ECE 484W

Final Project

Nicholas Barnes

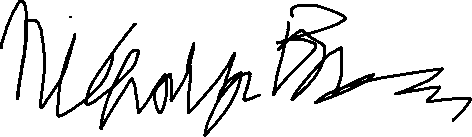
01067990

Team:

Nicholas Barnes

Garland Johnson

Due: 12/11/2020



Honor Code: X\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

The purpose of this assignment is to pass camera data through the NIOS processor to the on-board monitor on the VEEK-MT2S board. The Qt GUI developed over the semester is used to transmit an image, overlay it over the video feed, and change the brightness and contrast of the camera feed using the OpenCV Library.

# Table of Contents

1. Introduction

2. Design Methodology

3. Results and Analysis

4. Alternative Design

5. Additional Considerations in Design Process

6. Broader Impact

6.1 Economic Impact

6.2 Environmental Impact

6.3 Global Impact

6.4 Societal Impact

7. Individual Contribution

8. Reference

9. Appendix

# Introduction

The objective of this project is to complete three tasks utilizing the Qt GUI and the VEEK-MT2S . The tasks are as follows:

* 1. Add an overlay image to every frame of the video stream from the on-board camera. The overlay image will be transmitted from the GUI on the computer. (Requirement #1 )
  2. Adjust brightness and contrast of the video stream from the on-board camera. The brightness and contrast values will be transmitted from the GUI on the computer. (Requirement #2)
  3. Show the result on the on-board monitor. (Requirement #3)
  4. Software Components
     1. Qt

Qt is a framework written in C++ used to extend the capabilities of C++ with features like signals and slots. The Qt version used in this assignment is 5.13.2. The Qt Creator version used is 4.13.0.

* + 1. OpenCV

“OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library.” This library allows us to dynamically change the brightness and contrast of an image. The version of OpenCV used in this assignment is 4.11.

* 1. Hardware Components
     1. VEEK-MT2S

This is a Field-Programmable Gate Array that “presents a robust hardware design platform built around the Intel System-on-Chip (SoC) FPGA, which combines the latest dual-core Cortex-A9 embedded cores with industry-leading programmable logic for ultimate design flexibility.” [11] Implements a DE10-Standard with a built in touch screen interface.

* + 1. FIOS-G1100

Router used to establish communication between PC and DE-10 Standard board. Verizon service.

* + 1. Computer Specifications

These are the specifications of the computer running the Qt Software:

Model: Precision 5520

Processor: Intel Core I-7 CPU @ 2.90 GHz

RAM: 32 GB

System Type: 64 bit OS, x64-based processor.

# Design Methodology

* 1. QT GUI

The Final Assignment uses the GUI developed over the course of the semester. One section of the GUI handles TCP communication.

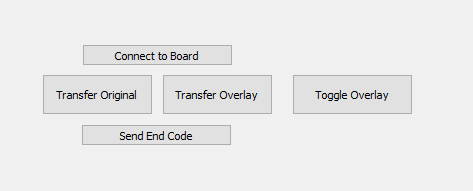


Figure 1. Board Communication Interface

Connect to Board opens a TCP socket with the server on the VEEK. Transfer Original/Overlay would send their respective images over the TCP connection, but are not used in this assignment.

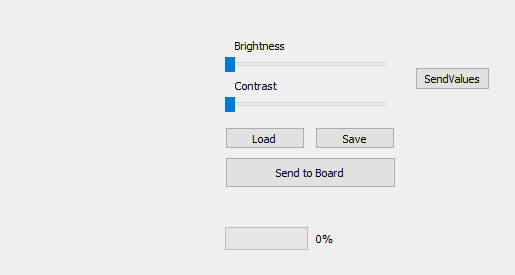


Figure 2. GUI that handles the overlay image and the brightness and contrast sliders.

“Load” loads an image to be overlaid over the camera feed. The image data is buffered using a ByteArray and a QBuffer, and sent in chunks of 1024 bytes.

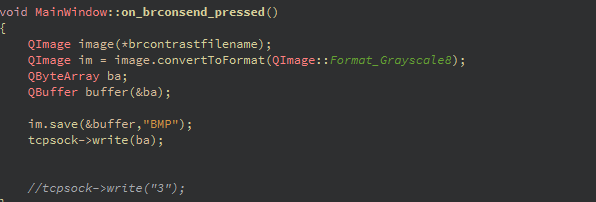


Figure 3. Writing to the TCP socket

After the image is finished sending, an end code “3” must be sent to the board by pressing the button “Send End Code.” The user can then adjust the brightness and contrast sliders and their values are sent to the board.

* 1. VEEK-MT2S C++ Server.

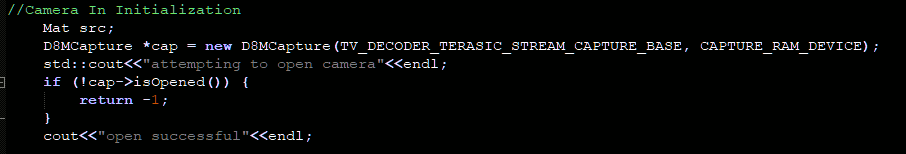
The C++ server on the VEEK-MT2S board is adapted from the camera\_in example provided in the VEEK-MT2S CD. Before video feed can be captured, the camera must be opened. The example does this by declaring a D8MCapture object which is mapped to the memory location of the on-board camera. 

Figure 4. Opening the Camera

After a successful opening of the camera, the program sets up the TCP server. The program applies TCP properties to the socket, binds the socket to the port 8080. And starts listening for a connection. The board starts listening for image packets of 1024 bytes after accepting a connection.

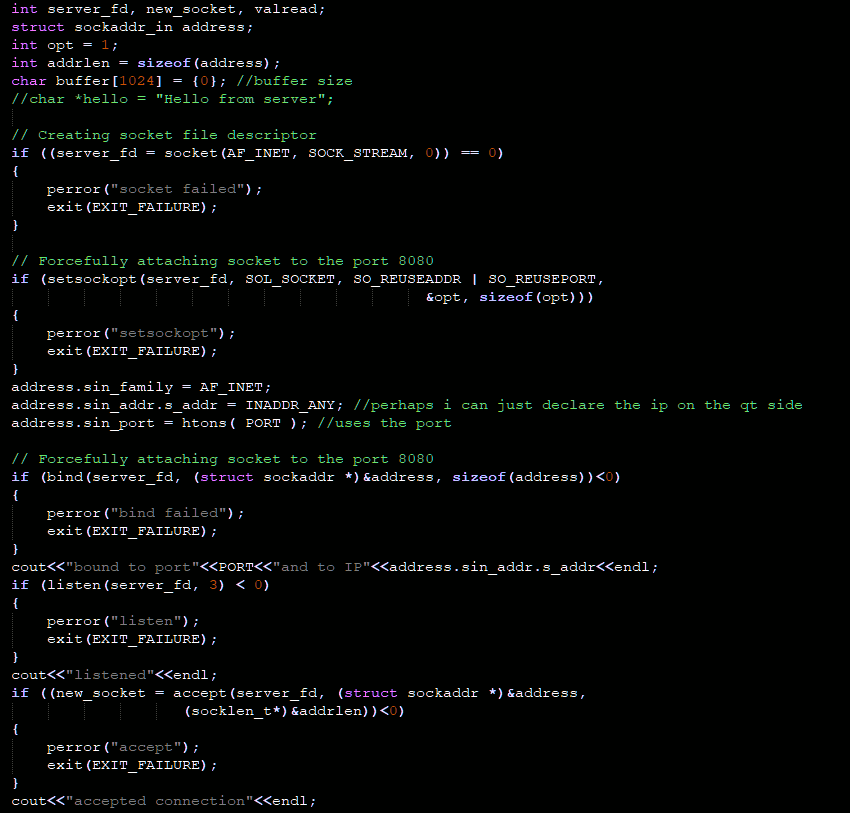


Figure 5. TCP Server Setup.

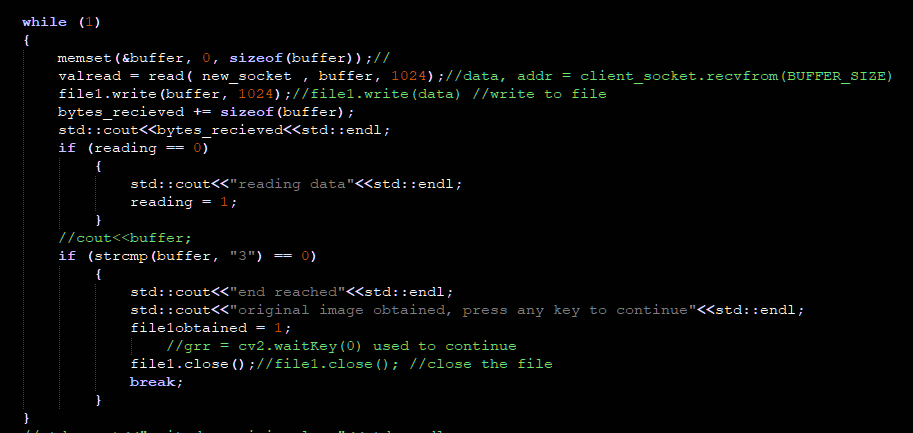


Figure 6. Accepting Image Data

After the end code is received, the image is saved to the board, resized, and converted to RGB from grayscale so that it can be properly display on the video capture.

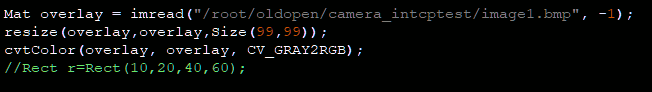


Figure 7. Adjusting the Overlay image to be overlaid onto video feed.

A loop is used to display a frame from the D8MCapture object “src.” The board then waits for brightness and contrast values from the Qt Client. The values received have 48 subtracted from them so their character data is displayed as an integer. Brightness is multiplied by 10 to make brightness changes more apparent.

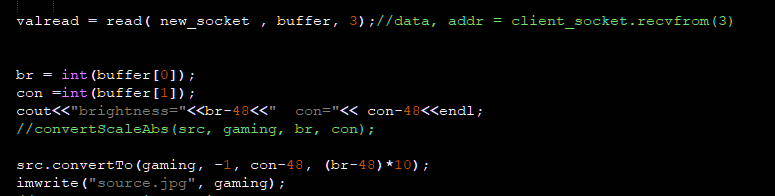


Figure 8. Brightness and Contrast Adjustment

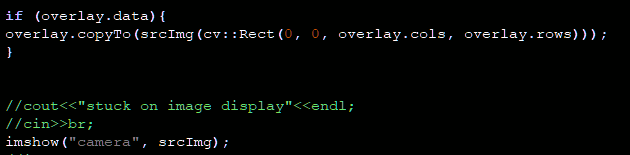
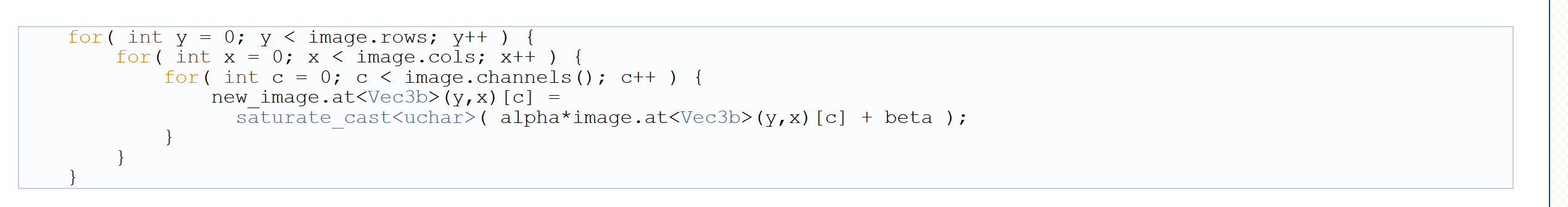
The image received from the client is then overlaid onto the camera frame using the CopyTo() function. The final frame is then displayed on the on-board monitor.

Figure 9. Overlaying Algorithm and Frame Display.

* + 1. Adjusting the brightness and contrast using OpenCV

This convertTo() function simplifies a linear transform using alpha (contrast) and beta (brightness) values. [5] The operation “g(i,j) = a \* f(i,j) +B” is applied each pixel in the image through the algorithm shown below where f(i,j) is the source image and g(i,j) is the destination image:

 Figure 10. In-depth look at the algorithm of convertTo()

The alpha, or contrast, value is also known as gain. Beta, or brightness, is also known as bias. Adjusting these two values with the sliders will produce an image with different brightness or contrast levels.

# Analysis and Results

1. Execution

The image sent to the board must be overlaid onto the camera frame. The loop executes in this order:

* + 1. Capture Frame from Camera
    2. Adjust the Brightness and Contrast
    3. Overlay the received image.

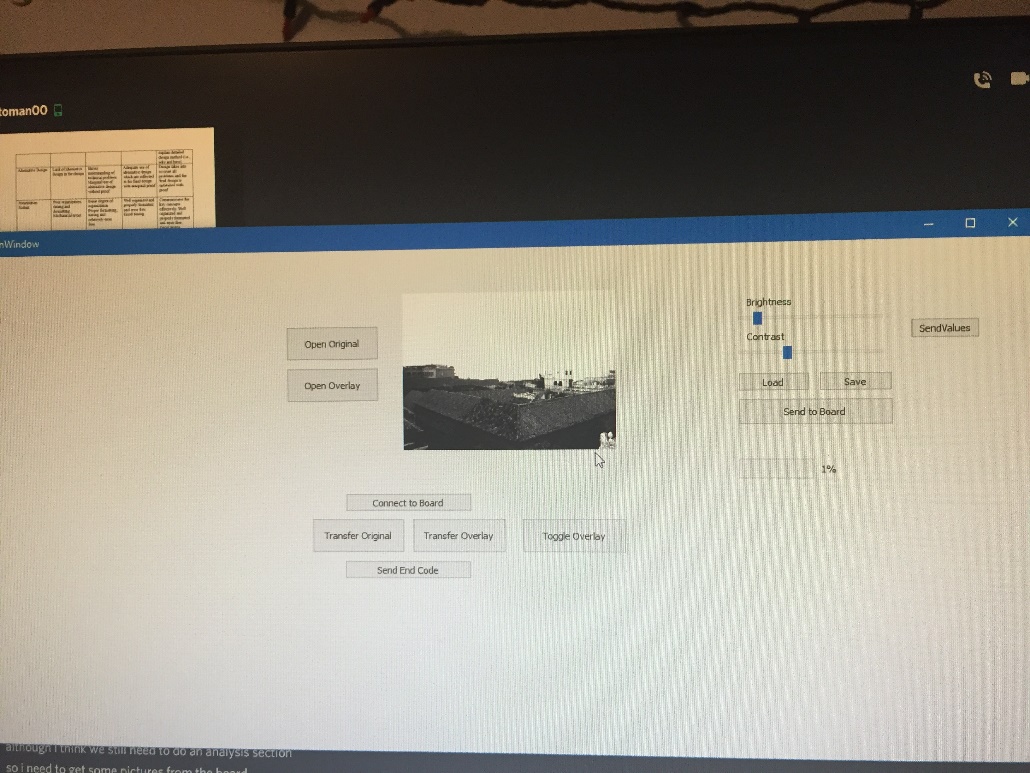


Figure 11. Normal Contrast Input

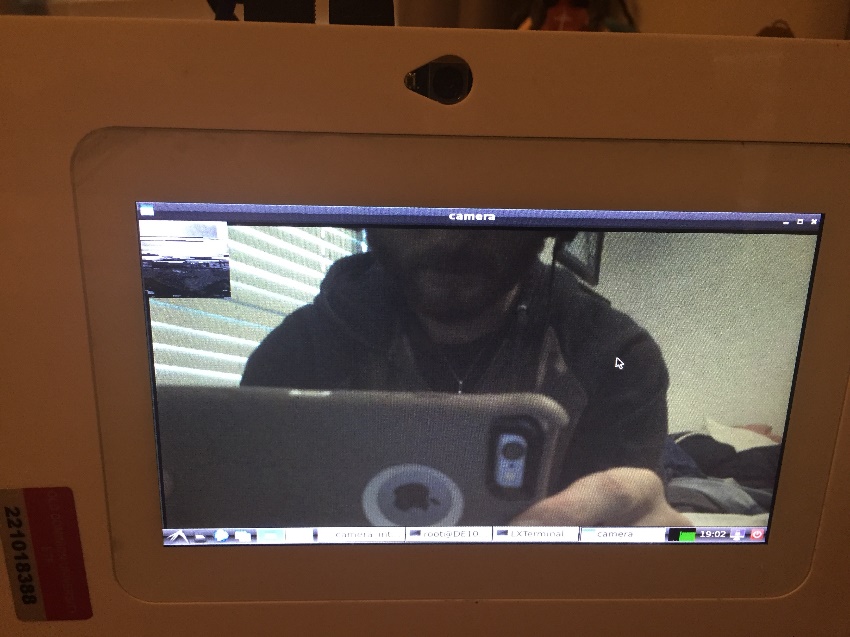


Figure 12. Normal Contrast Output.

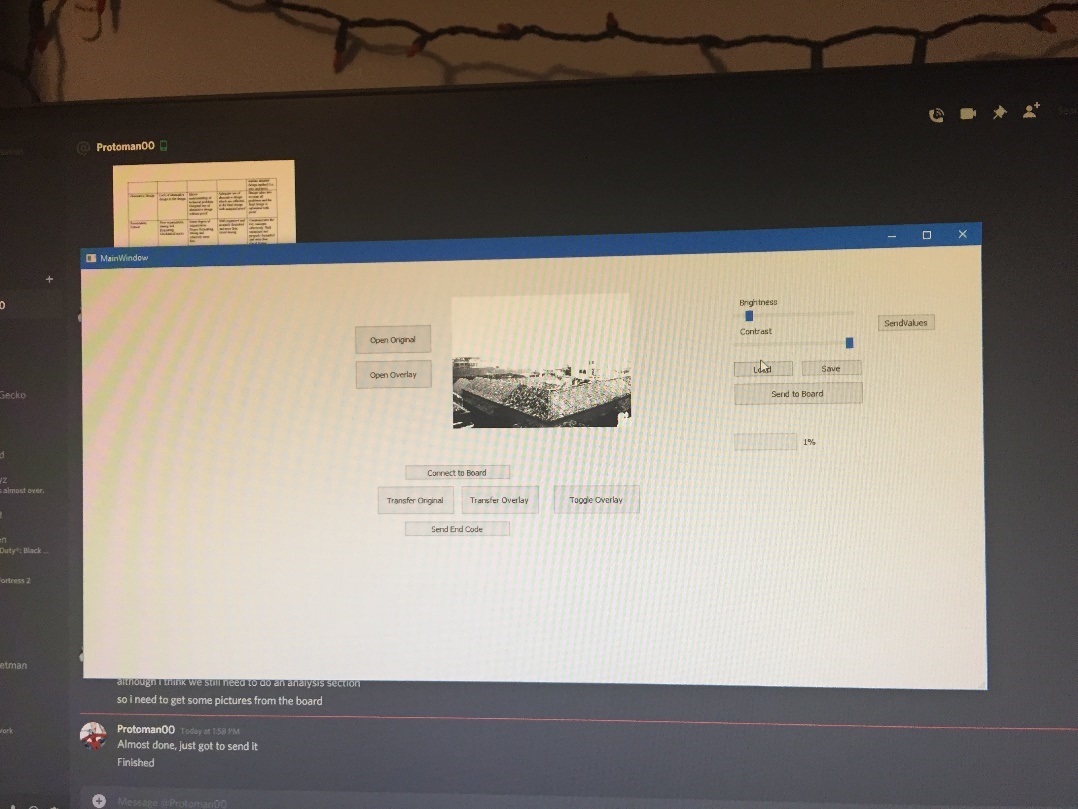


Figure 13. High Contrast Input.

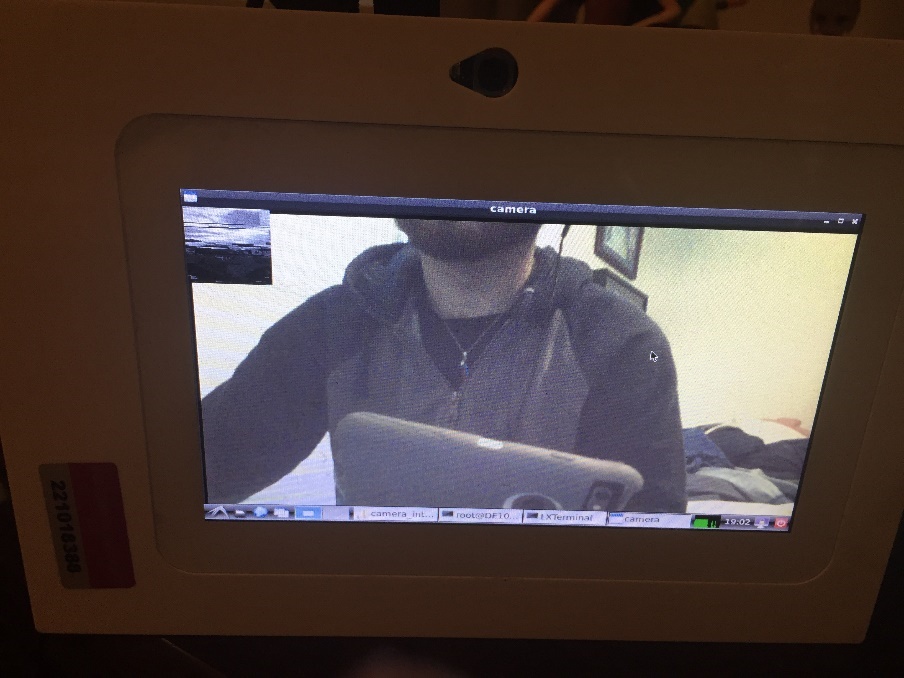


Figure 14. High Contrast Output.

B. Analysis

The images are properly overlaid onto the camera frame. However, for the camera frame to progress, another brightness/contrast value must be sent to the board. This is due to the loop waiting for a one byte packet that contains a brightness/contrast value.

The execution images are captured using an external camera. Thus, the quality of the images and the difference in results are not done proper justice.

Contrast is easy to demonstrate since its values range from 0-5. Brightness was also implemented with values 0-9. However, when demonstrating brightness with such small values it is difficult to tell the difference in a document.

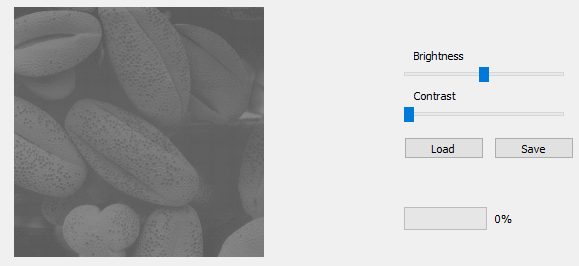


Figure 15. Image with no values

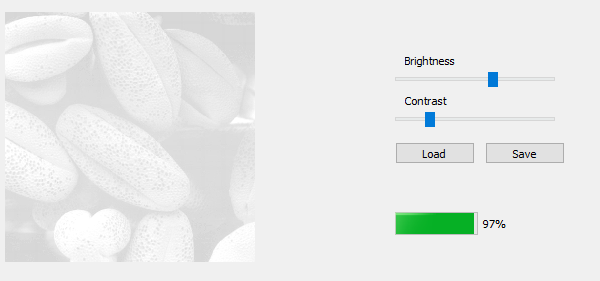


Figure 16. Image with High Brightness.\*

\*Note: Contrast needed to be a nonzero value for image to be properly seen.

See section 2.2.1 for more details on how the convertTo() function works.

# Alternative Design

* 1. Histogram Equalization

OpenCV has an algorithm for Histogram Equalization, equalizeHist(), which takes an image, normalizes its brightness, and increases its contrast.



Figure 15. Image from OpenCV Python Documentation

Although this method would create a clear image from the inputted image, it does not provide a way for manually adjusting the brightness and contrast.

* 1. QT Integration on VEEK-MT2S

QT could be used to display images on the VEEK-MT2S. This would allow opening images to be automated with no further input from the user. Toggling the image would imply keeping a copy of the original image, and loading whichever image is necessary into the GUI.

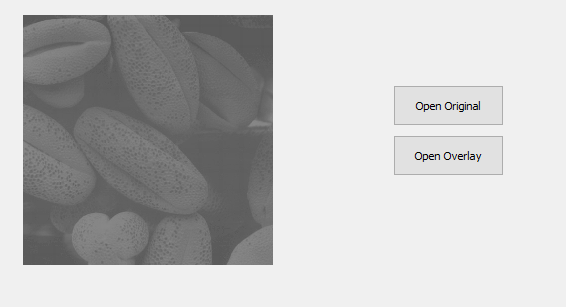
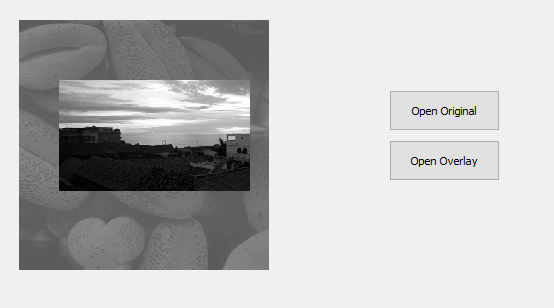
 

Figure 16. Before and After Toggle.

* 1. Python Integration.

We had a working TCP server, overlaying, and brightness/contrast control algorithm in python from Assignment 4. However, we were unable to get input from the camera in the Python server. Thus, we resorted to C++. If we figured out how to capture camera frames in python, we could have had more time to clean up the program, such as decreasing the time to send data to the board.

* 1. Sending dummy brightness/contrast values to the board

A possible solution to the video feed pausing when waiting for a packet is for the client to send dummy packets when the slider is not being adjusted. This would allow the video feed to continue playing while not changing the brightness and contrast of the feed.

# Additional Design Considerations

5.1 Economic Considerations:

Unlike OpenCV, Qt is not a true open source framework. Instead, those who choose to develop under Qt take a “Quid pro quo” agreement, or something for something. “In return for the value you receive from using Qt to create your application, you are expected to give back by contributing to Qt or buying Qt.[6]” Because of this, abiding loosely by the IEEE code of ethics Section 1.6 “to undertake technological tasks for others only if qualified by training or experience”, we should not pursue to release this program commercially or plan to make a profit off it. We are using the open source license for Qt for educational purposes only.

5.2 Social Considerations/Data Security:

The data we send over the network is unencrypted; anyone can see the data that we send. We must take care to not send personal information or potential security breaches across the socket. That said, the risk would be greater if the information was sent across an unsecured network. A program such as wireshark could be used to view the packets between the computer and the VEEK-MT2S.

So, when sending data over a network, we need to consider IEEE privacy standards. [IEEE Recommended Practice for Privacy Considerations for IEEE 802(R) Technologies]. Identifiers, or “A name, address, label, distinguishing index, or attribute specified in an IEEE 802® standard,” should be limited in their distribution over a network by principle of Data Minimization.

5.3 Sustainability

The code for this assignment has been adapted over many iterations. It started as a GUI to display images and adjust their brightness and contrasts. Next, it sent brightness and contrast values over a network. Finally, it sent images and brightness/contrast values and displayed it on the VEEK-MT2S. The code can be adapted further for future projects where image processing is necessary for video. For example, an autonomous vehicle that roams a dark area can have its brightness and contrast adjusted to improve visibility.

# Broader Impact

* 1. Economic Impact

Image processing is vital to the development of transportation technologies. In the dark, a self-driving car must differentiate between the road its driving on, vehicles surrounding it, and people that potentially enter the road. Perfecting this technology is key to ensuring the production of self-driving vehicles is profitable and safe [1].

Sending information across a network enables new technologies that can be sold on the market. Smart appliances can work together to provide a comfortable experience for the user [7] and will most likely get cheaper as time progresses.

* 1. Environmental Impact

Image processing is also used in wildlife monitoring. Systems such as Where’s the Bear “integrate recent advances in machine learning based image processing to automatically classify animals in images from re-mote, motion-detection camera traps [2].” Perfecting this technology could increase the efficiency of such systems, so that we can properly track how a particular species is doing in a habitat.

Networks should also be established in wildlife monitoring. Data is transmitted between sensors to provide an idea of how an ecosystem is doing, from agricultural to volcanic monitoring.[10]. It becomes a challenge to maintain sensor networks over long periods of time, but it is needed to ensure “continuous interaction with domain scientists.”

TCP can play an important role in disaster prevention, as while 5G has applications of high-quality drone video capture, it has some misalignment problems. [12] Woongsoo and co. proposed a deep learning TCP to improve stability and reliability for video capture in a disaster environment.

* 1. Global Impact

Recent events in 2020 see the West Coast of the United States burning all over. Detecting forest fires in any part of the world could be accomplished through image processing. While commercial fire sensors exist, they do not work well in open areas. Mahmoud and Ren propose an image processing algorithm that utilizes temporal variation to detect forest fires [3]. This could be effective in the United States but could also be effective in continent-wide fires such as the ones in Australia that also occurred in 2020.

In terms of networking, ensuring network security is important to maintain election fidelity. The US Department of Homeland Security, “through its Cybersecurity and Infrastructure Security Agency, provide resources to State and local election officials… to reduce cyber and physical risk to their election systems.” [9]

* 1. Societal Impact

Image processing already sees use in security and red-light cameras. However, combined with neural networks could provide a deeper identification of potential security breaches [4]. On a more basic level, increasing the brightness and contrast of an image is useful for detecting shapes in the dark, which has applications in night security.

People rely on networks to reliably transfer information from one user to another. It is also important to encrypt that information to secure users’ privacy. There is difficulty in ensuring proper encryption when many people try to send a lot of information at once. For example, environments like Twitter have millions of users sending information at once, and researchers such as Zhu propose many methods of fast, online identification methods for encryption network behaviors. [8]

# Individual Contribution

I started work on this project on November 23rd. I quickly concluded that I could not use the python server and algorithm developed for Assignment 4 due the difficulty in accessing the on-board camera. As a result, I redid the TCP server code and Image Processing algorithm in C++. I was able to get the brightness and contrast algorithm redone quickly, and on December 4th, I left it to Garland to get the overlaying algorithm done over the weekend. He did not have much success, but with the code we had he created the presentation to be recorded on the 7th. I was able to communicate with Jiajun the night of December 6th to get the overlaying algorithm done.

I did most if not all of the programming, but Garland was a real help with the presentation. I made sure to give him a chance to work on the programming, but I don’t think it’s fault that he couldn’t figure out the overlaying algorithm. I could not figure it out in the week after the brightness and contrast algorithm was complete and thought he would have better luck. It all worked out in the end, though.

# References

# [1] D. D. A. A. M. Shrey Shah, "Automated Driving CarUsing Image Processing," International Journal of Recent Trends in Engineering and Research, vol. 3, no. 4, p. 160, 2017.

# [2] N. G. C. K. R. W. Andy Rosales Elias, "Where’s The Bear? – Automating Wildlife ImageProcessing Using IoT and Edge Cloud Systems," Computer Science Department, University of California, Santa Barbara, CA, 2016.

# [3] H. R. Mubarak A. I. Mahmoud, "Forest Fire Detection Using a Rule-Based Image Processing Algorithm and Temporal Variation," Mathematical Problems in Engineering, vol. 2018, p. 8 pages, 2018.

# [4] G. R. Kotapalle and S. Kotni, "Security using image processing and deep convolutional neural networks," 2018 IEEE International Conference on Innovative Research and Development (ICIRD), Bangkok, 2018, pp. 1-6, doi: 10.1109/ICIRD.2018.8376292.

# [5] “OpenCV: Changing the contrast and brightness of an image!,” Opencv.org. [Online]. Available: https://docs.opencv.org/3.4.10/d3/dc1/tutorial\_basic\_linear\_transform.html. [Accessed: 22-Sep-2020].

# [6] The Qt Company, “Download Qt Open Source,” Www.qt.io. [Online]. Available: https://www.qt.io/download-open-source?hsCtaTracking=9f6a2170-a938-42df-a8e2-a9f0b1d6cdce%7C6cb0de4f-9bb5-4778-ab02-bfb62735f3e5. [Accessed: 22-Sep-2020].

[7] Takenaka, Takeshi, Yamamoto, Yoshinobu, Fukuda, Ken, Kimura, Ayaka, and Ueda, Kanji. "Enhancing Products and Services Using Smart Appliance Networks." CIRP Annals 65.1 (2016): 397-400. Web.

[8] Hejun, Zhu, and Liehuang, Zhu. "Online and Automatic Identification of Encryption Network Behaviors in Big Data Environment." *Concurrency and Computation* 31.12 (2018): N/a. Web.

[9] "Cybersecurity and U.S. Campaigns and Elections: Department of Homeland Security Report on Election Security." *The Congressional Digest* 98.8 (2019): 5. Web.

[10] Dyo, Vladimir, Ellwood, Stephen, Macdonald, David, Markham, Andrew, Trigoni, Niki, Wohlers, Ricklef, Mascolo, Cecilia, Pásztor, Bence, Scellato, Salvatore, and Yousef, Kharsim. "WILDSENSING." *ACM Transactions on Sensor Networks (TOSN)* 8.4 (2012): 1-33. Web.

# 8. References (Continued)

[11] Terasic Technologies, “Terasic - SoC Platform - Cyclone - DE10-Standard,” *Com.tw*. [Online]. Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=1081. [Accessed: 05-Oct-2020].

[12] Na, Woongsoo, Bae, Byungjun, Cho, Sukhee, and Kim, Nayeon. "DL-TCP: Deep Learning-Based Transmission Control Protocol for Disaster 5G MmWave Networks." IEEE Access 7 (2019): 145134-45144. Web.

# Appendix

9.1 camera\_in.cpp

//CameraIN Headers

#include <opencv2/opencv.hpp>

#include <sys/time.h>

#include "D8MCapture.h"

#include "hps\_0.h"

//------------------------------

//Server Headers

#include <unistd.h>

#include <stdio.h>

#include <sys/socket.h>

#include <stdlib.h>

#include <netinet/in.h>

#include <string.h>

#include <fstream>

#include <iostream>

#define PORT 8080

//

//#pragma comment (lib, "Ws2\_32.lib")

//---------------------------

using namespace cv;

using namespace std;

#ifndef CAPTURE\_RAM\_DEVICE

#define CAPTURE\_RAM\_DEVICE "/dev/f2h-dma-memory"

#endif /\* ifndef CAPTURE\_RAM\_DEVICE \*/

int main()

{

//Camera In Initialization

Mat src;

D8MCapture \*cap = new D8MCapture(TV\_DECODER\_TERASIC\_STREAM\_CAPTURE\_BASE, CAPTURE\_RAM\_DEVICE);

std::cout<<"attempting to open camera"<<endl;

if (!cap->isOpened()) {

return -1;

}

cout<<"open successful"<<endl;

char str[100];

static struct timeval last\_time;

struct timeval current\_time;

static float last\_fps;

float t;

float fps;

namedWindow("camera");

//---------------------

int server\_fd, new\_socket, valread;

struct sockaddr\_in address;

int opt = 1;

int addrlen = sizeof(address);

char buffer[1024] = {0}; //buffer size

//char \*hello = "Hello from server";

// Creating socket file descriptor

if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0)

{

perror("socket failed");

exit(EXIT\_FAILURE);

}

// Forcefully attaching socket to the port 8080

if (setsockopt(server\_fd, SOL\_SOCKET, SO\_REUSEADDR | SO\_REUSEPORT,

&opt, sizeof(opt)))

{

perror("setsockopt");

exit(EXIT\_FAILURE);

}

address.sin\_family = AF\_INET;

address.sin\_addr.s\_addr = INADDR\_ANY; //perhaps i can just declare the ip on the qt side

address.sin\_port = htons( PORT ); //uses the port

// Forcefully attaching socket to the port 8080

if (bind(server\_fd, (struct sockaddr \*)&address, sizeof(address))<0)

{

perror("bind failed");

exit(EXIT\_FAILURE);

}

cout<<"bound to port"<<PORT<<"and to IP"<<address.sin\_addr.s\_addr<<endl;

if (listen(server\_fd, 3) < 0)

{

perror("listen");

exit(EXIT\_FAILURE);

}

cout<<"listened"<<endl;

if ((new\_socket = accept(server\_fd, (struct sockaddr \*)&address,

(socklen\_t\*)&addrlen))<0)

{

perror("accept");

exit(EXIT\_FAILURE);

}

cout<<"accepted connection"<<endl;

int reading = 0; //flag

ofstream file1;//file1 = open("image1.bmp", "wb") //getfile1

file1.open("image1.bmp",ios::binary);

int file1obtained = 0; //flag

int br = 0;

int con = 0;

int bytes\_recieved = 0; //counter to keep track of all of the data

while (1)

{

memset(&buffer, 0, sizeof(buffer));//

valread = read( new\_socket , buffer, 1024);//data, addr = client\_socket.recvfrom(BUFFER\_SIZE)

file1.write(buffer, 1024);//file1.write(data) //write to file

bytes\_recieved += sizeof(buffer);

std::cout<<bytes\_recieved<<std::endl;

if (reading == 0)

{

std::cout<<"reading data"<<std::endl;

reading = 1;

}

//cout<<buffer;

if (strcmp(buffer, "3") == 0)

{

std::cout<<"end reached"<<std::endl;

std::cout<<"original image obtained, press any key to continue"<<std::endl;

file1obtained = 1;

//grr = cv2.waitKey(0) used to continue

file1.close();//file1.close(); //close the file

break;

}

}

//std::cout<<"exited receiving loop"<<std::endl;

//------------------------------------

Mat overlay = imread("/root/oldopen/camera\_intcptest/image1.bmp", -1);

resize(overlay,overlay,Size(99,99));

cvtColor(overlay, overlay, CV\_GRAY2RGB);

//Rect r=Rect(10,20,40,60);

Mat gaming,dst;

//--CameraIN loop-----------------

while (1) {

if (!cap->read(src))

return -1;

memset(&buffer, 0, sizeof(buffer));//

/\*

gettimeofday(&current\_time, NULL);

t = (current\_time.tv\_sec - last\_time.tv\_sec)

+ (current\_time.tv\_usec - last\_time.tv\_usec) / 1000000.;

fps = 1. / t;

fps = last\_fps \* 0.8 + fps \* 0.2;

last\_fps = fps;

last\_time = current\_time;

sprintf(str, "%2.2f, %2.6f", fps, t);

putText(src, str, Point(20, 40), FONT\_HERSHEY\_DUPLEX, 1,

Scalar(0, 255, 0));\*/

valread = read( new\_socket , buffer, 3);//data, addr = client\_socket.recvfrom(3)

br = int(buffer[0]);

con =int(buffer[1]);

cout<<"brightness="<<br-48<<" con="<< con-48<<endl;

//convertScaleAbs(src, gaming, br, con);

src.convertTo(gaming, -1, con-48, (br-48)\*10);

imwrite("source.jpg", gaming);

//cvtColor(gaming,gaming,COLOR\_GRAY2BGR);

Mat srcImg = imread("source.jpg",-1);

if (overlay.data){

overlay.copyTo(srcImg(cv::Rect(0, 0, overlay.cols, overlay.rows)));

}

//cout<<"stuck on image display"<<endl;

//cin>>br;

imshow("camera", srcImg);

//imshow("newframe", dst);

//cin>>br;

//-- bail out if escape was pressed

int c = waitKey(10);

if ((char) c == 27) {

con = 0;//break;

}

}

delete cap;

destroyAllWindows();

//------------------------------------------

return 0;

}

9.2 mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include<QFileDialog>

#include<QMessageBox>

#include<QImage>

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

#include<QtNetwork/QUdpSocket>

#include<QTcpSocket>

#include<QtNetwork/QNetworkDatagram>

#include<QBuffer>

#include<QTimer>

#include<QDebug>

#include <iostream>

using namespace cv;

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

//connect(&timer,&QTimer::timeout(),this, &on\_tf\_original\_clicked );

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_but\_original\_clicked() //When clicked, load the original image.

{

QString originalFilename=QFileDialog::getOpenFileName(

this,

tr("Open Original File"),

"",

"Bitmap (\*.bmp)"

);

QImage original(originalFilename); //loads an image from the filename

ui->lbl\_original->setPixmap(QPixmap::fromImage(original).scaled(250,250,Qt::KeepAspectRatio)); //set the image to the respective label

//oriTfrFilename->replace(0,99, originalFilename); //set the new filename when the image is loaded. Will not work for filenames over 100 characters

oriTfrFilename.replace(0,99, originalFilename);

}

void MainWindow::on\_but\_overlay\_clicked() //When clicked, load the overlay image.

{

QString overlayFilename= QFileDialog::getOpenFileName(

this,

tr("Open Overlay File"),

"",

"Bitmap (\*.bmp)"

);

QImage overlay(overlayFilename); //loads an image from the filename

ui->lbl\_overlay->setPixmap(QPixmap::fromImage(overlay).scaled(250,250,Qt::KeepAspectRatio)); //set the image to the respective label

overTfrFilename.replace(0,99, overlayFilename);

//overTfrFilename->replace(0,99, overlayFilename); //set the new filename when the image is loaded. Will not work for filenames over 100 characters.

}

void MainWindow::on\_btn\_load\_clicked()

{

QString brconFilename= QFileDialog::getOpenFileName(

this,

tr("Open Overlay File"),

"",

"Bitmap (\*.bmp)"

);

t2imageLoad = new QImage();

brcontrastfilename = new QString();

brcontrastfilename->replace(0,99, brconFilename); //set the new filename when the image is loaded. Will not work for filenames over 100 characters.

t2imageLoad->load(brconFilename);

QImage brcon(brconFilename);

ui->lbl\_t2display->setPixmap(QPixmap::fromImage(brcon).scaled(250,250,Qt::KeepAspectRatio));

}

void MainWindow::on\_sldr\_brightness\_valueChanged(int value)

{

//When the slider is adjusted, the image's X changes by a value.

//Each time the slider is changed

//--Change the Brightness

//--Display the Image

brightness\_value = value;

QString Data = "000";

// read an image

cv::Mat image = cv::imread(brcontrastfilename->toStdString(), 0);

cv::Mat image\_higher\_brightness;

//Change the brightness

image.convertTo(brconImage, -1, contrast\_value, brightness\_value);

QString cv = QString::number(contrast\_value);

QString bv = QString::number(brightness\_value);

Data.replace(1,1,cv);

Data.replace(2,1,bv);

tcpsock->write(Data.toStdString().c\_str(), 3);

//socket->writeDatagram(packettobesent.toStdString(),host,80);

brconOut = QImage((uchar\*) brconImage.data, brconImage.cols, brconImage.rows, brconImage.step, QImage::Format\_Grayscale8);

ui->lbl\_t2display->setPixmap(QPixmap::fromImage(brconOut).scaled(250,250,Qt::KeepAspectRatio));

}

void MainWindow::on\_sldr\_contrast\_valueChanged(int value)

{

//When the slider is adjusted, the image's Y changes by a value.

QString Data = "000";

contrast\_value = value;

// read an image

cv::Mat image = cv::imread(brcontrastfilename->toStdString(), 0);

cv::Mat image\_higher\_contrast;

QString cv = QString::number(contrast\_value);

QString bv = QString::number(brightness\_value);

Data.replace(1,1,cv);

Data.replace(2,1,bv);

//Change the brightness

image.convertTo(brconImage, -1, contrast\_value, brightness\_value);

tcpsock->write(Data.toStdString().c\_str(), 3);

//convert to QImage

brconOut = QImage((uchar\*) brconImage.data, brconImage.cols, brconImage.rows, brconImage.step, QImage::Format\_Grayscale8);

//display to label

ui->lbl\_t2display->setPixmap(QPixmap::fromImage(brconOut).scaled(250,250,Qt::KeepAspectRatio));

}

void MainWindow::on\_but\_save\_clicked()

{

QString imagePath = QFileDialog::getSaveFileName(

this,

tr("Save File"),

"",

tr("Bitmap (\*.bmp)" )

);

//brconOut.save(imagePath);

cv::imwrite(imagePath.toStdString(), brconImage );

//t2imageLoad->save(imagePath);

}

void MainWindow::on\_pushButton\_clicked()

{

//socket = new QUdpSocket;

tcpsock = new QTcpSocket;

//socket->connectToHost("192.168.1.16", 5354); //udp

tcpsock->connectToHost("192.168.1.249", 8080); //tcp

bool connectSuccess = tcpsock->waitForConnected(5000);

if(!connectSuccess)

{

//exit(-1);

qDebug() << "not Connected!";

}

// QString Data = "023";

//socket->write(Data.toStdString().c\_str(), 3);

}

void MainWindow::on\_tf\_original\_clicked()

{

QString choice = "1";

// initialize data

tcpsock->write(choice.toStdString().c\_str());

QImage image(oriTfrFilename);

QImage im = image.convertToFormat(QImage::Format\_Grayscale8);

QByteArray ba;

QBuffer buffer(&ba);

im.save(&buffer,"BMP");

tcpsock->write(ba);

//tcpsock->write("3");

}

void MainWindow::on\_tf\_overlay\_clicked()

{

//function transfers contents of overTfrFilename to tcp server

QString choice = "2";

// initialize data

tcpsock->write(choice.toStdString().c\_str());

QImage image(overTfrFilename);

QImage im = image.convertToFormat(QImage::Format\_Grayscale8);

QByteArray ba;

QBuffer buffer(&ba);

im.save(&buffer,"BMP");

tcpsock->write(ba);

//tcpsock->write("3");

}

void MainWindow::on\_endcodesend\_pressed()

{

tcpsock->write("3");

}

void MainWindow::on\_overlay\_toggle\_pressed()

{

QString flag;

if (toggle ==0)

{

QString flag = "1";

}

else

QString flag = "0";

// initialize data

tcpsock->write(flag.toStdString().c\_str());

}

void MainWindow::on\_brconsend\_pressed()

{

QImage image(\*brcontrastfilename);

QImage im = image.convertToFormat(QImage::Format\_Grayscale8);

QByteArray ba;

QBuffer buffer(&ba);

im.save(&buffer,"BMP");

tcpsock->write(ba);

//tcpsock->write("3");

}

void MainWindow::on\_sendvalues\_toggled(bool checked)

{

}

9.3 main.cpp

#include "mainwindow.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

MainWindow w;

w.show();

return a.exec();

}

9.4 mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

//For Task 2-------------------------------

#include<QWidget>

#include<QSlider>

#include<QScrollBar>

#include<QSpinBox>

#include<QProgressBar>

#include<QVBoxLayout>

#include<QFileDialog>

#include<QMessageBox>

#include<QImage>

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

#include<QtNetwork/QUdpSocket>

#include<QtNetwork/QNetworkDatagram>

#include<QTimer>

#include <QTcpSocket>

#include<QDebug>

#include <iostream>

QT\_BEGIN\_NAMESPACE

namespace Ui { class MainWindow; }

QT\_END\_NAMESPACE

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

QString oriTfrFilename;

QString overTfrFilename;

private slots:

void on\_but\_original\_clicked();

void on\_but\_overlay\_clicked();

void on\_btn\_load\_clicked();

void on\_but\_save\_clicked();

void on\_sldr\_brightness\_valueChanged(int value);

void on\_sldr\_contrast\_valueChanged(int value);

void on\_pushButton\_clicked();

void on\_tf\_original\_clicked();

void on\_tf\_overlay\_clicked();

void on\_endcodesend\_pressed();

void on\_overlay\_toggle\_pressed();

void on\_brconsend\_pressed();

void on\_sendvalues\_toggled(bool checked);

private:

Ui::MainWindow \*ui;

QPixmap t2Image;

QImage \*t2imageLoad;

QString \*brcontrastfilename;

cv::Mat brconImage;

int contrast\_value;

int brightness\_value;

QImage brconOut;

QUdpSocket \*socket;

QTcpSocket \*tcpsock;

QFile \* FiletoSend;

QByteArray outBlock;

int toggle = 0;

//QTimer &timer;

};

#endif // MAINWINDOW\_H

9.5 mainwindow.ui

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>MainWindow</class>

<widget class="QMainWindow" name="MainWindow">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>1254</width>

<height>595</height>

</rect>

</property>

<property name="windowTitle">

<string>MainWindow</string>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QPushButton" name="but\_original">

<property name="geometry">

<rect>

<x>400</x>

<y>90</y>

<width>111</width>

<height>41</height>

</rect>

</property>

<property name="text">

<string>Open Original</string>

</property>

</widget>

<widget class="QPushButton" name="but\_overlay">

<property name="geometry">

<rect>

<x>400</x>

<y>140</y>

<width>111</width>

<height>41</height>

</rect>

</property>

<property name="text">

<string>Open Overlay</string>

</property>

</widget>

<widget class="QLabel" name="lbl\_original">

<property name="geometry">

<rect>

<x>30</x>

<y>10</y>

<width>311</width>

<height>271</height>

</rect>

</property>

<property name="text">

<string/>

</property>

</widget>

<widget class="QLabel" name="lbl\_overlay">

<property name="geometry">

<rect>

<x>70</x>

<y>80</y>

<width>191</width>

<height>111</height>

</rect>

</property>

<property name="text">

<string/>

</property>

</widget>

<widget class="QSlider" name="sldr\_brightness">

<property name="geometry">

<rect>

<x>930</x>

<y>80</y>

<width>160</width>

<height>16</height>

</rect>

</property>

<property name="minimumSize">

<size>

<width>160</width>

<height>0</height>

</size>

</property>

<property name="minimum">

<number>0</number>

</property>

<property name="maximum">

<number>9</number>

</property>

<property name="singleStep">

<number>1</number>

</property>

<property name="value">

<number>0</number>

</property>

<property name="sliderPosition">

<number>0</number>

</property>

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

</widget>

<widget class="QSlider" name="sldr\_contrast">

<property name="geometry">

<rect>

<x>930</x>

<y>120</y>

<width>160</width>

<height>16</height>

</rect>

</property>

<property name="minimum">

<number>0</number>

</property>

<property name="maximum">

<number>9</number>

</property>

<property name="singleStep">

<number>1</number>

</property>

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

</widget>

<widget class="QLabel" name="lbl\_brightness">

<property name="geometry">

<rect>

<x>940</x>

<y>60</y>

<width>61</width>

<height>16</height>

</rect>

</property>

<property name="text">

<string>Brightness</string>

</property>

</widget>

<widget class="QLabel" name="lbl\_contrast">

<property name="geometry">

<rect>

<x>940</x>

<y>100</y>

<width>61</width>

<height>16</height>

</rect>

</property>

<property name="text">

<string>Contrast</string>

</property>

</widget>

<widget class="QPushButton" name="btn\_load">

<property name="geometry">

<rect>

<x>930</x>

<y>150</y>

<width>80</width>

<height>22</height>

</rect>

</property>

<property name="text">

<string>Load</string>

</property>

</widget>

<widget class="QPushButton" name="but\_save">

<property name="geometry">

<rect>

<x>1020</x>

<y>150</y>

<width>80</width>

<height>22</height>

</rect>

</property>

<property name="text">

<string>Save</string>

</property>

</widget>

<widget class="QLabel" name="lbl\_t2display">

<property name="geometry">

<rect>

<x>540</x>

<y>10</y>

<width>361</width>

<height>271</height>

</rect>

</property>

<property name="text">

<string/>

</property>

</widget>

<widget class="QProgressBar" name="progressBar">

<property name="geometry">

<rect>

<x>930</x>

<y>250</y>

<width>118</width>

<height>23</height>

</rect>

</property>

<property name="minimum">

<number>0</number>

</property>

<property name="maximum">

<number>100</number>

</property>

<property name="value">

<number>0</number>

</property>

</widget>

<widget class="QPushButton" name="pushButton">

<property name="geometry">

<rect>

<x>470</x>

<y>290</y>

<width>151</width>

<height>22</height>

</rect>

</property>

<property name="text">

<string>Connect to Board</string>

</property>

</widget>

<widget class="QPushButton" name="tf\_original">

<property name="geometry">

<rect>

<x>430</x>

<y>320</y>

<width>111</width>

<height>41</height>

</rect>

</property>

<property name="text">

<string>Transfer Original</string>

</property>

</widget>

<widget class="QPushButton" name="tf\_overlay">

<property name="geometry">

<rect>

<x>550</x>

<y>320</y>

<width>111</width>

<height>41</height>

</rect>

</property>

<property name="text">

<string>Transfer Overlay</string>

</property>

</widget>

<widget class="QPushButton" name="endcodesend">

<property name="geometry">

<rect>

<x>469</x>

<y>370</y>

<width>151</width>

<height>22</height>

</rect>

</property>

<property name="text">

<string>Send End Code</string>

</property>

</widget>

<widget class="QPushButton" name="overlay\_toggle">

<property name="geometry">

<rect>

<x>680</x>

<y>320</y>

<width>121</width>

<height>41</height>

</rect>

</property>

<property name="text">

<string>Toggle Overlay</string>

</property>

</widget>

<widget class="QPushButton" name="brconsend">

<property name="geometry">

<rect>

<x>930</x>

<y>180</y>

<width>171</width>

<height>31</height>

</rect>

</property>

<property name="text">

<string>Send to Board</string>

</property>

</widget>

<widget class="QPushButton" name="sendvalues">

<property name="geometry">

<rect>

<x>1120</x>

<y>90</y>

<width>75</width>

<height>23</height>

</rect>

</property>

<property name="text">

<string>SendValues</string>

</property>

</widget>

</widget>

<widget class="QMenuBar" name="menubar">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>1254</width>

<height>21</height>

</rect>

</property>

</widget>

<widget class="QStatusBar" name="statusbar"/>

</widget>

<resources/>

<connections>

<connection>

<sender>sldr\_brightness</sender>

<signal>valueChanged(int)</signal>

<receiver>progressBar</receiver>

<slot>setValue(int)</slot>

<hints>

<hint type="sourcelabel">

<x>883</x>

<y>80</y>

</hint>

<hint type="destinationlabel">

<x>890</x>

<y>227</y>

</hint>

</hints>

</connection>

</connections>

</ui>

9.6 D8MCapture.cpp

#include <stdint.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/mman.h>

#include <socal/hps.h>

#include <opencv2/opencv.hpp>

#include <sys/ioctl.h>

#include "D8MCapture.h"

using namespace cv;

#define HW\_REGS\_BASE (ALT\_STM\_OFST)

#define HW\_REGS\_SPAN (0x04000000)

#define HW\_REGS\_MASK (HW\_REGS\_SPAN - 1)

#define BUFF\_SPAN (800 \* 480 \* 4 \* 2)

#define H2F\_MASK (H2F\_SPAN - 1)

#define IORD(base, index) (\*(((uint32\_t \*)base)+index))

#define IOWR(base, index, data) (\*(((uint32\_t \*)base)+index) = data)

#define REG\_CONTROL 0

#define REG\_STATUS 1

#define REG\_MEM\_ADDR 2

#define REG\_FRAME\_DIM 3

#define REG\_DETECTED\_FRAME\_DIM 4

// bit mask for CONTROL

#define CONTROL\_CAPTURE\_BIT 0x01

#define CONTROL\_DUMMY\_DATA\_BIT 0x02

#define CONTROL\_AUTO\_FRAME\_DIM\_BIT 0x04

// bit mask for STATUS

#define STATUS\_DONE\_BIT 0x01

#define STATUS\_FIFO\_FULL\_BIT 0x02

#define STATUS\_INVALID\_FRAME\_BIT 0x04

cv::D8MCapture::D8MCapture()

{

opened = false;

frame\_index = 0;

}

cv::D8MCapture::D8MCapture(uint32\_t capture\_base, const char \*capture\_ram\_device)

{

opened = false;

frame\_index = 0;

open(capture\_base, capture\_ram\_device);

}

cv::D8MCapture::~D8MCapture()

{

release();

}

bool cv::D8MCapture::retrieve(OutputArray image, int flag)

{

bool bDone = false;

int width, height;

uint32\_t value;

value = IORD(capture\_controller, REG\_DETECTED\_FRAME\_DIM);

width = (value >> 16) & 0xFFFF;

height = value & 0xFFFF;

//printf("width:%d, height:%d\r\n", width, height);

bDone = true;

image.create(480, 800, CV\_8UC4);

Mat src = image.getMat();

if (frame\_index == 0)

memcpy(src.ptr(), capture\_sdram1, width \* height \* 4);

else

memcpy(src.ptr(), capture\_sdram2, width \* height \* 4);

return bDone;

}

bool cv::D8MCapture::wait\_done(int timeout\_s)

{

bool bDone = false;

int i = timeout\_s \* 1000 \* 1000 / 100;

uint32\_t status;

while (!bDone && i >= 0) {

status = IORD(capture\_controller, REG\_STATUS);

if ((status & STATUS\_DONE\_BIT) == STATUS\_DONE\_BIT) {

//printf("done, status=%xh\r\n", status);

bDone = true;

}

usleep(100);

i--;

}

capture\_status = status;

return bDone;

}

bool cv::D8MCapture::grab()

{

bool bDone = false;

bDone = wait\_done(1);

if (frame\_index == 0) {

IOWR(capture\_controller, REG\_MEM\_ADDR, capture\_sdram\_base);

frame\_index = 1;

} else {

IOWR(capture\_controller, REG\_MEM\_ADDR,

capture\_sdram\_base + (800 \* 480 \* 4));

frame\_index = 0;

}

start\_capture();

return bDone;

}

void cv::D8MCapture::start\_capture()

{

uint32\_t Command;

Command = 0;

Command |= CONTROL\_CAPTURE\_BIT | CONTROL\_AUTO\_FRAME\_DIM\_BIT;

IOWR(capture\_controller, REG\_CONTROL, Command);

Command &= ~CONTROL\_CAPTURE\_BIT;

IOWR(capture\_controller, REG\_CONTROL, Command);

Command |= CONTROL\_CAPTURE\_BIT;

IOWR(capture\_controller, REG\_CONTROL, Command);

}

bool cv::D8MCapture::read(OutputArray image)

{

bool bSuccess = true;

if (bSuccess == true)

bSuccess = grab();

if (bSuccess == true)

bSuccess = retrieve(image);

return bSuccess;

}

bool cv::D8MCapture::isOpened()

{

return opened;

}

bool cv::D8MCapture::open(uint32\_t capture\_base, const char \*capture\_ram\_device)

{

int fd;

fd = ::open(capture\_ram\_device,O\_RDWR);

if (fd < 0) {

printf("ERROR: could not open %s...\n", capture\_ram\_device);

return false;

}

if (ioctl(fd, 0, &capture\_sdram\_base) != 0) {

printf("ERROR: could not read buffer phy...\n");

close(fd);

return false;

}

close(fd);

if ((mem\_fd = ::open("/dev/mem", (O\_RDWR | O\_SYNC))) == -1) {

printf("ERROR: could not open \"/dev/mem\"...\n");

return (1);

}

h2f\_lw\_virtual\_base = mmap(NULL, HW\_REGS\_SPAN, (PROT\_READ | PROT\_WRITE),

MAP\_SHARED, mem\_fd, HW\_REGS\_BASE);

if (h2f\_lw\_virtual\_base == MAP\_FAILED) {

printf("ERROR: mmap() failed...\n");

close(mem\_fd);

return false;

}

capture\_sdram1 = (uint8\_t \*) mmap(NULL, BUFF\_SPAN, (PROT\_READ | PROT\_WRITE),

MAP\_SHARED, mem\_fd, capture\_sdram\_base);

if (capture\_sdram1 == MAP\_FAILED) {

printf("ERROR: axi mmap() failed...\n");

close(mem\_fd);

return false;

}

capture\_sdram2 = capture\_sdram1 + (800 \* 480 \* 4);

capture\_controller = (uint32\_t\*) ((uint8\_t\*) h2f\_lw\_virtual\_base

+ ((ALT\_LWFPGASLVS\_OFST + capture\_base) & HW\_REGS\_MASK));

IOWR(capture\_controller, REG\_MEM\_ADDR, capture\_sdram\_base);

start\_capture();

opened = true;

return true;

}

void cv::D8MCapture::release()

{

if (munmap(capture\_sdram1, BUFF\_SPAN) != 0) {

printf("ERROR: munmap() failed...\n");

}

if (munmap(h2f\_lw\_virtual\_base, HW\_REGS\_SPAN) != 0) {

printf("ERROR: munmap() fggailed...\n");

}

close(mem\_fd);

opened = false;

}

9.7 D8MCapture.h

#ifndef D8MCAPTURE\_H\_

#define D8MCAPTURE\_H\_

#include <stdint.h>

#include <opencv2/opencv.hpp>

namespace cv {

class D8MCapture {

public:

D8MCapture();

D8MCapture(uint32\_t capture\_base, const char \*capture\_ram\_device);

bool grab();

bool isOpened();

bool open(uint32\_t capture\_base, const char \*capture\_ram\_device);

bool read(OutputArray image);

bool retrieve(OutputArray image, int flag = 0);

void release();

virtual ~D8MCapture();

//D8MCapture & operator>>(Mat &image);

private:

bool opened;

int mem\_fd;

void \*h2f\_lw\_virtual\_base;

uint32\_t \*capture\_controller = NULL;

int frame\_index;

uint8\_t \*capture\_sdram1 = NULL;

uint8\_t \*capture\_sdram2 = NULL;

uint32\_t capture\_sdram\_base;

uint32\_t capture\_status;

bool wait\_done(int timeout\_s);

void start\_capture();

};

} /\* namespace cv \*/

#endif /\* D8MCAPTURE\_H\_ \*/

9.8 hps.h

#ifndef \_ALTERA\_HPS\_0\_H\_

#define \_ALTERA\_HPS\_0\_H\_

/\*

\* This file was automatically generated by the swinfo2header utility.

\*

\* Created from SOPC Builder system 'soc\_system' in

\* file './soc\_system.sopcinfo'.

\*/

/\*

\* This file contains macros for module 'hps\_0' and devices

\* connected to the following masters:

\* h2f\_axi\_master

\* h2f\_lw\_axi\_master

\*

\* Do not include this header file and another header file created for a

\* different module or master group at the same time.

\* Doing so may result in duplicate macro names.

\* Instead, use the system header file which has macros with unique names.

\*/

/\*

\* Macros for device 'sysid\_qsys', class 'altera\_avalon\_sysid\_qsys'

\* The macros are prefixed with 'SYSID\_QSYS\_'.

\* The prefix is the slave descriptor.

\*/

#define SYSID\_QSYS\_COMPONENT\_TYPE altera\_avalon\_sysid\_qsys

#define SYSID\_QSYS\_COMPONENT\_NAME sysid\_qsys

#define SYSID\_QSYS\_BASE 0x1000

#define SYSID\_QSYS\_SPAN 8

#define SYSID\_QSYS\_END 0x1007

#define SYSID\_QSYS\_ID 2899645186

#define SYSID\_QSYS\_TIMESTAMP 1492071627

/\*

\* Macros for device 'jtag\_uart', class 'altera\_avalon\_jtag\_uart'

\* The macros are prefixed with 'JTAG\_UART\_'.

\* The prefix is the slave descriptor.

\*/

#define JTAG\_UART\_COMPONENT\_TYPE altera\_avalon\_jtag\_uart

#define JTAG\_UART\_COMPONENT\_NAME jtag\_uart

#define JTAG\_UART\_BASE 0x2000

#define JTAG\_UART\_SPAN 8

#define JTAG\_UART\_END 0x2007

#define JTAG\_UART\_READ\_DEPTH 512

#define JTAG\_UART\_READ\_THRESHOLD 8

#define JTAG\_UART\_WRITE\_DEPTH 512

#define JTAG\_UART\_WRITE\_THRESHOLD 8

/\*

\* Macros for device 'ledr', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'LEDR\_'.

\* The prefix is the slave descriptor.

\*/

#define LEDR\_COMPONENT\_TYPE altera\_avalon\_pio

#define LEDR\_COMPONENT\_NAME ledr

#define LEDR\_BASE 0x3000

#define LEDR\_SPAN 16

#define LEDR\_END 0x300f

#define LEDR\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define LEDR\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define LEDR\_CAPTURE 0

#define LEDR\_DATA\_WIDTH 10

#define LEDR\_DO\_TEST\_BENCH\_WIRING 0

#define LEDR\_DRIVEN\_SIM\_VALUE 0

#define LEDR\_EDGE\_TYPE NONE

#define LEDR\_FREQ 50000000

#define LEDR\_HAS\_IN 0

#define LEDR\_HAS\_OUT 1

#define LEDR\_HAS\_TRI 0

#define LEDR\_IRQ\_TYPE NONE

#define LEDR\_RESET\_VALUE 0

/\*

\* Macros for device 'sw', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'SW\_'.

\* The prefix is the slave descriptor.

\*/

#define SW\_COMPONENT\_TYPE altera\_avalon\_pio

#define SW\_COMPONENT\_NAME sw

#define SW\_BASE 0x4000

#define SW\_SPAN 16

#define SW\_END 0x400f

#define SW\_IRQ 1

#define SW\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define SW\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define SW\_CAPTURE 1

#define SW\_DATA\_WIDTH 10

#define SW\_DO\_TEST\_BENCH\_WIRING 0

#define SW\_DRIVEN\_SIM\_VALUE 0

#define SW\_EDGE\_TYPE ANY

#define SW\_FREQ 50000000

#define SW\_HAS\_IN 1

#define SW\_HAS\_OUT 0

#define SW\_HAS\_TRI 0

#define SW\_IRQ\_TYPE EDGE

#define SW\_RESET\_VALUE 0

/\*

\* Macros for device 'key', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'KEY\_'.

\* The prefix is the slave descriptor.

\*/

#define KEY\_COMPONENT\_TYPE altera\_avalon\_pio

#define KEY\_COMPONENT\_NAME key

#define KEY\_BASE 0x5000

#define KEY\_SPAN 16

#define KEY\_END 0x500f

#define KEY\_IRQ 0

#define KEY\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define KEY\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define KEY\_CAPTURE 1

#define KEY\_DATA\_WIDTH 4

#define KEY\_DO\_TEST\_BENCH\_WIRING 0

#define KEY\_DRIVEN\_SIM\_VALUE 0

#define KEY\_EDGE\_TYPE ANY

#define KEY\_FREQ 50000000

#define KEY\_HAS\_IN 1

#define KEY\_HAS\_OUT 0

#define KEY\_HAS\_TRI 0

#define KEY\_IRQ\_TYPE EDGE

#define KEY\_RESET\_VALUE 0

/\*

\* Macros for device 'alt\_vip\_vfr\_vga', class 'alt\_vip\_vfr'

\* The macros are prefixed with 'ALT\_VIP\_VFR\_VGA\_'.

\* The prefix is the slave descriptor.

\*/

#define ALT\_VIP\_VFR\_VGA\_COMPONENT\_TYPE alt\_vip\_vfr

#define ALT\_VIP\_VFR\_VGA\_COMPONENT\_NAME alt\_vip\_vfr\_vga

#define ALT\_VIP\_VFR\_VGA\_BASE 0x31000

#define ALT\_VIP\_VFR\_VGA\_SPAN 128

#define ALT\_VIP\_VFR\_VGA\_END 0x3107f

/\*

\* Macros for device 'alt\_vip\_cl\_mixer\_0', class 'alt\_vip\_cl\_mixer'

\* The macros are prefixed with 'ALT\_VIP\_CL\_MIXER\_0\_'.

\* The prefix is the slave descriptor.

\*/

#define ALT\_VIP\_CL\_MIXER\_0\_COMPONENT\_TYPE alt\_vip\_cl\_mixer

#define ALT\_VIP\_CL\_MIXER\_0\_COMPONENT\_NAME alt\_vip\_cl\_mixer\_0

#define ALT\_VIP\_CL\_MIXER\_0\_BASE 0x32000

#define ALT\_VIP\_CL\_MIXER\_0\_SPAN 512

#define ALT\_VIP\_CL\_MIXER\_0\_END 0x321ff

/\*

\* Macros for device 'seg7', class 'TERASIC\_SEG7'

\* The macros are prefixed with 'SEG7\_'.

\* The prefix is the slave descriptor.

\*/

#define SEG7\_COMPONENT\_TYPE TERASIC\_SEG7

#define SEG7\_COMPONENT\_NAME seg7

#define SEG7\_BASE 0x33000

#define SEG7\_SPAN 32

#define SEG7\_END 0x3301f

/\*

\* Macros for device 'ir\_rx', class 'TERASIC\_IR\_RX\_FIFO'

\* The macros are prefixed with 'IR\_RX\_'.

\* The prefix is the slave descriptor.

\*/

#define IR\_RX\_COMPONENT\_TYPE TERASIC\_IR\_RX\_FIFO

#define IR\_RX\_COMPONENT\_NAME ir\_rx

#define IR\_RX\_BASE 0x34000

#define IR\_RX\_SPAN 8

#define IR\_RX\_END 0x34007

/\*

\* Macros for device 'spi', class 'altera\_avalon\_spi'

\* The macros are prefixed with 'SPI\_'.

\* The prefix is the slave descriptor.

\*/

#define SPI\_COMPONENT\_TYPE altera\_avalon\_spi

#define SPI\_COMPONENT\_NAME spi

#define SPI\_BASE 0x35000

#define SPI\_SPAN 32

#define SPI\_END 0x3501f

#define SPI\_IRQ 2

#define SPI\_CLOCKMULT 1

#define SPI\_CLOCKPHASE 0

#define SPI\_CLOCKPOLARITY 1

#define SPI\_CLOCKUNITS "Hz"

#define SPI\_DATABITS 16

#define SPI\_DATAWIDTH 16

#define SPI\_DELAYMULT "1.0E-9"

#define SPI\_DELAYUNITS "ns"

#define SPI\_EXTRADELAY 0

#define SPI\_INSERT\_SYNC 0

#define SPI\_ISMASTER 1

#define SPI\_LSBFIRST 0

#define SPI\_NUMSLAVES 1

#define SPI\_PREFIX "spi\_"

#define SPI\_SYNC\_REG\_DEPTH 2

#define SPI\_TARGETCLOCK 20000000

#define SPI\_TARGETSSDELAY "0.0"

/\*

\* Macros for device 'tv\_decoder\_alt\_vip\_cl\_cvi\_0', class 'alt\_vip\_cl\_cvi'

\* The macros are prefixed with 'TV\_DECODER\_ALT\_VIP\_CL\_CVI\_0\_'.

\* The prefix is the slave descriptor.

\*/

#define TV\_DECODER\_ALT\_VIP\_CL\_CVI\_0\_COMPONENT\_TYPE alt\_vip\_cl\_cvi

#define TV\_DECODER\_ALT\_VIP\_CL\_CVI\_0\_COMPONENT\_NAME tv\_decoder\_alt\_vip\_cl\_cvi\_0

#define TV\_DECODER\_ALT\_VIP\_CL\_CVI\_0\_BASE 0x36000

#define TV\_DECODER\_ALT\_VIP\_CL\_CVI\_0\_SPAN 128

#define TV\_DECODER\_ALT\_VIP\_CL\_CVI\_0\_END 0x3607f

/\*

\* Macros for device 'tv\_decoder\_alt\_vip\_cl\_scl\_1', class 'alt\_vip\_cl\_scl'

\* The macros are prefixed with 'TV\_DECODER\_ALT\_VIP\_CL\_SCL\_1\_'.

\* The prefix is the slave descriptor.

\*/

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_1\_COMPONENT\_TYPE alt\_vip\_cl\_scl

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_1\_COMPONENT\_NAME tv\_decoder\_alt\_vip\_cl\_scl\_1

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_1\_BASE 0x37000

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_1\_SPAN 512

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_1\_END 0x371ff

/\*

\* Macros for device 'audio', class 'TERASIC\_AUDIO\_WM8731'

\* The macros are prefixed with 'AUDIO\_'.

\* The prefix is the slave descriptor.

\*/

#define AUDIO\_COMPONENT\_TYPE TERASIC\_AUDIO\_WM8731

#define AUDIO\_COMPONENT\_NAME audio

#define AUDIO\_BASE 0x38000

#define AUDIO\_SPAN 16

#define AUDIO\_END 0x3800f

/\*

\* Macros for device 'tv\_decoder\_alt\_vip\_cl\_scl\_0', class 'alt\_vip\_cl\_scl'

\* The macros are prefixed with 'TV\_DECODER\_ALT\_VIP\_CL\_SCL\_0\_'.

\* The prefix is the slave descriptor.

\*/

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_0\_COMPONENT\_TYPE alt\_vip\_cl\_scl

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_0\_COMPONENT\_NAME tv\_decoder\_alt\_vip\_cl\_scl\_0

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_0\_BASE 0x39000

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_0\_SPAN 512

#define TV\_DECODER\_ALT\_VIP\_CL\_SCL\_0\_END 0x391ff

/\*

\* Macros for device 'tv\_decoder\_alt\_vip\_cl\_swi\_0', class 'alt\_vip\_cl\_swi'

\* The macros are prefixed with 'TV\_DECODER\_ALT\_VIP\_CL\_SWI\_0\_'.

\* The prefix is the slave descriptor.

\*/

#define TV\_DECODER\_ALT\_VIP\_CL\_SWI\_0\_COMPONENT\_TYPE alt\_vip\_cl\_swi

#define TV\_DECODER\_ALT\_VIP\_CL\_SWI\_0\_COMPONENT\_NAME tv\_decoder\_alt\_vip\_cl\_swi\_0

#define TV\_DECODER\_ALT\_VIP\_CL\_SWI\_0\_BASE 0x3a000

#define TV\_DECODER\_ALT\_VIP\_CL\_SWI\_0\_SPAN 128

#define TV\_DECODER\_ALT\_VIP\_CL\_SWI\_0\_END 0x3a07f

/\*

\* Macros for device 'tv\_decoder\_TERASIC\_STREAM\_CAPTURE', class 'TERASIC\_STREAM\_CAPTURE'

\* The macros are prefixed with 'TV\_DECODER\_TERASIC\_STREAM\_CAPTURE\_'.

\* The prefix is the slave descriptor.

\*/

#define TV\_DECODER\_TERASIC\_STREAM\_CAPTURE\_COMPONENT\_TYPE TERASIC\_STREAM\_CAPTURE

#define TV\_DECODER\_TERASIC\_STREAM\_CAPTURE\_COMPONENT\_NAME tv\_decoder\_TERASIC\_STREAM\_CAPTURE

#define TV\_DECODER\_TERASIC\_STREAM\_CAPTURE\_BASE 0x3b000

#define TV\_DECODER\_TERASIC\_STREAM\_CAPTURE\_SPAN 32

#define TV\_DECODER\_TERASIC\_STREAM\_CAPTURE\_END 0x3b01f

/\*

\* Macros for device 'ts\_i2c', class 'i2c\_opencores'

\* The macros are prefixed with 'TS\_I2C\_'.

\* The prefix is the slave descriptor.

\*/

#define TS\_I2C\_COMPONENT\_TYPE i2c\_opencores

#define TS\_I2C\_COMPONENT\_NAME ts\_i2c

#define TS\_I2C\_BASE 0x50000

#define TS\_I2C\_SPAN 32

#define TS\_I2C\_END 0x5001f

#define TS\_I2C\_IRQ 3

/\*

\* Macros for device 'ts\_interrupt', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'TS\_INTERRUPT\_'.

\* The prefix is the slave descriptor.

\*/

#define TS\_INTERRUPT\_COMPONENT\_TYPE altera\_avalon\_pio

#define TS\_INTERRUPT\_COMPONENT\_NAME ts\_interrupt

#define TS\_INTERRUPT\_BASE 0x51000

#define TS\_INTERRUPT\_SPAN 16

#define TS\_INTERRUPT\_END 0x5100f

#define TS\_INTERRUPT\_IRQ 4

#define TS\_INTERRUPT\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define TS\_INTERRUPT\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define TS\_INTERRUPT\_CAPTURE 1

#define TS\_INTERRUPT\_DATA\_WIDTH 1

#define TS\_INTERRUPT\_DO\_TEST\_BENCH\_WIRING 0

#define TS\_INTERRUPT\_DRIVEN\_SIM\_VALUE 0

#define TS\_INTERRUPT\_EDGE\_TYPE FALLING

#define TS\_INTERRUPT\_FREQ 50000000

#define TS\_INTERRUPT\_HAS\_IN 1

#define TS\_INTERRUPT\_HAS\_OUT 0

#define TS\_INTERRUPT\_HAS\_TRI 0

#define TS\_INTERRUPT\_IRQ\_TYPE EDGE

#define TS\_INTERRUPT\_RESET\_VALUE 0

/\*

\* Macros for device 'i2c\_opencores\_camera', class 'i2c\_opencores'

\* The macros are prefixed with 'I2C\_OPENCORES\_CAMERA\_'.

\* The prefix is the slave descriptor.

\*/

#define I2C\_OPENCORES\_CAMERA\_COMPONENT\_TYPE i2c\_opencores

#define I2C\_OPENCORES\_CAMERA\_COMPONENT\_NAME i2c\_opencores\_camera

#define I2C\_OPENCORES\_CAMERA\_BASE 0x52000

#define I2C\_OPENCORES\_CAMERA\_SPAN 32

#define I2C\_OPENCORES\_CAMERA\_END 0x5201f

#define I2C\_OPENCORES\_CAMERA\_IRQ 5

/\*

\* Macros for device 'camera\_pwdn\_n', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'CAMERA\_PWDN\_N\_'.

\* The prefix is the slave descriptor.

\*/

#define CAMERA\_PWDN\_N\_COMPONENT\_TYPE altera\_avalon\_pio

#define CAMERA\_PWDN\_N\_COMPONENT\_NAME camera\_pwdn\_n

#define CAMERA\_PWDN\_N\_BASE 0x53000

#define CAMERA\_PWDN\_N\_SPAN 16

#define CAMERA\_PWDN\_N\_END 0x5300f

#define CAMERA\_PWDN\_N\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define CAMERA\_PWDN\_N\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define CAMERA\_PWDN\_N\_CAPTURE 0

#define CAMERA\_PWDN\_N\_DATA\_WIDTH 1

#define CAMERA\_PWDN\_N\_DO\_TEST\_BENCH\_WIRING 0

#define CAMERA\_PWDN\_N\_DRIVEN\_SIM\_VALUE 0

#define CAMERA\_PWDN\_N\_EDGE\_TYPE NONE

#define CAMERA\_PWDN\_N\_FREQ 50000000

#define CAMERA\_PWDN\_N\_HAS\_IN 0

#define CAMERA\_PWDN\_N\_HAS\_OUT 1

#define CAMERA\_PWDN\_N\_HAS\_TRI 0

#define CAMERA\_PWDN\_N\_IRQ\_TYPE NONE

#define CAMERA\_PWDN\_N\_RESET\_VALUE 1

/\*

\* Macros for device 'i2c\_opencores\_mipi', class 'i2c\_opencores'

\* The macros are prefixed with 'I2C\_OPENCORES\_MIPI\_'.

\* The prefix is the slave descriptor.

\*/

#define I2C\_OPENCORES\_MIPI\_COMPONENT\_TYPE i2c\_opencores

#define I2C\_OPENCORES\_MIPI\_COMPONENT\_NAME i2c\_opencores\_mipi

#define I2C\_OPENCORES\_MIPI\_BASE 0x54000

#define I2C\_OPENCORES\_MIPI\_SPAN 32

#define I2C\_OPENCORES\_MIPI\_END 0x5401f

#define I2C\_OPENCORES\_MIPI\_IRQ 6

/\*

\* Macros for device 'mipi\_reset\_n', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'MIPI\_RESET\_N\_'.

\* The prefix is the slave descriptor.

\*/

#define MIPI\_RESET\_N\_COMPONENT\_TYPE altera\_avalon\_pio

#define MIPI\_RESET\_N\_COMPONENT\_NAME mipi\_reset\_n

#define MIPI\_RESET\_N\_BASE 0x55000

#define MIPI\_RESET\_N\_SPAN 16

#define MIPI\_RESET\_N\_END 0x5500f

#define MIPI\_RESET\_N\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define MIPI\_RESET\_N\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define MIPI\_RESET\_N\_CAPTURE 0

#define MIPI\_RESET\_N\_DATA\_WIDTH 1

#define MIPI\_RESET\_N\_DO\_TEST\_BENCH\_WIRING 0

#define MIPI\_RESET\_N\_DRIVEN\_SIM\_VALUE 0

#define MIPI\_RESET\_N\_EDGE\_TYPE NONE

#define MIPI\_RESET\_N\_FREQ 50000000

#define MIPI\_RESET\_N\_HAS\_IN 0

#define MIPI\_RESET\_N\_HAS\_OUT 1

#define MIPI\_RESET\_N\_HAS\_TRI 0

#define MIPI\_RESET\_N\_IRQ\_TYPE NONE

#define MIPI\_RESET\_N\_RESET\_VALUE 1

/\*

\* Macros for device 'spi\_mpu', class 'altera\_avalon\_spi'

\* The macros are prefixed with 'SPI\_MPU\_'.

\* The prefix is the slave descriptor.

\*/

#define SPI\_MPU\_COMPONENT\_TYPE altera\_avalon\_spi

#define SPI\_MPU\_COMPONENT\_NAME spi\_mpu

#define SPI\_MPU\_BASE 0x56000

#define SPI\_MPU\_SPAN 32

#define SPI\_MPU\_END 0x5601f

#define SPI\_MPU\_IRQ 7

#define SPI\_MPU\_CLOCKMULT 1

#define SPI\_MPU\_CLOCKPHASE 1

#define SPI\_MPU\_CLOCKPOLARITY 1

#define SPI\_MPU\_CLOCKUNITS "Hz"

#define SPI\_MPU\_DATABITS 8

#define SPI\_MPU\_DATAWIDTH 16

#define SPI\_MPU\_DELAYMULT "1.0E-9"

#define SPI\_MPU\_DELAYUNITS "ns"

#define SPI\_MPU\_EXTRADELAY 0

#define SPI\_MPU\_INSERT\_SYNC 0

#define SPI\_MPU\_ISMASTER 1

#define SPI\_MPU\_LSBFIRST 0

#define SPI\_MPU\_NUMSLAVES 1

#define SPI\_MPU\_PREFIX "spi\_"

#define SPI\_MPU\_SYNC\_REG\_DEPTH 2

#define SPI\_MPU\_TARGETCLOCK 1000000

#define SPI\_MPU\_TARGETSSDELAY "0.0"

/\*

\* Macros for device 'mpu\_int', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'MPU\_INT\_'.

\* The prefix is the slave descriptor.

\*/

#define MPU\_INT\_COMPONENT\_TYPE altera\_avalon\_pio

#define MPU\_INT\_COMPONENT\_NAME mpu\_int

#define MPU\_INT\_BASE 0x57000

#define MPU\_INT\_SPAN 16

#define MPU\_INT\_END 0x5700f

#define MPU\_INT\_IRQ 9

#define MPU\_INT\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define MPU\_INT\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define MPU\_INT\_CAPTURE 1

#define MPU\_INT\_DATA\_WIDTH 1

#define MPU\_INT\_DO\_TEST\_BENCH\_WIRING 0

#define MPU\_INT\_DRIVEN\_SIM\_VALUE 0

#define MPU\_INT\_EDGE\_TYPE RISING

#define MPU\_INT\_FREQ 50000000

#define MPU\_INT\_HAS\_IN 1

#define MPU\_INT\_HAS\_OUT 0

#define MPU\_INT\_HAS\_TRI 0

#define MPU\_INT\_IRQ\_TYPE EDGE

#define MPU\_INT\_RESET\_VALUE 0

/\*

\* Macros for device 'i2c\_opencores\_light', class 'i2c\_opencores'

\* The macros are prefixed with 'I2C\_OPENCORES\_LIGHT\_'.

\* The prefix is the slave descriptor.

\*/

#define I2C\_OPENCORES\_LIGHT\_COMPONENT\_TYPE i2c\_opencores

#define I2C\_OPENCORES\_LIGHT\_COMPONENT\_NAME i2c\_opencores\_light

#define I2C\_OPENCORES\_LIGHT\_BASE 0x58000

#define I2C\_OPENCORES\_LIGHT\_SPAN 32

#define I2C\_OPENCORES\_LIGHT\_END 0x5801f

#define I2C\_OPENCORES\_LIGHT\_IRQ 8

/\*

\* Macros for device 'light\_int', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'LIGHT\_INT\_'.

\* The prefix is the slave descriptor.

\*/

#define LIGHT\_INT\_COMPONENT\_TYPE altera\_avalon\_pio

#define LIGHT\_INT\_COMPONENT\_NAME light\_int

#define LIGHT\_INT\_BASE 0x59000

#define LIGHT\_INT\_SPAN 16

#define LIGHT\_INT\_END 0x5900f

#define LIGHT\_INT\_IRQ 10

#define LIGHT\_INT\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define LIGHT\_INT\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define LIGHT\_INT\_CAPTURE 1

#define LIGHT\_INT\_DATA\_WIDTH 1

#define LIGHT\_INT\_DO\_TEST\_BENCH\_WIRING 0

#define LIGHT\_INT\_DRIVEN\_SIM\_VALUE 0

#define LIGHT\_INT\_EDGE\_TYPE RISING

#define LIGHT\_INT\_FREQ 50000000

#define LIGHT\_INT\_HAS\_IN 1

#define LIGHT\_INT\_HAS\_OUT 0

#define LIGHT\_INT\_HAS\_TRI 0

#define LIGHT\_INT\_IRQ\_TYPE EDGE

#define LIGHT\_INT\_RESET\_VALUE 0

#endif /\* \_ALTERA\_HPS\_0\_H\_ \*/

9.9 makefile

TARGET = camera\_in

ALT\_DEVICE\_FAMILY ?= soc\_cv\_av

HWLIBS\_ROOT = ../hwlib

CFLAGS = -ggdb -Wall -std=c++11 `pkg-config --cflags opencv` -I$(HWLIBS\_ROOT)/include -I$(HWLIBS\_ROOT)/include/$(ALT\_DEVICE\_FAMILY) -D$(ALT\_DEVICE\_FAMILY)

LDFLAGS = -ggdb -Wall -std=c++11 `pkg-config --libs opencv` -lpthread

//CROSS\_COMPILE = arm-linux-gnueabihf-

CC = $(CROSS\_COMPILE)g++

all: $(TARGET)

$(TARGET): $(TARGET).o D8MCapture.o

$(CC) -o $@ $^ $(LDFLAGS)

%.o: %.cpp

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

clean:

rm -f $(TARGET) \*.a \*.o \*~