Release Notes for the QNX Neutrino 6.4.0 BSP for TI OMAP-L138 EVM/TI AM1808#

1.System Requirements#

Target Requirements

1. QNX Neutrino RTOS 6.4.0
2. Board version: TI OMAP-L138 EVM or TI AM1808
3. L138 processor
4. 64MB DDR SDRAM
5. 8 MB SPI flash
6. 512 MB Micron NAND flash
7. 8 MB NOR flash

Host Requirements

1. QNX Momentics 6.4.0
2. Terminal emulation program (Qtalk, Momentics IDE Terminal, tip, HyperTerminal, etc.)
3. RS-232 serial port and serial cable, or a USB-to-serial cable
4. Ethernet link

2.Getting Started#

2.1 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.

2.2 Connect your Hardware#

Connect the serial cable to the serial port of the TI OMAP L138 EVM board to the serial port of your host machine. There is 1 serial port on TI OMAP L138 EVM board. Usually you should see some Uboot output on the console when you connect cable to the correct 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

2.3 Setup you environment#

1. Power on your target. You should see the u-boot output on your console. 2. Connect an ethernet cable to the Baseboard's port.

3. Boot the IFS image#

You can use TFTP download to transfer an OS image to the board, as described below.

3.1 Boot via tftp#

This method requires that you put the raw image generated by BSP (by default at $BSP_ROOT/images/ifs-omapl138.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-omapl138.raw
=> setenv loadaddr 0xC0008000
=> setenv bootcmd 'tftpboot $loadaddr $bootfile; go $loadaddr'
=> setenv bootdelay 2
=> saveenv
Saving Environment to SPI Flash...
Erasing SPI flash...Writing to SPI flash...done
=> boot
```

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

```
## Starting application at 0xC0008000 ...
Welcome to QNX Neutrino 6.4.0 on the TI OMAPL138EVM Board
#
```

Congratulations! QNX 6.4.0 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. Still the network and flash drivers need to be supported for the TI OMAP L138 EVM board.

4. Driver Command Summary#

Component	Buildfile Command	Required Binaries	Required Libraries	Source Location
Startup	startup-omapl138 -L 0xC3E00000,0x200000 -vv	startup-omapl138	libstartup.a	src/hardware/startup/boards/omapl138
Serial	devc-ser8250 -e -F -S -b115200 -c150000000/16 0x01d0d000^2,61 waitfor /dev/ser1 4 reopen /dev/ser1	devc-ser8250	none	src/hardware/devc
I2C	i2c-dm6446 -a0x1 -i15 -p0x01c22000 -f24000000 -l0x1000 --b10000 --u0 (for controller 1)	i2c-dm6446	libi2c-master.a	src/hardware/i2c/dm6446

	i2c-dm6446 -a0x2 -i51 -p0x01e28000 -f24000000 -l0x1000 --b10000 --u1 (for controller 2)			
USB	io-usb -dohci-omapl1xx ioport=0x01E25000,irq=59 -domapl1xx-mg ioport=0x01E00400,irq=58	devu-ohci-omapl1xx.so devu-omapl1xx-mg.so	libusbdi.so io-blk.so io-usb usb devb-umass libcam.so fs-dos.so fs-qnx4.so fs-ext2.so cam-disk.so	"prebuiltonly"
Network	io-pkt-v4 -ddm644x-omapl1xx -ptcpip	io-pkt-v4 ifconfig	devn-dm644x-omapl1xx.so	src/hardware/devn
RTC	rtc -s -v omapl1xx rtc -v omapl1xx	rtc date	libutil.a	src/utils/r/rtc
SPI	spi-master -d dm644x-omapl1xx base=0x01c41000,irq=20,clock=150000000,edma=0,en0def=1,spicntrlr=omapl1xx spi-master -u1 -d dm644x-omapl1xx base=0x01F0E000,irq=56,clock=150000000,edma=0,en0def=1,spicntrlr=omapl1xx	spi-master	spi-dm644x-omapl1xx.so	src/hardware/spi/dm644x
DMA			libdma-omapl1xx.so libdma-omapl1xxS.a	src/lib/dma/omapl1xx
NAND	fs-etfs-jacinto-omapl1xx -D addr=0x62000000,cs=3 -e fs-etfs-jacinto-omapl1xx -D addr=0x62000000,cs=3 -m /fs/etfs	fs-etfs-jacinto-omapl1xx etfsctl		src/hardware/etfs
NOR	devf-generic -s 0x60000000,8M	devf-generic flashctl		src/hardware/flash
DSPLink (l138 only)			libdsplink160-clientS.a	"prebuiltonly"
SATA	devb-ahci blk auto=partition dos exe=all qnx6 sync=optional cam ahci omapl138,ioport=0x01e18000,irq=67	devb-ahci	libcam.so cam-disk.so io-blk.so fs-dos.so fs-qnx4.so fs-ext2.so	"prebuiltonly"
MMCSD	devb-mmcsd-omapl1xx cam mmcsd bs="wp_gpio_bank=4 wp_gpio=1 ins_gpio_bank=4 ins_gpio=0 ins_gpio_irq=46"	devb-mmcsd-omapl1xx	libcam.so cam-disk.so io-blk.so fs-dos.so	src/hardware/devb/mmcsd

	verbose=3 blk cache=2M			
Graphics	io-display -dvid=0,did=0	io-display	libGLES_CL.so.1 libffb.so.2 libdisputil.so libgf.so.1 libimg.so.1 devg-omapl1xx.so img_codec_bmp.so img_codec_gif.so img_codec_jpg.so img_codec_png.so img_codec_sgi.so img_codec_tga.so libFF-T2K.so.2	src/hardware/devg

Serial#

To start the Serial Driver, run:

```
devc-ser8250 -e -F -S -b115200 -c150000000/16 0x01d0d000^2,61
```

RTC#

To set the RTC time as that of the System time, run:

```
rtc -s -v omapl1xx
```

To set the System time as that of the RTC time, run:

```
rtc -v omapl1xx
```

NAND Flash#

To erase the NAND flash present on the UI board, run:

```
fs-etfs-jacinto-omapl1xx -D addr=0x62000000,cs=3 -e
```

To mount the ETFS file system on a mount point, run:

```
fs-etfs-jacinto-omapl1xx -D addr=0x62000000,cs=3 -m /fs/etfs
```

NOR Flash#

Run

```
devf-generic -s 0x60000000,8M
```

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.

Note :Before Erasing or Programming the NOR flash, the NOR flash sector needs to be unlocked first.

Run the following commands before erasing or programming the NOR flash:

The following command unmounts the fs0p0 partition.

```
flashctl -p /dev/fs0p0 -o 0 -l 7M -u
```

This is because the -U (unlock) option will fail if the partition (eg: fs0p0) is mounted.

The following command unlocks the raw partition.

```
flashctl -p /dev/fs0 -o 0 -l 7M -U
```

The above sequence of commands is a must after every device reset (as the Intel Strata P30 NOR flash is locked after the Reset), if you need to erase or program the NOR flash. If you need to read from the NOR flash, the above sequence of commands is not needed.

The commands for Erase and Format are as follows:

```
flashctl -p /dev/fs0 -o 0 -l 7M -ev
```

This command erases the NOR flash starting from an offset of 0 to a length of 7MB.

Though the nor flash is 8MB, we erase only 7MB as the last 1MB of flash contains some OTP regions which cannot be erased and which make the erase fail.

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 7M -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.

Network#

To start network driver, run:

```
io-pkt-v4 -ddm644x-omapl1xx -ptcpip
```

Note : Both the Ethernet Ports (connected to the MII interface on the Base board and RMII interface on the UI board) work, but not simultaneously. This is a hardware limitation. By default, the Ethernet Port on the Base board (MII interface) will work. To make the Ethernet Port on the UI board (RMII interface) to work, build the BSP's Startup with the -U option. "startup-omapl138 -wt -L 0x80000000,0x20000 -vv -U"

you should see following output when you run ifconfig

```
# ifconfig en0
en0: flags=80008802<BROADCAST,SIMPLEX,MULTICAST,SHIM> mtu 1500
        address: 00:0e:99:02:f9:4e
        media: Ethernet 100baseTX
        status: active
```

To bring a network interface up, type following command

```
ifconfig en0 10.90.74.246/24 up
```

Here 10.90.74.246 is the IP address assigned to your target and 24 is the netmask bits (i.e. 255.255.255.0) The LEDs of the RJ45 port in the Base Board shall glow periodically.

SPI#

To start spi driver for spi0 interface, run:

```
spi-master -d dm644x-omapl1xx
base=0x01c41000,irq=20,clock=150000000,edma=0,en0def=1,spicntrlr=omapl1xx
```

you can see a file /dev/spi0 after the above step.

To start spi driver for spi1 interface, run:

```
spi-master -u1 -d dm644x-omapl1xx
base=0x01F0E000,irq=56,clock=150000000,edma=0,en0def=1,spicntrlr=omapl1xx
```

you can see a file /dev/spi1 after the above step.

For reading the on-board spi-flash you need to use /dev/spi1.

USB#

To start both USB OHCI (1.1) and USB 2.0 drivers, run:

```
io-usb -dohci-omapl1xx ioport=0x01E25000,irq=59 -domapl1xx-mg
ioport=0x01E00400,irq=58
```

wait for /dev/io-usb/io-usb

To list and mount mass storage device, run:

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

```
mount -t <file_system_name> /dev/<device_name> <path to be mounted at>
```

SATA#

To start the Serial ATA (SATA) Driver, run:

```
devb-ahci blk auto=partition dos exe=all qnx6 sync=optional cam ahci
omapl138,ioport=0x01e18000,irq=67
```

wait for /dev/hd0 device, if the SATA HDD has valid partitions already, then it will list them as /dev/hd0, /dev/hd0tnn (nn is the number) etc

mount the partition to a local folder, for example :

mount /dev/hd0t79 /tmp

MMCSD#

To start the MMCSD driver,run `devb-mmcsd-omapl1xx cam mmcsd bs="wp_gpio_bank=4 wp_gpio=1 ins_gpio_bank=4 ins_gpio=0 ins_gpio_irq=46" verbose=3 blk cache=2M`

NAND/NOR flash cannot be used simultaneously with MMCSD support. To enable mmcsd support ,build the BSP's with startup option -s

"startup-omapl138 -wt -L 0x80000000,0x20000 -vv -s"

GRAPHICS#

To start the Graphics driver,run `io-display -dvid=0,did=0`

Required config files: display.conf , img.conf, omapl1xx.conf

Raster controller: The Raster Controller has been validated with the Sharp LQ043T1DG01 TFT LCD. The Raster controller supports a single layer and runs only at the RGB565 pixel format.

Various Graphics applications like img_decode_simple, vsync, demo-alpha, demo-chroma can be tested using this Graphics Driver support.

Photon can be run and the io-graphics driver can be loaded, which can be later used for various applications like Terminal, Calculator etc.

5. Known Issues for This BSP#

1. Time in the On board RTC (omapl1xx) is reset to the default value, after a hard reset of the Board. This is because, there is no separate isolated power supply connected to the RTC on the board.
2. Time in the RTC is retained only in the case of a soft reset.
3. The network driver will not behave properly if issued with a flood ping of more than 50000 bytes (refer PR : 74266)
4. SATA Hot plug support is not there.
5. The SPI1_SCS0 is shorted with the LCD_PWM in the board, hence the SPI flash which is on the SPI_SCS0 and the LCD controller wont work simultaneously. Hence the Application which uses the SPI flash should bring back the line high after accessing the SPI flash to make the LCD Graphics controller to work.
6. The USB 2.0 driver gives a sloginfo warning " devu-omapl1xx-mg.so : MENTOR_ProcessArgs - Unknown option irq=58 " because of incorrect parsing by io-usb.