Release Notes for the QNX Neutrino 6.4.0 BSP for Texas Instruments DM355 EVM 1.0.0#

System requirements#

Target system#

- QNX Neutrino RTOS 6.4.0
- Board version: Texas Instruments DM355 EVM
- ROM Monitor version UBoot 1.2.0
- MT29F16G08F NAND flash

Host development system#

- QNX Momentics 6.4.0
- Terminal emulation program (Qtalk, Momentics IDE Terminal, tip, HyperTerminal, etc.)
- RS-232 serial port
- NULL-modem 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:	0x80010000
NAND chip0 base address	0x02000000
NAND chip1 base address	0x02004000

The interrupt vector table can be found in the buildfile located at `src/hardware/startup/boards/dm355/build`

Getting Started#

Step 1: Connect your hardware#

1. Connect the serial cable to the serial port of the TI DM355 EVM board and to the first serial port on the 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.

2. Connect an RJ-45 ethernet cable between the ethernet port on the TI DM355 EVM and your local network.

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.

Step 3: Transfer the OS image to the target #

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

- Baud: 115200
- Bits: 8
- Stop bits: 1
- Parity: none

2. Start your target. You should see output similar to the following:

```
this is MT29F16G08FAA device
UBL: detected valid U-Boot magic number
UBL: booting to U-Boot



U-Boot 1.2.0 (Oct 17 2007 - 15:38:02)

DRAM:  128 MB
NAND:  NAND device: Manufacturer ID: 0x2c, Chip ID: 0xd3 (Micron NAND 1GiB 3,3V 8-bit)
Bad block table found at page 524224, version 0x01
Bad block table found at page 524160, version 0x01
nand_read_bbt: Bad block at 0x1f6e0000
nand_read_bbt: Bad block at 0x3ad00000
nand_read_bbt: Bad block at 0x3fa80000
NAND device: Manufacturer ID: 0x2c, Chip ID: 0xd3 (Micron NAND 1GiB 3,3V 8-bit)
Bad block table found at page 524224, version 0x01
Bad block table found at page 524160, version 0x01
nand_read_bbt: Bad block at 0x27ea0000
nand_read_bbt: Bad block at 0x2ce40000
nand_read_bbt: Bad block at 0x35340000
2048 MiB
In:    serial
Out:   serial
Err:   serial
ARM Clock :- 216MHz
DDR Clock :- 171MHz
Hit any key to stop autoboot:  3
```

Note: Press any key to stop autobooting, and then you'll see the DM355 EVM prompt: DM355 EVM #

Step 4: Setting up the environment#

This method requires a raw image, which the buildfile creates by default. On your target, type the following, filling in the appropriate IP addresses and ifs file:

```
DM355 EVM # setenv netmask 255.255.240.0
DM355 EVM # setenv ipaddr 10.42.104.17
DM355 EVM # setenv gatewayip 10.42.96.1
DM355 EVM # setenv serverip 10.42.104.16
DM355 EVM # setenv bootfile ifs-dm355.raw
DM355 EVM # setenv 'bootcmd tftpboot 80010000;go 80010000'
DM355 EVM # saveenv
```

Restart your target, You should see output similar to the following:

```
Hit any key to stop autoboot:  0
TFTP from server 10.42.104.16; our IP address is 10.42.104.17
Filename 'ifs-dm355.raw'.
```

```
Load address: 0x80010000
Loading: T T T T T T T ##################################################################
         ##########################################################
done
Bytes transferred = 637324 (9b98c hex)
## Starting application at 0x80010000 ...
Welcome to QNX Neutrino trunk on a TI DM355 Platform
```

Step 5: Start working with Neutrino OS#

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 "Summary of driver commands" section.

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-dm355`	.	.	`src/hardware/startup/boards/dm355`
Serial	`devc-ser8250 -e -F -S -b115200 -c24000000/16 0x01c20000^2,0x28`	devc-ser8250	.	`src/hardware/devc/ser8250`
Flash (NAND)	`fs-etfs-dm355 -D addr=0x2004000,reserve=4`	fs-etfs-dm355 etfsctl	.	`src/hardware/etfs/nand2048/dm355`
Network	`io-pkt-v4 -ddm9000 ioport=0x401400,irq=45 -ptcpip`	io-pkt-v4 ifconfig ping*	devn-dm9000.so libsocket.so devnp-shim.so	`src/hardware/devn/dm900`
I2C	`i2c-dm355`	i2c-dm355	.	`src/hardware/i2c/dm355`
Graphics	`io-display -dvid=0,did=0`	io-display dm355.conf	devg-dm355.so	`src/hardware/devg/dm355`
Audio	`io-audio -d dm644x-davinci evmboard=1,aspdev=1&`	io-audio	deva-ctrl-dm644x-davinci.so libasound.so	`src/hardware/deva/ctrl/dm644x`
SD	`devb-mmcsd-dm355 devb-mmcsd-dm355 mmcsd ioport=0x1e00000,irq=27,dma=30,dma=31`	devb-mmcsd-dm355	libcam.so io-blk.so fs-qnx4.so	`src/hardware/devb/mmcsd`
SPI	`spi-master -d dm644x base=0x01c66000,irq=42,edmairq=0xc111,edmachannel=17,clock=10800`	spi-master	spi-dm644x.so	`src/hardware/spi/dm644x`

	spi-master -d dm644x base=0x01c66800,irq=17,edmairq=0xc10f,edmachannel=15,clock=10800			
USB Device	io-usb-dcd -dusbumass-davinci ioport=0x1c64000,irq=12,verbose=2 devu-umass_client-block -vv -l lun=0,devno=1,iface=0,fname=/dev/hd0 or io-usb-dcd -vvvv -d/lib/dll/devu-usbser-davinci.so ioport=0x1c64000,irq=12 devc-serusb_dcd -d iface_list=0 or io-usb-dcd -dusbumass_ser-davinci ioport=0x1c64000,irq=12,verbose=2 devu-umass_client-block -vv -l lun=0,devno=1,iface=0,fname=/dev/hd0 devc-serusb_dcd -d iface_list=1	io-usb-dcd devu-umass_client-block devc-serusb_dcd devu-umass_ctrl	devu-usbumass-davinci.so devu-usbumass_ser-davinci.so devu-usbser-davinci.so	QNX do not provide support of USB device

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.

Graphic and USB have additional details:

Graphics#

The driver supports the following analog outputs:

- NTSC 720x480 interlaced output
- PAL 720x576 interlaced output

The VPSS controller supports the following 4 layers or "OSD windows" on the DM355, (bottom to top),

Layer 0	Video Window 0 supports YUV422 only
Layer 1	Video Window 1 supports YUV422 only
Layer 2	OSD Window 0 supports RGB565, ARGB8888, or YUV422

Layer 3	OSD Window 1 supports RGB565, ARGB8888, or YUV422, or alphalayer for layer 2

The user can control the following features for all layers:

- window start position and size
- horizontal & vertical zoom (2X, 4X)

The OSD windows can support the follow features:

- Alpha blending, 16 levels.
- Transparency - when a pixel matches transparent color, the pixel will be transparent and the underlying video pixels will be displayed.

OSD Rectangular Cursor support can be enabled/configured using dm355.conf.

Note: There is no hardware accelerated 2D or 3D rendering.

Blending and chroma keying can only occur between OSD layers and YUV layers - not between OSD layers or between VID layers. This means you can do blending or chromakey between:

- Layer 2 and Layer 0/1
- Layer 3 and Layer 0/1

You cannot blend/chroma between Layer 2 and 3 or between layer 0 and 1. GLOBAL alpha blending can occur for Layer 2 or Layer 3 with layer 0/1. MAP alpha blending can occur for Layer 2 with layer 0/1 (Layer 3 is used as map)

- only 3 MSB of alpha are used in map (HW restriction).
- patch 1048 must be installed to make this feature work

SRC_PIXEL alpha blending can occur for Layer 2 or Layer 3 with layer 0/1 if the pixel format is ARGB8888 (note only 3 bits of alpha are valid).

USB peripheral#

The following offers in depth detail regarding support of the USB peripheral.

1. Quick Start

Here is a quick overview on how to get the USB mass storage and CDC serial composite device up and running:

- Install Hotfixes 918365 and 935892.
- Prepare the inf file for the serial function, see below.
- Copy C:\windows\system32\drivers\usbser.sys to the same location as the inf file.
- Remove the J9 and J10 jumpers from the dm355 evm board.
- On the target run:

```
/sbin/devb-ram ram  capacity=16384,nodinit,cache=0m
waitfor /dev/hd0
/usr/bin/mkdosfs -F 16 /dev/hd0
   /sbin/io-usb-dcd -vvvv -d/lib/dll/devu-usbumass- davinci.so ioport=0x1c64000,irq=12,verbose=2
/sbin/devu-umass_client-block -vv -l lun=0,devno=1,iface=0,fname=/dev/hd0
/sbin/devc-serusb_dcd -d iface_list=1 -vv
```

- Attach the peripheral to the Windows XP host.
- Point the Hardware Installation Wizard at the inf file prepared earlier.
- Ensure that a new USB Mass Storage Device and a new COM port were detected by Windows XP. The mass storage device should appear as an additional drive letter, such as E:
- Using a terminal emulator open the new COM port, it should appear as COM<X>, where <X> is the largest number - XP assigns com port numbers in order.
- On the target run:

```
stty +edit < /dev/serusb1
on -t serusb1 ksh
```

- Press Enter in the terminal emulator - a ksh prompt should be displayed, and the shell should be usable.

2. Components

The followingi components are used by the USB stack

Binary	Description
lib/libusbdci.so.2 lib/libusbdci.a lib/libusbdciS.a lib/libusbdci.so	USB driver library, used by class drivers and function drivers to communicate with the USB stack (io-usb-dcd)
lib/dll/devu-usbumass-davinci.so lib/dll/devu-usbumass_ser-davinci.so lib/dll/devu-usbser-davinci.so	USB Peripheral drivers for the davinci USB controller. The variants have descriptors coded for different applications: usbumass - mass storage device usbumass_ser - mass storage and cdc serial composite device usbser - standalone cdc serial device The source is available at src/hardware/devu/dc/davinci
sbin/io-usb-dcd	USB stack with peripheral support.
sbin/devu-umass_client-block	Mass storage function driver, standard block filesystem variant.
sbin/devc-serusb_dcd	CDC serial function driver.
bin/umass_ctrl	Example application that connects to devu-umass_client-block resmgr interface and waits for device insertion/removal notification.

Note that the serial number string descriptor can be overridden on the io-usb-dcd command line via the 'ser' option to *-davinci e.g. ser=19/07/2000.

3. DM355 EVM Configuration

The following jumpers need to be removed in order to run as a peripheral:

- j9 - USB VBUS
- j10 - USB ID

See the DM355 EVM documentation for more detail.

4. Serial Support

Required Windows XP Hotfixes

The released version of the serial driver (usbser.sys) on Windows XP sp2 does not support devices with interface association descriptors. Microsoft has released hotfix 918365 to address this issue. See http://support.microsoft.com/kb/918365 This hotfix includes version 5.1.2600.2930 of usbser.sys that is required during hardware installation.

An update to the USB common class generic parent driver (usbccgp.sys) may also be required. Microsoft hotfix 935892 includes verion 5.1.2600.3116. See http://support.microsoft.com/kb/935892 Both these hotfixes should be installed before attaching the peripheral for the first time.

inf file

Windows XP and Windows CE .Net include support for serial over USB. However, a suitable inf file needs to be provided to the hardware detection wizard upon the first insertion of the device.

The inf file needs to be crafted specifically for the device. The vid, pid and mi in the inf file must match the vendor and product ids provided by the standard descriptor in order for the driver to be installed.

To determine the vid, pid and mi that the Hardware Wizard is advertising, one can do the following. Once the peripheral is attached to windows,

- Select Start->Run...
- In the open box, enter "devmgmt.msc"
- Select Okay

Then, in Device Manager

- Select the device
- Select Action->Properties
- Select the Details Tab
- Select "Device Instance Id" from the drop down selection box

The value will be shown the text box. Unless the descriptors have been altered, the serial interface should be MI_0 for the standalone serial device, and MI_1 for the composite device.

Note: An example inf file is included below:

```
; Copyright (C) 2008

[Version]
Signature="$Windows NT$"
Class=Ports
ClassGuid={4D36E978-E325-11CE-BFC1-08002BE10318}
Provider=%ACME%
DriverVer=08/17/2004,0.0.2.0

[Manufacturer]
%ACME%=ACMESerialDeviceList

[ACMESerialDeviceList]
;; uncomment for devu-usbser-davinci.so:
;%ASERIAL%=ACMESerialInstall, USB\VID_19E9&PID_0014&MI_00
;; uncommont for devu-usbumass_ser-davinci.so:
%ASERIAL%=ACMESerialInstall, USB\VID_19E9&PID_0018&MI_01

[DestinationDirs]
DefaultDestDir=10,System32\Drivers

[ACMESerialInstall]
```

```
CopyFiles=ACMESerialCopyFiles
AddReg=ACMESerialAddReg

[ACMESerialCopyFiles]
usbser.sys

[ACMESerialAddReg]
HKR,,DevLoader,,*ntkern
HKR,,NTMPDriver,,usbser.sys
HKR,,EnumPropPages32,,"MsPorts.dll,SerialPortPropPageProvider"

[ACMESerialInstall.Services]
AddService = usbser,0x0002,ACMESerialService

[ACMESerialService]
DisplayName = %ASERIAL_DISPLAY_NAME%
ServiceType = 1                 ; SERVICE_KERNEL_DRIVER
StartType = 3                   ; SERVICE_DEMAND_START
ErrorControl = 1                ; SERVICE_ERROR_NORMAL
ServiceBinary = %10%\System32\Drivers\usbser.sys
LoadOrderGroup = Base

[Strings]
ACME = "Acme"
ASERIAL = "Serial"
ASERIAL_DISPLAY_NAME = "USB Serial Driver"
```

Starting the Driver

Then, launch io-usb-dcd with either the usbser, or usbumass_ser variant,

```
/sbin/io-usb-dcd -vvvv -d/lib/dll/devu-usbser-davinci.so
 ioport=0x1c64000,irq=12,verbose=2
```

and start the serial driver:

```
/sbin/devc-serusb_dcd -d iface_list=0 -vv
```

for usbser, or

```
/sbin/devc-serusb_dcd -d iface_list=1 -vv
```

for the usbumass_ser variant.

Setting up windows

Once the drivers are running on the target platform, the usb cable can be attached to the windows host. An appropriate inf file needs to be provided to the hardware wizard upon the first connection.

On the first insertion Windows should detect the new hardware, and launch the "Found New Hardware Wizard".

- Select the "No, not this time" radio button, then the "Next" button.
- Select the "Install from a list or specific location (Advanced)" radio button, then the "Next" button.
- Select the "Search for the best driver in these locations" radio button. Check the "Include this location in the search" check box. Select the "Browse" button, and navigate to the folder that contains the inf file. Select the "Next" button.
- Select "Continue Anyway" if the wizard warns that the driver has not passed Windows Logo testing.

- The Hardware Wizard may request a copy of usbser.sys. It can be found at C:\windows\system32\drivers \usbser.sys.
- Select "Finish" on the "Completing the Found New Hardware Wizard" dialog

Using the serial connection

After the device is inserted in to the host, and the port is opened on the host, /dev/serusb1 will be created. At this point the serial connection can be used as a normal serial device. For example. ksh can be attached to this device as follows:

```
stty +edit < /dev/serusb1
on -t serusb1 ksh
```

5. Mass Storage Support

inf file

No special configuration is needed on the Windows XP host to connect to and use a mass storage device.

Running the Driver

First the raw storage space required to host the mass storage disk needs to be configured. For this example we will use a small ra

```
/sbin/devb-ram ram  capacity=16384,nodinit,cache=0m
waitfor /dev/hd0
/usr/bin/mkdosfs -F 16 /dev/hd0
```

Then, launch io-usb-dcd with either the usbumass, or usbumass_ser variant, and start the mass storage client:

```
/sbin/io-usb-dcd -vvvv -d/lib/dll/devu-usbumass-davinci.so ioport=0x1c64000,irq=12,verbose=2
/sbin/devu-umass_client-block -vv -l lun=0,devno=1,iface=0,fname=/dev/hd0
```

If usbumass_ser is being used, also start the serial client: /sbin/devc-serusb_dcd -d iface_list=1 -vv

Note: The driver does not support the ability to have both Windows and target board access the filesytem at the same time. While the device is being accessed through Windows, it can not be expected that files on the target board can be accessed. There will be a conflict as both filesystems believe they have exclusive access to the raw media for filesytem updates. Doing so will likely corrupt the filesystem as both filesytems will update the media without the other filesystems knowledge. Once the board is disconnected or ejected from Windows, then it is safe to mount and access the filesystem on the target board. The example umass_ctrl is provided to facilitate this process.

Setting up windows

Once the drivers are running on the target platform, the usb cable can be attached to the windows host. The host should detect the mass storage device and create a new drive letter. Windows uses the serial string to differentiate between new and previously seen devices.

6. Uninstalling the driver from Windows

Windows XP ties drivers to USB devices based on the vid and pid. If the device implementation has changed and a new inf file is required, or the install procedure needs to be tested, the peripheral can be un-installed from Windows XP as follows.

First, copy the following into a new text file, and rename the file to devmgmt.bat.

```
set devmgr_show_nonpresent_devices=1
```

```
start devmgmt.msc
```

Then after disconnecting the peripheral, run devmgmt.bat to launch the device manager. From the view menu, select Show hidden devices. Then find the old device instances and delete them. Depending on the device type they may be under,

- Networking adapters
- Disk Drives
- Other Devices
- Storage Volumes
- Universal Serial Bus Controllers

The vid and pid of the device can be confirmed via the properties.

Known Issues for this BSP#

Windows Connection Issue

Problem Description

If the target board is disconnected while the COM port is open and re- connected, the serial connection won't become active until the open session is closed. After closing the COM port, serial becomes active but the COM port will fail to appear in the available list even though the port is listed in the device manager.

Workaround

After removing the peripheral close or disconnect the terminal emulator before re-inserting the target.