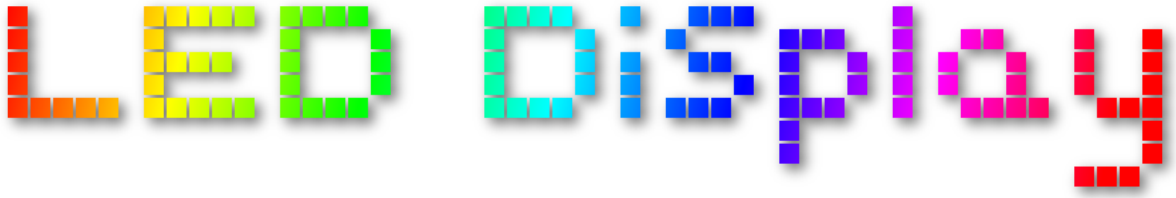
Design Report

For



Version 1.0

Prepared by Alexander Eklund, Colin Clifford, Peter Brown

University of Idaho Capstone Design

Last modified November 20th 2014

Table of Contents

Table of Contents **1**
1. Executive Summary **1**
2. Background **2**
3. Problem Definition **2**
3.1 Goals and Deliverables 2
3.2 Specifications & Constraints 3
3.3 Assumptions and Dependencies 3
4. Project Plan **3**
5. Concepts Considered **4**
5.1 Raspberry Pi 4
5.2 HDMI + FPGA 5
5.3 PCI Card 6
6. Concept Selection **6**
7. System Architecture **7**
7.1 Conceptual Design 7
7.2 Components 7
8. Future Work **8**

1. Executive Summary

The system being developed is an ***LED*** Matrix capable of displaying video. Potential applications of such a device are signs and large video displays used at concerts and sporting events. Our solution is a proof of concept for an open source design that business owners and hobbyists could use to build and maintain their own sign or video display. Current vendor solutions are very expensive to rent or buy and maintain, so such a design could be very helpful to entities that are interested in LED displays, but have a limited budget.

Our existing design from previous design projects is a step in the right direction, but its scalability and ease of use is limited. Our new design will send an HDMI video signal straight to an FPGA, which will more more-scalable and faster.

The LED Matrix can be assembled from one or more 32x32 LED panels. Upon completion of this project, functionality will be demonstrated with HDMI video played on these panels.

The LED matrix will be driven by an FPGA that will take an HDMI signal directly. HDMI enjoys widespread adoption, so is ideal for ease-of-use. And by driving an FPGA directly we can bypass speed issues inherent in the Raspberry Pi.

2. Background

The LED panels that our project uses are the building blocks for vibrant street signs and enormous LED walls and displays. LED signs are an effective way of advertising, and can be found standing next to streets, on buses, and buildings. LED displays are used extensively in the entertainment industry. They can be found hanging in the middle of most professional and high level college sporting venues, used to playback clips of recent plays in slow motion, and give the audience a live, close up view of the action. Where LED displays really stand out is at live music events and dance clubs. An LED wall towering behind a live band or DJ shines very bright in an evening setting, and can help create the right atmosphere and visual content to go along with music. Video clips, evolving geometric shapes, graphs of complex functions, and strobing effects can look impressive when edited and played in time to music on LEDs, and for many types of rock and dance music complement a live show very well.

Major sporting venues are generally going to have access to a budget that will allow them to meet their LED display needs through traditional vendors. However, small businesses and gigging musicians alike could benefit from an open source design that provides code and plans to create an LED display of a size to fit their needs.

To truly make the design layman friendly, providing some parts to make wiring, giving structure to, and storing the system a breeze would be necessary. This is where a business model could potentially fit into this “open source” design. Provide code and plans freely, and work as a vendor of the parts necessary to build the LED displays. Users would help each other work through issues on an internet forum, limiting the need for customer service.

3. Problem Definition

3.1 Goals and Deliverables

Our goal is to drive real-time full-motion color video to at least one, and ideally several, LED display panels with an FPGA.

The FPGA will accept HDMI input from any HDMI source and will identify as a monitor supporting the 640x480 resolution.

Our end of year deliverables will include full source code, as well as a working demonstration.

It’s not clear how fancy our end of semester deliverables will be. We may or may not have HDMI -> LED panels working, but we’ll be able to demonstrate that an HDMI signal is being received.

3.2 Specifications & Constraints

Our specs haven't changed too much since our preliminary specs. We basically want it to look good on as many panels as possible, not crash whatever it's plugged into, and overall be a reliable device that's as easy to use as a computer monitor.

Here's our most important targets:

Attribute	Marginal Value	Target Value
Color Depth	6	12
Tile-ability	Four 32x32 panels	Lots of panels
Picture Quality & Framerate	Jittery or Artifacts	Smooth
HDMI Input source resolution	640x480	640x480+
Reliable		Works without crashing whatever it's plugged into
Configurable	Not Configurable	Easy to change aspect-ratio or panel configurations.

3.3 Assumptions and Dependencies

Software Dependencies

- A recent version of Xilinx ISE and an OS to run it is required for any changes to the FPGA's programming. These are free.

Hardware Dependencies

- Adafruit 32x32 RGB LED matrix Panel, ID:607 http://www.adafruit.com/products/607
- Digilent Atlys FPGA http://www.digilentinc.com/Products/Detail.cfm?Prod=ATLYS
- Any HDMI Source that supports outputting in 640x480 resolution
- Type-A HDMI Cable

4. Project Plan

4.1.1 Team responsibilities

4.1.1.1 Alex

- Research Concepts
- Project pitch
- Research FPGAs
- HDMI research
- Communications with sponsor
- Snapshot day preparation
- Design review preparation
- Design report preparation
- Coding HDL for solution

4.1.1.2 Colin

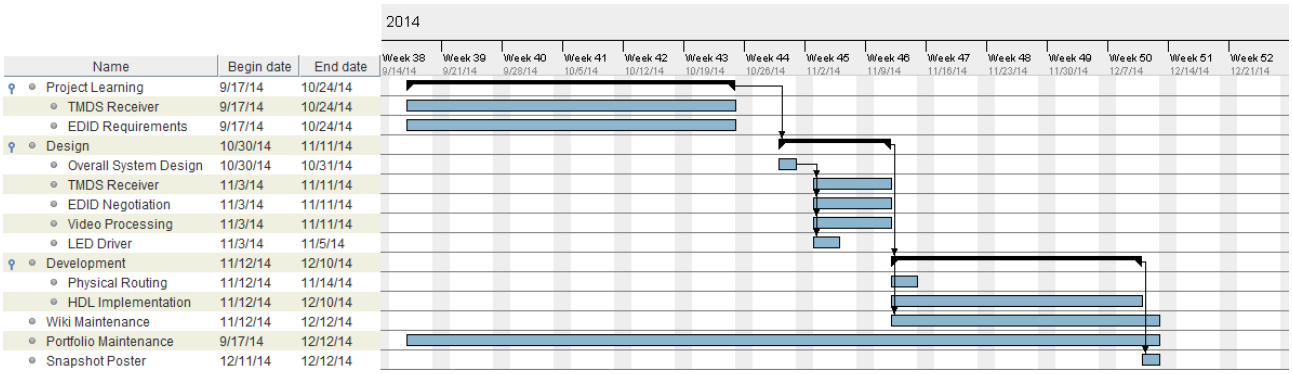
- Research FPGAs

- HDMI research
- Wiki page
- Snapshot day preparation
- Design review preparation
- Design review presentation
- Design report preparation
- Coding HDL for solution

4.1.1.3 Peter

- Research FPGAs
- HDMI research
- Portfolio maintenance
- Snapshot day preparation
- Design review preparation
- Design review presentation
- Design report preparation
- Coding HDL for solution

4.1.2 Gantt chart



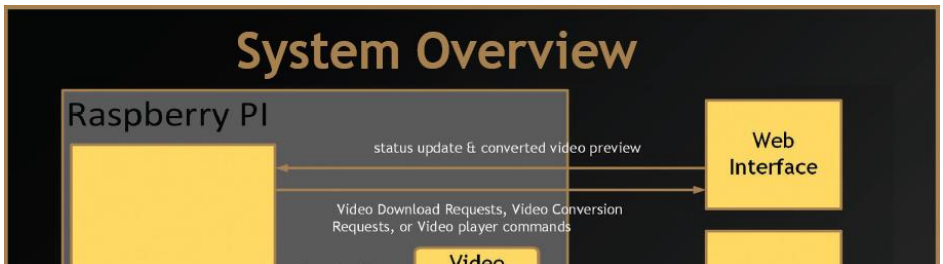
Name	Begin date	End date
Project Learning	9/17/14	10/24/14
TMDS Receiver	9/17/14	10/24/14
EDID Requirements	9/17/14	10/24/14
Design	10/30/14	11/11/14
Overall System Design	10/30/14	10/31/14
TMDS Receiver	11/3/14	11/11/14
EDID Negotiation	11/3/14	11/11/14
Video Processing	11/3/14	11/11/14
LED Driver	11/3/14	11/5/14
Development	11/12/14	12/10/14
Physical Routing	11/12/14	11/14/14
HDL Implementation	11/12/14	12/10/14
Wiki Maintenance	11/12/14	12/12/14
Portfolio Maintenance	9/17/14	12/12/14
Snapshot Poster	12/11/14	12/12/14

5. Concepts Considered

5.1 Raspberry Pi

One obvious direction the project could have taken was a direct continuation of the previous semester's work. The design from last semester had a number of issues and potential features to work on.

The design ran on a Raspberry Pi and a DE2 Nano FPGA. A web based interface was hosted by the Raspberry Pi. One could access the interface through a browser, and choose videos to play. Videos were downloaded to the Pi, converted to the necessary format, and sent to the DE2 which drove the LED panels based on the video data.



Possible improvements:

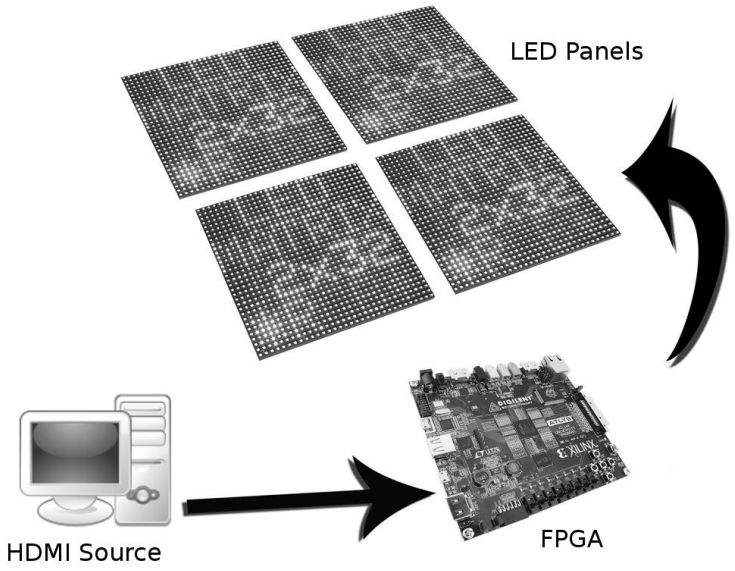
- Host the web interface on a fast machine that could have converted the videos to be played in seconds instead of a frame every few seconds. Thus videos could be played on demand without waiting many minutes for the conversion.
- Move the FPGA's frame buffer from inferred memory to SDRAM on the DE2. This would have saved FPGA resources for additional LED control modules.
- Get audio to play with the video. While not necessary for signs, many applications would require audio to play with videos.

There were major downsides to the previous semester's design that we wanted to free the LED panels from. The design required both the user and the Raspberry Pi to have a reliable internet connection. This proved to be a pain at the design expo last semester, when we could not demonstrate the web interface and control the video on the panels. Another issue was the bottleneck created by the Raspberry Pi's low basic I/O speed, which limited the amount of video data it could get out to the LED display. This was a major limiting factor to the possible size of the display. Ultimately continuing with the web based Raspberry Pi solution seemed like perfecting a dead end design.

5.2 HDMI + FPGA

The FPGA design involves driving an FPGA straight from an HDMI video source. This is the design we used, so there are details about it throughout this document.

Our vision is that a user could just plug an HDMI cable into their computer and immediately see some video output on the LED panels. This would be accomplished by plugging the HDMI cable directly into an FPGA which would directly drive one or more LED panels.



This will work with any device capable of HDMI output—as long as it supports a 640x480 output signal either through receiving EDID display capability data from the FPGA, or from direct configuration.

The FPGA then stores a video frame in an internal buffer, scale it down, and use it to drive the LED panels.

This design could be used with a computer for the HDMI source for sure, and possibly a video-game console or DVD player as well.

5.3 PCI Card

I love computers!

6. Concept Selection

- How did you arrive at your final selection?
- Include morphological charts, decision matrices

7. System Architecture

7.1 Conceptual Design

The idea behind our design is simple. We wanted to make playing anything you want on the LED panels as easy as using a normal computer monitor. The natural conclusion is to imitate the connectivity of everyday HDMI/DVI displays. We had already had success in driving LED panels from an FPGA, so if we could find an FPGA board capable of receiving HDMI or DVI, we would be golden. We found the Atlys, a board with all the HDMI ins and outs one could need, and a beefy FPGA to boot. So the plan is simple. The Atlys receives HDMI from whatever source, decodes the signal and puts the data in memory. Each video frame is converted to the size required by the LED panel setup, and the panels are then driven from the Atlys over basic I/O.

7.2 Components

1. **HDMI**
   HDMI satisfies the plug-n-play requirement as it is a video signal standard that enjoys widespread adoption. HDMI cables can be purchased for cheap online, and most modern computers or entertainment systems are capable of producing HDMI output.

   Additionally HDMI is fast enough to more than meet all our timing needs.

2. **Atlys FPGA**
   FPGAs in general are fast and parallelizable. The Atlys more than meets our speed needs, with plenty of FPGA fabric and a high clock-rate. Particularly it's capable of reaching the 200 MHz pixel clock that is needed for receiving 640x480 HDMI video.

   Additionally the Atlys comes with both an HDMI input and an HDMI output port built in. HDMI ports are important for getting HDMI to work.

   Finally the Atlys comes with a high-speed VHDC connector for general purpose output. We need as many output ports as we can get to drive as many LED panels as possible. Even with the VHDC port it's unclear how many panels we'll be able to hook up.

3. **LED Panels**
   Since the entire point of our project is to drive LED panels, LED panels are really important.

   We're going to be using 32x32 Adafruit LED panels, just because that's what we already have and what the previous design projects were working with.

8. Future Work

Recommendations for continued work

- Features that didn't find their way into the current design
- Estimate size and duration of the required effort