ARM7/TDMI LPC2148

フリーの開発環境 Eclipse + GCC

低価格の OpenJTAG

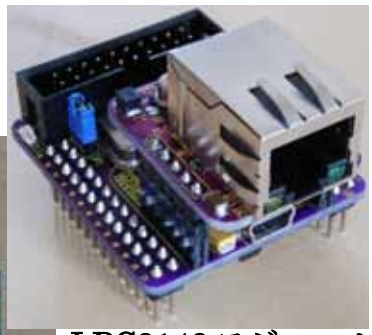
マニュアル

株式会社日昇テクノロジー

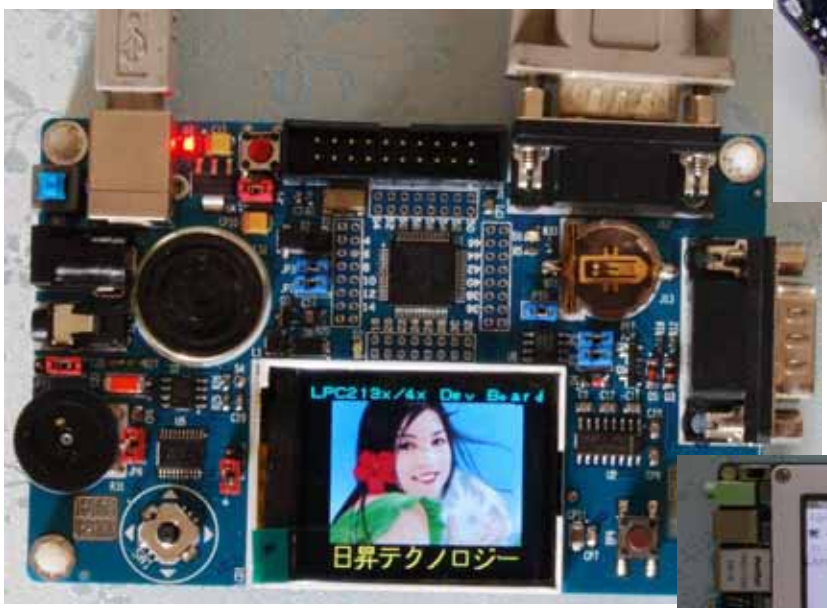
http://www.csun.co.jp

info@csun.co.jp

2009/8/1



LPC2148 モジュール



LPC2148 開発キット

LPC2148 は ARM9 を USB で繋ぐ

第一章 ARM7TDMI/LPC2148 とは 3
第二章 ARM7TDMI/LPC2148 開発キット 6
第三章 初体験 7
3.1 デフォルトのサンプル(テトリス) 7
3.2 書き込みツールのインストール 7
3.3 書き込み 12
3.4 OpenJTAGで書き込み 16
3.5 OpenJTAGのドライバをインストールする 18
3.6 OpenOCDのダウンロードとインストール 20
3.7 telnetで書き込み 23
3.8 GCCサンプルの紹介 24
3.9 USBダウントローダ 25
第四章 クロス開発環境 28
4.1 GCCツールチェーン 28
4.2 Integrated Development Environment(Eclipse) 30
4.3 プロジェクトを作る 34
4.4 Eclipseプラグイン(Zylin Embedded CDT)インストール 37
4.5 ビルドの設定 42
4.6 ビルド 45
4.7 GDBの設定 47
4.8 OpenOCDの設定 53
4.9 デバッグ 57
4.10 デバッグ終了 61
第五章 SPIイーサネットモジュールENC28J60 63
第六章 USBでARM9(Linux)に接続 65
第七章 微弱無線モジュールnRF24L01(2.4GHz) 69

※ 使用されたソースコードはhttp://www.csun.co.jp/からダウンロードできます。

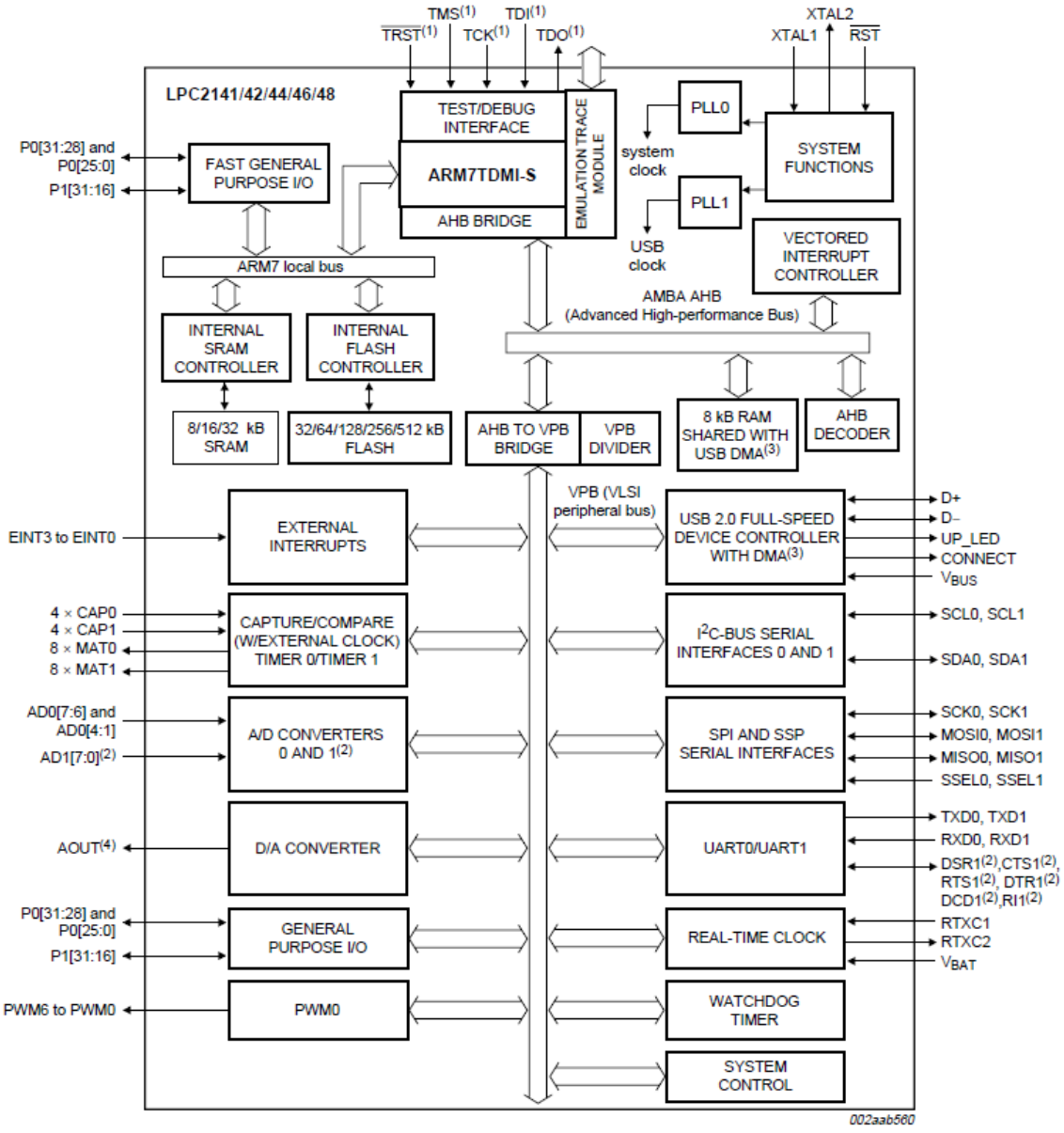
第一章 ARM7TDMI/LPC2148 とは

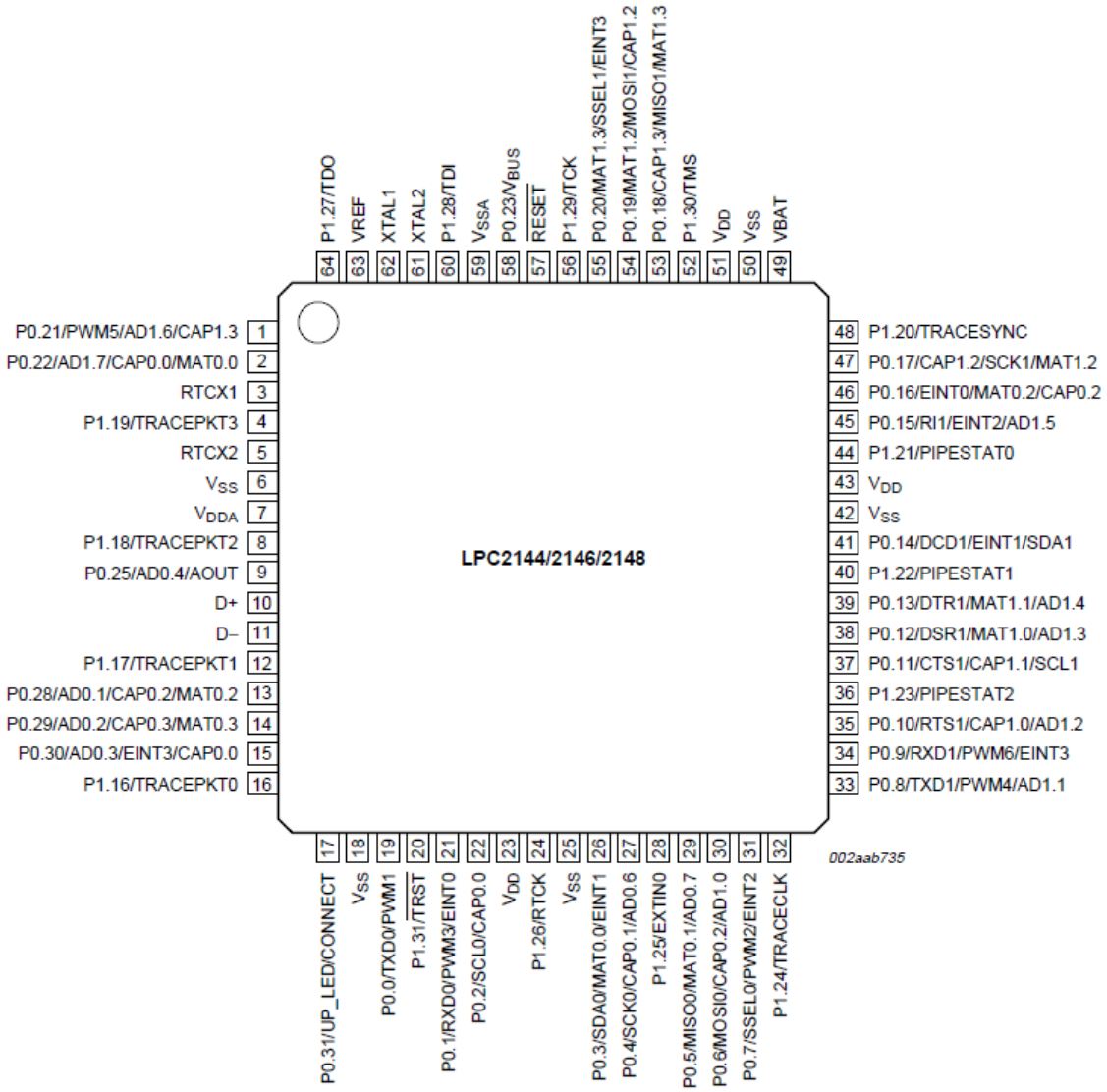
■LPC2148 は ARM7TDMI コアを使った低消費電力・高速 NXP 社のマイコンです。
■USB ターゲットを CPU に内蔵しています。USB を使ったシステムの開発・評価に最適です。
■H8 や SH と比べると価格も安くて、実にスピードも速い、容量も大きい、消費電力も小さいです。
■CPU は 3.3V 動作なのですが I/O ポートは 5V トレラントのため、5V 系ロジックを直結できます。
■ルネサス SH7144 と比べて

- スピードは 60MHz なので 20%速い
- フラッシュは 512K バイトなので 2 倍
- RAM は 42K あるので約 5 倍
- 消費電力は 60MHz フルスピードで 50mA 程度です。約 2～4 分の 1 です。クロックを落とせばもっと減ります。
- **ROM の書き換え回数 10 万回、事実上の無制限、データ保持 20 年。**

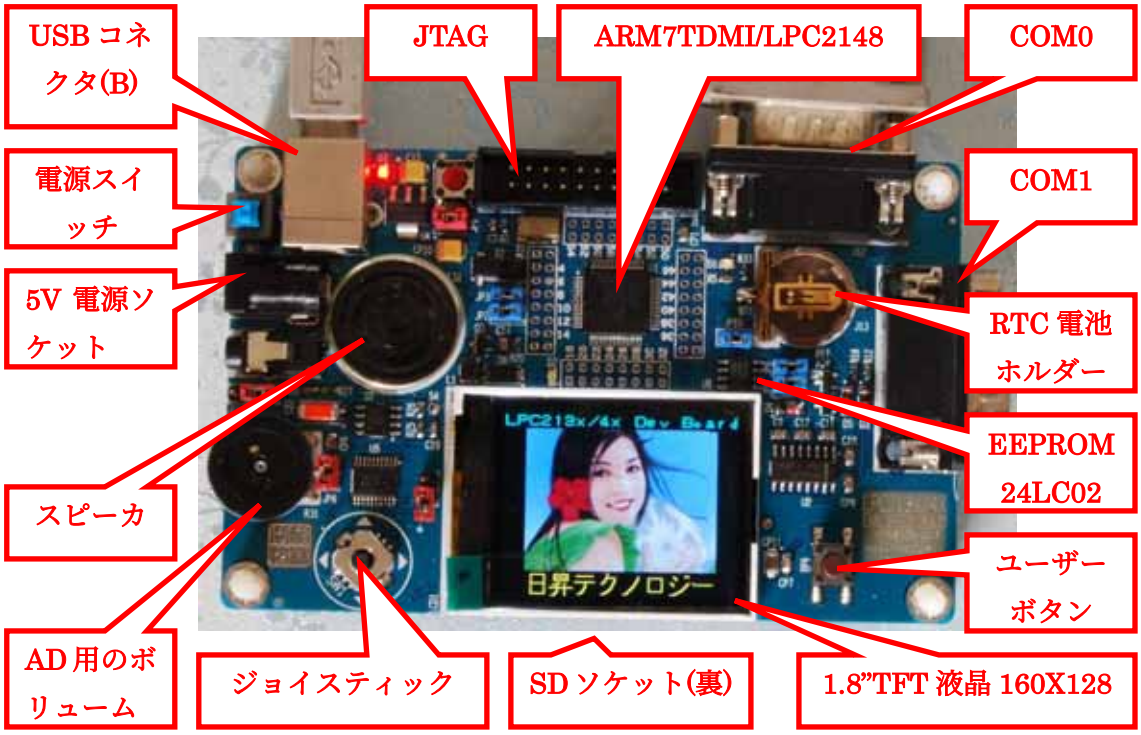
■LPC2148 の主な仕様

- ARM7TDMI-S 16/32bit RISC マイコン
- FLASH:512K バイト（H8/3069F と同じメモリ容量）
- RAM:42K バイト（H8/3069F の約 2.5 倍） ◎ＲＡＭ内蔵の１チップＣＰＵでは最大級
- 60MHz 動作（12MHzｘ５逓倍）
- USB2.0 対応インターフェース内蔵（max 12Mbps）
- 10 ビット AD コンバータ 2 ユニット内蔵、14Ch
- 10 ビット DA コンバータｘ1ｃｈ内蔵
- UART(16C550)ｘ2，I2Cｘ2，32 ビットタイマｘ2，PWMｘ6，WDT, RTC など通信系のインターフェース(SPI/SSP)は 2ch 内蔵されています。
- 5V トレラント I/O(ＣＰＵは 3.3V 動作ですが、I/O ポートは 5V の入力を受けられます)
- JTAG インターフェース内蔵
- Fast I/O 機能：通常は CPU コアと I/O のクロックが異なるため速い CPU コアでも I/O コントロールにウェイトが入ってしまい、高速に I/O を操作できません。この LPC2148 では Fast I/O 機能により高速に I/O ポートを操作できます。他のＣＰＵの約 3.5 倍のスピードでポートの操作ができます。


LPC2141/42/44/46/48
TMS(1)
TRST(1)
TCK(1)
TDI(1)
TDO(1)
XTAL2
XTAL1
RST
TEST/DEBUG INTERFACE
EMULATION TRACE MODULE
ARM7TDMI-S
AHB BRIDGE
PLL0
PLL1
system clock
USB clock
SYSTEM FUNCTIONS
VECTORED INTERRUPT CONTROLLER
P0[31:28] and P0[25:0]
P1[31:16]
FAST GENERAL PURPOSE I/O
ARM7 local bus
AMBA AHB (Advanced High-performance Bus)
INTERNAL SRAM CONTROLLER
INTERNAL FLASH CONTROLLER
8/16/32 kB SRAM
32/64/128/256/512 kB FLASH
AHB TO VPB BRIDGE
VPB DIVIDER
8 kB RAM SHARED WITH USB DMA(3)
AHB DECODER
VPB (VLSI peripheral bus)
EINT3 to EINT0
EXTERNAL INTERRUPTS
USB 2.0 FULL-SPEED DEVICE CONTROLLER WITH DMA(3)
D+
D−
UP_LED
CONNECT
VBUS
4 × CAP0
4 × CAP1
8 × MAT0
8 × MAT1
CAPTURE/COMPARE (W/EXTERNAL CLOCK) TIMER 0/TIMER 1
I2C-BUS SERIAL INTERFACES 0 AND 1
SCL0, SCL1
SDA0, SDA1
AD0[7:6] and AD0[4:1]
AD1[7:0](2)
A/D CONVERTERS 0 AND 1(2)
SPI AND SSP SERIAL INTERFACES
SCK0, SCK1
MOSI0, MOSI1
MISO0, MISO1
SSEL0, SSEL1
AOUT(4)
D/A CONVERTER
UART0/UART1
TXD0, TXD1
RXD0, RXD1
DSR1(2),CTS1(2), RTS1(2), DTR1(2) DCD1(2),RI1(2)
P0[31:28] and P0[25:0]
P1[31:16]
GENERAL PURPOSE I/O
REAL-TIME CLOCK
RTXC1
RTXC2
VBAT
PWM6 to PWM0
PWM0
WATCHDOG TIMER
SYSTEM CONTROL
002aab560


LPC2144/2146/2148
P0.21/PWM5/AD1.6/CAP1.3 1
P0.22/AD1.7/CAP0.0/MAT0.0 2
RTCX1 3
P1.19/TRACEPKT3 4
RTCX2 5
VSS 6
VDDA 7
P1.18/TRACEPKT2 8
P0.25/AD0.4/AOUT 9
D+ 10
D− 11
P1.17/TRACEPKT1 12
P0.28/AD0.1/CAP0.2/MAT0.2 13
P0.29/AD0.2/CAP0.3/MAT0.3 14
P0.30/AD0.3/EINT3/CAP0.0 15
P1.16/TRACEPKT0 16
P0.31/UP_LED/CONNECT 17
VSS 18
P0.0/TXD0/PWM1 19
P1.31/TRST 20
P0.1/RXD0/PWM3/EINT0 21
P0.2/SCL0/CAP0.0 22
VDD 23
P1.26/RTCK 24
VSS 25
P0.3/SDA0/MAT0.0/EINT1 26
P0.4/SCK0/CAP0.1/AD0.6 27
P1.25/EXTIN0 28
P0.5/MISO0/MAT0.1/AD0.7 29
P0.6/MOSI0/CAP0.2/AD1.0 30
P0.7/SSEL0/PWM2/EINT2 31
P1.24/TRACECLK 32
P0.8/TXD1/PWM4/AD1.1 33
P0.9/RXD1/PWM6/EINT3 34
P0.10/RTS1/CAP1.0/AD1.2 35
P1.23/PIPESTAT2 36
P0.11/CTS1/CAP1.1/SCL1 37
P0.12/DSR1/MAT1.0/AD1.3 38
P0.13/DTR1/MAT1.1/AD1.4 39
P1.22/PIPESTAT1 40
P0.14/DCD1/EINT1/SDA1 41
VSS 42
VDD 43
P1.21/PIPESTAT0 44
P0.15/RI1/EINT2/AD1.5 45
P0.16/EINT0/MAT0.2/CAP0.2 46
P0.17/CAP1.2/SCK1/MAT1.2 47
P1.20/TRACESYNC 48
VBAT 49
VSS 50
VDD 51
P1.30/TMS 52
P0.18/CAP1.3/MISO1/MAT1.3 53
P0.19/MAT1.2/MOSI1/CAP1.2 54
P0.20/MAT1.3/SSEL1/EINT3 55
P1.29/TCK 56
RESET 57
P0.23/VBUS 58
VSSA 59
P1.28/TDI 60
XTAL2 61
XTAL1 62
VREF 63
P1.27/TDO 64
002aab735

第二章 ARM7TDMI/LPC2148 開発キット



開発キットのインターフェース

- RS232(16C550) ×2
- USB2.0 device ×1
- JTAG/ICE
- SD カードソケット
- AD テスト用のボリューム
- I2C EEPROM
- ユーザーLED ×1
- ユーザーボタン×1
- ジョイスティック×1
- スピーカー×1
- 1.8 インチ TFT 液晶、分解能 160×128

外形寸法

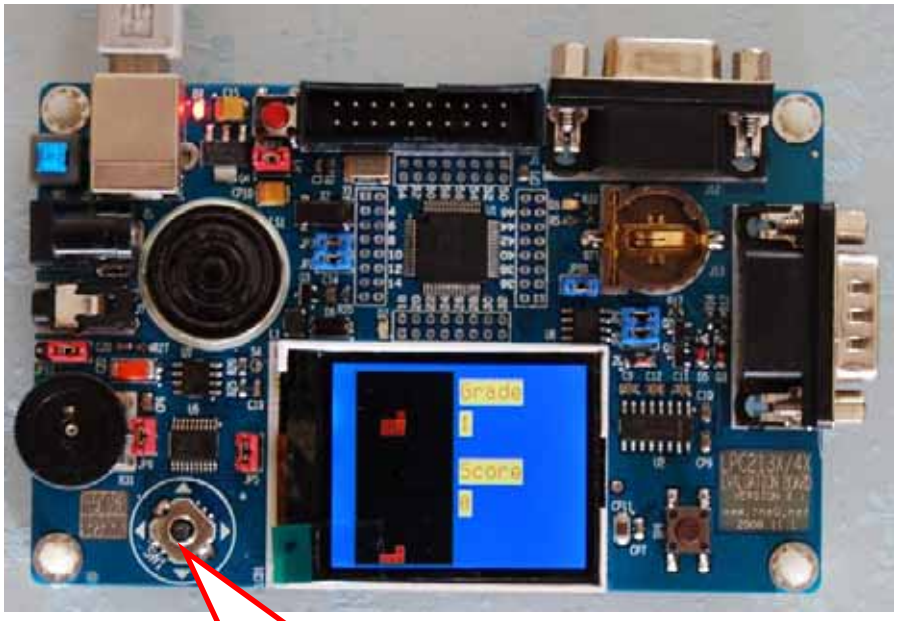
- 外形寸法: 110×70(mm) ※突起物は除く

供給電源

- 5VDC 電源、プラグ 2.1mmφ、極性はセンタープラス  です。電源スイッチと電源指示 LED 付き。USB 給電可。

第三章 初体験

3.1 デフォルトのサンプル(テトリス)



Example-2148GCC.rar

は LPC2148 開発キットのサンプルです。回路図とソースコードも含みます。なかのほかのサンプルを体験してみよう。

3.2 書き込みツールのインストール

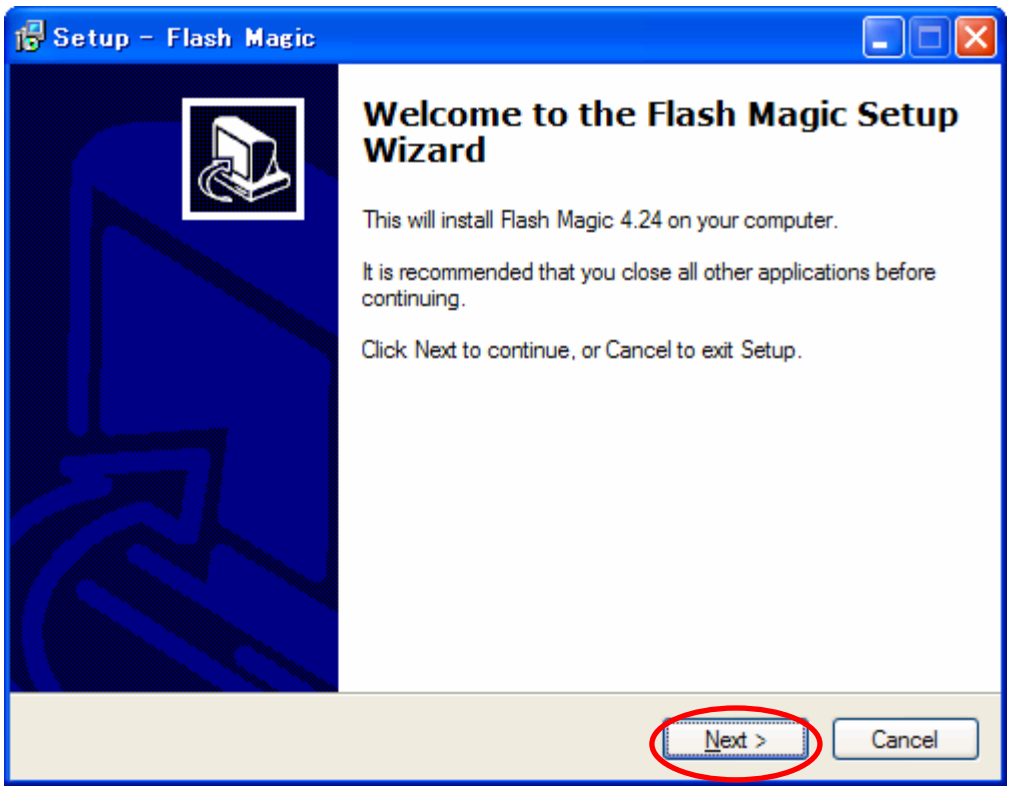
FlashMagic とは

LPC シリーズはフラッシュ ROM を内蔵しているため、ISP（In-System Program）機能によりユーザ・プログラムを書き込むことができます。そのためのプログラミング・ツー

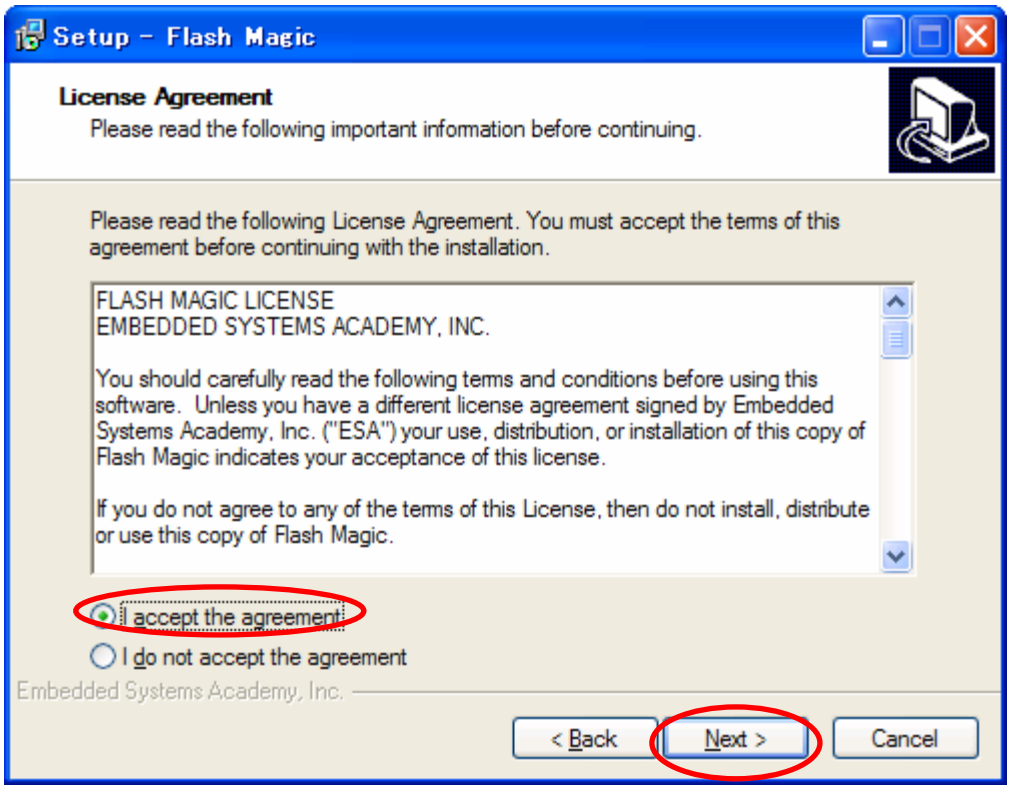
ルが FlashMagic です。FlashMagic は次の URL からダウンロードできます。

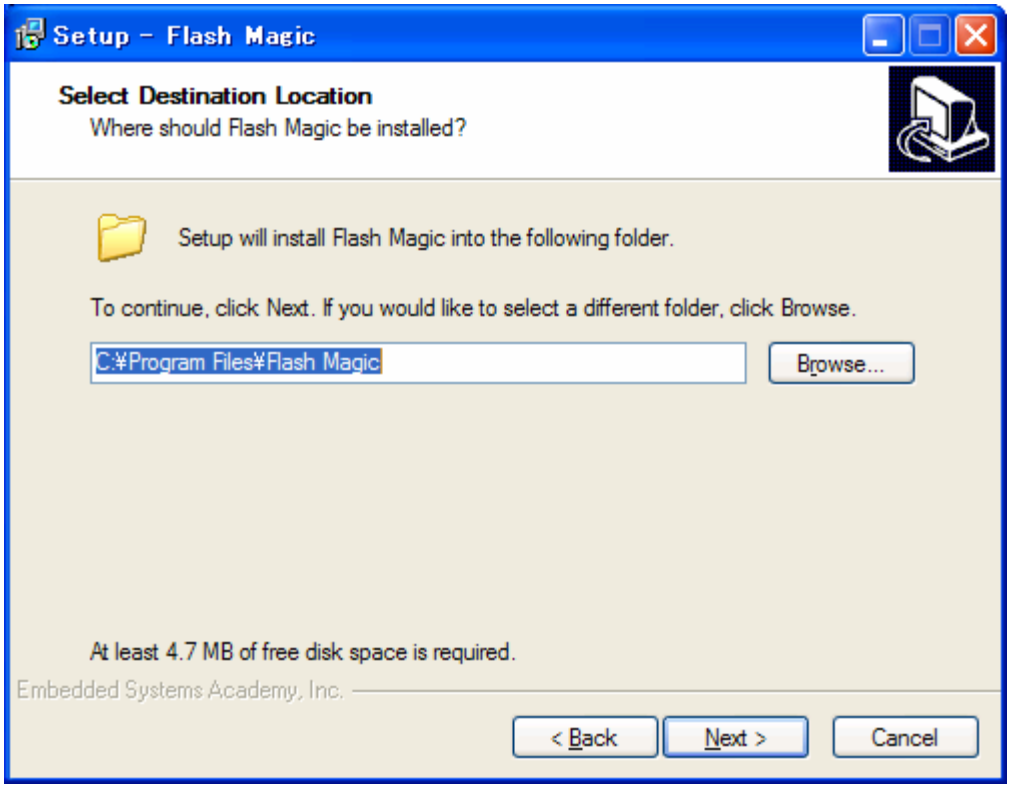
http://www.flashmagictool.com/

FlashMagic.exe を実行すると、LPC2148 ボードの書き込みツールをインストールします。LPC2148 の Flash を更新すれば、ほかのサンプルを体験できます。

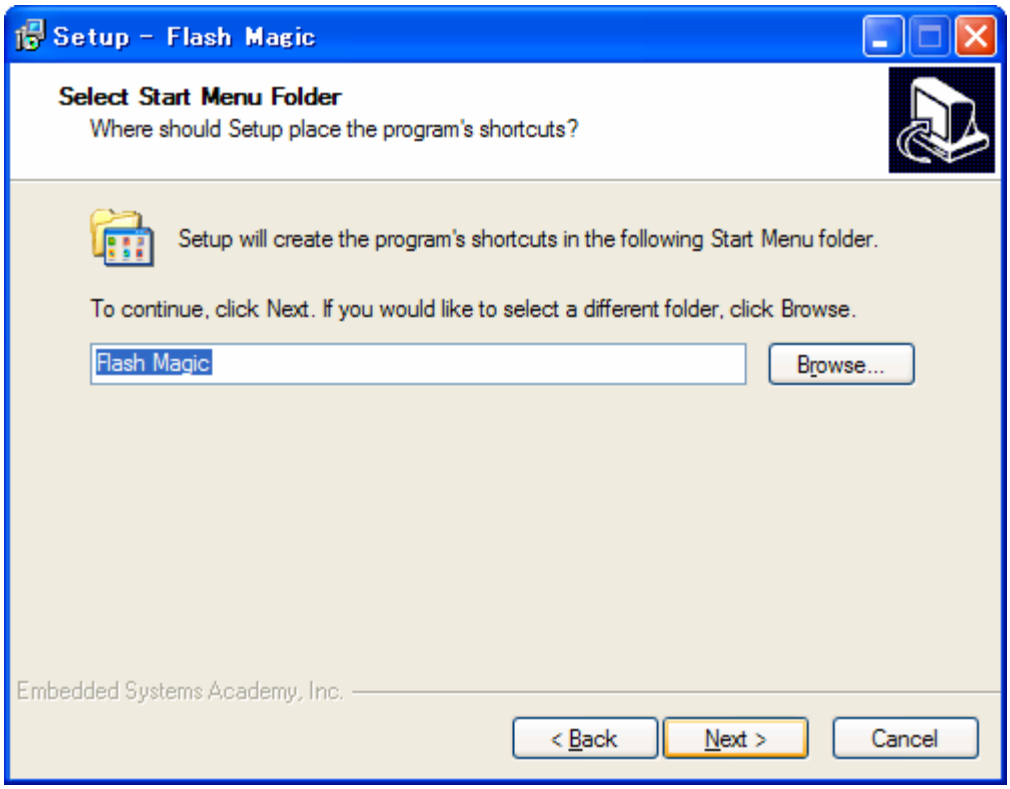


「Next」ボタンを押すと、英文のライセンスが出てきます。同意できる場合は、「**I accept the agreement**」を選択して、「Next」ボタンを押します。

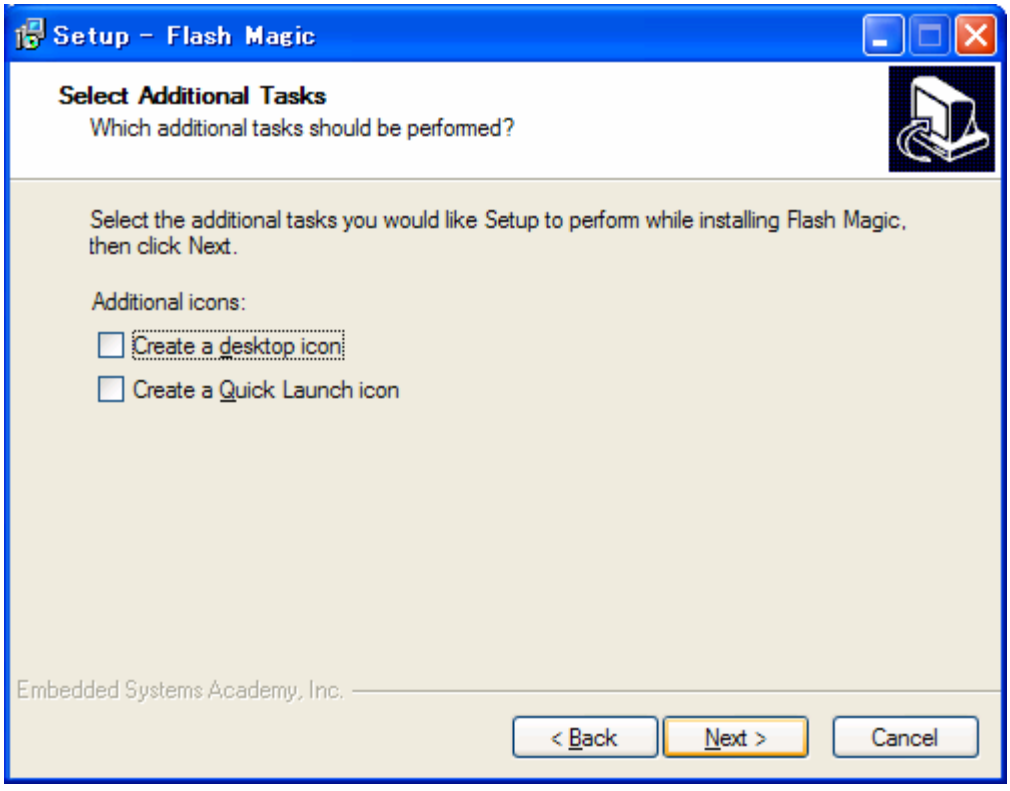




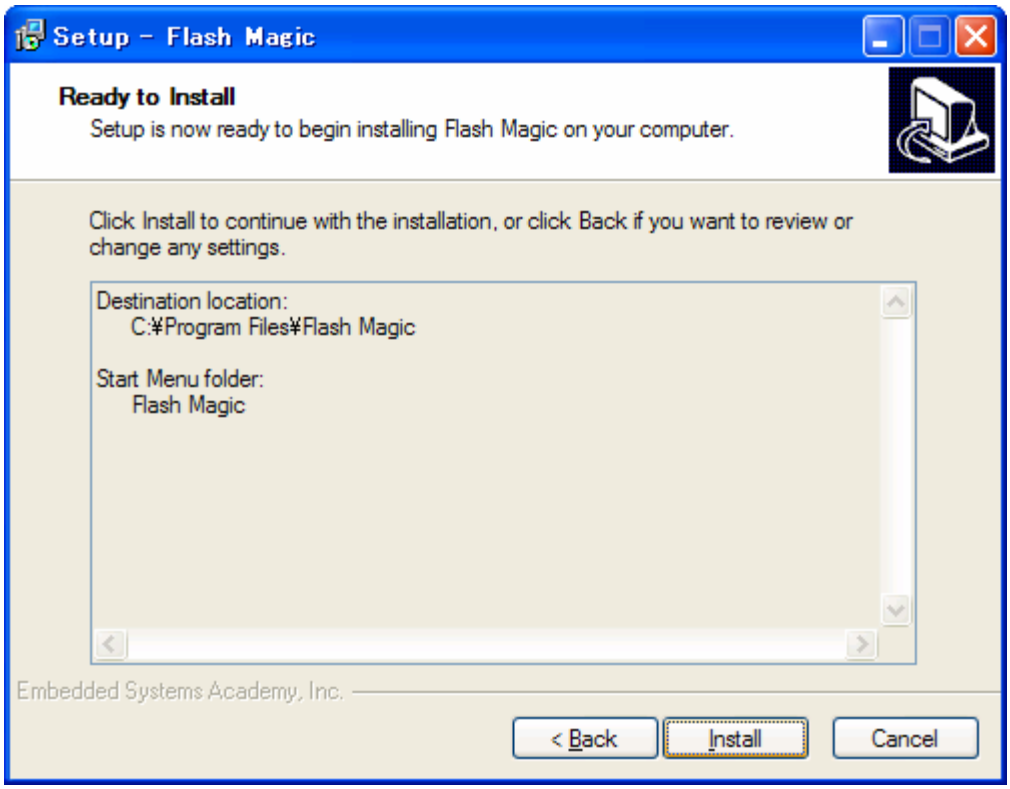
インストール先フォルダを変更せず、そのまま進んでください。



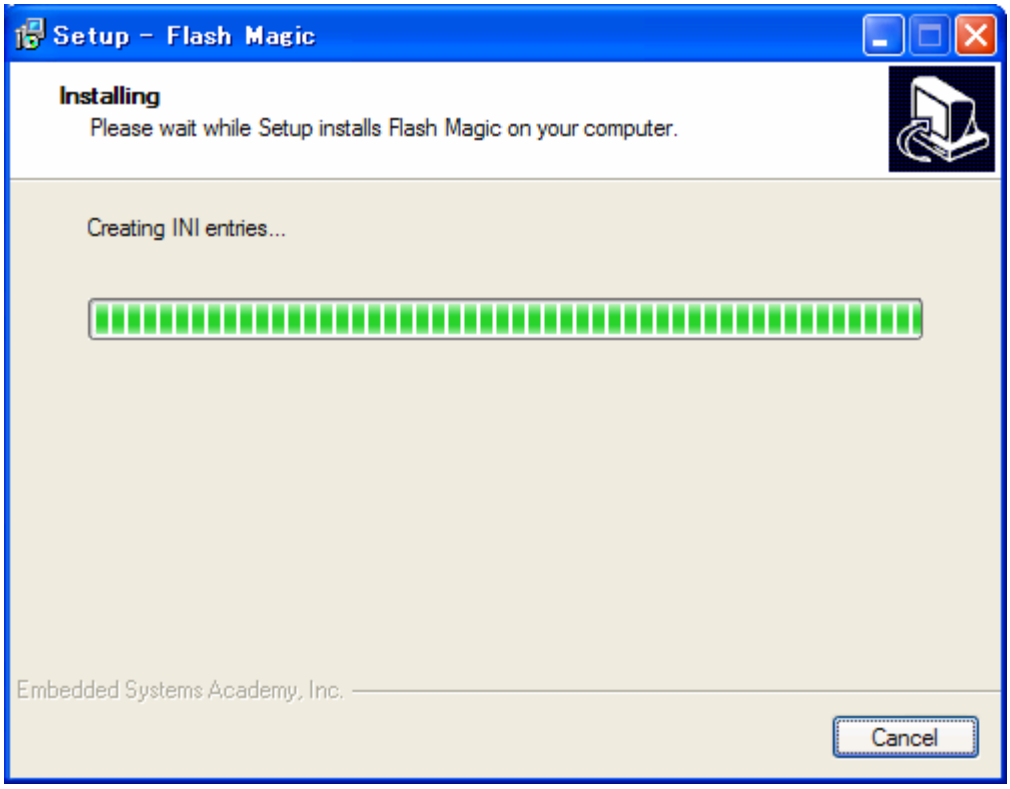
メニュー・フォルダも変更せず、そのまま進んでください。



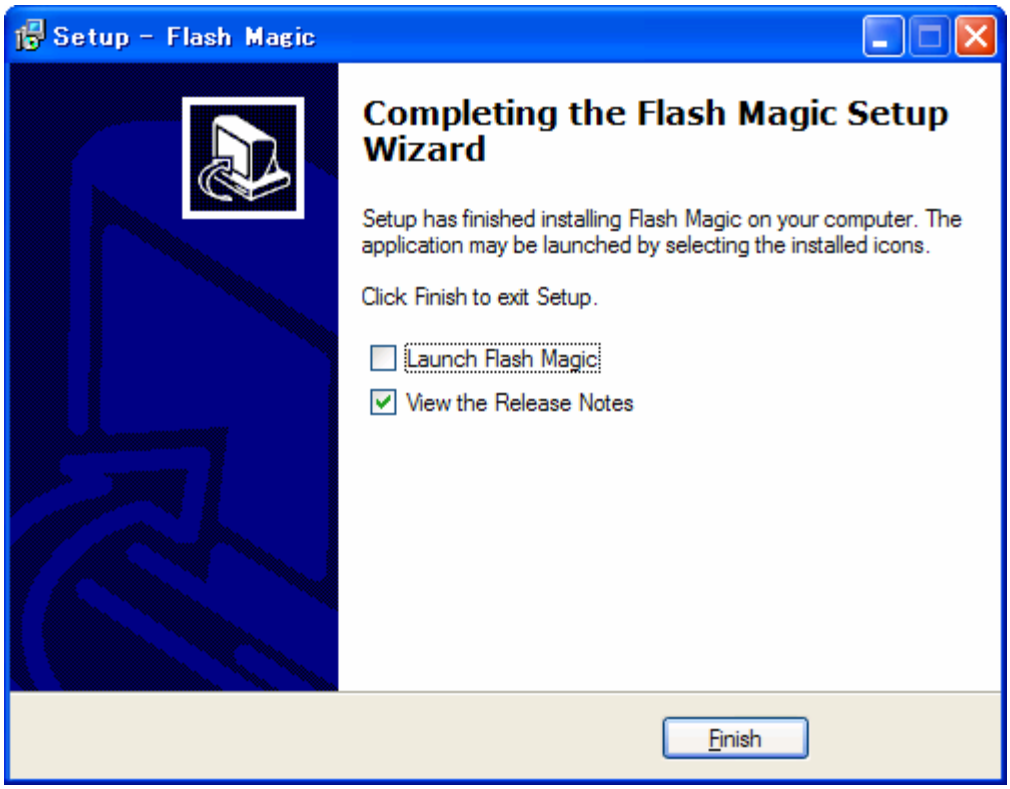
そのまま進んでください。



インストール前の確認、「install」ボタンを押してください。

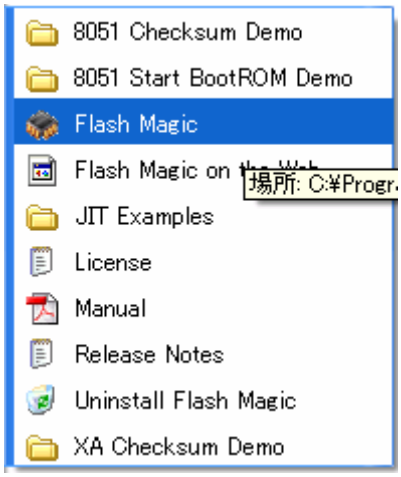


インストール中の画面です。

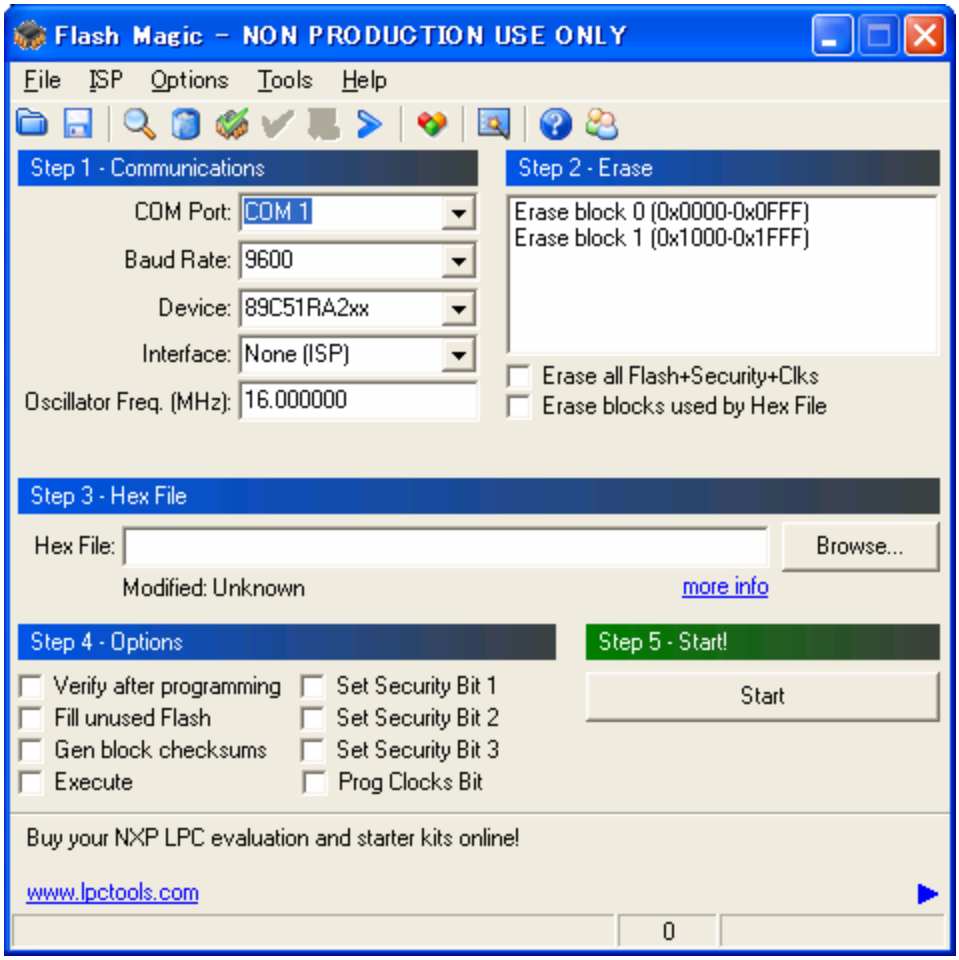


最後に「Finish」をクリックすると、ウィザードが閉じてインストールが終了します。

3.3 書き込み

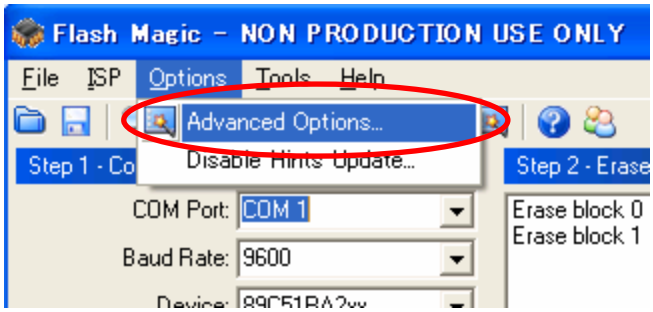


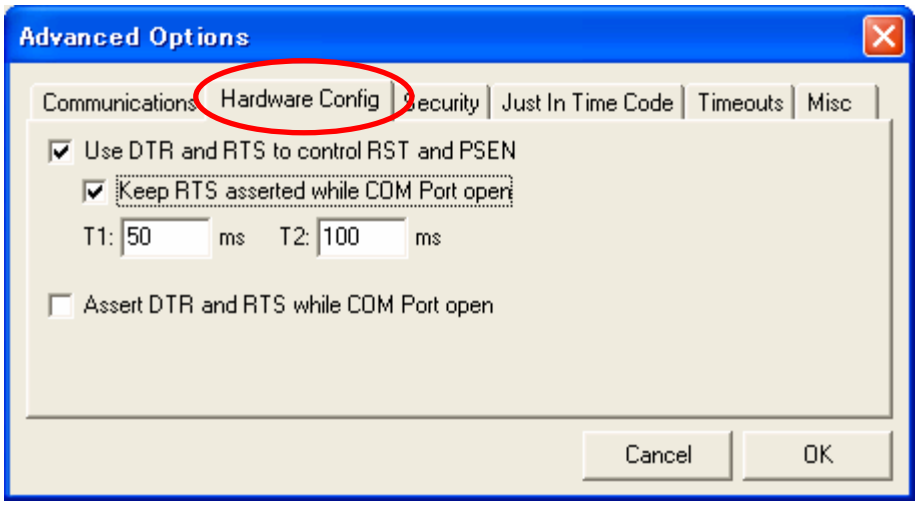
Windows のメニュー「スタート」→「Flash Magic」→「Flash Magic」を選択してください



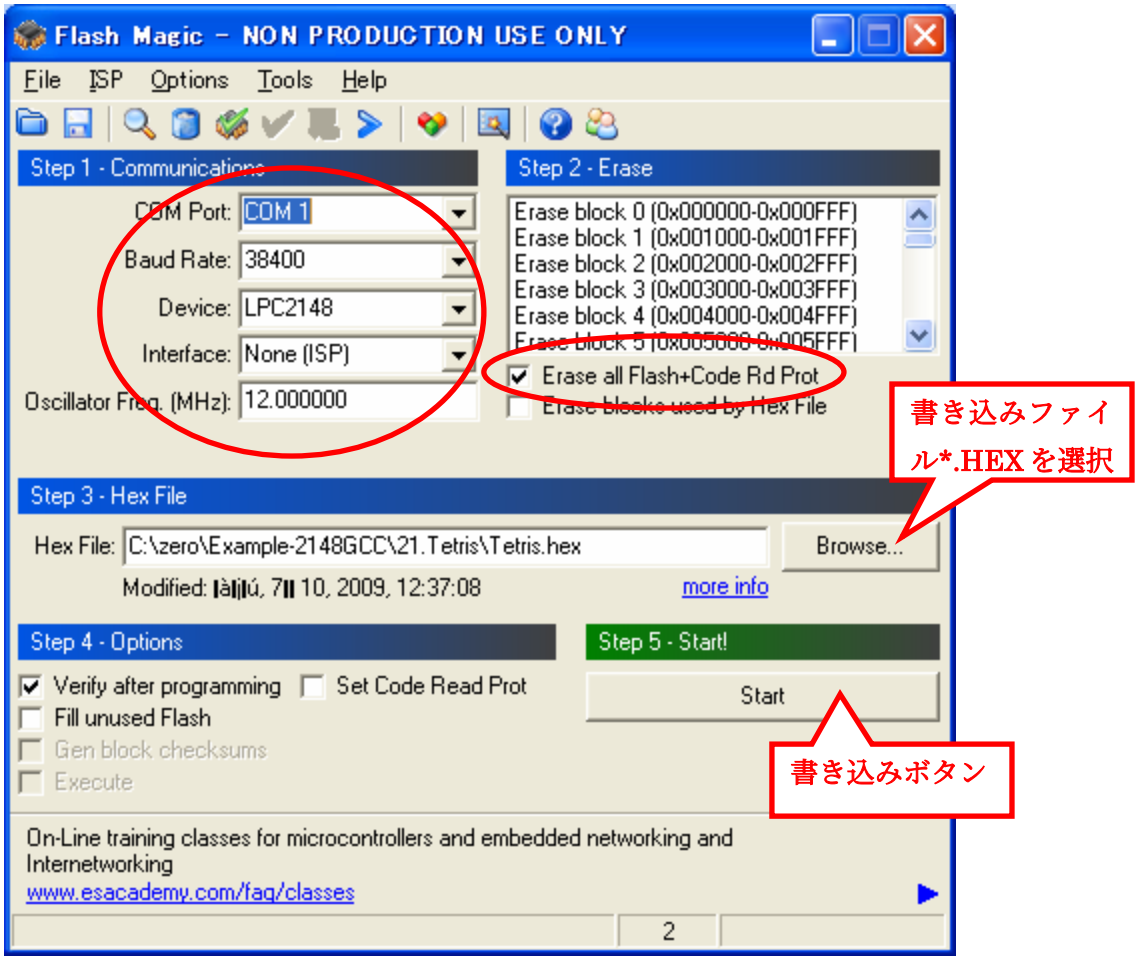
Flash Magic の初画面です。

Flash Magic のメニュー「Options」→「Advanced Options」を選択してください。

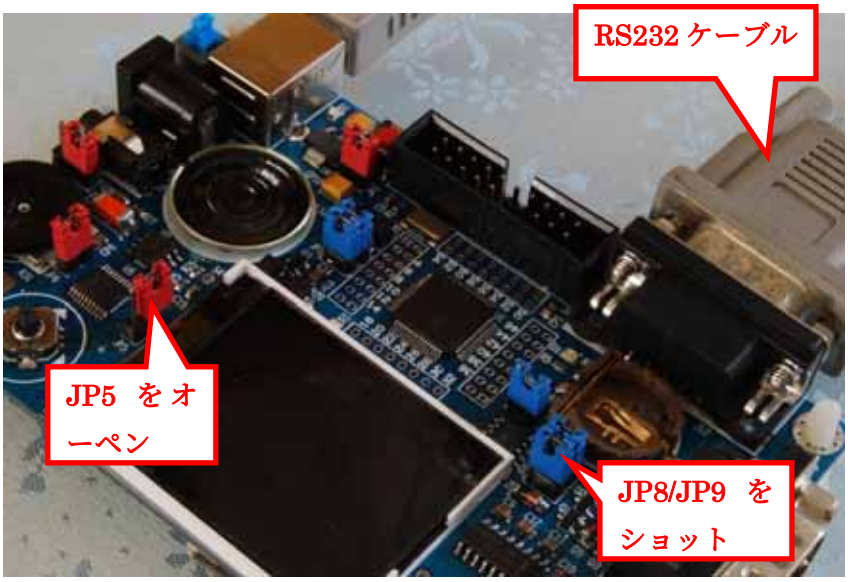




画面の通りに設定して、「OK」ボタンを押してください。

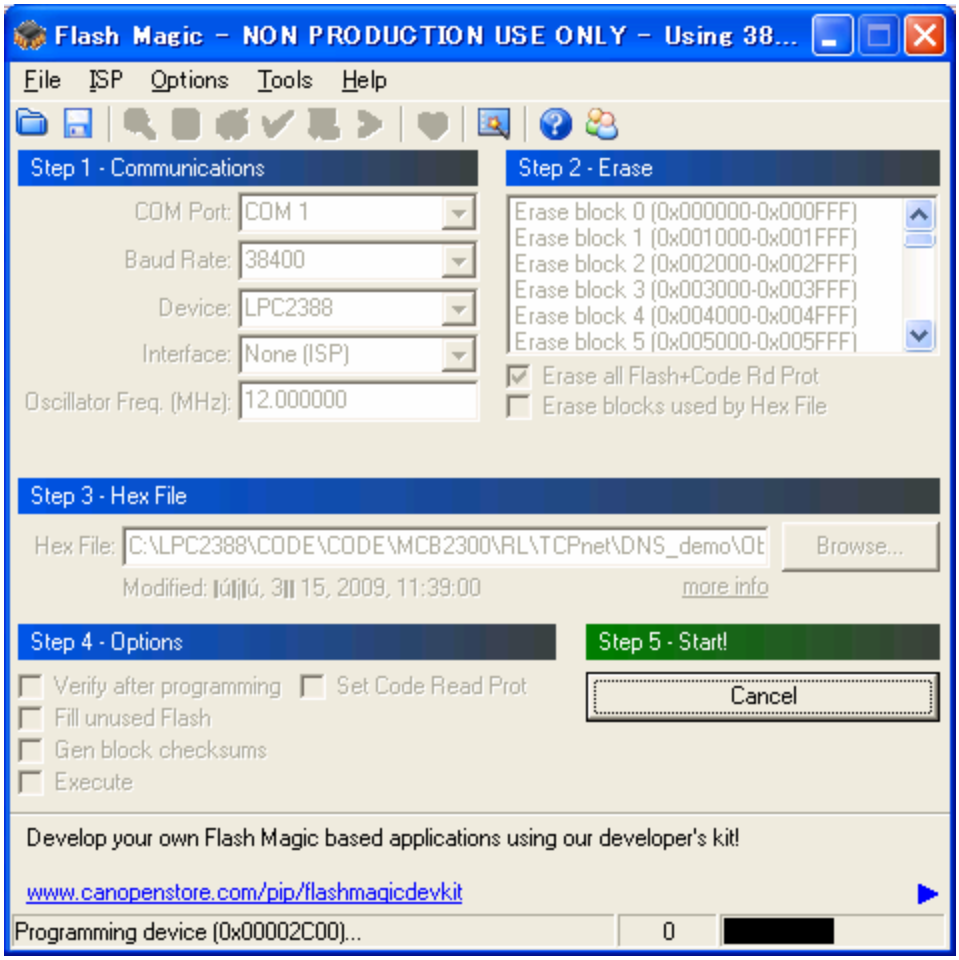


画面の通りにパラメータを設定して、「Browse」ボタンで書き込みファイル*.hex を選択してください。



書き込みの設定

書き込みボタン「Start」ボタンを押す前に、LPC2148 ボードの JP5/JP8/JP9 設定と RS232 ケーブルの接続を確認してください。「Start」ボタンを押すと、書き込み開始

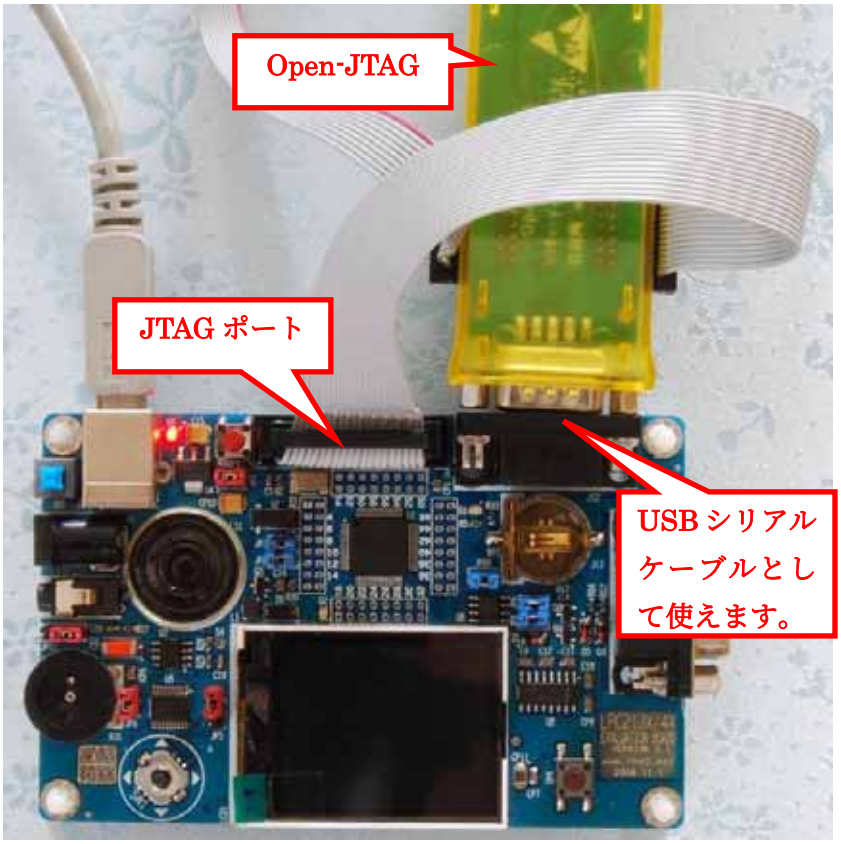


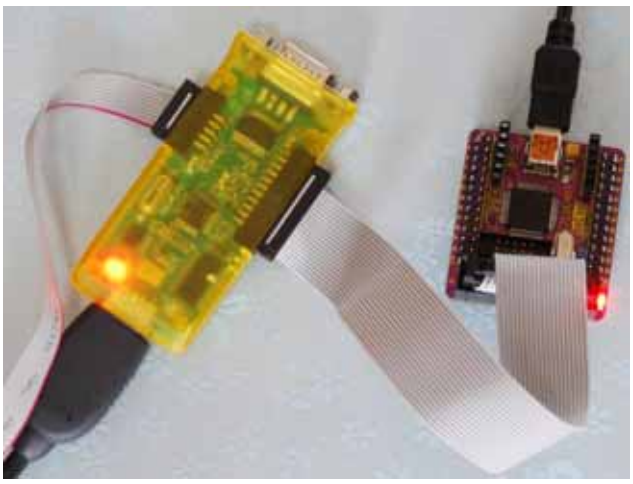
書き込み中の画面です。

実行の時、JP5/JP8/JP9 の設定。

3.4 OpenJTAG で書き込み

Open-JTAG は ARM 用の USB-JTAG エミュレータです。ARM7、ARM9、Cortex-M3, XSCALE に対応、OpenOCD をサポートします。USB-RS232 機能もあります。COM と LPT ポートがないノートパソコンに最適。

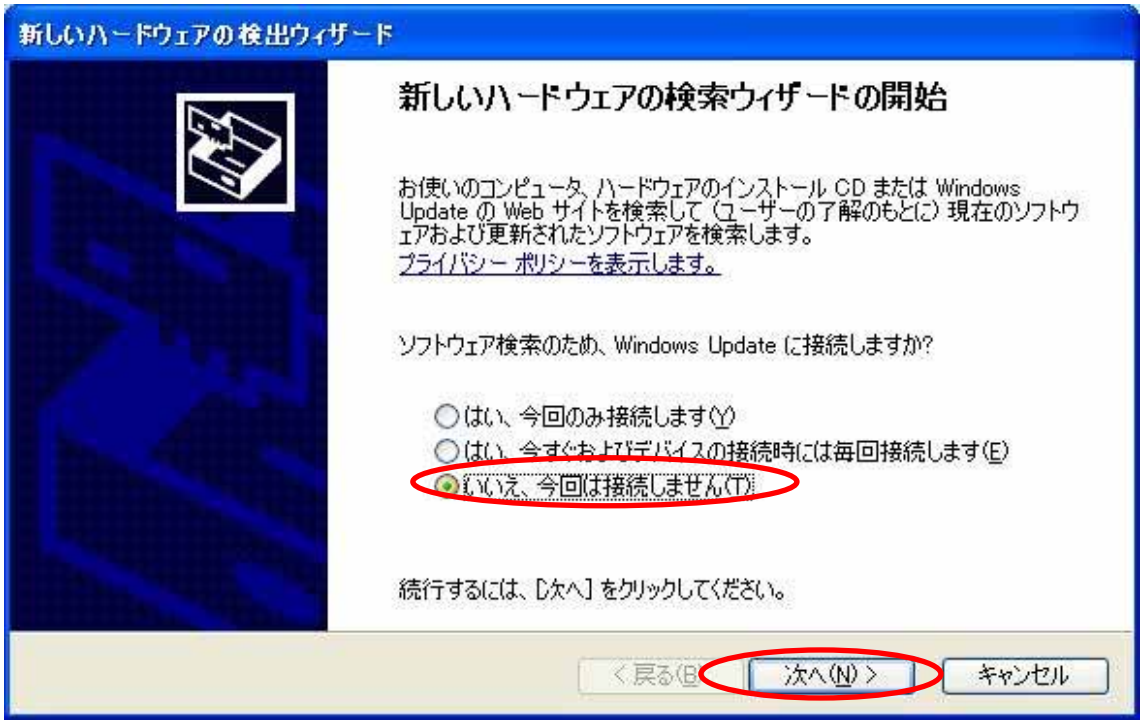


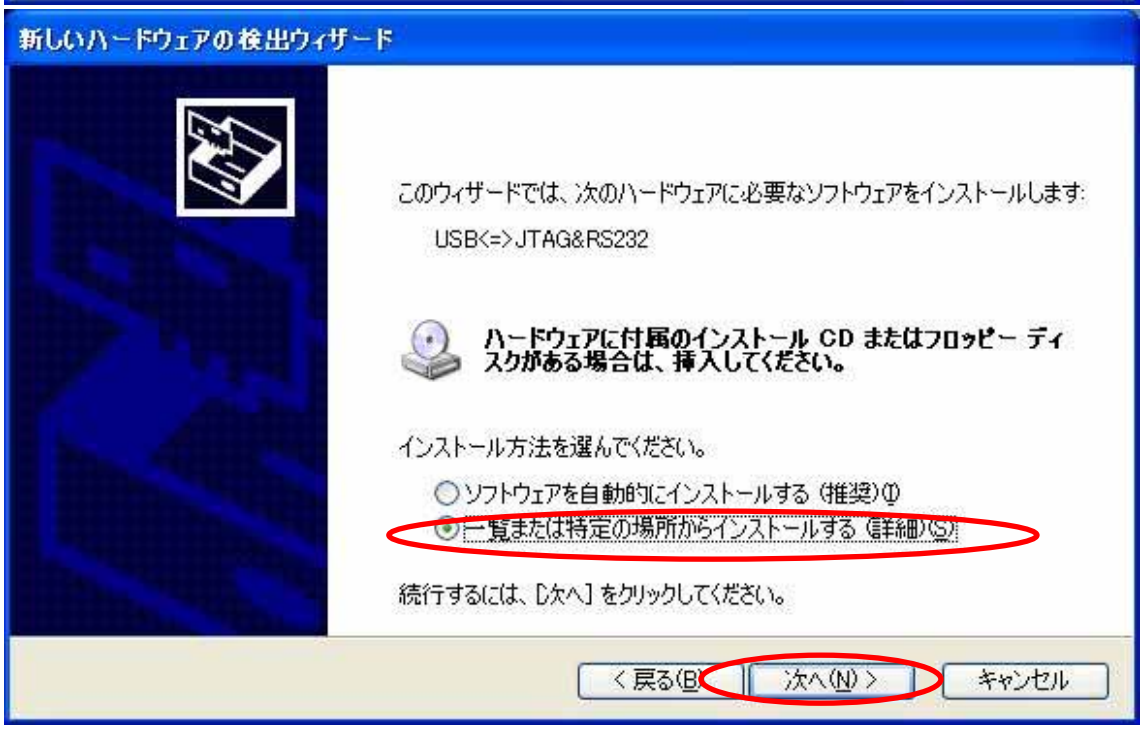


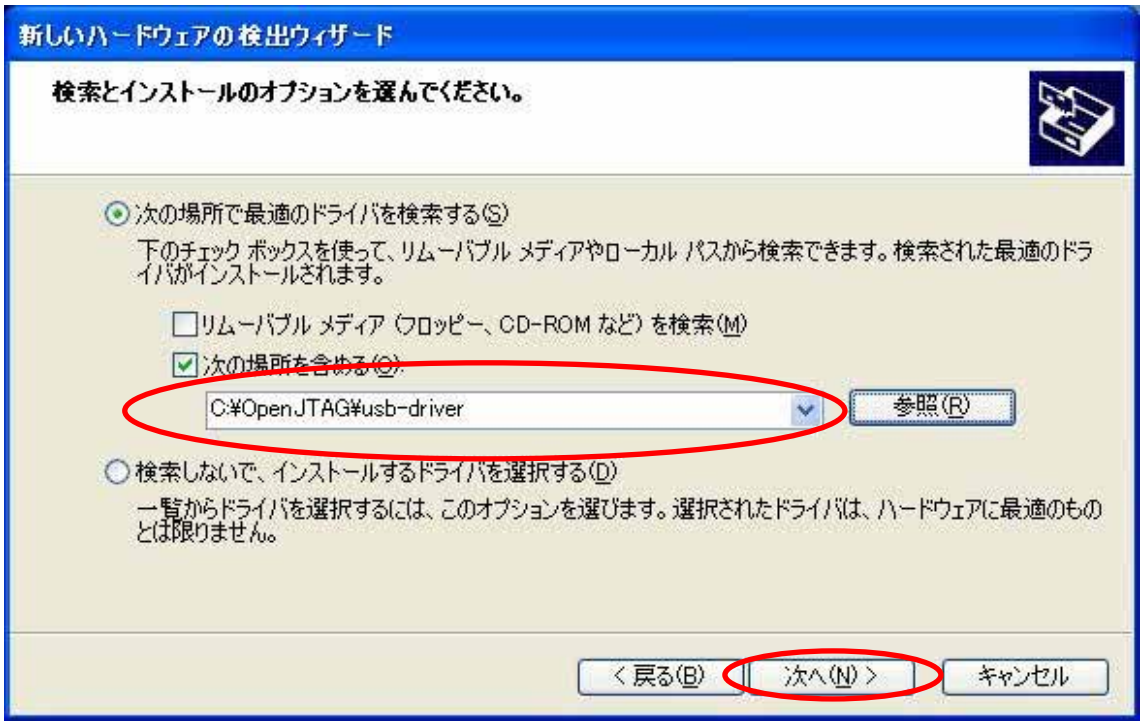
LPC2148 モジュールはシリアルポートがありませんので、open-JTAG で書き込みしかできません。

3.5 OpenJTAG のドライバをインストールする

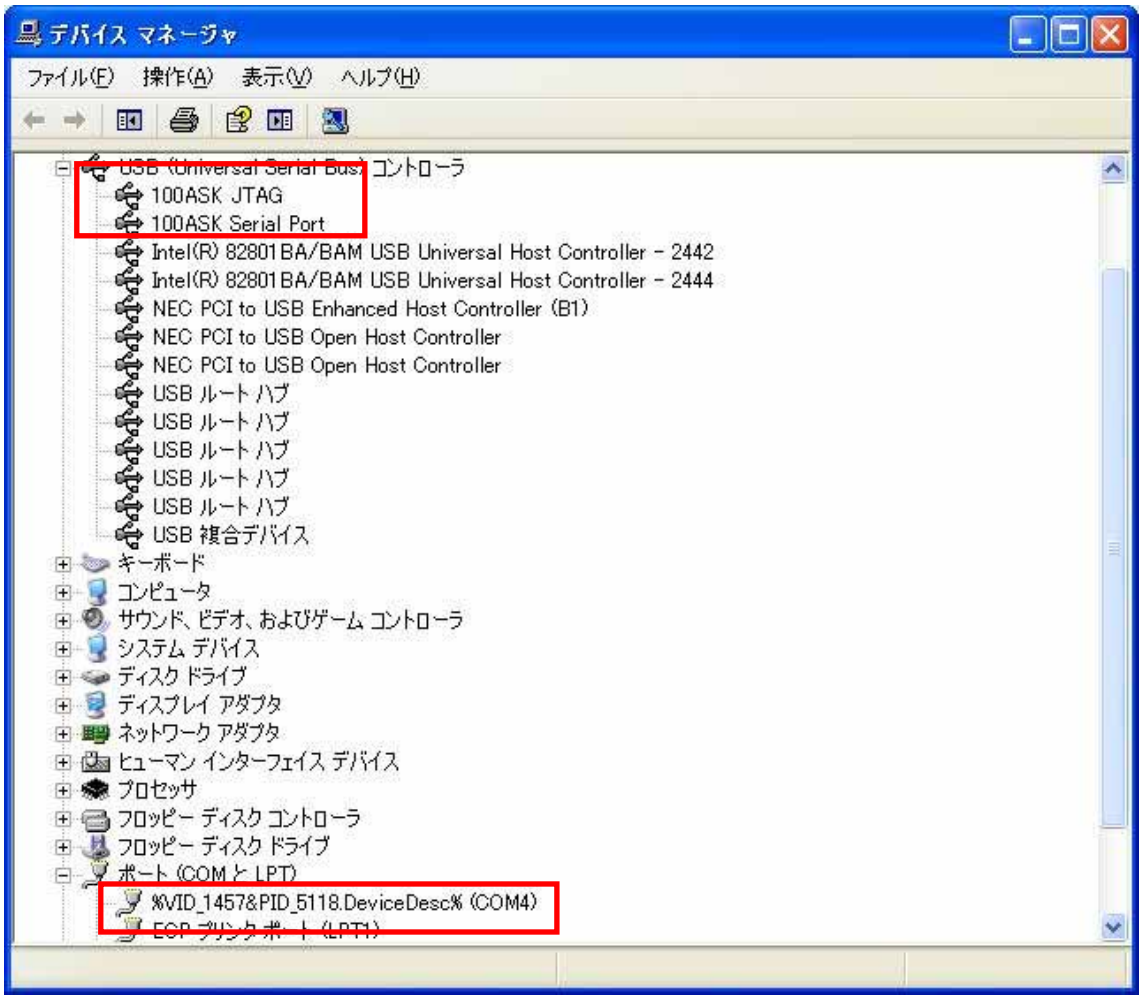
OpenJTAG をパソコンの USB ポートに挿入して、下の通りにドライバをインストールしてください。







USB ドライバのインストールは３回があります。インストール完了すると、デバイスマネージャで三つのデバイスが見えます。



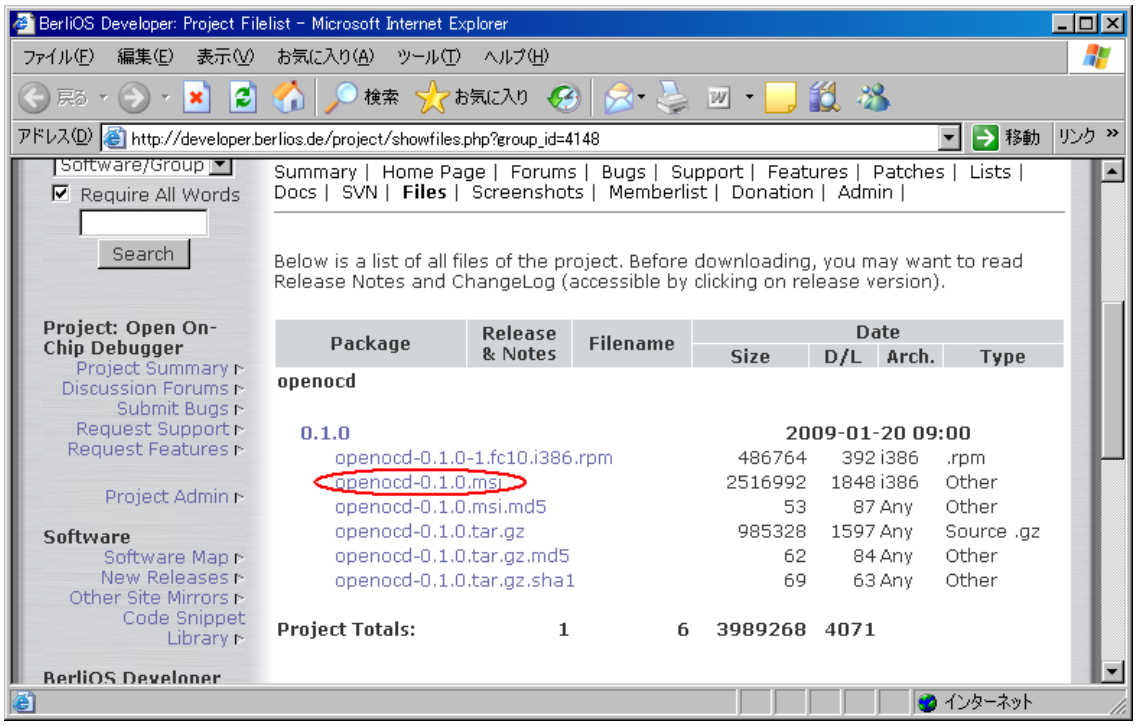
※ OpenJTAG は USB シリアルポートとして使えます。

3.6 OpenOCD のダウンロードとインストール

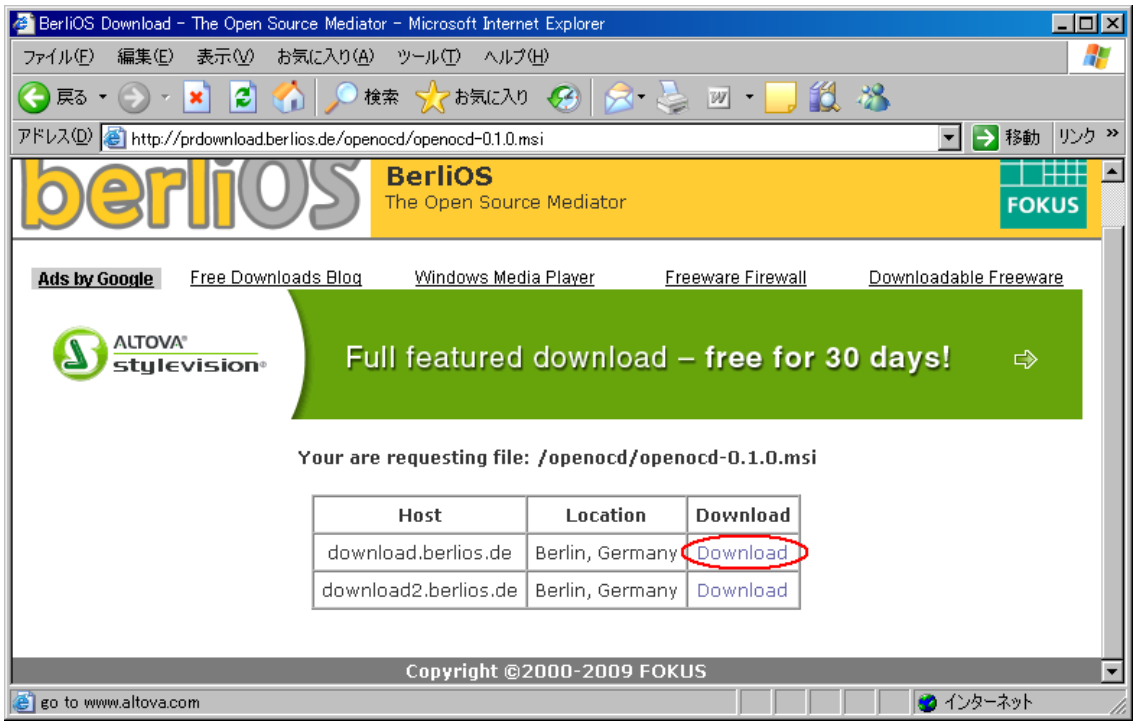
1）下記のリンクをクリック

http://developer.berlios.de/project/showfiles.php?group_id=4148

2）"openocd-0.1.0.msi"をクリック



3）リンクの Download をクリックし Windows 用インストーラパッケージをダウンロードする



5）動作確認のため下記を実行します

1．Open-JTAG をターゲット(LPC2148 基板)とＰＣに接続

２．ターゲットに電源を入れます

３．コマンドプロンプトでディレクトリを移動

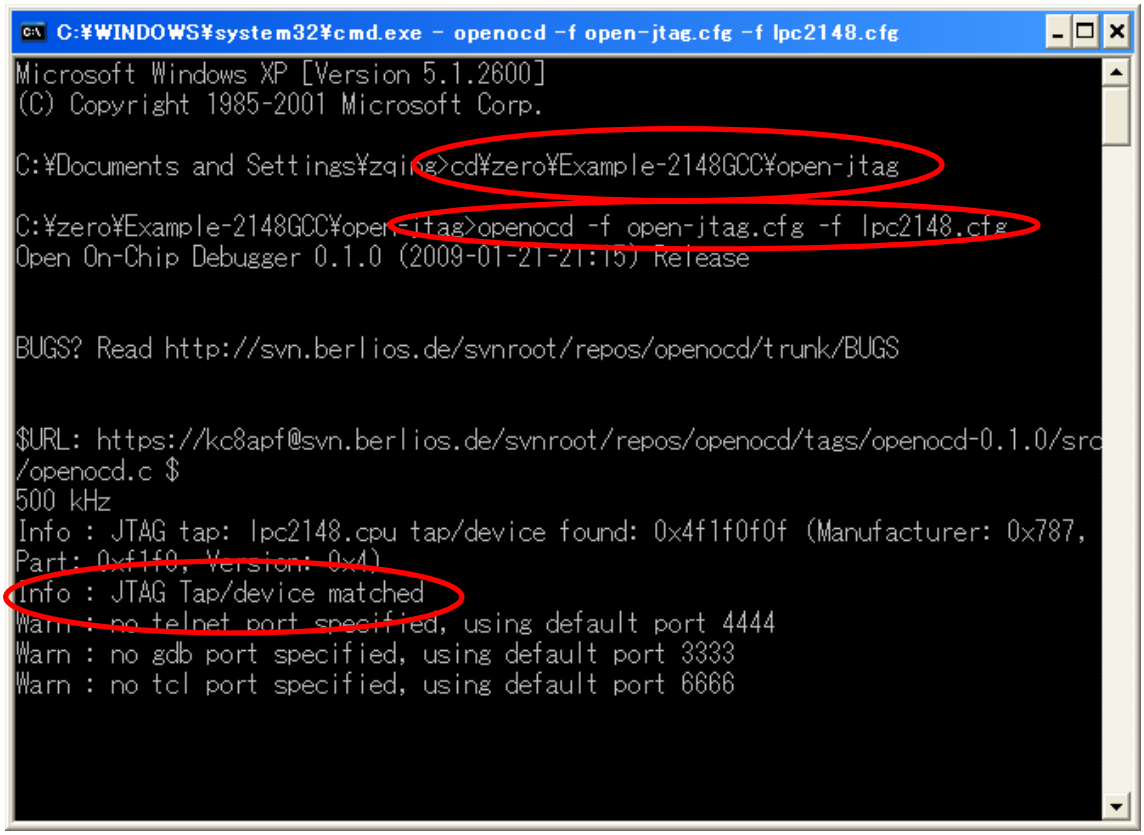
（cd Example-2148GCC¥open-jtag）

４．下記のコマンドを入力します

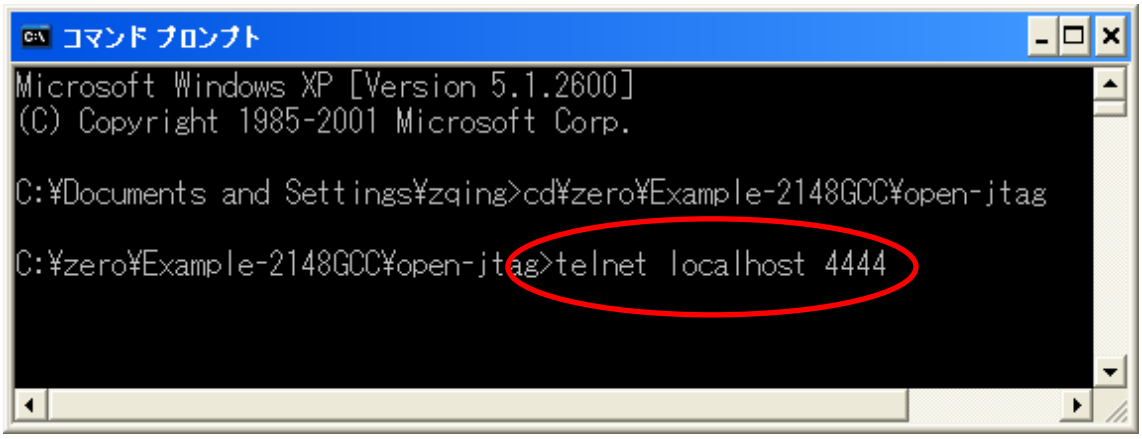
（openocd -f open-jtag.cfg –f lpc2148.cfg）

画面のように"Info：JTAG Tap/device matched"と表示されればＯＫです

（この時点で ARM LPC2148 と通信が出来ています）

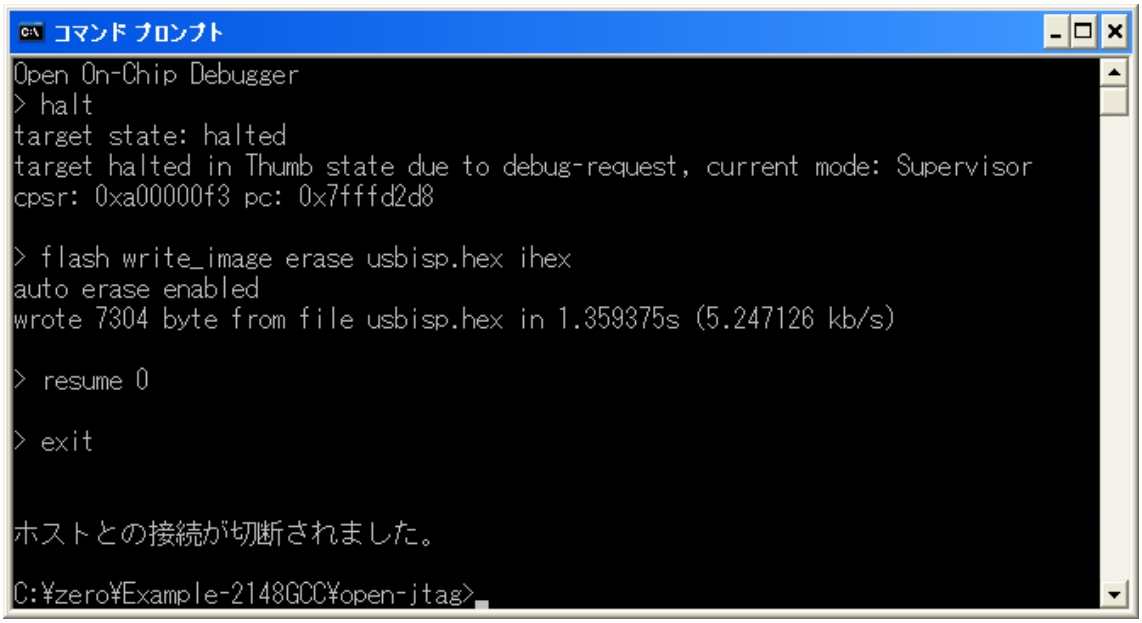


3.7 telnet で書き込み



新コンソールで telnet localhost 4444 コマンドを入力します。telnet で下記のコマンドを入力します。

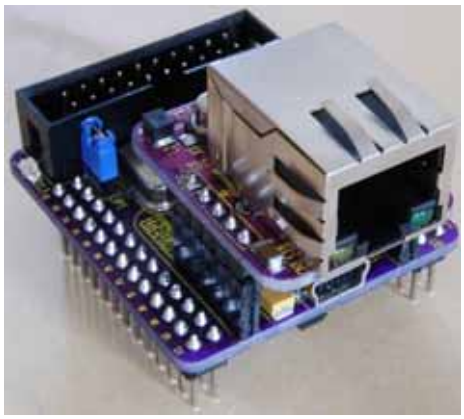
halt	ターゲットを停止
flash write_image erase Tetris.hex ihex	*.hex ファイルを書き込み
resume 0	実行
exit	telnet をクロス

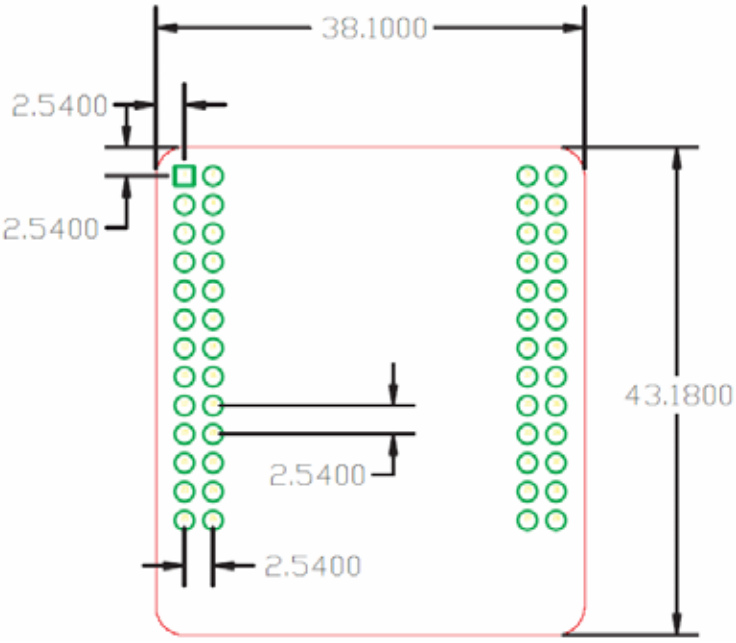


3.8 GCC サンプルの紹介

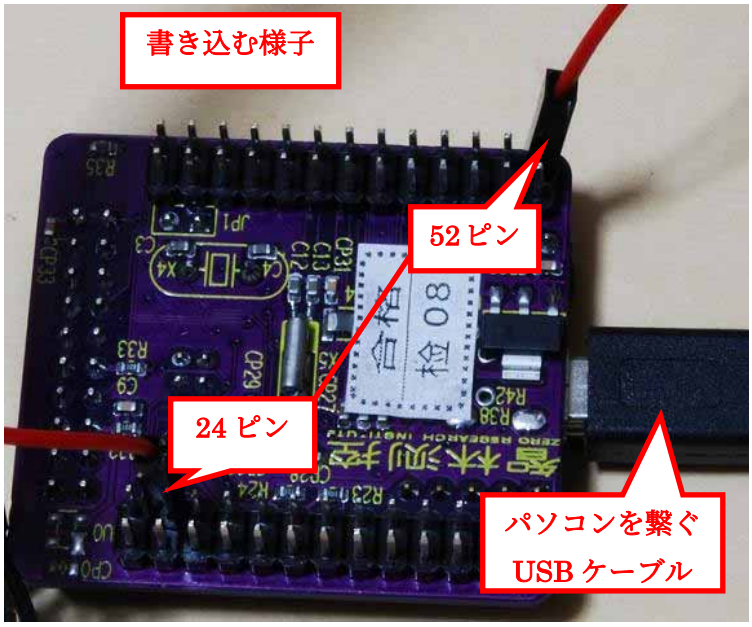
1. LED	LED 点灯
2. GPIO-IN	入力実験
3. PWM	PWM で LCD のバックライトを点灯
4. LCD	LCD で英語を表示
5. LCD-Japanese	日本語を表示
6. LCD_BMP	ピクチャを表示
7. RTC	RTC 実験
8. IIC	IIC の EEPROM のライト・リード
9. ADC	ADC 実験
10.DAC	DAC 実験
11.UART_FIFO	UART のポーリング
12.UART_INT	UART の割り込み
13.Timer0-INT	タイマーの割り込み
14.EXTINT	外部の割り込み
15.AD_INT	ADC の割り込み
16.Sound	DAC で音声
17.SD-FatFs	SD メモリのデモ
18.USBtarget	
examples/msc.hex	LPC2148 経由 SD メモリが USB メモリとして
examples/serial.hex	LPC2148 が USB シリアルポートとして使える
ドライバは USBtarget/examples/usbser.inf です。	
19.IAP	プログラムで Flash を更新
20.Webserver	ウェブサーバ、イーサネット ENC28J60 用
21.Tetris	デフォルトのサンプル(テトリス)
22.nRF24L01	微弱無線モジュール(nRF24L01)の通信実験

3.9 USB ダウントローダ





LPC2148 モジュールには USB ダウンローダが書き込まれています。パソコン上で開発したユーザ・プログラムを USB 経由で LPC2148 のフラッシュ・メモリに書き込むことができます。



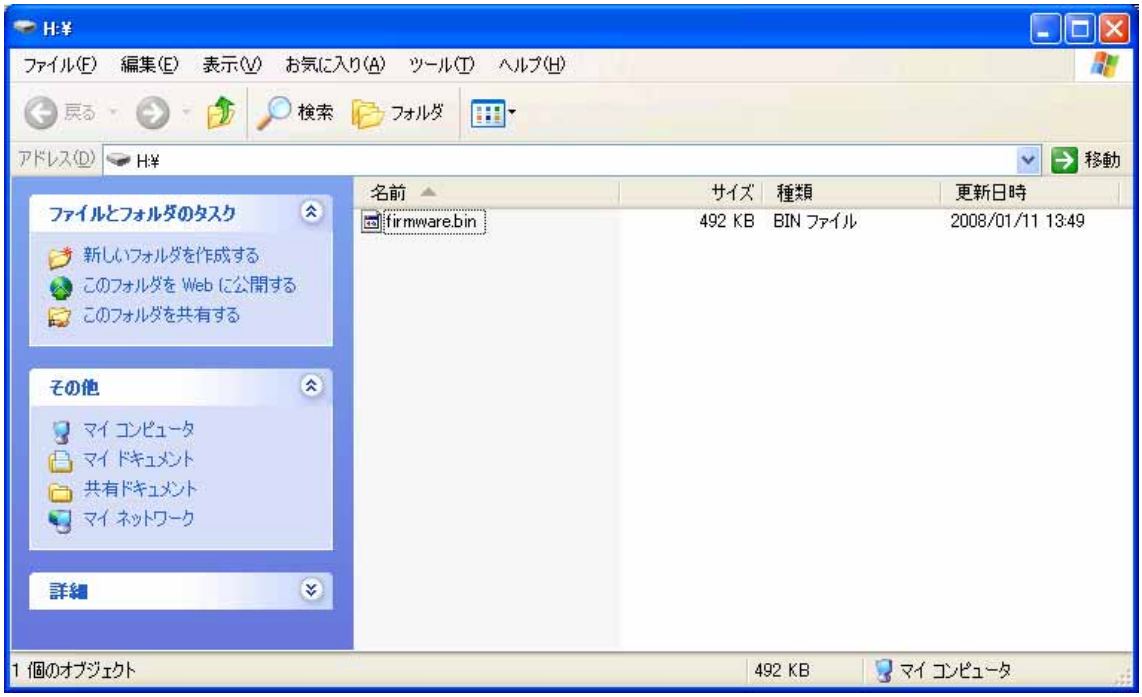
USB ダウンローダ領域は 0~0x1FFFF、ユーザ・プログラム領域は 0x2000 からです。

USB ダウンローダはソースも含めて公開されています。Example/USB ISP は USB ダウンローダのプロジェクトです。

この仕組みによって LPC2148 モジュールは、JTAG デバッグがなくても USB 経由でアプリケーション・プログラムの書き換えが可能です。ユーザはパソコンと LPC2148 モジュールだけで ARM プロセッサのシステムを開発できます。

誤って 0 番地に上書きすると、USB ブートローダ壊れてしまうので、注意が必要です。この修復には JTAG デバッグが必要になります。

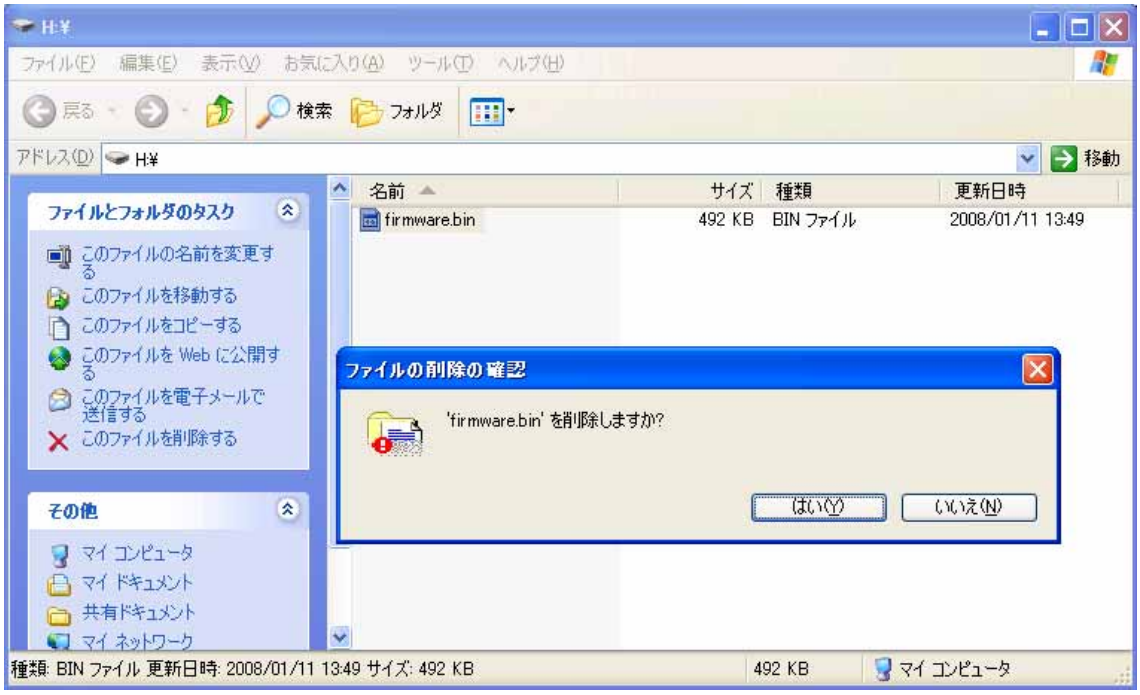
LPC2148 モジュールが起動の時、裏面の拡張ヘッダの 24,52 ピンをショットすれば、USB ダウンローダモードに入ります。



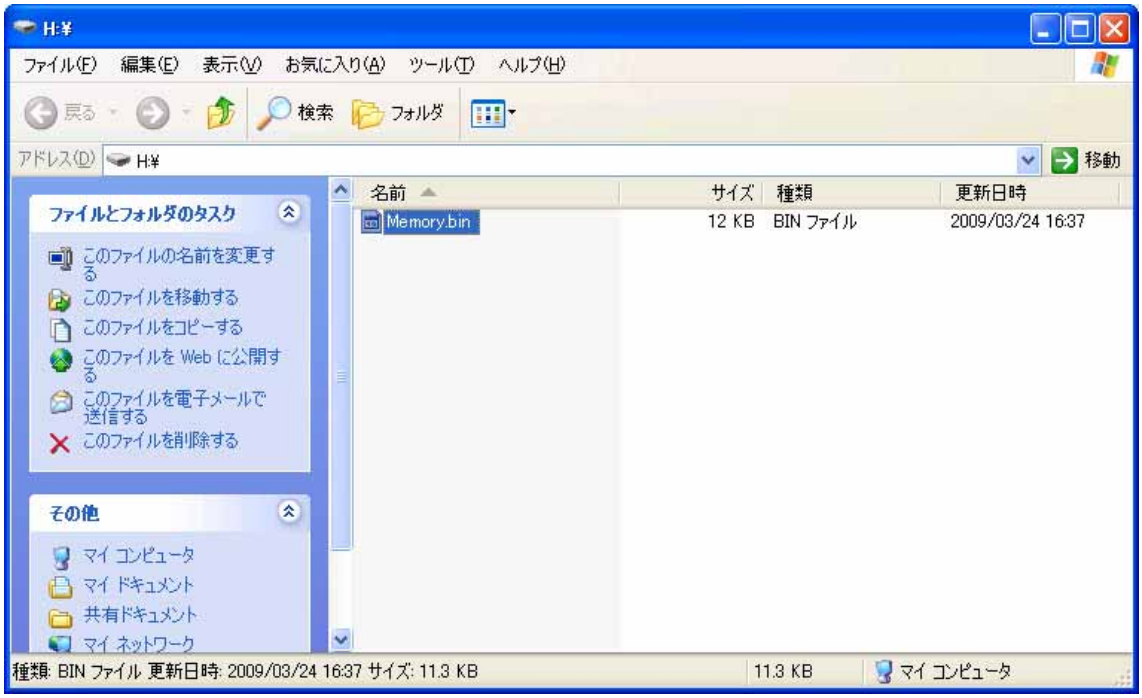
パソコンは LPC2148 モジュールを USB メモリとして認識します。LPC2148 の USB メモリに「firmware.bin」というファイルがあります。サイズは 492KB です。このファイルはユーザが使える FlashROM のイメージです。使えるサイズは 492KB です。

ユーザ・プログラムを書き換える手順：

まず、「firmware.bin」というファイルを削除してください。



ユーザ・プログラムのイメージファイル*.bin を USB メモリにコピーしてください。



LPC2148 の拡張ヘッダの 24,52 ピン間の短絡線を外して再起動すると、ユーザ・プログラムを実行します。

第四章 クロス開発環境

4.1 GCC ツールチェーン

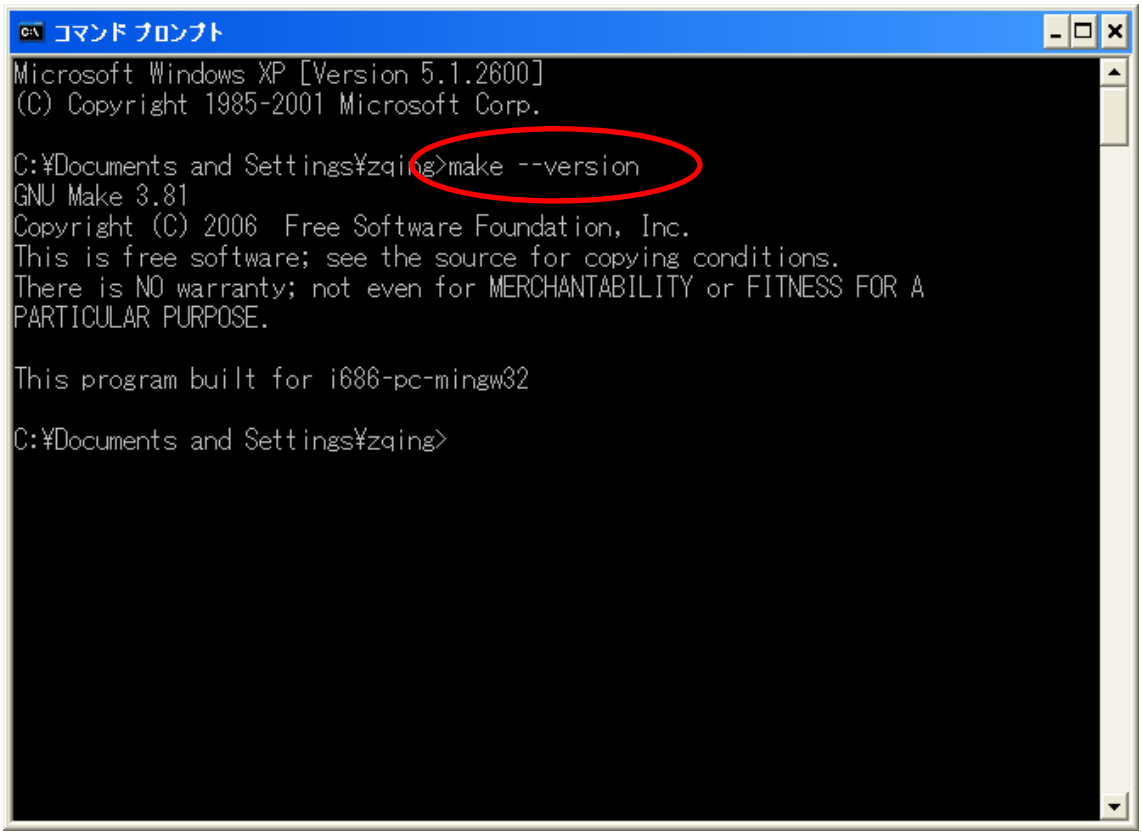
各種ユーティリティ：

http://www.yagarto.de/download/yagarto/yagarto-tools-20070303-setup.exe

GCC ツールチェーン

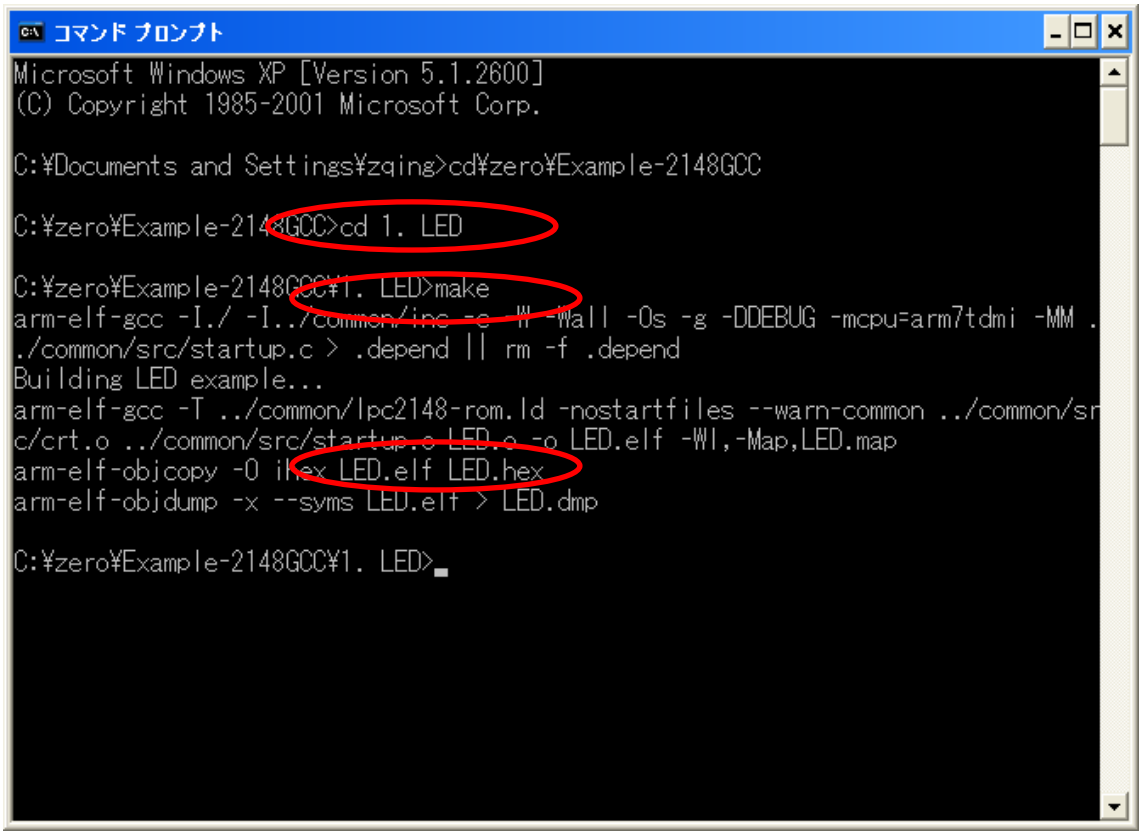
http://sourceforge.net/projects/yagarto/files/YAGARTO%20for%20Windows/yagarto-bu-2.19.1_gcc-4.3.3-c-c%2B%2B_nl-1.17.0_gi-6.8.50_20090329.exe/download

インストールが出来たら make の確認をするためコマンドプロンプトを起動し、右記のコマンドを入力します(make --version)。画面に下記のメッセージが出てくればＯＫです。



サンプルのコンパイル：

1. コマンドプロンプトでディレクトリを移動
   （cd Example-2148GCC¥1. LED）
2. 下記のコマンドを入力します
   （make）

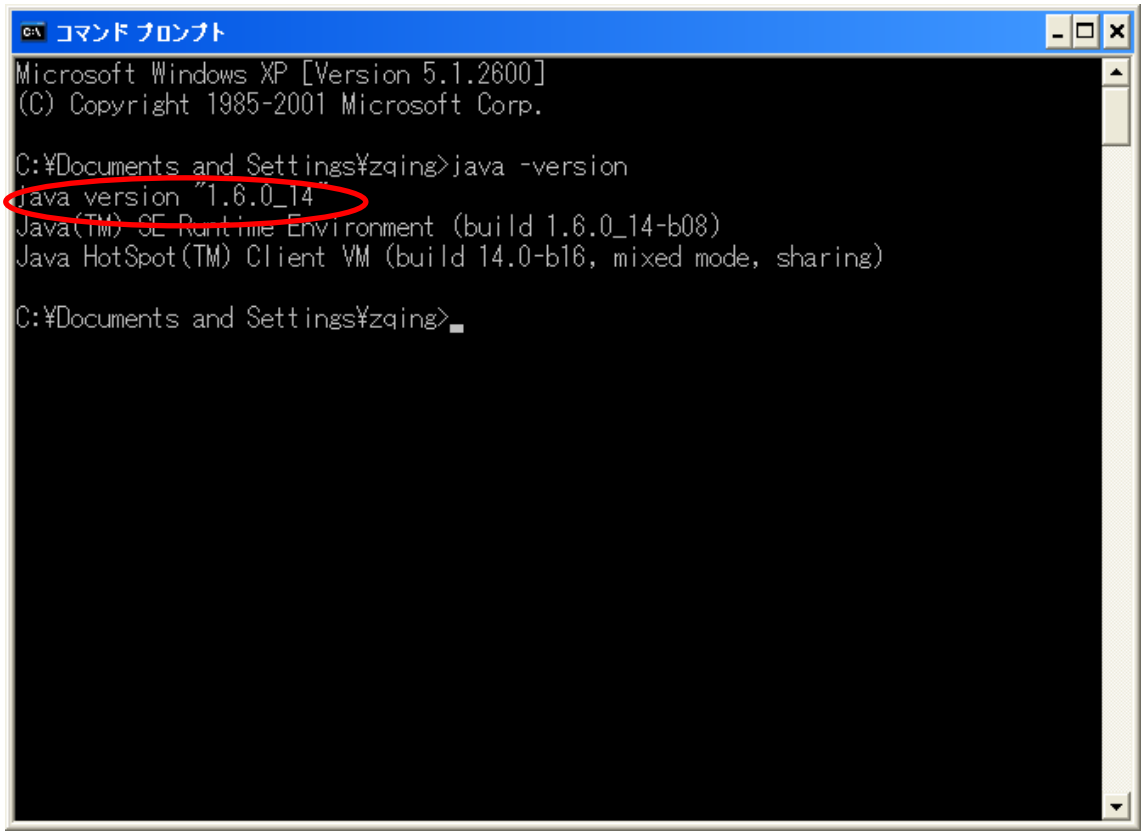


コンパイル成功したら、*.hex ファイルを生成させます。生成された*.hex ファイルを LPC2148 に書き込みましょう。

4.2 Integrated Development Environment(Eclipse)

JRE バージョン確認：

確認コマンド： java -version



JRE がなければ、あるいは 1.4.2 以下なら、JRE のインストールが必要です。

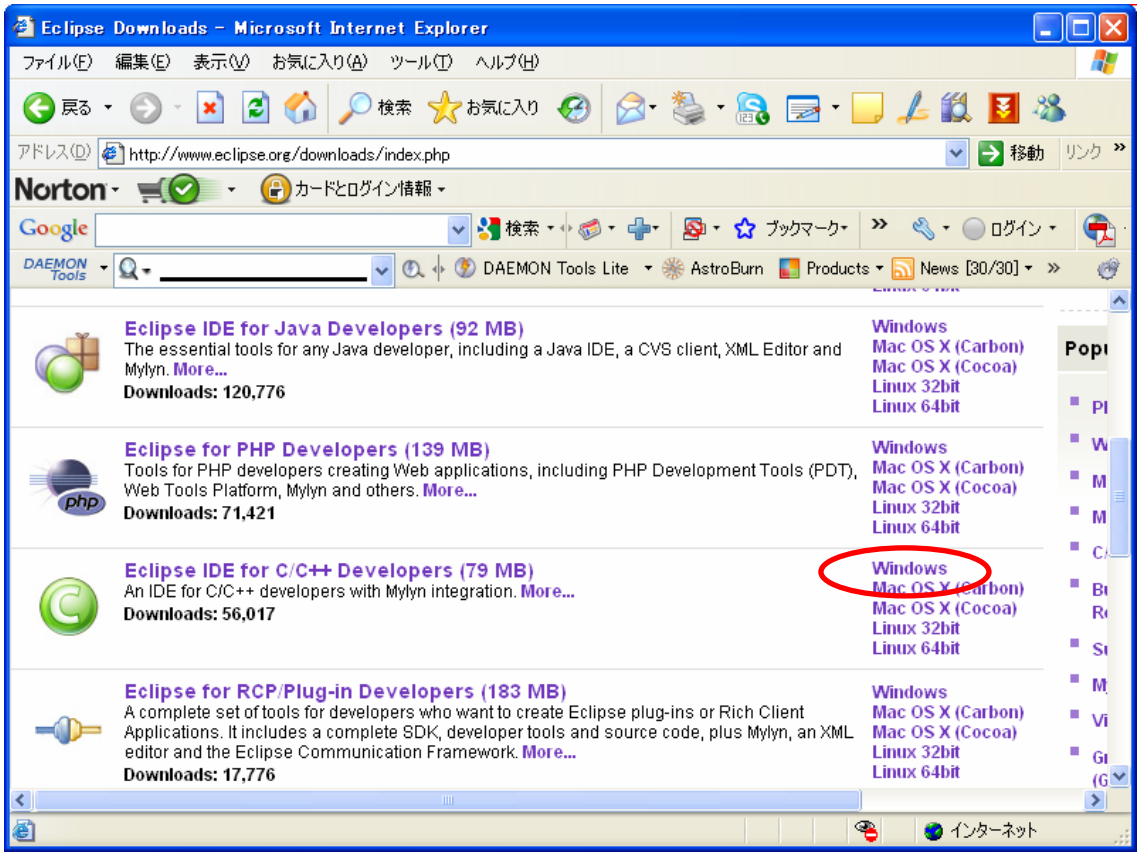
http://java.sun.com/javase/downloads/index.jsp

Eclipse のインストール：

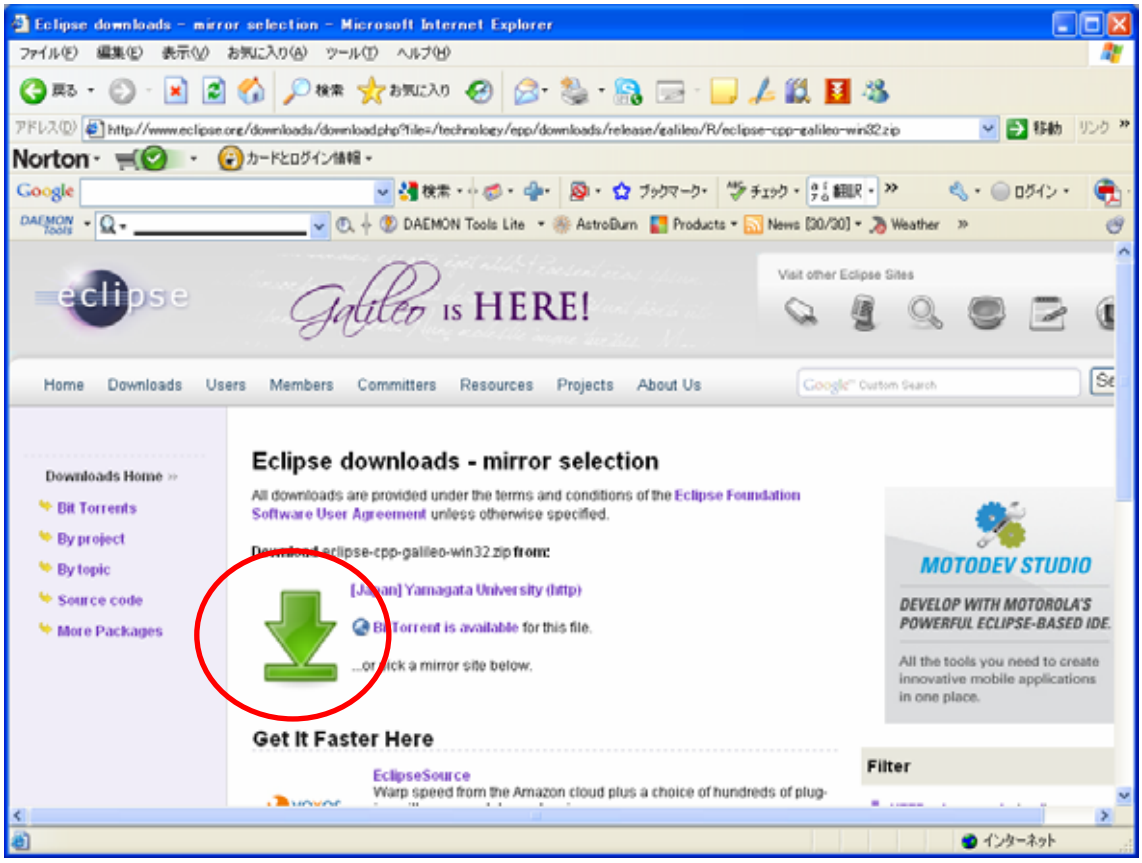
１）下記のリンクをクリック

http://www.eclipse.org/downloads/index.php

２）Eclipse IDE for C/C++ Debelopers(79MB)の Windows をクリック

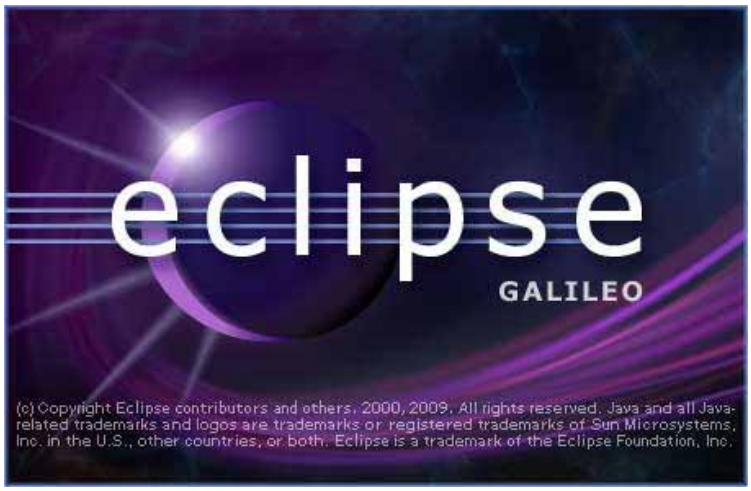

Eclipse Downloads - Microsoft Internet Explorer
ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
戻る 検索 お気に入り
アドレス(D) http://www.eclipse.org/downloads/index.php
移動
Norton
カードとログイン情報
Google
検索 ブックマーク ログイン
DAEMON Tools
DAEMON Tools Lite AstroBurn Products News [30/30]
Eclipse IDE for Java Developers (92 MB)
The essential tools for any Java developer, including a Java IDE, a CVS client, XML Editor and Mylyn. More...
Downloads: 120,776
Windows
Mac OS X (Carbon)
Mac OS X (Cocoa)
Linux 32bit
Linux 64bit
Eclipse for PHP Developers (139 MB)
Tools for PHP developers creating Web applications, including PHP Development Tools (PDT), Web Tools Platform, Mylyn and others. More...
Downloads: 71,421
Windows
Mac OS X (Carbon)
Mac OS X (Cocoa)
Linux 32bit
Linux 64bit
Eclipse IDE for C/C++ Developers (79 MB)
An IDE for C/C++ developers with Mylyn integration. More...
Downloads: 56,017
Windows
Mac OS X (Cocoa)
Linux 32bit
Linux 64bit
Eclipse for RCP/Plug-in Developers (183 MB)
A complete set of tools for developers who want to create Eclipse plug-ins or Rich Client Applications. It includes a complete SDK, developer tools and source code, plus Mylyn, an XML editor and the Eclipse Communication Framework. More...
Downloads: 17,776
Windows
Mac OS X (Carbon)
Mac OS X (Cocoa)
Linux 32bit
Linux 64bit
インターネット

3）画面の下矢印をクリックしダウンロード

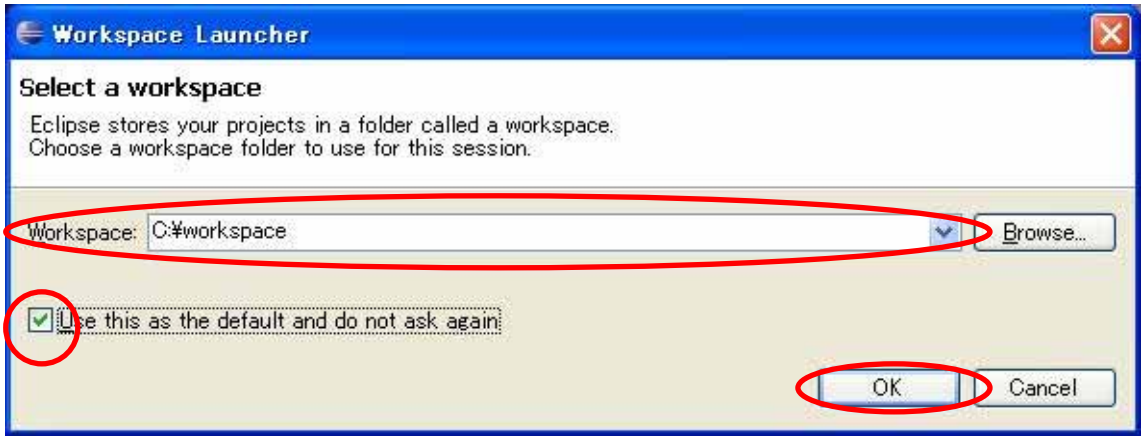


4）ダウンロードしたファイル"eclipse-cpp-**galileo**-win32.zip"を解凍し、そのなかの”eclipse"フォルダを適当な場所(C:¥eclipse)へ移動する。

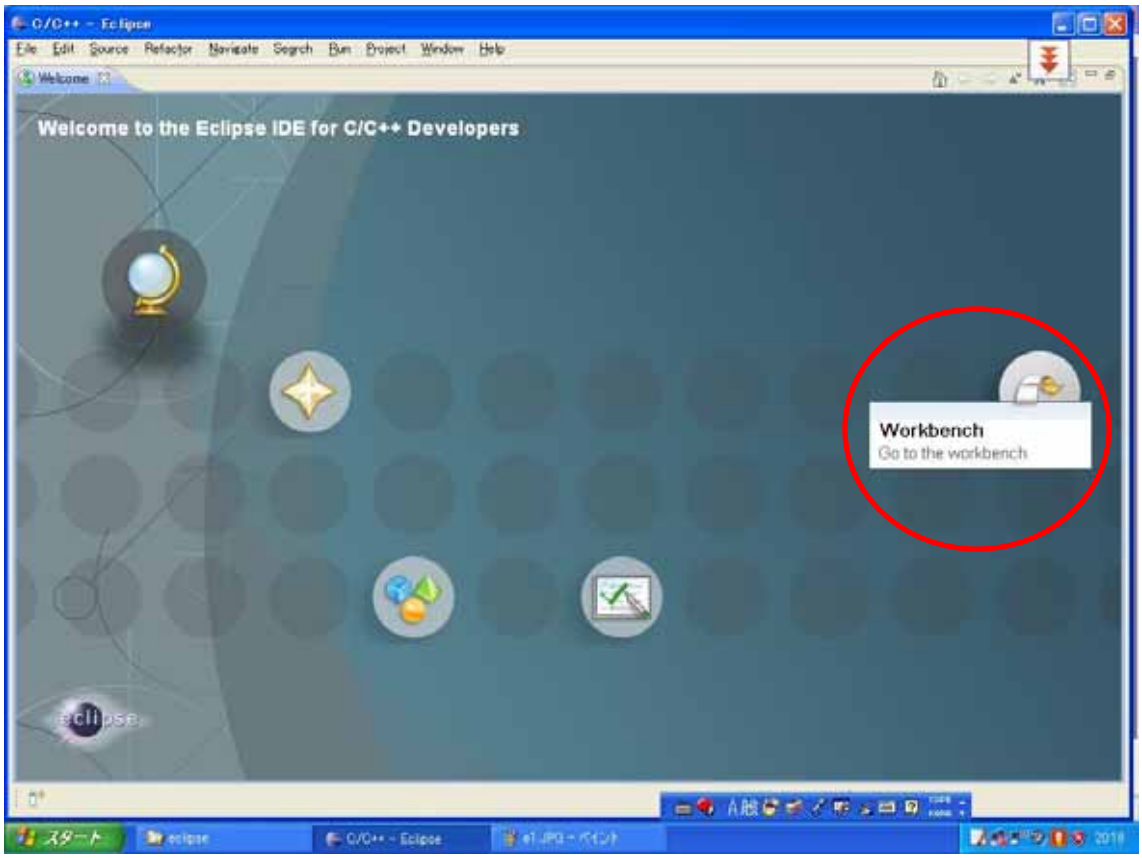
5）Eclipse を起動する。

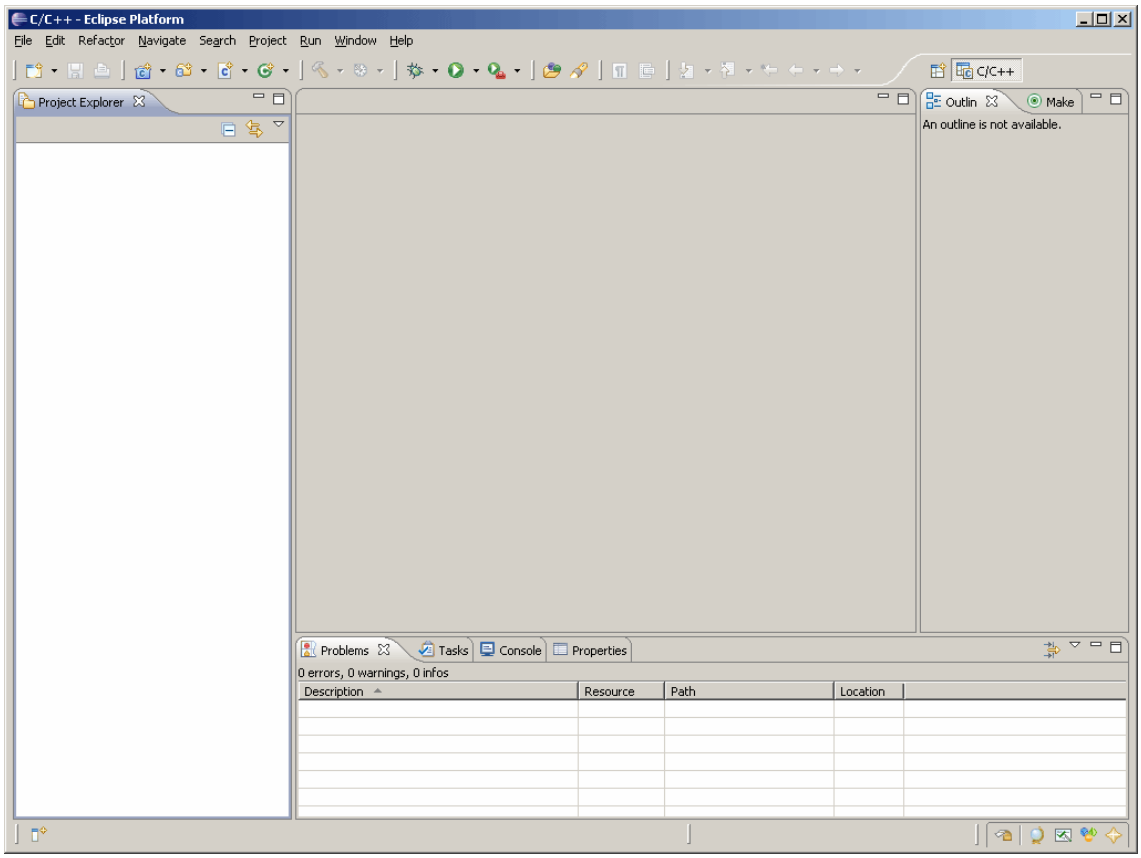


6）最初に Workspace の場所を聞いてきます。適当なフォルダに変更してください。



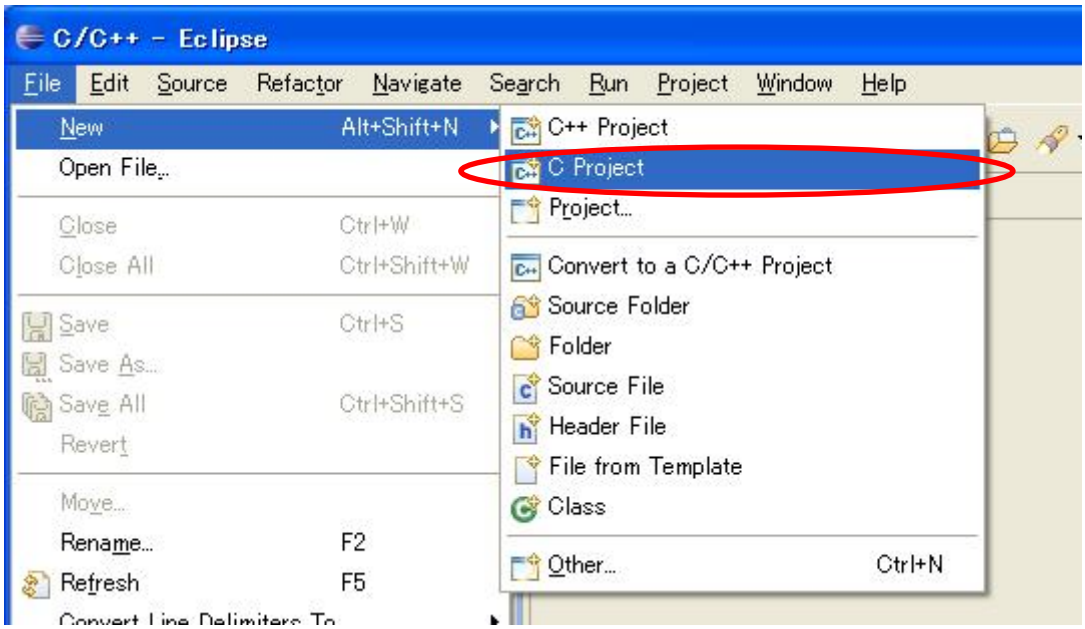
画面の Workbench をクリックします。



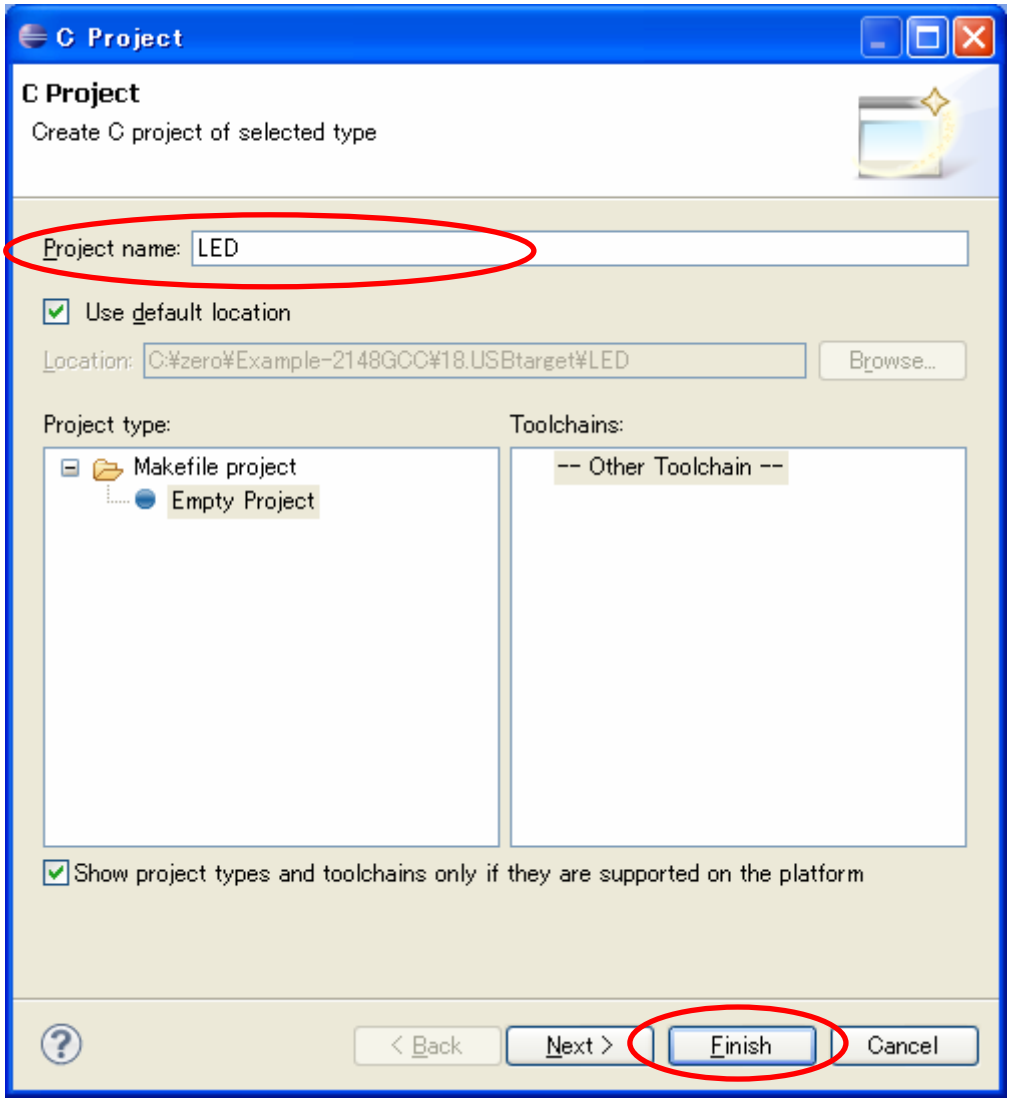


4.3 プロジェクトを作る

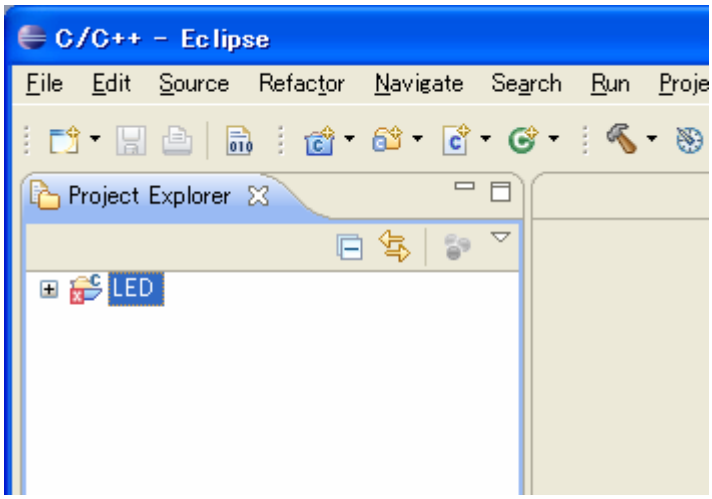
新規プロジェクトを作成するため"File"→"New"→"C Project"を選択します



プロジェクト名を聞かれるので適当な名前(LED)を入力し Finish ボタンを押します。

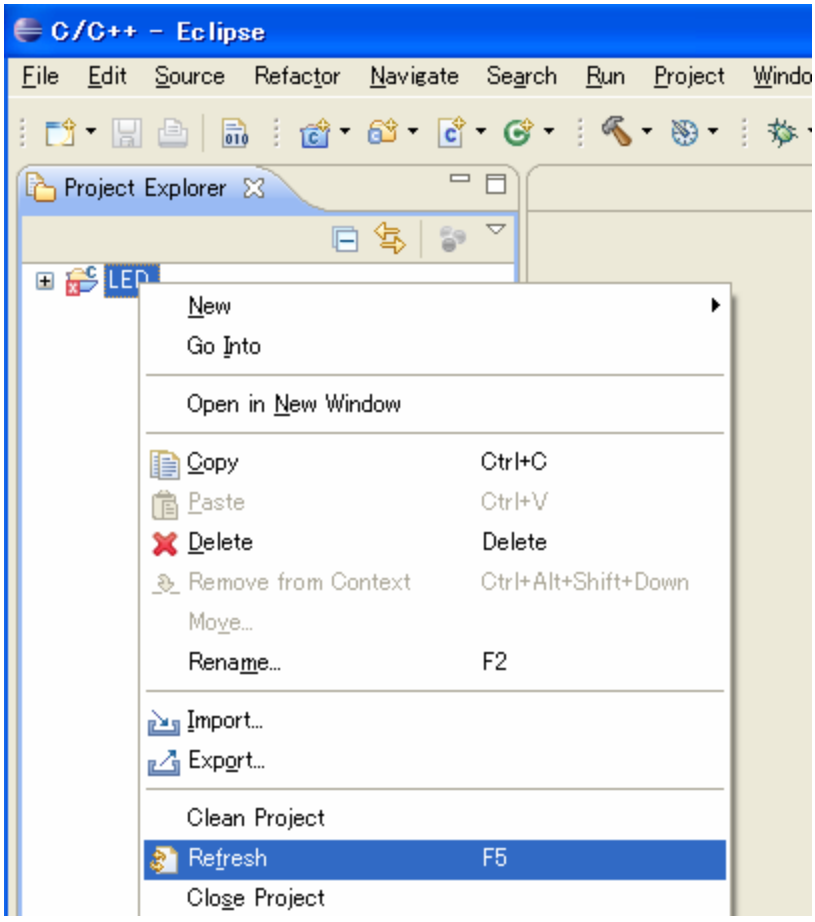


Project Explorer にプロジェクト LED が追加されましたが中身が何もないので、"×"がついています。

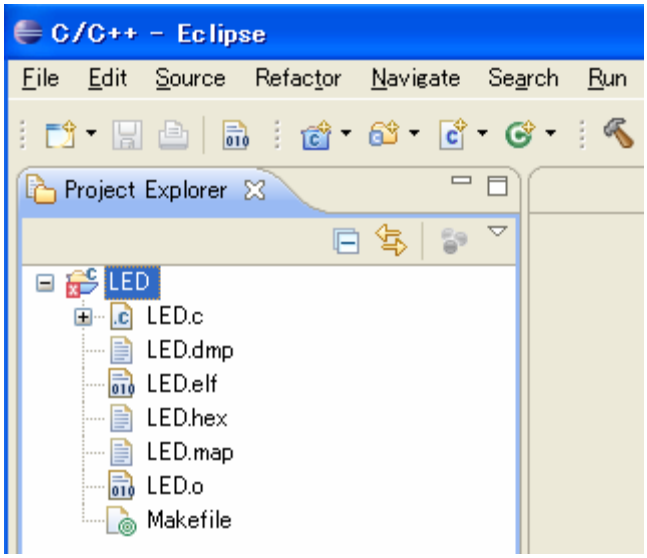


サンプルのフォルダ Example-2148GCC¥1. LED のなかのファイルを"C:¥workspace¥LED"にコピーしてください。

Eclipse の ” File"→"Refresh"を選択します。

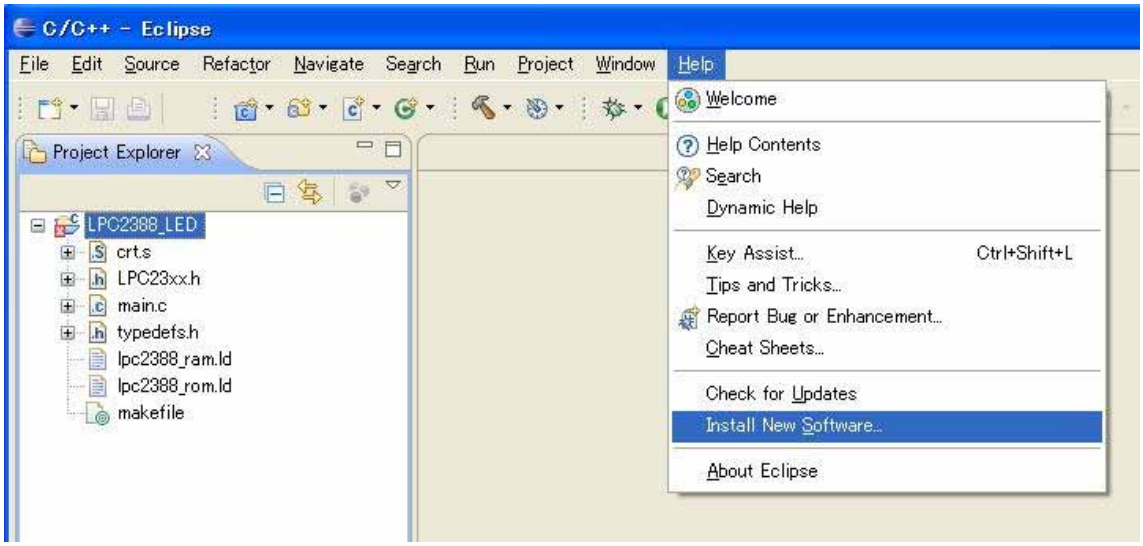


Project Explorer の"LED"プロジェクトの左にある＋をクリックするとファイルの一覧が表示されます。

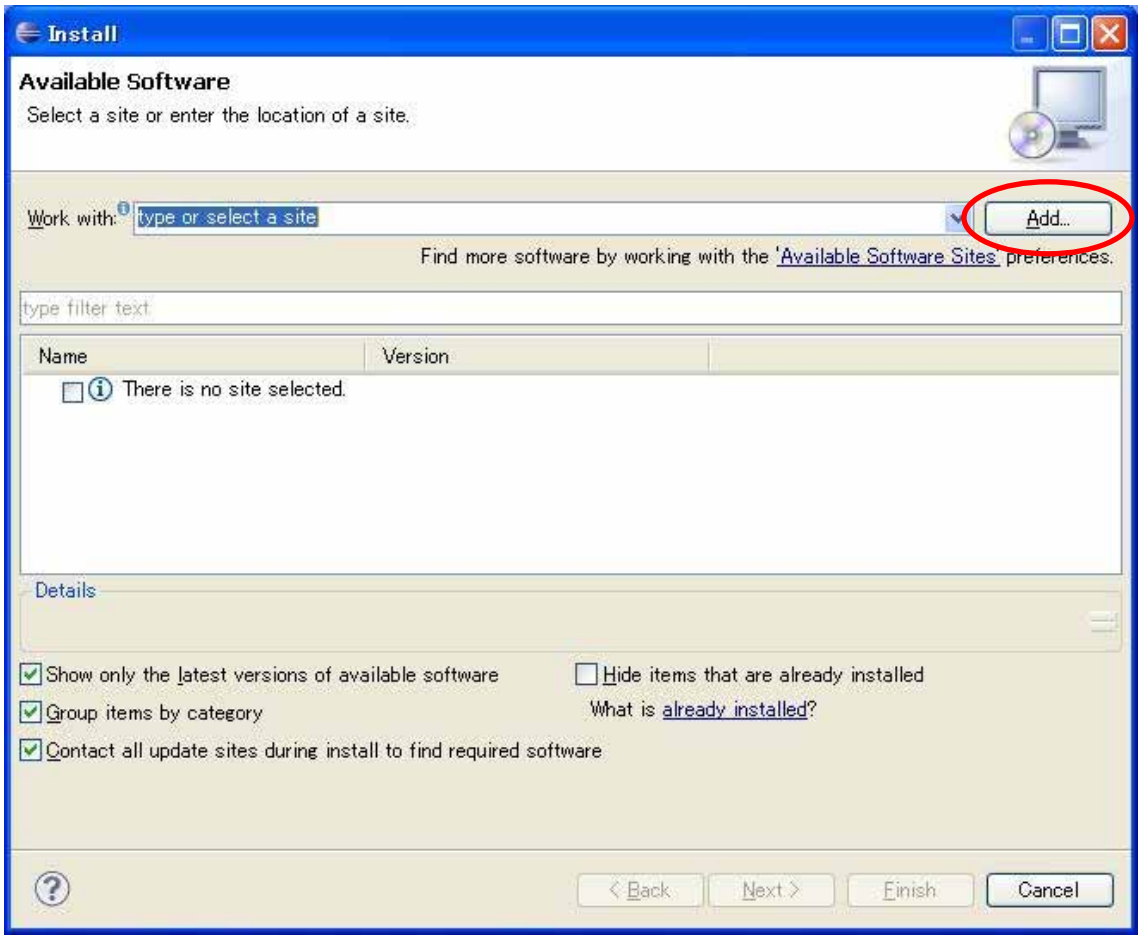


4.4 Eclipse プラグイン(Zylin Embedded CDT)インストール

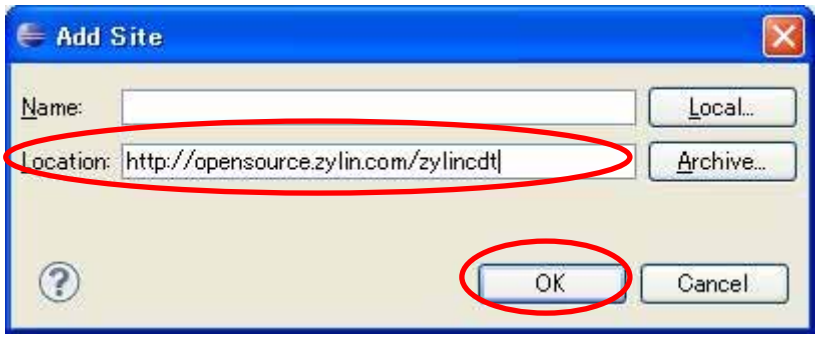
Eclipse の"Help"→"Install New Software"を選択します



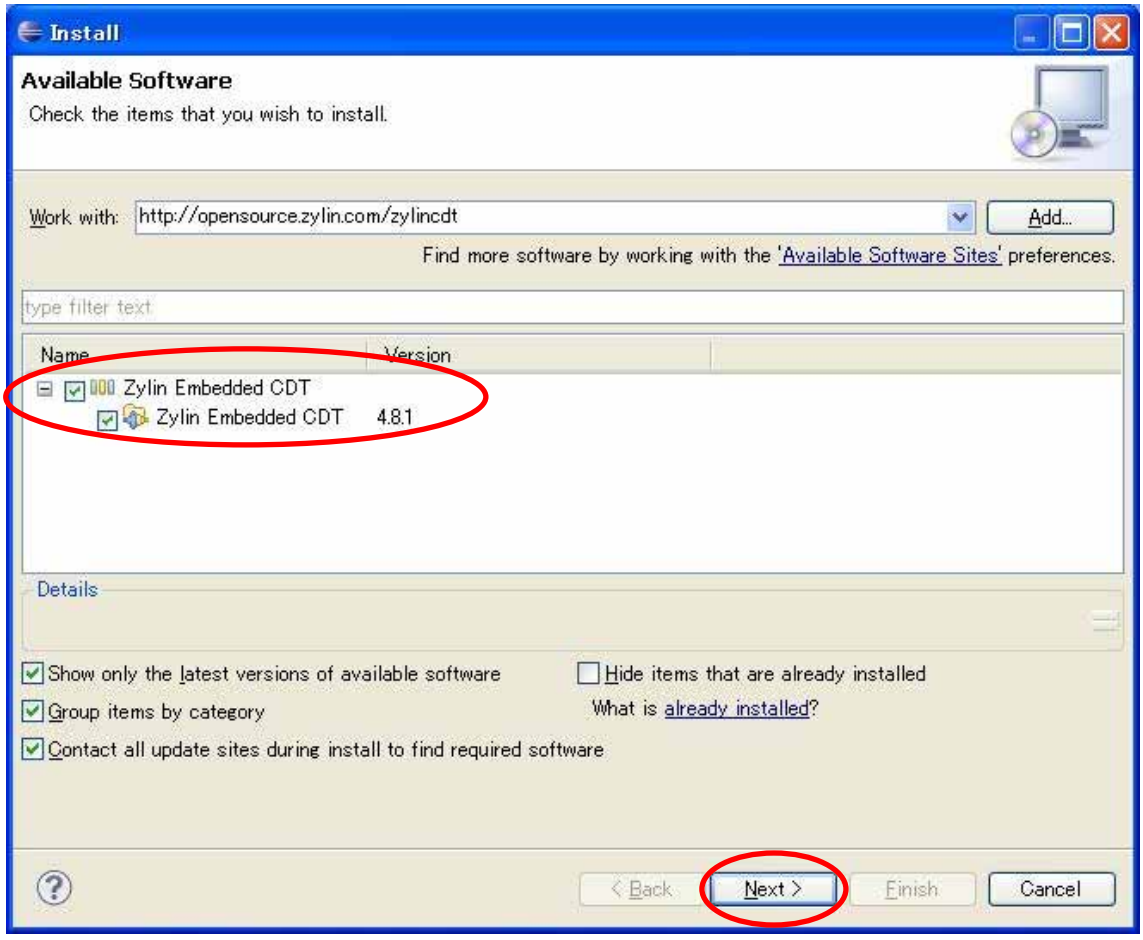
Add ボタンを押します。

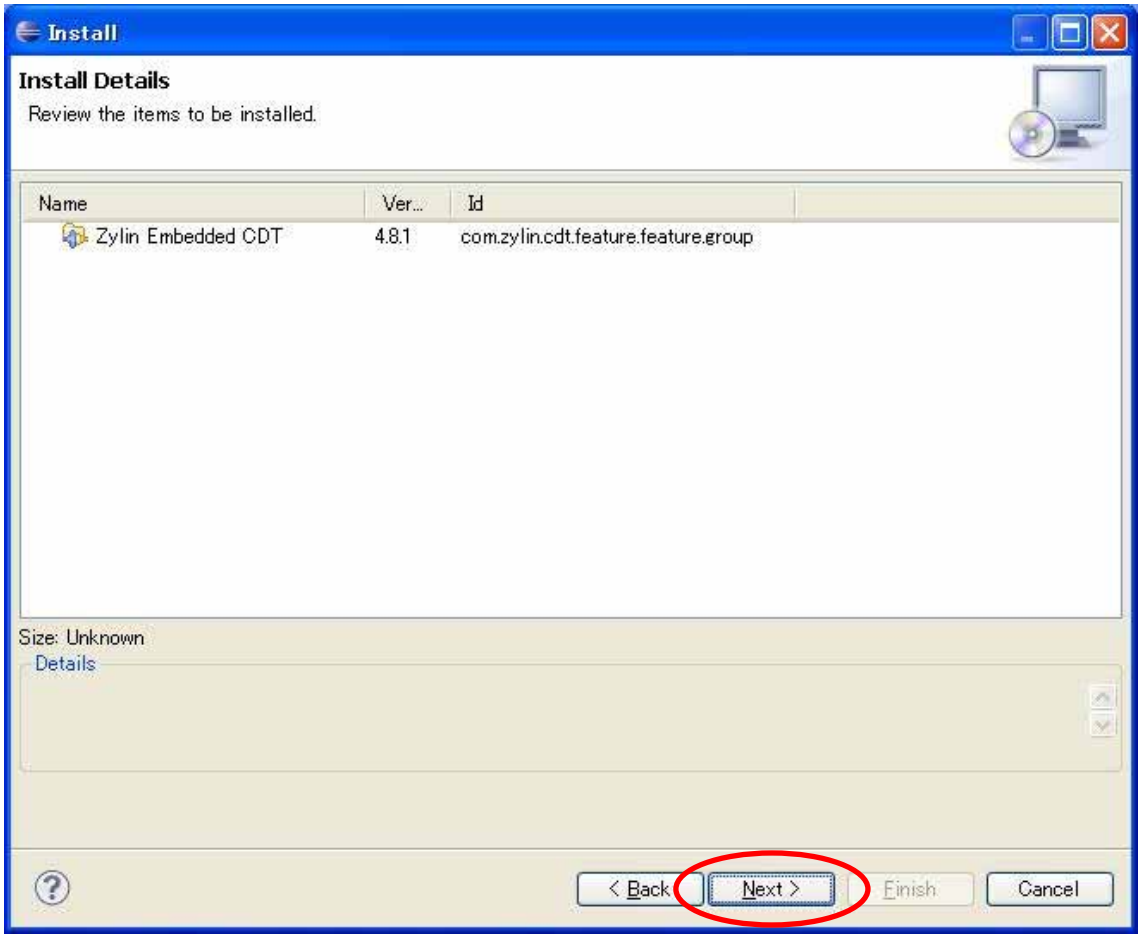


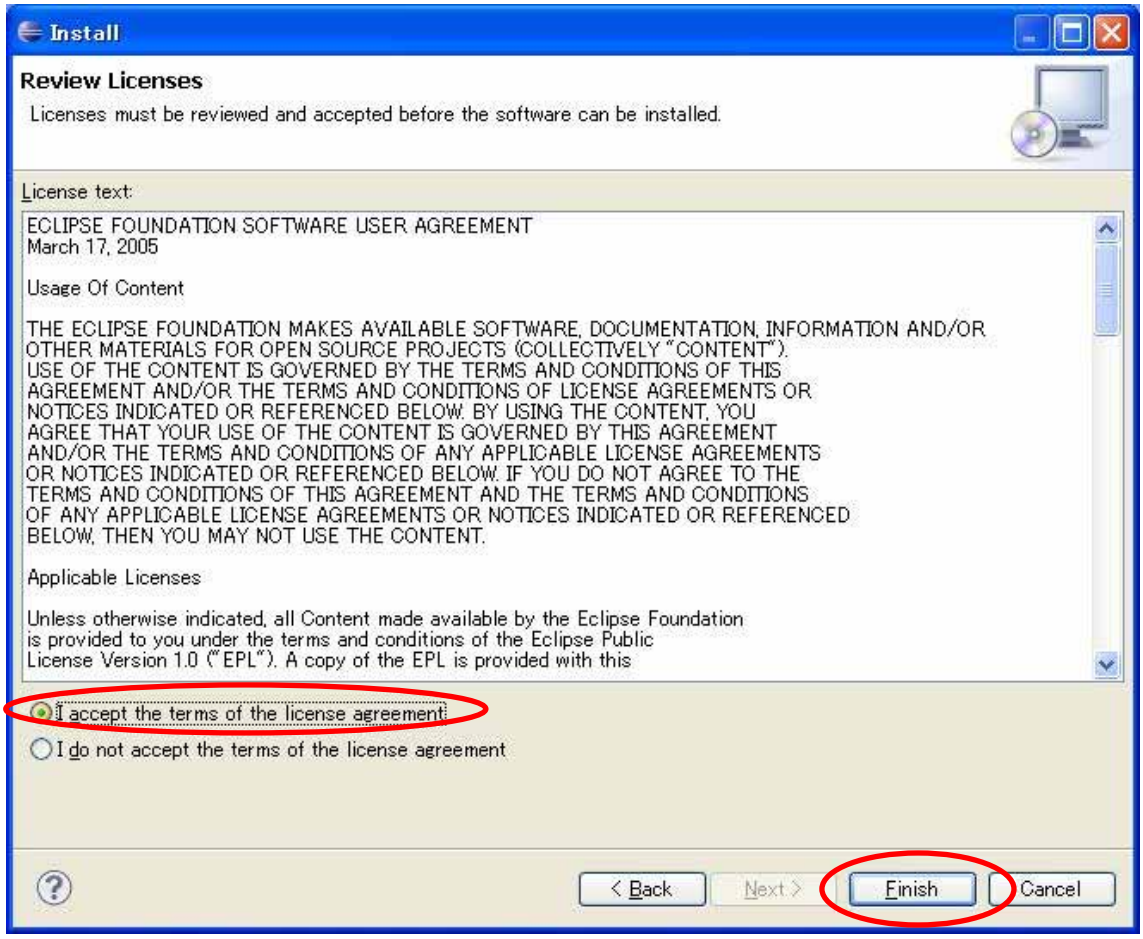
Add Site の"Location"に"http://opensource.zylin.com/zylincdt "と入力し OK ボタンを押す。

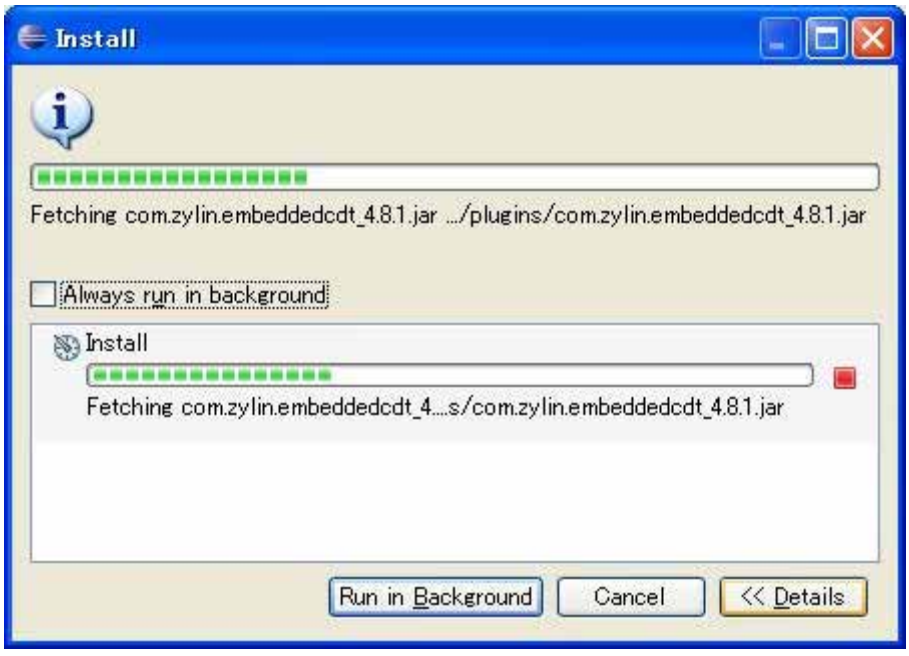


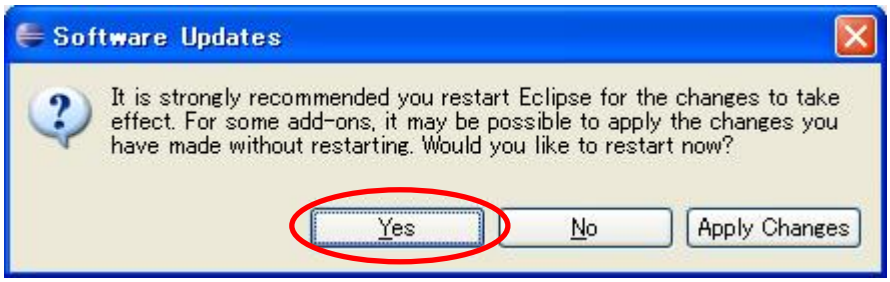
Install に"http://opensource.zylin.com/zylincdt "が追加されるのでチェックボックスにクリックしチェックを入れて Next ボタンを押す。




Install
Install Details
Review the items to be installed.
Name
Ver...
Id
Zylin Embedded CDT
4.8.1
com.zylin.cdt.feature.feature.group
Size: Unknown
Details
< Back
Next >
Finish
Cancel


Install
Review Licenses
Licenses must be reviewed and accepted before the software can be installed.
License text:
ECLIPSE FOUNDATION SOFTWARE USER AGREEMENT
March 17, 2005
Usage Of Content
THE ECLIPSE FOUNDATION MAKES AVAILABLE SOFTWARE, DOCUMENTATION, INFORMATION AND/OR
OTHER MATERIALS FOR OPEN SOURCE PROJECTS (COLLECTIVELY "CONTENT").
USE OF THE CONTENT IS GOVERNED BY THE TERMS AND CONDITIONS OF THIS
AGREEMENT AND/OR THE TERMS AND CONDITIONS OF LICENSE AGREEMENTS OR
NOTICES INDICATED OR REFERENCED BELOW. BY USING THE CONTENT, YOU
AGREE THAT YOUR USE OF THE CONTENT IS GOVERNED BY THIS AGREEMENT
AND/OR THE TERMS AND CONDITIONS OF ANY APPLICABLE LICENSE AGREEMENTS
OR NOTICES INDICATED OR REFERENCED BELOW. IF YOU DO NOT AGREE TO THE
TERMS AND CONDITIONS OF THIS AGREEMENT AND THE TERMS AND CONDITIONS
OF ANY APPLICABLE LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED
BELOW, THEN YOU MAY NOT USE THE CONTENT.
Applicable Licenses
Unless otherwise indicated, all Content made available by the Eclipse Foundation
is provided to you under the terms and conditions of the Eclipse Public
License Version 1.0 ("EPL"). A copy of the EPL is provided with this
I accept the terms of the license agreement
I do not accept the terms of the license agreement
< Back
Next >
Finish
Cancel

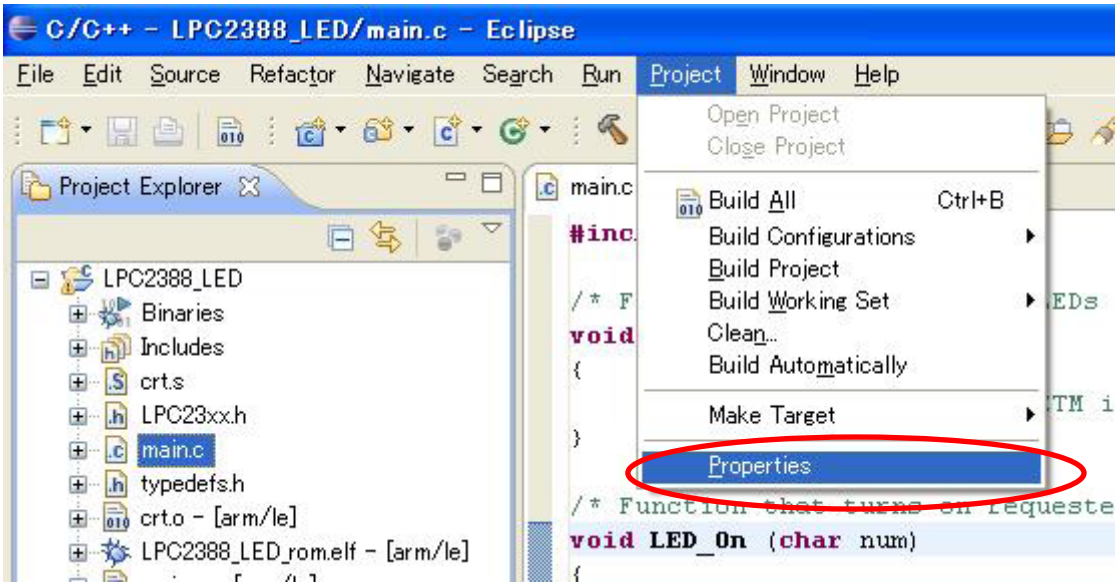

Install
Fetching com.zylin.embeddedcdt_4.8.1.jar .../plugins/com.zylin.embeddedcdt_4.8.1.jar
Always run in background
Install
Fetching com.zylin.embeddedcdt_4....s/com.zylin.embeddedcdt_4.8.1.jar
Run in Background
Cancel
<< Details



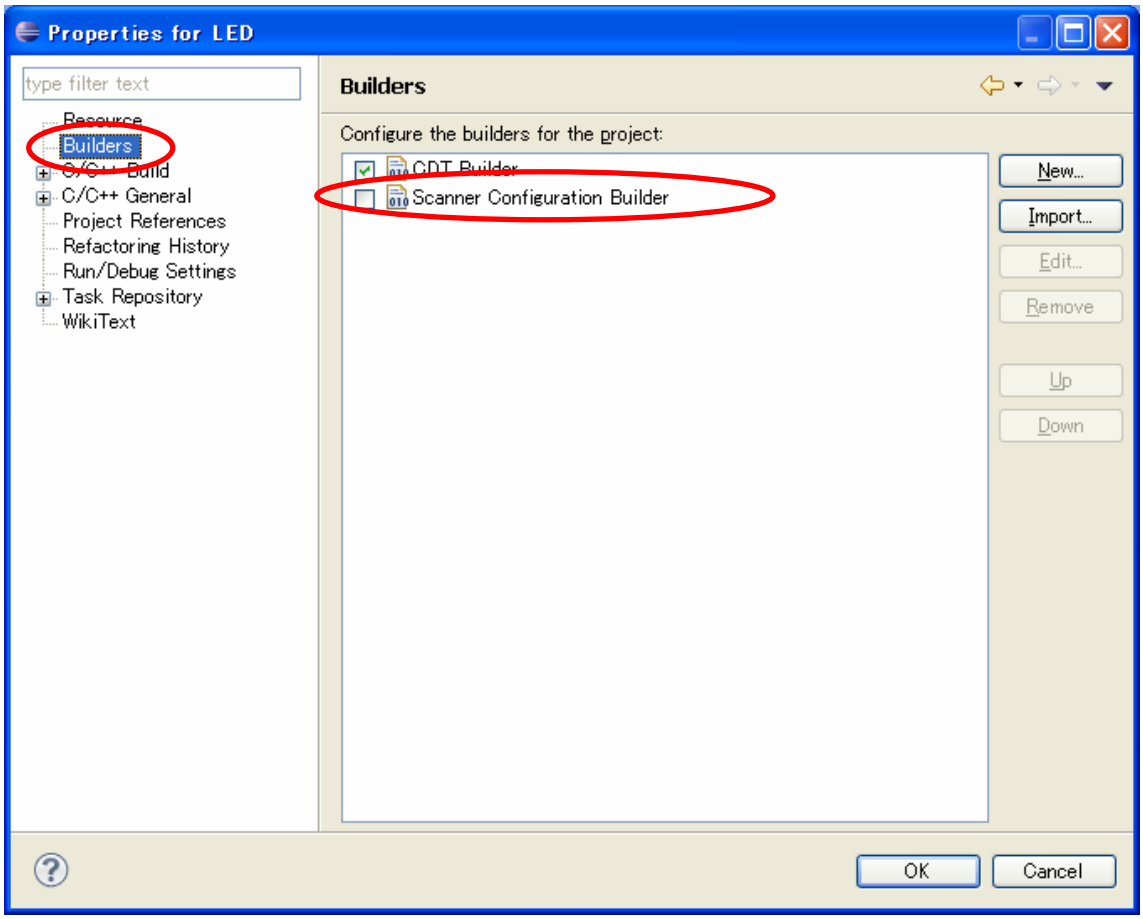
インストール完了したら、Yes ボタンを押して、Eclipse を再起動させます。

4.5 ビルドの設定

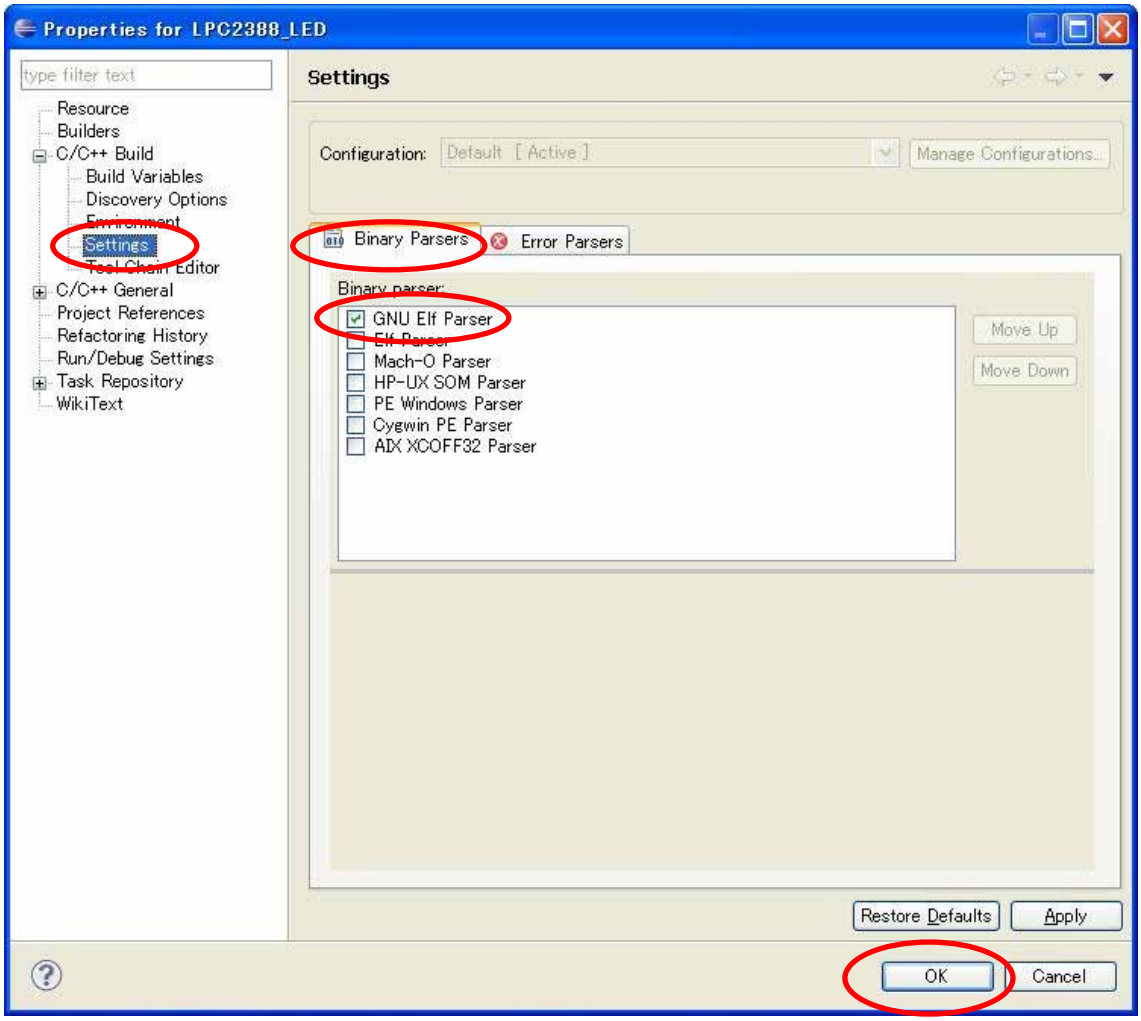
Eclipse の"Project"→"Preferences"を選択する。



Preferences の"Build"を選択し"Scanner Configuration Builder"のチェックマークを外して

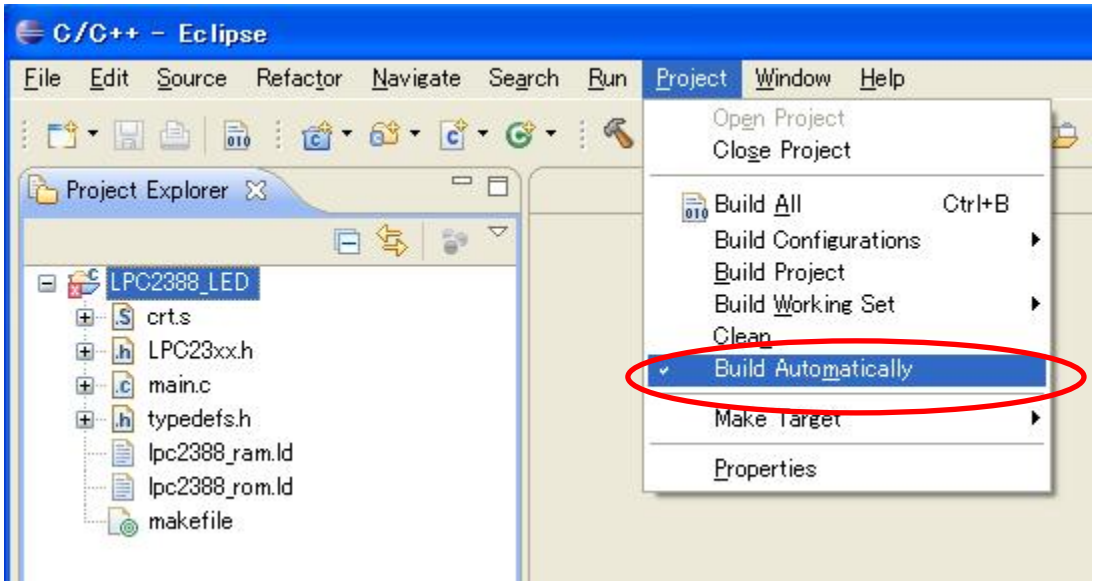


Preferences の"C/C++ Build"→"Settings"を選択し"Binary Parsers"タブの"GNU Elf Parser"にチェックを入れて OK ボタンを押します

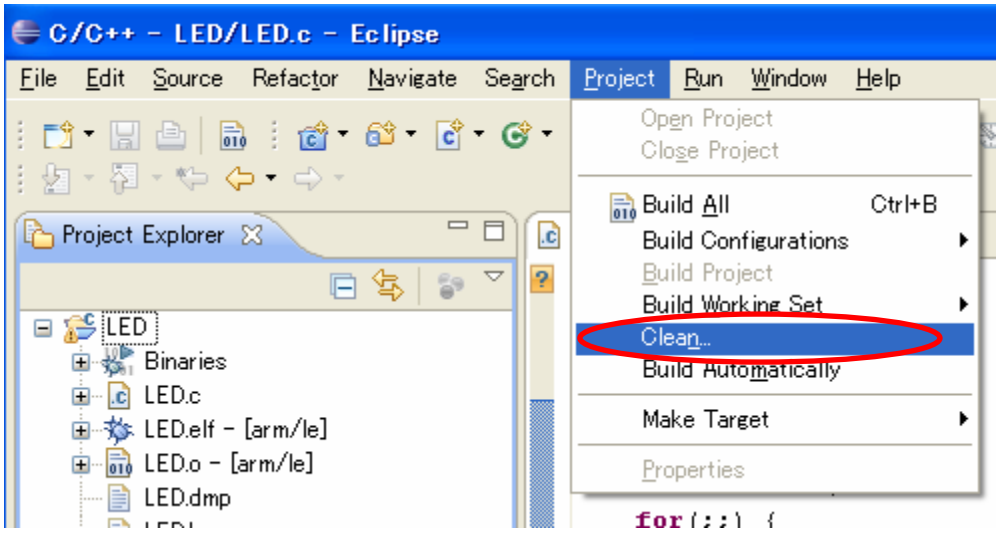


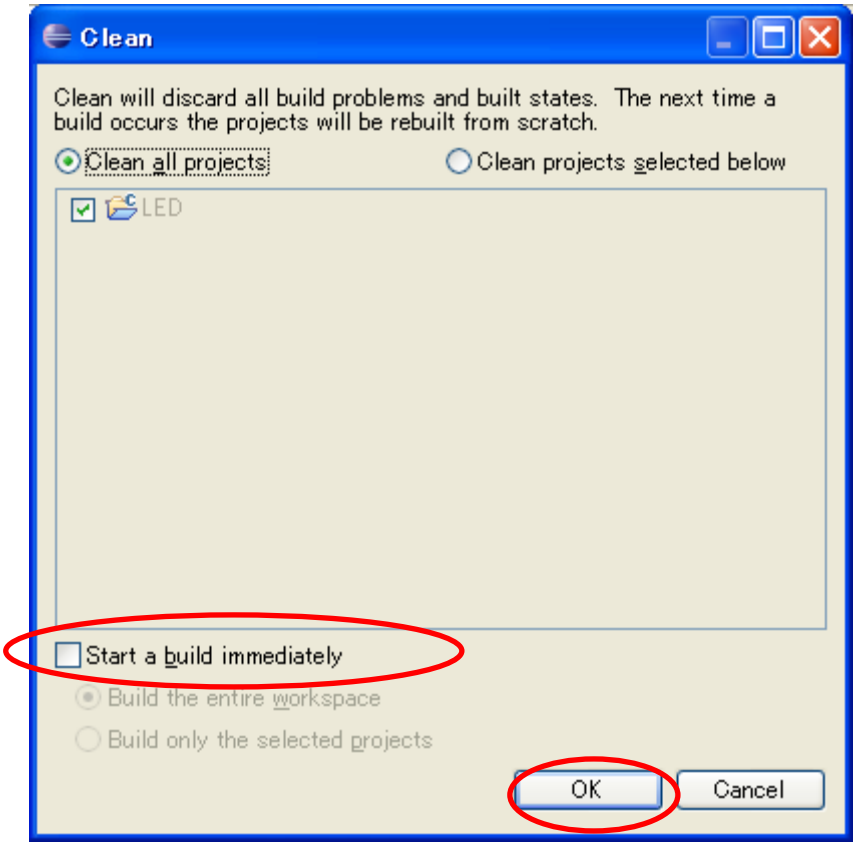
4.6 ビルド

Eclipse の"Project"→"Build Automatically"のチックを外してください。



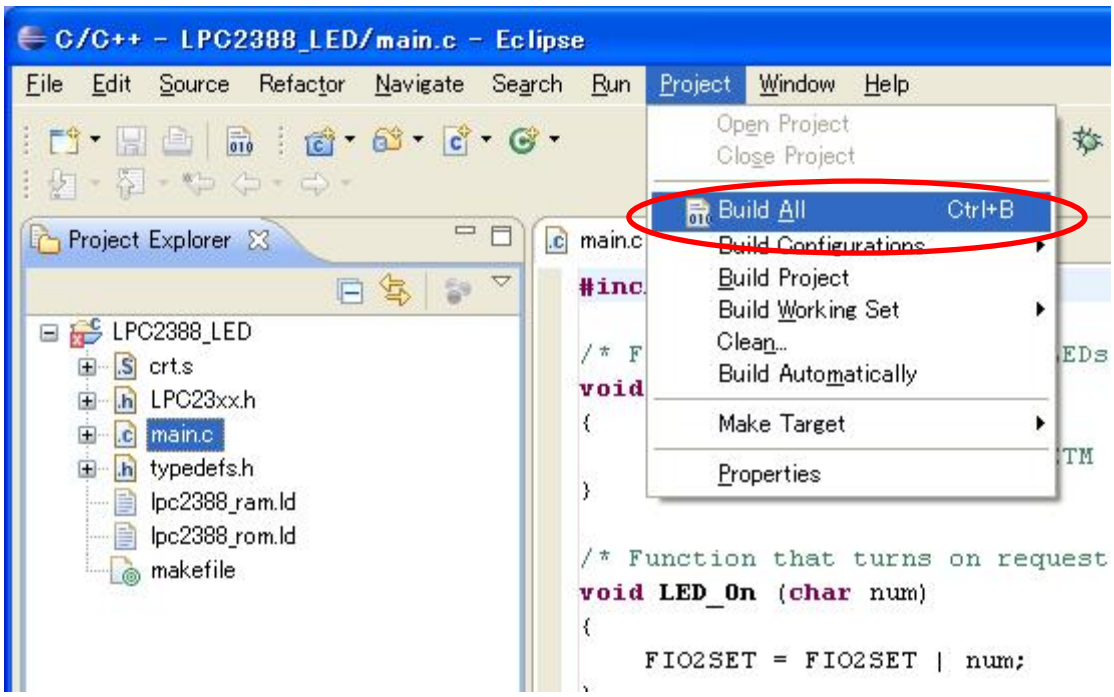
"Project"→"Clean"を選択するクリアが行われます。

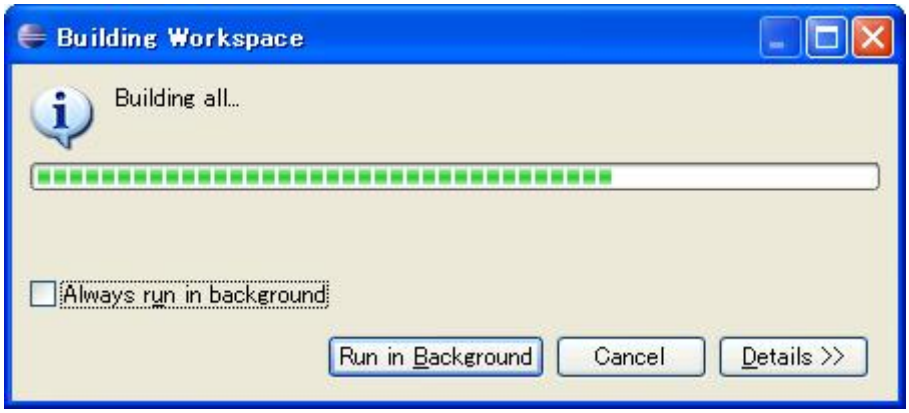




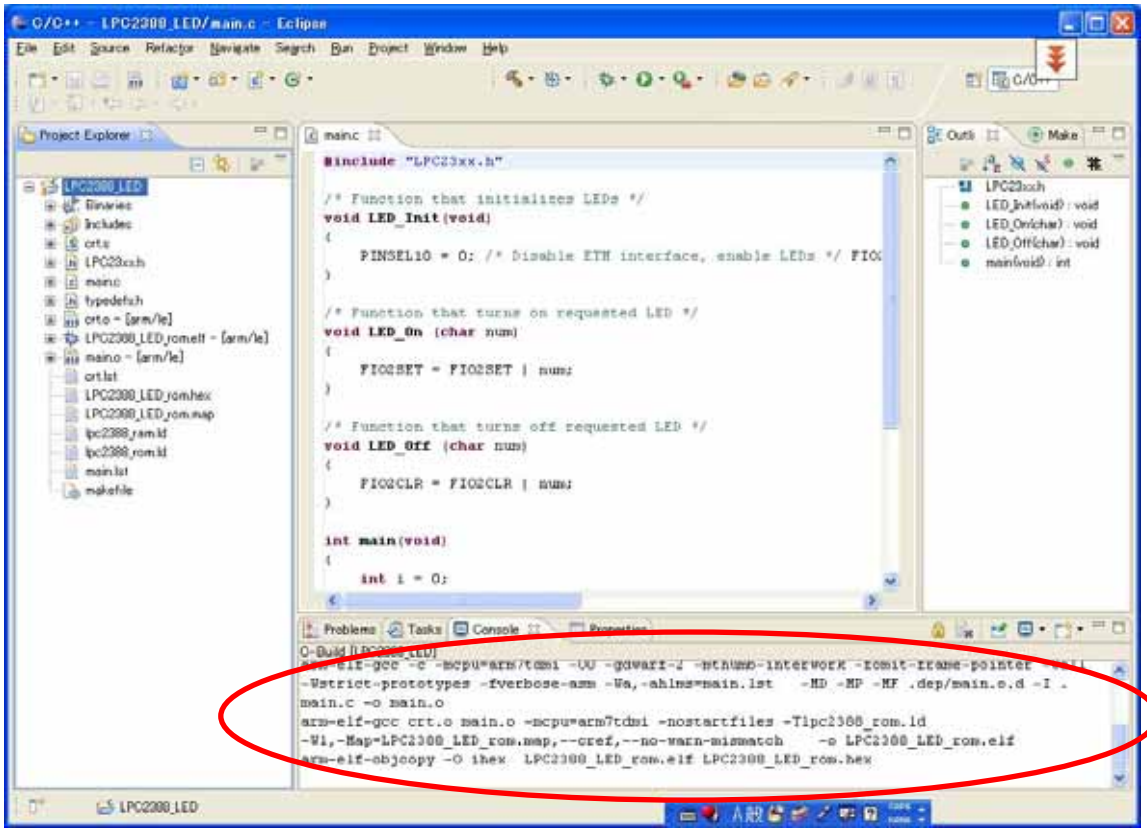
「Start a build immediately」のチェックマークを外して、「Ok」を押します。

"Project"→"Build All"を選択するとビルドが行われます。





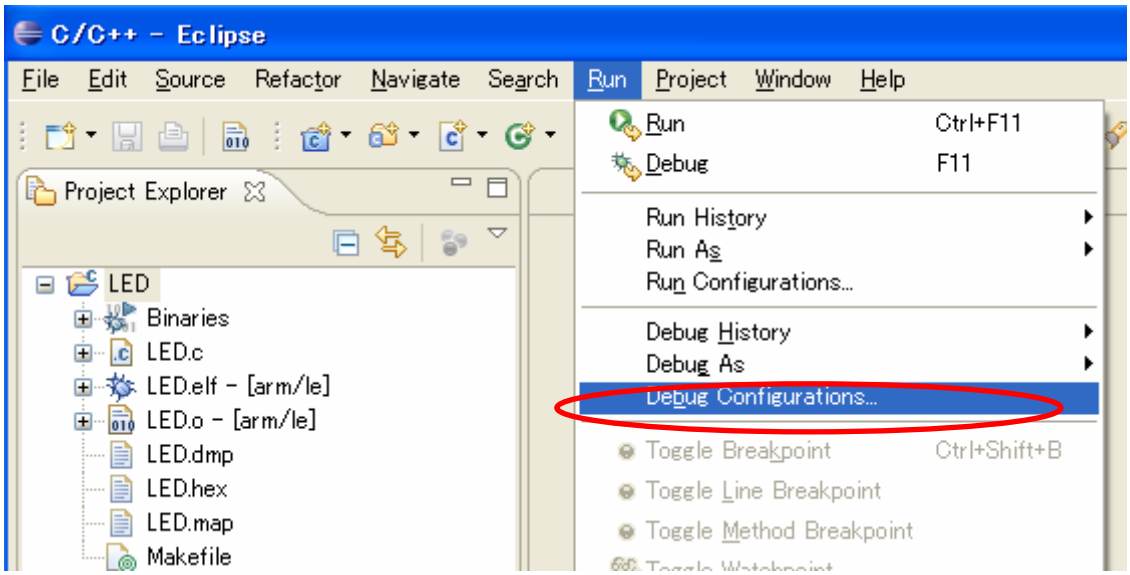
コンパイル中です。



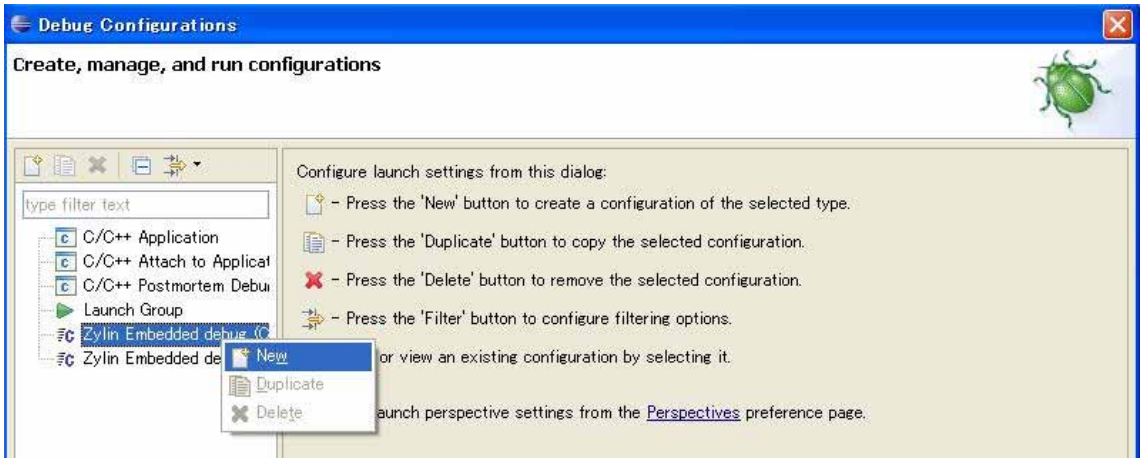
コンパイルが成功すれば、実行ファイル LED.elf と LED.hex を生成されます。

4.7 GDB の設定

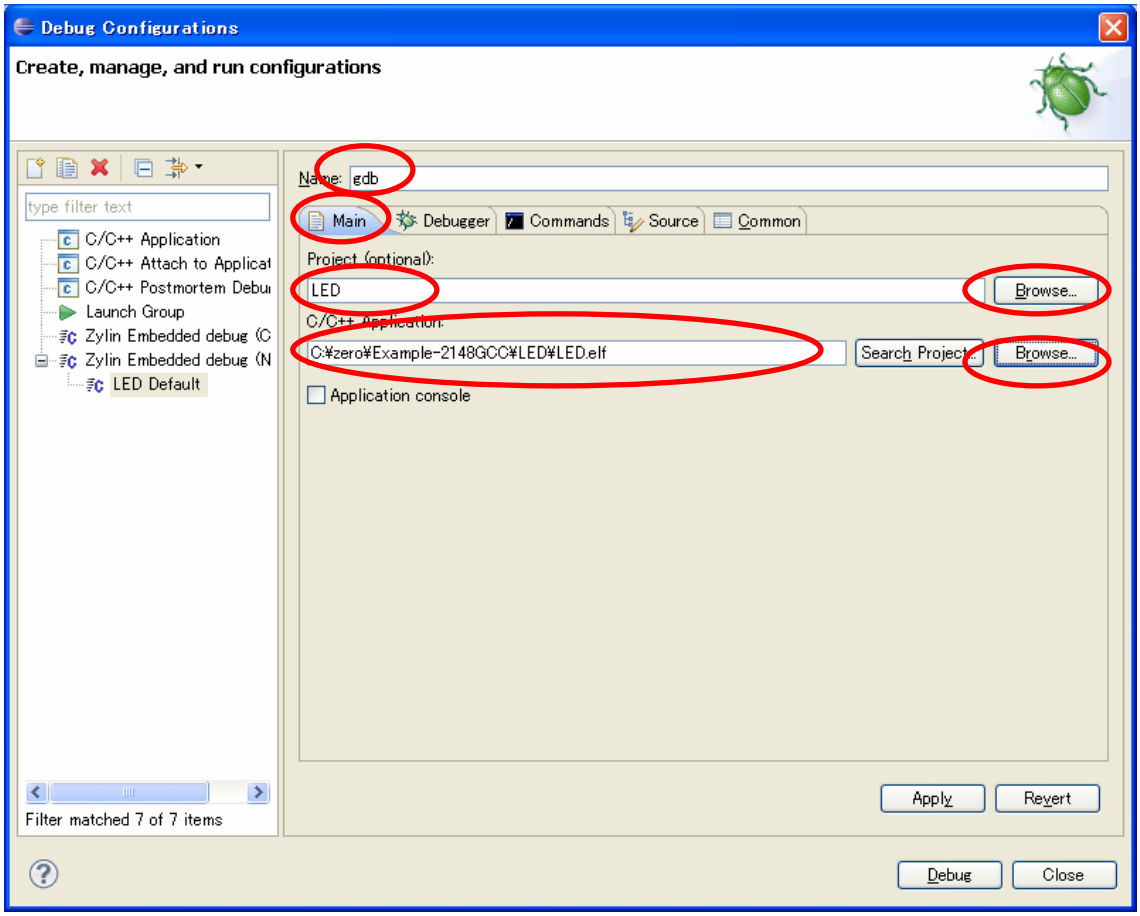
Eclipse の"Run"→"Debug Configurations..."を選択します。



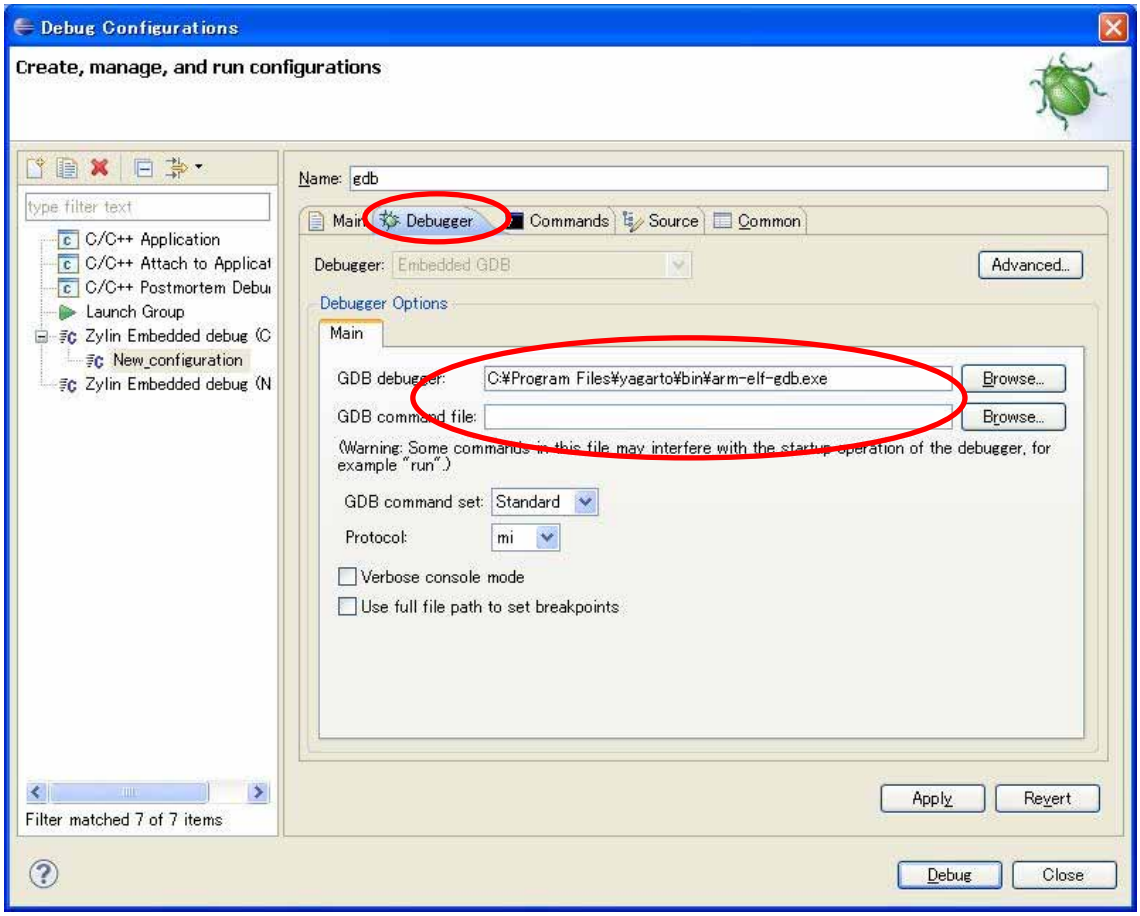
Debug Configurations の"Zylin Enbedded debug(Native)"を右クリックし"New"を選択する。



Name に適当な名前を入れる。例、"gdb"と入れます。Main タブの"Project"に"LED"、"C/C++ Appication:"に"C:¥workspace¥LED¥LED.elf"と入力します。

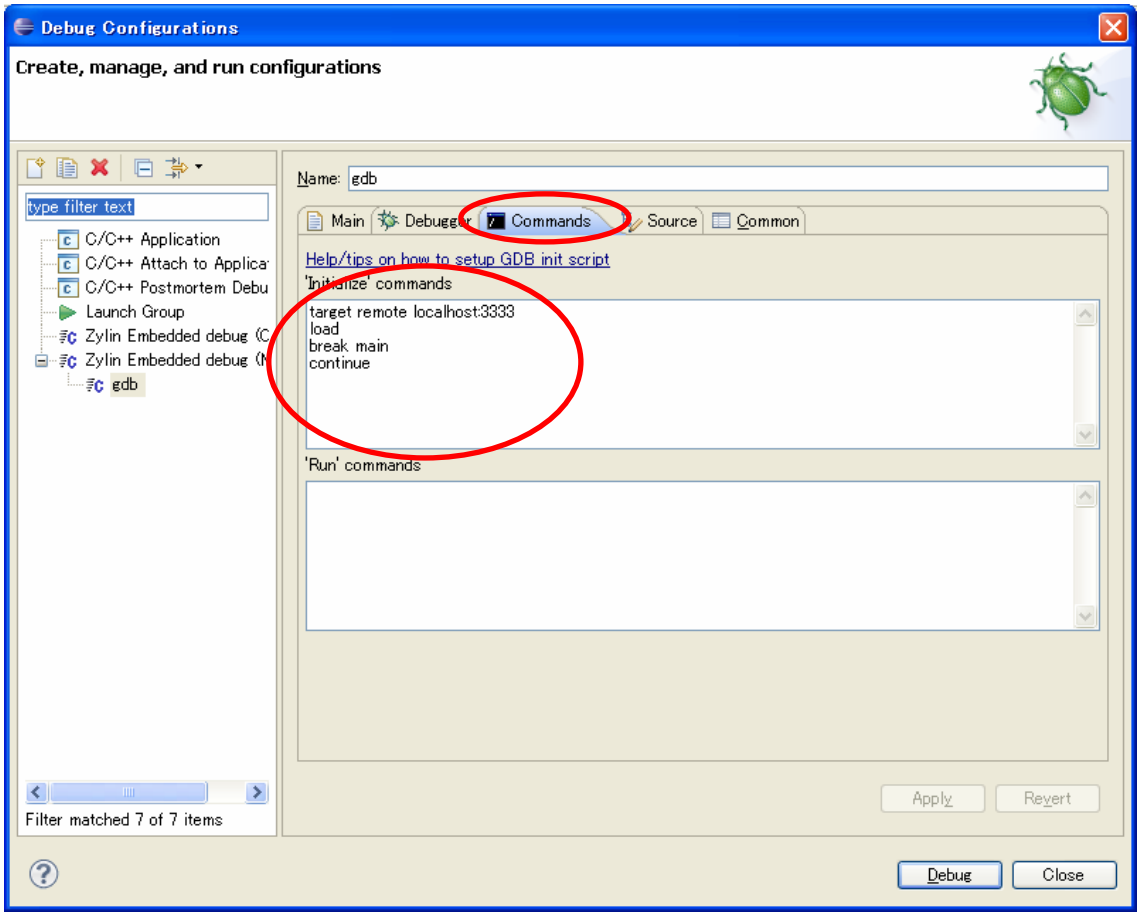


Debugger タブの"GDB debugger:"に"arm-elf-gdb"、"GDB command file:"に何も入力しません。

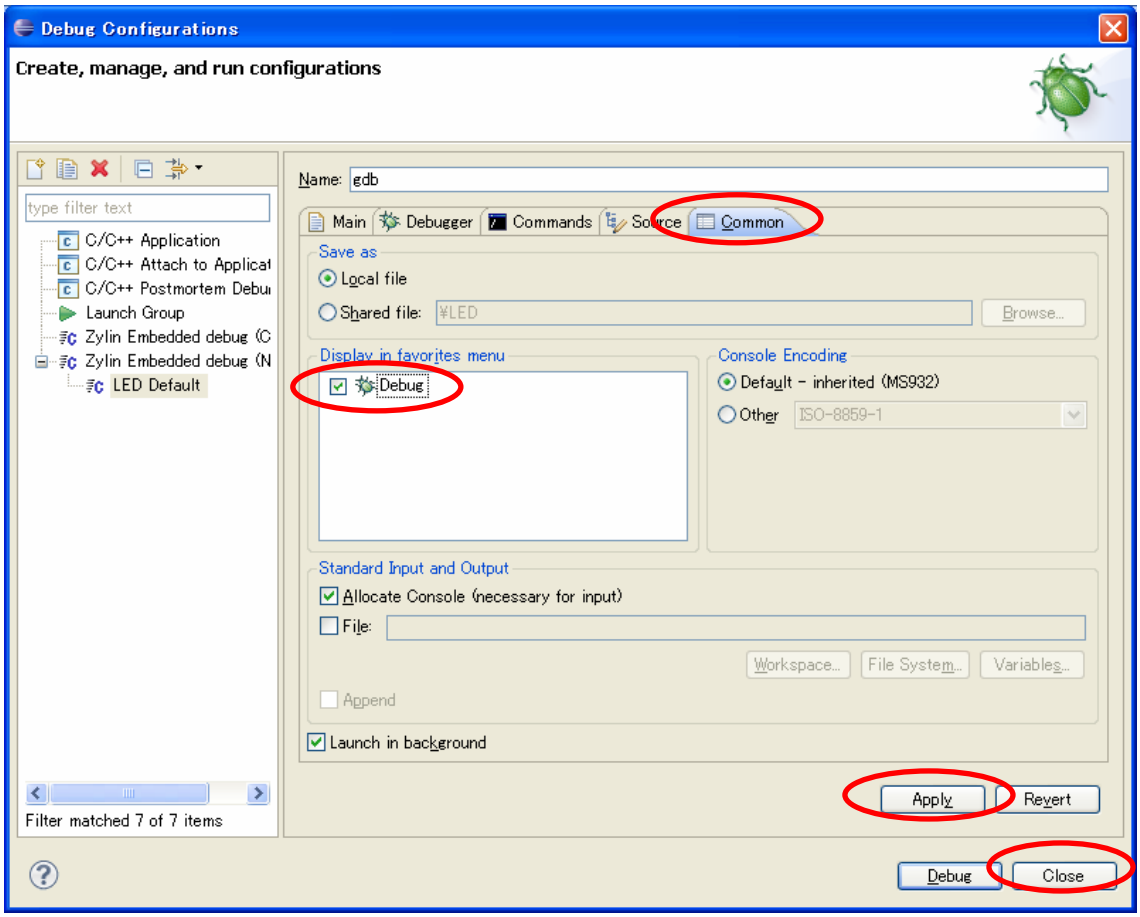


Commands タブの"'Initialize' commands"に下記の画面の様に入力します

```
target remote localhost:3333
monitor halt
monitor step
load
break main
continue
```



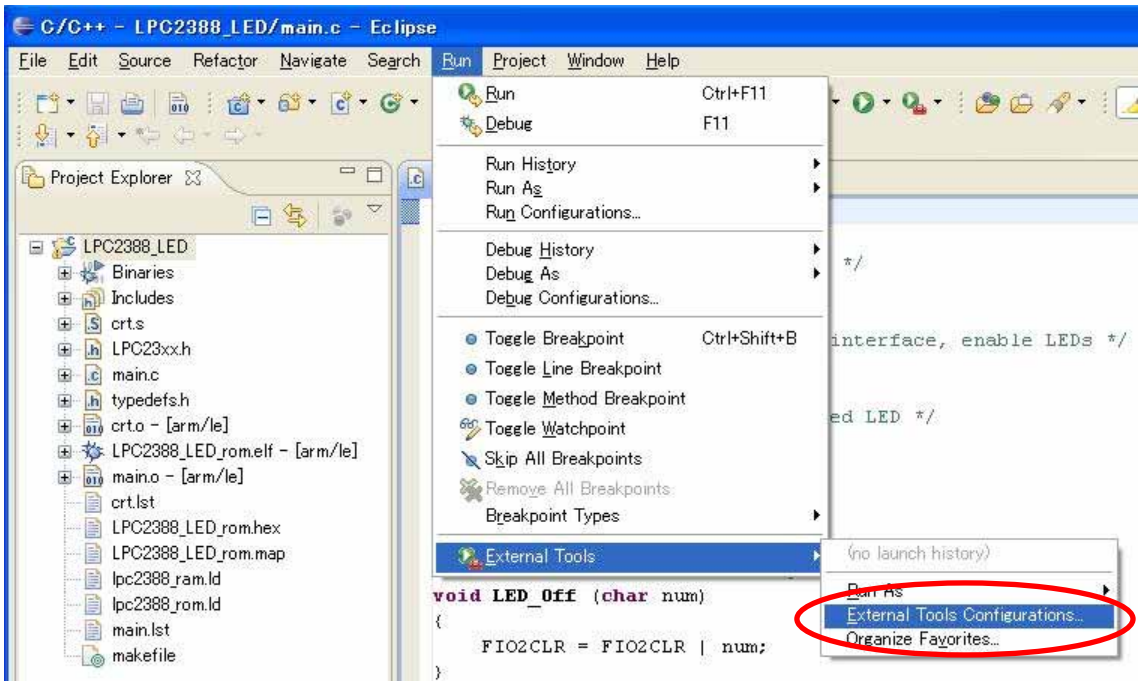
Common タブの"Display in favorites menu"の Debug にチェックを入れます。全てを入力し終えたら"Apply"ボタンを押し、"Close"ボタンを押します。



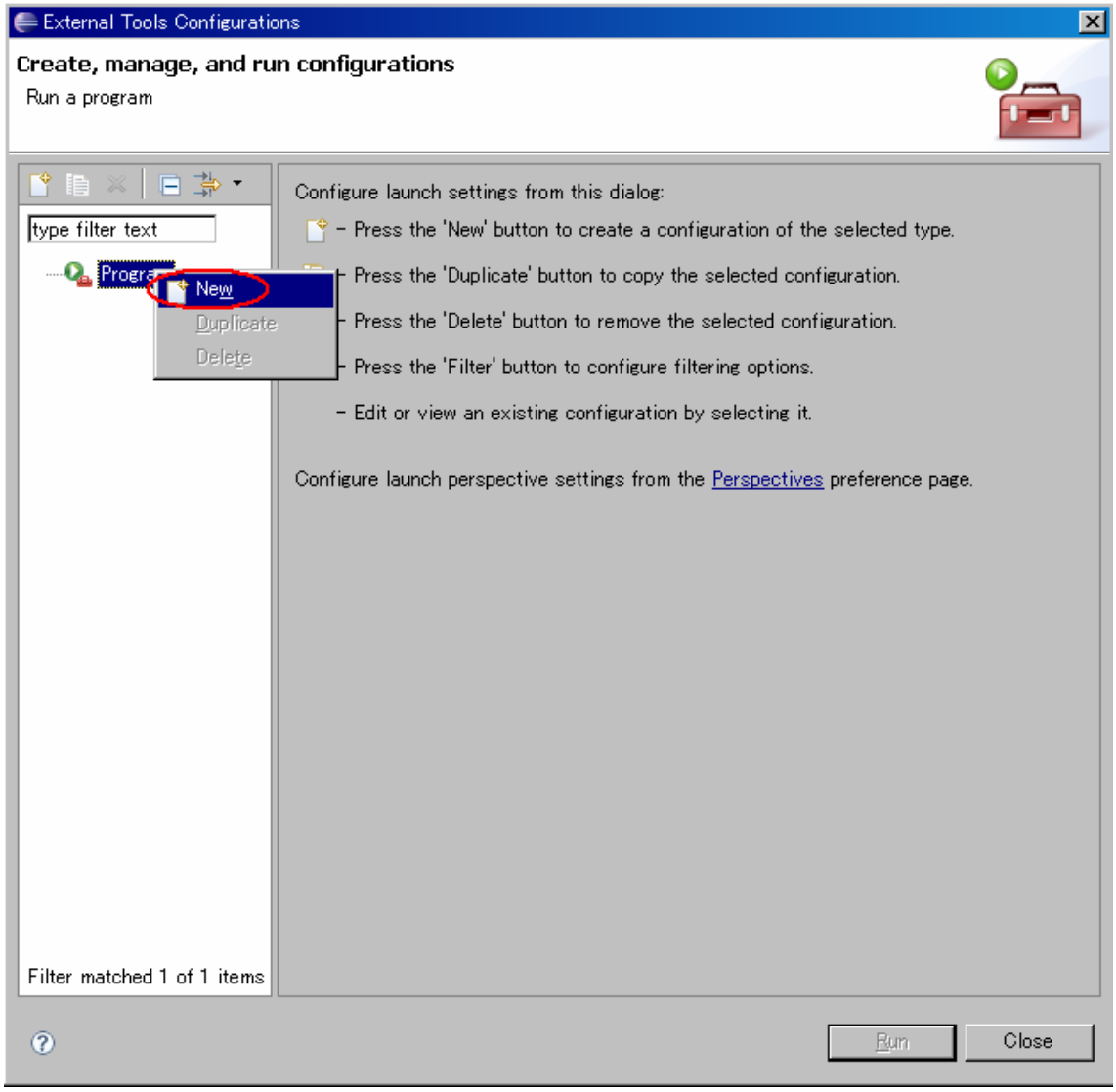
全てを入力し終えたら"Apply"ボタンを押し、"Close"ボタンを押します。

4.8 OpenOCD の設定

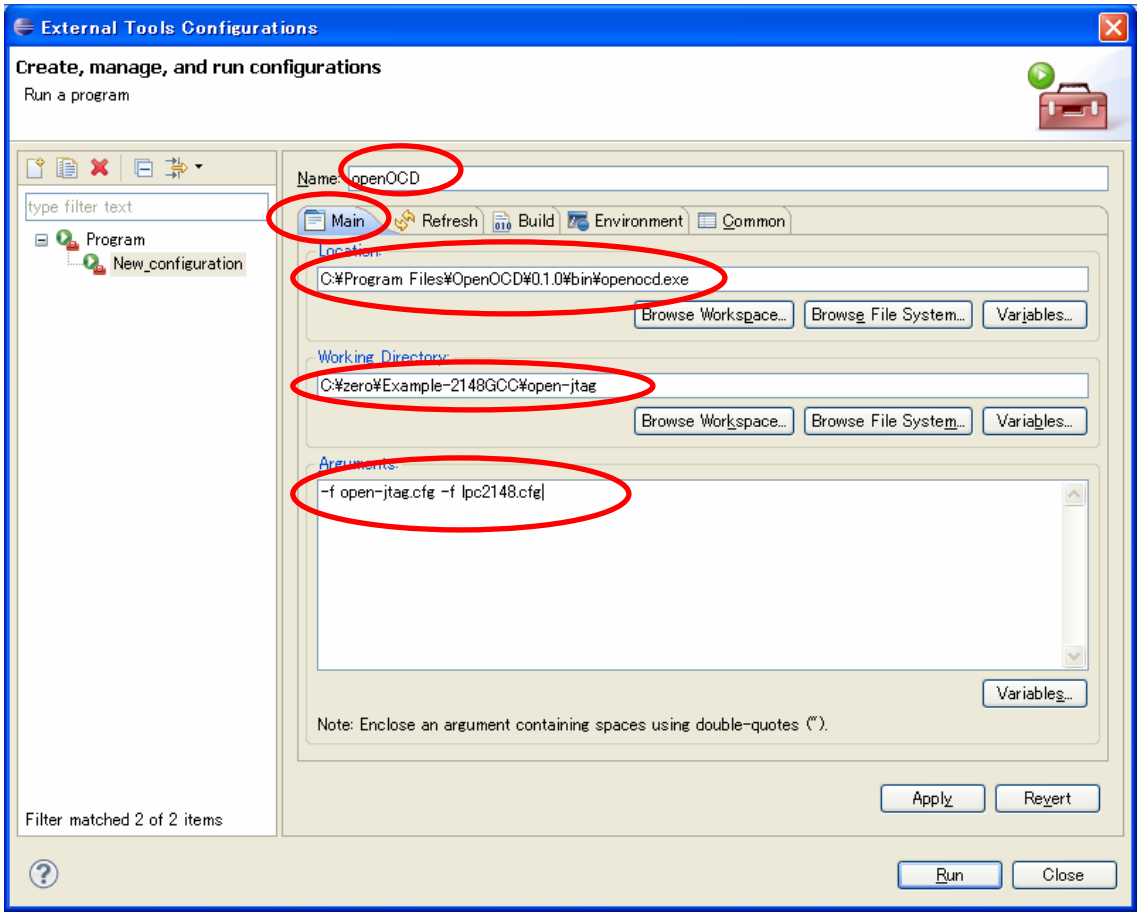
Eclipse の"Run"→"External Tools."→"External Tools Configrurations..."を選択します。

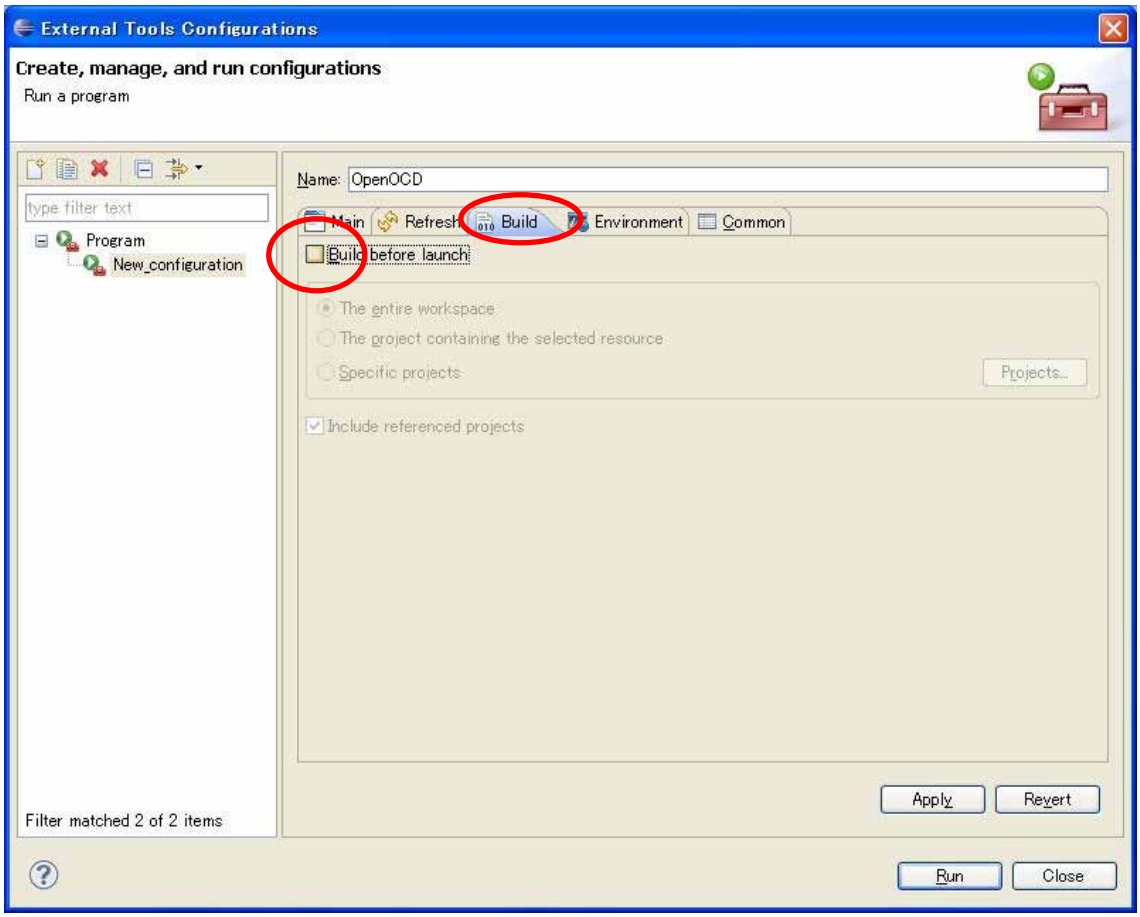


External Tools Configrurations の"Program"を右クリックし、"New"を選択します。

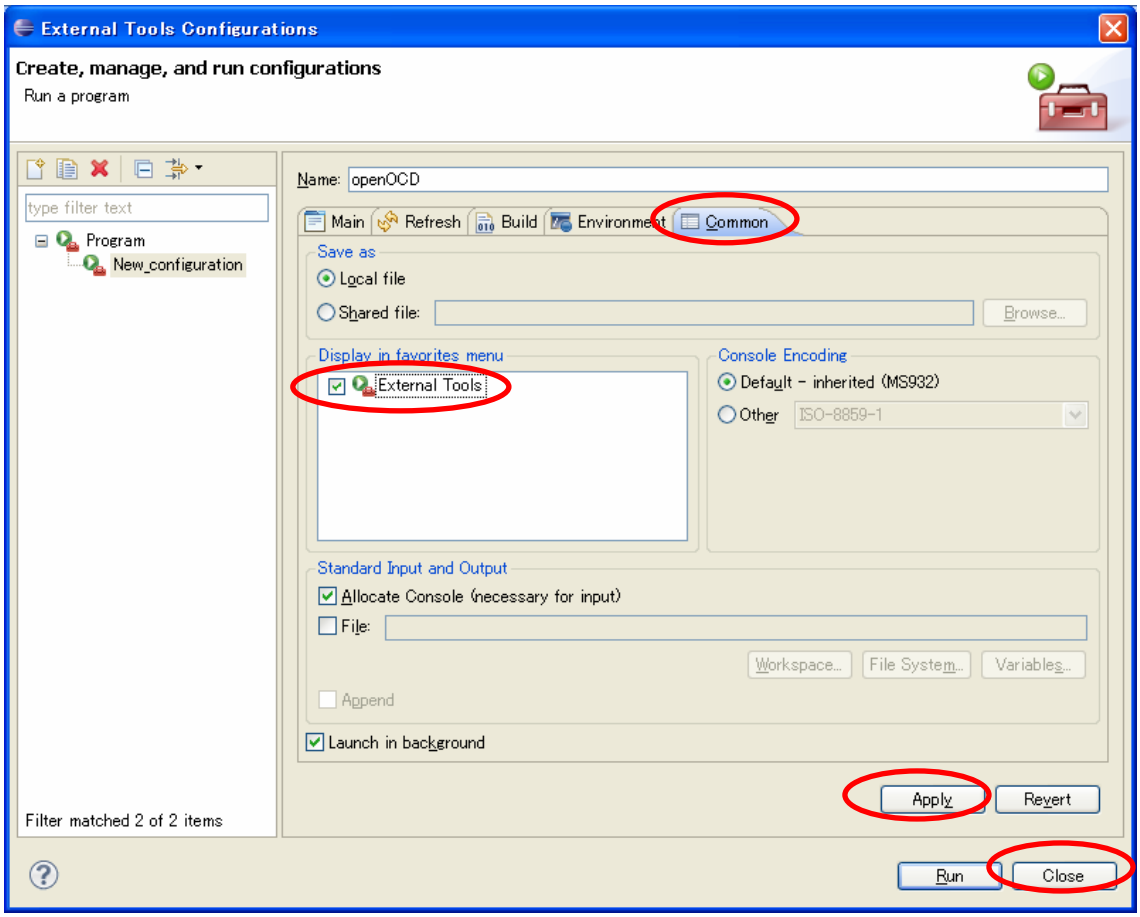


Main タブの"Name"に適当な名前を入力してください。私は ” OpenOCD"と入れました。
"Location:"に"C:¥Program Files¥OpenOCD¥0.1.0¥bin¥openocd"、
"Working Directory:"に"C:¥Example-2148GCC¥open-jtag"、
"Arguments:"に-f open-jtag.cfg -f lpc2148.cfg と入力します。





Build タブをクリックし"Build before launch"にチェックを外れます。



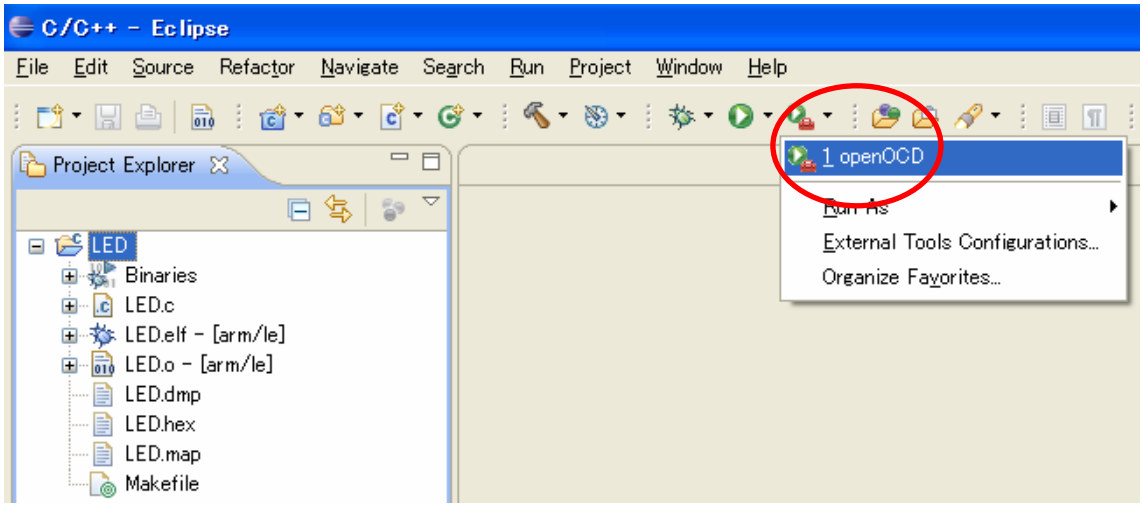
Common タブをクリックし"Display in favarites menu"の"External Tools"にチェックを入れます。全てを入力し終えたら"Apply"ボタンを押し、"Close"ボタンを押します。

4.9 デバッグ

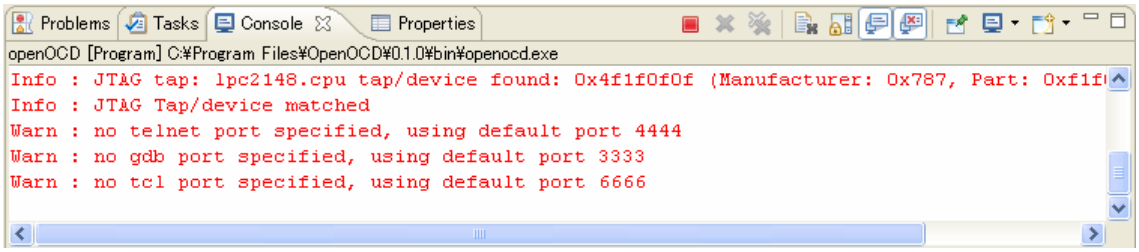
電源投入

1. OpenJTAG をターゲット(LPC2148 ボード)とパソコンに接続
2. ターゲットに電源を入れます

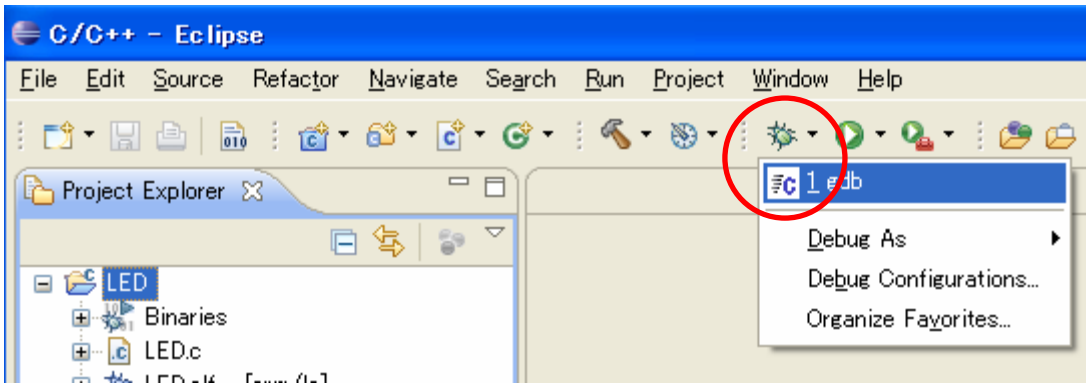
External Tools の▼ボタンをクリックし、OpenOCD を選択

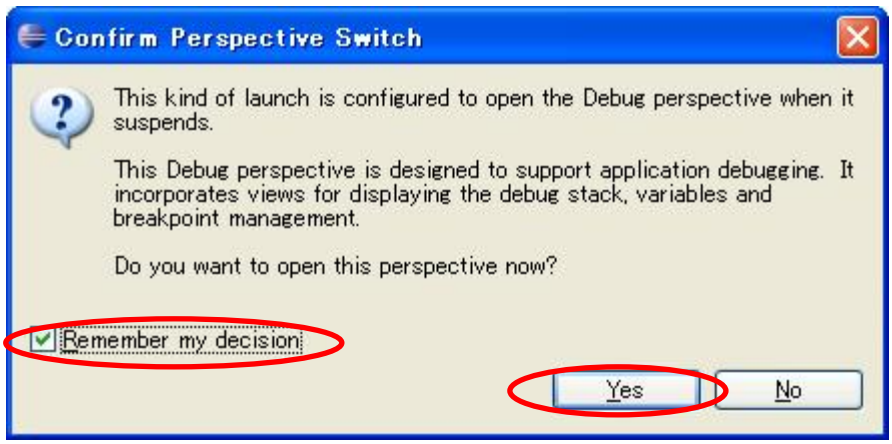


Console ウインドに下記のメッセージが出力

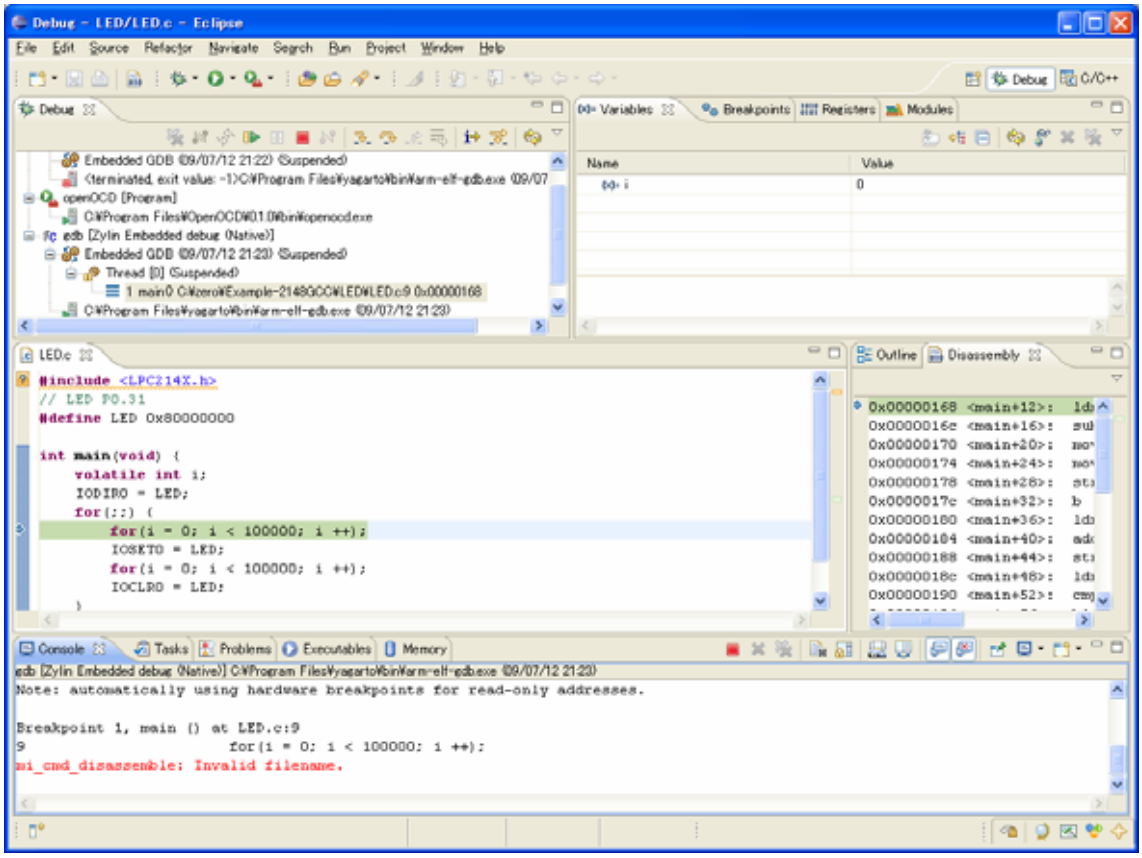


Debug の▼ボタンをクリックし、"gdb"を選択。

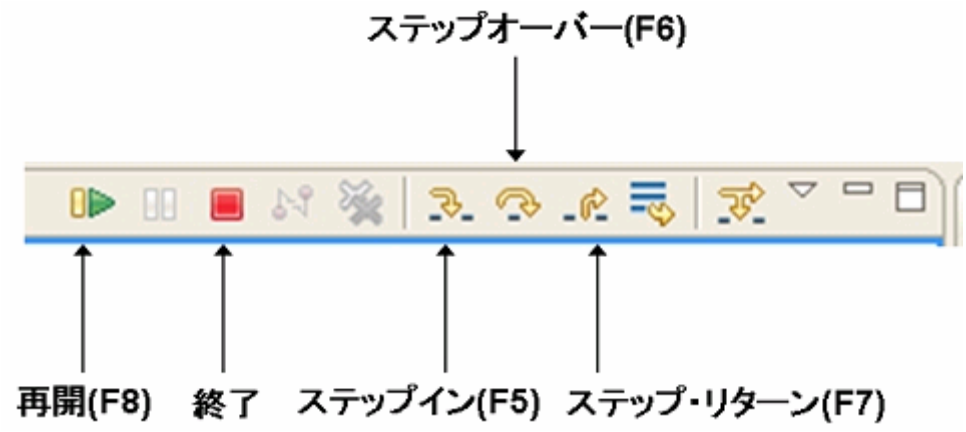




Yes ボタンを押して、デバッグが開始します。



Eclipse に Debug 用のコマンドあるいはショットカット一覧
詳しくは Eclipse のドキュメントを参照



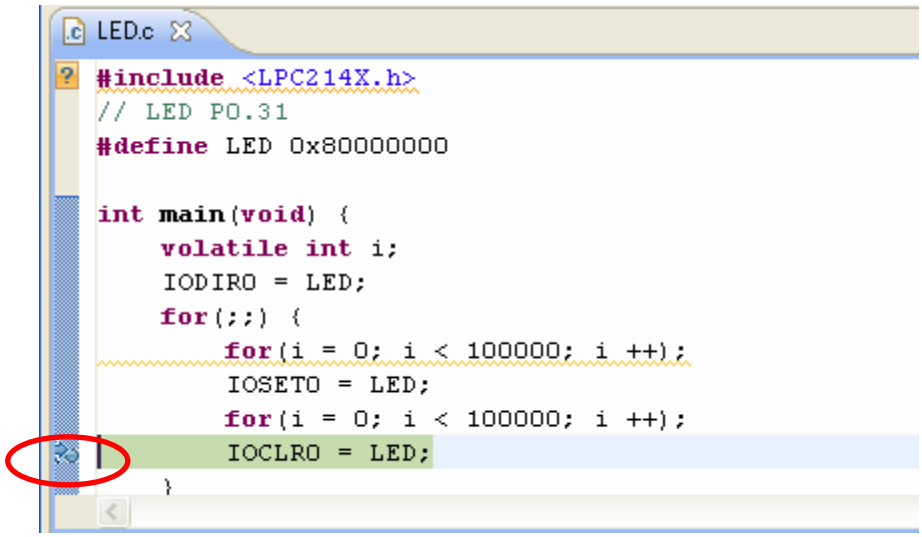
ステップ実行において良く使われる操作の一覧を以下に示します。

操作名	ショートカットキー
再開	F8
ステップイン	F5
ステップオーバー	F6
ステップ・リターン	F7

ステップ実行とは関係ありませんが、前回起動したクラスを再度実行したデバッグする場合は、以下のショートカットキーが便利です。

操作名	ショートカットキー
前回の起動を実行	Ctrl + F11
前回の起動をデバッグ	F11

ブレークポイントでプログラムが中断した状態から、次のブレークポイントまで実行させたり、1 行ずつ実行させたりできます。コードを繰り返して実行することにより、LED ランプが点滅します。

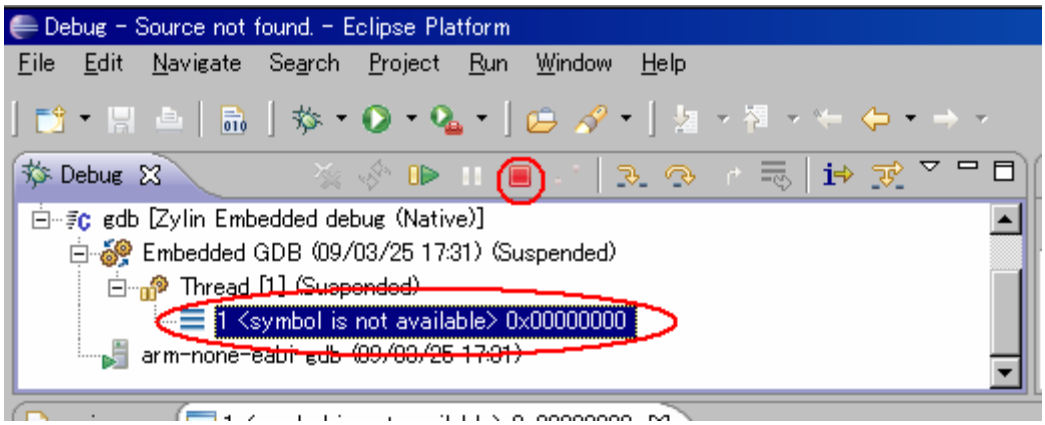


ダブルクリックで breakpoint を設置/キャンセルします。

4.10 デバッグ終了

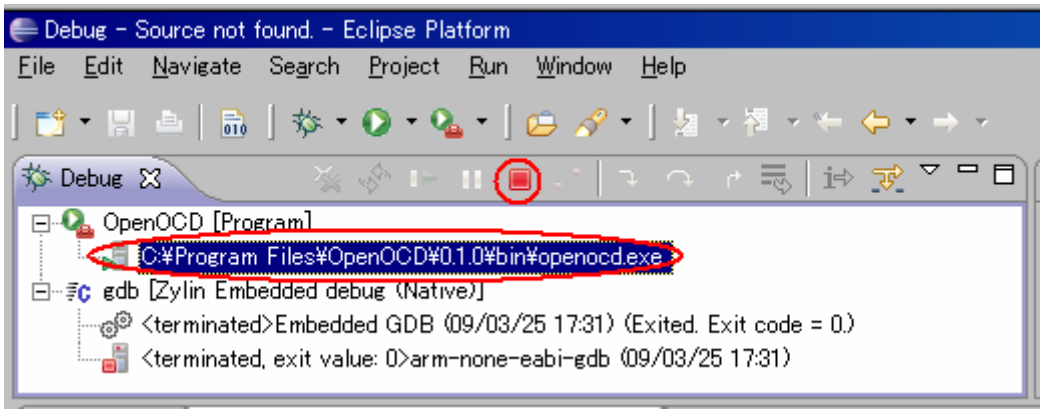
1）gdb の停止

Debug ウインドウの gdb の Thread を選択し、停止ボタンと押します



2）OpenOCD の停止

Debug ウインドウの OpenOCD の Thread を選択し、停止ボタンと押します



3）電源停止

ターゲットの電源を停止

4）OpenJTAG をターゲットから取り外す

5）上記が面倒であれば Eclipse を終了しターゲットの電源停止、open-JTAG を取り外しでも OK です。

※ ほかのサンプルは Example-2148GCC.rar を参照してください。

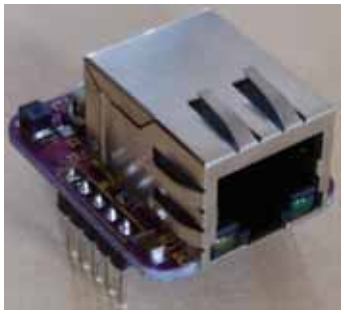
第五章 SPI イーサネットモジュール ENC28J60

MICROCHIP 社から ENC28J60 という、SPI 接続のイーサネットコントローラ(MAC+PHY)が発売されました。10BASET-T ですので、速度は早くありませんが、SPI インターフェイスでマイコンと接続できるのが特徴です。

SPI インターフェイスは、SO、SI、SCK、CS の４本で構成され、速度も ENC28J60 の場合は 8MHz-10MHz でマイコンと接続することが出来ます。SPI は AVR、PIC、ARM、H8 などのマイコンに標準的なインターフェイスですので、手軽にマイコンと接続出来ます。10BASET-T のイーサネットコントローラとしては RTL8019AS などが有名ですが、こちらはデータバス接続となりますので、２０本近くの配線が必要になりますので、ピン数の少ないマイコンには荷が重過ぎます。

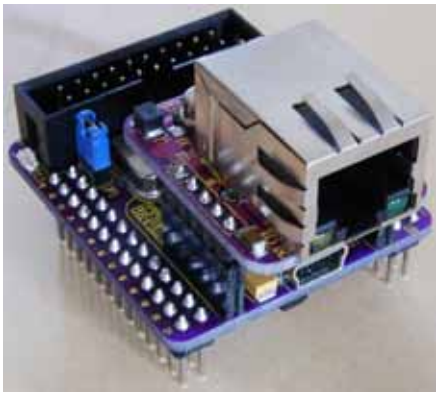
LPC2148 モジュール

＋

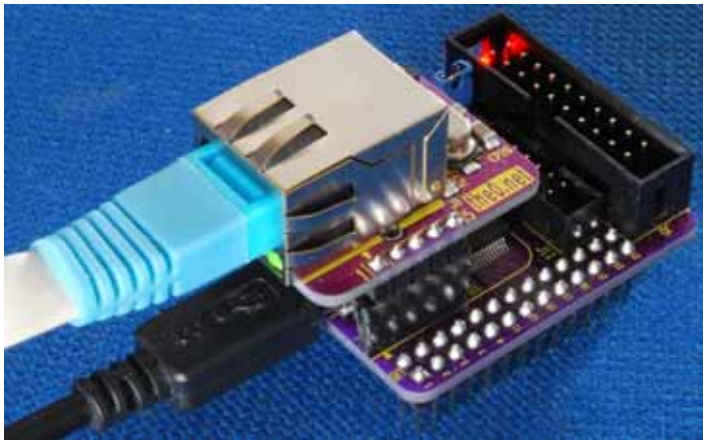


ENC28J60

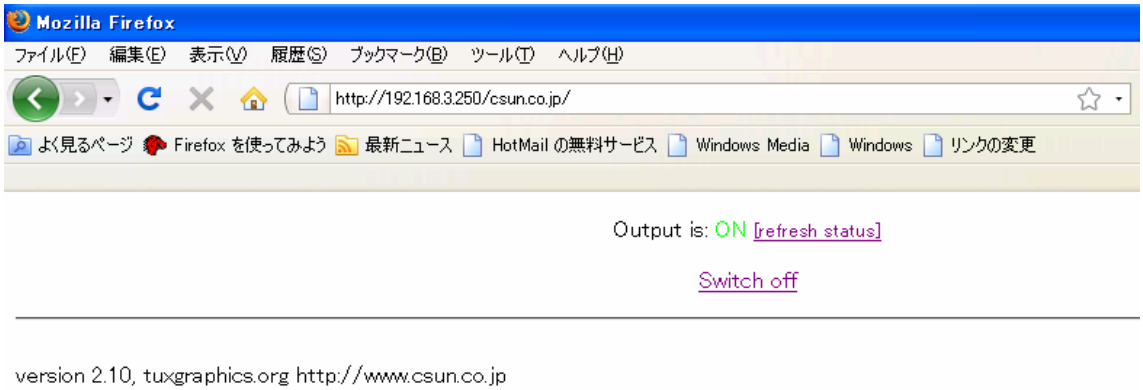
＝



サンプル： WebServer/WebServer.bin



このサンプルを LPC2148 モジュールにダウンロードして、パソコンのブラウザーで URL「http://192.168.3.250/csun.co.jp」を入力すると



サンプルの IP アドレスとパスワードが WebServer.c にあります。目分のネットワークによって直してください。

WebServer.c

```
～ (略) ～

static uint8_t mymac[6] = {0x54,0x55,0x58,0x10,0x00,0x24};
// how did I get the mac addr? Translate the first 3 numbers into ascii is: TUX
static uint8_t myip[4] = {192,168,3,250};
// listen port for tcp/www (max range 1-254)
#define MYWWWPORT 80
//
// listen port for udp
#define MYUDPPORT 1200

#define BUFFER_SIZE 1500//450
static uint8_t buf[BUFFER_SIZE+1];

// the password string (only the first 5 char checked), (only a-z,0-9,_ characters):
static char password[]="csun.co.jp"; // must not be longer than 9 char

～ (略) ～
```

第六章 USB で ARM9(Linux)に接続

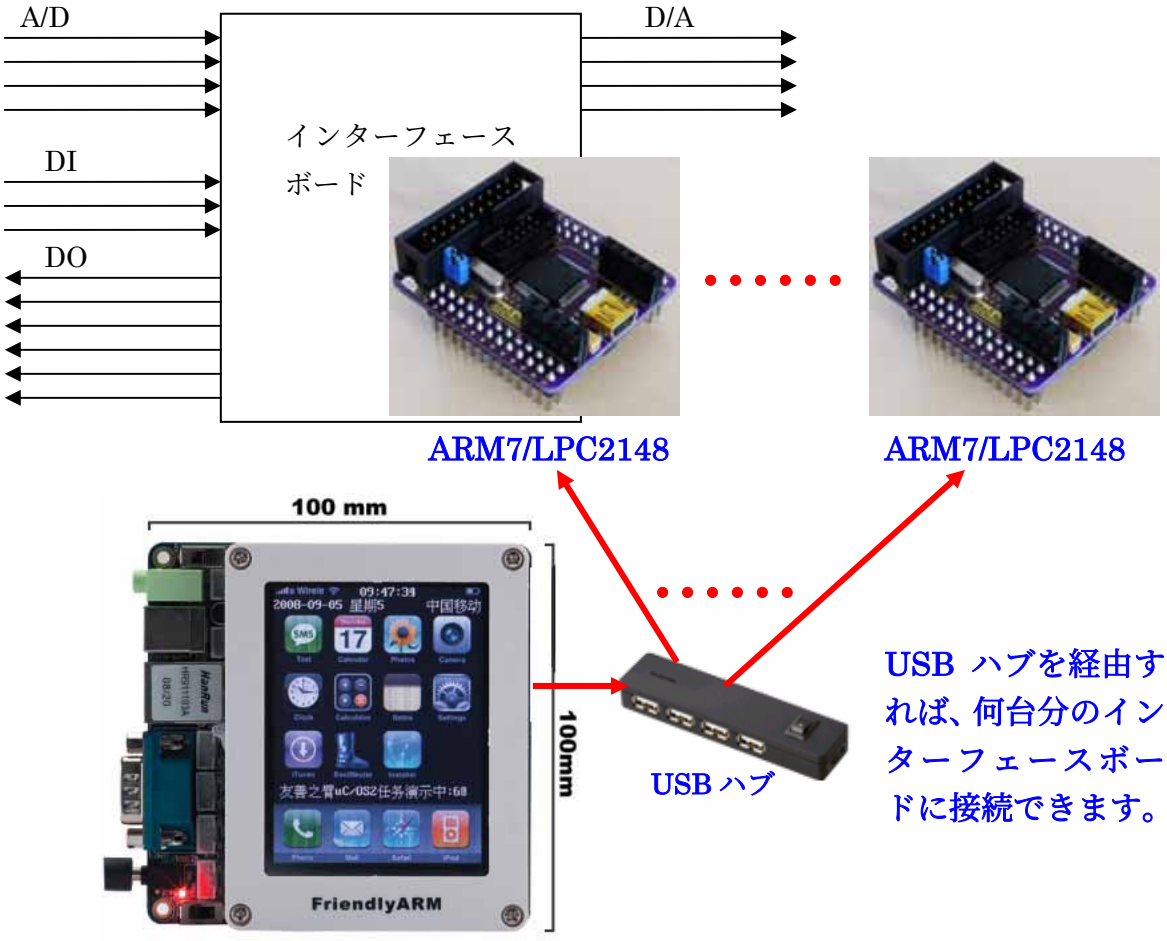
弊社の ARM9(MINI2440)は標準 OS に Linux を採用します。Linux には、信頼性が高いネットワークスタックが実装され、利用できます。従って、ネットワークに接続する信頼性の高い遠隔制御機器が、容易に作成できる利点があります。Linux にも USB スタックが実装され、多種類の USB デバイスを利用できます。例えば、USB プリンター、USB 無線 LAN、USB メモリ、SD カードなど。パソコンの Linux 上のアプリケーションが ARM9 上で利用できます。ゼロから開発せずに、例えば GUI と Web サーバなどが組み込み用機器で利用できるわけで、これは非常に大きな利点といえます。

Linux の便利さの反面、複雑、重い、反応速度が遅いです。反応速度は大体数十 ms ぐらいです。この反応速度は人間との会話に満足できますが、機械制御のリアルタイム性に足りないかもしれません。

ARM7 シリーズはリアルタイム制御に向けのマイコンです。OS なしあるいは簡単な RTOS を搭載します。1us~1ms 以上の反応速度が実現できます。ARM7 は制御のため、豊富な制御用の周辺機能を持ちます。例えば AD, DA, PWM, GPIO, カウンターなど。

ARM7/LPC2148 には USB ターゲットポートがあります。最大通信速度 12Mbps。LPC2148 は USB デバイスとして使えます。ARM9 は USB ハブを経由すれば、何台分の LPC2148 にも接続できます。拡張性がいいです。パソコンも LPC2148 デバイスを使えます。

システムは ARM9/MINI2440 と ARM7/LPC2148 を同時に採用すれば、Linux の便利な機能と ARM7 のリアルタイム性を組み合わせ、高度複雑なアプリケーションとリアルタイム制御が両立できるシステムを作れます。



写真は MINI2440 と LPC2148 を USB で繋ぐ様子。

LPC2148 は CDC ACM デバイスとして動きます。ARM9 側から見ると、LPC2148 は USB シリアルポートです。ARM9 のターミナルで下のコマンドを入力

 # armcomtest –d /dev/ttyACM0

キーボードで情報を入力します。ARM9 はこの情報を ARM7 へ送信します。ARM7 は ARM9 が送った情報をそのまま ARM9 へ返信します。ARM9 はこの情報をコンソールで表示します。

LPC2148 側のサンプルは **USBtarget/examples/serial.hex** です。

ARM9 の Linux のコンフィグの手順：

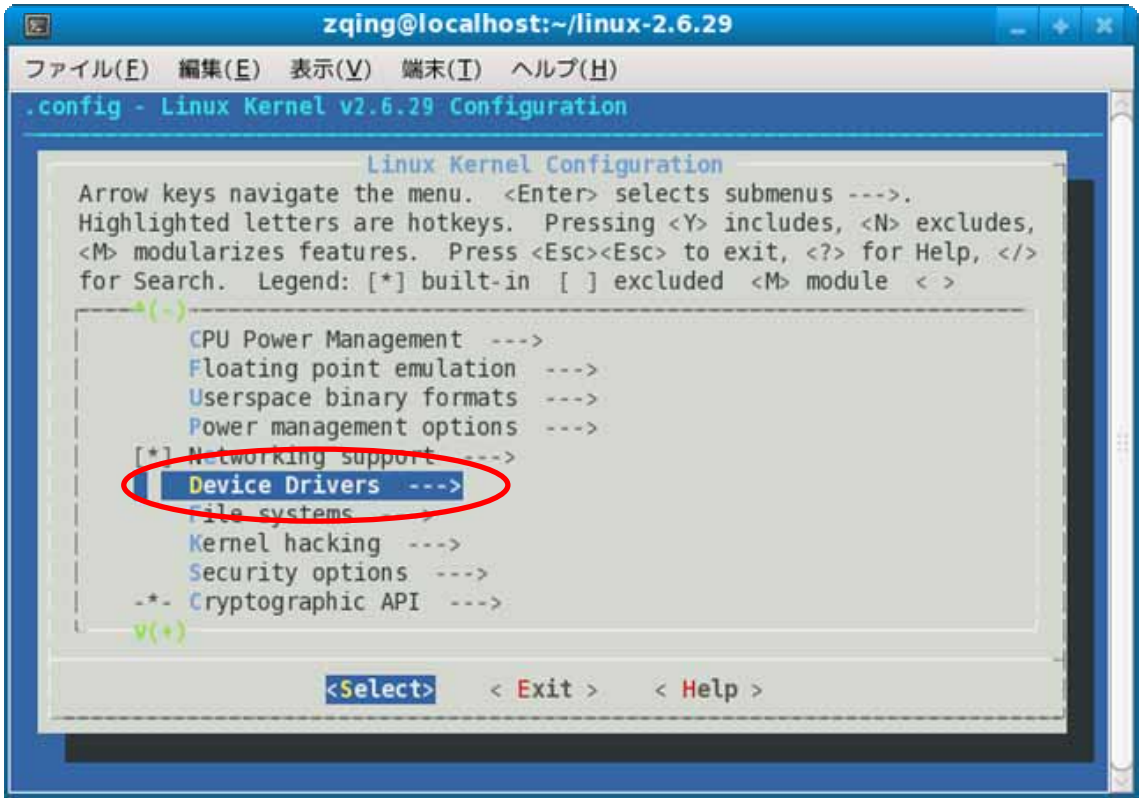
下のコマンドで Linux のコンフィグに入ります。

$ cd linux-2.6.29

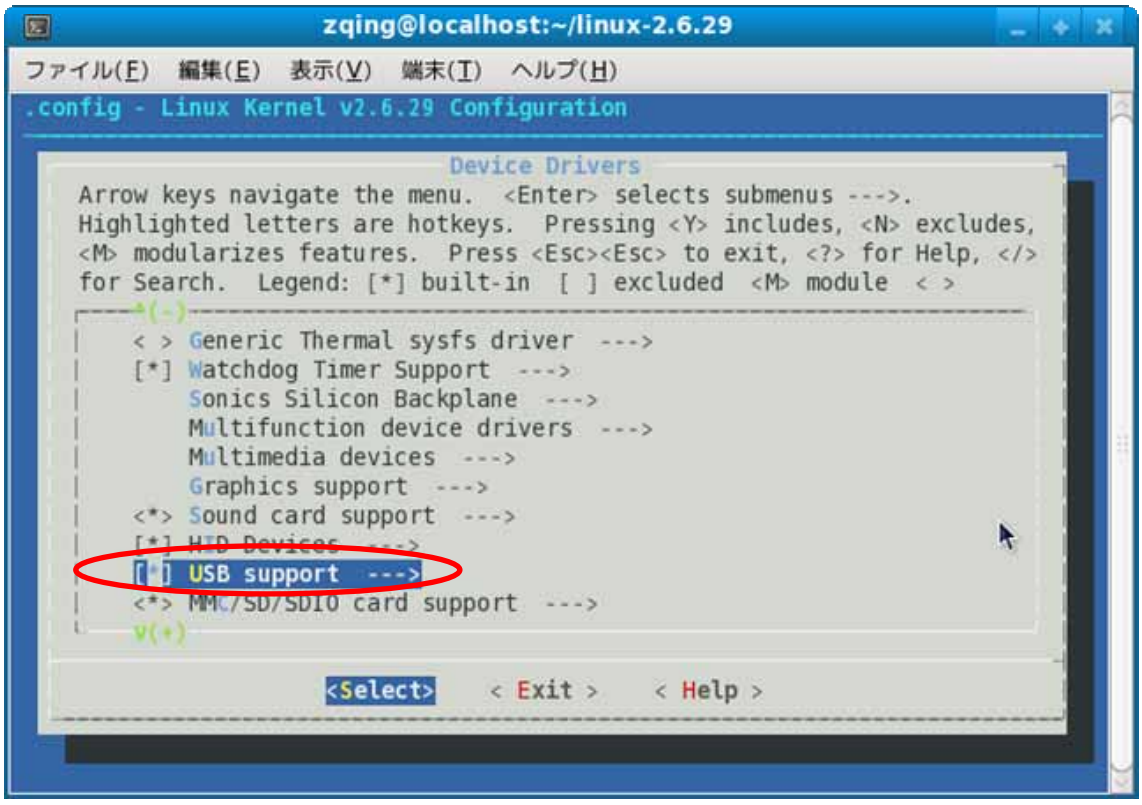
$ make clean

$ cp config_mini2440_n35 .config

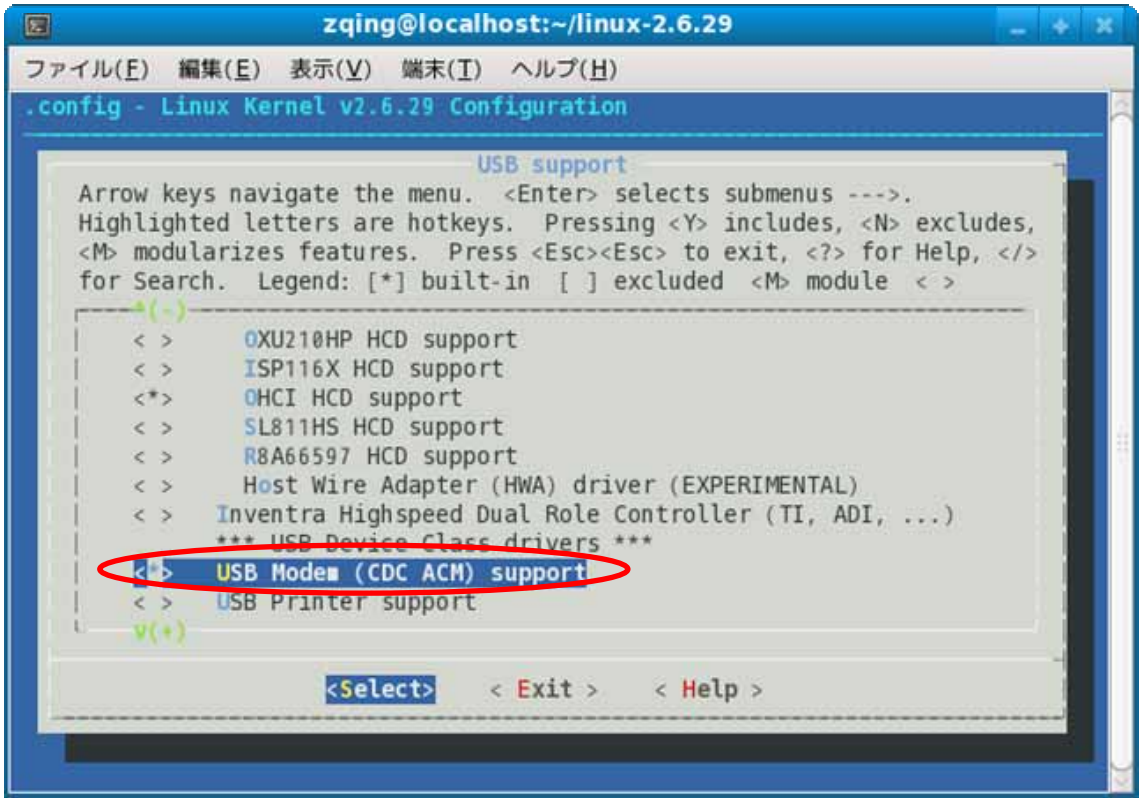
$ make menuconfig



「Device Drivers」を選択、



「USB support」を選択、



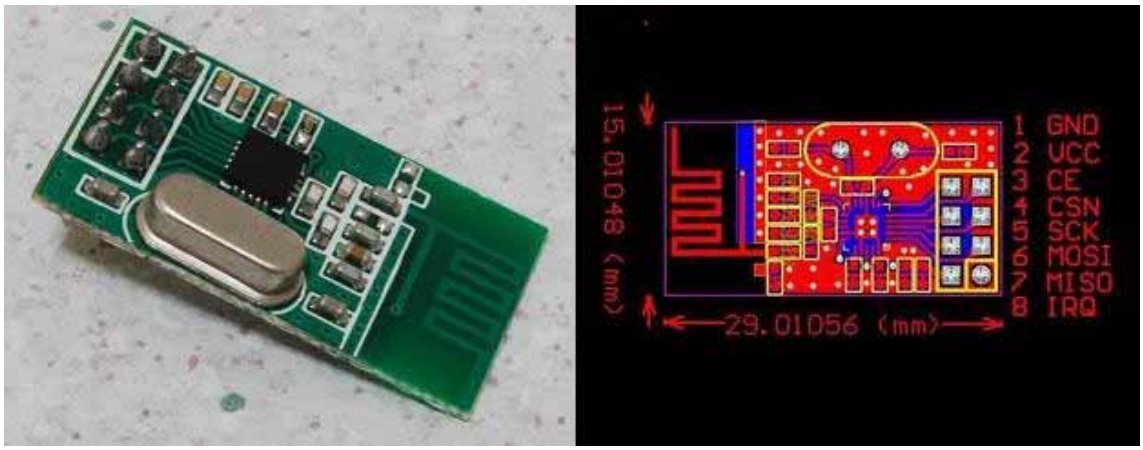
「USB Modem(CDC ACM) support」を選択します。“ Exit ” & “ Save ” します。

下のコマンドで Linux カーネルをコンパイルします。

$ make zImage

コンパイル完了すれば、**arch/arm/boot** フォルダには CDC ACM を含むカーネル **zImage** を生成します。この **zImage** ファイルを ARM9 に書き込んでください。詳しい情報は ARM9 のマニュアルをご参照ください。

第七章 微弱無線モジュール nRF24L01(2.4GHz)



弊社が販売している微弱無線モジュール nRF24L01(2.4GHz)

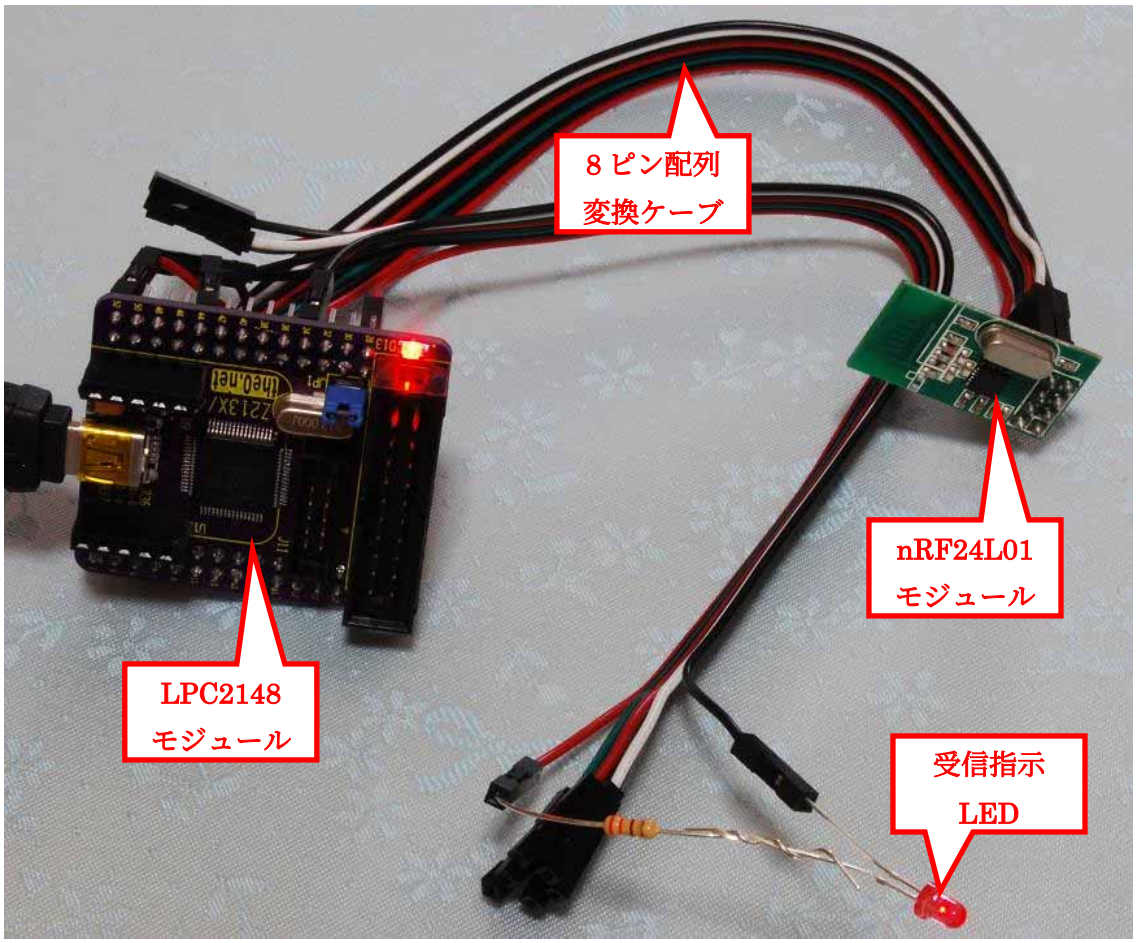
- **Nordic Semiconductor 社の NRF24L01 を搭載**
- **最大伝送速度: 2Mbps、VoIP 応用可能**
- **使用周波数帯: 2.4GHz ISM 帯**
- **変調方式: GFSK**
- **定格電圧: 1.9～3.6V**
- **SPI インターフェース**
- **使いやすい 2.54mmDIP**

nRF24L01 モジュールと LPC2148 モジュールの接続：

nRF24L01	LPC2148 モジュール	nRF2401	LPC2148 モジュール
1. GND	26. GND	2. VCC	52. 3.3V
3. CE	43. P1.24	4. CSN	9. P0.7(SSEL0)
5. SCK	6. P0.4(SCK0)	6. MOSI	8. P0.6(MOSI0)
7. MISO	7. P0.5(MISO0)	8. IRQ	44. P1.25(EXTIN0)

※ 始めの数字はモジュールのピン番号です。

受信指示 LED：LPC2148 モジュールの PIN34(P0.31)と PIN27(VBAT/3.3V)



送信用のプログラム：**nRF24L01/send.bin**

受信用のプログラム：**nRF24L01/rev.bin**

この実験は二つのセット(LPC2148+nRF24L01)が必要です。一つは送信です。もう一つは受信です。送受信成功すれば、受信指示 LED が点滅します。