# Enabling Hardware Accelerated Playback for Intel® Atom™ Processor N2000/D2000 Series

Case Study Using MPlayer on Fedora 16

June 2012

## Document Revision History

| Revision | Date | Description |
|---|---|---|
| 0.01 | 6/13/12 | Initial Version |
| 0.03 | 6/14/12 | General cleanup after testing instructions. |
| | | |
| | | |
| | | |

# Table of Contents

# 1 Executive Summary

The Cedar Trail platform is based on the Intel® Atom™ Processor N2000 and D2000 Series (formerly called Cedarview) and the Intel® NM10 Express Chipset (formerly called Tiger Point) . This processor series is based on a 32nm process and features new levels of performance-per-watt opening the door to always-on, always-connected embedded devices.

The N2000/D2000 series processors include an integrated graphics controller with advanced media handling capabilities such as smooth full HD (up to 1080p) video playback along with support for wide range of outputs such as VGA, LVDS, HDMI*, DP* and eDP[1]. Media playback is optimum when the video decoding is handled by the video engine in the integrated graphics controller. Using the video engine can significantly reduce the CPU workload and also help improves the quality of playback.

Intel has released Graphics/Media drivers (referred to as PVR-CDV drivers in this document) for the MeeGo open-source distribution for the Cedar Trail platform. The PVR-CDV drivers exploit the video and graphics acceleration capabilities of the media engine. This paper is a case study on integrating these drivers with the Fedora distribution and then demonstrating the video acceleration capability of the system with the MPlayer video player. The steps to demonstrate Graphics acceleration using Glxgears demo utility are also provided.

The case study was done on a system with Intel Atom N2800 processor and Intel NM10 Express Chipset. However, the findings are applicable in general to any Cedar Trail system.



[1] Current version of PVR-CDV drivers do not support DP and eDP ports

# 2 Overview of Hardware Accelerated Decode

To meet the demands of low power and high performance, the Cedarview processor provides dedicated graphics and video decode acceleration hardware for delivering fast video and graphics rendering. Hardware acceleration frees up most of the CPU bandwidth for other time critical tasks. In addition to lowering CPU utilization, hardware acceleration also helps improve the quality of media play back by reducing or eliminating frame drop and reducing Audio/Video synchronization issues. The diagram below provides a simplified overview of the Graphics & Video acceleration stack.

VA-API is a standard API that exposes offloading of video decoding to acceleration hardware. "libVA" is a open source library implementation of the VA-API specification. This library provides access to the hardware used for acceleration of video processing. It enables hardware accelerated video decode at various entry-points (VLD, IDCT, Motion Compensation, deblocking) for the prevailing coding standards today (MPEG-2, MPEG-4 ASP/H.263, MPEG-4 AVC/H.264, and VC-1/WMV3). The VA hardware driver is the Cedarview hardware specific video decode driver.

OpenGL ES API is used to expose 3D Graphics acceleration to the Graphics application[2]. libGLES is the open source library implementation of the OpenGL ES API. The DRI driver converts libGLES commands to the graphics hardware accelerator specific commands.

The X server provides the basic services for managing windows displays and input devices. It provides the basic framework for building GUIs.

DRM (Direct Rendering Manager) kernel module is the Kernel space portion of the "Direct Rendering Infrastructure" (DRI). The DRM kernel module provides synchronized access to the graphics controller from the Video and Graphics Drivers and the X server.

[2] OpenGL API is partially supported by PVR-CDV drivers. However OpenGL ES is the recommended API for 3D Graphics applications

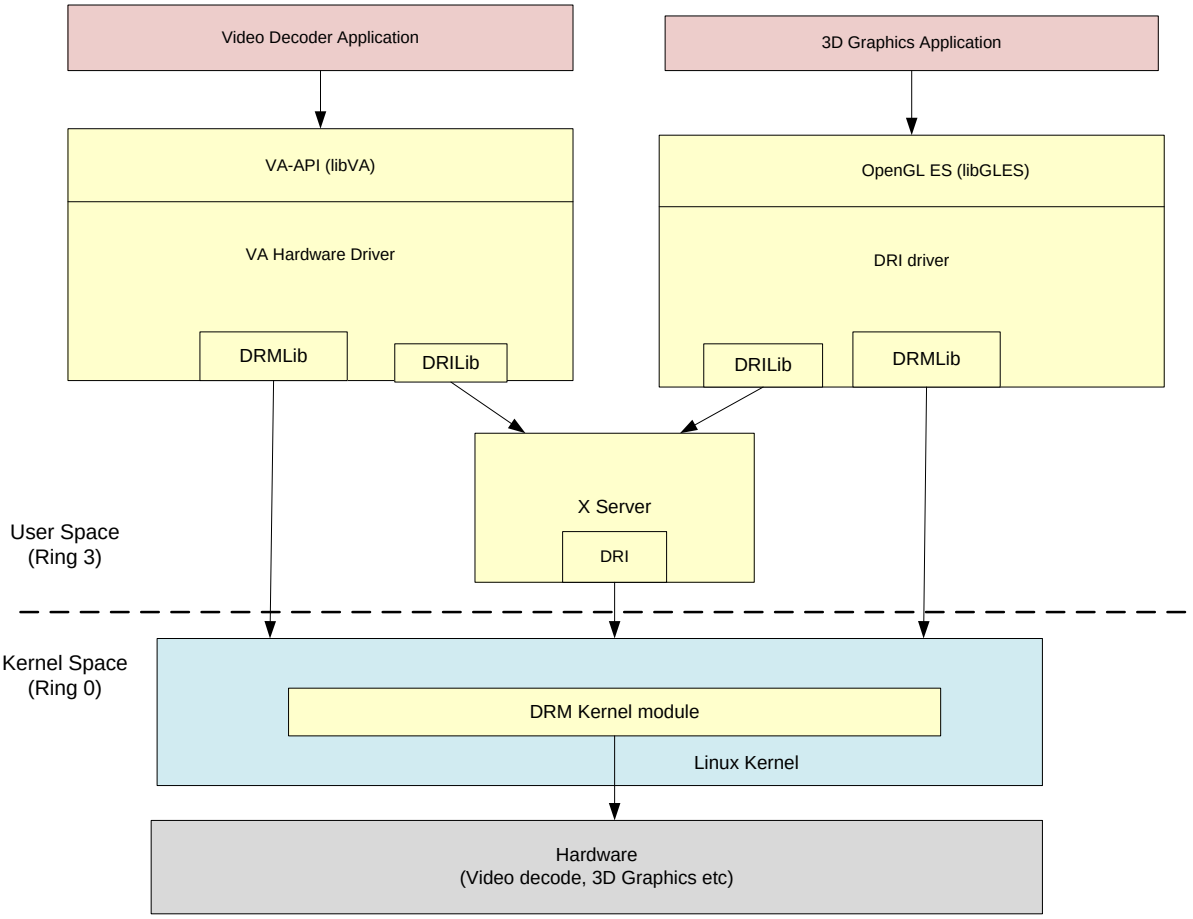


**Figure 1. Overview of Hardware Accelerated Decode**

# 3 Build and Installation Overview

The build and installation of the Cedar Trail software stack is done as follows:

- First, build and install the Cedar Trail target kernel with required patches. The target kernel is built by patching the 3.1.0 kernel release from kernel.org with PVR-CDV kernel patches. Note that the target kernel can be natively built either on the target itself or cross-compiled on a host machine. This document provides instructions for both cases.
- Installation of drivers, libraries, and other packages needed for hardware accelerated playback is done next. The graphics driver used in this case study is from the MeeGo v1.2 release for Cedar Trail platform. The driver has the following dependencies.

    - Mesa GL 7.11
    - X Server 1.11
    - Kernel 3.1.0
    - Patched libwsbm 1.1.0+ (provided in release)
    - llibc >= 2.11.90
    - libva >= 1.0.15
    - pixman >= 0.22.0
    - gcc >= 4.5.1
    - libdrm >= 2.4.25

- Build and install MPlayer on Target machine. The MPlayer is patched to use VA-API based hardware acceleration for Video decoding instead of using CPU for video decode.

# 4 Step-by-Step Instructions

## 4.1 Cedar Trail Platform Preparation

1. Ensure that the board has DDR memory installed (Minimum of 1 GB recommended)
2. Connect a VGA or HDMI monitor to the board.

   This study was done with a VGA and HDMI monitor, however other displays supported by the Cedar Trail platform and driver can also be used such as: LVDS or DP:
3. Connect a USB keyboard and a mouse to the board.
4. Connect a hard disk to the board.
5. Connect a DVD player to the board using SATA cable.
6. Connect the power supply to the board.

## 4.2 Fedora 16 Installation on Cedar Trail Target Platform

Follow the following steps to install the Linux Fedora-16, 32 bit (i686) OS on the Cedar Trail target board.

1. Disconnect the LAN cable during the installation because Fedora 16 will try to go online and update the packages during installation and that will put the wrong kernel version on the system.
2. Turn ON the power supply connected to the target board
3. Power-On the board. (On most systems this would typically require momentarily pushing the Power-ON button)
4. You should now see the boot menu on the VGA/HDMI monitor as seen below:

   **“No boot device has been detected, please press any key to reboot!”**
5. Insert Fedora 16 i686 DVD in the DVD player.
6. Press enter and it starts installing the Fedora OS in the hard disk.
7. Select “Install new system” or “upgrade” from the menu. Select “test the Media” before installation
8. Click “next” when Fedora 16 Menu comes
9. Select “English” as a installation language
10. Select “U.S.English” for keyboard
11. Select “Basic Storage Devices” and click “next”
12. Select “Fresh Installation” to start a new installation (the old data will be lost)
13. Use the default (localhost.localdomain) host name and click “next”
14. Select your time zone and click “next”
15. Type the root password, store it in a safe place and click “next”
16. Select “Replace existing Linux System” and click “next”

17. Select “Write Changes to Disk”
18. Select “Software Development”, “Installation Repo”, “Customize Now”, and click next.

    NOTE! Selecting “Software Development” allows the right tools to be installed without using yum later which may install updated packages that we don’t want right now.
19. Under “Desktop Environments” select Xfce, and deselect GNOME because the F16 uses GNOME 3 and that requires OpenGL. The Cedarview driver provides support for OpenGLES 2.0, but limited “big” OpenGL support.
20. Once Installation is complete, take the DVD from the player and select “reboot”
21. Reconnect LAN cable.
22. After reboot, you will see the welcome menu, click “Forward”
23. Click Forward in the screen “License Information”
24. Create user name and password and click “Forward”
25. Set appropriate date and time in the calendar, click “Forward”
26. Click “Finish”
27. Login and select Default on desktop option
28. Disable “SELinux”

    **vi /etc/selinux/config**

    *SELINUX=disabled*

    NOTE. This will not take effect until the system is rebooted.
29. Enable “eth0” to activate the internet connection using Network Manager. It should be a button on the top right of screen.

## 4.3 Build Kernel on Host and Install on Target

Follow these steps only if you intend to cross-compile the target kernel by building it on a Host machine. Cross-compiling can result in significantly faster builds than compilation on the Cedar Trail target itself.

The steps below assume that Fedora-14 is installed in Host along with packages required for compilation. If your host does not have a Fedora-14 installed, then you can follow steps similar to section 4.2 to install it on the Host, except that you can use either 32-bit or 64-bit Fedora-14 Linux.

Follow the following steps to build the Kernel to be installed on Cedar Trail target board.

1. Go to the website:

   http://www.kernel.org/pub/linux/kernel/v3.x/

2. Look for the following file and download it

   linux-3.1.tar.gz
3. Get the patch tarball from your FAE (cdv-gfx-drivers-1.0.1_bee.tar.bz2)

4. Extract the contents of the patch in home directory

   **tar xjvf cdv-gfx-drivers-1.0.1_bee.tar.bz2**

5. Untar the kernel in home directory

   **tar xzf linux-3.1.tar.gz**

   It creates a directory named " linux-3.1" and untars all the files into this directory.

6. Apply the kernel patches by using the patch utility

   **cd linux-3.1**

   **patch –p1 < ../cdv-gfx-drivers-1.0.1_bee/src/kernel-ttm-clear-high.patch**

   **patch –p1 < ../cdv-gfx-drivers-1.0.1_bee/cedarview-kernel-v1.0.1_bee.patch**

7. Edit drivers/staging/Kconfig and add the following line:

   **source "drivers/staging/cdv/Kconfig"**

8. Edit drivers/staging/Makefile and add the following line:

   **obj-$(CONFIG_DRM_INTEL_CDV) += cdv/**

9. We are now ready to build the kernel

10. Note that the next step will use the config file from the current installations /boot directory which could be a PAE image. We will want to make sure that we reset that option with menuconfig.

11. Configure and Compile the kernel

    **linux32 make menuconfig**

The kernel configuration window pops-up and looks like the following:

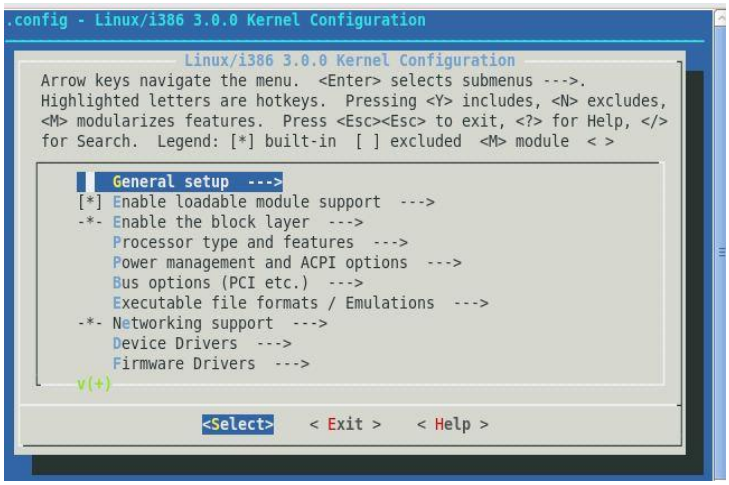


**Figure 2: Menu Config Utility Screen**

1. Select "Processor type and features --- >" by using the arrow key and enter
2. Select "High Memory Support (64GB) --->" by using the arrow key and enter
3. If you see "High Memory Support (4GB) -->" instead, skip the next 2 step.
4. Move the cursor to 4GB and hit spacebar to select
5. Exit one level
6. select "Device driver --- >" by using arrow key and enter
7. select "staging drivers --- >" by using arrow key and enter

8. Select “Intel CDV (load along with IMG driver)” and hit the space bar. This will load the release driver.

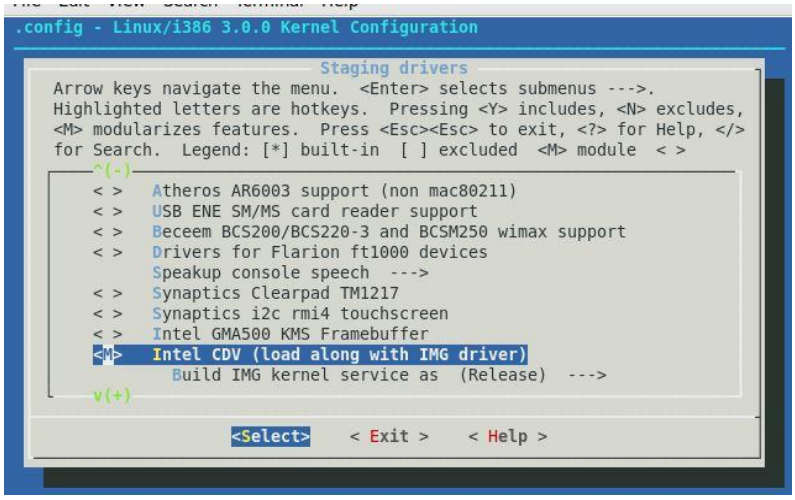


**Figure 3: Menu config Utility - Intel CDV Driver selection**

9. Exit and save

13 Make rpm to load into the target

**linux32 make rpm**

14 During build process “rpmbuild” directory is created in your home directory and there are four directories created under rpmbuild.

**a. cd ~/rpmbuild/RPMS/i386**

b. Copy the following two files to a USB drive

kernel-3.1.0-1.i386.rpm

kernel-headers-3.1.0-1.i386.rpm

15. Now we install the new kernel on the target system.

a. First Login as root on the target system

b. Create a “cedartrail” dir on target

**mkdir cedartrail**

c. Copy the following files from the USB to cedartrail directory

kernel-3.1.0-1.i386.rpm

kernel-headers-3.1.0-1.i386.rpm

d. Install the kernel

**rpm –ivh kernel-3.1.0-1.i386.rpm**

**rpm –ivh kernel-headers-3.1.0-1.i386.rpm**

Now you should see a new directory called ‘3.1.0’ under /lib/module

16. Make the ramdisk

**mkinitrd /boot/initramfs-3.1.0.img 3.1.0**

17. Add the new kernel to the grub files

**grub2-mkconfig –o /boot/grub2/grub.cfg**

18. Edit **/etc/default/grub** as follows:

    Add "video=LVDS-1:d" in the kernel line if your display is not LVDS. This is a work-around to an issue that causes driver to believe that LVDS display is connected.

    **grub2-mkconfig –o /boot/grub2/grub.cfg**

19. Unpack and install cedarview-libwsbm binary tarball relative to your root filesystem. Assume patch tarball was extracted in /home/jim:

    **cd /**
    **tar xvf /home/jim/cdv-gfx-drivers-1.0.1_bee/bin/cedarview-libwsbm-1.1.0.tar**

20. Install libva and dependencies from your package manager.

    **yum –y install libva**

21. Unpack and install the appropriate userspace (Xorg, 2D, 3D) Cedarview drivers relative to your root filesystem. While the PowerVR driver is a Mesa replacement for EGL and GL ES, mesa-libGL of the stated version is required for OpenGL operation. Note that development headers are installed as well.

    **tar xvf /home/jim/cdv-gfx-drivers-1.0.1_bee/bin/cedarview-userspace-v1.0.1_bee.tar.bz2**

22. Unpack the PVR VA-API driver for Cedartrail-accelerated H.264, MPEG-2, and VC1 streams:

    **tar xvf /home/jim/cdv-gfx-drivers-1.0.1_bee/bin cedarview-vaapi-v1.0.1_bee.tar**

23. Reboot and make sure you select the correct new image when grub displays the list of choices. You can now use the following command to set the default:

    **grub2-set-default <title or number>**

## 4.4 Build Kernel on Target and Install

Follow these steps only if you intend to natively build the Kernel to be installed on Cedar Trail target board itself. The build time is approximately 3 hours. Now we have a hard disk with Fedora 16 Xfce installed for the target based on the steps described in section 4.2.

1. Go to the website:

   http://www.kernel.org/pub/linux/kernel/v3.x/

2. Look for the following file and download it

   linux-3.1.tar.gz

3. Get the patch tarball from your FAE (cdv-gfx-drivers-1.0.1_bee.tar.bz2)
4. Extract the contents of the patch in home directory

   **tar xjvf cdv-gfx-drivers-1.0.1_bee.tar.bz2**

5. Untar the kernel in home directory

   **tar xzf linux-3.1.tar.gz**

   It creates a directory named " linux-3.1" and untars all the files into this directory.

6. Apply the kernel patches by using the patch utility

   **cd linux-3.1**

   **patch –p1 < ../cdv-gfx-drivers-1.0.1_bee/src/kernel-ttm-clear-high.patch**

   **patch –p1 < ../cdv-gfx-drivers-1.0.1_bee/cedarview-kernel-v1.0.1_bee.patch**

7. Edit drivers/staging/Kconfig and add the following line:

   **source "drivers/staging/cdv/Kconfig"**

8. Edit drivers/staging/Makefile and add the following line:

   **obj-$(CONFIG_DRM_INTEL_CDV) += cdv/**

9. We are now ready to build the kernel

10. Note that the next step will use the config file from the current installations /boot directory which is a PAE image. We will want to make sure that we reset that option with menuconfig.

11. Configure and Compile the kernel

    **make menuconfig**

The kernel configuration window pops-up and looks like the following:

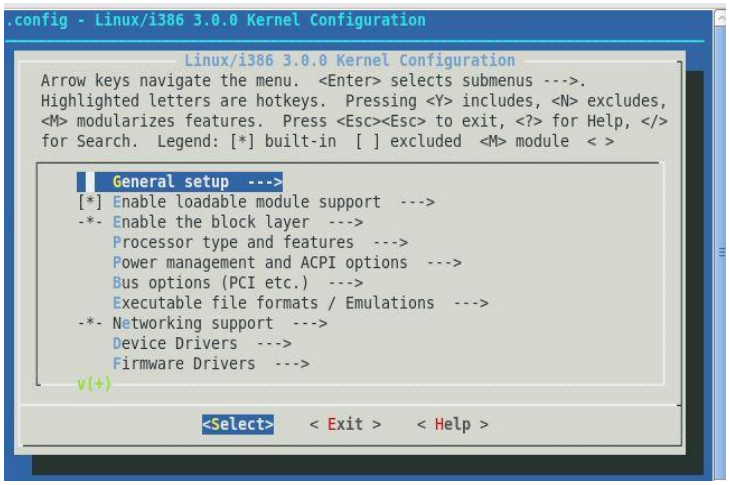


**Figure 4: Menu Config Utility Screen**

24. Select "Processor type and features --- >" by using the arrow key and enter
25. Select "High Memory Support (64GB) --->" by using the arrow key and enter
26. Move the cursor to 4 GB and hit spacebar to select
27. Exit one level
28. Select "Device driver --- >" by using arrow key and enter
29. Select "staging drivers --- >" by using arrow key and enter
30. Select "Intel CDV (load along with IMG driver)" and hit the space bar. This will load the release driver.

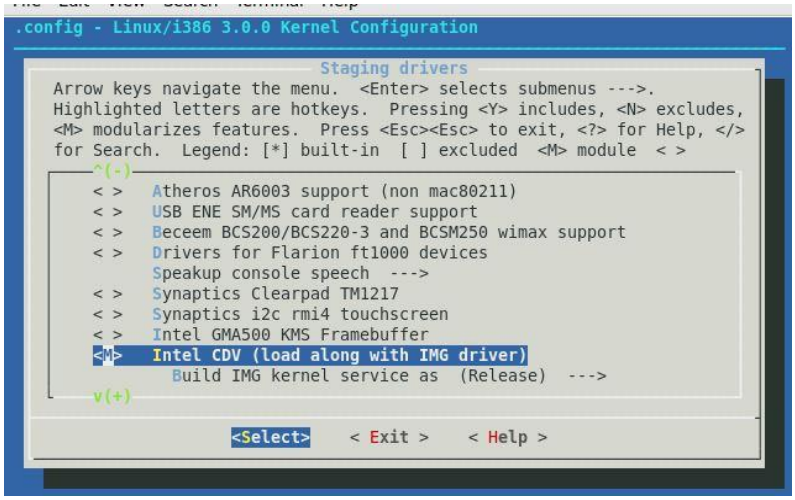


**Figure 5: Menu config Utility - Intel CDV Driver selection**

31. Exit and save

32. Make and Install the Kernel

    **make**

    **su –c "make modules_install"**

    **su –c "make install"**

    If the make is successful, then the kernel should be there. Check that ramdisk was made in the /boot dir.

33. Unpack and install cedarview-libwsbm binary tarball relative to your root filesystem. Assume patch tarball was extracted in /home/jim:

    **cd /**
    **tar xvf /home/jim/cdv-gfx-drivers-1.0.1_bee/bin/cedarview-libwsbm-1.1.0.tar**

34. Install libva and dependencies from your package manager.

    **yum –y install libva**

35. Unpack and install the appropriate userspace (Xorg, 2D, 3D) Cedarview drivers relative to your root filesystem. While the PowerVR driver is a Mesa replacement for EGL and GL ES, mesa-libGL of the stated version is required for OpenGL operation. Note that development headers are installed as well.

    **tar xvf /home/jim/cdv-gfx-drivers-1.0.1_bee/bin/cedarview-userspace-v1.0.1_bee.tar.bz2**

36. Unpack the PVR VA-API driver for Cedartrail-accelerated H.264, MPEG-2, and VC1 streams:

    **tar xvf /home/jim/cdv-gfx-drivers-1.0.1_bee/bin cedarview-vaapi-v1.0.1_bee.tar**

37. Edit **/etc/default/grub** as follows:

    Add "video=LVDS-1:d" in the kernel line if your display is not LVDS. This is a work-around to an issue that causes driver to believe that LVDS display is connected.

    **grub2-mkconfig –o /boot/grub2/grub.cfg**

38. Reboot and make sure you select the correct new image when grub displays the list of choices. You can now use the following command to set the default:

    **grub2-set-default <title or number>**

## 4.5 Install Packages on Target

There are several dependencies for the Graphics driver and MPlayer. The steps below show how to install the required packages on the target system. **ASSUMPTION IS INITIAL INSTALLATION WAS WITH "Software Development" chosen.**

1. Install Xorg Packages as root

   **yum –y install libXi-devel libXfont-devel libXext-devel**

   **yum –y install libdrm-devel**

   **yum –y install libpciaccess-devel pixman-devel**

   **yum -y install expat-devel**

   **yum –y install libva-devel freeglut-devel**

## 4.6 MPlayer compilation on Target

We are going to build MPlayer with hardware accelerated video decoding in the target system. Before proceeding, ensure that the system has a internet connection.

1. Install yasm and git
   **yum –y install yasm git**

2. Install some alsa dependencies
   **yum –y install alsa***

3. Clone MPlayer git repository
   **git clone git://gitorious.org/vaapi/mplayer.git**

   If the command executes successfully, then MPlayer tree is downloaded in target under "mplayer" directory

4. Configure MPlayer
   **cd mplayer**
   **git checkout –t origin/hwaccel-vaapi**
   **./configure**

   Hit "enter" at "No FFmpeg" checkout prompt

   **cd ffmpeg**
   **git checkout –b ffmpeg-0.6.3** [you can use the latest version tag]

5. Get out of ffmpeg directory
   **cd ..**

39. Now we are in Mplayer directory. Configure MPlayer with options as follows
    **./configure --prefix=/usr/local --enable-xv --enable-gl --enable-vaapi --disable-vdpau --disable-mencoder --disable-faad --enable-runtime-cpudetection**

6. Compile Mplayer
   **make**
   **make install**

# 5 Playback Video and Check CPU Utilization

1. Ensure that Mplayer is built with vaapi enabled
   **mplayer –vo help**
   "vaapi" should be among the list of video output drivers listed by the above command.

2. Play the video with Mplayer using hardware acceleration.
   **mplayer -vo vaapi -va vaapi -ao alsa <example_video_clip>.mp4**

3. Check that CPU utilization using "top" command
   Open another terminal and type
   **top**

   Max CPU utilization for Mplayer application should be about 20% (with 1080p videos)

# 6 Run Glxgears and Check CPU Utilization

4. Check if OpenGL based hardware acceleration is enabled
   **glxinfo | grep renderer**
   The string "OpenGL renderer string: PowerVR SGX545" should be output, indicating that the PVR-CDV OpenGL driver has been correctly installed.

   **glxinfo|grep direct**
   The string "direct rendering: Yes" should be output, indicating that Graphics hardware acceleration is enabled

5. Launch the glxgears demo
   **glxgears -fullscreen**
   A full screen animation of three rotating gears would start to play
   Hit "Esc" key to stop glxgears

6. Launch the glxgears demo (non-Full screen)
   **glxgears**

   The glxgears animation will start in a new window.

7. Check that CPU utilization using "top" command
   Open another terminal and type
   **top**
   Max CPU utilization for glxgears application should be less than 2 %.

# 7 References

1) VA-API,
http://www.freedesktop.org/wiki/Software/vaapi

2) MeeGo. The MeeGo project site hosts the Cedar Trail Graphics Drivers and Kernel patches (Released as part of Cedar Trail Netbook platform). Refer to this site to check for any updates.
<https://meego.com/>

| **Acronyms** | |
|---|---|
| API | Application Programming Interface |
| AVC | Audio Video Coding |
| DP | DisplayPort |
| eDP | Embedded Display Port |
| HDMI | High Definition Multimedia Interface |
| IDCT | Inverse Discrete Cosine Transform |
| LVDS | Low Voltage Differential Signaling |
| HD | High Definition |
| VA-API | Video Acceleration API |
| VGA | Video Graphics Array |
| VLD | Variable Length Decoding |
| WMV3 | FourCC notation for Windows Media Video 9 |

**Table 1: Acronym Table**