San Jose State University

Department of Electrical Engineering

EE296A Xilinux/OpenCV porting on Zybo Board

Step 1: Installing an operating system and configuring hardware

- ✓ Download boot partition kit for the Xilinux Zynq Zybo board.
  http://xillybus.com/downloads/xillinux-eval-zybo-1.3c.zip

- ✓ Download the SD card image.
  http://xillybus.com/downloads/xillinux-1.3.img.gz

- ✓ Download Xilinx Vivado HL WebPACK
  http://www.xilinx.com/support/download.html
  Vivado HLx 2016.2: Windows Web Installer (EXE - 50.41 MB)

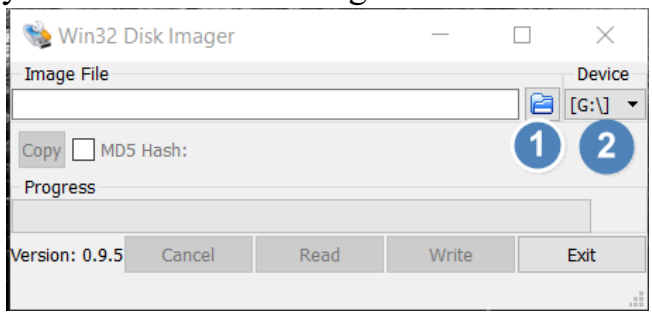
1. Unzip the previously downloaded xillinux-eval-zybo-1.3c.zip file into a working directory.
2. Start Vivado. With no project open, Pick Tools > Run Tcl Script and choose xillydemo-vivado.tcl in the verilog/ subdirectory, which you extracted in the previous step. A sequence of events takes place for less than a minute. The success of the project's deployment can be verified by choosing the "Tcl Console" tab at Vivado's window's bottom, and verify that it says
   INFO: Project created: xillydemo
3. After the project has been created, run an implementation: Click "Generate Bitstream" on the Flow Navigator bar to the left.

4. A popup window asking if it's OK to launch synthesis and implementation is likely to appear – pick "Yes". Vivado runs a sequence of processes. This takes normally takes a few minutes. Several warnings are issued, some of which may be classified critical. There should be no errors. A popup window, informing that the bitstream generation was completed successfully will appear, giving choices of what to do next. Any option is fine, including picking "Cancel". The bitstream file, **xillydemo.bit**, can be found at verilog\vivado\xillydemo.runs\impl_1

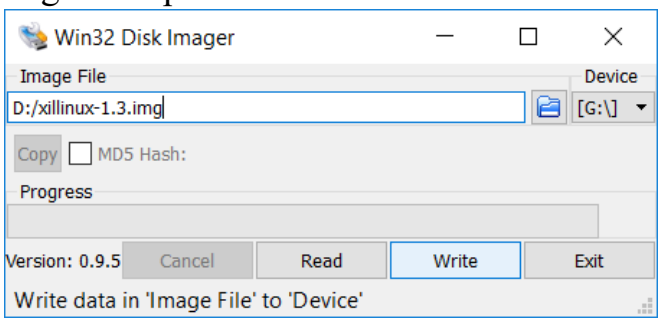
Loading the (Micro)SD with the image

Download Win32 Disk Imager
https://sourceforge.net/projects/win32diskimager/

Insert the SD card into the card reader of your laptop. Extract **xillinux-1.3.img** from **xillinux-1.3.img.gz** that you downloaded previously to a working directory. Start Win32DiskImager



Select **xillinux-1.3.img** by clicking on **1,** and select your SD card by clicking on **2**. After selecting both options click on the Write button below.



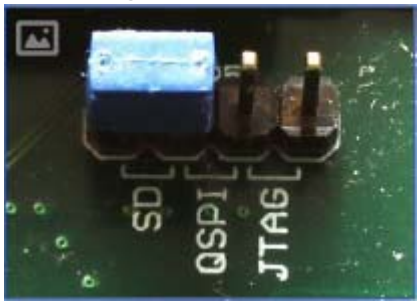
After W32DiskImager finishes its operation copy **boot.bin** and **devicetree.dtb** from the bootfiles subdirectory, into the (Micro)SD card's boot partition (the first partition). Bootfiles directors is located inside **xillinux-eval-zybo-1.3c** which was extracted previously. Also copy

xillydemo.bit that was generated in a previous step 4 (from the Verilog\ subdirectory).

The SD card should have the following files

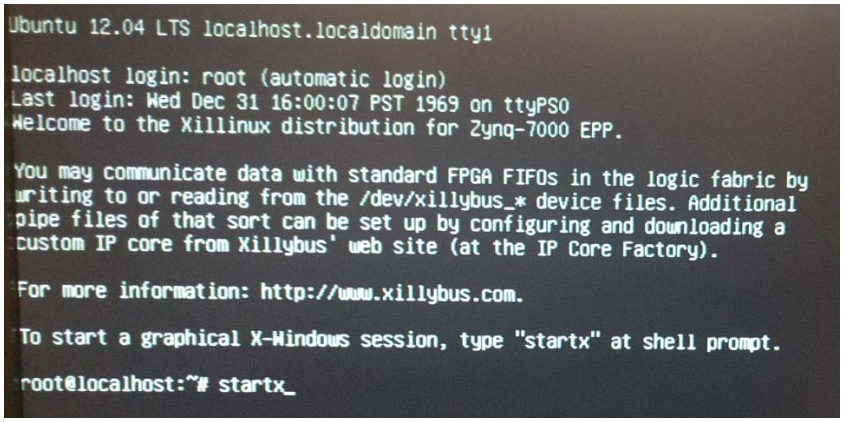
- uImage
- boot.bin
- devicetree.dtb
- xillydemo.bit
-

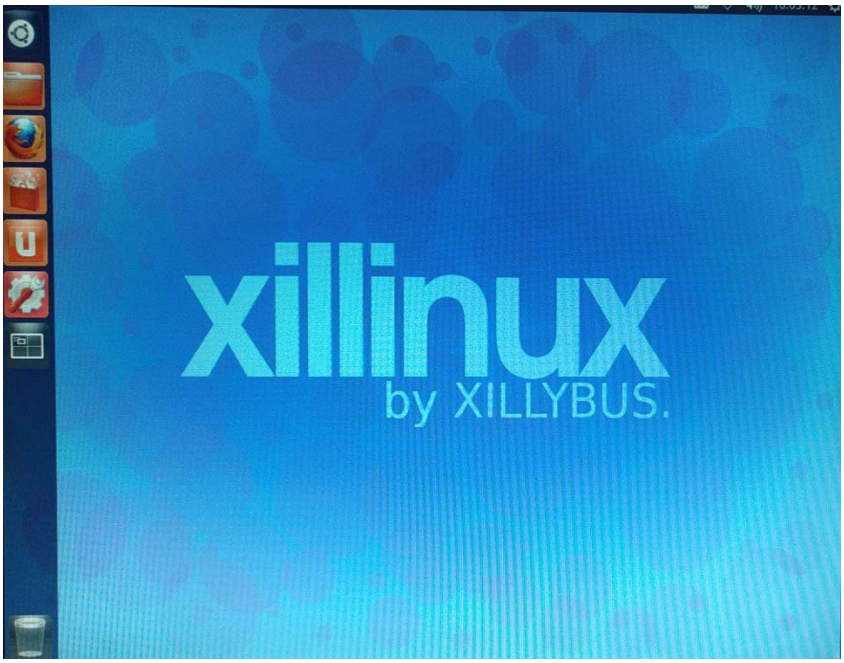
Insert the SD card in the Zybo board. The boot mode is selected by a jumper near to the VGA connector, which should be set on the two pins marked with “SD”, as shown on this image:



The other jumpers are set depending on the desired operating mode. For example, the power supply jumper can be set to take power from an external 5V source, or from the UART USB jack – both are fine for booting Xillinux.

At the end of the boot process, Type “startx” at command prompt to launch a Gnome graphical desktop. The desktop takes some 15-30 seconds to initialize.

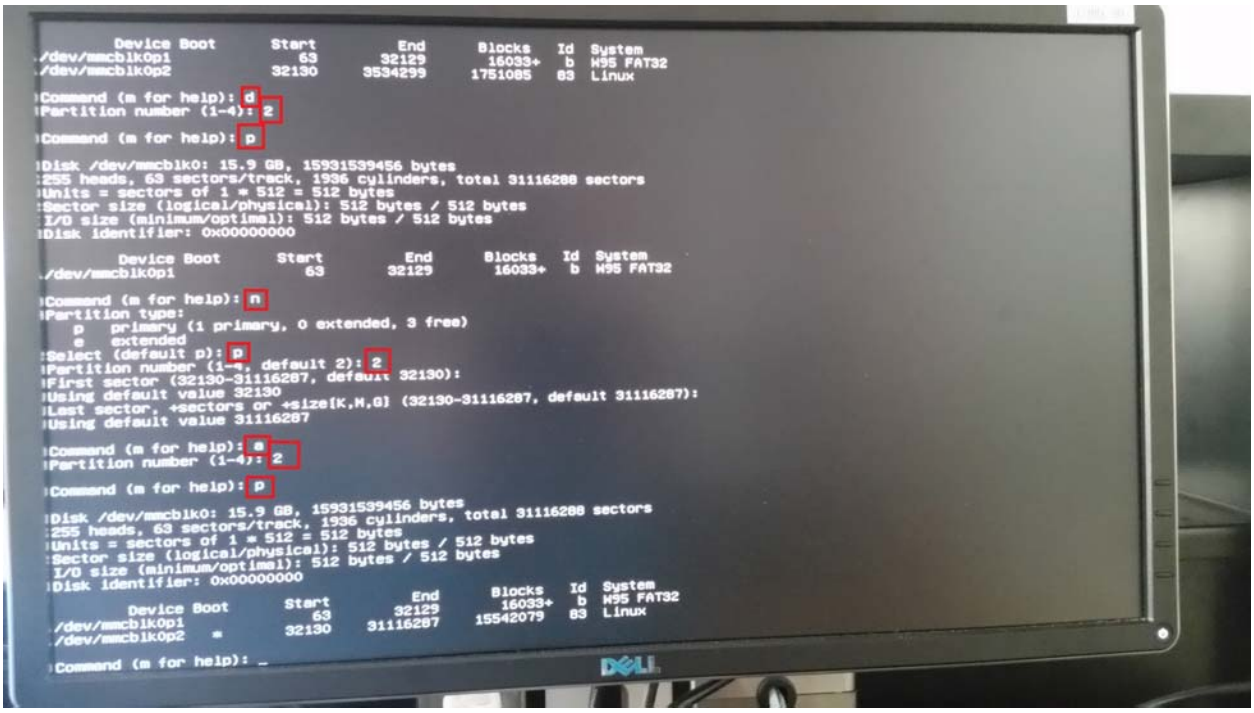




Step 1.5: Increasing partition space to fill SD card.

Because the image we flashed onto our SD card would only allow 1.67Gb of space, we need to increase it to maximum capacity to fully install OpenCV.

1. sudo fdisk /dev/mmcblk0



2. p , lists partitions
3. d, deletes
   - d
   - 2

4. n, makes new
   - n
   - p , select primary partition
   - 2 , make it the 2nd partition
   - enter , sets minimum
   - enter , sets maximum/default

5. a , toggle bootable flag
   - a
   - 2

6. w , write to disk and change it permanently

7. sudo reboot now, reboots device
8. sudo resize2fs /dev/mmcblk0p2, expands filesystem to take up all space on partition

Step 2: Installing OpenCV

NOTE: Text inside a black box is a command that you would have to enter in Terminal. Text inside the red boxes is also a command but they will take more than one line in terminal. Text in black boxes will take only one line.

1. Remove any installed versions of ffmpeg and x264.

```
sudo apt-get remove ffmpeg x264 libx264-dev
```

2. Edit sources.list file

```
gedit /etc/apt/sources.list
```

It should contain the following content

Major bug fix updates produced after the final release of the
distribution.
deb http://ports.ubuntu.com/ubuntu-ports/ precise-updates main restricted
deb-src http://ports.ubuntu.com/ubuntu-ports/ precise-updates main restricted

deb http://ports.ubuntu.com/ubuntu-ports/ precise multiverse
deb-src http://ports.ubuntu.com/ubuntu-ports/ precise multiverse
deb http://ports.ubuntu.com/ubuntu-ports/ precise multiverse
deb-src http://ports.ubuntu.com/ubuntu-ports/ precise-updates multiverse

3. Get all the dependencies for x264 and ffmpeg.

```
sudo apt-get update

sudo apt-get install build-essential checkinstall git cmake libfaac-dev libjack-jackd2-dev libmp3lame-dev libopencore-amrnb-dev libopencore-amrwb-dev libsdl1.2-dev libtheora-dev libva-dev libvdpau-dev libvorbis-dev libx11-dev libxfixes-dev libxvidcore-dev texi2html yasm zlib1g-dev
```

4. Download and install gstreamer.

```
sudo apt-get install libgstreamer0.10-0 libgstreamer0.10-dev gstreamer0.10-tools gstreamer0.10-plugins-base libgstreamer-plugins-base0.10-dev
```

gstreamer0.10-plugins-good gstreamer0.10-plugins-ugly gstreamer0.10-plugins-bad gstreamer0.10-ffmpeg

5. Download and install gtk.

```
sudo apt-get install libgtk2.0-0 libgtk2.0-dev
```

6. Download and install libjpeg.

```
sudo apt-get install libjpeg8 libjpeg8-dev
```

7. Enable Swap on your installation

```
sudo fallocate -l 4G /swapfile
chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile
```

8. Append the following entry in /etc/fstab file to enable swap on system reboot.

```
/swapfile   none    swap    sw    0   0
```

9. Edit **/etc/sysctl.conf** file and append following configuration in file.

```
vm.swappiness=10
```

10. Now reload the sysctl configuration file, This should return something similar to what you inputted in the sysctl.conf file.

```
sudo sysctl -p
```

11. Create a directory to hold source code.

```
cd ~
mkdir src
```

12. Download and install install x264.

```
cd ~/src
```

wget ftp://ftp.videolan.org/pub/videolan/x264/snapshots/x264-snapshot-20120528-2245-stable.tar.bz2

cd x264-snapshot-20120528-2245-stable

13.Configure and build the x264 libraries.

```
./configure --enable-shared --enable-pic
make
sudo make install
```

14.Download ffmpeg version 0.11.1

```
cd ~/src
wget http://ffmpeg.org/releases/ffmpeg-0.11.1.tar.bz2
tar xvf ffmpeg-0.11.1.tar.bz2
cd ffmpeg-0.11.1
```

15.Configure and build ffmpeg. This will take a while.

```
./configure --enable-gpl
--enable-libfaac --enable-libmp3lame --enable-libopencore-amrnb --enable-
libopencore-amrwb --enable-libtheora --enable-libvorbis --enable-libx264 --
enable-libxvid --enable-nonfree --enable-postproc --enable-version3 --enable-
x11grab --enable-shared --enable-pic

make
sudo make install
```

16.Download and install install a recent version of v4l (video for linux)

```
go to: https://www.linuxtv.org/downloads/v4l-utils/v4l-utils-0.8.8.tar.bz2
in terminal:
mv ~/Downloads/v4l-utils-0.8.8.tar.bz2 ~/src
tar xvf v4l-utils-0.8.8.tar.bz2
```

```
cd v4l-utils-0.8.8
make
sudo make install
```

17. Download and install install OpenCV 2.4.2.

```
cd ~/src

wget http://pkgs.fedoraproject.org/repo/pkgs/opencv/OpenCV-2.4.2.tar.bz2/059ef86fc1724d69b75832a0d2929ff5/OpenCV-2.4.2.tar.bz2

tar xvf OpenCV-2.4.2.tar.bz2
```

18. Create a new build directory and run cmake

```
cd OpenCV-2.4.2/
mkdir build
cd build
cmake -D CMAKE_BUILD_TYPE=RELEASE ..
```

19. Build and install OpenCV. This will also take a long while.

```
make
sudo make install
```

20. Tell linux where the shared libraries for OpenCV are located by entering the following shell command

```
export LD_LIBRARY_PATH=/usr/local/lib
```

21. Add the following line to the end of file /etc/ld.so.conf.d/opencv.conf

```
/usr/local/lib
```

22. After editing and saving the file, enter the following command on shell

```
sudo ldconfig /etc/ld.so.conf
```

23. Add the following lines to the end of file /etc/bash.bashrc

```
PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig
```

```
export PKG_CONFIG_PATH
```

After completing the previous steps, your system should be ready to compile code that uses the OpenCV libraries.