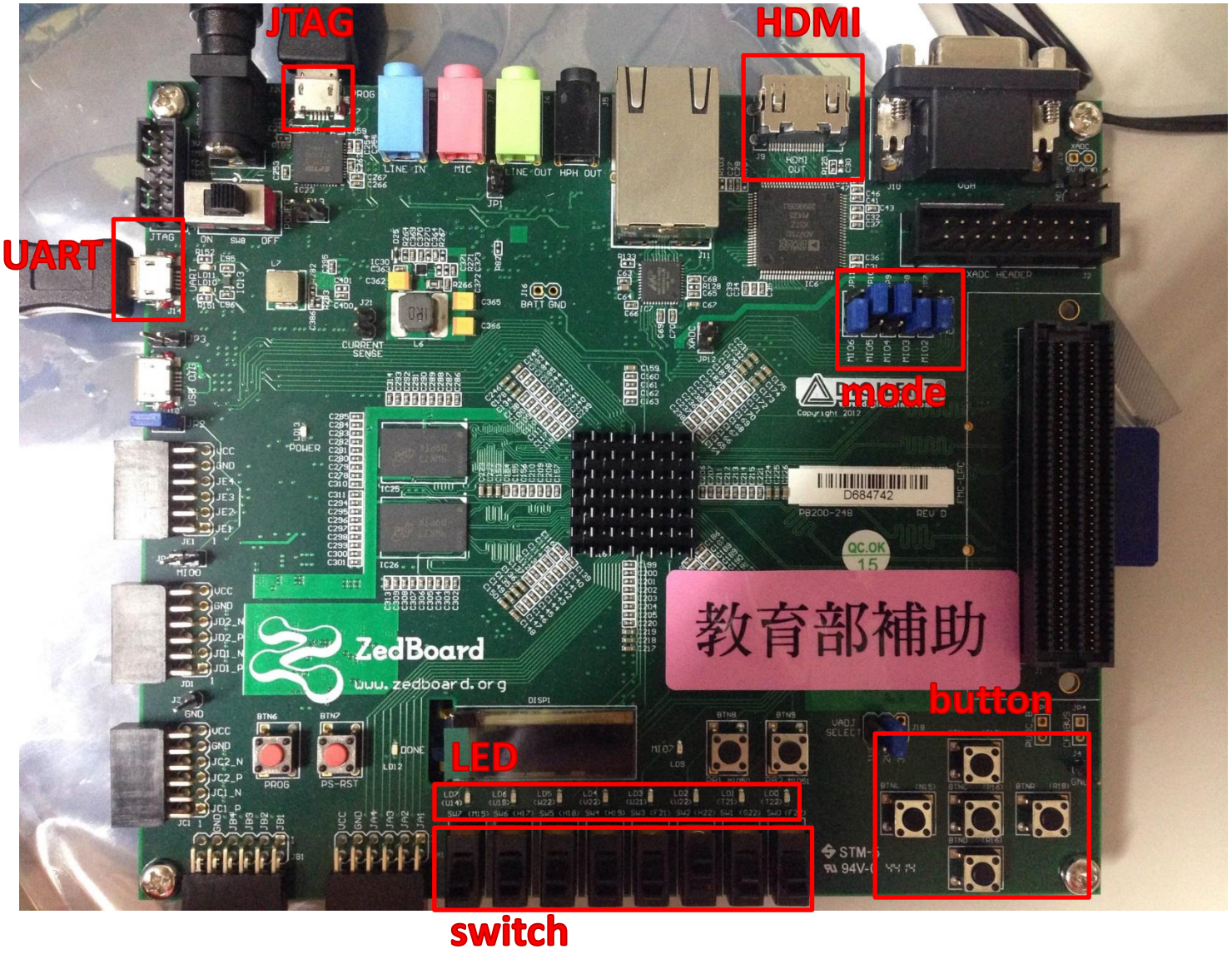
LAB 1
Platform, OS and basic I/O setup

What would you get?

1. Zedboard
2. MicroUSB to USB cable (male to male)
3. OTG USB cable (male MicroUSB to female USB)
4. SD card
5. 12V power supply

6. Cat 5e lan cable
7. HDMI cable
8. HDMI to DVI adapter
9. MicroUSB to USB cable

Zedboard


JTAG
HDMI
UART
mode
button
LED
switch
教育部補助
ZedBoard
www.zedboard.org

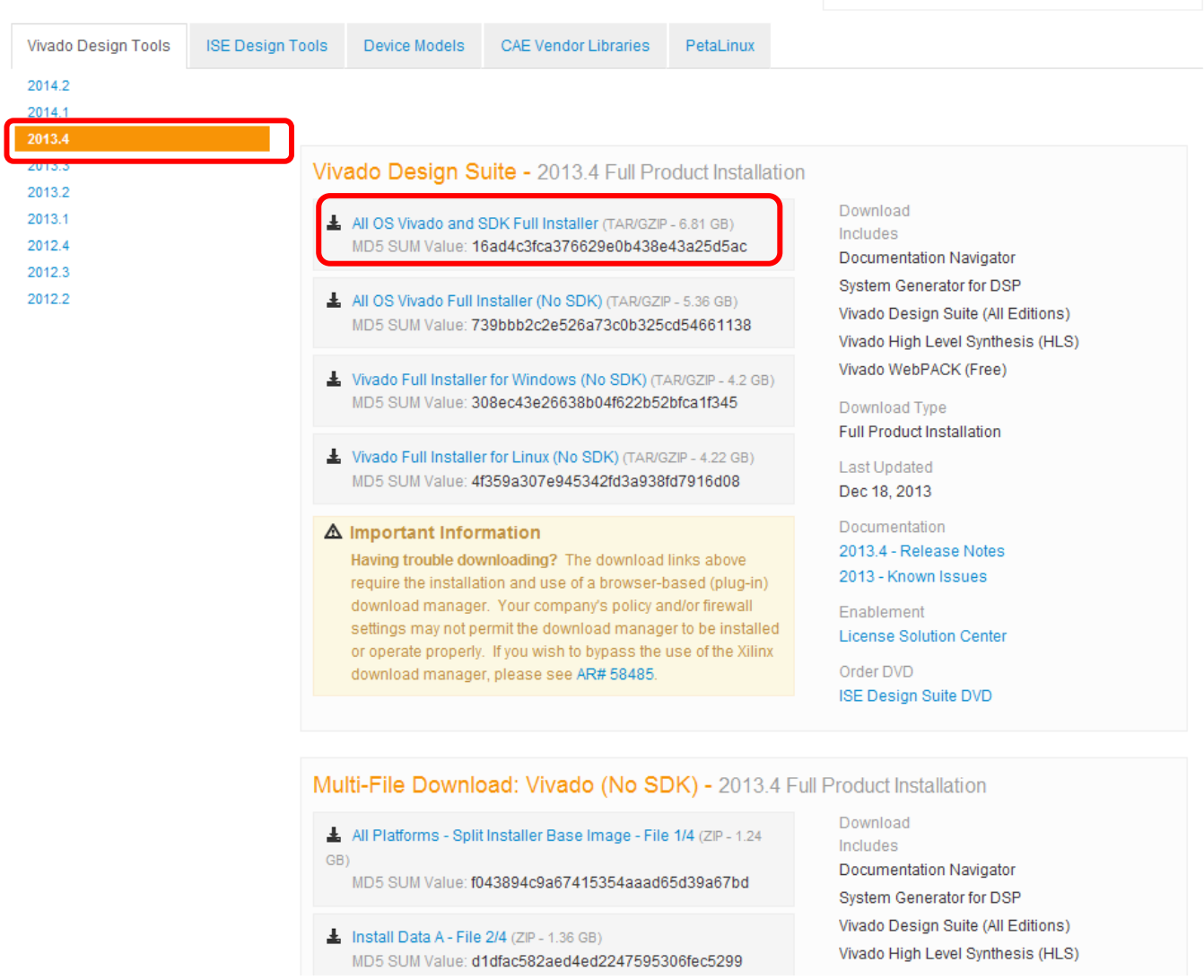
Environment

- OS: Ubuntu 14.04 64-bit
- Vivado Design Suite
  - Vivado: for hardware design
  - Xilinx SDK: for software design
- Petalinux Tools
  - Linux kernel
- JTAG Driver on Ubuntu
  - http://svenand.blogdrive.com/archive/172.html#.VqG0Evl9670

LAB 1.1

Platform setup and development environment setup

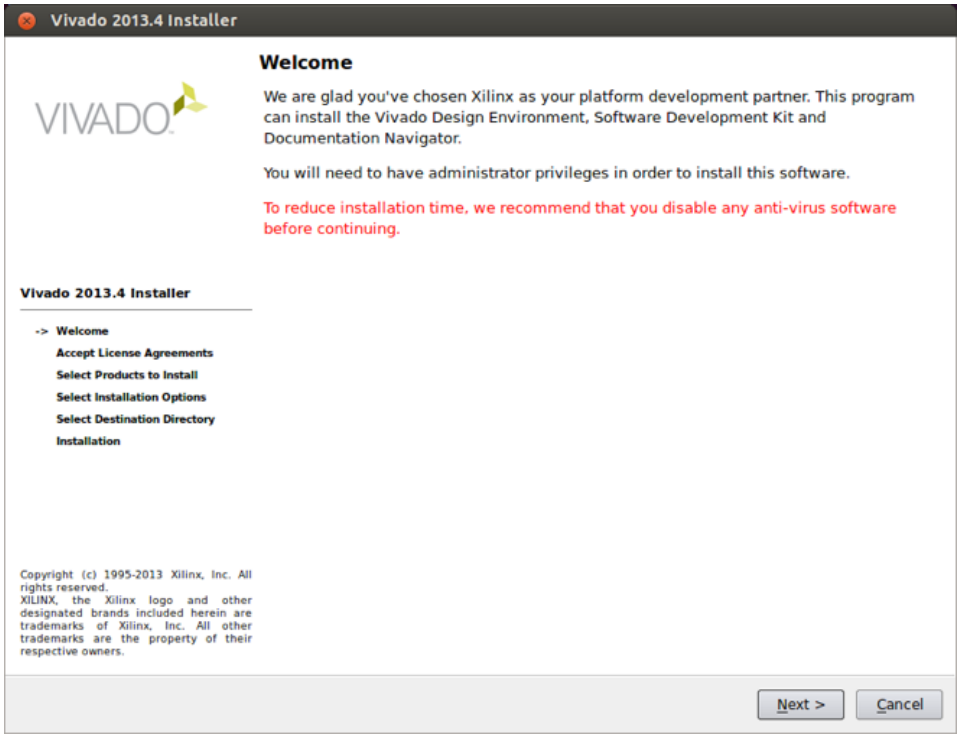
Vivado Design Suite - Download



http://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/vivado-design-tools.html

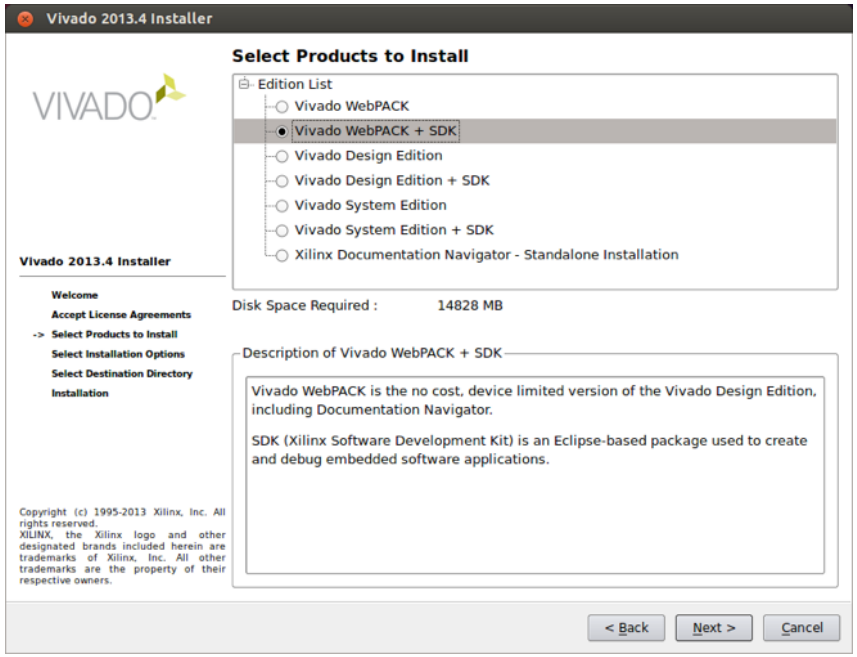
Vivado Design Suite – Install

- Host> cd Downloads
- Host> tar xvf Xilinx_Vivado_SDK_2013.4_1210_1.tar
- Host> sudo mkdir /opt/Xilinx
- Host> cd Xilinx_Vivado_SDK_2013.4_1210_1
- Host> sudo ./xsetup



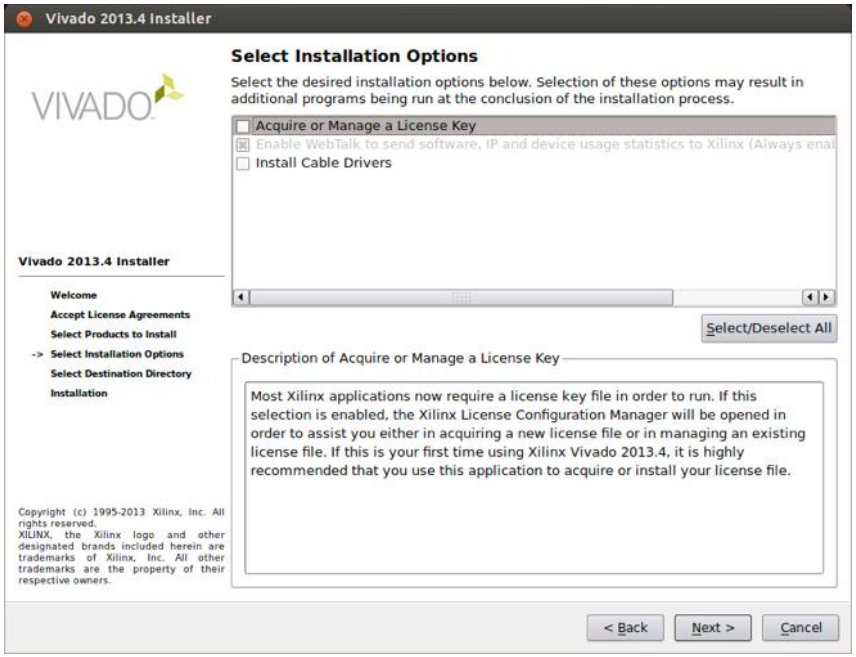
Vivado Design Suite – Install

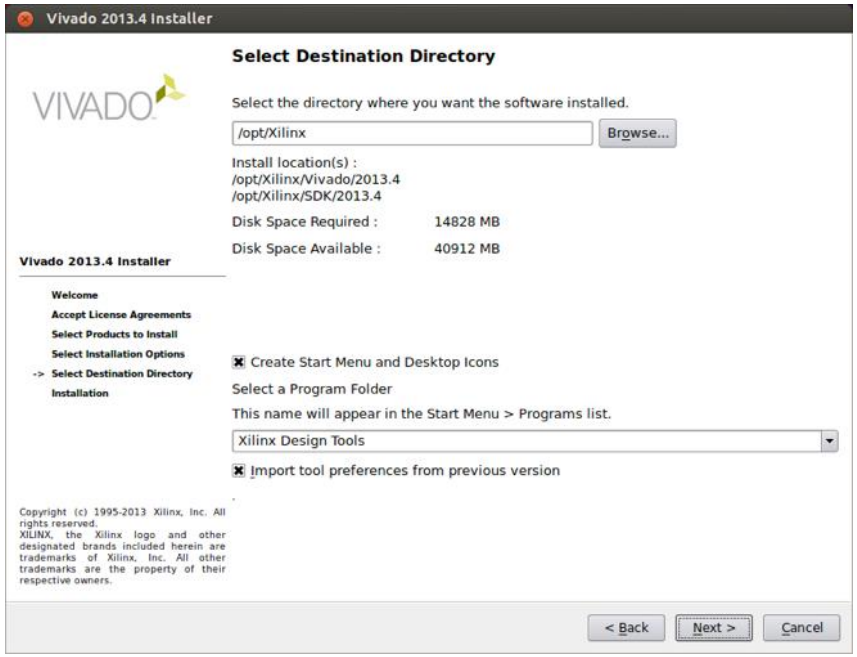
Install WebPack + SDK



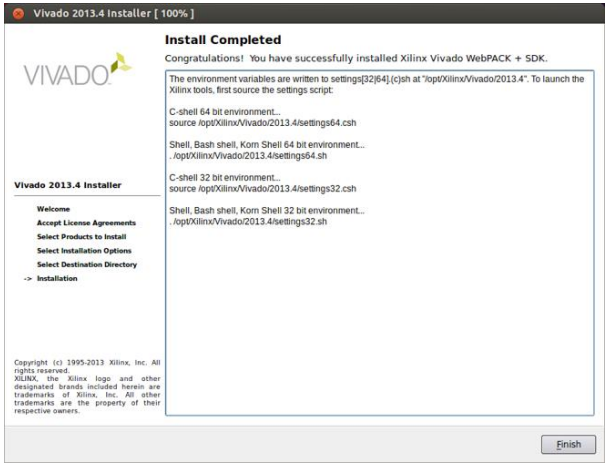
Vivado Design Suite – Install

Don’t choose two installation options





Vivado Design Suite – Install



- Host> vi ~/.bashrc
  - Add

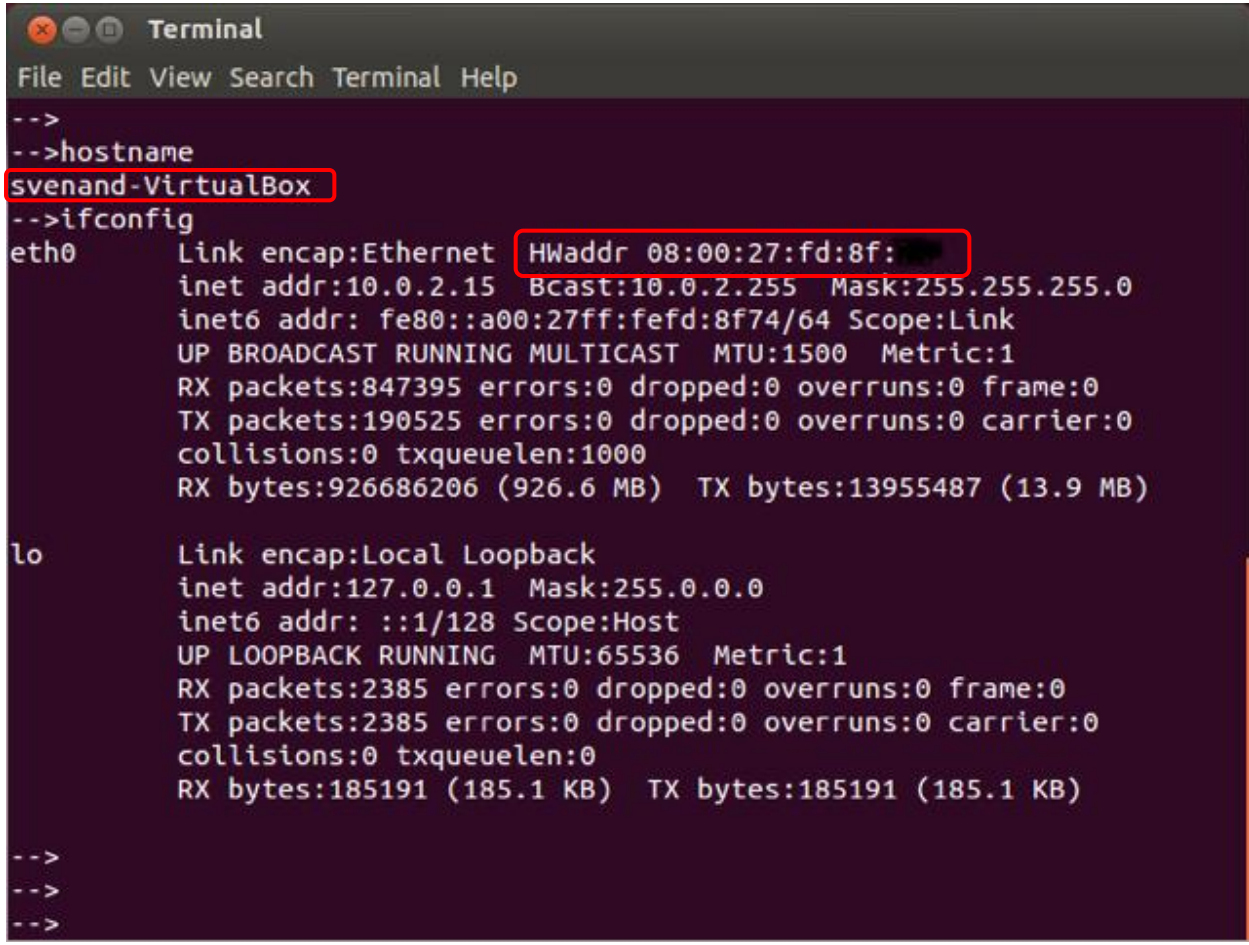
  “source /opt/Xilinx/Vivado/2013.4/settings64.sh” (64bit Ubuntu)

  or

  “source /opt/Xilinx/Vivado/2013.4/settings32.sh” (32bit Ubuntu)

Vivado Design Suite – Acquire License

- Host> hostname
- Host> ifconfig



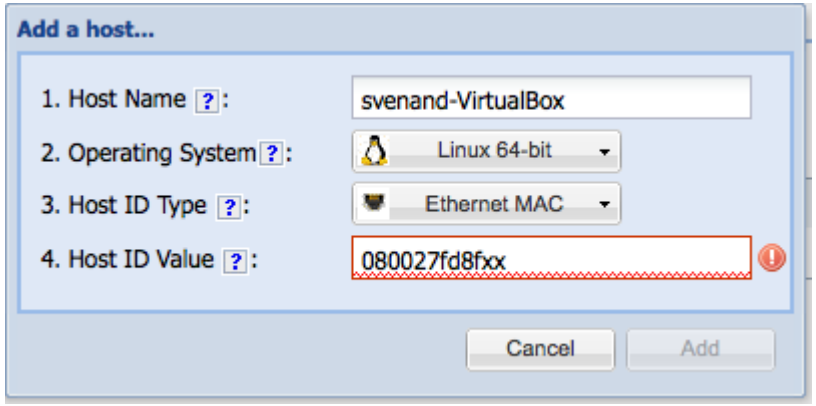
Vivado Design Suite – Acquire License

Check PetaLinux Tools and ISE WebPACK

Certifcate Based Licenses

	Product	Type	License	Available Seats	Status	Subscription End Date
☑	ISE WebPACK License	Certificate - No Charge	Node	1/1	Current	None
☑	PetaLinux Tools License	Certificate - Evaluation	Node	1/1	Current	365 days
☐	PetaLinux Tools License, Floating License	Certificate - Evaluation	Floating	1/1	Current	365 days
☐	Vivado HLS Evaluation License	Certificate - Evaluation	Node	1/1	Current	30 days

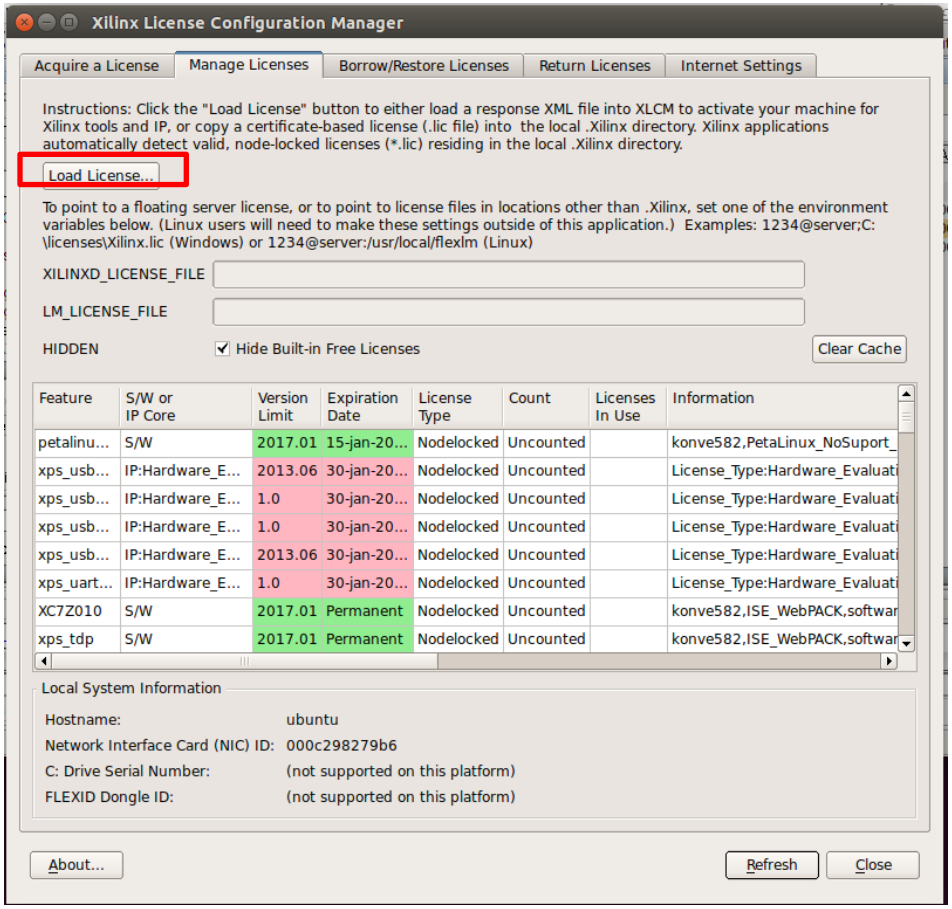
Input your hostname and MAC address



http://www.xilinx.com/getlicense

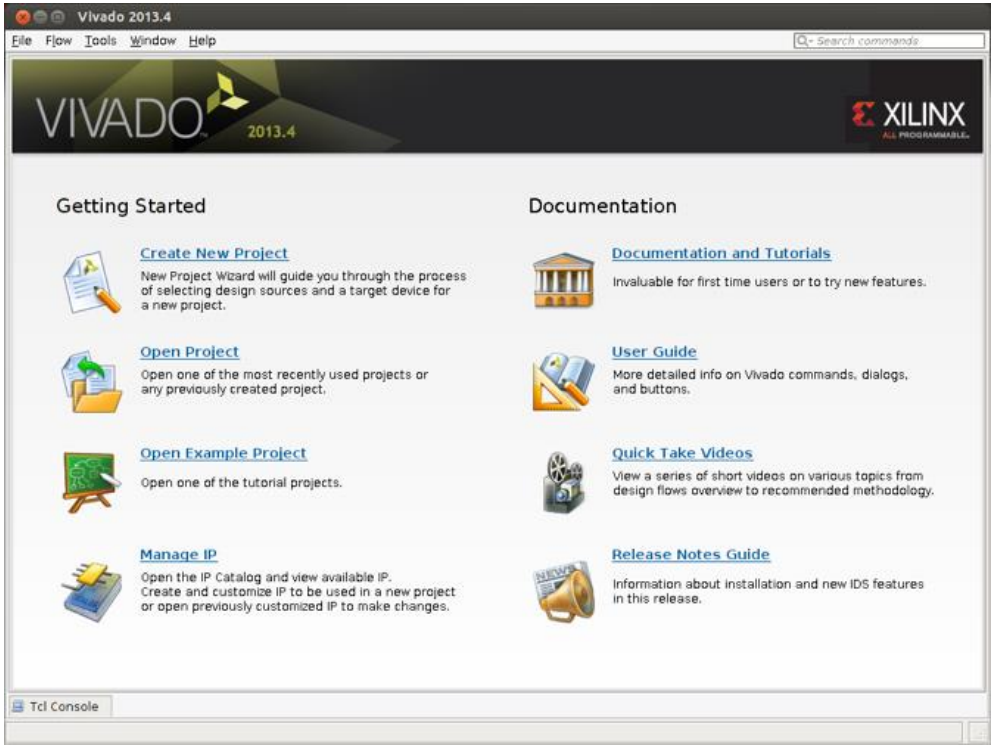
Vivado Design Suite – Manage License

- Download the license from the e-mail
- Host > vivado &
- Help -> Manage License
- Click *Load License*

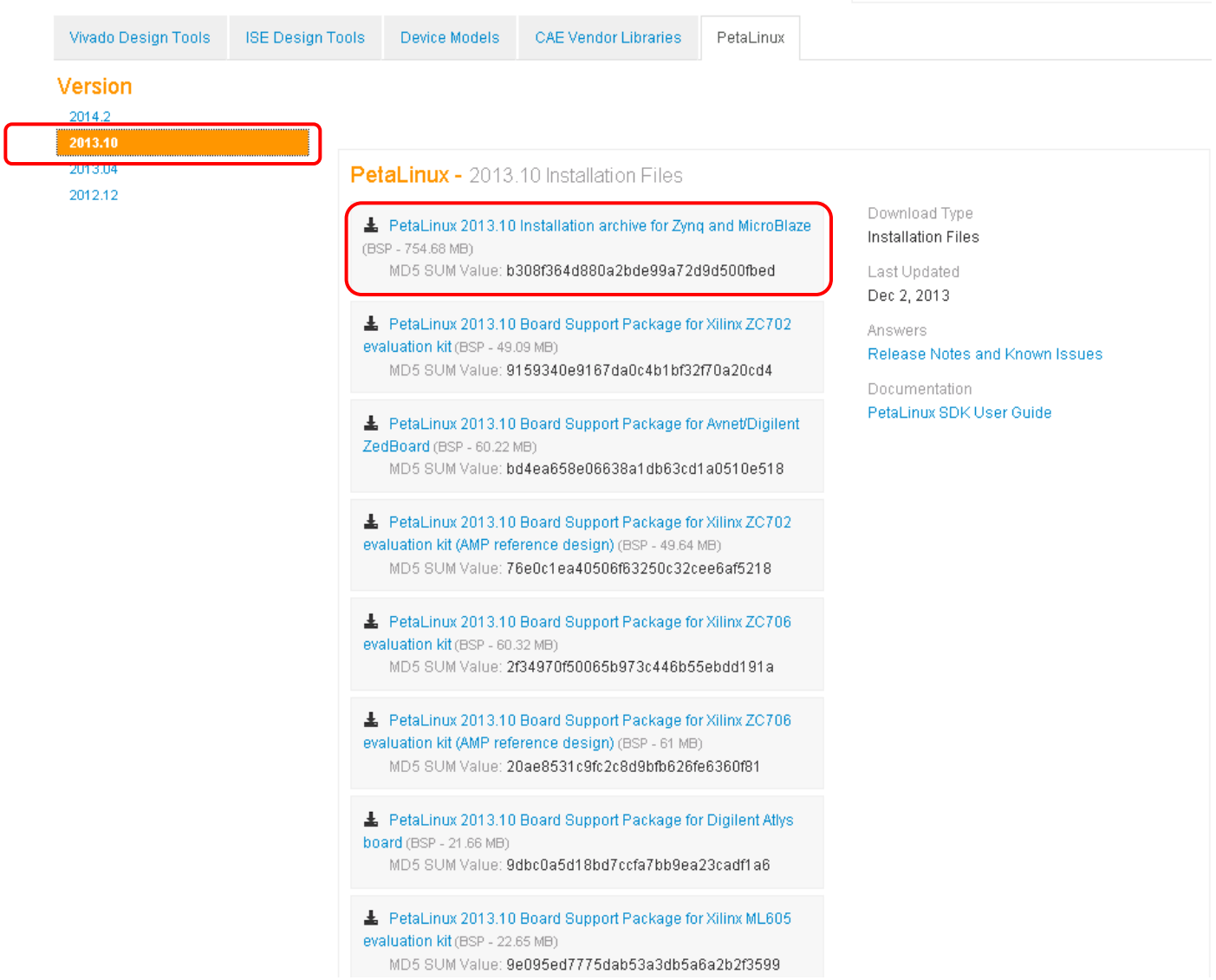


Vivado Design Suite – Start Vivado and SDK

- Host> vivado &
- Host> xsdk &
- ‘GLIBCXX_3.4.9’ not found:
  - Solution: http://ubuntuforums.org/showthread.php?t=808045



PetaLinux Tools - Download

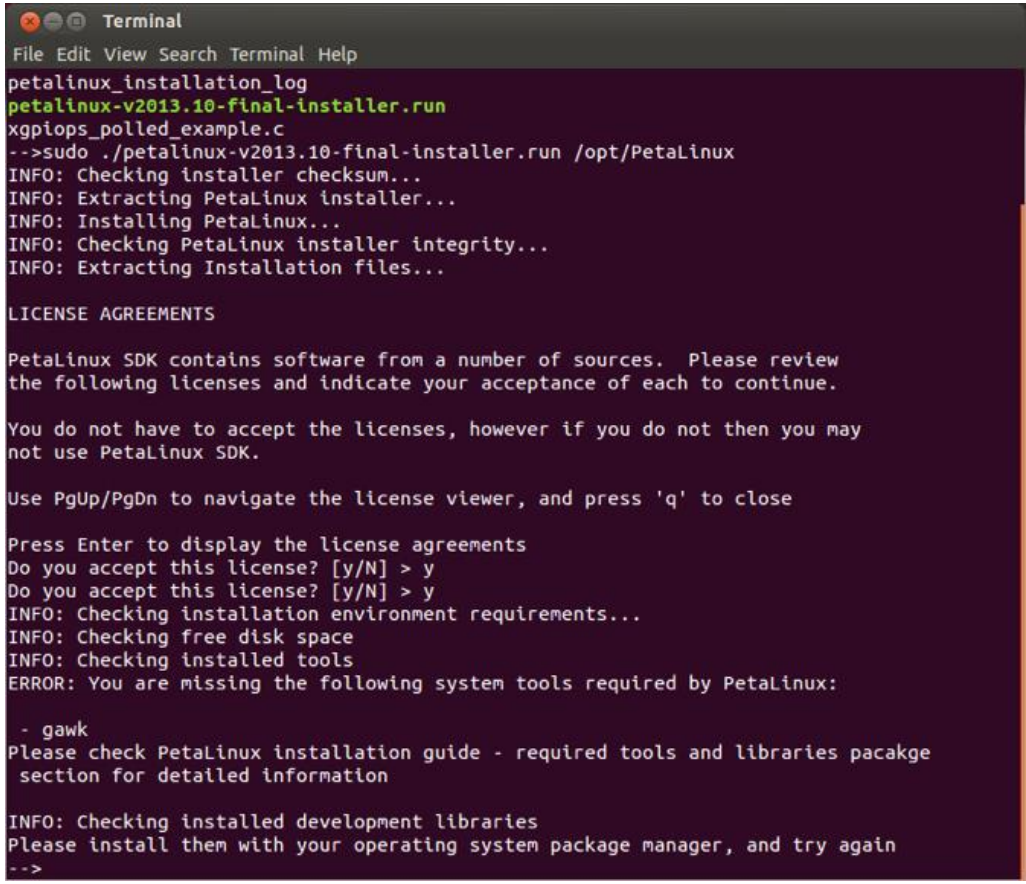


http://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/embedded-design-tools/2013-10.html

PetaLinux Tools - Install

- Host> cd ~/Downloads
- Host> sudo ./petalinux-v2013.10-final-installer.run /opt/PetaLinux
- Host> vi ~/.bashrc
  - Add

  “source /opt/PetaLinux/petalinux-v2013.10-final/settings.sh”



PetaLinux Tools – Required Package

- Host> sudo apt-get install gawk
- Host> sudo apt-get install bison
- Host> sudo apt-get install flex
- Host> sudo apt-get install zlib1g-dev
- Host> sudo apt-get install tofrodos
- Host> sudo apt-get install libstdc++6:i386
- Host> sudo apt-get install libncurses5:i386
- Host> sudo apt-get install libncurses5w-dev:i386
- Host> sudo apt-get install gcc
- Host> sudo ln -s /usr/bin/make /usr/bin/gmake

Tool/Library	YUM/RPM Package for RHEL/CentOS/Fedora	APT Package for Debian/Ubuntu	RPM Package for SuSE
dos2unix	dos2unix	tofrodos	dos2unix
ip	iproute	iproute	iproute2
gawk	gawk	gawk	gawk
gcc	gcc	gcc	gcc
git	git	git-core	git-core
make	gnutls-devel	make	make
netstat	net-tools	net-tools	net-tools
ncurses	ncurses-devel	ncurses-dev	ncurses-devel
tftp server	tftp-server	tftpd	tftp-server
zlib	zlib-devel	zlib1g-dev	zlib-devel
flex	flex	flex	flex
bison	bison	bison	bison
32bit libs	libstdc++-4.4.6-4.el6.i686 glibc.i686 libgcc.i686	ia32-libs lib32ncursesw5	32-bit runtime environment

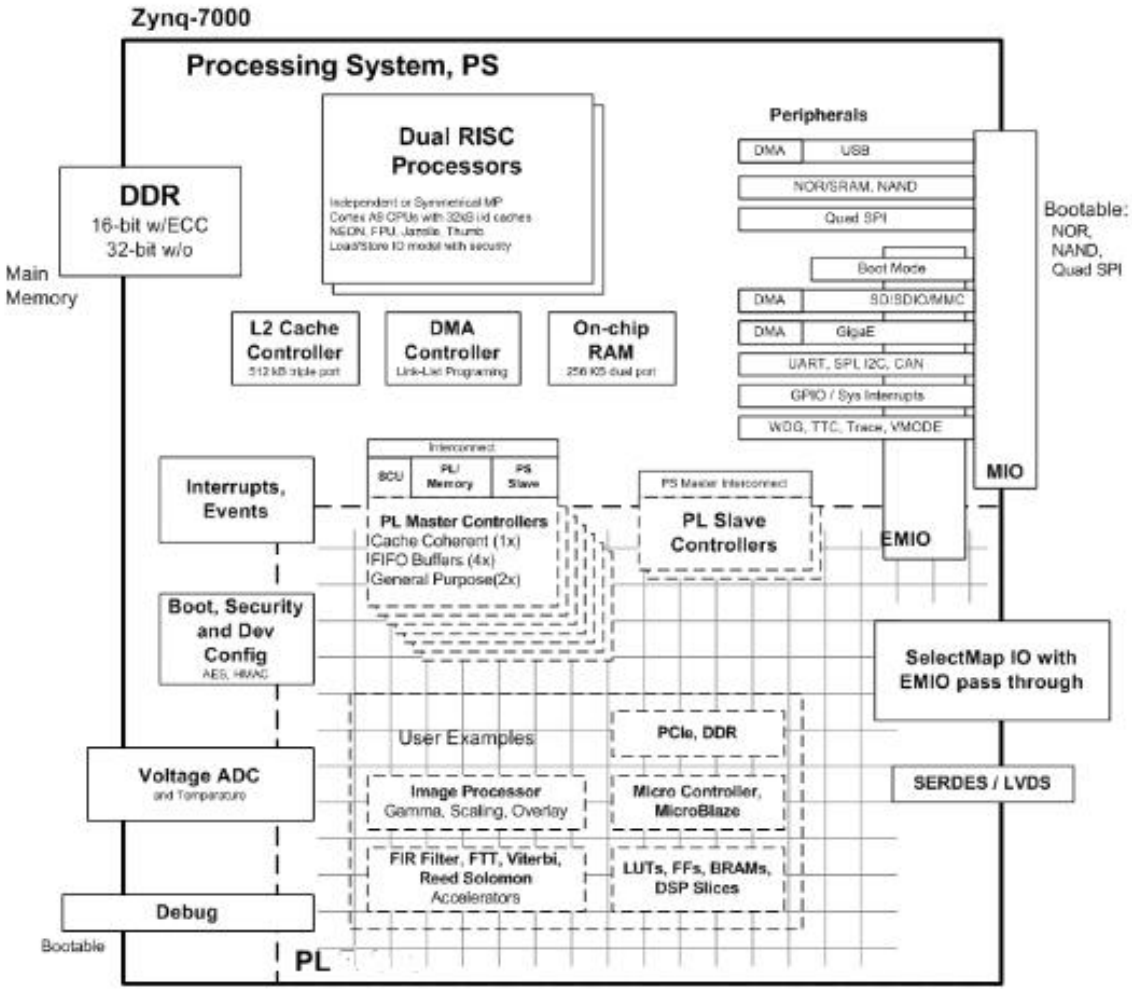
PetaLinux Tools – Create Project

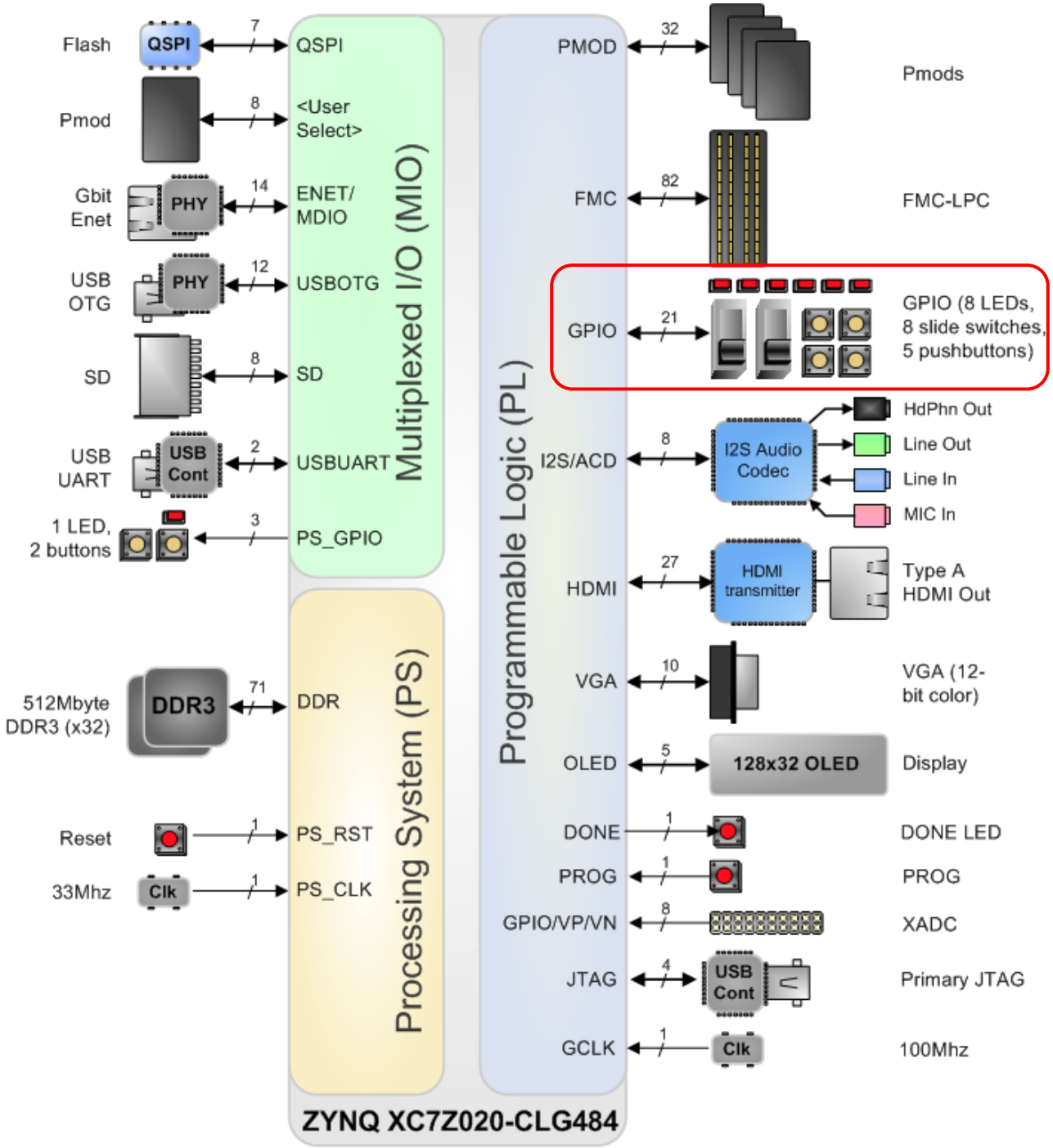
- Host> petalinux-create -t project --name software

```
petalinux@ubuntu:~/emblnx/labs/lab6$ petalinux-create -t project --name software
INFO: Create project: software
INFO: New project successfully created in /home/petalinux/emblnx/labs/lab6/softw
are
```

Zynq 7000 AP SoC PS and PL Block Diagram

- PS: **P**rocessing **s**ystem
  - Dual ARM Cortex-A9 processor based
- PL: **P**rogrammable **l**ogic
  - Artix™ FPGA




Flash
QSPI
7
QSPI
Pmod
8
<User Select>
Gbit Enet
PHY
14
ENET/ MDIO
USB OTG
PHY
12
USBOTG
SD
8
SD
USB UART
USB Cont
2
USBUART
1 LED, 2 buttons
3
PS_GPIO
Multiplexed I/O (MIO)
512Mbyte DDR3 (x32)
DDR3
71
DDR
Reset
1
PS_RST
33Mhz
Clk
1
PS_CLK
Processing System (PS)
PMOD
32
Pmods
FMC
82
FMC-LPC
GPIO
21
GPIO (8 LEDs, 8 slide switches, 5 pushbuttons)
I2S/ACD
8
I2S Audio Codec
HdPhn Out
Line Out
Line In
MIC In
HDMI
27
HDMI transmitter
Type A HDMI Out
VGA
10
VGA (12-bit color)
OLED
5
128x32 OLED
Display
DONE
1
DONE LED
PROG
1
PROG
GPIO/VP/VN
8
XADC
JTAG
4
USB Cont
Primary JTAG
GCLK
1
Clk
100Mhz
Programmable Logic (PL)
ZYNQ XC7Z020-CLG484

LAB 1.2

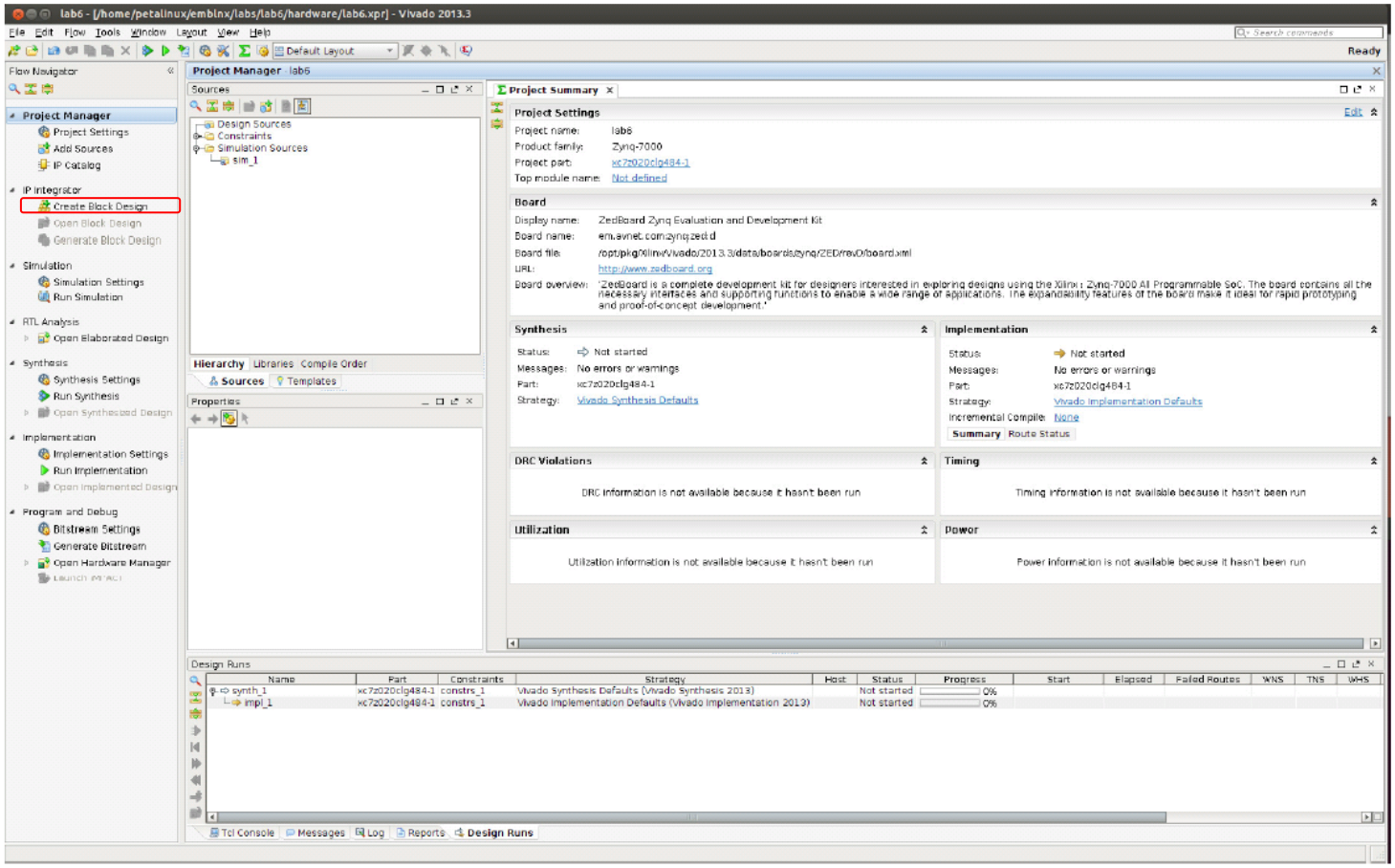
Bare-metal design

Create Vivado Project

- Host> vivado &
- Click ***Create New Project*** and click ***Next***
- Enter ***Project Name*** and click ***Next***
- Select ***RTL Project*** and click ***Next***
- Under the Specify area select ***Boards***
- Select ***Zynq*** from the Library drop-down list
- Select ***Zedboard Zynq Evaluation and Development Kit*** and click ***Next***
- Click ***Finish***

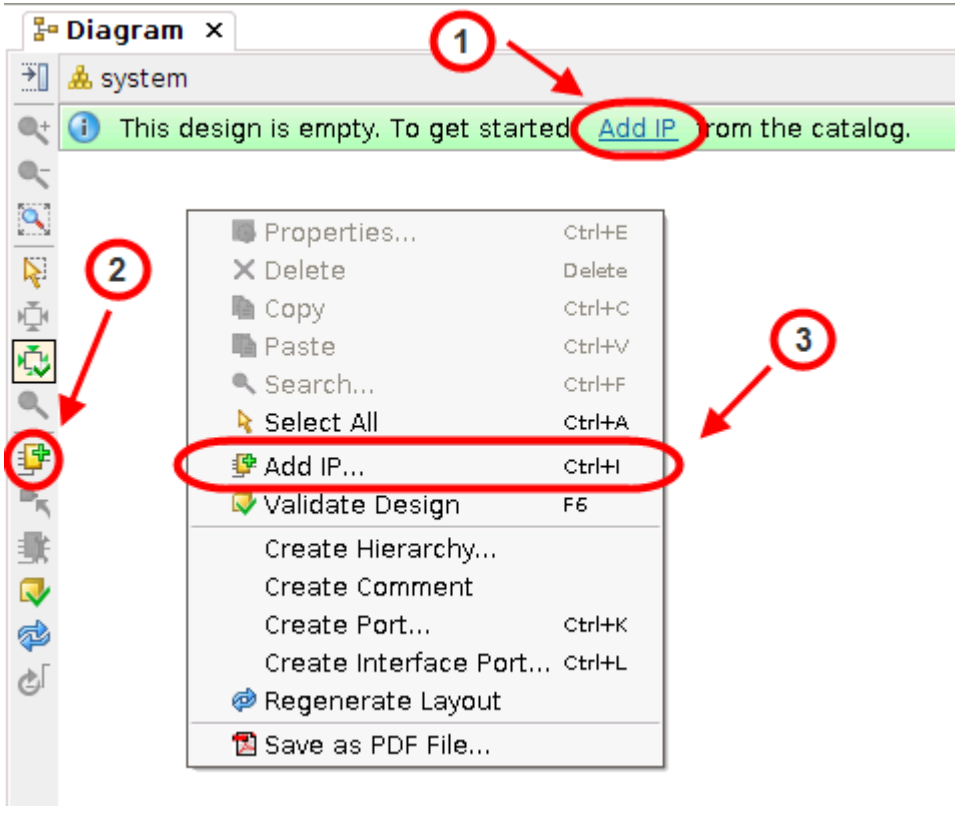
Create Block Design

- Click ***Create Block Design***
- Enter ***Design name***, such as system, and click ***OK***.



Create Processing System

- Click ***Add IP***
- Search ***zynq*** and double click ***ZYNQ7 Processing System***
- Click ***Run Block Automation*** and select ***/processing_system7_0*** and Click ***OK***



Configure Processing System

- Double-click the ***ZYNQ7 Processing System*** block
- (Optional) Deselect all the I/O peripherals except ***UART 1***
  - De-select ***USB0, ENET 0*** and ***SD 0*** under ***MIO Configuration → I/O Peripherals***
  - De-select ***GPIO MIO*** under ***MIO Configuration → I/O Peripherals → GPIO***
  - Deselect ***Quad SPI Flash*** under ***MIO Configuration → Memory Interfaces***
  - Deselect ***Timer 0*** under ***MIO Configuration → Application Processor Unit***
- Check and Select ***M_AXI_GP0 interface*** under ***PS-PL Configuration → GP Master AXI***
- Check and Select ***FCLK_RESET0_N*** under ***General → Enable Clock Resets***
- Check and Select ***FCLK_CLK0*** under ***Clock Configuration → PL Fabric Clocks***

Create Switch GPIO

- Click the Add IP icon 
- Double-click the ***AXI GPIO***
- Click on the ***AXI GPIO*** block to select it, and in the properties tab, change the name to ***sw_8bit***
- Double click on the ***AXI GPIO*** block
- Click on ***Generate Board Based IO Constraints***, and under ***Board Interface***, for ***GPIO***, click on ***Custom*** to view the dropdown menu options, and select ***sws_8bits***
- Click ***OK***

Configure AXI Interface

- Click ***Run Connection Automation*** and select ***/SWs_8Bits/S_AXI*** and Click ***OK***
- Double-click the ***AXI Interconnect*** block
- Select ***3*** as the number of master interfaces
- Click ***OK***
- Connect
  - *AXI Interconnect* ***M01_ARESETN*** → *Proc Sys Reset* ***peripheral_aresetn***
  - *AXI Interconnect* ***M02_ARESETN*** → *Proc Sys Reset* ***peripheral_aresetn***
  - *AXI Interconnect* ***M01_ACLK*** → *ZYNQ7 Processing System* ***FCLK_CLK0***
  - *AXI Interconnect* ***M02_ACLK*** → *ZYNQ7 Processing System* ***FCLK_CLK0***

Create Button and LED GPIOs

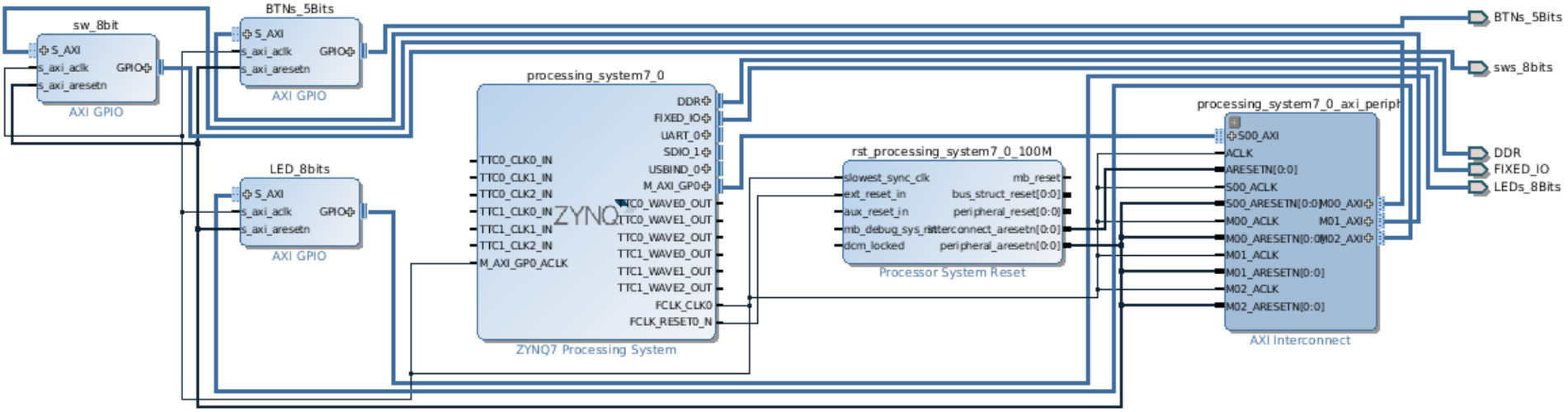
- Create two more GPIO for Buttons and LEDs.
  - BTNs_5Bits (select the ***board interface*** as ***BTNs_5Bits***)
  - LEDs_8Bits (select the ***board interface*** as ***LEDs_8Bits***)
- Follow the previous steps for each additional peripherals (***BTNs_5Bits*** and ***LEDs_8Bits***)

Peripheral Name	Board Interface	S_AXI Connection	s_axi_aclk	s_axi_aresetn
BTNs_5Bits	BTNs_5Bits	M01_AXI	FCLK_CLK0	peripheral_aresetn
LEDs_8Bits	LEDs_8Bits	M02_AXI	FCLK_CLK0	peripheral_aresetn

Connect GPIOs to External Pins

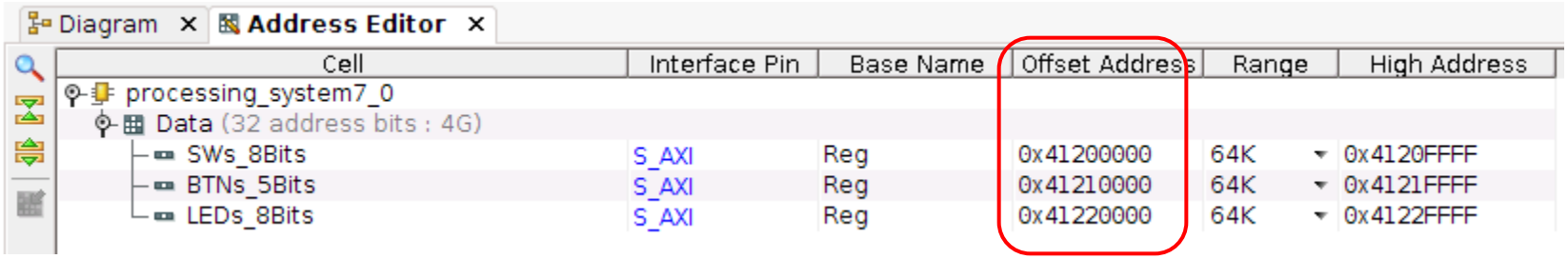
- Right-click the ***GPIO*** port of the ***SWs_8Bits*** instance and select ***Make External***
- Select the ***GPIO*** port and change the name to ***SWs_8Bits*** in the ***External Interfaces Properties Name*** field
- Click ***Run Connection Automation*** and select ***/BTNs_5Bits/GPIO***
- Select ***BTNs_5Bits*** in the ***Select Board Interface***
- Click ***OK***
- Similarly, use ***Run Connection Automation*** to create an external port for the ***LEDs_8Bits*** instance

Block Design



Assign Address

- Select the ***Address Editor*** tab and click 
- Click 
- Select ***File*** → ***Save Block Design***
- Select ***Tools*** → ***Validate Design***



Generate Bitstream

- Right Click on the block diagram file, ***system.bd***, and select ***Create HDL Wrapper***, then click ***Copy and Overwrite***
- Click ***Run Synthesis*** then click ***Save***
- Select ***Open Synthesized Design*** and click ***OK***
- Click on ***Generate Bitstream*** then click ***Yes***
- Select ***Open Implemented Design*** option and click ***OK***

Export to Xilinx SDK

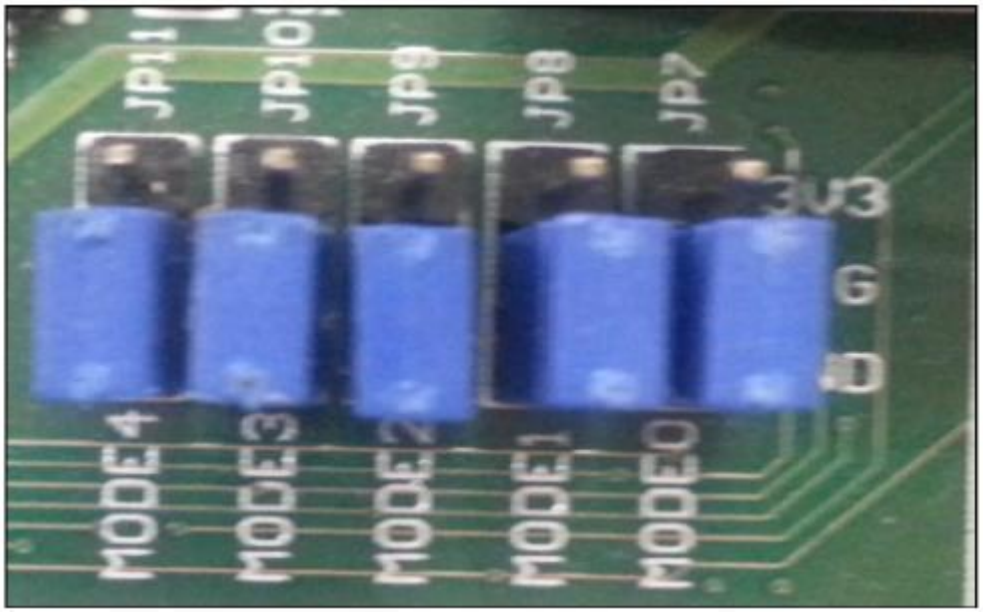
- Click ***File*** → ***Export*** → ***Export Hardware for SDK***
- Check the ***Launch SDK*** and ***Include bitstream*** box and click ***OK***
- Click ***Yes***

Write a Bare-metal Application

- In SDK, select ***File*** → ***New*** → ***Application Project***
- Name the project ***Lab_1.2.1*** and, select ***Create New*** in the ***Board Support Package*** section and type the name ***standalone_bsp*** and click ***Next***
- Select ***Empty Application*** and click ***Finish***
- The library will automatically generated at ~*lab1.2.1**lab1.2.1.sdk**SDK**SDK_Export**standalone_bsp**ps7_cortexa9_0**include* directory
- Create a new *C* program at ***Lab_1.2.1*** in the project view
- Finish the program

Program FPGA

- Make sure that the jumper settings are set to JTAG Mode
- Select ***Xilinx Tools*** → ***Program FPGA***
- Check the ***Bitstream*** path is pointing to *~\lab1.2.1\lab1.2.1.sdk\SDK\SDK_Export\hw_platform_0\system.bit* and click ***Program***
- Right click on project ***Lab_1.2.1*** and select ***Run As*** → ***Launch on Hardware (GDB)***



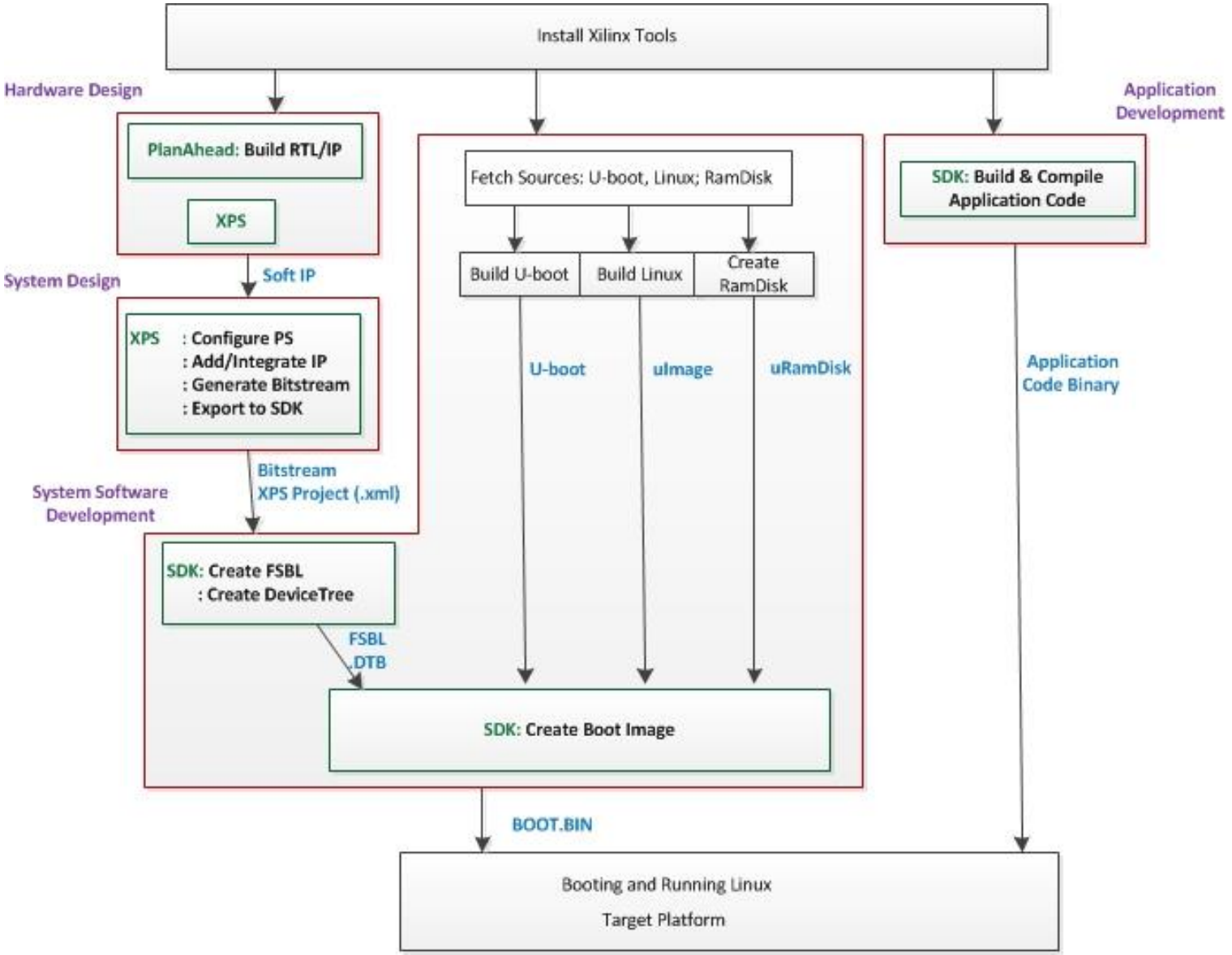
Observe UART Output

- Start the ***GtkTerm*** at host
- Select ***/dev/ttyACM0*** at ***Port*** and ***115200*** at ***Baud Rate***

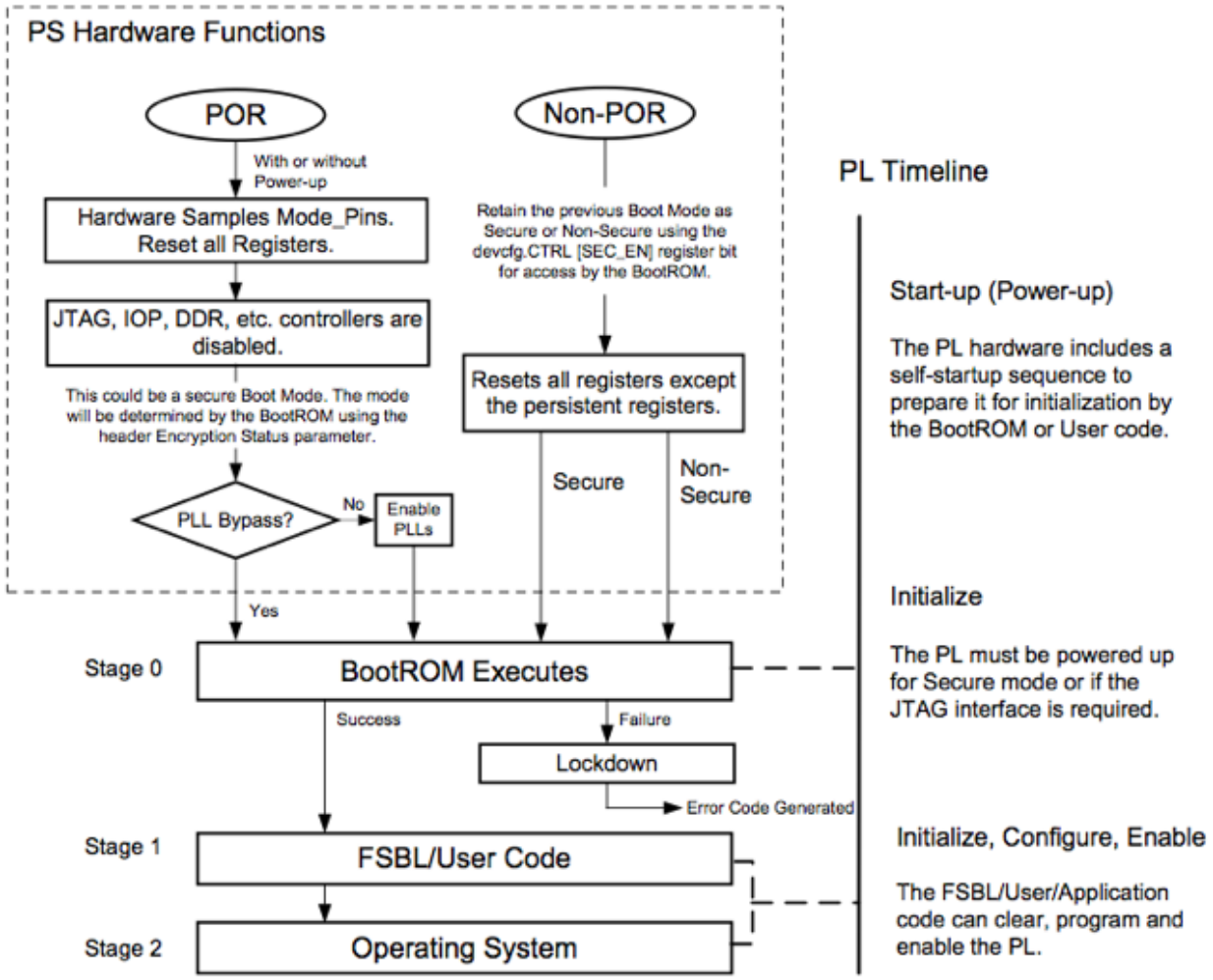
LAB 1.2.2

Hello world program on basic petalinux

Design Flow


Install Xilinx Tools
Hardware Design
Application Development
PlanAhead: Build RTL/IP
XPS
Soft IP
System Design
XPS : Configure PS
: Add/Integrate IP
: Generate Bitstream
: Export to SDK
Bitstream
XPS Project (.xml)
System Software Development
SDK: Create FSBL
: Create DeviceTree
FSBL
DTB
Fetch Sources: U-boot, Linux; RamDisk
Build U-boot
Build Linux
Create RamDisk
U-boot
uImage
uRamDisk
SDK: Create Boot Image
SDK: Build & Compile Application Code
Application Code Binary
BOOT.BIN
Booting and Running Linux
Target Platform

Zynq Boot Flow



Hardware Design

- In Vivado, create a new design with only ***ZYNQ7 Processing System*** block or using the hardware design from Lab 1.2.1
  - The following components are the minimal requirement to boot up Linux:
  - ***ENET 0*** (optimal but suggested) and ***SD 0*** under ***MIO Configuration*** → ***I/O Peripherals***
  - ***Quad SPI Flash*** under ***MIO Configuration*** → ***Memory Interfaces***
  - ***Timer 0*** under ***MIO Configuration*** → ***Application Processor Unit***
  - (Please check the setting of above item is the same as it in default ***ZYNQ7 Processing System*** block)
- ***Run Synthesis*** and/or ***Generate Bitstream*** again if you revise the design or create a new design
- Click ***File*** → ***Export*** → ***Export Hardware for SDK***
- Check the ***Launch SDK*** box and click ***OK***
- Click ***Yes***

Add PetaLinux SDK Repository

- In SDK, select ***Xilinx Tools*** → ***Repositories***
- Select ***Repositories*** then click ***New***
- Browse and Select *~/opt/petalinuxv2013.3.10-final/components/edk_user_repository* then click ***OK***
- Click ***OK***

Create PetaLinux BSP (Board Support Package)

- Select ***File*** → ***New*** → ***Board Support Package***
- Select ***hw_platform_0*** at ***Hardware Platform*** field and ***petalinux*** at ***Board Support Package OS*** field, and then click ***Finish***
- Select ***petalinux*** in the ***Board Support Package Settings*** window and check the following options
  - ***stdin***: ***ps7_uart_1***
  - ***stdout***: ***ps7_uart_1***
  - ***main_memory***: ***ps7_ddr_0***
  - ***flash_memory***: ***ps7_qspi_0***
  - ***sdio***: ***ps7_sd_0***
  - ***ethernet***: ***ps7_ethernet_0***
- Click ***OK***

Create First Stage Boot Loader (FSBL)

- Select ***File*** → ***New*** → ***Application Project***
- Enter ***zynq_fsbl*** as the ***Project name***
- Set the follow options:
  - ***Hardware Platform***: ***hw_platform_0***
  - ***Target Software OS Platform***: ***standalone***
  - ***Board Support Package***: ***Create New: zynq_fsbl_bsp***
- Click ***Next***
- Select ***zynq FSBL*** in the ***Available Templates*** box
- Click ***Finish***

Create and Configure Petalinux Project

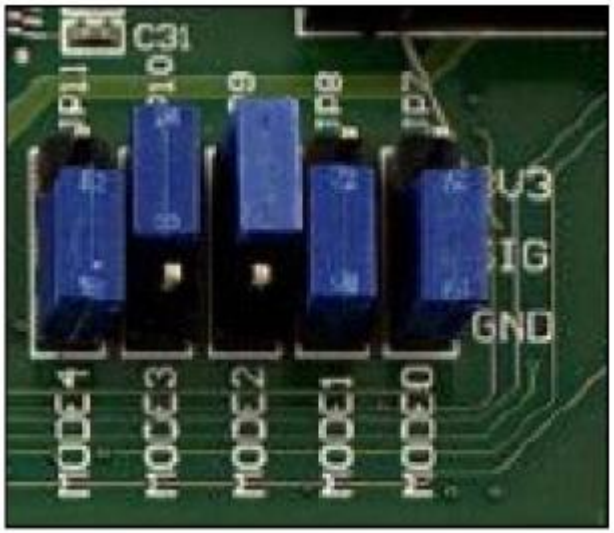
- Start a ***Terminal*** at host
- Host> cd lab1.2.2
- Host> petalinux-create -t project --name software
- Host> cd ~/lab1.2.2/lab1.2.2.sdk/SDK/SDK_Export/petalinux_bsp_0
- Host> petalinux-config --get-hw-description -p ~/lab1.2.2/software
- Host> cd ~/lab1.2.2/software
- Host> petalinux-config
- Select ***Host name*** and change to your group number
- Select ***System boot device*** and change to ***SD card***
- Select ***OK*** then select ***Exit*** and select ***Yes***
- Host> petalinux-build
- Uboot would be generated at *~/lab1.2.2/software/images/linux*

Create Boot Image

- In SDK, right-click the ***zynq_fsbl*** project and select ***Create Boot Image***
- Click ***Add*** then click ***Browse*** and select *~/lab1.2.2/software/images/linux/u-boot.elf*
- Check the order of boot: bootloader -> bitstream -> u-boot
- Click ***OK***
- Click ***Create Image***
- ***mv*** *~/lab1.2.2/lab1.2.2.sdk/SDK/SDK_Export/zynq_fsbl/bootimage/output.bin ~/lab1.2.2/lab1.2.2.sdk/SDK/SDK_Export/zynq_fsbl/bootimage/BOOT.BIN*
- Copy *~/lab1.2.2/lab1.2.2.sdk/SDK/SDK_Export/zynq_fsbl/bootimage/BOOT.BIN* and *~/lab1.2.2/software/images/linux/image.ub* to SD card

Boot the OS on Zedboard

- Set the jumper to SD card mode (*JP9* and *JP10* are set to High)
- Plug in SD card
- Start the ***GtkTerm*** at host
- Select ***/dev/ttyACM0*** at ***Port*** and ***115200*** at ***Baud Rate***
- U-Boot-PetaLinux> run sdboot
- The user name and password are both *root* for petalinux



Design a Hello World Program

- At host
  - Host> cd /tftpboot
  - Host> vi lab1.2.2.c
  - Write a Hello World program
  - Host> arm-xilinx–linux-gnueabi-gcc lab1.2.2.c –o lab1.2.2
  - Setup the Host IP and use LAN cable to connect Host and Target
    - Host> ifconfig eth0 192.168.0.1
- At Zedboard
  - Group X> ifconfig eth0 192.168.0.10
  - Group X> tftp –g 192.168.0.1 –r lab1.2.2
  - Group X> chmod 755 lab1.2.2
  - Group X> ./lab1.2.2

Download the Hello World Program to Zedboard

- At Host
  - Set up the Host IP and use LAN cable to connect Host and Target
    - Host> ifconfig eth0 192.168.0.1
- At Zedboard
  - Group X> ifconfig eth0 192.168.0.10
  - Group X> tftp –g 192.168.0.1 –r lab1.2.2
  - Group X> chmod 755 lab1.2.2
  - Group X> ./lab1.2.2

Mount From SD card

- root@GroupID:~# mount /dev/mmcblk0p1 /mnt
- root@GroupID:~# cd /mnt/
- root@GroupID:~# ./lab1.2.2

LAB 1.3

Basic I/O related program on petalinux

Hint

- Two ways to access
  - Map the physical address of I/O to virtual address
  - Enable User space I/O and map the I/O under ***/dev*** to application's address space

Lab Assignment

- Today's work (10%)
  - Install vivado & xsdk
- Lab 1.2.1 (30%)
  - Design hardware for lighting on the LEDs with switching on the corresponding switches
  - Show the button status from UART
- Lab 1.2.2 (30%)
  - Change host name of petalinux to you group number as "Group #"
  - Show "Hello Group #" under OS
  - Minimize the file size of BOOT.BIN
- Lab 1.3 (20%)
  - Use LEDs to show the result of add, subtract, multiply and divide
  - Switches as input number
  - Four Buttons denote add, subtract, multiply and divide
- Bonus (10%)
  - Implement Lab 1.3 in bare-metal design
- Report (10%)

Deadline

- Demo:
  - Lab1.1 & 1.2.1: 3/3
  - Lab1.2.2 & 1.3: 3/10
- Report: 3/10 23:59
- Discussion
  - https://2016_embedded_system_lab.hackpad.com/