ECE532 Group 2

FPGA Snapchat Filters

Kazi, Shariq, Shirley

Project Overview

Implement real time snapchat filters in an FPGA

Initial Goal

- Use real time image processing to detect face of a person
- Use pixel coordinates from detected face to draw an object over face
- Use bluetooth input to cycle between preset images

Problems

Image Detection Algorithms

- Originally Viola-Jones
- Ultimately abandoned due to complexity along with team not having any experience in field to offset time required to implement in software and eventually hardware
- Resources on FPGA in regards to memory was deemed not suitable for training algorithm with potentially thousands of positive and negative face images

AXI Stream Interface

- The Xilinx AXI Stream interface was somewhat difficult and tricky
- Time was required to fully understand and detect pixels properly

Availability of Video Board

One board shared across group lead to delays in testing individual blocks as testing video required the video board

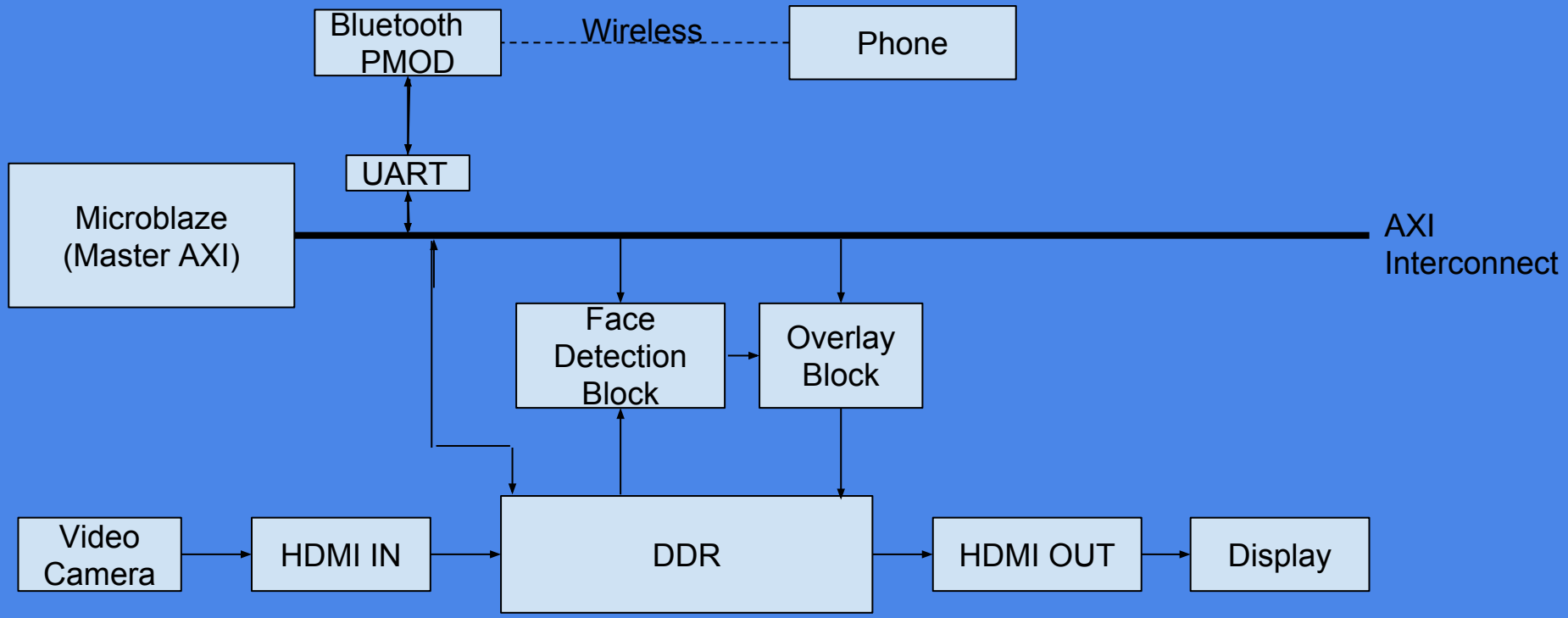
Poor Project Planning

Team spent time on efforts that were ultimately abandoned due to changes in project or due to poor planning.

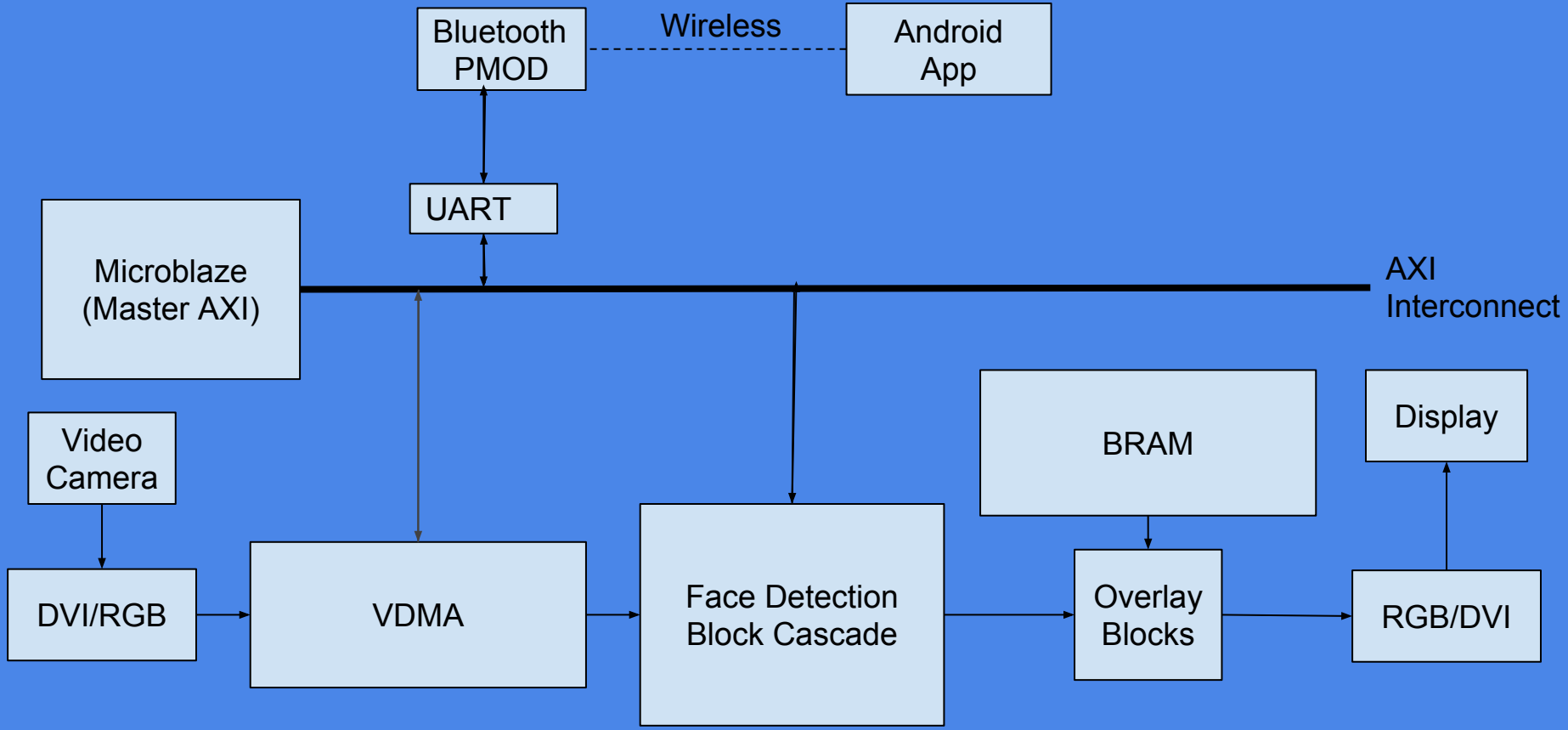
The team also aimed to have too many components working for this project (bluetooth, face detection, image overlay) and time management became an issue.

Block Diagram

Initial Block Diagram



Final Block Diagram



Project Components

System Components

- Bluetooth Stack/Android App
- HDMI IN & OUT
- Face Block Detection Cascade
- Overlay Block
- DDR3, VDMA, BRAM

Bluetooth

Bluetooth interface accomplished using UART IPs with Android App developed to supply input to change filters

HDMI IN & OUT

- Video in & out via HDMI ports
- Video in to AXI Stream & AXI Stream to video out (Xilinx IPs)
- DVI to RGB & RGB to DVI Conversion (Digilent IP)
- VDMA (Xilinx IP)
  - One VDMA to provide overlay image information
  - One VDMA to store framebuffer for display to screen

Block Detection (Custom IP)

- Inspect the video in stream input
- Output x & y boundaries to the overlay block
- Count x & y position of every pixel
- Detecting 4 blocks indicating boundaries of the filter (which determines boundary of face)
- FSM detecting max/min position of pixels with specific color on every line
-

Overlay (Custom IP)

- Change image pixels streaming through
- Take in input image from BRAM memory preloaded by COE file
- Output altered image to output VDMA
- Scale images to appropriate size
- Count x & y position for every pixel

Distribution of work

- Shirley was responsible for the overlay block
- Shariq worked on getting bluetooth input working along with the Android app
- Shariq and Kazi worked on the block detection algorithm
- Lots of overlap in helping each other debug as well as writing software to enable IPs to do what we wanted

How we achieved our goals

- Team made sure to create proper testbenches for all custom hardware blocks
- Team used ILAs to debug real time signals from HDMI input to verify video input was being processed properly
- Continually analyzed originally proposed tasks and adjusted expectations after doing further research and getting a better understanding of the project beyond that of the proposal

Lessons Learned

1. Designing with hardware synthesis tools requires good planning of time and design to minimize amount of time spent synthesizing solution by making sure design was working via testbenches and unit tests.
2. Time should be allotted to understanding protocols and interfaces of external IPs as not them completely can lead to debugging that can be avoided if these are understood beforehand
3. Project goals should be changed if feasibility due to time constraints is at stake as was the case here as the team early on realized that Viola-Jones was not viable given team’s skillset

Demo

Demo time