# Release Notes of the QNX 6.4.0 BSP for Centrality Titan EVB Trunk#

## System requirements#

### Target system#

- Centrality Titan Motherboard version 1.2.0
- CPU Core module board version 2.2.0 (with 64MB of RAM)
- CPU TT640
- Sharp 4.3" LCD panel (resolution: 800x480)
- ROM monitor: NBoot v5.1
- USB cable
- Serial cable (straight-through)
- One serial port (UART RS-232)
- 64 MB NAND flash (Samsung K9F1208U0M)
- DM9000A Ethernet

**Note**:If the Titan board is populated with more than 64 MB of RAM, you'll need to change the startup code (i.e. init_raminfo.c) to add the proper RAM size.

### Host development system#

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

## System Layout#

The table below depicts the memory layout for the image and for the on-board flash.

| Item | Address | |
|---|---|---|
| OS image loaded at: | 0xC0200000 | |
| OS image begins execution at: | 0xC0200000 | |
| IPL offset: | In on-board NAND flash, at EBoot location | |
| Ethernet base address | 0x14000000 (IRQ: 301) | |

## Getting Started#

### Step 1: Connect your hardware#

Connect the Centrality Titan board's power supply as per the instructions provided with the hardware. Connect the straight-through serial cable between the UART1 port (P23) on the Titan, and a free serial port on the host PC you wish to use.

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

Connect the supplied USB cable to the P20 connector (USB client) on the Centrality Titan board. Connect the other end of the USB cable to a free USB port on a Windows PC.

### Step 2: Build 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.

The pre-built IPL binary provided with the binary components needs to be padded to 16 KB before transferring it to the board via NBoot. Use the following script **`mkflashimage`** to pad the IPL binary to 16 KB:

```
#script to build a binary IPL for the Titan evaluation board
   set -v
   rm -f ipl-titan.bin
#convert IPL into an S-record
   ${QNX_HOST}/usr/bin/ntoarm-objcopy -Osrec ipl-titan ipl-tmp-titan.srec
#convert S-Record IPL to binary
   ${QNX_HOST}/usr/bin/ntoarm-objcopy -Obinary ipl-tmp-titan.srec ipl.tmp
   mkrec -r -ffull -s16k ipl.tmp &gt; ipl-titan.bin
#Clean up temporary files
   rm -f *tmp*
```

This step is necessary only when transferring the pre-built IPL via NBoot.

### Step 3: Transfer the IPL to the on-board NAND flash#

Before transferring the OS image to the board, you'll need to program the IPL to the flash.

The QNX IPL will replace EBoot. It is possible to update the IPL by using the AtlasMgr5.exe utility from Centrality.

### Step 3.1: Program the IPL#

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

- Baud: 38400
- Bits: 8
- Stop bits: 1
- Parity: none

The Titan board has a two-stage boot process. The first loader to run is called NBoot, which will present the user with a menu of further boot and configuration options. The second loader called EBoot won't be used. The following procedure will replace the EBoot loader completely with a QNX Initial Program Loader, which can then be used to load a QNX Neutrino OS image serially. To perform this procedure, you will need a Microsoft Windows Utility called AtlasMgr5.exe (for NBoot v5.1), available from Centrality. You will also need to install a USB device driver on the Windows HOST for the Titan board. This driver is also available from Centrality.

**Note**:The Centrality tools are usually provided with the Centrality Titan board.

### Step 3.2: Transfer the IPL#

1. Transfer the binary IPL, to a MS Windows HOST that is connected to the Titan board via the USB link. The binary IPL file is called ipl-titan.bin.

2. Apply power to the Centrality Titan board. On the serial console, you should see the following NBoot output when using NBoot v5.1:

```
Centrality NBOOT v5.1
Nov 14 2006 19:16:19
ClockInfo: 0x0

1): Launch EBOOT
2): Launch NK
3): Launch DM
4): Erase and Reset TOC
5): Toggle NBOOT Menu
6): Set Clock
7): Config CS
Input:
```

3. Using NBoot version 5.1:

- Make sure that you have correctly installed the USB device driver on the MS Windows host and that the USB cable is plugged in.
- From the MS Windows host, run the AtlasMgr5.exe program. In the **`Transport`** area, make sure that **`USB`** is selected.
- Then simultaneously reset the board and quickly click on the **`Open`** button of the AtlasMgr5.exe program until the NBoot output appears in the **`USB Termina`**l section of the AtlasMgr5.exe program.
- In the **`Target Media`** section, select **`CS0`**. In the **`Update Functions`** section, select **`Update EBoot`**.

4. In the Open dialog box that appears, navigate to the ipl-titan.bin file, and select it.The IPL will now be programmed to the EBoot area of the CS0 NAND flash. The AtlasMgr utility should display the following message when the IPL is successfully programmed to the flash:

```
Update EBoot Successful!
```

5. Now, reset the Titan board.

6. From the NBoot menu, select #1 Launch EBOOT option.

7. Change the baud rate of your terminal to 115200.

8. You should then see the following IPL output (or similar):

```
QNX Neutrino Initial Program Loader for Centrality Titan board
Commands:

Press 'D' for serial download, using the 'sendnto' utility
```

**Step 4: Transfer the OS image to the target#**

Once the IPL is running on the board, use the following steps to transfer the OS image to the board. You can transfer the OS image into RAM and execute it from RAM.

- Locate the QNX OS image generated in Step 2: Build the BSP
- Select the IPL option D to download the OS image serially to the board.
- Use the QNX sendnto utility to transfer the OS image from the host to the Centrality Titan board.

For instance:

**sendnto -d com1 ifs-titan.bin**

Once the transfer is completed, the OS image will be executed and QNX Neutrino will boot on the board. You should also see the following welcome message on your terminal screen:

```
Welcome to QNX Neutrino 6.x on the Centrality Communications Titan Board
```

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

### Change the clock settings (optional)#

Note the NBoot menu is displayed on the board, you can use the following steps to change the clock settings for the Titan board.

Select the NBoot> option #6 to change the clock settings. You should see the following output:

```
a) DDR_500_250_250_200_125
b) DDR_125_125_125_200_62.5
c) DDR_250_125_125_125_62.5
d) DDR_500_200_200_100_100
e) MDDR_500_250_250_166_125
f) MDDR_125_125_125_166_62.5
g) MDDR_250_125_125_125_62.5
h) MDDR_500_240_240_60_120
i) SDR_500_250_250_133_125
j) SDR_125_125_125_133_62.5
k) SDR_250_125_125_125_62.5
l) SDR_500_240_240_60_120
m) DDR_96_48_48_24_24
n) MDDR_300_150_150_75_133
z) return
Input:
```

The NBoot uses the following setting by default:

a) DDR_500_250_250_200_125

Select the desired frequency option. Once the OS image is loaded into RAM the clock settings selected from the NBoot will be applied.

## 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-titan` | . | . | `src/hardware/startup/boards/titan` |
| Serial | `devc-sertitan -e -F -b115200` | devc-sertitan | . | `src/hardware/devc/sertitan` |

| | | | | |
|---|---|---|---|---|
| ETFS Flash (NAND) | `fs-etfs-titan512 -r32768 -m /fs/etfs` | fs-etfs-titan512<br>etfsctl | . | `src/hardware/etfs/nand512/titan512` |
| Network | `io-pkt-v4 -ddm9000-titan ioport=0x14000000,irq=301 -ptcpip` | io-pkt-v4<br>ifconfig<br>nicinfo*<br>ping* | devn-dm9000-titan.so<br>libsocket.so<br>devnp-shim.so | `src/hardware/devn/dm9000-titan` |

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.

NAND flash has addtional details:

**NAND Flash#**

Command to run the NAND flash ETFS driver on the on-board NAND flash:

```
fs-etfs-titan512 -r32768 -m /fs/etfs
```

where -r <cluster-number> is the number of clusters (a cluster contains 2 pages of 512 B) to reserve for the IPL, the OS image and potential bad blocks skipped when the IPL and the OS image are copied to the flash. You should then see the /dev/etfs1 and /dev/etfs2 partitions. The /dev/etfs1 partition contains the IPL and the OS image, so be sure to not erase it.

To format the second NAND flash partition (i.e. /dev/etfs2):

```
etfsctl -d /dev/etfs2 -S -f -c
```

You should now see the mountpoint /fs/etfs/, which you can use to copy files to.

For more information about these commands, see the Neutrino Utilities Reference

**Known Issues#**

- The default behaviour of NBoot is to check for a system "warm" reset before going on to display the NBoot menu. Due to this, on issuing shutdown from the QNX shell prompt, the Titan EVB boots directly into Windows CE without displaying the NBoot Menu. **`Workaround`**: To gain access to NBoot menu, press the external reset switch.
- Upon bootup, maintain a terminal baud rate of 38400 to gain access to the NBoot Menu messages. To display QNX IPL menu, please change the baud rate of the terminal to 115200.
- The DM9000A Ethernet driver does not maintain a pool of packet buffers but instead allocates packets from io-pkt-v4 when required. This may cause a slight performance degradation under heavy traffic conditions.
- In those instances where the the ROM monitor's MAC address is different from the one you pass in when running **`io-net`** and/or **`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-net`** and/or **`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.