Release Notes of the QNX 6.4.0 BSP for Freescale MPC8349E MDS Trunk#

System requirements#

Target system

- QNX Neutrino RTOS 6.4.0
- Board version: Freescale MPC8349E-MDS (CPU Rev1.1 or higher) with PQ-MDS-PIB and PQ-MDS-PMCPCI card adaptor
- 8 MB flash
- 256 MB DDR RAM
- ROM Monitor version U-Boot 1.1.3

Host development system

- QNX Momentics 6.4.0
- Terminal emulation program (Qtalk, Momentics IDE Terminal, tip, HyperTerminal, etc.)
- RS-232 serial port and Straight-through serial cable
- Ethernet link

System Layout#

The tables below depict the memory layout for the image and for the flash.

Item	Address
OS image loaded at:	0x00800000
OS image begins execution at:	0x00200000
Flash base address	0xfe000000
(IPL+ OS image) base address in flash	0xfe700000

Getting Started#

Starting Neutrino#

Step 1: Build the BSP
You can build a BSP OS image from the source code. For instructions about building a BSP OS image, please refer to the chapter Working with a BSP in the Building Embedded Systems manual.

Step 2: Connect your hardware
1.Connect the serial cable to the first serial port of the MPC8439E MDS CPU card to the first serial port of your host machine (e.g. ser1 on a Neutrino host).

Note: If you have a Neutrino host with a serial mouse, you may have to move the mouse to the second serial port on your host, because some terminal programs require the first serial port.

On your host machine, start your favorite terminal program with these settings:

- Baud: 115200

- Bits: 8
- Stop bits: 1
- Parity: none
- Flow control: none

Then, apply power to the target. You should see output similar to the following:

```
U-Boot 1.1.3 (FSL Development) (Apr  6 2006 - 20:19:16) MPC83XX

Clock configuration:
  Coherent System Bus:  330 MHz
  Core:                 495 MHz
  Local Bus Controller: 330 MHz
  Local Bus:             82 MHz
  DDR:                  330 MHz
  SEC:                  110 MHz
  I2C1:                 330 MHz
  I2C2:                 330 MHz
  TSEC1:                330 MHz
  TSEC2:                330 MHz
  USB MPH:              110 MHz
  USB DR:               110 MHz
CPU: MPC8349E, Rev: 11 at 495 MHz
Board: Freescale MPC8349ADS
I2C:   ready
DRAM:
   SDRAM on Local Bus: 64 MB
   DDR RAM: 256 MB
FLASH:  8 MB
In:    serial
Out:   serial
Err:   serial
Net:   Freescale TSEC0: PHY is Marvell 88E1111 (1410cc2)
Freescale TSEC1: PHY is Marvell 88E1111 (1410cc2)
Freescale TSEC0, Freescale TSEC1
Hit any key to stop autoboot:  0
=>
```

Step 3: Setup the environment

On your target, type the following, filling in the appropriate IP addresses and ifs file:

```
=> setenv ipaddr 192.168.200.2
=> setenv serverip 192.168.200.1
=> setenv bootfile ifs-8349mds.raw
=> setenv loadaddr 0x200000
=> setenv bootcmd 'tftpboot $loadaddr $bootfile; go $loadaddr'
=> setenv bootdelay 2
=> saveenv
Saving Environment to Flash...
Un-Protected 2 sectors
Erasing Flash...
.. done
Erased 2 sectors
Writing to Flash...
256 KBdone
Protected 2 sectors
```

Step 4: Boot the IFS image

You can use TFTP download (the default) or serial download to transfer the image from your host to the target:

Step 4A: TFTP download

This method requires a raw image, which the buildfile creates by default.

Once the above setup is complete, you can run the load command at the => prompt to download the image:

```
=> boot
```

At this point you should see the ROM monitor download the boot image, indicated by a series of number signs. You'll also see output similar to this when it completes downloading:

```
Speed: 100, full duplex
Using Freescale TSEC0 device
TFTP from server 10.42.97.136; our IP address is 10.42.104.42
Filename '/root/ifs-8349mds.raw'.
Load address: 0x200000
Loading: ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##########################################
done
Bytes transferred = 1554228 (17b734 hex)
## Starting application at 0x00200000 ...

System page at phys:0000b000 user:0000b000 kern:0000b000
Starting next program at v00244860
Welcome to QNX Neutrino trunk on the Freescale MPC8349E-QS board
```

Step 4B: Using the QNX IPL code download

Note:

- This section assumes that the IPL has already been flashed into the CPU card. For more details on how to flash the IPL to the board, refer to the Creating a flash partition section.
- The IPL needs to boot high. This means that the position 8 of the SW4 switch needs to be ON.

This method requires an binary image. You have to modify the buildfile to create this format. Change this:

[virtual=ppcbe,raw]

to this:

[virtual=ppcbe,binary]

Rebuild the image.

After you apply power to the board, the IPL will run and you should see the following message on your host screen:

```
Welcome to QNX Neutrino on the Freescale MPC8349 MDS board
Hit 'd' to download a Neutrino image with sendnto
Press any other key to boot from flash
```

Press **d** to transfer the OS image over the serial link.

Note:
Pressing any other key would tell the IPL to boot OS image from flash. This assumes that the OS image has already been copied into the flash. For more details on how to flash the OS image into the board, refer to the Creating a flash partition section.

Copy the OS image from your host to the target using this command:

- On a Windows host: **`sendnto -b115200 -dCOM1 ifs-mpc8349mds.bin`**
- On a QNX Neutrino host: **`sendnto -b115200 -d/dev/ser1 ifs-mpc8349mds.bin`**
- On a Linux host: **`sendnto -b115200 -d/dev/ttyS0 ifs-mpc8349mds.bin`**

You should now see the Neutrino welcome message on your terminal screen:

```
Welcome to QNX Neutrino 6.3 on the Freescale MPC8349 MDS board
```

You can now test the OS simply by executing any shell builtin command or any command residing within the OS image (e.g. ls).

Once the initial image is running, you can update the OS image using the network and flash drivers. For sample command lines, please see the "Summary of driver commands" section.

Creating a flash partition#

WARNING:
Use extreme caution if you attempt to modify the contents on the flash. If critical contents (e.g. the IPL code at the bottom of the memory area) become damaged, QNX Neutrino may no longer boot from the flash.

You may be able to repair a flash by reprogramming it using an ICE or a flash programmer.

The **`mkflash`** script creates a 1MB flash image (in binary format) that you can download to the MPC8349 MDS board. This 1 MB image contains the following:

- configuration word
- reset code
- IPL
- OS image

Note:

- This leaves 7 MB of free space for a flash filesystem partition.
- 1 MB of space should be enough for the IPL, the reset code, the configuration word and a small OS image that starts the flash filesystem driver in order to pull the other drivers from flash.

Flash partitions#

To reformat the flash partition, do the following (for an 8 MB flash part):

1. Enter the following command to start the flash filesystem driver:

```
devf-generic -s0xfe000000,8M
```

2. To prepare the area for the filesystem partition, use the flashctl utility:

Note:

- In order to avoid erasing the IPL and OS image from the flash, use the -l ("el" -- length) option to specify the length of flash to erase. This leaves the top part of the flash intact with the IPL and OS image.
- In order to avoid erasing the ROM monitor (which is located in the first megabyte of flash), use the -o (offset) to erase starting at offset 1 MB (i.e. -o1M).

You can use both the **-o** and **-l** options at the same time. In the following example, we will erase the ROM monitor, but we will keep the IPL and OS image.

```
flashctl -p/dev/fs0 -l7M -ve
```

3. Format the partition:

```
flashctl -p/dev/fs0p0 -l7M -vf
```

4. Slay, then restart the driver.

These steps initialize the /dev/fs0p0 partition. The 8 MB flash partition is then as follows:

- **/dev/fs0p0** (mounted as /fs0p0):

A 7 MB fully initialized filesystem space which you can copy files to. For instance: **cp -v files /fs0p0/**

- **/dev/fs0p1**:

An unformated 1 MB flash partition in which you can copy the flash image generated by the mkflash script. For instance:**cp -v 1MB-flash-image /dev/fs0p1**

Driver Command Summary#

The following table summarizes the commands to launch the various drivers.

Component	Buildfile Command	Required Binaries	Required Libraries	Source Location
Startup	`startup-mpc8349e-qs`	.	.	`src/hardware/startup/boards/mpc8349e-qs`
Serial	`devc-ser8250 -e -F -c330000000 -b115200 0xe0004500,0x09 0xe0004600,0x0A`	devc-ser8250	.	`src/hardware/devc/ser8250`
Flash (NOR)	`devf-generic -s0xfe000000,8M`	devf-generic flashctl	.	`src/hardware/flash/boards/generic`
PCI	`pci-mpc83xx`	pci-mpc83xx pci	.	`src/hardware/pci/mpc83xx`

Network	`io-pkt-v4-hc -dmpc85xx mac=xxxxxxxxxxxx -ptcpip`	io-pkt-v4-hc ifconfig `x,verbose`	devnp-mpc85xx.so libsocket.so	"Binary form only:" `prebuilt/ppcbe/lib/dll/devnp-mpc85xx.so`
Network:MPC Security Engine (AKA SEC)	`io-pkt-v4-hc -dmpcsec -p tcpip-v6 ipsec -dmpc85xx mac=001100111111`	io-pkt-v4-hc ifconfig	devnp-mpc85xx.so devnp-mpcsec.so libsocket.so	"Binary form only:" `prebuilt/ppcbe/lib/dll/devnp-mpcsec.so`
I2C	`For I2C interface 1: i2c-mpc5200 -p0xe0003000 -c88000000 -i14 For I2C interface 2: i2c-mpc5200 -p0xe0003100 -c88000000 -i15`	i2c-mpc5200	.	`src/hardware/i2c/mpc8520`
SPI	`spi-master -d mpc8349 base=0xe0000020,irq=16`	spi-master	spi-mpc8349.so	`src/hardware/spi/mpc8349`

Some of the drivers are commented out in the default buildfile. To use the drivers in the target hardware, you'll need to uncomment them in your buildfile, rebuild the image, and load the image into the board.

Serial:#

> **Note**:
> Please make ensure the -c parameter matches the Coherent System Bus clock frequency.

Network:#

without encryption:
`io-pkt-v4-hc -dmpc85xx mac=xxxxxxxxxxxx,verbose -ptcpip`

with encryption in software:
`io-pkt-v4-hc -p tcpip-v6 ipsec -dmpc85xx mac=00112233AABB`

with encryption in hardware (for MPC8572E):
`io-pkt-v4-hc -dmpcsec.so -p tcpip-v6 ipsec -dmpc85xx.so mac=00112233AABB`

> **Note**:
> The latest sources for `devnp-mpc85xx.so` and `devnp-mpcsec.so` are available from the networking project.

PCI:#

> **Note**:
> This driver must be started after the I2C driver is active.

The Freescale MPC8349E MDS CPU boards might come by default configured as a PCI device. This will cause the PCI server to not detect PCI devices connected to the bus. Use the following settings to configure the CPU board as a PCI host:

- SW3.4 set to 1 (OFF). The default is ON.
- SW6.7 set to 0 (ON). The default is OFF.

Leave every other switch at the factory-default positions.

Known Issues#

- In those instances where the the ROM monitor's MAC address is different from the one you pass in when running io-pkt, the host can cache the ROM monitor's address. This can result in a loss of connectivity.**`Workaround`**: If you need to specify a MAC address to io-pkt, we recommend that you use the same MAC address that the ROM monitor uses. This will ensure that if the host caches the ROM monitor's MAC address, you'll still be able to communicate with the target. Otherwise you might need to delete the target's arp entry on your host.

- The PCI driver is tested and works on MPC8349E REV 3.0 board, but doesn't work on REV 1.1 board.