



# 1. 「MPLAB X IDE」と「MPLAB XC8」のダウンロード

(以下は 2012 年 12 月 14 日現在の情報です。WEB ページは予告なく変更されます。)

マイクロチップテクノロジーの WEB ページからダウンロードします。検索サイト(Google, Bing, YAHOO など)から「マイクロチップテクノロジー」で検索してください。「マイクロチップ・テクノロジー・ジャパン株式会社」のアドレスが見つかるので開いてください。

開発ツールをダウンロードする前に日本語の資料を入手しておきましょう。ダウンロードのメニューから「ツール/ユーザガイド」選択します。

ダウンロードファイルの一覧から「MPLAB X IDE ユーザガイド」, 「PICkit3 プログラマ/デバッグ ユーザガイド」, 「PICkit3 インサーキットデバッグの使い方」をダウンロードしてください。

トップ > ダウンロード > ツール/ユーザガイド

## ダウンロード

- > リリース日順
- > ダウンロード回数順
- > データシート
- > リファレンス マニュアル
- > アプリケーションノート
- ツール/ユーザガイド**
- > 製品パンフレット
- > 技術記事
- > 動画
- > その他

## ツール/ユーザガイド

これらの文献は読者が内容をお読みになる時の参考のために日本語に翻訳されています。翻訳に誤りが存在する場合、Microchip Technology Inc. 及び全ての子会社、関連会社、役員、従業員、代理業者は一切の責任を負いかねます。最新の英語版オリジナル文献を必ずご参照することを強くお勧めします。

ダウンロード検索:  **検索** 件数: 15件 並び替え: リリース日 (新) 順

19件中1~15件目の条件に合致したダウンロードファイルを表示しています。

名称	文書ID	DL回数	▼ リリース日
<a href="#">MPLAB® X IDEユーザガイド</a>	52027A_JP	681	2012/07/19
<a href="#">PICkit™ 3 プログラマ/デバッグ ユーザガイド</a>	51795B_JP	346	2012/03/02
<a href="#">MPLAB REAL ICE™ インサーキット エミュレータの使い方</a>	51997A_JP	74	2011/12/26
<a href="#">MPLAB ICD 3インサーキット デバッグの使い方</a>	52011A_JP	200	2011/12/26
<a href="#">PICkit™ 3インサーキット デバッグの使い方</a>	52010A_JP	257	2011/12/26
<a href="#">不適正なUSB デバイスドライバのアンインストール</a>	51417D_JP	60	2011/03/23
<a href="#">dsPIC® DSC Speex 音声 エンコーディング / デコーディング ライブラリのユーザーガイド</a>	70328A_JP	24	2010/02/19

次に、開発ツールをダウンロードします。ページの下に「製品一覧」という項目があります。その「開発ツール」の欄に「MPLAB X IDE」という項目がありますのでクリックしてください。ブラウザによりますが、別タブで「MPLAB X IDE」のページが開きます(英文サイト)。

## 製品一覧

≡ ページの先頭へ戻る

> PIC®マイクロコントローラ	> 開発ツール	> メモリ	> 安全とセキュリティ
8ビットPIC® MCU 16ビットPIC® MCU & dsPIC® DSC 32ビットPIC® MCU	コンパイラ エミュレータ <b>MPLAB® X IDE</b> プログラマ	シリアルEEPROM シリアルSRAM シリアルフラッシュ パラレルフラッシュ	ホードドライバ 煙&CO検知器
> アンプ&リニア	> インターフェイス	> モータードライバ	> 熱管理
オペアンプ 計装アンプ コンパレータ PGA & SGA	CAN Ethernet 赤外線 LIN シリアルペリフェラル USB	> 電源管理	温度センサ ブラシレスDCファンコントローラ 温度センサ(SMSC) リモートダイオードファンコントローラ(SMSC)
> データコンバータ	> レガシー8051/80C51 MCU	バッテリー管理 チャージポンプ CPU/システム スーパーバイザ LDOLレギュレータ パワー-MOSFETドライバ PWMコントローラ スイッチングレギュレータ 電圧検出器	> タッチ関連製品
A/Dコンバータ DAC&デジタルポテンショメータ 電力計測 電力/電流センサ(SMSC)		> リアルタイムクロック	キー&スライダ タッチスクリーンコントローラ
			> 無線
			赤外線 パワーアンプ RF

**CHECK OUT OUR  
END OF YEAR SALE  
GOING ON RIGHT NOW**

**End of Year Sale**  
Going on now through Jan 4, 2013

**EDN 2012 Hot 100**  
Microchip wins the EDN 2012 Hot 100. Check out the hottest compiler on the market.

**ZenWheels Micro Car**  
Stuff a Stocking with a ZenWheels Car

### MPLAB® X Links



- MPLAB® X IDE Resources:
  - MPLAB® X IDE User's Guide
  - Get Started with MPLAB® X Wiki
  - Getting Started with MPLAB® Presentation
  - MPLAB® X IDE Forum
- Compiler Downloads:
- Related Pages:



### MPLAB® X Integrated Development Environment (IDE)

MPLAB® X IDE is a software program that runs on a PC (Windows®, Mac OS®, Linux®) to develop applications for Microchip microcontrollers and digital signal controllers. It is called an Integrated Development Environment (IDE), because it provides a single integrated "environment" to develop code for embedded microcontrollers.

MPLAB® X Integrated Development Environment brings many changes to the PIC® microcontroller development tool chain. Unlike previous versions of MPLAB® which were developed completely in-house, MPLAB® X is based on the open source NetBeans IDE from Oracle. Taking this path has allowed us to add many frequently requested features very quickly and easily while also providing us with a much more extensible architecture to bring you even more new features in the future.

























#### MPLAB® X IDE Features

- Provides a new Call Graph for navigating complex code
- Supports Multiple Configurations within your projects
- Supports Multiple Versions of the same compiler
- Support for multiple Debug Tools of the same type
- Supports Live Parsing
- Import existing MPLAB® 8 projects and use either IDE for the same source
- Supports hyperlinks for fast navigation to declarations and includes
- Supports Live Code Templates
- Supports the ability to enter File Code Templates with license headers or template code
- MPLAB® X can Track Changes within your own system using local history
- Within MPLAB® X, a user can configure their own Code Format Style








[Read more MPLAB® X Features](#)

Awards:

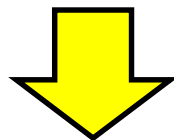
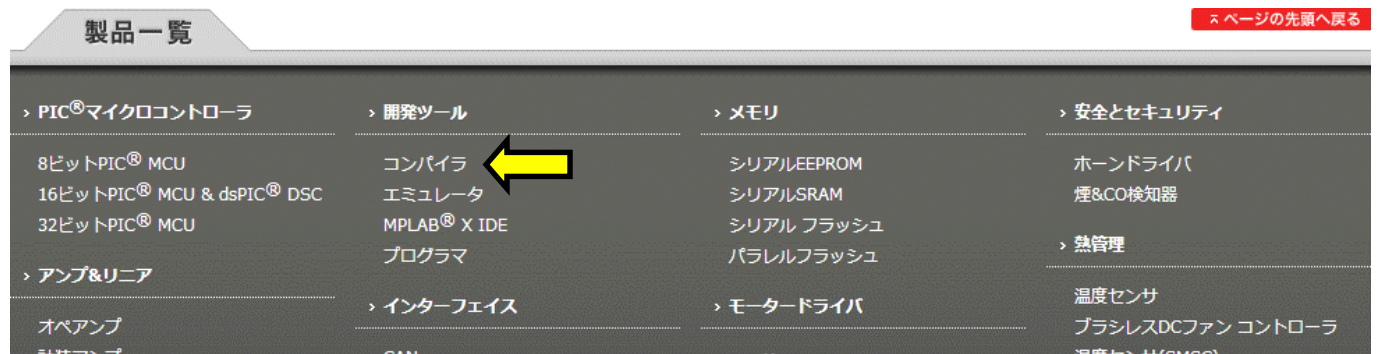
このページに「MPLAB X FREE DOWNLOAD」というボタンがあります。そこをクリックすると、ダウンロードファイルの一覧が開きます。使用するパソコンの OS に対応する「MPLAB X IDE v1.51」と「MPLAB XC8 v1.12」の二つのファイルをダウンロードしてください。(v以降の数字はバージョンによって変化します。このマニュアルのパソコンの OS は Windows を使っています。)

Title	Date Published	Size	D/L
<b>Windows (x86/x64)</b>			
MPLAB® X IDE v1.51 	11/08/2012	312Mb	
MPLAB® X IDE Release Notes / User' Guide v1.51 (supersedes info in installer)	09/11/2012	150KB	
MPLAB® C18 Lite Compiler for PIC18 MCUs v3.40	05/30/2012	73Mb	
MPLAB® XC8 Compiler v1.12 	12/4/2012	168Mb	
MPLAB® XC16 Compiler v1.10	05/30/2012	98Mb	
MPLAB® XC32 Compiler v1.11a	10/04/2012	105Mb	
<b>Linux 32-Bit and Linux 64-Bit (Requires 32-Bit Compatibility Libraries)</b>			
MPLAB® X IDE v1.51 	11/08/2012	260Mb	
MPLAB® X IDE Release Notes / User' Guide v1.51 (supersedes info in installer)	09/11/2012	150KB	
MPLAB® C18 Lite Compiler for PIC18 MCUs v3.40	05/30/2012	70Mb	
MPLAB® XC8 Compiler v1.12 	12/4/2012	172Mb	
MPLAB® XC16 Compiler v1.10	05/30/2012	81Mb	
MPLAB® XC32 Compiler v1.11	10/04/2012	104Mb	
<b>Mac (10.X)</b>			
MPLAB® X IDE v1.51 	11/08/2012	231Mb	
MPLAB® X IDE Release Notes / User' Guide v1.51 (supersedes info in installer)	09/11/2012	150KB	
MPLAB® C18 Lite Compiler for PIC18 MCUs v3.40	05/30/2012	73Mb	
MPLAB® XC8 Compiler v1.12 	12/4/2012	169Mb	
MPLAB® XC16 Compiler v1.10	05/30/2012	75Mb	
MPLAB® XC32 Compiler v1.11	10/04/2012	107Mb	

マイクロチップ・テクノロジー・ジャパンのページから日本語の資料を入手しましたが、英文の資料が最新版になるのでこちらも入手しておきましょう。先ほどの「MPLAB X IDE」のページの「MPLAB X Documentation」というボタンをクリックしてください。ドキュメントの一覧が開きます。「MPLAB X IDE User's Guide」と「Using PICKit3 for MPLAB X IDE」をダウンロードしてください。

Title	Date Published	Size	D/L
MPLAB® X IDE User's Guide 	11/29/2011	515 KB	
Using PICKit™ 3 for MPLAB® X IDE 	11/10/2008	399 KB	
Using MPLAB® ICD 3 In-Circuit Debugger for MPLAB® X IDE	07/01/2011	573 KB	
MPLAB® ICD 3 for MPLAB® X IDE User's Guide	07/01/2011	573 KB	
Using MPLAB® REAL ICE In-Circuit Emulator for MPLAB® X IDE	07/01/2011	573 KB	

「MPLAB XC8」のドキュメントも入手しておきましょう。日本語のページに戻って、ページの下の「製品一覧」という項目の「開発ツール」の欄にある「コンパイラ」という項目をクリックしてください。ブラウザによりますが、別タブで「MPLAB XC Compilers」のページが開きます(英文サイト)。このページの下の部分に「MPLAB XC8 User Guide」があります。クリックしてダウンロードしてください。なお、「MPLAB XC8」の日本語の資料はまだないようです。



## 2. 開発環境のインストール

「MPLAB X IDE」も「MPLAB XC8」も英語のソフトです。説明文は全て英文になります。

最初に「MPLAB X IDE」をインストールします。ダウンロードしたファイルをクリックしてください。あとは画面の指示に従ってインストールします。

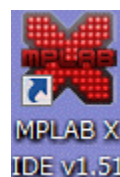
次に「MPLAB XC8」をインストールします。ダウンロードしたファイルをクリックしてください。あとは画面の指示に従ってインストールします。

いずれも途中でいくつかのダイアログが開きますが、基本的にそのままの設定でかまいません。使用許諾に同意するかどうか尋ねるダイアログのときは「agree」を選択してください。また、プロダクトキーの入力が求められることがあります。FREE版で使用するときは未入力でもかまいません(未入力の場合にFREE版になる)。

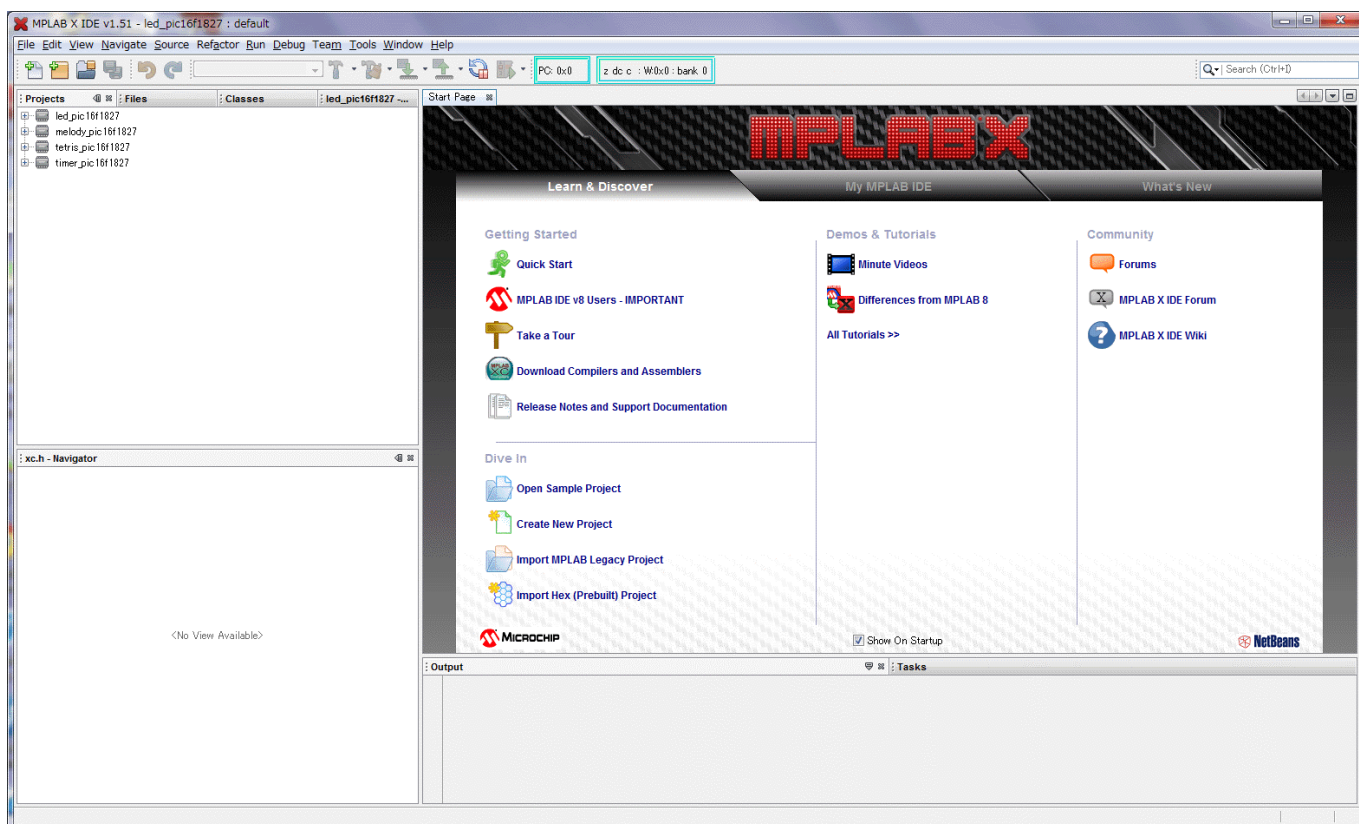
## 3. プロジェクトの新規作成

プロジェクトを新規作成し、ソースリストを入力、ビルドした後、PIC16F1827にダウンロードするところまで見てみましょう。ここで作成するプロジェクト名は「00\_tutorial」です。

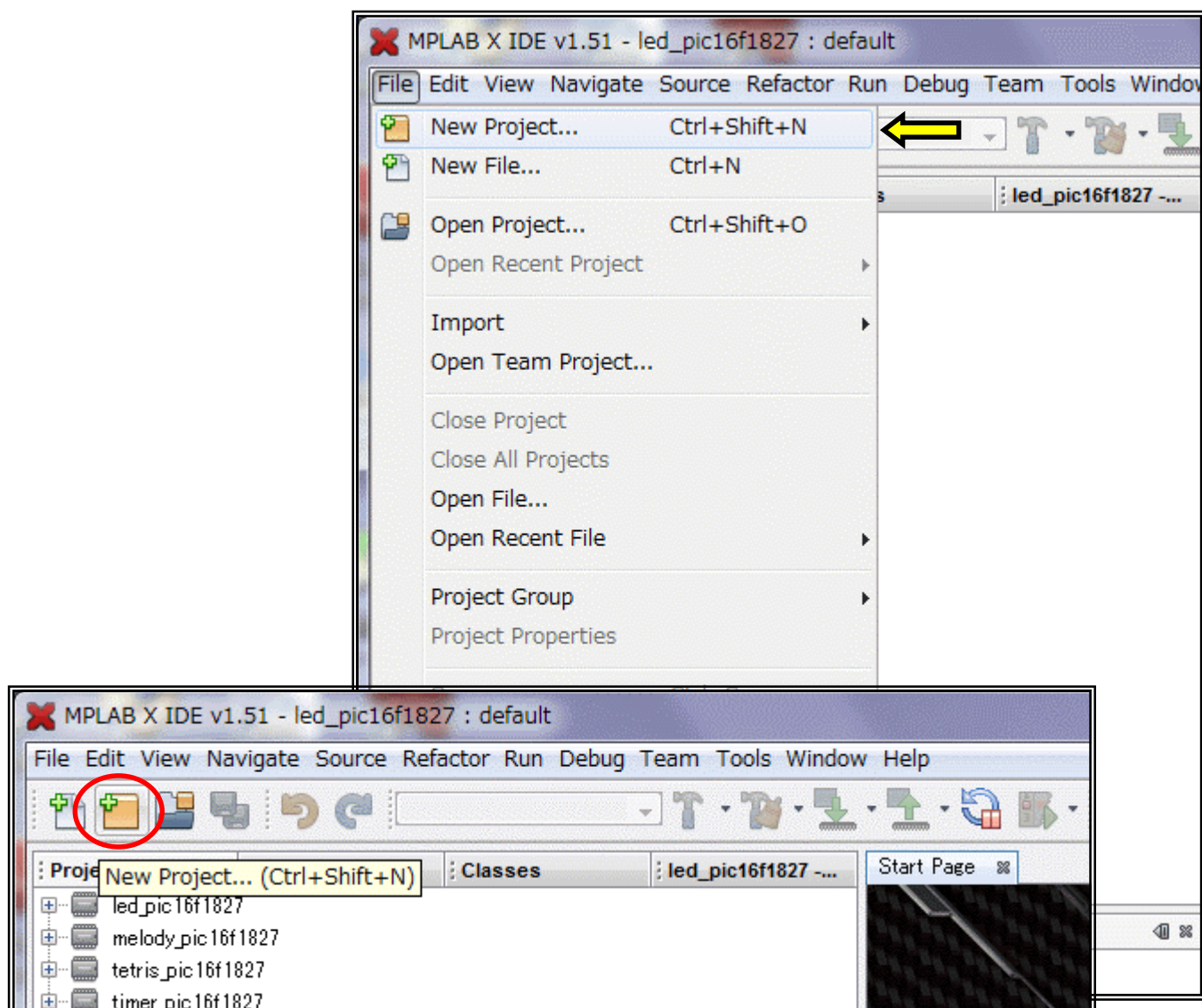
デスクトップに「MPLAB X IDE」のショートカットアイコンが作られていますのでクリックして起動してください。(アイコンの数字はバージョンによって変更されます。アイコンが作られていないときはスタートメニューから起動してください。)



しばらくすると、次のようなウィンドウが開きます。ここからスタートしていきます。(画面をキャプチャしたこのパソコンはすでにプロジェクトをいくつか作っていますので、作成済みのプロジェクトがプロジェクトウィンドウに表示されています。)

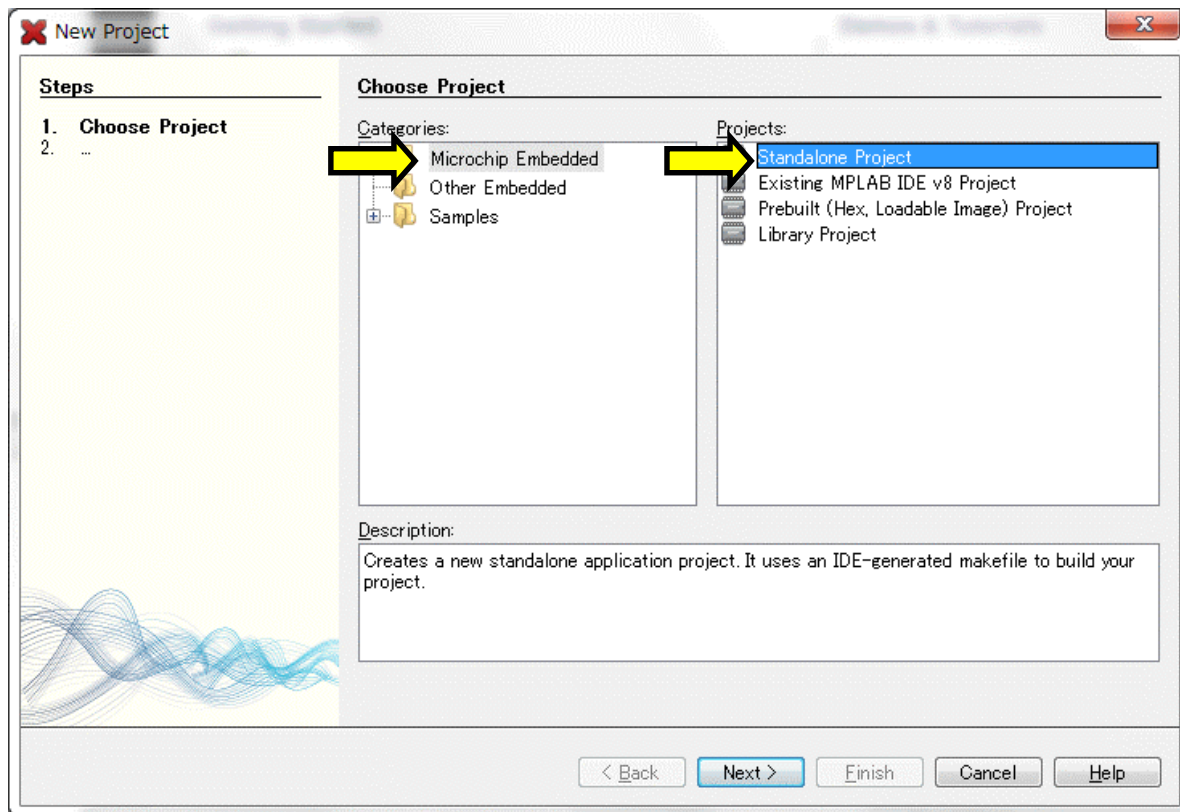


メニューから[File]→[New Project]と選択するか、「New Project」ボタンをクリックしてください。

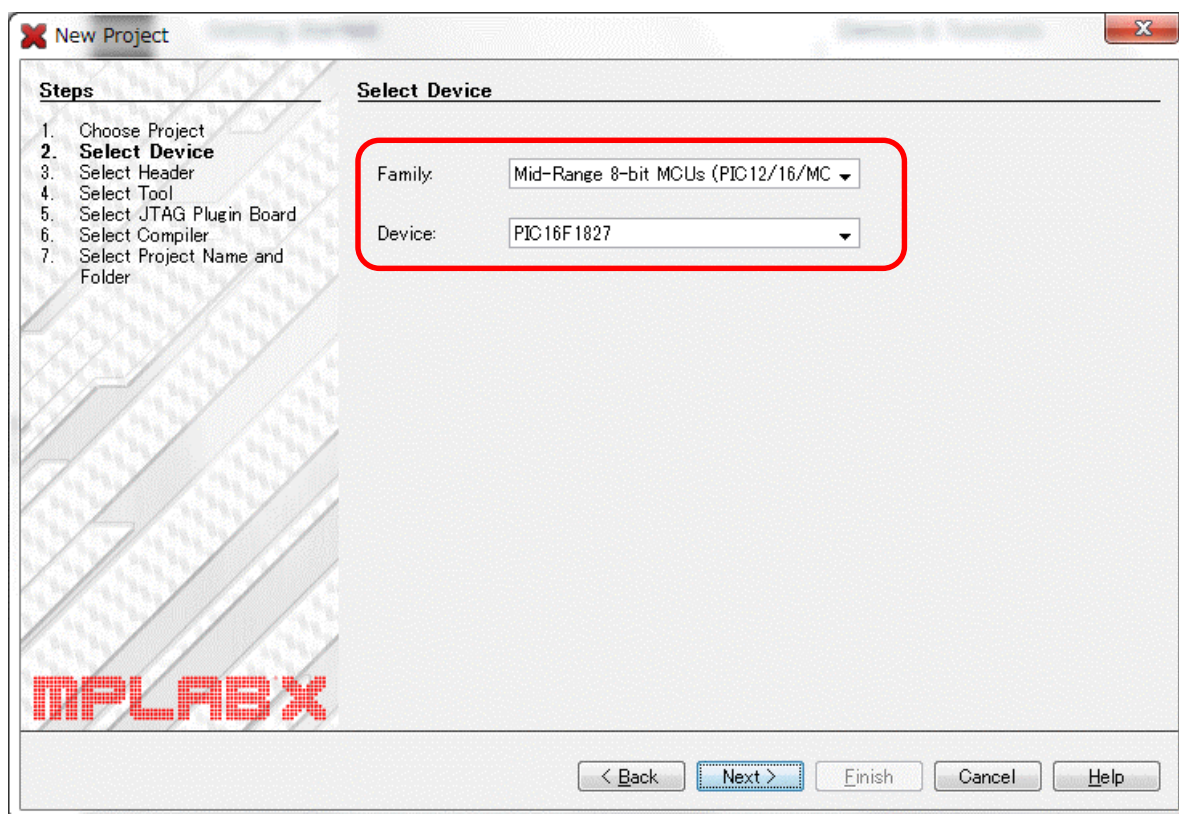




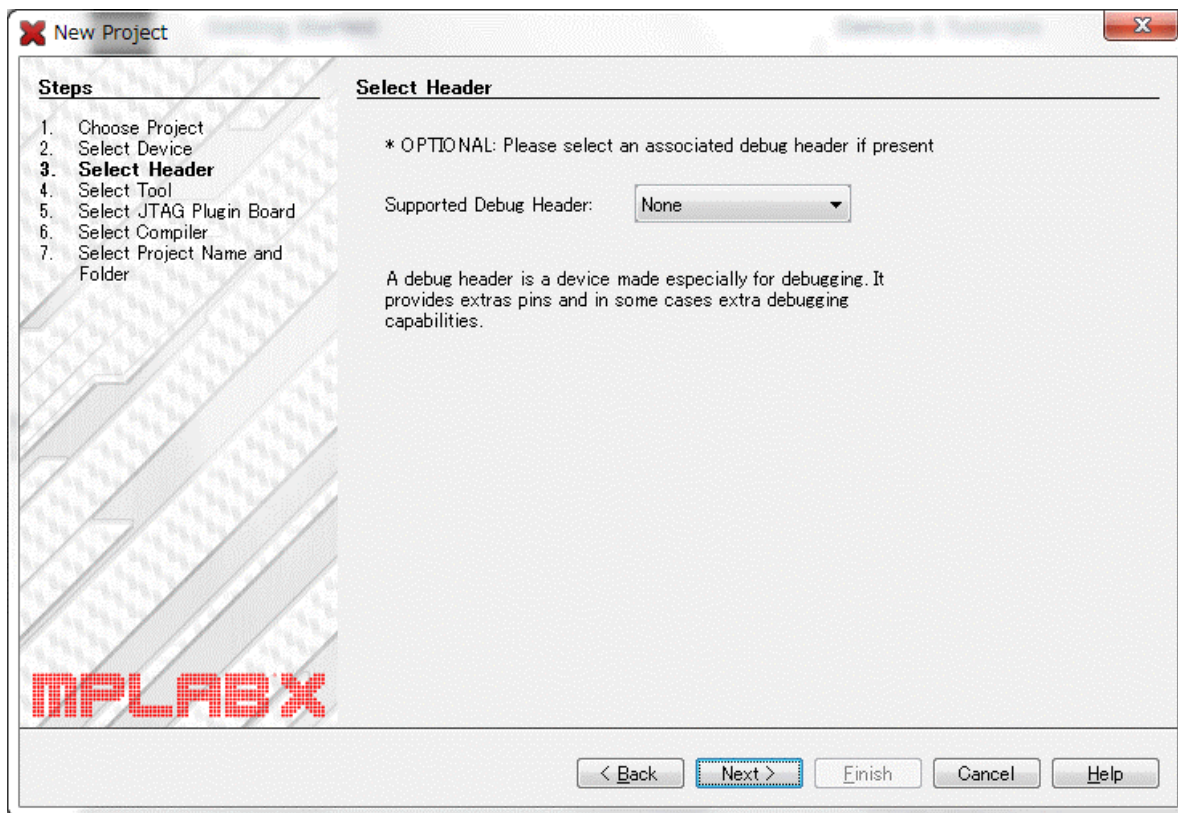
「1. Choose Project」ダイアログで、「Categories:」は「Microchip Embedded」、「Projects:」は「Standard Project」を選択し、「Next>」をクリックします。



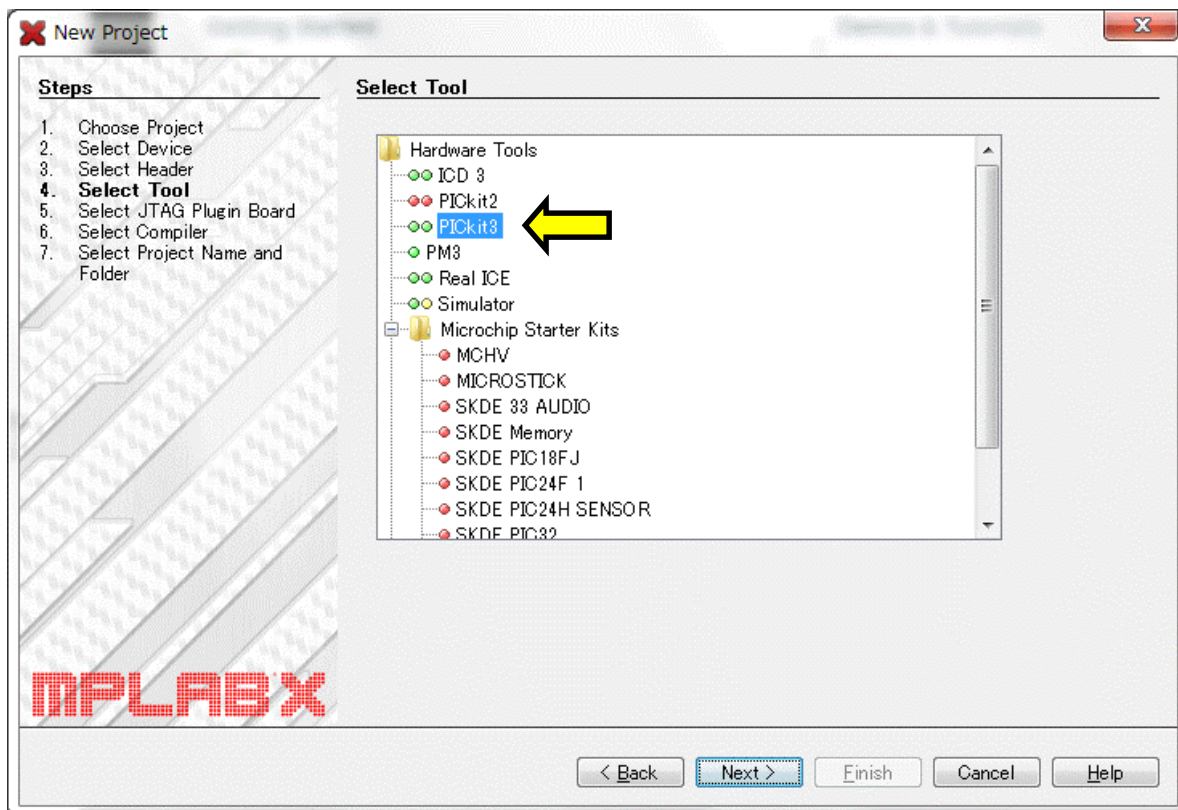
「2. Select Device」ダイアログで、「Family:」は「Mid-Range 8-bit MCUs (PIC12/16/MCP)」、「Device:」は「PIC16F1827」を選択し、「Next>」をクリックします。



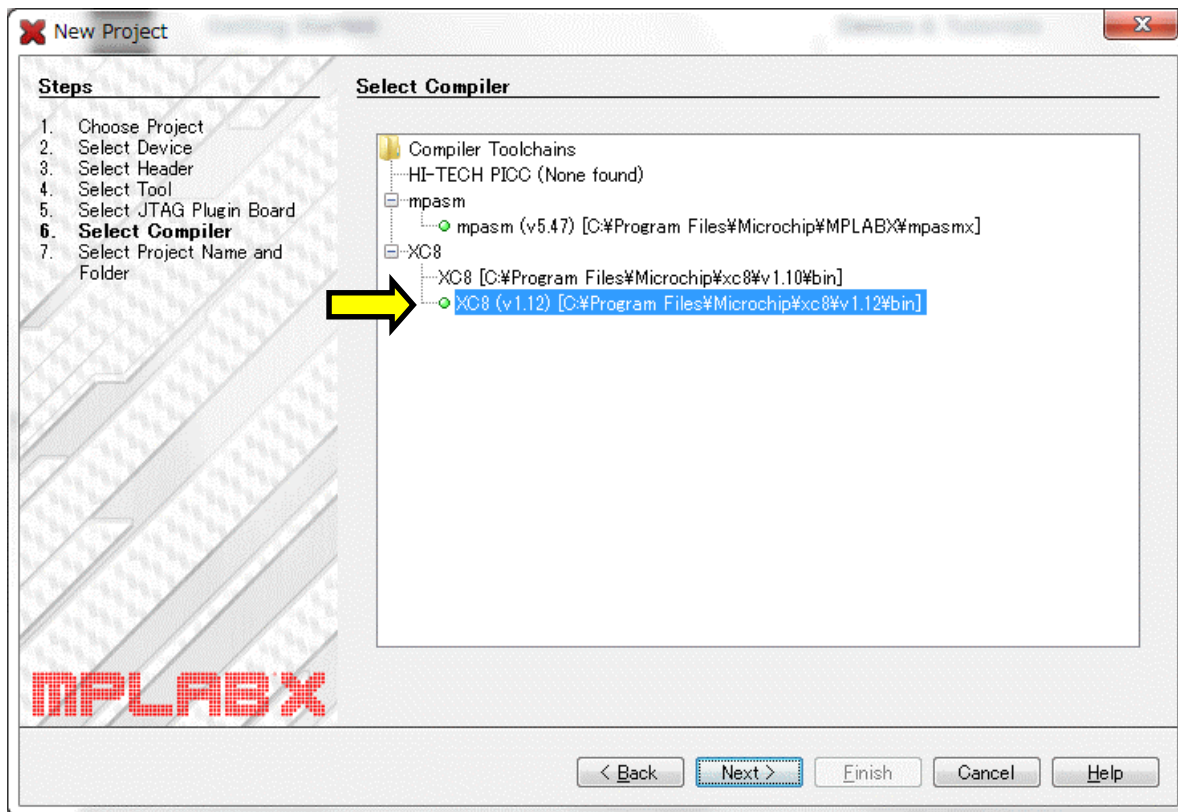
「3. Select Header」ダイアログは変更しません。「Next>」をクリックします。



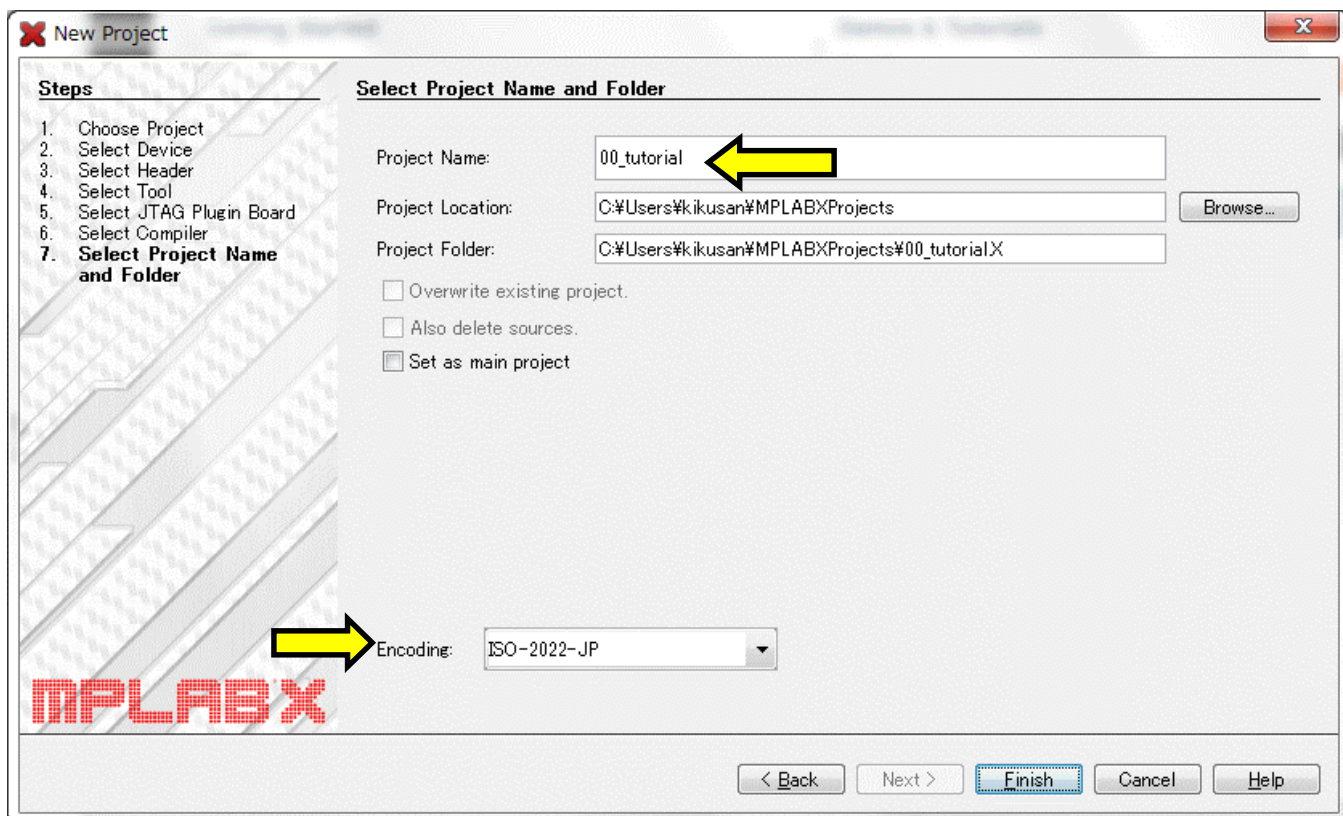
「4. Select Tool」ダイアログで「PICkit3」を選択し、「Next>」をクリックします。



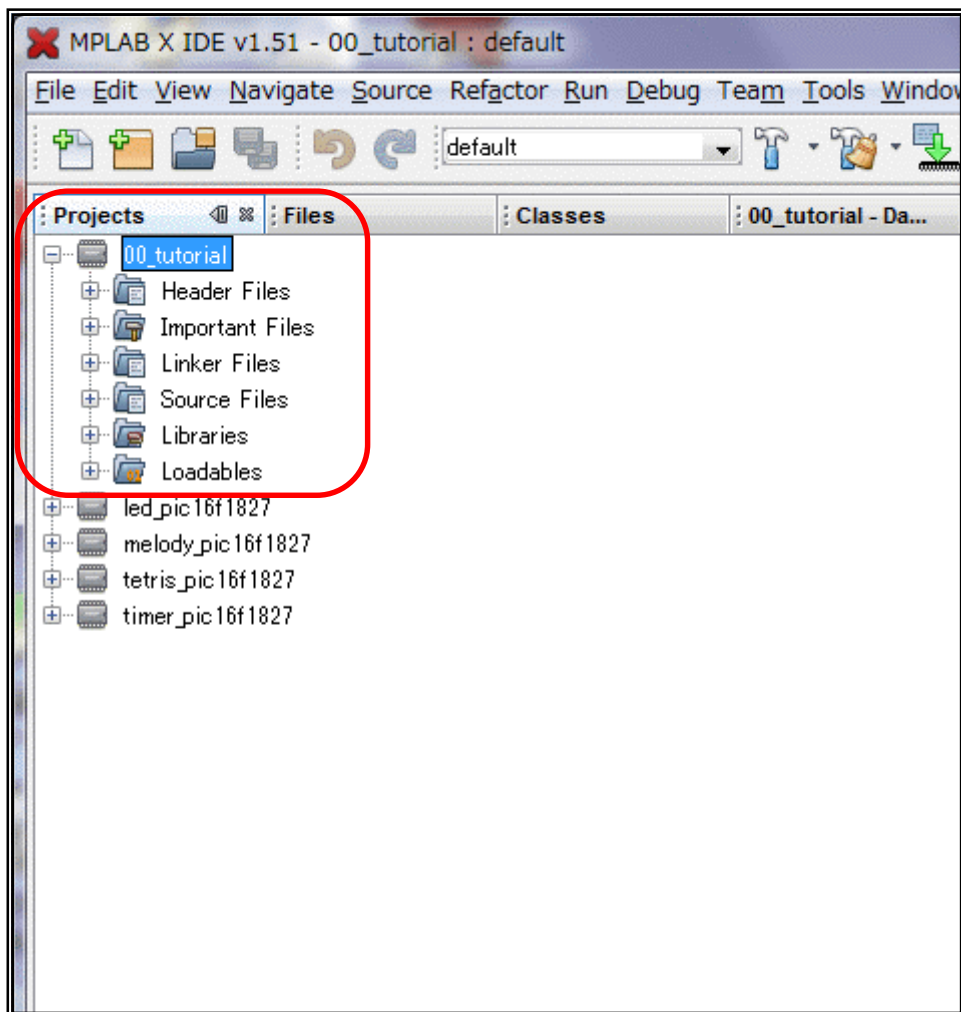
「6. Select Compiler」ダイアログで「XC8(v1.12)」を選択し、「Next>」をクリックします。(数字はバージョンによって変更されます。)



「7. Select Project Name and Folder」ダイアログで、「Project Name:」に「00\_tutorial」を入力し、「Encoding:」を「ISO-2022-JP」か「ISO-2022-JP-2」に変更します(ソースリストに日本語のコメントを入力できるようになります)。最後に「Finish」をクリックします。



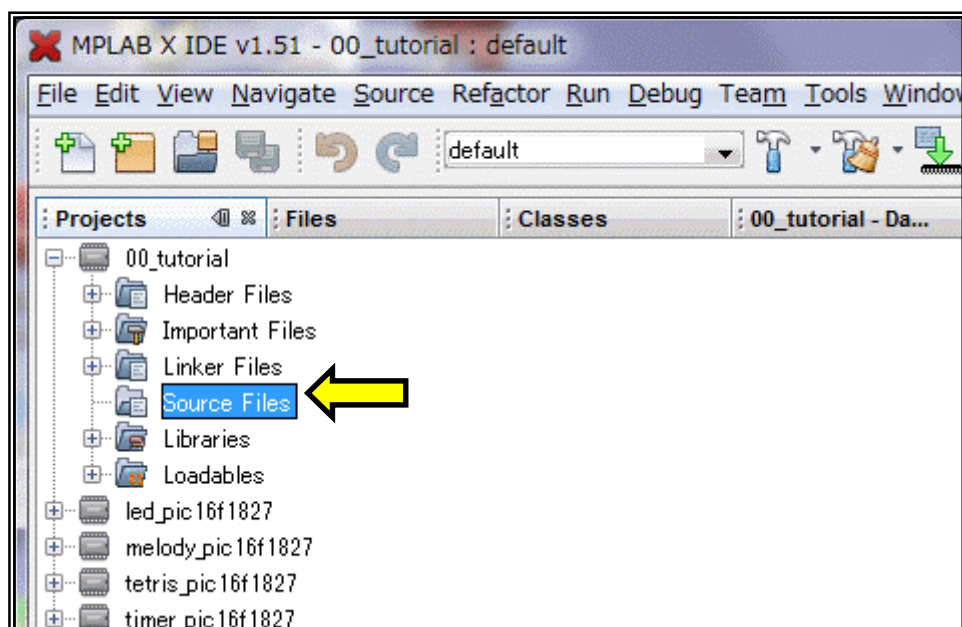
「Projects」ウィンドウに「00\_tutorial」が追加されます。



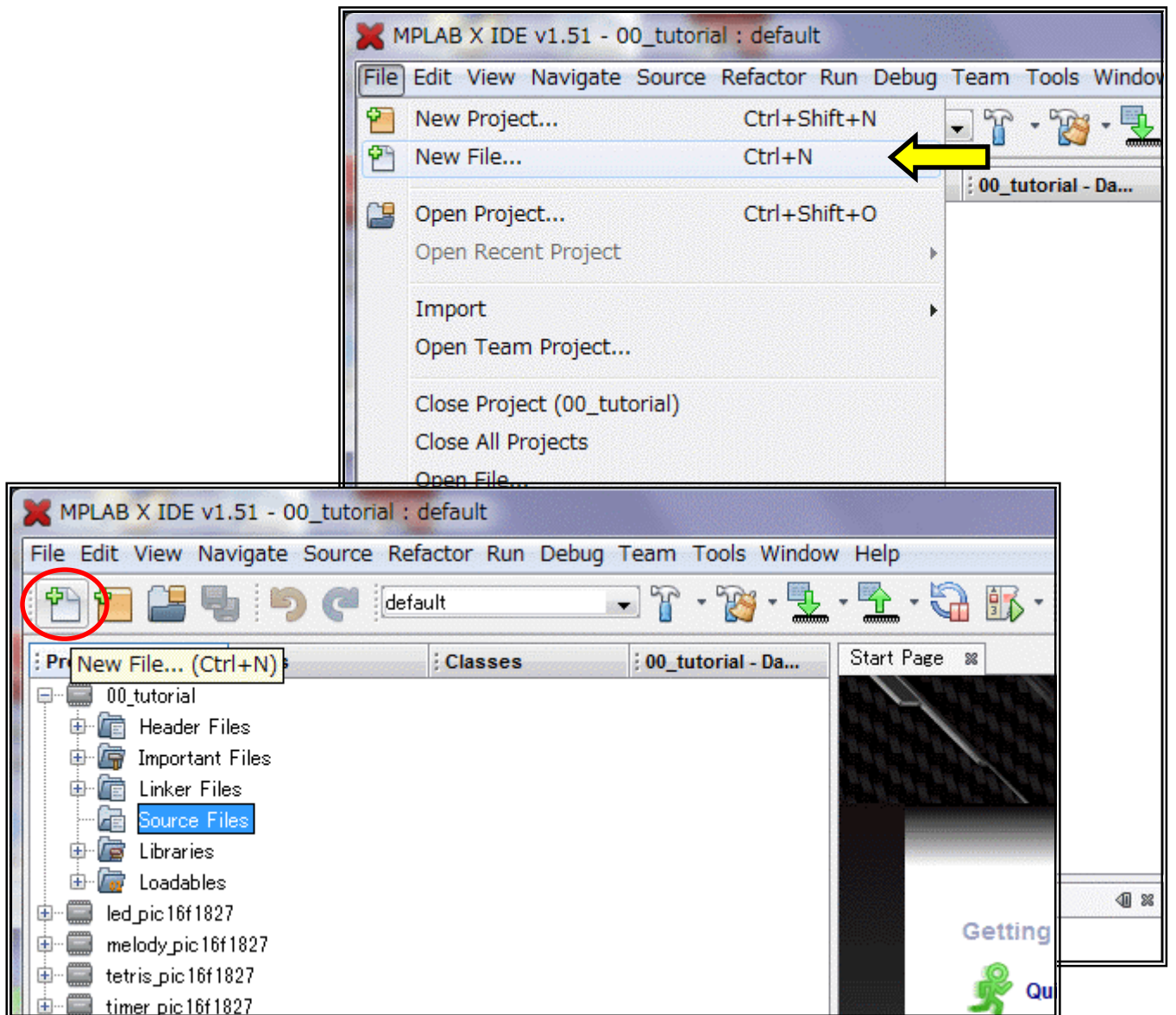
#### 4. ソースリストの入力

続いてソースリストを入力します。ここで作成するソースファイル名は「00\_tutorial.c」です。

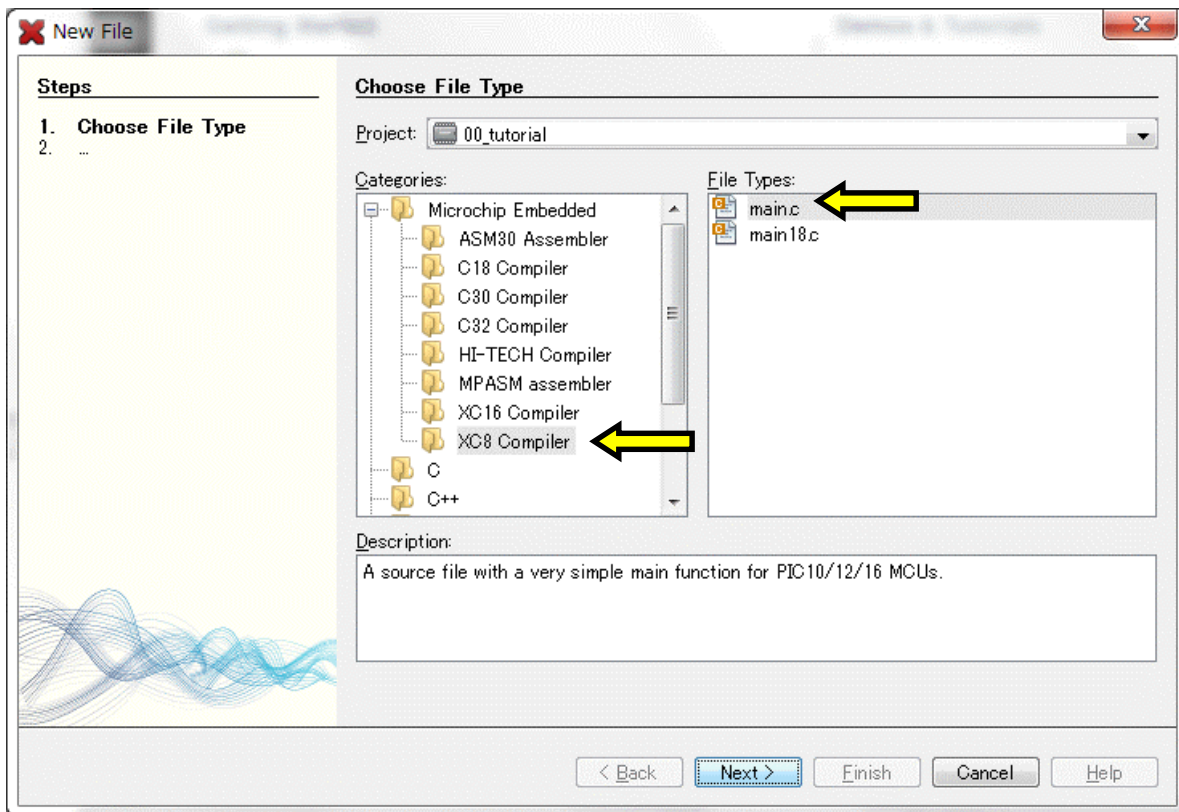
まず、プロジェクトの「Source Files」をクリックします。



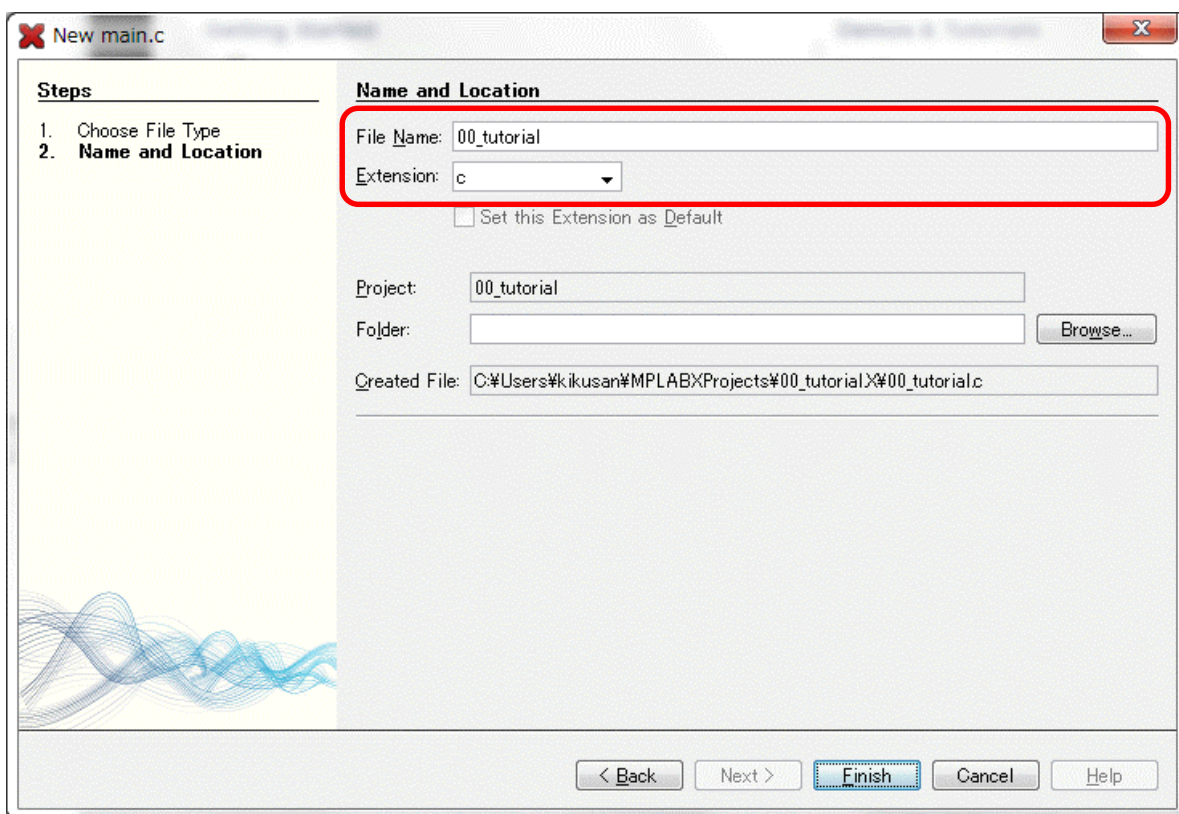
次に、メニューから[File]→[New File]と選択するか、「New File」ボタンをクリックしてください。



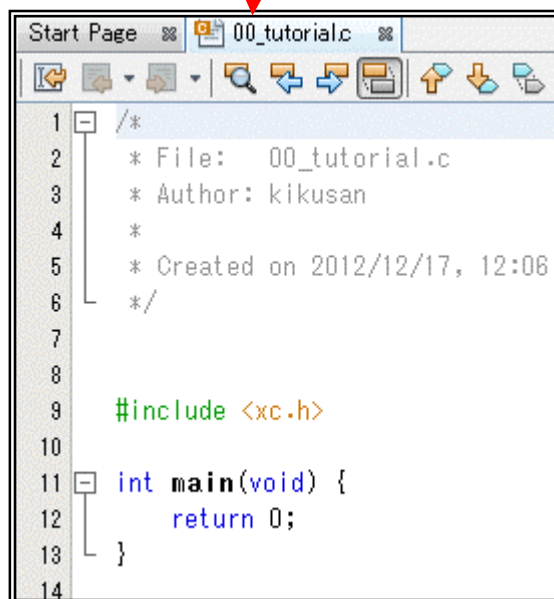
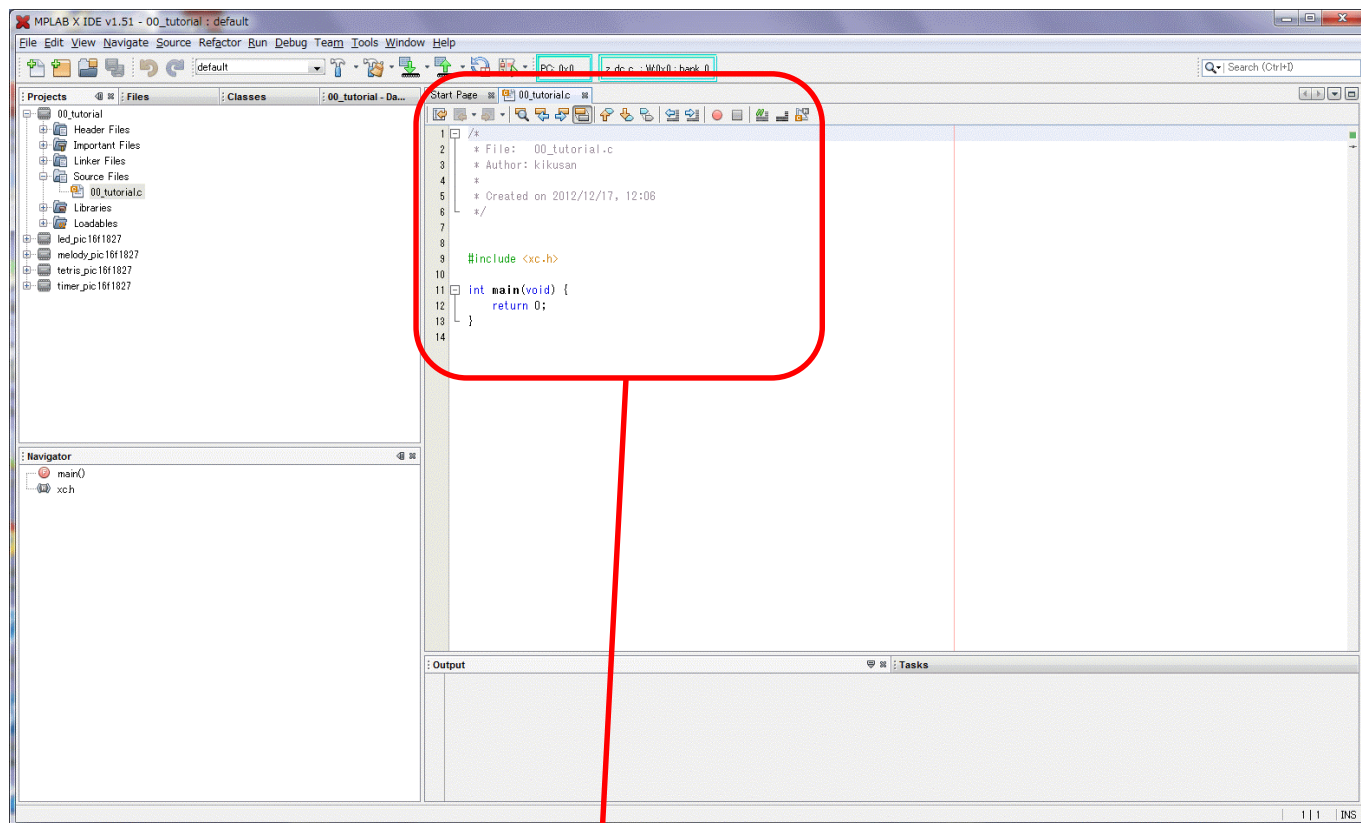
「1. Choose File Type」ダイアログで、「Categories:」は「XC8 Compiler」, 「File Types:」は「main.c」を選択し、「Next>」をクリックします。



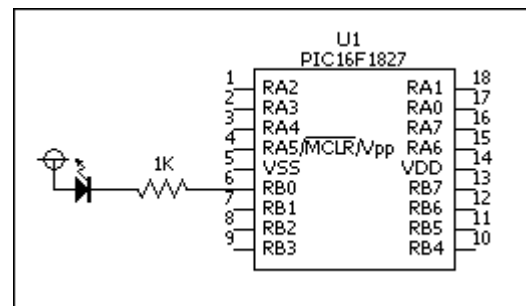
「2. Name and Location」ダイアログで、「File Name:」に「00\_tutorial」を入力, 「Extension:」は「c」を選択します。最後に「Finish」をクリックします。



すると、ソースファイルが自動生成されます。



このソースファイルを変更していきます。今回のプログラムはRB0につないだLEDを点滅するというものです。回路は右のようになります。(この回路は説明のために考えたもので、プリント基板には実装されていません。実際にLEDを光らせたい場合はユニバーサルエリアに組み立ててください。巻末の回路図参照。)



次のようにソースファイルを変更してください。今回は手順の説明なので、プログラムの詳細はコメントを見てください。もちろん、コメントは入力しなくても動作します。

```

1  /*
2  * File:   00_tutorial.c
3  * Author: kikusan
4  *
5  * Created on 2012/12/17, 12:06
6  */
7
8  // PIC16F1827 コンフィグレーションビットの設定
9  #pragma config FOSC = INTOSC, WDTE = OFF, PWRTE = ON, MCLRE = OFF, CP = OFF
10 #pragma config CPD = OFF, BOREN = ON, CLKOUTEN = OFF, IESO = ON, FCMEN = ON
11 #pragma config WRT = OFF, PLLEN = ON, STVREN = ON, BORV = LO, LVP = OFF
12
13 // インクルードファイル
14 #include <xc.h>
15
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable, 8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RB0 = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RB0 = 1; // LEDオフ
33             for(i=0; i<100000; i++){ // ウェイト
34                 }
35         }
36     }

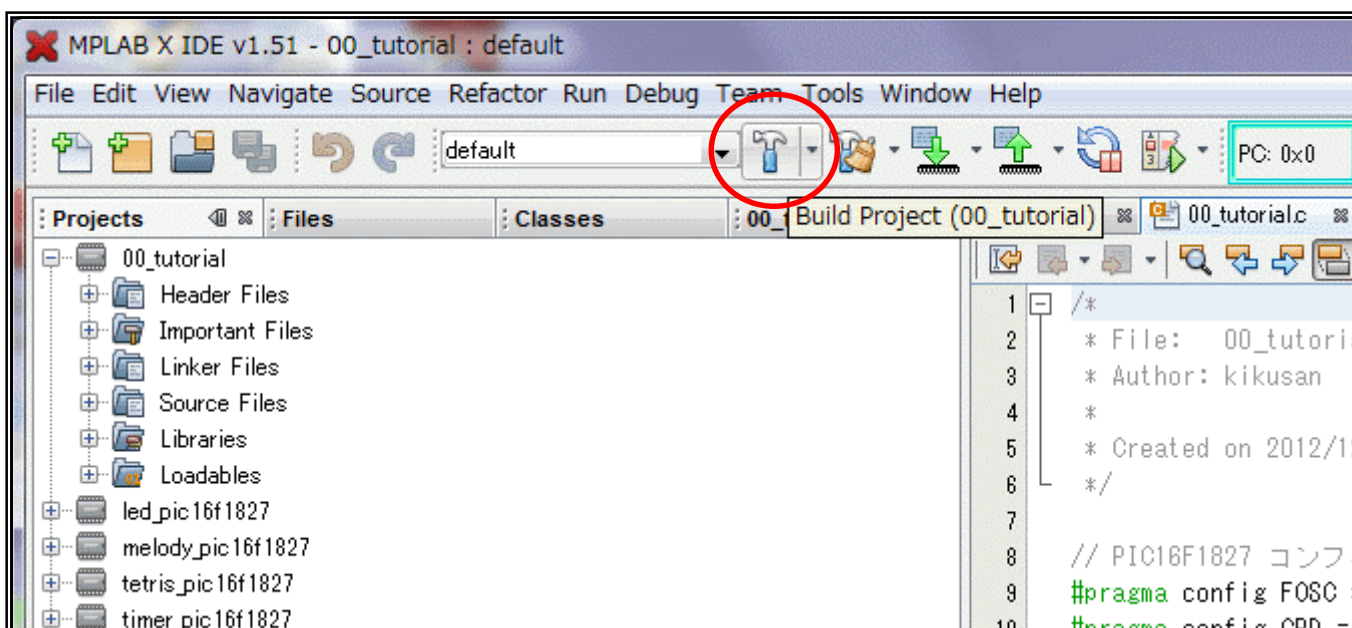
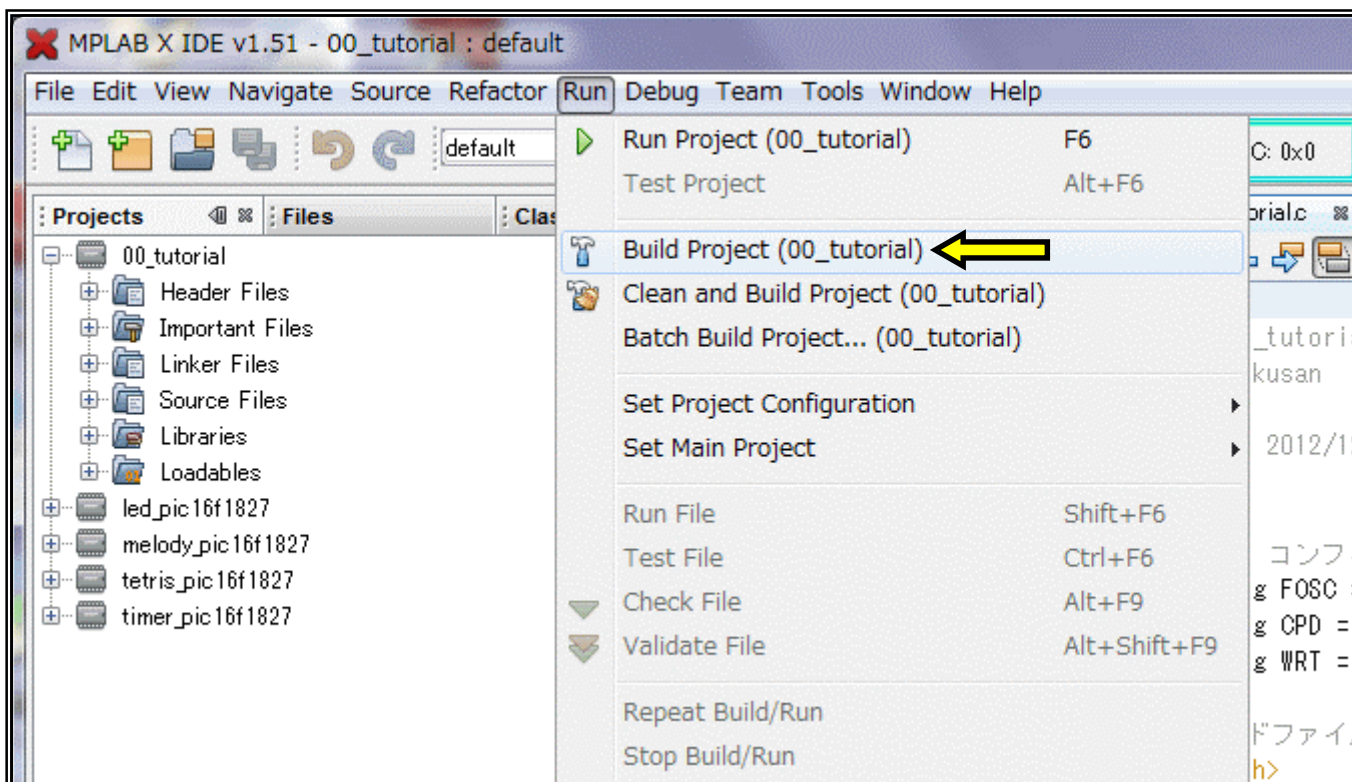
```

入力の途中でも、ときどきキーボードから「Ctrl」+「S」(ファイルの上書き保存)でセーブする習慣をつけましょう。入力が終わったときにもセーブします。



## 5. ビルド

ソースリストの入力が終わったらビルドします。メニューから[Run]→[Build Project]と選択するか、「Build Project」ボタンをクリックしてください。



ビルドの状態は「Output」ウィンドウに表示されます。ビルドが成功すると次のように表示されます。

```
Output - 00_tutorial (Build, Load) | Tasks
make -f nbproject/Makefile-default.mk SUBPROJECTS= .build-conf
make[1]: Entering directory `C:/Users/kikusan/MPLABXProjects/00_tutorial.X'
make -f nbproject/Makefile-default.mk dist/default/production/00_tutorial.X.production.hex
make[2]: Entering directory `C:/Users/kikusan/MPLABXProjects/00_tutorial.X'
"C:\Program Files\Microchip\xc8\v1.12\bin\xc8.exe" --pass1 --chip=16F1827 -O -G --asmlist --double=24 --float=24 --opt=default,+asm,-asmfile,+speed,-space,-debug,9 --addr
Microchip MPLAB XC8 C Compiler V1.12
Copyright (C) 2012 Microchip Technology Inc.
License type: Node Configuration

:: warning: Omniscient Code Generation not available in Free mode

Memory Summary:
Program space      used   87h ( 135) of 1000h words ( 3.3%)
Data space        used    6h (  6) of 180h bytes ( 1.6%)
EEPROM space      used    0h (  0) of 100h bytes ( 0.0%)
Configuration bits used    2h (  2) of   2h words (100.0%)
ID Location space used    0h (  0) of   4h bytes ( 0.0%)

Running this compiler in PRO mode, with Omniscient Code Generation enabled,
produces code which is typically 40% smaller than in Free mode.
See http://microchip.com for more information.

make[2]: Leaving directory `C:/Users/kikusan/MPLABXProjects/00_tutorial.X'
make[1]: Leaving directory `C:/Users/kikusan/MPLABXProjects/00_tutorial.X'

BUILD SUCCESSFUL (total time: 10s)
Loading code from C:/Users/kikusan/MPLABXProjects/00_tutorial.X/dist/default/production/00_tutorial.X.production.hex...
Loading symbols from C:/Users/kikusan/MPLABXProjects/00_tutorial.X/dist/default/production/00_tutorial.X.production.cof...
Loading completed
```

BUILD SUCCESSFUL (total time: 10s)  
Loading code from C:/Users/kikusan/MPLABXProjects/  
Loading symbols from C:/Users/kikusan/MPLABXProjec  
Loading completed

エラーが出るとすれば、今回はタイプミスがほとんどだと思います。エラーが表示されたら、エラー箇所を探して修正してください。

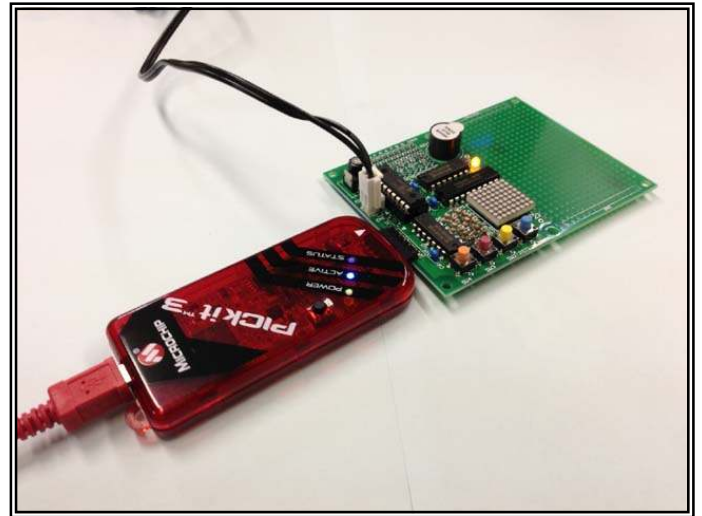
## 6. PICKit3 を使って PIC16F1827 にダウンロード

ビルドが成功したら PIC16F1827 にプログラムをダウンロードしましょう。

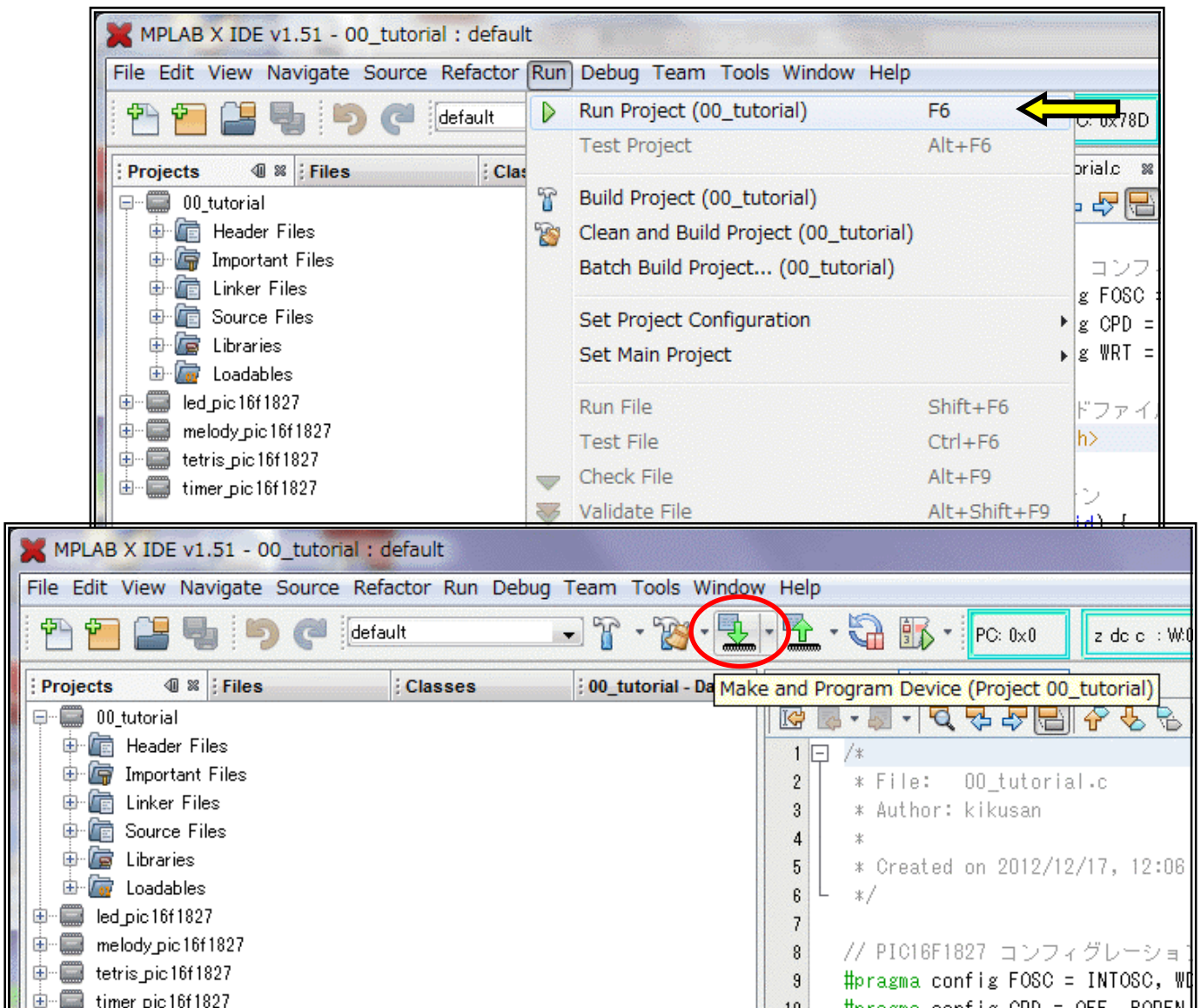
まず、「PICKit3」とパソコンを、「PICKit3」に付属している USB ケーブルで接続します。「PICKit3」のドライバは「MPLAB X IDE」のインストールの際にパソコンにコピーされます。ドライバのインストールが必要な場合は、パソコンの画面の指示に従ってインストールしてください。

次に tk-PIC1827 の CN2 と「PICKit3」を接続します。

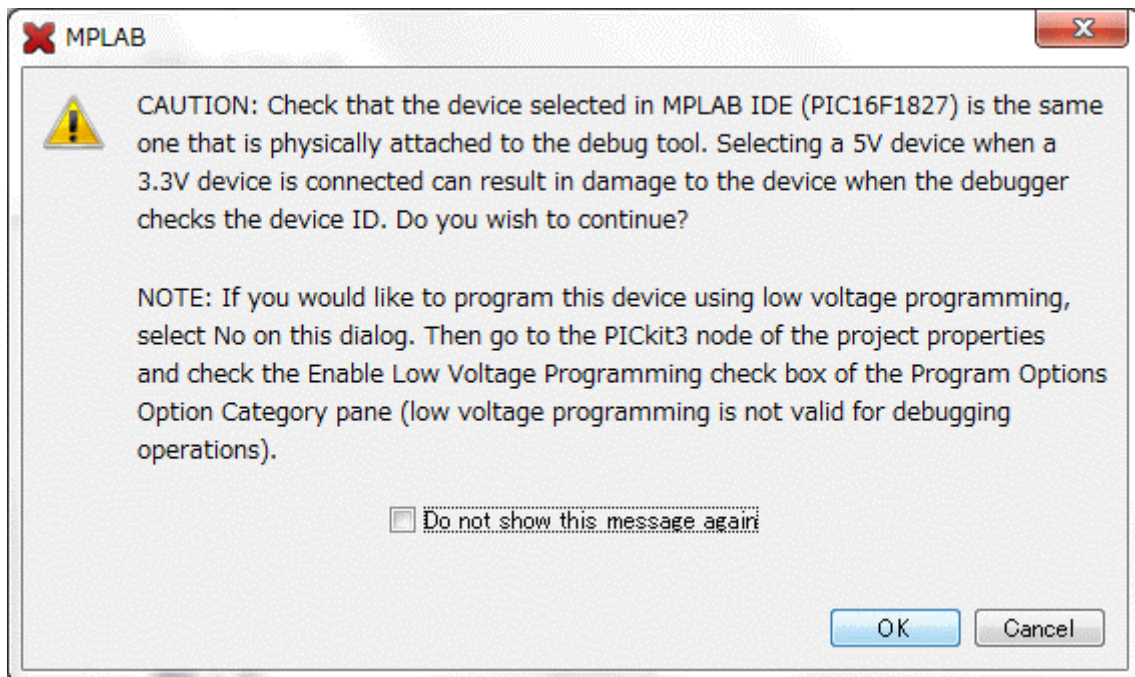
最後に tk-PIC1827 の電源を入れてください。



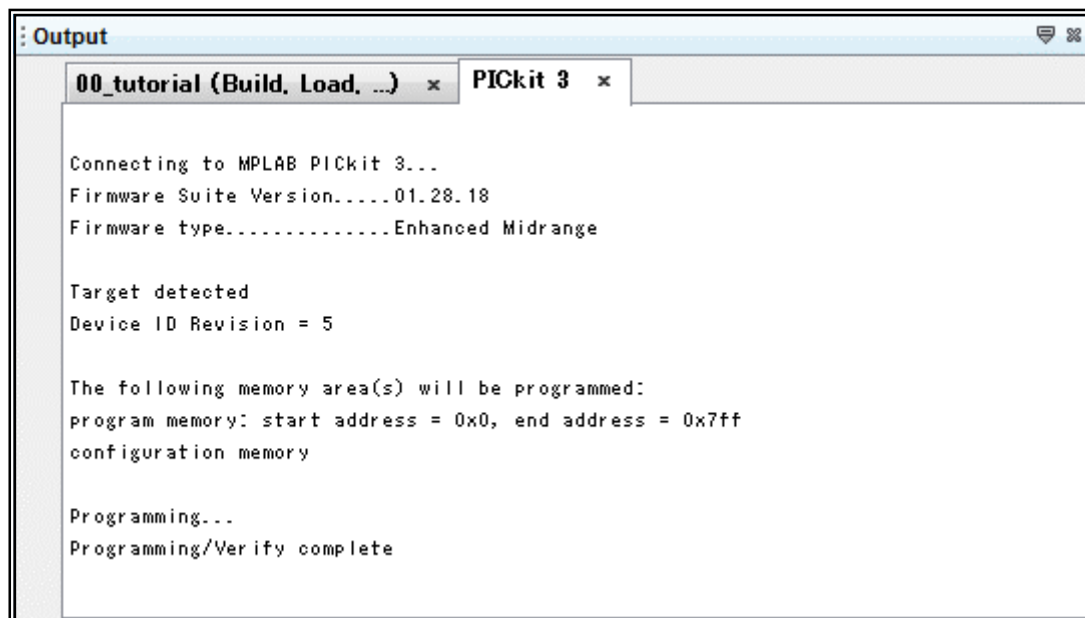
接続が終わったら、メニューから[Run]→[Build Project]と選択するか、「Make and Program Device」ボタンをクリックしてください。必要な場合は自動的にビルドされ、ダウンロードが始まります。



途中で、次のような警告のダイアログが開くことがあります。 「OK」をクリックしてください。



「PICkit3」の情報は「Output」ウィンドウの「PICkit3」タブに表示されます。右のように表示されれば PIC16F1827 へのダウンロードは成功です。



ダウンロードが終わるとプログラムがスタートします。回路図どおり RB0 に LED を接続していれば、LED が点滅します。あとは「PICkit3」を外した状態でも、電源を入れればダウンロードしたプログラムがスタートします。

#### ■ 「PICkit3」接続時の注意事項 ■

「PICkit3」は PIC16F1827 の RA5 (MCLR/Vpp), RB6 (ICSPCLK), RB7 (ICSPDAT) を使って通信します。このマニュアルで取り上げたプログラム例ではこれらのポートを使用しないので問題ありませんが、これらのポートを使う回路を追加した場合ダウンロードに支障が出る可能性があります。その際は、ダウンロードの際に回路を切り離せるようにするとよいでしょう。

なお、回路設計時の注意事項の詳細は、マイクロチップテクノロジーの資料か、このマニュアルの付録をご覧ください。

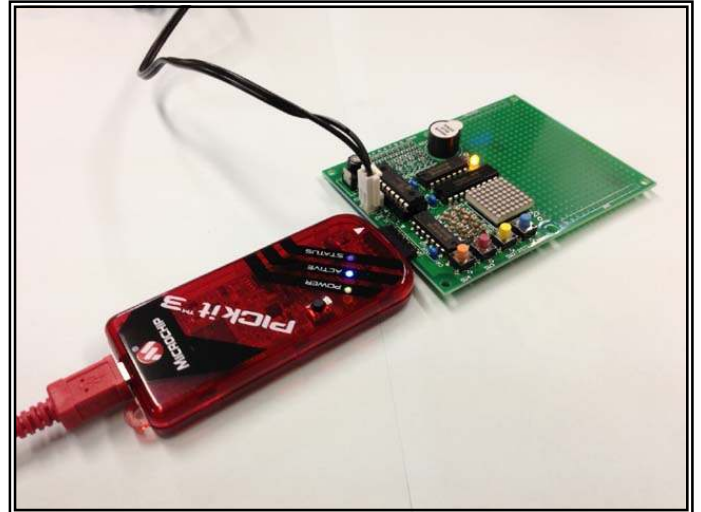
## 7. PICkit3 を使ったデバッグ

「PICkit3」はプログラムのダウンロードだけでなく、プログラムを実機で動かしながらデバッグする、デバッガとして使うことができます。

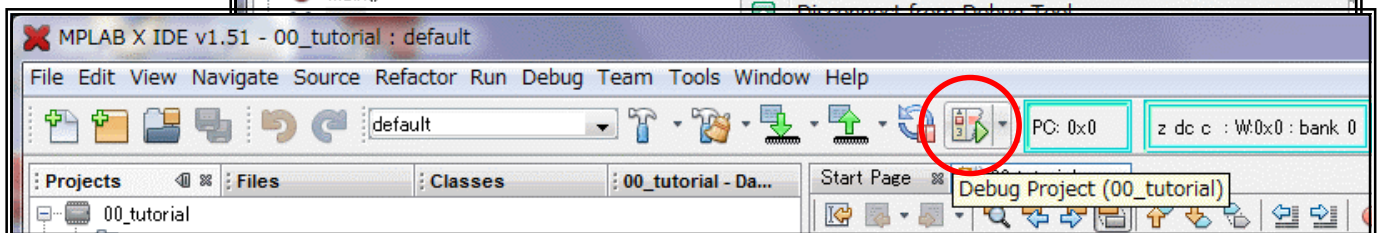
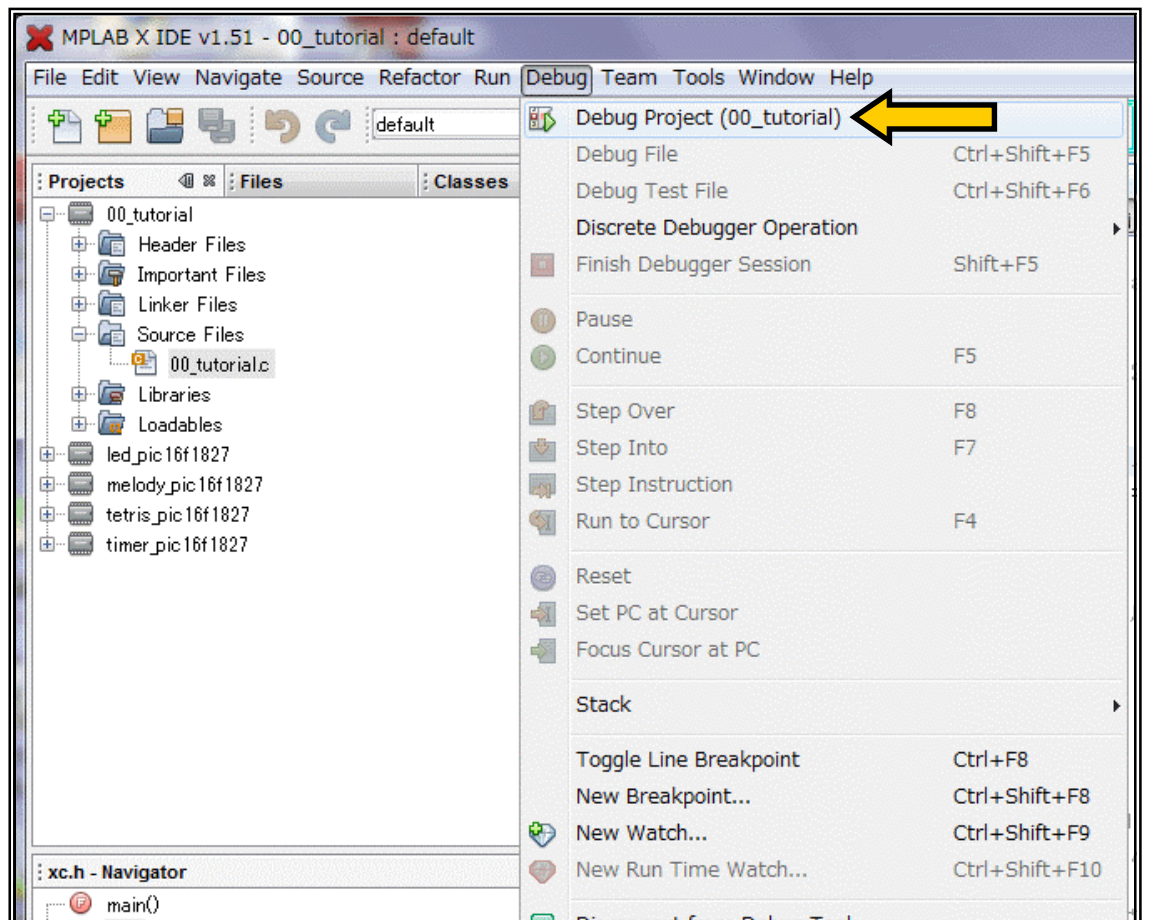
まず、「PICkit3」とパソコンを、「PICkit3」に付属している USB ケーブルで接続します。「PICkit3」のドライバは「MPLAB X IDE」のインストールの際にパソコンにコピーされます。ドライバのインストールが必要な場合は、パソコンの画面の指示に従ってインストールしてください。

次に tk-PIC1827 の CN2 と「PICkit3」を接続します。

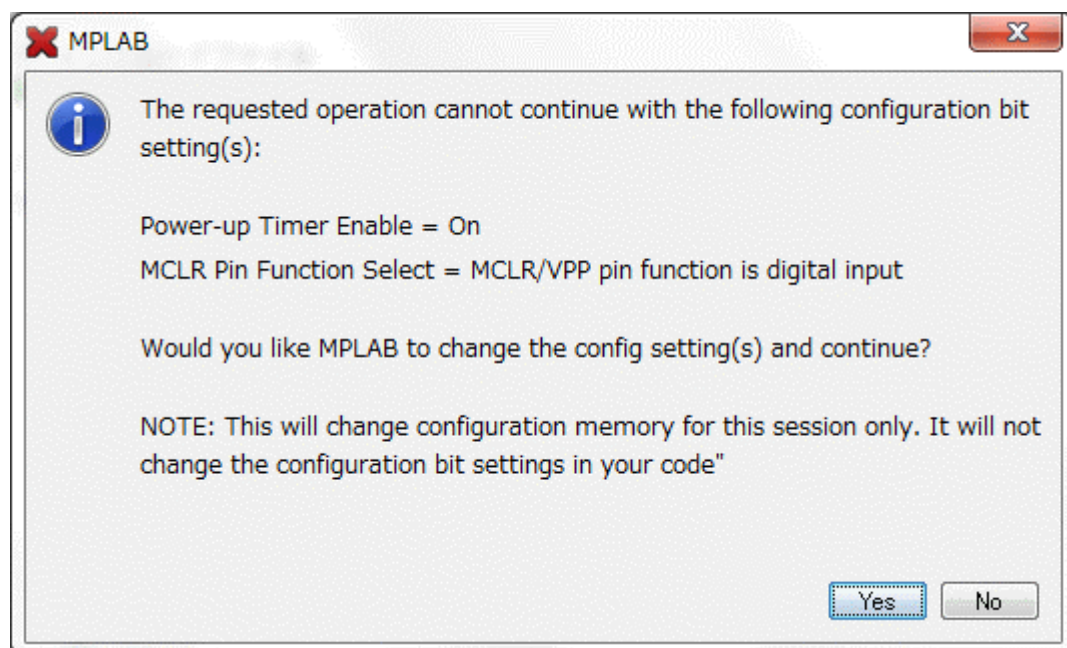
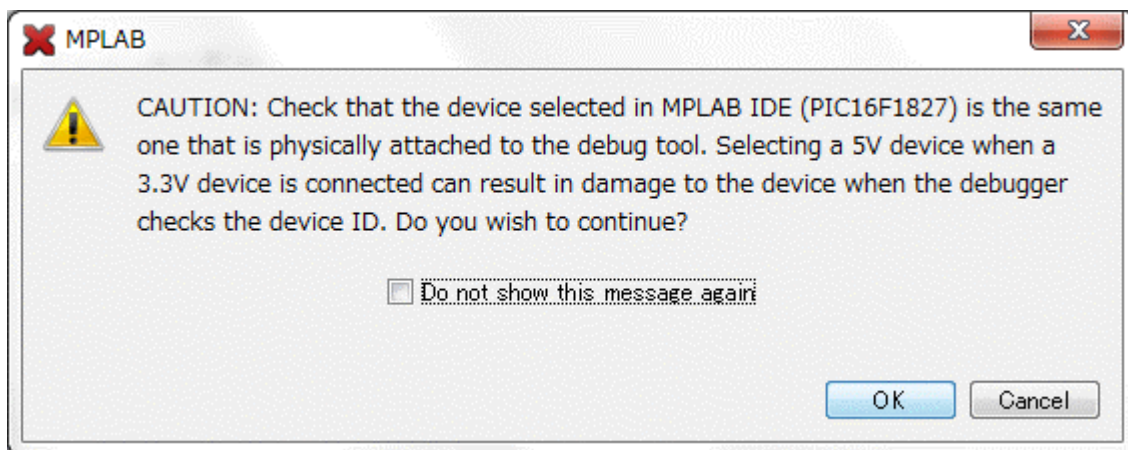
最後に tk-PIC1827 の電源を入れてください。



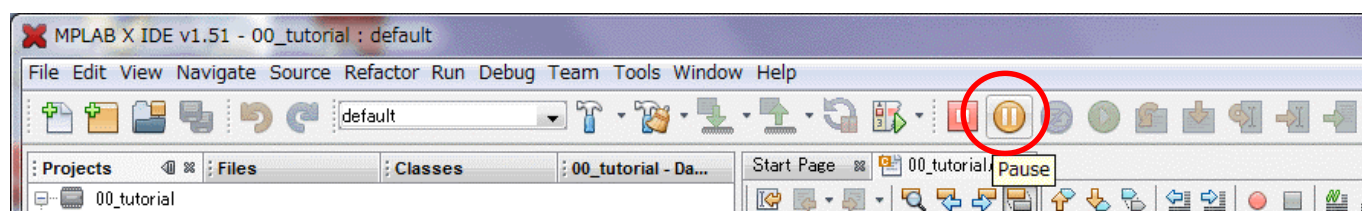
接続が終わったら、メニューから [Debug] → [Debug Project] と選択するか、「Debug Project」ボタンをクリックしてください。必要な場合は自動的にビルドされ、ダウンロードが始まります。



途中で、次のような警告や確認のダイアログが開くことがあります。 「OK」, 「Yes」をクリックしてください。

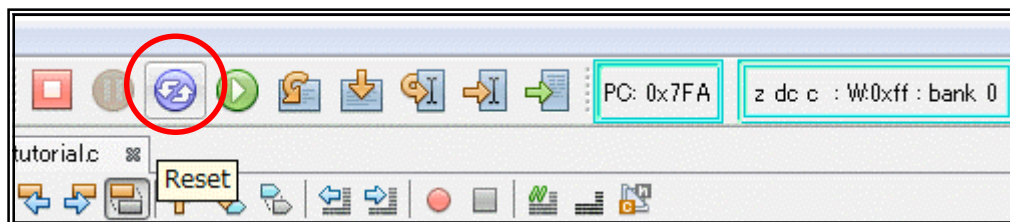


ダウンロードが終わると自動的にプログラムがスタートしますので、まずはプログラムを一時停止します。「Pause」ボタンをクリックします。

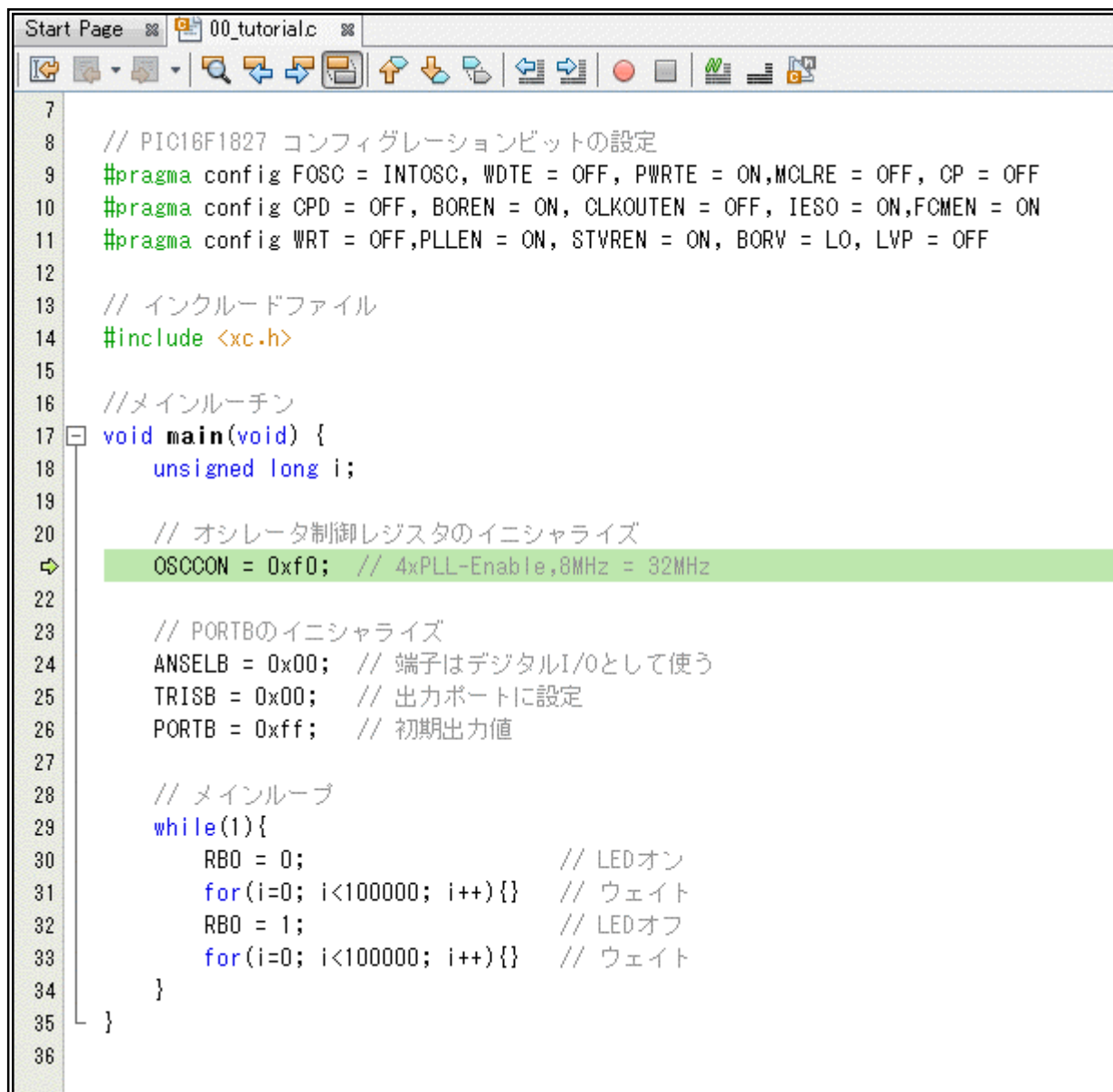


## ■ リセット

プログラムをスタートアドレスから動かしてみよう。「Reset」ボタンをクリックしてください。



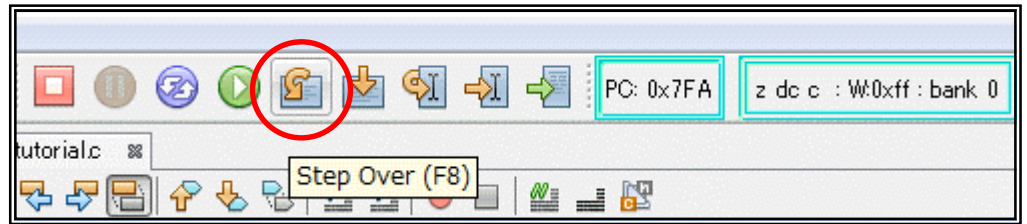
そうすると、スタートアドレスの行の色が変わり、左端に「→」が表示されます(下図の 21 行)。これが、現在のプログラムカウンタが示している行をあらわします。



```
Start Page 00_tutorial.c
7
8 // PIC16F1827 コンフィグレーションビットの設定
9 #pragma config FOSC = INTOSC, WDTE = OFF, PWRTE = ON, MCLRE = OFF, CP = OFF
10 #pragma config CPD = OFF, BOREN = ON, CLKOUTEN = OFF, IESO = ON, FCMEN = ON
11 #pragma config WRT = OFF, PLLEN = ON, STVREN = ON, BORV = LO, LVP = OFF
12
13 // インクルードファイル
14 #include <xc.h>
15
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable, 8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RBO = 1; // LEDオフ
33             for(i=0; i<100000; i++){ // ウェイト
34                 }
35     }
36
```

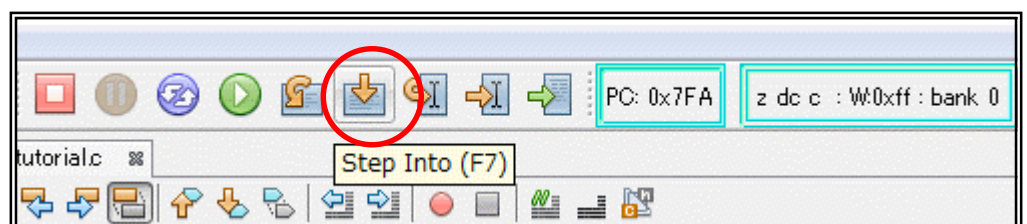
## ■ ステップ実行

デバッグの基本は一行ずつ実行してプログラムが設計どおり動いていくか確認することです。「Step Over」ボタンをクリックすると一行ずつ実行します。



```
Start Page 00_tutorial.c
7
8 // PIC16F1827 コンフィグレーションビットの設定
9 #pragma config FOSC = INTOSC, WDTE = OFF, PWRTE = ON, MCLRE = OFF, CP = OFF
10 #pragma config CPD = OFF, BOREN = ON, CLKOUTEN = OFF, IESO = ON, FCMEN = ON
11 #pragma config WRT = OFF, PLLEN = ON, STVREN = ON, BORV = LO, LVP = OFF
12
13 // インクルードファイル
14 #include <xc.h>
15
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable, 8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RBO = 1; // LEDオフ
33             for(i=0; i<100000; i++){ // ウェイト
34
35         }
36 }
```

なお、似たような機能に「Step Into」ボタンがあります。「Step Over」は関数をコールしている行を一行として実行しますが、「Step Into」は関数の中に入って一行ずつ実行します。状況に応じて使い分けましょう。





## ■ レジスタファイルの値の確認

デバッグの際に、レジスタファイルの値を確認したいことがあります。そのときはマウスカーソルを、値を見たいレジスタファイルのラベルに重ねてください。例えば、レジスタファイル「ANSELB」にマウスカーソルを重ねると、次のように表示されます。

```

22 // address = 0x18D, ANSELB = 0xFE
23
24 ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25 TRISB = 0x00; // 出力ポートに設定

```

「ANSELB」のアドレスが 0x18D 番地で、現在の値は 0xFE なのがわかります。ここで、「Step Over」ボタンをクリックして一行実行します。そのあと、同じように「ANSELB」にマウスカーソルを重ねると、次のように表示されます。

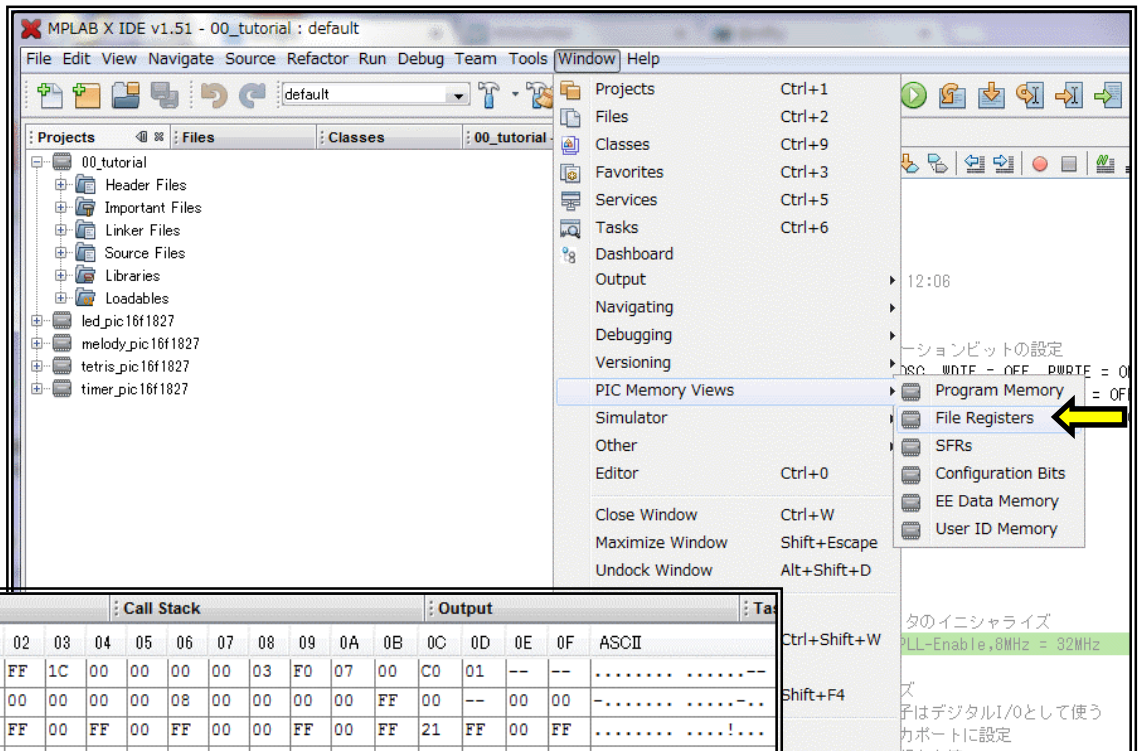
```

22 // address = 0x18D, ANSELB = 0x00
23
24 ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25 TRISB = 0x00; // 出力ポートに設定

```

プログラムどおり、0x00 になったことがわかります。

また、レジスタファイルの一覧を見るウィンドウも用意されています。メニューから [Window] → [PIC Memory Views] → [File Registers] と選択してください。



Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
000	00	00	FF	1C	00	00	00	00	03	F0	07	00	C0	01	--	--	.....
010	--	00	00	00	00	00	08	00	00	00	00	FF	00	--	00	00	-.....
020	FF	01	FF	00	FF	00	FF	00	00	FF	00	FF	21	FF	00	FF	.....!
030	FF	00	EF	00	FB	00	FF	04	00	FF	00	BF	00	FD	00	FF	.....
040	FF	50	7B	00	FF	00	FF	20	00	FF	00	FF	00	FF	01	FF	.P{....
050	FF	00	DF	20	FF	00	FF	00	00	FF	00	FF	00	FF	00	BF	.....
060	FF	00	FF	00	DF	02	EF	00	04	FF	00	FF	00	DF	40	FF	.....@.
070	02	CE	00	00	FF	00	F7	00	00	BB	00	FF	00	DF	00	FF	.....
080	00	00	79	1C	00	00	00	00	03	F0	07	00	FF	FF	--	--	..y.....
090	--	00	00	00	00	FF	0C	16	00	F0	FF	00	03	00	00	--	-.....
0A0	FF	00	FF	00	FF	00	FF	00	80	FD	02	FF	00	DF	00	FF	.....
0B0	FF	20	FF	00	FF	00	FF	20	00	FF	00	FF	42	FD	00	FF	.....B...
0C0	FF	00	FF	00	DF	02	FF	02	04	FF	08	F5	00	FF	00	FF	.....
0D0	FF	00	FF	00	FF	41	FE	02	A0	FF	00	FF	20	FF	10	FE	.....A..
0E0	FF	00	FF	01	DF	00	FF	20	00	FF	00	FF	28	FF	00	FF	.....
0F0	02	CE	00	00	FF	00	F7	00	00	BB	00	FF	00	DF	00	FF	.....

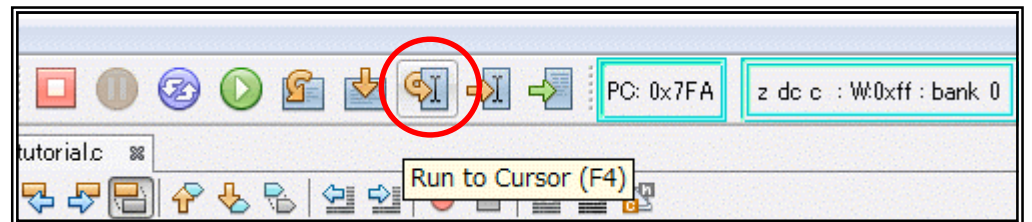
## ■ カーソル行まで実行

「Step Over」や「Step Into」を使えば一行ずつプログラムを実行することができますが、ある行までプログラムを実行してそこで止めたい、ということがあります。

右の例ではプログラムカウンタが 21 行にあります。メインループの最後の行、33 行まで実行してプログラムを止めたい場合、カーソルを 33 行にもって行って「Run to Cursor」ボタンをクリックします。

```
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable,8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RBO = 1; // LEDオフ
33         } // ウェイト
34     }
35 }
```

カーソルをセット



そうすると、プログラムカウンタの位置からスタートし、カーソルのある行まで実行すると停止します。

```
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable,8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RBO = 1; // LEDオフ
33         } // ウェイト
34     }
35 }
```

## ■ ブ레이크機能

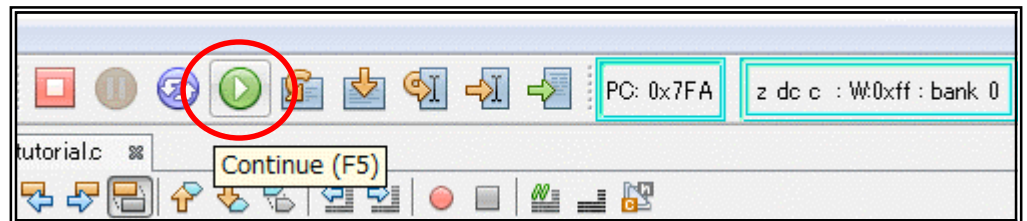
同じような機能で、ブ레이크機能があります。プログラムの実行を止めたい行(ブ레이크ポイント)を登録し、プログラム実行中にその行に到達したら停止します。

右の例で、33行で停止したい場合、「33」という数字のところを左クリックします。すると、33行が赤くかわり、ブ레이크が張られたことが明示されます。

この状態で、「Continue」ボタンをクリックするとプログラムカウンタの行から実行します。

```
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable,8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RBO = 1; // LEDオフ
33             for(i=0; i<100000; i++){ // ウェイト
34         }
35     }
```

左クリック



カーソル行まで実行すると停止します。

なお、ブ레이크ポイントは一つまで指定できます。また、ブ레이크ポイントが指定されている間は「Run to Cursor」機能は使用できません。

```
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable,8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32         RBO = 1; // LEDオフ
33         for(i=0; i<100000; i++){ // ウェイト
34     }
35 }
```

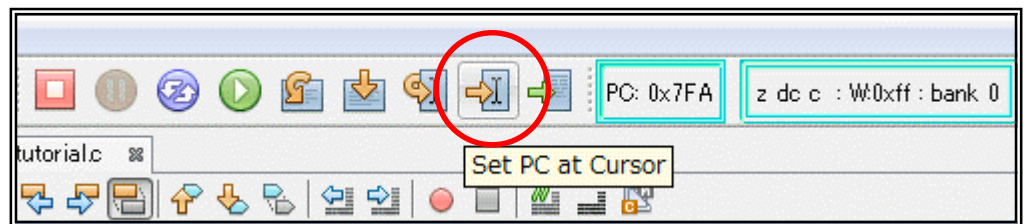
## ■ プログラムカウンタの指定

プログラムカウンタの位置を任意の場所に設定することができます。右の例では現在33行にプログラムカウンタがあります。これを24行に設定しましょう。

まずプログラムカウンタを設定したい24行にカーソルをあわせます。

この状態で「Set PC at Cursor」ボタンをクリックすると、プログラムカウンタがカーソル行にセットされます。

```
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable,8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RBO = 1; // LEDオフ
33             for(i=0; i<100000; i++){ // ウェイト
34         }
35     }
```



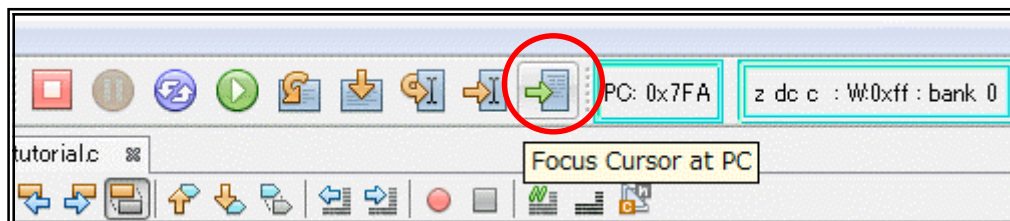
```
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable,8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RBO = 1; // LEDオフ
33             for(i=0; i<100000; i++){ // ウェイト
34         }
35     }
```

## ■ プログラムカウンタの位置を表示する

画面上でプログラムコードを読んでいくと、今プログラムカウンタがどこにあるのかわからなくなることがあります。そのようなときに、ワンクリックでプログラムカウンタのある行の画面に戻ることができます。

右の例ではプログラムカウンタの位置が画面に表示されていません。ここで、「Focus Cursor at PC」ボタンをクリックすると、プログラムカウンタが示している33行を含む形式でソースリストが画面に表示されます。

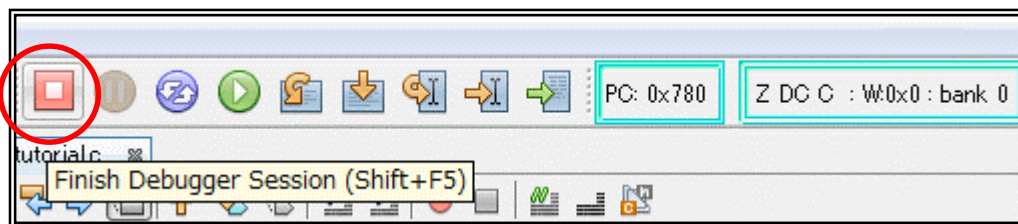
```
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable,8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
```



```
16 //メインルーチン
17 void main(void) {
18     unsigned long i;
19
20     // オシレータ制御レジスタのイニシャライズ
21     OSCCON = 0xf0; // 4xPLL-Enable,8MHz = 32MHz
22
23     // PORTBのイニシャライズ
24     ANSELB = 0x00; // 端子はデジタルI/Oとして使う
25     TRISB = 0x00; // 出力ポートに設定
26     PORTB = 0xff; // 初期出力値
27
28     // メインループ
29     while(1){
30         RBO = 0; // LEDオン
31         for(i=0; i<100000; i++){ // ウェイト
32             RBO = 1; // LEDオフ
33             for(i=0; i<100000; i++){ // ウェイト
34         }
35     }
```

## ■ デバッガの終了

デバッグ機能を終えて通常の MPLAB に戻るときは、「Finish Debugger Session」ボタンをクリックします。



なお、デバッグ時にダウンロードしたプログラムには、ユーザプログラムと一緒にデバッガ実行プログラムがダウンロードされており、リセット（電源オン）でデバッガ実行プログラムが起動します。そのため、このままだとユーザプログラムを起動することができません。それで、あらためてプログラムをビルドし「PICkit3」を使ってユーザプログラムのみをダウンロードしてください。

### ■ 「PICkit3」接続時の注意事項 ■

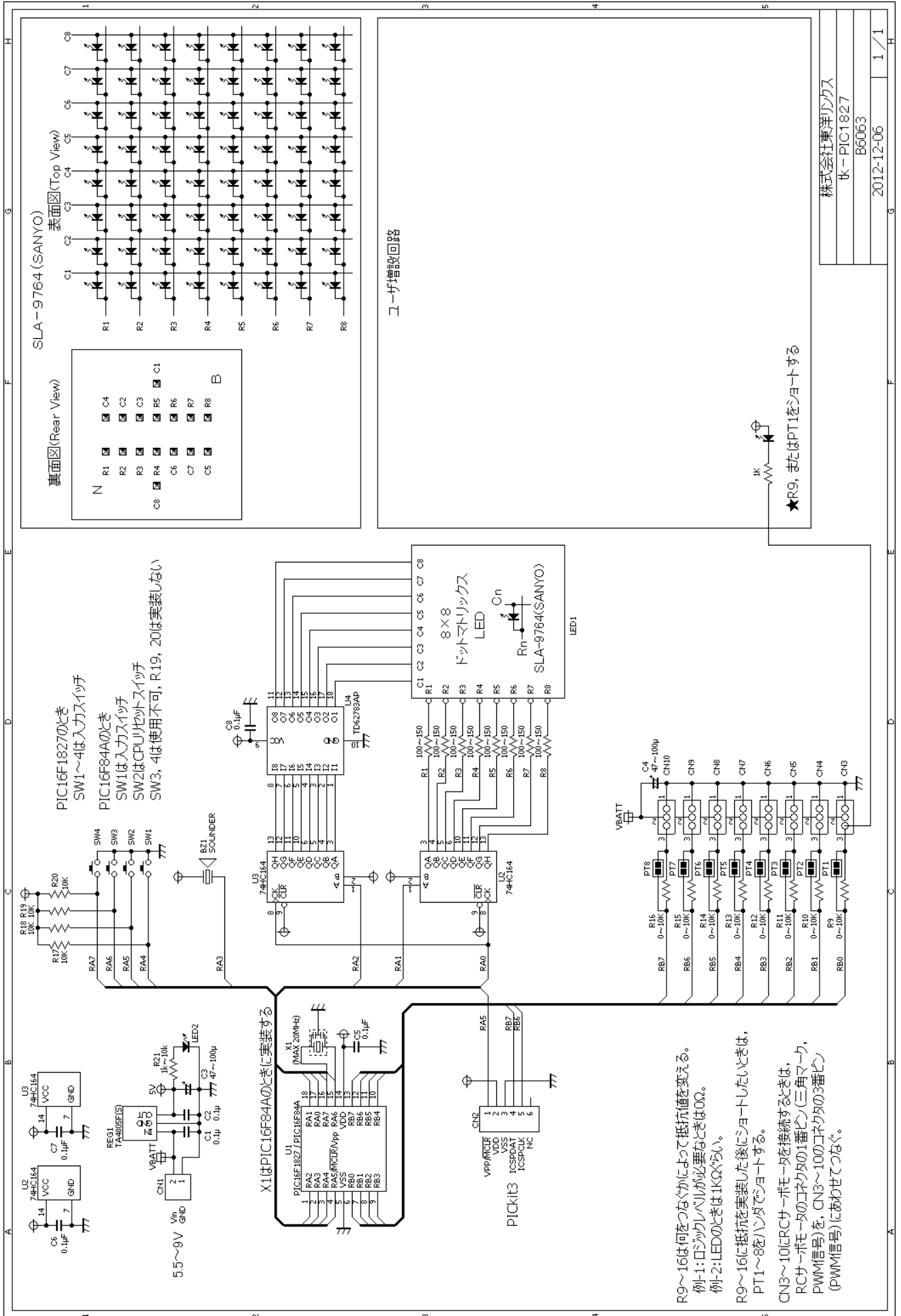
「PICkit3」は PIC16F1827 の RA5 (MCLR/Vpp), RB6 (ICSPCLK), RB7 (ICSPDAT) を使って通信します。このマニュアルで取り上げたプログラム例ではこれらのポートを使用しないので問題ありませんが、これらのポートを使う回路を追加した場合デバッグできません。「PICkit3」を使ってデバッグするのであれば、回路設計時に RA5, RB6, RB7 は使用しないでください。

なお、回路設計時の注意事項の詳細は、マイクロチップテクノロジーの資料か、このマニュアルの付録をご覧ください。



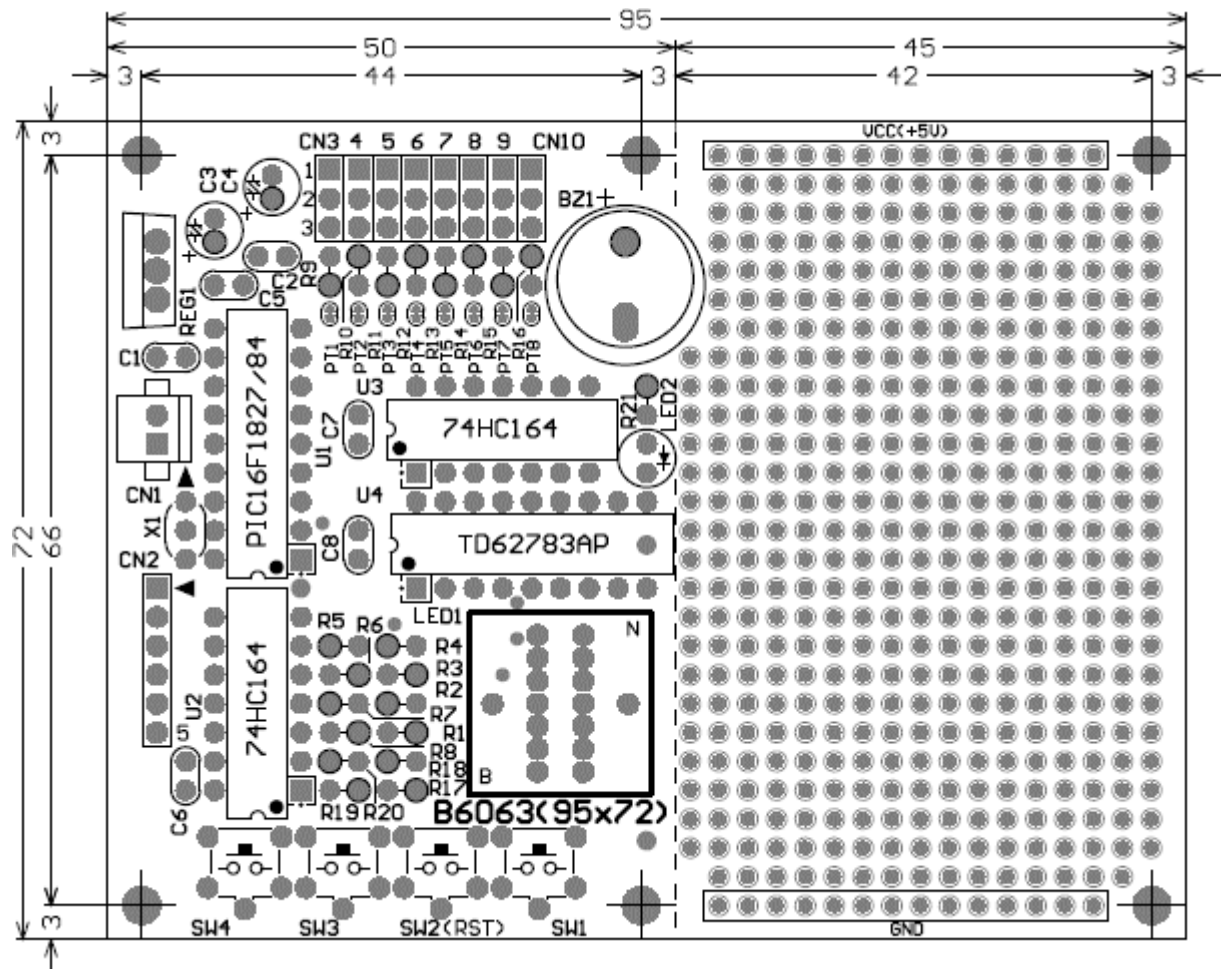
このマニュアルではごく一部の機能しかご紹介していません。「MPLAB X IDE」, 「MPLAB XC8」, 「PICkit3」には、他にも便利な機能がたくさんあります。HELP ファイルや、マイクロチップテクノロジーの資料を見て活用してください。

# 付録1：回路図



株式会社東洋エクス  
 株 - PIC1827  
 B6063  
 2012-12-06  
 1 / 1

付録-2 : シルク図(実装図)





## 回路とコネクタのピン配置



1ピン インジケータ

\*デバッガとして使う場合、オシレータが必要です。  
\*\*デバイスにAVDDラインとAVSSラインがある場合、デバッガを使うには、両ラインを接続する必要があります。

### ターゲット コネクタのピン配置

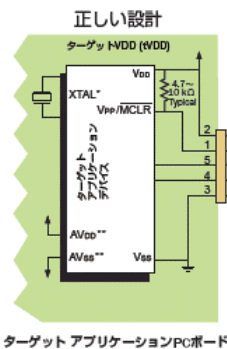
ピン	信号
1	MCLR/Vpp
2	VDDターゲット
3	Vssグラウンド
4	PGD (ICSPDAT)
5	PGC (ICSPCLK)
6	未接続*

\*将来用に予約済み

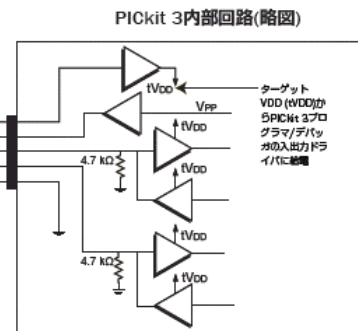
### PICkit 3コネクタのピン配置

ピン	信号
1	MCLR/Vpp
2	VDDターゲット
3	Vssグラウンド
4	PGD (ICSPDAT)
5	PGC (ICSPCLK)
6	未接続*

\*将来用に予約済み



ターゲット アプリケーションPCボード



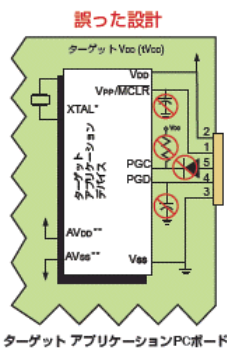
PICkit 3内部回路(略図)

### 推奨設定

コンポーネント	設定
オシレータ	・ OSCピットを正しく設定 ・ 動作している事
電源	ターゲットが供給
WDT	無効(デバイスごとに異なる)
コード保護	無効
テーブル読み出し保護	無効
LVP	無効
BOD	VDD > BOD VDD min
JTAG	無効
AVDDとAVSS	接続する必要あり
PGCx/PGDx	該当する場合、正しいチャンネルが選択されている事
プログラミング	VDD電圧がプログラミング仕様を満たす事

Note: 詳細は、PICkit 3オンラインヘルプを参照してください。

### ターゲット回路設計上の注意



ターゲット アプリケーションPCボード

- VDD に静電容量が 100  $\mu$ F を超えるコンデンサを接続しないでください。PICkit 3 から電源を供給する場合、総負荷によってはターゲットに遅やかに給電されません。
- MCLR にコンデンサを接続しないでください。コンデンサを接続すると、Vpp が高速に遷移できません。
- PGC/PGD にプルアップ抵抗を接続しないでください。両ラインには MPLAB REAL ICE 内に 4.7 k $\Omega$  のプルダウン抵抗が存在するため、プルアップ抵抗を接続すると分圧されます。
- PGC/PGD に多重化しないでください。両ラインは PICkit 3 への通信専用です。
- PGC/PGD にコンデンサを接続しないでください。コンデンサを接続すると、プログラミングとデバッグの通信中にデータラインとクロックラインが高速に遷移できません。
- PGC/PGD にダイオードを接続しないでください。ダイオードを接続すると、PICkit 3 とターゲット PIC<sup>®</sup> MCU の間で双方向通信ができなくなります。
- 推奨ケーブル長より長いケーブルを使わないでください。最大ケーブル長については、PICkit 3 オンラインヘルプまたはユーザガイドの「Hardware Specification」を参照してください。

上記の資料は、マイクロチップテクノロジーの資料から抜粋しました。tk-PIC1827 は次のとおりこれらの条件を満たしています。

- Vpp/MCLR (RA5) を 10K $\Omega$  程度でプルアップする。コンデンサを接続しない。  
→ R18 (10K $\Omega$ ) でプルアップしている。コンデンサは接続していない。デバッガ使用時のアプリケーションでは SW2 を入力スイッチとしては使わないでください。
- PGC (ICSPCLK, RB6)/PGD (ICSPDAT, RB7) に、プルアップ抵抗、コンデンサ、ダイオードを接続しない。  
→ RB6 と RB7 はユーザに開放しているため未使用。デバッガ使用時のアプリケーションでは RB6, RB7 は使わないでください。

★部品が足りない場合やお問い合わせはメールか FAX でお願い致します。

## **(株)東洋リンクス**

〒102-0093 東京都千代田区平河町 1-2-2 朝日ビル

TEL : 03-3234-0559 FAX : 03-3234-0549

URL : <http://www2.u-netsurf.ne.jp/~toyolinx>

E-Mail : [toyolinx@va.u-netsurf.jp](mailto:toyolinx@va.u-netsurf.jp)

※本書の内容は将来予告無しに変更することがあります(2013年2月作成)

20130212