

NXP LPC32XX WinCE 6.0 manual

Table of contents

| | | |
|----------|---|-----------|
| 1 | Introduction..... | 3 |
| 1.1 | <i>Copyrights and limitations</i> | 3 |
| 1.2 | <i>Hardware prerequisites.....</i> | 3 |
| 1.2.1 | Phytec board with the LPC3250 MCU | 3 |
| 1.2.2 | Network switch and Ethernet cables..... | 3 |
| 1.2.3 | RS232 serial cable | 3 |
| 1.3 | <i>Software prerequisites to using this BSP.....</i> | 3 |
| 1.3.1 | Platform Builder for WinCE 6.0..... | 3 |
| 1.3.2 | Microsoft Visual Studio..... | 3 |
| 1.3.3 | Terminal program | 4 |
| 2 | BSP information..... | 4 |
| 2.1 | <i>EBOOT.....</i> | 4 |
| 2.2 | <i>WinCE 6.0</i> | 4 |
| 2.2.1 | OEM Adaption Layer (OAL) | 4 |
| 2.2.2 | WinCE drivers | 5 |
| 2.2.3 | KITL..... | 12 |
| 3 | Installing the BSP..... | 13 |
| 4 | Application example build..... | 13 |
| 4.1 | <i>Create a WinCE project, select BSP and components.....</i> | 13 |
| 4.2 | <i>Add BSP items from the catalog.....</i> | 20 |
| 4.3 | <i>Building the WinCE image</i> | 26 |
| 4.4 | <i>Deploying the WinCE bootloader and image.....</i> | 27 |
| 4.4.1 | Configuring SIL to boot the WinCE Ethernet bootloader..... | 27 |
| 4.4.2 | Configuring the WinCE Ethernet bootloader | 29 |
| 4.4.3 | Setting up Platform Builder download and KITL transports..... | 31 |
| 5 | Deployment options | 34 |
| 5.1 | <i>Setting up the Ethernet bootloader to boot from an SD card.....</i> | 34 |
| 5.2 | <i>Setting up the Ethernet bootloader to boot from an NAND FLASH.....</i> | 34 |
| 5.3 | <i>Booting the WinCE image from an SD card.....</i> | 35 |
| 5.4 | <i>Booting the WinCE image from NAND FLASH</i> | 35 |
| 6 | Miscellaneous | 37 |
| 6.1 | <i>Issues</i> | 37 |

1 Introduction

This document explains how to install the NXP LPC32XX WinCE 6.0 Board Support Package (BSP) and how to perform a build with the BSP. This BSP has been ported for the Phytex PHY3250 board, but can easily be re-reported to other boards and products with minor changes.

1.1 Copyrights and limitations

This BSP is provided free of charge and with no support from NXP for development of WinCE products using the LPC32XX Microcontrollers (MCUs).

1.2 Hardware prerequisites

The following hardware is required to deploy, debug, and run the executable WinCE 6.0 images.

1.2.1 Phytex board with the LPC3250 MCU

The WinCE 6.0 BSP has been ported to this specific board. The PHY3250 board includes the Stage 1 Loader (S1L), which is used to initialize the board and kickstart the WinCE 6.0 bootloader (EBOOT).

1.2.2 Network switch and Ethernet cables

A network switch and Ethernet cables are required to download the WinCE 6.0 image to the board and for connector via KITL and the kernel debugger.

1.2.3 RS232 serial cable

A serial cable between a PC and the PHY3250 board is required to interface with S1L and to view WinCE debug messages.

1.3 Software prerequisites to using this BSP

Before using this BSP to generate WinCE images, several tools must be installed on the development system.

1.3.1 Platform Builder for WinCE 6.0

This package contains the tools, images, and software to build the WinCE 6.0 operating system. An evaluation version can be downloaded from Microsoft's website.

1.3.2 Microsoft Visual Studio

Platform Builder requires Visual Studio to build WinCE 6.0 images. An evaluation version can be downloaded from Microsoft's website.

1.3.3 Terminal program

A terminal program is required to interface with S1L and to view WinCE 6.0 debug messages when KITL isn't used. Although any terminal program can be used, TeraTerm is recommended. TeraTerm is free and can be downloaded from various websites.

2 BSP information

This BSP provides basic drivers for most of the LPC32XX peripherals and the Phytex PHY3250 interfaces. Although the drivers may work in a specific application, they may require additional work or modifications in a production system.

2.1 EBOOT

EBOOT is the WinCE bootloader provide with Platform Builder. EBOOT provides support for loading and starting the WinCE image from multiple sources including Ethernet or flash media. It also provides an interactive method for changing boot preferences, boot timeouts, and WinCE boot support features.

The Phytex port of EBOOT relies on the Stage 1 Loader (S1L) to pre-initialize the board and memory prior to executing. See the phy32xx_bl.pdf file included with the LPC32XX BSP for more information on S1L. Once S1L has initialized the board, it will load and start EBOOT from one of the supported S1L boot sources. Although EBOOT relies on S1L for system pre-initialization, the pre-initialization code is available for free from NXP's website and can be easily added to the startup code of the EBOOT source.

The PHY3250 version of EBOOT supports the following features:

- Persistent configuration of EBOOT and kernel parameters
- Kernel image boot from NAND FLASH, SD/MMC cards, or Ethernet
- Ethernet network configuration
- Kernel parameters configuration (KITL, VMINI)

2.2 WinCE 6.0

This section describes the kernel and driver support included with the PHY3250 WinCE 6.0 BSP.

2.2.1 OEM Adaption Layer (OAL)

The OAL provides the basic functionality necessary to start up the WinCE kernel. The OAL component notes are listed in this section.

2.2.1.1 OAL drivers

2.2.1.1.1 Interrupt driver

The interrupt driver handles system wide interrupt events and hardware IRQ to SYSINTR mappings. Individual DMA channel interrupts are assigned to SYSINTR values making DMA channel interrupt management easier.

Several systems interrupts such as the timer, profiler, and USB host are statically mapped, but most interrupts are dynamically mapped as the drivers acquire them.

2.2.1.1.2 Real time clock (RTC)

A driver to handle the on-board RTC is provided as part of the OAL. The RTC driver uses the LPC32XX RTC for “seconds” storage. Because the resolution of the RTC is only 1 second, timer 1 is used to provide a high resolution system time function for WinCE.

The use of timer 1 allows the time functions to support millisecond resolution.

2.2.1.1.3 Timer tick function

Timer 3 is used to provide the system tick interrupt used by WinCE.

2.2.1.2 Other OAL support functions

2.2.1.2.1 Profiler

An application call profiler is available as part of the OAL. To use the profiler, applications must be instrumented with profiler code. The profiler uses system timer 2.

2.2.1.2.2 DMA channel allocation and control

A DMA channel allocation driver is provided in the OAL for drivers to request and enable specific DMA channels. This driver doesn’t actually handle the DMA transfer or channel specific functions, only the allocation and de-allocation of individual DMA channels. This driver is used by other drivers via the KernelIOControl(...) system call.

2.2.1.2.3 System clock control and query

A clock query and control driver handles system clock speed query functions so driver timing can automatically adapt to the current system speed. Individual peripheral clocks can be enabled and disabled through these functions. Clock query and control functions are accessed via the KernelIOControl(...) system call.

2.2.1.2.4 System suspend

The OEM power off and on functions are not fully implemented. I2C power support is provided in the

2.2.2 WinCE drivers

The WinCE drivers provide most of the system functionality needed to control the LPC32XX and PHY3250 hardware.

2.2.2.1 Synchronous Serial Port (SSP) drivers

Two SSP drivers exist for SSP interfaces access on SSP0: and SSP1:.In the Phytex board implementation, SSP0: connects to the serial EEPROM and SSP1: is not connected. WinCE applications can access devices on the SSP interfaces directly via these drivers.

Peripherals on the SSP buses can be accessed directly from WinCE applications using the SSP0 and SSP1 filenames.

2.2.2.2 Driver notes

Driver dependencies:

No other driver dependencies

Power management:

Full power management is provided with this driver

Registry configuration options:

No configurable options

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_SSP0, BSP_LPC32XX_SSP1

Notes:

The drivers can be instantiated in WinCE by simply enabling the drivers in the catalog. As SSP1 is not connected in hardware, the source code for SSP1 must be completed to support the SSP1 configuration if it is connected in a design. Look for the “TBD”s in the driver – these are the areas that need modification for a design using SSP1.

2.2.2.3 I2C drivers

Two I2C drivers exist for I2C interfaces access on I2C1; and I2C2;. Other system drivers may use these drivers for accessing peripherals on the I2C bus (such as the audio driver).

The I2C driver is separated into a separate I2C library and driver interface functions. The library is also used by the USB On-The-Go driver.

Peripherals on the I2C buses can be accessed directly from WinCE applications using the I2C1 and I2C2 filenames.

2.2.2.4 Driver notes

Driver dependencies:

No other driver dependencies

Power management:

Because the I2C driver is used by other drivers, the I2C peripherals are always powered. The Power up/down functions in the driver do not do anything. On system suspend, the system will place the I2C drivers into a low power state. This allows drivers that require use of the I2C peripheral to continue using the driver when the system is powering down.

Registry configuration options:

No configurable options

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_I2C1, BSP_LPC32XX_I2C2

Notes:

The drivers can be instantiated in WinCE by simply enabling the drivers in the catalog. The drivers may require some slight modifications to drive strength settings for a customer design.

2.2.2.5 DMA driver library

A DMA driver library exists for peripherals that need DMA support. The DMA functions provide support for DMA channel allocation and control, memory management, and data transfer via linked list and direct buffer. The DMA driver is used directly by other drivers.

2.2.2.6 Driver notes

Driver dependencies:

No other driver dependencies

Power management:

Not applicable

Registry configuration options:

Not applicable

Source code configuration options:

No configurable options

Catalog IDs: Not applicable

Notes:

This library is used by other drivers when DMA is needed.

2.2.2.7 LCD display driver

The LCD driver provides support for the display with the PHY3250 board. A LCD configuration file stores timing and configuration parameters that can be used with registry settings to select a specific display type.

2.2.2.8 Driver notes

Driver dependencies:

No other driver dependencies

©2006-2007 NXP Semiconductors. All rights reserved.

Power management:

Power management is not fully supported for this driver.

Registry configuration options:

The “LCDPanelType” registry setting allows simple registry selection of predefined panel configurations stored in the LCD configuration file. As of this release, the only support panel type is 0, which corresponds to the QVGA LCD display used on the Phytex board.

The “UseIRAM” registry setting configures the LCD driver to use IRAM for the LCD frame buffer. This allows a small system performance improvement over SDRAM if the LCD frame buffer can fit inside the 256K of internal RAM of the LPC3250.

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_DISPLAY

Notes:

The lcd_panel_types.cpp file contains a pre-defined LCD configuration parameters array that can be defined for specific displays. New panel parameters can easily be added to this array to support new panel and LCD controller configurations. The LCD backlight is enabled and disabled as part of this driver using a GPIO. This GPIO is specific to the Phytex 3250 board and may require modification for a custom design.

2.2.2.9 LCD backlight driver

The LCD backlight driver provides backlight control for the LCD. Currently, only on and off control is provided by the driver via the GPIO signal on the LPC3250.

2.2.2.10 Driver notes

Driver dependencies:

Requires the backlight MDD included with WinCE Platform Builder

Power management:

Basic on/off power management is provided in this driver.

Registry configuration options:

In a system with power management support, backlight timeout values can be adjusted in the registry.

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_BACKLIGHT

Notes:

The GPIO used for the backlight is specific to the Phytec 3250 board and may need to require modification on a custom design.

2.2.2.11 FLASH media driver

Support for FLASH devices through the LPC3250 SLC NAND controller is provided by the FLASH media driver. The FLASH media driver provides support for 1-bit Error correction, bad block marking, and reserved blocks.

SIL, the bootloader on the Phytec board, allows marking of blocks as reserved to prevent WinCE from using them in the FLASH media driver.

2.2.2.12 Driver notes

Driver dependencies:

No other driver dependencies

Power management:

Basic on/off power management is provided in this driver.

Registry configuration options:

If the WinCE system is configured for hive storage, the FLASH media driver will be configured for the hive registry. Options for setting up the hive and FLASH file system are in the registry.

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_FLASH

Notes:

This driver is specifically targeted for the ST FLASH on the Phytec board. Large block FLASH or other vendor's FLASH will require modifications to the driver. Reserved blocks can be marked with the Stage 1 Loader (SIL) to prevent WinCE from accidentally erasing the bootloader.

2.2.2.13 Key scanner driver

Matrix keypad support is provided through the key scanner driver. This driver uses the LPC3250 key scanner peripheral to monitor key states and provide key press information to WinCE.

2.2.2.14 Driver notes

Driver dependencies:

No other driver dependencies

Power management:

Full power management is provided with this driver.

Registry configuration options:

The registry is setup for locale 409. Other localizations will require modifications to the registry to support the key scanner.

Source code configuration options:

The keyscanner driver can easily be configured for any matrix size of 1x1 to 8x8 with an adjustable number of debounce and delay cycles to vary key detect rates.

Catalog IDs: BSP_LPC32XX_KSCAN

Notes:

Although the driver is not board specific, the matrix to key map file is board specific. This file (kb_phy3250_map) is intended to be modified for custom designs.

2.2.2.15 Touch controller driver

The touch controller driver provides a position sensing ability for touches to the LCD touchscreen.

2.2.2.16 Driver notes

Driver dependencies:

No other driver dependencies

Power management:

Full power management is provided with this driver.

Registry configuration options:

No configurable options

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_TOUCH

Notes:

None

2.2.2.17 Audio driver

The wavedev driver supports the I2S peripheral of the LPC3250 when used with the UDA1380 I2S CODEC. The audio driver currently supports output mode only for uncompressed PCM data at varying data rates and data sizes.

The wavedev driver requires use of the I2C1 driver and the DMA driver library.

2.2.2.18 Driver notes

Driver dependencies:

Requires the I2C1 driver. Also requires the DMA library.

Power management:

Partial power management is provided with this driver. Full power management is provided for the I2S peripheral. Although functions are available for powering up and down the CODEC, they are not implemented.

Registry configuration options:

No configurable options

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_WAVEAPI

Notes:

Audio input is not supported in this driver.

2.2.2.19 SD card controller driver

The SD card controller driver provides support for the SD card controller peripheral of the LPC3250 and the SD card slot of the Phytex board.

The SD card controller driver requires use of the DMA driver library.

2.2.2.20 Driver notes

Driver dependencies:

Requires the DMA library.

Power management:

Partial power management is provided with this driver.

Registry configuration options:

Standard WinCE SD card registry options are available.

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_SDCARD

Notes:

The SD card driver currently only supports single sector read and write operations. Multiple sector read operations are converted to a loop of single sector read operations in the driver, while single sector write operations are selected with the “SingleBlockWrites”

©2006-2007 NXP Semiconductors. All rights reserved.

registry setting in the registry options. Do not disable the “SingleBlockWrites” registry setting. For block reads, the following STOP TRANSMISSION command is ignored.

Known issues:

Currently, multiple block read and write commands are not supported.

2.2.2.21 USB host support

USB host support is provided through the OHCI library provided with WinCE. Control of the hub and some USB functions is provided through the OTG ISP1301 peripheral device. Currently, USB function support is not available.

2.2.2.22 Driver notes

Driver dependencies:

Requires the I2C0 driver and the WinCE OHCI USB host library

Also enable the BSP_LPC32XX_USBOTG and BSP_LPC32XX_USBFN catalog items to use host support.

Power management:

Partial power management is provided with this driver.

Registry configuration options:

Standard WinCE USB registry options are available.

Source code configuration options:

No configurable options

Catalog IDs: BSP_LPC32XX_USBD

Notes:

USB host support is supplied through the USB OTG driver. USB support is not yet fully implemented.

Known issues:

Full speed transfers are not operational yet. This will prevent some USB devices from working (such as mass storage class devices), but Human Interface Devices should work.

2.2.3 KITL

A KITL driver provides support for the LPC3250 ethernet peripheral when used with the Ethernet PHY used on the Phytex board. The KITL driver support polling and interrupt modes and is configured through EBOOT.

3 Installing the BSP

To install the LPC3250 WinCE 6.0 BSP, unzip the BSP file into the \WINCE600\PLATFORM directory. After the file is unzipped, the following path should be available:

\WINCE600\PLATFORM\LPC3250

Run the ins_lwce.bat script once in the .\LPC3250 directory to copy common files used in other BSPs to the LPC3250 BSP. *This step is needed because files already provided with WinCE 6.0 Platform Builder are not provided as part of the LPC3250 BSP. The ins_lwce.bat file can be deleted after it has been executed.*

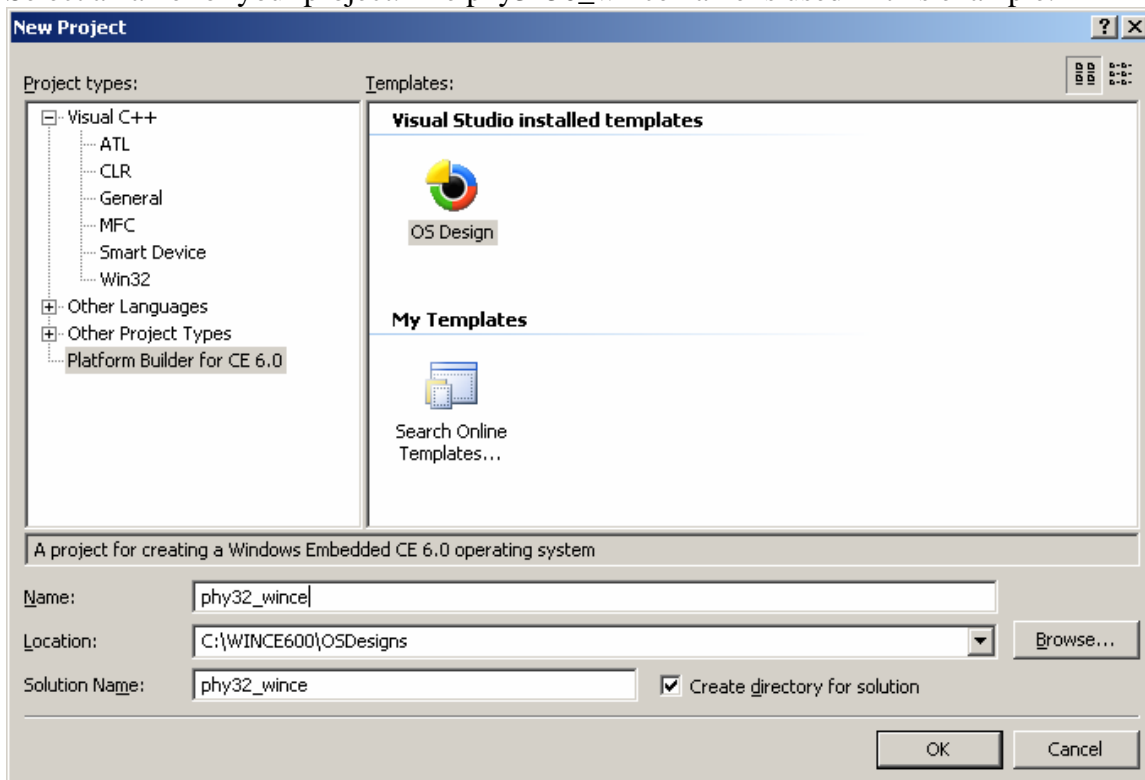
That's it! The LPC3250 BSP has been installed. If Visual Studio is currently open, close it and restart it before using the LPC3250 BSP.

4 Application example build

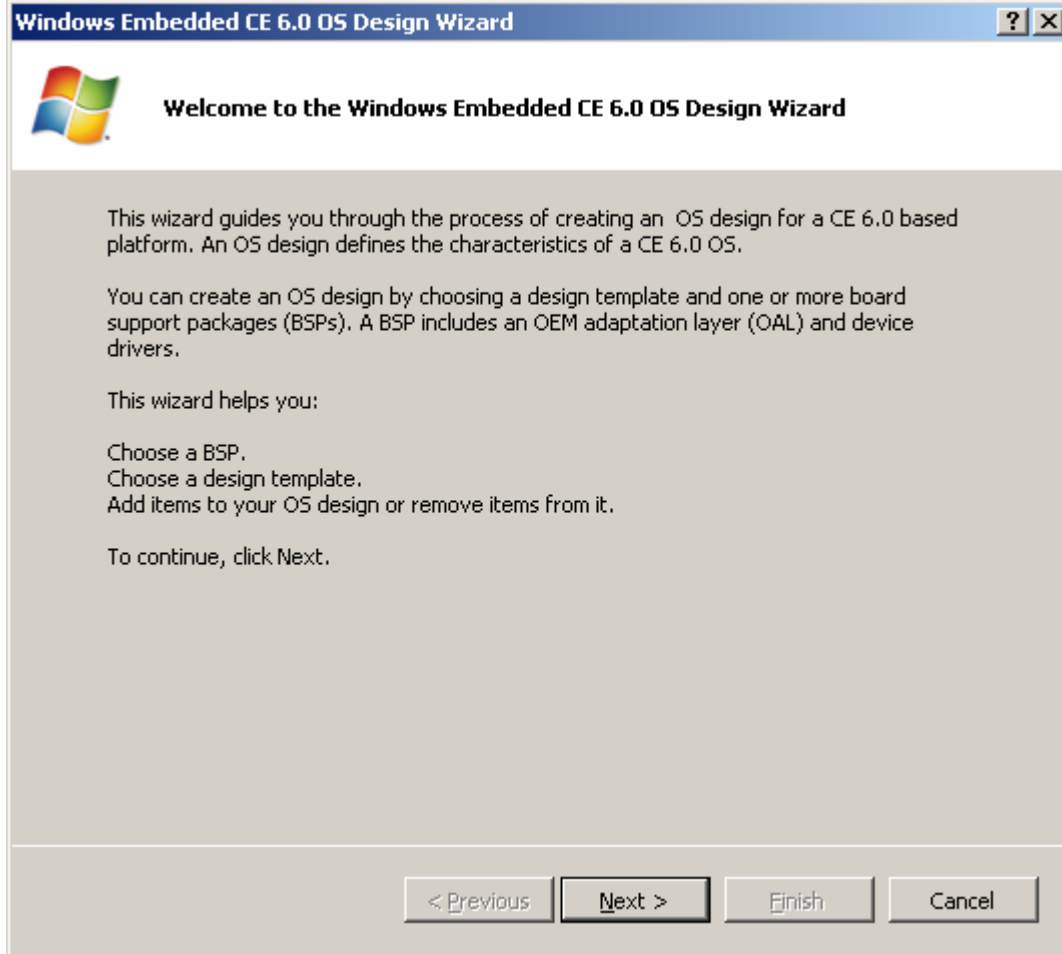
One the BSP has been installed, make sure the Visual Studio and Platform Builder are up to date with the latest patches and fixes. This section will go through a complete build procedure for the "PDA" system using the LPC3250 BSP.

4.1 Create a WinCE project, select BSP and components

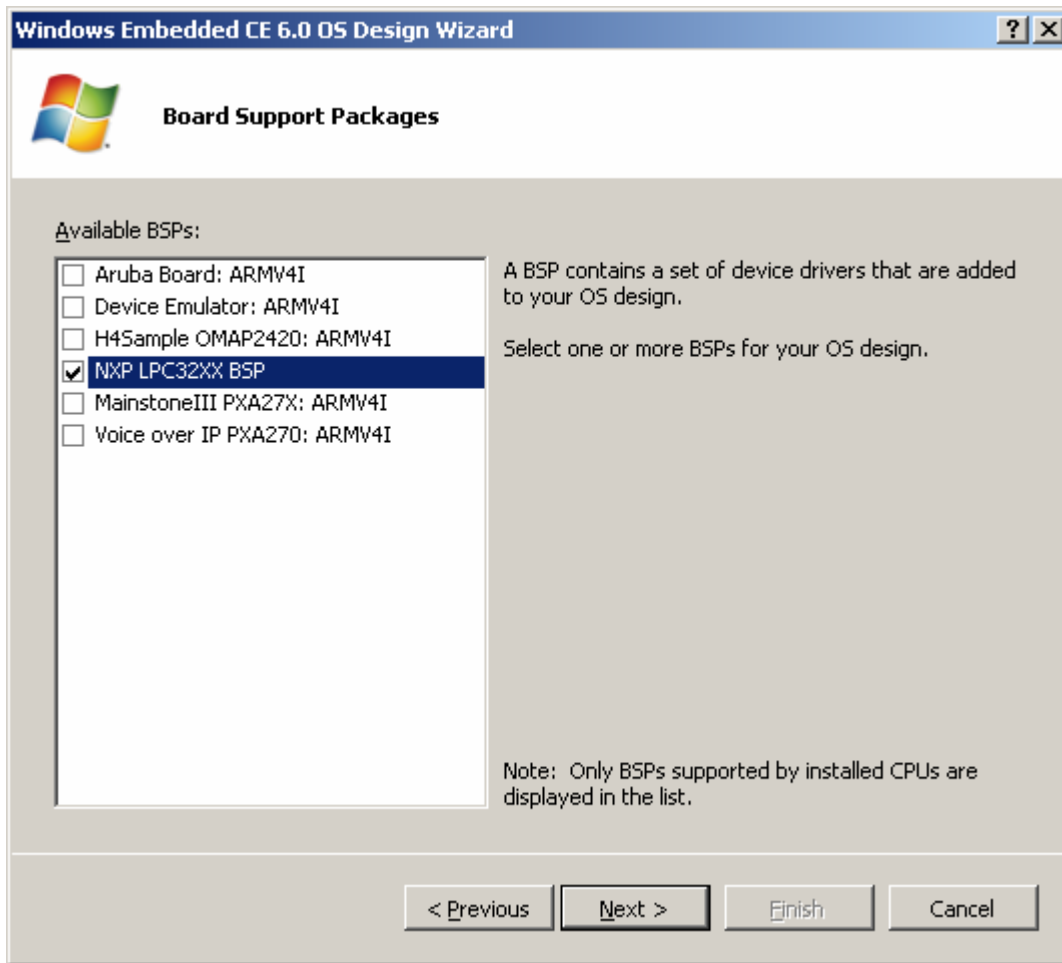
Start Visual Studio and select the FILE menu item and then New->Project. Select the Platform Builder for CE6.0 Project type and the OS design installed template. Select a name for your project. The phy3250_wince name is used in this example.



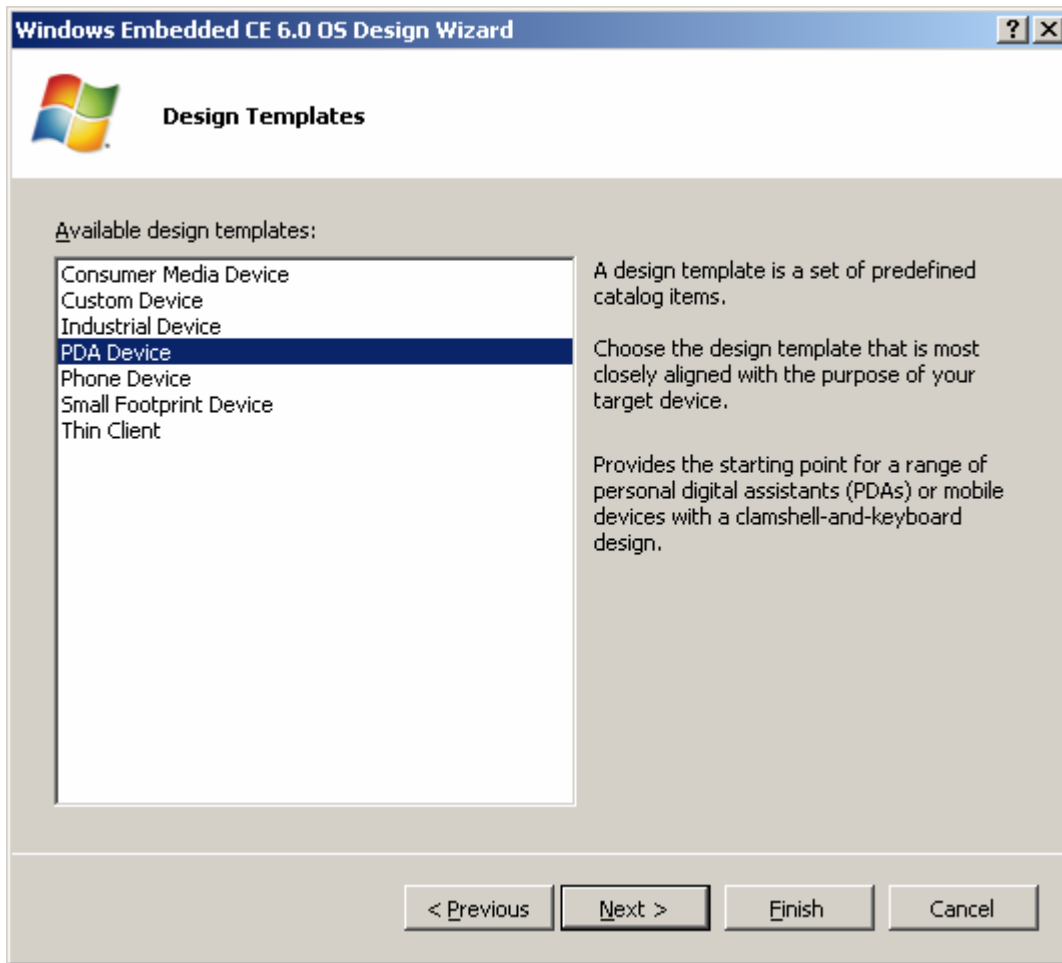
Selecting the OK button will take you to the start of the Design Wizard. Press NEXT.



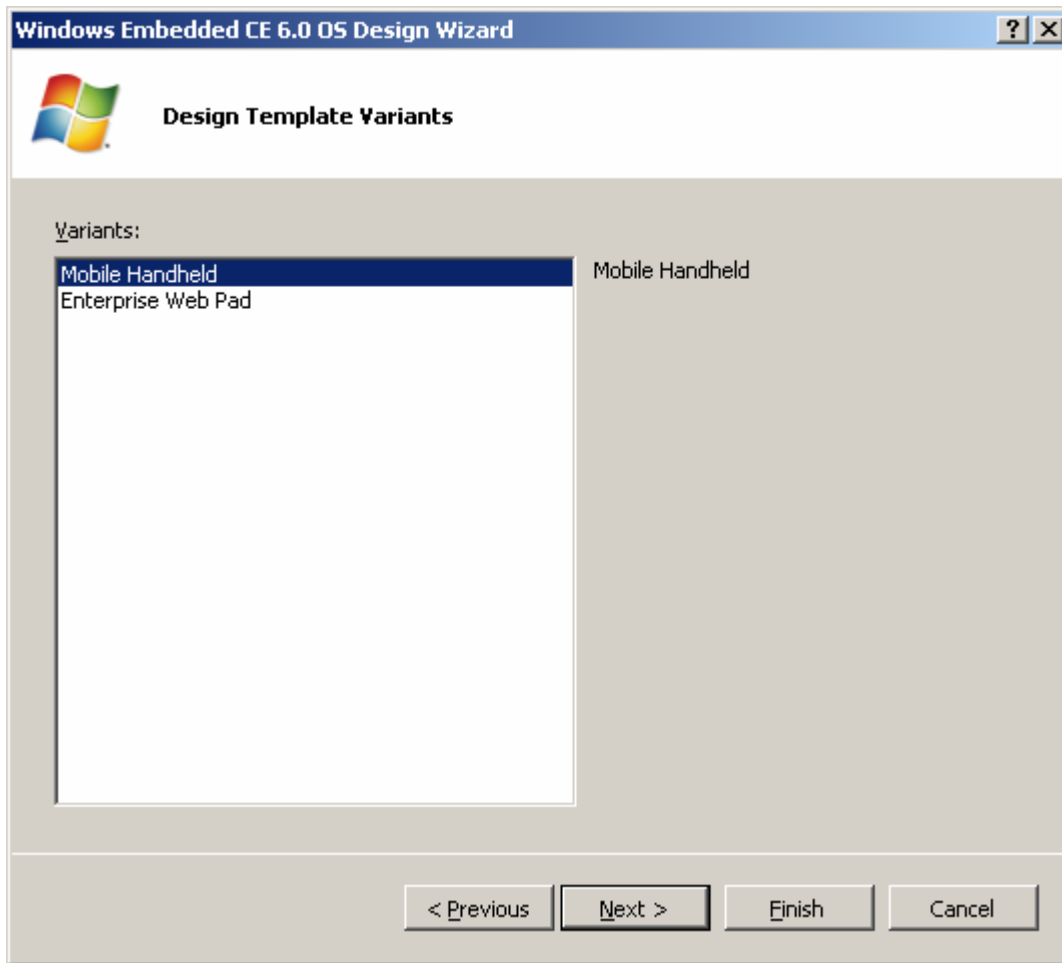
Select the NXP LPC32XX BSP and press NEXT.



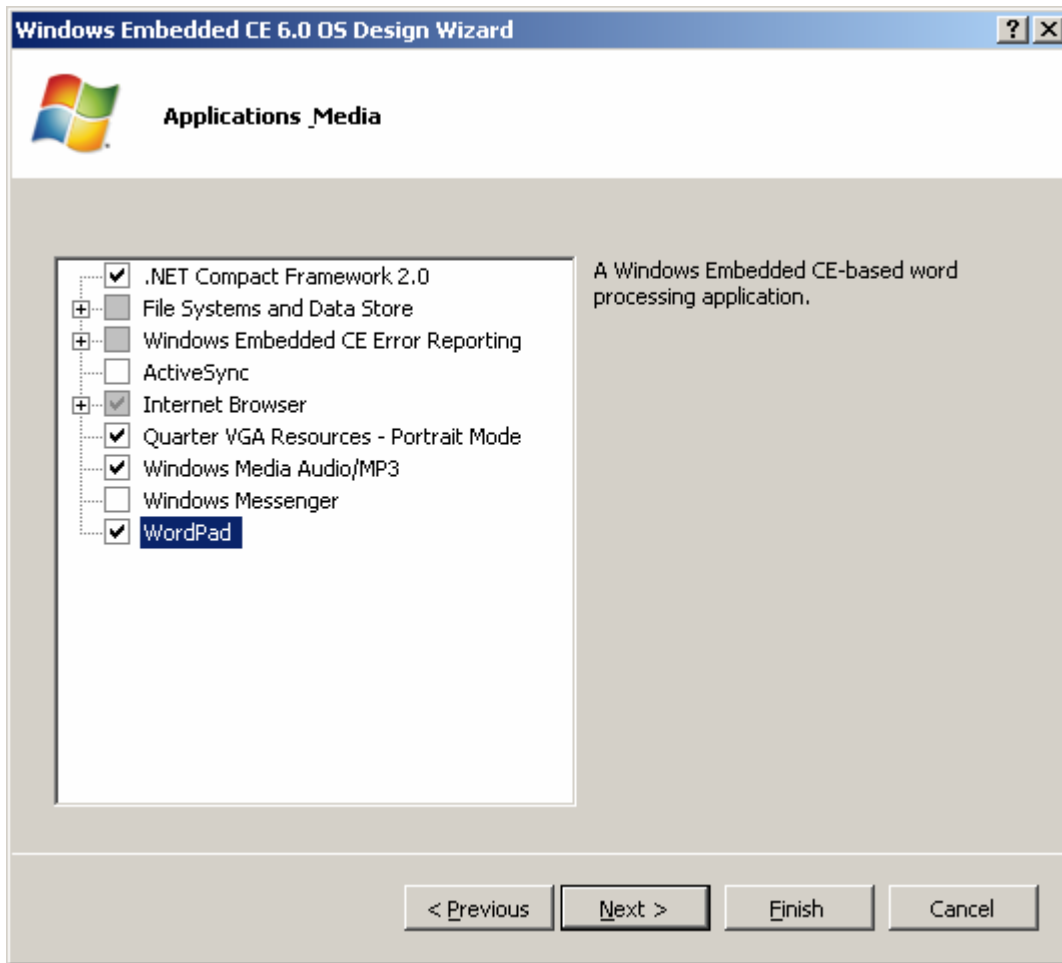
For the design template, select the PDA Device template. Although the default template does require a few components that are not available in the BSP, the build will still work and be usable on the board. Press NEXT.



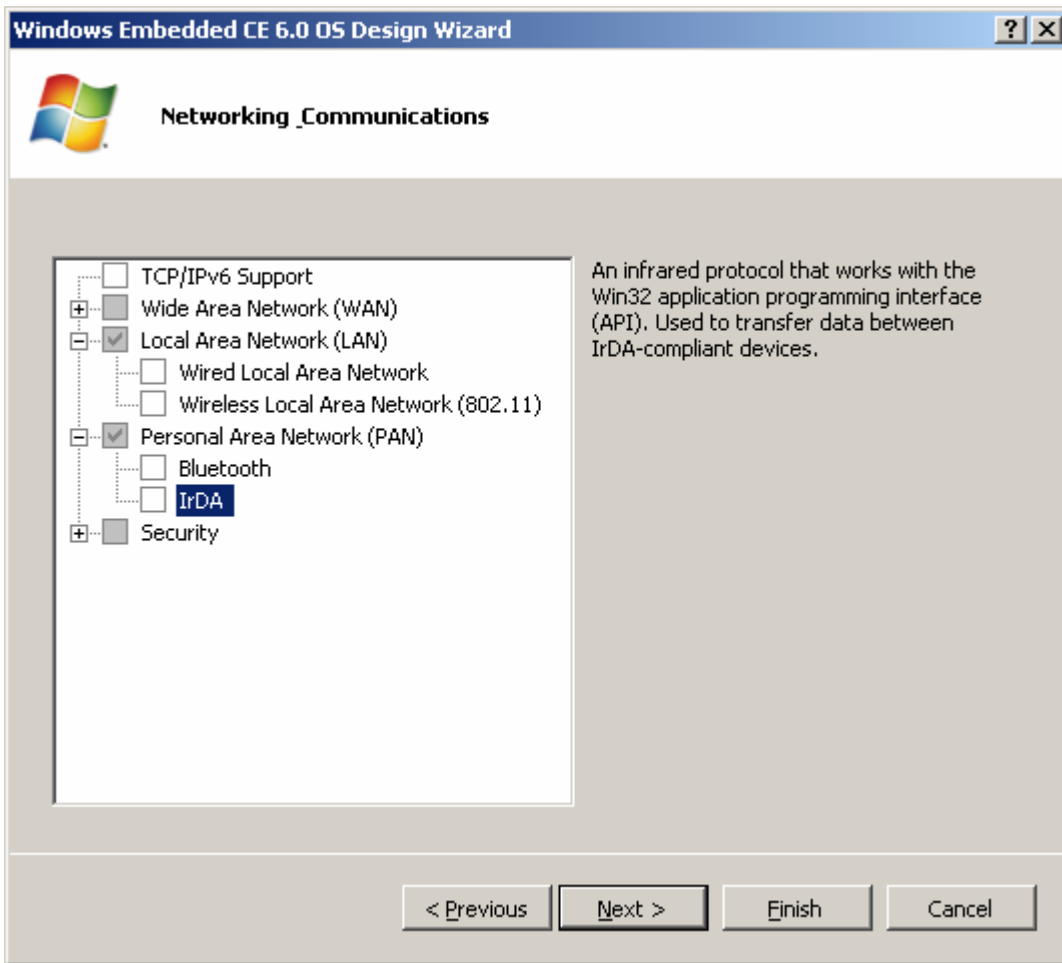
Select the Mobile Handheld variant. Press NEXT.



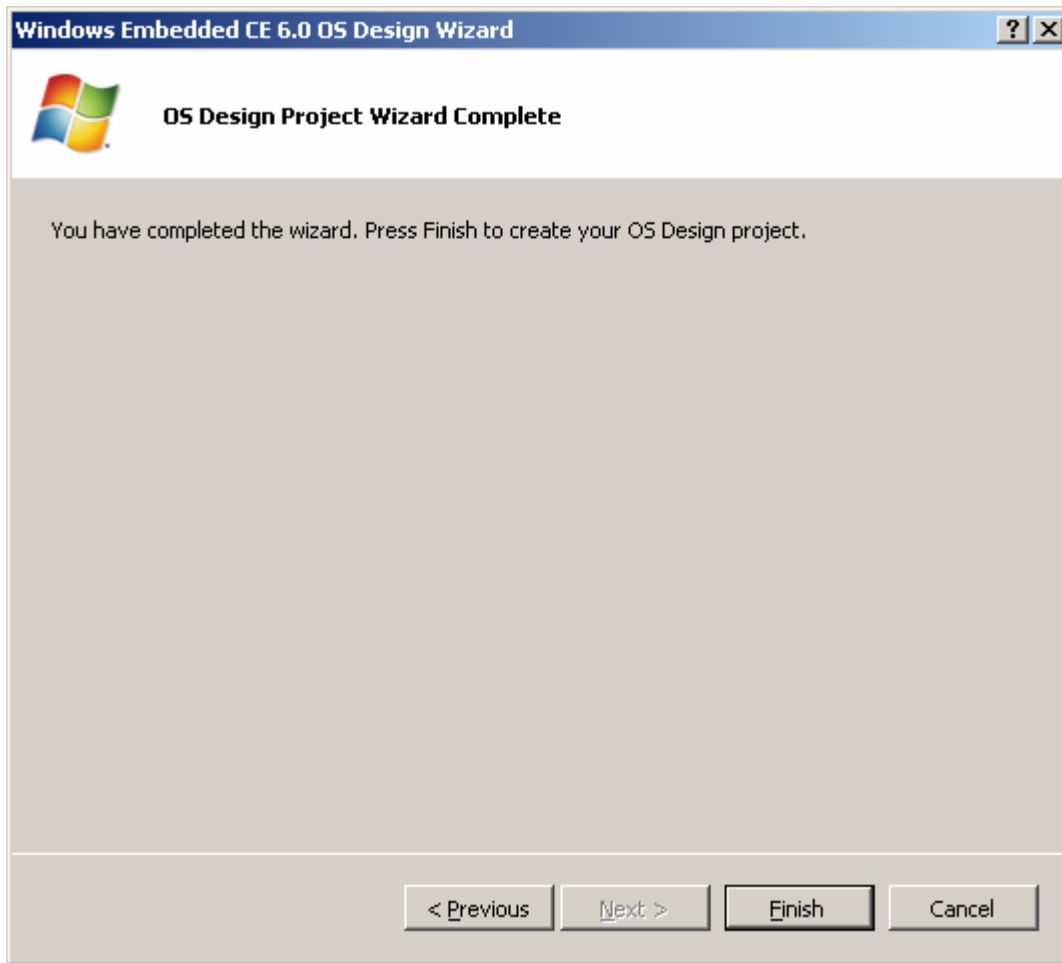
Disable ActiveSync and enable the Windows Media Audio/MP3 and Wordpad options.
Press NEXT.



Disable the network, Bluetooth, and IrDA options. Press NEXT.

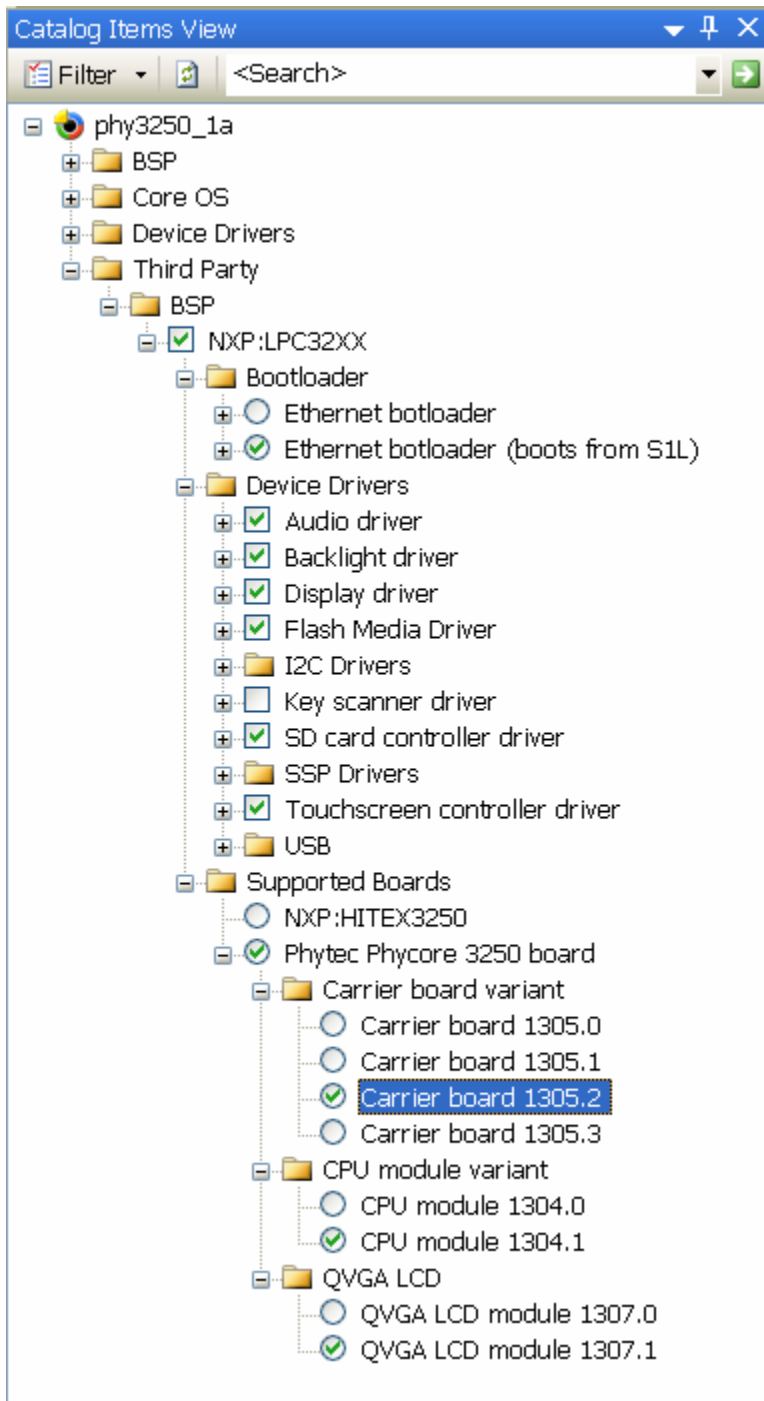


Press FINISH to complete the wizard.

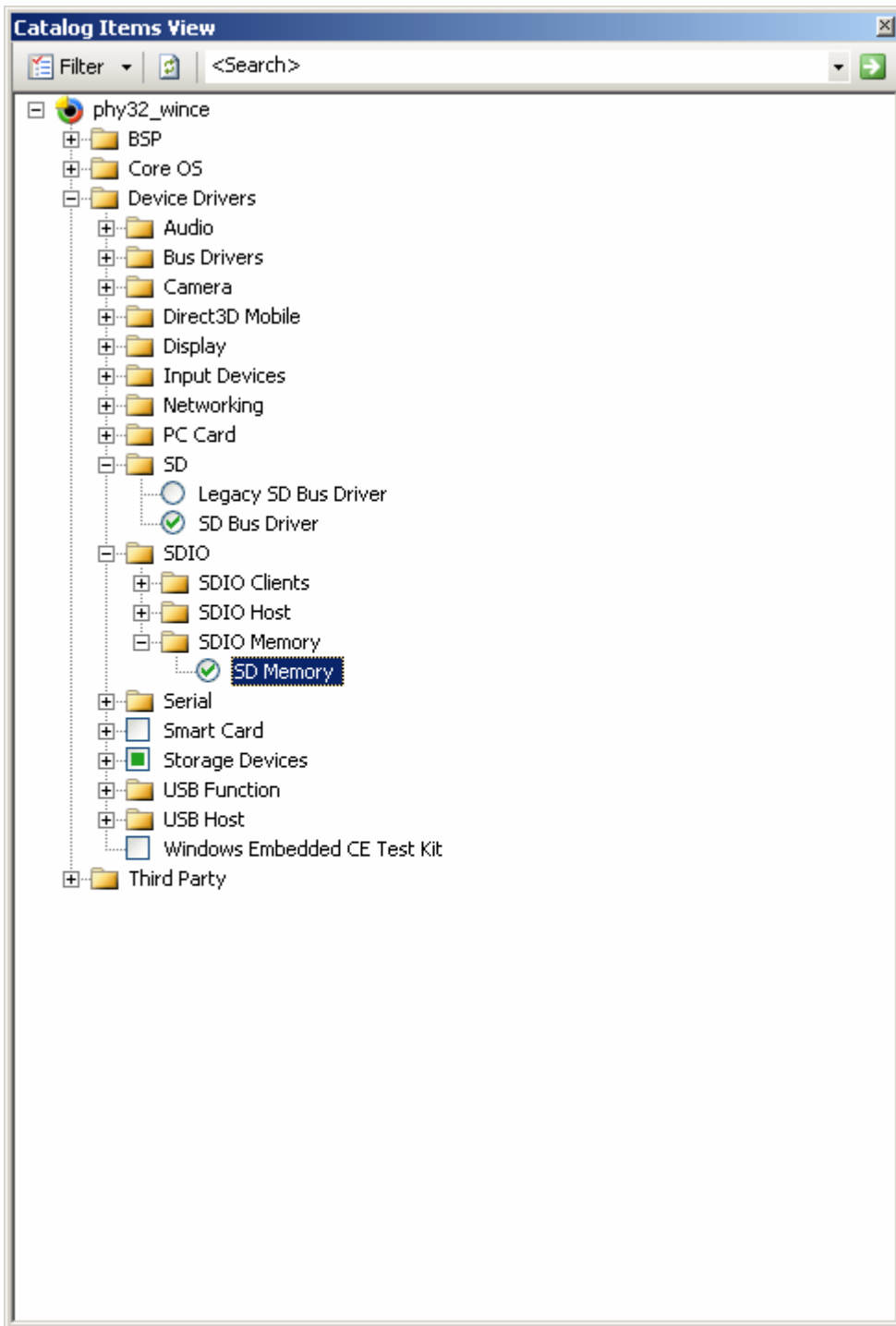


4.2 Add BSP items from the catalog

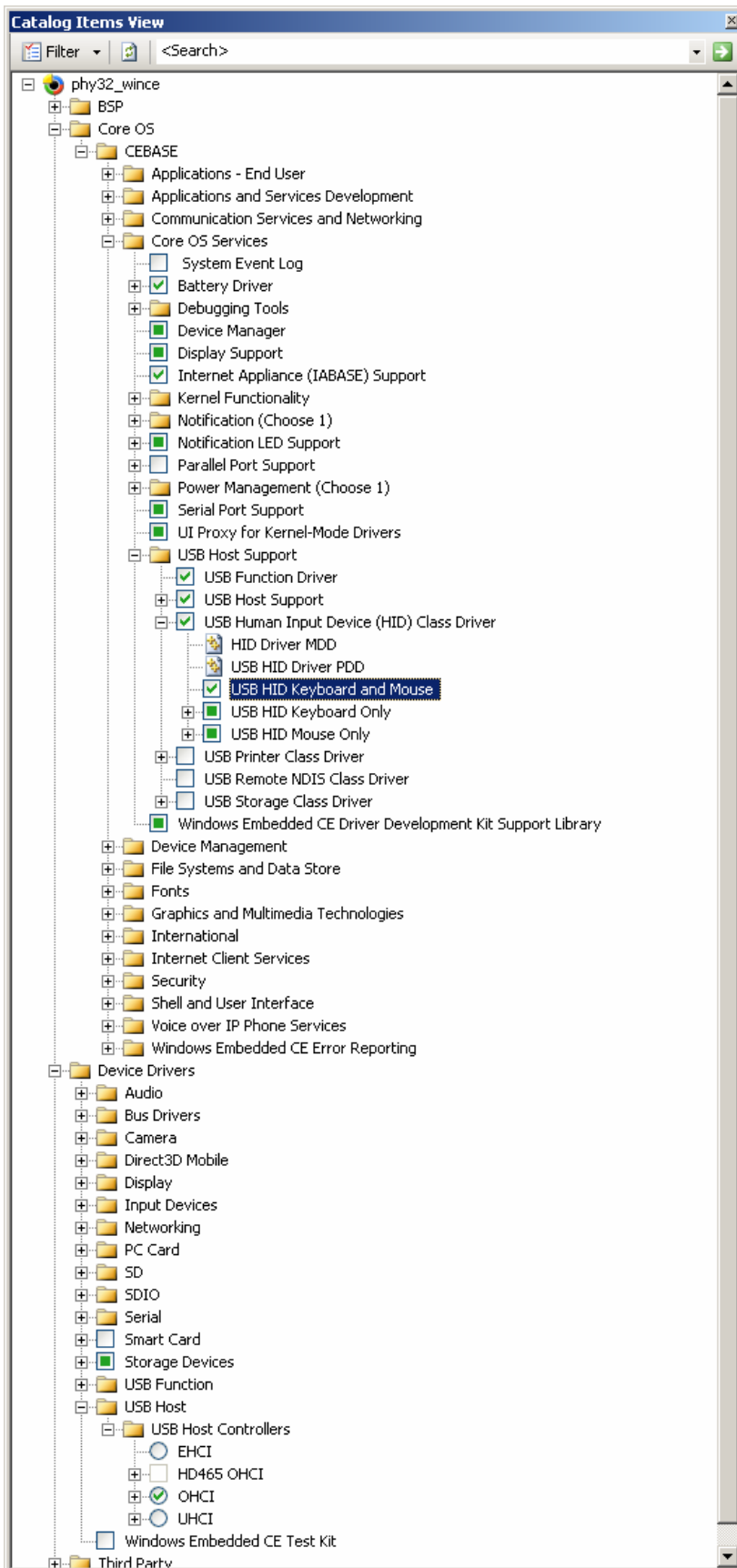
After the wizard has complete, you can further refine selections for the WinCE image from the catalog and select which LPC3250 drivers you want to use in the image. Note that a section for selecting the board versions – this section must be correctly setup prior to building!



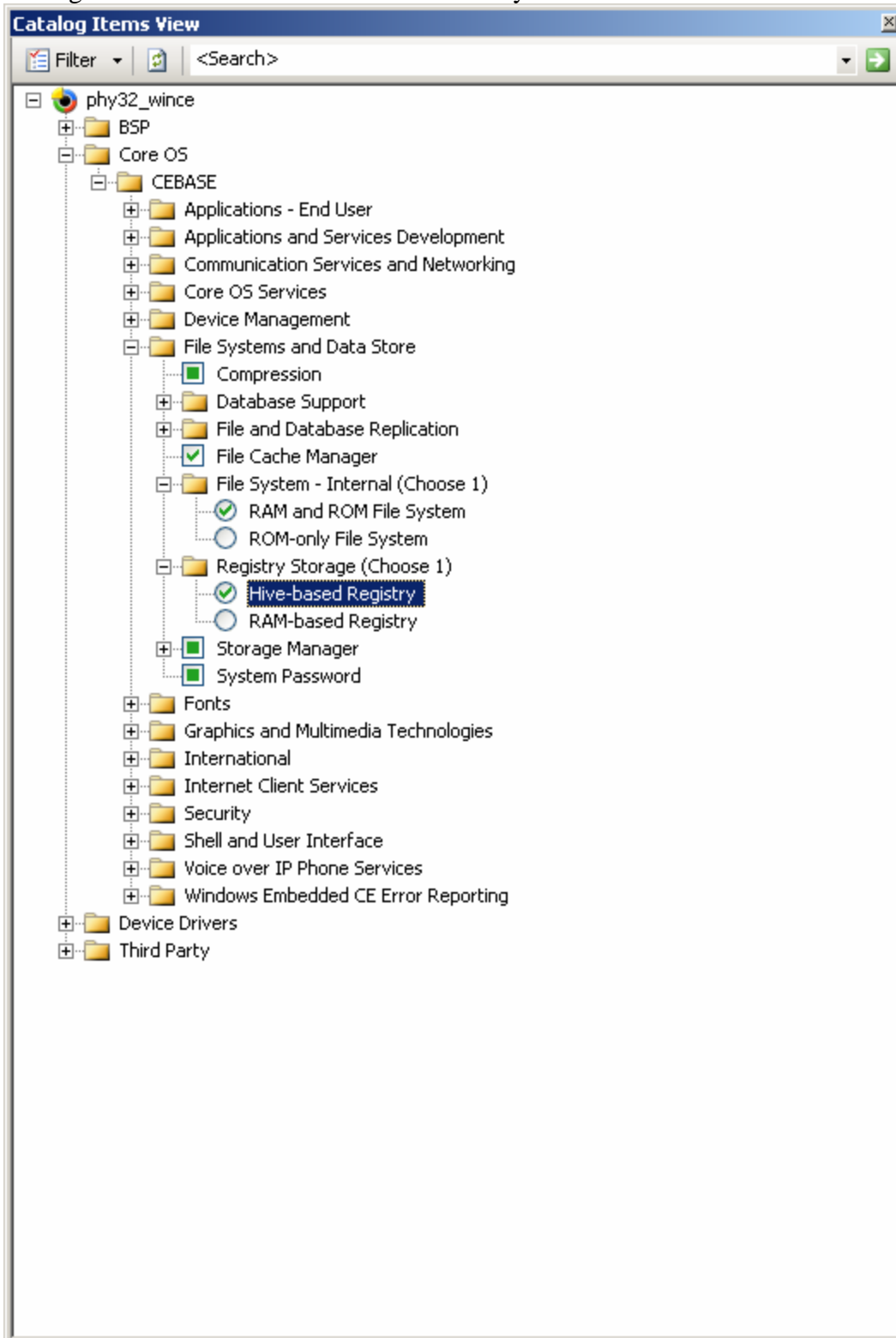
To use the LPC3250 SD card controller, the WinCE SD bus driver and the SD memory support must also be selected from the catalog.



To use the USB host, OHCI support and USB support must be selected from the catalog. Keyboard and mouse support are selected for the USB host.

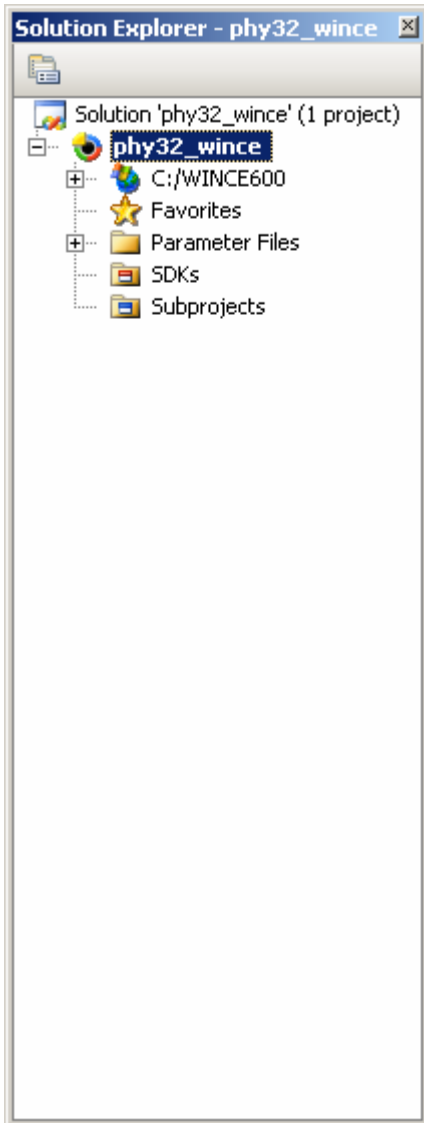


Configure support for the hive registry by enabling the Hive-based registry button in the catalog. Also enable the RAM and ROM file system button.

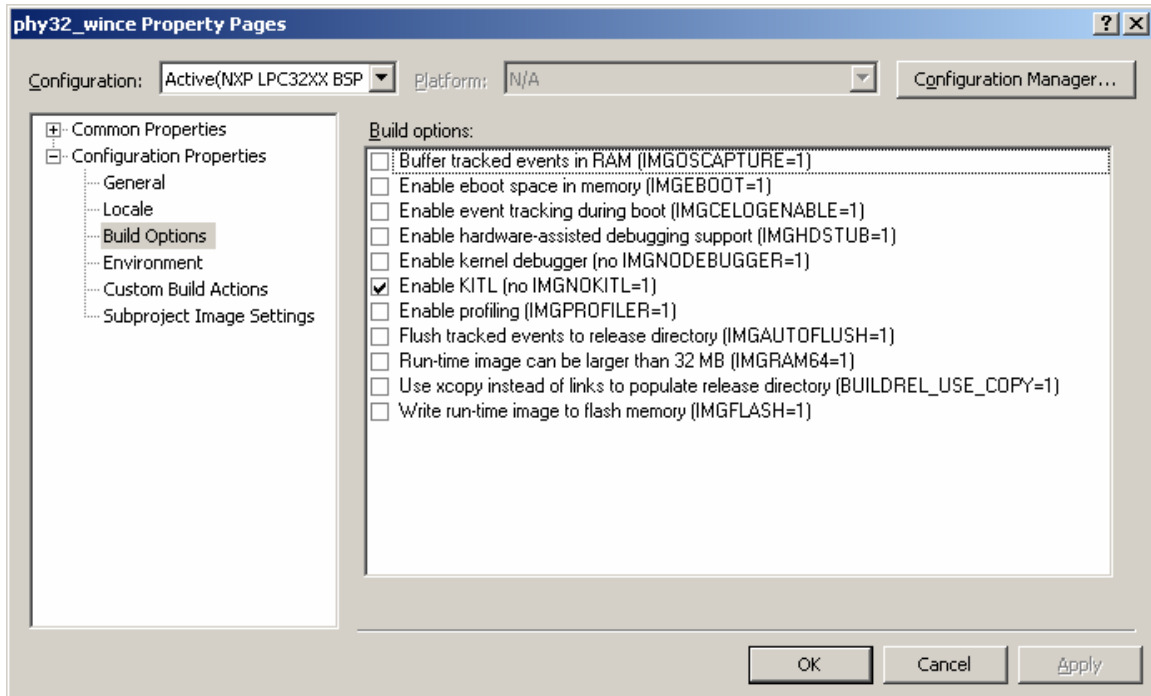


At this point, the necessary catalog items to build a feature-rich WinCE image and WinCE bootloader have been configured. Before starting the WinCE image build process, some build options need to be setup.

In the Solution Explorer Window, right click on the project name (phy3250_wince) and select Properties. If the Solution Explorer window isn't visible, it can be enabled from the Visual Studio View menu.



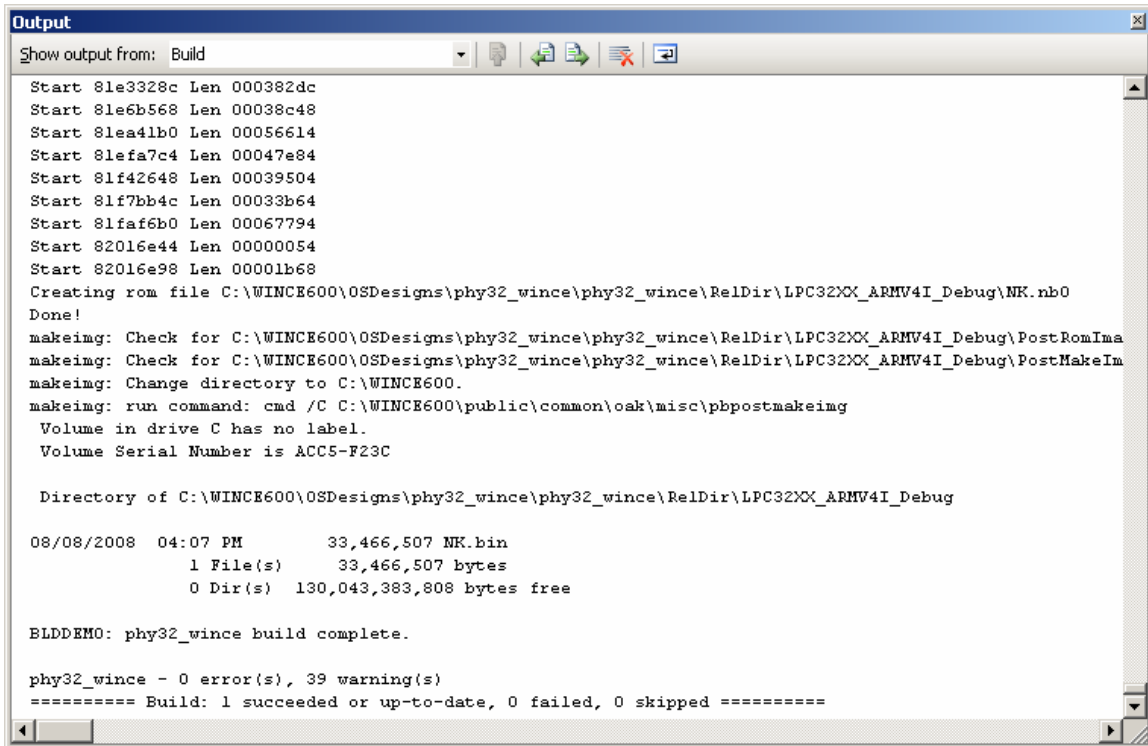
Select the options as shown in the image below. This will enable support for the ethernet debugger in the image. Press the OK button.



4.3 Building the WinCE image

After the image has been configured in the catalog, the image can be built. Depending on the speed of your machine, build times may vary.

To start the image build process, select Build Solution from the Visual Studio Build menu. After a few minutes, the build will complete with a number of files generated in the build output directory (`(\WINCE600\OSDESIGNS\PHY32_WINCE\PHY32_WINCE\RELDIR\LPC32XX_ARMV41_DEBUG)`). Text similar to the messages below should appear in the Visual Studio output window after the build is complete.



```

Output
Show output from: Build
Start 81e3328c Len 000382dc
Start 81e6b568 Len 00038c48
Start 81ea41b0 Len 00056614
Start 81efa7c4 Len 00047e84
Start 81f42648 Len 00039504
Start 81f7bb4c Len 00033b64
Start 81faf6b0 Len 00067794
Start 82016e44 Len 00000054
Start 82016e98 Len 00001b68
Creating rom file C:\WINCE600\OSDesigns\phy32_wince\phy32_wince\RelDir\LPC32XX_ARMV4I_Debug\NK.nb0
Done!
makeimg: Check for C:\WINCE600\OSDesigns\phy32_wince\phy32_wince\RelDir\LPC32XX_ARMV4I_Debug\PostRomIma
makeimg: Check for C:\WINCE600\OSDesigns\phy32_wince\phy32_wince\RelDir\LPC32XX_ARMV4I_Debug\PostMakeIm
makeimg: Change directory to C:\WINCE600.
makeimg: run command: cmd /C C:\WINCE600\public\common\oak\misc\pbpostmakeimg
Volume in drive C has no label.
Volume Serial Number is ACC5-F23C

Directory of C:\WINCE600\OSDesigns\phy32_wince\phy32_wince\RelDir\LPC32XX_ARMV4I_Debug

08/08/2008  04:07 PM          33,466,507 NK.bin
             1 File(s)      33,466,507 bytes
             0 Dir(s)  130,043,383,808 bytes free

ELDDemo: phy32_wince build complete.

phy32_wince - 0 error(s), 39 warning(s)
===== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped =====

```

4.4 Deploying the WinCE bootloader and image

These steps explain how to deploy the WinCE bootloader and WinCE image on the Phytex 3250 board. To use this method, the Stage 1 Loader must be installed on the Phytex board. More information about S1L can be found with the documentation included with the Phytex 3250 software CD.

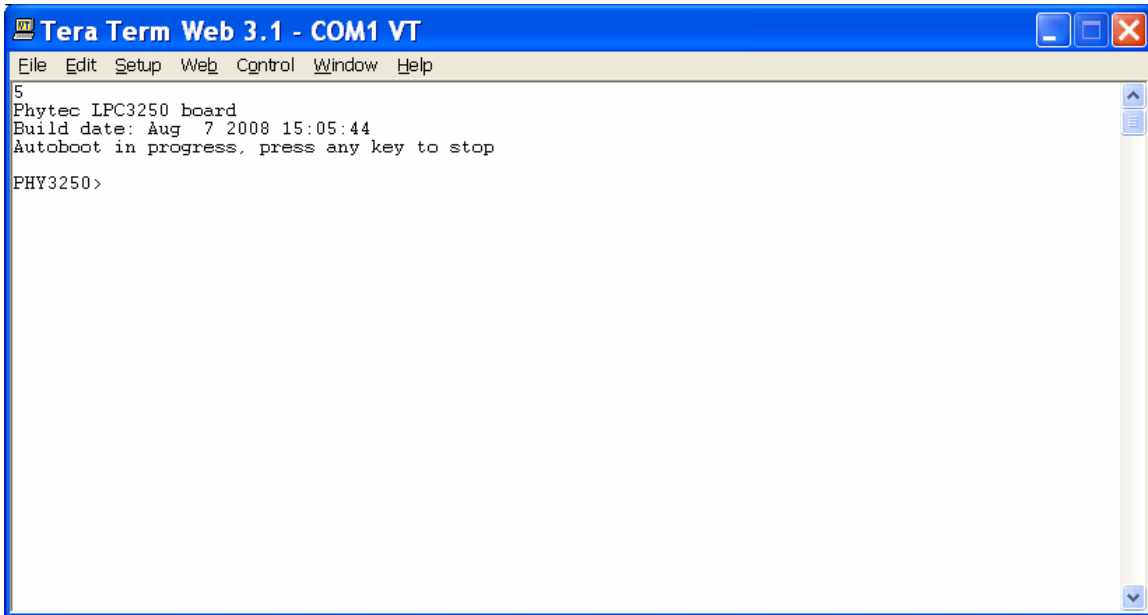
This specific example uses the SD card to boot the WinCE Ethernet bootloader and the Ethernet bootloader uses the network to download the WinCE image from Platform Builder.

4.4.1 Configuring S1L to boot the WinCE Ethernet bootloader

Locate the EBOOT.NB0 file in the WinCE build output directory. Copy the file to the root directory of an SD card.

Connect a serial cable between the bottom serial power on the Phytex board and a PC. Start a terminal program (such as TeraTerm) on the PC and setup serial communications for 115.2K-8-N-1.

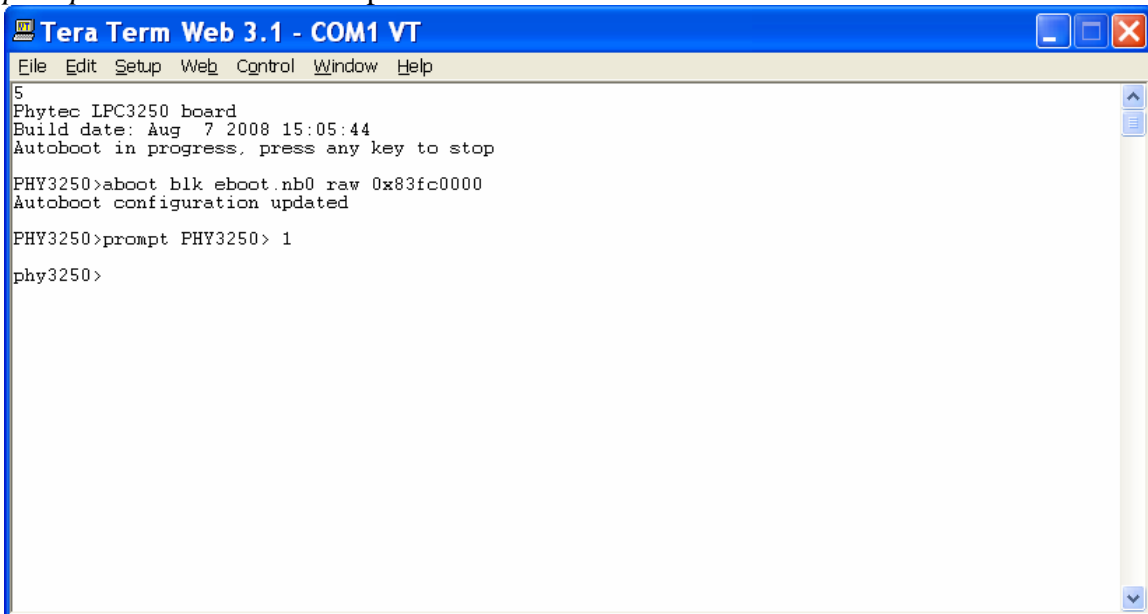
Insert the card into the SD slot of the Phytex board and power up or reset the board. A banner and the S1L prompt should appear similar to the image below. If the prompt doesn't appear, but the banner appears, you may need to press a key to get to the prompt. If the prompt doesn't appear, S1L may not be loaded onto the board. See the documentation included with the board on how to restore the S1L image to the board.



```

Tera Term Web 3.1 - COM1 VT
File Edit Setup Web Control Window Help
5
Phytec LPC3250 board
Build date: Aug 7 2008 15:05:44
Autoboot in progress, press any key to stop
PHY3250>
    
```

At the S1L prompt, type *about blk eboot.nb0 raw 0x83fc0000* to setup S1L to boot and execute the Ethernet bootloader from the SD card whenever the board is reset. Type *prompt PHY3250> 1* to setup S1L to boot after 1 second.



```

Tera Term Web 3.1 - COM1 VT
File Edit Setup Web Control Window Help
5
Phytec LPC3250 board
Build date: Aug 7 2008 15:05:44
Autoboot in progress, press any key to stop
PHY3250>about blk eboot.nb0 raw 0x83fc0000
Autoboot configuration updated
PHY3250>prompt PHY3250> 1
phy3250>
    
```

Reset or power cycle the Phytec board and the Ethernet bootloader will boot after a few seconds. As soon as the WinCE Ethernet bootloader header appears on the terminal, press the space key to stop WinCE from booting. You will enter the WinCE boot menu.

```

Tera Term Web 3.1 - COM1 VT
File Edit Setup Web Control Window Help
5
Phytec LPC3250 board
Build date: Aug 7 2008 15:05:44
Autoboot in progress, press any key to stop

Microsoft Windows CE Bootloader Common Library Version 1.4 Built Aug 8 2008 13:50:51
System ready!
Preparing for download...
Microsoft Windows CE EBOOT 1.0 for NXP LPC32XX Built Aug 8 2008 at 13:52:54
INFO: Boot configuration found
Hit space to enter LPC32XX bootloader menu.
.

-----
NXP LPC32XX Main Menu
-----
[1] Change boot timeout
[2] Set baud rate
[3] Show Current Settings
[4] Select Boot Device
[5] Select Debug Device
[6] Network Settings
[7] Force clean boot
[9] Save Settings
[0] Exit and Continue

Selection:

```

4.4.2 Configuring the WinCE Ethernet bootloader

From the menu, select [4] Select Boot Device and then select [1] LPC32XX RMII Ethernet to setup image boot over Ethernet.

```

Tera Term Web 3.1 - COM1 VT
File Edit Setup Web Control Window Help
[0] Exit and Continue

Selection: 4

-----
Select Boot Device
-----
[1] LPC32xx RMII ethernet
[2] SD/MMC Card
[3] NAND FLASH
[0] Exit and Continue

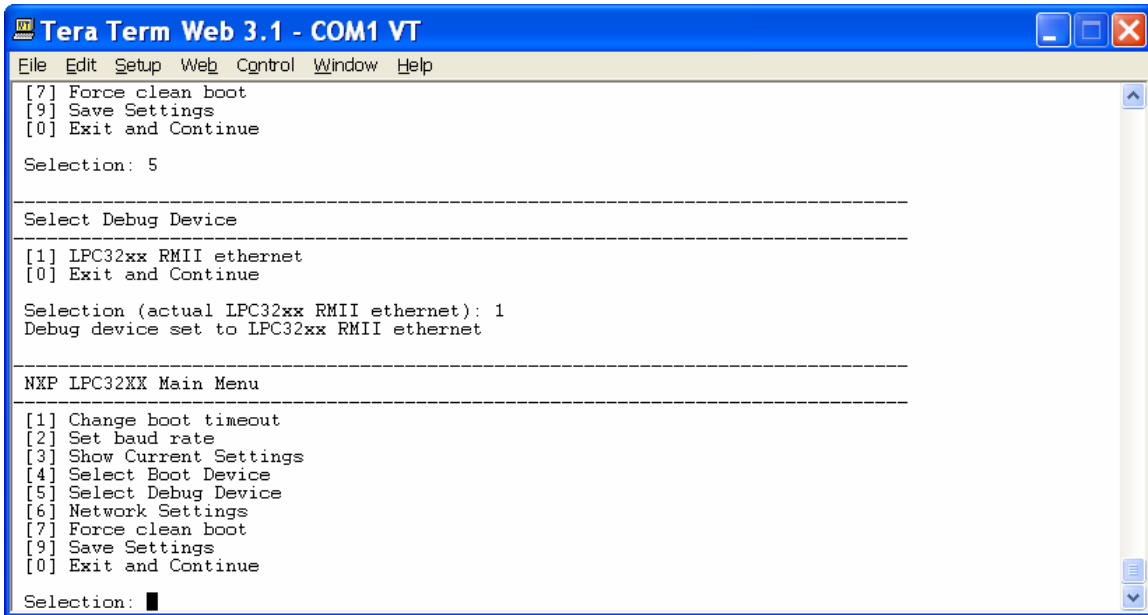
Selection (actual LPC32xx RMII ethernet): 1
Boot device set to LPC32xx RMII ethernet

-----
NXP LPC32XX Main Menu
-----
[1] Change boot timeout
[2] Set baud rate
[3] Show Current Settings
[4] Select Boot Device
[5] Select Debug Device
[6] Network Settings
[7] Force clean boot
[9] Save Settings
[0] Exit and Continue

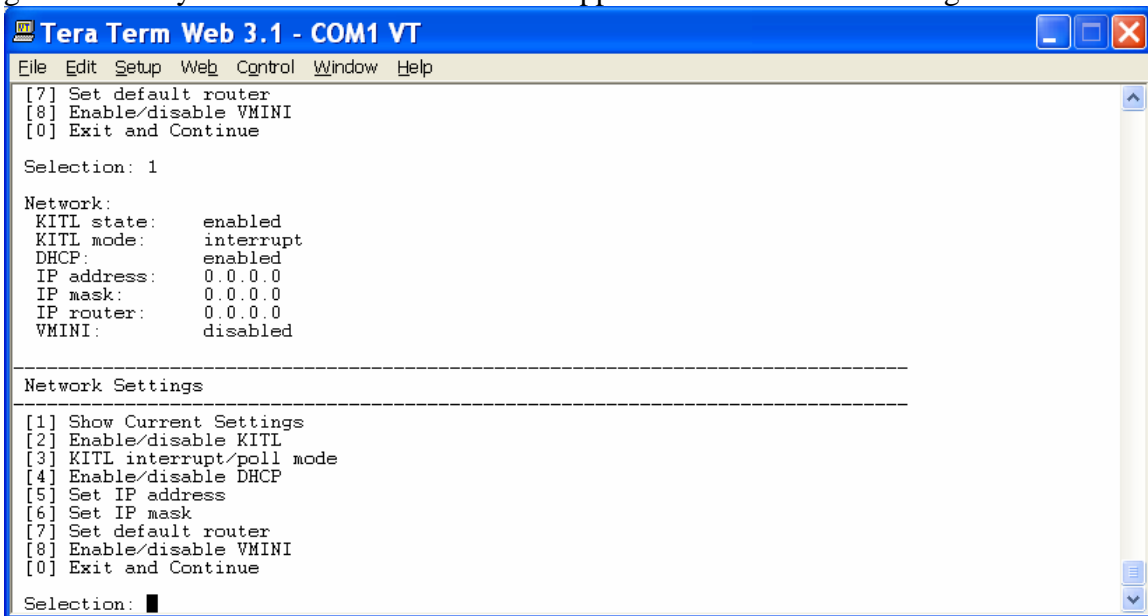
Selection:

```

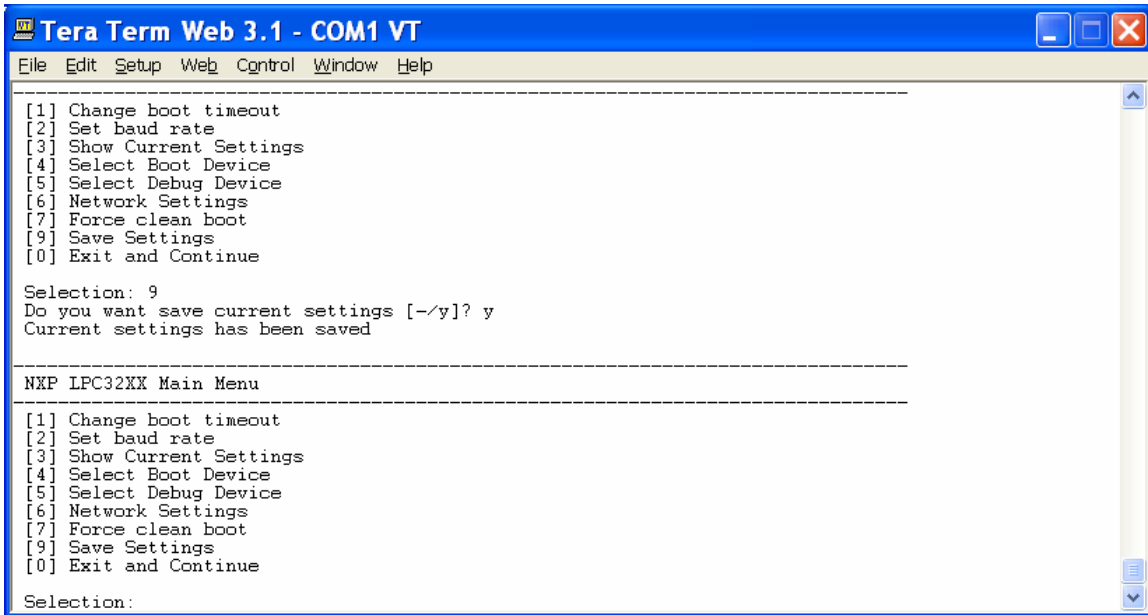
Select [5] Select Debug Device and then select [1] LPC32XX RMII Ethernet to select the Ethernet as the KITL transport.



Selection [6] Network Settings. Enable KITL with option [2]. Enable KITL interrupt mode with option [3]. Also enable DHCP with option [4] or setup the IP support with options [5], [6], and [7]. Disable VMINI, option [8]. If you select option [1], you should get a summary of the selections that should appear similar to the following:

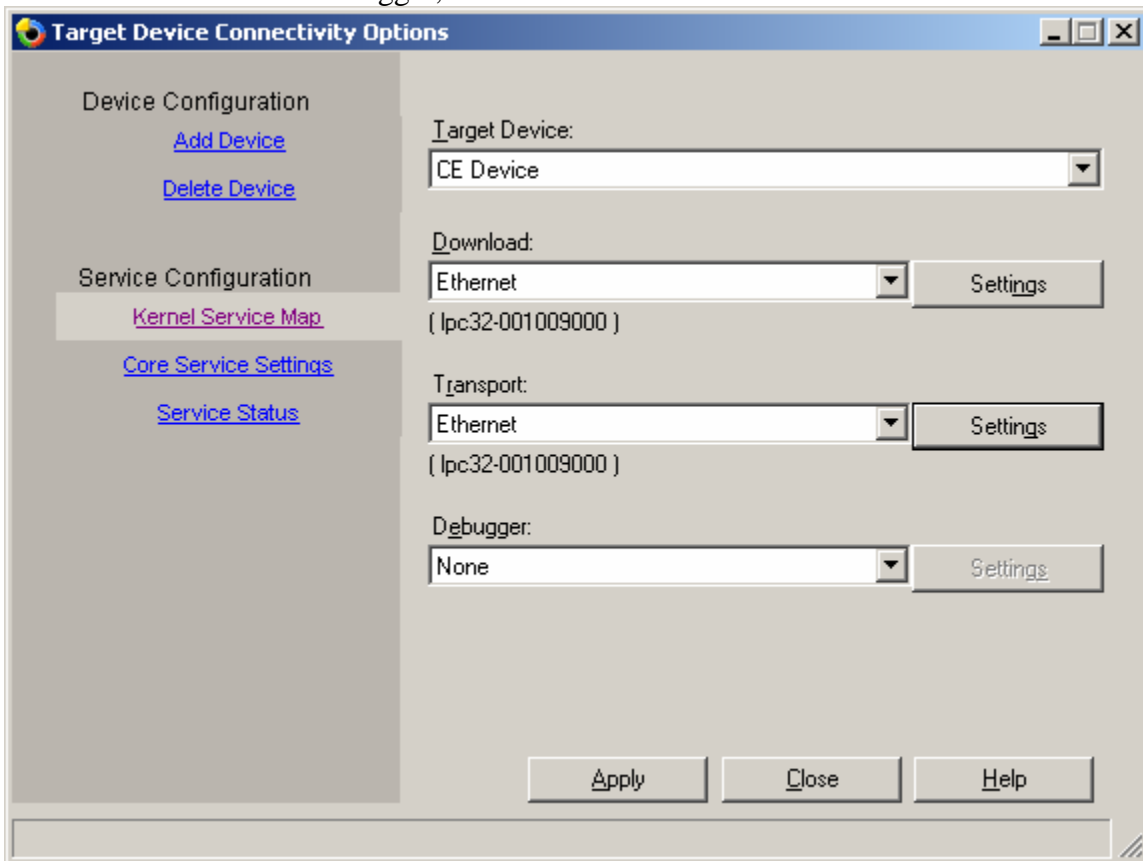


Press [0] to return to the main menu. Press [9] to save the settings.

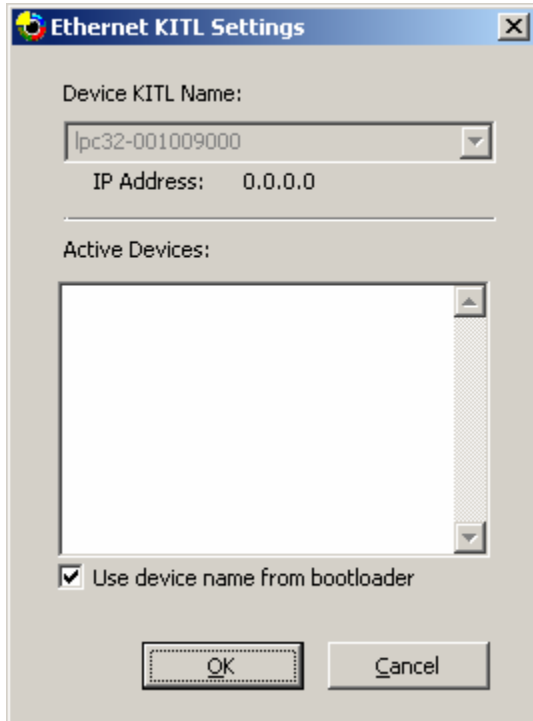


4.4.3 Setting up Platform Builder download and KITL transports

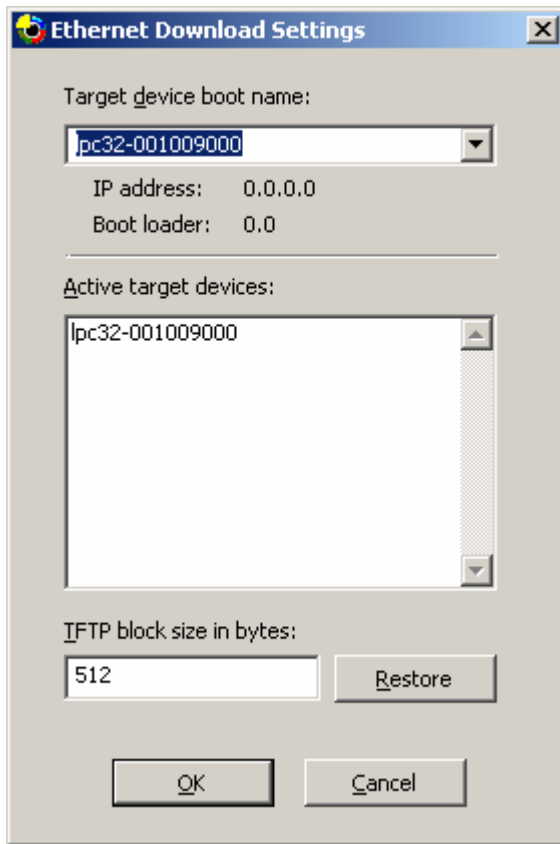
From the Visual Studio Target menu, select Connectivity Options. For the Target device, select CE Device. For the Download option, select Ethernet. For the Transport option, select Ethernet. For the Debugger, select None.



Press the Settings button next to the Transport box. For the window that appears, make sure the Use device name from bootloader option is enabled. Press the OK button.

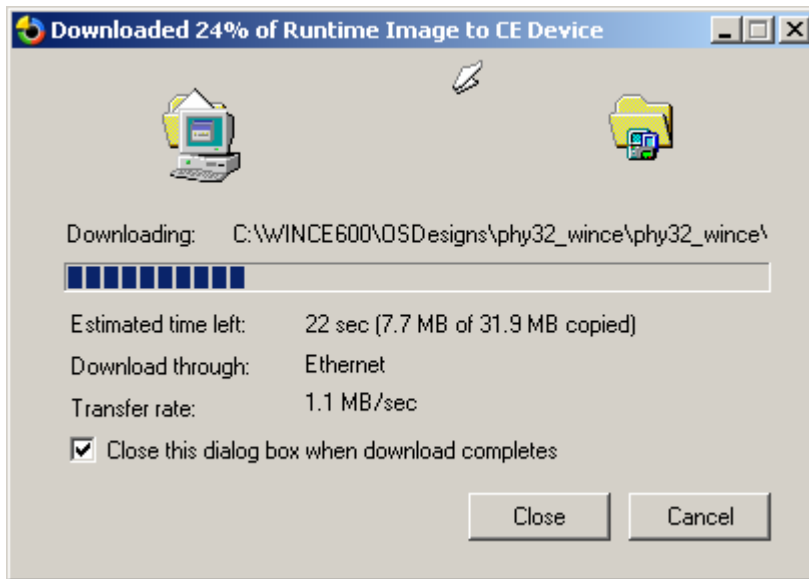


Press the Settings button next to the Download box. Make sure an Ethernet cable is connected between the board and an Ethernet hub or switch. Keeping this window open, press reset on the Phytex board. After a few seconds, a new target device name will appear in the window. Select the device and press the OK button and return to the Connectivity Options window.



Press the the Apply and Close buttons on the connectivity Options window to save the download and transport setups.

From the Visual Studio Target menu, select Attach Device. After a few seconds, the WinCE image should start downloading to the board. If it doesn't start downloading, the WinCE bootloader may have timed out so reset the board and the board will attempt to connect again. As the file is downloading, the download progress window will appear.



After the image is downloaded, execution of WinCE will start on the board. Using the Visual Studio Target menu, you can use remote tools to examine how WinCE is running, start programs, or transfer files.

This completes the application build example.

5 Deployment options

EBOOT and the WinCE image can be booted from NAND FLASH or an SD card instead of through Ethernet. Setup of these options requires adjusting boot settings in S1L or the WinCE Ethernet bootloader.

5.1 Setting up the Ethernet bootloader to boot from an SD card

To boot the Ethernet bootloader from an SD card, copy the EBOOT.NB0 file generated for your design into the root directory of an SD card. Insert the SD card into the board SD card slot. Power up or reset the board and press any key to get to the S1L prompt.

At the prompt, type the following commands to setup for SD card boot.

```
About blk eboot.nb0 raw 0x83fc0000
Prompt PHY3250> 1
```

The next time the board is powered up or reset, the Ethernet bootloader will attempt to load and execute from the SD card.

5.2 Setting up the Ethernet bootloader to boot from an NAND FLASH

To boot the Ethernet bootloader from NAND FLASH, copy the EBOOT.NB0 file generated for your design into the root directory of an SD card. Insert the SD card into

the board SD card slot. Power up or reset the board and press any key to get to the S1L prompt.

At the prompt, type the following commands to setup for SD card boot.

```
load blk eboot.nb0 raw 0x83fc0000
nsave
about flash raw 0x83fc0000
Prompt PHY3250> 1
```

The next time the board is powered up or reset, the Ethernet bootloader will attempt to load and execute from the NAND FLASH.

S1L manages where in FLASH the EBOOT.NB0 image is saved to and loaded from. Once programmed into FLASH, the SD card isn't needed to boot EBOOT.

5.3 Booting the WinCE image from an SD card

The Ethernet bootloader can load and execute the WinCE image from an SD card by selecting the SD card as the boot device in the Ethernet bootloader.

Copy the NK.BIN file generated for your design into the root directory of an SD card. Insert the SD card into the board SD card slot. Once the WinCE Ethernet bootloader is running, press any key to get to the menu.

Select the follow options from the Ethernet bootloader to setup SD card boot of the WinCE image:

```
[4] Select Boot Device
      [2] SD/MMC Card
          Enter image name(actual 'nk.bin'): NK.BIN
[9] Save Settings
```

The next time the board is powered up or reset, the Ethernet bootloader will attempt to load and execute the WinCE image from the SD card. *SD card boot is slow.*

5.4 Booting the WinCE image from NAND FLASH

The Ethernet bootloader can load and execute the WinCE image from NAND FLASH by selecting FLASH as the boot device in the Ethernet bootloader. S1L is used to butn the WinCE image into FLASH.

Copy the NK.BIN file generated for your design into the root directory of an SD card. Insert the SD card into the board SD card slot. Power up or reset the board at get to the S1L prompt. At the prompt, type the following:

```
Load blk nk.bin raw 0x80000000
Nburn 100 0
```

Start the Ethernet bootloader using one of the methods in Sections 5.1 or 5.2 and get to the bootloader menu. Select the following options from the Ethernet bootloader to FLASH boot of the WinCE image:

- [4] Select Boot Device
- [3] NAND FLASH
- [9] Save Settings

The next time the board is powered up or reset, the Ethernet bootloader will attempt to load and execute the WinCE image from the NAND FLASH.

6 Miscellaneous

6.1 Issues

If you are having issues booting WinCE, make sure the Stage 1 Loader is updated to the latest version. S1L can be downloaded as part of the CDL on NXP's website. S prebuilt binary for S1L is included with the CDL.