Multi-Media ARM9

Windows CE 5.0

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

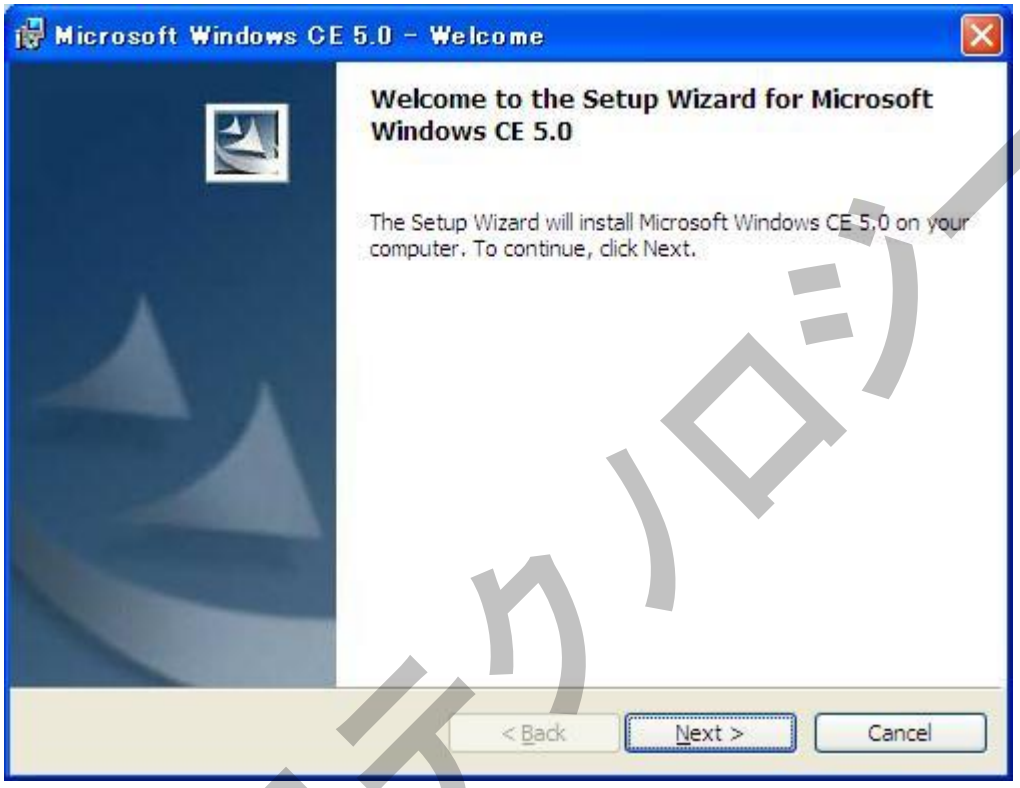
2010/05/10

※使用されたソースコードは http://csun.co.jp/#tabview=tab3&subview=xxx0 からダウンロードできます。

第一章 WinCE5.0 の構築環境

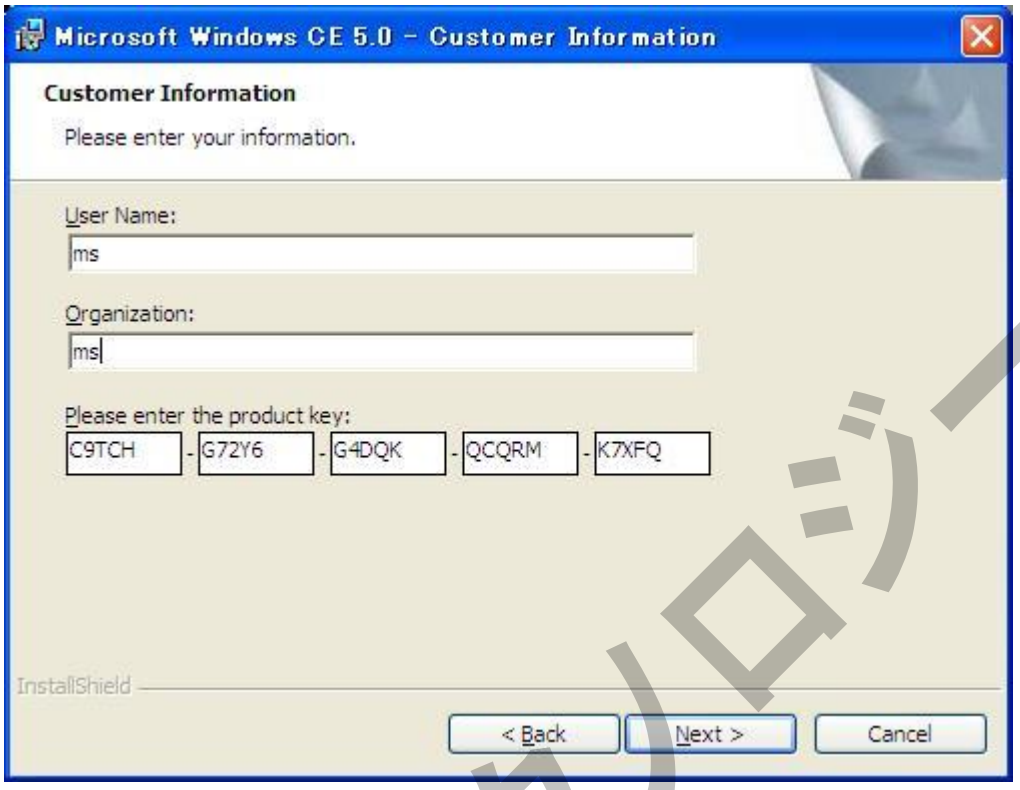
1.1 Platform Builder 5.0 をインストールする

Platform Builder 5.0-setup.exe を実行します。

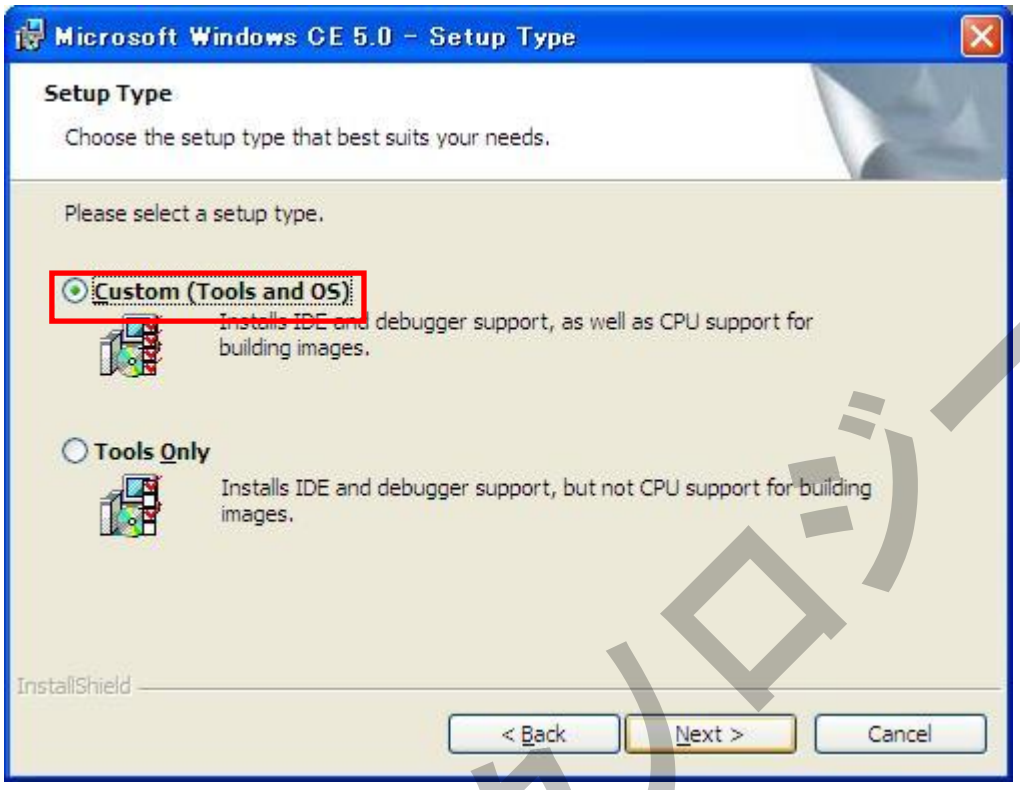


[Next]ボタンを押すと、次のようなダイアログが現れます。同意できる場合は、「I accept the terms in the license agreement」を選択して、「Next」ボタンを押します。

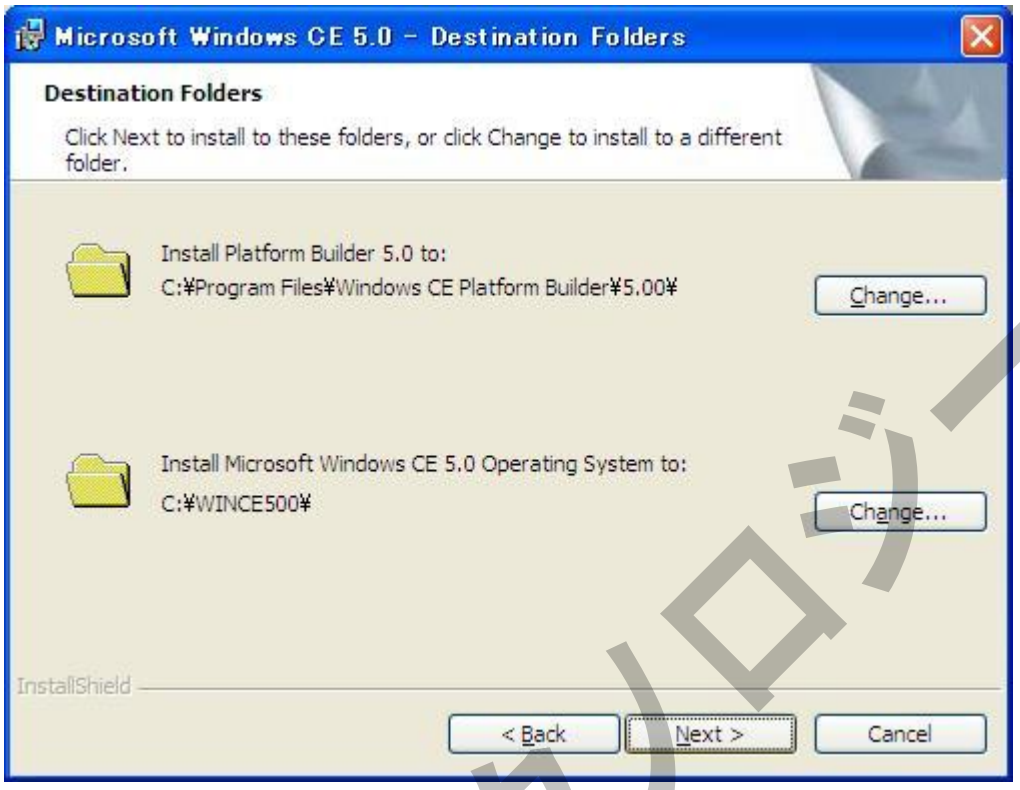
名前と会社名を入力して、「Next」ボタンを押します。



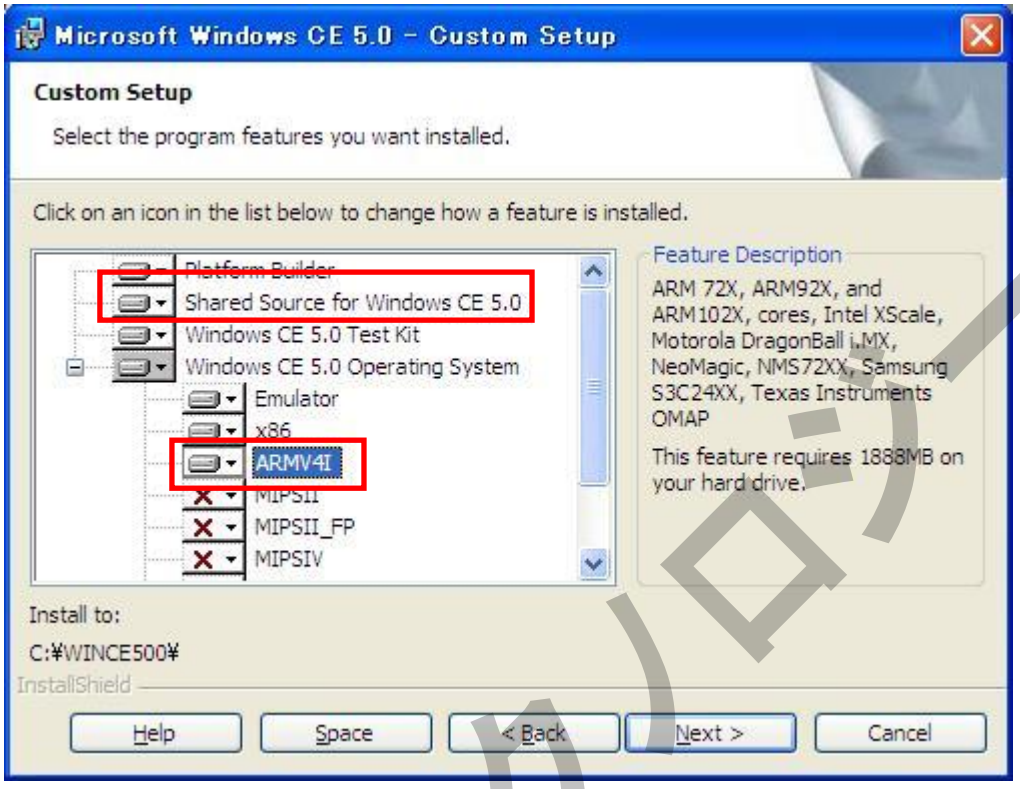
Custom(tools and OS)を選択して、「Next」ボタンを押します。



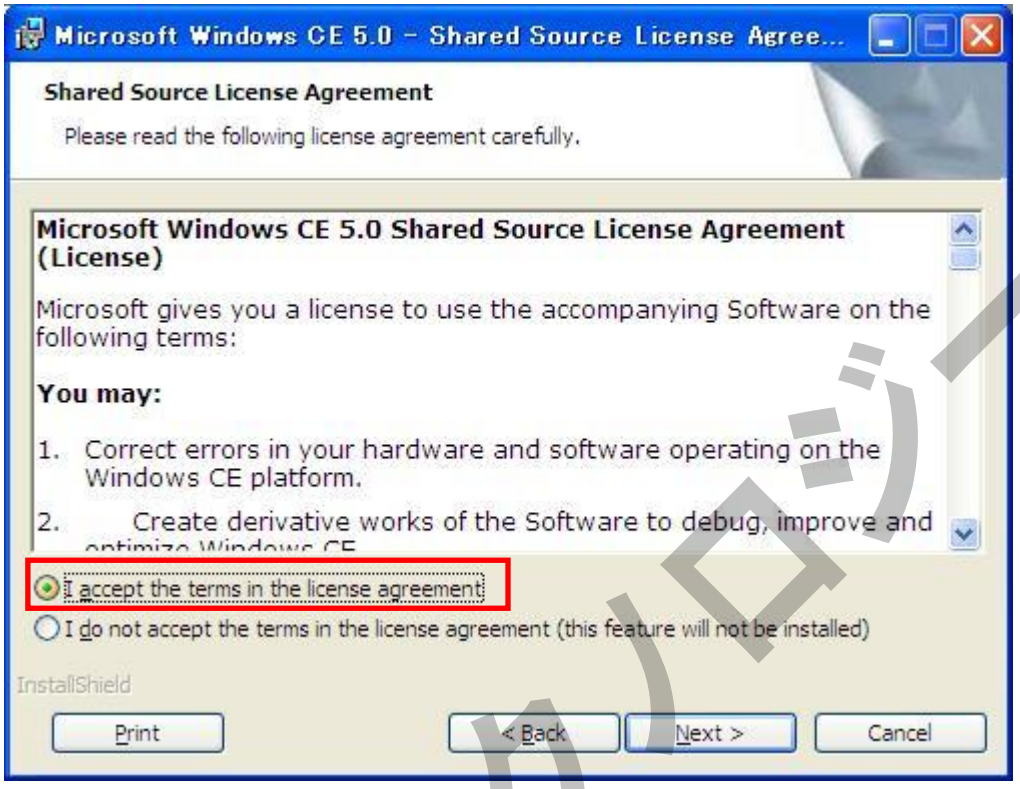
インストール先フォルダの指定画面です。変更せず、そのまま進んできださい。



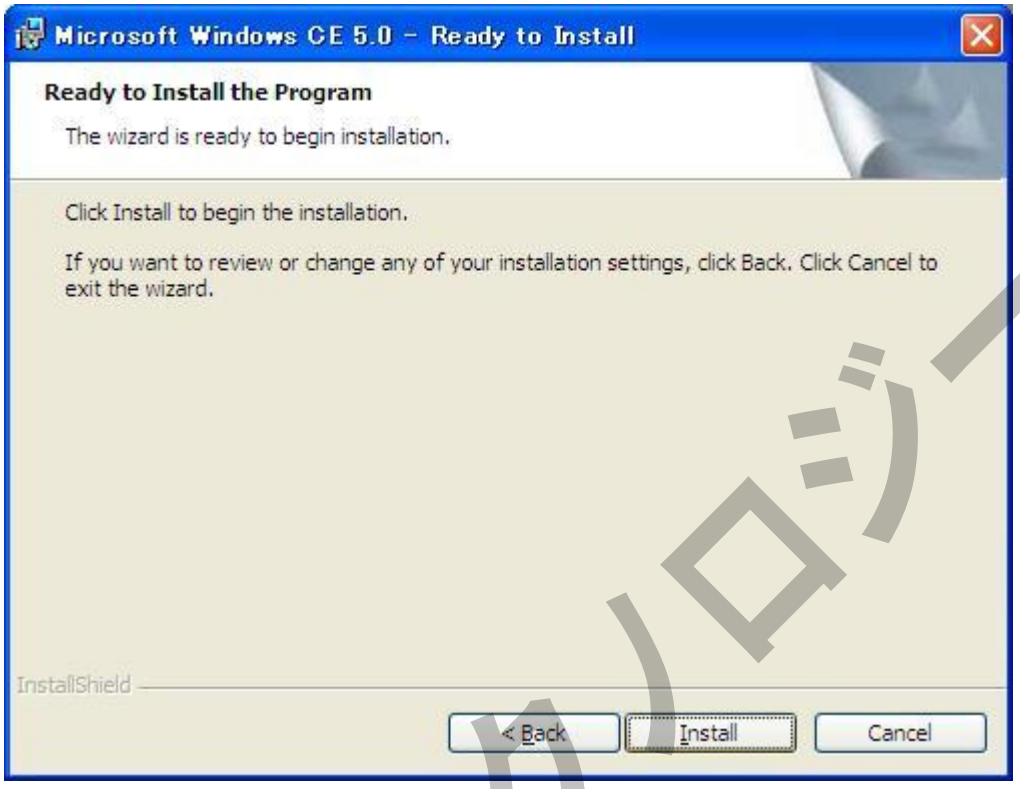
インストールしたいパッケージを選択します。特に[Shared Source Windows CE 5.0]と[ARMV4I]を添加します。「Next」ボタンを押します。



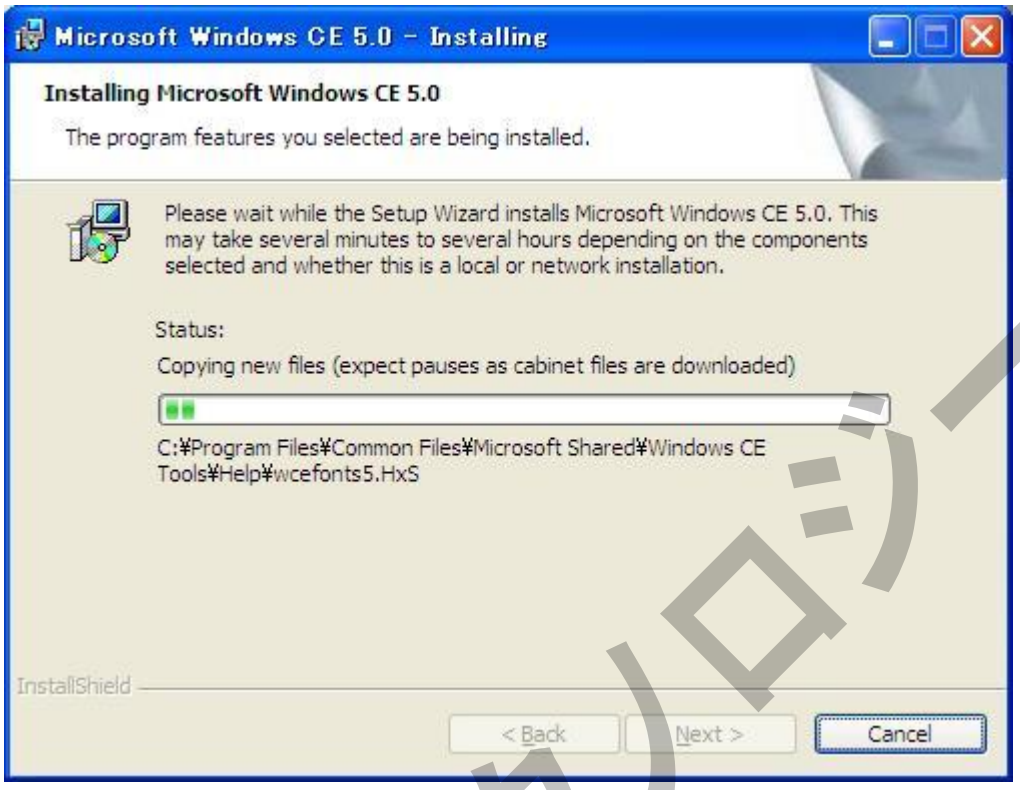
ライセンスを読み、同意できる場合は、「**I accept the terms in the license agreement**」を選択して、「Next」ボタンを押します。



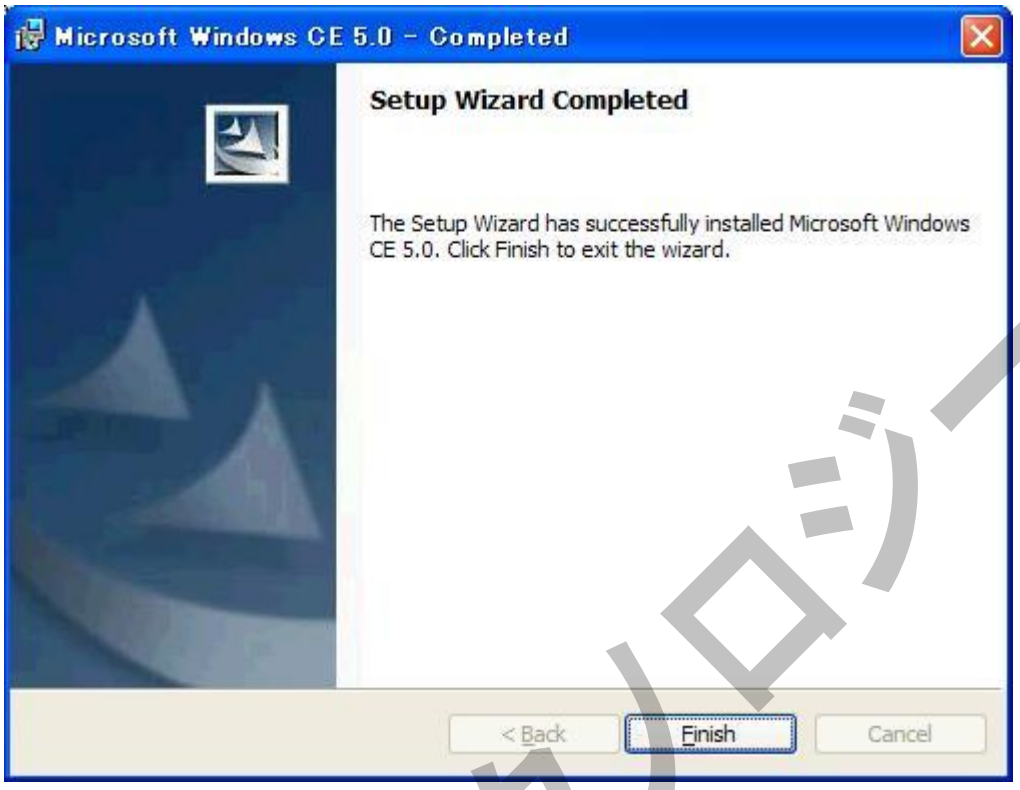
インストールの準備できました。「Install」ボタンを押して、インストールをスタートします。



時間がかかりますので、我慢してください。

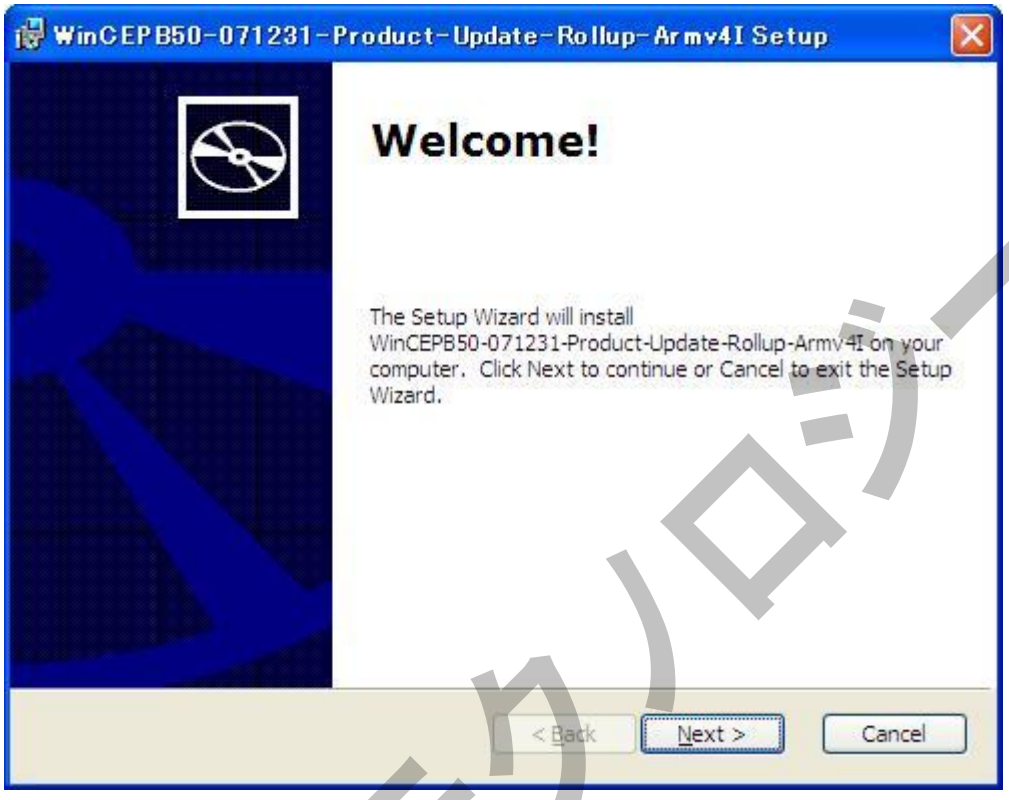


インストールが完了した画面です。「Finish」ボタンを押します。

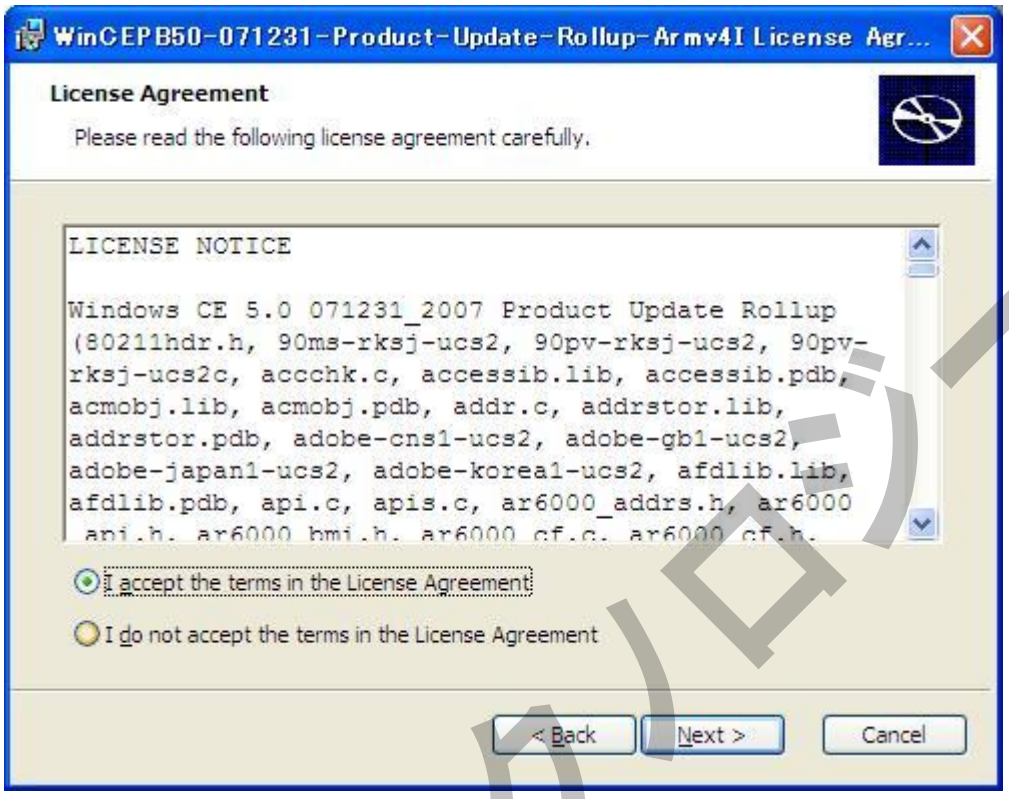


1.2 パッチをインストール(VS2008 のため)

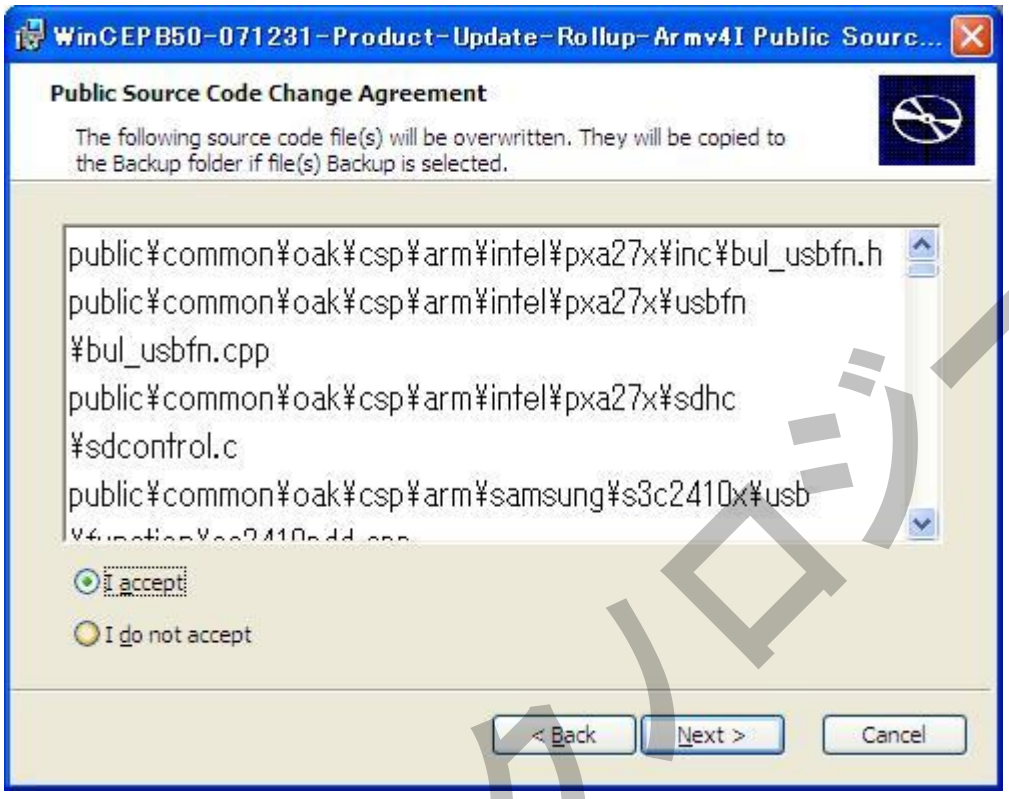
WinCEPB50-071231-Product-Update-Rollup-Armv4I.msi を実行します。



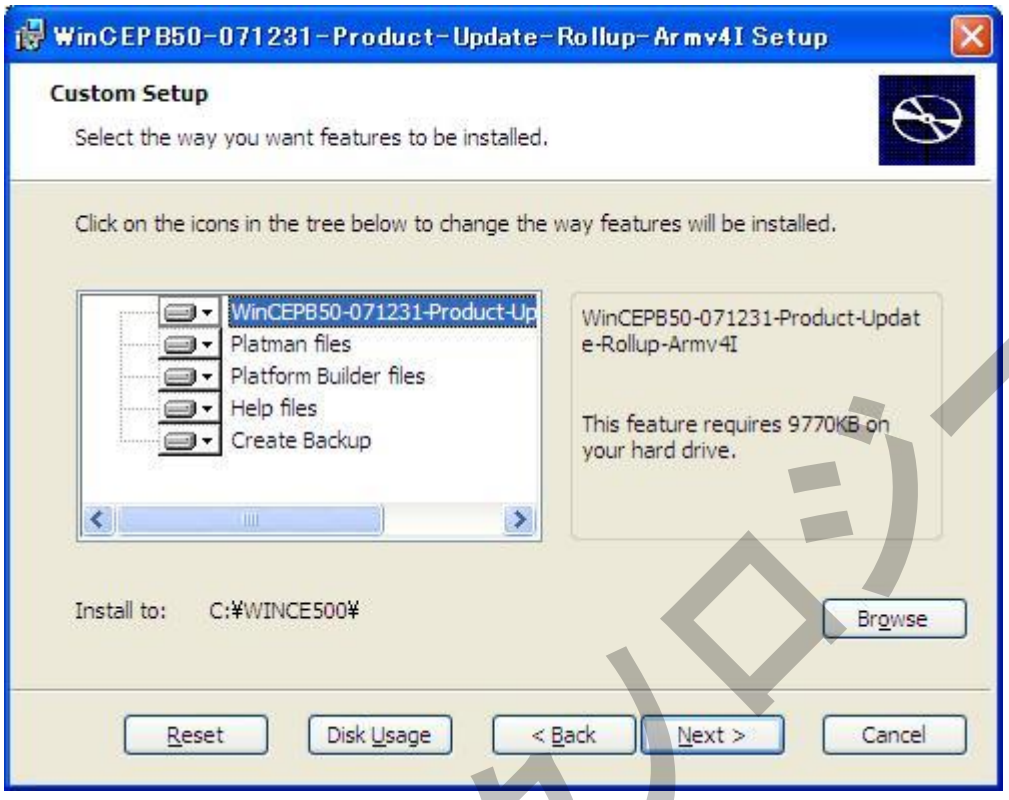
英文のライセンスを読み、同意できる場合は、I accept the terms in the license agreement を選択して、「Next」ボタンを押します。



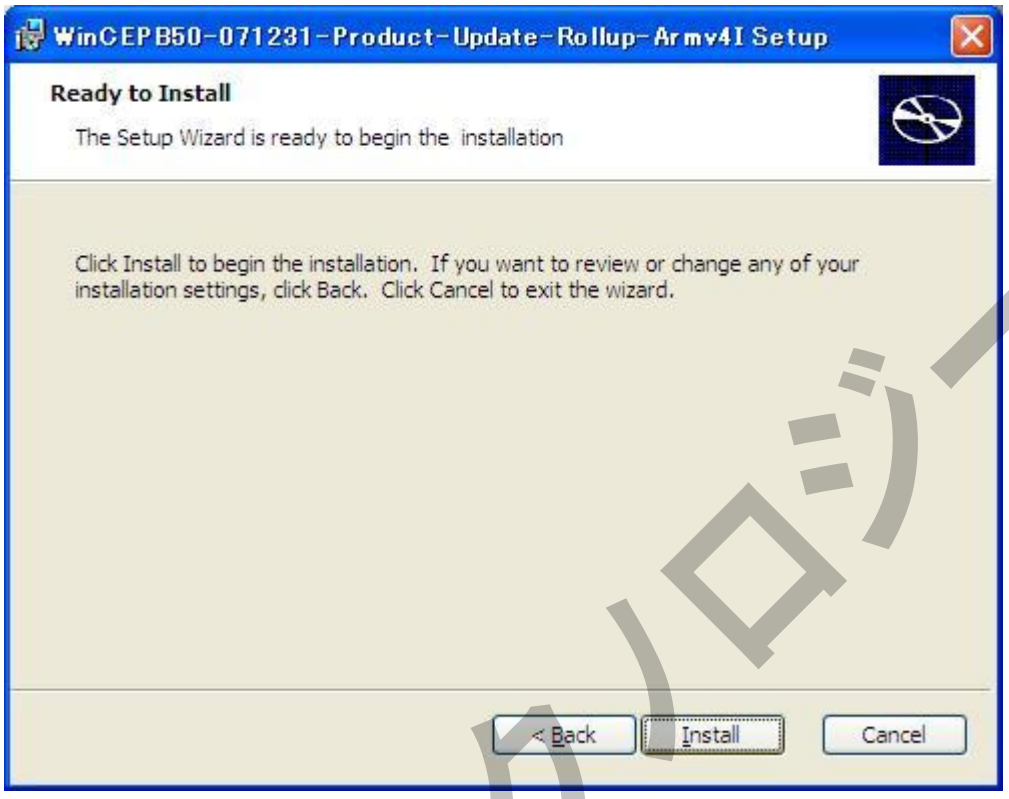
英文のライセンスを読み、同意できる場合は、I accept を選択して、「Next」ボタンを押します。



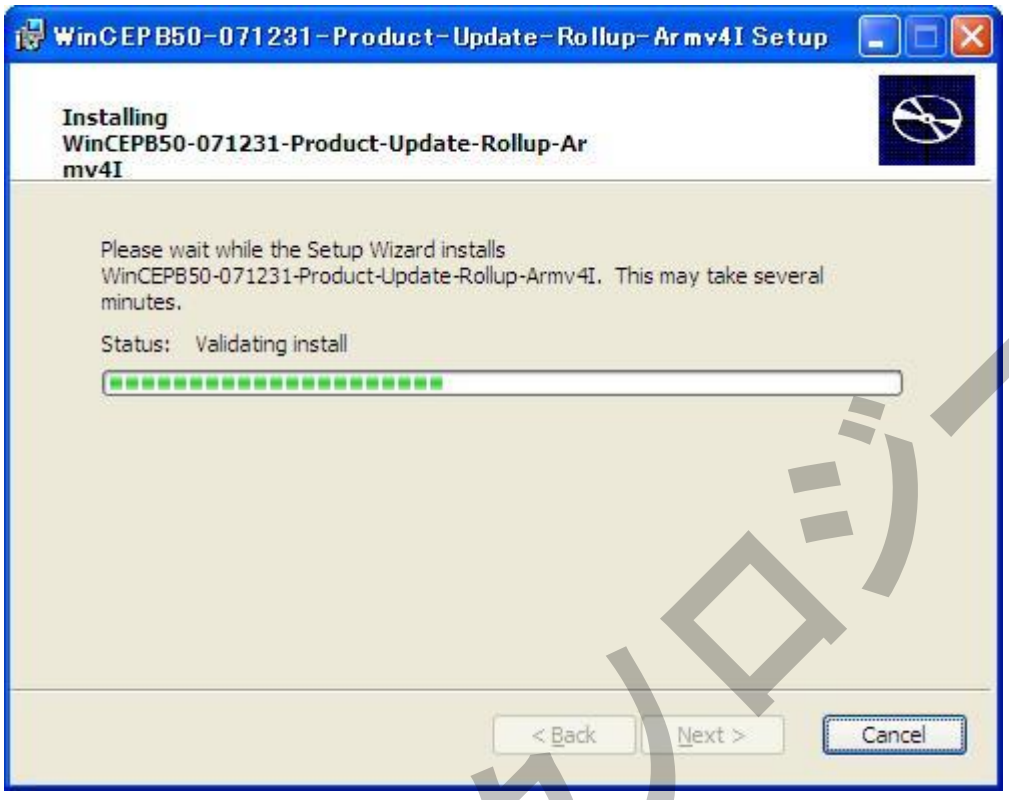
全部を選択して、「Next」ボタンを押します。



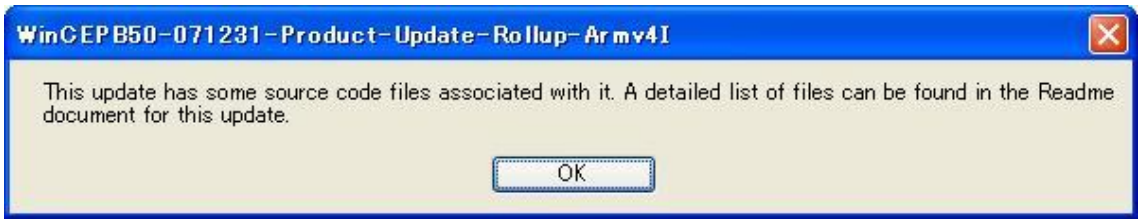
インストールの準備できました。「Install」ボタンを押して、インストールをスタートします。



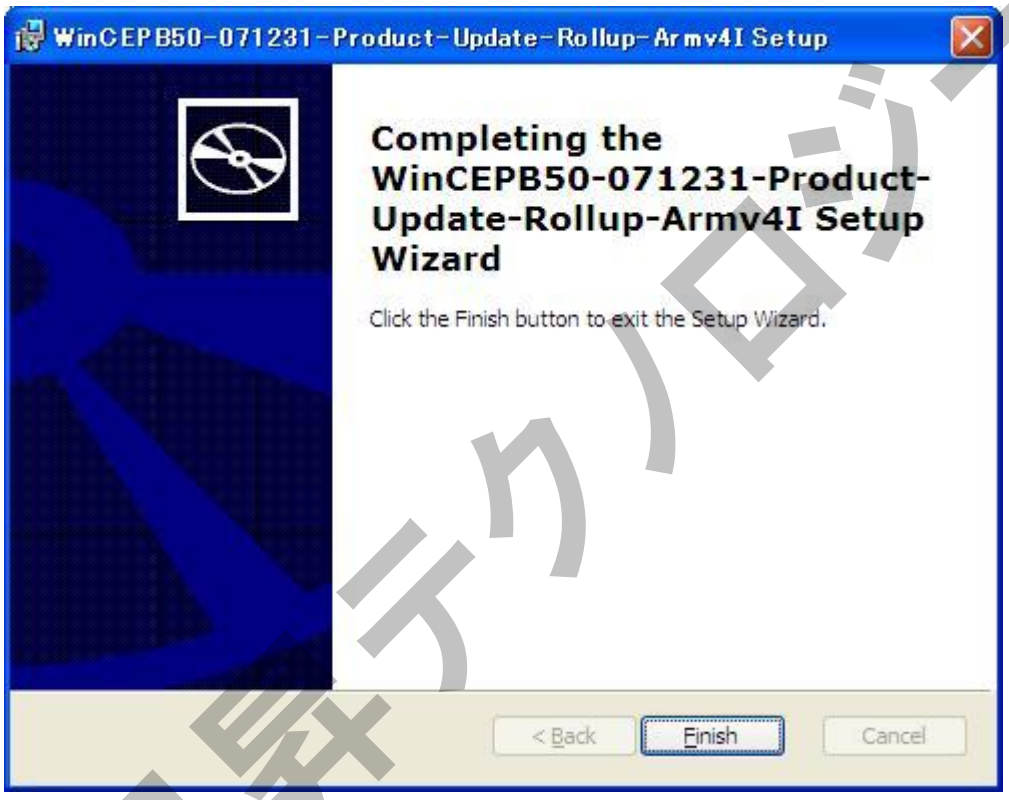
時間がかかりますので、我慢してください。



インストール途中で、この画面が出てきます。「OK」ボタンをおした、消してください。



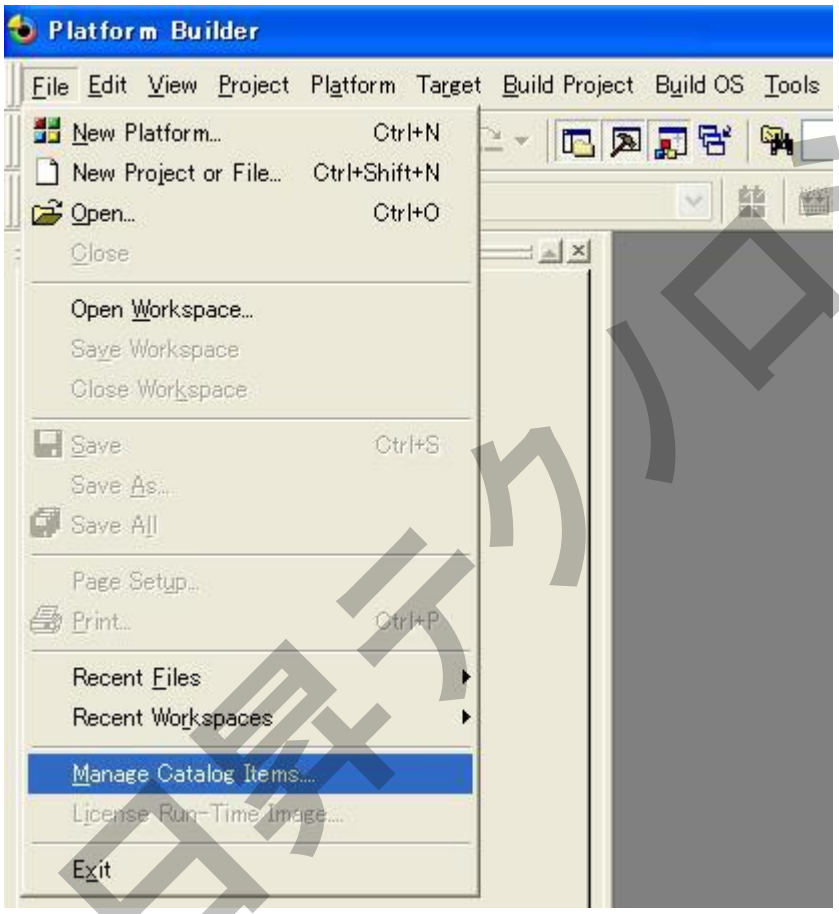
インストールが完了した画面です。「Finish」ボタンを押します。



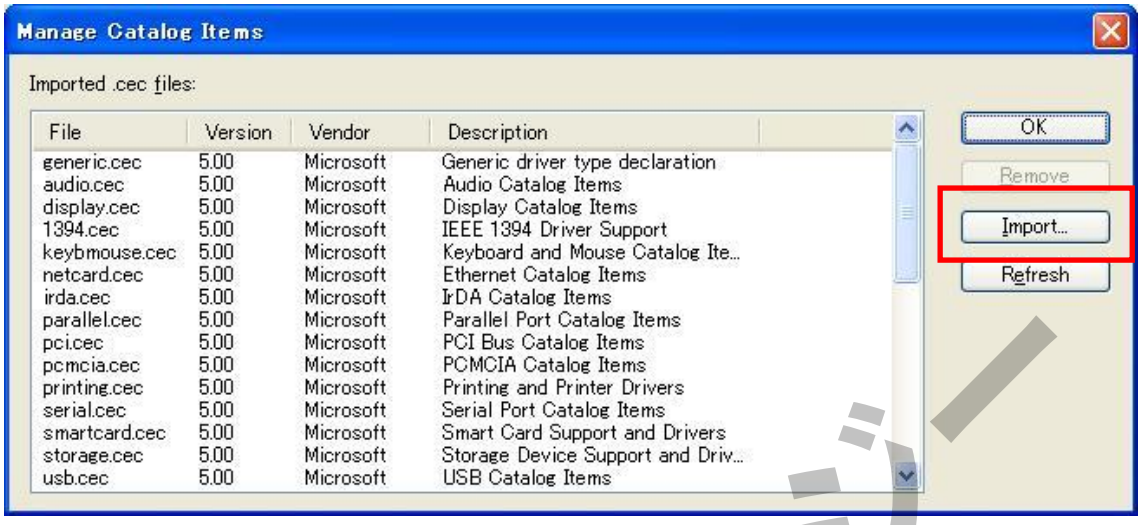
1.3 ARM9 ボードの BSP をインストールする

ARM9 ボードの型番(QQ2440v3/mini2440)と使用された液晶(3 インチ又は 7 インチ)にとって、BSP ファイル(*.rar)を選択します。選択された BSP を C:¥WINCE500¥PLATFORM で解凍します。

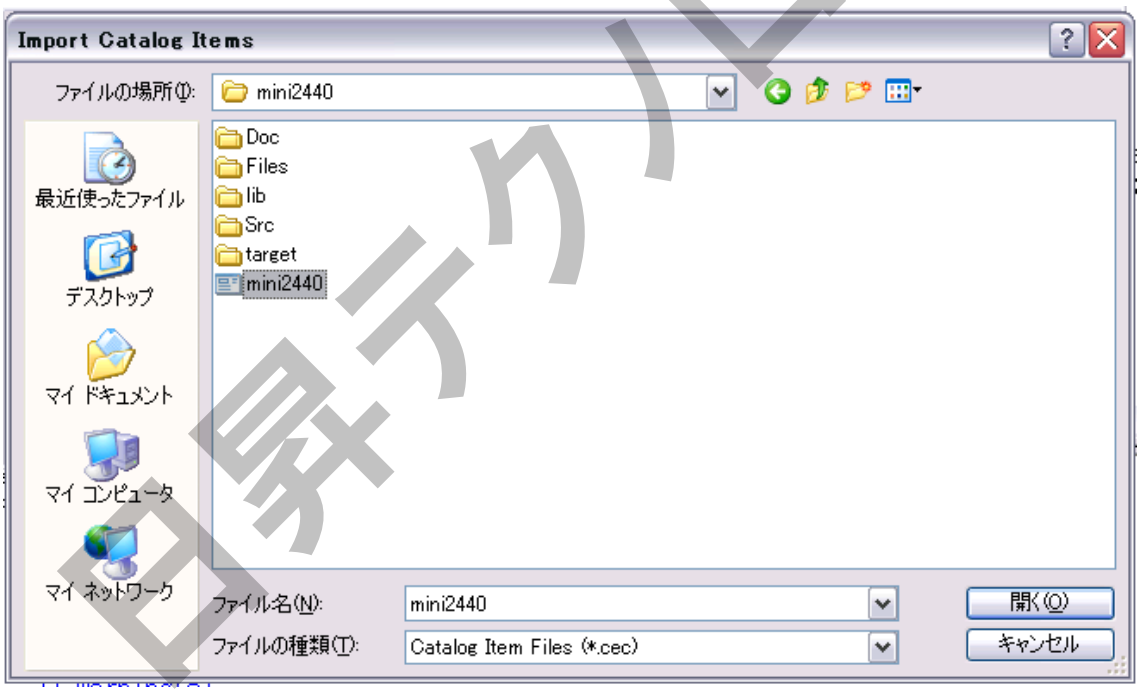
先ずインストールした Platform Builder 5.0 を実行します。メニュー「**File**」→「**Manage Catalog Items**」を選択します。



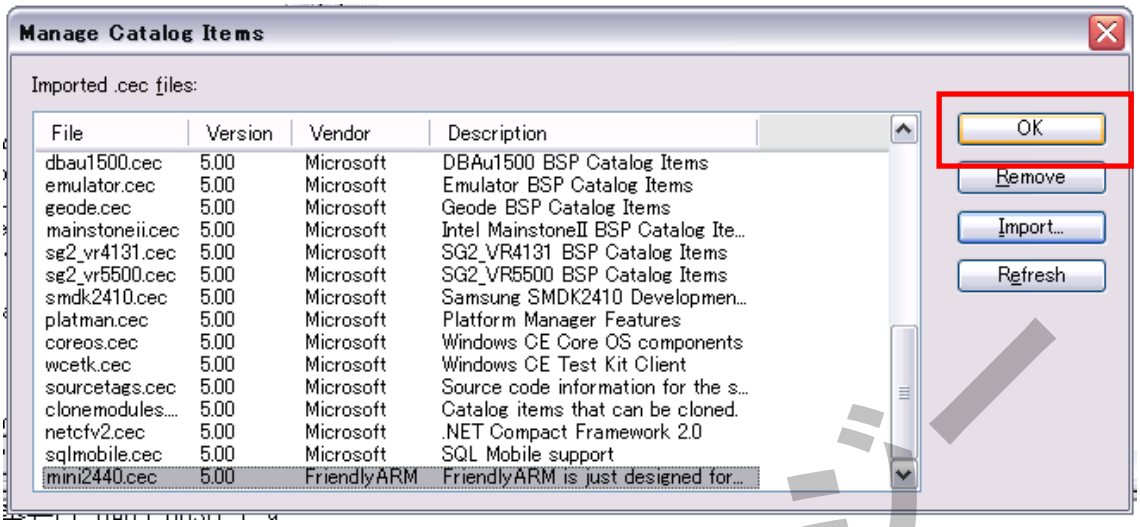
「**Import**」ボタンを押します。



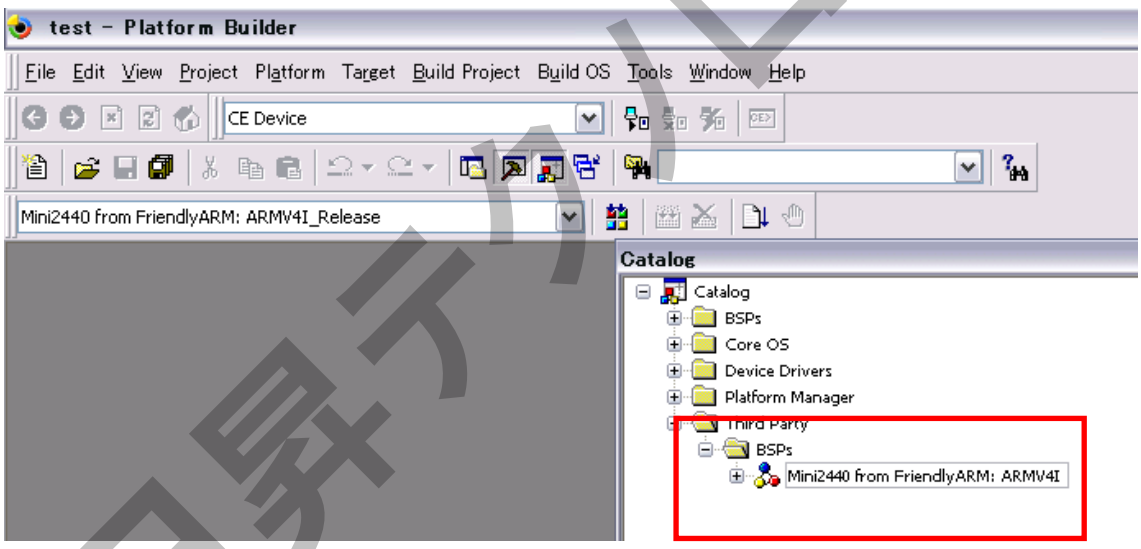
BSP フォルダの「mini2440.cec」というファイルを選択します。



「OK」ボタンを押します。

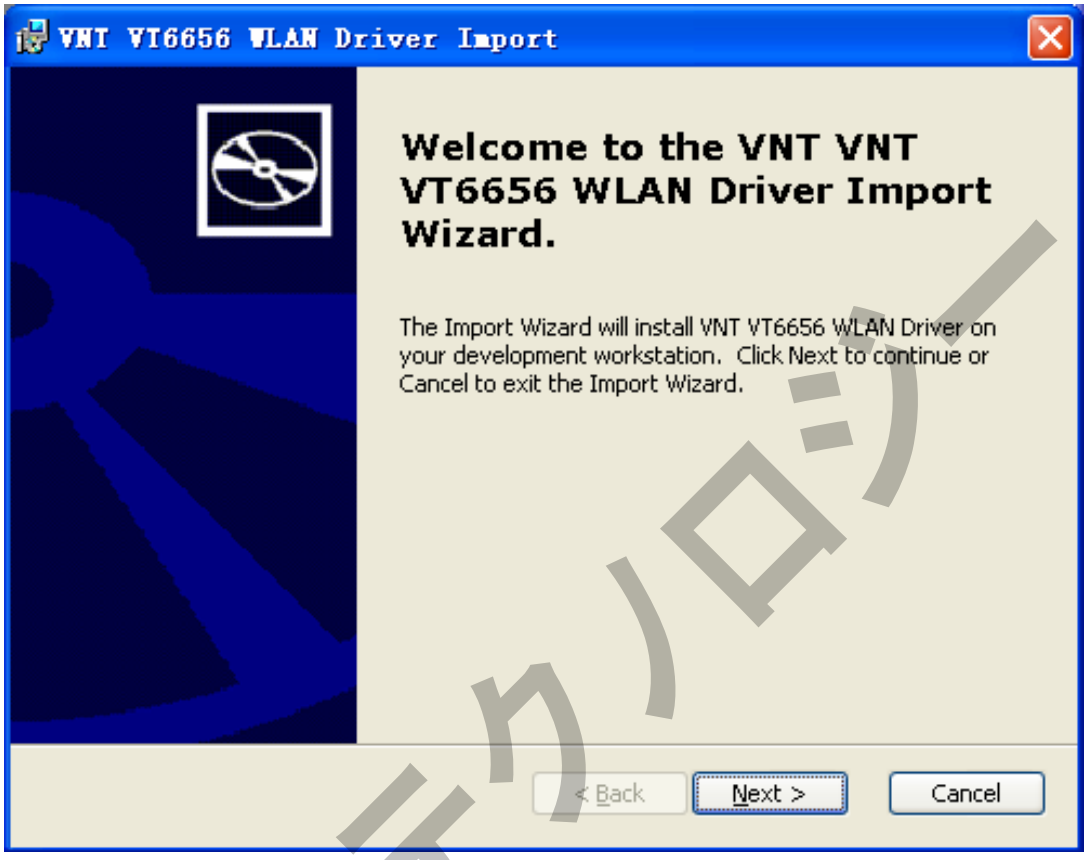


Platform Builder 5.0 の「Catalog」に「**mini2440**」という BSP が見えます。

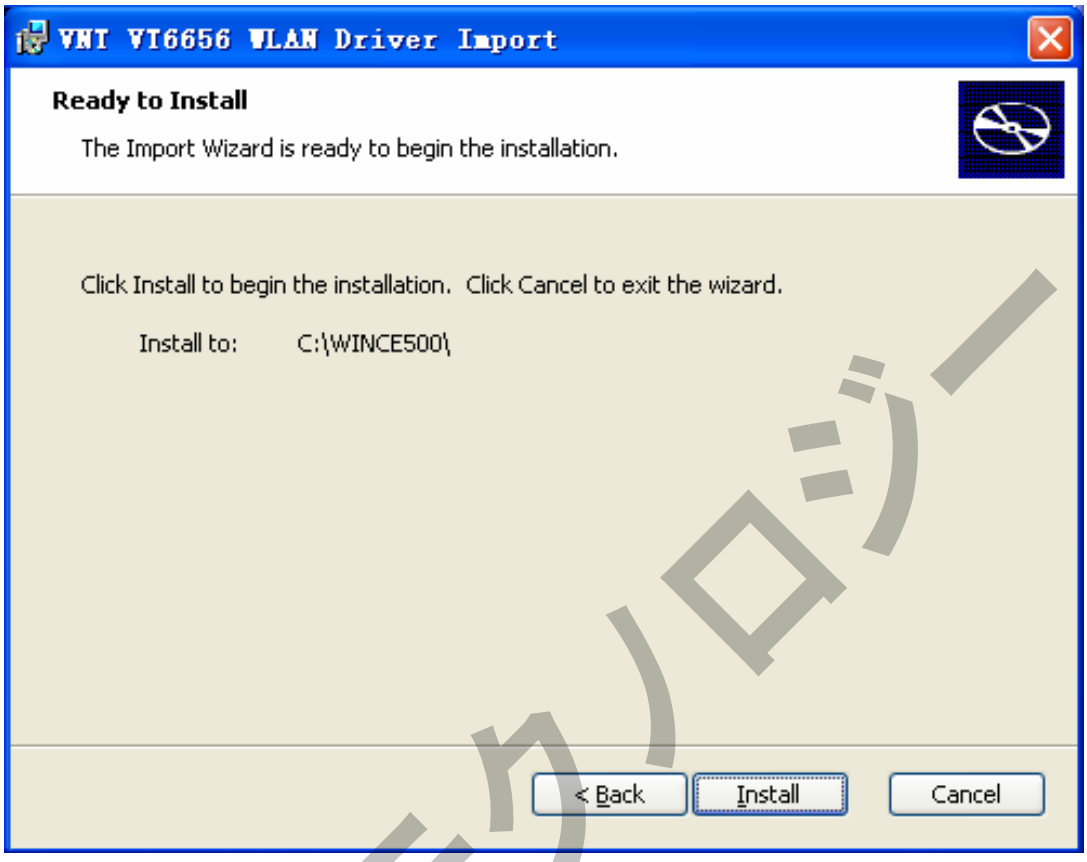


1.4 USB 無線 LAN のドライバをインストールする

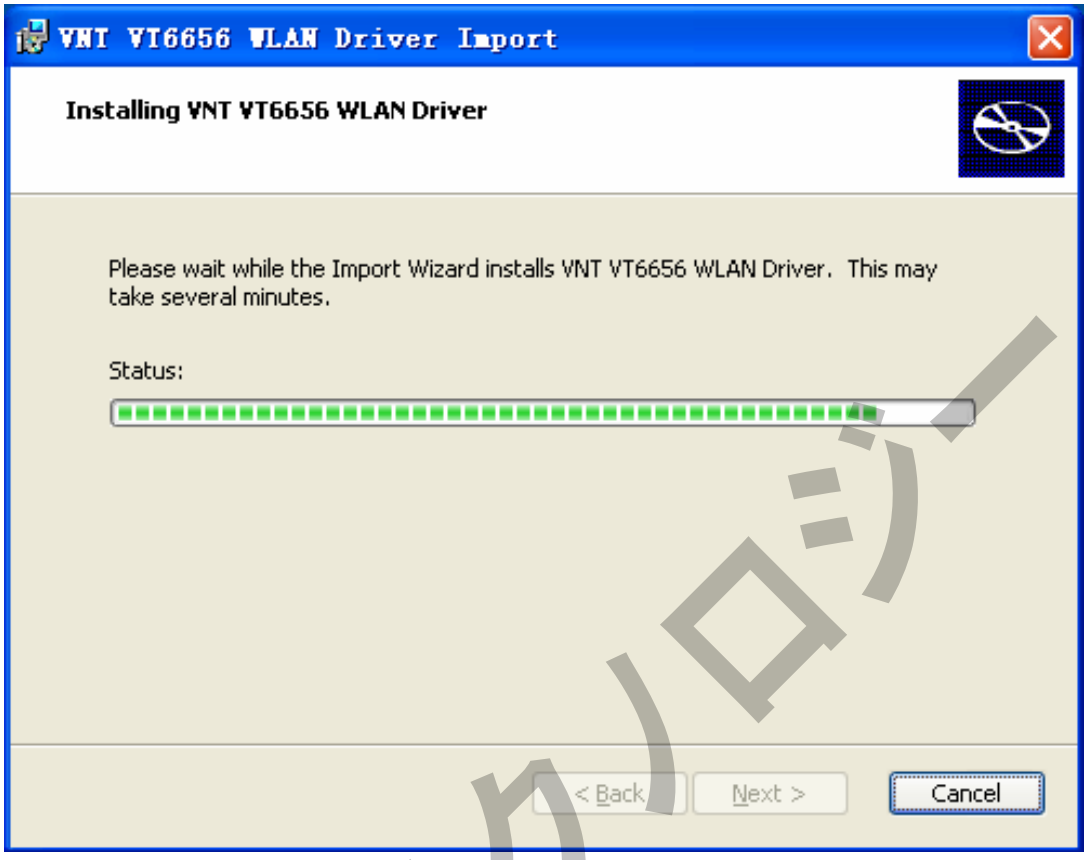
VNUWLC5-ARM.msi を実行します。「Next」ボタンを押します。



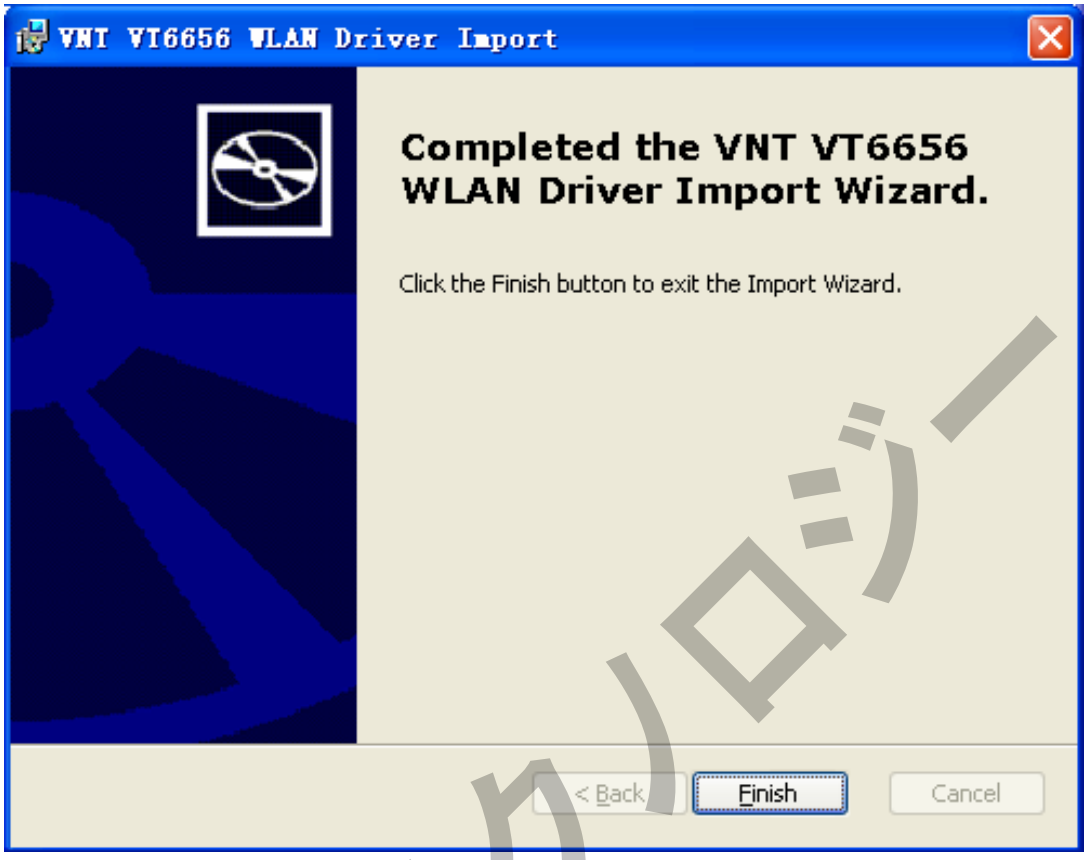
インストールの準備できました。「Install」ボタンを押して、インストールをスタートします。



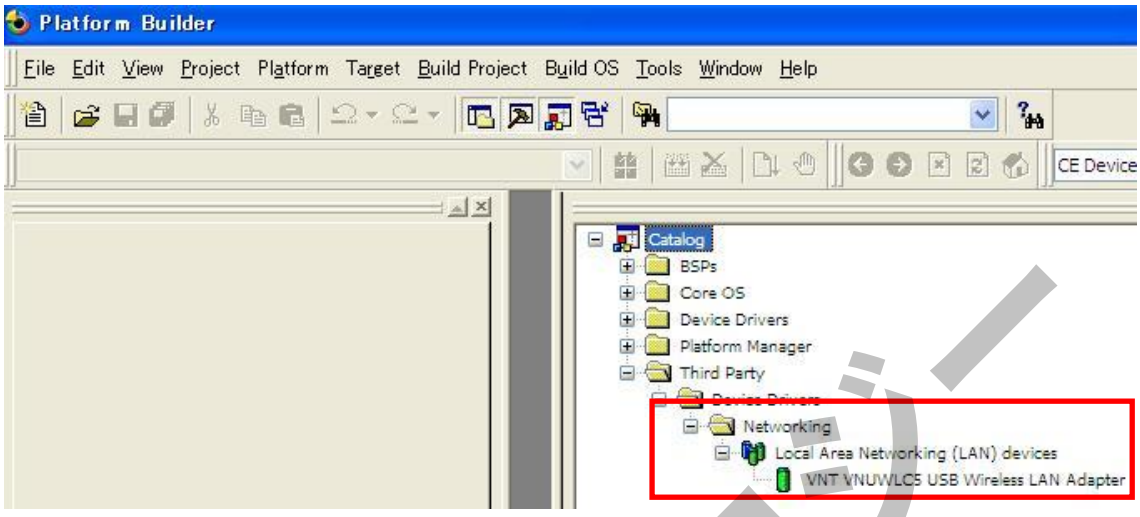
インストールがしばらく完了します。



インストールが完了した画面です。「Finish」ボタンを押します。



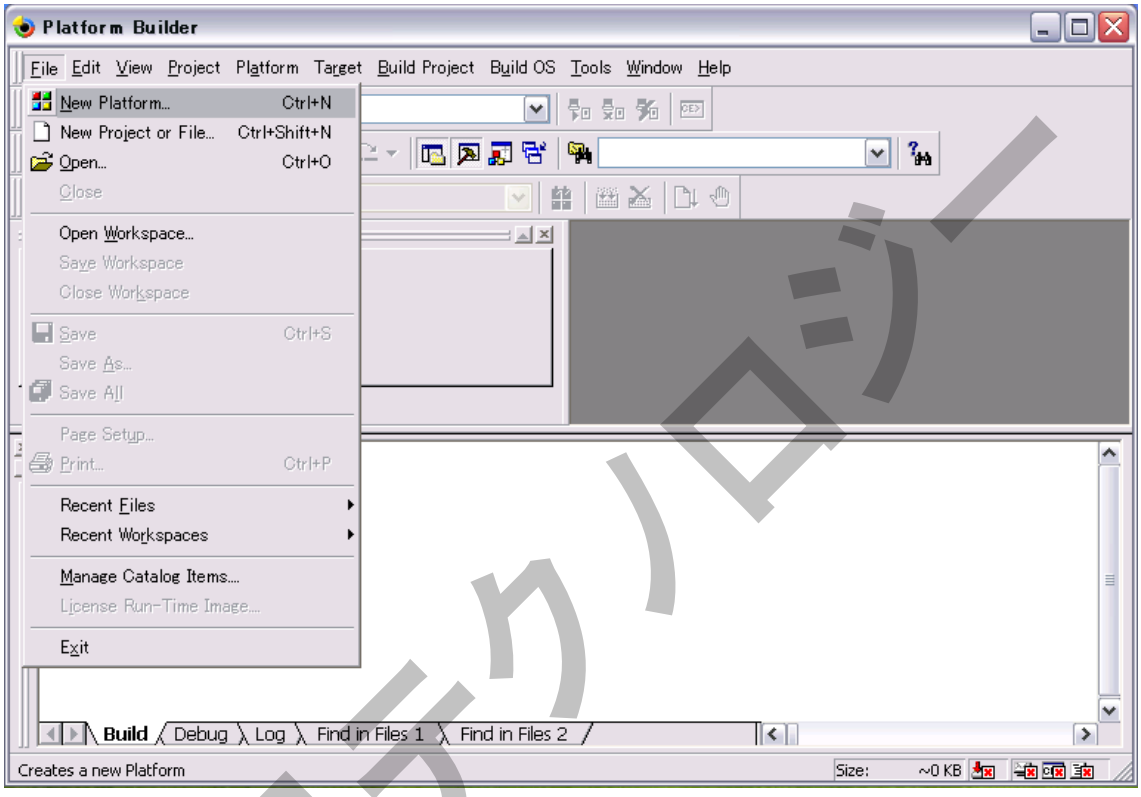
再び Platform Builder 5.0 を実行すると、「Catalog」に USB 無線 LAN のドライバが見えます。



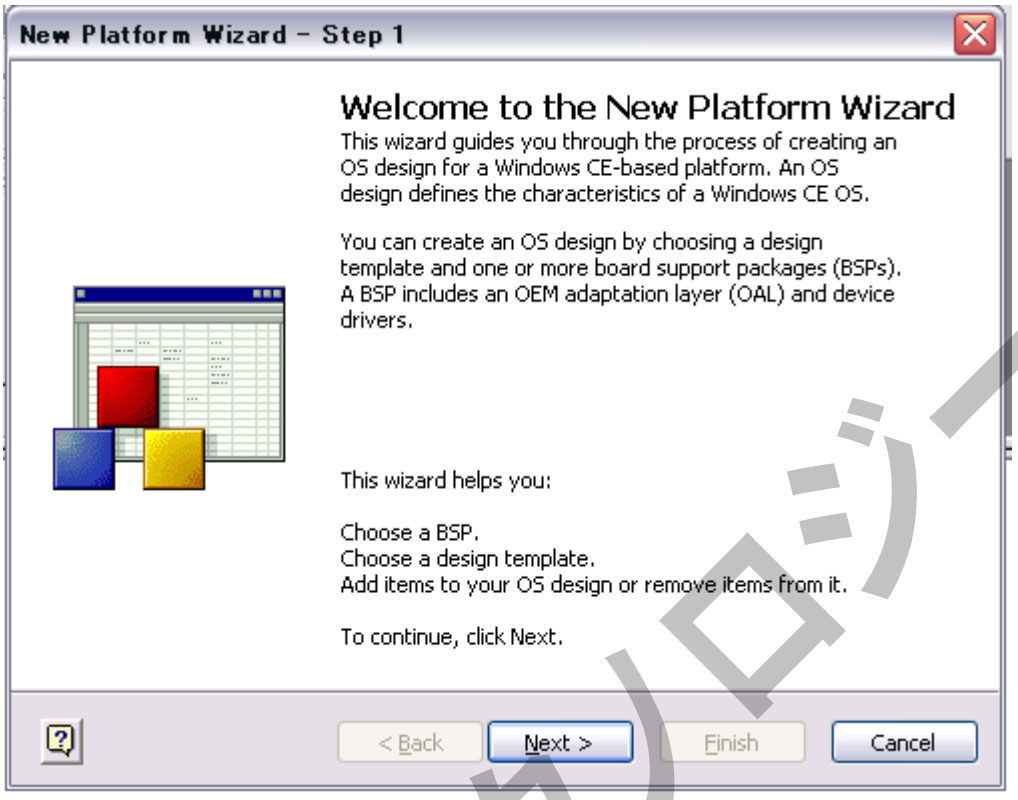
第二章 WinCE5.0 を構築する

2.1 新規プラットフォームを作ります

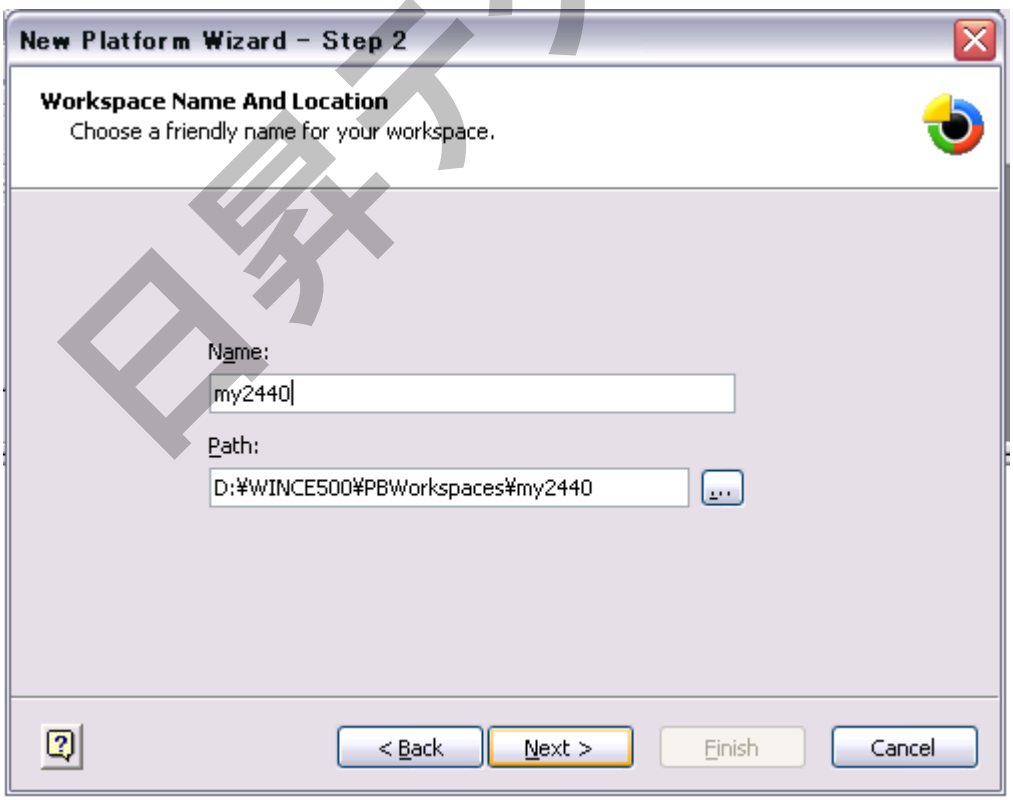
Platform Builder 5.0 を実行して、「**File**」→「**New Platform**」を選択します。



「Next」ボタンを押します。

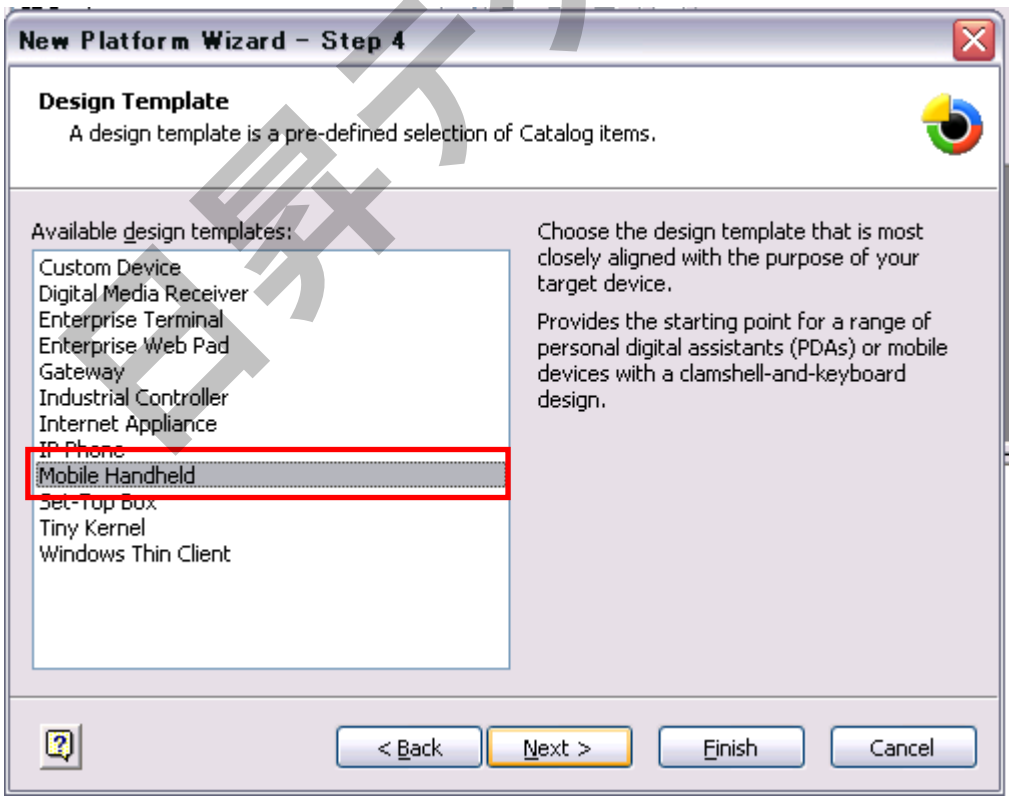


ワークスペースの名前と場所を入力します。

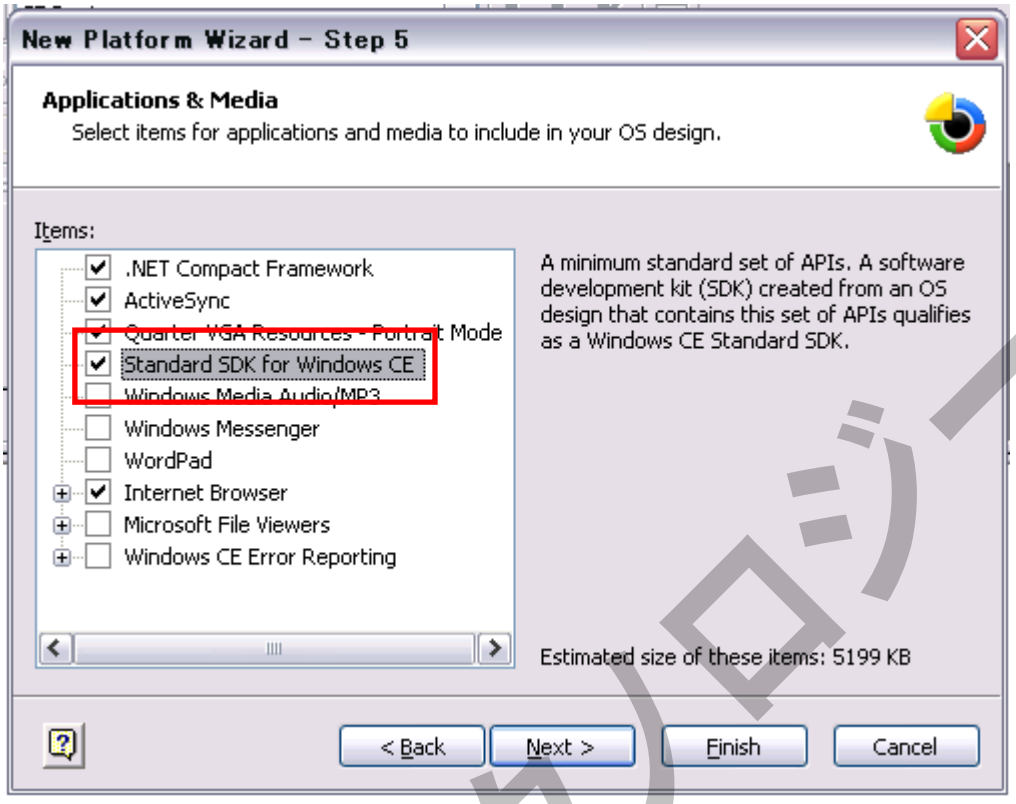


Mini2440 という BSP を選択します。「Next」ボタンを押します。

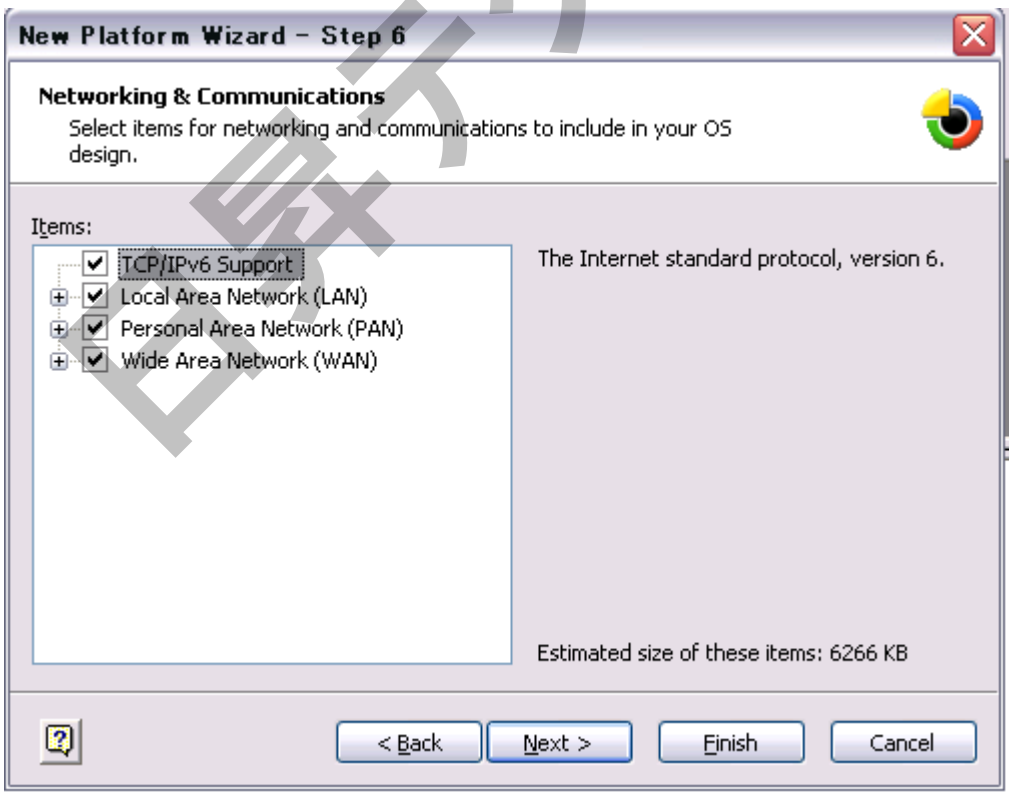
「Mobile Handheld」を選択します。「Next」ボタンを押します。



「Standard SDK for Windows CE」という項目を追加します。「Next」ボタンを押します。



ネットワークの設定、そのまま進んでください。

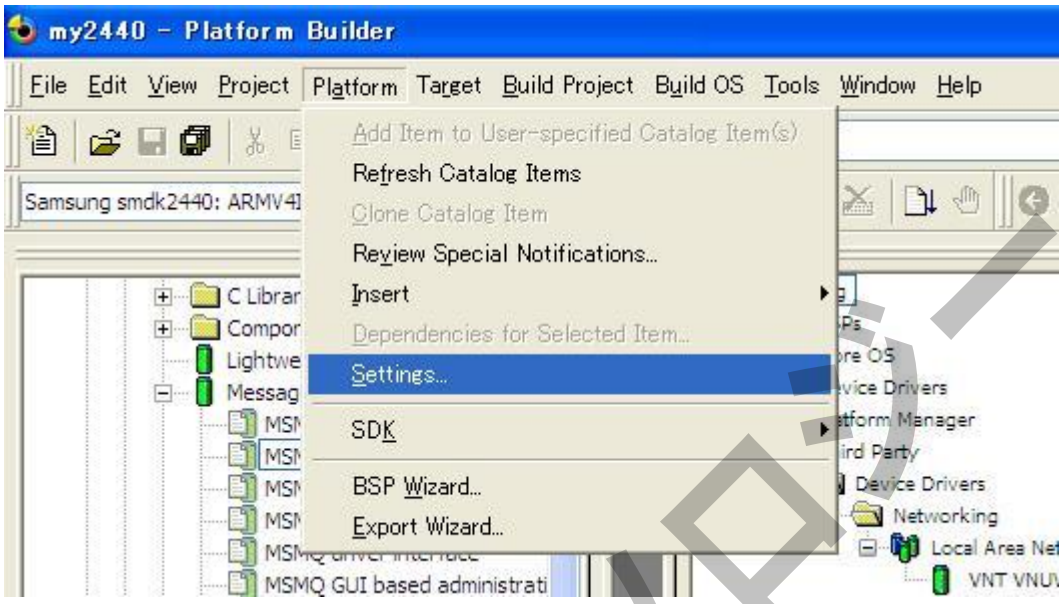


無線関連の設定、そのまま進んでください。

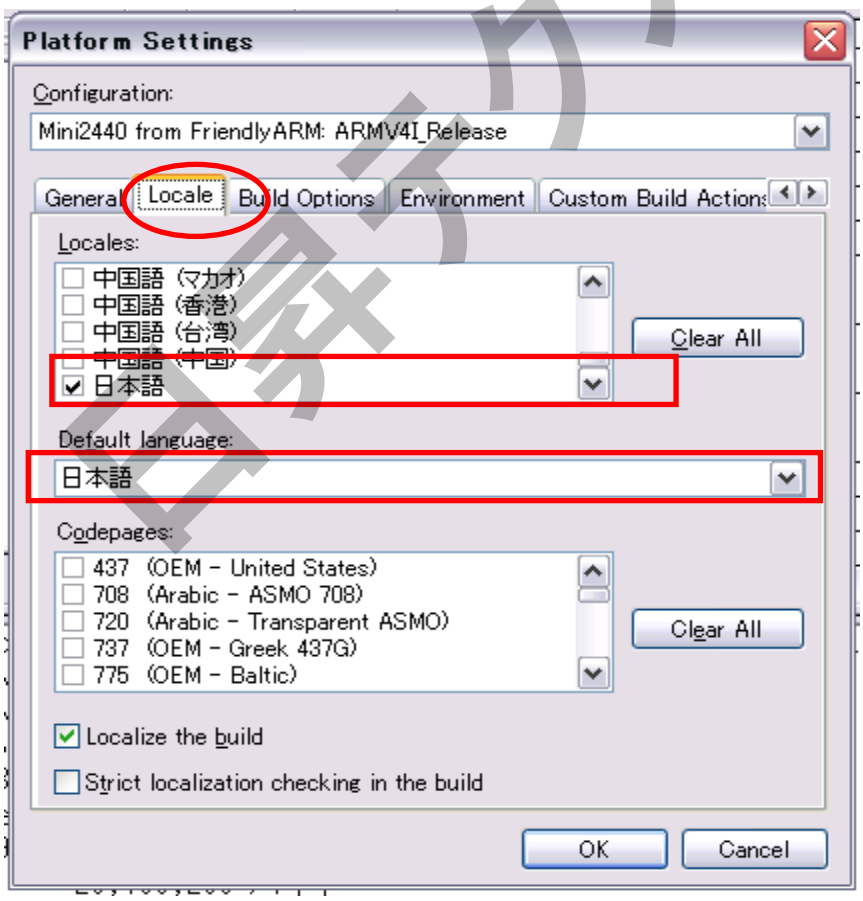
プラットフォーム作成が完了します。

2.2 使用したい言語を選択する

Platform Builder 5.0 のメニュー「Platform」→「Settings」を選択します。

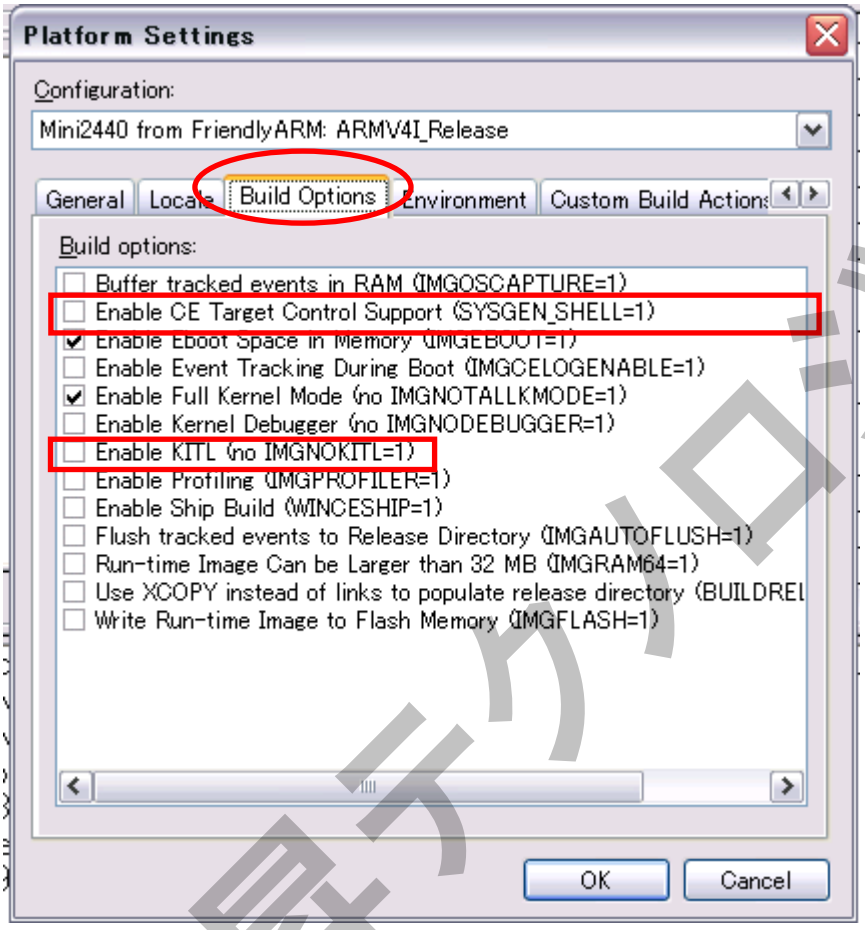


Locale で日本語を選択します。



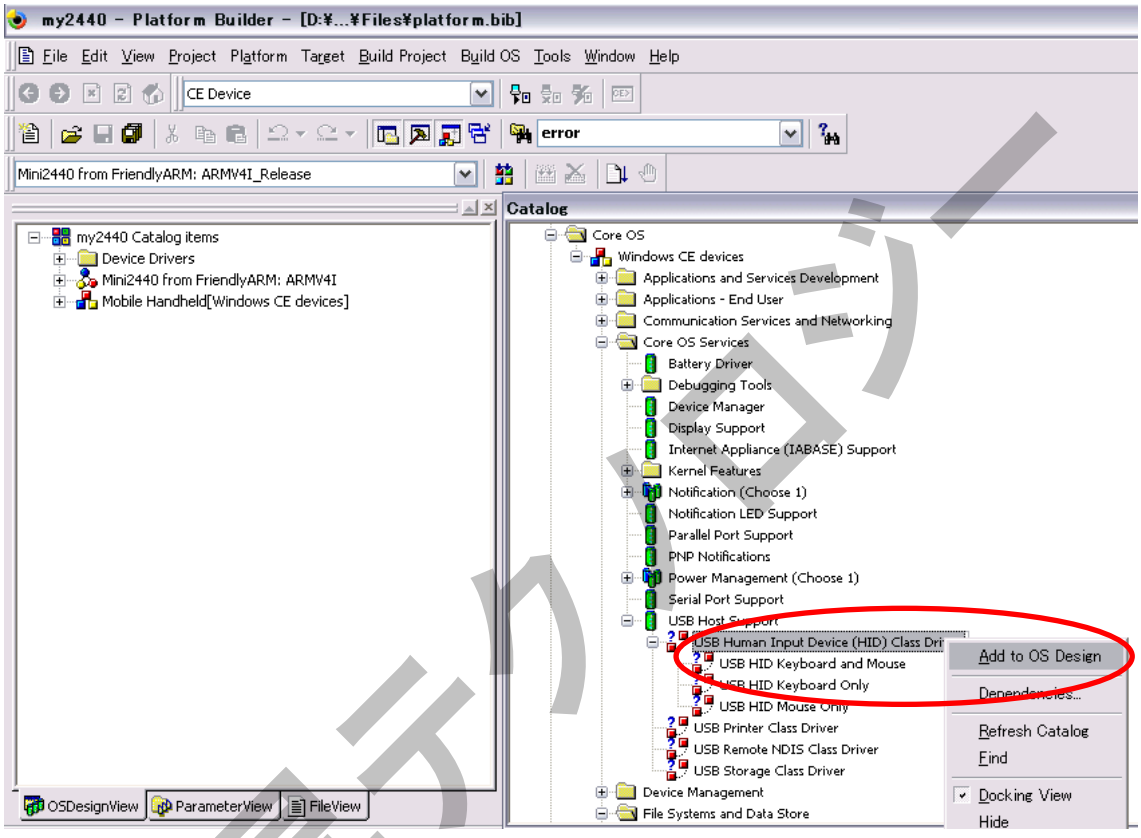
2.3 ビルドオプションを設定する

「Build Options」で「**Enable CE Target Control Support**」と「**Enable KITL**」のチェックを消します。

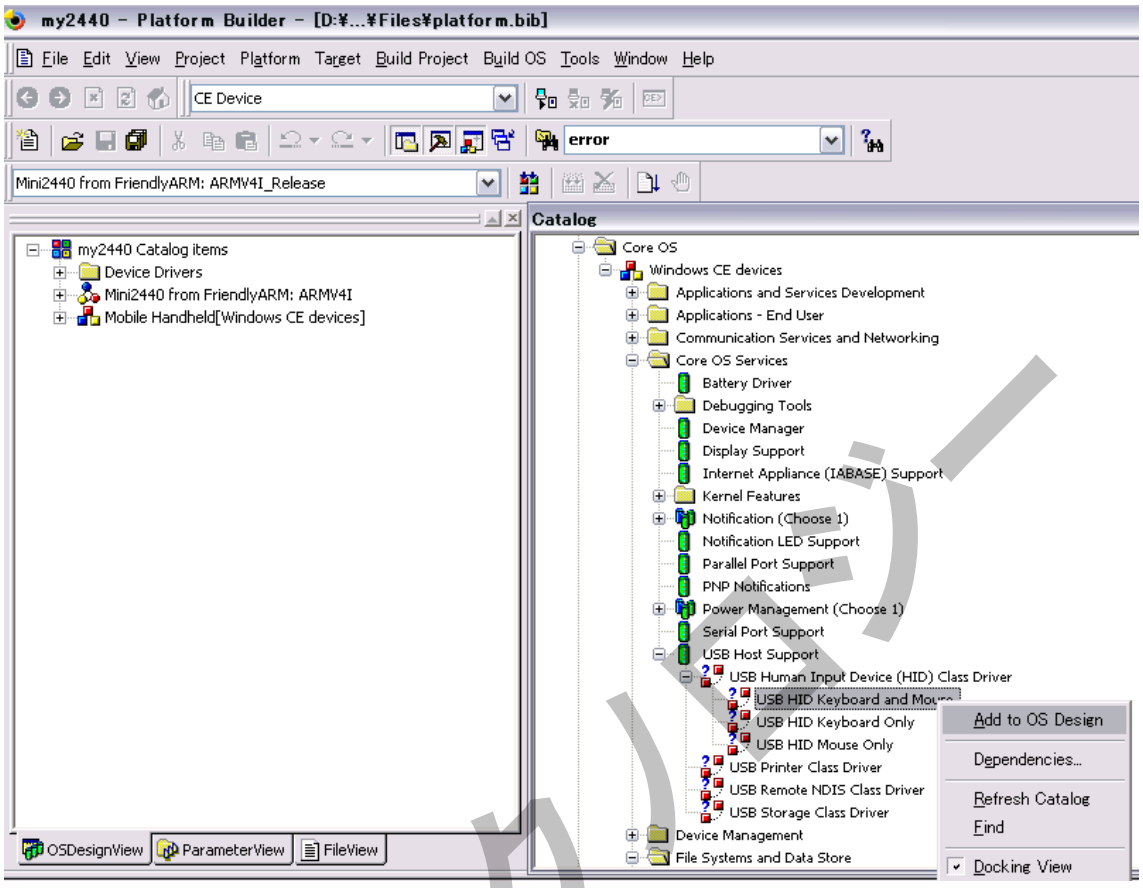


2.4 マウスとキーボードを添加する

「Catalog」→「Core OS」→「Windows CE devices」→「Core OS Services」→「USB host support」→「USB Human Input Device (HID) Class Driver」を選択して、マウスの右ボタンをクリックして、メニューの中で「Add to OS Design」を選択します。

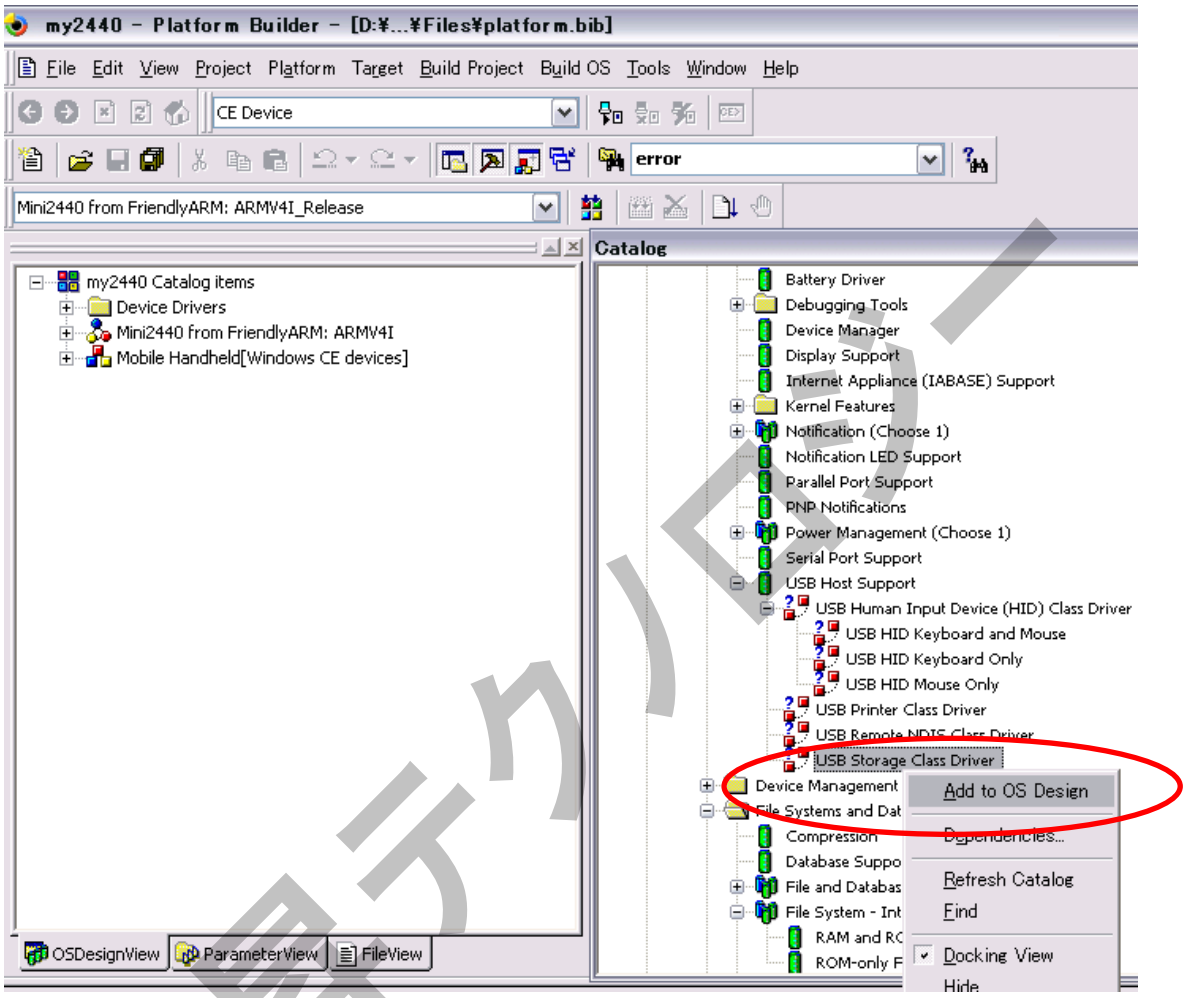


続いてサブ選択肢の「USB HID Keyboard and Mouse」を選択して、マウスの右ボタンをクリックして、メニューの中で「Add to OS Design」を選択します。


my2440 - Platform Builder - [D:¥...¥Files¥platform.bib]
File Edit View Project Platform Target Build Project Build OS Tools Window Help
CE Device
error
Mini2440 from FriendlyARM: ARMV4I_Release
my2440 Catalog items
Device Drivers
Mini2440 from FriendlyARM: ARMV4I
Mobile Handheld[Windows CE devices]
Catalog
Core OS
Windows CE devices
Applications and Services Development
Applications - End User
Communication Services and Networking
Core OS Services
Battery Driver
Debugging Tools
Device Manager
Display Support
Internet Appliance (IABASE) Support
Kernel Features
Notification (Choose 1)
Notification LED Support
Parallel Port Support
PNP Notifications
Power Management (Choose 1)
Serial Port Support
USB Host Support
USB Human Input Device (HID) Class Driver
USB HID Keyboard and Mouse
USB HID Keyboard Only
USB HID Mouse Only
USB Printer Class Driver
USB Remote NDIS Class Driver
USB Storage Class Driver
Device Management
File Systems and Data Store
Add to OS Design
Dependencies...
Refresh Catalog
Find
Docking View
OSDesignView
ParameterView
FileView

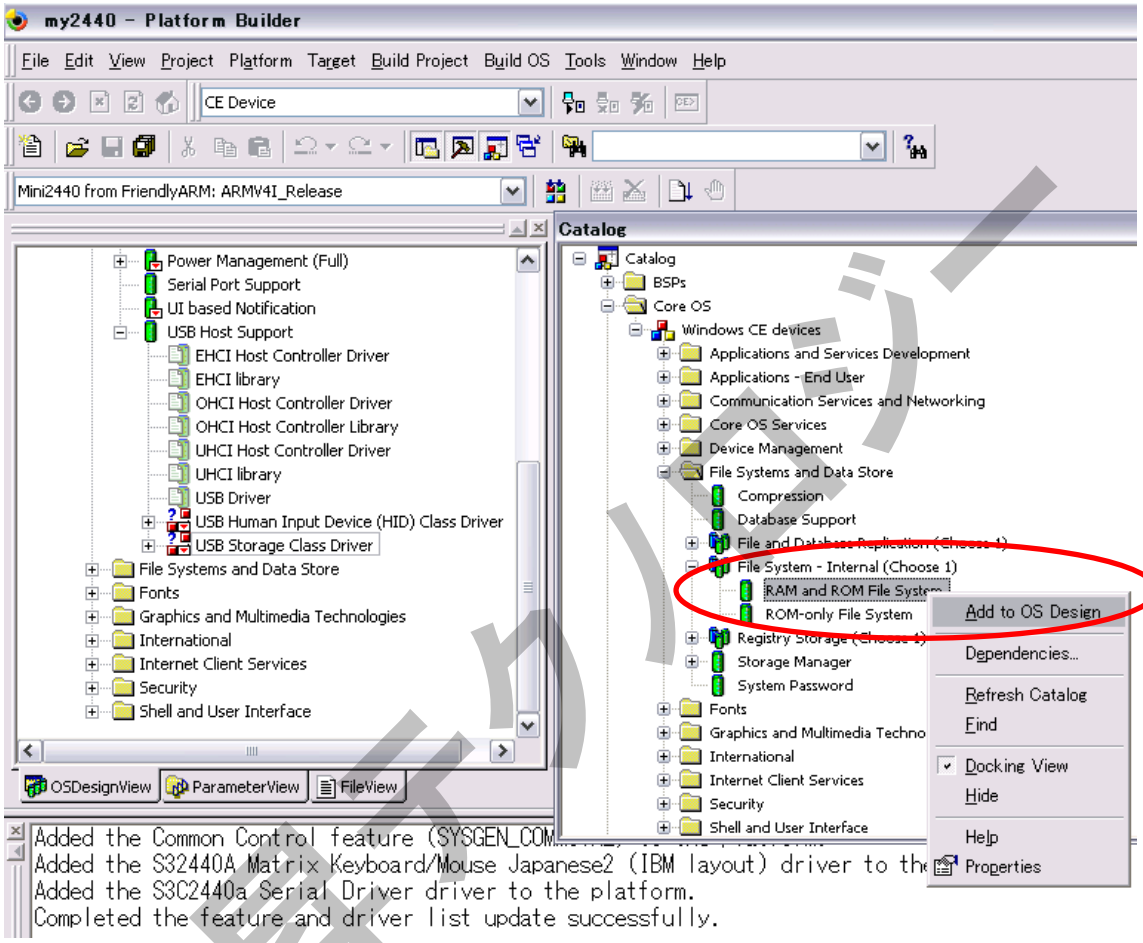
2.5 USB メモリを添加する

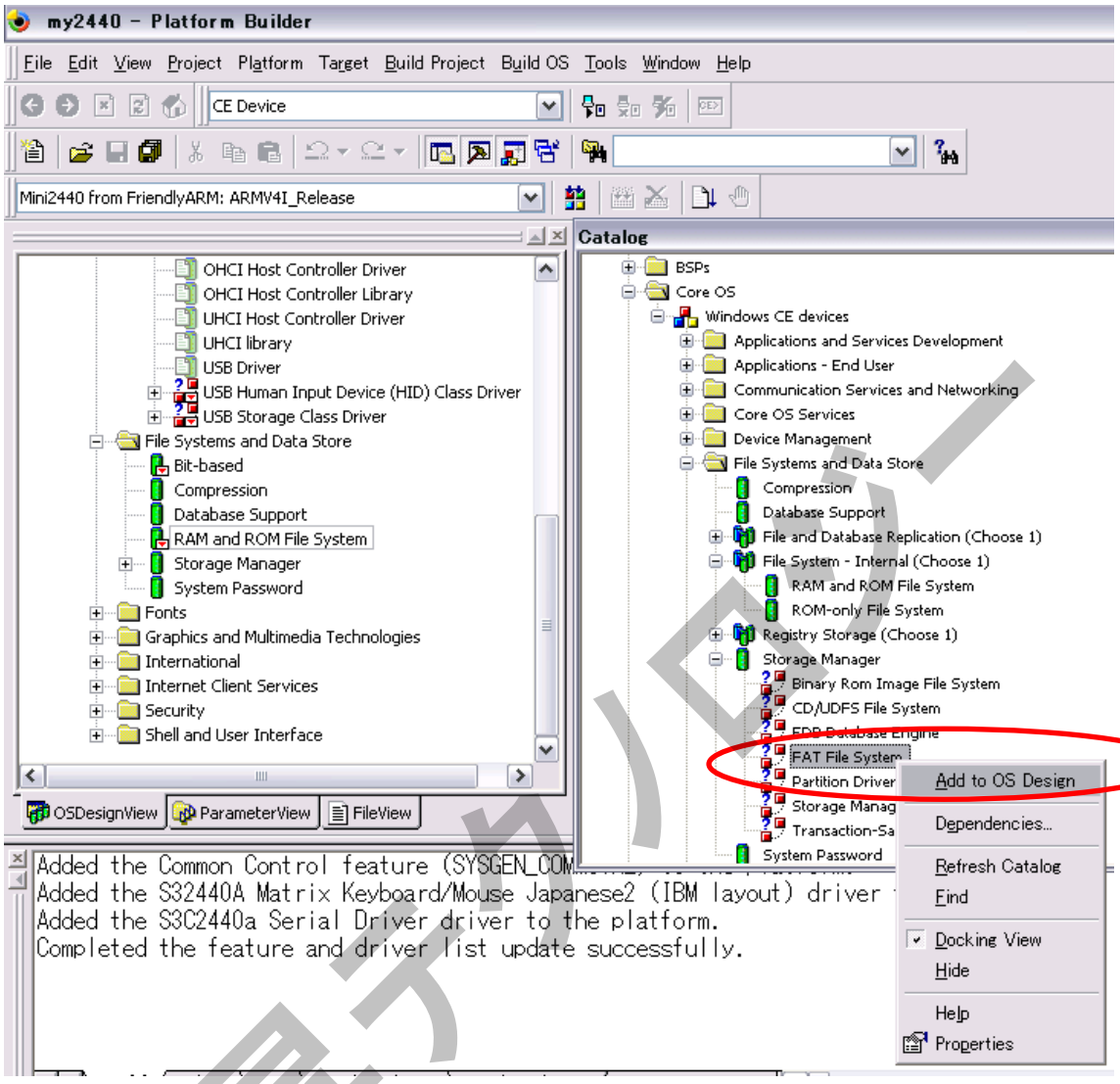
同じ方法で USB メモリを添加します。



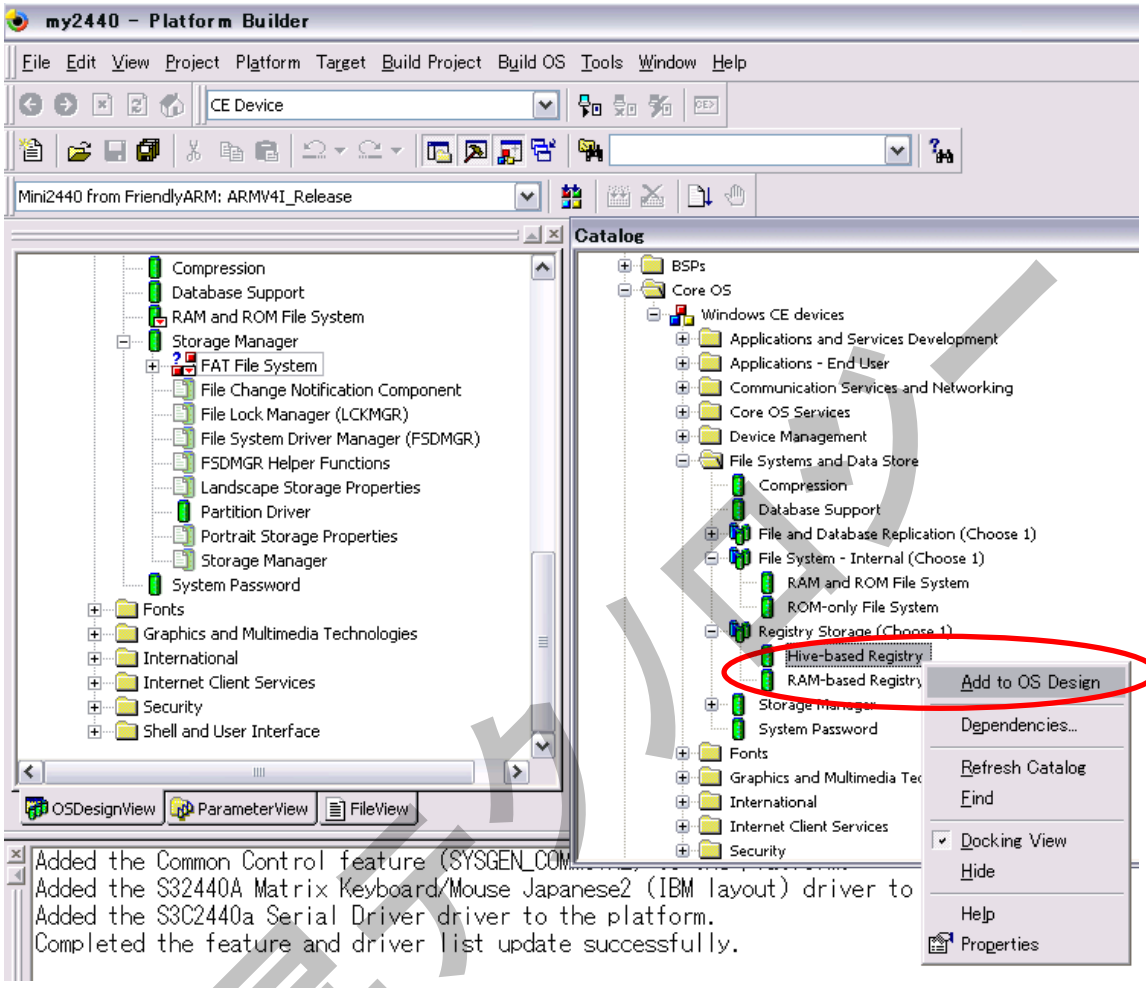
2.6 ファイルシステムを添加する

ファイルシステムを添加します。「RAM and ROM file system」と「FAT system」を追加します。

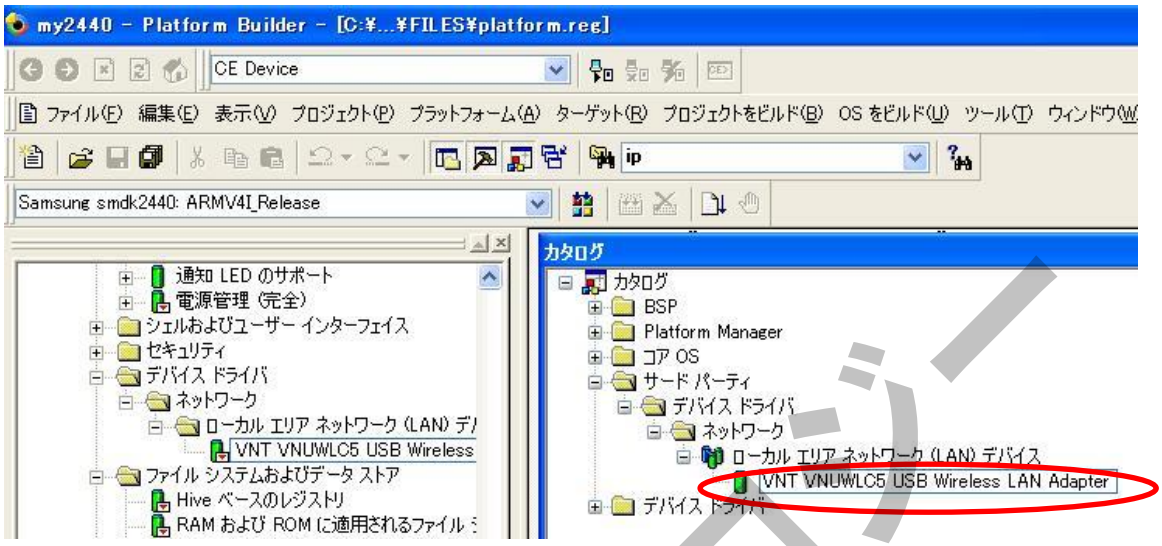



my2440 - Platform Builder
File Edit View Project Platform Target Build Project Build OS Tools Window Help
CE Device
Mini2440 from FriendlyARM: ARMV4I_Release
OHCI Host Controller Driver
OHCI Host Controller Library
UHCI Host Controller Driver
UHCI library
USB Driver
USB Human Input Device (HID) Class Driver
USB Storage Class Driver
File Systems and Data Store
Bit-based
Compression
Database Support
RAM and ROM File System
Storage Manager
System Password
Fonts
Graphics and Multimedia Technologies
International
Internet Client Services
Security
Shell and User Interface
OSDesignView
ParameterView
FileView
Catalog
BSPs
Core OS
Windows CE devices
Applications and Services Development
Applications - End User
Communication Services and Networking
Core OS Services
Device Management
File Systems and Data Store
Compression
Database Support
File and Database Replication (Choose 1)
File System - Internal (Choose 1)
RAM and ROM File System
ROM-only File System
Registry Storage (Choose 1)
Storage Manager
Binary Rom Image File System
CD/UDFS File System
FAT File System
Partition Driver
Storage Manag
Transaction-Sa
System Password
Add to OS Design
Dependencies...
Refresh Catalog
Find
Docking View
Hide
Help
Properties
Added the Common Control feature (SYSGEN_COM
Added the S32440A Matrix Keyboard/Mouse Japanese2 (IBM layout) driver
Added the S3C2440a Serial Driver driver to the platform.
Completed the feature and driver list update successfully.

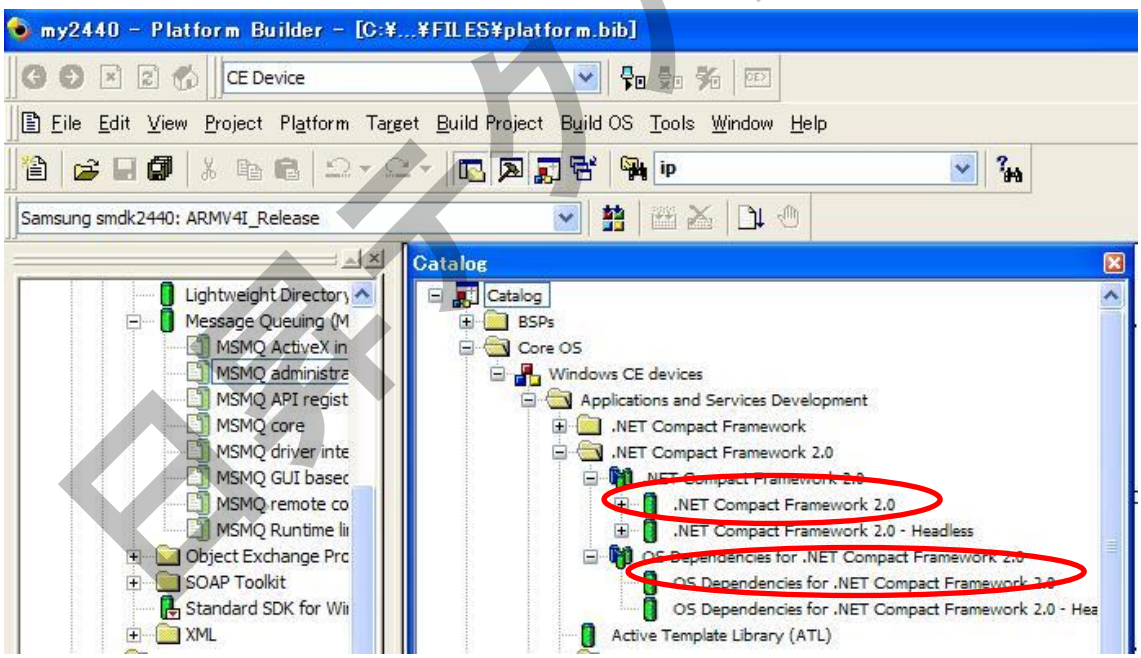
2.7 レジストリ記憶域を追加する



2.8 USB 無線 LAN ドライバを追加する

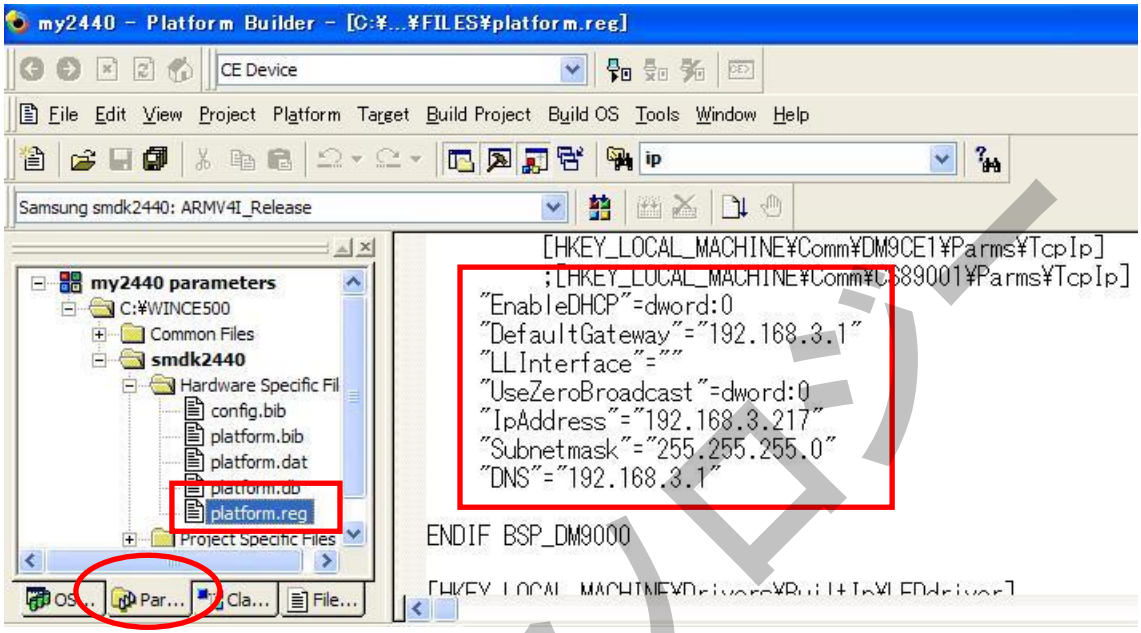


2.9 .NET Compact Framework 2.0 を追加する



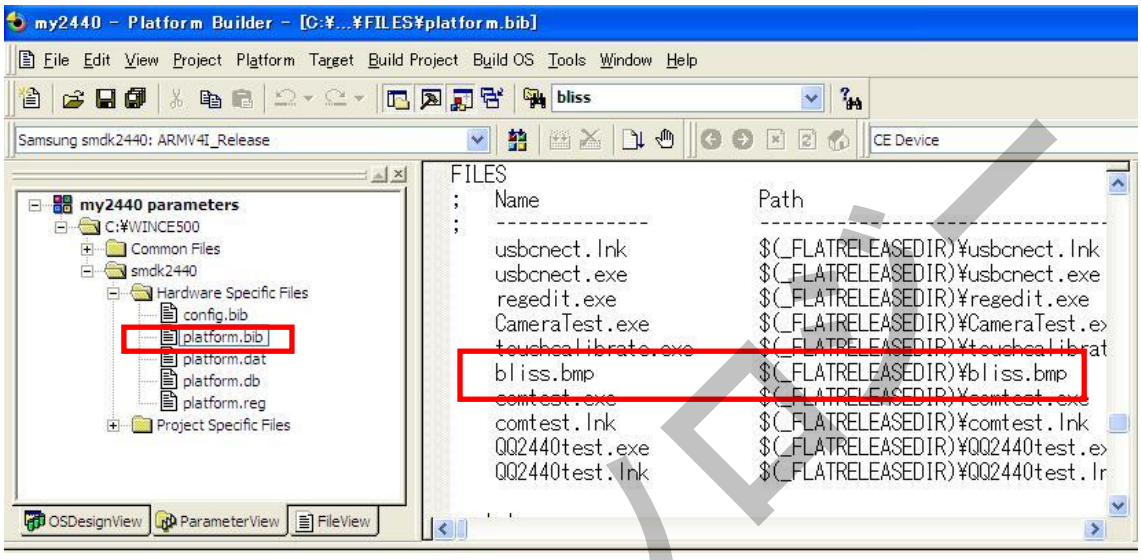
2.10 デフォルト IP アドレスを設定

デフォルト IP アドレス、ゲットウェイと DNS などを編集します。「platform.reg」というファイルを開きます。自分のネットワーク環境によって、直します。

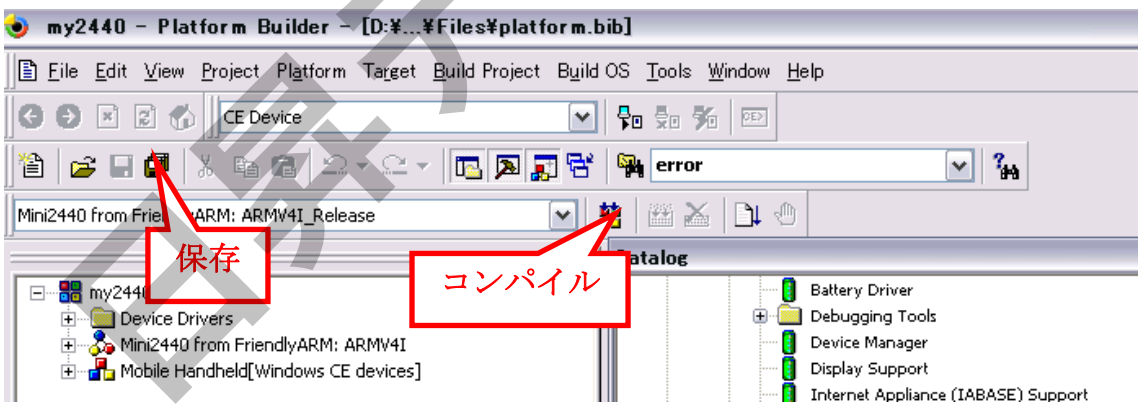


2.10 WinCE の背景画面を変更

WinCE の背景画面も変更できます。「platform.bib」というファイルを開きます。自分が好きな bmp ピクチャを「C:¥WINCE500¥PLATFORM¥mini2440¥FILES」にコピーします。ピクチャの名前は「bliss.bmp」に変更します。



以上は配置の例だけです。自分が好きな機能も追加できます。配置の方法はほかの資料を参考してください。配置が完了したら、保存とコンパイルします。

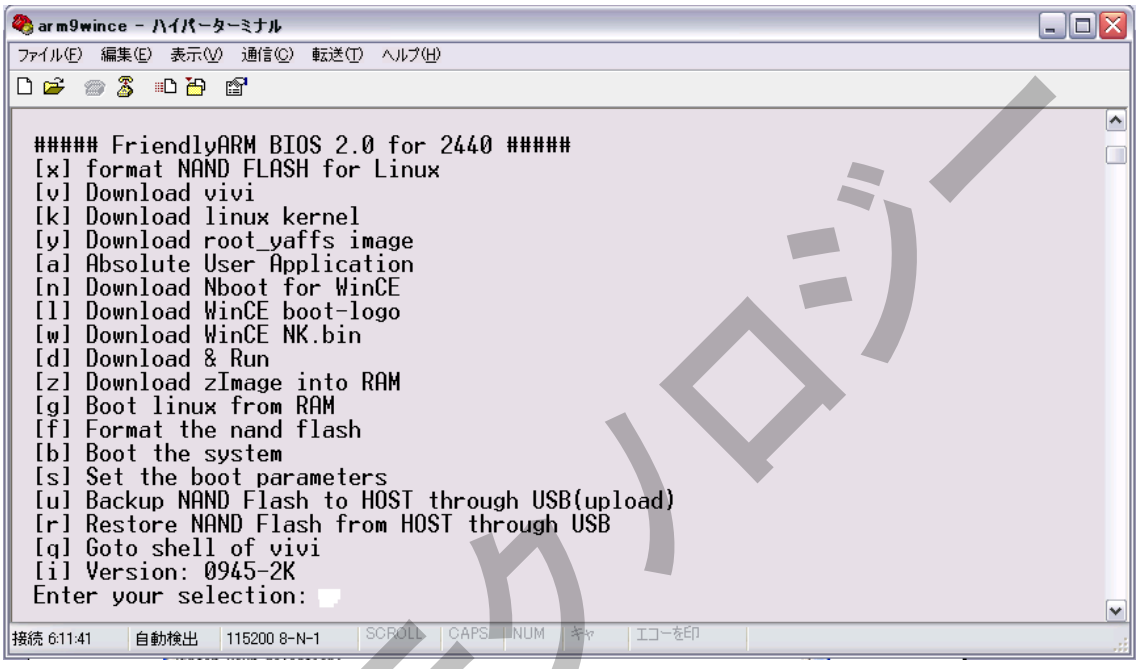


コンパイルは時間がかかります、我慢してください。成功すれば、「C:¥WINCE500¥ PBWorkspaces¥my2440¥RelDir¥mini2440_ARMV4I_Release」で NK.bin というファイルを生成します。このファイルは WinCE5.0 のイメージファイルです。

第三章 WinCE を書き込む

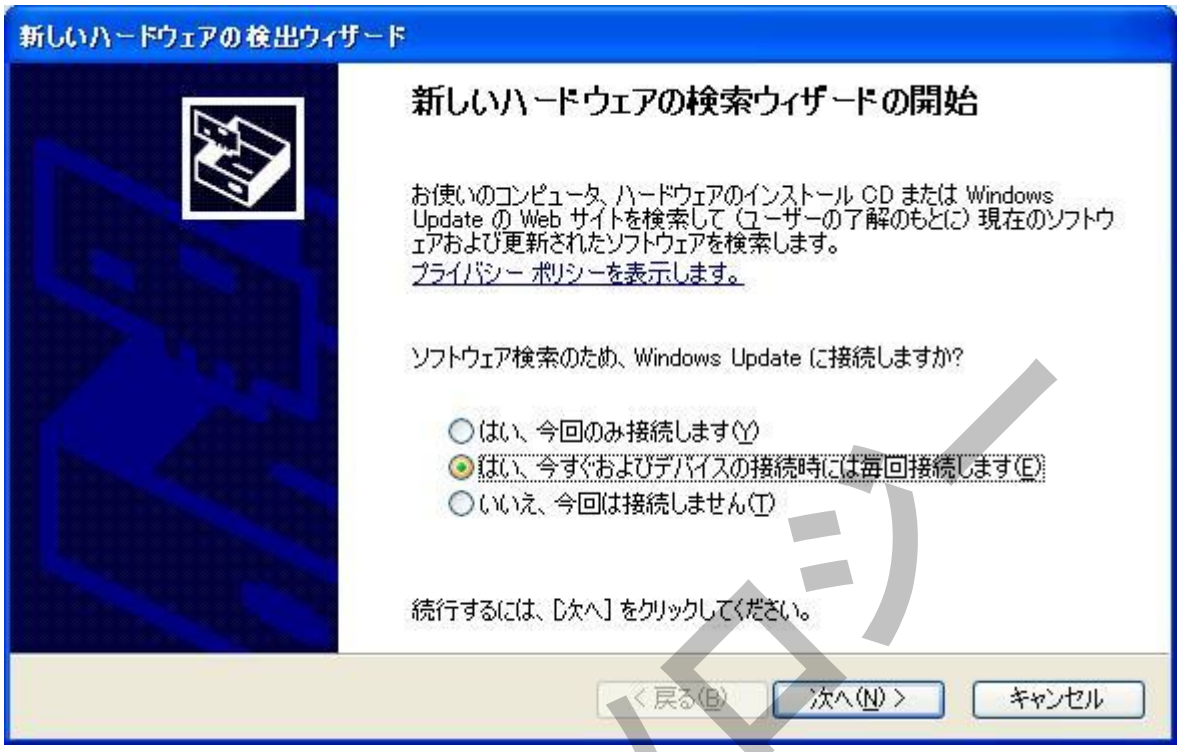
3.1 NOR Flash から起動

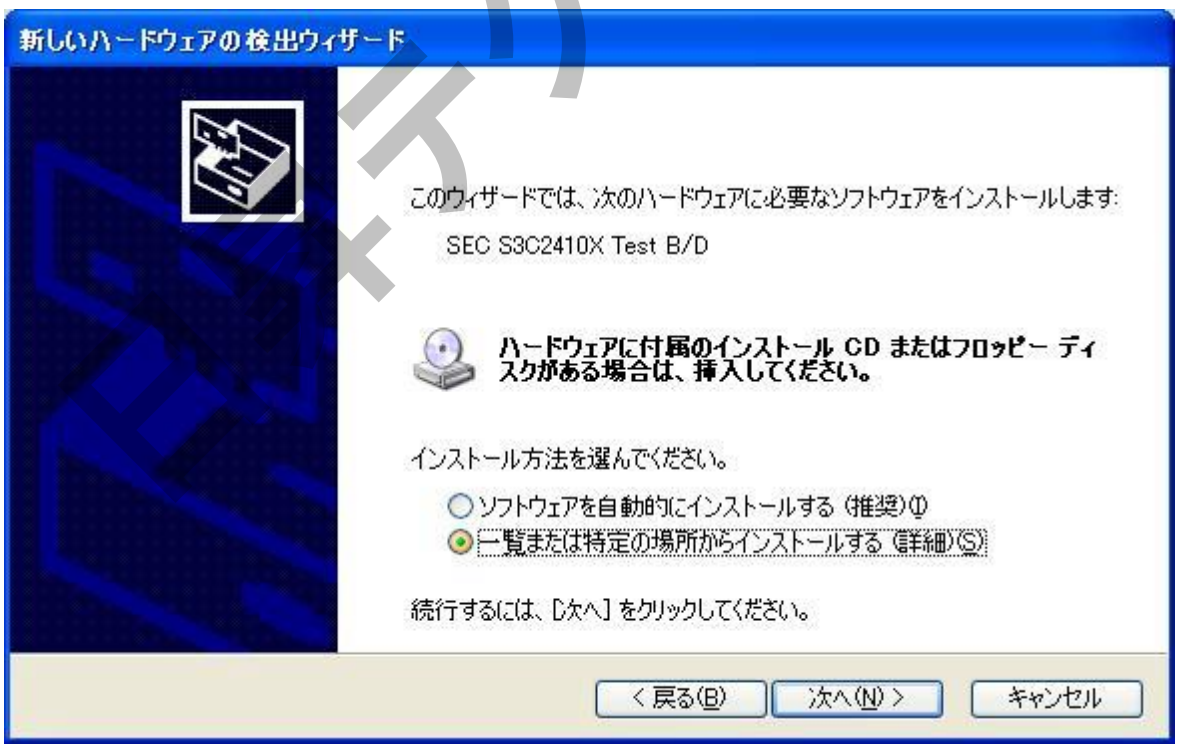
ARM9 ボードの S2 スイッチ又は J1 を Nor Flash に設定して、電源を入れて、ARM9 ボードは Nor Flash から起動します。

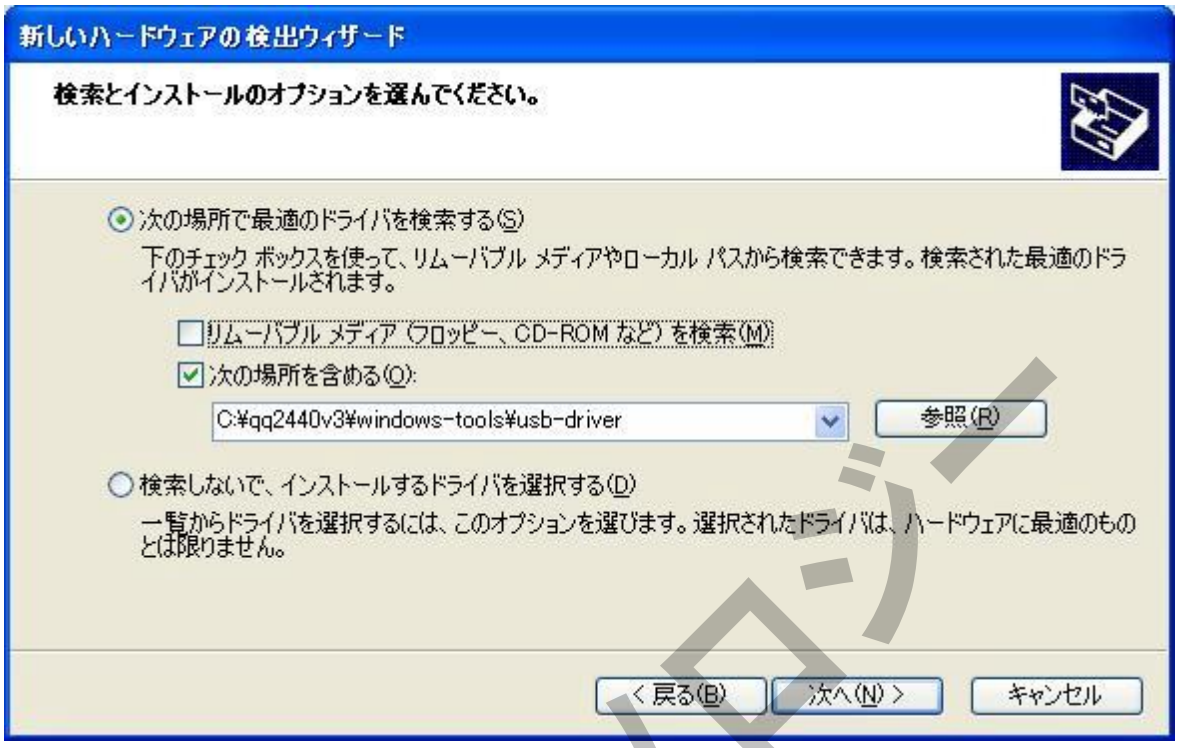


3.2 USB ドライバのインストール

開発されたOSとプログラムをUSB通じてARM9ボードにダウンロードします。その為、USBケーブルでARM9ボードのUSBスレーブポートとパ ソコンのUSBポートを繋ぐことが必要です。繋ぐと、パソコンは新しいデバイスを発見して、USBドライバをインストールします。


新しいハードウェアの検出ウィザード
新しいハードウェアの検索ウィザードの開始
お使いのコンピュータ、ハードウェアのインストール CD または Windows Update の Web サイトを検索して (ユーザーの了解のもとに) 現在のソフトウェアおよび更新されたソフトウェアを検索します。
プライバシー ポリシーを表示します。
ソフトウェア検索のため、Windows Update に接続しますか?
はい、今回のみ接続します(Y)
はい、今すぐおよびデバイスの接続時には毎回接続します(E)
いいえ、今回は接続しません(T)
続行するには、[次へ] をクリックしてください。
< 戻る(B)
次へ(N) >
キャンセル


新しいハードウェアの検出ウィザード
このウィザードでは、次のハードウェアに必要なソフトウェアをインストールします:
SEC S3C2410X Test B/D
ハードウェアに付属のインストール CD またはフロッピー ディスクがある場合は、挿入してください。
インストール方法を選んでください。
ソフトウェアを自動的にインストールする (推奨)(I)
一覧または特定の場所からインストールする (詳細)(S)
続行するには、[次へ] をクリックしてください。
< 戻る(B)
次へ(N) >
キャンセル


新しいハードウェアの検出ウィザード
検索とインストールのオプションを選んでください。
次の場所で最適のドライバを検索する(S)
下のチェック ボックスを使って、リムーバブル メディアやローカル パスから検索できます。検索された最適のドライバがインストールされます。
リムーバブル メディア (フロッピー、CD-ROM など) を検索(M)
次の場所を含める(O):
C:¥qq2440v3¥windows-tools¥usb-driver
参照(R)
検索しないで、インストールするドライバを選択する(D)
一覧からドライバを選択するには、このオプションを選びます。選択されたドライバは、ハードウェアに最適のものとは限りません。
< 戻る(B)
次へ(N) >
キャンセル

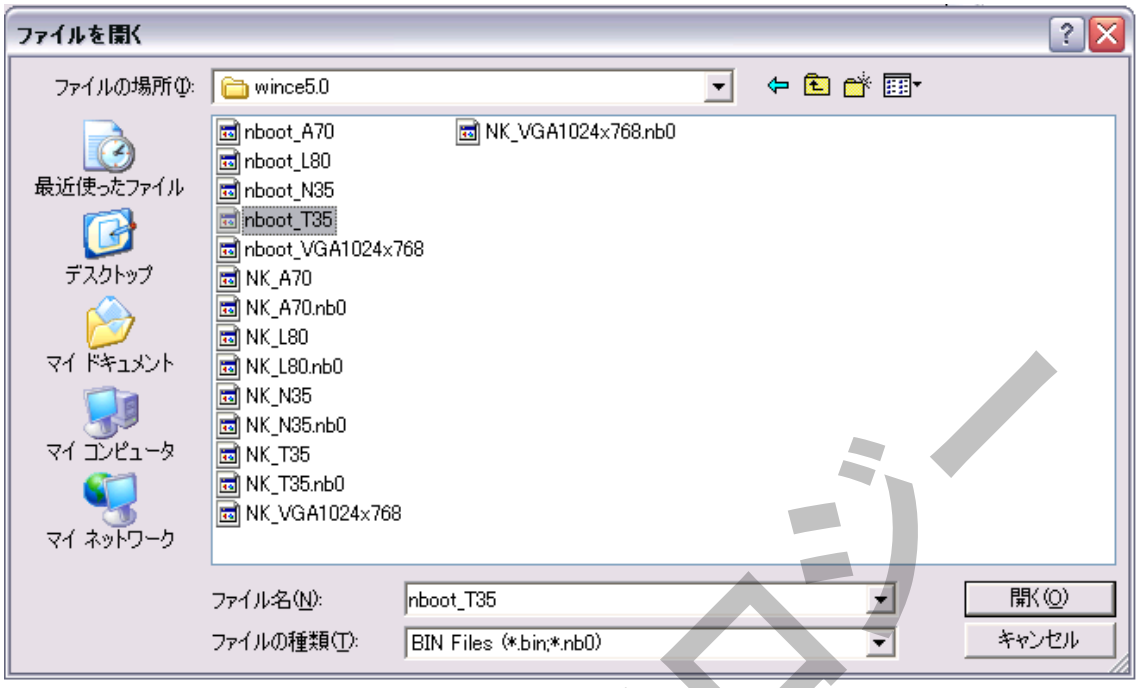

新しいハードウェアの検出ウィザード
ソフトウェアをインストールしています。お待ちください...
SEC SOC Test Board
< 戻る(B)
次へ(N) >
キャンセル

USBドライバをインストール完了あと、パソコンのダウンロード・ツールDNW.exeを実行して、mini2240とパソコンを繋ぐことが確認できます。

※ USBドライバはバッグがあります。ARM9ボードが再起動、又はリセットの時、ホスト側は死んだかもしれません。その原因で、ARM9ボードが起動完了した後、USBケーブルでホストを繋ぎます。

3.3 NBOOT の書き込み

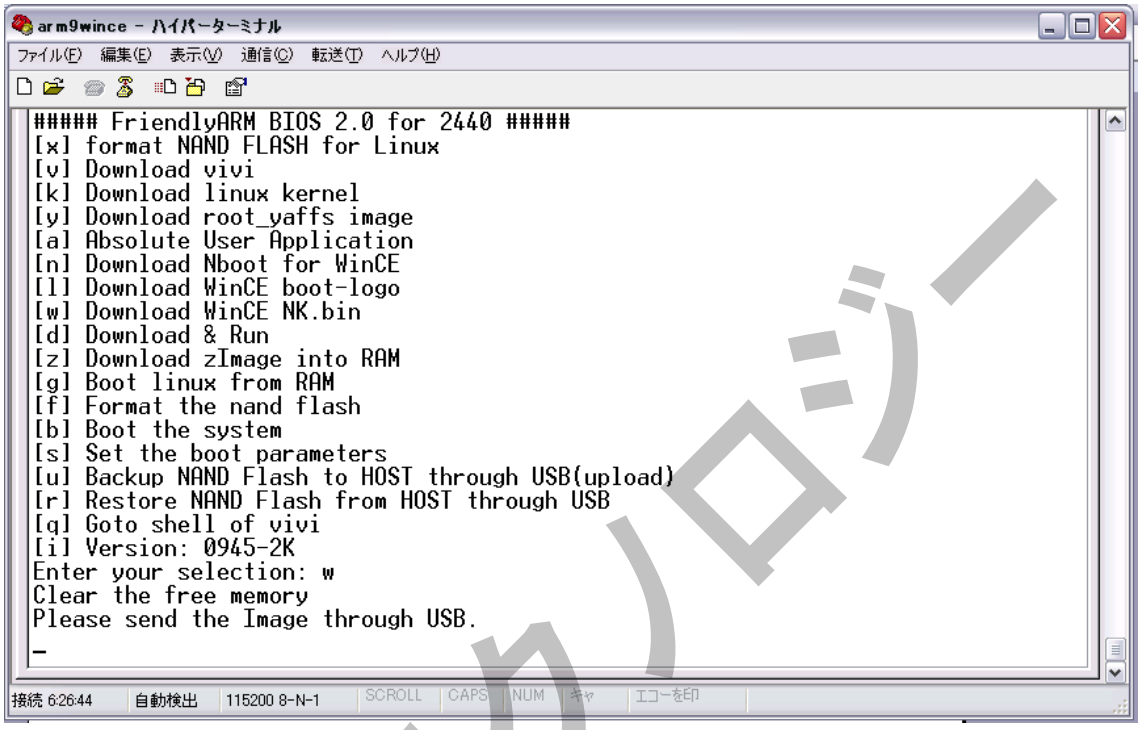
ブートロード supervivi 又は NBOOT を書き込みます。NBOOT は簡単なブートロードですが、WinCE をブートするのは速いです。



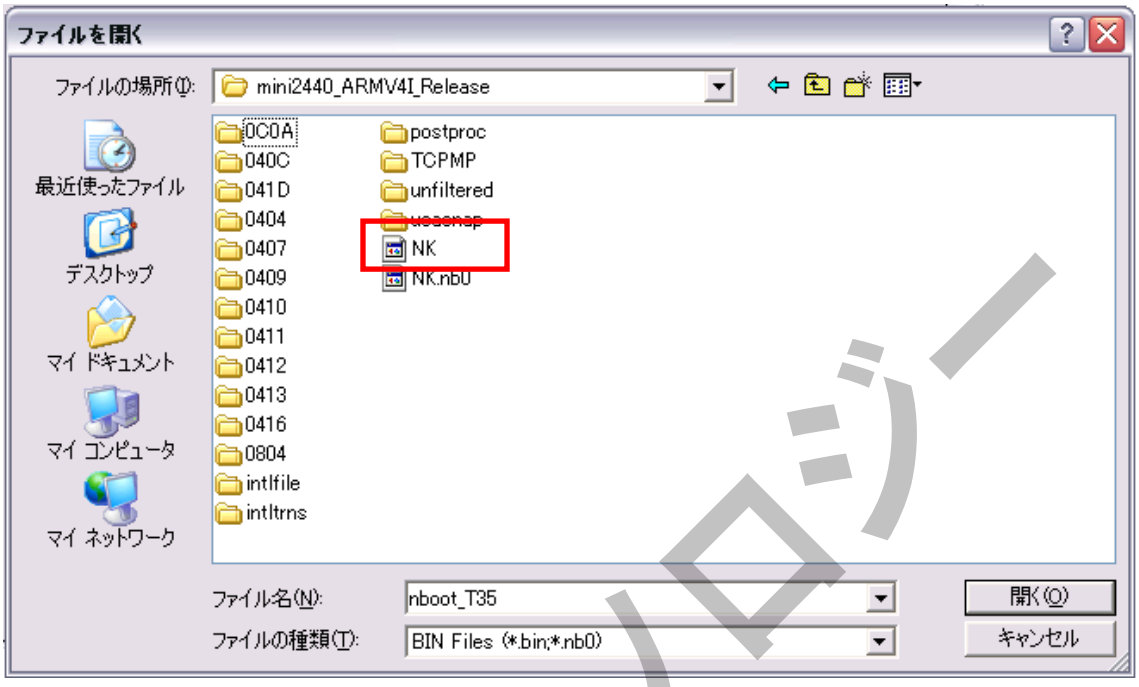
ブートロードを書き込み完了すると、自動的にメニューに戻ります。

3.4 WinCE の書き込み

メニューの中で、機能号[w]を選択して、



DNW を待っています。DNW のメニュー「USB Port」→「Transmit」を選択して、生成された WinCE(NK.bin)を選択して、書き込みます。



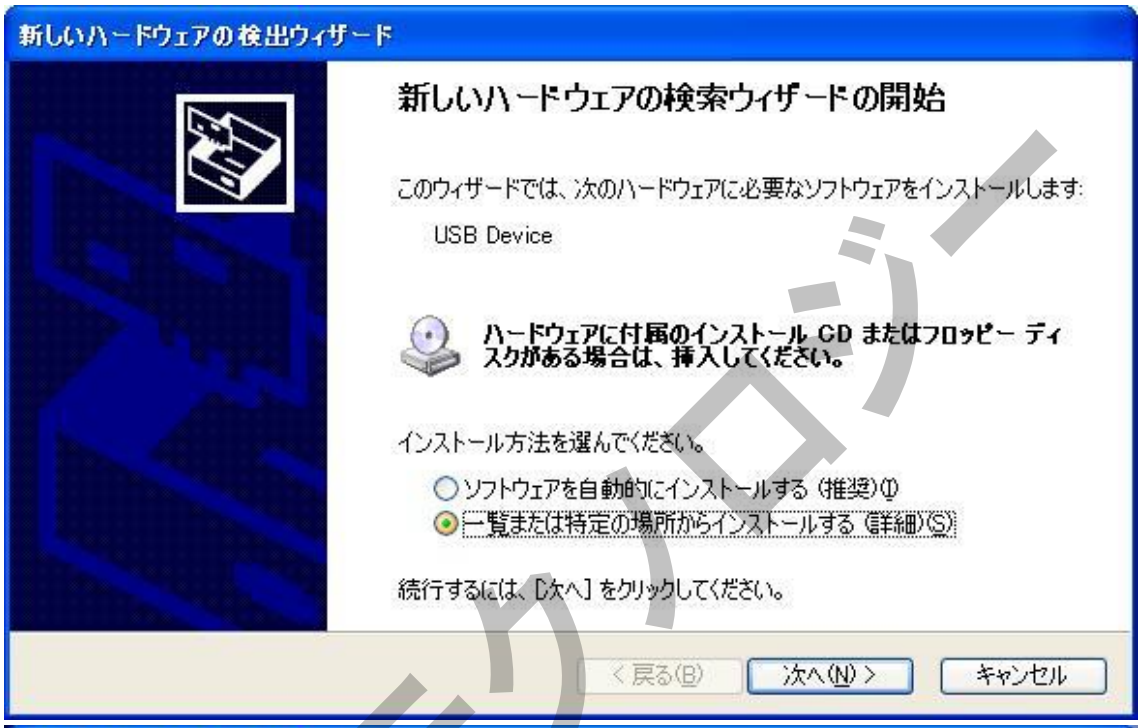
書き込み完了すると、自動的に WinCE を起動します。初起動のため、時間がかかります、少々待ちください。WinCE5.0 の起動画面：

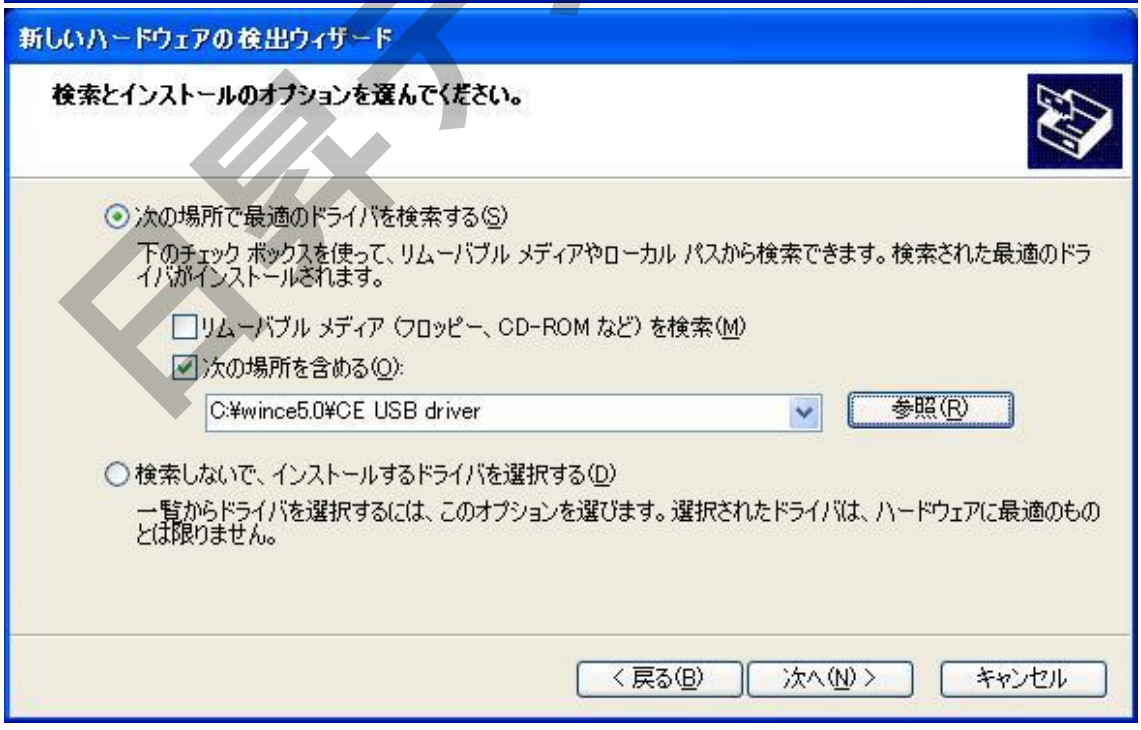
※ Windows CE は Windows XP との使い方がほぼ同じです。使ってみましょう。

第四章 パソコンと同期通信する

4.1 USB ドライバをインストールする

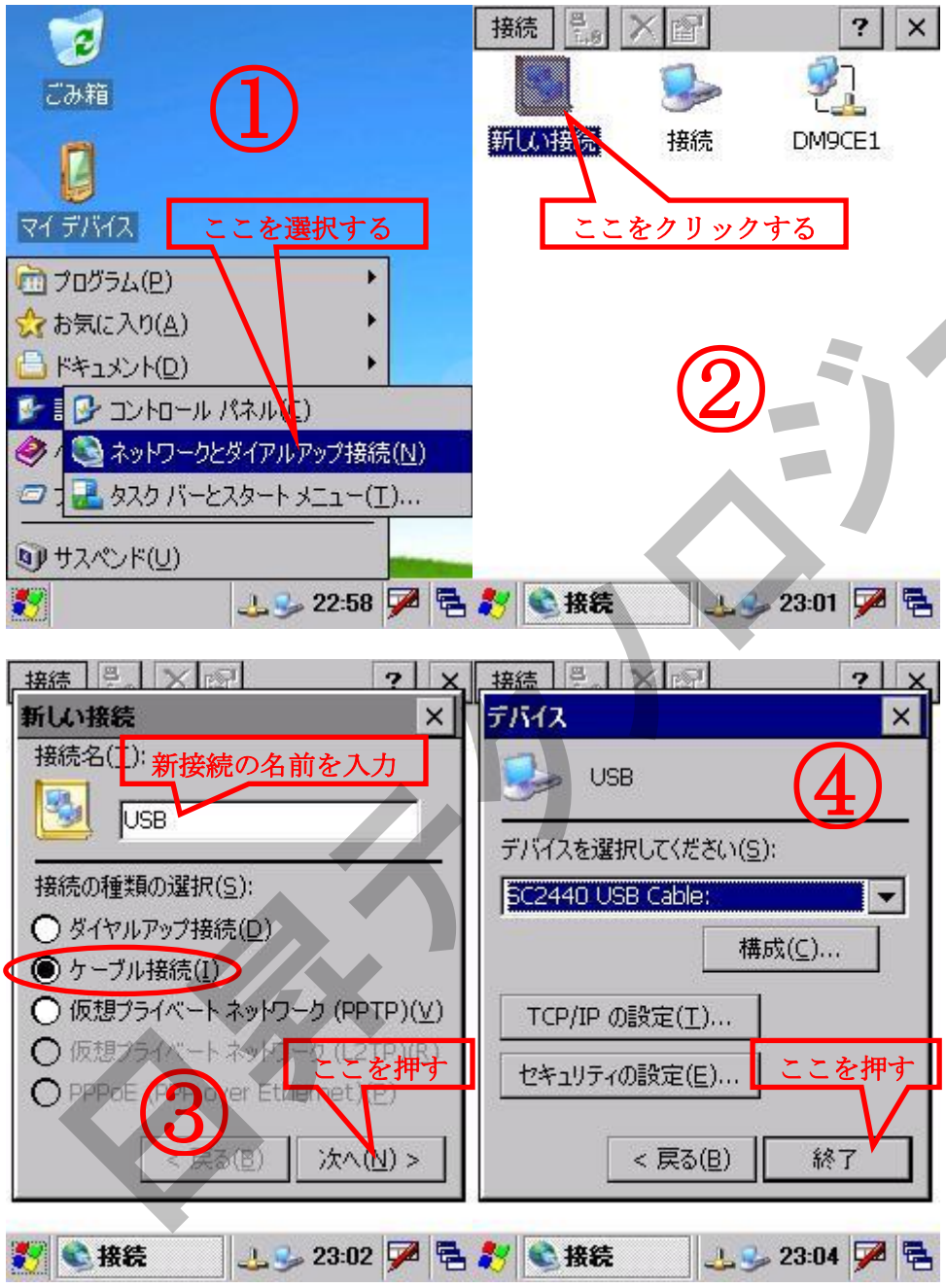
ARM9 ボードは WinCE が起動して。USB ケーブルでパソコンを繋ぐと、

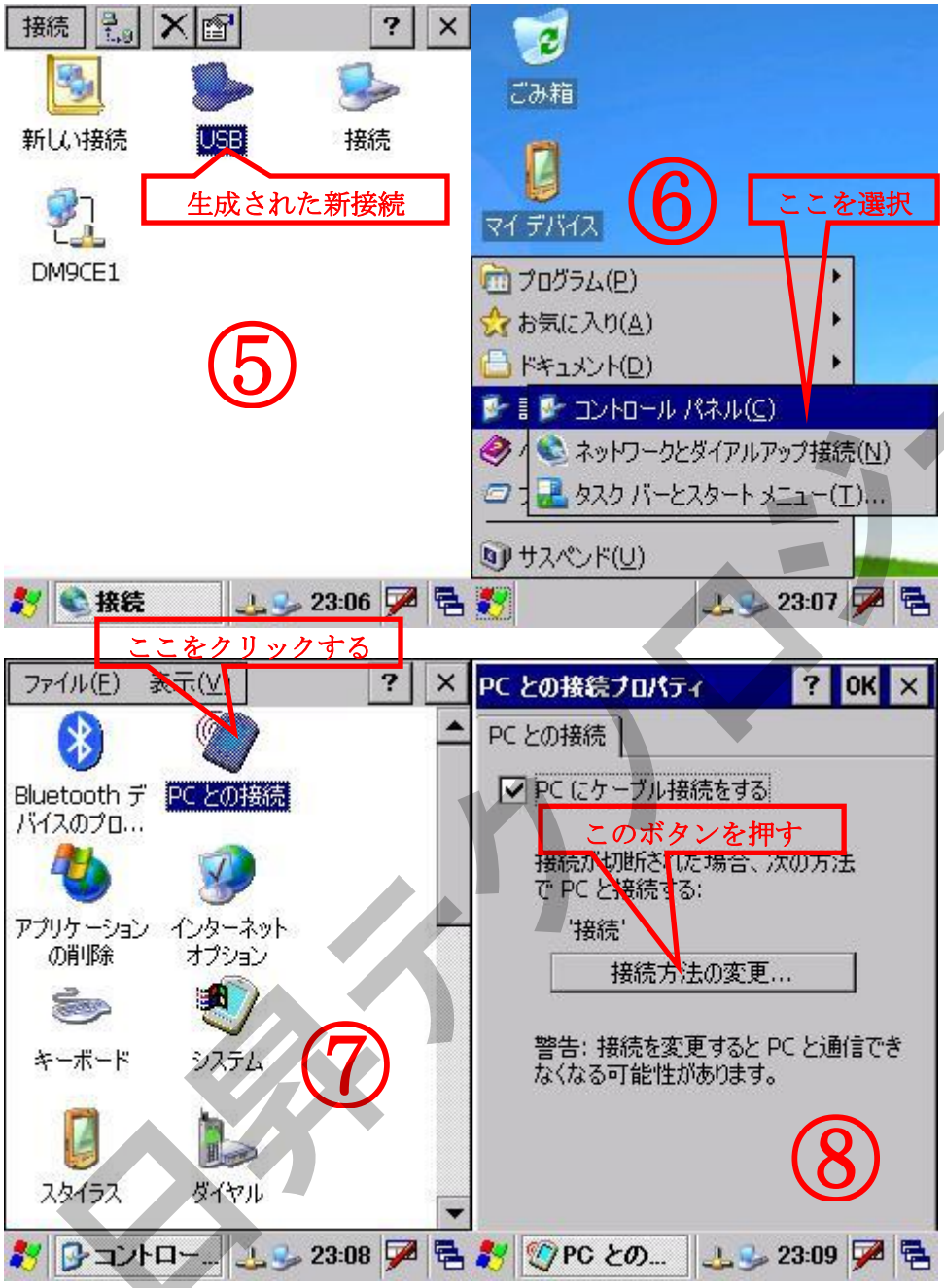


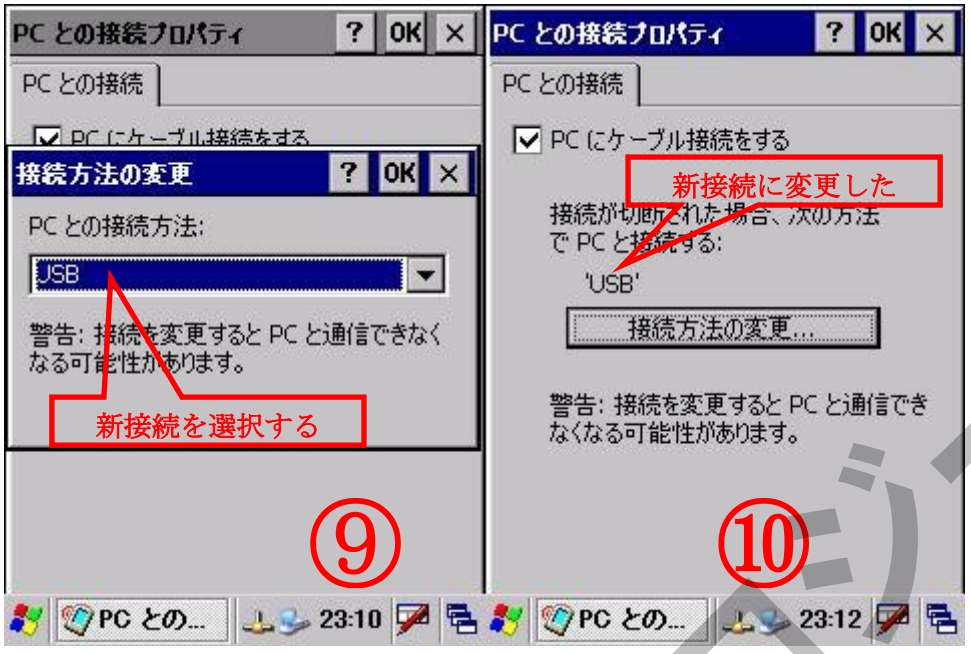


WinCE 用 USB ドライバをインストールします。

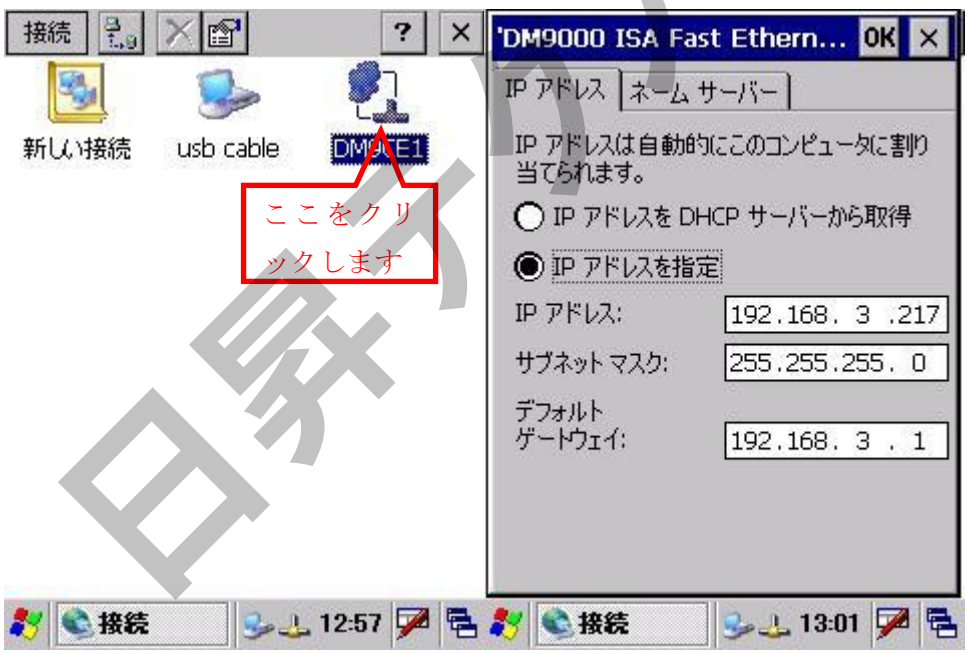
4.2 WinCE 側の設定



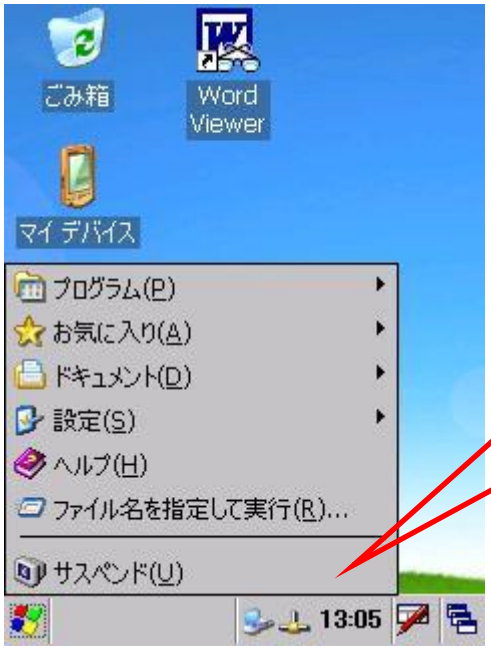

接続
新しい接続
USB
接続
DM9CE1
生成された新接続
⑤
ごみ箱
マイ デバイス
⑥
ここを選択
プログラム(P)
お気に入り(A)
ドキュメント(D)
コントロール パネル(C)
ネットワークとダイアルアップ接続(N)
タスク バーとスタート メニュー(T)...
サスペンド(U)
接続
23:06
23:07
ここをクリックする
ファイル(F) 表示(V)
Bluetooth デバイスのプロ...
PC との接続
アプリケーションの削除
インターネットオプション
キーボード
システム
スタイラス
ダイヤル
⑦
PC との接続プロパティ
OK
PC との接続
PC にケーブル接続をする
このボタンを押す
接続が切断された場合、次の方法で PC と接続する:
'接続'
接続方法の変更...
警告: 接続を変更すると PC と通信できなくなる可能性があります。
⑧
コントロー...
23:08
PC との...
23:09



WinCE の IP アドレスを確認します。



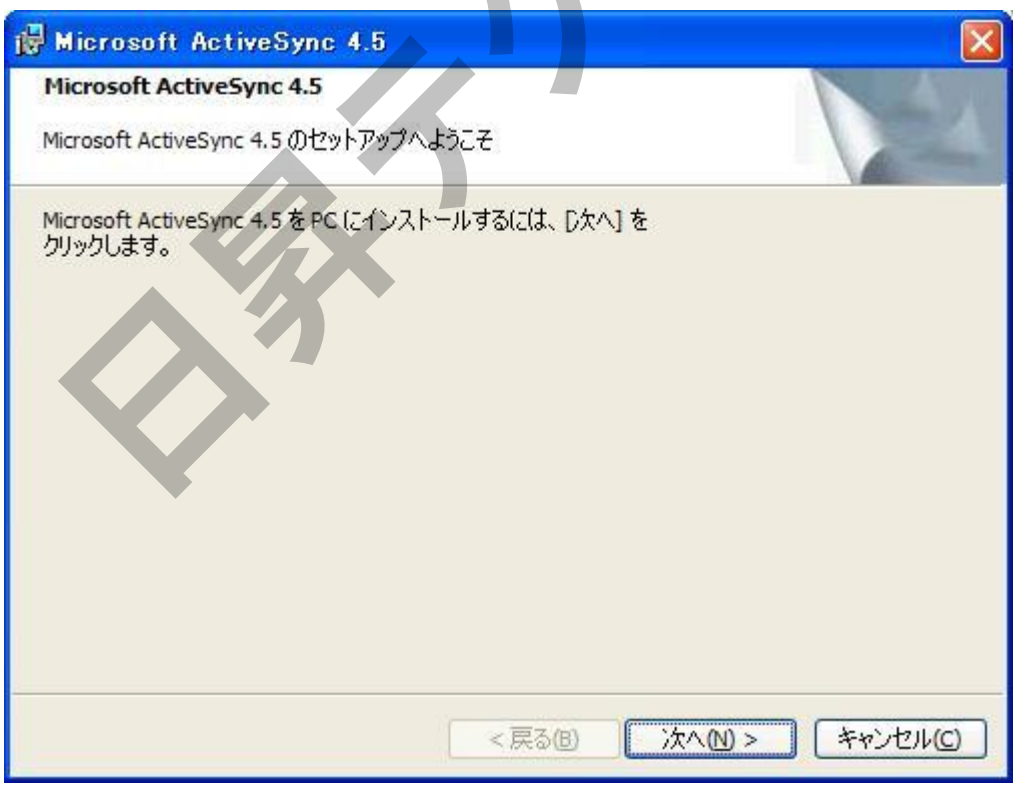
通信の時、ホストの IP アドレスは ARM9 と同じサブネットワークが必要です。この例、ホストの IP アドレスは 192.168.3.xxx のような形です。



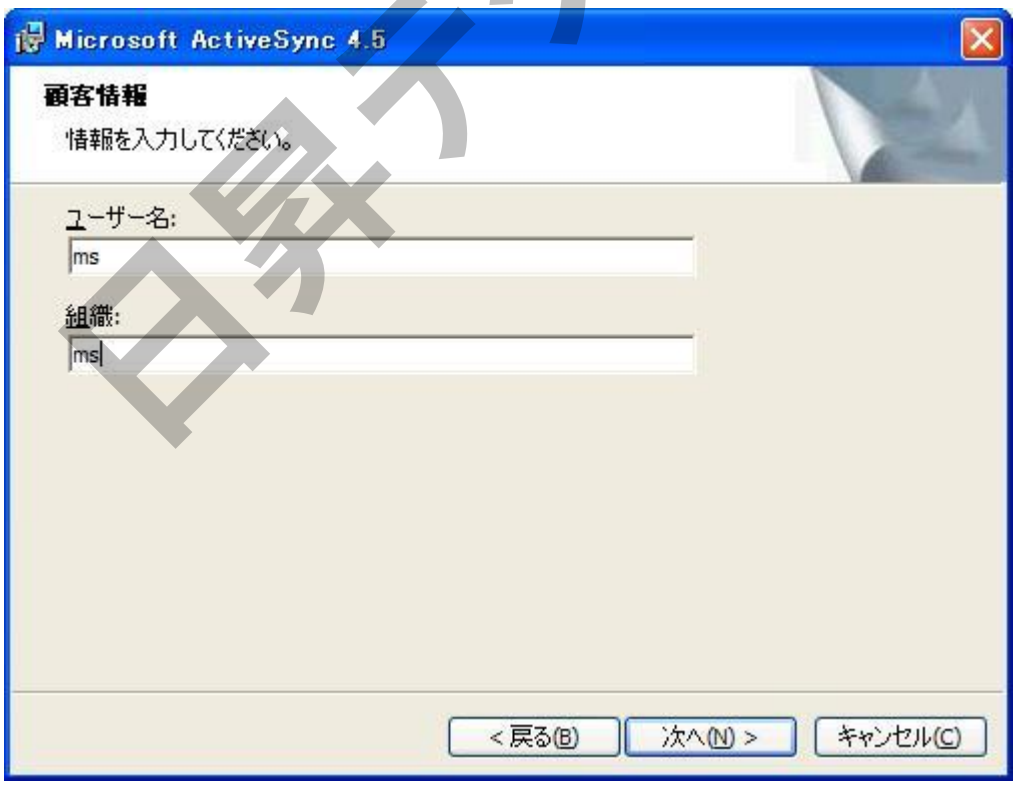
変更完了すると、サスペンドをクリックして、変更されたパラメターを保存します。次回起動の時も有効です。

4.3 ActiveSync をインストールする

ActiveSync_setup4.5.msi を実行します。「次へ」ボタンを押します。

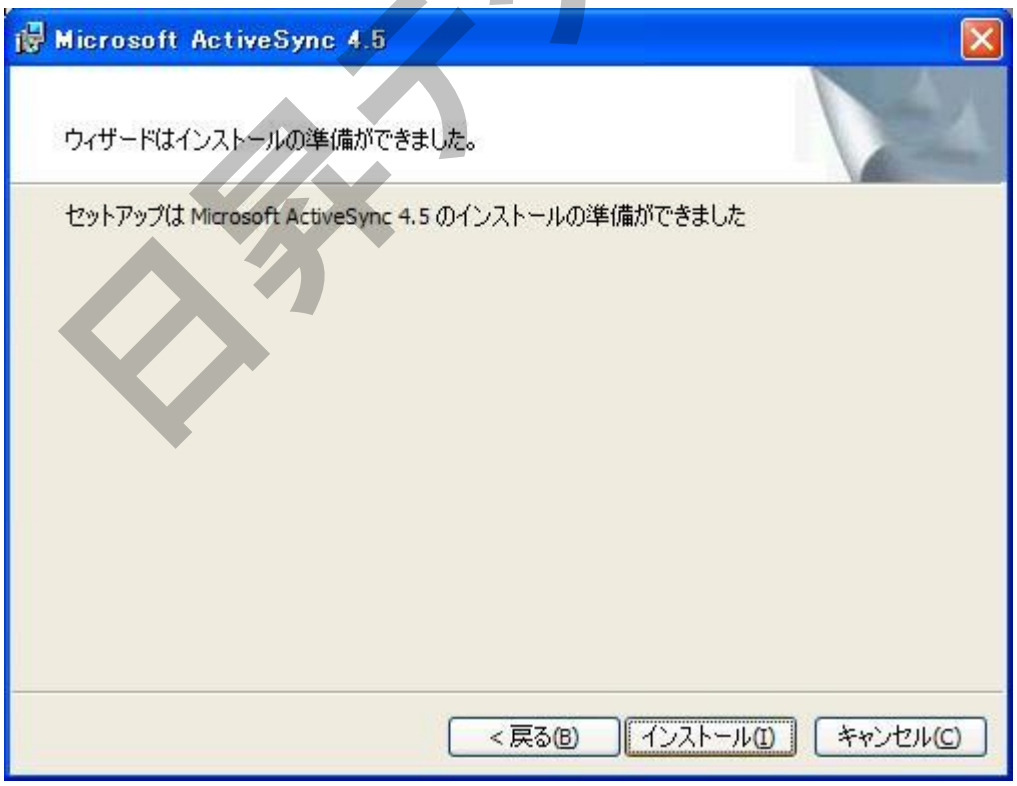


同意できる場合は、「**使用許諾契約書の条項に同意します**」を選択して、「次へ」ボタンを押します。

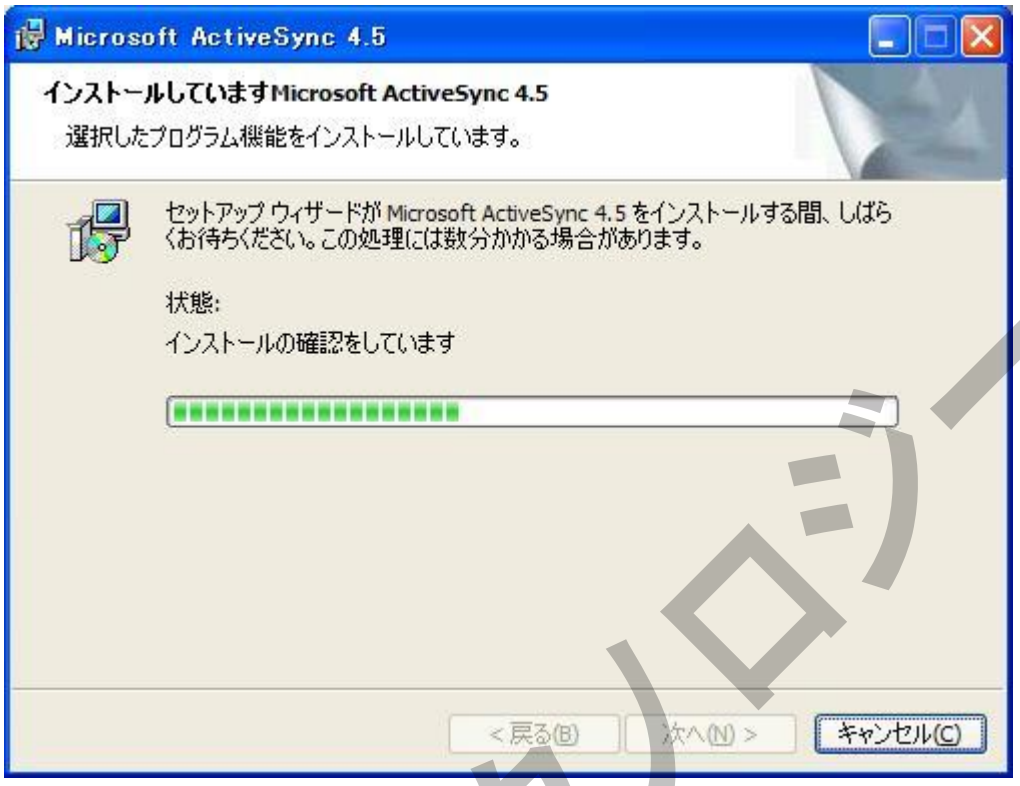


ユーザと組織の名前を入力して、「次へ」ボタンを押します。

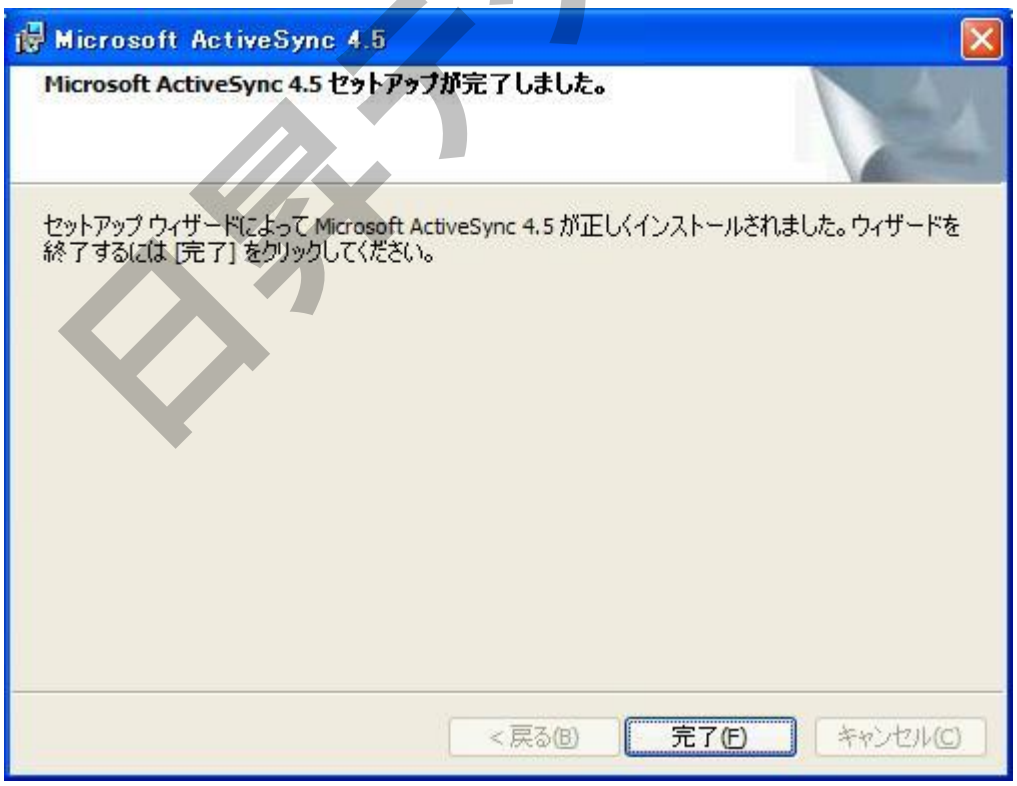
インストール先を変更せず、そのまま進んでください。



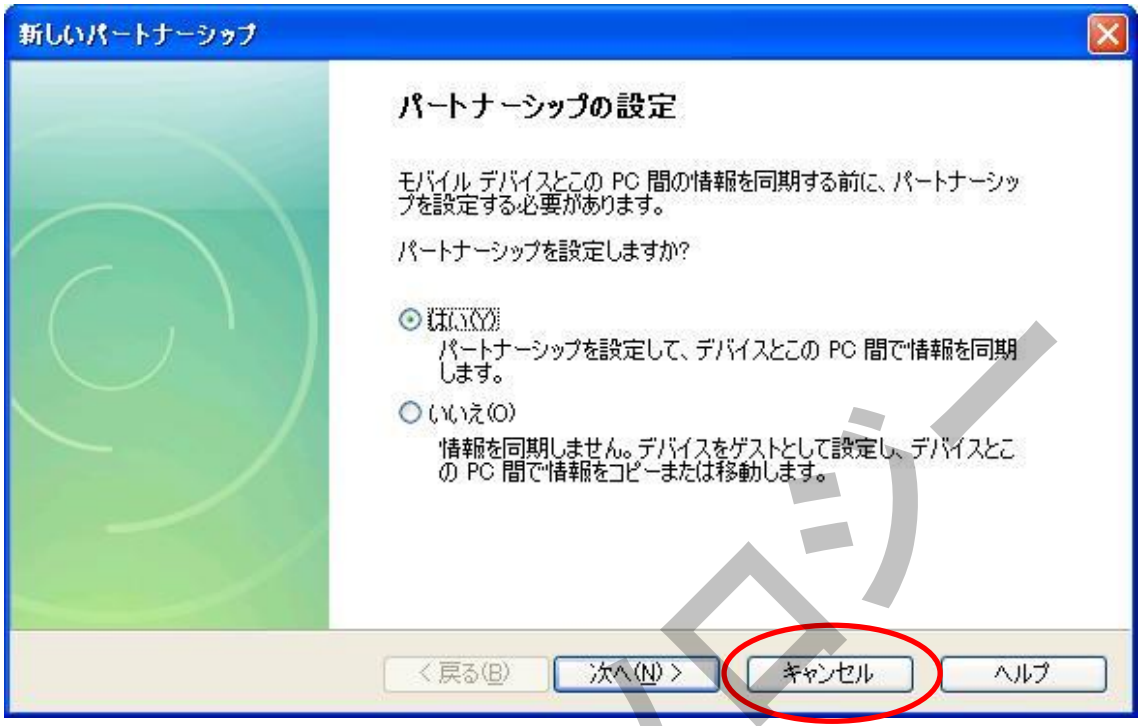
インストールの準備ができました、「インストール」ボタンを押します。



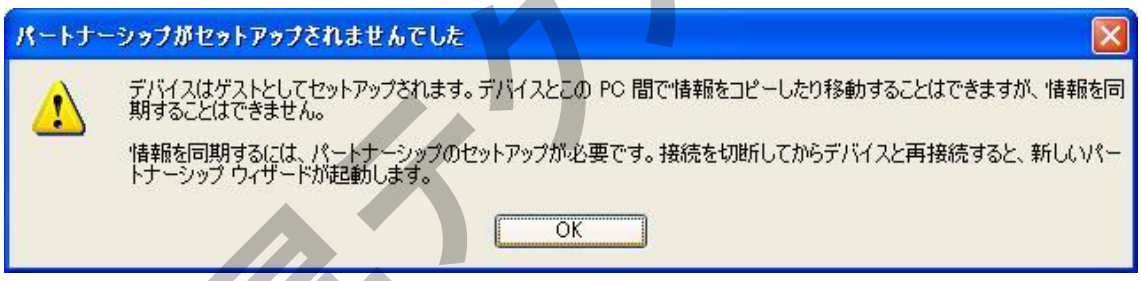
インストール中です、少々待ちください。



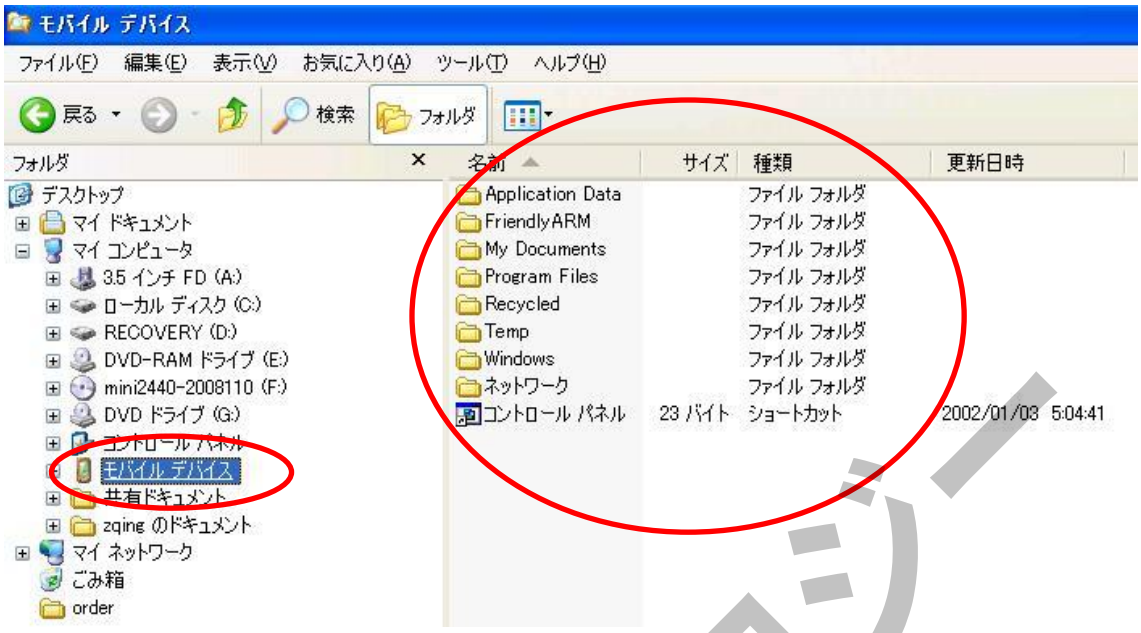
インストールが完了しました。



この画面の「キャンセル」を押します。

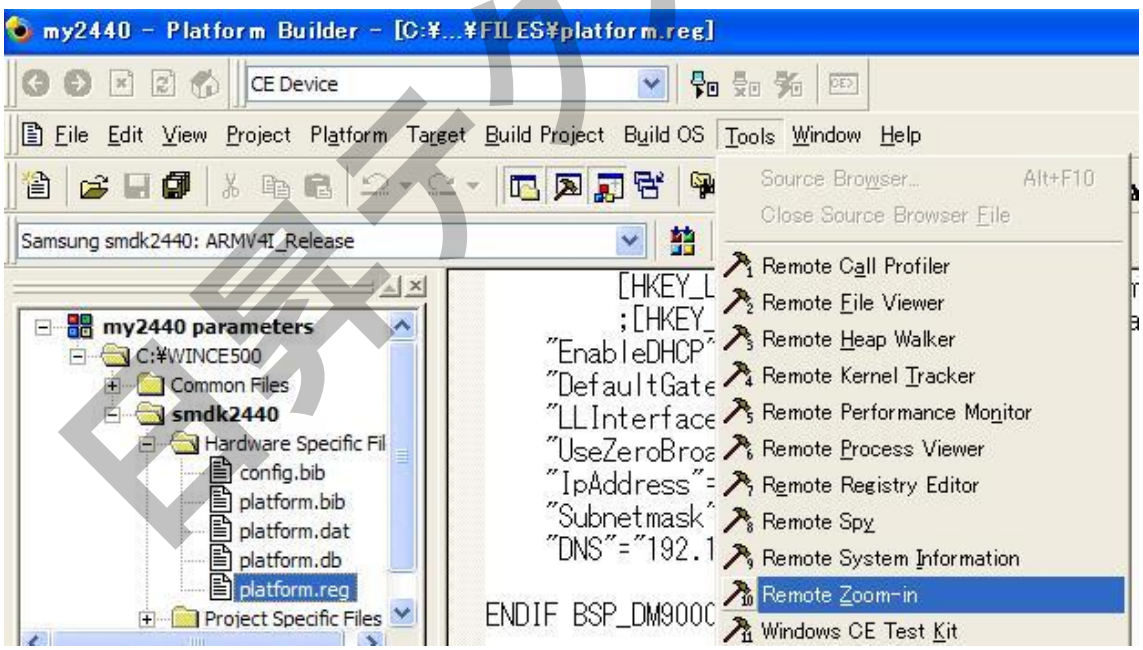


「OK」ボタンを押します。ARM9 ボードのフォルダが見えます。パソコンは ARM9 ボードのフォルダへアクセスできます。



4.4 WinCE の画面を取る

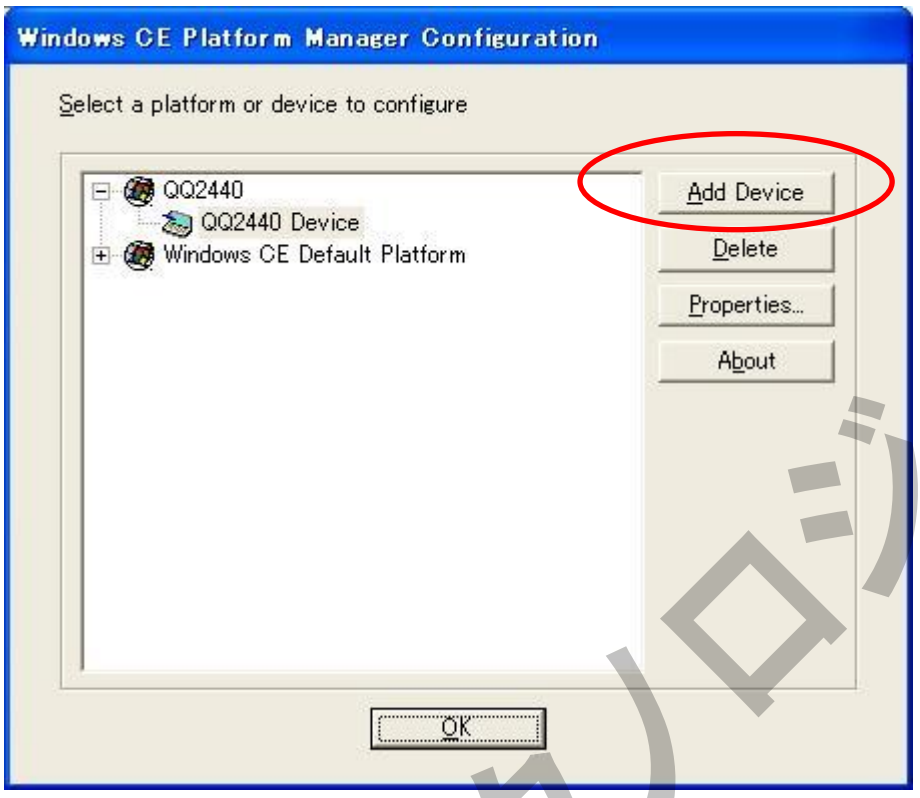
Platform Builder 5.0 を起動します。「ツール」→「リモート ズームイン」で



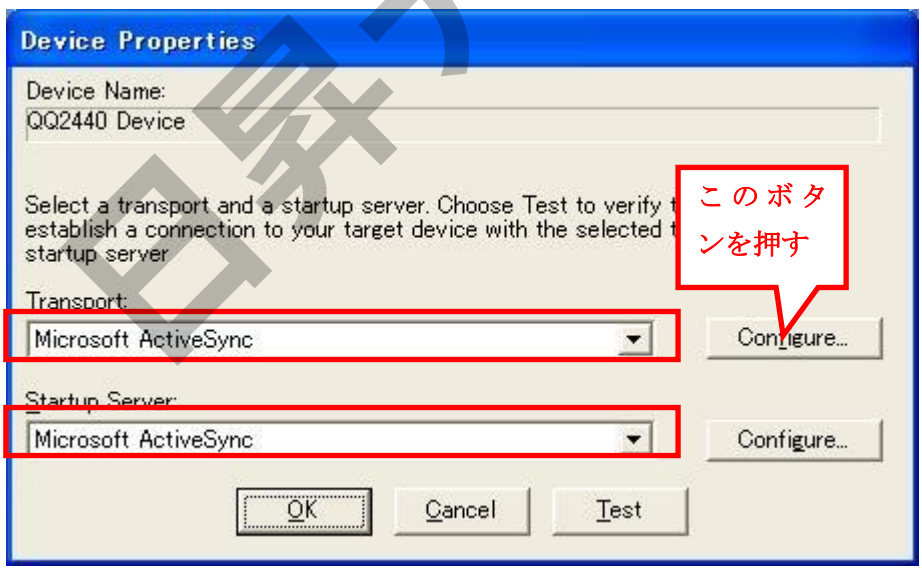
Windows CE リモート ズームインを実行させます。

Windows CE リモート ズームインの「接続」→「Windows CE Platform Manager の構成」を選択します。

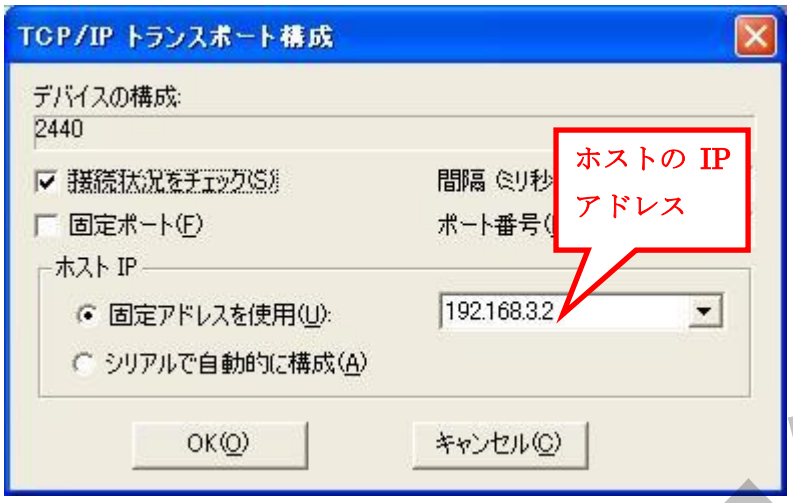
「QQ2440」というデバイスを追加します。



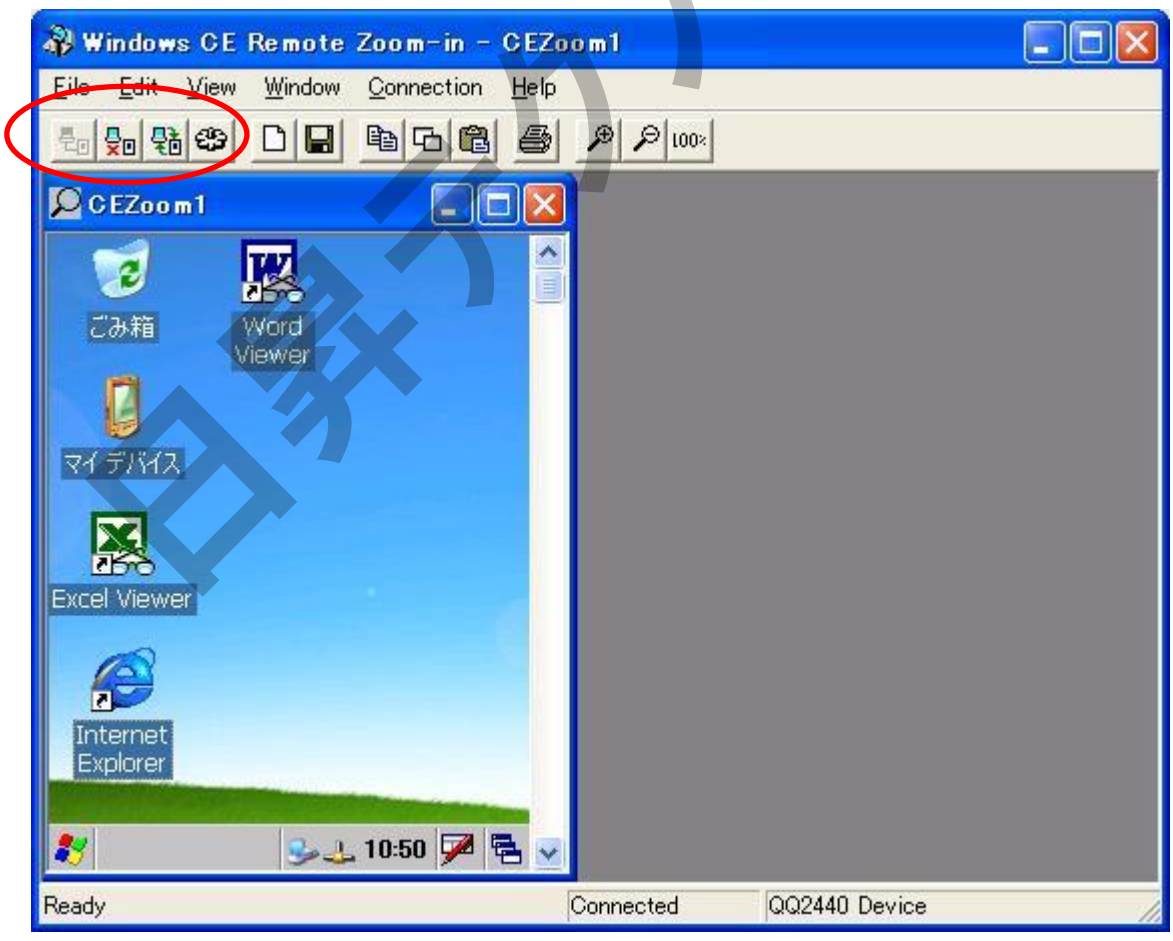
「Properties」ボタンをおして、「Transport」と「Startup Server」を画面のような項目選択します。



ホストの IP アドレスと ARM9 ボードの IP アドレスは同じサブネットワークですか、確認してください。異なれば、ホスト側又は WinCE 側の IP アドレスを変更します。

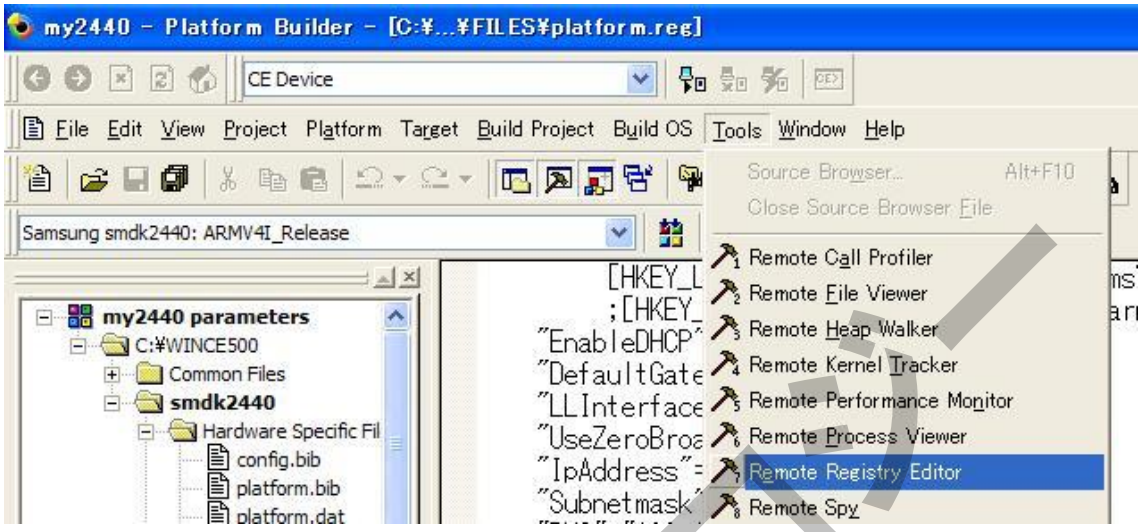


設定終了すれば。赤い丸の中のボタンを使用して、WinCE 側の画面を取れます。画面を取るのは LAN ケーブルが必要です。

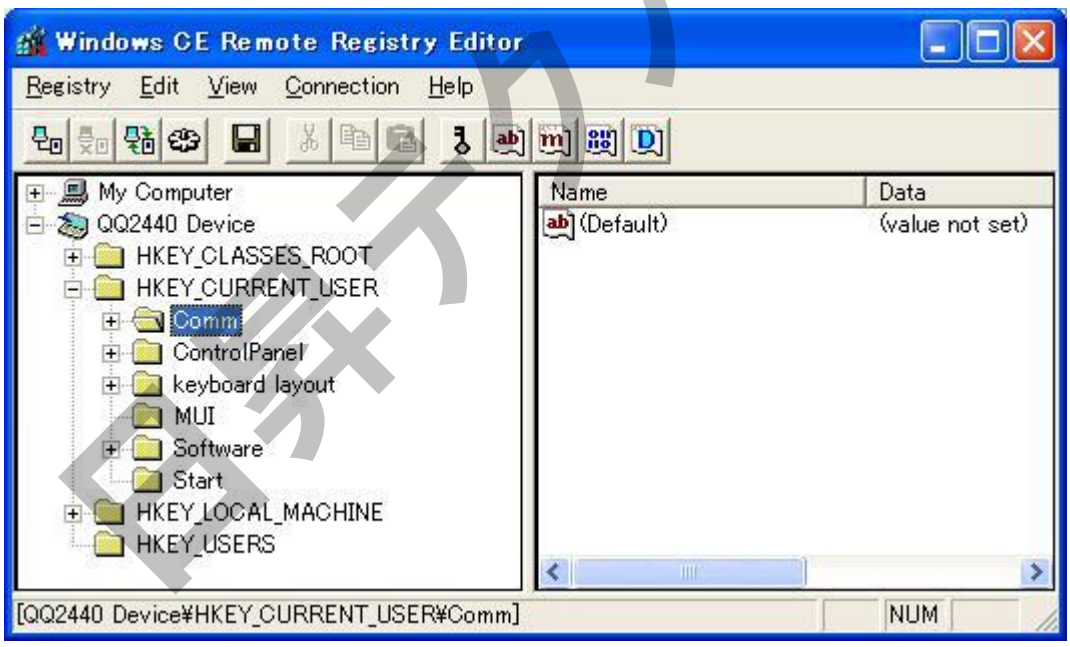


4.5 WinCE のレジストリを編集

Platform Builder 5.0 を起動します。「ツール」→「リモート レジストリ エディタ」で



Windows CE リモート レジストリ エディタを実行させます。



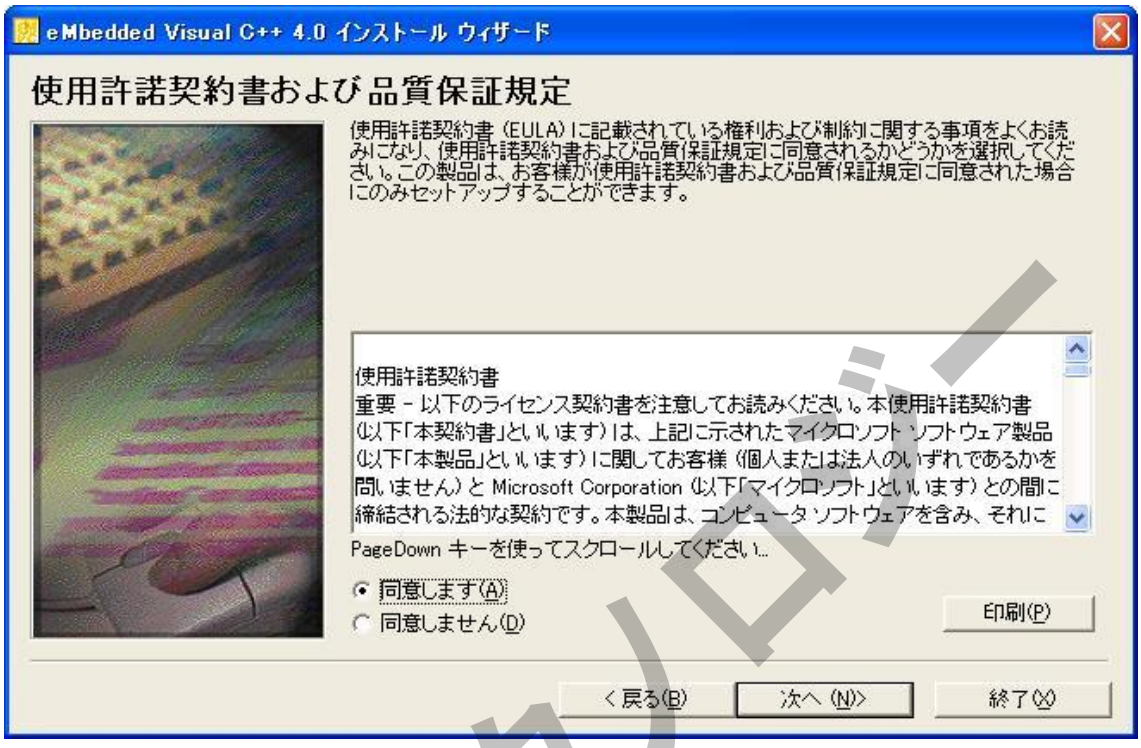
ホスト側で WinCE のレジストリを編集できます。

第五章 開発環境をインストールする

5.1 eMbedded Visual C++ 4.0 をインストールする

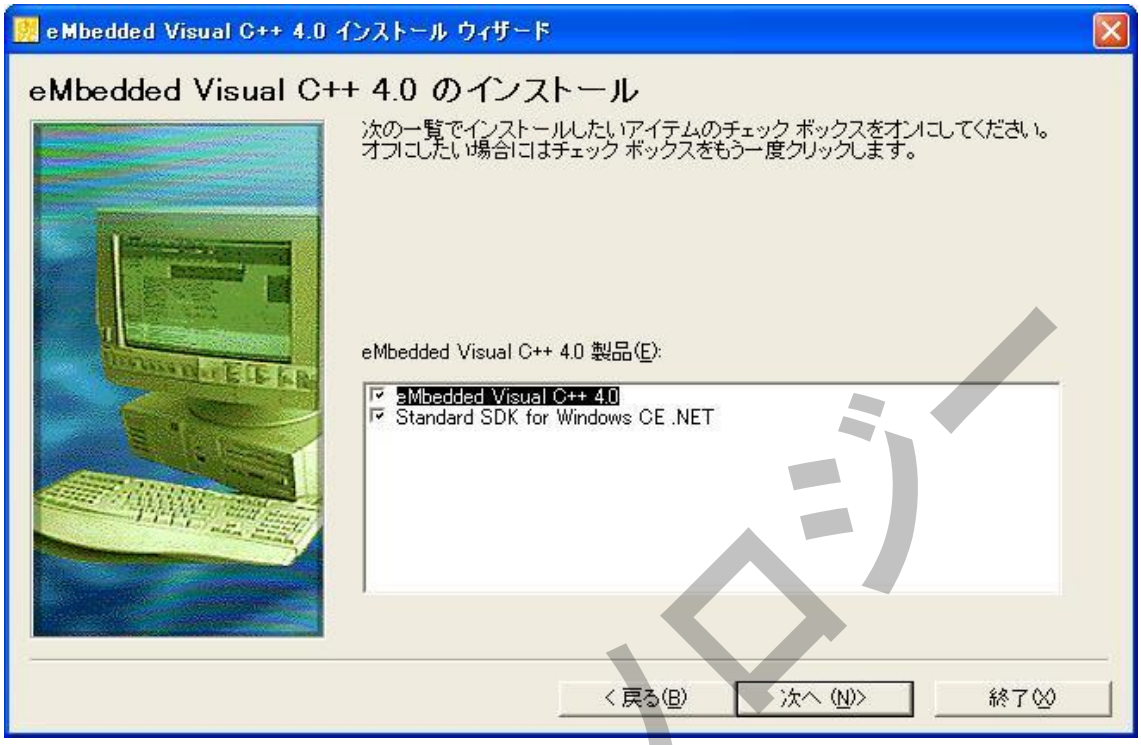
JA_eVC4.exe を実行して、eMbedded Visual C++ 4.0 を解凍します。解凍されたフォルダの中の setup.exe を実行します。「次へ」ボタンを押します。

契約書を読んで、同意できる場合は、「**同意します**」を選択して、「次へ」ボタンを押します。

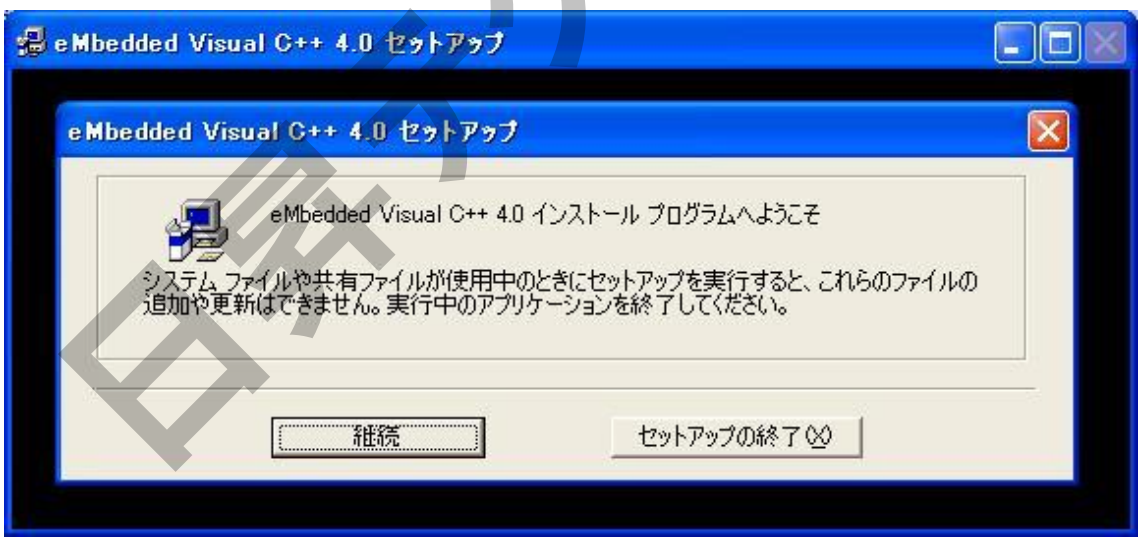


プロダクト ID とユーザー名を入力します。「次へ」ボタンを押します。

「次へ」ボタンを押します。



「継続」ボタンを押します。

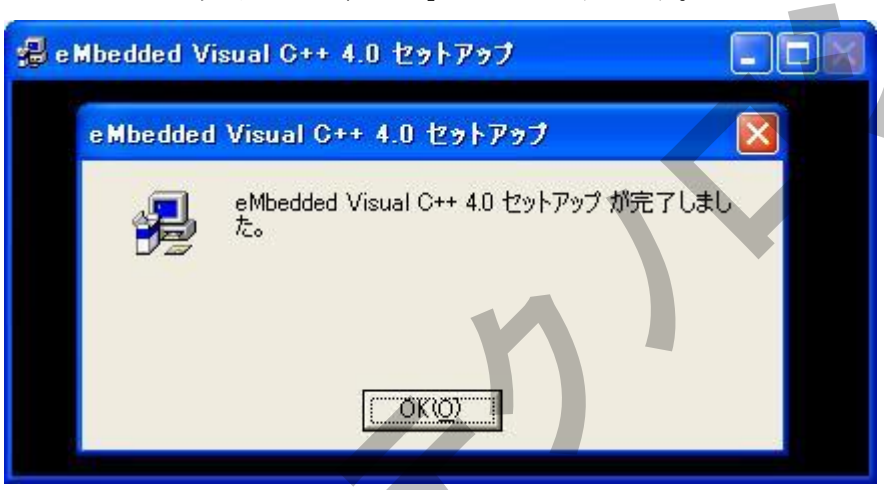


「OK」ボタンを押します。

「継続」ボタンを押します。

eMbedded Visual C++ 4.0 がインストール中です。

インストールが完了したら、「OK」ボタンを押します。



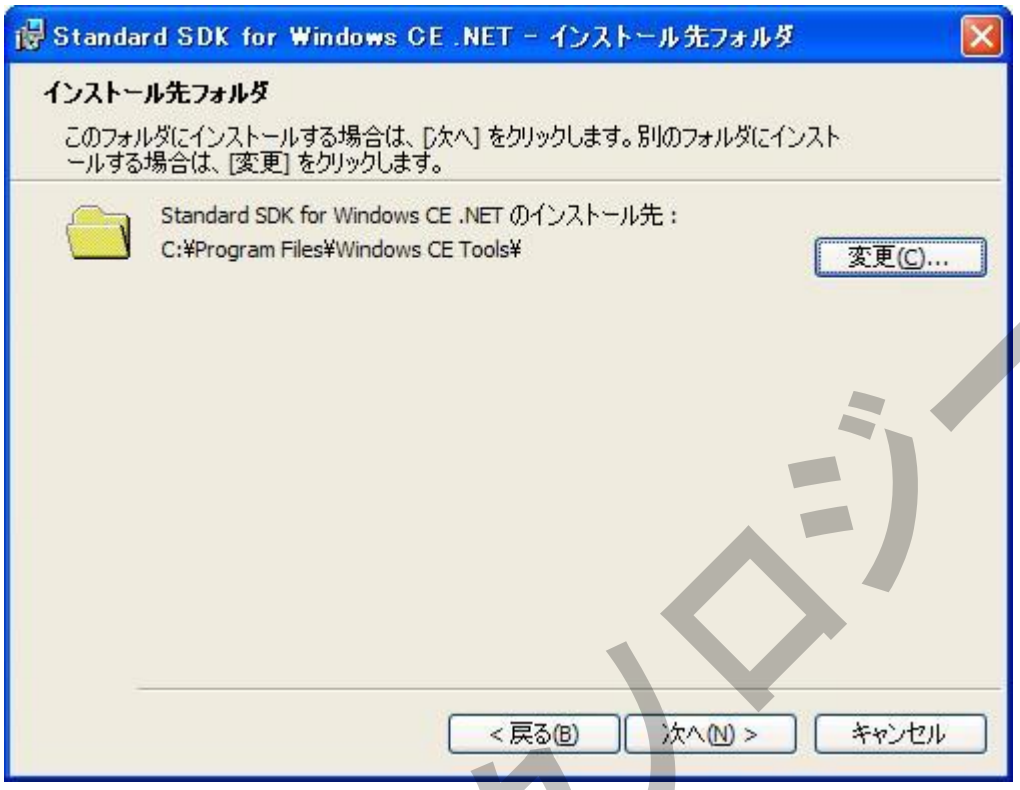
続きは Standard SDK をインストールします。「次へ」ボタンを押します。

契約書を読んで、同意すれば、「次へ」ボタンを押します。

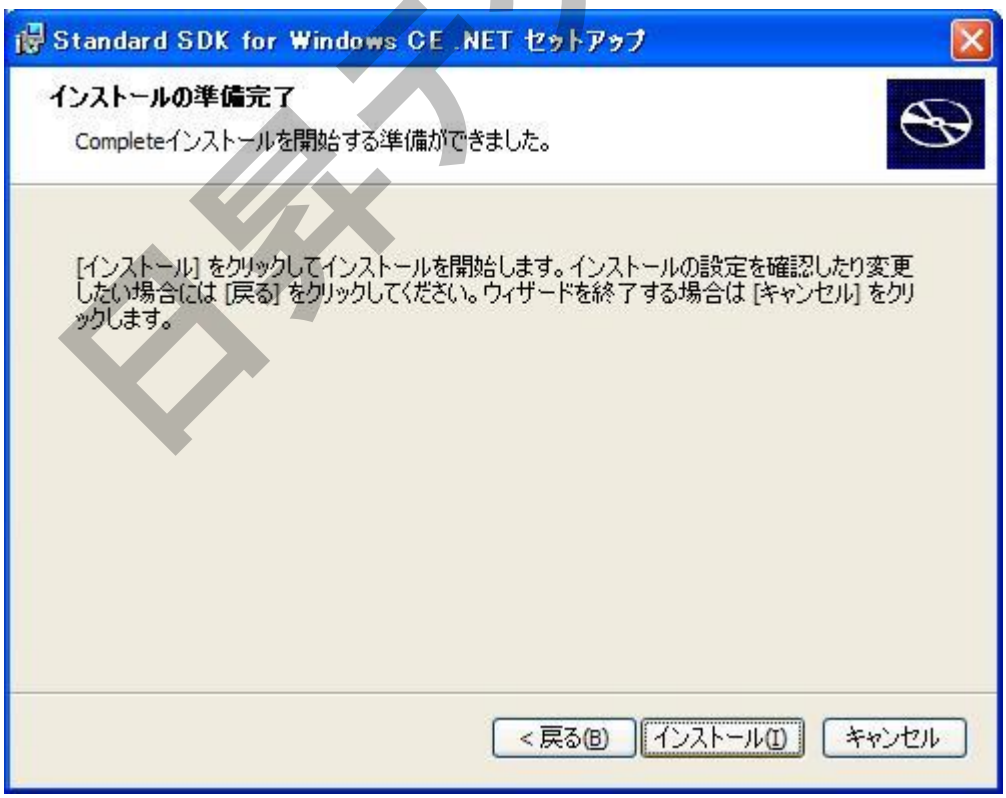
ユーザー情報を入力します。「次へ」ボタンを押します。

「完全」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。



インストールの準備完了、「インストール」ボタンを押します。

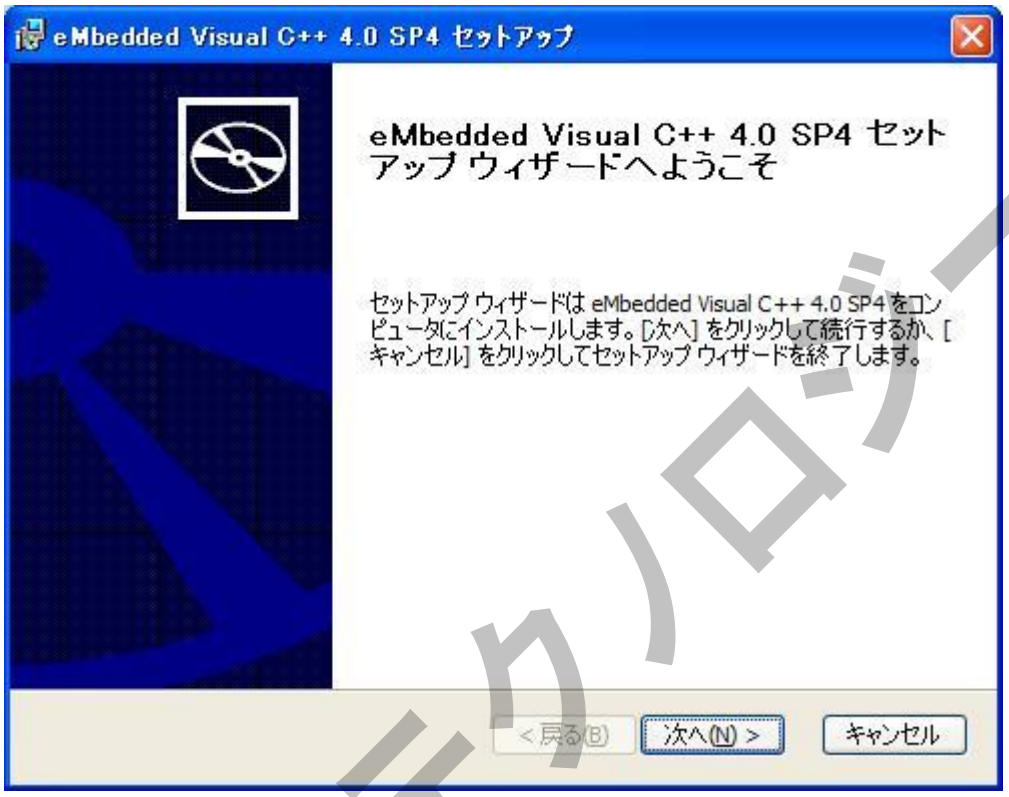


インストール中

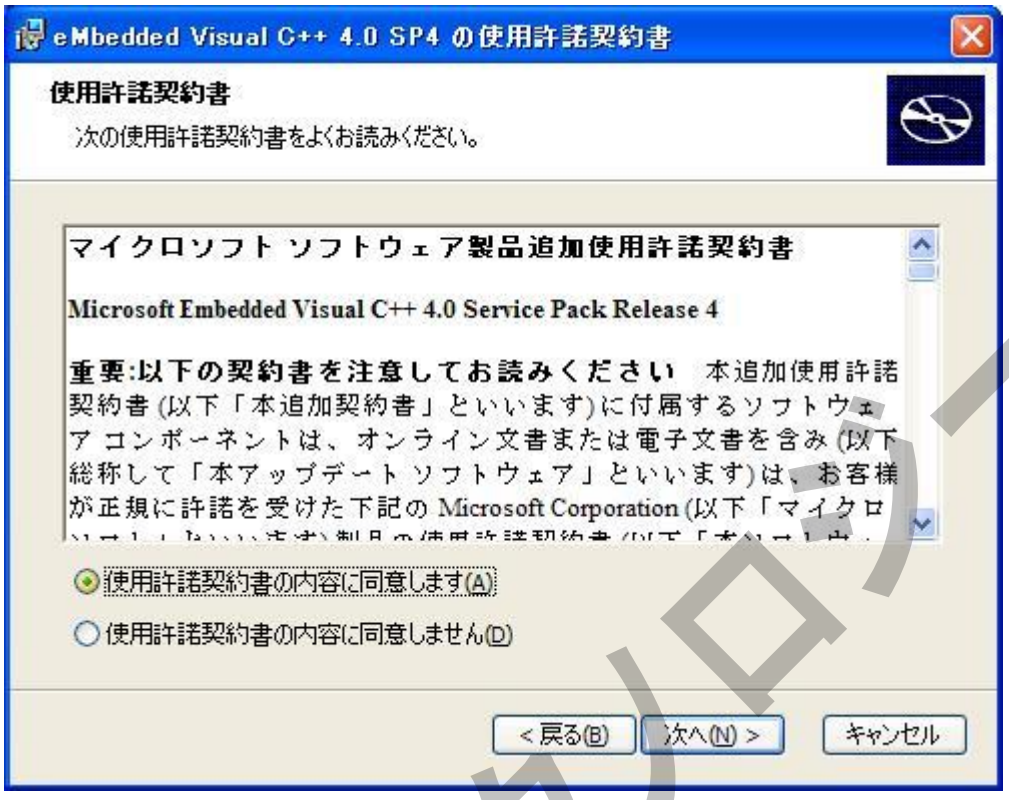
「終了」ボタンを押します。

5.2 パッチをインストールする

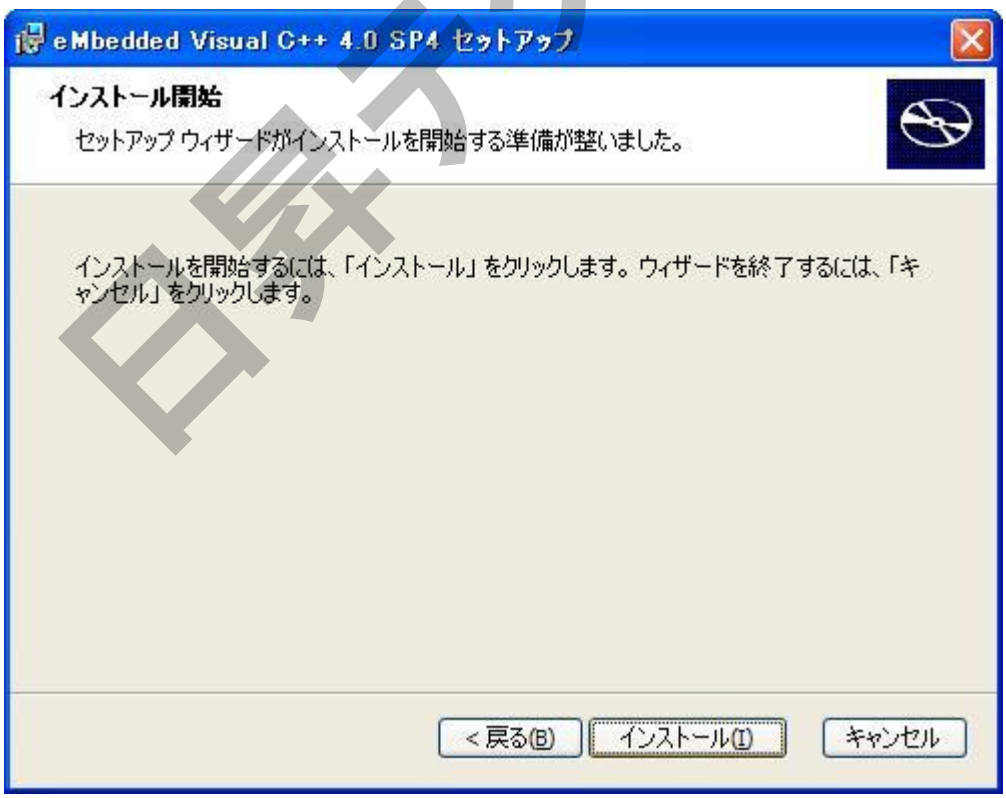
evc4sp4JPN.exe を実行して、eMbedded Visual C++ 4.0 のパッチを解凍します。解凍されたフォルダの中の setup.exe を実行します。「次へ」ボタンを押します。



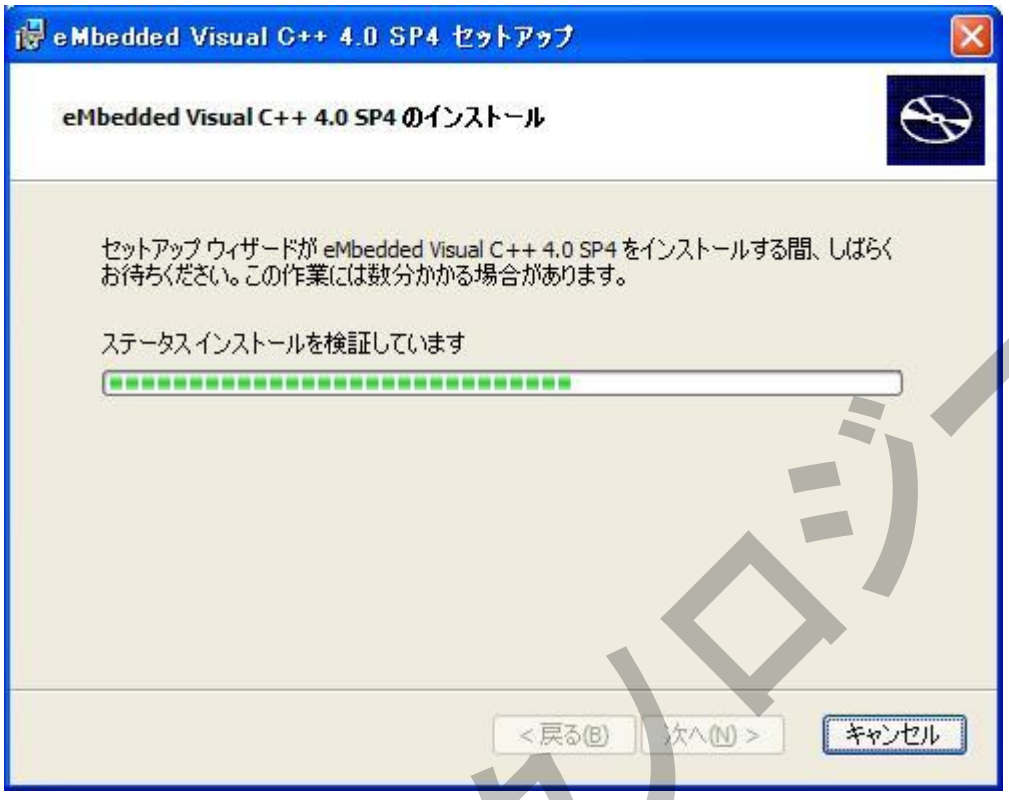
契約書を読んで、同意すれば、「次へ」を押します。



「インストール」ボタンを押して、インストールを開始します。



インストール中です。

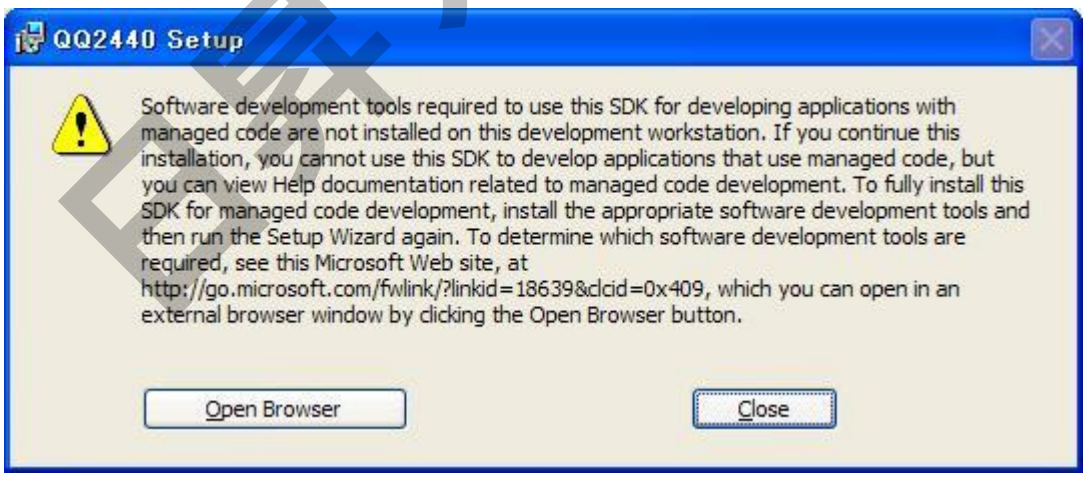


「完了」ボタンを押します。

5.3 ARM9 ボードの SDK をインストール

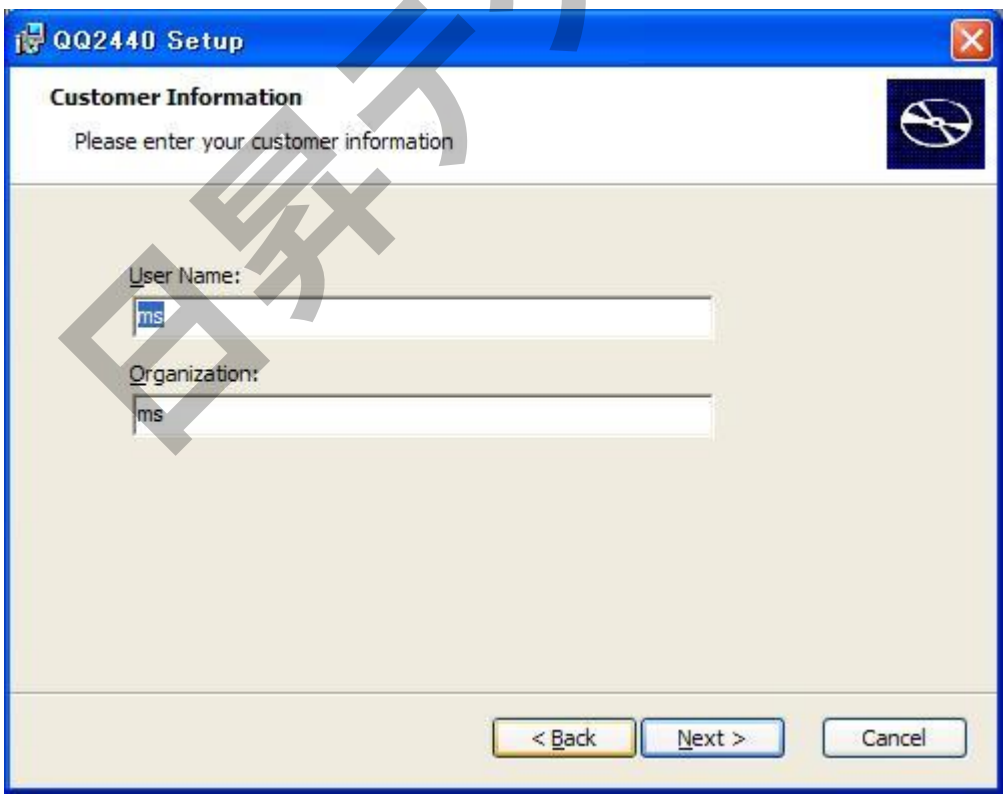
QQ2440_SDK.msi を実行します。「Next」ボタンを押します。

「Close」ボタンを押します。

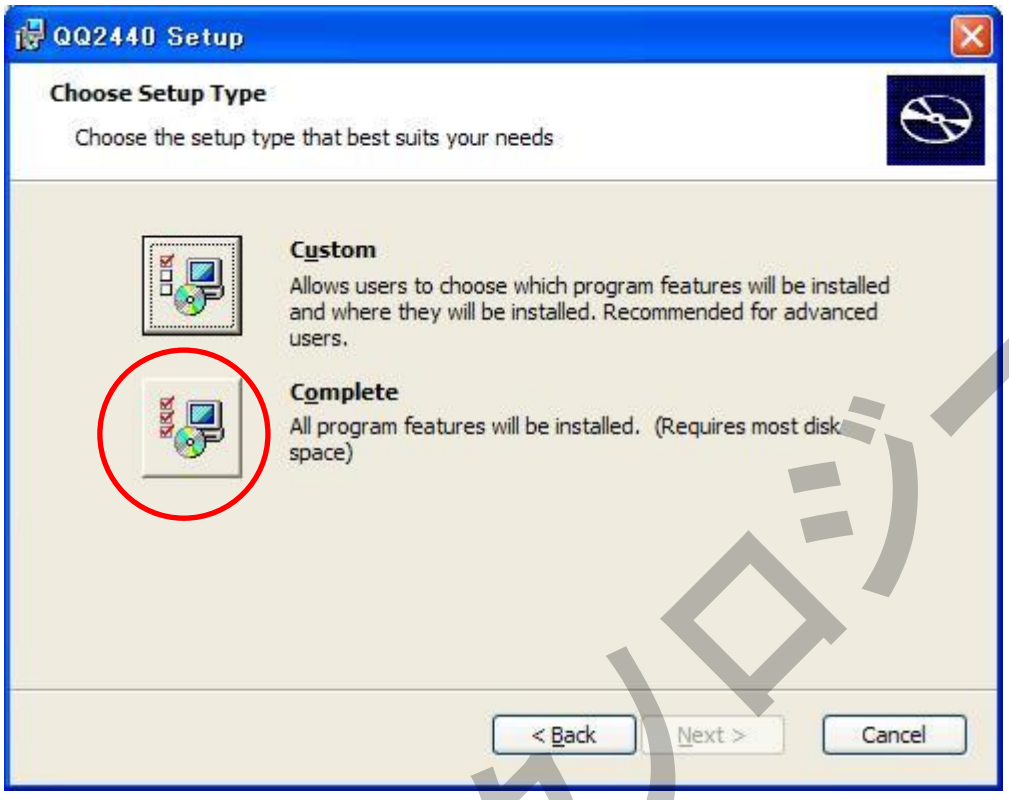


契約書を読んで、同意すれば、「Next」ボタンを押します。

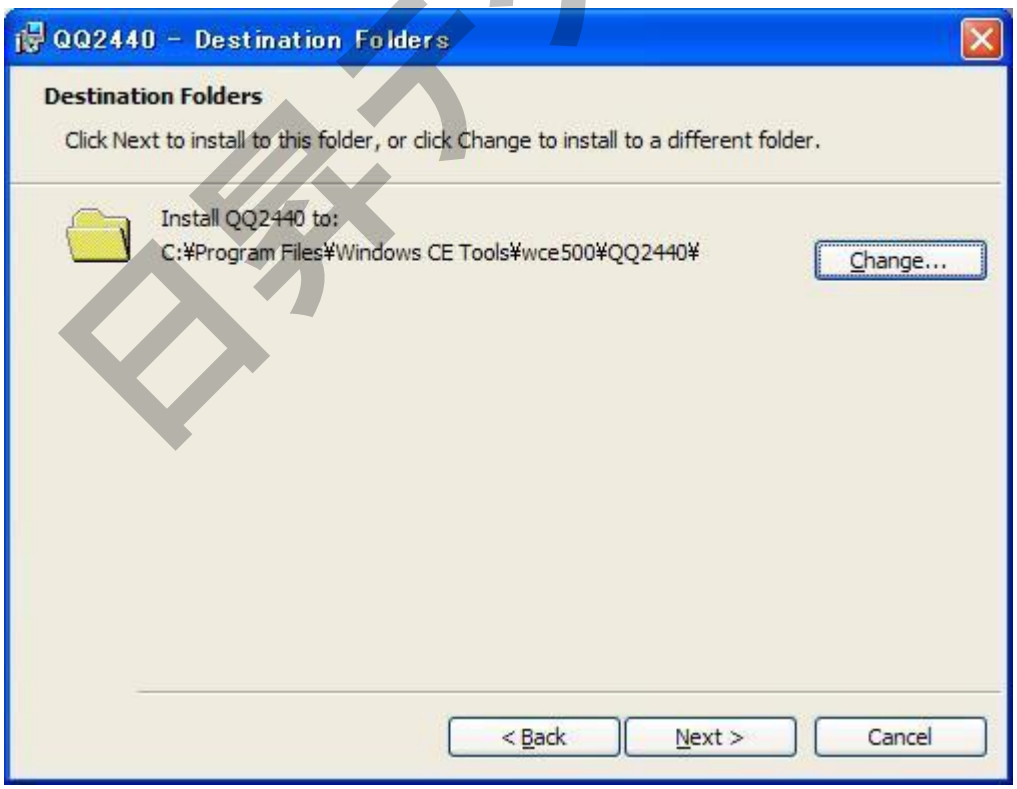
ユーザー情報を入力してください。



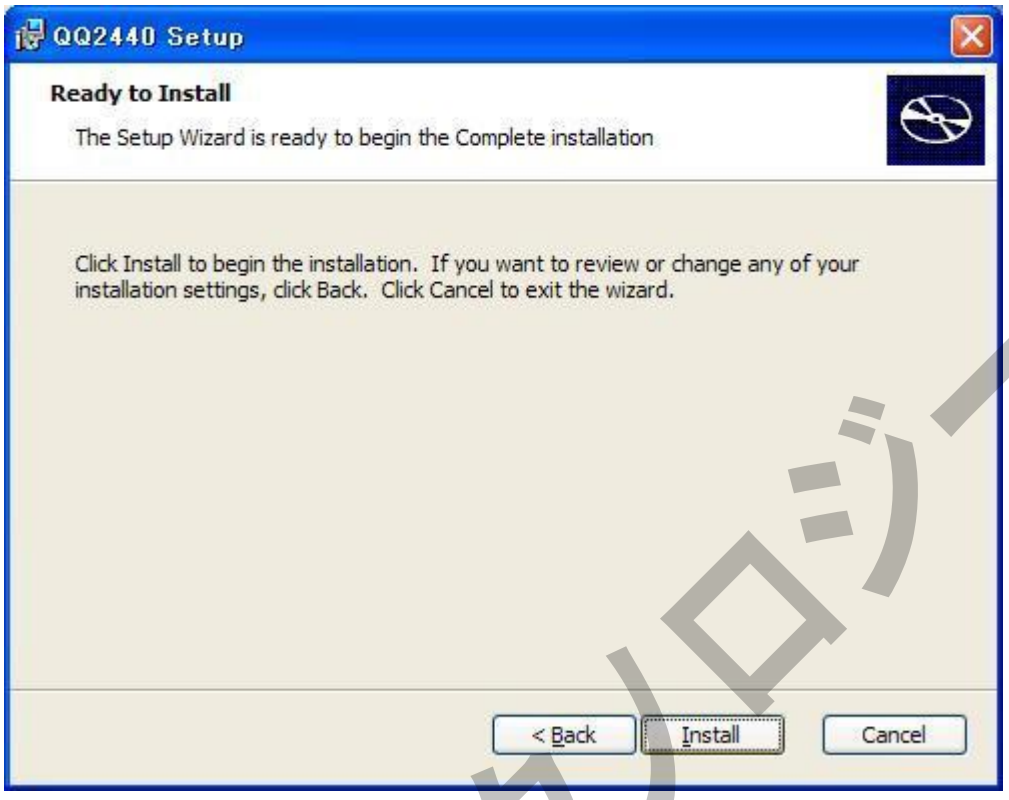
「Complete」ボタンを押します。



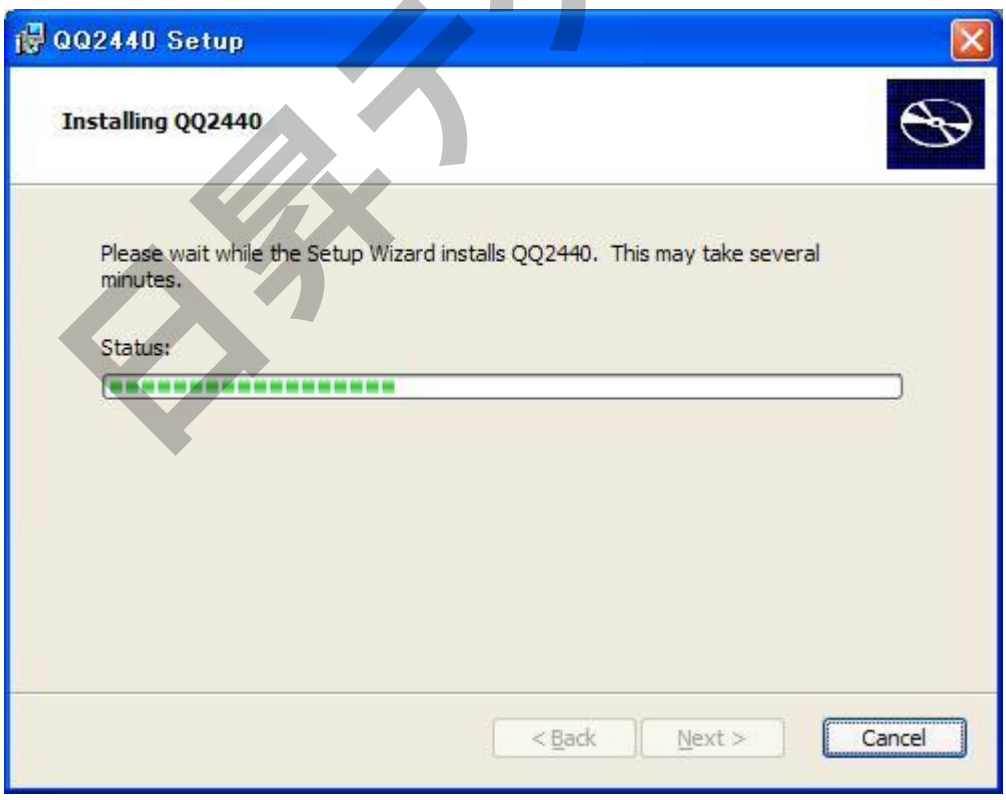
インストール先フォルダを変更せず、そのまま進んでください。



「Install」ボタンを押して、インストールを開始します。



インストール中です。

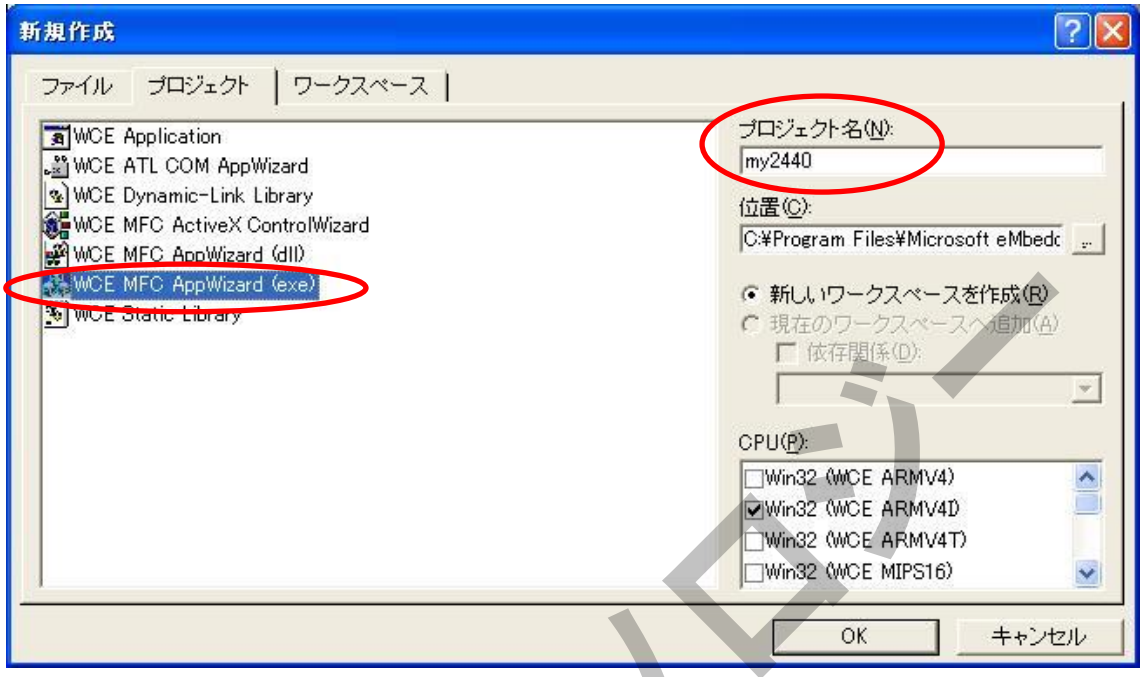


「Finish」ボタンを押します。

5.4 EVC の Hello!

今から WinCE 環境で簡単なプログラム Hello を作ります。一行のコードも書くことが必要ないです。eMbedded Visual C++ 4.0 を起動します。「ファイル」→「新規作成」を選択します。

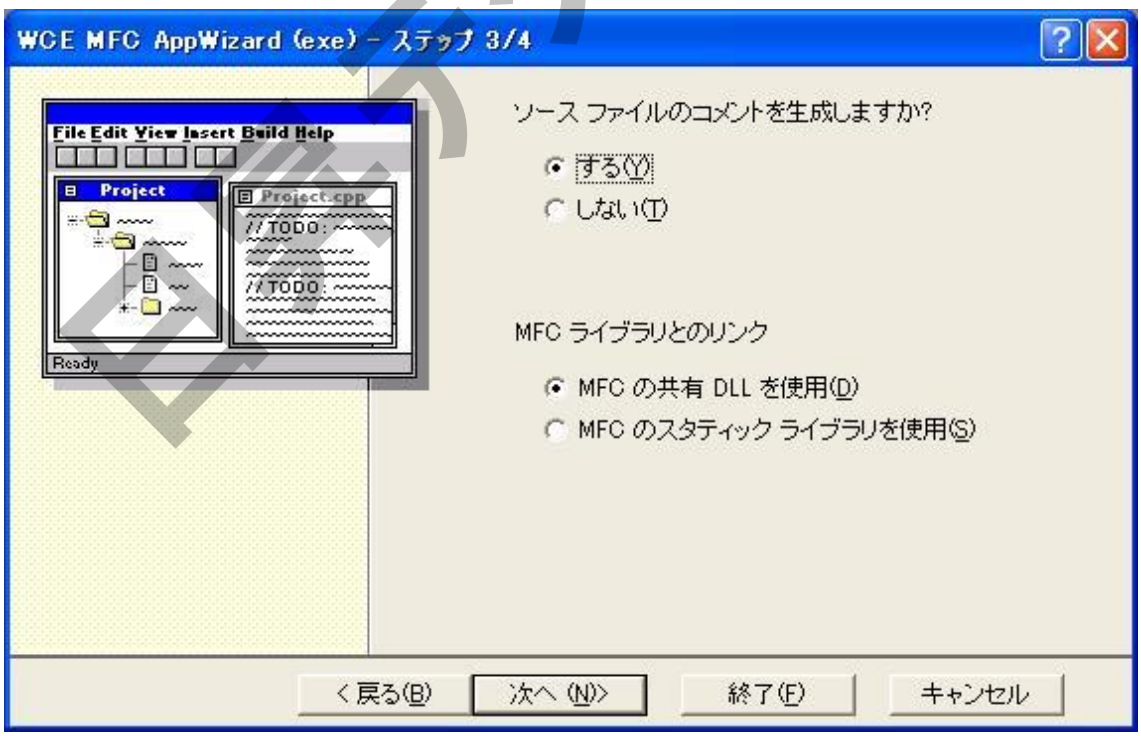
「WCE MFC AppWizard(exe)」を選択して、プロジェクト名も入力して、「OK」ボタンを押します。



変更せず、「次へ」押します。

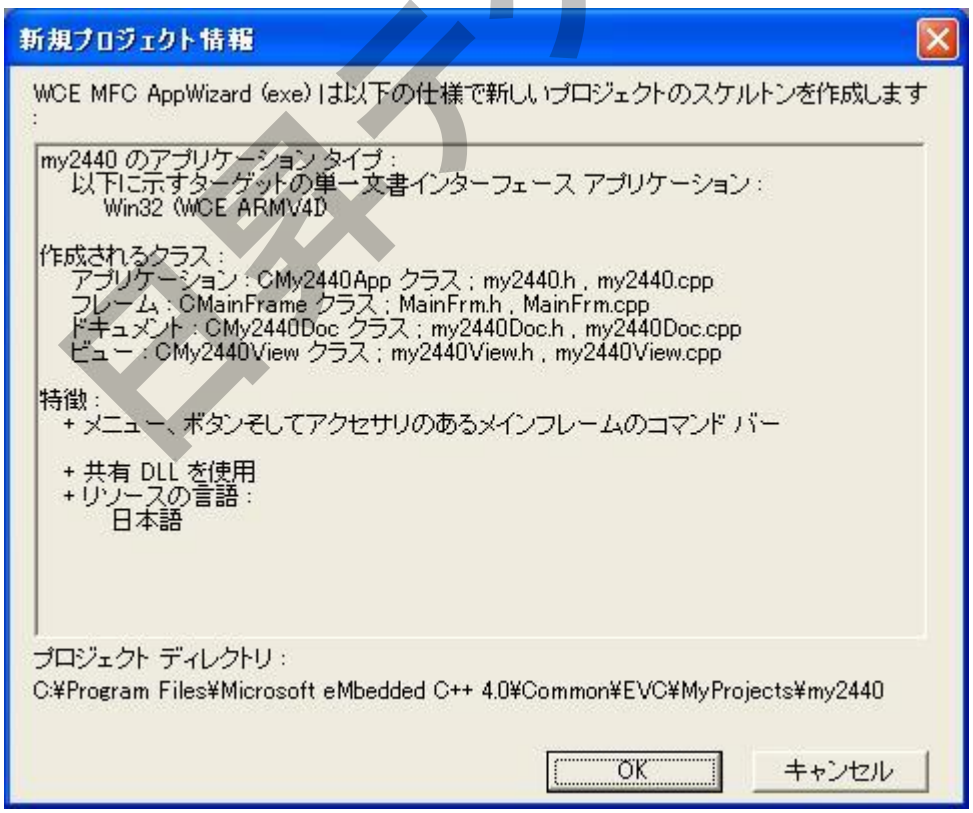
変更しないまま、「次へ」ボタンを押します。

変更しないまま、「次へ」ボタンを押します。



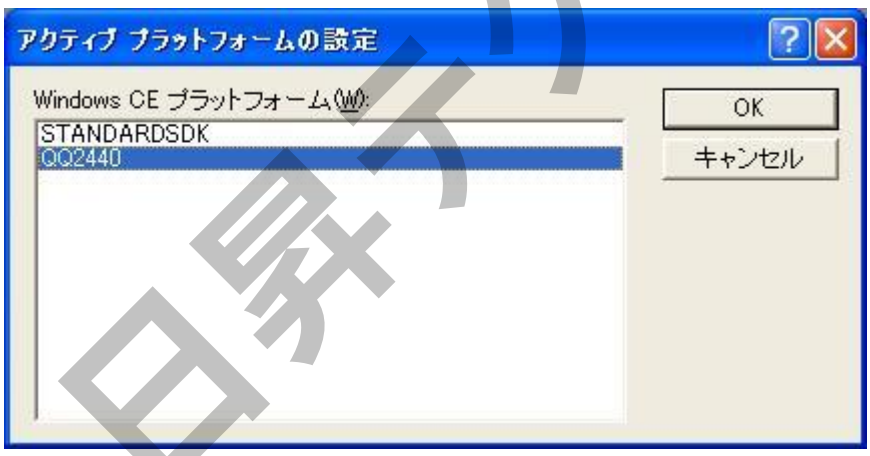
「終了」ボタンを押します。

最後の確認、「OK」ボタンを押します。

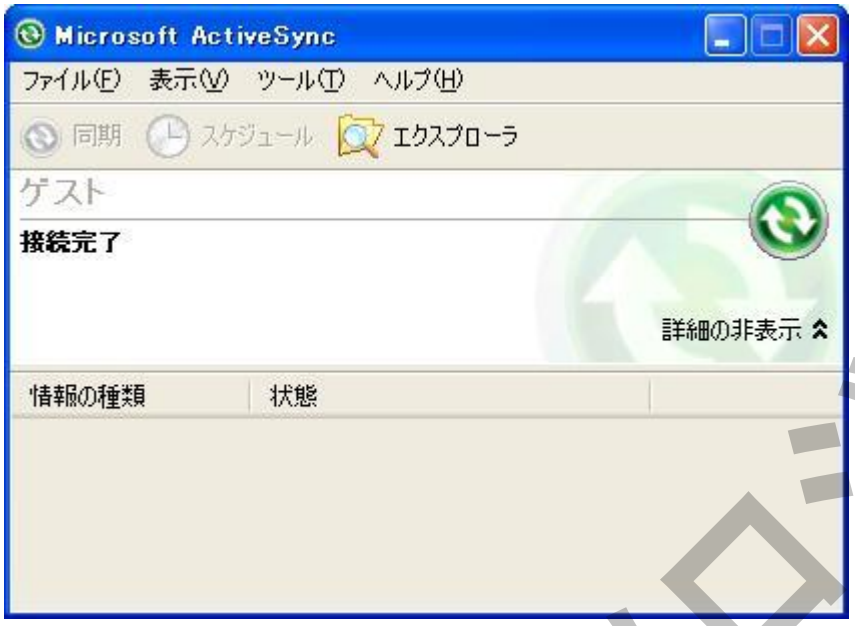


メインメニューに戻ります。「ビルド」→「アクティブ プラットフォームの設定」を選択します。

QQ2440 を選択して、「OK」ボタンを押します。



ビルドの前、ホストと ARM9 ボードを繋ぎますか、通信できますか、同じサブネットワークですか、確認しください。

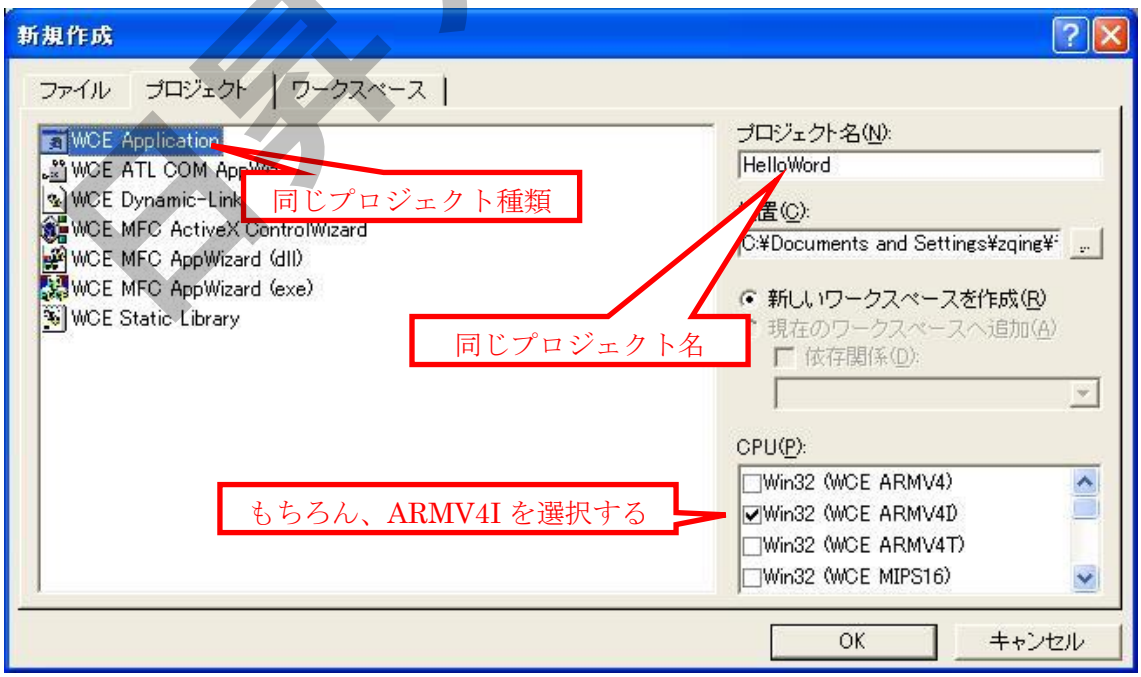


メインメニュー「ビルド」→「リビルド」を選択します。

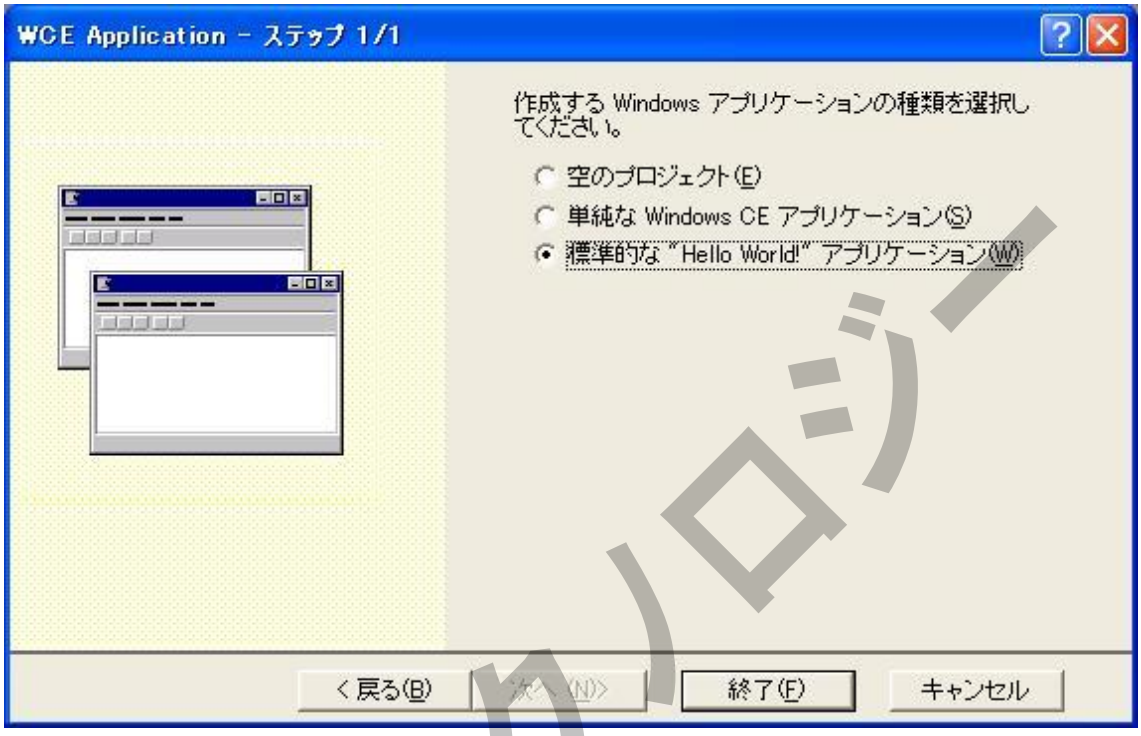
ビルドが完了すれば、自動的に生成された実行ファイルを ARM9 ボードにロードします。ARM9 ボードの「マイ デバイス」でこの実行ファイルが見えます。実行してみましょう。

5.5 X86 プラットフォームの EVC プロジェクトを移植

X86 プラットフォームの EVC プロジェクトはたくさんがあります。EVC example.rar ファイルはたくさんの例があります。初の例 HelloWord を ARM9 に移植するのを紹介します。まず EVC で空きプロジェクトを作れます。元のプロジェクトと同じプロジェクト種類を選択して、同じプロジェクト名を入力します。



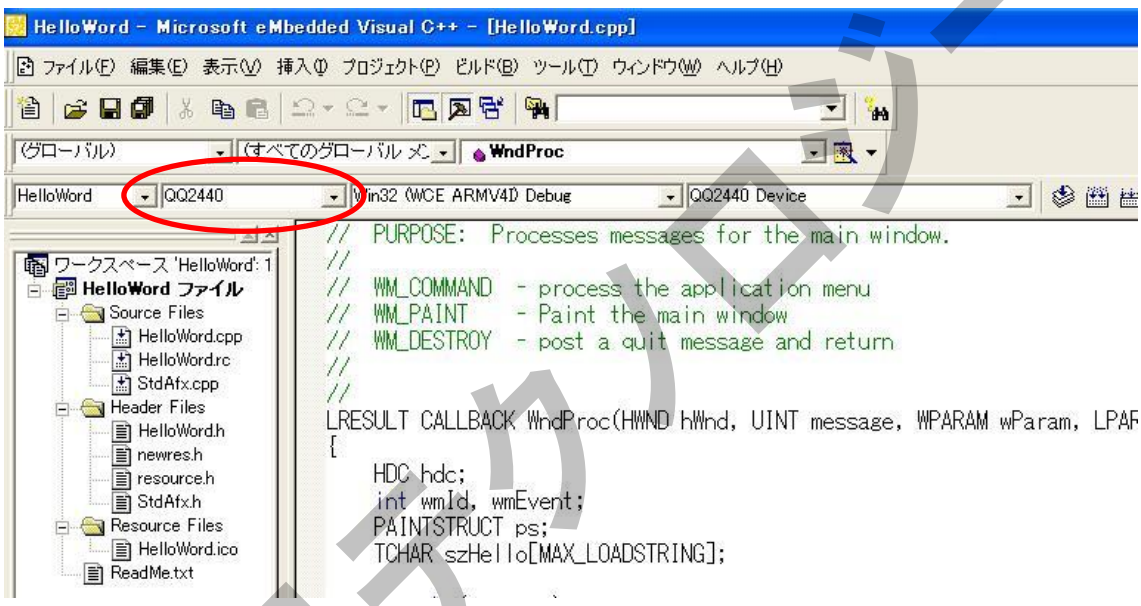
元のプロジェクトと同じアプリケーションの種類を選択します。SDI ですか、ダイアログ ベースですか。この例は標準的な"Hello World"アプリケーションです。「終了」ボタンを押します。EVC も終了します。



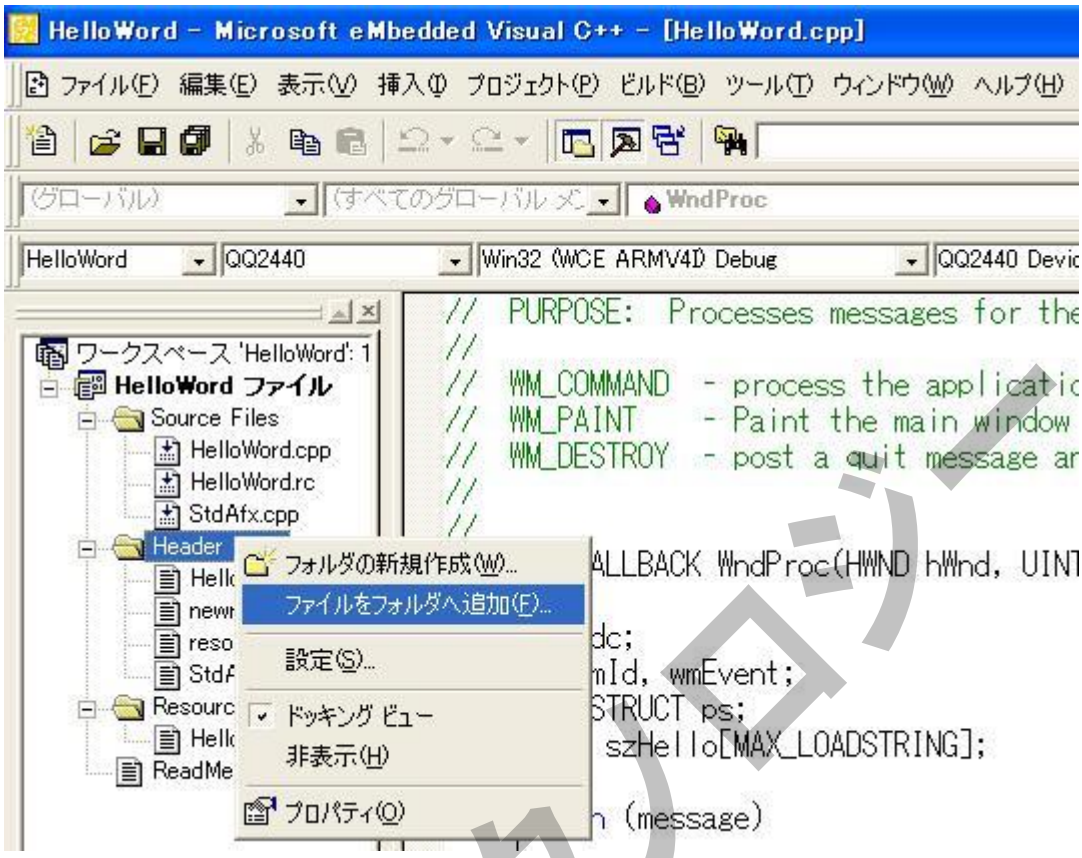
元のプロジェクトフォルダのファイルを新プロジェクトフォルダにコピーします。*.vcl, *.vcp, *.vcw 三つのファイルをコピーしません。

EVC を起動して、再び HelloWord プロジェクトを開きます。この警告情報が出てきます。ARM9 用のものを生成しました。「OK」ボタンを押します。

「QQ2440」を選択します。



ビルドを開始してみましょう。エラーがあれば、一つの原因はプロジェクトの設定と元のプロジェクトが異なります、直してみます。もう一つの原因は新規クラスを入りません、下の画面を通じて、プロジェクトにファイルを追加します。



ビルドが成功すれば、自動的に ARM9 にロードします。実行してみましょう。

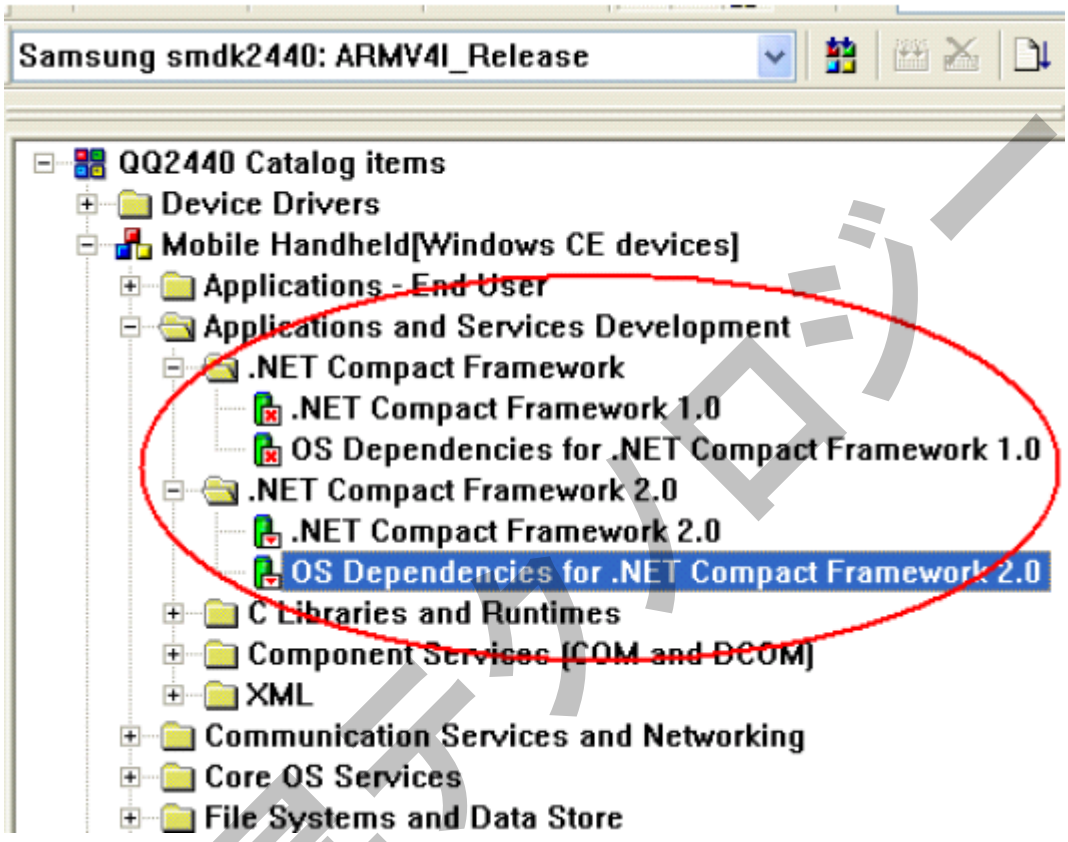
これらの例は中国語 EVC 教科書の付属例ですので、動くとき中国語のを表示します。

你好，世界

第六章 Visual Studio 2008 でプログラムを開発

Visual Studio 2005/2008 でプログラムを開発する前に、WinCE カーネルに.NET 2.0 があるかどうか、確認してください。

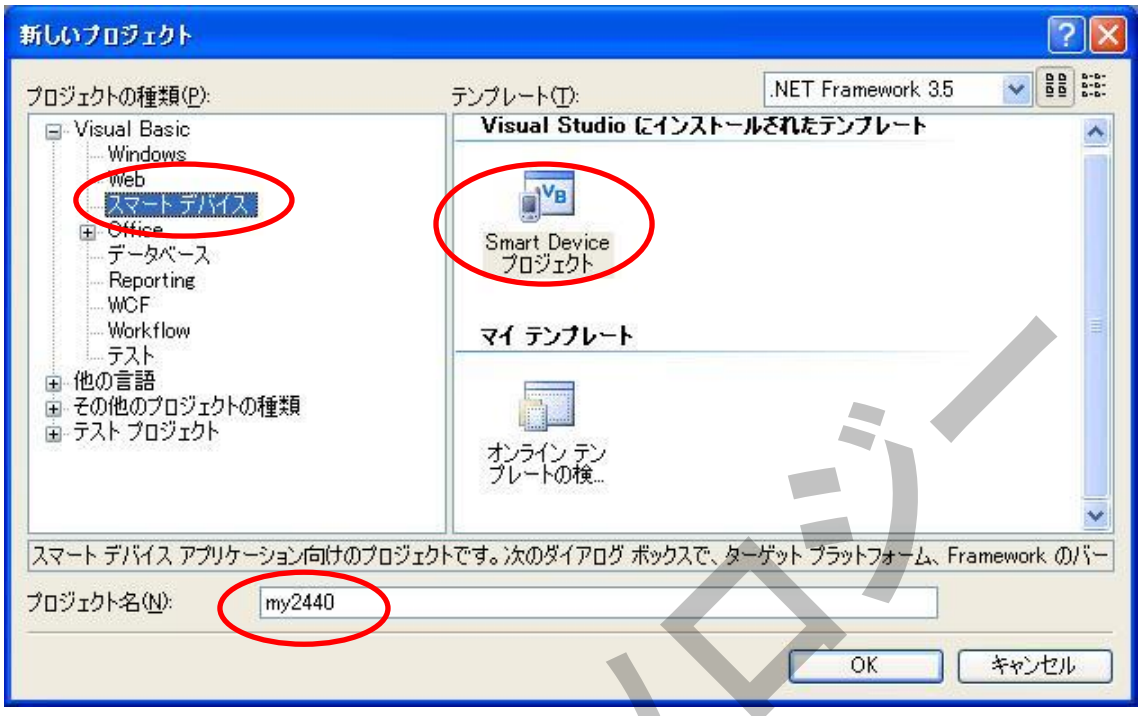
※ デフォルトの設定は.NET 2.0 があります。



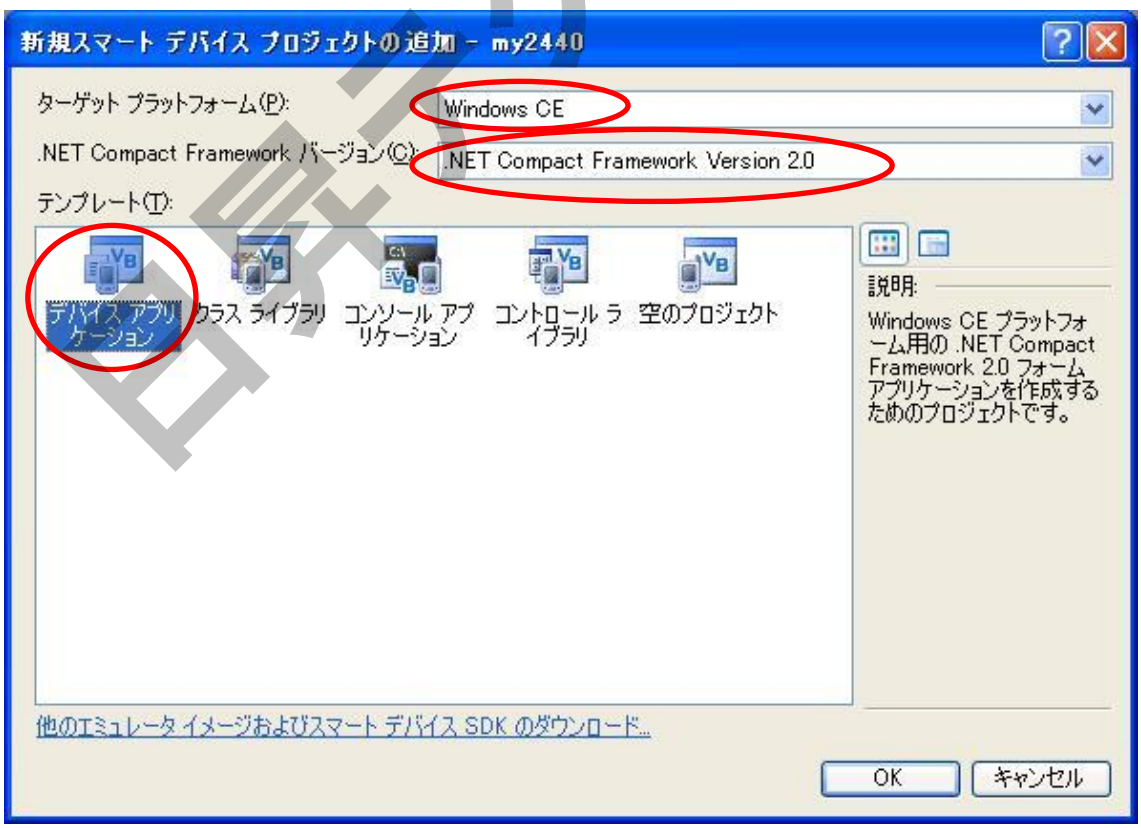
6.1 プロジェクトを作る

VS2008 を起動します。「ファイル」→「新しいプロジェクト」を選択します。

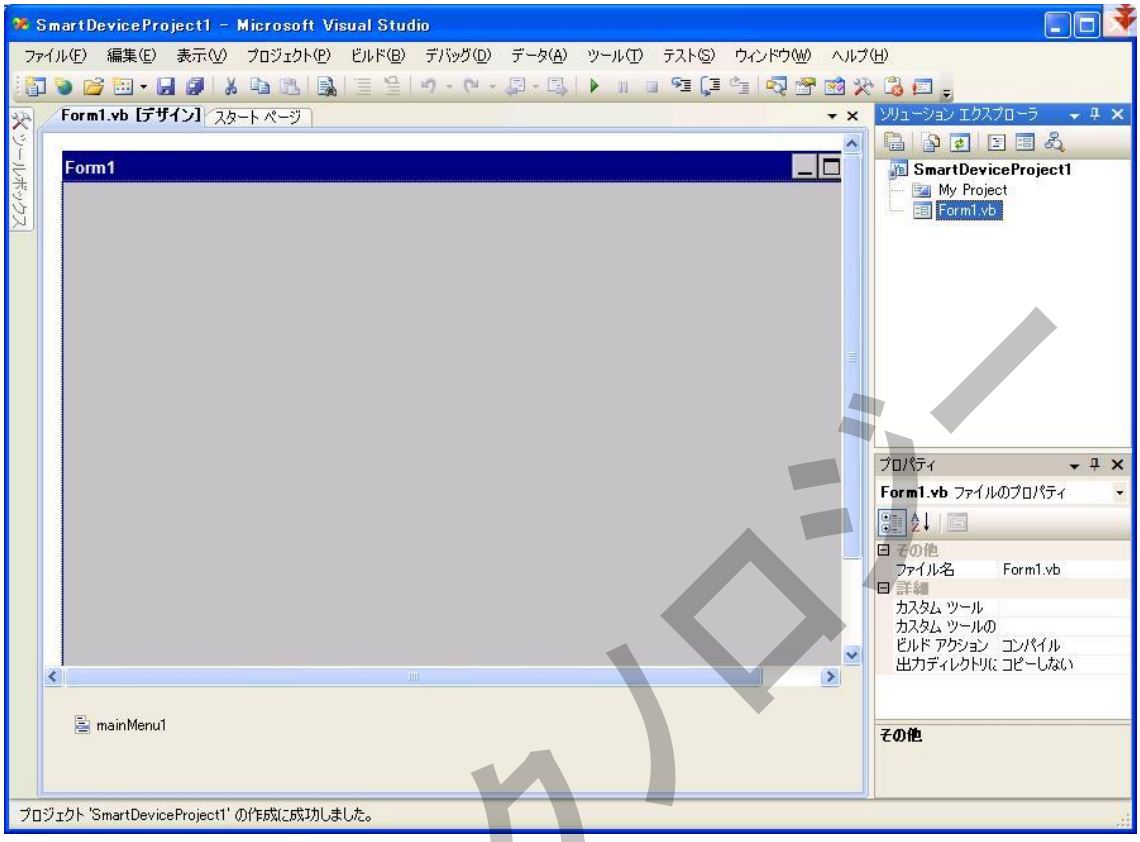
スマートデバイスを選択します。プロジェクト名も入力します。「OK」ボタンを押します。



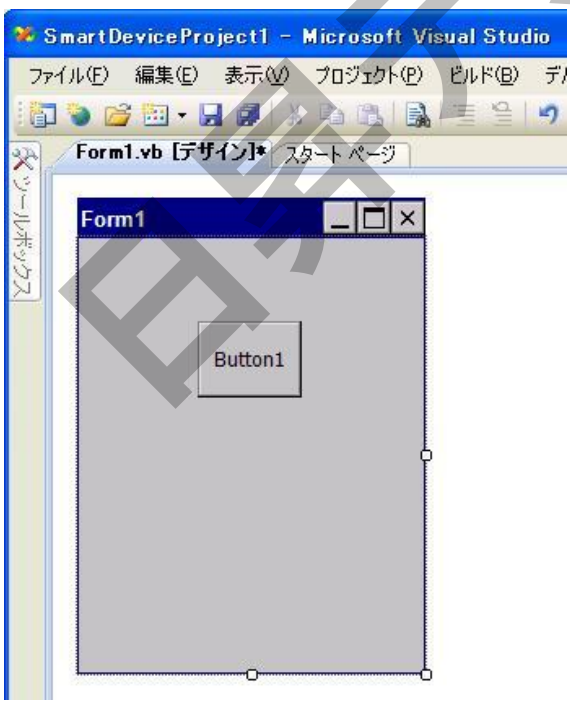
「Windows CE」、「.NET Compact Framework Version 2.0」、「デバイス アプリケーション」を選択します。「OK」ボタンを押します。



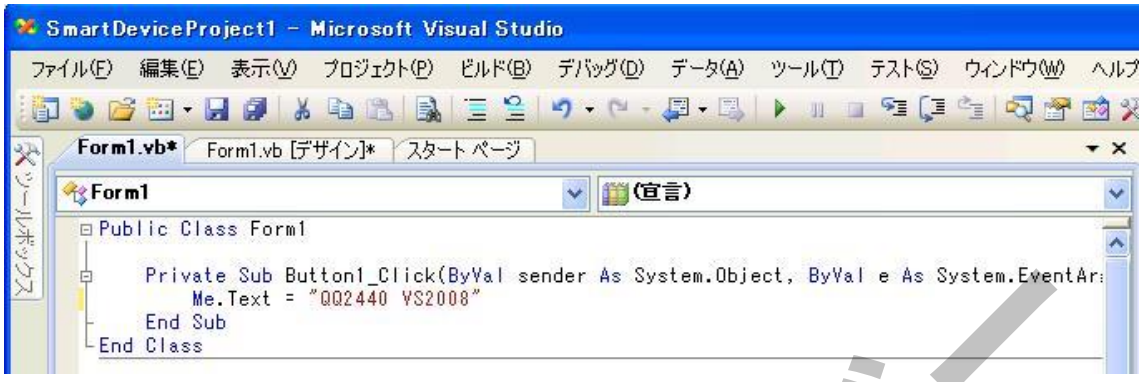
プログラムが動く画面が出てきます。



画面のサイズを調整して、あるボタンを載せます。



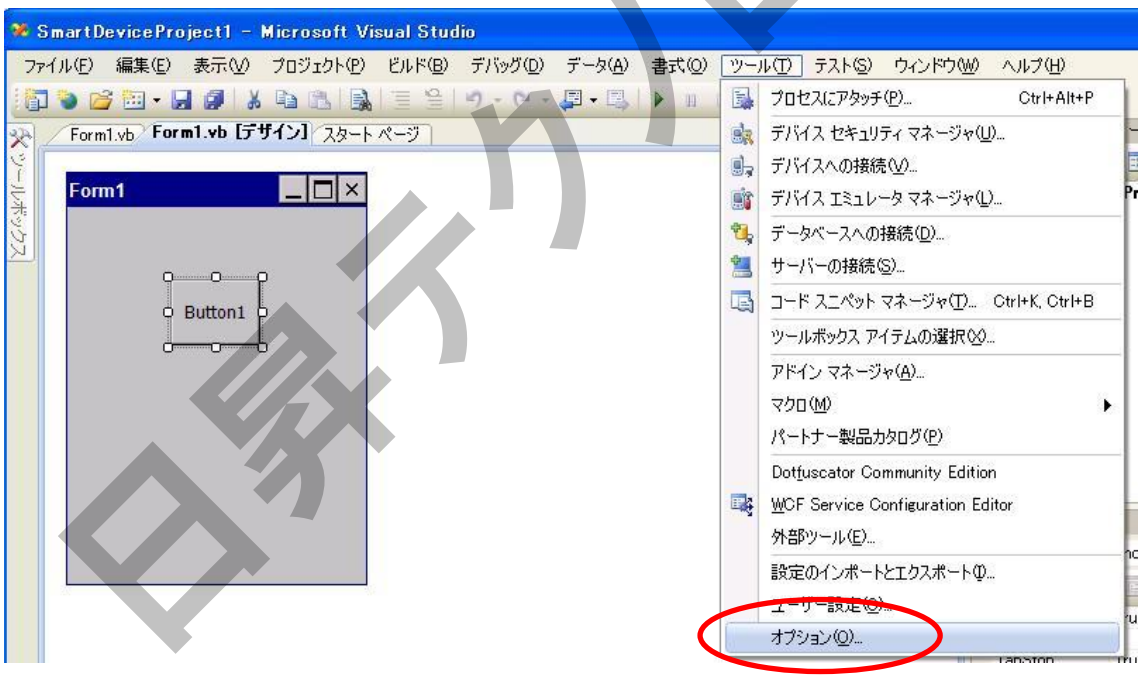
このボタンをダブルクリックすると、ボタンのコードを編集する画面が出てきます。次のコードを入ります。



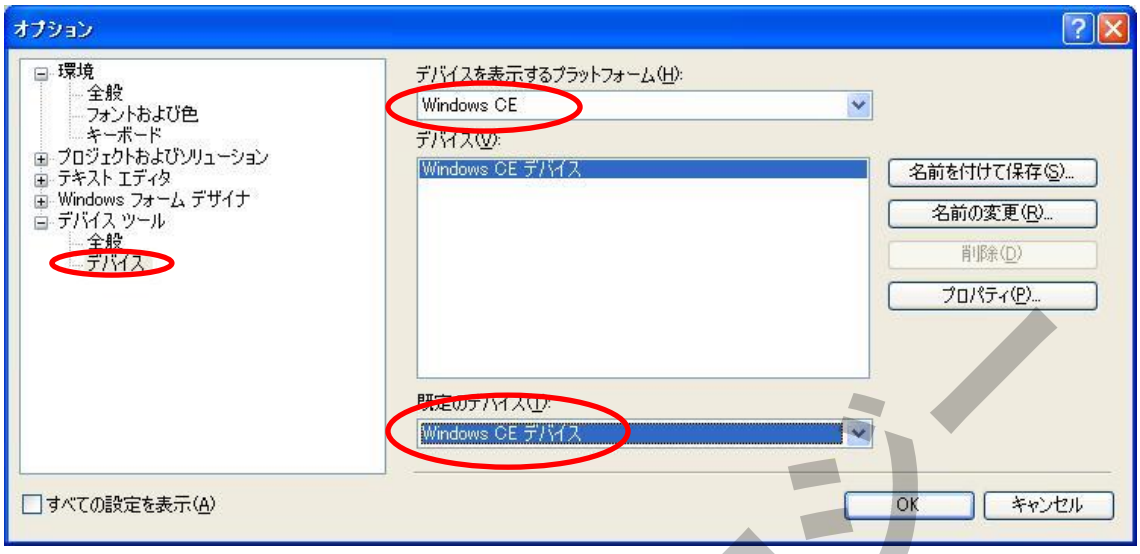
保存して、ARM9 ボードにロードして、実行してみましょう。

6.2 ARM9 ボードにロードする

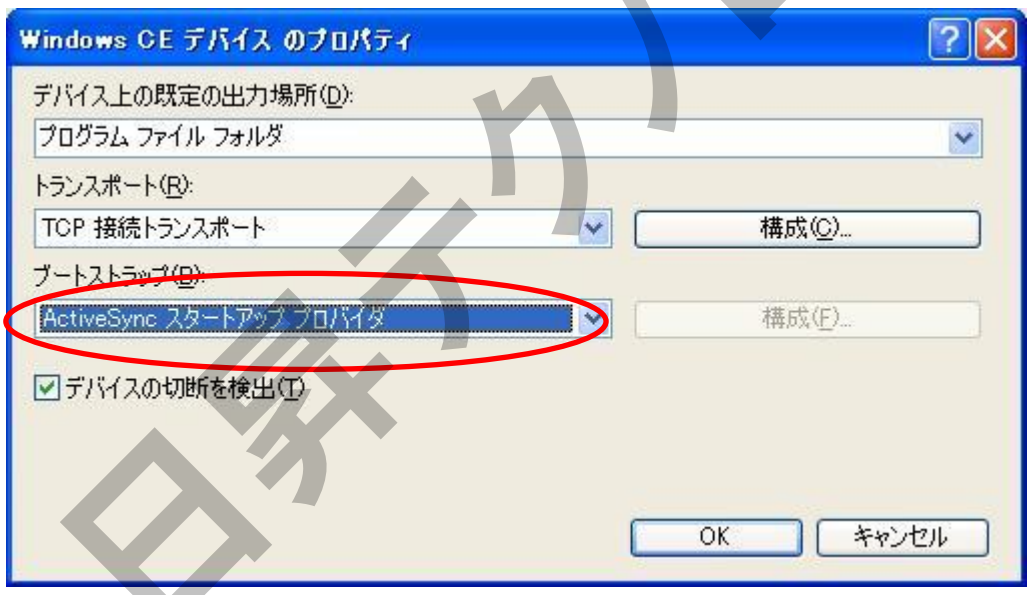
「ツール」→「オプション」を選択します。



「デバイス」と「Windows CE」を選択します。「プロパティ」ボタンを押します。

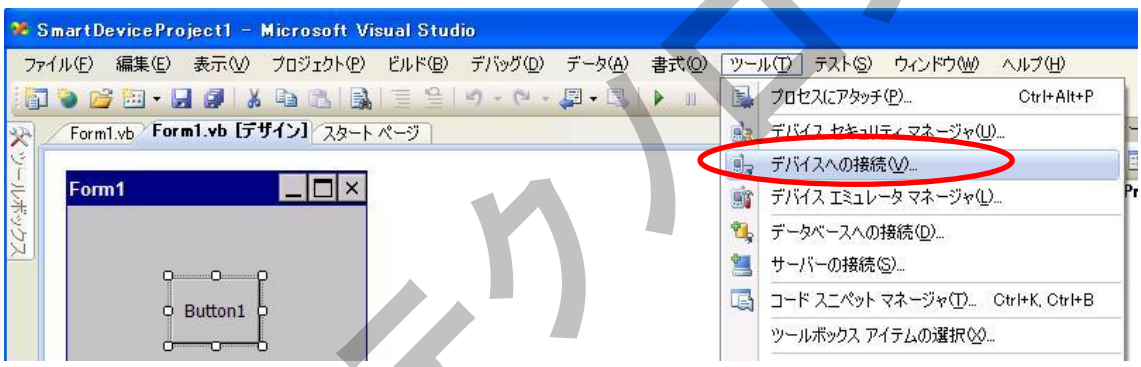


「ActiveSync スタートアップ プロバイダ」を選択します。「構成」ボタンを押します。



このような画面を確認して、「OK」ボタンを押します。VS2008 のメインメニューに戻ります。

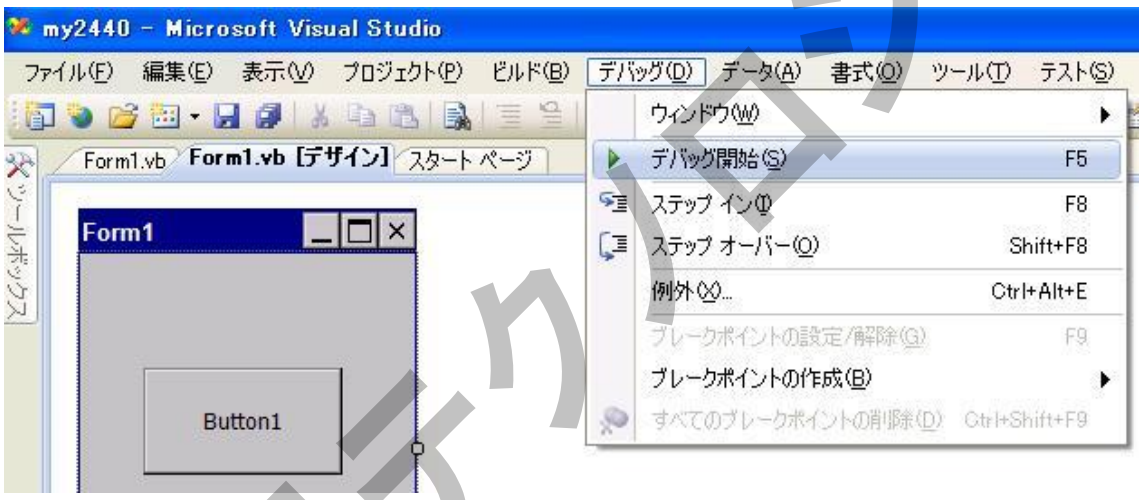
「ツール」→「デバイスへの接続」を選択します。



「Windows CE デバイス」を選択します。「接続」ボタンを押します。

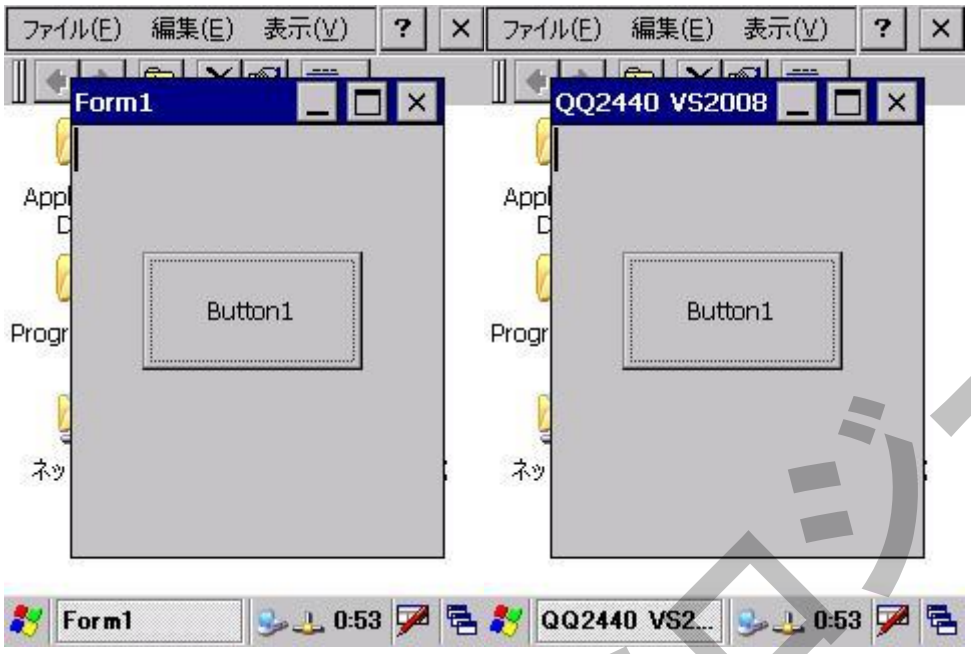
接続中です。

接続成功すれば、「デバッグ」→「デバッグ開始」を選択します。

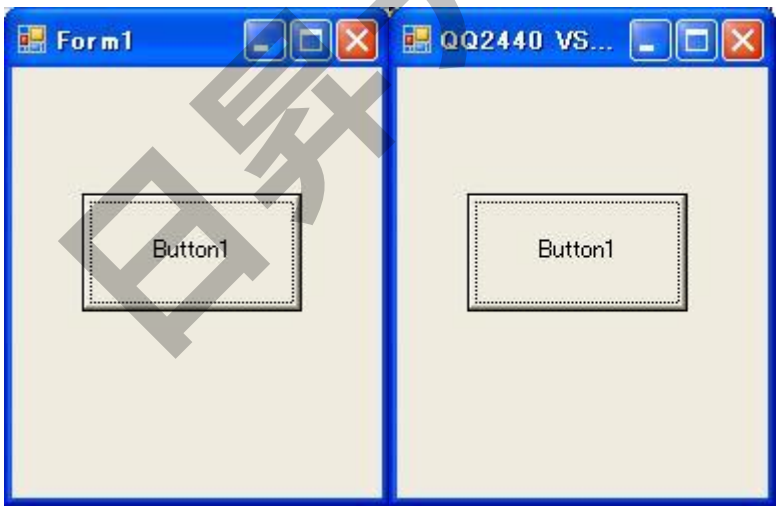


「Windows CE デバイス」を選択します。「配置」ボタンを押して、プログラムを ARM9 ボードにロードします。

プログラムを ARM9 ボードで実行させる様子：



※ VS2008 で生成された実行ファイルも直接に ARM9 ボードにコピーして、実行できます。VS2008 を利用すれば、Visual Basic 以外、Visual C#と Visual C++も作れます。生成された同じ実行ファイルも ARM9 で実行できます。



同じ実行ファイルをホスト側で実行する様子。

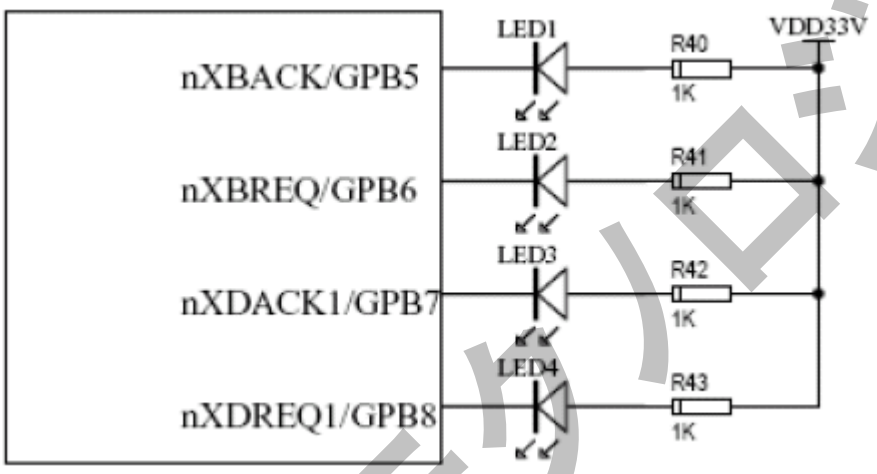
第七章 LED ドライバとテスト例

7.1 ソースの場所

LED のドライバは BSP(mini2440¥Src¥DRIVERS¥LEDdriver)の中にあります。ソースファイルは LEDdriver.cpp です。

7.2 ハードウェアを了解する

ARM9 ボードの LED の回路図：



LED は S3C2440 の GPB ポートを使用しています。

GPB の特性は(S3C2440.pdf ページ 276)

GPB8	Input/output	nXDREQ1
GPB7	Input/output	nXDACK1
GPB6	Input/output	nXBREQ
GPB5	Input/output	nXBACK

GPBCON(S3C2440.pdf ページ 284)：

GPB8	[17:16]	00 = Input 10 = nXDREQ1	01 = Output 11 = Reserved
GPB7	[15:14]	00 = Input 10 = nXDACK1	01 = Output 11 = Reserved
GPB6	[13:12]	00 = Input 10 = nXBREQ	01 = Output 11 = reserved
GPB5	[11:10]	00 = Input 10 = nXBACK	01 = Output 11 = reserved

GPBDAT(S3C2440.pdf ページ 284)：

GPBDAT	Bit	Description
GPB[10:0]	[10:0]	When the port is configured as input port, the corresponding bit is the pin state. When the port is configured as output port, the pin state is the same as the corresponding bit. When the port is configured as functional pin, the undefined value will be read.

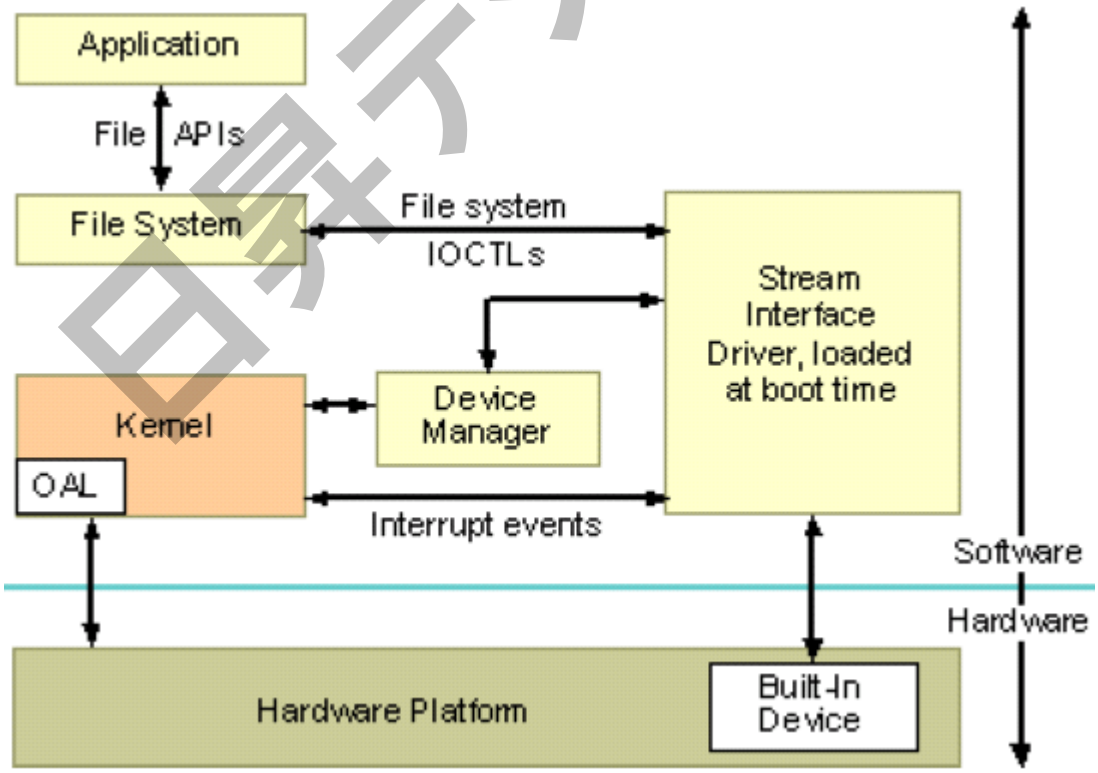
LED 点灯のステップ：

(1) GPBCON の設定、GPB を出力する。

(2) LED 点灯すれば、GPBDAT の対応されたビットを 0 にする。

(3) LED 消灯すれば、GPBDAT の対応されたビットを 1 にする。

7.3 WinCE ドライバの原理



ドライバが必要な関数(PB5 のドキュメントから)

Programming element	Description
XXX_Close (Device Manager)	This function is required to access the device with CreateFile. If you implement **XXX_Close**, you must implement **XXX_Open**.
XXX_Deinit (Device Manager)	This function is required by drivers loaded by ActivateDeviceEx, ActivateDevice, or RegisterDevice.

XXX_Init (Device Manager)	This function initializes a device. It is called by Device Manager. This function is required by drivers loaded by **ActivateDeviceEx**, **ActivateDevice**, or **RegisterDevice**.
XXX_IOControl (Device Manager)	This function sends a command to a device. This function might or might not be required, depending on the device capabilities that the driver exposes. This function requires an implementation of **XXX_Open** and **XXX_Close**.
XXX_Open (Device Manager)	This function opens a device for reading, writing, or both. An application indirectly invokes this function when it calls **CreateFile** to obtain a handle to a device. This function is required to access the device with **CreateFile**.
XXX_PowerDown (Device Manager)	Optional. This function ends power to the device. It is useful only with devices that can be shut off under software control.

XXX_PowerUp (Device Manager)	Optional. This function restores power to a device.
XXX_Read (Device Manager)	This function reads data from the device identified by the open context. This function might or might not be required, depending on the device capabilities that the driver exposes. This function requires an implementation of **XXX_Open** and **XXX_Close**.
XXX_Seek (Device Manager)	This function moves the data pointer in the device. This function might or might not be required, depending on the device capabilities that the driver exposes. This function requires an implementation of **XXX_Open** and **XXX_Close**.
XXX_Write (Device Manager)	This function writes data to the device. This function might or might not be required, depending on the device capabilities that the driver exposes. This function requires an implementation of **XXX_Open** and **XXX_Close**.

7.4 ドライバを BSP に添加する

(1) mini2440¥Src¥DRIVERS フォルダで LEDdriver フォルダを作ります。dirs ファイルにフォルダ LEDdriver を入れます。
(2) フォルダ LEDdriver で makefile ファイルを作ります。
(3) フォルダ LEDdriver で LEDDriver.def ファイルを作ります。
(4) mini2440¥FILES¥platform.bib ファイルを編集します。
(5) mini2440¥FILES¥platform.reg ファイルを編集します。
具体的にどうやって直すか、BSP のソースコードをご覧ください。

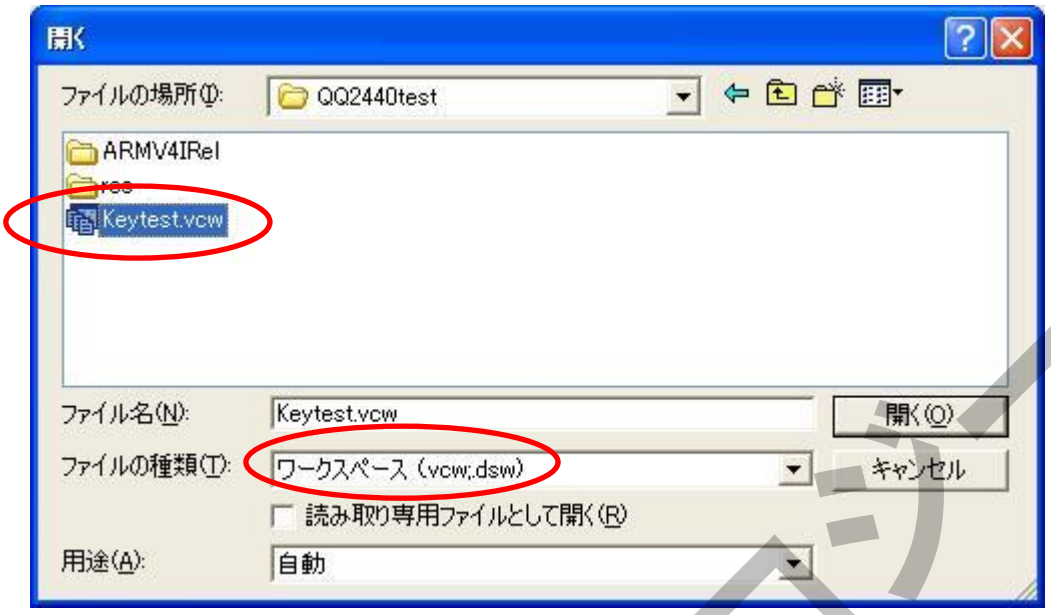
BSP に添加完了すれば、WinCE を再構築してください。

7.5 LED をテストする

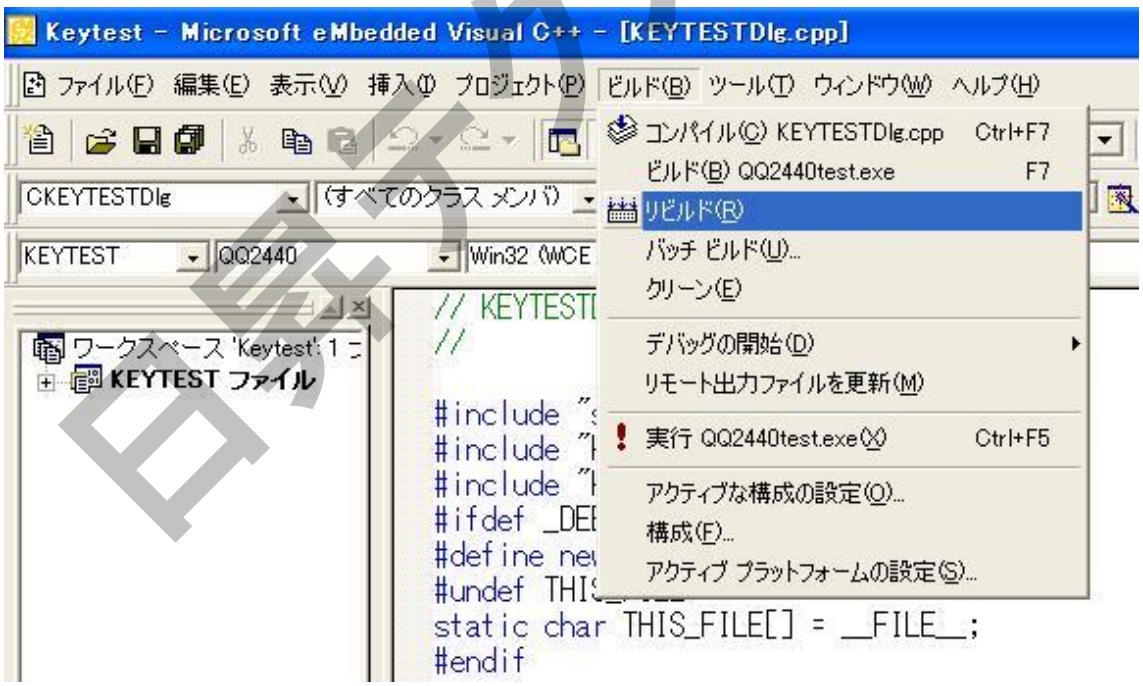
QQ2440test.rar は LED テスト用プロジェクトです。解凍してください。
eMbedded Visual C++ 4.0 を起動します。「ファイル」→「開く」を選択します。


Microsoft eMbedded Visual C++
ファイル(F) 編集(E) 表示(V) 挿入(I) プロジェクト(P) ビルド(B) ツール(T)
新規作成(N)... Ctrl+N
開く(O)... Ctrl+O
閉じる(C)
ワークスペースを開く(W)...
ワークスペースの保存(V)
ワークスペースを閉じる(K)

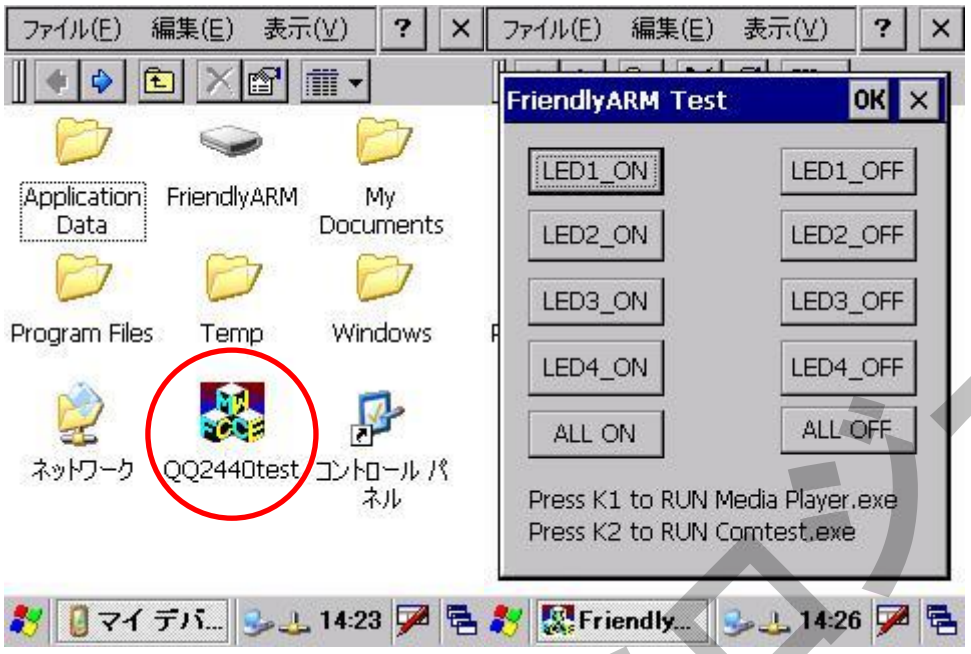
解凍されたフォルダでワークスペースを開きます。



「ビルド」→「リビルド」を選択します。ビルド完了すると、自動的に ARM9 ボードにロードします。（先ずホストと ARM9 の通信を確認してください。）



ARM9 で QQ2440test が見えます。実行してみましょう。



7.6 プログラムを WinCE のカーネルに組み込む

(1) 生成された実行ファイル QQ2440test.exe を mini2440¥FILES にコピーします。

(2) QQ2440test.lnk というファイルを作ります。このファイルはテキストファイルです。
内容は：**23#¥Windows¥QQ2440test.exe**
23 は「#」以降の文字の個数です。
このファイルも mini2440¥FILES にコピーします。

(3) mini2440¥FILES ¥Platform.bib ファイルを編集して、下の内容を入ります。
QQ2440test.exe $(_FLATRELEASEDIR)¥QQtest.exe NK U
QQ2440test.lnk $(_FLATRELEASEDIR)¥QQtest.lnk NK U

(4) mini2440¥FILES ¥Platform.dat ファイルを編集して、下の内容を入ります。
Directory("¥windows¥デスクトップ"):-File("QQ2440 テスト.lnk","¥windows ¥QQ2440test.lnk")

BSP に添加完了すれば、WinCE を再構築してください。

※ これらを BSP に添加済みました。ご覧ください。