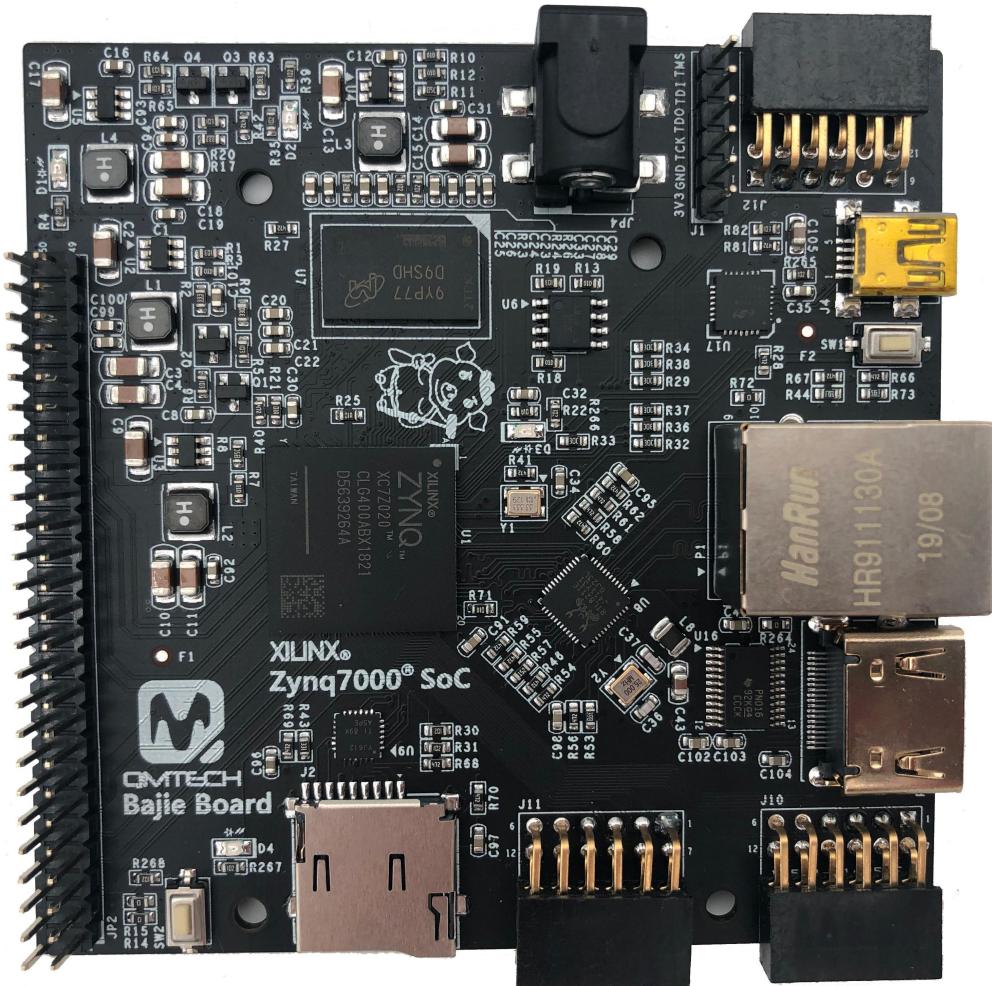


QMTECH ZYNQ7000 BAJIE BOARD

USER MANUAL



Preface

The QMTECH® ZYNQ7000 Bajie Board uses Xilinx Zynq®-7000 device which integrates the software programmability of an ARM®-based processor with the hardware programmability of an FPGA, enabling key analytics and hardware acceleration while integrating CPU, DSP, ASSP, and mixed signal functionality on a single device. Consisting of single-core Zynq-7000S and dual-core Zynq-7000 devices, the Zynq-7000 family is the best price to performance-per-watt, fully scalable SoC platform for your unique application requirements.

For more information, updates and useful links, please visit QMTECH Official Website:

<http://www.chinaqmtech.com>



QMTECH

QMTECH ZYNQ7000 Bajie Board

User Manual(Linux) V01

Table of Contents

1. INTRODUCTION.....	3
1.1 DOCUMENT SCOPE.....	3
2. GETTING STARTED.....	3
2.1 STEPS TO CUSTOMIZE THE PETALINUX WITHOUT HDMI DISPLAY.....	4
2.2 TEST THE ETHERNET UNDER LINUX ENVIRONMENT.....	11
2.3 STEPS TO CUSTOMIZE THE PETALINUX WITH HDMI DISPLAY.....	13
3. REFERENCE.....	25
4. REVISION.....	26

上海勸謀電子科技有限公司



QMTECH

QMTECH ZYNQ7000 Bajie Board

User Manual(Linux) V01

1. Introduction

1.1 Document Scope

This user manual introduces the procedure to make the PetaLinux environment running on the QMTECH ZYNQ7000 Bajie Board. The PetaLinux environment mainly covers three parts: u-boot, Linux OS and file system. All of those parts are developed with PetaLinux 2019.2 under Ubuntu 18.04.1 (64bit) environment. The prerequisites before working with the PetaLinux are shown as below:

1. Preferred and verified Ubuntu version is Ubuntu 18.04.1 (64bit).

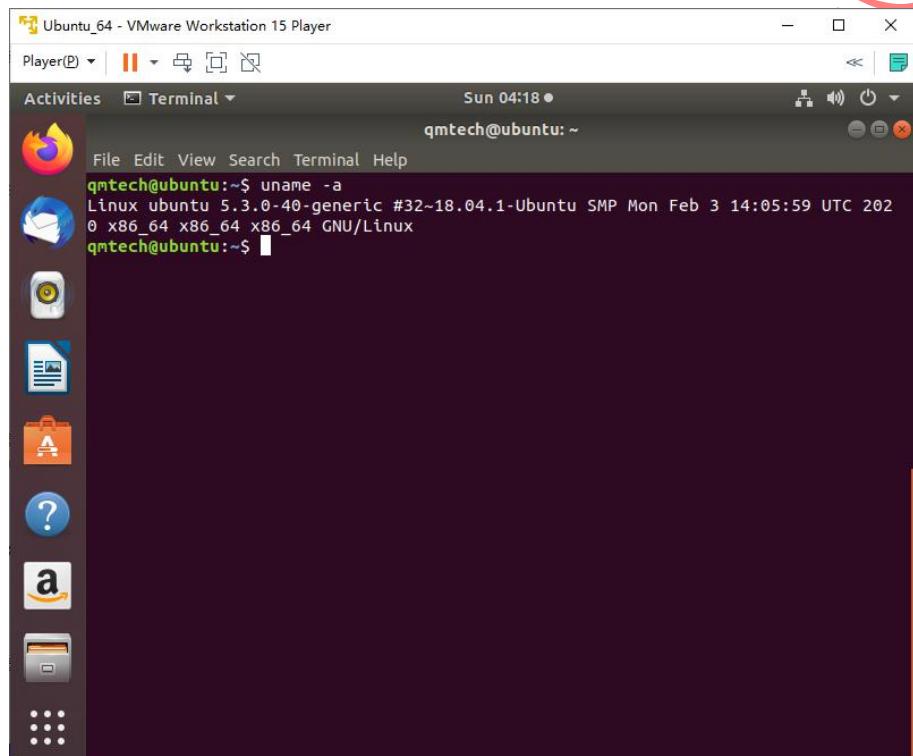


Figure 1-1. Ubuntu Version

2. Users shall install the required packages before install the PetaLinux 2019.2 in Ubuntu. The detailed required packages are mentioned in **UG1144: Table 2: Packages and Linux Workstation Environments**.
Notification: Users need to have root access to install the required packages mentioned in that table. The PetaLinux tools need to be installed as a non-root user.
3. Users shall have the basic knowledge about the usage of the Linux environment. Know how to use the cross-compile toolchain including arm-gcc-linux-, makefile, etc.

2. Getting Started

This chapter describes the detailed steps to create a customized PetaLinux. Comparing to the existing ZYNQ development board e.g. Xilinx ZC702, there are many differences in the QMTECH ZYNQ7000 Bajie Board. For example, there's only one 16bit width DDR3 memory chip connected to PS ARM core. And HDMI display interface is implemented at the PL side. Hence, the u-boot/ Linux device tree for the Bajie Board needs to be updated here.

2.1 Steps to Customize the Petalinux without HDMI Display

The first step is to customize the hardware info required by Petalinux. The hardware info could be retrieved from below Vivado project:

\$Bajie_Board\zynq_linux\Petalinux_No_HDMI\Project06_Petalinux_20200412_V02_No_HDMI.zip.
Below image shows the customized ZYNQ system in Vivado 2018.3. Make sure the project could successfully pass the three steps: Synthesis, implementation and Generate Bitstream.

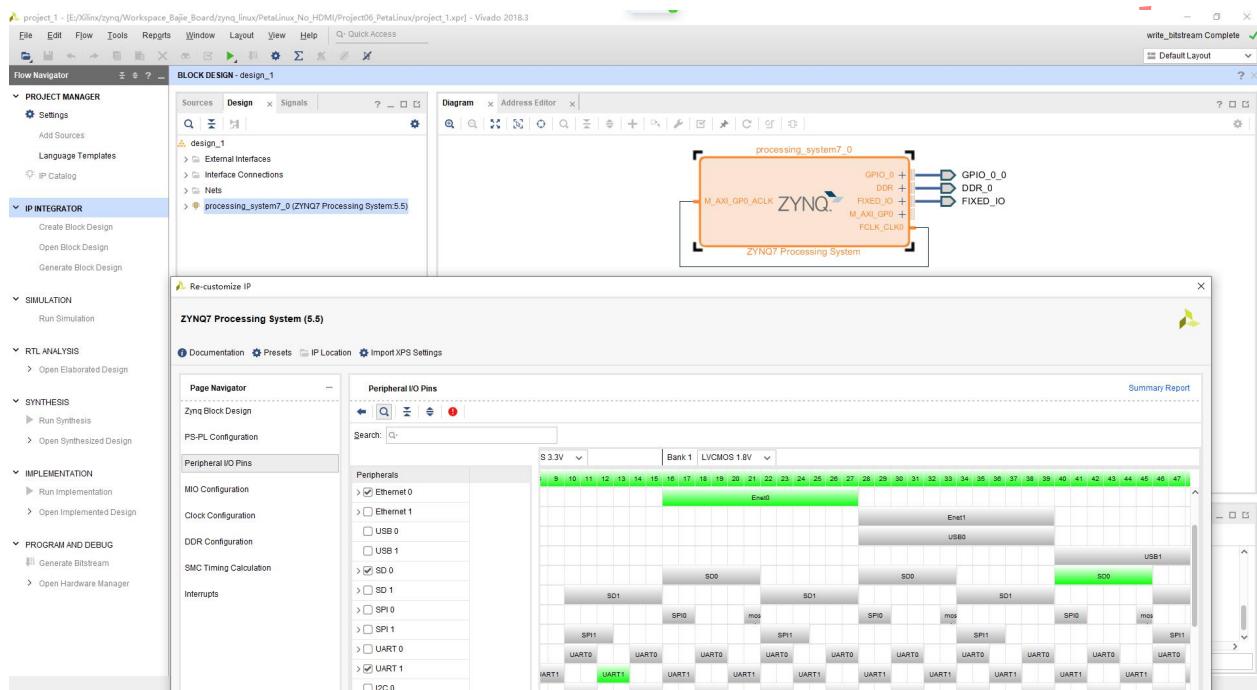


Figure 2-1. Modifications

Then users could retrieve the customized hardware info by clicking the [File] -> [Export] -> [Export Hardware]. And remember check the [Include Bitstream] before click the [OK] button.

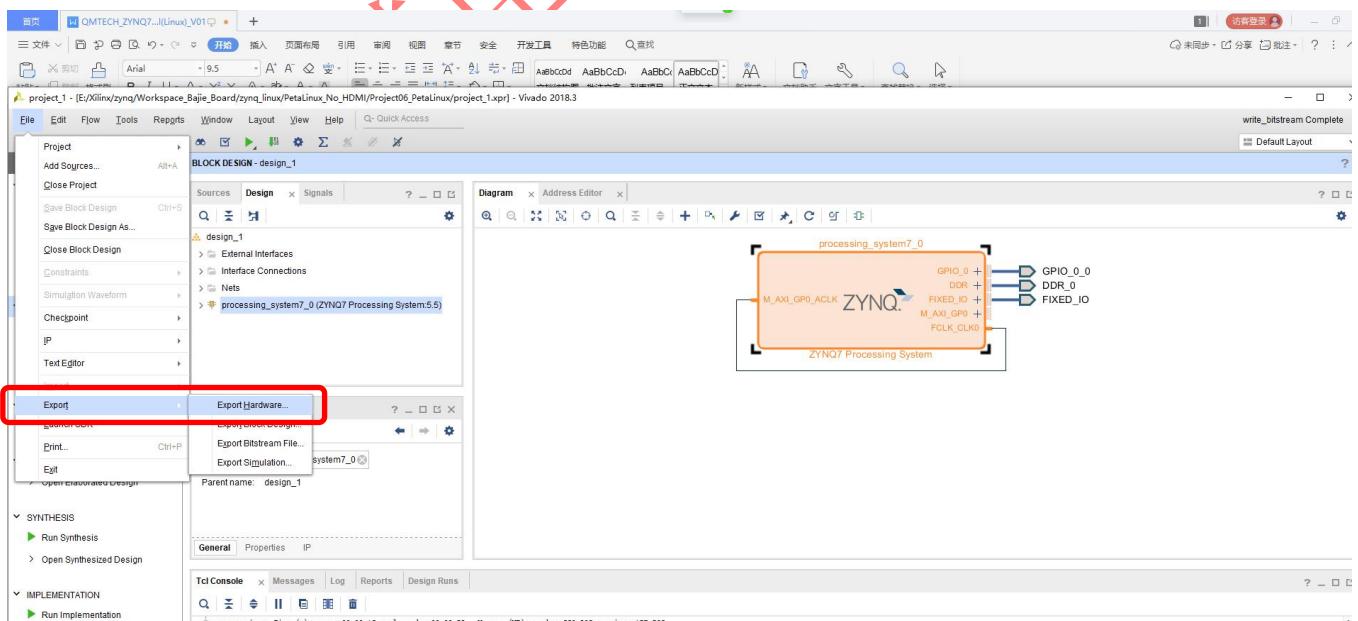


Figure 2-2. Export Hardware Info



The generated hardware info file could be found here:

\$Bajie_Board\zynq_linux\PetaLinux_No_HDMI\Project06_PetaLinux\project_1.sdk\design_1_wrapper.hdf

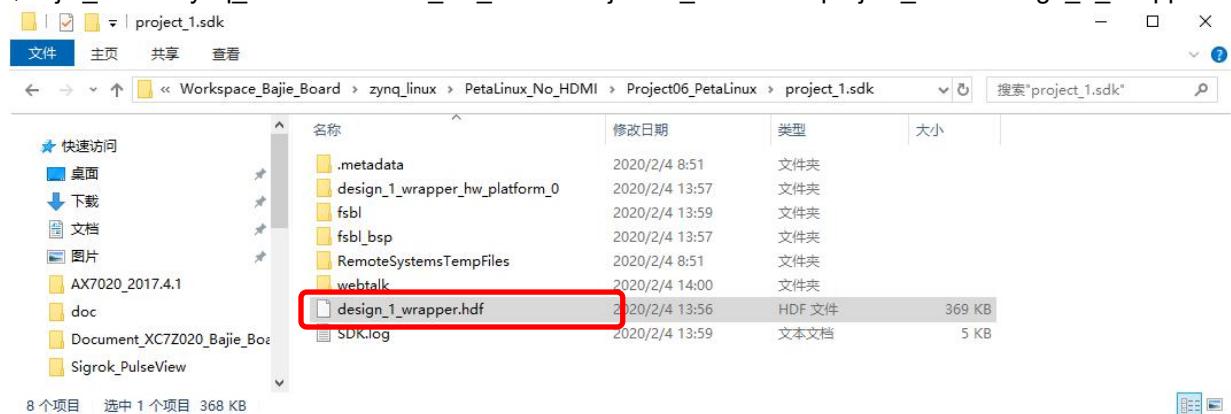


Figure 2-3. Target Hardware Info File

Copy the design_1_wrapper.hdf file into Ubuntu environment. E.g. put the file in folder \$workspace/Petalinux_workspace/linux_base.sdk.

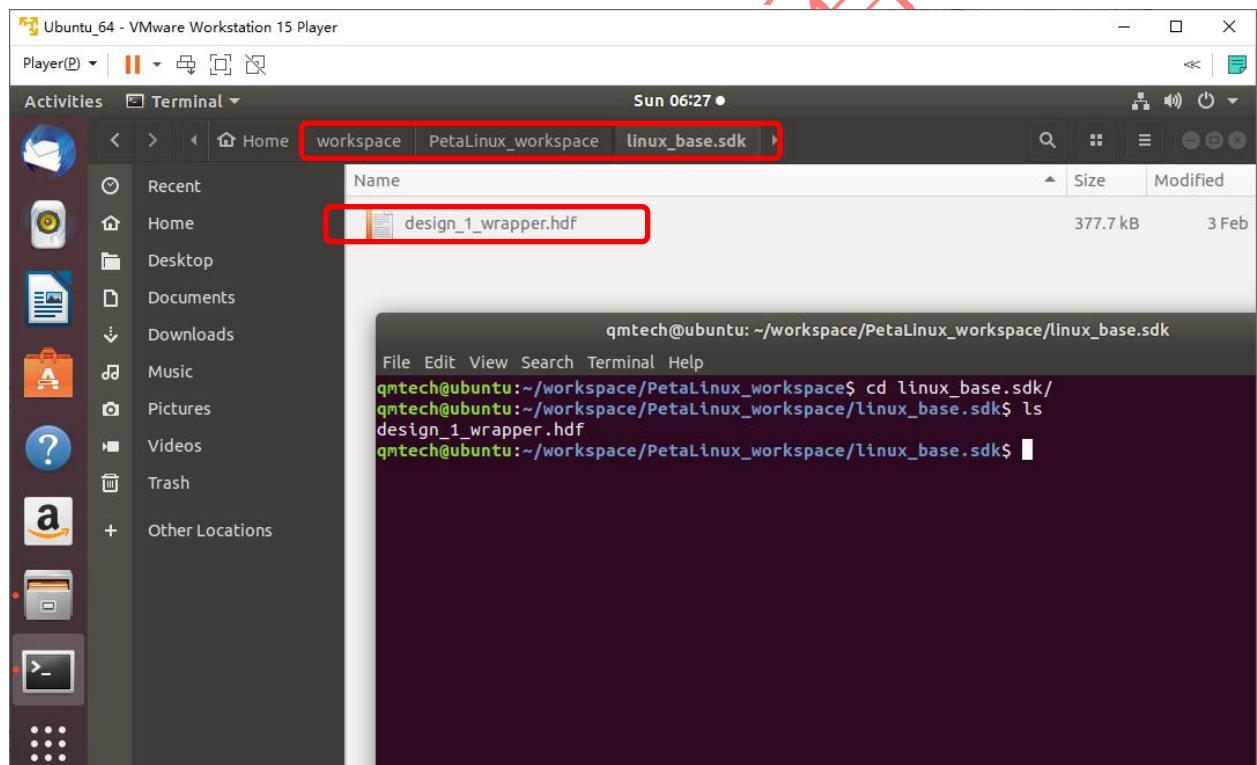


Figure 2-4. Import Hardware Info File

In our example, the PetaLinux 2019.2 package is installed in folder /opt/pkg/petalinux. Users need change the below commands according to the detailed directory that contains the PetaLinux.

Type command in the terminal to source the PetaLinux: `/opt/pkg/petalinux/settings.sh`. Create a new folder named as `qmtech_peta_01` by typing command: `petalinux-create --type project --template zynq --name qmtech_peta_01`. This new folder is in the same directory as the `linux_base.sdk` and used as the PetaLinux workspace.

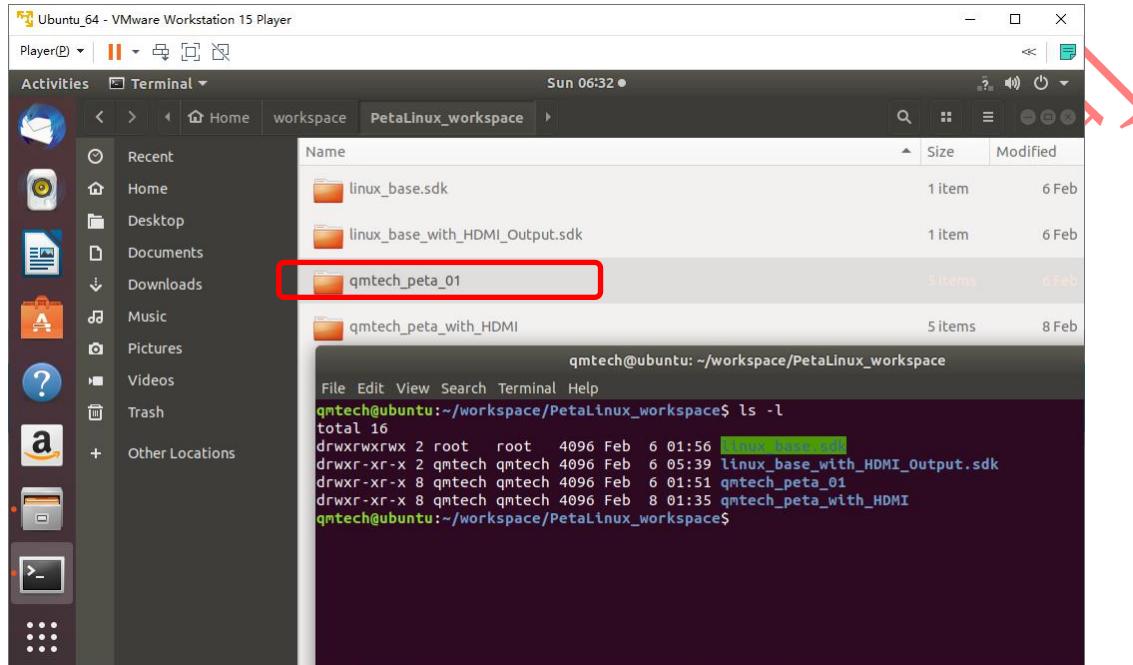


Figure 2-5. Create PetaLinux Working Folder

Then enter into the folder `qmtech_peta_01`.

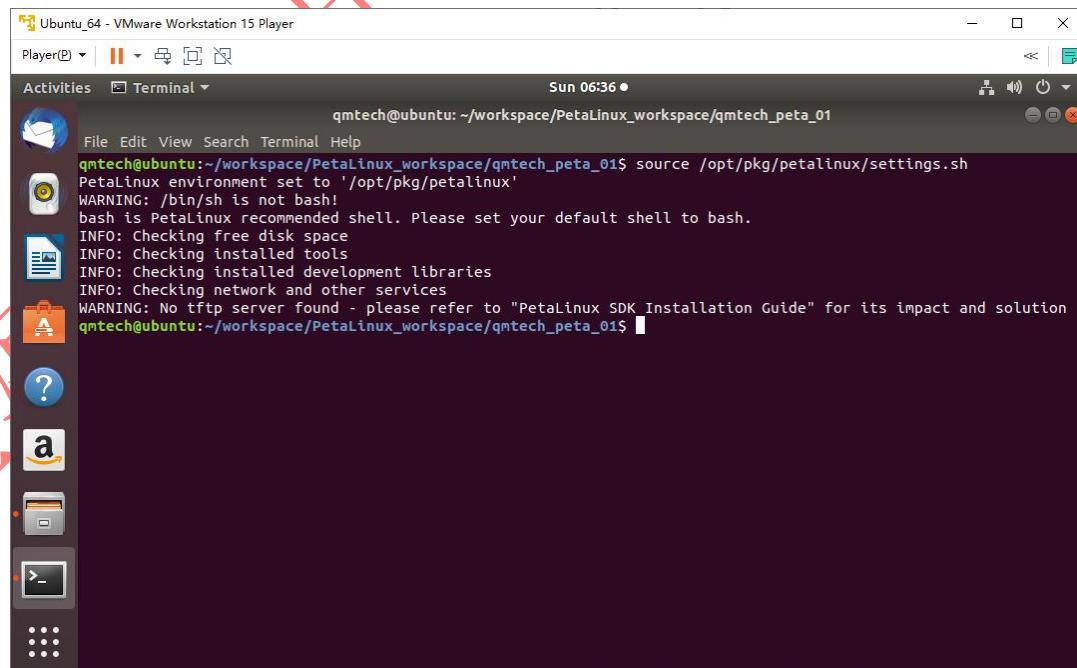


Figure 2-6. Enter Workspace

Import the hardware info into PetaLinux by command: `petalinux-config --get-hw-description ./linux_base.sdk`. Below configuration image will display and nothing needs to be changed here.

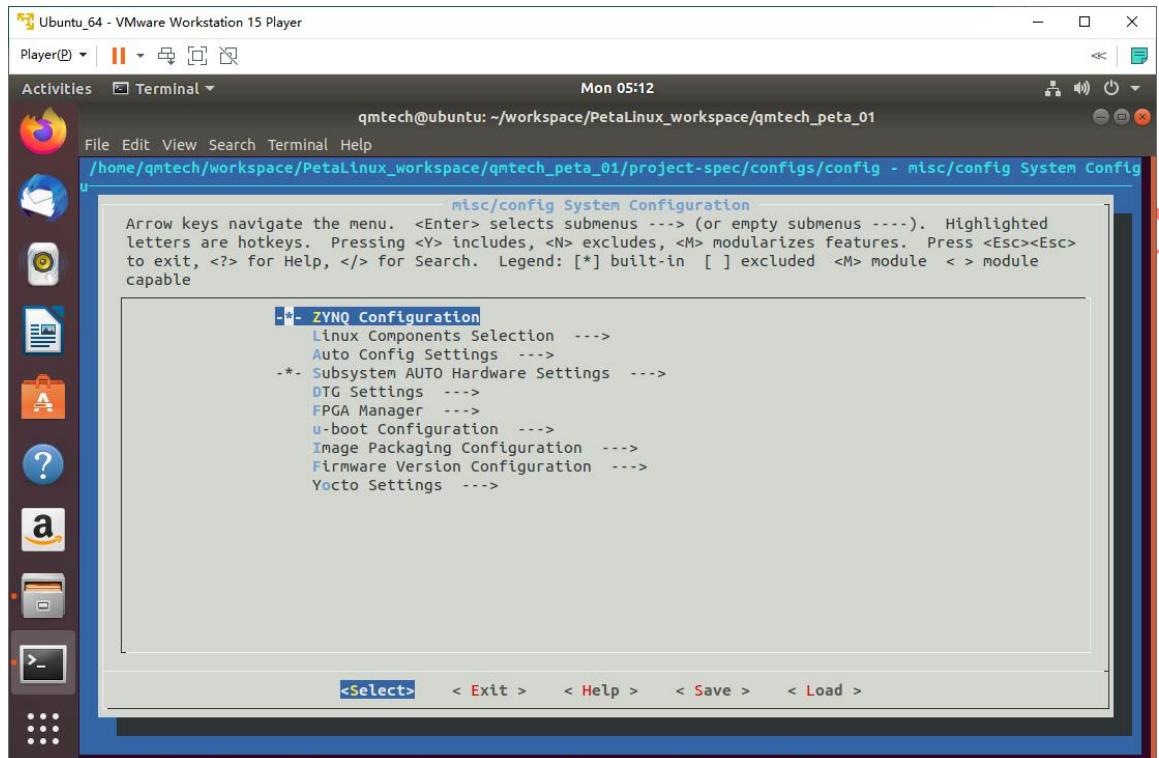


Figure 2-7. ZYNQ Configuration

```
Ubuntu_64 - VMware Workstation 15 Player
Player(P) | || - < > X
Activities Terminal Mon 05:13
qmtech@ubuntu: ~/workspace/PetaLinux_workspace/qmtech_peta_01
File Edit View Search Terminal Help
INFO: Rename design_1_wrapper.hdf to system.hdf
[INFO] generating Kconfig for project
[INFO] menuconfig project
ERROR: Failed to menu config project component
ERROR: Failed to config project.
ERROR: Get hw description Failed!.
qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_01$ pwd
/home/qmtech/workspace/PetaLinux_workspace/qmtech_peta_01
qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_01$ petalinux-config --get-hw-description ./linux_base.sdk
INFO: Getting hardware description...
INFO: Rename design_1_wrapper.hdf to system.hdf
[INFO] generating Kconfig for project
[INFO] menuconfig project

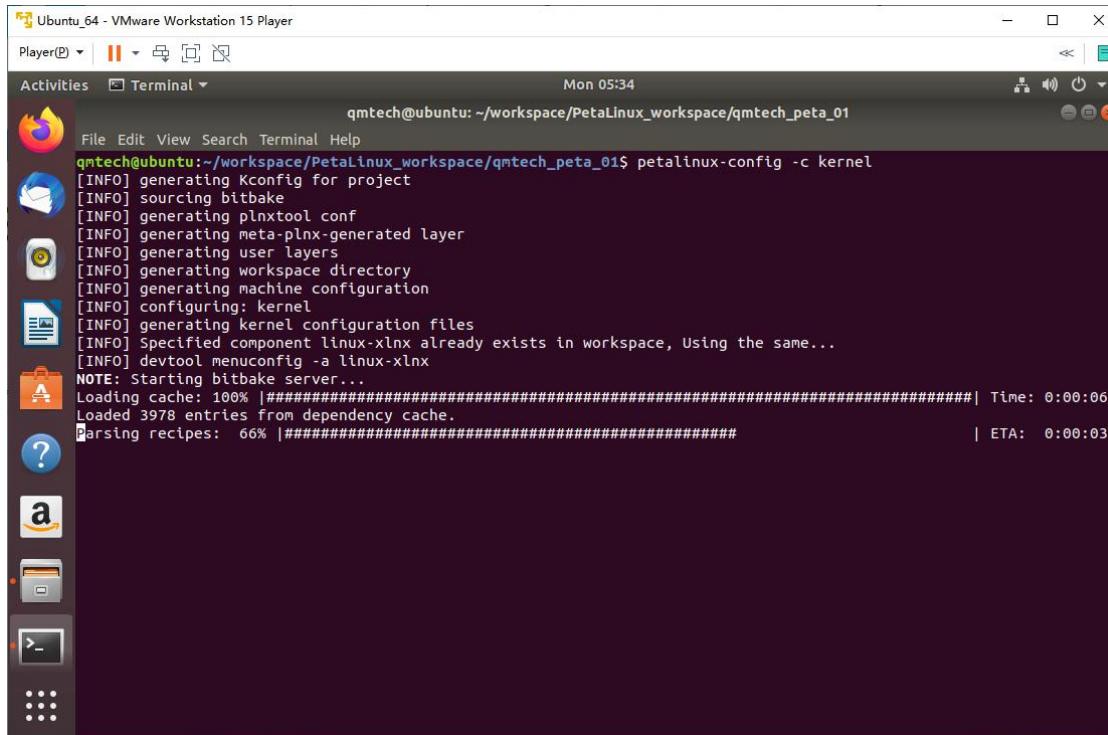
*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

[INFO] sourcing bitbake
[INFO] generating plntool conf
[INFO] generating meta-plnx-generated layer
[INFO] generating user layers
[INFO] generating workspace directory
[INFO] generating machine configuration
[INFO] generating bbappends for project . This may take time !
[INFO] generating u-boot configuration files
[INFO] generating kernel configuration files
[INFO] generating kconfig for Rootfs
[INFO] silentconfig rootfs
[INFO] generating petalinux-user-image.bb
qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_01$
```

Figure 2-8. Import Hardware Info

Once the hardware info is successfully imported, users may start to configure the PetaLinux kernel by command: **petalinux-config -c kernel**

This step may take a really long time. Be patient here and make sure the VM's network status is perfect.



```
Ubuntu_64 - VMware Workstation 15 Player
Player(P) | ||| □ ⊞ ⊖ ⊖
Activities Terminal Mon 05:34
qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_01$ petalinux-config -c kernel
[INFO] generating Kconfig for project
[INFO] sourcing bitbake
[INFO] generating pnxtool conf
[INFO] generating meta-pnx-generated layer
[INFO] generating user layers
[INFO] generating workspace directory
[INFO] generating machine configuration
[INFO] configuring: kernel
[INFO] generating kernel configuration files
[INFO] Specified component linux-xlnx already exists in workspace, Using the same...
[INFO] devtool menuconfig -a linux-xlnx
NOTE: Starting bitbake server...
Loading cache: 100% |#####| Time: 0:00:06
Loaded 3978 entries from dependency cache.
Parsing recipes: 66% |#####| ETA: 0:00:03
```

Figure 2-9. Configure PetaLinux Kernel

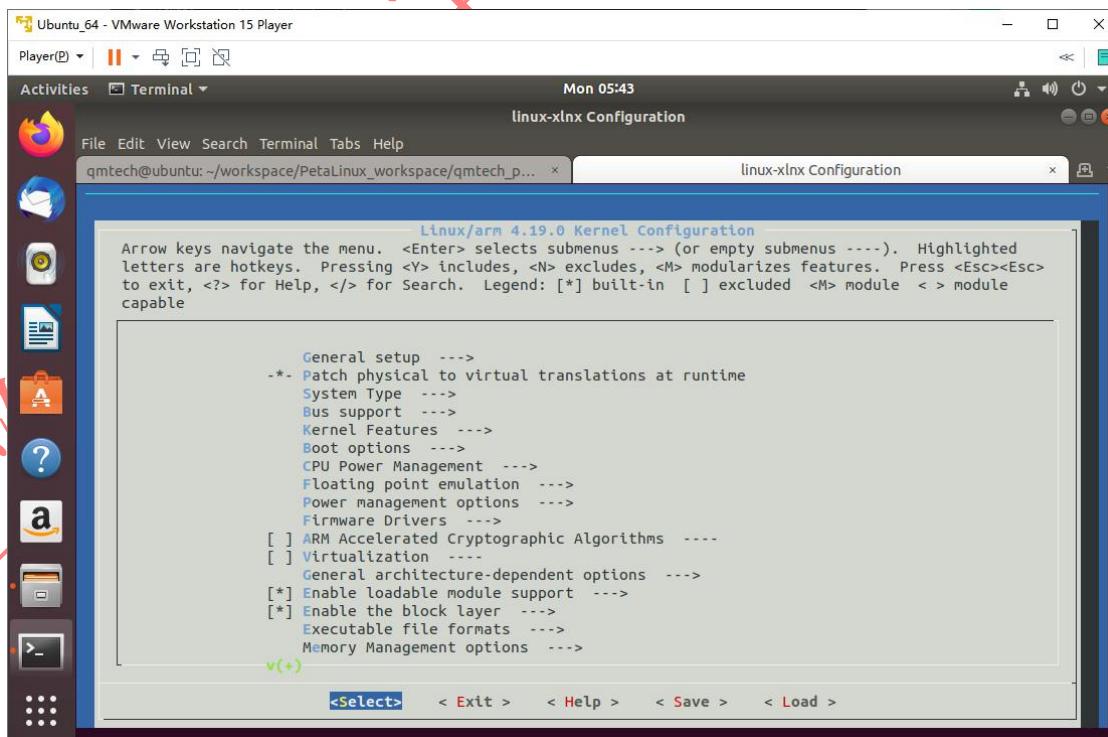


Figure 2-10. No Need to change the Kernel Configurations

Once the PetaLinux Kernel info is successfully configured, users may start to configure the PetaLinux file system by command: **petalinux-config -c rootfs**
 Below image will display and nothing needs to be changed here.

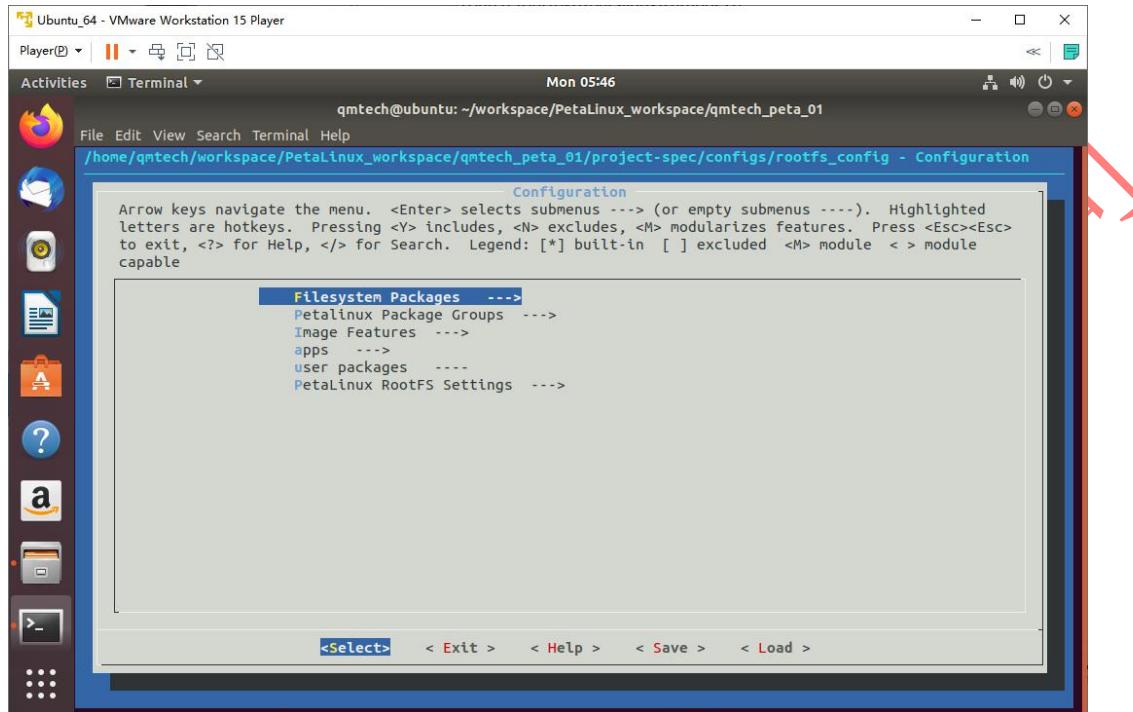


Figure 2-11. No Need to change the rootfs

Start to build the PetaLinux by command: **petalinux-build**

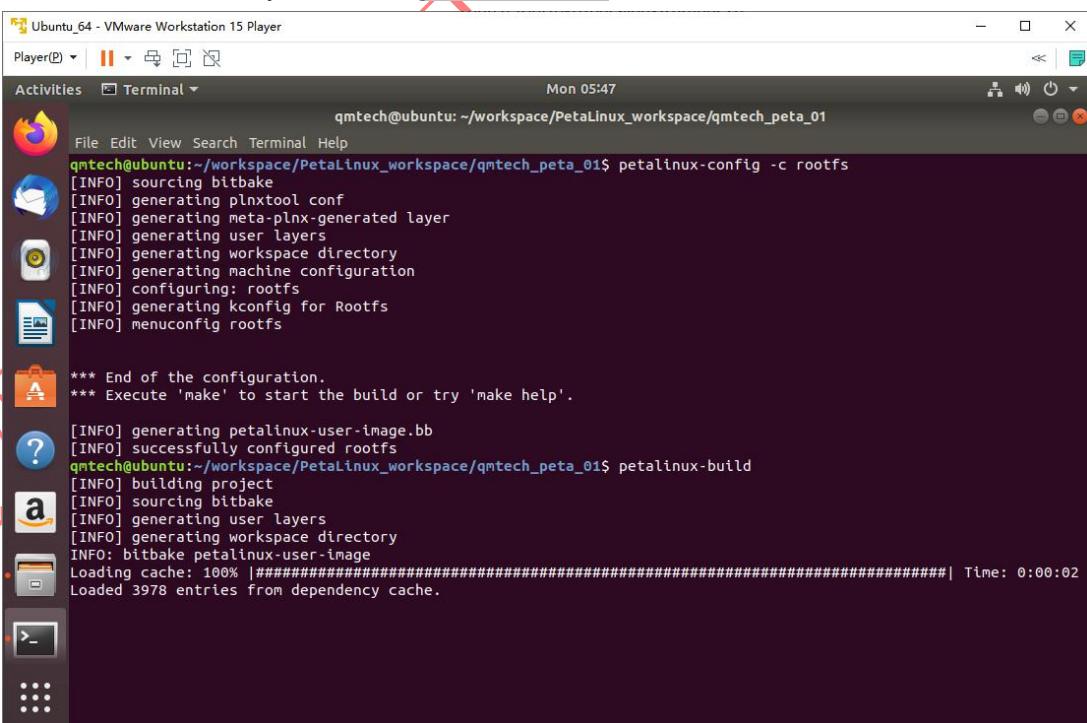


Figure 2-12. Build PetaLinux

After the PetaLinux is successfully built, users could generate the target BOOT.bin and image.ub by command: **petalinux-package --boot --fsbl ./images/linux/zynq_fsbl.elf --fpga -u-u-boot --force**

Ubuntu_64 - VMware Workstation 15 Player

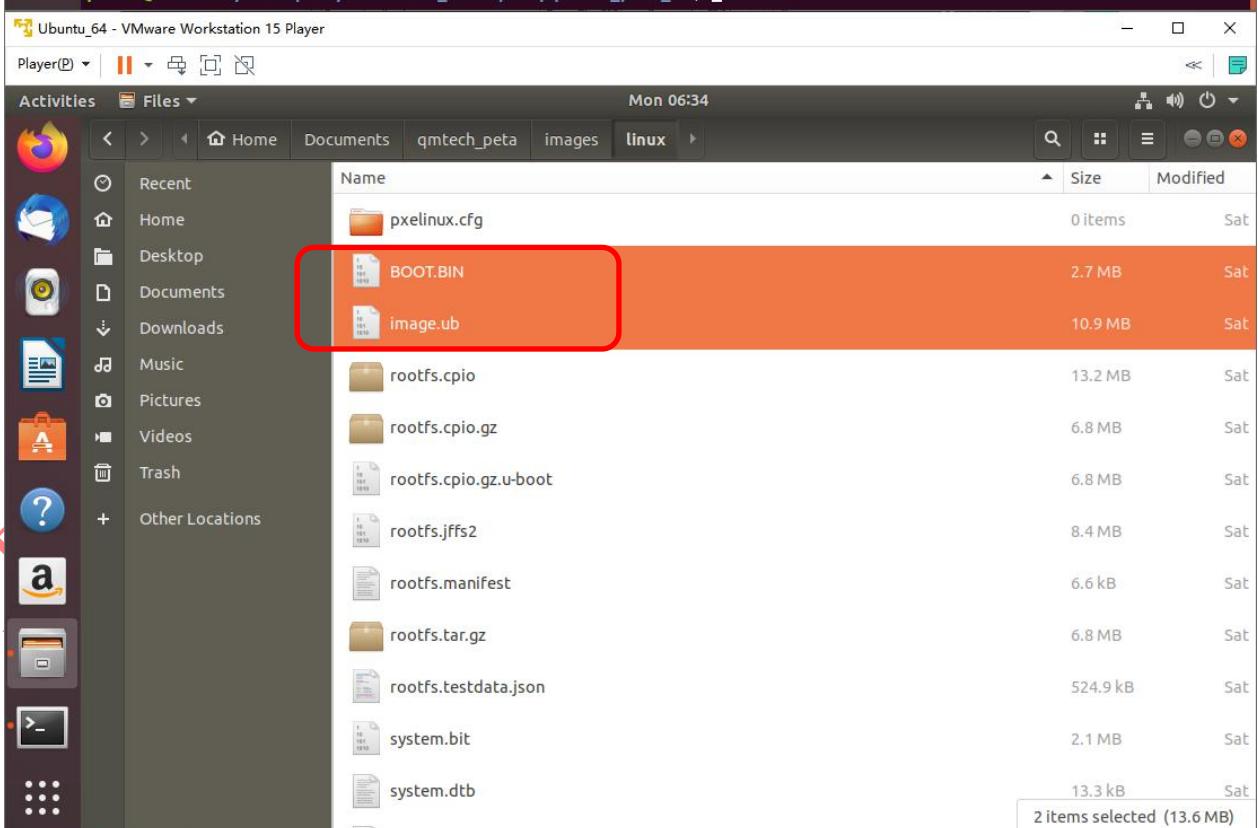
Player(P) | II |

Activities Terminal Mon 06:34

qmtech@ubuntu: ~/workspace/PetaLinux_workspace/qmtech_peta_01

```
File Edit View Search Terminal Help
INFO: bitbake petalinux-user-image
Loading cache: 100% [#####################################################################
] Time: 0:00:02
Loaded 3978 entries from dependency cache.
Parsing recipes: 100% [#####################################################################
] Time: 0:00:49
Parsing of 2893 .bb files complete (2890 cached, 3 parsed). 3980 targets, 169 skipped, 0 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies
Initialising tasks: 100% [#####################################################################
] Time: 0:00:04
Checking sstate mirror object availability: 100% [#####################################################################
] Time: 0:05:13
Sstate summary: Wanted 125 Found 10 Missed 230 Current 754 (8% match, 86% complete)
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 3214 tasks of which 3147 didn't need to be rerun and all succeeded.
INFO: Copying Images from deploy to images
NOTE: Failed to copy built images to tftp dir: /tftpboot
[INFO] successfully built project
qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_01$ petalinux-package --boot --fsbl ./images/linux/zynq_fsbl.elf --fpga --u-boot --force
INFO: File in BOOT BIN: "/home/qmtech/workspace/PetaLinux_workspace/qmtech_peta_01/images/linux/zynq_fsbl.elf"
INFO: File in BOOT BIN: "/home/qmtech/workspace/PetaLinux_workspace/qmtech_peta_01/project-spec/hw-description/design_1_wrapper.bit"
INFO: File in BOOT BIN: "/home/qmtech/workspace/PetaLinux_workspace/qmtech_peta_01/images/linux/u-boot.elf"
INFO: Generating Zynq binary package BOOT.BIN...
*****
Xilinx Bootgen v2019.2
**** Build date : Oct 23 2019-22:59:42
** Copyright 1986-2019 Xilinx, Inc. All Rights Reserved.

INFO: Binary is ready.
WARNING: Unable to access the TFTPBOOT folder /tftpboot!!!
WARNING: Skip file copy to TFTPBOOT folder!!!
qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_01$
```



Copy the BOOT.bin and image.ub into MicroSD card and then insert the MicroSD card into ZYNQ7000 Bajie Board. Plug the MiniUSB cable into the Bajie Board and then power on the board with 5V DC power source. Below image shows the serial output from the board.

```

U-Boot 2019.01 (Feb 08 2020 - 09:32:41 +0000) Xilinx Zynq ZC702
CPU: Zynq 7z020
Siilicon: v3.1
Model: QMTECH Zynq7000 Bajie Development Board
DRAM: ECC disabled 512 MiB
MMC: mmc@e0100000: 0
Loading Environment From FAT... *** Warning - bad CRC, using default environment

In:    serial@e0001000
Out:   serial@e0001000
Err:   serial@e0001000
Model: QMTECH Zynq7000 Bajie Development Board
Net:   ZYNQ GEM: e000b000, phyaddr 1, interface rgmii-id
eth0: ethernet@e000b000
U-BOOT for qmtech_peta_with_HDMI

ethernet@e000b000 Waiting For PHY auto negotiation to complete..... TIMEOUT !
Hit any key to stop autoboot: 4 000 3 000 2 000 1 000 0
Device: mmc@e0100000
Manufacturer ID: 3
OEM: 5344
Name: SU026
Bus Speed: 15873016
Mode : SD High Speed (50MHz)
Rd Block Len: 512
SD version 3.0
High Capacity: No
Capacity: 1.8 GiB
Bus Width: 4-bit
Erase Group Size: 512 Bytes
10954408 bytes read in 1453 ms (7.2 MiB/s)
## Loading kernel from FIT Image at 10000000 ...
Using 'conf@system-top.dtb' configuration
Verifying Hash Integrity ... OK
Trying 'kernel@0' kernel subimage
Description: Linux kernel
Type: Kernel Image
Compression: uncompressed
Data Start: 0x100000f4
Data Size: 4122984 Bytes = 3.9 MiB
Architecture: ARM
OS: Linux
Load Address: 0x00000000
Entry Point: 0x00000000
Hash algo: sha1
Hash value: 15e57fb316251305f93123f92a6939902842e77
Verifying Hash Integrity ... sha1+ OK
## Loading ramdisk from FIT Image at 10000000 ...
Using 'conf@system-top.dtb' configuration

```

COM68 OPENED, 115200, 8, NONE, 1, OFF Rx: 31,953 Bytes Tx: 0 Bytes

Figure 2-13. PetaLinux Boot Log

2.2 Test the Ethernet Under Linux Environment

Plug the ethernet cable into the Bajie Board and power on it. Below log will be displayed on the terminal tool when the ethernet link is ready:



```

haveged: haveged: tot tests(BA8): A:1/1 B:1/1 continuous tests(B): last entropy estimate 8.00087
haveged: haveged: fills: 0, generated: 0
random: crng init done
Public key portion is:
ssh-rsa
AAAAB3NzaC1yc2EAAQABAAAQADQwtIFFJABkTCPbEiy4FBuJHNycwlbCAvqN0kUxNR3pt11S2S01xAvTS4H991
kUo69dwCsvodhjF07p1pGUBr/Z3WLoZoF14TKDn/
0z0BgiJ5U2mKeEG1aKw727cfQ7Xy2hi9uef411st1N9IUNL4WIVp3aN22/zr2aZj30UGlhXZPo4TR5+04Tsn/
TLXQRpsdQNu8Npa8e2v+5qM+qSrucqHt10gpK3mtpp7Jyc/LFa61nm0013d6HuqF8G/jqlvZm96Uve4kdiUgkbL
+BJK2diJyppGSNqMcThbMuKrTH2/buTQznoIuYhsootsBN0JDqzRQSAtvnof1 root@qmtech_peta_with_HDMI
Fingerprint: sha1!! 41:cd:3b:93:aa:ad:b7:1c:6:c:0:c:62:f1:29:31:8:b:a8:b5:b9:a8:c
dropbear.
hwclock: can't open '/dev/misc/rtc': No such file or directory
Starting internet superserver: inetd.
Starting syslogd/klogd: done
Starting tcf-agent: OK

Petalinux 2019.2 qmtech_peta_with_HDMI /dev/ttys0
qmtech_peta_with_HDMI login: root
Password:
root@qmtech_peta_with_HDMI:~# macb e000b000.ethernet eth0: link up
(100/FULL)
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
root@qmtech_peta_with_HDMI:~#
root@qmtech_peta_with_HDMI:~#

```

Figure 2-14. Log Info

Users may type below ethernet test related commands to check the status of ethernet interface.

```

qmtech_peta_with_HDMI login: root
Password:
root@qmtech_peta_with_HDMI:~# macb e000b000.ethernet eth0: link up
(100/FULL)
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
root@qmtech_peta_with_HDMI:~#
root@qmtech_peta_with_HDMI:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0a:35:00:1e:53
          inet  addr:192.168.0.2  Bcast:255.255.255.255 Mask:255.255.255.0
          inet6 addr: fe80::2a:3ff:fe0:1e53/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500 Metric:1
            RX packets:3 errors:0 dropped:0 overruns:0 frame:0
            TX packets:49 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:1198 (1.1 KiB)  TX bytes:11616 (11.3 KiB)
            Interrupt:28 Base address:0xb000

root@qmtech_peta_with_HDMI:~#
root@qmtech_peta_with_HDMI:~# ping 192.168.0.1
PING 192.168.0.1 (192.168.0.1): 56 data bytes
64 bytes from 192.168.0.1: seq=0 ttl=64 time=0.762 ms
64 bytes from 192.168.0.1: seq=1 ttl=64 time=0.714 ms
64 bytes from 192.168.0.1: seq=2 ttl=64 time=0.707 ms
64 bytes from 192.168.0.1: seq=3 ttl=64 time=0.691 ms
64 bytes from 192.168.0.1: seq=4 ttl=64 time=0.707 ms

```

Figure 2-15. Echo Test under Linux

2.3 Steps to Customize the Petalinux with HDMI Display

This chapter describes the steps to customize a new Petalinux with HDMI display interface enabled. With this feature, users could develop some GUI based applications. Here, only lists the keys steps that enable the Petalinux HDMI interface.

The first step is to customize the hardware info required by Petalinux. The hardware info could be retrieved from below Vivado project:

\$Bajie_Board\zynq_linux\Petalinux_With_HDMI\Project06_Petalinux_With_HDMI_20200412_V02.zip.

Below image shows the customized ZYNQ system in Vivado 2018.3. Make sure the project could successfully pass the three steps: Synthesis, implementation and Generate Bitstream.

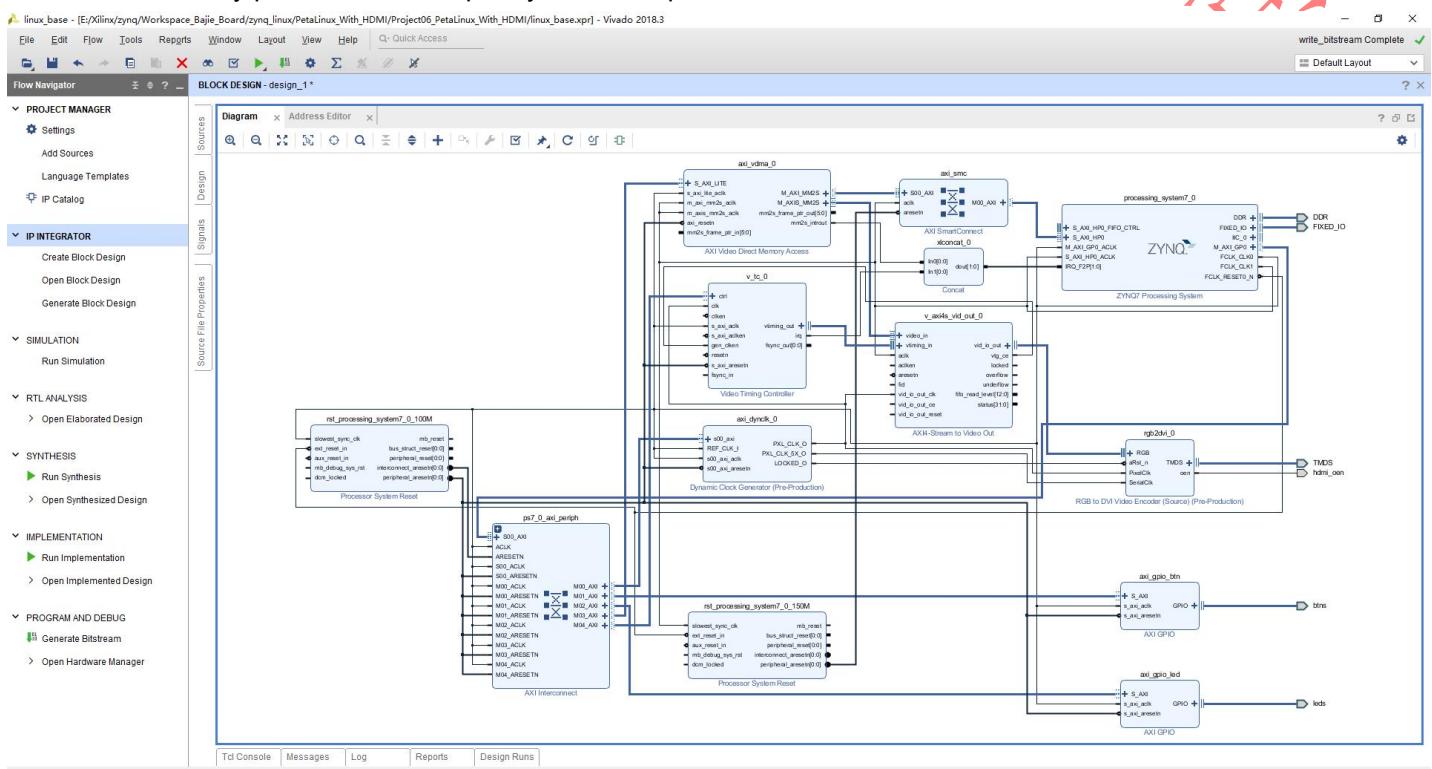
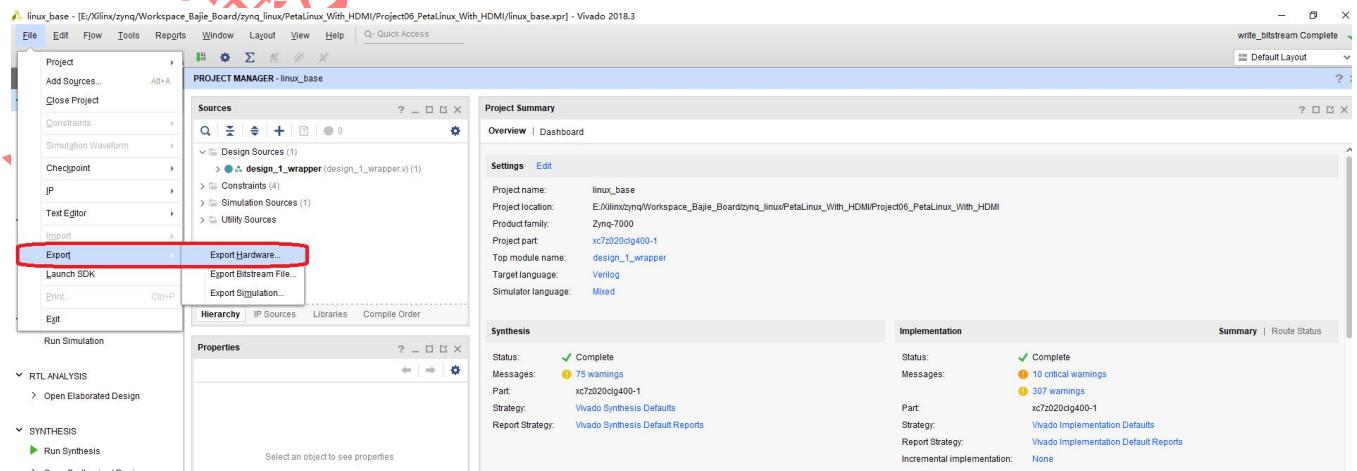


Figure 2-16. Hardware Architecture

Then users could retrieve the customized hardware info by clicking the [File] -> [Export] -> [Export Hardware]. And remember check the [Include Bitstream] before click the [OK] button.



Copy the design_1_wrapper.hdf file into Ubuntu environment. E.g. put the file in folder \$workspace/PetaLinux_workspace/linux_base_with_HDMI_Output.sdk.

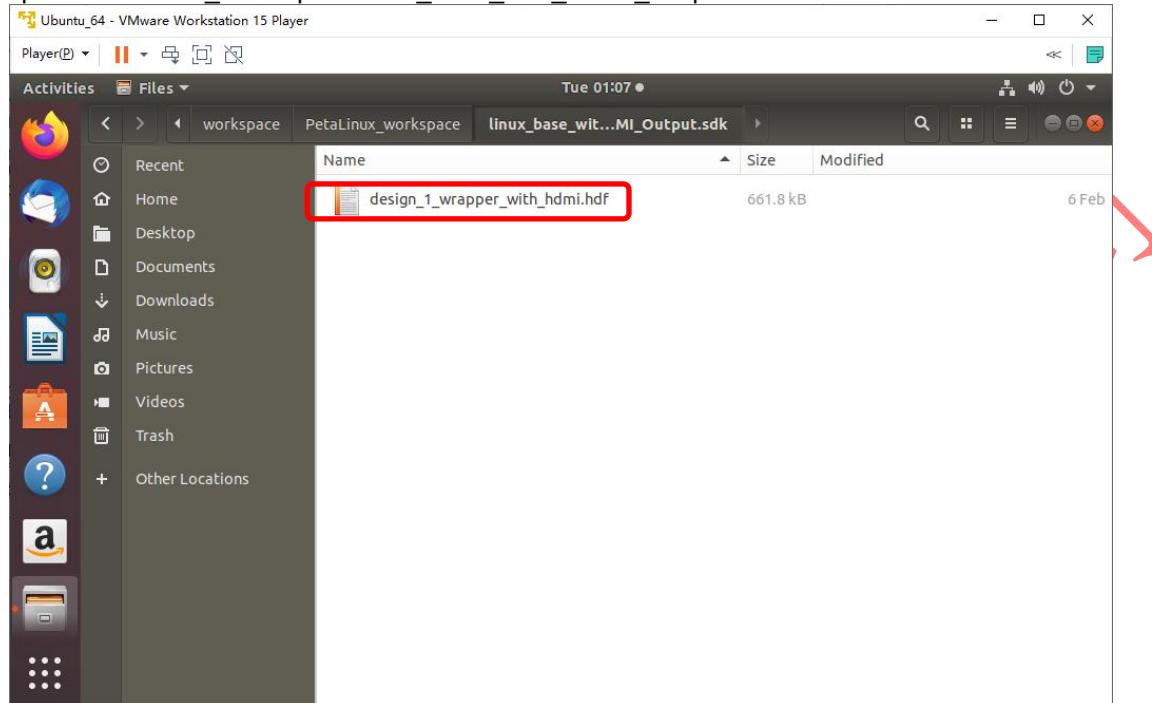


Figure 2-17. Create PetaLinux Working Folder

Type command in the terminal to source the PetaLinux: /opt/pkg/petalinux/settings.sh. Create a new folder named as qmtech_peta_with_HDMI by typing command: petalinux-create --type project --template zynq --name qmtech_peta_with_HDMI. This new folder is in the same directory as the linux_base_with_HDMI_Output.sdk and used as the PetaLinux workspace.

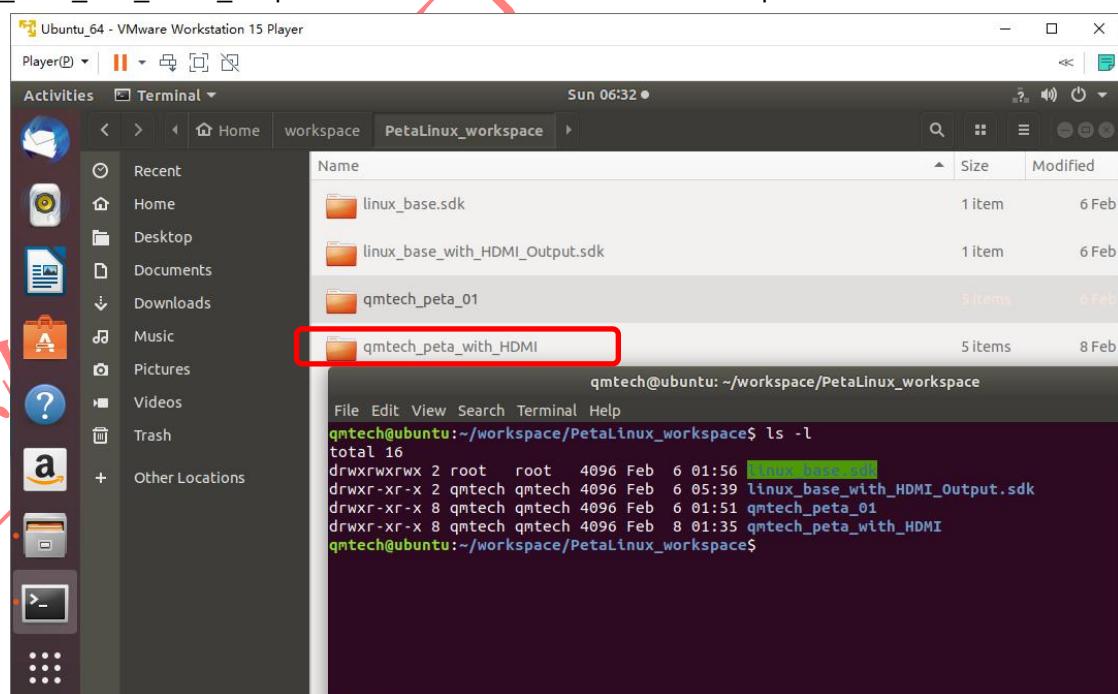


Figure 2-18. Create PetaLinux Working Folder

Import the hardware info into PetaLinux by command: **petalinux-config --get-hw-description ..\linux_base_with_HDMI_Output.sdk**. Below configuration image will display and nothing needs to be changed.

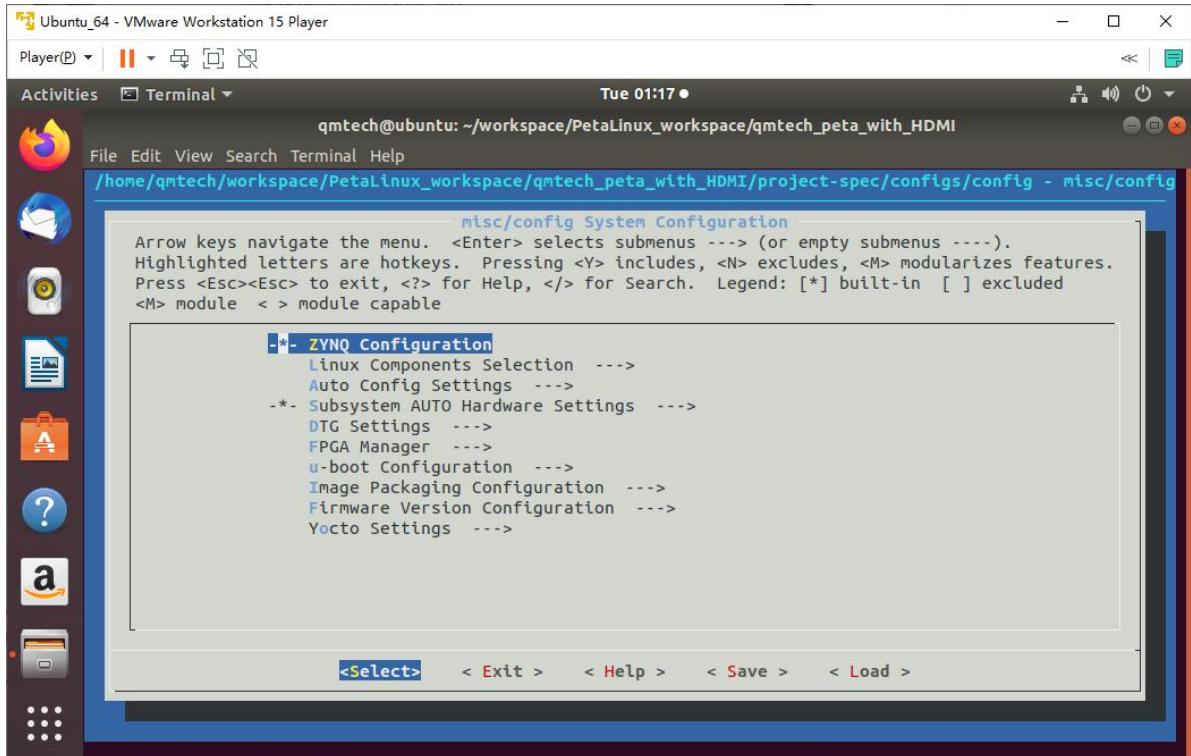


Figure 2-19. ZYNQ Configuration

Ubuntu_64 - VMware Workstation 15 Player

Player(P) | || | ☰ | Activities Terminal Tue 01:18 ●

qmtech@ubuntu: ~/workspace/PetaLinux_workspace/qmtech_peta_with_HDMI\$

File Edit View Search Terminal Help

WARNING: No tftp server found - please refer to "PetaLinux SDK Installation Guide" for its impact and solution

qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_with_HDMI\$ clear

qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_with_HDMI\$ petalinux-config --get-hw-description
./linux_base_with_HDMI_Output.sdk
INFO: Getting hardware description...
INFO: Rename design_1_wrapper_with_hdmi.hdf to system.hdf
[INFO] generating Kconfig for project
[INFO] menuconfig project

*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

[INFO] sourcing bitbake
[INFO] generating plnxtool conf
[INFO] generating meta-plnx-generated layer
[INFO] generating user layers
[INFO] generating workspace directory
[INFO] generating machine configuration
[INFO] generating bbappends for project . This may take time !
[INFO] generating u-boot configuration files
[INFO] generating kernel configuration files
[INFO] generating kconfig for Rootfs
[INFO] silentconfig rootfs
[INFO] generating petalinux-user-image.bb

qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_with_HDMI\$

Figure 2-20. Import Hardware Info



Once the hardware info is successfully imported, users may start to modify the Linux kernel to enable the HDMI display port. Mainly two parts needs to be updated here.

- [Dynamic Clock Generator](#)
- [RGB to DVI Video Encoder](#)

The above two drivers are maintained by Digilent. Users may find the latest drivers in their official GitHub. To enable the dynamic clock generator, users need to put the **clk-dgInt-dynclk.c** into below folder.

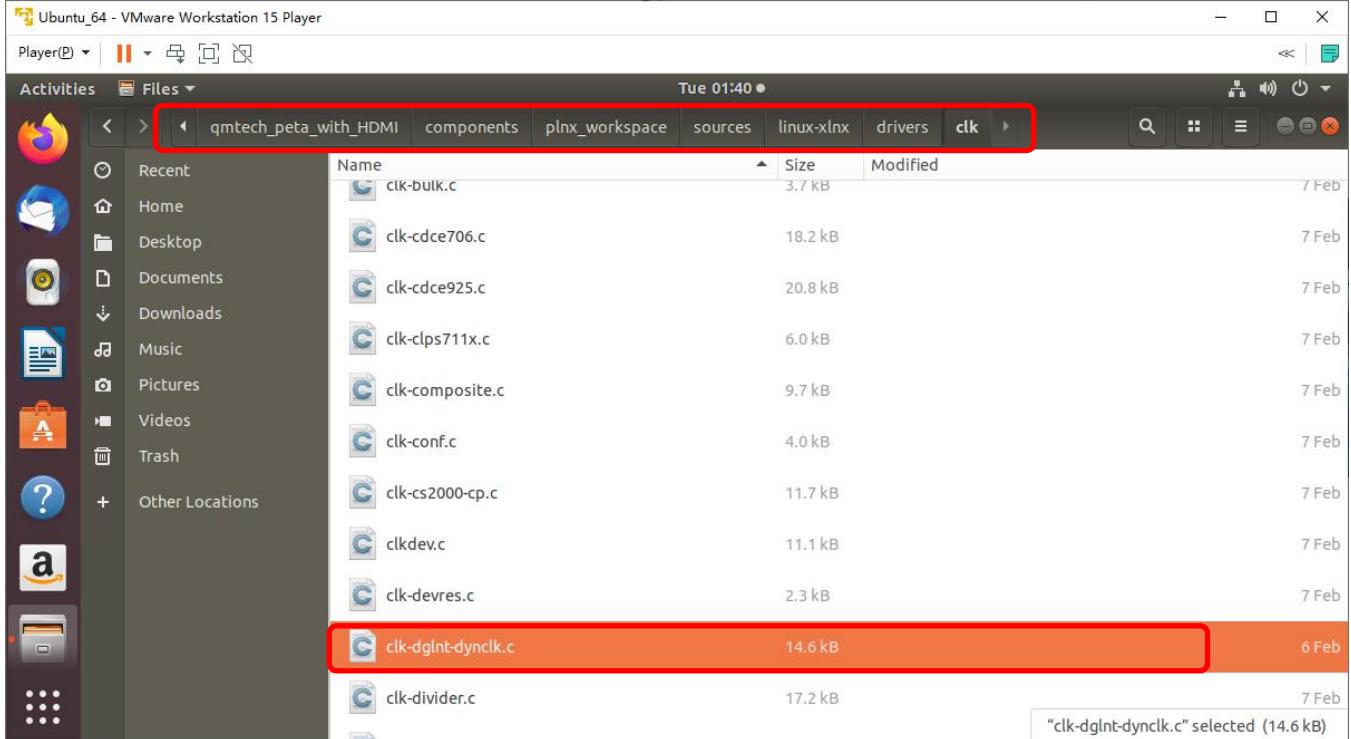


Figure 2-21. Add Clock Driver

Modify the KConfig file in the **clk** folder and add below code line to make the Digilent clock driver module enabled.

```

config COMMON_CLK_AXI_CLKGEN
    tristate "AXI clkgen driver"
    depends on ARCH_ZYNQ || MICROBLAZE || COMPILER_TEST
    help
    ---help---
        Support for the Analog Devices axi-clkgen pcore clock generator for Xilinx FPGAs. It is commonly used in Analog Devices' reference designs.

config COMMON_CLK_DGINT_DYNCLK
    tristate "Digilent axi_dynclk Driver"
    depends on ARCH_ZYNQ || MICROBLAZE
    help
    ---help---
        Support for the Digilent AXI Dynamic Clock core for Xilinx FPGAs.

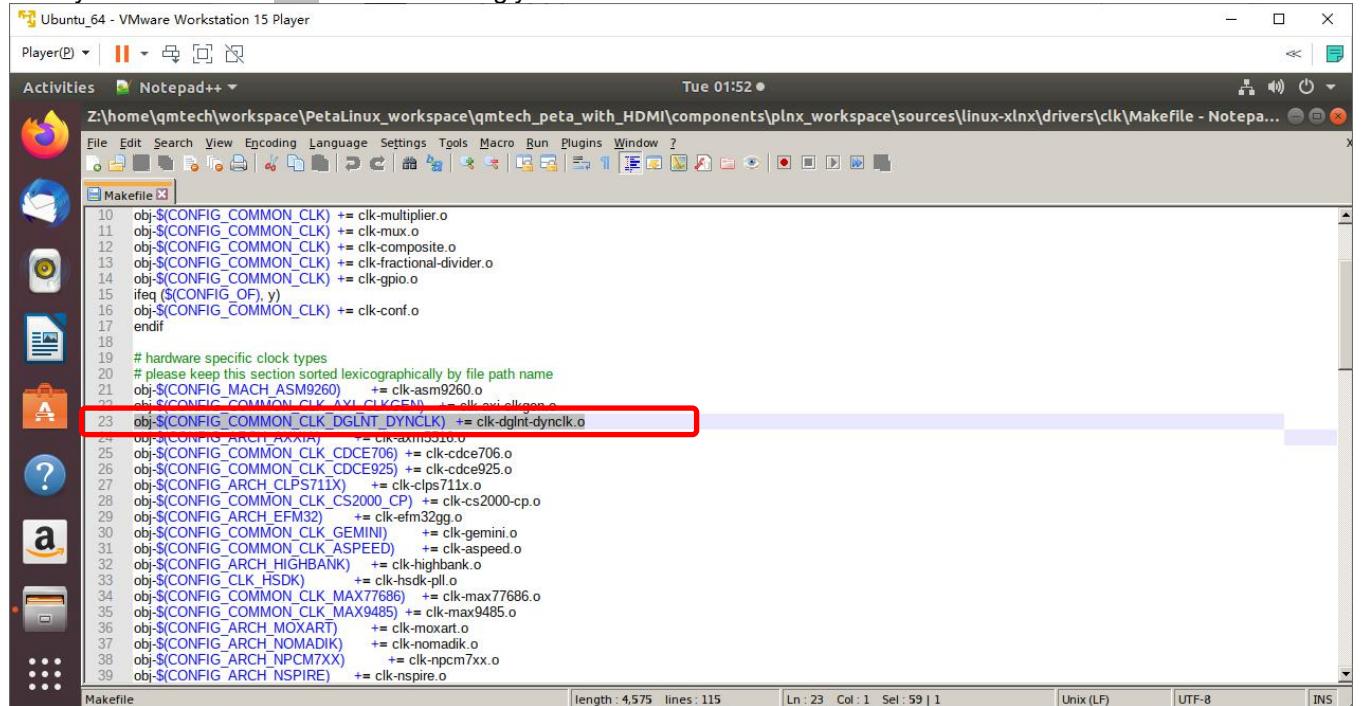
config CLK_QORIQ
    bool "Clock driver for Freescale QorIQ platforms"
    depends on (PPC_E500MC || ARM || ARM64 || COMPILER_TEST) && OF
    ---help---
        This adds the clock driver support for Freescale QorIQ platforms using common clock framework.

config COMMON_CLK_XGENE
    bool "clock driver for APM XGene SoC"
    ---help---

```



Modify the Makefile in **clk** folder accordingly.



```
obj-$(_CONFIG_COMMON_CLK) += clk-multiplier.o
obj-$(_CONFIG_COMMON_CLK) += clk-mux.o
obj-$(_CONFIG_COMMON_CLK) += clk-composite.o
obj-$(_CONFIG_COMMON_CLK) += clk-fractional-divider.o
obj-$(_CONFIG_COMMON_CLK) += clk-gpio.o
ifeq ($(_CONFIG_OF), y)
obj-$(_CONFIG_COMMON_CLK) += clk-conf.o
endif

# hardware specific clock types
# please keep this section sorted lexicographically by file path name
obj-$(_CONFIG_MACH_ASM9260) += clk-asm9260.o
obj-$(_CONFIG_COMMON_CLK_AXI_CLKGEN) += clk_axi_clkgen.o
obj-$(_CONFIG_COMMON_CLK_DGLNT_DYNCLK) += clk-dglnt-dynclk.o
obj-$(_CONFIG_ARCH_AAA7A) += clk-axm10526.o
obj-$(_CONFIG_COMMON_CLK_CDCE706) += clk-cdce706.o
obj-$(_CONFIG_COMMON_CLK_CDCE925) += clk-cdce925.o
obj-$(_CONFIG_ARCH_CLPS711X) += clk-clps711x.o
obj-$(_CONFIG_COMMON_CLK_CS2000_CP) += clk-cs2000-cp.o
obj-$(_CONFIG_ARCH_EFM32) += clk-efm32gg.o
obj-$(_CONFIG_COMMON_CLK_GEMINI) += clk-gemini.o
obj-$(_CONFIG_COMMON_CLK_ASPEED) += clk-aspeed.o
obj-$(_CONFIG_COMMON_CLK_HGBANK) += clk-hgbank.o
obj-$(_CONFIG_CLK_HSDK) += clk-hsdk-pll.o
obj-$(_CONFIG_COMMON_CLK_MAX77686) += clk-max77686.o
obj-$(_CONFIG_COMMON_CLK_MAX9485) += clk-max9485.o
obj-$(_CONFIG_ARCH_MOXART) += clk-moxart.o
obj-$(_CONFIG_ARCH_NOMADIK) += clk-nomadik.o
obj-$(_CONFIG_ARCH_NPCM7XX) += clk-npcm7xx.o
obj-$(_CONFIG_ARCH_NSPIRE) += clk-nspire.o
```

Figure 2-22. Modify the Makefile

To enable the RGB to DVI decoder, users need to put the **diligent_encoder.c** into below folder.

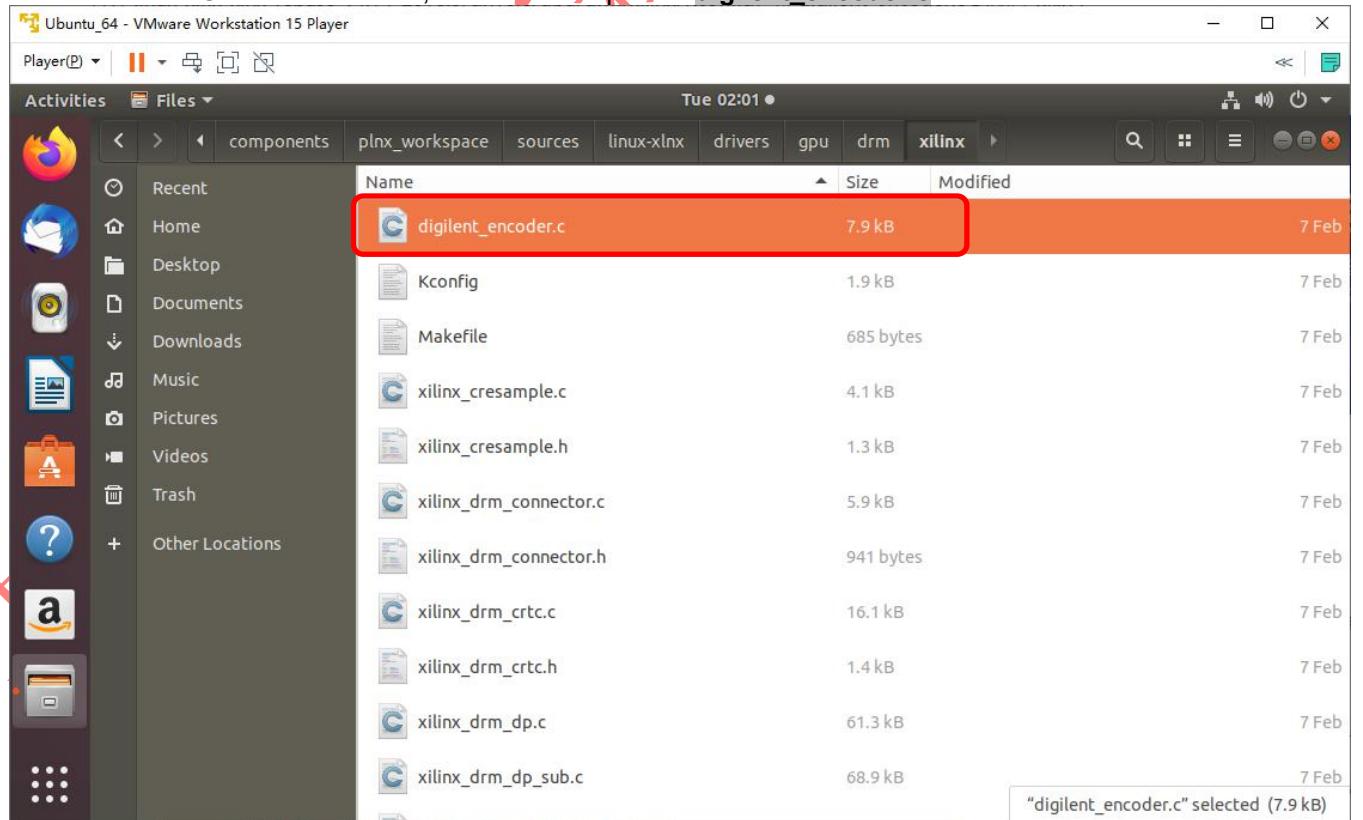
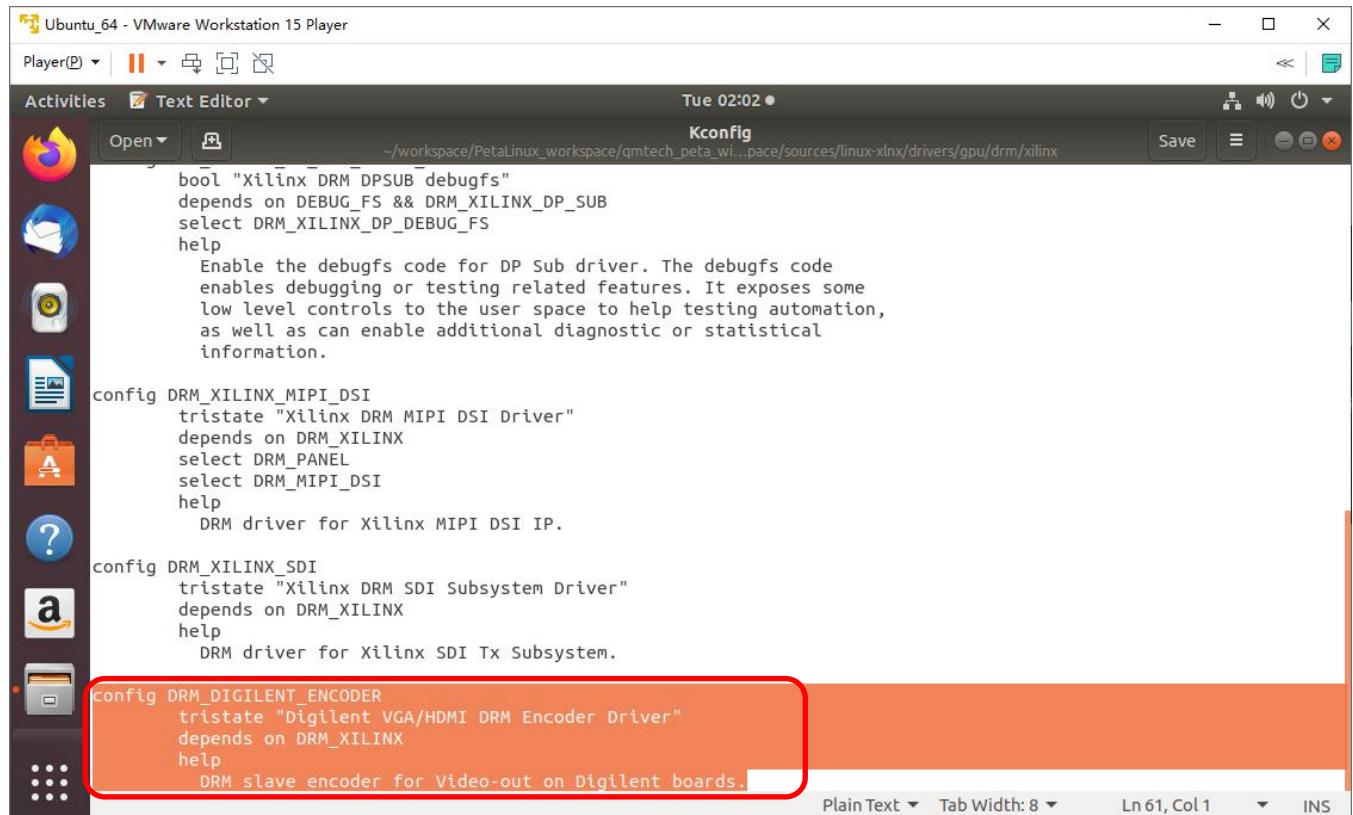


Figure 2-23. Add Driver File

Modify the KConfig file in the **xilinx** folder and add below code line to make the Digilent VGA/HDMI DRM Encoder Driver module enabled.



```
bool "Xilinx DRM DPSUB debugfs"
depends on DEBUG_FS && DRM_XILINX_DP_SUB
select DRM_XILINX_DP_DEBUG_FS
help
    Enable the debugfs code for DP Sub driver. The debugfs code
    enables debugging or testing related features. It exposes some
    low level controls to the user space to help testing automation,
    as well as can enable additional diagnostic or statistical
    information.

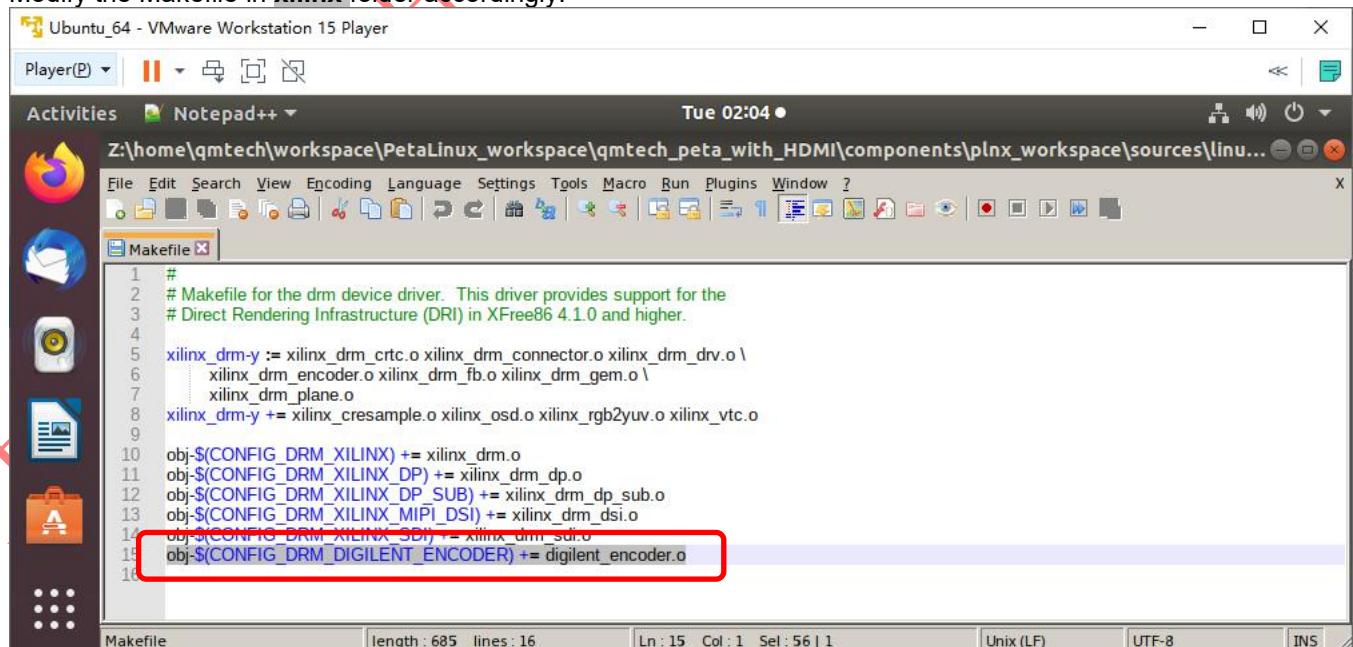
config DRM_XILINX_MIPI_DSI
    tristate "Xilinx DRM MIPI DSI Driver"
    depends on DRM_XILINX
    select DRM_PANEL
    select DRM_MIPI_DSI
    help
        DRM driver for Xilinx MIPI DSI IP.

config DRM_XILINX_SDI
    tristate "Xilinx DRM SDI Subsystem Driver"
    depends on DRM_XILINX
    help
        DRM driver for Xilinx SDI Tx Subsystem.

config DRM_DIGILENT_ENCODER
    tristate "Digilent VGA/HDMI DRM Encoder Driver"
    depends on DRM_XILINX
    help
        DRM slave encoder for Video-out on Digilent boards.
```

Figure 2-24. Modify the Kconfig File

Modify the Makefile in **xilinx** folder accordingly.



```
# Makefile for the drm device driver. This driver provides support for the
# Direct Rendering Infrastructure (DRI) in XFree86 4.1.0 and higher.

xilinx_drm-y := xilinx_drm_crtc.o xilinx_drm_connector.o xilinx_drm_drv.o \
                xilinx_drm_encoder.o xilinx_drm_fb.o xilinx_drm_gem.o \
                xilinx_drm_plane.o
xilinx_drm-y += xilinx_cresample.o xilinx_osd.o xilinx_rgb2yuv.o xilinx_vtc.o

obj-$CONFIG_DRM_XILINX += xilinx_drm.o
obj-$CONFIG_DRM_XILINX_DP += xilinx_drm_dp.o
obj-$CONFIG_DRM_XILINX_DP_SUB += xilinx_drm_dp_sub.o
obj-$CONFIG_DRM_XILINX_MIPI_DSI += xilinx_drm_dsi.o
obj-$CONFIG_DRM_XILINX_SDI += xilinx_drm_sdi.o
obj-$CONFIG_DRM_DIGILENT_ENCODER += digilent_encoder.o
```

Figure 2-25. Modify the Makefile



Users then start to configure the Petalinux kernel by command: **petalinux-config -c kernel**
 This step may take a really long time. Be patient here and make sure the VM's network status is perfect.
Enable Digilent axi_dynclk Driver in Device Drivers → Common Clock Framework.

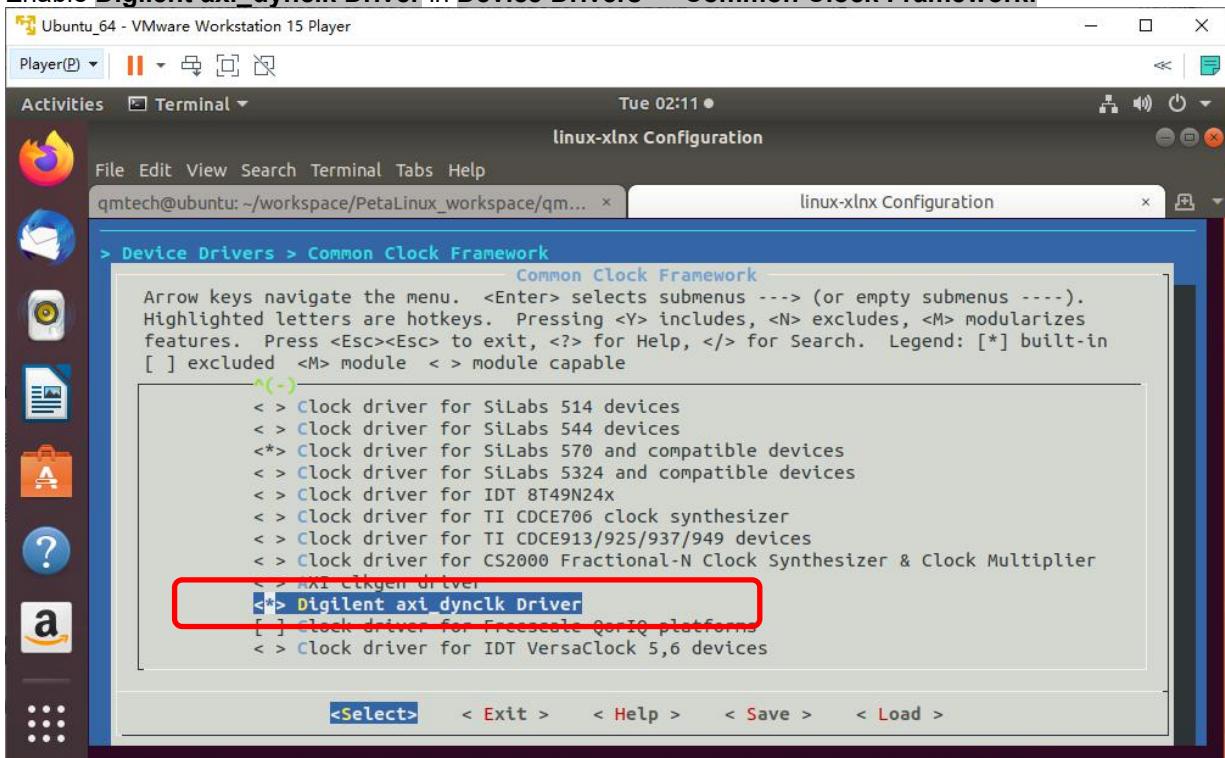
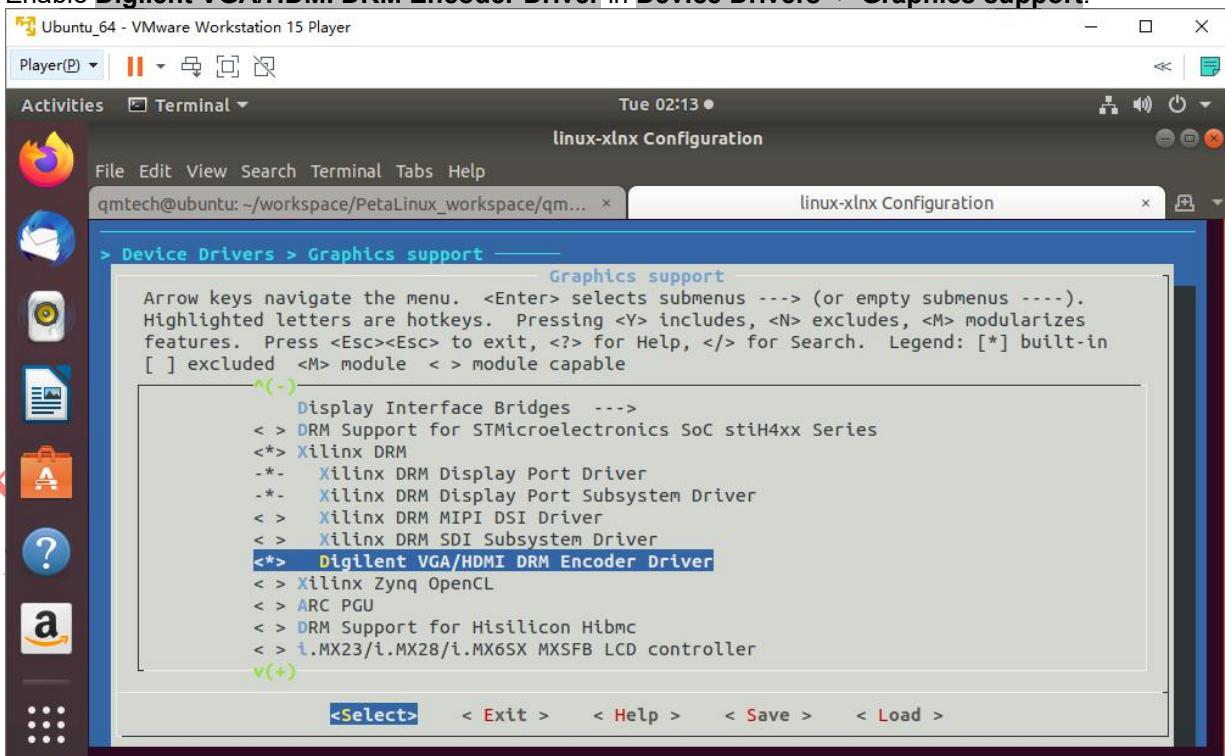
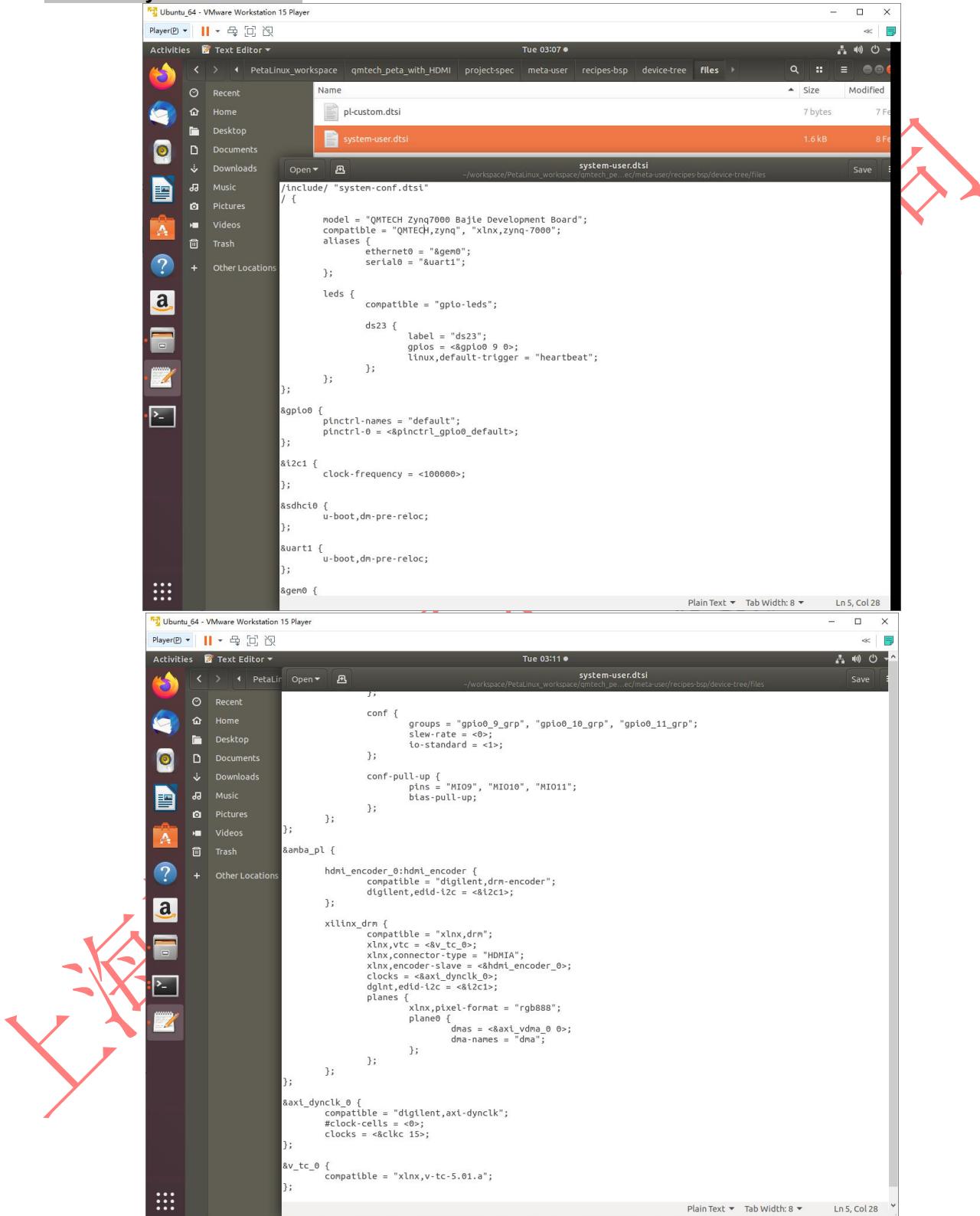


Figure 2-26. Enable Clock Driver

Enable **Digilent VGA/HDMI DRM Encoder Driver** in **Device Drivers -> Graphics support**.



Update the device tree in file **\$PROJECT_DIR/project-spec/meta-user/recipes-bsp/device-tree/files/system-user.dtsi**:



```
Ubuntu_64 - VMware Workstation 15 Player
Activities Text Editor Tue 03:07 •
Recent Name pl-custom.dtsi 7 bytes 7 kB
Desktop system-user.dtsi 1.6 kB 8 kB
Documents
Downloads
Music
Pictures
Videos
Trash
+ Other Locations
Open Save
system-user.dtsi
~/workspace/PetaLinux_workspace/qmtech_peta_with_HDMI/project-spec/meta-user/recipes-bsp/device-tree/files
/include/ "system-conf.dtsl"
{
    model = "QMTECH Zynq7000 Bajie Development Board";
    compatible = "QMTECH,zynq", "xlnx,zynq-7000";
    aliases [
        ethernet0 = "&gen0";
        serial0 = "&uart1";
    ];
    leds {
        compatible = "gpio-leds";
        ds23 {
            label = "ds23";
            gpios = <&gpio0 9 0>;
            linux,default-trigger = "heartbeat";
        };
    };
    8gpio0 {
        pinctrl-names = "default";
        pinctrl-0 = <&pinctrl_gpio0_default>;
    };
    &l2c1 {
        clock-frequency = <1000000>;
    };
    &sdhci0 {
        u-boot,dm-pre-reloc;
    };
    &uart1 {
        u-boot,dm-pre-reloc;
    };
    &gen0 {
        ...
    };
}
Plain Text Tab Width: 8 Ln 5, Col 28
```



```
Ubuntu_64 - VMware Workstation 15 Player
Activities Text Editor Tue 03:11 •
Recent system-user.dtsi
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
+ Other Locations
Open Save
system-user.dtsi
~/workspace/PetaLinux_workspace/qmtech_peta_with_HDMI/project-spec/meta-user/recipes-bsp/device-tree/files
        conf {
            groups = "gpio0_9_grp", "gpio0_10_grp", "gpio0_11_grp";
            slew-rate = <0>;
            io-standard = <1>;
        };
        conf-pull-up {
            pins = "MIO9", "MIO10", "MIO11";
            bias-pull-up;
        };
    };
    &amba_pl {
        hdmi_encoder_0:hdmi_encoder {
            compatible = "digilent,drm-encoder";
            digilent,edid-l2c = <&l2c1>;
        };
        xilinx_drm {
            compatible = "xlnx,drm";
            xlnx,vtc = <&v_tc_0>;
            xlnx,connector-type = "HDMI";
            xlnx,encoder-slave = <&hdmi_encoder_0>;
            clocks = <&xil_dynclk_0>;
            dgtnt,edid-l2c = <&l2c1>;
            planes {
                xlnx,pixel-format = "rgb888";
                plane0 {
                    dmabases = <&xil_vdma_0 0>;
                    dma-names = "dma";
                };
            };
        };
        &xil_dynclk_0 {
            compatible = "digilent,xil-dynclk";
            #clock-cells = <0>;
            clocks = <&clkc 15>;
        };
        &v_tc_0 {
            compatible = "xlnx,v-tc-5.01.a";
        };
    };
}
Plain Text Tab Width: 8 Ln 5, Col 28
```

Figure 2-27. Update Device Tree

Once the PetaLinux Kernel info is successfully configured, users may start to configure the PetaLinux file system by command: **petalinux-config -c rootfs**

Below image will display and nothing needs to be changed here.

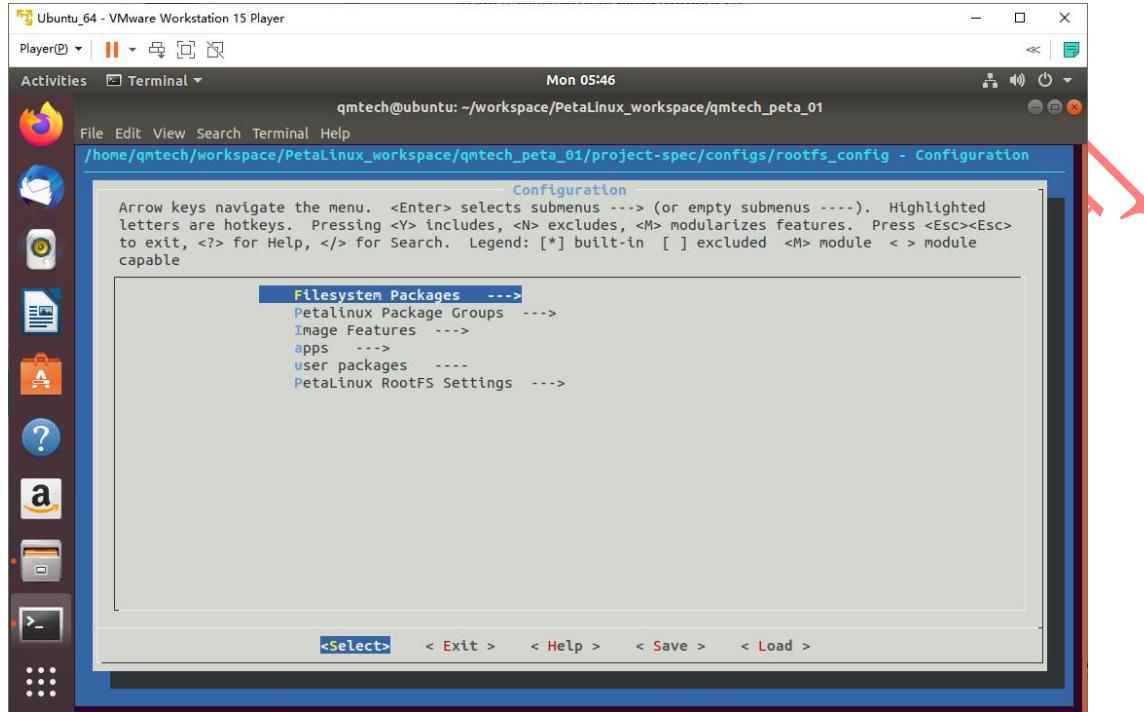


Figure 2-28. No Need to change the rootfs

Start to build the PetaLinux by command: **petalinux-build**

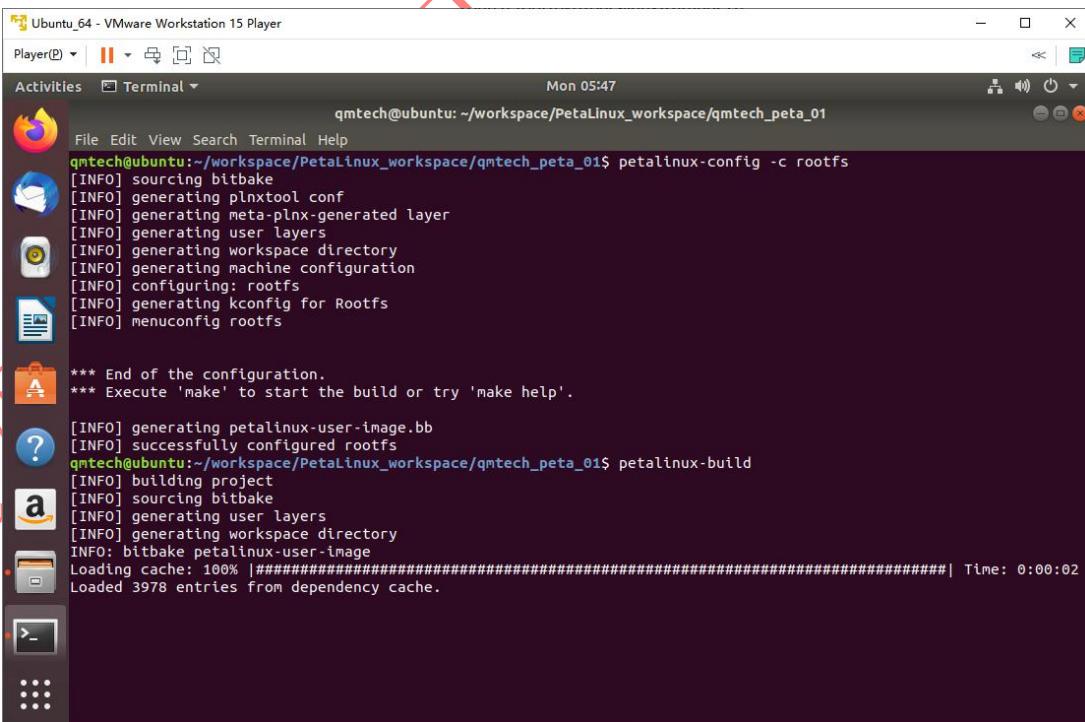


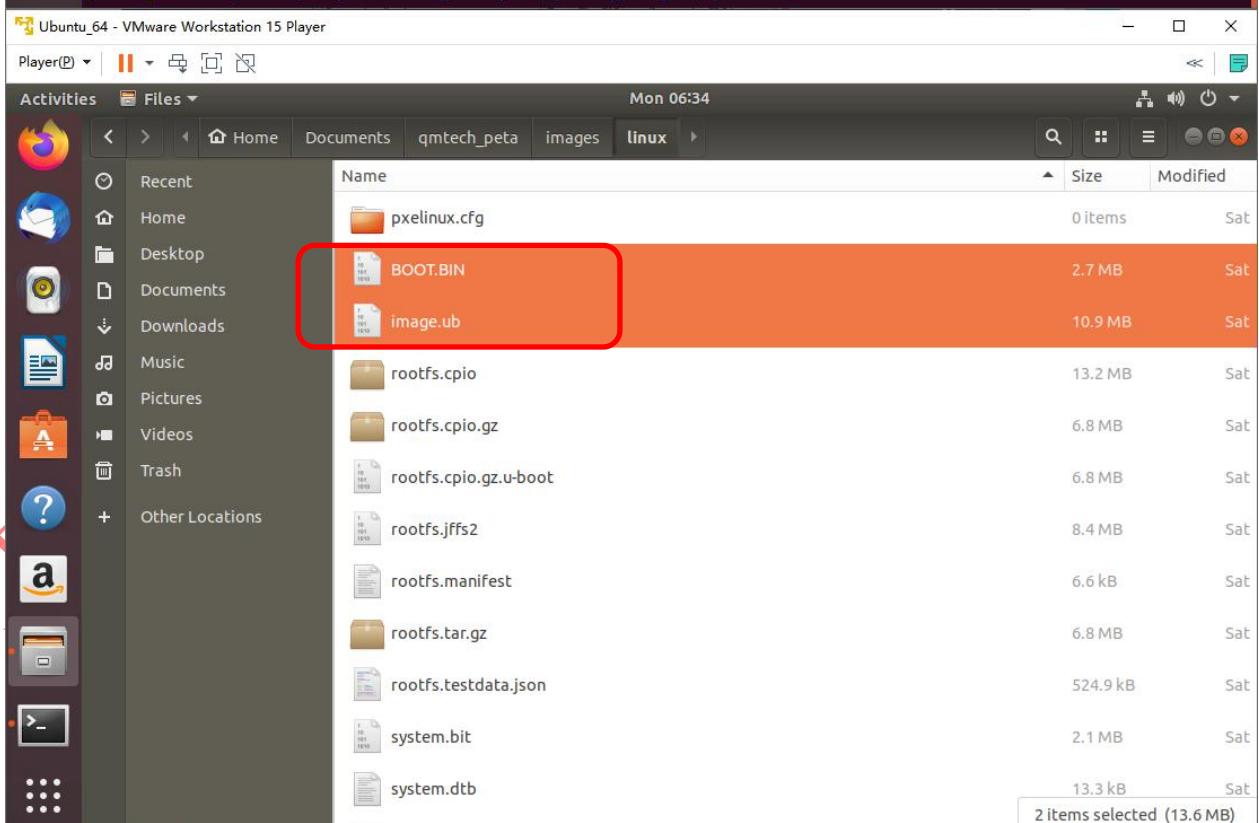
Figure 2-29. Build PetaLinux

After the PetaLinux is successfully built, users could generate the target BOOT.bin and image.ub by command: **petalinux-package --boot --fsbl ./images/linux/zynq_fsbl.elf --fpga --u-boot --force**

```

Ubuntu_64 - VMware Workstation 15 Player
Player(P) | || | ↴ ↵ 🔍
Activities Terminal ▾ Mon 06:34
qmtech@ubuntu: ~/workspace/PetaLinux_workspace/qmtech_peta_01
File Edit View Search Terminal Help
INFO: bitbake petalinux-user-image
Loading cache: 100% [#####################################################################
Loaded 3978 entries from dependency cache.
Parsing recipes: 100% [#####################################################################
Time: 0:00:02
Parsing of 2893 .bb files complete (2890 cached, 3 parsed). 3980 targets, 169 skipped, 0 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies
Initialising tasks: 100% [#####################################################################
Time: 0:00:04
Checking sstate mirror object availability: 100% [#####
Sstate summary: Wanted 125 Found 10 Missed 230 Current 754 (8% match, 86% complete)
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 3214 tasks of which 3147 didn't need to be rerun and all succeeded.
INFO: Copying Images from deploy to images
NOTE: Failed to copy built images to tftp dir: /tftpboot
[INFO] successfully built project
qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_01$ petalinux-package --boot --fsbl ./images/linux/zynq_fsbl.elf --fpga --u-boot --force
INFO: File in BOOT BIN: "/home/qmtech/workspace/PetaLinux_workspace/qmtech_peta_01/images/linux/zynq_fsbl.elf"
INFO: File in BOOT BIN: "/home/qmtech/workspace/PetaLinux_workspace/qmtech_peta_01/project-spec/hw-description/design_1_wrapper.bit"
INFO: File in BOOT BIN: "/home/qmtech/workspace/PetaLinux_workspace/qmtech_peta_01/images/linux/u-boot.elf"
INFO: Generating Zynq binary package BOOT.BIN...
***** Xilinx Bootgen v2019.2
**** Build date : Oct 23 2019-22:59:42
** Copyright 1986-2019 Xilinx, Inc. All Rights Reserved.

INFO: Binary is ready.
WARNING: Unable to access the TFTPBOOT folder /tftpboot!!!
WARNING: Skip file copy to TFTPBOOT folder!!!
qmtech@ubuntu:~/workspace/PetaLinux_workspace/qmtech_peta_01$ 
```



Copy the BOOT.bin and image.ub into MicroSD card and then insert the MicroSD card into ZYNQ7000 Bajie Board. Plug the MiniUSB cable into the Bajie Board and then power on the board with 5V DC power source. Below image shows the serial output from the board.

```

U-Boot 2019.01 (Feb 08 2020 - 09:32:41 +0000) Xilinx Zynq ZC702
CPU: Zynq 7z020
Siilicon: v3.1
Model: QMTECH Zynq7000 Bajie Development Board
DRAM: ECC disabled 512 MiB
MMC: mmc@e0100000: 0
Loading Environment From FAT... *** Warning - bad CRC, using default environment

In:    serial@e0001000
Out:   serial@e0001000
Err:   serial@e0001000
Model: QMTECH Zynq7000 Bajie Development Board
Net:   ZYNQ GEM: e000b000, phyaddr 1, interface rgmii-id
eth0: ethernet@e000b000
U-BOOT for qmtech_peta_with_HDMI

ethernet@e000b000 Waiting For PHY auto negotiation to complete..... TIMEOUT !
Hit any key to stop autoboot: 4 000 3 000 2 000 1 000 0
Device: mmc@e0100000
Manufacturer ID: 3
OEM: 5344
Name: SU026
Bus Speed: 15873016
Mode : SD High Speed (50MHz)
Rd Block Len: 512
SD version 3.0
High Capacity: No
Capacity: 1.8 GiB
Bus Width: 4-bit
Erase Group Size: 512 Bytes
10954408 bytes read in 1453 ms (7.2 MiB/s)
## Loading kernel from FIT Image at 10000000 ...
Using 'conf@system-top.dtb' configuration
Verifying Hash Integrity ... OK
Trying 'kernel@0' kernel subimage
Description: Linux kernel
Type: Kernel Image
Compression: uncompressed
Data Start: 0x100000f4
Data Size: 4122984 Bytes = 3.9 MiB
Architecture: ARM
OS: Linux
Load Address: 0x00000000
Entry Point: 0x00000000
Hash algo: sha1
Hash value: 15e57fb316251305f93123f92a6939902842e77
Verifying Hash Integrity ... sha1+ OK
## Loading ramdisk from FIT Image at 10000000 ...
Using 'conf@system-top.dtb' configuration

```

COM68 OPENED, 115200, 8, NONE, 1, OFF Rx: 31,953 Bytes Tx: 0 Bytes

Figure 2-30. PetaLinux Boot Log



Below info will be displayed on the HDMI monitor that the Bajie Board connected.

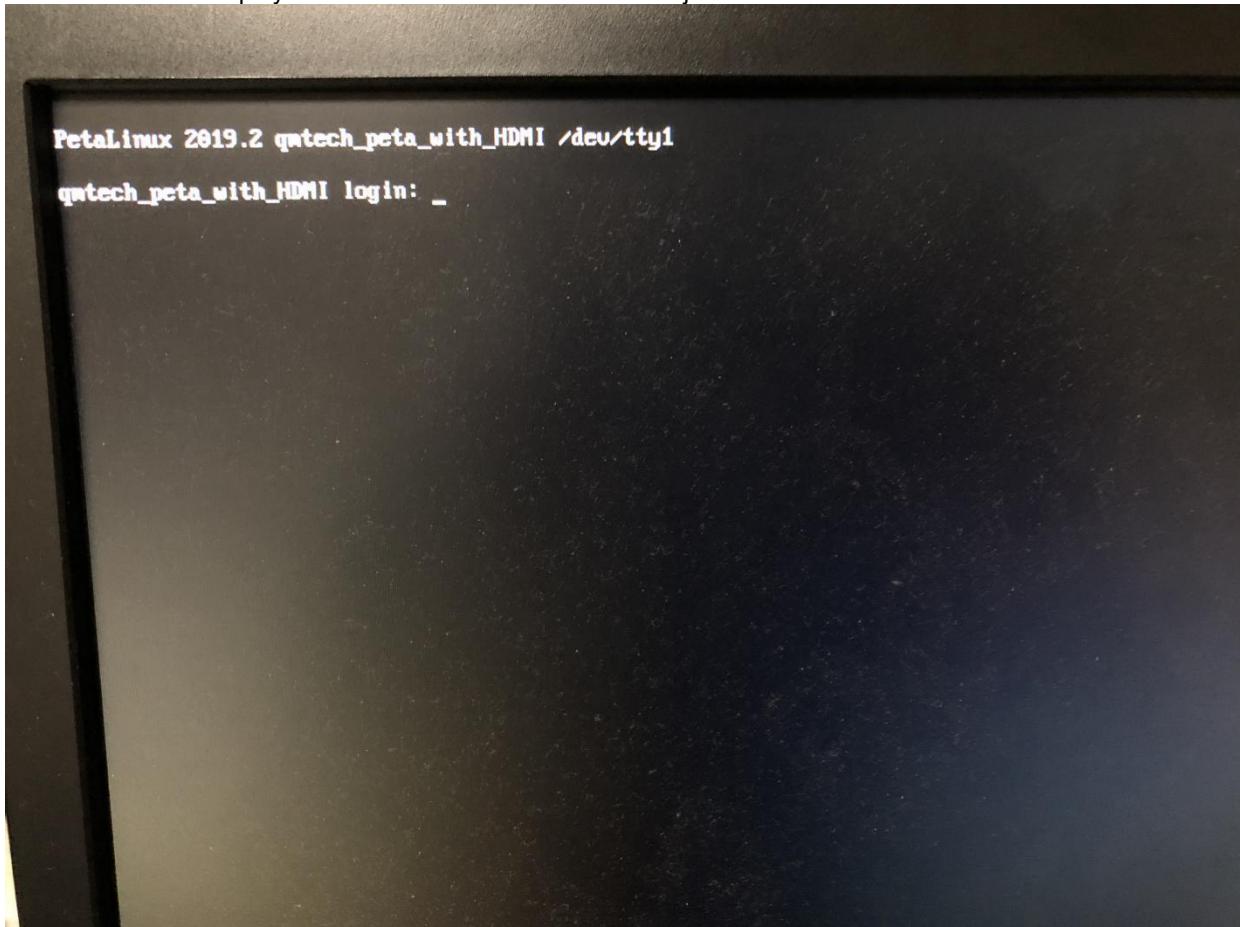


Figure 2-31. HDMI Display

上海勤謀



QMTECH

QMTECH ZYNQ7000 Bajie Board

User Manual(Linux) V01

3. Reference

- [1] ug585-Zynq-7000-TRM.pdf
- [2] ds187-XC7Z010-XC7Z020-Data-Sheet.pdf
- [3] ug865-Zynq-7000-Pkg-Pinout.pdf
- [4] MT41K256M16TW-107:P.pdf
- [5] tps563201.pdf
- [6] RTL8211E-VL.PDF

上海勤謀電子科技有限公司



4. Revision

Doc. Rev.	Date	Comments
0.1	03/03/2020	Initial Version.
1.0	03/21/2020	V1.0 Formal Release.

上海勤謀電子科技有限公司



QMTECH

QMTECH ZYNQ7000 Bajie Board

User Manual(Linux) V01