LSP 1.20 DaVinci Linux EVM Installation

User's Guide

Literature Number: SPRUFG0
March 2008

Contents

LSP 1.20 DaVinci Linux EVM Installation

This is the installation guide for LSP 1.20 DaVinci Linux. LSP 1.20 is supported on the following EVMs: DM644x, DM355.

1 LSP 1.20 Installation – DM355 and DM6446 EVMs

1.1 DM355 EVM

The Bootloader (ublDM355-nand.bin and u-boot-1.2.0-dm355.bin), Flash/NAND programmer utilities (NAND_programmer.out) and Kernel (uImage-dm355) binaries for the DM355 EVM are part of the DVSDK release at `PSP_##_##_##_###/bin/dm355` location. The sample Ramdisk image (ramdisk.gz) can be found at the location `PSP_##_##_##_###/bin`. The U-Boot command reference guide - commands.txt can be found at the location `PSP_##_##_##_###/docs`

Installation Method I: Using Code Composer Studio (CCS)

1. System Requirements
   - MontaVista Pro 4.0.1 Toolchain, based on GCC 3.4.3 installed on RedHat Red Hat Enterprise Linux WS release 3 or Red Hat Linux 9.0
   - CCS 3.3.38.2 installed on Windows XP with Service Pack 2
2. Download the configuration files for CCS from the Spectrum Digital site for DM355: http://c6000.spectrumdigital.com/evmdm355/revc/
   - CCS configuration file – `evmdm355_xds510usb.ccs` or `evmdm355_xds560.ccs`
   - EVM GEL file – `evmdm355.gel`
3. Connect the XDS560 or XDS510 to the TI JTAG.
4. Setup CCS 3.3 with the CCS configuration and GEL files corresponding to the DM355 EVM.
5. Open CCS 3.3, Alt+C to connect to the device. Ctrl+R to reset from CCS.
6. Load program. Choose NAND_programmer.out from the release and run it (F5).

Note: There is a logic in the NAND programmer to erase (scrub) all the blocks (4096), to make sure there are no bad blocks.

7. NAND programmer will ask you for input for the UBL binary. Choose the `ublDM355-nand.bin` file from the release.
8. NAND programmer will then ask you for the U-Boot binary. Choose the `uboot-1.2.0-dm355.bin` file from the release.
9. Turn off the board and disconnect the CCS from the device -> Alt+C and disconnect XDS560 or XDS510 from the device.
10. Connect the RS232 cable between your PC and EVM.
11. You will see console messages, UBL identifying the U-Boot magic number and giving control to U-Boot 1.2.0.
12. Since this is the first time you are bringing up U-Boot 1.2.0 on your EVM, you will notice a delay about 10 seconds, while detecting the NAND device. U-Boot 1.2.0 is going through the NAND blocks to populate the bad block table (BBT) in a specific block of NAND device for future. You will not notice this delay for the subsequent power resets.
13. Configure the TFTP Server. Specify a TFTP default directory (Ex. C:\TFTPServer). Copy the `uImage-dm355` and `ramdisk.gz` files to this directory from the release.

14. Now follow the `commands.txt` file to install `uImage-dm355` and/or filesystem binaries. You also need to follow `commands.txt` file to set up the bootcmd and bootargs ENV variables.

Installation Method II: Upgrade U-Boot Binary from U-Boot

Software Requirements

- U-Boot image
- Tftpserver

For DM355 with SLC (512MB/2GB)

```
tftp 0x80700000 u-boot-1.2.0-dm355_evm.bin
nand erase 0x140000 0x20000
nand write 0x80700000 0x140000 0x20000
```

For DM355 with MLC (2GB)

```
tftp 0x80700000 u-boot-1.2.0-dm355_evm.bin
nand erase 0x280000 0x20000
nand write 0x80700000 0x280000 0x20000
```

1.2 DM6446 EVM

The Bootloader (u-boot-1.2.0-davinci-nor.bin and u-boot-1.2.0-davinci-nand.bin), Flash/NAND programmer utilities (DV_NORWriter.out and DV_NANDWriter.out) and Kernel (uImage-dm6446) binaries for DM6446 EVM are part of the DVSDK release at `PSP_##_##_##_###/bin/dm6446` location. The sample Ramdisk image (ramdisk.gz) can be found at the location `PSP_##_##_##_###/bin`. The U-Boot command reference guide - commands.txt can be found at the location `PSP_##_##_##_###/docs`

Installation Method I: Using Code Composer Studio (CCS)

1. System Requirements
   - MontaVista Pro 4.0.1 Toolchain, based on GCC 3.4.3 installed on RedHat Red Hat Enterprise Linux (RHEL) WS release 3 or Red Hat Linux 9.0
   - CCS 3.3.38.2 installed on Windows XP with Service Pack 2
2. Download the configuration files for CCS from the Spectrum Digital Early site for DM6446 http://c6000.spectrumdigital.com/davincievm/revf/
   - CCS configuration file – `DaVinci_XDS560_icepick_armdsp.ccs`
   - EVM GEL file – `davincievm_arm.gel`
3. Jumper and/or Switch settings

   DM6446 EVM settings for NOR boot mode:

   J4: in FLASH mode

   S3: in FLASH mode 1011111110

   DM6446 EVM settings for NAND boot mode:

   J4: in NAND mode

   S3: in NAND mode 0000111110
4. Connect the XDS560 or XDS510 to the TI JTAG.
5. Setup CCS 3.3 with the CCS configuration and GEL files corresponding to the DM6446 EVM.
6. Open CCS 3.3, Alt+C to connect to the device. Ctrl+R to reset from CCS.
7. Load program. Choose `DV_NORWriter.out` for NOR mode (or `DV_NANDWriter.out` for NAND mode) from the release and run it (F5).
8. NOR programmer (`DV_NORWriter.out`) will ask you whether UBL is needed and input for the UBL binary. Choose 'y' and `ubl_davinci_nor.bin` from the release as the input file.
   a. NAND programmer (`DV_NANDWriter.out`) will ask input for the UBL binary. Choose the `ubl_davinci_nand.bin` file from the release.
9. NOR programmer will then ask for the U-Boot binary. Choose the `u-boot-1.2.0-davinci-nor.bin` file with flash support enabled (default u-boot-1.2.0 build).
   a. NAND programmer will then ask for the U-Boot binary. Choose the `u-boot-1.2.0-davinci-nand.bin` file with the NAND support enabled.

b. NAND/NOR programmer will ask for further inputs such as load address, start address. You can provide “0” as the value, to accept default values.

10. Turn off the board and disconnect the CCS from the device -> Alt+C and disconnect XDS560 or XDS510 from the device.
11. Connect the RS232 cable between your PC and EVM.
12. You will see console messages, UBL identifying the U-Boot magic number and giving control to U-Boot 1.2.0.
13. In the NOR boot mode, since this is the first time you are bringing up U-Boot 1.2.0 on your EVM, you will notice a delay about 6 seconds, while detecting the NOR device. In the NAND boot mode, you will notice a delay about 10 secs, while detecting the NAND device. U-Boot 1.2.0 is going through the NAND blocks to populate the bad block table (BBT) in a specific block of NAND device for future. You will not notice this delay for the subsequent power resets.
14. Configure the TFTP Server. Specify a TFTP default directory (Ex. C:\TFTPServer). Copy the `uImage-dm6446` and `ramdisk.gz` files to this directory from the release.
15. Now follow `commands.txt` file to install `uImage-dm6446` and/or filesystem binaries. You also need to follow `commands.txt` file to set up the bootcmd and bootargs ENV variables.

Installation Method II: Upgrade U-Boot Binary from U-Boot

Software Requirements

- U-Boot images for both NAND and NOR
- Tftpserver

For NAND boot mode

```
tftp 0x80700000 u-boot-1.2.0-davinci.bin
nand erase 0x18000 0x20000
nand write 0x80700000 0x18000 0x20000
```

For NOR boot mode

```
tftp 0x80700000 u-boot-1.2.0-davinci.bin
protect off 0x02000000 +0x20000
erase 0x02000000 +0x20000
cp.b 0x80700000 0x02000000 0x20000
protect on 0x02000000 +0x20000
```

2 Build Procedure for DM355 and DM6446 EVMs

2.1 NAND UBL

This is a CCS based application that resides in the NAND flash, as identified by the ROM Bootloader (RBL). This application is used to initialize the EVM, read the U-Boot binary from the NAND flash, load it to the RAM and give control to the start address of U-Boot in the RAM.

The source tar ball for NAND UBL (ublNAND.tgz) is part of the DVSDK release at `PSP_##_##_##_###/board_utilities` location.

Following are the steps to build NAND UBL for DM355:

1. The tar file `ublNAND.tgz` needs to be extracted to a machine with CCS 3.3.38.2 installed.
2. Project->Open and choose the `build/ubldm355_debug.pjt` file. This opens the CCS project for NAND UBL.
3. You can choose to rebuild the application using “Project/Rebuild All” option.
4. The NAND UBL binary is located in “build” directory – `ublDM355-nand.bin`

Note: The LSP 1.20 release does not support NAND programmer and NAND UBL for DM6446. This release will continue to use the Flash writer and NAND UBL binaries from ti-davinci LSP 1.10 release.

2.2 U-Boot 1.2.0

This is a Linux Bootloader application which resides in the NAND flash, as identified by the UBL. This application has Ethernet and NAND read/write support. The application provides a command line, where certain commands can be executed. E.g.: setenv, saveenv, nand dump, nand write, nand erase etc.

1. Extract the `u-boot-1.2.0.tgz` file to a Linux machine, with MontaVista PE 4.0.1 installed (toolchain).
2. For DM355, follow the steps below to build a new u-boot 1.2.0 binary:

```
cd u-boot-1.2.0
make distclean
make dm355_evm_config
make
```

3. For DM6446, the default configuration is to build for NOR boot mode. Follow the steps below to build a new u-boot 1.2.0 binary for NOR boot mode:

```
cd u-boot-1.2.0
make distclean
make davinci_config
make
```

4. For DM6446, follow the steps below to build a new u-boot 1.2.0 binary for NAND boot mode:

```
cd u-boot-1.2.0
vim include/configs/davinci.h
```

Comment the following macro definition to disable the NOR boot mode and thus enable the NAND boot mode:

```
/*#define CFG_ENV_IS_IN_FLASH     1 */    /* U-Boot env in NOR Flash   */
```

Now do the following:

```
make distclean
make davinci_config
make
```

2.3 Linux 2.6.10 Kernel for DM355 and DM6446 EVMs

2.3.1 Build With Default Configuration

The LSP installation binary mvl_4.0.1_demo_lsp_setuplinux_##_##_##_###.bin should be installed on a Linux machine, with MontaVista Pro 4.0.1 toolchain installed. The preferred installation directory for the LSP is /opt/mv_pro_4.0.1. This binary installs the LSP tar ball `DaVinciLSP-##_##_##_###.tar.gz` in the /opt/mv_pro_4.0.1 directory (or the directory you chosen to install). The `DaVinciLSP-##_##_##_###.tar.gz` should be extracted in the /opt/mv_pro_4.0.1 directory (or the directory you have chosen to install). It is recommended to make a local copy of the Linux Kernel directory - `montavista/pro/devkit/lsp/ti-davinci` for actual build and any changes.

1. For DM355, follow the steps below to build the kernel image:

```
cd <local-dir>/montavista/pro/devkit/lsp/ti-davinci
make distclean
make davinci_dm355_evm_defconfig
make uImage
```

2. For DM6446, follow the steps below to build the kernel image:

```
cd <local-dir>/montavista/pro/devkit/lsp/ti-davinci
make distclean
make davinci_dm644x_defconfig
make uImage
```

3. The `uImage` binary is located in the `arch/arm/boot` directory.

2.3.2 USB Driver Configuration – Host/Device Modes

The default DM355/DM6446 device configuration for USB driver is Host Mode. The following configuration options are required to switch to Ethernet Gadget or File-backed Storage Gadget modes.

USB driver (Host Mode, Default)

```
CONFIG_USB=y
CONFIG_USB_DEVICEFS=y
CONFIG_USB_ARCH_HAS_HCD=y
CONFIG_USB_MUSB_HDRC=y
CONFIG_USB_MUSB_SOC=y
CONFIG_USB_MUSB_HOST=y
CONFIG_USB_MUSB_HDRC_HCD=y
CONFIG_USB_TI_CPPI_DMA=y
CONFIG_USB_INVENTRA_STATIC_CONFIG=y
CONFIG_USB_INVENTRA_HCD_LOGGING=0
CONFIG_USB_STORAGE=y
CONFIG_USB_HID=y
CONFIG_USB_HIDINPUT=y
```

USB driver (Peripheral Mode, Ethernet Gadget)

```
CONFIG_USB_ARCH_HAS_HCD=y
CONFIG_USB_MUSB_HDRC=y
CONFIG_USB_MUSB_SOC=y
CONFIG_USB_MUSB_PERIPHERAL=y
CONFIG_USB_GADGET_MUSB_HDRC=y
CONFIG_USB_TI_CPPI_DMA=y
CONFIG_USB_INVENTRA_STATIC_CONFIG=y
CONFIG_USB_INVENTRA_HCD_LOGGING=0
CONFIG_USB_GADGET=y
CONFIG_USB_GADGET_DUALSPEED=y
CONFIG_USB_ETH=y
CONFIG_USB_ETH_RNDIS=y
```

USB driver (Peripheral Mode, File-backed Storage Gadget)

```
CONFIG_USB_ARCH_HAS_HCD=y
CONFIG_USB_MUSB_HDRC=y
CONFIG_USB_MUSB_SOC=y
CONFIG_USB_MUSB_PERIPHERAL=y
CONFIG_USB_GADGET_MUSB_HDRC=y
CONFIG_USB_TI_CPPI_DMA=y
CONFIG_USB_INVENTRA_STATIC_CONFIG=y
CONFIG_USB_INVENTRA_HCD_LOGGING=0
CONFIG_USB_GADGET=y
CONFIG_USB_GADGET_DUALSPEED=y
CONFIG_USB_FILE_STORAGE=m
```

2.3.3 Building Device Drivers as Kernel Modules

The default configuration will have all the drivers statically built as part of kernel image – `uImage`. In order to make a driver (e.g. Audio, Video – V4L2) a dynamically loadable kernel module, please follow the steps below:

1. The ti-davinci LSP toolchain 4.0.1 should have been installed on your Host machine
2. You should have the Linux Kernel source copied to a local directory, from the LSP installation location - `montavista/pro/devkit/lsp/ti-davinci`.
3. `cd <local-dir>/montavista/pro/devkit/lsp/ti-davinci`
4. `make davinci_dm355_evm_defconfig` for DM355 or `make davinci_dm644x_defconfig` for DM6446
5. Open the configuration file: `.config`

Note: Alternatively, you can achieve the same by issuing “make menuconfig” or “make xconfig” commands and go through the options.

6. Set the corresponding driver to a “module”

Sound/Audio Driver

```
CONFIG_SOUND_DAVINCI=m
CONFIG_SOUND_DAVINCI_TLV320AIC33=m
```

Video-V4L2-Display Driver

```
It is not recommended to build V4L2 Display Driver as a module, as it is dependent on Encoder
and Display Managers.
```

Video-V4L2-Capture Driver

```
CONFIG_VIDEO_TVP5146=m
CONFIG_VIDEO_MT9T001=m
CONFIG_VIDEO_DAVINCI_CAPTURE=m
```

MMC/SD Driver

```
CONFIG_MMC_DAVINCI=m
```

SPI/EEPROM Driver

```
CONFIG_SPI_DAVINCI_BITBANG=m
CONFIG_SPI_DAVINCI=m
(Note: Only on DM355)
```

Ethernet – DM9000A Driver

```
CONFIG_DM9000=m
(Note: Only on DM355)
```

Ethernet – CPMAC Driver

```
CONFIG_TI_DAVINCI_EMAC=m
(Note: Only on DM6446)
```

DaVinci NAND Driver

```
CONFIG_MTD_NAND_DAVINCI=m
```

USB Driver (Host Mode)

```
CONFIG_USB=m
CONFIG_USB_DEVICEFS=y
CONFIG_USB_ARCH_HAS_HCD=y
CONFIG_USB_MUSB_HDRC=m
CONFIG_USB_MUSB_SOC=y
CONFIG_USB_MUSB_HOST=y
CONFIG_USB_MUSB_HDRC_HCD=y
CONFIG_USB_TI_CPPI_DMA=y
CONFIG_USB_INVENTRA_STATIC_CONFIG=y
CONFIG_USB_INVENTRA_HCD_LOGGING=0
CONFIG_USB_STORAGE=m
CONFIG_USB_HID=m
CONFIG_USB_HIDINPUT=y
```

USB Driver (Peripheral Mode, Ethernet Gadget)

```
CONFIG_USB_ARCH_HAS_HCD=y
CONFIG_USB_MUSB_HDRC=m
CONFIG_USB_MUSB_SOC=y
CONFIG_USB_MUSB_PERIPHERAL=y
CONFIG_USB_GADGET_MUSB_HDRC=y
CONFIG_USB_TI_CPPI_DMA=y
CONFIG_USB_INVENTRA_STATIC_CONFIG=y
CONFIG_USB_INVENTRA_HCD_LOGGING=0
CONFIG_USB_GADGET=m
CONFIG_USB_GADGET_DUALSPEED=y
CONFIG_USB_ETH=m
CONFIG_USB_ETH_RNDIS=y
```

USB Driver (Peripheral Mode, File-backed Storage Gadget)

```
CONFIG_USB_ARCH_HAS_HCD=y
CONFIG_USB_MUSB_HDRC=m
CONFIG_USB_MUSB_SOC=y
CONFIG_USB_MUSB_PERIPHERAL=y
CONFIG_USB_GADGET_MUSB_HDRC=y
CONFIG_USB_TI_CPPI_DMA=y
CONFIG_USB_INVENTRA_STATIC_CONFIG=y
```

```
CONFIG_USB_INVENTRA_HCD_LOGGING=0
CONFIG_USB_GADGET=m
CONFIG_USB_GADGET_DUALSPEED=y
CONFIG_USB_FILE_STORAGE=m
```

7. Save the `.config` file and exit
8. `make clean`
9. `make uImage`
10. `make modules`
11. Load the new uImage to your DM355/DM6446 EVM. You can follow the commands.txt to install the new uImage on your DM355/DM6446 EVM.
12. After you login at the kernel login prompt, download the corresponding driver kernel module binaries to your DM355/DM6446 EVM.
13. Execute the command `insmod` in the order given for the corresponding driver kernel module. The order of insertion is very important.

    Sound/Audio Driver

    ```
    sound/oss/dm644x/dm644x_audio_driver.ko
    ```

    ideo-V4L2-Capture Driver

    ```
    drivers/media/video/davinci/mt9t001.ko
    drivers/media/video/davinci/tvp5146.ko
    drivers/media/video/davinci/davinci_capture.ko
    ```

    MMC/SD Driver

    ```
    drivers/mmc/davinci_mmc.ko
    ```

    SPI/EEPROM Driver

    ```
    drivers/spi/davinci_spi_bitbang.ko
    drivers/spi/davinci_spi_master.ko
    (Note: Only on DM355)
    ```

    Ethernet – DM9000A Driver

    ```
    drivers/net/dm9000.ko
    (Note: Only on DM355)
    ```

    Ethernet – CPMAC Driver

    ```
    drivers/net/davinci_emac_driver.ko
    (Note: Only on DM6446)
    ```

    DaVinci NAND Driver

    ```
    drivers/mtd/nand/nand_davinci.ko
    ```

    USB Driver (Host Mode)

    ```
    drivers/usb/input/usbhid.ko
    drivers/usb/storage/usb-storage.ko
    drivers/usb/musb/musb_hdrc.ko
    drivers/usb/core/usbcore.ko
    ```

    USB Driver (Peripheral Mode, Ethernet Gadget)

    ```
    drivers/usb/musb/musb_hdrc.ko
    drivers/usb/gadget/g_ether.ko
    ```

    USB Driver (Peripheral mode, File-backed Storage Gadget)

    ```
    drivers/usb/musb/musb_hdrc.ko
    drivers/usb/gadget/g_file_storage.ko
    ```

14. Verify the corresponding driver functionality.
15. Execute the command `rmmod` in the reverse order of insertion, for the corresponding driver.
16. Alternatively, if you want to create a filesystem (ramdisk) with the kernel modules, follow the steps below:

    Step a. On your host machine, `mkdir /mnt/ramdisk`

    Step b. Goto folder carrying the ramdisk.gz

    Step c. `gzip -d ramdisk.gz`

    Step d. `mount -o loop -t ext2 ./ramdisk /mnt/ramdisk`

Step e. After step 10 above, `make modules_install INSTALL_MOD_PATH=/mnt/ramdisk/`

Step f. Go back to the folder where the ramdisk directory is located

Step g. `umount /mnt/ramdisk`

Step h. `gzip ramdisk`

Step i. Now you have the new filesytem with the kernel modules

Use this filesystem (ramdisk.gz) to load on to your EVM target. You can skip step 13 above.

2.3.4 Building Drivers in Interrupt/Polling/DMA Modes

USB Driver

The default mode of operation is “DMA”. To switch to the “Interrupt or PIO” mode of operation:

1. Issue the following command to change the Kernel options: `make menuconfig`
2. Select the `Device Drivers` option and then `USB support`
3. To enable Interrupt or PIO mode of operation, check the option below:
   `Disable DMA (always use PIO)`

SPI Driver

The default mode of operation is “polling”. To enable interrupt mode of operation, change the following source file:

Open

`drivers/spi/davinci_spi_master.c`

Add the following line:

```
#define CONFIG_SPI_INTERRUPT
```

Note: Commenting out the above line in the codebase will make SPI driver go back to “polling” mode of operation.

The SPI Driver is supported only on DM355 devices.

MMC/SD Driver

The default mode of operation is “ DMA”. To disable DMA mode of operation and go to PIO (or interrupt) mode, change the following source file:

`Open linux-2.6.10/arch/arm/mach-davinci/board-dm355-evm.c`

Comment out the following DMA resource definition entries for MMC0 and MMC1 on DM355. During boot up, MMC driver should mention whether it is booting with DMA enabled or disabled.

```
static struct resource mmc0_resources[] = {
        [0] = {                 /* registers */
                .start   = DM355_MMC0_BASE,
                .end    = DM355_MMC0_BASE + SZ_1K - 1,
                .flags   = IORESOURCE_MEM,
        },
        [1] = {                 /* interrupt */
                .start   = IRQ_DM355_MMCINT0,
                .end    = IRQ_DM355_MMCINT0,
                .flags   = IORESOURCE_IRQ,
        },
#if 0
        [2] = {                 /* dma rx */
                .start   = DM355_DMA_MMC0RXEVT,
                .end    = DM355_DMA_MMC0RXEVT,
                .flags   = IORESOURCE_DMA,
        },
        [3] = {                 /* dma tx */
                .start   = DM355_DMA_MMC0TXEVT,
```

```
                .end    = DM355_DMA_MMC0TXEVT,
                .flags   = IORESOURCE_DMA,
        },
#endif
};

static struct resource mmc1_resources[] = {
        [0] = {                 /* registers */
                .start   = DM355_MMC1_BASE,
                .end    = DM355_MMC1_BASE + SZ_1K - 1,
                .flags   = IORESOURCE_MEM,
        },
        [1] = {                 /* interrupt */
                .start   = IRQ_DM355_MMCINT1,
                .end    = IRQ_DM355_MMCINT1,
                .flags   = IORESOURCE_IRQ,
        },
#if 0
        [2] = {                 /* dma rx */
                .start   = DM355_DMA_MMC1RXEVT,
                .end    = DM355_DMA_MMC1RXEVT,
                .flags   = IORESOURCE_DMA,
        },
        [3] = {                 /* dma tx */
                .start   = DM355_DMA_MMC1TXEVT,
                .end    = DM355_DMA_MMC1TXEVT,
                .flags   = IORESOURCE_DMA,
        },
#endif
};
```

Comment out the following DMA resource definition entries for MMC0 on DM6446. During boot up, MMC driver should mention whether it is booting with DMA enabled or disabled.

```
static struct resource mmc0_resources[] = {
        [0] = {                 /* registers */
                .start          = DAVINCI_MMC_SD_BASE,
                .end            = DAVINCI_MMC_SD_BASE + SZ_1K - 1,
                .flags          = IORESOURCE_MEM,
        },
        [1] = {                 /* interrupt */
                .start          = IRQ_MMCINT,
                .end            = IRQ_MMCINT,
                .flags          = IORESOURCE_IRQ,
        },
#if 0
        [2] = {                 /* dma rx */
                .start          = DAVINCI_DMA_MMCRXEVT,
                .end            = DAVINCI_DMA_MMCRXEVT,
                .flags          = IORESOURCE_DMA,
        },
        [3] = {                 /* dma tx */
                .start          = DAVINCI_DMA_MMCTXEVT,
                .end            = DAVINCI_DMA_MMCTXEVT,
                .flags          = IORESOURCE_DMA,
        },
#endif
};
```

2.3.5 Video – fbdev, V4L2 Capture and Display Options

1. The default input device for V4L2 capture is TVP5146. To change the input device to MT9T001/MT9T031, change the bootargs to include:

   ```
   v4l2_video_capture=[option[:option]]
   ```

   Valid Options:

   ```
   device=[MT9T031|MT9T001|TVP5146]
   ```

2. LogicPD LCD support in fbdev

   The PRGB signals are multiplexed on the same pins as the PWM signals, so you cannot use the PWM driver if a PRGB display is selected. The kernel configuration is updated to take care of this dependency. When LogicPD LCD Encoder support is enabled, PWM gets disabled. When PWM is

enabled LogicPD LCD Encoder support gets disabled.

Alternatively the PWM and LogicPD LCD Encoder options can be disabled as follows:

a. Issue the following command to change the Kernel options `make menuconfig`
b. Select the `Device Drivers` option
c. To disable PWM, select the `Character Devices` option and then uncheck the `DaVinci PWM Driver Support` option
d. To disable LCD Encoder, select the `Multimedia devices` and then select `Video For Linux` option. Uncheck the `Logic PD Encoder support` option

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Clocks and Timers	www.ti.com/clocks	Digital Control	www.ti.com/digitalcontrol
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265