Freescale Semiconductor
Application Note

Document Number: AN3220
Rev. 0, 01/2006

U-Boot Porting Guide

(MPC5200B)

by: Lamar Hansford
Infotainment, Multimedia, and Telematics Division

This document outlines the procedures for porting the U-boot bootloader to newly implemented hardware systems using the Freescale MPC5200B. The MPC5200B microcontroller is based on an e300 core using the PowerPC™ instruction set.

Table of Contents

1 U-Boot: What is it? 1
2 File Organization 1
2.1 The Include Directory 2
2.2 The Board Directory 3
2.3 The CPU Directory 3
2.4 The lib_ppc Directory. 3
3 Step-By-Step Configuration for New Systems. . . 4
3.1 Configuring <board>.h. 4

1 U-Boot: What is it?

U-boot is a multi-functional open-source bootloader that may be obtained from the following website:

http://sourceforge.net/projects/u-boot

The U-boot bootloader allows the developer to either bootload an operating system from a variety of devices, or utilize a low level platform for testing hardware device settings.

For instructions on downloading and building U-boot, please see the instructions at the U-boot website above.

2 File Organization

After installation, the U-boot installation directory will appear as follows:

freescale™ semiconductor

```
u-boot/dtt
u-boot/net
u-boot/rtc
u-boot/disk
u-boot/post
u-boot/board
u-boot/tools
u-boot/drivers
u-boot/lib_arm
u-boot/lib_ppc
u-boot/lib_microblaze
u-boot/lib_generic
u-boot/common
u-boot/lib_i386
u-boot/lib_m68k
u-boot/lib_mips
u-boot/lib_nios
u-boot/examples
u-boot/include
u-boot/lib_nios2
```

Affected directories in porting a new MPC5200B design are:

- include
- board
- cpu
- lib_ppc

2.1 The Include Directory

The *include* directory contains all header files used globally by U-boot. When porting new processors and boards, the developer should ensure that a header file exists for the CPU under development, along with a board configuration file.

When compiling U-boot to support the Lite5200B evaluation kit that utilizes the MPC5200B, two header files must exist, as follows:

- MPC5xxx.h
- Configs/lite5200b.h

These files are specified in the top level makefile when the configuration is first created. The makefile is called the *mkconfig* routine, specifying ARCH, CPU, and BOARD.

2.1.1 <cpu>.h

The <cpu>.h file contains processor-specific definitions required for U-boot to access the CPU. This includes register definitions for both ANSI-C and PowerPC™ assembly.

2.1.2 configs/<board>.h

The configs/<board>.h file includes the U-boot software modules to be built, including hardware configuration for memory map and peripherals.

2.2 The Board Directory

An individual directory for each board configuration supported by U-boot is found in the *board* directory. This sub-directory includes board-specific configuration routines such as:

- Flash initialization
- DRAM configurations
- External device configurations such as ATA, PCI, FPGAs, etc.

2.3 The CPU Directory

The *CPU* directory contains initialization routines for internal peripherals. These include:

- Flash configuration
- Chip select configuration
- Low level serial drivers
- Low level USB driver

2.4 The lib_ppc Directory

The *lib_ppc* directory contains the board.c file, that performs the following functions:

- Memory map initialization including:
  - Logical memory map configuration
  - Dynamic memory allocation routines (malloc)
- Cache initialization

- Watchdog initialization
- U-boot peripheral initialization

3 Step-By-Step Configuration for New Systems

When designing a new system with the MPC5200B, follow these steps:

1. Define a physical memory map with the following sections
   - Flash memory
   - Dram memory
   - Peripheral memory space (Starts at 0xF0000000)
2. Define a memory link map with the following sections
   - .text – Program executable and data sections of U-boot
   - .env – Reserve space for environmental variables
   - Global data – This data is reserved for global variables
   - Monitor – Reserved space in ram for the U-boot bootloader
   - Malloc – Reserve space for dynamically allocated memory
   - Bootmap – Reserve space for operating system image
3. Configure U-boot by modifying the <board>.h header file

3.1 Configuring <board>.h

The <board>.h file contains #defines that controls the memory management, software modules, and parameters used by configuration routines in U-boot.

For new designs using the MPC5200B, developers should create a new header file based on the include\configs\lite5200b.h file. Once accomplished, see the following sections that address the modifications required.

3.1.1 Hardware Configuration

3.1.1.1 MBAR Configuration

3.1.1.1.1 CFG_MBAR

The CFG_MBAR value should be set to 0xF0000000.

3.1.1.1.2 CFG_DEFAULT_MBAR

This value relates to the MBAR register at reset. It should reflect 0x80000000.

3.1.1.2 Flash Configuration

3.1.1.2.1 Local Plus Chip Selects

The following settings are used to configure proper chip select settings:

- CFG_BOOTCS_START
- CFG_BOOTCS_SIZE
- CFG_BOOTCS_CFG
- CFG_CS0_START
- CFG_CS0_SIZE
- CFG_CS1_START
- CFG_CS1_SIZE
- CFG_CS1_CFG
- CFG_CS_BURST
- CFG_CS_DEADCYCLE

3.1.1.2.2 CFG_FLASH_BASE

CFG_FLASH_BASE value defines the flash base address used to configure chip select settings for false in the Local Plus controller.

3.1.1.2.3 CFG_FLASH_SIZE

CFG_FLASH_SIZE value indicates the size of the entire flash memory space. If present, the value includes multiple flash chips.

3.1.1.2.4 CFG_MAX_FLASH_BANKS

CFG_MAX_FLASH_BANKS value indicates the number of flash chips present in the system and is used to determine chip select configuration in the Local Plus controller.

3.1.1.2.5 CFG_MAX_FLASH_SECT

CFG_MAX_FLASH_SECT is the number of flash sectors in each chip used during programming of the flash chip.

3.1.1.2.6 CFG_FLASH_SECT_SIZE

CFG_FLASH_SECT_SIZE is the size of each flash sector used during programming of the flash chip.

3.1.1.3 DRAM Configuration

DRAM settings are used to configure chip select and vendor-specific DRAM settings.

3.1.1.3.1 CFG_SDRAM_BASE

Configure the SDRAM base address using this definition.

3.1.1.3.2 CFG_DRAM_CHIP_SIZE

Configure the size of each SDRAM chip.

3.1.1.3.3 CFG_DRAM_TOTAL

Configure the size of the total SDRAM memory space.

3.1.1.3.4 Vendor-Specific DRAM Configuration

These settings are used by the *sdram_start* routine located in the *board/<board>* directory. These settings are used to program the vendor-specific modes. The actual routine may require adjustment for different DRAM memories. The settings are written before DRAM is initialized. These stagings include:

- SDRAM_MODE
- SDRAM_EMODE
- SDRAM_CONTROL
- SDRAM_CONFIG1
- SDRAM_CONFIG2
- SDRAM_TAPDELAY

3.1.1.4 Internal SRAM

Internal SRAM settings are used to configure an initial software stack so C-routines can be used during initial boot configuration. These settings are related to the MPC5200B and MBAR; therefore, they should not be changed.

- CFG_INIT_RAM_ADDR
- CFG_INIT_RAM_END

3.1.1.5 CFG_IPBSPEED_133

The IPBus speed should be configured to 133. Changing the setting impacts flash timing.

3.1.1.6 CFG_XLB_PIPELINING

This value should be set to 1 for optimal throughput.

3.1.1.7 CFG_GPS_PORT_CONFIG

The GPIO pin configuration is configured here. The MPC5200B requires special GPIO configuration if two DRAM chips are to be used.

3.1.2 Software Configuration

3.1.2.1 Logical Memory Configuration

U-boot uses macros to locate the logical memory spaces described in Section 3.1, “Configuring <board>.h,” on page 4. Memory is allocated from the top of SDRAM to the bottom in the following order:

1. Text (U-boot)
2. Malloc
3. Global data + board initialization data structure
4. Stack

The space reserved for environmental settings must be placed into memory with the CFG_FLASH_BASE macro.

Memory allocation is achieved using settings described in the following sections.

3.1.2.1.1 U-Boot Monitor Space

CFG_MONITOR_BASE defines the base address for U-boot in SDRAM.

3.1.2.1.2 Global Data

CFG_GBL_DATA_SIZE defines the amount of space reserved for global data.

3.1.2.1.3 Malloc

CFG_MALLOC_LEN defines the size of memory allocated for heap operations.

3.1.2.1.4 Stack Space

Stack space grows down from allocated memory; it is not bounded.

3.1.2.1.5 Environment

Configuration environment can be placed anywhere in memory using the following settings:

- CFG_ENV_ADDR
- CFG_ENV_SIZE

3.1.2.1.6 Operating System Space

Operating system space is reserved through the CFG_BOOTMAPSZ definition.

3.1.2.2 CONFIG_COMMANDS

This setting selects what software module is to be compiled into U-boot. The available software commands can be found in the *include/cmd_confdefs.h* header file. Some options include:

- bdi info
- loads
- loadb
- iminfo
- icache, dcache
- flinfo, erase, protect
- md, mm, nm, mw, cp, cmp,
- crc, base, loop, mtest
- bootp, tftpboot, rarpboot
- saveenv
- kgdb
- PCMCIA support
- IDE harddisk support
- pciinfo
- irqinfo
- bootd
- coninfo
- EEPROM read/write support
- ask for env variable
- run command in env variable
- echo arguments
- I^2C serial bus support
- Register dump
- IMMR dump support
- support for RTC, date/time...
- DHCP support
- Include BedBug debugger
- Floppy disk support
- SCSI support
- Autoscript support
- MII support
- DCR support on 4xx

- ELF (VxWorks) load/boot cmd
- Misc functions, like sleep
- USB support
- Disk-on-chip support
- JFFS2 support
- Digital therm and thermostat
- SDRAM DIMM SPD info printout
- Diagnostics
- FPGA configuration support
- RTS/CTS hw flow control
- save S record dump
- SPI utility
- Floppy DOS support
- VFD support (TRAB)
- NAND support
- BMP support
- Port I/O
- ping support
- MMC support
- FAT support
- List all found images
- Integer (and string) test
- NFS support
- Reiserfs support
- Cisco discovery protocol
- Load part of multi image
- Tundra universe support
- EXT2 support

3.1.2.3 CONFIG_PSC_CONSOLE

Settings in the CONFIG_PSC_CONSOLE section allow control over which PSC is configured as console and the default baud rate settings of the port.

3.1.2.4 CONFIG_PCI

When enabled, this #define controls configuration of the PCI device. When CONFIG_PCI is undefined, the PCI software module will not be compiled into the image.

3.1.2.4.1 ADD_PCI_CMD

ADD_PCI_CMD value should be set to CFG_CMD_PCI to add PCI command support to the U-boot prompt.

3.1.2.5 Partitions

Supported partitions should be added here. The Lite5200B supports:

- MAC
- DOS
- ISO

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

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)
support@freescale.com

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 Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN3220
Rev. 0, 01/2006

freescale™ semiconductor