ITRI
Industrial Technology
Research Institute

Petalinux Tutorial

What are Petalinux Tools

- PetaLinux is an Embedded Linux System Development Kit targeting Xilinx® FPGA-based System-on-Chip designs.
- Xilinx PetaLinux Tools are available at no-charge, make it easy for developers to configure.
- The PetaLinux tool simplifies the development of Linux-based products using the following tools
  - Command-line interfaces
  - Application, Device Driver & Library generators and development templates
  - Bootable system Image builder
  - Debug agents
  - GCC tools
  - Integrated QEMU Full System Simulator
  - Automated tools
  - Support for Xilinx System Debugger

Design Flow Overview

- The table below provides an example design workflow, demonstrating the order in which the tasks should be completed and the corresponding tool or workflow for that task.

Design Flow Step	Tool / Workflow
Hardware Platform Creation	Vivado
Create PetaLinux Project	petalinux-create -t project
Initialize PetaLinux Project	petalinux-config --get-hw-description
Configure System-Level Options	petalinux-config
Configure the Linux Kernel	Petalinux-config -c kernel
Configure the Root File System	petalinux-config -c rootfs
Build the System	petalinux-build
Package for Deploying the System	petalinux-package

Installation Requirements

- 8 GB RAM (recommended minimum for Xilinx tools)
- 2 GHz CPU clock or equivalent (minimum of 8 cores)
- 100 GB free HDD space
- Supported OS:
  - Red Hat Enterprise Workstation/Server 7.2, 7.3, 7.4, 7.5 (64-bit)
  - CentOS 7.2, 7.3, 7.4, 7.5 (64-bit)
  - Ubuntu Linux 16.04.3, 16.04.4 (64-bit)

Packages and Linux Workstation Environments

- Ubuntu
  - sudo apt-get install -y gcc git make net-tools libncurses5-dev tftpd zlib1g-dev libssl-dev flex bison libselinux1 gnupg wget diffstat chrpath socat xterm autoconf libtool tar unzip texinfo zlib1g-dev gcc-multilib build-essential libsdl1.2-dev libglib2.0-dev zlib1g:i386 screen pax gzip
- Redhat/CENTOS:
  - sudo apt-get install -y gcc git make net-tools libncurses5-dev tftpd zlib1g-dev libssl-dev flex bison libselinux1 gnupg wget diffstat chrpath socat xterm autoconf libtool tar unzip texinfo zlib1g-dev gcc-multilib build-essential libsdl1.2-dev libglib2.0-dev zlib1g:i386 screen pax gzip

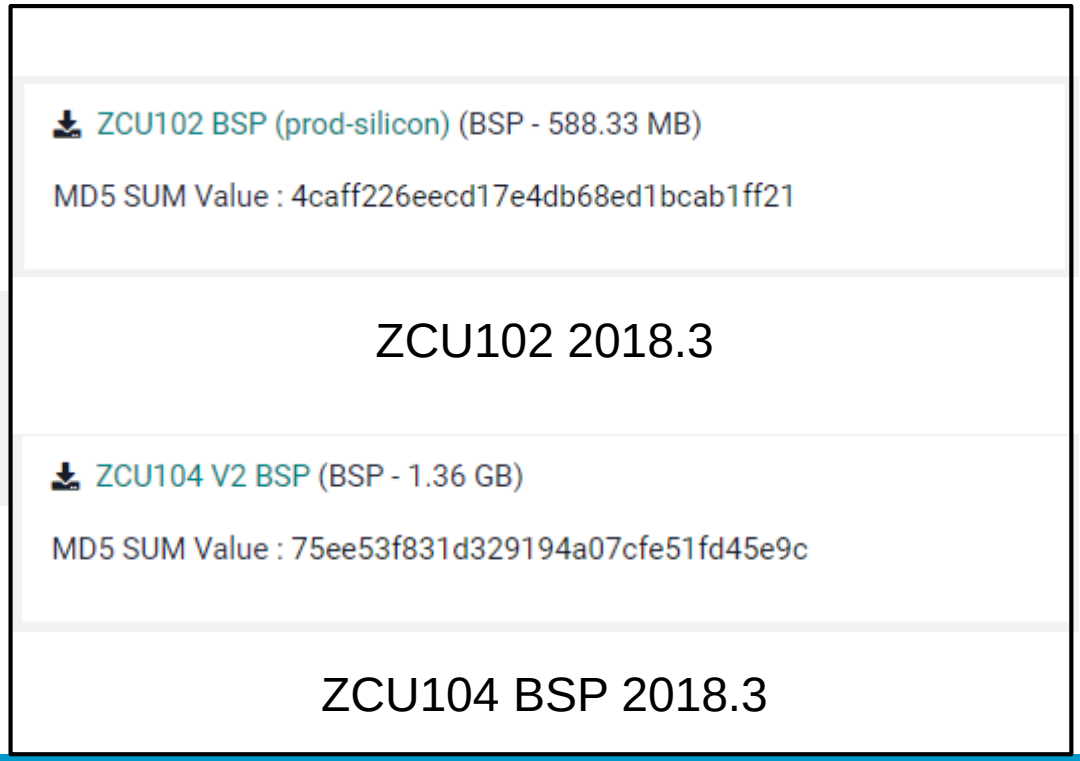
Tool / Library	CentOS 7.2/7.3/7.4	RHEL7.2/7.3/7.4	Ubuntu 16.04.3
gcc-multilib	-	-	gcc-multilib
build-essential	-	-	build-essential
libsdl1.2-dev	-	-	libsdl1.2-dev
libglib2.0-dev	-	-	libglib2.0-dev
SDL-devel	SDL-devel	SDL-devel	-
glibc-devel	glibc-devel	glibc-devel	-
32-bit glibc	glibc-2.17-157.el7_3.4.i686 glibc-2.17-157.el7_3.4.x86_64	glibc-2.17-157.el7_3.4.i686 glibc-2.17-157.el7_3.4.x86_64	-
glib2-devel	glib2-devel	glib2-devel	-
automake	automake	automake	-
screen	screen	screen	screen
pax	pax	pax	pax
gzip	gzip	gzip	gzip
libstdc++	libstdc++-4.8.5-11.el7.x86_64 libstdc++-4.8.5-11.el7.i686	libstdc++-4.8.5-11.el7.x86_64 libstdc++-4.8.5-11.el7.i686	-

Tool / Library	CentOS 7.2/7.3/7.4	RHEL7.2/7.3/7.4	Ubuntu 16.04.3
dos2unix	dos2unix-6.0.3-4.el7.x86_64.rpm	dos2unix-6.0.3-4.el7.x86_64.rpm	tofrodos_1.7.13+ds-2.debian.tar.xz
ip	iproute-3.10.0-74.el7.x86_64.rpm	iproute-3.10.0-74.el7.x86_64.rpm	iproute2 4.3.0-1ubuntu3
gawk	gawk-4.0.2-4.el7.x86_64.rpm	gawk-4.0.2-4.el7.x86_64.rpm	gawk (1:4.1.3+dfsg-0.1)
gcc	gcc-4.8.5-11.el7.x86_64	gcc-4.8.5-11.el7.x86_64	-
g++ (gcc-c++)	gcc-c++-4.8.5-11.el7.x86_64	gcc-c++-4.8.5-11.el7.x86_64	-
git	git 1.8.3	git 1.8.3	git 1.7.1 or above
make	make 3.81	make 3.82	make 3.81
netstat	net-tools 2.0	net-tools 2.0	net-tools
ncurses devel	ncurses -devel 5.9-13	ncurses -devel 5.9-13	libncurses5 -dev
tftp server	tftp-server	tftp-server	tftpd
zlib devel (also, install 32-bit of this version)	zlib-devel-1.2.7-17.el7.x86_64.rpm	zlib-devel-1.2.7-17.el7.x86_64.rpm	zlib1g:i386
openssl devel	openssl -devel 1.0	openssl -devel 1.0	libssl -dev
flex	flex 2.5.37	flex 2.5.37	flex
bison	bison-2.7	bison-2.7.4	bison
libselinux	libselinux 2.2.2	libselinux 2.2.2	libselinux1
gnupg	gnupg	gnupg	gnupg
wget	wget	wget	wget
diffstat	diffstat	diffstat	diffstat
chrpath	chrpath	chrpath	chrpath
socat	socat	socat	socat
xterm	xterm	xterm	xterm
autoconf	autoconf	autoconf	autoconf
libtool	libtool	libtool	libtool
tar	tar:1.24	tar:1.24	tar:1.24
unzip	unzip	unzip	unzip
texinfo	texinfo	texinfo	texinfo
zlib1g-dev	-	-	zlib1g-dev

Download Packages

- PetaLinux release package is downloaded. You can download PetaLinux installer from Xilinx.com Download Center.
- PetaLinux BSPs
  - PetaLinux reference board support packages (BSPs) are provided in the form of installable BSP files, and include all necessary design and configuration files, pre-built and tested hardware and software images, ready for downloading on your board or for booting in the QEMU system emulation environment.
- Version 2018.3

Petalinux 2018.3



Run PetaLinux Tools Installer

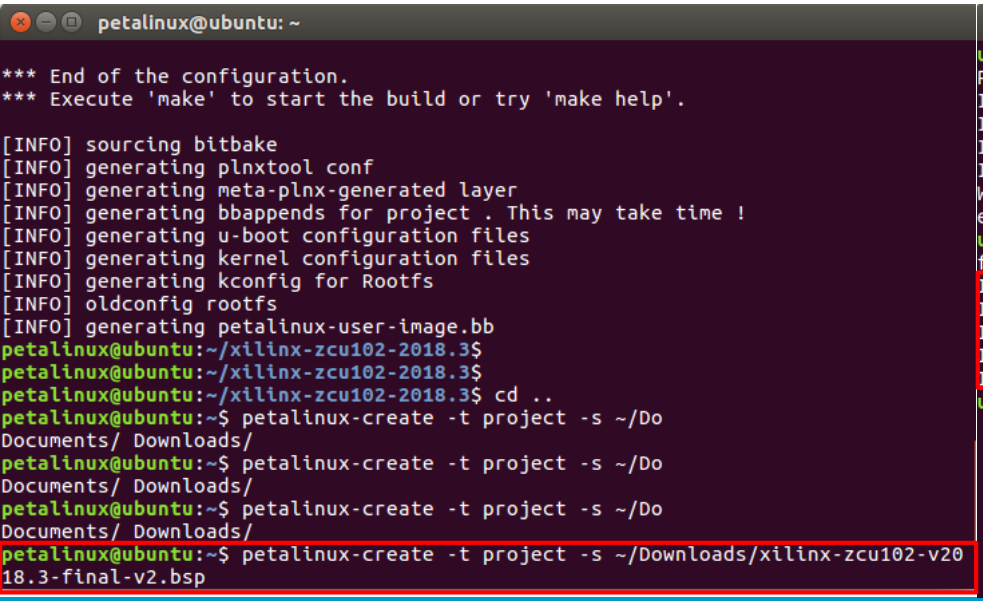
- $ mkdir -p /opt/pkg/petalinux/2018.3
- $ ./petalinux-v2018.3-final-installer.run /opt/pkg/petalinux/2018.3
  - #Note: Reading and agreeing to the PetaLinux End User License Agreement (EULA) is a required and integral part of the PetaLinux Tools installation process. You can read the license agreement prior to running the installation. If you wish to keep the license for the records, the licenses are available in plain ASCII text in the following files:
- Source the appropriate settings script:
  - source <path-to-installed-PetaLinux>/settings.sh

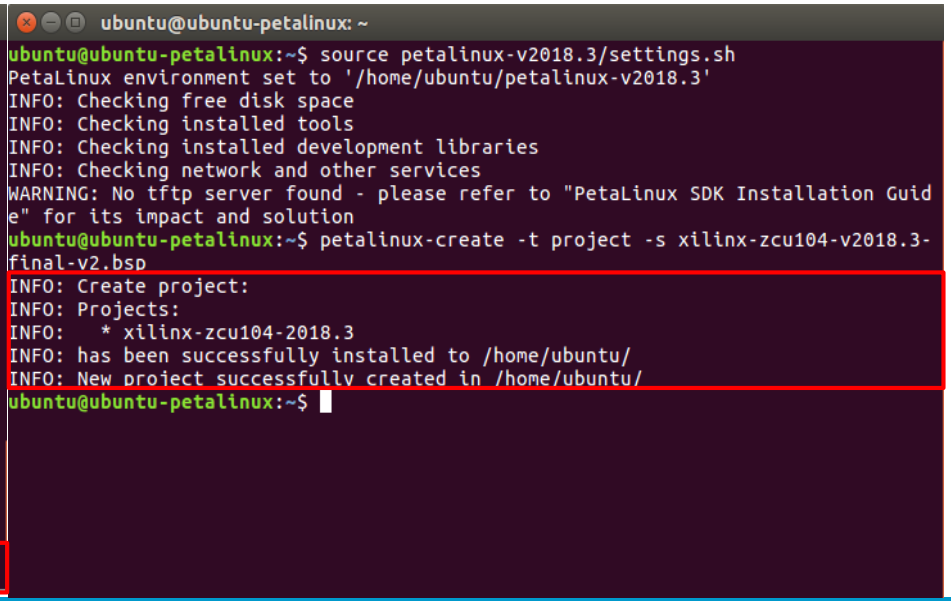
PetaLinux BSP Installation Steps

- Run petalinux-create command on the command console:
  - petalinux-create -t project -s <path-to-bsp>

#Note:

Option	Function description
-t, --type <TYPE>	Specify the TYOE of object to create
-n, --name <NAME>	Create object with the specified name
-p, --project <PROJECT>	PetaLinux project directory path
--force	Overwrite existing files on disk
-h, --help	Display usage information

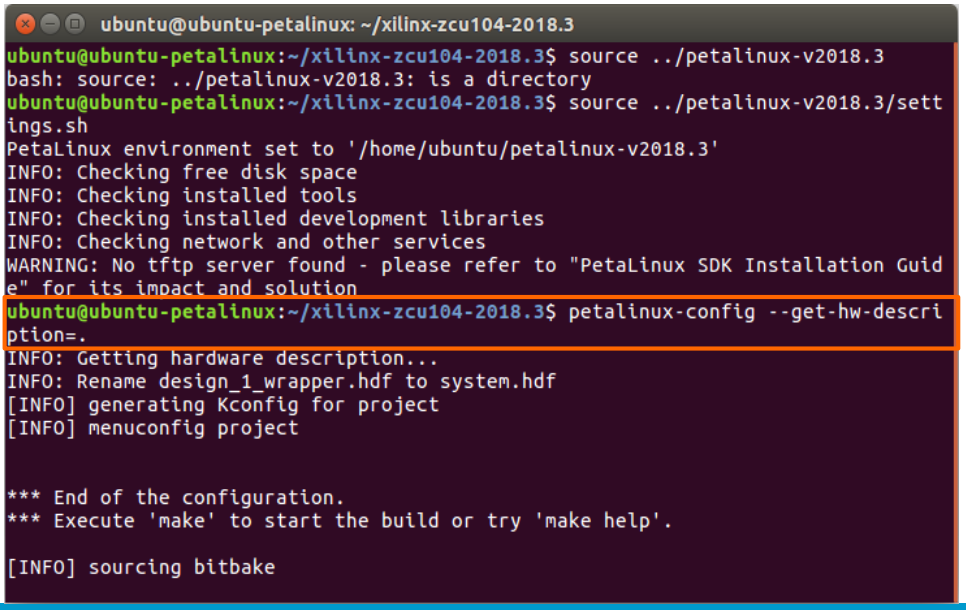


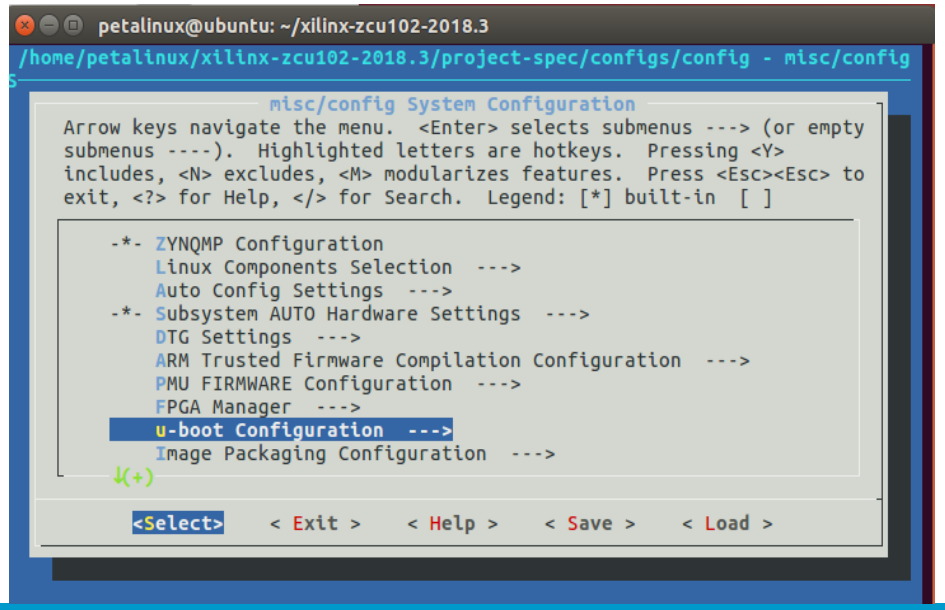


Steps to Import Hardware Configuration

- Change into the directory of your PetaLinux project.
  - $ cd <plnx-project-root>
- Import the hardware description with petalinux-config command, by giving the path of the directory containing the .hdf file as follows:

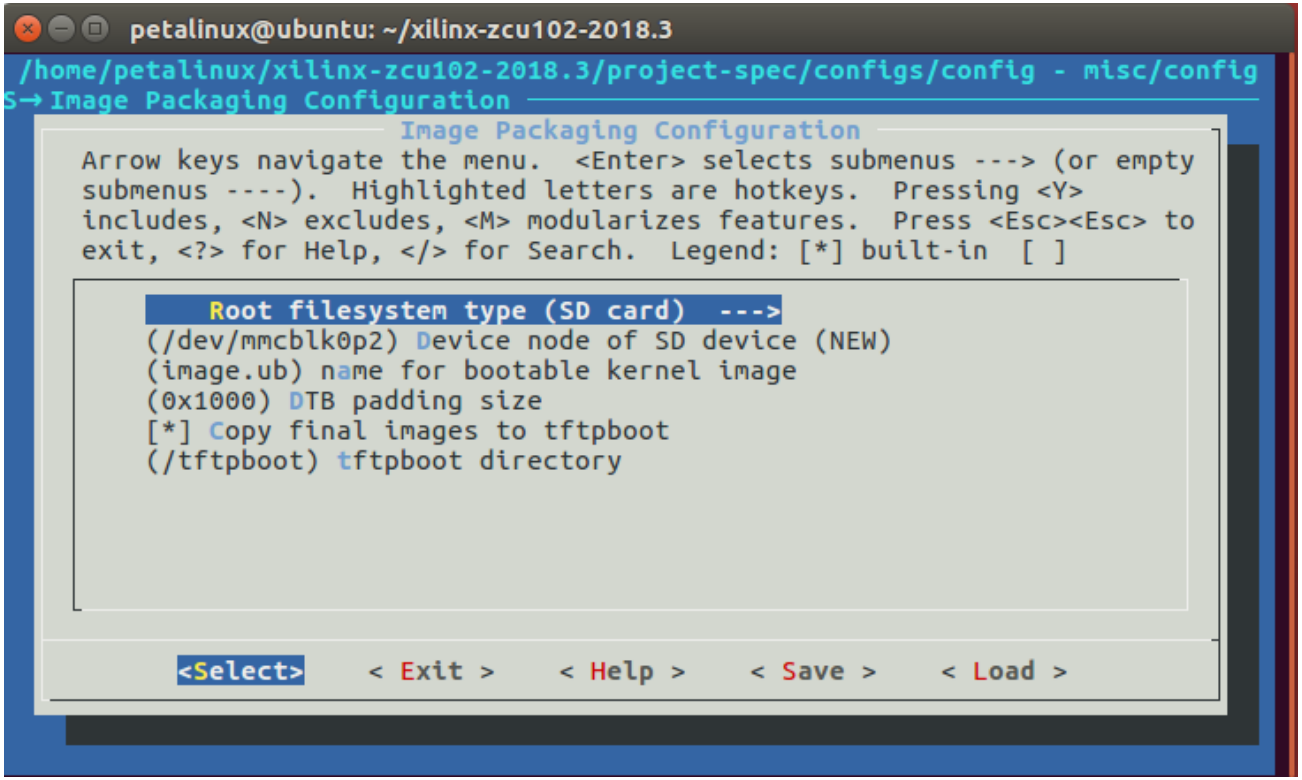
  (My .hdf file has been placed in xilinx-zcu-1022018.3)
  - $ petalinux-config --get-hw-description=<path-to -directory-containing-hardware-description-file>





Root File System Type Configuration

- Configuring SD Card ext filesystem Boot
  - $ Select **Image Packaging Configuration** ---> **Root filesystem type.**
    - Select SD card as the RootFS type.
    - Exit menuconfig and save configuration settings.



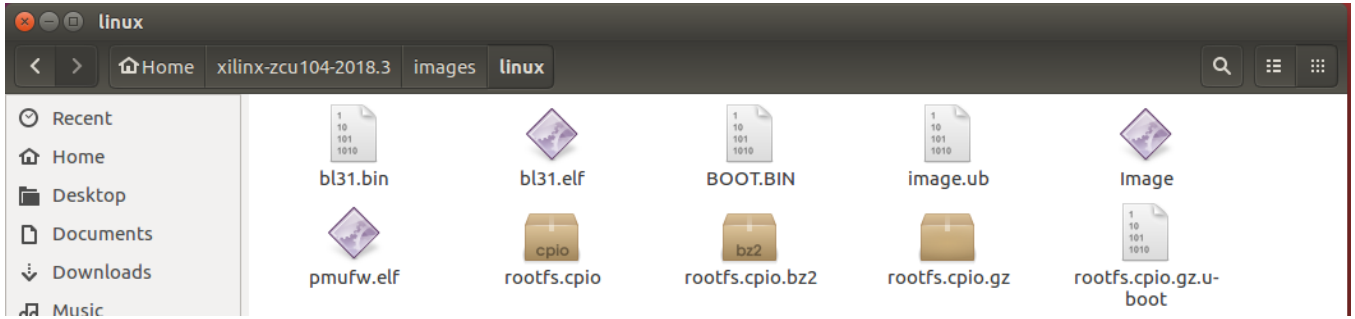
Steps to Build PetaLinux System Image

- Run petalinux-build to build the system image
  - $ petalinux-build

```
NOTE: Tasks Summary: Attempted 7107 tasks of which 5292 didn't need to be rerun
and all succeeded.

Summary: There was 1 WARNING message shown.
INFO: Copying Images from deploy to images
INFO: Creating images/linux directory
NOTE: Failed to copy built images to tftp dir:  /tftpboot
[INFO] successfully built project
petalinux@ubuntu:~/xilinx-zcu104-2018.3$
```

- Generate Boot Image
  - $ petalinux-package --boot --format BIN --fsbl images/linux/zynqmp_fsbl.elf --u-boot images/linux/u-boot.elf --pmufw images/linux/pmufw.elf --fpga zcu102.bit –force
  - The final image is < path-to-bsp >/images/linux/



Steps to Boot a PetaLinux Image on Hardware with SD Card

- Copy the following files from /pre-built/linux/images/ into the root directory of the first partition which is in FAT32 format in the SD card:
  - ✓ BOOT.bin
  - ✓ Image.ub
- Copy rootfs.ext4 and rootfs.cpiofile to rootfs partition of SD card and extract the file system.
  - $ sudo mount rootfs.ext4 /mnt –o loop
  - $ sudo cp -rf /mnt/* /media/rootfs
  - $ cp images/linux/rootfs.cpio /media/rootfs/
- Connect the serial port on the board to your workstation.
- Open a console on the workstation and start the preferred serial communication program (For example: kermit, minicom, gtkterm) with the baud rate set to 115200 on that console.