i.MX23 EVK Windows Embedded CE 6.0

User's Guide

Document Number: 924-76401
Rev. 2010.05

freescale™
semiconductor

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064
Japan
0120 191014 or
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Contents

Chapter 5
Preparing for Downloading and Debugging

Chapter 6
Power Management

Chapter 7
Downloading and Debugging Images

About This Book

This document is a user's guide for the Freescale iMX233-EVK Windows Embedded CE 6.0 board support package (BSP). This document will describe how to install the BSP and how to use Microsoft Platform Builder for Windows CE 6.0 to build, download, and debug OS images. Information on the configuration and usage of features included within the Freescale BSP will also be provided.

Audience

This guide is intended for users of Microsoft Platform Builder who want to build and execute Windows CE 6.0 OS images based on the Freescale BSP. The audience also includes Windows CE application developers that want to leverage API interfaces provided by the Freescale BSP.

Organization

This document is organized into the following chapters.

Chapter 1	Describes how to install/uninstall the Freescale BSP.
Chapter 2	Describes the contents and organization of the Freescale BSP.
Chapter 3	Describes how to configure the Freescale BSP for building Windows Embedded CE 6.0 OS images.
Chapter 4	Provides instructions on how to build Windows Embedded CE 6.0 OS images using the Freescale BSP.
Chapter 5	Describes the preparation necessary for downloading and debugging OS images.
Chapter 6	Describes how to configure the power management related setting.
Chapter 7	Describes the procedures for downloading and debugging OS images..

Suggested Reading

Additional information regarding Microsoft Windows CE can be found in these documents:

- i.MX233 Windows Embedded CE 6.0 Reference Manual
- i.MX233 Windows Embedded CE 6.0 Release Notes
- http://msdn.microsoft.com/embedded/windowsce
- Windows Embedded Training Resources: http://www.windowsembedded.com/training
- MCP Certification for Windows Embedded CE: http://www.windowsembedded.com/certification
- Windows Embedded CE 6.0 Online Help

Conventions

Use this section to name, describe, and define any conventions used in the book. This document uses the following notational conventions:

- `Courier monospaced type` indicate commands, command parameters, code examples, expressions, datatypes, and directives.
- *Italic type* indicates replaceable command parameters.

- All source code examples are in C.

Definitions, Acronyms, and Abbreviations

The following list defines the abbreviations used in this document.

API	application programming interface
BSP	board support package
CSP	chip support package
DHCP	dynamic host configuration protocol
DVFS	dynamic voltage and frequency scaling
EBOOT	Ethernet bootloader
EVB	platform evaluation board
ICE	in-circuit emulator
IDE	integrated development environment
IST	interrupt service thread
IPU	image processing unit
KITL	kernel independent transport layer
MMC	MulitMedia Card
MSI	Microsoft Installer
OAL	OEM adaptation layer
OEM	original equipment manufacturer
OS	operating system
PQOAL	production quality OEM adaptation layer
RTC	real time clock
SD	secuire digital card
SDRAM	synchronous dynamic random access memory
SoC	system on a chip

References

The following documents were referenced to produce this document.

- Windows Embedded CE 6.0 Online Help

Chapter 1
BSP Installation

The BSP is distributed as single Microsoft Install (.msi) file that includes support for the i.MX23-EVK Platform. Please refer to the Windows Embedded CE 6.0 *i.MX23-EVK BSP Release Notes* for additional instructions and information before installing and using this BSP.

1.1 Installing the Windows Embedded Tools and the i.MX23-EVK BSP

The following steps will install the Microsoft Windows Embedded CE 6.0 development tools and the Freescale i.MX23-EVK BSP to provide a complete i.MX23-EVK development environment.

1. Install Visual Studio 2005 and the Windows Embedded CE 6.0 Platform Builder plugin from the installation discs. Please refer to the *Release Notes* on the Visual Studio and Platform Builder installation discs for installation instructions.

   During Platform Builder installation, be sure to select which versions of the operating system to include. The **ARMV4I** version must be installed on the local hard drive for this BSP. All other versions are not required.

   Refer to the Release Notes for information about any additional Microsoft-supplied QFEs or Service Packs that also need to be installed.

2. If you have previously installed an earlier version of the Freescale i.MX23-EVK BSP, remove any existing BSP files by following the instructions in Uninstalling the Freescale i.MX23-EVK BSP.
3. Download the Freescale i.MX23-EVK BSP and install the contents into the existing WINCE600 top-level folder (the MSI installer will automatically create the necessary subfolders so that all of the files will be installed into the correct location).

1.2 Uninstalling the Freescale i.MX23-EVK BSP

This section describes how to remove an installation BSP from the Windows Embedded CE 6.0 source code tree and Platform Builder development environment.

Uninstalling the BSP will remove all files that were installed. If you have made any changes to these files, they will be removed. Be sure to save off any modified files you want to keep before uninstalling the BSP.

The following steps will remove the BSP:

1. Close Platform Builder.
2. Either rerun the original MSI installer and select the “Remove” option or open up the “Add or Remove Programs” Control Panel applet, select the Freescale BSP item, and the select “Remove”.
3. Manually remove the following BSP files and directories:

   ```
   \WINCE600\OSDesigns\iMX233-EVK-PDK1_8-Mobility
   ```

```
\WINCE600\OSDesigns\iMX233-EVK-PDK1_8-SmallFootprint
\WINCE600\PLATFORM\COMMON\SRC\SOC\COMMON_FSL_V2_PDK1_8
\WINCE600\PLATFORM\COMMON\SRC\SOC\MX233_FSL_V2_PDK1_8
\WINCE600\PLATFORM\iMX233-EVK-PDK1_8
\WINCE600\SUPPORT_PDK1_8
```

4. Manually remove the residual BSP library files that were created after the BSP was installed. To find the libraries, use Windows Explorer with the search name `*FSL_V2*` for the following library path and remove all the `LIB`, `PDB`, and `DEF` files found by the search:

   ```
   WINCE600\PLATFORM\COMMON\LIB\ARMV4I
   ```

1.3 Load Sample OS Design Solution

After installing the BSP, you can use the sample OS solutions provided in the BSP package to build a Windows Embedded CE 6.0 OS image.

1. In Platform Builder, select **File → Open → Project/Solution....**
2. Select the i.MX23-EVK BSP sample workspace by loading the following file:

   ```
   WINCE600\OSDesigns\iMX233-EVK-PDK1_8-Mobility\iMX233-EVK-PDK1_8-Mobility.sln
   WINCE600\OSDesigns\iMX233-EVK-PDK1_8-UUT\iMX233-EVK-PDK1_8-UUT.sln
   WINCE600\OSDesigns\iMX233-EVK-PDK1_8-SmallFootprint\iMX233-EVK-PDK1_8-SmallFootprint
   .sln
   ```

The process of loading this solution should also automatically load the associated i.MX23-EVK BSP catalog.

Chapter 2
BSP Contents and Organization

The Freescale i.MX23-EVK BSP is a collection of code and support files that can be integrated into the Microsoft Platform Builder development environment to create Windows Embedded CE 6.0 OS images for i.MX23-EVK board. The BSP contains the following elements:

- Boot loader for downloading OS images
- OEM Adaptation Layer (OAL) for providing the kernel hardware interface
- Device drivers to support on-chip and on-board peripherals
- Image configuration and build files

The BSP includes a set of directories and files that are installed into an existing Windows Embedded CE 6.0 source tree.The BSP directory structure follows the production-quality OAL (PQOAL) and production-quality drivers (PQD) structure recommended by Microsoft.

2.1 System-on-a-Chip Support Package (SOC)

The Freescale SOC directory contains a collection of Freescale chipset-level code that can be leveraged to develop platform based on the i.MX23-EVK SOC and the PQOAL components customized for the i.MX23-EVK. The driver code and definitions in the SOC directory can be reused in a new platform design for some SOC without modification. To keep the SOC sources platform agnostic, drivers in the SOC directory utilize hardware abstraction routines that must be ported to a specific platform or board. The SOC source code is compiled into a set of static libraries that are ultimately linked with platform-specific libraries to create drivers for the system.

2.1.1 Production Quality Driver (PQD)

Windows Embedded CE 6.0 supports PQD components that simplify and shorten the process of developing a Driver. For more information on PQQL development concepts, refer to the topic "*Production-Quality Drivers*" in the *Windows Embedded CE 6.0 Help*.

The following directories contain the SOC driver source code for the i.MX23-EVK:

```
WINCE600\PLATFORM\COMMON\SRC\SOC\COMMON_FSL_V2_PDK1_8
WINCE600\PLATFORM\COMMON\SRC\SOC\MX233_FSL_V2_PDK1_8
```

SOC driver code in the `MX233_FSL_V2` directory is reusable across all platforms based on i.MX23-EVK.

2.1.2 Production Quality OAL (PQOAL)

Windows Embedded CE 6.0 supports PQOAL components that simplify and shorten the process of developing an OAL. For more information on PQOAL development concepts, refer to the topic "*Production-Quality OAL*" in the *Windows Embedded CE 6.0 Help*.

Where possible, the Freescale BSP leverages the PQOAL architecture and components provided by Microsoft to reduce the OAL code that needs to be modified and maintained by the OEM. In addition, PQOAL components customized for the i.MX23-EVK are available in the following directories:

```
WINCE600\PLATFORM\COMMON\SRC\SOC\MX233_FSL_V2_PDK1_8\OAL
```

PQOAL code in the `MX233_FSL_V2\OAL` directory is reusable across all platforms based on i.MX23-EVK.

2.2 i.MX23-EVK Platform Files

All of the driver and OAL content that is specific to the i.MX23-EVK hardware platform is located in the following directory:

```
WINCE600\PLATFORM\iMX233-EVK-PDK1_8
```

The i.MX23-EVK platform directory implements the hardware abstraction routines invoked by driver code in the Freescale SOC directory. In addition, this directory implements certain aspects of the PQOAL that may need to be modified by the OEM for their specific platform.

2.3 Sample OS Design Solutions

Design solutions are used by Platform Builder to encapsulate the OS components and build options necessary for the Windows Embedded CE 6.0 tools to generate an OS image. Default solutions have been included in the BSP within the following directory:

```
WINCE600\OSDesigns\iMX233-EVK-PDK1_8-Mobility
```

The solution below is included in the BSP that provides the workspace for building the MfgTool Windows Embedded CE 6.0 firmware run-time image:

```
WINCE600\OSDesigns\iMX233-EVK-PDK1_8-UUT
```

There is another solution included in the BSP that provides the starting point for the smallest functional Windows Embedded CE 6.0 run-time image:

```
WINCE600\OSDesigns\iMX233-EVK-PDK1_8-SmallFootprint
```

The solution was created using the **New → Project** feature within Platform Builder and utilizing the Custom Device Design Template. For more information about creating an OS solution, refer to the topic "*Creating an OS Design with the Windows Embedded CE OS Design Wizard*" in the *Windows Embedded CE 6.0 Help*.

2.4 Support Files

Support files that complement the BSP source tree are located within the following directory:

```
WINCE600\SUPPORT_PDK1_8
```

This directory is used to store applications and tests targeted for the supported platforms. Support files necessary to configure the development tools are also placed here.

Chapter 3
Configuring OS Images

You can use Platform Builder to select one of the two default build configurations provided in the BSP sample solution. These configurations control the type of OS image (debug or release) that will be generated by the Platform Builder tools. In addition, the build configuration encapsulates all of the platform environment variables and custom build instructions that will be used during OS image creation. This section will describe the configuration options available for the i.MX23-EVK BSP.

NOTE

The sample solution provided within the i.MX23-EVK BSP is properly configured to generate a default image targeted for the i.MX23-EVK hardware. It is not necessary to adjust any of the image configuration settings prior to building the BSP.

3.1 Image Build Type

The type of OS image generated by Platform Builder can be controlled using the **Build → Configuration Manager...** menu item and choosing the appropriate **Active Solution Configuration** item from the drop-down list. The sample workspace provided with the i.MX23-EVK BSP provides the following two image build types:

- **Freescale i_MX233 EVK Release**—Retail build that includes KITL.This image type provides a smaller image with faster execution at the expense of limited debug capability.
- **Freescale i_MX233 EVK Debug**—Debug build that includes KITL and kernel debugger support. This image type provides full debug capability at the expense of a larger image size and slower execution.

NOTE

The Standard toolbar can enabled/disabled by selecting the **Tools → Customize...** menu item followed by the **Toolbars** tab and then selecting/deselecting the **Standard** toolbar item.

For more information about the build types available for Windows CE, refer to the topic "*Build Configurations*" in the *Windows Embedded CE 6.0 Help*.

3.2 BSP Environment Variables

There are three types of BSP environment variables used to configure the BSP: BSP_NOXXX, BSP_XXX and IMGXXX.

3.2.1 BSP_NOXXX Variables

The i.MX23-EVK BSP supports the Windows Embedded CE 6.0 dependency feature, which simplifies the BSP configuration process. Support for this feature means that the selection of certain SYSGEN components in the workspace (e.g. SYSGEN_DISPLAY, SYSGEN_AUDIO) will trigger the automatic selection of certain BSP drivers (e.g. Display, Audio). The drivers that have a SYSGEN dependency will

have a corresponding BSP_NOXXX variable defined in the platform batch file listed below. These variables provide a means for excluding a driver from the OS image that might otherwise be included, due to a SYSGEN dependency.

```
WINCE600\PLATFORM\iMX233-EVK-PDK1_8\iMX233-EVK-PDK1_8.bat
```

NOTE

It is not possible to remove the selection of these drivers by clicking the items in Catalog UI. Instead, users need to set the corresponding BSP_NOXXX variables as '1' in **Environment** dialog (launched from Platform Builder menu **Project > Properties > Configuration Properties > Environment**).

Table 3-1. BSP_NOXXX Variables in Batch File

Variable Name	Description	Settings
BSP_NOCSPDDK	Used to exclude i.MX23-EVK CSP driver development kit (CSPDDK) support. The CSPDDK is required for most BSP device drivers.	BSP_NOCSPDDK = 1 Excludes CSPDDK from the OS image. Only use this configuration when building an OS design that does not include BSP device drivers. DO NOT define for sample workspace.
BSP_NOAUDIO	Used to exclude support for audio driver.	BSP_NOAUDIO = 1 Excludes audio driver support from the OS image.
BSP_NONAND_FMD	Used to exclude support for NAND driver	BSP_NONAND_FMD=1 Excludes NAND driver
BSP_NOUSB	Used to exclude support for all usb driver, including client, host and otg.	BSP_NOUSB=1 Excludes USB driver
BSP_NONLED	Used to exclude support for LED notification driver	BSP_NONLED=1 Excludes NLED driver
BSP_NODISPLAY	Used to exclude support for display driver	BSP_NODISPLAY=1 Excludes DISPLAY driver
BSP_NOTOUCH	Used to exclude support for Touch driver	BSP_NOTOUCH=1 Excludes Touch Driver
BSP_NOBATTERY	Used to exclude support for Battery Driver	BSP_NOBATTERY=1 Excludes Battery Driver
BSP_NOSSP1_SDHC	Used to exclude support for SDHC driver	BSP_NOSSP1_SDHC=1 Excludes SD driver
BSP_NOKEYPAD	Used to exclude support for Keypad driver	BSP_NOKEYPAD=1 Excludes Keypad driver

NOTE

The opposite setting of "BSP_NOXXX=1" would be either "BSP_NOXXX=" which is set in the batch file by default, or not to define the BSP_NOXXX variable at all. Thus, the driver will be included by SYSGEN selection.

Though the drivers listed above have direct SYSGEN dependency, not all of them can be automatically selected by SYSGEN dependency for the possible reasons below.

1. The driver has multiple subordinate selections, for example there are two selections LMS350GF10(QVGA) and LM430HF02(WQVGA) for Display.
2. The driver depends on another driver, for example Battery depends on LRADC driver, and Touch depends on LRADC driver.
3. The driver conflicts with other drivers, for example CSP driver conflicts with SPI_SDHC.

3.2.2 BSP_XXX Variables

The i.MX23-EVK BSP uses BSP_XXX variables defined in Catalog to configure the drivers that can not automatically be selected by SYSGEN dependency. In these cases, users can use Platform Builder Catalog UI to select/deselect drivers by clicking the catalog items. The *Windows Embedded CE 6.0 BSP for i.MX23-EVK Reference Manual* describes the BSP_XXX variables for individual driver.

NOTE

Users might not always select the desired drivers successfully in Catalog UI. Instead, users will possibly get a small red ‘x’ for some cases, which means there are some dependency or conflict involved in the selection. In this case, users need to right-click the catalog item and see the details in ‘Reasons for Exclusion of Item’. In order to get a successful selection, follow the info to select the required SYSGEN components or drivers and to deselect conflicting components.

NOTE

When creating custom workspace using Platform Builder wizard, users are always expected to go through BSP catalog items under **Third party > BSP > Freescale i.MX233-EVK: ARMV4I** to manually select those desired drivers which may not be automatically selected by SYSGEN dependency.

3.2.3 IMG_XXX Variables

The IMG_XXX variables are used to control Make Run-Time Image procedure, and can be viewed and configured within the **Environment** dialog as follows:

1. Open the sample solution for the i.MX23-EVK BSP.
2. From the **Project** menu, choose **Properties**.
3. Expand **Configuration Properties** if necessary and select the **Environment** item.
4. Table 3-2 provides a summary of the IMG_XXX environment variables available on i.MX23-EVK BSP.

Table 3-2. IMG_XXX Variables

Variable Name	Description	Settings
IMGNAND	Used by EBOOT and OS build files to determine if images are targeted for NAND Flash.	IMGNAND = 1 Link EBOOT and OS images for NAND Flash.

3.2.4 Catalog Environment Variables

The i.MX23-EVK BSP utilizes the Platform Builder Windows CE Catalog to allow users to configure BSP components in the OS design. The Windows Embedded CE 6.0 BSP for i.MX23-EVK BSP Reference Manual describes the environment variables associated with each of the BSP features exposed in the i.MX23-EVK BSP Catalog.

Chapter 4
Building OS Images

This chapter provides instructions for building Windows Embedded CE 6.0 OS images using the BSP.

4.1 Building the Freescale SOC Libraries

The Freescale SOC libraries that support the i.MX23-EVK are generated during the **Build → Advanced Build Commands → Sysgen** or **Build → Advanced Build Commands → Build Current BSP and Subprojects** build procedures. Windows CE ships with pre-built SOC libraries for various ARM processors, but the libraries for the i.MX23-EVK must be built from the sources since they are not included with the standard Microsoft distribution.

Note that the sample OS design solution that is provided is already preconfigured to build the Freescale SOC that is required. That is, the sample i.MX23-EVK OS design solution will automatically build the i.MX23-EVK SOC sources.

Follow these steps to build the Freescale SOC libraries:

1. Open the sample solution.
2. Select the desired build type discussed in Section 3.1, Image Build Type.
3. Select a **Sysgen** build if you have not yet performed a Sysgen operation before. Otherwise, you may select to just **Build Current BSP and Subprojects** if you have already performed a Sysgen before and just need to rebuild the Freescale SOC libraries.

For a release build type, the SOC libraries will be placed in:

```
WINCE600\PLATFORM\COMMON\LIB\ARMV4I\RETAIL
```

For a debug build type, the SOC libraries will be placed in:

```
WINCE600\PLATFORM\COMMON\LIB\ARMV4I\DEBUG
```

4.2 Building Run-Time Images

The remaining steps to build an OS image follow the standard procedures described in the Platform Builder documentation. For more information, refer to the topic "*Building a Run-Time Image*" in the *Windows Embedded CE 6.0 Help*.

4.2.1 Building the BSP for the First Time

1. Open the sample solution.
2. Select the desired build type discussed in Section 3.1, Image Build Type.
3. Using the **Build → Global Build Settings** menu, configure the build options as follows:
   - **Copy Files to Release Directory After Build** selected
   - **Make Run-Time Image After Build** selected
4. Select **Build → Advanced Build Commands → Sysgen** to start the build process.

For a release build type, the resulting OS image files will be placed in the following directory depending upon the OS design that is being used:

```
WINCE600\OSDesigns\iMX233-EVK-PDK1_8-Mobility\RelDir\Freescale_i_MX233_EVK_ARMV4I_Release
```

For a debug build type, the resulting OS image files will be placed in the following directory:

```
WINCE600\OSDesigns\iMX233-EVK-PDK1_8-Mobility\RelDir\Freescale_i_MX233_EVK_ARMV4I_Debug
```

4.2.2 Clean Build for the BSP

By default, Platform Builder performs incremental builds of the BSP components, even during the **Sysgen** build procedure. It may be desirable under certain circumstances to force a clean build for the BSP.

A clean build of the all the Freescale BSP components can be accomplished as follows:

1. **Copy Files to Release Directory After Build** selected.
2. **Make Run-Time Image After Build** unselected.
3. Select **Build → Advanced Build Commands → Rebuild Current BSP and Subprojects** to perform a clean build of the BSP platform directory (including the SOC libraries) and complete the creation of a new OS image.

4.2.3 Incremental BSP Build

The **Sysgen** build phase results in pre-built OS component binaries being copied to the release directory for the current build configuration. It is not necessary to perform a **Sysgen** again unless components are being added or removed from your **OS design**. Instead, the user can perform an incremental build of the Freescale BSP components to quickly build an updated OS image as follows:

1. Open a solution.
2. Select the desired build type discussed in Section 3.1, Image Build Type.
3. Using the **Build → Global Build Settings** menu, configure the build options as follows:
   - **Copy Files to Release Directory After Build** selected
   - **Make Run-Time Image After Build** selected
4. Select **Build → Advanced Build Commands → Build Current BSP and Subprojects** to perform an incremental build of the BSP platform directory (including the SOC libraries) and complete the creation of an OS image.

Chapter 5
Preparing for Downloading and Debugging

The target and development workstation must be properly configured and initialized before OS images can be downloaded and executed. This section will discuss the steps required to prepare the target and development workstation so that Platform Builder can be used to download and debug images on the target.

5.1 Serial Debug Messages

Serial debug messages are used by the boot loader and OS images to report status and error information. In addition, the boot loader uses serial input to allow for user interaction. This section will describe the configuration of the desktop workstation and Freescale BSP to support serial debug messages.

5.1.1 Desktop Workstation Serial Debug Port

Any terminal emulation application can be used to display messages sent from the serial port of the target. Configure your terminal application with the following communications parameters:

- Bits per second: 115200
- Data bits: 8
- Parity: None
- Stop bits: 1
- Flow control: none

5.2 i.MX23-EVK Board Configuration

This section will describe the switch and jumper configuration required for proper operation of the BSP on the i.MX23-EVK board. For specific BSP features, the i.MX23-EVK board may need to be reconfigured to support the necessary interface.

5.2.1 i.MX23-EVK Boot Mode Settings

S36 on EVK board should set 0000, which means boot from USB recover mode.

5.3 EBOOT Installation and Configuration

EBOOT is used to download and execute OS images. This section will describe the procedure for building, installing, and configuring EBOOT.

5.3.1 Building an EBOOT Image for NAND Flash

1. Visual Studio 2005 -> Build->Make run-time image
2. eboot.msb will be created automatically

5.3.2 Burn EBOOT onto NAND flash using PB

1. Power on board with boot mode (0000).
2. Run batch file below.

   ```
   WINCE600\SUPPORT_PDK1_8\TOOL\iMX233-EVK\SBIMAGE\loader.bat.
   ```
3. Quickly switch to serial terminal and press “space“ within 3 seconds.
4. Press “F” to perform “NAND Low level format”.
5. When format is done, reset board and repeat steps 2~3.

NOTE

When USB5V is available (S14 is ON, and USB cable is connected with a Host machine), a press of S24 RESET button can reset board. Otherwise, another press of S2 POWE button is needed.

6. Refer to Section 5.4, Configuring USB RNDIS Connection for Downloading and Debugging to setup RNDIS.
7. Download eboot.msb using Platform Builder.

NOTE

Platform Builder does not recognize msb file type, so eboot.msb is not in **Target file name for debugger** drop list of Property Pages. Users can manually input “eboot.msb“ in the text box and get PB download it, if the file exists in flat release directory.

8. Change boot mode to (0100)
9. Reboot board and EBOOT should boot from NAND.

5.3.3 Burn EBOOT onto SD Card using Freescale cfimager Utility

1. Plug SD Card reader to PC with SD\MMC Card.
2. Open a command prompt and change directory to:

   ```
   WINCE600\SUPPORT_PDK1_8\TOOL\COMMON\CFIMAGER
   ```
3. Run command below.

   ```
   cfimager.exe -a -f eboot.sb -d <card reader drive without colon>
   ```

NOTE

The parameter "-a" is added here to ensure the successful flash, as cfimager will format the card before flashing image with this parameter. If the card has already been formatted by cfimager and only EBOOT update is intended, the parameter "-a" should be omitted.

NOTE

Images eboot.sb and nk.sb could be found in following directory.

```
WINCE600\SUPPORT_PDK1_8\TOOL\iMX233-EVK\SBIMAGE
```

4. Plug the SD\MMC Card to board
5. Powe on board with boot mode (1001), and EBOOT should boot from SD card.

5.3.4 Burn NK image onto SD Card using Freescale cfimager Utility

The following steps should be followed to boot OS image without a bootloader

1. Plug SD Card reader to PC with SD\MMC Card.
2. Open a command prompt and change directory to:
   ```
   WINCE600\SUPPORT_PDK1_8\TOOL\COMMON\CFIMAGER
   ```
3. Run command below.
   ```
   cfimager.exe -a -f nk.sb -d <card reader drive withou colon>
   ```
4. Plug the SD\MMC Card into board
5. Powe on board with boot mode (1001), and NK should boot from SD card.

NOTE

The command above will format the card and flash nk.sb onto the card. As eboot.sb and nk.sb use the same one partition, only one of them can be on the card and get launched. To get both EBOOT and NK flashed on the card, the following command should be used.

```
cfimager -a -f eboot.sb -d <card reader drive letter without colon>
-e nk.nb0
```

Image nk.nb0 will be flashed into another partition than the one for eboot.sb, so both images will be on the card. The eboot.sb will be launched by ROM. Then in EBOOT menu, ensure "[5] Select Boot Device : NK from SD/MMC" then press "L" to get NK launched.

5.4 Configuring USB RNDIS Connection for Downloading and Debugging

To configure a USB RNDIS connection that can be used for downloading and debugging images both eboot and Visual Studio must be configured correctly:

5.4.1 Configuring Eboot

1. Press “space” to enter eboot menu
2. Choose 0 to configure IP Address, such as 192.168.0.1
3. Choose 1 to configure IP Mask, such as 255.255.255.0
4. Choose 3 to disable DHCP
5. Choose 6 to set MAC address, such as 00.11.22.33.44.00
6. Choose E to choose “USB RNDIS”
7. Choose S to save configuration
8. Choose D to start download
9. First Time PC host require install RNDIS driver, driver is WINCE600\PUBLIC\COMMON\OAK\DRIVERS\ETHDBG\RNDISMINI\HOST\usb8023.inf
10. After install rndis driver, the new network card will be added at your PC host. You need configure IP Address for that. such as 192.168.0.2.

5.4.2 Configuring Visual Studio

1. From the Platform Builder **Target** menu, choose **Connectivity Options**.
2. Choose **Kernel Service Map**.
3. In the **Target Device** box, choose a target device.

   If a connection to a target device is already configured, the settings associated with the connection to the target device appear in the **Download** box and the **Transport** box.
4. In the **Download** box, choose **None** as the download service.
5. Launch EBOOT on the target. After EBOOT initialization completes, you should begin to see BOOTME messages appear on the serial debug output. Observe the device name created by EBOOT on the serial debug output.
6. To the right of the **Download** box, choose **Settings**. The device name of your target should appear in the **Active Devices** box.
7. Select your target from the **Active Devices** box, and then choose **OK**.
8. In the **Transport** box, choose **Ethernet** as a kernel transport.
9. To the right of the **Transport** box, choose **Settings**.
10. Check the box next to **Use device name from bootloader** and then choose **OK**.
11. If your run-time image includes support for the kernel debugger stub, KdStub, from the **Debugger** box, choose **KdStub**. If your run-time image does not include support for a debugger, from the **Debugger** box, choose **None**.
12. Choose **Core Service Settings**.
13. To instruct Platform Builder to download a run-time image each time Platform Builder connects with the target device, under **Download Image**, choose **Always**.
14. Select **Enable KITL on device boot**.
15. Select **Enable access to desktop files**.
16. Choose **Apply**.
17. Choose **Close**.

5.5 Configuring USB Serial Connection for Downloading and Debugging

To configure an USB Serial connection that can be used for downloading and debugging images, The eboot and Visual Studio need configured correctly:

5.5.1 Eboot Configure

1. Press “space” to entry eboot menu
2. Choose E to choose “USB Serial”
3. Choose S to save configuration
4. Choose D to start download

5.5.2 Configuring Visual Studio and PC host

1. Disable ActiveSync USB Connection
2. From the Platform Builder **Target** menu, choose **Connectivity Options**.
3. Choose **Kernel Service Map**.
4. In the **Target Device** box, choose a target device.

   If a connection to a target device is already configured, the settings associated with the connection to the target device appear in the **Download** box and the **Transport** box.

5. In the **Kernel Download** box, choose **USB** as a kernel transport.
6. In the **Kernel Transport** box, choose **USB** as a kernel transport.
7. If your run-time image includes support for the kernel debugger stub, KdStub, from the **Debugger** box, choose **KdStub**. If your run-time image does not include support for a debugger, from the **Debugger** box, choose **None**.
8. Choose **Core Service Settings**.
9. To instruct Platform Builder to download a run-time image each time Platform Builder connects with the target device, under **Download Image**, choose **Always**.
10. Select **Enable KITL on device boot**.
11. Select **Enable access to desktop files**.
12. Choose **Apply**.
13. Choose **Close**.

Chapter 6
Power Management

This section describes how to use power management of i.MX23-EVK board. It include battery, suspend resume, DVFS and so on.

6.1 AC & Battery

S12 choose real battery or fake battery. Left is fake battery and Right is real battery.

J6 external power suppy just provide a fake battery and power on external chip, such Ethnet ENC28J60 and UART voltage convert chip.

AC charger can only connect to board by J4 USB macro-AB port.

6.2 Suspend

Suspend only support at battery only case. When AC plug, suspend will be disabled. When AC plug, battery charging will be on. Battery driver detect battery voltage, die temperature and change charging current. AC supply can provide enough power and it is unnecessary to entry suspend status.

6.3 Off state

When battery is low, system should be put to off state. All device lost status except RTC. When power on, go through whole reset flow.

Off state is not supported by Windows Embedded CE natively.

Press S2 Power button for 3 second to let system entry OFF state.

Chapter 7
Downloading and Debugging Images

This section describes the procedures for downloading and debugging OS images on the i.MX23-EVK board. It is assumed that you have followed the steps to build an OS image and configured your development hardware prior to attempting a download and debug session.

7.1 Downloading an Image into SDRAM using EBOOT

To download an image into the SDRAM of the target, follow these steps:

1. If EBOOT is not resident on the target, follow the procedure in *Section 4.2, EBOOT Installation and Configuration* to program EBOOT into the target.
2. Prepare the target hardware by following the instructions in Section 5.2, i.MX23-EVK Board Configuration. Configure the CPU board for direct external boot from NAND flash.
3. Open the desired workspace within Platform Builder.
4. *Deselect* **Build → Properties → Configuration Properties → Build Options → Write Run-time Image to Flash Memory**.
5. Within **Build → Properties → Configuration Properties → Environment**, remove the IMGNAND environment variable if it exists.
6. Build the image following the steps provided in Chapter 4, "Building OS Images".
7. If a target device connection has not been created within Platform Builder, follow the steps in Section 5.4, Configuring USB RNDIS Connection for Downloading and Debugging to establish a connection.
8. From the Platform Builder **Target** menu, select **Attach Device** to begin the download.

7.2 Downloading an OS Image into NAND Flash using EBOOT

To download an image into the NAND of the target, follow these steps:

1. If EBOOT is not resident on the target, follow the procedure in *Section 5.2, EBOOT Installation and Configuration* to program EBOOT into the target.
2. Prepare the target hardware by following the instructions in Section 5.2, i.MX23-EVK Board Configuration. Configure the CPU board for direct external boot from NAND flash.
3. Open the desired workspace within Platform Builder.
4. Within **Build → Properties → Configuration Properties → Environment**, set the environment variable IMGNAND=1.
5. Build the image following the steps provided in Chapter 4, "Building OS Images".
6. If a target device connection has not been created within Platform Builder, follow the steps in Section 5.4, Configuring USB RNDIS Connection for Downloading and Debugging to establish a connection.
7. From the Platform Builder **Target** menu, select **Attach Device** to begin the download.
8. After download, eboot will ask you whether burn image into nand flash, you press "y".

9. After eboot complete burn image to nand flash, reboot board.

7.3 Running an OS Image from NAND Flash using EBOOT

1. Press “space” to entry eboot menu.
2. Press “5” to choose boot from NAND
3. Press “L” to launch OS image from NAND flash.