Chopin

Piano Player with Virtual Keyboard

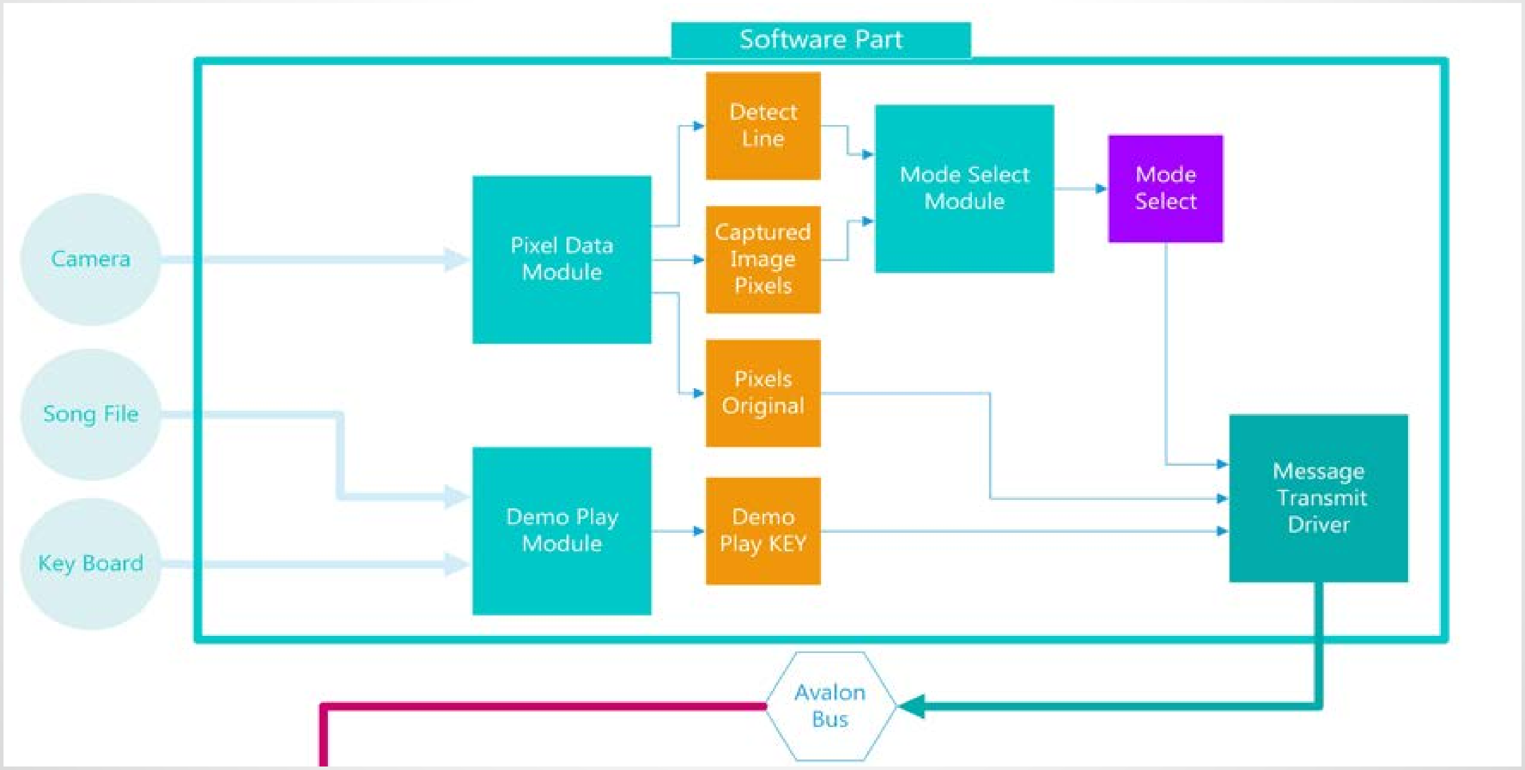
Daran Cai
Linjun Kuang
Wei Xia
Wenyuan Zhao

Overview

- This project implement a piano player with a virtual keyboard.
- The key feature is that we use a camera and a normal screen to realize a touch screen effect
- This project supports two play modes, one is free mode, the other is autoplay mode.
- Also, we support pressing two keys at the same time

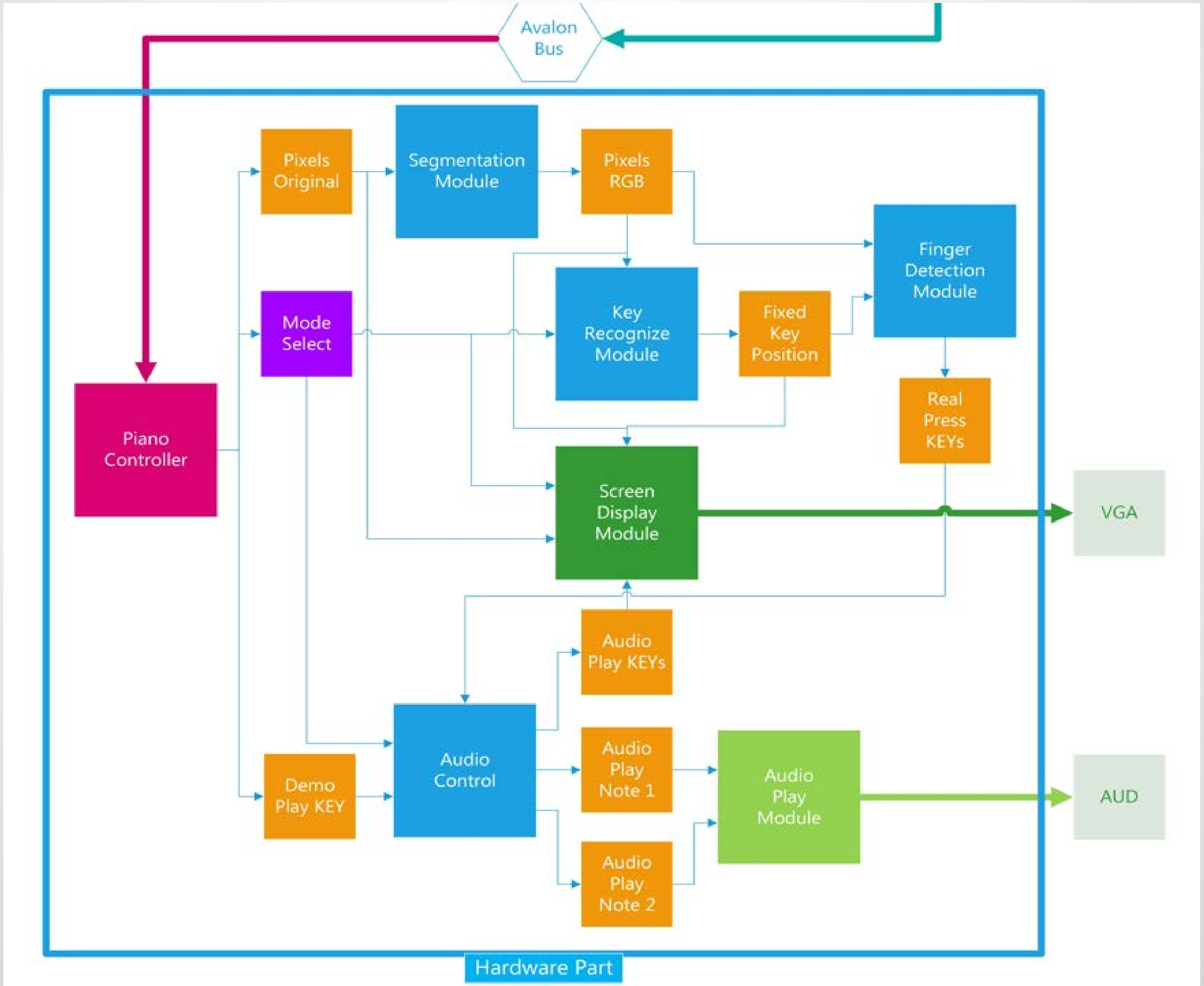
High level block diagram

Software Implementation



High level block diagram

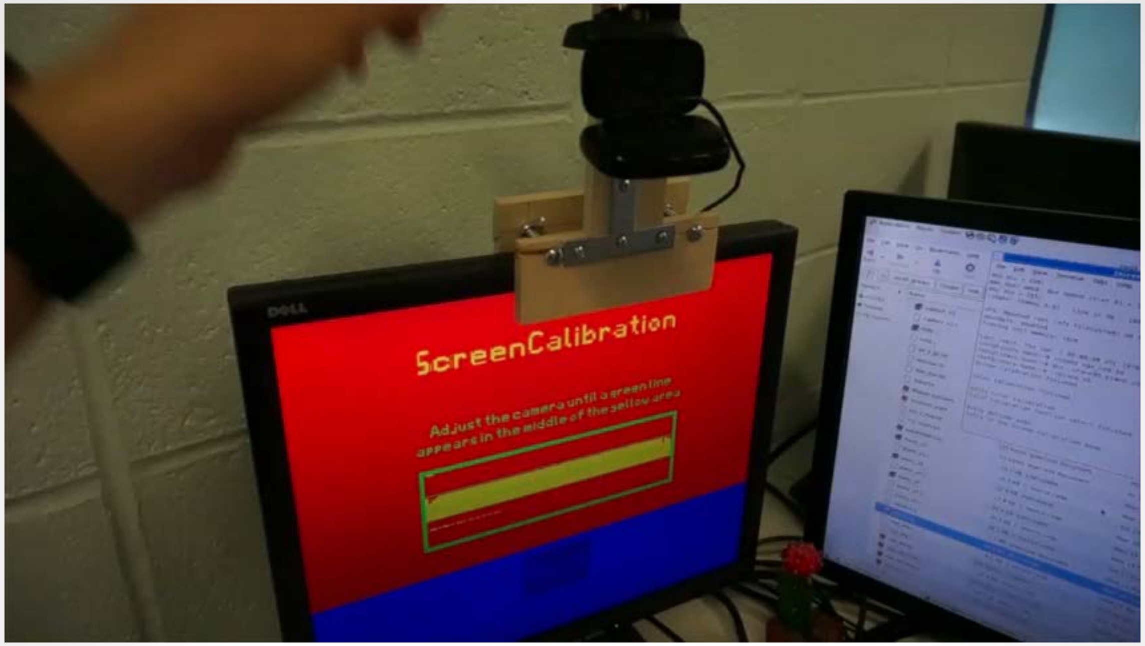
Hardware Implementation



Software vs Hardware

Software	Hardware
Kernel compilation Camera driver Calibration (Screen & Color) User Interface control Data transmission	Segmentation Image denoise Key recognition Finger detection Vga display • background • Keyboard display • Text display • Calibration display • Press feedback Audio (Karplus-Strong)

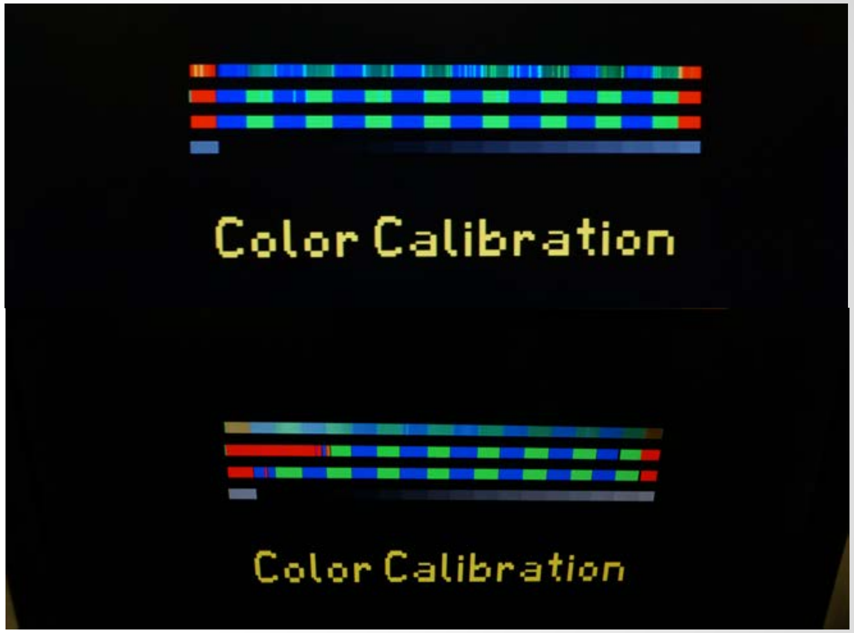
Screen Position Calibration

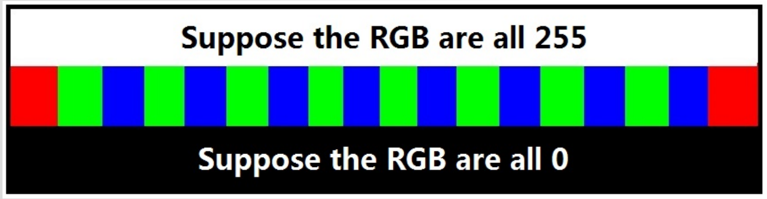

ScreenCalibration
Adjust the camera until a green line appears in the middle of the yellow area
DELL

In order to make the function works, we have to make sure the camera can capture the area for touch detection. Thus, we provide user a convenient way to calibrate the screen position.

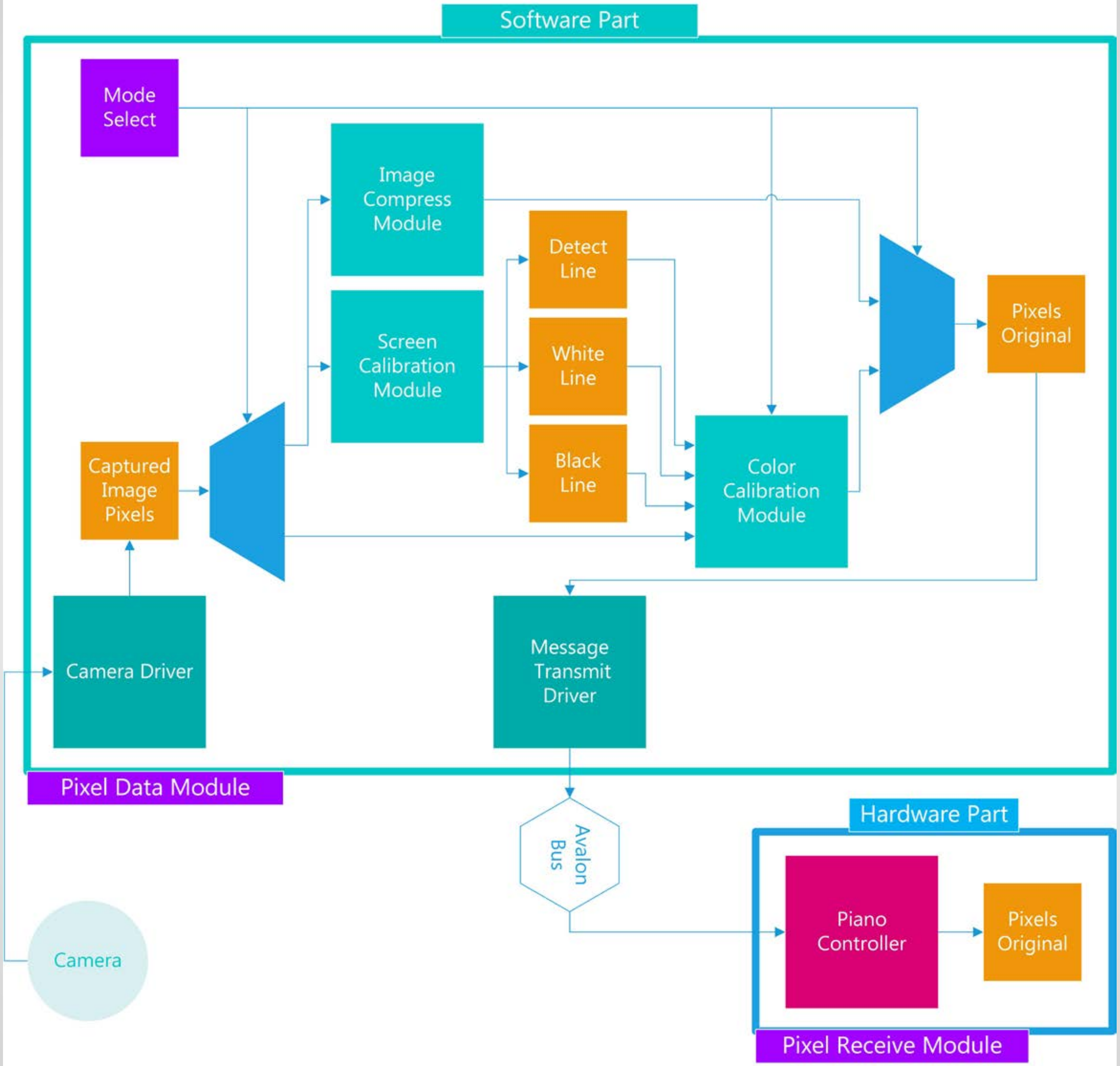
- Segment the captured image and display the touch area can be captured in screen.
- Give the visual feedback for the successful calibration.
- When the user calibrate the screen position successfully, the "Next" button will appeared.

Color calibration

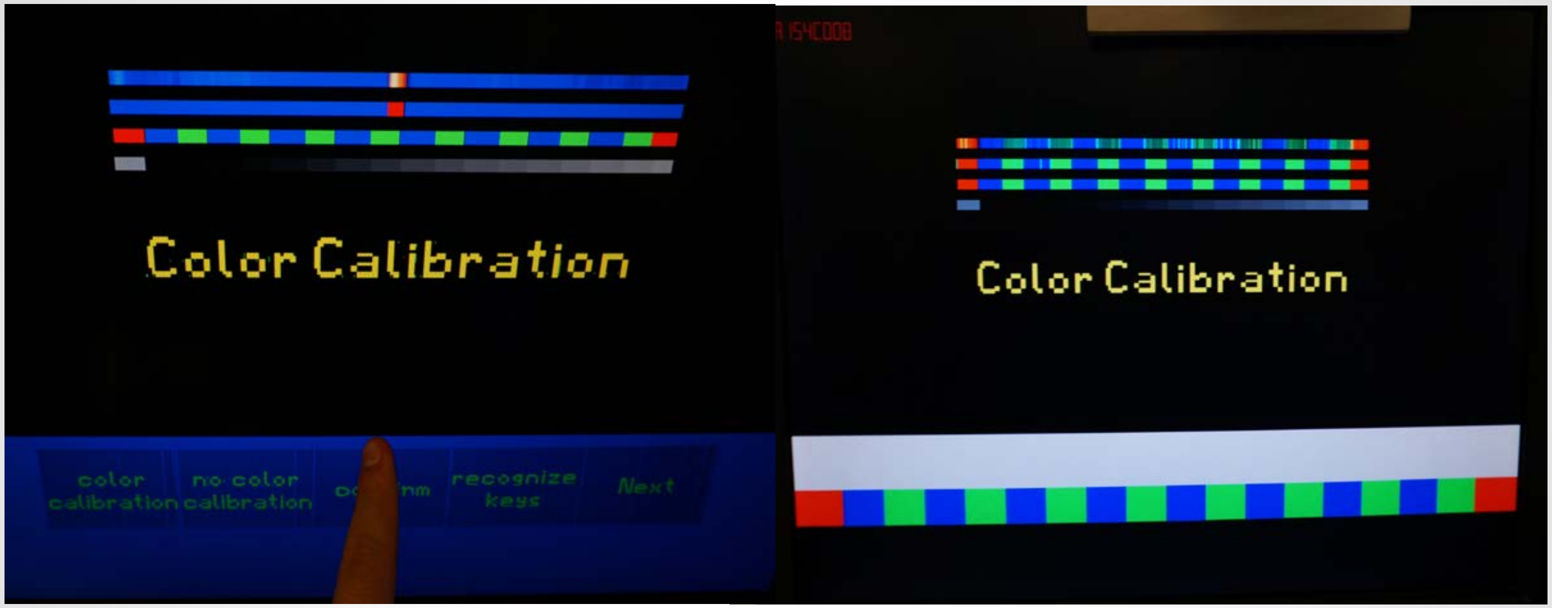




Due to the particular angle between camera and screen, the plane of the screen will reflect some light from the environment to the camera which will effect the color segmentation of the captured image. Therefore, an Algorithm for color Calibration is needed.

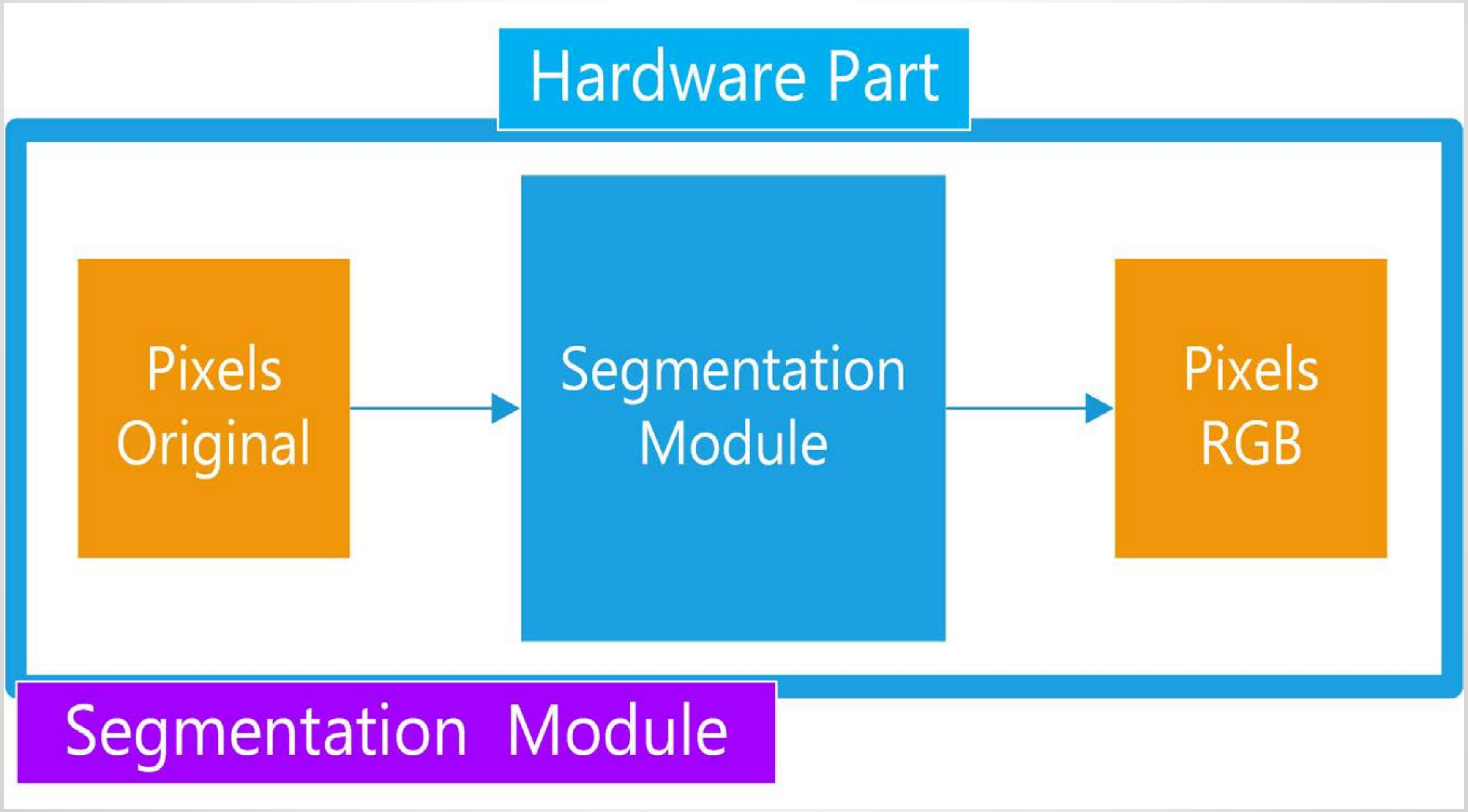

Software Part
Mode Select
Image Compress Module
Screen Calibration Module
Detect Line
White Line
Black Line
Color Calibration Module
Pixels Original
Captured Image Pixels
Camera Driver
Message Transmit Driver
Pixel Data Module
Avalon Bus
Hardware Part
Piano Controller
Pixels Original
Pixel Receive Module
Camera

Finger Detection & Keys Recognition

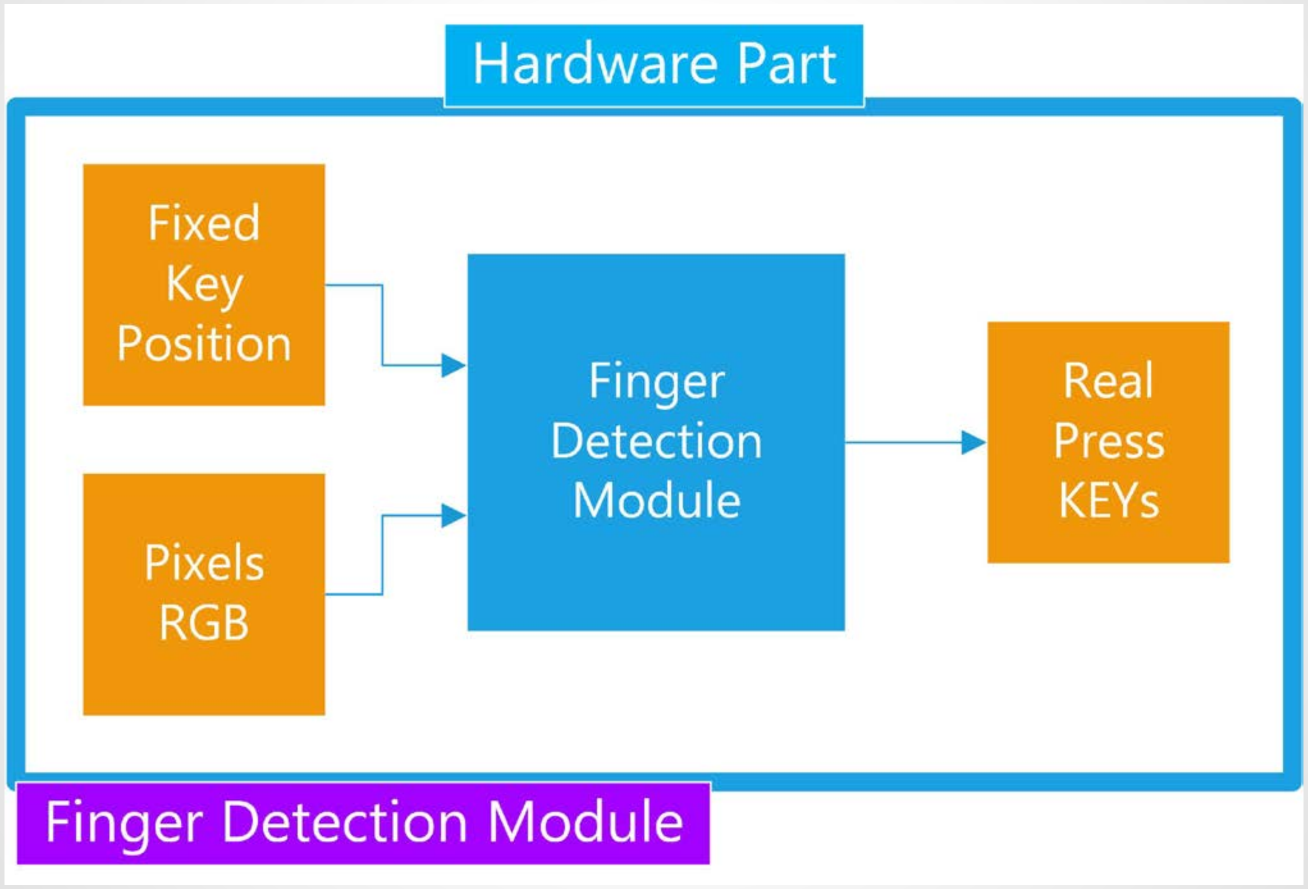


All Algorithm are implemented in hardware, which means the real circuit. In this way, we can provide a high accuracy performance and very short response time.

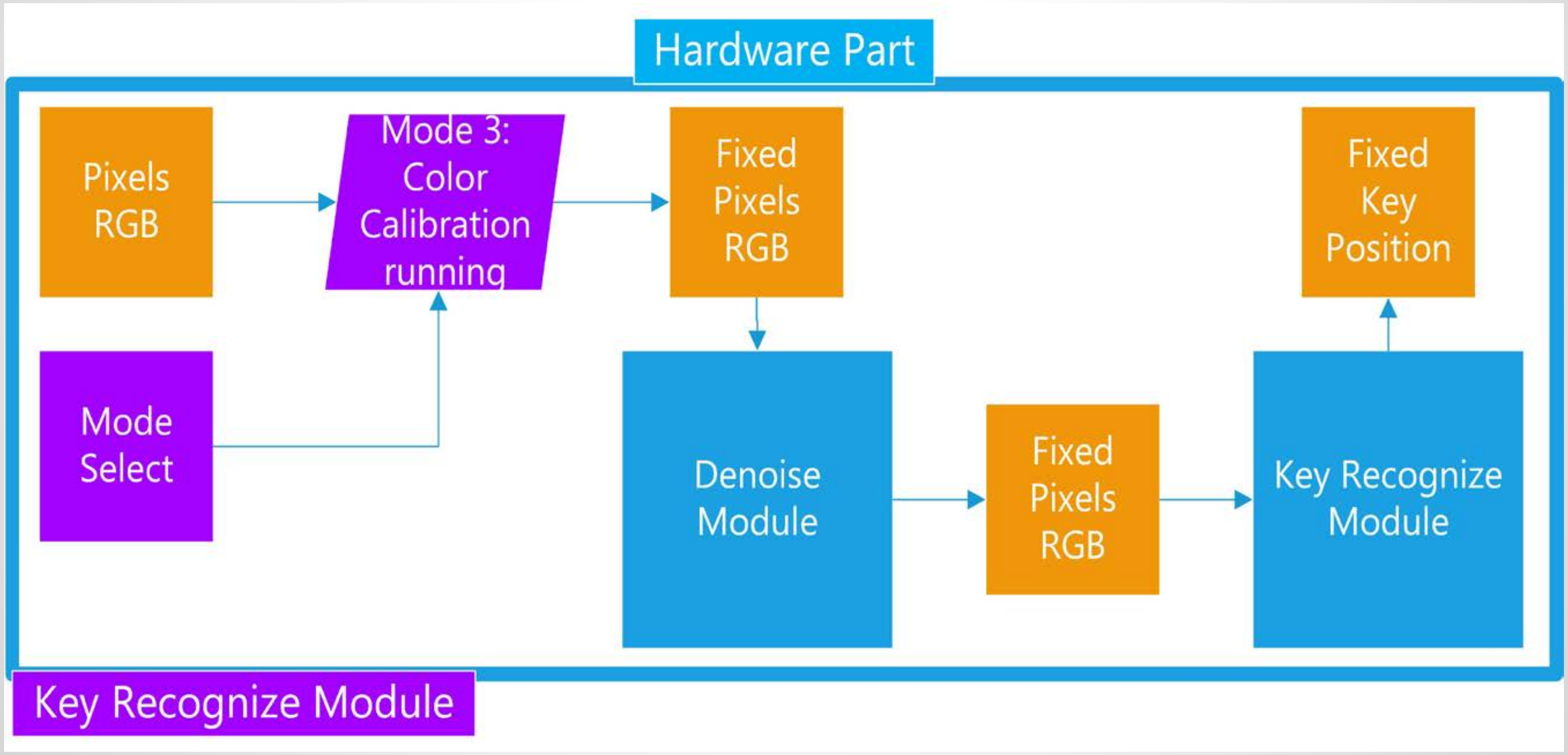
Segmentation



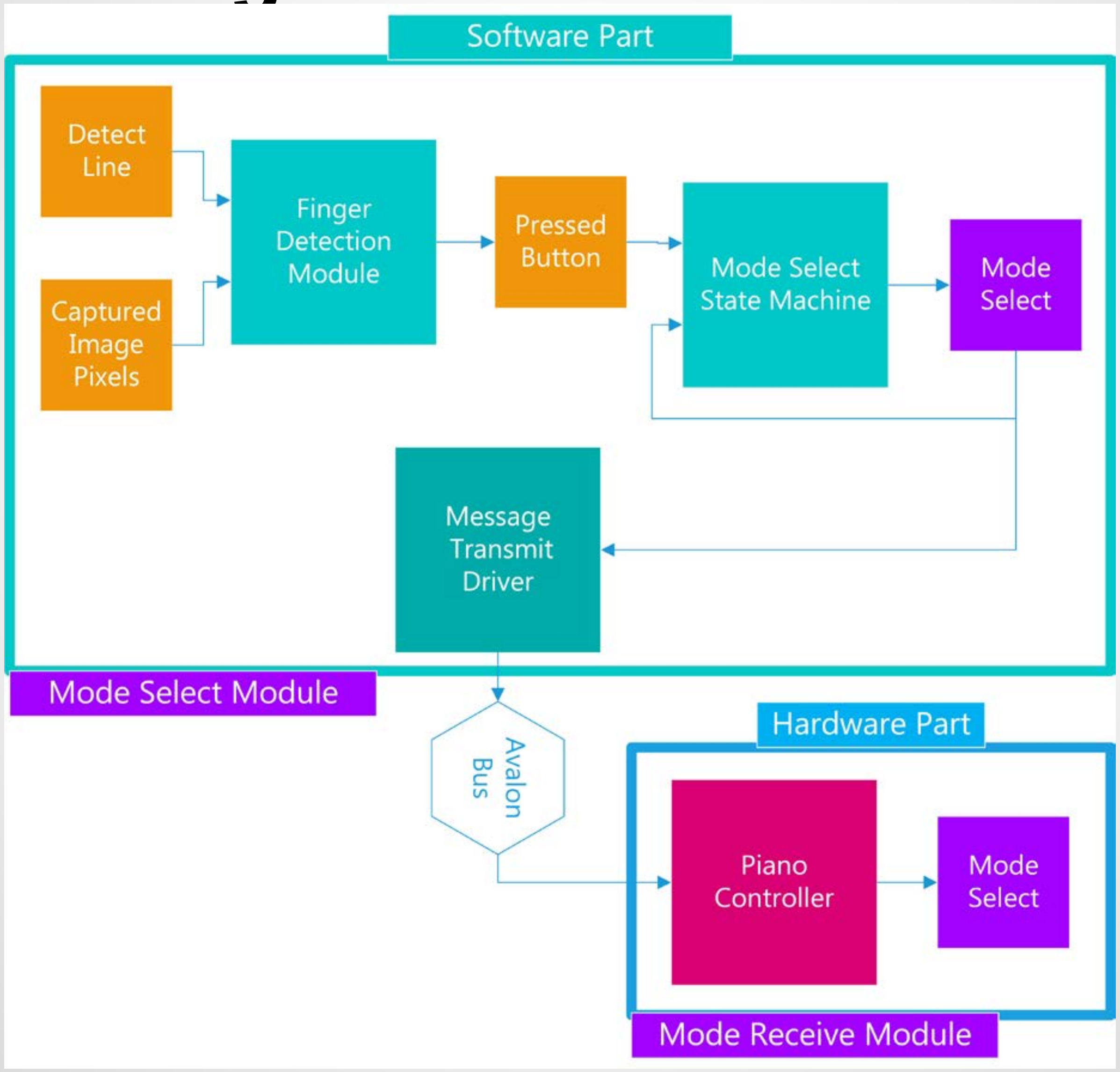
Finger detection



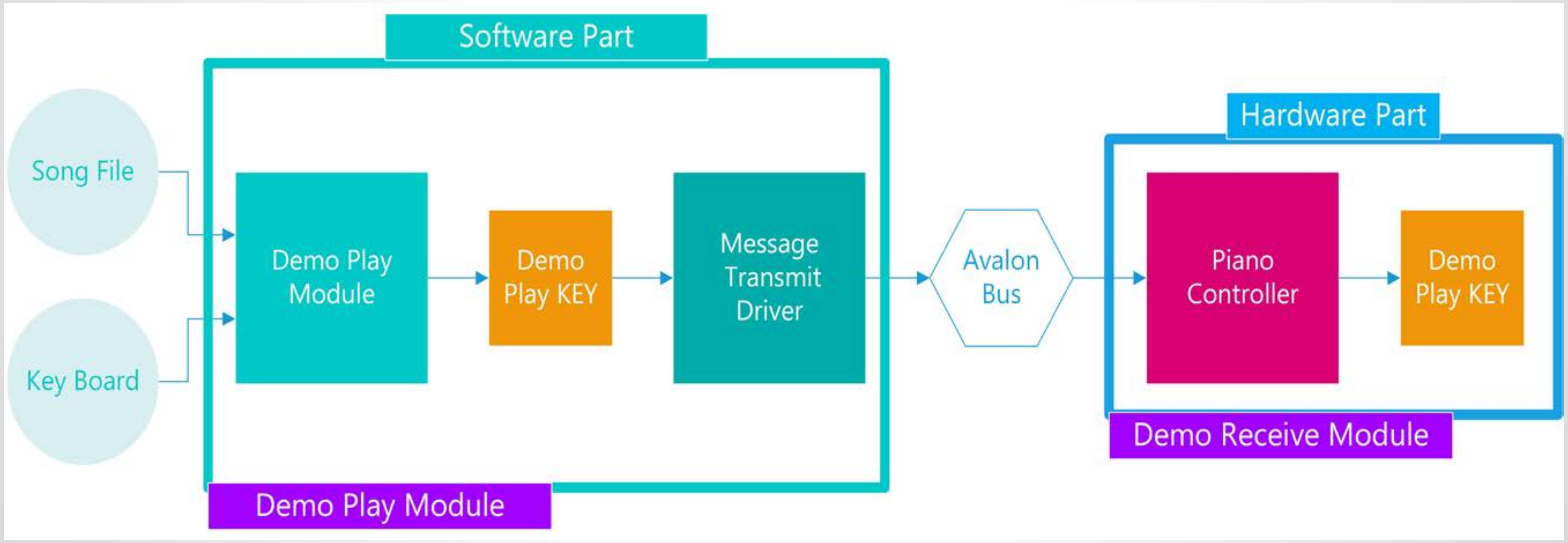
Key recognition



Control logic



Demo play


Software Part
Song File
Key Board
Demo Play Module
Demo Play KEY
Message Transmit Driver
Demo Play Module
Avalon Bus
Hardware Part
Piano Controller
Demo Play KEY
Demo Receive Module

VGA display

Our user interface module has several inputs, including original pixels, pixels RGB, fixed key position, three ROMs, mode select, and audio play note.

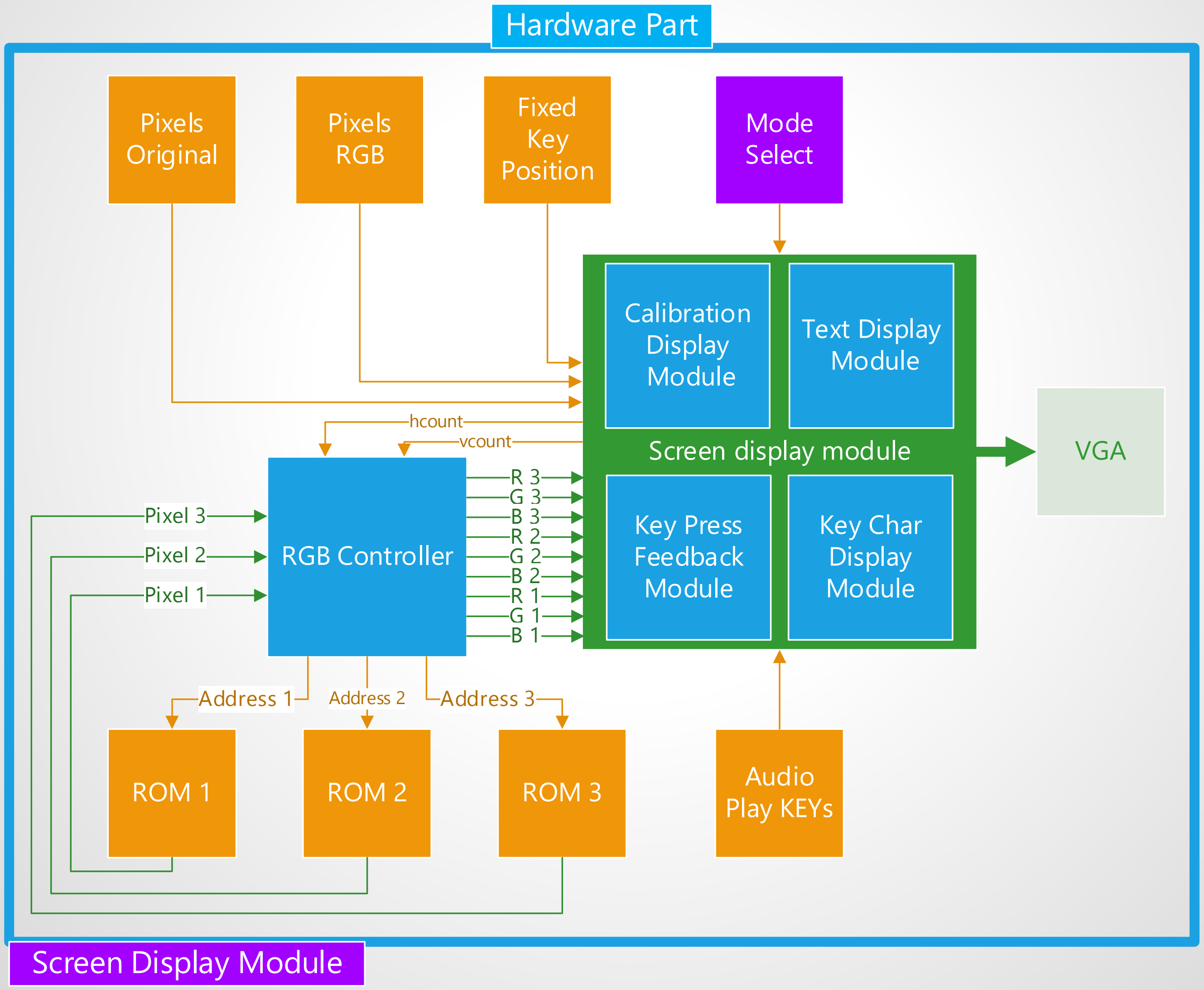
There are four parts in our screen display module:

- Calibration display module
- text display module
- key press feedback module
- key char display module

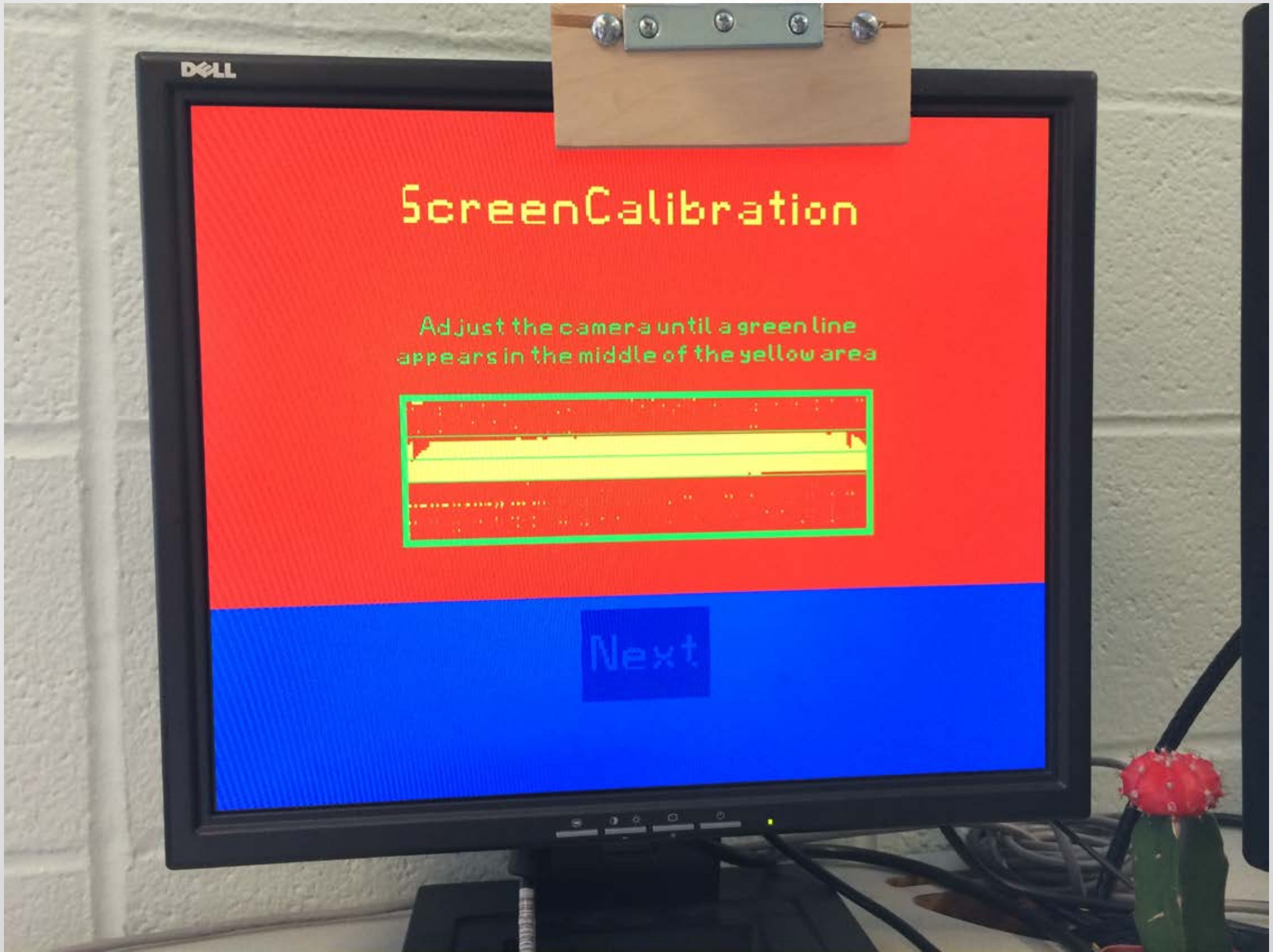
Our user interface have four pages.

- Screen calibration page
- Color calibration page
- Welcome page
- Real play page

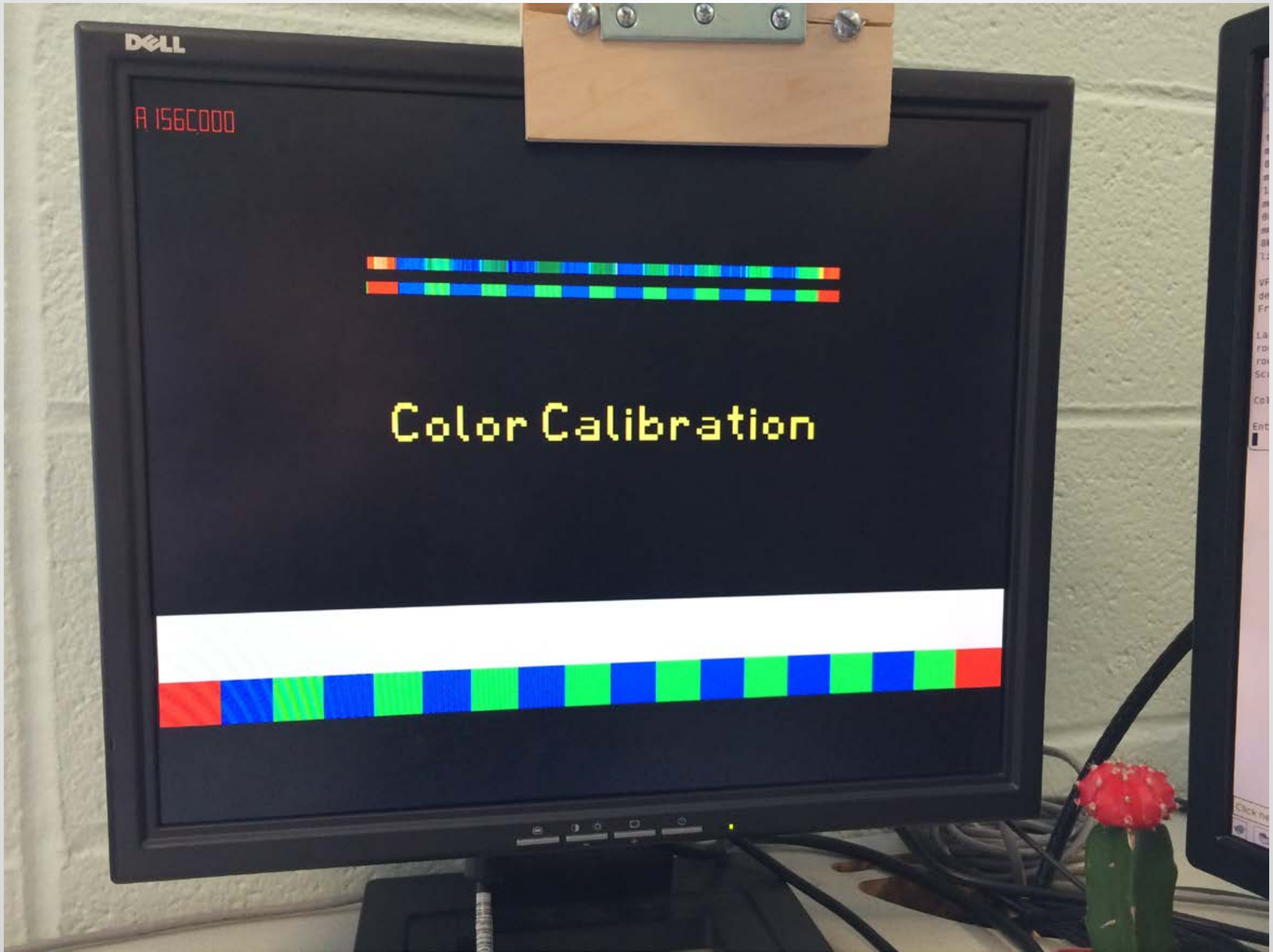
Hign level block diagram



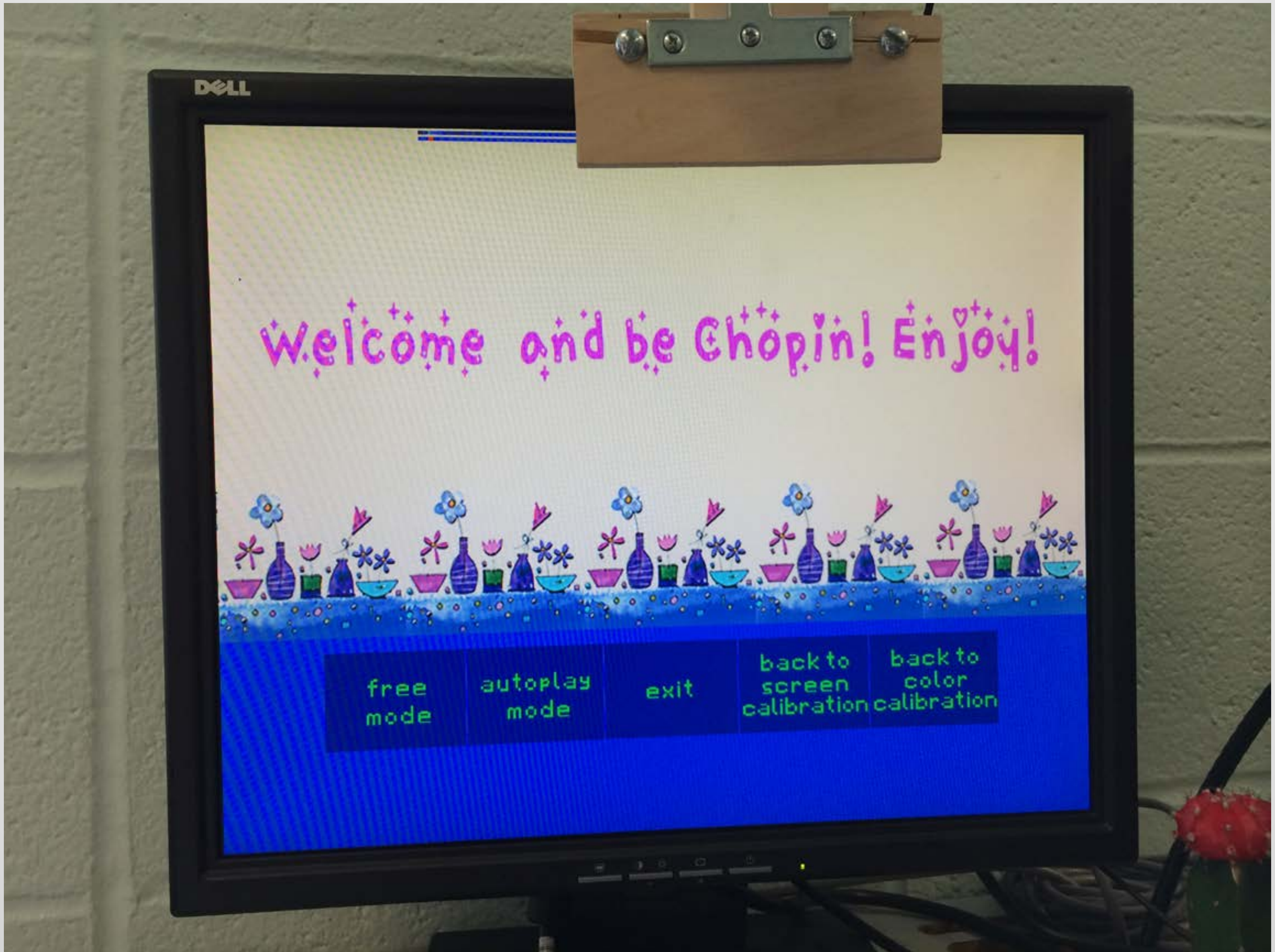
First page: screen calibration


DELL
ScreenCalibration
Adjust the camera until a green line appears in the middle of the yellow area
Next

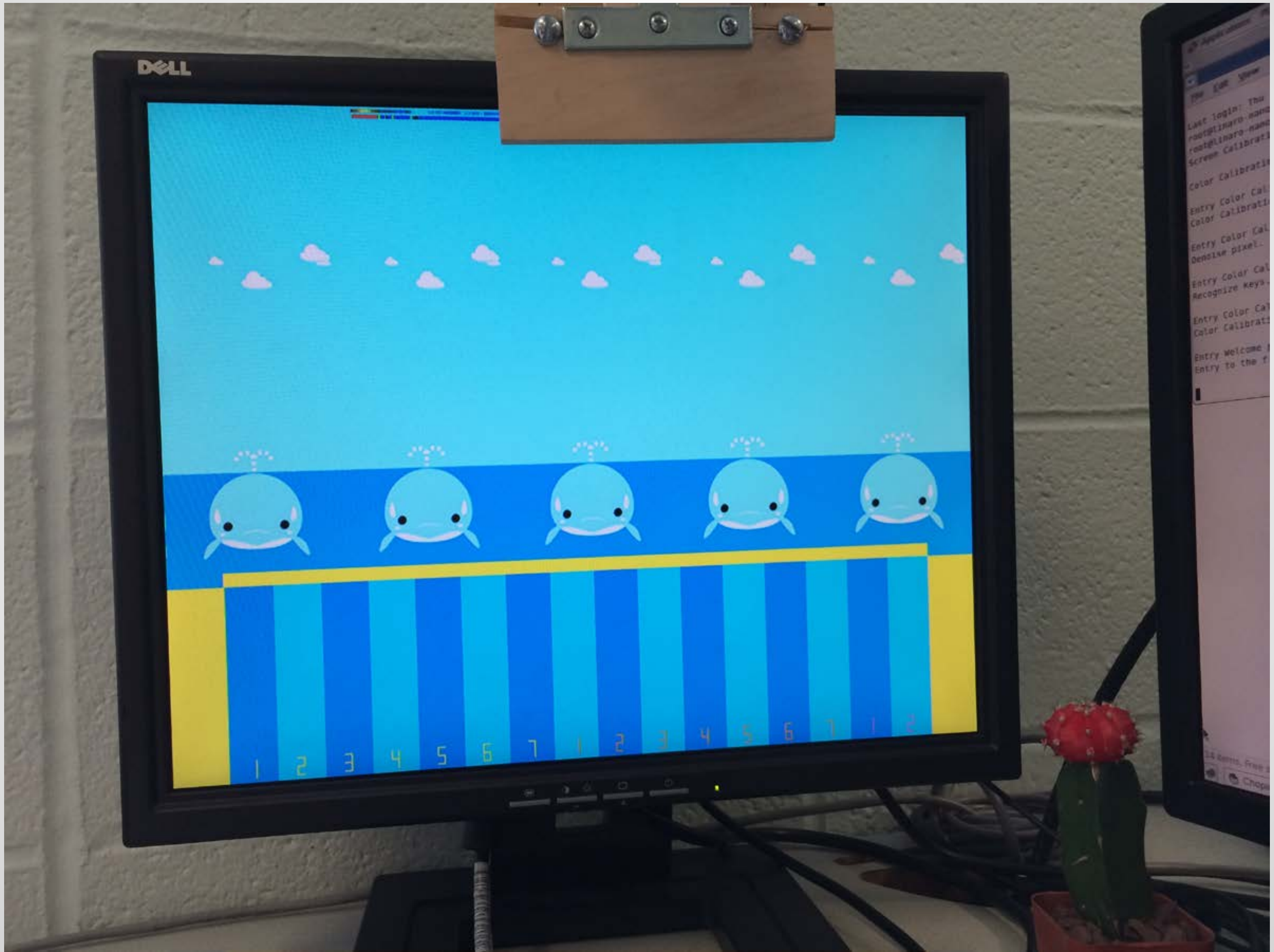
Second page: color calibration



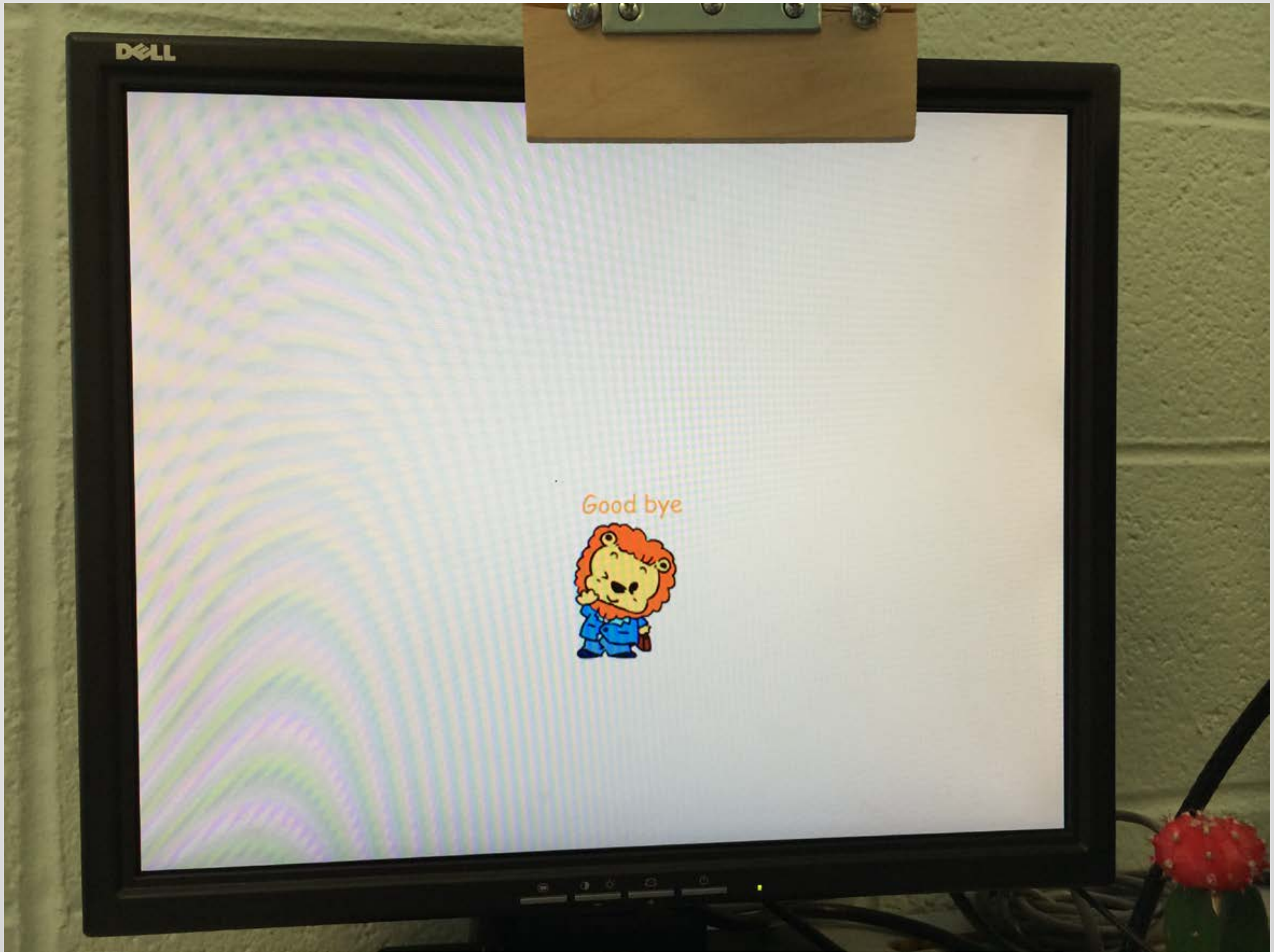
Third page: welcome page


DELL
Welcome and be Chopin! Enjoy!
free mode
autoplay mode
exit
back to screen calibration
back to color calibration

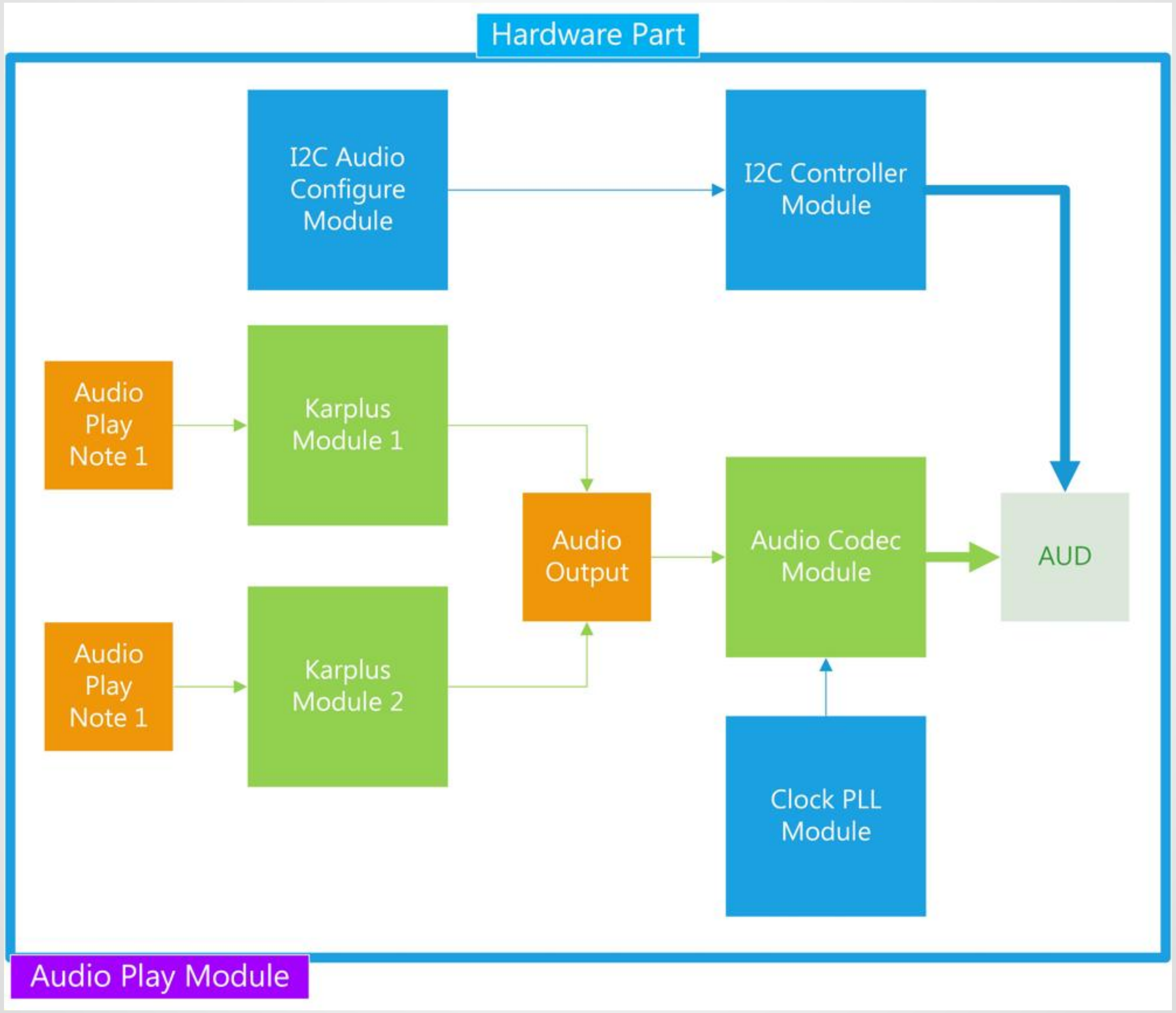
Fourth page: real play page



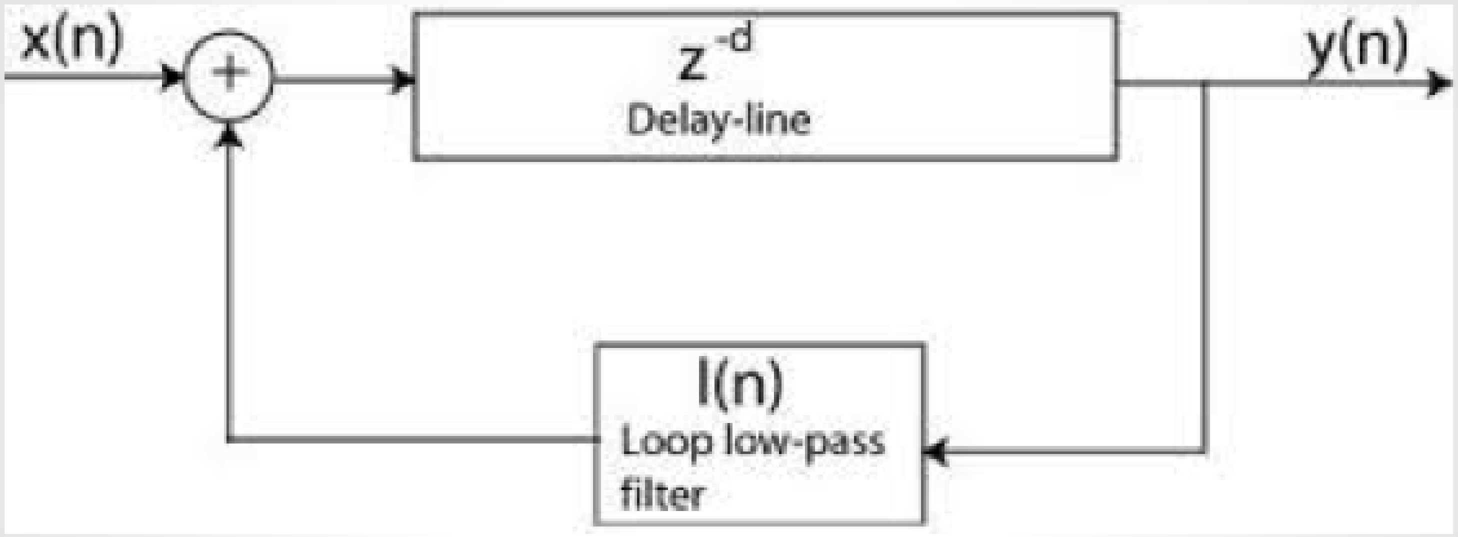
Goodbye page



Audio


Hardware Part
I2C Audio Configure Module
I2C Controller Module
Audio Play Note 1
Karplus Module 1
Audio Output
Audio Codec Module
AUD
Audio Play Note 1
Karplus Module 2
Clock PLL Module
Audio Play Module

Karplus-Strong Algorithm



- Three strings to emulate piano sound
- Sound mixing

$$freq = 44.1KHz(Sample\ Rate)/length$$

I2C Protocol & Codec

- Master-slave protocol
- 11.2896MHz working frequency
- 44.1kHz sampling rate
- 16-bit audio words

Work distribution

Daran Cai
Compile Linux kernel
Camera driver
Calibration (screen & color)
Finger detection and key recognition algorithm
Software control logic

Linjun Kuang & Wei Xia
VGA display for keyboard and background
RGB controller
Matlab code for converting picture to mif file
Drawing bit map and creating text using matlab
Calibration and text display

Wenyuan Zhao
Finger detection in hardware
Karplus-strong algorithm
Audio codec part and i2c configuration
Mixing two sounds to support press two keys

Challenges

- Compile Linux kernel to support camera
- Storage problem
- Robustness of algorithm
- Karplus-Strong algorithm

Lessons Learned

- Plan/think ahead
- Use tools like signaltap to do hardware debugging
- Think as a user
- Pay attention to the completeness of project
- Watch out for multiple levels of logic

Thank you

Simple Demo