Logic Product Development
User Manual

Logic Part Number: 1008544
Rev. A, 1/2008

MPC8360E-RDK Linux BSP User Manual

LOGIC

freescale™ semiconductor

Table of Contents

1 About This Manual

This User Manual provides information on the basic features supported by the BSP and provides you with instructions about how to accomplish these tasks:

- Install the BSP on a host development system
- Run Linux Target Image Builder (LTIB) to build target images
- Deploy built images to the MPC8360E-RDK board
- Boot Linux on the MPC8360E-RDK board
- Update U-Boot using U-Boot or an Abatron BDI (**required** to boot Linux 2.6.19+ kernels)

1.1 Audience

This document is addressed to developers who want to take advantage of the Freescale Linux Target Image Builder (LTIB) for the MPC8360E-RDK Board Support Package (BSP).

1.2 Organization

This document is organized into the following sections:

- Section 2 provides an introduction to the MPC8360E-RDK BSP
- Section 3 provides basic information on LTIB
- Section 4 provides important target set-up information
- Section 5 provides host and target-specific build and deployment information

1.3 Conventions

This document uses the following notational conventions:

- Courier monospaced type indicates commands, command parameters, code examples, expressions, data types, and directives.
- Italic type indicates replaceable command parameters.
- All source code examples are in C.

1.4 Definitions, Acronyms, and Abbreviations

The following list defines the abbreviations used in this document.

ATM	Asynchronous Transfer Mode
AAL	ATM Adaption Layer
APC	ATM Pace Control
BCSR	Board Control and Status Register
BSP	Board Support Package
COP	Common On-chip Processor
CSB	Coherency System Bus
CW	CodeWarrior IDE for PowerPC
DDR	Double Data Rate RAM
DIP	Dual-In-Line Package
DMA	Direct Memory Access
DTB	Device Table Binary File (Open Firmware file used by 2.6.19+ kernels)
FLASH	Non-volatile reprogrammable memory
GPCM	General Purpose Chip-select Machine
GPL	General Purpose Line
HRCW	Hardware Reset Configuration Word
HS	High Speed

LBC	Local Bus Controller
LB	Local Bus
LSB	Least Significant Bit
LS	Low Speed
LTIB	Linux Target Image Builder
MII	Media Independent Interface
NFS	Network File System
OCP	On Chip Peripherals
OTG	On-The-Go
PCI	Peripheral Components Interconnect
PCS	Platform Creation Suite
PB	83xx Processor Board
QE	83xx QUICC Engine
RTC	Real Time Clock
RCWL	Reset Configuration Word Low
RCWH	Reset Configuration Word High
RMII	Reduced Media Independent Interface
SDRAM	Synchronous Dynamic Random Access Memory
SEC	Security Engine
TBD	To Be Defined
TFTP	Trivial File Transfer Protocol
UCC	Universal Communication Controllers in QE
UEC	UCC Ethernet Controller
UPMB	User Programmable Machine B
UTOPIA	Universal Test and Operation Physical Interface for ATM

2 Introduction

2.1 LTIB Overview

The Linux Target Image Builder (LTIB) is a tools framework used to manage, configure, extend and build Linux software elements to easily build a Linux target image and a root file system. LTIB runs on an x86 PC running the Linux OS.

This BSP operates with LTIB running on a host development system with the following:

- Ethernet card
- Serial port
- 1 GB of free disk space required
- NFS Server
- TFTP Server
- rsync
- perl

NOTE: Be aware that some host side packages may not function properly on every Linux distribution. The following are platforms where LTIB was tested.

- Redhat: 7.3, 8.0, 9.0
- Fedora Core: 1, 2, 3, 5, 6
- Debian: 3.1r0 (stable), unstable
- SuSE: 8.2, 9.2, 10.0

2.2 BSP Overview

This MPC8360E-RDK BSP is designed for use with LTIB. Once the BSP is installed and running with its basic configuration, you can use LTIB to customize your project.

The BSP components provide the tools, device drivers, and additional features needed for your embedded Linux project.

Linux 2.6.19 kernel

- Linux kernel 2.6.19 supporting E300 core
- DUART1 support for console control
- I2C bus devices probe support
- IPIC support
- JFFS2/NFS/ramdisk file system deployment support
- PCI host working at 33MHz
- MPC8360E QE framework support in the kernel
- MPC8360E PB UCC1 and UCC2 working as Ethernet port, it supports the following:
  - MPC8360E QE UCC1 and UCC2 work as Gigabit Ethernet Controller
  - Auto-negotiation to provide a mechanism for transferring information from the local link port to the peer link port to establish speed ,duplex, and Master/Slave preference during a link session
  - Speed transition among 1000M,100M, and 10M
  - Can be configured to advertise both full-duplex and half-duplex
  - RGMII 1000M, 100M, and 10M Ethernet interface
  - Support Multicast Mode
  - Support NAPI mode
  - UCC1 and UCC2 is enabled in the BSP default
- UCC3 works at Fast Protocol mode as ATM drivers
- MPC8360E PB UCC7 and UCC4 working as Ethernet port, it supports the following:
  - MPC8360E QE UCC7 and UCC4 work as 100M/10M Ethernet Controller

- Auto-negotiation to provide a mechanism for transferring information from the local link port to the peer link port to establish speed, duplex, and Master/Slave preference during a link session
- Speed transition between 100M and 10M
- Can be configured to advertise both full-duplex and half-duplex
- MII 100M and 10M Ethernet interface
- Support Multicast Mode
- Support NAPI mode
- UCC7 and UCC4 are not enabled in the BSP by default

- Support MPHYs port on OC-3 card as ATM drivers
- SPI driver operation with onboard flash
- Support SEC2.x device driver demo
- MPC8360E USB1.1 controller working as device under Linux Gadget framework
- CodeTEST SWIC and HWIC code trace support
- Support MPC8360E IEEE1588 functionality demo demonstration
- Linux kernel debug for CodeWarrior and Abatron BDI-2000
- Video frame buffer support for Logic's LCD-10.4-VGA-10 Display Kit

U-Boot 1.1.6

- Enable MPC8360E rev 2.0, rev 2.1
- Local bus support
- I2C bus
- DDR memory fixed initialize
- DUART
- UCC1 working as Ethernet and TFTP port at 1000M, 100M, 10M
- Flash enable
- NAND enable
- Reset command
- Boot-up from NAND, Flash, and network
- Enable Cache and MMU in U-Boot

gcc-3.4.3-glibc-2.3.3 for 8360e, binutils-2.15

CodeWarrior 8.8 version

3 LTIB Basics

3.1 Installing the BSP

Please follow the steps below to install LTIB on your host machine.

1. As root, mount the ISO image on your machine:
   `mount –o loop <target-bsp.iso> <mount point>`

2. As a non-root user, install the LTIB:
   `<mount point>/install`

   You will be prompted to input the desired LTIB install path. Be sure the user has the correct permissions for the install path.

There are no uninstall scripts. To uninstall LTIB you need to remove the `/opt/freescale/pkgs`, `/opt/freescale/ltib` and `<install_path>/ltib` directories manually.

3.2 Running LTIB

To run LTIB, change to the directory into which you installed it and run `./ltib`.

```
cd <install_path>/ltib

./ltib
```

The first time LTIB runs on your machine a number of host packages are built and installed that support LTIB. This may take a few minutes.

Important Note: Please be sure to set the “Target System Configuration” options for your network environment the first time you build.

To modify the project configuration run:

`./ltib –-configure` (or `-c`; type `--help` to see configuration options)

This will re-prompt you for the platform/board configuration. In the board configuration screens, change settings and select packages as appropriate. When you exit the configuration screen your target image will be adjusted accordingly.

Once you build your project you will get the following image files:

- `<install path>/rootfs` – directory, the root file system that will be deployed on your board
- `<install path>/rootfs.ext2.gz.uboot` – ramdisk file system that can be flashed to your board
- `<install path>/rootfs.jffs2` – JFFS2 file system that can be flashed to your board
- `<install path>/rootfs/boot/uImage` – kernel image that can be loaded with U-Boot

If you want to fully reconfigure and recompile all the packages, you can do the following (this is generally not necessary):

1. Clean up all the configure files and objects thoroughly:
   `./ltib –m distclean`

2. You will be prompted to confirm your choice. Type `yes` to perform a distclean.

3. Run litb
```
./ltib
```

More information on LITB can be found in <*install path*>`/ltib/doc` or on the web at http://savannah.nongnu.org/projects/ltib.

4 Target Configuration

4.1 Supported Target Revisions

The target system is the MPC8360E-RDK board. This BSP is known to work on the following board(s) revision(s):

PIB: Prototype
PB: MPC8360E-RDK; PN 1008393
LCD: LCD-10.4-VGA_10; PN 80000174

4.2 Target System Memory Map

After system startup, the bootloader maps system memory as shown below.

Start	End	Definition
0x00000000	0x0fffffff	DDR2 SDRAM
0x60000000	0x63ffffff	NAND
0x80000000	0x9fffffff	PCI memory
0xe0000000	0xe01fffff	IMMR
0xff800000	0xffffffff	8MB flash

The flash starts at address **0xff800000**.

Start	End	Definition
0xff800000	0xff800100	HRCW
0xff800100	0xff83ffff	U-Boot
0xff840000	0xff85ffff	U-Boot environment
0xff860000	0xffbdffff	Linux kernel
0xff9c0000	0xfffdffff	Empty
0xfffe0000	0xffffffff	Device Table Binary

The NAND flash is not directly accessible; therefore, it does not reside in the MPC8360E-RDK address space. Instead, code in U-Boot (and the Linux kernel) accesses NAND using the Local Bus Controller via Chip Select 1 and UPMB. The addresses below are offsets from the start of the NAND chip.

Start	End	Definition
0x00000000	0x009fffff	Root file system
0x00a00000	0x03ffffff	Empty

4.3 Target Set-up

1. Connect your board to the network via the UCC1(J30) port.
2. Connect your board to your host machine via the serial port(P1A).
3. Set the terminal port, baud rate to 115200bps, data to 8 bit, parity to none, stop to 1 bit, and flow control to none.
4. Connect your board to the power supply; then power on your board.
5. The U-Boot terminal should show clock and memory settings, make sure the values are correct.

Clock	Value (MHz)
CSB	333
Core	660
QE	495
DDR	333
Memory	**Value (MB)**
DDR	256
NAND	64
Flash	8

Table 4.1: MPC8360E-RDK clock and memory settings

5 Target Deployment

5.1 Host Set-up

Proper host set-up is critical for your BSP to function appropriately. The host must be running tftp for deployment to work. For deployments using an NFS root file system, make sure the host is running NFS. The following instructions are generic. Your system may be different and the commands should be adjusted accordingly.

1. Turn off any firewalls for tftp to work. `iptables -F` or type "`setup`" at the command line.

2. Install tftp-server

3. Install nfs-server

4. Create the tftboot directory.
   `mkdir /tftpboot`

5. Link `rootfs` to an exportable directory once you have built your project.
   `ln –s <install_path>/ltib/rootfs /tftpboot/ltib`

6. Copy over kernel, bootloader, and flash file system images for your deployment to the `/tftpboot` directory
   ```
   cp <install_path>/ltib/rootfs/boot/* /tftpboot
   cp <install_path>/ltib/<flashfs> /tftpboot
   cp <cd mount point>/bootloaders/* /tftpboot
   ```

7. Edit `/etc/exports` and add the following line:
   `/tftpboot/ltib/ <target board IP>(rw,no_root_squash, async)`

8. Edit `/etc/xinetd.d/tftp` to enable tftp like this:
   ```
   {
   disable = no
   socket_type = dgram
   protocol = udp
   wait  = yes
   user  = root
   server = /usr/sbin/in.tftpd
   server_args = /tftpboot
    }
   ```

9. Restart the nfs and tftp servers on your host
   ```
   /etc/init.d/xinetd restart
   /etc/init.d/nfsserver restart
   ```

10. Connect board to the network.

11. Connect the target to the host via a serial connection.

12. Start `minicom` and set it up to talk to the MPC8360E-RDK board.
    - Serial Set-up: Select the correct serial device; Hardware & Software Flow control = No; bps = 115,200

- Modem & dialing: Delete text for the following: Init String, Reset String, Hang-up String, No flow control

13. Power-on board and see the console prompt.

5.2 Flashing U-Boot

To boot Linux 2.6.19+ kernels, it is **required** to update U-Boot to version 1.1.6. A U-Boot 1.1.6 image is located on your CD at `/images/u-boot.bin` or <*`install_path`*>`/ltib/rootfs/boot/u-boot.bin`. In addition, a Linux kernel image and root file system images are located on your CD in the `/images` directory.

5.2.1 Programming U-Boot using U-Boot

You can use U-Boot to upgrade the version of U-Boot. To accomplish this upgrade, you will need to download the U-Boot image to the DDR RAM on the board via tftp. See the instructions below.

Important Note: If these instructions are not followed exactly, the process can cause the MPC8360E-RDK to fail to boot. If this occurs, your only option is to reprogram U-Boot using a JTAG debugger.

```
=> tftp 0x400000 /tftpboot/u-boot.bin

=> protect off 1:0-1

=> erase 1:0-1

=> cp.b 0x400000 0xff800000 <u-boot size in hex>

=> protect on 1:0-1
```

5.2.2 Programming U-Boot using an Abatron BDI-2000

To program the MPC8360E-RDK using an Abatron BDI-2000:

1. Copy the u-boot.bin file to the TFTP server `/tftpboot` directory used by the Abatron

2. Modify the Abatron configuration file (in the `/tftpboot` directory on the TFTP server used by the Abatron) and set the RCW to `0xB0600004 0x0A04000F`

3. Attach the Abatron BDI-2000 to the MPC8360E-RDK

4. Power-on the MPC8360E-RDK board

5. Telnet into the Abatron and execute the following commands:

```
BDI> reset halt

BDI> unlock 0x00000

BDI> unlock 0x20000

BDI> erase 0x00000
```

```
BDI> erase 0x20000

BDI> burn 0x0 u-boot.bin bin

BDI> reset run
```

5.3 Configuring U-Boot

5.3.1 NFS file system

```
=>setenv ipaddr 10.193.20.177

=>setenv serverip 10.193.20.88

=>setenv gatewayip 10.193.20.254

=>setenv bootfile uImage

=>setenv loadaddr 0x200000

=>setenv ftdfile mpc8360ece.dtb

=>setenv ftdaddr 0x400000

=>setenv nfsboot 'setenv bootargs root=/dev/nfs rw
nfsroot=$serverip:/tftpboot/$ipaddr
ip=$ipaddr:$serverip:$gatewayip:255.255.255.0:mpc8360e:eth0:off
console=ttyS0,115200;tftp $loadaddr $bootfile;tftp $ftdaddr
$ftdfile;bootm $loadaddr - $ftdaddr'

=>setenv bootcmd 'run nfsboot'

=>saveenv
```

5.3.2 Ramdisk file system

```
=>setenv ipaddr 10.193.20.177

=>setenv serverip 10.193.20.88

=>setenv gatewayip 10.193.20.254

=>setenv loadaddr 0x200000

=>setenv bootfile uImage

=>setenv ftdfile mpc8360ece.dtb

=>setenv ftdaddr 0x400000

=>setenv ramdiskfile rootfs.ext2.gz.uboot

=>setenv ramdiskaddr 0x1000000
```

```
=>setenv ramboot 'setenv bootargs root=/dev/ram console=ttyS0,115200
ramdisk_size=240000;tftp $loadaddr $bootfile;tftp $ramdiskaddr
$ramdiskfile;tftp $ftdaddr $ftdfile;bootm $loadaddr $ramdiskaddr
$ftdaddr'

=>setenv bootcmd 'run ramboot'

=>saveenv
```

5.3.3 JFFS2 file system

```
=>setenv ipaddr 10.193.20.177

=>setenv serverip 10.193.20.88

=>setenv gatewayip 10.193.20.254

=>setenv flashkerneladdr 0xff860000

=>setenv flashftdaddr 0xfffe0000

=>setenv romboot 'setenv bootargs root=/dev/mtdblock4 rootfstype=jffs2
rw console=ttyS0,115200;bootm $flashkereneladdr - $flashftdaddr'

=>setenv bootcmd 'run romboot'

=>saveenv
```

5.4 NFS Deployment

1. Copy the kernel image from <*install_path*>/ltib/rootfs/boot/uImage to the /tftpboot directory created during host set-up.

2. Copy the file <*install_path*>/ltib/rootfs/boot/mpc8360ece.dtb to the /tftpboot directory created during host set-up.

3. At the u-boot prompt, use printenv to ensure target IP address, tftp server IP address, MAC address, bootfile, loadaddr, ftdfile, and fdtaddr are set properly.
   =>printenv -- Lists U-Boot environment settings

4. If the settings are not correct, use setenv to set them or type help at the u-boot prompt for other options. Use saveenv to save any changes you made to the U-Boot environment.

5. Make sure the U-Boot configuration is okay, see section 5.3.1.

6. Download the Linux kernel binary to DDR RAM.
   =>tftp $loadaddr /tftpboot/uImage

7. Download the Device Table Binary to DDR RAM.
   =>tftp $ftdaddr /tftpboot/mpc8360ece.dtb

8. To boot Linux, issue the following U-Boot command:
   =>bootm $loadaddr - $ftdaddr

5.5 Ramdisk Deployment

Note that the default ramdisk is too large to fit in the on-board 8MB NOR flash and the Linux kernel does not have support for loading a ramdisk out of NAND. Because of these limitations, the ramdisk has to be loaded into RAM via TFTP each time you want to boot a kernel using a ramdisk.

1. Copy the kernel image from <*install_path*>/ltib/rootfs/boot/uImage to the /tftpboot directory created during host set-up.

2. Copy the file <*install_path*>/ltib/rootfs/boot/mpc8360ece.dtb to the /tftpboot directory created during host set-up.

3. At the u-boot prompt, use printenv to ensure target IP address, tftp server IP address, MAC address, bootfile, loadaddr, ftdfile, ftdaddr, ramdiskaddr, and ramdiskfile are set properly.
```
=>printenv
```

4. Copy the ramdisk image from <*install_path*>/ltib/rootfs.ext2.gz.uboot to the /tftpboot directory created during host set-up.

5. If the settings are not correct, use setenv to set them or type help at the u-boot prompt for other options. Use saveenv to save any changes you made to the U-Boot environment.

6. Make sure the U-Boot configuration is okay, see section 5.3.2.

7. Download the Linux kernel binary to RAM.
```
=>tftp $loadaddr /tftpboot/uImage
```

8. Burn the Linux kernel into flash. The 'flinfo' command can be used to show the addresses that correspond to "1:3-14" as in the third through fourteenth flash sectors of the first NOR flash device.
```
=>protect off 1:3-14
=>erase 1:3-14
=>cp.b $loadaddr 0xff860000 <kernel size in hex>
=>protect on 1:3-14
```

9. Download the Device Table Binary to RAM.
```
=>tftp $loadaddr /tftpboot/mpc8360ece.dtb
```

10. Burn the Device Table Binary file into flash.
```
=> cp.b $loadaddr 0xfffe0000 <Device Table Binary size in hex>
```

11. Download the Ramdisk file system image to RAM.
```
=>tftp $ramdiskaddr /tftpboot/rootfs.ext2.gz.uboot
```

12. To boot Linux, issue the following U-Bboot command:
 if the kernel is not burned into flash:
```
=>bootm 0x20000 $ramdiskaddr 0xfffe0000
```

 if the kernel is in flash:
```
=>bootm 0xff860000 $ramdiskaddr 0xfffe0000
```

5.6 JFFS2 Deployment

Note that the JFFS2 root file system image is too large to place in the on-board 8MB NOR flash. Instead, the file system is installed within the first 10MB of the 64MB NAND device. This leaves the remaining 54MB of space in the NAND empty and available to the Linux kernel as `/dev/mdtblock5`.

1. Copy the kernel image from `<install_path>/ltib/rootfs/boot/uImage` to the `/tftpboot` directory created during host set-up.

2. Copy the file `<install_path>/ltib/rootfs/boot/mpc8360ece.dtb` to the `/tftpboot` directory created during host set-up.

3. At the `u-boot` prompt, use `printenv` to ensure target IP address, tftp server IP address, MAC address, bootfile, loadaddr, ftdfile, and ftdaddr are set properly.
```
=>printenv
```

4. Copy the JFFS2 image from `<install_path>/ltib/rootfs.jffs2` to the `/tftpboot` directory created during host set-up.

5. If the settings are not correct, use `setenv` to set them or type `help` at the `u-boot` prompt for other options. Use `saveenv` to save any changes you made to the U-Boot environment.

6. Make sure U-Boot configuration is okay, see section 5.3.3.

7. Download the Linux kernel binary to RAM using the following:
```
=>tftp $loadaddr /tftpboot/uImage
```

8. Burn the Linux kernel into flash. The 'flinfo' command can be used to show the addresses that correspond to "1:3-14" as in the third through fourteenth flash sectors of the first NOR flash device.
```
=>protect off 1:3-14
=>erase 1:3-14
=>cp.b $loadaddr 0xff860000 <kernel size in hex>
=>protect on 1:3-14
```

9. Download the Device Table Binary file to RAM.
```
=>tftp $loadaddr /tftpboot/mpc8360ece.dtb
```

10. Burn the Device Table Binary into flash.
```
=>protect off 1:63-66
=>erase 1:63-66
=>cp.b $loadaddr 0xfffe0000 <device table binary size in hex>
=>protect on 1:63-66
```

11. Download the JFFS2 file system image to RAM using the following:
```
=>tftp 0x1000000 /tftpboot/rootfs.jffs2
```

12. Burn the JFFS2 image to NAND flash.
```
=>nand erase 0 0xa00000
=>nand write.jffs2 0x1000000 0 <JFFS2 size in hex>
```

13. To boot Linux, issue the following U-Boot command:
```
=>bootm 0xff860000 – 0xfffe0000
```

5.7 Booting Linux 2.6.11 Kernels with U-Boot 1.1.6

To boot a Linux kernel prior to version 2.6.19 (version 2.6.11 for example) while using U-Boot 1.1.6, omit the third parameter in the ‘bootm’ command. This omission is required because the pre-2.6.19 kernels do not use an Open Firmware device table.

As an example, to boot a Linux 2.6.11 kernel with an NFS or JFFS2 root file system, the ‘bootm’ command would look like:

```
=> bootm $loadaddr –
```

To boot using a ramdisk, the ‘bootm’ command would look like:

```
=> bootm $loadaddr $ramdiskaddr
```

6 Revision History

REV	EDITOR	REVISION DESCRIPTION	APPROVAL	DATE
1	Jed Anderson, Peter Barada	First Release	ME	09/26/07
2	Peter Barada	-Add information for U-Boot 1.1.6 and Linux 2.6.19 (specifically DTB) -Section 4.2: addition of Device Table Binary to system memory map -Sections 5.3–5.6: changes for U-Boot 1.1.6 and Linux 2.6.19	PB	12/06/07
A	Peter Barada	-Make updating to U-Boot 1.1.6 a requirement -Section 5.2: Add information for where U-Boot, Linux, and root file system images can be found on CD -Add Section 5.7 for booting pre-2.6.19 Linux kernels -Remove “Preliminary” markings from document	PB	01/02/08

Please check www.logicpd.com for the latest revision of this manual, product change notifications, and additional application notes.