ECE532

Final Project Group Report

Gerry (Chueh-An) Chen

994036805

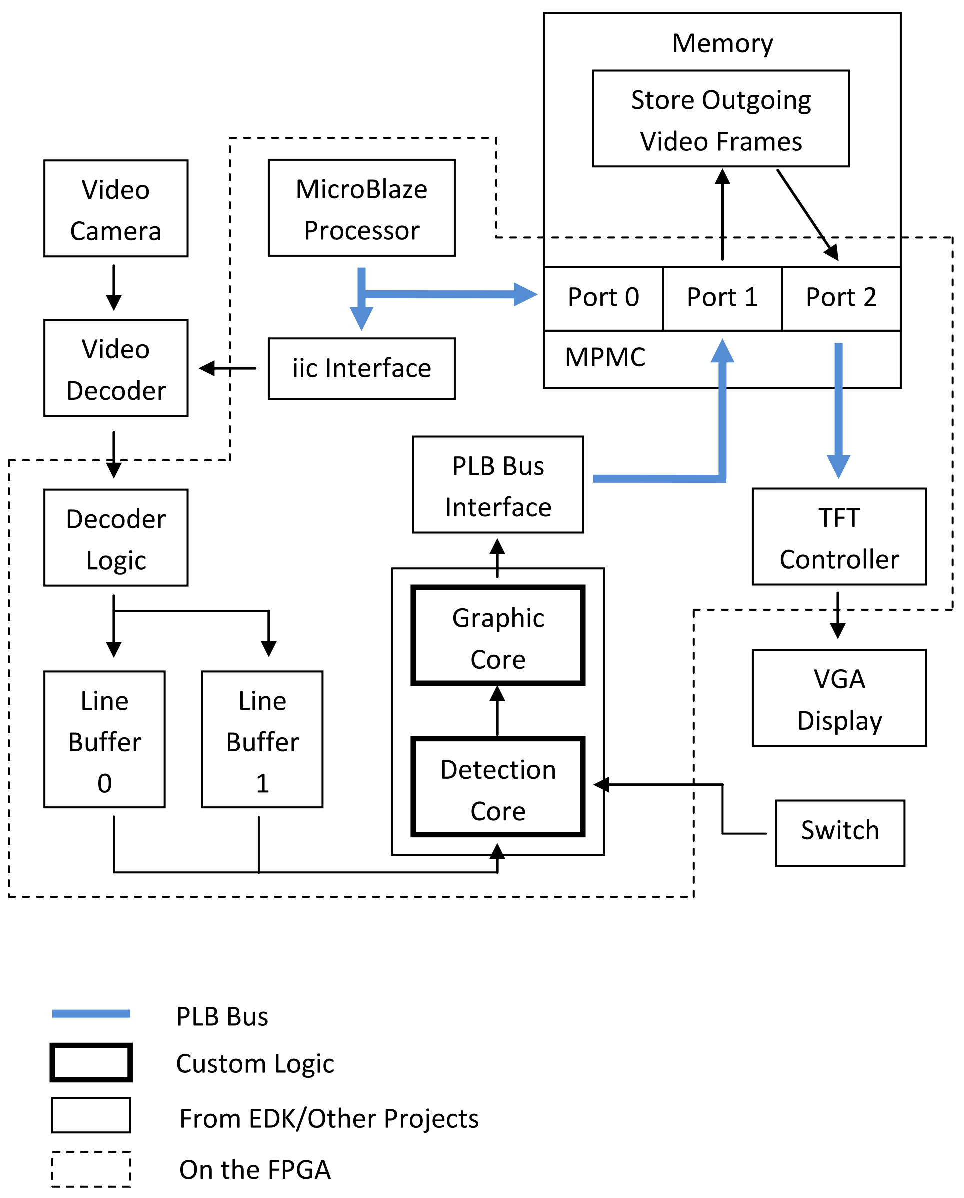
Jerry (Chi Kin) Chong

994653843

Project Overview

The goal of this project is to implement the detection algorithm and the graphic algorithm of the idioscope. This design features a multi-touch detection functionality and displays the key(s) being hit on the VGA monitor. The block diagram below illustrates the design architecture of the project.

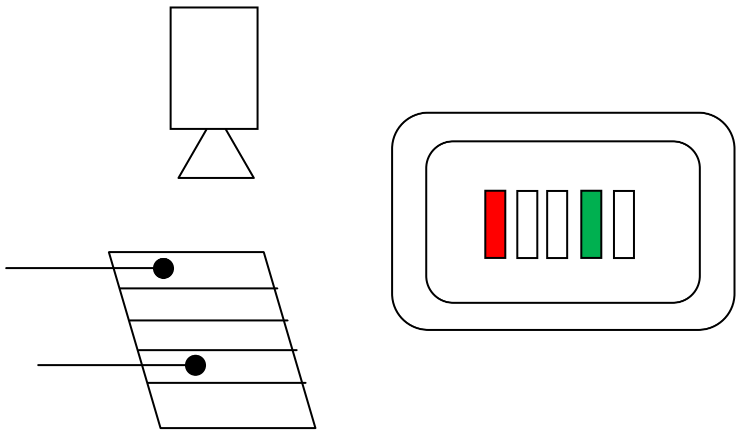
System Block Diagram



Brief Block Descriptions

Block	Functionality	Origin
Video Camera	Captures the video and stream it to the video decoder	Self supplied
Video Decoder	Process input video from the camera into frames and send to the decoder logic	Given
Decoder Logic	Converts pixel data format from YCrCb format to RGB format	Provided by Jeffery Goeders
Line Buffer	Buffers the incoming pixel data	Provided by Jeffery Goeders
Detection Core	Process the video frame and determine the key(s) that is/are being hit	Custom Logic
Graphic Core	Output the colour of each pixel of the output frame based on the output of the detection core	Custom Logic
PLB Bus Interface	Provides an interface to perform read/write access to the memory	Provided by Jeffery Goeders with minor modification
iic Interface	Sends instruction to the video decoder	Xilinx IP
MicroBlaze	Configures iic	Xilinx IP
MPMC	Provides multi port access to memory	Xilinx IP
Memory	Stores the outgoing video frames	Given
TFT Controller	Acquires frame data from memory and display the frame on VGA monitor	Xilinx IP
VGA Display	Displays output video frame	Given
Switch	Enables detection core	Given on the board

Project Outcome



The above diagram illustrates the basic principal on how our idioscope project work. When the user hit a key or multiple keys on the physical paper keyboard, the camera will capture the video and the video will be processed by the hardware FPGA. The hardware will then determine which key(s) is/are being hit and display the result on the VGA display. The key(s) that is/are being hit will be highlighted on the display in different colours.

Our idioscope project is a success. It is able to detect the green pointer under reasonable lighting. Also, after the hardware architecture was changed due to slow processing speed, the hardware algorithm is able to process the data quickly. The multi-touch algorithm was extremely successful which allows seven keys to be hit simultaneously.

Future Suggestions

In terms of hardware modification and improvement, it would be interesting to implement the audio block in the hardware. Having the audio block, we would be able to output various audio frequencies depending on the keys being hit. In addition, when multiple keys are being hit, we could mix the audio frequencies using FFT blocks to output various audio signals. This would add more completion to the idioscope project where one can actually play it as a simple instrument.

Details Block Descriptions

Video Camera

The video camera serves the purpose of capturing the video. The video camera was placed above the physical paper keyboard and it is connected to the video decoder through the composite cable.

Video Decoder

The video decoder is a hardware chip on the daughter card provided to us. The purpose is to process the incoming stream video and process it into pixel data.

Decoder Logic

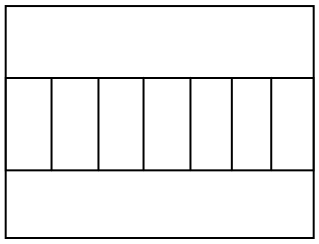
The decoder logic is provided by classmate Jeffery Goeders and it is also on the digilent website. The decoder logic converts the YCrCb pixel data into RGB data format which allows us to manipulate the pixel colour more easily

Line Buffer

The purpose of the line buffer is to temporarily store the incoming pixel data.

Detection Core

The detection core is a custom logic that is implemented by us. The core takes the incoming pixel data as input and searches for any green pixel in the frame. Once the green pixel is located, the algorithm will determine whether a key is being hit depending on the location of the pixel. The frame is partitioned as follows:



If a green pixel is present in any of the "box" in the middle region, then a key is hit.

To detect a green pixel, the following algorithm is as follows

If G > R + B, then we consider the pixel is green.

The Enable signal is connected to a push button on the board as an external port. The detection core is enabled by pressing down the push button.

To verify the functionality of this block, a verilog test bench module is implemented to generate the test vectors. Then it is verified through the waveform in ModelSim.

Graphic Core

The graphic core controls the colour and the placement of the pixel at the output display. Depending on the key that is being hit, the graphic core algorithm determines different colours for the pixels that are being written to the memory. When a key is not hit, the output pixel colour is white and if a key is hit, it is highlighted with a different colour.

For example, if Key 1 is hit, the pixel colour will be red, else it will be white.

To verify the functionality of this block, a verilog test bench module is implemented to generate the test vectors. Then it is verified through the waveform in ModelSim.

PLB Bus Interface

This block is provided by Jeffery Goeders with minor modification. The purpose of this block is to provide an interface to perform read/write access to the memory. It is important to have this interface to write the pixel data to the memory. In the project, it is currently doing 16 dword burst write to the memory location 0x40000000 constantly.

Iic Interface

The iic module is used to configure the video decoder. The Microblaze processor sends instructions to iic to start the video decoding process.

MicroBlaze Processor

The MicroBlaze processor is provided by Xilinx. It is a processor that can execute software instruction set. In this project, it is used to send commands to iic block to control the video decoder.

MPMC

MPMC is used to create multi-port access to the memory. In this project, we used MPMC to create three port to access the memory. One port for MicroBlaze processor, one for write access to memory, and one for TFT controller to read the video frame stored in the memory.

Memory

The memory used in this project is a 512MB DRAM. It is only used to store the outgoing video frame.

TFT Controller

The TFT controller block is an essential part of this project. Its purpose is to take the video frame and output to the VGA display. In addition, the TFT controller reads from memory location 0x40000000. The TFT controller refreshes the output video at the rate of 60Hz which is faster than our original goal which is 30Hz.

VGA Display

The VGA monitor displays the output result of the video frame.

Switch

The switch is a push button on the board. When this push button is pressed down, it enables the detection core to start the algorithm. Otherwise the hardware system does not do anything.

Description of Design Tree

Folder	Description
System	Constains the XPS project. Custom cores are unders the folder /pcore Key files include video_to_ram.v and core.v Detection_core.v and graphic_core.v are used in the old architecture; however, these two files have been integrated into core.v for the new system architecture.
Doc	Includes the final documentation of the project and the power point slide