آرتمیس کاریا
www.artemiskaria.ir
آرتمیس کاریا (آلینکس) - فناوری FPGA

AK-SDR P201/3 Series User Manual Openwifi Porting Tutorial

Version Records:

Data	Version	Description
2023.2.16	V1.0	initial version

This tutorial will continue to revise, optimize and increase based on the actual Experience, that is to provide you with more and better Demos.

If you find some errors or any suggestion, contact with us.

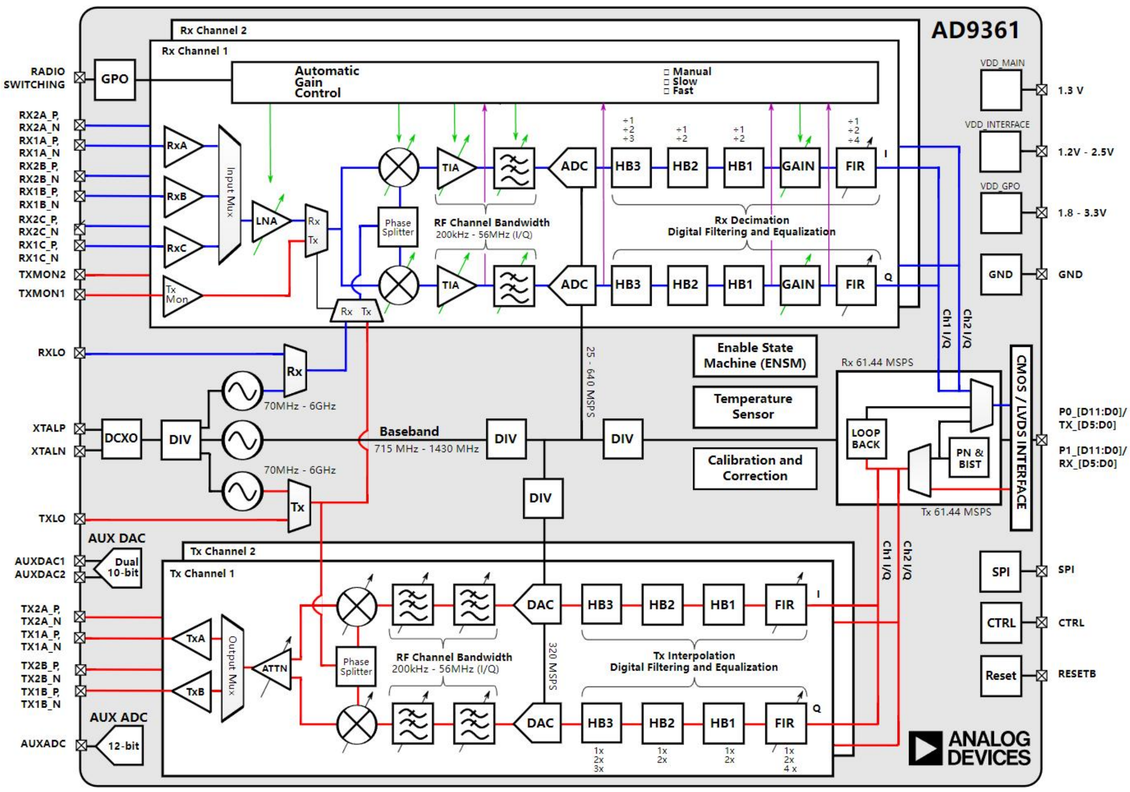
Content

Part 1: ADI-AD936X Series Introduction

The AD9361 , AD9364 and AD9363 are high-performance, highly integrated RF Agile Transceivers™. Their programmability and broadband capability make them ideal for a wide range of transceiver applications. Combining an RF front-end with a flexible mixed-signal baseband section and integrated frequency synthesizer, these devices simplify design by providing a configurable digital interface to the processor.

The AD9361 and AD9364 both operate in the 70 MHz to 6.0 GHz range, while the AD9363 operates in the 380 MHz to 3.8 GHz range, covering most licensed and unlicensed bands.

The AD9361 and AD9364 have channel bandwidths from 200 kHz to 56 MHz, while the AD9363 has a frequency range of 200 kHz to 20 MHz. Both the AD9361 and AD9363 are 2 Rx, 2 Tx devices, while the AD9364 is a 1 Rx, 1 Tx device.



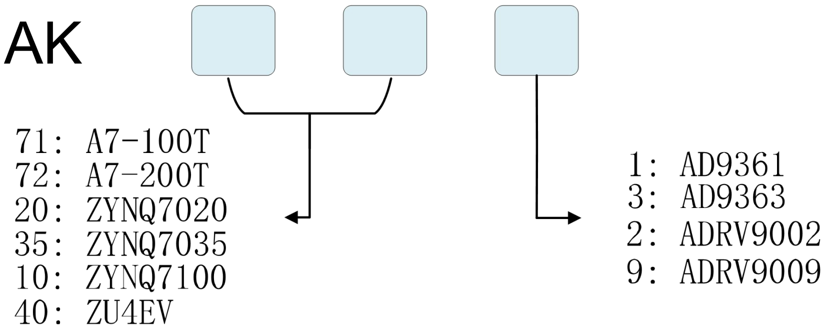
The AD9361 and AD9364 are highly integrated radio frequency (RF) transceivers that can be configured for a wide range of applications. These devices integrate all the RF, mixed-signal, and digital modules needed to provide all transceiver functionality in a single device. Programmability makes this broadband transceiver suitable for a wide range of communication standards, including frequency division duplex (FDD) and time division duplex (TDD) systems. This programmability also allows the device to connect to a variety of baseband processors (BBPs) using a single 12-bit parallel data port, dual 12-bit parallel data ports, or a 12-bit low-voltage differential signaling (LVDS) interface.

The AD9361 and AD9364 also offer a self-calibration and automatic gain control (AGC) system that maintains high performance levels under changing temperature and input signal conditions. In addition, the devices include multiple test modes that allow system designers to insert test signals and create internal loopback patterns that designers can use to debug their designs during prototyping and optimize their radio configurations for specific applications.

Part 2: AK201 and AK203 of AKSDR

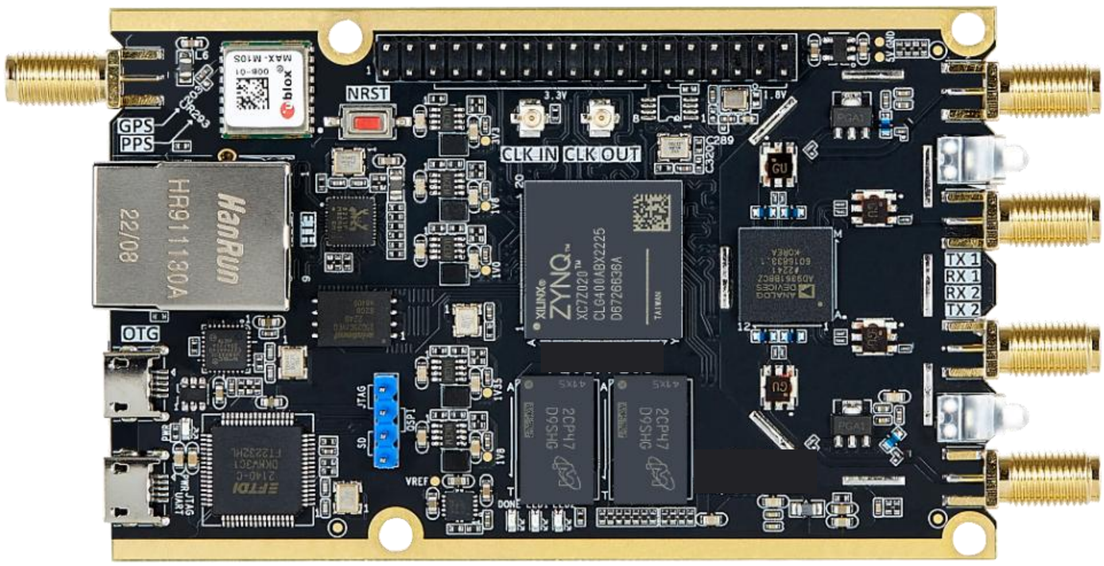
Part 2.1: AKSDR Series Products Introduction

The AKSDR series includes a number of RF front ends products based on xilinx FPGA chip and ADI's AD9363/AD9361/ADVR9002/ADVR9009. For example, the FPGA Chips are Artix7's A7-100T and A7-200T, ZYNQ7000 series 7020 and 7100, and ZYNQ UltraScale series 4EV, etc.There is a wide range of products, and to facilitate the selection process, we have defined the model specifications for the entire product family as shown in the following diagram, which details the differences between the models. This document is based on the P201, i.e. ZYNQ7020+AD9361 architecture.



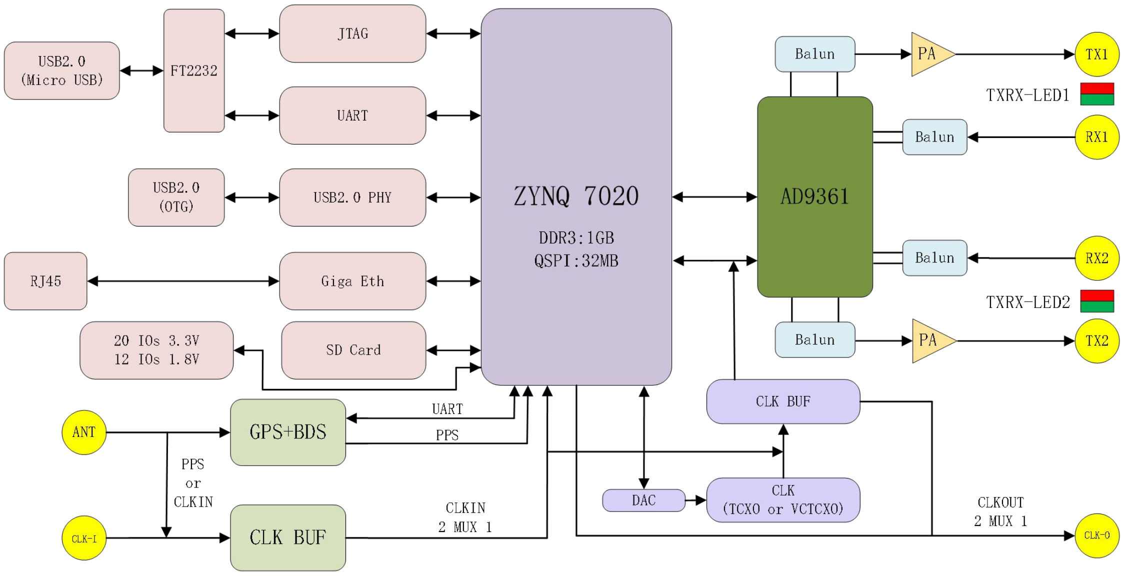
AKSDR Module Definition

The AK201 and AK203 platforms are upgraded on the basis of ADI official pluto, the RF port adopts 2T2R mode, which is more powerful, and on this basis, it also integrates GPS, clock input and output and many other functions.



Part 2.2: Product Function Block Diagram

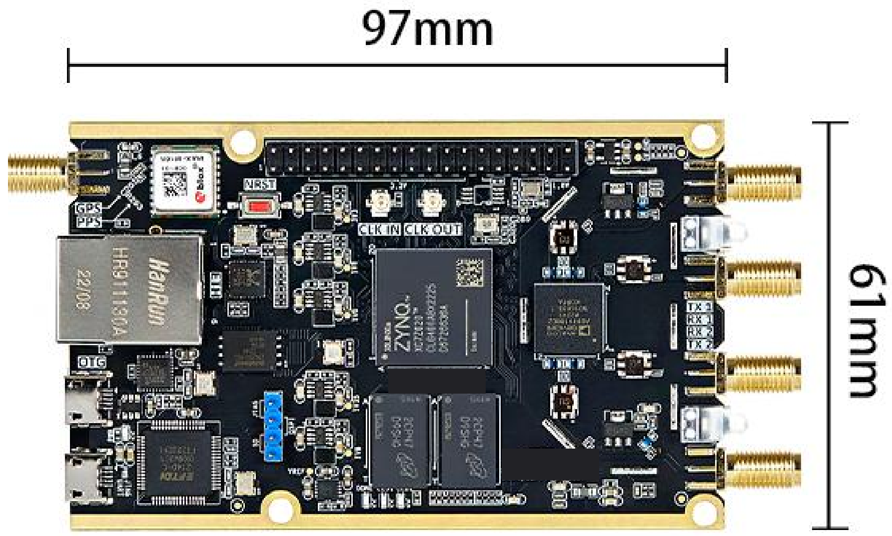
In this section, we will show the configuration details of the product in detail through the product block diagram and product specifications, as shown below.

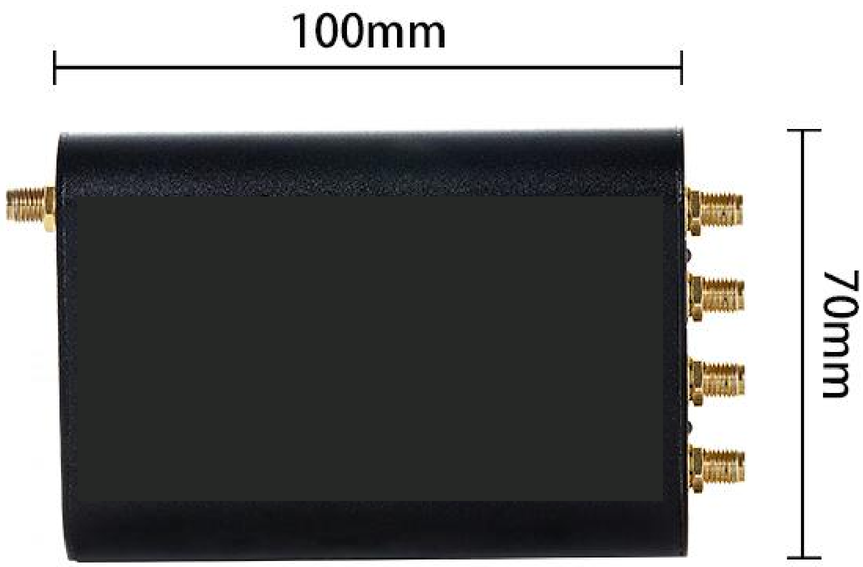


	AK201	AK203
RF Chip	AD9361	AD9363
Frequency range	70M-6Ghz	325M-3.8Ghz
Signal Bandwidths	200K-56Mhz	200K-20Mhz
Power Amplifier	2 way TX	
FPGA Chip	XC7Z020-2CLG400I	
Processor Core	Dual Core Cortex-A9, 766Mhz	
Logic Cells	85K	
DDR3	1GB	
QSPI FLASH	32MB	
RF Clock	Default TCXO/0.5PPM, VCTCXO is adjustable	
Clock Input/Output	1 input, 1 output	
Gigabit Ethernet	1	
USB OTG	1	
UART	1	
JTAG	1	
SD Card Slot	1	
GPS/PPS	Default GPS, PPS is adjustable	
Extended IO	20 x 3.3V IO, 12 x 1.8V IO	
Form Factors	Board: 97 x 61mm Shell: 100 x 70mm	
Power supply	5V/1A (USB powered)	

Part 2.3: Form Factors

Bare Board form Form Factors is 97mm * 61mm, and the shell box form factor is 100mm * 70mm



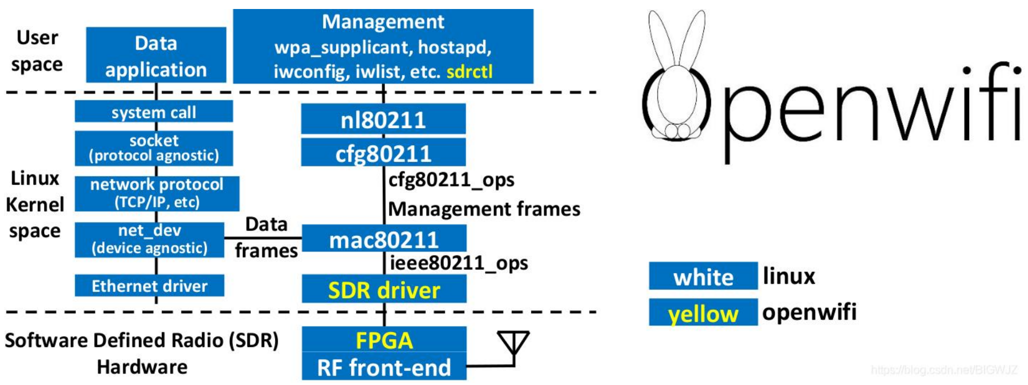


Part 3: Openwifi Porting

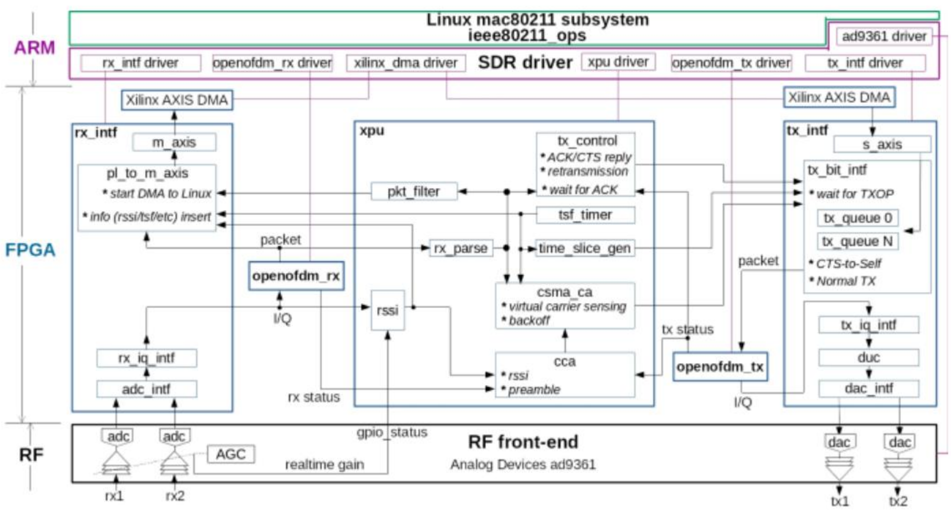
Openwifi porting is done under Ubuntu system, users need to prepare the Ubuntu system host or virtual machine (the author's Ubuntu version is 18.04). At the same time, you need to install Vivado version 2021.1, we provide the installation package for Vivado software, but the Ubuntu system is larger and you need to download it by yourself.

Part 3.1: Openwifi Introduction

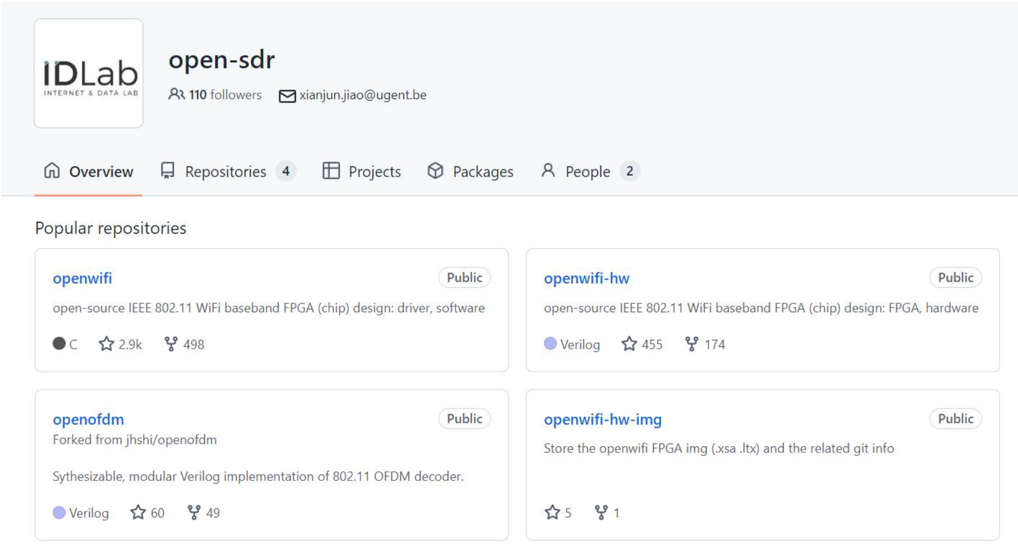
Openwifi is an open source project about wifi system, is a rare and excellent open source project about wifi, the project includes wifi baseband, lowmac, linux driver and other three parts, where the baseband, lowmac part is implemented in FPGA, wifi driver part is running under Linux, so openwifi system is a complete wireless network card running under Linux implementation version , so for those who wish to learn the physical layer implementation part of the protocol of wifi or want to understand the operation of the wireless network card function , has a very high reference value .



Openwifi Overall functional framework diagram:



This block diagram of Openwifi's functionality has clearly illustrated the various functions. Openwifi is a wifi project running under Linux, its function is the whole function of a wireless network card. The Openwif project is to show us how to build a wireless network card and provide a driver for the network card under linux, able to run under Linux as a commercial wireless network card, providing wifi access or link service. Also, let's talk a bit more about the composition of the openwifi project on github:



openwifi: This is the source code for the driver part of Openwifi with Linux and the linux system.

openwifi-hw: This is the source code for the FPGA part of Openwifi, which is the hardware part and also the lowmac part.

openofd m: This is the source code for the baseband part of Openwifi, which also runs in the FPGA and is eventually integrated into the openwif-hw project, which is also considered part of openwif-hw, and is easier to understand and learn here as a separate project.

We only need to use openwifi and openwifi-hw for porting.

Part 3.2: Openwifi Experiment Introduction

Since the file provided by ADI-Pluto-github does not have a file for our FPGA development board (AKSDR Series Board), we modified and added it based on the file provided by Openwifi file.

We need to compile the Openwifi file based on the modified Openwifi file to make it compatible with our board's firmware and able to run online.

You can check the source files we provide and github to learn.

Github: https://github.com/open-sdr

Part 3.3: Openwifi Porting Guideline

If you are using our source files, please refer to our step-by-step guide.

You can also refer to the steps on the official website, but the source files on the official website may not include the files for our FPGA development board (AKSDR Series Board), so you can add them yourself.

The official website link is as follows

https://github.com/open-sdr/openwifi

Once the burn is complete, the system will recognise the BOOT and rootfs folders and mount them on the Ubuntu system.

We have added the files for the user to adapt to our FPGA development board (AKSDR Series Board), so the user can go to the openwifi directory, execute the following command, and wait for it to finish.

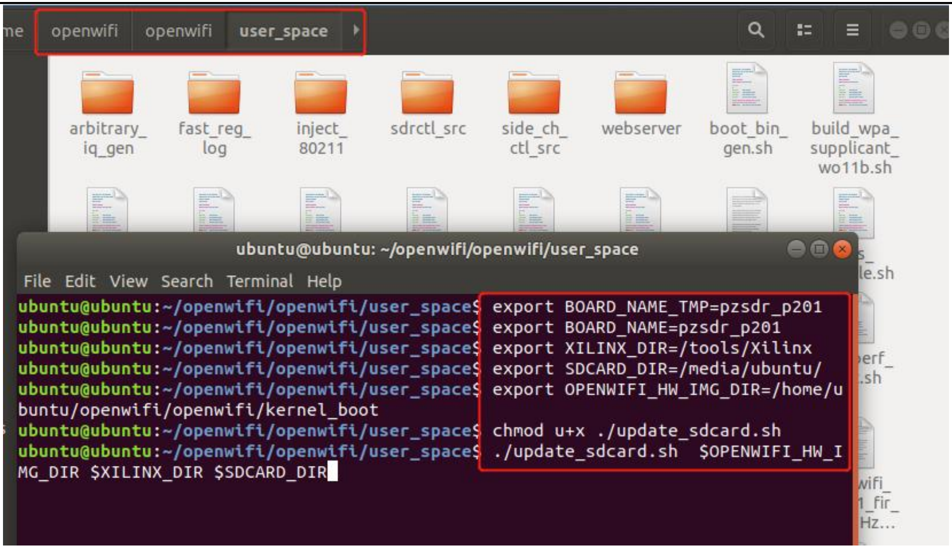
Note: The compilation process needs to be connected to the Internet, and the user needs to be able to access the Google website under Ubuntu, otherwise the compilation may go wrong.

The command is as follows (user path: user to modify according to their actual path):

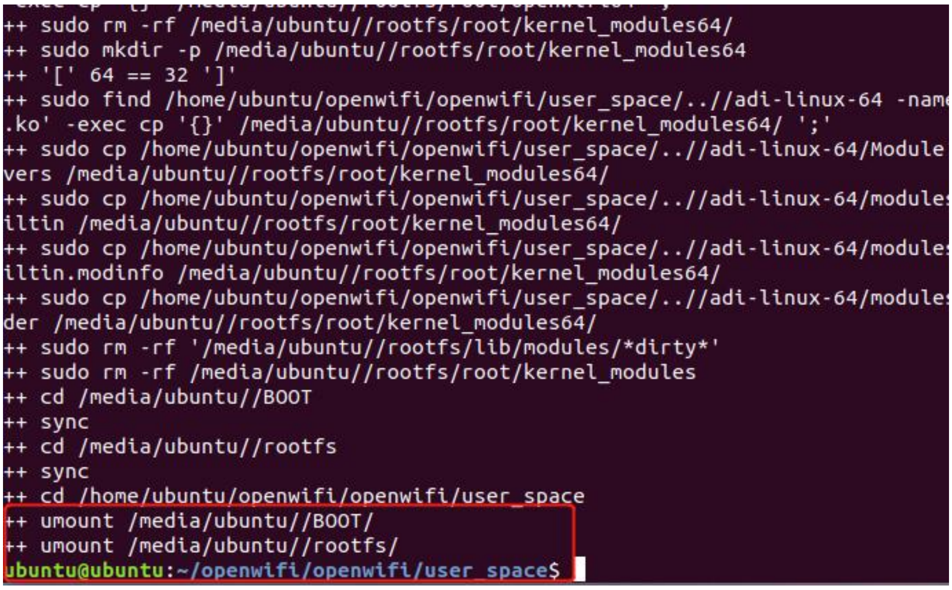
```
export BOARD_NAME_TMP=pzsdr_p201
export BOARD_NAME=pzsdr_p201
export XILINX_DIR=/tools/Xilinx  (Vivado installation path)
export SDCARD_DIR=/media/ubuntu/
export OPENWIFI_HW_IMG_DIR=/user path/openwifi/kernel_boot
chmod u+x ./update_sdcard.sh
./update_sdcard.sh  $OPENWIFI_HW_IMG_DIR $XILINX_DIR $SDCARD_DIR
```

Note: If you modify the Vivado project and recompile it with xsa, you need to copy it to the following path

Path: /openwifi/kernel_boot/boards/pzsdr_p201



The author runs the command as follows:



The command is finally unmounted, but just in case we mount the BOOT partition again, copy all the files under /BOOT/openwifi/pzsdr_p201 to BOOT, then eject the SD card, plug it into the development board and power it up and run it.

Execute the following command in the serial terminal and wait for the openwifi signal:

/root/openwifi/setup_once.sh (Only need to run once for new board)

cd /root/openwifi

./wgd.sh

ifconfig sdr0 up

iwlist sdr0 scan

./fosdem.sh

Currently openwifi is not networked, the following commands need to be executed for networking, and you need to connect to a device with internet access via a network cable (it is recommended to use a router and change the network segment to 192.168.10.1):

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo iptables -t nat -A POSTROUTING -o NICY -j MASQUERADE
sudo ip route add 192.168.13.0/24 via 192.168.10.122 dev ethX
```

ethX是连接开发板以太网的 PC NIC 名称。**NICY**是连接互联网（WiFi 或其他以太网）的 PC NIC 名称。

sudo eth0 192.168.10.1

sudo sysctl -w net.ipv4.ip_forward=1

sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE

sudo ip route add 192.168.13.0/24 via 192.168.10.122 dev eth0

At this point, the Openwifi experiment is complete.