

Advantech AE Technical Sharing Documentation

| | | | |
|------------------------|---|-------------------|------------------------------|
| Date | 2019 / 3 / 27 | SR# | 1-3774989526 |
| Category | ■FAQ □ SOP | Related OS | Linux version 4.10 and above |
| Abstract | How to install AD5593R driver and test the function under AdvLinux 1.0.8? | | |
| Keyword | AdvLinux 1.0.8 , AD5593R, | | |
| Related Product | UNO-420 | | |

■ Problem Description:

The UNO-420 uses the AD5593R chip. How do you drive the AD5593R die and verify functionality after installing the operating system?



advad5593r_source_
v1.00.tar.gz



advad5593r_source_
v1.00.tar.gz.md5

Ps: Remember to add the load when you are testing.

■ Brief Solution - Step by Step:

(一) 、How to install the driver:

Fig1. Directory Construction:

The directory construction is:

```
+advad5593r_source_v1.00
|
+--drivers                      // directory of the device driver
|   |
|   +--ad5593r                  // ad5593r driver, depends on I2C & SMBUS
|   |
|   +--Makefile                 // Makefile for install the driver
|
+--example                      // directory of the demo example
|   |
|   +--Makefile                 // Makefile for building the examples
|   |
|   +--test_iio_ad5593r.c       // example for ad5593r iio port
|   |
|   +--README                   // README file for build and how to use the examples
|
+--README                       // README file for driver installation
|
+--changelog.txt                // changelog file for driver installation
```

1.) Uncompressed the package:

- Executing the following instructions and uncompressed the package.
tar -zxvf advad5593r_source_v1.00.tar.gz
cd advad5593r_source_v1.00

2.) Install the driver:

- Go to the driver directory:
cd driver
make

make install

3.) After “make install” the source code, you will see the ad5593r driver is ready.

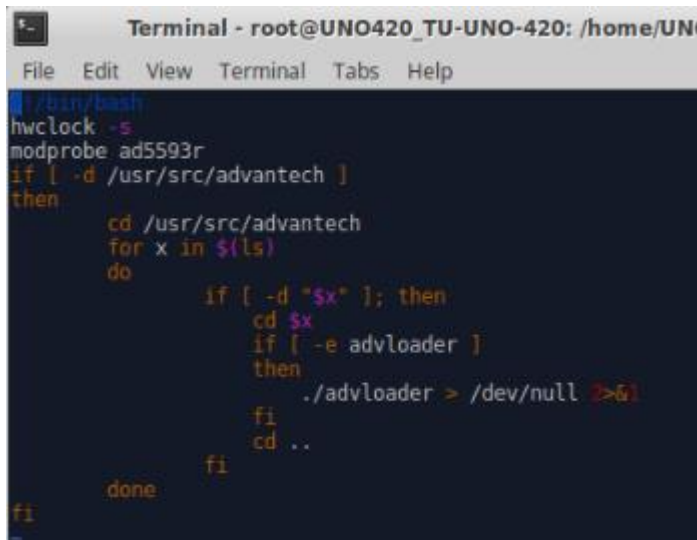
And now we need to add the module into Linux Kernel

modprobe ad5593r

By this way to add module will unmount when you restart the system. If you want to auto mount the driver please go to next step.

4.) We need to edit the rc.local file by #vi /etc/rc.local

Please add “modprobe ad5593r” in the /etc/rc.local file by vi application.



```

Terminal - root@UNO420_TU-UNO-420: /home/UNC
File Edit View Terminal Tabs Help
~/bin/bash
hwclock -s
modprobe ad5593r
if [ -d /usr/src/advantech ]
then
    cd /usr/src/advantech
    for x in $(ls)
    do
        if [ -d "$x" ]; then
            cd $x
            if [ -e advloader ]
            then
                ./advloader > /dev/null &&
            fi
            cd ..
        fi
    done
fi

```

Then save and reboot system.

In case, you can check by #cat /proc/cmdline

(二) 、How to set the I/O mode:

We must set the I/O mode before we test it.

In this module (AD5593R) mode or function of all channels (channel0 ~ channel7) are currently supported and you can set the following modes to every pin:

- CH_MODE_UNUSED (the pin is unused) value is 0
- CH_MODE_ADC (the pin is ADC input) value is 1
- CH_MODE_DAC (the pin is DAC output) value is 2
- CH_MODE_DAC_AND_ADC

(The pin is DAC output but can be monitored by an ADC; since there is no disadvantage this this should be considered as the preferred DAC mode) value is 3

- CH_MODE_GPIO (the pin is registered with GPIOLIB) value is 8

1) Setting operation modes.

Step1 >> Edit file by #vi /etc/default/grub

Modify

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash overlayroot=disabled"
to
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash overlayroot=disabled ad5593r.cha_mode=88001233"
```

And you can set the mode of the I/O by the ad5593r.cha_mode value.

Parameter "88001233" means that:

- channel[0] and channel[1] works in CH_MODE_DAC_AND_ADC mode
- channel[2] works in CH_MODE_DAC mode
- channel[3] works in CH_MODE_ADC mode
- channel[4] and channel[5] works in CH_MODE_UNUSED mode
- channel[6] and channel[7] works in CH_MODE_GPIO mode

Please accord to your application to set your parameter value.

2) Step2 >> update grub file.

```
#sudo update-grub
```

```
#sync
```

3) Step3 >> reboot and insmod ad5593r driver.

```
#reboot
```

```
#modprobe ad5593r
```

4) Also, you can choose to set config dynamically when the driver is loaded by "insmod":

```
insmod ad5593r.ko cha_mode="88001233"
```

The driver use cha_mode="88888888" by default.

(三) 、 How to test the AD5593R in DAC, ADC, GPIO mode?

Before test please make sure the ad5593r driver and iio tools should be installed.

To install iio tools, run `# sudo apt-get install libiio*`

✧ The DAC & ADC test way:

1) Show device name

```
# cd /sys/bus/iio/devices/iio\:device0
```

```
# cat name
```

2) Calculate DAC Scales:

The output range of the DAC is 0 V to VREF or 0 V to 2 × VREF. out_voltage_scale is used to select the high or low range. Without external reference the driver selects the internal 2500mV VREF reference. So the scale computes as $2500\text{mV}/2^{12} = 0.610351562$ or $5000\text{mV}/2^{12} = 1.220703124$.

3) Calculate ADC Scales:

The input range of the ADC is 0 V to VREF or 0 V to 2 × VREF. in_voltage_scale is used to select the high or low range. Without external reference the driver selects the internal 2500mV VREF reference. So the scale computes as $2500\text{mV}/2^{12} = 0.610351562$ or $5000\text{mV}/2^{12} = 1.220703124$.

4) Show current scale

Scale to be applied to out_voltageY_raw/in_voltageY_raw in order to obtain the measured voltage in millivolts.

```
# cat out_voltage_scale
```

5) Show available scales

Scale to be applied to out_voltageY_raw/in_voltageY_raw in order to obtain the measured voltage in millivolts.

```
# cat in_voltage_scale_available
```

6) Change scales

Scale to be applied to out_voltageY_raw/in_voltageY_raw in order to obtain the measured voltage in millivolts.

```
# echo 1.220703124 > in_voltage_scale
```

```
# cat in_voltage_scale
```

7) Get channel Y input voltage

Raw (unscaled, no bias etc.) output voltage for channel Y.

$U = \text{out_voltage0_raw} * \text{out_voltage_scale} = 1638 * 0.610351562 = 999.76 \text{ mV}$

```
# echo 1638 > out_voltage0_raw
```

* The GPIO test way:

8) GPIO Testing

The GPIO sysfs interface allows users to manipulate any GPIO from user space. User space utilizes a sysfs control interface to dynamically request and release individual GPIOs. Once a GPIO has been requested, writing to the newly created path allows you to control the direction and the data while reading from it returns the GPIO data (which usually corresponds to a 0 or 1 which represents the signal level).

```
# cd /sys/class/gpio/
```

```
# echo 504 > export (The number 504 is according to Fig2)
```

-You can export 504 to 511.

```
# cd gpio504/
```

```
# cat direction
```

```
# cat value
```

```
# echo low > direction
```

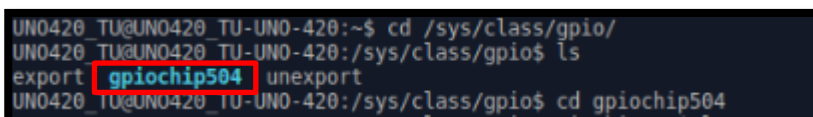
```
# cat direction
```

```
# cat value
```

```
# echo high > direction
```

```
# cat value
```

Fig2. Gpiochip504



```
UN0420_TU@UN0420_TU-UN0-420:~$ cd /sys/class/gpio/
UN0420_TU@UN0420_TU-UN0-420:/sys/class/gpio$ ls
export gpiochip504 unexport
UN0420_TU@UN0420_TU-UN0-420:/sys/class/gpio$ cd gpiochip504
```