすることが出来ます、そして、"Apply"[適応]をクリックして下さい。デフォルトは CCS ベーシックです。

Creat[作成] ソース・ファイル、インクルード・ファイル、グローバル定義と指定出力ファイルを追加/削除することが出来る新しいプロジェクトを作成

Open All File[すべてのファイルを開く]

指定された .PJT ファイルとプロジェクトで使用される全てのファイルがオープンされます。このためには全てのインクルード・ファイルがわかるようにプログラムがコンパイルされていないわけにはいけません。

Close Project[プロジェクトを閉じる]



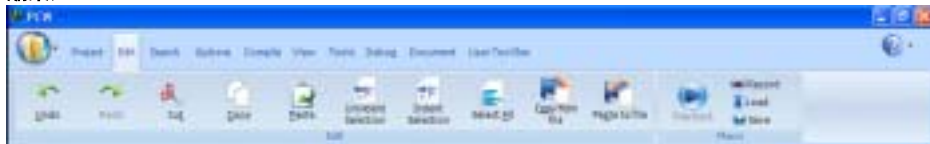
そのプロジェクトに関連したファイルを閉じます。

Find Text in Project[テキストを検索]



与えられたテキストの文字列に対して、プロジェクトの全てのファイルを検索します。


編集メニュー




Undo[アンドゥ]  最後の編集動作を行わない。


Redo[リドゥ]  最後の編集動作を再度実行

Cut[カット]  ファイルから選択したテキストをクリップ・ボードへ移動

Copy[コピー]  選択したテキストをクリップ・ボードへコピー

Paste[ペースト]  クリップ・ボードの内容をカーソル位置へコピー

Unindent Selection[アンインデント選択]  選択されたコードのエリアをインデントしません。


Indent Selection[インデント選択]  選択されたコードのエリアをインデント


Select All[全てを選択]  全テキストのハイライト


Copy from File[ファイルからコピー]  ファイルの内容をカーソル位置へコピー


Paste to File[ファイルへペースト]  選択したテキストをファイルにペースト


Macros[マクロ] 記録、保存とキーとマウス・ストロークのローディングのためのマクロ

Playback[プレイバック]  最後に記録されたもの、又は、ロードされたマクロをプレイバックします。

Record[レコード]  プレイバックされるべきキーストロークの記録を開始、又は、セーブ、そして、後でロードされます。


Stop[ストップ]  プレイバックされるべきキーストロークの記録を中止、又は、セーブ、そして、後でロードされます。

Load[ロード]  セーブされたプレイバックされるマクロをロード

Save[セーブ]  ロード、又は、後でプレイバックをされるべき記録されたマクロをセーブ

サーチ・メニュー



Find[検索]  選択されたサーチ条件で指定した文字列で開かれたファイルの次の出来事を検索


Find Next[次を検索]  指定した文字列をプロジェクト内で検索

Ignore Case[無視するケース] 検索はチェックされている場合は無視


Search Forward[前方検索] ファイル内で前方を検索


Search Backward[後方検索] ファイル内で後方を検索


Search Entire File[全ファイルを検索] 全ファイルを検索


Find Text in Project[プロジェクトのテキスト検索]  指定した文字列をファイル・ウィンドウ内で検索して開きます。プロジェクトが選択されますと現在、または、最後にコンパイルされたプロジェクト内の全てのファイルを捜がします。インクルード・ディレクトリが選択されると、それはインクルード・ディレクトリー内の全てのファイルを捜します。与えられたディレクトリが選択されると、現在のディレクトリの中の全てのファイルを検索します。CCS Examples が選択されますと、サンプル・ディレクトリの全てのファイルを検索します。

Replace[置き換え]  リプレース[置き換え]ウィンドウを開きます。

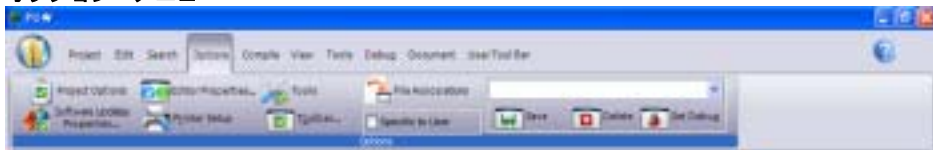
Find Next "Word"[次のことばの検索]  開かれたファイル内で選択されたテキストの次の指定箇所を検索

Goto Line..[カーソルを指定番号へ]  行番号を指定し、カーソルを指定行に移動

Toggle Bookmark[トグル・ブックマーク]  カーソル位置でブックマーク(0 から 9)をセット

Goto Bookmark[ブックマークへ]  カーソルを指定したブックマーク(0-9)へ移動

オプション・メニュー



Project Options[プロジェクト・オプション]

プロジェクト・オプション・ダイアログを開きます。インクルード・ファイル、インクルード・ディレクトリ、グローバル定義と出力ファイルを追加/移動などのセッティング
ファイル、インクルード・ファイル、グローバル定義と出力ファイルの4つのウィンドウを持っています。

File Tab[ファイル・タブ]

ファイル・タブはターゲット・チップと現在のプロジェクトの全てのソースファイルをリストアップします。Add 又は、remove ボタンを使ってプロジェクトにファイルを追加、削除して下さい。プロジェクトがマルチ・コンパイルされたユニットを使用している場合は、multiple compilation units をチェックして下さい。そのリンカーを使用する場合は separate linker オプションをチェックして下さい。これらのオプションはタブのボタンで新規プロジェクトのためにデフォルトとしてセッティングするためにチェックすることでデフォルトを設定することが出来ます。変更を保存するためには apply ボタンをクリックしておいて下さい。

Include File Tab[インクルード・ファイル・タブ]

インクルード・ファイル・タブはそれぞれのディレクトリの仕様がそのプロジェクトのためのインクルード・ファイルを検索するのに使用されたものです。この情報は PJT ファイルに保存されます。ブラウズ・ボタンを使用してフォルダをブラウズして、リストに追加するには add ボタンを使用して下さい。アイテムを上下に移動するには move up, move down ボタンを削除するには delete ボタンを使用します。デフォルトのインクルード・ディレクトリ設定を使用するには default ボタンを使用して下さい。これらのオプションはタブの下の新規プロジェクト・ボックスのためのこれらのセッティングをチェックすることでデフォルトにすることが出来ます。Apply ボタンで変更を保存して下さい。

Global Defines[グローバル定義]

グローバル定義は #defines の設定がコンパイル時に使用されるようにします。これはいくつかの #defines をプログラムの先頭で定義するのと同じです。これは例えばデバッグの定義がコードを変えないで使用することが出来ます。これらの定義はチェックにしますと PCW[以下、ウィンドウズ版を PCW と言います。]コンパイラだけに適応されます。これらは MPLAB でのコンパイルには適応されません。

Output File[出力ファイル]

出力ファイルは出力ファイル形式の選択をします。:

- Standard COD – 標準 PICmicro® MCU cod デバッグ・ファイル
- Expanded COD – アドバンスド・デバッグ情報を持った COD ファイル
- COFF File – デバッグ情報を持った Microchip COFF ファイル
- リスト・ファイル・オプション
- CCS Basic - CCS フォーマット・リスト・ファイル
- With Opcodes – マシン・コードを持った標準フォーマット
- Old Standard – 古い MPASM フォーマット
- Symbolic – アセンブリ内に C シンボルを含む
- オブジェクト・ファイル・オプション

8 bit HEX - 8 Bit Intel HEX file

16 bit HEX - 16 bit Intel HEX file

Binary - Straight binary (ヒューズ情報なし)

エラー・ファイル・オプション

Original – 古いマイクロチップ標準

Standard – 現在のマイクロチップ標準

Multi-lines – show all errors 又は、show all warnings のチェック・ボックスを選択するかで全てのエラー・メッセージと/又は、警告を表示します。

Other File Options – ボックスを選ぶことでコール・ツリー、ファイル、スタティクス・ファイルとスクラッチ・ファイルを作成



Editor Properties[エディター・プロパティ]

クリックしますと、Editor Properties ウィンドウがオープンされ、ユーザーが各種オプションを設定することが出来ます。

クリックしますと新しいEditor Properties Windowが開きます。エディターの特性をセットアップするための多くのオプションを持っています。Editor Properties Windowには、以下で説明される3つのタブがあります:

General[一般]

Syntax Highlighting – チェックが入りますと、エディターはCのキーワードとコメントをハイライトします。

Auto Highlight brackets - チェックが入りますと、エディターは1つの場所にカーソルが置かれた時、自動的にマッチしたブラケット(丸括弧)をハイライトします。

Auto Indent – 選択され、そして、ENTERを押しますとカーソルは前の行の最初の文字の下次の行に移動します。選択されない時はENTERによってつねに次の行の最初に移動します。

Show Active Line – 選択されますとアクティブ行の上にカーソルが示されます。

Auto positioning the "{" – 選択されるとき、マッチした { 又は、) をハイライトします。エディターは { } をカウントし始め、そして、マッチしたそれをハイライトします。単に項目の1つにカーソルを置きますとマッチしたものがハイライトされます。

Use Code Folding – 選択されますと、コードの可読性を向上させるために1行に減少させるために関数に対してエディターが新しいコードの折りたたみ機能を使用します。

Order Open Files – ファイル・タブをアルファベットにアレンジします。

Recall Open Files – 選択されますと、PCW[ウィンドウズ版一般を言います。]はいつも最後に閉じた時に開いていたと同じファイルを開きます。選択されないと、PCWは開いているファイルなしでいつも始まります。

Tab sizeはタブ位置の間のキャラクタ数を測定します。 Tabsはタブとタブがスペースに変換されるかどうかによって等しくされるか、またはタブとして残っているスペースの数を設定できます。

Window設定オプションで、ユーザはエディタ(水平で垂直な)にスクロールバーを選択できます。

Display[表示]

Visible left Margin – 選択しとエディターの左マージンが見えるようになります。

Visible Right Margin - 選択しとエディターの右マージンが見えるようになります。

Show Line Numbers – 行番号を選択しとエディターに示されます。

Left Margin Width – 左マージンの幅

Right Margin Width – 右マージンの位置

Editor Font – エディターのフォントを選択

Option Editor Properties Display Editor Font で設定して下さい。

日本語フォント名自体が「文字化け」して正しく表示されませんが文字化けしているフォントが日本語フォントです。

またEDITERにおいてサイズを大きく設定しているとカーソルと文字の位置が合わない場合がある様です。種類とサイズを最適に合わせていただければと思います。

Font Size – エディター・フォントのサイズ

Font Style – エディター・フォントのスタイル選択 (イタリック/太字/下線)

Color Scheme – ツール・バーのスタイル選択(ブルー/シルバー/オリーブ/XP)

サンプル・ボックスに選択されたオプションのサンプルが表示されます。

Color[カラー]

このタブでユーザは要素をハイライトさせる構文のために色を選択して、色を選択するのにフォアグラウンドと背景色を使用します。 カラー・スピード・セッティングを使用してエディターのための背景色を選択して下さい。

Tools[ツール]



ICD, Mach X, MPLAB, その他のプログラムの様な外部にあるもののセットに使用。各ツールに対して名前、場所とコマンド・ライン・オプションを指定します。

Files Associations[ファイル関連付け]



CCS のプログラムへのファイルの関連をセットアップ。拡張子とプログラムを選択します。

Software Update Properties[ソフトウェア・アップデートのプロパティ]



アップデート・ユティリティがどの頻度で、どのようなアクションを取るかを設定出来ます。

Properties[プロパティ] ユーザーとアチーブ・ファイルを特定します。

Printer Setup[プリンター設定]



プリンターの設定のためのアイアログ・ボックスを開く

Toolbar Setup[ツール・バーのセットアップ]



このウィンドウは2つのタブを持っています。 ツールバー・タブとキーボード・タブ

Toolbars

このタブはツール・バーに追加出来るアイコンのリストを示します。 選択は標準とユーザー・ツール・バー

Use Large Icons – 選択しますと大きなアイコンが使用されます。

User Toolbar Name – このボックスを使ってユーザー・ツールバーのための名前を指定

Keyboard

このタブは異なったメニュー・カテゴリーでのコマンドのためのキーボード・ショートカットをリストします。 ユーザーは必要に応じて変えることが出来ます。

新しいショートカットを割り当てるため、それを取り除く、又は、全てのボタンをリセットするために割り当て、取り除きを使用するか、又は、全ボタンをリセットします。


デスクトップ

保存、削除、又は、デバッグ・デスクトップとして現在のデスクトップをセットするために保存、削除、又は、セット・デバッグ・アイコンを使用します。


Save Desktop[デスクトップの保存]



現在のデスクトップ・レイアウトを保存

Delete Desktop[デスクトップの削除] 

保存されたデスクトップのリストからデスクトップを取り除く


Set Debug Desktop[デバッグ・デスクトップのセット] 

デバッグ・モードに入る時ロードされるデスクトップをセット

UserチェックボックスへのSpecificが選択される時、PCWは各ユーザのために異なった設定を保存します。

コンパイル・メニュー



Compile[コンパイル] 

現在のコンパイラを使ってステータス・バーの現在のプロジェクトをコンパイルします。

Compile Window [コンパイル・ウィンドウ]

コンパイル・ウィンドウはアクティブなプロジェクトをF9、又は、コンパイル・リボンの'Compile'アイコンをクリックすることで現れます。このウィンドウはコンパイラのバージョン情報と登録の情報を示します。

Project[プロジェクト]

現在コンパイルされようとするメイン・プロジェクト・ファイルを表示します。

Compile[コンパイル]

そのプロジェクトに追加されようとするプロジェクトで使われる全てのファイルを表示します。

Outputs[出力]

このウィンドウはコンパイルがエラーなしで完了しますと各種出力メッセージが表示されます。

RAM and ROM[RAMとROM]

ターゲットのチップで利用可能な総量にたいしてRAM とROMの使用率がパーセンテージで表示されます。

Output Files[出力ファイル]

コンパイル中にプロジェクトがコンパイラによって生成、又は、修正されたファイルが表示されます。


Miscellaneous Information [その他の情報]

Files – プロジェクトで使用されたファイルの数。 .Cと.Hのみ。

Statements – プロジェクトのCステートメントの数

Lines – プロジェクトでの行数。 Cステートメントと全てのコメントを含みます。

Time – プロジェクトをコンパイルするのに必要な時間。

Build[ビルド] 

プロジェクトをビルド。最後にビルドされてから変更されたコンパイル・ユニットのみ。

Build All[全てのビルド]



全てのユニットをコンパイル、そして、現在のコンパイラを使って、現在のプロジェクトをビルド（名前は右下）

Clean[クリーン]



現在のプロジェクトの出力ファイルを削除

Microchip 12 bit

12ビットPICマイクロコントローラのために12ビット・コンパイラ(PCB)を選択

Microchip 14 bit

14ビットPICマイクロコントローラのために14ビット・コンパイラ(PCM)を選択

Microchip 16 bit

16ビットPICマイクロコントローラのために16ビット・コンパイラ(PCH)を選択

Microchip 24 bit

24ビットPICマイクロコントローラのために24ビット・コンパイラ(PCD)を選択

Compiler[コンパイラ] 必要なコンパイラ(PIC12,16,18等)を選ぶためのプルダウン・メニュー

Lookup Part[パーツ選択] デバイスと必要なコンパイラが自動的に選択されます。

Program Chip[プログラム・チップ]



ボタンをクリックすることで、ターゲットをプログラムするためのツールを選択し、そのセットアップを行います。CCSのICD、又は、Mach Xプログラマーのオプションのリスト。そして、SIOWプログラムに接続されます。

Debug[デバッグ]



ボタンをクリックすることで、ターゲットをデバッグするためのツールを選択し、そのセットアップを行います。

デバッグのために .hexファイルを入力、そして、.asmファイルを出力します。

C/ASM List[C/ASM リスト]



リード・オンリー(READ ONLY)モードでリスティング・ファイルをオープンします。

リスト・ファイルを見るにはファイルはコンパイルしなければいけません。コンパイル後

このファイルをアップデートします。リスティング・ファイルは各Cソース行とその行のために生成された関連したアセンブリー・コードを表示します。

*サンプル :

```
.....delay_ms(3)
0F2:  MOVLW 05
      MOVWF 08
0F4:  DESCZ 08,F
0F5:  GOTO 0F4
.....while input(pin_0);
0F6:  BSF 0B,3
```

Symbol Map[シンボル・マップ]



リード・オンリー(READ ONLY)モードでシンボル・ファイルをオープンします。シンボル・ファイルを見るにはソースファイルをコンパイルしなければなりません。コンパイルされる度にシンボル・ファイルは更新されます。シンボル・ファイルは各レジスター位置と各位置でどのようなプログラム変数をセーブしたかを表示します。

最後にコンパイルされたプログラムのRAMメモリー・マップを表示します。

マップは各 RAM 位置の使用を表示します。ある位置では現在の実行されている状況による再使用されていますのでマルチになっています。

サンプル：

```
08      @SCRATCH
09      @SCRATCH
0A      TRIS_A
0B      TRIS_B
0C      MAIN.SCALE
0D      MAIN.TIME
0E      GET_SCALE.SCALE
0E      PUTHEX.N
0E      MAIN.@SCRATCH
```

Call Tree[コール・ツリー]



View メニューの Call Tree コマンドでコールツリーが表示されますが、修正することはできません。リード・オンリー(READ ONLY)モードでツリー・ファイルをオープンします。コールツリーを見るにはソースファイルをコンパイルしなければなりません。コンパイルされる度にコールツリーは更新されます。コールツリーは各関数が使用する ROM 及び RAM、関数の中でコールされる関数を表示します。

(inline)は@で始まるインライン・プロシージャの後に現れます。

プロシージャ名のは s/n の形式で数字が表示されます。ここで、s はプロシージャのページ番号、n は 必要とされるコードのサイズです。

もし、s の個所が?となっている場合は、コンパイラが ROM 領域を使い果たしたことを示します。もし、s の個所が?となっている場合は、コンパイラが ROM 領域を使い果たしたことを示します。

サンプル： Main 0/30

```
INIT 0/6
WAIT_FOR_HOST 0/23 (Inline)
DELAY_US 0/12
SEND_DATA 0/65
```

Statistics[スタティスティクス]



リード・オンリー(READ ONLY)モードでスタッツ・ファイルをオープンします。スタッツ・ファイルを見るにはソースファイルをコンパイルしなければなりません。コンパイルされる度にスタッツ・ファイルは更新されます。スタッツ・ファイルは各関数が使用する ROM 及び RAM を表示します。

コンパイラはマッケイブ(McCabe)のサイクロマチック数とハルステッド(Halstead)の複雑度測定を使用することでコードの経路複雑度と計算量を評価します。派生している保全性指標 [Maintainability Index]はソースコードの構造的で原文の複雑さに関する良い洞察を与えます。

ハルステッド(Halstead)の複雑度測定:

ハルステッドの複雑度測定はソースコードから直接計算量を強調することで、プログラム・モジュールの複雑さを測定するために開発されました。。それは、複雑さを決定するのにモジュール内で演算子とオペランドを使用します。

ハルステッド測定はプログラムのソースコードから直接引き出された 4 つのスケーラー数に基づきます。

n1 = 個別演算子の数

n2 = 個別オペランドの数

N1 = 演算子の合計数

N2 = オペランドの合計数

上記の数を使って、次の5つの測定が引き出されます。

測定	シンボル	公式
Program Length	N	$N=N1+N2$
Program Vocabulary	n	$n=n1+n2$
Volume	V	$V=N*(LOG2n)$
Difficulty	D	$D=(n1/2)*(N2/n2)$
Effort	E	$E=D*V$

マッケイブ(McCabe)の循環的複雑度:

プログラム・モジュールを通して線形的に独立した経路の数を測定します。マッケイブのサイクルマチック数 $v(G)$ はコードの断片を通してフロー制御の複雑さを示しています。

簡単には、もし、D が決定ノード数であれば、その時は、
 $v(G)=D+1$

Maintainability Index[保守性指標]:

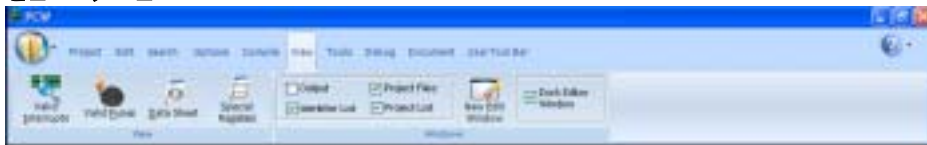
プログラムの保守性は、Maintainability Index(MI)を形成する広く使用されて一般的に利用可能な測定の組み合わせを使用することで計算されます。1セットのプログラムの基本的なMIはサイクルマチック数の複雑さ、平均したハルステッド・ボリューム、コードの平均行とコメント行の平均したパーセントから得られた多項式です、



Debug File[デバッグ・ファイル]

リード・オンリー(READ ONLY)モードでデバッグ・ファイルをオープンします。リスティング・ファイルは各Cソース行とその行のために生成された関連したアセンブリー・コードを表示します。

ビュー・メニュー



Valid Interrupts[有効インターラプト(割り込み)]

現在のプロジェクトで使用されているデバイスの#INT_keywordを使って有効なインターラプトのリストを表示します。その他のチップのインターラプト(割り込み)はドロップ・ダウン・メニューを使って見ることが出来ます。

Valid Fuses[有効ヒューズ]

現在のプロジェクトで使用されているデバイスに関連した#FUSES のディレクティブを使って有効なヒューズのリストを表示します。

Data Sheets[データシート]

このツールは選択されたコンパイラでサポートされているMicrochip社のパーツのデータシートをアクロバット・リーダー形式で表示します。

Part Errata[パーツ・エラッタ]

データシート同様に関連するパーツのエラッタ・データベースを見ることが出来ます。

重要: データシートやエラッタは常にMicrochip社のものですので、必ずMicrochip社から最

新しい情報とデータシートをご参照下さい。CCS コンパイラでのこの機能はバージョンによっては古いデータシートとなりますので、参考に過ぎません。

Special Registers[スペシャル・レジスター]

パーツに関連したスペシャル・ファンクション・レジスターを表示します。

New Edit Windows[ニュー・エディット・ウィンドウ]

ファイルを並べて見るために新しい編集ウィンドウをタイルにすることが出来ます。

Dock Editor Windows[ドック・エディター・ウィンドウ]

このチェック・ボックスを選択しますと、そのエディター・ウィンドウをIDEにドックすることが出来ます。

Project Files[プロジェクト・ファイル]

このチェック・ボックスが選択された時、プロジェクト・ファイルのスライドアウト・タブが表示されます。これにより素早く全てのプロジェクト・ソース・ファイルと出力ファイルにアクセスすることが出来ます。

Project List[プロジェクト・リスト]

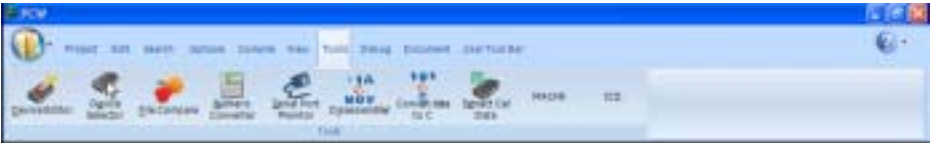
このチェック・ボックスを選択しますと、プロジェクト・ファイル・スライドアウト・タブを表示します。プロジェクト・スライドアウト・タブは全ての最近のプロジェクト・ファイルを表示します。

Output[出力] このチェック・ボックスが選択しますと、コンパイラから生成された警告メッセージとエラー・メッセージを表示させることが出来ます。

Identifier List[識別子リスト]

このチェック・ボックスが選択しますと、拡張子のスライドアウト・タブを表示します。これは関数、タイプ、変数と定義のようなプロジェクト識別子に素早くアクセスすることが出来ます。

ツール・メニュー



Device Editor[デバイス・エディター]

このツールはコンパイルをコントロールするためにコンパイラによって使用されるデバイス・データベースを編集します。サポートされている各プロセッサに対する主要な特性(チップ・メモリ、インターラプト(割り込み)、ヒューズとその他ペリフェラル)を編集するのに使用されます。

Device Selector[デバイス・セレクター]

デバイスのパラメトリック選択を行えるようにデバイスのデータベースを使用します。このツールは選択評価基準に基づいた全ての適切なデバイスを表示します。

File Compare[ファイル比較]

2つのファイルを比較します。ソースファイル又は、テキスト・ファイルが選択されますと、行ごとの比較が行われます。リスト・ファイルが選択されますと、比較はより意味のあるものにするためにRAMとROM又は、いずれかを無視するようにセットされます。

*例えば、プログラムの最初にASM行が追加されますと、ノーマル比較は各行にフラッグを立てます。ROMアドレスを無視することで、特別な行のみが変更されてフラッグされます。2つの出力形式が利用出来ます。1つは表示のため、1つはファイル又は、印刷のためです。

Numeric Converter[ニューメリック変換]

このユーティリティーは異なったフォーマットのデータを変換するために使用します。バイナリー、Hex、IEEE と符号付き整数(signed)と符号無し整数(unsigned)の様な各種フォーマットのデータを同時に見ることが出来ます。

Serial Port Monitor[シリアル・ポート・モニター]

PIC をシリアル・ポートに簡単に接続するのに使用できるツールです。データは ASCII、又は、Hex フォーマットとして見ることが出来ます。全ての hex ファイルを PIC に送信することが出来ますので、ブートローディング・アプリケーションに便利なツールです。

Disassembler[ディスアセンブラー] HEX ファイルを入力として、そして、ASM を出力します。

*ASM はインライン ASM として使用できる形式です。このコマンドは HEX ファイルを入力とし、選択された部分を抽出してインライン・アセンブリーとして C プログラムに挿入することが出来ます。オプションはアセンブリー・フォーマットを選択することが出来ます。

- 12 又は、14bit オペコード
- アドレス、C、MC ASM ラベル
- HEX 又は、Binary
- Simple, ASM, C numbers

Convert Data to C[Cヘデータ変換]

このツールはテキスト・ファイルからデータを入力し、#ROM 又は、CONST ステートメントのフォームでコードを生成します。

Extract Calibration[キャリブレーション・データ抽出]

HEX ファイルを入力し、キャリブレーション・データを C インクルード・ファイルへ抽出します。この機能はある種の PIC チップからプログラム・メモリーのトップにキャリブレーション・データを保持するのに便利です。

Mach-X[マッハX] Mach-X.exeプログラムを呼び出し、現在のプロジェクトのhexファイルをそのチップへダウンロードします。


ICD[ICD] ICD.exeプログラムを呼び出し、現在のプロジェクトのhexファイルをそのチップへダウンロードします。

デバッグ・メニュー



Enable Debugger[イネーブル(有効)・デバッガー]

デバッガをイネーブル(有効)し、デバッガー・ウィンドウを開き、コードをターゲット・チップにダウンロードします。そして、デバッガでターゲットをリセットします。

Reset[リセット]  デバッガ上でターゲットをリセットします。

Run / Halt[実行/停止]   ターゲット・チップの実行/停止。

実行されている間、どのデバッグ・ウィンドウも現在の情報をアップデートしません。プログラムはブレークポイントに達した時、または、stopボタンを押した時に停止します。

Single Step[シングル・ステップ]

1 つのソースコードを1度に実行します。エディターでソースコード、又は、リスト・ファイル・タブのいずれがアクティブになっているかによってCソースコードの1行、又は、ASMコードが実行されます。

Step Over[ステップ・オーバー]

ターゲット・コードをステップ・オーバーします。関数コールをステップ・オーバーするのに便利です。

Run to Cursor[カーソルへ実行]

カーソルの位置までターゲット・コードを実行。コードの希望する位置にカーソルを置き、そして、このボタンをクリックしますとそのアドレスまでコードを実行します。

Snapshot[スナップ・ショット]

各種デバッグ情報を記録します。ウォッチ、RAM値、データEEPROM、ROM値、周辺機器[ペリフェラル]の状態の様なデバッグ情報をログすることが出来ます。このログをセーブ、印刷、上書き又は、追加することが出来ます。

Run script[スクリプト実行]

このツールはIDEの統合デバッガはC-スタイル・スクリプトを実行します。プログラムの関数や変数はアクセスすることが出来、そして、デバッガはその結果のリポートを作成します。



コマンド・ライン・オプション force_new を使って 2 つの PCW を実行する。

```
c:¥Program Files¥ PICC ¥ pcw ----->> 最初の PCW を開始
```

```
c:¥Program Files¥ PICC ¥ pcw + force_new ----->> 2 番目の PCW を開始
```

これで 2 つの ICD ユニットを使って接続された 2 つのターゲットで書く PCW のプロジェクトを別個にデバッグすることが出来ます。各デバイスに使用されている USB ポートはデバッガの Debug Configure タブの'port'で見ることが出来ます。

どの ICD デバッガが Debug Configure タブの'port'で設定されたポートを使用するか選択することが出来ます。

ノート: PC から ICD をはずす前にデバッガを向こうにしなければいけません。

同じ PC から 2 つの ICD コントロール・パネルを実行

ICD.exe アプリケーション・プログラムを実行することで複数の ICD コントロール・パネルを一度に使用することが出来ます。開始時にウィンドウが自動的に各 ICD に USB ポートを割り当てます。このポートを切り替えるにはコントロール・パネルの'Configure port'を使って下さい。

Debug Windows[デバッグ・ウィンドウズ]

このドロップ・メニューが特定のデバッグ・タブを見ることが出来ます。見たいドロップ・ダウン・リストのタブ名をクリックしますと、デバッガ・ウィンドウにタブが現われます。

ブレーク

ブレークポイントをセットするのに、ソースかリスト・ファイルの必要なライン位置へエディタ・

カーソルを動かしてください。そして、デバッガのブレーク・タブを選択し、+アイコンをクリックしてください。

ブレークの動作はハードウェア・ユニットの種類で異なります。

例えば、ICD で PIC16 を使うと、ただ 1 つのブレーク設定が可能な事と、プロセッサの実行はストップ動作の前の状態でブレークポイントをセットされたライン(アセンブラのライン)までを実行します。

コンフィギュア

コンフィギュア・タブはどのハードウェア・デバッガが接続されるかを選択します。他のコンフィギュレーション・オプションは使われるハードウェア・デバッガにより異なります。また、コンフィギュア・タブはターゲットにコードを手動で再ロードすることが出来ます。

もし、デバッガ・ウィンドウが開かれていて、"Reload target after every compile"ボックスが選択されていますと、プログラムがコンパイルされる度にプログラムがターゲットにダウンロードされます。

デバッガ・プロファイルはウォッチされる変数、デバッガ・ウィンドウの位置とサイズとブレークポイント設定のようなすべてのデバッガ・タブの選択が含まれています。プロファイルはファイルにセーブされ、そして、コンフィギュア・タブからロードできます。セーブ又は、ロードされた最後のプロファイル・ファイルはプロジェクト .PJT ファイルにセーブされます。

コンフィギュア・ウィンドウは下記の各種セッティングを提供します。:

Compile Reload: デフォルトは True。True の場合、プロジェクトが再コンパイルされ毎に自動的にターゲットに再ロードされます。

Mouse Over Eval: デフォルトは True。もし、True の場合、式はマウス・オーバー・イベントとして評価されます。マウスを式の上において値を見て下さい。

Timeout Mouse Over: デフォルトは True。True の場合、式はマウス・オーバー・イベントとして評価されます。マウスを式の上において値を見て下さい。True の場合、マウスをマウスが上に置かれますとタイムアウト、そうでない場合、マウスの移動で消えます。デフォルトは True。

Mouse over radix: マウス・オーバーの結果に対して、hex, binary, decimal 等を指定することが出来ます。

Usersteram enabled: True の時。ICD ジャック(ターゲット・ジャックのピン 6)のピン 1 がユーザー・ストリウムのためにピン B3 に接続されている場合。他の目標に使用されているか、または、ICD に接続されない時、このオプションを無効にしてください。

Echo On Monitor: True の時。タイプされた文字がモニター・ウィンドウでエコーされます。

デフォルトは True。

Multiple Breakpoints Enabled: この機能は PIC18 ターゲットのみで利用出来ます。

PIC16 では false にセットされ、変更することが出来ません。

デフォルトで false にセットされるので、1 つのハードウェア・ブレークポイントのみ利用出来ます。複数のブレーク・ポイント機能のために true にセットした時は必ず適応(apply ボタン)させて、デバッガを閉じて、再スタートさせて下さい。この機能が変更された時、必ず適応(apply ボタン)させて、デバッガを閉じて、再スタートさせて下さい。

Monitor Font Size: モニター画面のフォントサイズをセット

Target Type: ターゲットのタイプをセット。デフォルトはありません。正しいプログラミング・モードを使用するために正しいオプション(pic18fj 又は、Nonpic18fj)をセットし、そして、デバッガを有効にします。

ICD F/W: デバッガのファームウェア・バージョンを表示

Processor: ターゲットのマイクロコントローラを表示

Port: デバッガのポート番号/デバイス名。

Baud Rate: PC がデバッガと通信する速度

CPU Fuse Warning: CPU のヒューズがマイクロプロセッサ・モードにセットされている時に警告します。デフォルトは True です。

Low Voltage Warning: 現在のプログラム/イレース設定がターゲット電圧(ローVDD(3V))に対応し

ていない時に警告します。

Mach X supplies VDD: このオプションは MachX をターゲットのデバッグに使用する時のみ利用出来ます。True にセットされますと MachX がターゲットに VDD を供給します。デフォルトは False です。

DATA EEPROM

デバッグの Data EEPROM タブはターゲットの Data EEPROM を示します。赤い番号はプログラムが最後に停止されてから変更された状態を示しています。変更したい Data EEPROM の位置でダブル・クリックして変更したい内容を入力してください。全ての番号はヘキサです。

ICD ユーザーのための特別ノート:

ICD ユニットを使用するとき、CCS ファームウェアが ICD にインストールされていなければいけません。ファームウェアをインストールするには "Configure Hardware" をクリックし、そして、ICD firmware ロードするためには中央のトップのボタンをクリックして下さい。

エバルエーション[評価]

このタブは C 記述の評価のためのものです。これはウォッチ機能をよく似ていますが、大きな構成や配列のために用いられ、さらに空き領域があることが異なります。

エバルエーションはターゲットの C 関数を呼ぶことができます。この場合、すべてのパラメータを与えなければいけません。関数の結果は Result ウィンドウに表示されます。

この機能はすべてのデバッグ・プラットフォームで利用可能ではありません。

ログ

ログ機能はブレイク、ウォッチとスナップショットの組み合わせです。ブレイク番号と評価する数式を各ブレイク毎に指定してください。プログラムは数式が評価され結果がログ・ウィンドウに記録された後、再スタートします。複数の数式はセミコロンで別々に指定します。ログ・ウィンドウはファイルにセーブすることが出来ます。ファイルの中の各数式結果はスプレッド・シート形式のプログラムにインポートし易い様にタブでセパレートされています。

モニター

モニター・ウィンドウはターゲットからのデータを示し、そして、ターゲットに送るデータの入力が出れます。これはターゲット上で下記の様に行われます。

```
#use RS232(DEBUGGER)
...
printf("Test to run? ");
test=getc();
```

B3 ピンはデフォルトでこの機能をインプリメントするためにターゲットで使用され、ICD ジャック(ターゲット・ジャックのピン 6)のピン 1 に接続されます。もし、異なったピンを使用する場合は、#use rs232(debugger, xmit=pin_xx,rcv=pin_xx)ディレクティブで指定して下さい。

デフォルトで有効。必要な場合は、Debug Configure ウィンドウで無効して下さい。エコー・バックする必要の無い文字が入力された場合は、エコー・オプションで false にセットして下さい。Ctrl+F、又は、print("/f")でモニター画面をクリアすることが出来ます。

ペリフェラル

このタブはターゲット・スペシャル・ファンクション・レジスターの状態を示します。このデータは関数によって構成されます。レジスターの下にレジスターの各フィールドがどのビット・パターンがリストされます。

RAM

デバッグ RAM タブはターゲット RAM を示します。赤い番号はプログラムが最後に停止されてから変更された状態を示しています。ブラック・アウトされている位置は物理的にレジスタが存在しないか、デバッグ中に利用できない事を表わしています。RAM を変更するには変更したい位置で

ダブル・クリックして変更内容を入力してください。全ての番号はヘキサです。

ROM

ROM タブはターゲットのプログラム・メモリーの内容をヘキサ値とディスアSEMBルの両方で示します。このデータは最初にロードした HEX ファイルからのもので、ユーザーが要求しないかぎりターゲットからは変更されません。ターゲットから再ロードするにはウィンドウで右クリックしてください。

タスク

タスク・ウィンドウは現在実行中とサイクル時間等の RTOS タスクの詳細を表示します。ツ

ウォッチ

ウォッチ・タブがウォッチする新しい式を入力する選択がされているときに+アイコンをクリックして下さい。ヘルパー・ウィンドウがポップアップしウォッチするプログラムの識別子を見つけることができます。通常の C 表現は下記のようにウォッチされます。

```
X  
X+Y  
BUFFER[X]  
BUFFER[X].NAME
```

ウォッチに入る時にソース・ファイルにエディタ・カーソルのあるところの式がどのように影響するかを見て下さい。

例えば、2つの関数 F1 と F2 があるとすれば、ウォッチ式として単に I を入力します。I によって得られるものは、どの関数にカーソルが有るかによります。

そこで、いかなる関数名を持った変数と変数(F1I のように)を指定する期間を実行出来ます。

デフォルトでは結果のフォーマットはデシマルです。他の指定子は：

expr, h - hexadecimal format

expr, b - binary format

expr, c - character format

expr, s - string format

expr, .x - floating with x decimal places

スタック

このタブは現在のスタックを示します。最後に呼び出された関数とすべてのパラメータはリストのトップに表示されます。

PIC16 でスタックを見るにはソース・ファイルで #DEVICE CCSICD=TRUE がなければいけません。

これによりコンパイラはエキストラ・コードを生成しデバッガでスタックが見えるようになります。

PIC18 でスタックを見るにはソース・ファイルに #device ICD=TRUE があるだけで十分です。

Show All 全てのデバッガ・ウィンドウを表示

Hide All 全てのデバッガ・ウィンドウを非表示

Reset Debug Windows 全てのデバッグ・ウィンドウをそれらの元の位置へリセット

ドキュメント・メニュー



Format source[フォーマット・ソース]



このユーティリティーはインデント、カラー構文ハイライトとその他のフォーマット・

オプションに対応して ソースファイルをフォーマットします。

Generate Document[ドキュメント作成]



これはソースコード・ドキュメンテーションとして、.RTFフォーマットで出力ファイルを作成するためにソースコードからのコメントにユーザーが作成した .RTFフォーマットのテンプレートをマージするドキュメント・ジェネレータ・プログラムを呼び出します。

RTF editor[RTFエディター]



プロジェクトに容易にドキュメンテーションを統合するためのフル機能のRTFエディターであるRTFエディター・プログラムを開きます。

Flow Chart[フロー・チャート]



素早く、容易にチャートを作るためのフローチャート・プログラムを開きます。このツールは回路図を含む簡単なグラフィックを作成するのに使用することが出来ます。

Spell Check Quotes[引用附]



引用部分のすべての言葉のスペル・チェックを行います。

Specc Check Comments[コメント]



ソースコードのすべてのコメントのスペル・チェックを行います。

Print all files[全ファイル印刷]



現在のプロジェクトの全てのファイルを印刷

ヘルプ・メニュー

クリックしますと以下のドロップ・ダウン・メニューが出ます。

Contents[内容]



ヘルプ・ファイル - 内容のテーブル

Index[索引]



ヘルプ・ファイル索引

Keyword at Cursor[カーソル位置のキーワード]

カーソル位置のキーワードをヘルプ・ファイルの索引で検索。シヨントカットはF1

Debugger Help[デバッガ・ヘルプ]



デバッガ機能のヘルプ・ファイル

Editor[エディター]

PCW(PCWH)で使用できるエディター・キーのリスト。シヨントカットは

Shif+F12

Data Types[データ・タイプ]

ベーシック・データ・タイプのヘルプ・ファイル

Operators[演算子]

PCW(PCWH)で使用される演算子のヘルプ・ファイル

Statements[ステートメント]

ステートメントのリスト

Preprocessor[プリプロセッサ]

プリプロセッサのリスト

Commands[コマンド]

コマンド

Built-in Functions[ビルトイン関数]

コンパイラで供給されているビルトイン関数のリスト

Technical Support[テクニカル・サポート]

CCSに英文で直接E-Mailでコンタクトするためのテクニカル・サポート・ウィザード

Check for Software Updates[ソフトウェア・アップデートをチェック]

自動的にソフトウェアの現バージョンを見るためにダウンロード・マネージャーを起動

Internet[インターネット] その他の追加情報のためのCCSウェブサイトへの直接リンク

About[アバウト] インストールされているコンパイラとIDEのバージョンを表示