**SDK and system resources of B3VCU**

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Reason For Changes | Version |
| Chen Yong | Jul 7, 2022 | Initial draft | 1.0.1 |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1 Scope 4](#__RefHeading___Toc1662_3499696116)

[2 System 4](#__RefHeading___Toc1664_3499696116)

[3 Enviroment 4](#__RefHeading___Toc1666_3499696116)

[3.1 VCU 4](#__RefHeading___Toc1669_3499696116)

[3.2 Programming computer 5](#__RefHeading___Toc1671_3499696116)

[4 B3VCU system resources 5](#__RefHeading___Toc2357_3499696116)

[4.1 External interface 5](#__RefHeading___Toc2359_3499696116)

[4.1.1 CAM UART 5](#__RefHeading___Toc2361_3499696116)

[4.1.2 VDU UART 5](#__RefHeading___Toc2363_3499696116)

[4.1.3 Spare UART 6](#__RefHeading___Toc2365_3499696116)

[4.1.4 Ethernet 6](#__RefHeading___Toc2367_3499696116)

[4.1.5 KeyPad 6](#__RefHeading___Toc2369_3499696116)

[4.1.6 Digital GPIO Output 6](#__RefHeading___Toc2371_3499696116)

[4.1.7 SDI Video Input 6](#__RefHeading___Toc2373_3499696116)

[4.1.8 SDI video Output 7](#__RefHeading___Toc2375_3499696116)

[4.1.9 *LED* 7](#__RefHeading___Toc2377_3499696116)

[4.2 Internal Interface 7](#__RefHeading___Toc2379_3499696116)

[4.2.1 I2C 7](#__RefHeading___Toc2381_3499696116)

[4.2.2 frmbuf\_rd 7](#__RefHeading___Toc2383_3499696116)

[4.2.3 mix 7](#__RefHeading___Toc2385_3499696116)

[4.2.4 uhdsdi\_tx 8](#__RefHeading___Toc2387_3499696116)

[4.2.5 sdi\_anc\_tx 8](#__RefHeading___Toc2389_3499696116)

[4.2.6 IIo sensor 8](#__RefHeading___Toc2391_3499696116)

[4.2.7 pdma 8](#__RefHeading___Toc2393_3499696116)

[4.2.8 Reserve memory 8](#__RefHeading___Toc2395_3499696116)

[4.2.9 Watchdog 8](#__RefHeading___Toc6414_1605989181)

[5 Programming 9](#__RefHeading___Toc6416_1605989181)

[5.1 A simple firmware 9](#__RefHeading___Toc6418_1605989181)

[5.1.1 Edit 9](#__RefHeading___Toc6420_1605989181)

[5.1.2 Compile 10](#__RefHeading___Toc6422_1605989181)

[5.1.3 Run firmware 10](#__RefHeading___Toc6424_1605989181)

# Scope

# System

VCU

VDU

CAM

Display

RTP/UDP

player

Hub

VCU Control

computer

Programming

computer

Figure is the full system physical linking. The devices and linking for programming are in the dotted line box.

# Enviroment

## VCU

Defaut setting:

* static IP (192.168.9.115)
* username: root
* password: root

The user can download and upload files to VCU with sftp client software.

The user can control the VCU with the serial port debug application(e.g: teraterm).

The user also can log in and control the VCU with putty or ssh.

## Programming computer

* This computer is for programming the software fo the VCU. The Ubuntu(20.04) is required and verified to work.
* This computer has the same subnet IP address with the VCU.
* The user need install the requirement software eg Filezilla(sftp client), GTKTerm(serial debug) or ssh for downloading, uploading and debugging the VCU files.
* The user run sdk.sh(from teamone) to install the sdk compiler.
* <your-sdk-dir>/environment-setup-cortexa72-cortexa53-xilinx-linux includes the enviroment setup for the compiler. The user need setup the enviroment of the compiler based on their SDK requirement.

# B3VCU system resources

## External interface

### CAM UART

This port is link to the TI camera for the NUC controlling.

***Physcial link****:* ***TIA/EIA-422***

***Device name****: /dev/ttySC0*

***working mode****:Duplex*

***flow control****: no*

### VDU UART

This port is link to the VDU for the VDU switching command.

***Format****:* ***TIA/EIA-422***

***Device name****: /dev/ttySC1*

***working mode****:Duplex*

***flow control****: no*

### Spare UART

***Format****: TIA-232*

***Device name****: /dev/ttyPS1, /dev/ttySC2, /dev/ttySC3*

***working mode****:Duplex*

***flow control****: no*

### Ethernet

1**0/100/1000 BASE-T**

* **Port 1:** VCU control port. default static IP 192.168.9.115
* **Port 2:** VCU RTP video stream port. Default Static IP 192.168.1.111
* **Port 3:** VCU RTP video stream port. Default Static IP 192.168.2.112

### KeyPad

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Key No | Gpio port | Key name | Linux code |  |
| KEY\_DIN\_1 | <axi\_gpio\_0 0> | key1 | KEY\_1(2) | Trailer ? |
| KEY\_DIN\_2 | <axi\_gpio\_0 1> | key2 | KEY\_2(3) | Reverse Gear |
| KEY\_DIN\_3 | <axi\_gpio\_0 2> | key3 | KEY\_3(4) | Forward Gear |
| KEY\_DIN\_4 | <axi\_gpio\_0 3> | key4 | KEY\_4(5) | TBD |
| KEY\_DIN\_5 | <axi\_gpio\_0 4> | key5 | KEY\_5(6) | TBD |
| KEY\_DIN\_6 | <axi\_gpio\_0 5> | key6 | KEY\_6(7) | TBD |
| KEY\_DIN\_7 | <axi\_gpio\_0 6> | key7 | KEY\_7(8) | TBD |
| KEY\_DIN\_8 | <axi\_gpio\_0 7> | key8 | KEY\_8(9) | TBD |

### Digital GPIO Output

|  |  |  |  |
| --- | --- | --- | --- |
| DO No | Linux GPIO No | Terminal | Description |
| DO1 | 492 | IR | On/OFF IR |
| DO2 | 493 | IR | On/OFF IR |
| DO3 | 494 | TBD |  |
| DO4 | 495 | TBD |  |
| DO5 | 496 | TBD |  |

### SDI Video Input

**Device name**: /*dev/media6 (/dev/video0)*

*/dev/media7 (/dev/video1)*

*/dev/media8 (/dev/video2)*

*/dev/media9 (/dev/video3)*

*/dev/media0 (/dev/video4)*

*/dev/media1 (/dev/video5)*

*/dev/media2 (/dev/video6)*

*/dev/media3 (/dev/video7)*

*/dev/media4 (/dev/video8)*

*/dev/media5 (/dev/video9)*

### SDI video Output

*SDI0 output: frmbuf\_rd --> mix -> uhdsdi\_tx (‘mix’ can mix video and osd)*

*SDI1 – SDI7 output: frmbuf\_rd->udhsdi\_tx. (without osd funciton)*

*There is a device “sdi\_anc\_tx” in all output. It can embed timestamp into the SDI signal.*

*all devices refer to internal interface:*

*frmbuf\_rd: refer to 4.2.2*

*mix: refer to 4.2.3*

*uhdsdi\_tx: refer to 4.2.4*

*sdi\_anc\_tx: refer to 4.2.5*

### *LED*

|  |  |  |
| --- | --- | --- |
| LED | Linux GPIO No | Description |
| Green | 458 | Normal Status |
| Red | 459 | BIT error |

## Internal Interface

### I2C

**device name**: /*dev*/i2c-0

**slave chip**: mb85rc256vfp **slave address**: 0x50

**description**: log data storage

### frmbuf\_rd

**Address**: sdi0(0xb02d0000) sdi1(0xb02f0000) sdi2(0xb0310000) sdi3(0xb0330000) sdi4(0xb03a0000) sdi5(0xb03c0000) sdi6(0xb03e0000) sdi7(0xb0400000)

**Support Format**: Y\_UV10, Y\_UV8

refer to document (pg278-v-frmbuf.pd) provided by Xilinx

### mix

**Address:** 0xb02e0000

**Primary video format**: YUV4:2:2 10bits

**Overlay layer1 format**: BGRA8

**Maximum resolution**: 1920x1080

refer to doucument (pg243-v-mix.pdf) provided by Xilinx

### uhdsdi\_tx

**Address**: sdi0(0xa0040000) sdi1(0xa0060000) sdi2(0xa0080000) sdi3(0xa00a0000) sdi4(0xa0100000) sdi5(0xa0120000) sdi6(0xa0140000) sdi7(0xa0160000)

**Mode**: 3G SDI 10bits

**format**: NV16\_10LE32

refer to document (pg289-v-smpte-uhdsdi-tx-ss.pdf) provided by Xilinx

### sdi\_anc\_tx

**Address**: sdi0(0xa01c0000) sdi1(0xa01d0000) sdi2(0xa01e0000) sdi3(0xa01f0000) sdi4(0xa0200000) sdi5(0xa0210000) sdi6(0xa0220000) sdi7(0xa0230000)

**data format**: 64bits unsigned interger. Microseconds of the epoch time.

### IIo sensor

CPU:/sys/bus/iio/devices/iio\:device0

Power Current Sensor:/sys/bus/iio/devices/iio\:device1

### pdma

It is a user interface to transfer a DAM address of the user space to the physical address.

**Device**: /*dev*/pdma0

**ioctl cmd**: 0

**ioctl parameters**: struct{guint64 param\_in; guint64 param\_out}

param\_in: the dma user-space address.

param\_out: the dma physical address

### Reserve memory

**reserve memory physcial address**:0x70000000

**reserve memory size**: 0xff00000

### Watchdog

**Device**: /*dev/watchdog0, /dev/watchdog1*

*watchdog0 is controlled by OS.*

*Watchdog1 can be controoled by the user.*

# Programming

## A simple firmware

### Edit

Install Atom as the SDK editor. Install Atom requirement plugin.

Create the Project Folder and Add this project folder into the Atom.

create a new file ‘Makefile’ and a folder ‘src’ in this project folder.

Create a source file ‘main.c’ in the folder ‘src’.

**edit Makefile:**

TARGET := $(notdir $(CURDIR))

CSOURCES := $(wildcard src/\*.c)

COBJECTS := $(CSOURCES:%.c=%.o)

CPPSOURCES := $(wildcard src/\*.cpp)

CPPOBJECTS := $(CPPSOURCES:%.cpp=%.o)

CFLAGS := -O

LDFLAGS :=

LIBS :=

.PHONY: $(TARGET)

$(TARGET): $(COBJECTS) $(CPPOBJECTS)

$(CXX) -o $@ $^ $(LDFLAGS) $(LIBS)

$(COBJECTS): %.o: %.c

$(CC) $(CFLAGS) -o $@ -c $<

$(CPPOBJECTS): %.o: %.cpp

$(CXX) $(CFLAGS) -o $@ -c $<

clean:

rm -rf \*~ .dep $(COBJECTS) $(CPPOBJECTS) $(TARGET)

**edit the main.c:**

#include <stdio.h>

int main()

{

printf(“hello world!\n”);

return 0;

}

### Compile

Open Terminal and Go to your project folder. Run “make”.

Then you get a executable file ‘<your-project-folder>’ in <your-project-folder>.

### Run firmware

* Download the executable file to /*usr/local/bin in the VCU with the sftp client software*
* *Open GTKTerm and connect to the VCU with the USB debug port.(e.g /dev/ttyUSB0)*
* *Disable the original VCU firmware in /etc/init.d/loaduserapp.sh and reboot the VCU*
* *go to the folder /usr/local/bin and run your application*