UNO-220

Software User Manual

V1.2

Revision History

Date	Version	Description	Author
2021/2/9	V1.2	Add UNO-220-P4N2AE support.	Yuchun.Chen
2020/6/10	V1.1	Initial draft.	Gino.Chiu

Contents

UNO-220 AdvRaspbian Image Feature List

- Support Advantech UNO-220 IO Board.

- RTC-RX8010
- TI TCA9554 IO extender
- Serial to RS-232/485
- Infineon TPM SLx9670
- CPU temperature monitor sample code (C & Python) and C binary testing tool.

- SSH server enabled

- Based on 2021-01-11-raspios-buster-armhf-full.img and 2021-01-11-raspios-buster-armhf.img from Raspberry official website

Write AdvRaspbian image to SD card

Prerequisite

1. AdvRaspbian Image
2. Micro SD card (Recommended 8GB or more)
3. Host PC (Recommended ubuntu 16.04 x86_64 or newer)

Write image to Micro SD card

1. **Open terminal and type "sudo fdisk –l " (Micro SD card must be inserted)**
   - Check Micro SD device name

```
Disk /dev/sde: 7.5 GiB, 8039432192 bytes, 15702016 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

2. **Type "sudo dd if={image file path} of={Micro SD device name} bs=4M status=progress conv=fsync"**
   - Start write image to device

```
root@gino-VirtualBox:/home/gino/Desktop/220# dd if=2019-09-26-raspbian-buster-fu
ll.img of=/dev/sde bs=4M status=progress conv=fsync
6811549696 bytes (6.8 GB, 6.3 GiB) copied, 1632 s, 4.2 MB/s
1624+0 records in
1624+0 records out
6811549696 bytes (6.8 GB, 6.3 GiB) copied, 1772.97 s, 3.8 MB/s
```

3. **Type "sudo fdisk –l {Micro SD device name}"**
   - Check disk partitions
4. **If the disk partitions are created correctly, eject the Micro SD card from the host PC and plug in to UNO220. Now you can use AdvRaspbian OS on UNO220.**

AdvRaspbian File List

```
├──── define.mk                                    # defined function
├──── macro.mk                                     # variables
├──── Makefile                                     # Makefile
├──── mountimage.sh
├──── writesdimage.sh
└──── files
    ├──── boot                                     # boot backup files
    │    ├──── cmdline.txt
    │    └──── config.txt
    └──── root                                     # root files for uno-220
        ├──── etc
        │    ├──── modules-load.d
        │    │    ├──── gpio-i2c.conf
        │    │    └──── rtc-i2c.conf
        │    ├──── udev
        │    │    └──── rules.d
        │    │         ├──── 50-i2c_gpio.rules
        │    │         └──── 50-i2c_rtc.rules
        │    └──── version
        └──── usr
             └──── sbin
                  ├──── uno220gpio
                  ├──── uno220uart
                  ├──── uno220uartrecv
                  └──── uno220uartsend
```

How to test UNO-220

RTC

Get RTC time

pi@raspberrypi:~ $ sudo hwclock -r

```
pi@raspberrypi:~ $
pi@raspberrypi:~ $ sudo hwclock -r
2020-05-04 03:42:40.829182+01:00
pi@raspberrypi:~ $
pi@raspberrypi:~ $
```

Set RTC by system time

pi@raspberrypi:~ $ sudo hwclock -w

```
pi@raspberrypi:~ $ sudo hwclock -w
pi@raspberrypi:~ $ sudo hwclock -r
2020-05-04 03:42:51.330649+01:00
pi@raspberrypi:~ $
```

GPIO

Show usage

pi@raspberrypi:~ $ sudo uno220gpio -h

```
pi@raspberrypi:~ $ sudo uno220gpio -h
  Usage:
    uno220gpio --export=[all|0~7]                 # Export GPIO
    uno220gpio --unexport=[all|0~7]               # Unexport GPIO
    uno220gpio --pin=[0~7] --direction=[in|out]   # Set GPIO Direction
    uno220gpio --pin=[0~7]                        # GPIO Read Operation
    uno220gpio --pin=[0~7] --value=[0|1]          # GPIO Write Operation
    uno220gpio --status
pi@raspberrypi:~ $
```

Get all GPIO Status

pi@raspberrypi:~ $ sudo uno220gpio

```
pi@raspberrypi:~ $ sudo uno220gpio
 pin       |   0   1   2   3   4   5   6   7
-----------------------------------------------
 export    |   0   0   0   0   0   0   0   0
 direction |   X   X   X   X   X   X   X   X
 value     |   X   X   X   X   X   X   X   X
pi@raspberrypi:~ $
```

Export all

pi@raspberrypi:~ $ sudo uno220gpio --export=all

pi@raspberrypi:~ $ sudo uno220gpio

```
pi@raspberrypi:~ $
pi@raspberrypi:~ $ sudo uno220gpio --export=all
pi@raspberrypi:~ $ sudo uno220gpio
 pin       |   0   1   2   3   4   5   6   7
-----------------------------------------------
 export    |   1   1   1   1   1   1   1   1
 direction |   I   I   I   I   I   I   I   I
 value     |   1   1   1   1   1   1   1   1
pi@raspberrypi:~ $
```

Set direction (ex: pin=0, direction=out)

pi@raspberrypi:~ $ sudo uno220gpio --pin=0 --direction=out

pi@raspberrypi:~ $ sudo uno220gpio

```
pi@raspberrypi:~ $
pi@raspberrypi:~ $ sudo uno220gpio --pin=0 --direction=out
pi@raspberrypi:~ $ sudo uno220gpio
 pin       |   0   1   2   3   4   5   6   7
-----------------------------------------------
 export    |   1   1   1   1   1   1   1   1
 direction |   0   I   I   I   I   I   I   I
 value     |   0   0   1   1   1   1   1   1
```

Set value (ex: pin=0, direction=out, value=1)

pi@raspberrypi:~ $ sudo uno220gpio --pin=0 --value=1

pi@raspberrypi:~ $ sudo uno220gpio

```
pi@raspberrypi:~ $ sudo uno220gpio --pin=0 --value=1
pi@raspberrypi:~ $ sudo uno220gpio
 pin       |   0   1   2   3   4   5   6   7
-----------------------------------------------
 export    |   1   1   1   1   1   1   1   1
 direction |   0   I   I   I   I   I   I   I
 value     |   1   1   1   1   1   1   1   1
pi@raspberrypi:~ $
```

Serial Port

Serial port test - PC (Ubuntu 16.04 x86-64) vs Pi

Connect PC's RS-232 TxD/RxD/GND pins to IO Board corresponding pins.

a. PC send data to Pi

Pi side command:

pi@raspberrypi:~ $ sudo uno220uartrecv

```
pi@raspberrypi:~ $ sudo uno220uartrecv
```

PC side command:

$ ./files/host-x86_64/host_send /dev/ttyUSB0 $(echo -ne "\x01\x02\x03")

Then, Pi will show received data prompt.

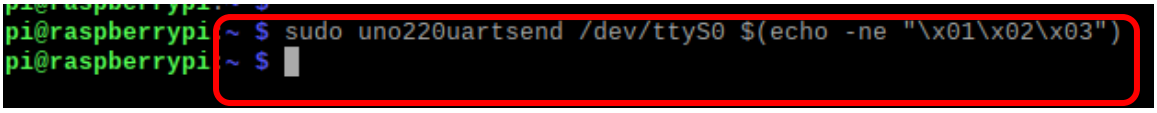
b. Pi send data to PC

PC side command:

$ sudo ./host_recv /dev/ttyUSB0

Pi side command:

pi@raspberrypi:~ $ sudo uno220uartsend /dev/ttyS0 $(echo -ne "\x01\x02\x03")



#Then, Pi will show received data prompt.