Release Notes for the QNX Neutrino 6.4.1 BSP for Freescale P2020DS 1.0.0#

1.System Requirements#

Target Requirements

1. QNX Neutrino RTOS 6.4.1
2. Board version: P2020DS - Stingray
3. P2020 processor
4. 2GB DDR SDRAM
5. 128 MB NOR flash

Host Requirements

1. QNX Momentics 6.4.1
2. Truncated Message Copy Issue — SMP PowerPC Kernels Patch Patch ID 1636. Download link
3. Terminal emulation program (Qtalk, Momentics IDE Terminal, tip, HyperTerminal, etc.)
4. RS-232 serial port and serial cable, or a USB-to-serial cable
5. Ethernet link

2.System Layout#

Start	End	Item
0x00100000		OS Image Loaded
0x00000000	0x7FFFFFFF	RAM
0xC0000000	0xDFFFFFFF	PCIe1 Memory
0xA0000000	0xBFFFFFFF	PCIe2 Memory
0x80000000	0x9FFFFFFF	PCIe3 Memory
0xE0000000	0xE7FFFFFF	Promjet
0xE8000000	0xEFFFFFFF	Nor Flash (on eLBC)
0xF0000000	0xF000FFFF	PCIe1 IO
0xF0010000	0xF001FFFF	PCIe2 IO
0xF0020000	0xF002FFFF	PCIe3 IO
0xF0030000	0xF012FFFF	NAND Flash Bank 1 (eLBC Chip Select 2)
0xF0130000	0xF022FFFF	NAND Flash Bank 2 (eLBC Chip Select 4)
0xF0230000	0xF032FFFF	NAND Flash Bank 3 (eLBC Chip Select 5)
0xF0330000	0xF042FFFF	NAND Flash Bank 4 (eLBC Chip Select 6)
0xF0430000	0xF0437FFF	PIXIS Registers (eLBC Chip Select 3)

3.Getting Started#

3.1 Building the BSP#

You can build a BSP OS image from the source code or the binary components contained in a BSP package. For instructions about building a BSP OS image, please refer to the chapter Working with a BSP in the Building Embedded Systems manual.

3.2 Connect your Hardware#

Connect the serial cable to the first serial port of the P2020DS board to the first serial port of your host machine. There are 2 serial ports on P2020DS. Use the one which is near the boundary of the board. Usually you should see some Uboot output on the console when you connect cable to the correct port. 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.

The correct terminal settings of the program handling serial connection should be:

baudrate	115200
data	8 bit
parity	none
stop	1bit
flow control	none

3.3 Setup you environment#

1. Power on your target. You should see the u-boot output on your console. 2. Conenct an ethernet cable to any of the 3 PHY port available on the back side of the board.

4. Boot the IFS image#

You can use TFTP download (the default) or serial download to transfer an OS image to the board, as described below.

4.1 Boot via tftp#

This method requires that you put the raw image generated by BSP (by default at $BSP_ROOT/images/ifs-p2020ds.raw) to a TFTP server. This server must be reachable via board and preferably should be on the same LAN. As soon as u-boot starts, press any key so that u-boot stops and doesnt boot the prebuild linux kernel. Configure u-boot parameters as follows:

```
=> setenv ipaddr 10.90.74.214
=> setenv serverip 10.90.74.42
=> setenv bootfile ifs-p2020ds.raw
=> setenv loadaddr 0x100000
=> setenv bootcmd 'tftpboot $loadaddr $bootfile; go $loadaddr'
=> setenv bootdelay 2
=> saveenv
Saving Environment to Flash...
Un-Protected 1 sectors
Erasing Flash...
flash erase done
Erased 1 sectors
Writing to Flash... done
Protected 1 sectors
=> boot
```

4.1 Boot via serial#

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

[virtual=ppcbe-spe,raw] to this:

[virtual=ppcbe-spe,srec] Rebuild the image. On your target, type:

```
=>: setenv loads_echo 0
=>: saveenv
=>: loads
```

On your host, copy the image to the serial port that's connected to the board. For example, on a Neutrino host: cp ifs-p2020ds.srec /dev/ser1 On a Windows host, you can use Hyperterminal's transfer feature to copy the image as a text file.

```
## First Load Addr = 0x00100000
## Last  Load Addr = 0x0023955B
## Total Size      = 0x0013955C = 1283420 Bytes
## Start Addr      = 0x00101E38
=>:
```

Type **`go start_addr`**

At this point, you should see output similar to this when it finishes downloading:

```
## Starting application at 0x00100000 ...
Welcome to QNX Neutrino 6.4.1 on the PowerPC P2020DS board
#
```

Congratulations! QNX 6.4.1 kernel is running on your system. You can 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 " Driver Command Summary" section.

5. Writing the IPL and IFS images to flash using the boot loader#

P2020DS supports bank switching in the NOR Flash. i.e. The upper half of the flash can appear as the lower half and vice-versa. The upper half of the NOR flash has the U-Boot image in the last 1MB. Another boot loader image i.e. IPL can be placed in the last 64KB of the lower half of the NOR flash. So both the boot loader images, U-Boot and IPL can simultaneously be placed in the NOR flash.

The IPL image (ipl-p2020ds) which was built is an SREC image. We should convert it to a Binary image using the QNX utility ntoppc-objcopy. Run the following command at the same location where we ran the "make" command to build the IFS image:

```
ntoppc-objcopy --input-format=srec --output-format=binary install/ppcbe/boot/sys/ipl-p2020ds images/ipl-p2020ds.bin
```

On your target board, you can use U-BOOT to transfer an IPL image ipl-p2020ds.bin and an IFS image ifs-p2020ds.raw to the target board RAM, and then program it to the flash.

1.Use tftp to download the images:

```
=> tftp 0x100000 ifs-p2020ds.raw
```

```
eTSEC1: No link.
eTSEC2: No link.
Speed: 100, full duplex
Using eTSEC3 device
TFTP from server 10.90.74.42; our IP address is 10.90.74.210
Filename 'ifs-p2020ds.raw'.
Load address: 0x100000
Loading: #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         ########################################
done
Bytes transferred = 5201536 (4f5e80 hex)

=> tftp 0x600000 ipl-p2020ds.bin
Speed: 100, full duplex
Using eTSEC3 device
TFTP from server 10.90.74.42; our IP address is 10.90.74.210
Filename 'ipl-p2020ds.bin'.
Load address: 0x600000
Loading: #############
done
Bytes transferred = 65536 (10000 hex)
```

2.Use protect to turn off the flash protection:

```
=> protect off all
Un-Protect Flash Bank # 1
```

3.Use erase to erase the first 5 MB of the second half of the flash (for IFS image) (from 0xec000000 to 0xec4fffff):

```
=> erase 0xec000000 0xec4fffff

..... done
Erased 40 sectors
```

Though the IPL image is 64 KB, we erase 128 KB of the flash, because NOR flash can be erased only sector wise and each sector is 128 KB

4.Use erase to erase the last 128 KB of the first half of the flash (for IPL image) (from 0xebfe0000 to 0xebffffff):

```
=> erase 0xebfe0000 0xebffffff

. done
Erased 1 sectors
```

5.Use cp.b to program the IPL image and the IFS image to flash:

```
=> cp.b 0x100000 0xec000000 0x4f5e80
Copy to Flash... 9....8....7....6....5....4....3....2....1....done
=>
=> cp.b 0x600000 0xebff0000 0x10000
Copy to Flash... 9....8....7....6....5....4....3....2....1....done
```

6. Switch off the target and change the switch setting of SW7(2:3) from 00 to 01. This switch setting, bank switches the NOR flash.

Now the IFS image will be at the start of the flash i.e. at 0xE8000000 and the IPL image will be in the last 64 KB of the flash i.e. at 0xEFFF0000. Now switch on the target. IPL image will start booting from the NOR flash as below:

```
Welcome to QNX Neutrino IPL on the Freescale P2020DS Stingray board
Scanning for image       @ 0xE8000000
Found image              @ 0xE8000100
Jumping to startup       @ 0x00104A60
board_smp_init: 2 cpu
Looking for Config EEPROM on i2c,0 @ I2C address 0x00000057 ... found
Validating contents ... NOTE: CRC check disabled ... Ok
Looking for RTC on i2c,1 @ I2C address 0x00000068 ...
System page at phys:0000b000 user:0000b000 kern:0000b000
Starting next program at v00150e2c
Welcome to QNX Neutrino 6.4.1 on the PowerPC P2020DS board
#
#
```

6. Creating a flash partition#

Follow the description mentioned below for the NOR flash.

7. Driver Command Summary#

Component	Buildfile Command	Required Binaries	Required Libraries	Source Location
Startup	startup-p2020ds -v -c0xFFE00000 -t100000000 -D0xffe04500	startup-p2020ds	libstartup.a	src/hardware/startup/boards/p2020ds
Serial	devc-ser8250 -e -c500000000 -b115200 0xffe04500,25 waitfor /dev/ser1 reopen /dev/ser1	devc-ser8250	none	src/hardware/devc
USB	io-usb -d ehci-p2020 ioport=0xFFE22100,irq=15 & waitfor /dev/io-usb/io-usb 10 devb-umass&	devu-ehci-p2020.so	libusbdi.so io-blk.so io-usb usb devb-umass libcam.so fs-dos.so	"prebuiltonly"

			fs-qnx4.so fs-ext2.so cam-disk.so	
I2C	i2c-mpc8572 -i26 -p0xffe03000 (for controller 1) i2c-mpc8572 -i26 -p0xffe03100 --u1 (for controller 2)	i2c-mpc8572	none	src/hardware/i2c/mpc8572
SMP	PATH=:/proc/boot:/bin:/usr/bin LD_LIBRARY_PATH=:/proc/boot:/lib:/usr/lib:/lib/dll procnto-booke-smp -v	procnto-booke-smp	none	prebuilt
PCI	pci-p2020	pci-p2020	none	/src/hardware/pci
Network	io-pkt-v4 -d mpc85xx	io-pkt-v4 ifconfig	devnp-mpc85xx.so	/src/hardware/devnp
NOR Flash	devf-generic -s 0xE8000000,128M,,,128k,2,1 -r	devf-generic flashctl	libmtd-flash.a	/src/hardware/flash
RTC	rtc -v ds3232 rtc -s -v ds3232	rtc date	libutil.a libutilS.a	/src/utils/r/rtc

USB#

devb-umass& (This will exit if it doesn't find any mass storage device, so start this only after plugging in a device)

Network#

To start network driver without encryption, run:

`io-pkt-v4-hc -dmpc85xx`

you should see following output when you run ifconfig

```
# ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 33192
        inet 127.0.0.1 netmask 0xff000000
tsec0: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
        capabilities rx=7<IP4CSUM,TCP4CSUM,UDP4CSUM>
        capabilities tx=0
        enabled=0
        address: 00:04:9f:00:d8:fe
        media: Ethernet none
tsec1: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
        capabilities rx=7<IP4CSUM,TCP4CSUM,UDP4CSUM>
        capabilities tx=0
        enabled=0
        address: 00:04:9f:00:d8:ff
        media: Ethernet none
tsec2: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
        capabilities rx=7<IP4CSUM,TCP4CSUM,UDP4CSUM>
        capabilities tx=0
        enabled=0
```

```
address: 00:11:8c:84:00:11
media: Ethernet none
```

To bring a network interface up, type following command

```
ifconfig tsec0 10.90.74.214/24 up
```

Here 10.90.74.214 is the IP address assign to your target and 24 is the netmask bits (i.e. 255.255.255.0) The LEDs of the RJ45 ports corresponding to an enabled interface shall glow periodically. This can be used to identify the correct port corresponding to each tsec.

PCI#

run **pci-p2020** to enable PCI functionality on your board. This server supports both PCI and PCI-ex devices. To use a PCI or PCI-express device on P2020DS, the device should be connected to the slot before PCI server is run. To detect whether the connect device is detected successfully or not, run following command line utility

```
pci -v
```

This shall output all PCI devices currently detected on your system.

RTC#

The RTC used in P2020DS board is DS3232. Make sure the i2c driver is up and running before running the RTC utility, as the RTC DS3232 is connected to the second i2c bus.

If the i2c driver is not running, issue the following command

```
i2c-mpc8572 -i26 -p0xffe03100 --u1
```

To run the RTC utility, use the following command:

```
rtc -v ds3232
```

This command updates the current time and date from the hardware clock in the board.

To set the hardware clock in the board with the current date and time, use the following command:

```
rtc -s -v ds3232
```

NOR Flash#

Run

```
devf-generic -s 0xE8000000,128M,,,128k,2,1 -r
```

to run the generic Flash filesystem driver on your board. After running this command, two partitions will get created. Normally the file names are as below: /dev/fs0 which is the default mountpoint for socket 0 and /dev/fs0p0 which has the raw access for socket 0, partition 0.

After these partitions get created, we should erase and format the flash using the flashctl utility.

The commands are as follows:

```
flashctl -p /dev/fs0 -o 0 -l 127M -ev
```

This command erases the nor flash starting from an offset of 0 to 127MB.

Though the nor flash is 128MB, we erase only 127MB as the last 1MB of flash contains the u-boot image.

After giving the above command, slay the driver using the `slay devf-generic` command and then restart it again.

The Nor flash is erased now and we can mount the given flash filesystem partition as the filesystem mountpoint /flash using the command below:

```
flashctl -p /dev/fs0p0 -o 0 -l 127M -f -n /flash
```

After this command is successfully run, slay the driver and restart it again. This formats the nor flash and mounts it over the filesystem mountpoint /flash.

We can create multiple filesystem partitions based on our requirement.

8. Known Issues#

1. The P2020DS board has a K9WBG08U1M PCB0 , 4k page size flash device installed. This device is not supported by the p2020 and hence the BSP does not have Flash support. This is a board limitation.
2. errors while loading the BSP into the IDE. Workaround:
    - In IDE BSP perspective, open the System Builder Projects view, right click on the system builder project "bsp-freescale-p2020-ds", select "Properties", select "Search Paths" from the left panel, select "System Files" tab at the right panel, change the first path which has "install/PLATFORM/boot/sys" in it, change the PLATFORM variable to "ppcbe". Now select "DLLs" tab at the right panel, change the first path which has "install/PLATFORM/lib/dll" in it, change the PLATFORM variable to "ppcbe-spe". Rebuild the system builder project.
3. When using the QNX IPL, the PCI Server doesn't detect devices behind bridge. (PR:72703). The current workaround is to slay and rerun the PCI server.
4. if you are using 6.4.1 SDP with 4.6 IDE then for successful debugging using gdb download gdb update update 5 from http://community.qnx.com/sf/frs/do/viewRelease/projects.toolchain/frs.gdb.gdb_6_8_u5