ＭＣＵ（マルチプル・コンパイレーション・ユニット）の例

以下はマルチプル・コンパイレーション・ユニット例の概要です。:

main.c	1番目のコンパイル単位のＣソースファイル
filter.c	2番目のコンパイル単位のＣソースファイル
report.c	3番目のコンパイル単位のＣソースファイル
project.h	プロジェクト全体（上記コンパイル単位全体）に適用されるインクルード・ファイル。全てのコンパイル単位に含まれなければいけません。
filter.h	filter.c用のヘッダファイル。filter.cで使用されている定義を使用する全てのコンパイル単位のＣソースファイルでインクルードしなけばなりません。
report.h	report.c用のヘッダファイル。report.cで使用されている定義を使用する全てのコンパイル単位のＣソースファイルでインクルードしなければなりません。
buildall.bat	全ての単位をコンパイルし、リンクするバッチ・ファイル
build.bat	コンパイルとリンクする必要があるファイルを再コンパイルするバッチファイル
project.pjt	プロジェクト単位を管理します。build.batで使用されます。

各ソースファイルの構成

main.c	filter.c	report.c
#include ファイル: project.h filter.h report.h 定義している関数: main() program 使用している関数: clear_data() filter_data() report_data_line() report_line_number	#include ファイル: project.h report.h 定義している関数: clear_data() filter_data() 使用している関数: report_error()	#include ファイル: project.h 定義している関数: report_line_number report_data_line() report_error()

各ユニットのコンパイルで、
*.o(リローケタブル・オブジェクト)
*.err（エラー・ファイル)
*.osym（ユニット・シンボル)
が作成されます。

リンク後に最終的に作成されるのは
project.hex（最終ロード・イメージ)
project.lst（C と ASM リスト)
project.sym（プロジェクト・シンボル)
project.cof（デバッガ・ファイル)

●例１　コマンド・ラインでプロジェクトをビルドする場合の例:

1. そのプロジェクト・ファイルをディレクトリー内へ移動
2. その buildall.bat ファイルを編集し、そして CCSC.EXE へのパスが正しいかを確認して下さい。
3. DOSプロンプトで、プロジェクトのファイルがあるフォルダに移動します
4. DOSプロンプトでBUILDALL.BATを実行

```
"c:¥program files¥picc¥ccsc" +FM +EXPORT report.c
"c:¥program files¥picc¥ccsc" +FM +EXPORT filter.c
"c:¥program files¥picc¥ccsc" +FM +EXPORT main.c
"c:¥program files¥picc¥ccsc" +FM LINK="project.hex=report.o,filter.o,main.o"
```

●例２　コマンド・ラインで必要なファイルを再コンパイルすることによる自動ビルドの例:

1. project.pjtファイルは以下のように記載します：

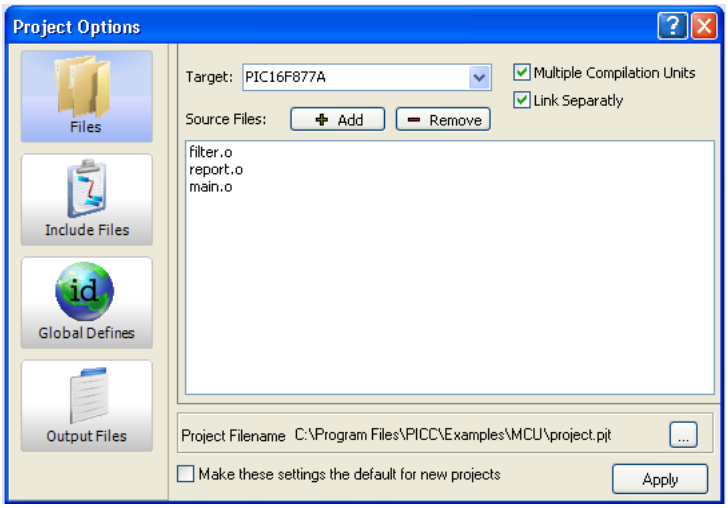
```
[Units]
Count=3
1=filter.o
2=report.o
3=main.o
Link=1
```

2. DOS プロンプトで、プロジェクトのファイルがあるフォルダに移動します
3. DOSプロンプトでBUILD.BATを実行

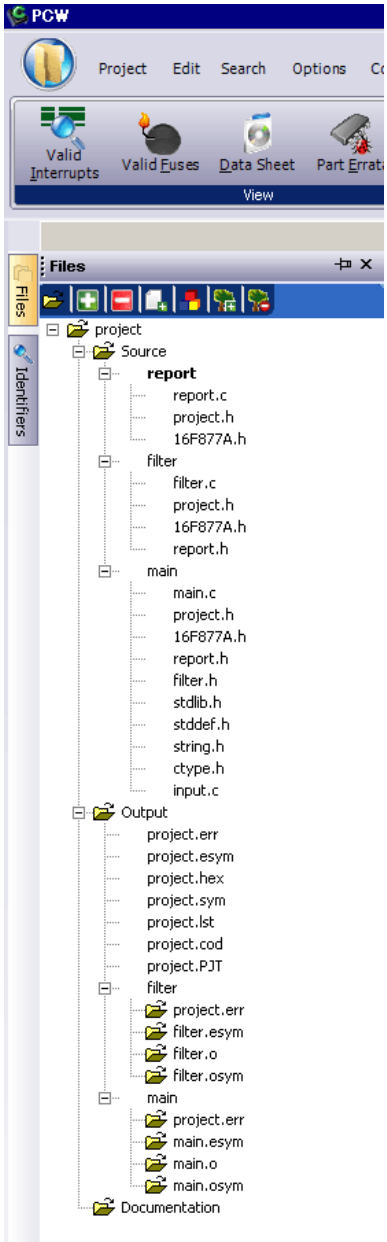
```
"c:¥program files¥picc¥ccsc" +FM BUILD=project.pjt
```

●例３　例２で作成した project.pjt ファイルを IDE で使用する例：

1. IDE(C:¥Program Files¥PIC¥PCW.exe)を起動し、project.pft ファイルを開きます
2. OPTIONS メニューから PROJECT OPTIONS を選択すると、以下のように、現在の設定が表示されます。



3. IDE の左にある[Files]をクリックすると、下図のように関連ファイルが表示されます。



4. このように IDE を使用すると、関連するソースファイルを管理するのに便利です。
   ※例えば修正したいソースファイルをダブルクリックすると、ソースが表示されます。
5. 修正したソースファイルを表示させて、
   Compile メニューから Combile を選択すると、そのソースファイルがコンパイルされて、オブジェクトファイル(*.o)が作成されます。
   Build を選択すると、上記例２のバッチファイルと同様に、修正されたソースファイルのコンパイル、および hex ファイルが作成されます。
   Build All を選択すると、上記例１のバッチファイルと同様に、ソースファイルが再度コンパイルされて、hex ファイルが作成されます。
6. ソースファイル編集等を行うと、project.pjt ファイルは以下のように修正されます
   [PROJECT]
   Processor=0x877A

```
Processor_Text=PIC16F877A
[Target Data]
OptionString=-p +FM
FileList=C:¥Ccs_Picw¥MCU¥project.c
[Opened Files]
1=report.c
2=filter.c
3=
[Units]
Count=3
1=report.o
2=filter.o
3=main.o
Link=1
```