PetaLinux

Table of contents

Petalinux Project Creation- Short HOWTO

Getting Linux working on Zynq is very simple, following steps are required

1. Vivado/SDK/PetaLinux 20xx.x installed (important do not mix versions!)
2. Create Vivado Project, configure PS, Export HDF (XSA for 19.2 and newer)
3. Create new PetaLinux project
4. Import HDF(XSA for 19.2 and newer) into project
5. petalinux-build
6. copy boot.bin and image.ub to SD Card (only 2 files no more)

This is generic how-to, everything is setup for your by the Vivado->PetaLinux flow. Note, the boot.bin generated by PetaLinux may not always work, in such case it is recommended to make the boot.bin with SDK-GUI or command line tools manually.

There is no need to install anything else, or to fetch anything from any github repos, etc.

PetaLinux 2023.2

> Petalinux 2023.2 is under review documentation for 2023.2 can be changed permanently at the moment

PetaLinux Installation

- (optional) Create new VM with supported Linux OS.
  - Petalinux is evaluated on Windows WSL with Ubuntu, see: AMD Tools and Win10 WSL
    - other VMs or native Linux Distribution is possible, but maybe changes are needed.
- Download PetaLinux from Xilinx Website: http://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/embedded-design-tools.html
  - Choose a PetaLinux Version, that's corresponding to the installed Vivado and SDK Version.
    - Example: Use Vitis (SDK+Vivado) 2023.2 with PetaLinux 2023.2
  - Note Petalinux is also available from Vivado/Vitis Installation setup
- Use UG1144 "PetaLinux Tools Documentation - Reference Guide" that's corresponding with your PetaLinux Version

1. Use bash as terminal:
   a. $ **sudo dpkg-reconfigure dash**
      i. press no
2. Check "PetaLinux Tools Installation Requirements" chapter and install missing tool /libraries
   a. Install packages(see https://support.xilinx.com/s/article/000035572?language=en_US PetaLinux_2023.2_OS_Package_List.xlsx)

i. Note:

1. From excel additional to python3 also python was recommended which is not longer possible to install
2. Additionally bc and was libtinfo5 added
3. Add subversion (need only in case svn is used)
4. Add u-boot-tools (need to generate own boot.scr file instead of petalinux version)

ii. **sudo apt-get install iproute2 gawk python3 build-essential gcc git make net-tools libncurses5-dev tftpd zlib1g-dev libssl-dev flex bison libselinux1 gnupg wget git diffstat chrpath socat xterm autoconf libtool tar unzip texinfo zlib1g-dev gcc-multilib automake zlib1g:i386 screen pax gzip cpio python3-pip python3-pexpect xz-utils debianutils iputils-ping python3-git python3-jinja2 libegl1-mesa libsdl1.2-dev pylint bc libtinfo5 subversion u-boot-tools -y**

3. Install petalinux (Note: do not start from shared folder, copy installer into home directory)

a. $ **sudo mkdir /tools**
b. $ **sudo mkdir /tools/Xilinx**
c. $ **sudo chown <owner>:<owner> /tools/Xilinx**
d. $ **copy FPGAs_AdaptiveSoCs_Unified_2023.2_1013_2256.tar.gz into ~/**
e. $ **tar -xvf FPGAs_AdaptiveSoCs_Unified_2023.2_1013_2256.tar.gz**
f. $ **cd FPGAs_AdaptiveSoCs_Unified_2023.2_1013_2256**
g. $ **./xsetup**

4. source enviroment

a. $ **source /tools/Xilinx/PetaLinux/2023.2/tool/settings.sh**

5. Deactivate Webtalk:

a. $ **petalinux-util --webtalk off**

- Note:
  - There is no need to install anything else, or to fetch anything from any github repos, etc.
  - It is recommended to test the installation by creating a dummy template project and building it.
    - Download one of the BSP Examples from Xilinx Website (Only to test your installation)
    - Install:$ **petalinux-create -t project -s <path-to-bsp>**
    - Build: $ **petalinux-build**

Creating a Project from Vivado Project

Some reference designs contains a preconfigured PetaLinux project as template. This can be used instead of creating a new project described on step 2.

Basic Note to TE Petalinux Templates, see: PetaLinux TE-Template#PetaLinux2023.2

New with 2023.2 and newer:

- ... TBD

New with 2022.2 and newer:

- Separate Device Tree for U-boot possible
- Device tree for uboot must be included into the Boot.bin

New with 2020.2 and newer:

- Trenz FSBL patches are available for petalinux now (beta, vitis template recommended at the moment)
- Xilinx changes:
  - **boot.src** is need for uboot now (will be generated with petalinux) see:
    - Distro Boot with Boot.scr
    - xilinx-wiki.atlassian "Using Distro Boot With Xilinx U-Boot"
  - "**petalinux-devtool**" command is need to update user layer with uboot and kernel changes

1. PetaLinux Working Environment (see UG1144)
   - PetaLinux Tools Installation is completed.
   - (Note: do not create project from shared folder, use home directory)
   - "/bin/sh" is bash
   - Set Working Environment:
     - $ **source <path-to-installed-PetaLinux>/settings.sh**
2. (optional to create project from scretch instead of Trenz Templates) Create a New Project (see UG1144):
   a. $ **petalinux-create --type project --template <CPU_TYPE> --name <PROJECT_NAME>**

      - <CPU_TYPE>: zynqMP, zynq, microblaze
      - <PROJECT_NAME>:The name of the project you are building
3. Import Hardware Configuration (see UG1144):
   - Export Hardware Definition File (*.xsa) from the Vivado Project into the PetaLinux root folder "<plnx-proj-root>":
     - change to PetaLinux root folder:
     - Run:$ **petalinux-config --get-hw-description**
4. (optional) Configure your PetaLinux:
   - While anywhere in the project folder tree:
   - Run:$ **petalinux-config**
   - Run:$ **petalinux-config -c u-boot**
   - Run:$ **petalinux-config -c kernel**
   - Run:$ **petalinux-config -c rootfs**
5. Build System Image (see UG1144):
   - While anywhere in the project folder tree:
   - Run:$ **petalinux-build**
6. Generate Boot.bin with Vitis tools (recommended)
   a. Take u-boot.elf, image.ub, bl31.elf (ZynqMP only) from "<plnx-proj-root>/images /linux" for BOOT.BIN generation, also boot.scr is used (put separate on SD or include into Boot.bin for QSPI boot only). It is recommended to create the FSBL and PMU Firmware (ZynqMP only) with Vitis tools.
7. Generate Boot.bin with Petalinux
   a. Run:$ **petalinux-package --boot <command options>**
   b. https://docs.xilinx.com/r/en-US/ug1144-petalinux-tools-reference-guide/petalinux-package-boot

Petalinux Configuration

Use local sstate cache and downlods

- Download (https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/embedded-design-tools.html):
  - aarch64 sstate-cache
    - for Versal, U+Zynq, RFSoC projects
  - arm sstate-cache
    - for 7 series Zynq projects
  - microblaze sstate-cache
    - for microblaze full projects
  - Downloads
    - forall types
- extract files for example in
  - mkdir ~/design/sstate-cache
  - cd ~/design/sstate-cache
  - copy archives into this folder
  - tar -xfv sstate_aarch64_2023.2_10121051.tar.gz
  - tar -xfv sstate_arm_2023.2_10121051.tar.gz
  - tar -xfv sstate_microblaze_2023.2_10121051.tar.gz
  - mkdir ~/design/sstate-cache/downloads_2023.2
  - cd ~/design/sstate-cache/downloads_2023.2
  - copy archives into this folder
  - tar -xfv downloads_2023.2_10121051.tar.gz
  - delete archive files
- change to the petalinux project
  - **petalinux-config**
    - for sstate-cache
      - -> Yocto Settings->Local sstate feeds settings->local sstate feeds url
        - ~/design/sstate-cache/sstate_aarch64_2023.2/aarch64
        - ~/design/sstate-cache/sstate_aarch64_2023.2/arm
        - ~/design/sstate-cache/sstate_aarch64_2023.2/mb-full
        - original: http://petalinux.xilinx.com/sswreleases/rel-v${PETALINUX_MAJOR_VER}/aarch64/sstate-cache
    - for downloads
      - -> Yocto Settings-> Add pre-mirror url
        - file://~/design/sstate-cache/downloads_2023.2/downloads
        - orginal:http://petalinux.xilinx.com/sswreleases/rel-v${PETALINUX_MAJOR_VER}/downloads
- Clear petalinux project
  - **petalinux-build -x mrproper**

Configuration menues and files

Most settings can be changed with menu-config:

- $ **petalinux-config**
- $ **petalinux-config -c u-boot**
- $ **petalinux-config -c kernel**
- $ **petalinux-config -c rootfs**

Manual changes can be done in the subfolder "<plnx-proj-root>/project-spec/meta-user/"

CONFIG	• project-spec/config/config	• changes with "**petalinux-config**" will be saved here

U-Boot	• recipes-bsp/u-boot/files • recipes-bsp/u-boot/files/platform-top.h • recipes-bsp/uboot-device-tree/files/ystem-user.dtsi	• changes with "**petalinux-config -c u-boot**" will be add to meta-user as recipes • optional overwrite, add UBoot settings on platform-top.h • Important:: Separate Device tree is disabled on default project generation, it must be enabled on **petalinux-config** menue under u-boot configuration otherwhise linux device tree will be used
Device Tree	• recipes-bsp/device-tree/files/system-user.dtsi • recipes-bsp/device-tree/files/pl-custom.dtsi	• overwrite or add device tree attributes
Kernel	• recipes-kernel/linux/linux-xlnx/	• changes with "**petalinux-config -c kernel**" will be add to meta-user as recipes
Apps	• project-spec/meta-user/recipes-apps	• add simple new app with "**petalinux-create -t apps -n myapp --enable**" • enable/disable with "**petalinux-config -c rootfs**"
ROOTFS	• project-spec/config/rootfs_config • project-spec/meta-user/conf/user-rootfsconfig	• "**petalinux-config -c rootfs**" will save changes in the general config spec, only apps will be saved in user-rootfsconfig also • user-rootfsconfig must be changed manually and will read after rootfs_config is used
Xilinx generated configuration	• project-spec/config/	Note: config and rootfs_config are shared at the moment, they include user and xilinx default changes from XSA import

Additional Descriptions

- Debian Installation
- Distro Boot with Boot.scr
- How to install the linux-rt (Real-Time) patch
- Petalinux General Hints and Note
- PetaLinux TE-Template

- Petalinux Troubleshoot
- Reference Designs with PetaLinux

References

1. PetaLinux Tools Documentation - Reference Guide (UG1144)
2. PetaLinux Tools Documentation - PetaLinux Command Line Reference (UG1157)
3. https://www.devicetree.org/
    a. https://www.devicetree.org/specifications/
4. https://github.com/Xilinx/linux-xlnx/tree/master/Documentation/devicetree/bindings

PetaLinux Troubleshoot

Petalinux 2023.2

General notes

Xilinx Release Notes available on: https://support.xilinx.com/s/article/000035572?language=en_US

BL31 gets Stuck during booting

Issue	Petalinux 2023.2 seems to have a Problem booting with UART1: It gets stuck on BL31 while trying to print a char to UART1 during booting. This error happens on Versal and Zynq Ultrascale+ devices. The error is that BL31 built by petalinux does not correctly take into account that UART1 is used, and tries to output on UART0
Workaround	This can be fixed by e.g. overriding yourself with creating **arm-trusted-firmware_%.bbappend** in meta-user/recipes-bsp/arm-trusted-firmware with content For Versal: ATF_CONSOLE = "pl011_1" For Zynq Ultrascale+: ATF_CONSOLE = "cadence1"

Modified FSBL is not containing patches

Issue	Trenz modified FSBL for petalinux is not taken into account from petalinux. This is only need on trenz project which has FSBL patch included inside the petalinux template. --><project folder>\os\petalinux\project-spec\meta-user\recipes-bsp\embeddedsw AMD has changed patch priority and petalinux ignores provided FSBL patch in \os\petalinux\project-spec\meta-user\recipes-bsp\embeddedsw
Workaround	In order for the patches to be used, the existing line BBFILE_PRIORITY_meta-user = "[VALUE]" must be set to 9 instead of the original value 7 in the Petalinux config file project-spec/meta-user /conf/layer.conf