May 2014

Intelligent LED Display

Alexander Joseph Ryan
Worcester Polytechnic Institute

Harrison Chabot Williams
Worcester Polytechnic Institute

Rayce Colemen Stipanovich
Worcester Polytechnic Institute

Intelligent LED Display

A Major Qualifying Project

Submitted to the Faculty of Worcester Polytechnic Institute

In partial fulfillment of the requirements for the

Degree in Bachelor of Science

In

Electrical and Computer Engineering

By

Harrison Williams

Rayce Stipanovich

Alex Ryan

Date 04/12/2012:

Sponsoring Organization:

Worcester Polytechnic Institute:

Project Advisors:

Professor William R. Michalson, Advisor

Abstract

The goal of this project is to increase the overall redundancy, and ease-of-use during installation and operation, of large-format LED video displays for the professional touring and outdoor display industry. Using design concepts found in large-scale redundant networks, the system dynamically scales video output to the LED display and provides adaptive real-time fault detection and failover behaviors to ensure reliability in rigorous outdoor environments. This ultimately simplifies installation of a system, eliminating the need for the individual addressing of panels and alignment of video content. The designed system is inherently redundant and the ability to sustain failure of its components increases with the size of the display making it ideal for live applications. The developed display also possesses a dynamic run-time scaling ability of the video output, removing any need for image alignment and manual configuration.

Acknowledgements

The team would like to acknowledge and express our gratitude towards several individuals for their contribution to the project – it is likely we would not have succeeded as much as we did without their help.

We would like to thank our advisor, Professor William Michalson, for his keen engineering insight that helped us avoid pitfalls.

We would also like to thank Paulo Carvalho (BME/RBE ’15) for his assistance with the project as an extra member – his manufacturing, hardware design and documentation skills were extremely helpful.

We would lastly like to thank Joe Germain for his assistance in creating the LED panel cases with his 3D printer.

Executive Summary

With the arrival of multicolored Light Emitting Diodes (LEDs), large format displays used for both informational and entertainment purposes have become a reality. Arrayed LED matrix panels exist in many shapes and sizes. Often times installed in static locations, these large format displays that consist of several smaller panels (32x32 or 40x40 LEDs) are commonly used as dynamic billboards, and can be found in sporting venues to show the score and live video of the game as well as in the entertainment industry for backlighting of the stage or to create dynamic performance elements on stage. Typically these displays are bulky and need to be installed in a certain order for them to work and interface with the software that processes the video that will be displayed on the screen. These drawbacks leads to a time consuming and costly installation executed by specially trained personnel.

The goal was to develop a unit that could be easily installed without advanced training through the use of self-addressing technology, be robust by use of fault detection and redundant data paths whilst still minimize the cost per panel. Note that common manufacturers' units run in the price range of $1,700 to $5,200 depending on the distance between LEDs as well as the number and quality of the colors that can be reproduced. The proposed system would addresses many of the shortcomings of today's mainstream panels that still suffer from lacks in redundancy, reliable life tracking, and the restrictions of addressing.

The large scale of the project required it to be divided into four main sections: The panel chassis and rigging hardware; the architecture type which determines the powering scheme of the LEDs; the firmware of the individual panels; and, the main video processing software. The independent nature of the tasks would allow for work to proceed in all simultaneously. A networking protocol both for communication between panels and to the controller was decided early in the developed and would allow for the different sections of the project to work together once completed.

Two unique sets of hardware were developed and differed in the LED powering scheme. The first used multiplexing of LED drivers to illuminate each row briefly before scanning on to the next one whilst the second used a direct driven method, where each of the LEDs in the matrix were given their own driver. Even though the later allowed for

an increase in the panel lumen output, provided better color fidelity and eliminated issues that would be traditionally encountered when trying to film the panel with a camera that has a rolling shutter function the direct drive has low power efficiency and requires a larger physical space to achieve these benefits. The direct-drive powering topology can be observed in figure 0.1.

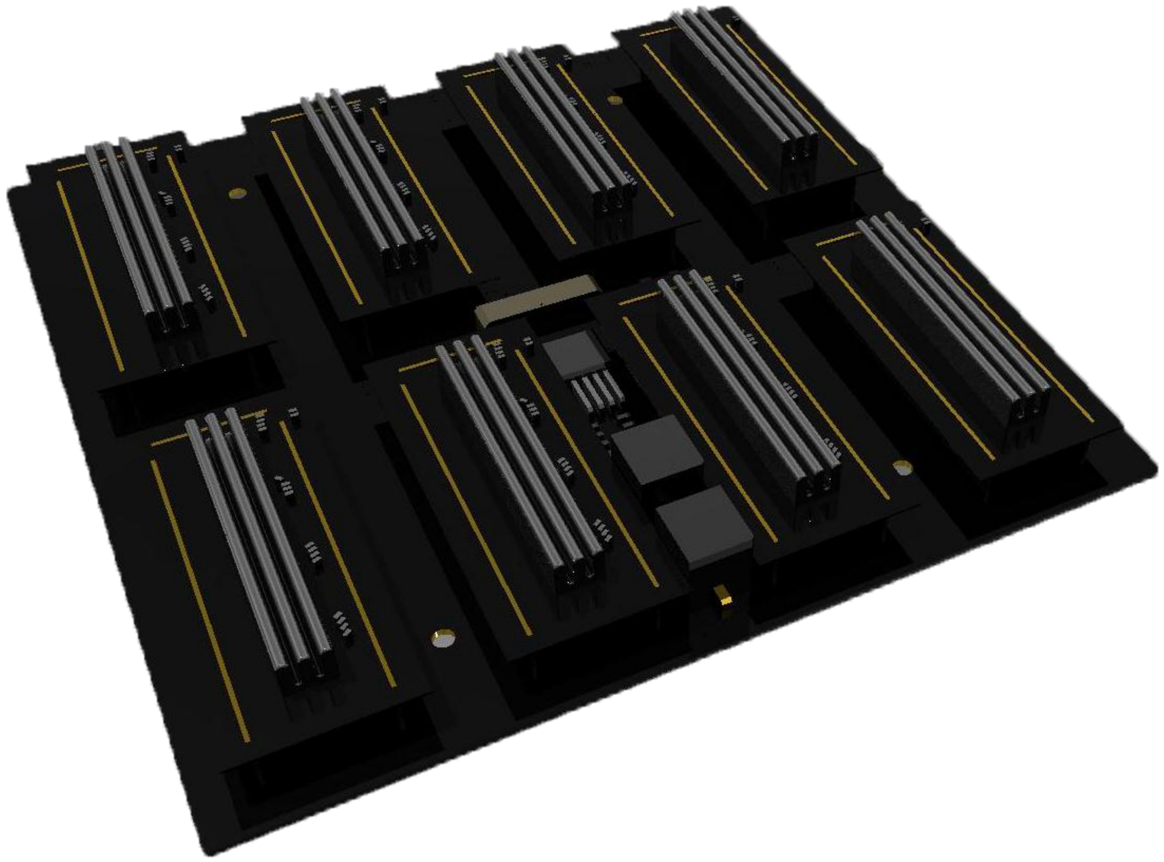


Figure 0 - 1: A 3D rendering of a panel design, showing eight LED driver boards connected to the LED matrix PCB below.

Regardless of the LED driving scheme a firmware capable of managing both the communication between the panels and the controller was required. The panels needed to not only receive and forward video at a high speed but also to report back fault conditions to the host. To implement this, the traditional IEEE Ethernet protocol was employed with a few layers removed in order to decrease overhead. Ethernet allowed for the routing of video packets in conditions where there are panel faults in the network and onto the proper destination. The IEEE Ethernet stack would provide the project with a well-documented and widely used standard as a foundation for further development.

The fourth constituent of the project was the video host and management server. This machine centralizes the control and management of all panels to a single easily operated interface. It constantly send data to all operational panels while monitoring their status. In case a failure is detected the software takes care of performing the necessary changes in the video output in order to maintain the system running.

Aligned with the goal of creating an easy to use and robust system through the use of self-addressing and fault detection techniques, the team divided the project into four main areas. Each of the project areas, which included panel configuration, architecture, firmware and higher level code, were tackled simultaneously to ensure completion of the project within the stipulated deadline. The final result was a product that provides innovative solutions to common industry problems at a low cost and that is under way to become market ready through more extensive testing and hardware iterations.

Table of Contents

List of Figures

List of Tables

List of Equations

Chapter 1: Introduction

1.1 Motivation

Large format video displays are used in a variety of different applications. With the advent of power-efficient Light Emitting Diode (LED) technologies, displays that are normally comprised of projection-based video systems are often replaced by lower-maintenance, more efficient LED display systems. Large format LED video displays commonly referred to as “video walls” [1] are now considered to be a staple in the professional touring music industry as well as in the advertising industry. These video walls are comprised of individual panels that are electrically and mechanically connected to form a larger display. As video resolutions increase to those that are above Full-HD 1080p video, the size and complexity of video walls increases, often adding multiple points of failure to a system and increasing setup time. Larger video resolutions generally result in a requirement to add more panels to an array, as increasing the pixel density of an array is often too expensive to consider as a viable option. In addition to the inherent complexity of large systems, in many situations within the professional touring music industry, a setup crew may only have a few hours to unpack, assemble, hang, and test a video wall.

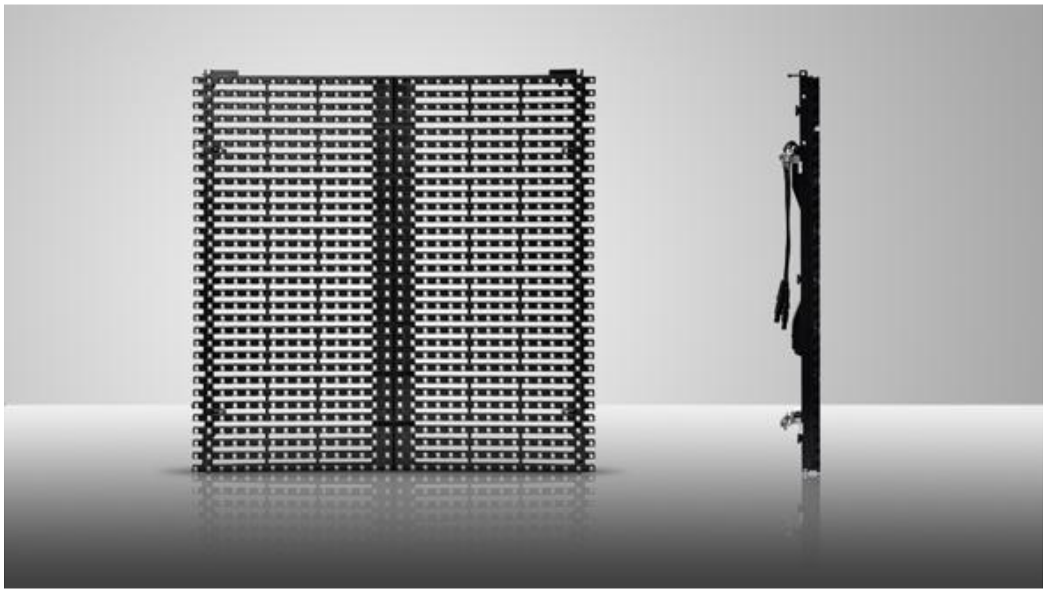


Figure 1 - 1: PixLED Linx-18F Outdoor Panel From [2]

LED video displays have also penetrated the advertising markets and are seen on the sides of buildings, billboards, or other installations. These offer great flexibility to advertisers, but often incur substantial maintenance costs. These arrays are often found outside and are exposed to harsh weather conditions [2]. Because of this, these systems require routine maintenance and the occasional repair. This can be difficult or dangerous depending on the location and installation type. The industry has pushing to develop robust panels as to decrease the rate of failure, but there are currently no solutions on the market to offer a significant amount of failure redundancy. Many systems only offer one-way communication from the controller handling the video processing and the individual panels. If there is a fault in the system, it usually be discovered by an individual observing the array and noticing a flaw in the video output. This likely has many desirability's for advertisers using video walls in remote locations, as faulty displays could potentially go unrepaired for significant amounts of time. As some systems rely on standard communications protocols such as TCP/IP to operate [3], two-way communication can be implemented to provide various forms of fault detection.

In the context of a live musical performance, if a system fails, it will be noticed by thousands of people. Discretely making repairs on such systems becomes very difficult, as the work environments are often poorly-lit, confined spaces. The ability to quickly diagnose, repair, and replace panels in an array becomes critical, both from the perspective of visual aesthetics as well as safety. The industry has pushed to develop weatherproof solutions for the LED output panels themselves, but the control interfaces and network infrastructure are often less fault-tolerant and are more-likely to fail. In many systems, a loss in a single connection can affect a large portion of an array or even the entire array itself. In systems with large numbers of connections, repairing and maintaining such a system requires elaborate methods of personal transport to replace broken panels. The ideal solution is to prevent large-scale faults that require such repairs in the first place.

The goal of this project is to build upon common technologies used in current video display systems by implementing inter-panel and panel-to-host communication, allowing for fault detection and other reliability improvements. These technologies will also be used to improve setup times by eliminating the need to address individual panels and align video content. This report will examine the design approach and methods used to achieve a redundant and "intelligent" large format video display array. Notable applications for LED panels include large format outdoor displays for touring video production, roadside advertisements, architectural displays, and large format reference displays for cinematography. In most cases, the panels consist of ruggedized and sealed PCB's and enclosures that are designed to interlock in such a way as to allow for tessellation. In some cases, panels such as the DigiLED FLEX dFX10, shown in figure 1.2, are designed to be flexible, enabling construction of curved surfaces.

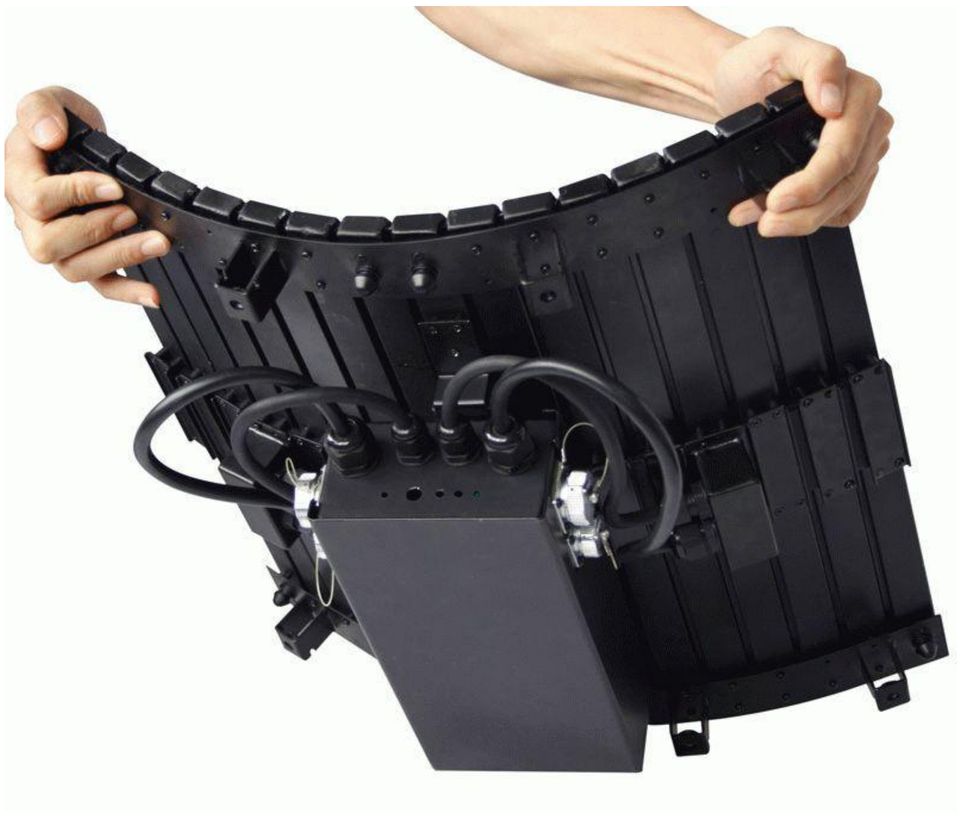


Figure 1 - 2: An example of arrayed LED matrix panels, which exist in many forms and cater to many markets. From [4]

1.2 Current State of the Art

There are several main focuses of research within the current industry. These are primarily power efficiency and video output quality. Many panel manufactures pursue methods to remain power efficient, ultimately enabling larger displays to be implemented for the same power budget. This has some inherent drawbacks. Having more panels within an array requires many more electrical connections. This, in itself, introduces multiple points of failure. Most panel manufacturers are currently exploring faster methods of data transfer [3] such as using the physical layer of the TCP stack, but few panels use the full IEEE Internet Protocol for communication.

Currently, panel manufacturers such as PixLED and DigiLED are pursuing methods of reducing the bandwidth required to send video content over Cat5 cables to a video panel. This allows for faster data transfer, but also introduces an issue of more points of failure. Panels such as the DigiLED MCK series use differential signaling to transmit data, but do not rely on the IP stack to transmit full-duplex data. These panels only offer methods of daisy-chaining panels in a row. While this ultimately simplifies the complexity and reduces the amount of cables required to transmit data, it introduces multiple points of failure. Often times, a failure in a panel early in the signal chain results in a cascading failure to a large section of the array. This is undesirable, as not only is the fault not localized, but it affects neighboring panels as well, causing large visible faults to the viewing audience.

Figure 1 - 3: DigiLED Navigator Panel Host Interface [5]

When combined with DigiLED panels, the DigiLED Navigator [5], shown in Figure 1.3, allows for real-time scaling and switching of video content. This is then sent to the array through Cat5e cables through a broadcast network. Each panel then reads only its part of the entire array's video frame based on its pre-determined address. In addition to this, the Navigator has the ability to send control signals to the panels to perform actions such as running diagnostics or the ability to remotely power-down the panels. These features add a significant amount of convenience to the product, but as these communications are only one-way, information cannot be received from the panels. Because of this, the typical solution for detecting faults is to have a physical observer or camera monitoring the panels form the audience's perspective. This ultimately is inefficient, as it is still prone to human error and requires an individual to detect an error or fault.

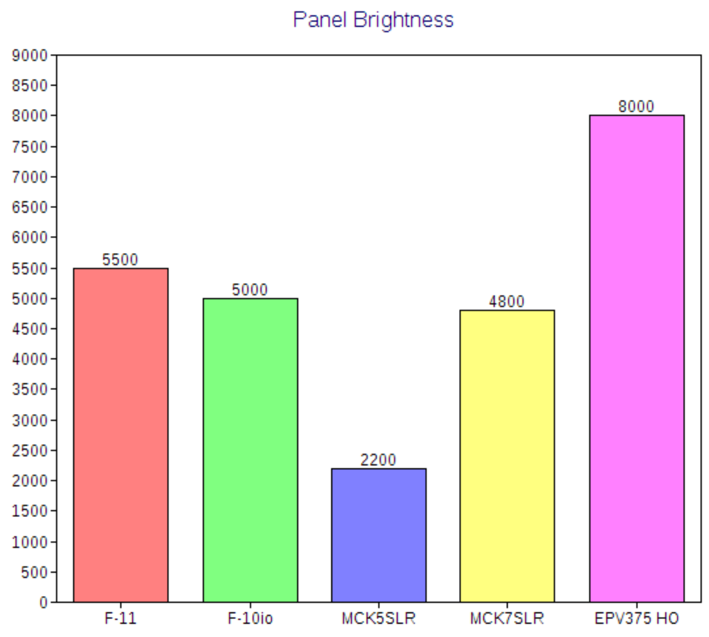


Figure 1 - 4: Current market panel brightness's. From [6]

The high-end panels currently on the market generally have a brightness output of around 5,000 lumens. This is a suitable brightness for most outdoor applications, and is even viewable in direct sunlight. Panel resolution and brightness are the two largest areas researched by panel manufacturers [6], and as shown by the Figure [x]. Because of this, little research has been conducted into redundancy, which is where this projects aims to focus on.

1.3 Proposed Contributions

This project aims to create an array of LED panels that are capable of displaying real-time video content. These panels will possess the ability to self-address in order to dynamically map their output to video content. This will be achieved by developing an algorithm that uses both the IEEE Ethernet stack and a proprietary inter-panel communication protocol. The panels will also feature redundant mesh networking through the implementation of IEEE 802.1w to prevent failures due to damaged cables or other interruptions to the signal path from the video server to the panels. The panels will use these systems to communicate with a central video processing host and will possess the ability to notify an operator of the existence of and location of faults in a panel array.

This project overcomes the current shortcomings in the current state-of-the-art by the following approaches:

1. Remove the need for the individual addressing of panels in an array, allowing the array to self-address upon startup.
2. Automatically and dynamically scale video content to match the dimensions of the video wall.
3. Alert maintenance staff of a failure and provide the location of the fault.
4. Offer inherent network redundancy through multi-point RSTP-supported mesh networks.
5. Offer higher-quality video output for rolling-shutter cameras through directly driving each LED individually as opposed to cycling through rows.
6. Offer an intelligible user interface that is straightforward.
7. Perform video processing on consumer-grade computers instead of expensive proprietary system processors.

This project will prove the benefits and viability of implementing redundant systems to improve the quality of large format video displays over those currently on the market. This design will be realized through a combination of common development boards, peripheral resources, and custom designed electrical and mechanical hardware. In addition to this, custom video processing and array control software will be written to handle the majority of the self-addressing and failover behavior logic.

1.4 Report Organization

This report is comprised of 6 chapters. They are: Chapter 1: Introduction, this chapter introduces the project, its design goals and methods, and explores our motivation to work on this project. This chapter also takes a look into the preexisting products and other implementations in the market today. This leads into Chapter 2: Network-Distributed Video Systems where it discusses how video can be transported across networks and how the data interacts with the network topology as a whole. It also looks into how to design a network specific to this project and how to select a video encoding method that is optimal for the given network configuration. In order to get video from the main server, or the host, a network had to be designed such that the video data could be transported quickly, without loss and with fault redundancy. The first step is to analyze different methods of video encoding and determine which would be most common and easy to use given that it would be implemented on a network of our choosing. Once the video encoding format had been chosen, a network format needed to be selected so as to optimize the transport of video, and be able to address the concerns stated previously. Use of the IEEE network protocol 802.1w was selected due to its ability to provide a decent solution to a majority of these issues. This lays the foundation that the other portions of the project could be designed to, and gives hard targets for code and hardware to perform at.

After that was done, Chapter 3: Proposed Design and Project Logistics takes a look in detail at the project goals and objectives with an emphasis on design criteria and decisions made while the project progressed in order to meet the standards. It also looks at which decisions had to be made so that the rest of the project could be designed. It will also explore the project management and tasks necessary for the project to be a success. Also this chapter will provide a design summary for the project as a whole.

As the report moves into Chapter 4: Implementation, it discusses in detail how each portion of the project was created and made to work with one another. Both the multiplexing and direct drive architectures are described along with the details necessary for how both styles interface with the panel embedded operating system. Details as to the software running on all of the panels is discussed and reviewed. The host controller software is also looked at in detail. Moving on to Chapter 5: Results sees the report addressing the project as it stands today and the status of completion in the various target areas, and any improvements that had to be made to the original designs in order to realize completion. This chapter looks at panel calibration with regards to still images, and the final frame rate of the panel. Also discussed is the panel interaction with rolling shutter cameras and the network utilization and stability. Finally, Chapter 6: Conclusions and Future Work looks into suggestions and techniques that could be utilized for continued work on the subject, and where improvements can be made should work be continued at a subsequent time.

Chapter 2: Network-Distributed Video Systems

This section provides a general background of the concepts used in the final design and describes the current technologies implemented in the project. It discusses some of the advantages and disadvantages of such that led to the overall design decisions chosen further discussed in Chapter 3.

2.1 Professional Video Standards and Protocol Selection

Most video walls in existence today use only a few video input standards. HDMI HD-SDI, and DVI are three of the most common protocols in use in the industry today. HD-SDI or High Definition Serial Digital Interface is used as a digital video link that was first designed in 1998 by the Society of Motion Picture and Television Engineers for use in high-definition broadcast grade video. It offers a nominal data rate of about 1.5-Gbps (1080i) over a distance of about 300 meters, far longer than traditional analog video standards can possibly provide. For example, VGA is able to produce a maximum resolution of just 640x480 pixels. SDI and its variants provide uncompressed unencrypted video data via a serial byte stream with intermittent timing bits.

Figure 2 - 1: HD-SDI and SDI protocols utilize BNC connectors. From [7]

Also in use is HDMI or High-Definition Multimedia Interface. Like HD-SDI, HDMI is a digital video transmission protocol that supports the communication of a fully HD (1080p or greater) video signal along with different types of digital audio. An HDMI cable is shown in figure 2.2. Unlike any of the SDI protocols, HDMI provides packetized data. While running over HDMI cables, signals are typically encrypted with HDCP or High-bandwidth Digital Content Protection. This is an encryption scheme to ensure that video data running over the cable is difficult or impossible to listen in on or copy. Originally developed by Intel, HDCP has seen widespread use in the market today. In theory, as long as the input video card in the server supports and is compatible with HDCP, the end user will have no issues using the system.

Figure 2 - 2: An HDMI Type A connector. From [8]

Due to the intrinsic nature of large format displays as a whole, any input protocol must be able to support the amount of video data a very high pixel count display would call for. Also a sizable amount of high quality video production and transmission gear has already adopted digital interfaces. A comparison chart expanding on the details and differences between the various digital video standards is found below.

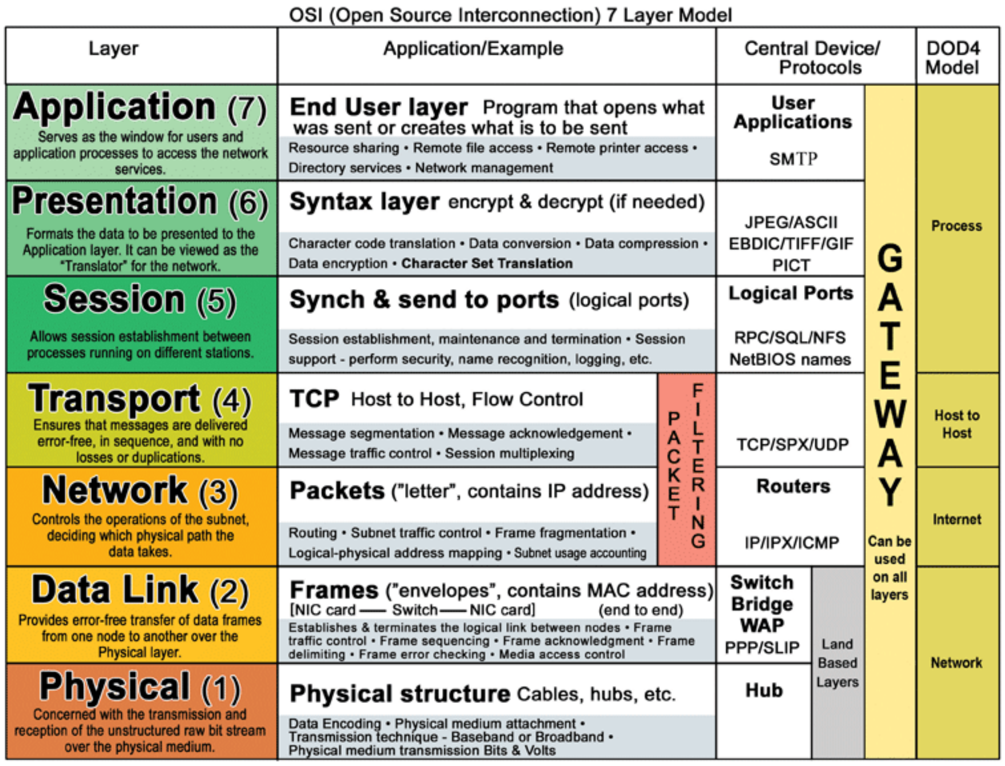
Table 2 - 1: A table showing different digital video standards and some of their specifications.

Standard:	Max Image Size:	Max Data Speed: (Mbps)	Audio:
SD-SDI	470i, 576i	360	No
ED-SDI	480p, 576p	540	No
HD-SDI	720p, 1080i	1520	No
3G-SDI	1080p	3041	No
DVI-D	UXGA (1600x1200)	5068	No
HDMI	WQXGA (2560×1600)	18432	Yes

The third most common digital video standard is DVI or Digital Visual Interface. DVI is flexible standard can accommodate both analog and digital either separately, DVI-A or DVI-D, or together, in DVI-I. There is no maximum cable length specified in the DVI standard, though in general for best performance, cables should be kept shorter than 15 feet. HDMI and DVI are very compatible with one another with some exceptions. For instance, DVI lacks the ability to carry digital audio, a main feature of HDMI. HDMI also lacks the ability to carry VGA, something found standard in DVI.

2.2 Distributed Redundant Networks

As networks become larger, their inherent complexity grows as well. Often times, larger networks also demand large amounts of up time and reliability. However, as size increases, so do the number of potential points of failure. Redundancy must be built into large networks to maintain stability and a high level of fault tolerance. Adding in redundant nodes is relatively simple, as networks are designed to handle multiple clients. However, adding multiple paths for data to flow becomes difficult when traditional network ARP tables are used. Using the OSI (Open Systems Interconnection) network model, shown in figure 2.3, hardware must keep a record of what devices it is connected to via Layer 1. This occurs on Layer 2 of the OSI model. As switches have many devices connected to them, they must maintain a large table of connections, called an ARP (Address Resolution Table), which. Associates an IP address on Layer 3 with the hardware addresses found on Layer 2.


OSI (Open Source Interconnection) 7 Layer Model

Layer	Application/Example	Central Device/ Protocols			DOD4 Model
Application (7) Serves as the window for users and application processes to access the network services.	**End User layer** Program that opens what was sent or creates what is to be sent Resource sharing • Remote file access • Remote printer access • Directory services • Network management	**User Applications** SMTP		**G A T E W A Y** Can be used on all layers	Process
Presentation (6) Formats the data to be presented to the Application layer. It can be viewed as the "Translator" for the network.	**Syntax layer** encrypt & decrypt (if needed) Character code translation • Data conversion • Data compression • Data encryption • **Character Set Translation**	JPEG/ASCII EBDIC/TIFF/GIF PICT			
Session (5) Allows session establishment between processes running on different stations.	**Synch & send to ports** (logical ports) Session establishment, maintenance and termination • Session support - perform security, name recognition, logging, etc.	**Logical Ports** RPC/SQL/NFS NetBIOS names			
Transport (4) Ensures that messages are delivered error-free, in sequence, and with no losses or duplications.	**TCP** Host to Host, Flow Control Message segmentation • Message acknowledgement • Message traffic control • Session multiplexing PACKET FILTERING	TCP/SPX/UDP			Host to Host
Network (3) Controls the operations of the subnet, deciding which physical path the data takes.	**Packets** ("letter", contains IP address) Routing • Subnet traffic control • Frame fragmentation • Logical-physical address mapping • Subnet usage accounting	**Routers** IP/IPX/ICMP			Internet
Data Link (2) Provides error-free transfer of data frames from one node to another over the Physical layer.	**Frames** ("envelopes", contains MAC address) [NIC card —— Switch —— NIC card] (end to end) Establishes & terminates the logical link between nodes • Frame traffic control • Frame sequencing • Frame acknowledgment • Frame delimiting • Frame error checking • Media access control	**Switch Bridge WAP** PPP/SLIP	Land Based Layers		Network
Physical (1) Concerned with the transmission and reception of the unstructured raw bit stream over the physical medium.	**Physical structure** Cables, hubs, etc. Data Encoding • Physical medium attachment • Transmission technique - Baseband or Broadband • Physical medium transmission Bits & Volts	**Hub**			

Figure 2 - 3: A table representation of the OSI model. From [9]

In a conventional network, switches maintain a small local ARP table of clients and their respective port location, as only one IP address can be assigned to a port. This ensures that data requests can be routed efficiently and accurately. When redundant paths are added in between two switches, a phenomenon known as a bridge loop occurs. Multiple paths can cause an overflow of the ARP tables in extreme cases. However, as switching hardware and clients generally send broadcast packets at regular intervals, having loops in a network would cause the packet to travel the network forever. As these packets are exclusively routed over Layer 2, the packets cannot have a TTL (Time to Live) value, and therefore are only dropped when the network reaches full capacity due to an exhaustion of switching capacity. The net effect of this situation is commonly referred to as a “runaway network.” The network soon becomes consumed with broadcast packets that cannot be destroyed, rendering the network unusable for standard traffic.

Solutions exist in the IEEE 802 specification to allow for path redundancy while eliminating the ability for the network to be consumed with superfluous packet traffic. Spanning-Tree Protocol, defined by IEEE standard 802.1d, adds a lightweight framework for switching hardware to virtually connect and disable various redundant pathways as needed by the network topology. As this protocol takes time to converge and disable the appropriate paths, the specification was revised to IEEE 802.1w or Rapid Spanning-Tree Protocol. RSTP is capable of optimizing a network full of redundant pathways within 30 seconds, making it Ideal for time-critical setup applications. This works by examining the redundant physical paths, and virtually disabling some paths until they are required to be activated again due to a failure of another neighboring path. Examples of this operating are visible in figures 2.4 and 2.5.

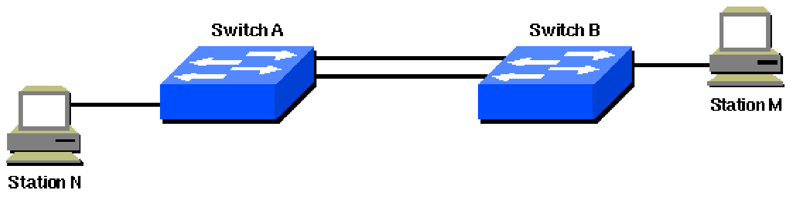


Figure 2 - 4: A physical connection before RSTP. From [10]

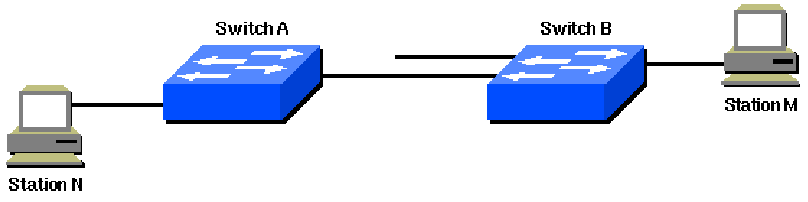


Figure 2 - 5: A virtually-removed connection after RSTP. From [10]

RSTP works by first selecting a central switch called the root bridge. In traditional network, this is configured on the hardware manually by a system technician. In some cases, there may also be redundant root bridges if the initial root bridge fails. In a context where the network topology is constantly changing, however, this may not be practical, as the network would require manual configuration for every major topology change that affected the presence of the root bridge.

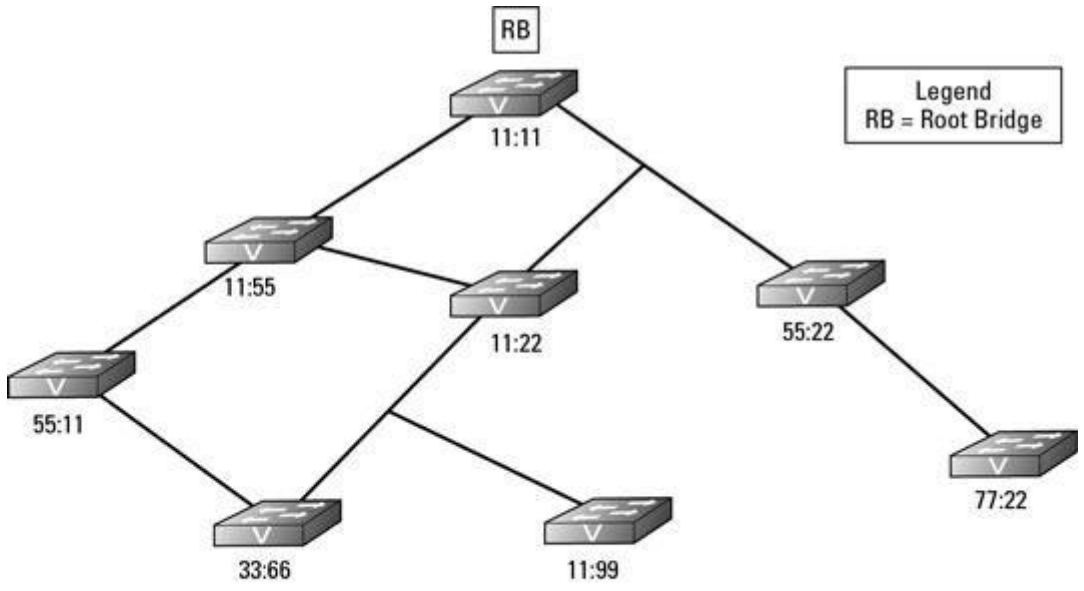


Figure 2 - 6: In a STP-enabled network, switches are assigned various weights, and paths are assigned various weights as well. Paths with the longest weights are disabled until lower-weight paths become unusable due to network topology changes or other path failure. From [10]

802.1d/w allow for options that support the automatic determination of a root bridge, however. In this situation, during the initialization of a network, every switching device undergoes what is aptly named, a “root war.” During the root war, the switch with the lowest MAC address is determined to be the root bridge. After this occurs, the rest of the path optimization occurs. A configuration such as this is ideal for large networks with highly-varying topologies from initialization to initialization. While this technology is commonly implemented in enterprise-level network infrastructures, this is seldom implemented in live audio and video applications. This can be primarily attributed to the fact that few video systems actually transmit video content over IP and secondarily attributed to the fact that the primary considerations of the industry have been improving individual panel reliability over network reliability.

Spanning-Tree also offers some protection features to ensure a fast recovery form a failed network. One such method implemented is Unidirectional Link Detection or UDLD [11]. This operates by regularly sending out packets on a specific port, querying neighboring hardware. If the hardware fails to respond before a timeout occurs, the link can be assumed to be unidirectional. In the context of an LED array, this is undesirable, as although video content may be able to still be displayed, any traffic returning to a controller to notify the system of a fault might be blocked. If a link is determined to be unidirectional, it is thrown into an error state, and any redundant pathways are enabled, ensuring bidirectional connectivity between nodes.

2.3 Large-Format Display Topologies

Large format displays come in many different shapes and sizes. Most displays in use today are flat and rectangular. This is the easiest format to implement due to the intrinsic issues that start to arise when curved or other non-quadrilateral formats are used. However making large format displays curved is not impossible. The ABC Times Square Studio, shown in Figure 2.7, is one such example where multiple different curves can be incorporated into one whole display. Displays such as the one in Times Square are traditionally purpose built for a single installation and are rarely used in different configurations once installation is complete.

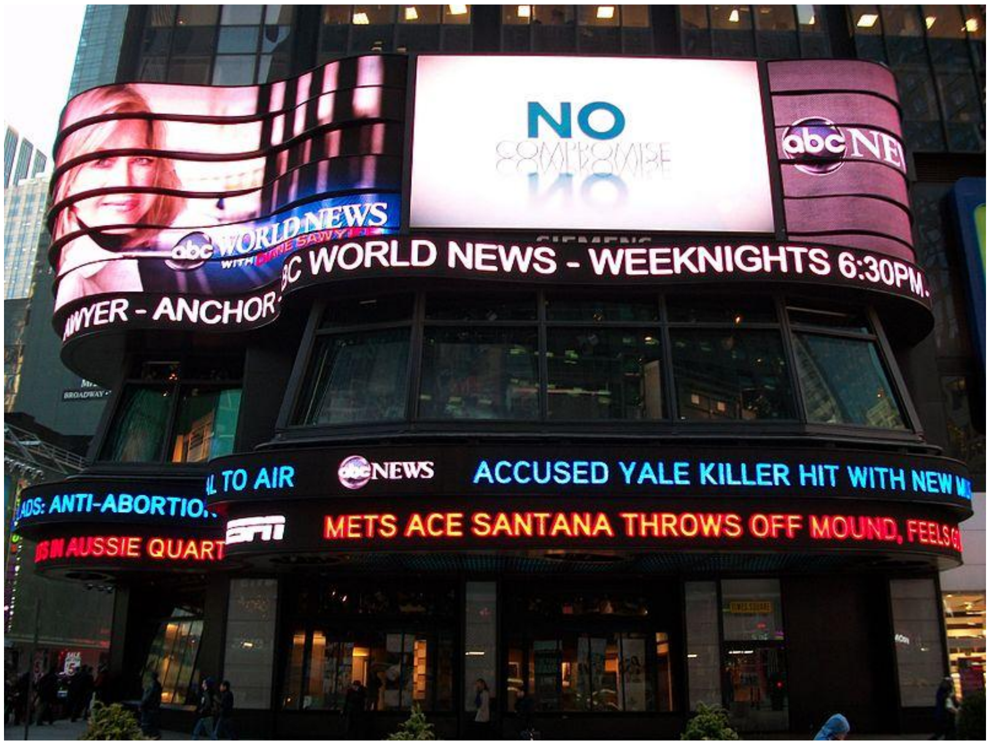

NO
COMPROMISE
abc NEWS
abc WORLD NEWS
WORLD NEWS - WEEKNIGHTS 6:30PM
AWYER - ANCHOR
TO AIR
ACCUSED YALE KILLER HIT WITH NEW
ANTI-ABORTION
AUSSIE QUART
METS ACE SANTANA THROWS OFF MOUND, FEELS

Figure 2 - 7: The ABC Times Square Studio in NYC, New York. From [12]

Other curved displays include the Yas Marina Hotel, shown in Figure 2.8, which is an example of a large LED display taken on a massive scale. The 85,000 square foot hotel is built over the top of the Abu Dhabi F1 Racetrack, and sports a multi-color LED exo-skeleton, allowing the hotel management to put on a massive light show at night. With 5,389 pivoting square LED panels, the hotel can display any manor of video or color on its exterior. The hotel remains today as one of the only of its kind in the world. [13]

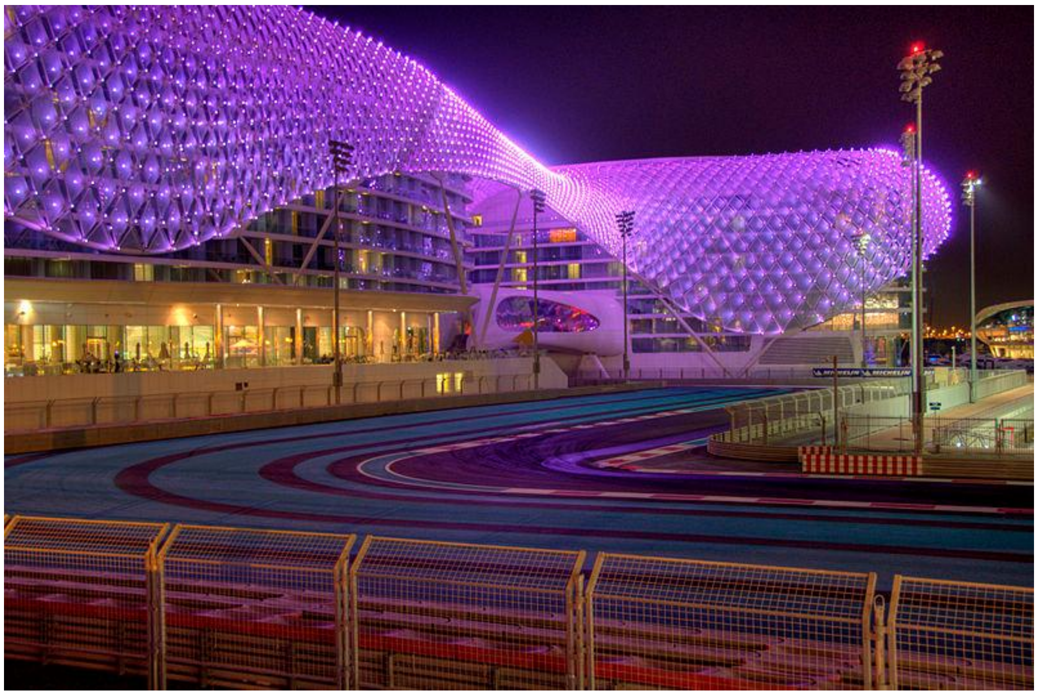


Figure 2 - 8: The Yas Marina Hotel in Abu Dhabi illuminated in a shade of purple. From [14]

While curved displays are a portion of the market, rectangular-based displays are most common in use today. Square panel based displays have the highest configurability and customization due to their building-block like nature, and their ability to be easily broken down into smaller units. The nature of square panels also have a positive impact in tessellation, in that most displays will have a rectangular aspect ratio, and building any rectangular display with square panels is quite simple. Square panels also ease the task of creating redundant networking connections. Each and every square panel can have 4 straight-line neighbors for connections plus an additional 4 diagonal neighbors if more connections are needed. With the 4 connections per panel situation, any two panels can fail without distinctly interrupting the connection to another.

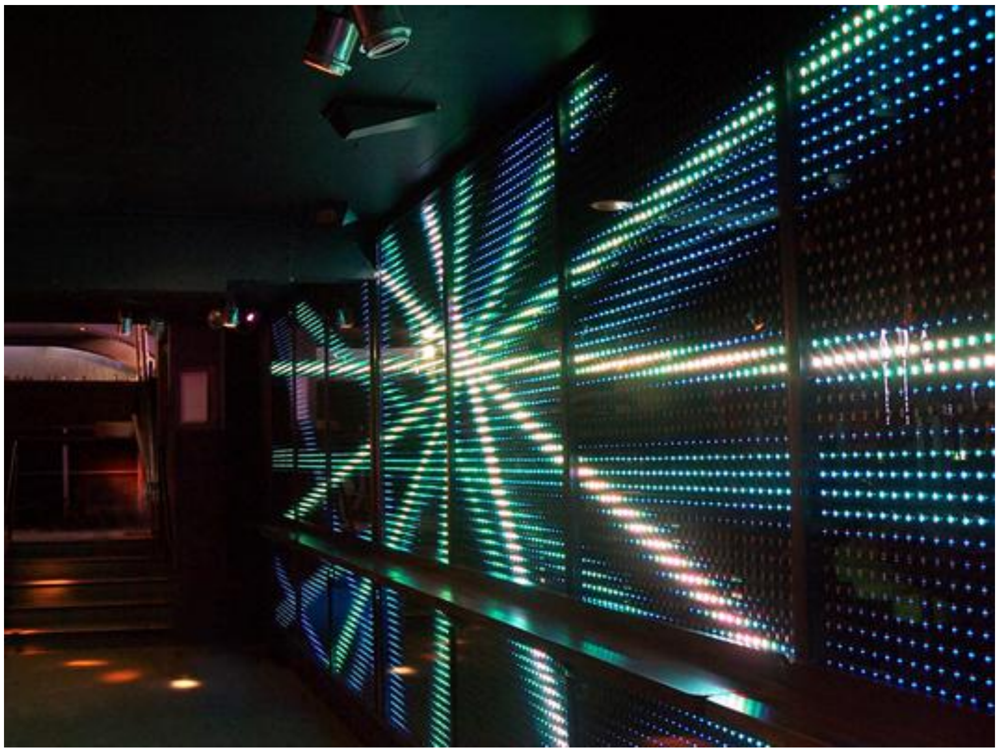


Figure 2 - 9: A rectangular set of panels mounted on a wall. From [15]

In addition to the ease of redundancy, square panels also add the convenience of rigging options. Because there are four flat edges to every panel, each neighboring panel can be brought flush together, increasing overall display coherency and decreasing the need for unique or difficult hardware for attachment. This makes square panels the unit of choice in most road shows and performances. One such setup is shown in Figure 2.9.

2.4 Video Processing Systems

In any application, it would be inefficient to route a video input source directly into each panel and perform the processing individually. To optimize efficiency, panel manufacturers often design a companion controller that serves to take in a video input source and convert it into a data format that the panels can then easily interpret. These devices usually also provide some form of transport layer and output system to directly connect to the panels. Generally, these devices do not provide power to the panels directly, although they usually are stored in close proximity to the power distribution systems for the panels.

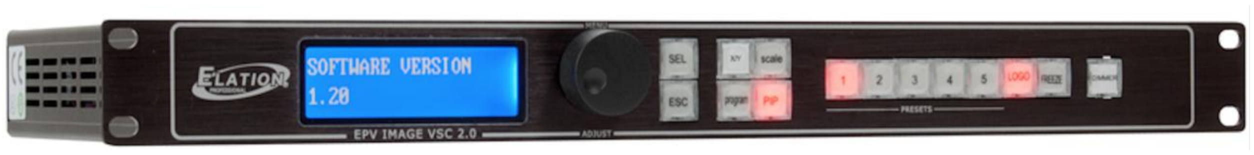


Figure 2 - 10: Elation Image VSC 2.0 Image processor used to condition video to be fed to a panel controller. From [16]

Manufacturers such as Elation [X] have proprietary video scaling and conversion hardware that is used to drive their panels. One of their image processing systems is shown in Figure 2.10. The typical system consists of a hardware video processor that conditions a video signal a resolution and color depth that is usable by the panels. This signal is then fed through conventional video cables to a panel controller that is responsible for taking in the processed video signal and converting to network frames that are routed to the array. This is usually done through implementation of differential signaling through RJ-45 twisted pairs. These devices usually only implement unidirectional communication protocols, and very few on the market implement full IP-based communication.

2.5 Summary

This chapter has shown some of the many different design points and pre-existing technology used in this project. In the first section an explanation of digital video protocols as well as a comparison of their strengths and weaknesses. Specifics to the most common protocols in use today were explained and a comparison of those details was made as well. The second section took a look at networking protocols and redundancy, and how to find network resilience in existing IEEE protocols. The IEEE specifications of 802.1D and 802.1w were explained and their effectiveness in the scope of this project was demonstrated. The third section saw an overview of large format display topologies and how the sizes and shapes of displays can affect the properties of the display as a whole. The fourth section showed how video processing has an impact on large format displays and the rolls that it plays in live video displays. This section also gave an example of a proprietary video processing rig and briefly explained how it functions.

Chapter 3: Proposed Design and Project Logistics

This section will describe the projects design constraints, overall system specification, and the methodology in determining these specifications and constraints.

3.1 Main Project Goals

LED video walls require large, complex networks that often fail. In addition to this, they require substantial amounts of effort during setup to properly address, arrange, and connect each panel. In a large network that is prone to failures, especially those in the live entertainment market, failures are visible instantly to thousands of people, making it incredibly desirable to resolve faults efficiently and quickly without compromising safety.

The goal of this project was to develop a self-addressing array of LED panels capable of recovering from network faults introduced by damaged or faulty connections. This would allow for such recovery and operator notification through the use of Rapid Spanning-Tree Protocol as a network backbone for a proprietary UDP-based protocol to relay control signals and video data to and from panels. In addition to enabling various improvements to network reliability and panel diagnostics, a secondary goal of the project was to improve the overall video quality of the output stage of the panels by investigating the viability of directly driving individual LED's, removing the need for multiplexing LED output.

This design has a significant amount of technical challenges to overcome in its implementation. They are as follows:

- **Integrating RSTP into a Compact Solution** – RSTP must be successfully implemented on each panel to effectively provide redundancy.
- **Designing a Robust UDP Layer 7 Protocol** – There currently exists no non-proprietary protocol for transmitting video frames to panels. There is also no protocol in place to handle the automatic addressing of panels and bi-directional communication to the controller.
- **Designing Fault-Tolerant, Ruggedized Hardware** – Implementation of all of the required features defined by the project will require the design and implementation of significantly specialized hardware.

- **Implementing Video Processing/Optimization** – Specialized host software will need to be written in order to efficiently control the array and provide a point of user feedback to indicate faults in a system.

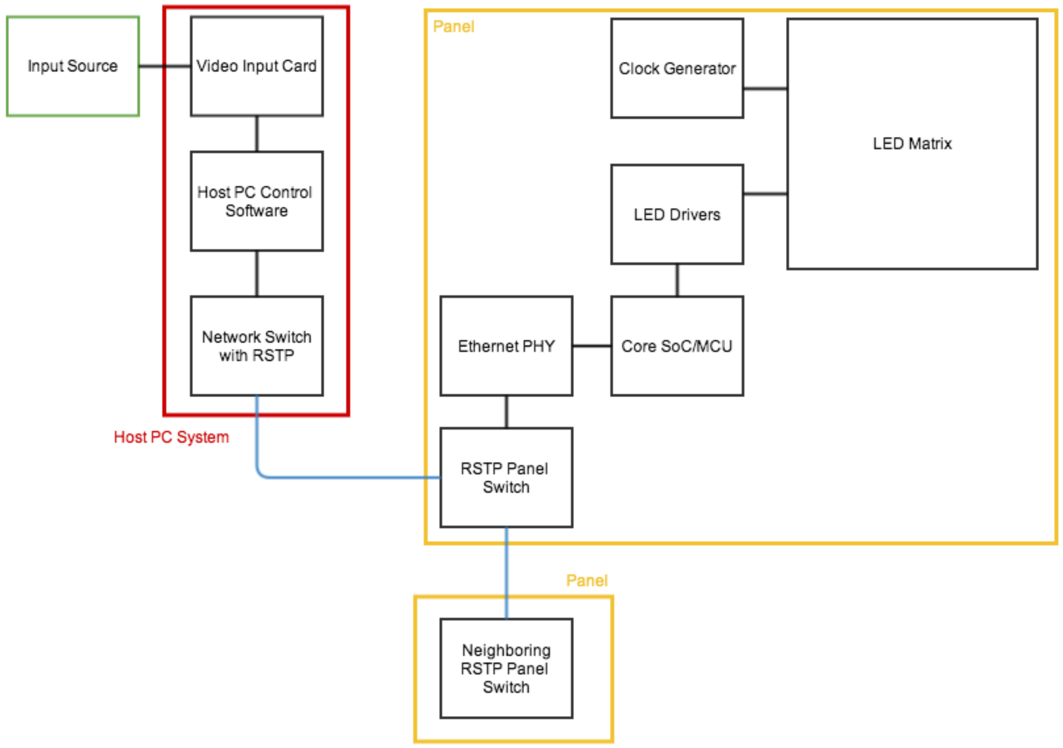


Figure 3 - 1: Proposed high-level system block diagram.

The above diagram, figure 3.1, illustrates a high-level implementation of the required subsystems of the project. As the purpose of the project is to illustrate the viability of a self-addressing, redundant panel array, only several panels will constructed to prove the concept. If required, more panels can be constructed to create larger networks. To decrease production time, a panel size of 16x16 pixels will be chosen. Larger video resolutions will be implemented by arraying multiple panels in various configurations and observing the behavior of the video content.

3.2 Project Objectives and Specifications

The primary objective for the project is to improve the redundancy and reliability of LED video walls and to remove the need to individually configure panels to work as a larger array. This will be achieved through the use of multiple network protocols, and the final system must be capable of several objectives:

- The system will use an algorithm to determine the relative location of panels in an array and dynamically map video content to the size and shape of such array.
- The system must be able to recover from a failure and be able to dynamically remap video content to the new array shape.
- The system must be able to display visually pleasing and accurate video output relative to the video input.
- The system must be able to be quickly setup and configured.
- The system must be able to notify an operator of a fault through methods other than visual detection of the failure.

These primary objectives will be reached through the design and development of several component subsystems that will be integrated together into a singular, streamlined final product. The systems designed will serve as a panel replacement as opposed to a supplemental system that could be installed onto another pre-built panel. This will enable granular control of the final design and will enable the team to develop a finished, efficiently optimized product.

The technical specifications for the final product in the areas of video quality, networking, robustness and power are defined in the subsections below.

3.2.1. Video Output:

- Variable size and shape panels
- Tri-color RGB LED with 24-bit color-depth
- At least 1000 total lumen output per panel
- Supports at least 30 frames per second
- Automatic color and brightness calibration
- Minimal effects on rolling-shutter type cameras.

3.2.2. Networking:

- Self-addresses for relative position in display configuration
- Automatic routing of video signals

- Fault-aware and diagnosing abilities
    - Dead pixel detection
    - Inter-panel fault communication
- Standard RJ-45 Ethernet connection to controller

3.2.3. Ruggedized:

- IP65 weatherproof and intrusion resistance certification.
- Riggable or stackable modular chassis
- Shock-resistant to at least IK04 [17] specifications.

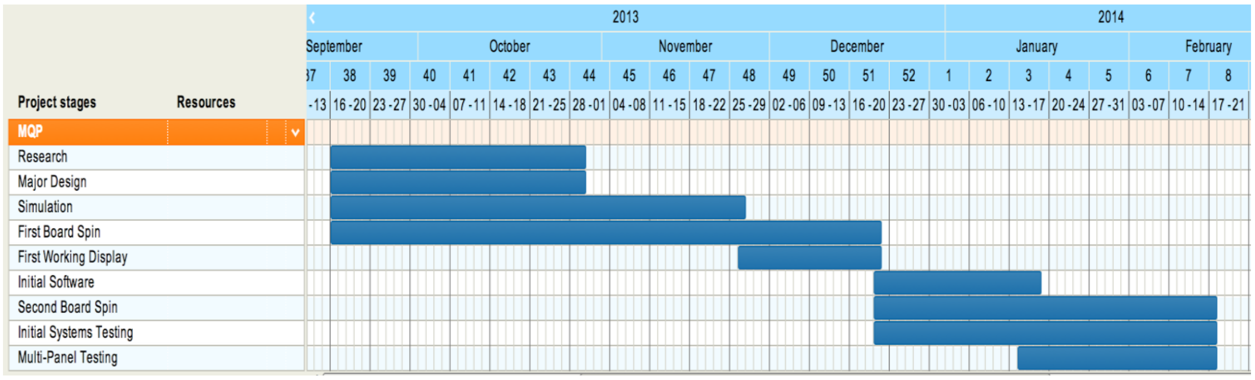
3.2.4. Power:

- Accepts 60-Hz, 90-250V AC Mains input with a bypass (out) connector
- Output capability >= 60W
- Noise filtering from/to mains to acceptable EMC standards levels
- Power monitoring and fault detection/reporting
- Protection from surges for display/logic systems
- All power rails created on-board
- Achieves >80% efficiency, 0.95 power factor

3.3 Project Management and Tasks

To optimize efficiency and improve the amount of work that could be done at any one time, the project was broken up into separate discrete sections that could be completed alone or in tandem with one or more other team members. Initial planning was done mostly to get the project started and on task with the major portions. The development strategy consisted mainly of the four team members working on four separate sections every week until the task at hand was done. This would continue until the project was completed. For some sections however, it was necessary for team members to work on their sections together and complete some tasks that were interdependent. In those cases, the team was able to communicate and coordinate times for meeting and work to be completed together. The team was able to hold numerous meetings and work sessions in the lab.

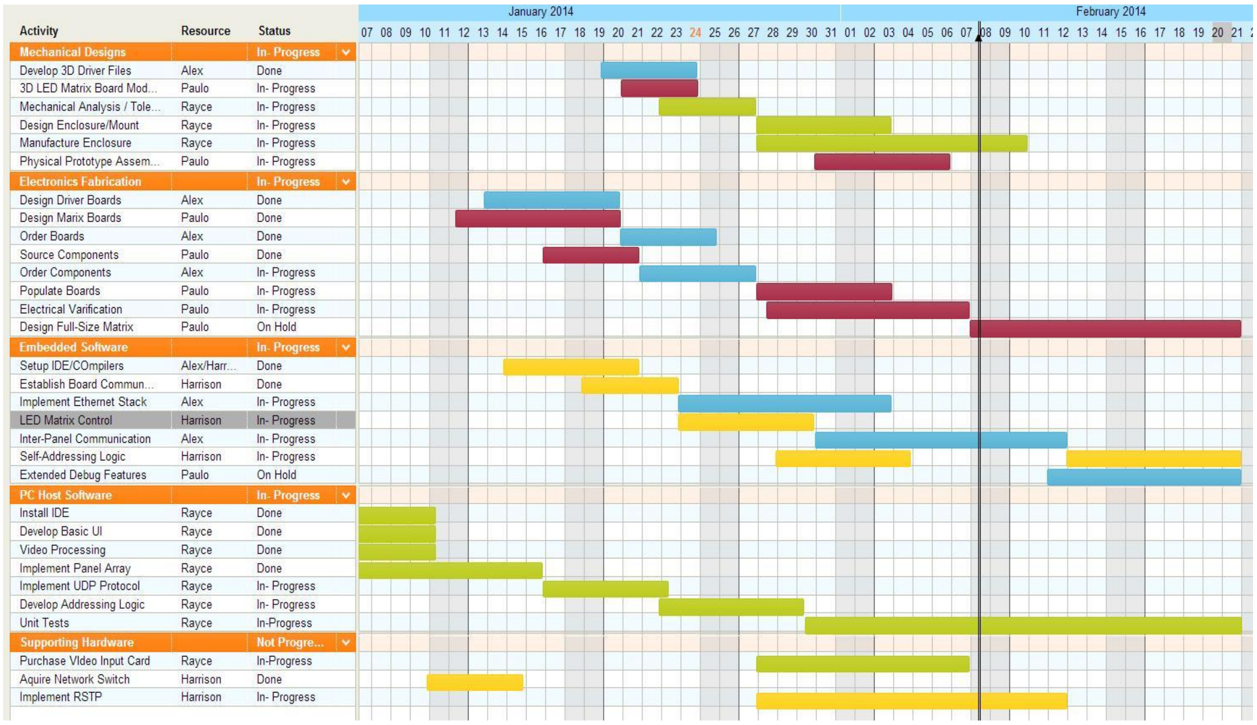
In addition to this planning, the team created a Gantt chart to mark out progress and define target dates and deadlines. The chart provided a broad task and gave it a specified time range in which to be completed in. This original chart the team designed is shown below in Figure 3.2. This shows the major sections that needed to be completed for project success. Both the major design and the research needed to be completed early and quickly. Next came a simulation of the circuit and software portions of the project. Extensive client simulation was also to be performed to ensure the successful deployment of RSTP and host video software. After that designs for the specific hardware was done and assembly of units began. Once that had been completed, unit testing and whole systems testing was performed.



Project stages	Resources
MQP	
Research	
Major Design	
Simulation	
First Board Spin	
First Working Display	
Initial Software	
Second Board Spin	
Initial Systems Testing	
Multi-Panel Testing	

Figure 3 - 2: Proposed general timeline and project schedule.

Unfortunately not everything always goes to plan, and some adjustments had to be made. An updated and more detailed schedule can be found below in figure X. Note that the first working display was achieved on February 8th with the successful completion of a multiplexing panel.



Activity	Resource	Status
Mechanical Designs		In- Progress
Develop 3D Driver Files	Alex	Done
3D LED Matrix Board Mod...	Paulo	In- Progress
Mechanical Analysis / Tole...	Rayce	In- Progress
Design Enclosure/Mount	Rayce	In- Progress
Manufacture Enclosure	Rayce	In- Progress
Physical Prototype Assem...	Paulo	In- Progress
Electronics Fabrication		In- Progress
Design Driver Boards	Alex	Done
Design Marix Boards	Paulo	Done
Order Boards	Alex	Done
Source Components	Paulo	Done
Order Components	Alex	In- Progress
Populate Boards	Paulo	In- Progress
Electrical Varification	Paulo	In- Progress
Design Full-Size Matrix	Paulo	On Hold
Embedded Software		In- Progress
Setup IDE/COmpilers	Alex/Harr...	Done
Establish Board Commun...	Harrison	Done
Implement Ethernet Stack	Alex	In- Progress
LED Matrix Control	Harrison	In- Progress
Inter-Panel Communication	Alex	In- Progress
Self-Addressing Logic	Harrison	In- Progress
Extended Debug Features	Paulo	On Hold
PC Host Software		In- Progress
Install IDE	Rayce	Done
Develop Basic UI	Rayce	Done
Video Processing	Rayce	Done
Implement Panel Array	Rayce	Done
Implement UDP Protocol	Rayce	In- Progress
Develop Addressing Logic	Rayce	In- Progress
Unit Tests	Rayce	In-Progress
Supporting Hardware		Not Progre...
Purchase VIdeo Input Card	Rayce	In-Progress
Aquire Network Switch	Harrison	Done
Implement RSTP	Harrison	In- Progress

Figure 3 - 3: Actual development schedule and timeline.

3.4 Design Decisions and Research

The large scale of the project warranted careful consideration of the design decisions. In order to construct a working system within the allotted time and budget: The team decided to follow two concurrent paths and segment the prototypes into separate modules by function. The use of well-established protocols for networking, fault detection, and fault resolution were also emphasized.

Two paths were concurrently explored and involved the topology of the LED display as either multiplexing or direct drive. The multiplexing panel would be fastest to develop due to the team's previous experience with similar systems. While the direct drive would theoretically attain better color fidelity and luminance. The multiplexing would be done by a combination of constant current LED drivers with MOSFETs that would cycle through each row and sequentially generate an image that would be sustained due its high refresh rate and a phenomena known as persistence-of-vision. In the direct drive each LED would be driven by one of the output pins of a constant current LED driver and be connected to power. This second topology would allow for every single LED to be powered on concurrently and remove the need for constantly cycling through the rows but with an added component count and the need to account for much higher currents. A more thorough design overview for the two panel topologies is presented in section 4.3 and 4.4.

To reduce the development time for the system the team decided to break down the project into separate modules by functionality. This would remove the necessity to redesign the entire system due to errors in any one subsystem. In both the cases of the multiplexing and direct drive panels the topology is equivalent and any component can be interchanged between them without the need for hardware modifications. The system consists from: An Ethernet shield board by STMicroelectronics a STM32F407 breakout board also by STMicroelectronics a custom shield board to interface between the microcontroller breakout board and the display; and, a custom display either multiplexing or direct drive. In the case of the direct drive the display was broken down into an LED containing board and eight daughter boards each of which contained four LED drivers.

Extensive analysis was conducted to determine the viability of using redundant network technologies in a mesh-networked video solution. Due to the mesh topology of the network, traditional ARP tables would not suffice and would eventually overflow with client lists. Any loop generated in the network would result in a cyclical list in the routing tables. Because of this, efficient packet routing could not be guaranteed, and the network would flood with broadcast packets. IEEE 802.1d offers protection from this. Spanning Tree Protocol is used to determine path connections and distances from a point known as a Root Bridge. From this point, path lengths are calculated and optimized. This information is then relayed to all active STP nodes, which then shut down redundant pathways, or pathways with too high of a cost (length). Generally, a network can take up to 5 minutes to stabilize and have paths converge. This is unacceptable for a live application. 802.1w or Rapid Spanning Tree Protocol offers much faster path convergence as high as around 30 seconds, depending on mesh topology and complexity.

For the network to converge efficiently, each switch can be assigned an ID number. Lower numbers have higher priority and therefore retain more weighting. Once the path finding is completed, very little network overhead is used for STP or RSTP. The network is periodically polled to ensure that it is in the same configuration as before. If a node detects a configuration change, an update packet is broadcast and paths are changed based on the availability of redundant options.

One inherent issue with STP/RSTP is that there always needs to be a Root Bridge (RB). There is no true, decentralized model. In addition to the lack of decentralization, in the context of a live touring installation, one panel must always be a RB. If this is the case, an array using multiple RBs accidentally or lacking any will not function. 802.1d/w offers a solution to this. After network initialization, all nodes will undergo “Root War.” The switch with the lowest MAC address will become the RB. Once the RB has been determined via a “Root War”, path calculations and optimization can occur. If the Root Bridge node fails, the network can recover by determining again a new RB and recalculating path weights.

IEEE 802.1d/w [18] are designed to handle a small number of redundant connections. This is not the case where each switch is connected to four other switches. This ultimately makes network mapping, path generation, and optimization become very time-consuming. Depending on the size of the client list, the paths may never converge and the network will fail. In the event that the network does become unstable, less connections can be used. A simple solution of connecting every column together in a daisy-chain like fashion would suffice. For redundancy, each column is then connected with two horizontal connections to the adjacent columns each. This greatly minimizes the number of path loops while maintaining at least two-point redundancy for all panels. This ultimately also reduces setup time and the overall weight of the flown rig. It also must be taken into consideration that different array shapes will have different network topologies, and therefore, maintaining 4-point connectivity on each panel is desirable as it offers the most extensibility, even if it is not needed in most applications.

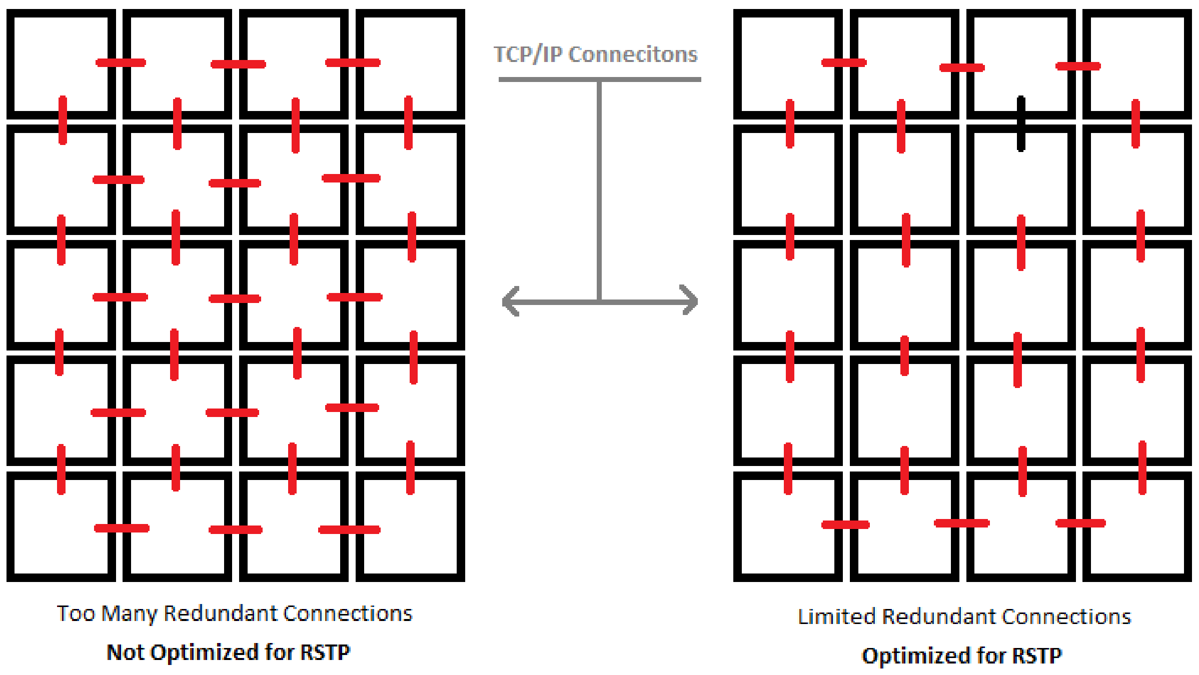


Figure 3 - 4: Network path representation in a typical configuration.

Above is an illustration showcasing one method of optimizing the array for RSTP while maintaining two points of redundancy per panel. Even if a link fails (illustrated on the right) both arrays will have a redundant layer, although path lengths and weightings may change based on the location of the point of failure. The video processing hardware has many latency requirements to satisfy real-time video applications. Minimal system delay and skew are requirements. However, large amounts of image processing are required to achieve acceptable video quality for large arrays including processes such as anti-aliasing, interpolation, and FIR Low-Pass blur filtering.

As the panels will most likely not be able to reproduce the video content with a one-to-one pixel ratio, image resampling must occur. Alone, this is relatively simple. However, simple image resampling does not produce an accurate representation of the image, as image noise would lead to inaccurate sampling. Anti-Aliasing must be performed on the source material to ensure an accurate sample is taken. Additional operations such as brightness and contrast adjustments must also be made on large pixel matrices. Because of such intensive real-time requirements, a C/C++ development environment was chosen to realize the host software that would handle conversion of video content and the control of the panels.

Preliminary research was conducted to understand the theoretical capacity of a single IP-based network. Assuming that each panel will be operating at a resolution of 32x32 pixels with a color depth of 24-bits, the overall video bandwidth can be calculated to be:

$$\frac{8\ \text{bits}}{1\ \text{byte}} \times \frac{3\ \text{bytes}}{1\ \text{pixel}} \times 32\ \text{pixels}\ \times\ 32\ \text{pixels}\ \times \frac{30\ \text{frames}}{1\ \text{second}}\ =\ 720\ \text{kbps} \tag{3 - 1}$$

Without using a checksum, the standard IP and UDP frame headers add another 28-32 bytes of data. A single frame of video content requires 24kbits of data. This exceeds the standard MTU of 1.2 Kbytes. Sending an entire video frame over one packet also adds unnecessary points of failure. In order to increase fault tolerance, smaller packets can be used. Initially, it was considered to send a packet of RGB content for each scan line of the video frame. While this offers more redundancy, the payload size of the data frame is only 2-3 times of the frame headers. This results in about 25% of the network being wasted on overhead, minimizing overall video throughput. Because of this, a compromise was made. As each panel can be broken up into 4 16x16 chunks, each video frame consisted of 4 packets containing 16x16 RGB data. This allowed for the payload to fit in the standard MTU while minimizing the relative network overhead. As the panels tested had a resolution of 16x16 pixels, only one chunk was required to be sent per video frame, verifying the protocol is inherently scalable for both applications, reducing the requirement for modifying the host video processing software.

This method is also preferred as it offers a more visually pleasing fault tolerance. As all colors are sent in one frame, the output video is less susceptible to temporal chromatic aberration. Also, a slight delay in a relatively small partition of the panel is less noticeable than the entire panel missing a frame. Various methods can be employed to minimize visual impact of dropped packets, such as holding the video content on the output buffer until a new packet is received. Although experimentation remains to verify this, it is a working assumption that a panel blacking out on a fault will be more noticeable and thus undesirable than a panel having video content freeze for 1/15 of a second.

$$\left(\frac{24\ kbps}{1\ frame}\times\frac{1\ frame}{4\ payloads}\ +\ 32\ bytes\ overhead\right)\times\frac{4\ packets}{1\ video\ frame}\times\frac{30\ frames}{1\ second}=750\ kbps \tag{3-2}$$

Each panel requires a total bandwidth of approximately 750 kbps, ignoring any control signals sent on the network. Because of this, a 100Base-TX network can support around 100-120 panels. This is ultimately dependent of RSTP network overhead, L2 broadcast overhead, and further requirements not implemented yet. For large arrays, Gigabit networks are more desirable and provide lower latency. This, combined with RSTP, creates a high-speed redundant environment compatible with common technologies.

3.5 Design Summary

Individual panel designs are being made to the specifications and requirements as they were set out earlier in the report. Network design decisions are in place so that an algorithm can be in charge of mapping the individual panels thereby removing the requirement for manual addressing. The network design will also be able to dynamically adjust and remap video to the new array shape in the event of a single or multi-panel failure. IEEE standards for protocols including 802.1D and 802.1w will be included in the network design. The panels themselves have been designed to be a part of this network, accepting Ethernet with the specified standards for their video input. The panel designs are split into two separate sub-designs depending on how they drive their LEDs, either directly driven, or multiplexed. Research has been done to indicate both have pros and cons. In either case, the video output of each panel must be of a quality such that video flicker is not observable to either the human eye, or to a camera that may be recording the wall. All connections on the panels and on the host must be easy to use and quick to setup so that the setup time is reduced as much as possible. Software and hardware is designed to alert the user of faults in the system and panels.

Chapter 4: Implementation

This chapter will detail the technical methods and designs used in the final implementation of the project. It will also further detail the methods used in determining the designs implemented.

4.1 System Hardware Configuration

4.1.1 Host Video Processing Rack

The video processing server had to be mounted in a safe rugged location. As such, the use of an 18U rolling road case rack was used. Having the main computer mounted in the bottom helped with the center of gravity of the case while maximizing workable space in the rest of the case. For testing purposes, a 24-port 2x1000 Base-TX managed switch, shown in Figure 4.1, was installed to implement the networking functions of the project. This switch offered RSTP support as well as serial level control of most networking functions. This meant that the host computer could effectively manage the network of panels in a way that replicates a distributed full scale solution. The switch is IEEE 802.1D, and 802.1w compliant, making it ideal for use in project development.

Figure 4 - 1: The 24 port managed switch used in the rack. From [19]

To power all the devices in the case, a power conditioner was added to filter and prevent damage due to fluctuation in voltage or current spikes in the AC mains. There are also inputs to all of the necessary ports located on an easily accessible panel in the front of the case. These connections include two networking ports, a BNC connector for SDI input and 2 HDMI connectors for HDMI video input. This panel is small but still has room for other ports if expansion or additions are necessary in the future. In addition to all of these features, the rack still has room to hold some additional cabling and adapters inside.

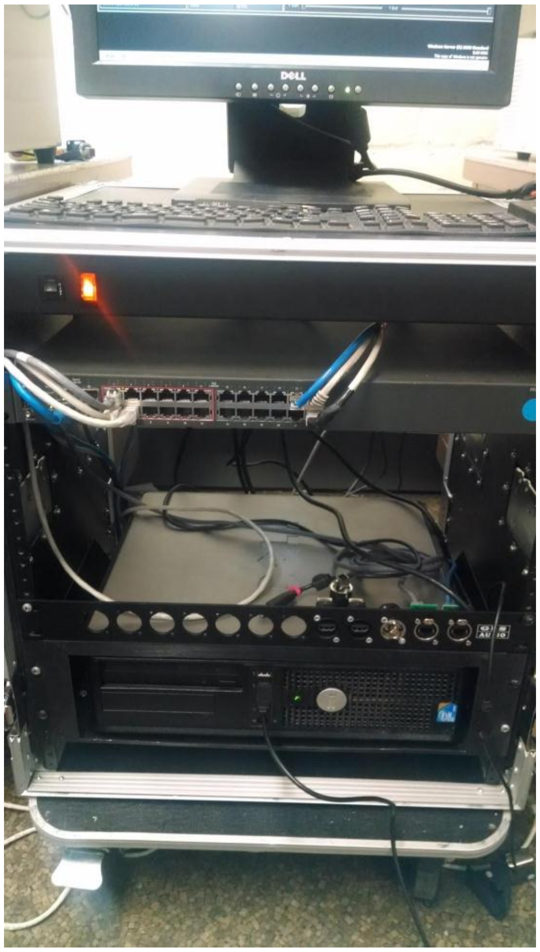


Figure 4 - 2: An image of the processing rack.

The host computer chosen was a dell OptiPlex 760, which is more than sufficient for the needs of the host software system. It possesses 4GB of RAM with Core 2 Duo Q8400 (2.66GHz 4MB L2 Cache) and is running Windows Server 2008 R2 for an operating system. This gives it an advanced capability to control the network environment and manage the connected panels. This computer served to run the video processing software.

4.1.2 Development Network Topology

Due to time and cost restrictions, it became impractical to implement switching hardware on individual panels. For the scale of the network attempted in this project, it proved more beneficial to independently test various types of RSTP networks through simulation, compared to purchasing or designing switching hardware. To accurately simulate a larger network, a series of tagged VLANS were configured on an AVAYA 5000 series switch.

This switch was configured to act as the root bridge, and have all VLANS share the same STP client pool. To more accurately simulate latency, these VLANS were isolated and configured to only be accessible to two hardware ports on the switch. This enabled the team to simulate multiple sized arrays of panels in various configurations. To simulate panel traffic over the network, one set of ports were left with the ability to access all tagged VLANS. A computer was then connected to these ports through redundant Ethernet adapters. Each of these adapters were configured to accept several tagged VLANS as individual virtual interfaces. This allowed the computer to act as a large number of virtual clients connected to different RSTP-enabled switches over the network. A small program was written in JAVA to generate random packet overhead generated by command signals and video signals being sent to and from virtual panels and a host pc. A combination of on-board system diagnostic utilities on the switch and packet sniffing software, WireShark, were used to measure the convergence times of various network configurations and sizes. While only a small number of RSTP-enabled switches could be simulated, many redundant paths were implemented in a mesh topology, ideally generating a network that was not optimized for RSTP. This would likely be more accurately representative of the typical large network formed by a large-format video wall using the panels designed in this project.

Table 4 - 1: Average RSTP Convergence Times

Simulated RSTP Switches	Simulated Clients	Average Convergence Time
1	3	1 ms
3	9	5 ms
6	12	478 ms
8	16	792 ms
10	16	1003 ms
14	20	1734 ms

The bandwidth calculations conducted in Chapter 3 show clearly show that the maximum carrying capacity of a Gigabit network is around 1000 to 900 panels. This would likely yield approximately 40 physical loops in the network for RSTP to resolve. The simulations ran had a maximum of 20 loops within a network for RSTP to resolve. On a network with minimal traffic, the research suggests that total convergence times for RSTP-enabled panels would be under 20 seconds. This is considerably faster than the time required for manual network configuration of the array topology in traditional array network paradigms. To further support the validity of using RSTP as a networking solution, these results were obtained during situations when 70% of network resources were allocated to the switching of video content and control packets. In an ideal initialization scenario, the panels would not emit any control frames, and the host pc would not transmit any video frames until it was detected that the network converged and stabilized. It is anticipated that total network convergence times would significantly decrease on a network that was able to afford all of its resources to the path finding and convergence of RSTP alone.

To adapt these findings to the context of this project, a singular-RSTP switch was used, emulating the network overhead found in networks that have RSTP enabled. This switch was then directly routed to each panel over a series of isolated, VLANS. These VLANS were virtually connected through software, simulating multiple smaller RSTP switches on a network. This switch, combined with the host computer serving as a DHCP server, provided all of the networking resources used in the project. One factor that was not simulated in the small scale of the project was relative packet latency between individual panels across an array. This theoretically could be minimized by assigning a central panel as a root bridge and as the primary entrant point for packets from the host PC. The array could then be interconnected as such to enable for as even of a temporal packet distribution as possible, removing or minimizing jitter and latency from the system. Depending on the magnitude of latency observable across the array, visual tearing might occur, generating effects similar to poor tracking on common VHS systems. To prevent this phenomenon, the video frames could be fitted with a timestamp. Panels would delay output of the video frame by the longest arrival time of any panel in the array from the absolute timestamp of the original send time from the host pc. This would introduce a potentially significant delay in the output of the panels, but would serve to temporally align the output frame display times. This technique is similar to phase alignment in large audio systems. In the context of large entertainment events, video content with delays of around 200 ms or less is acceptable, as sound takes time to travel, the light emitted from the panels would be time aligned with the perceived audio by an individual at a significant distance from the video wall. Individuals placed even further from the array would perceive the video content as occurring before the sound reached them. Ultimately, it is these admissions that allow for a large degree in flexibility of signal processing, although it is still desirable to minimize required system latency. Introducing additional latency is a feature that will be accessible to the user for such audio alignment purposes.

4.1.3 Video Input Hardware

As most computers to not come equipped with a HDMI or SDI input interface, one was selected for the project. In order to improve speeds, a PCI Express card was selected over a USB option. Specifically, a BlackMagic DeckLink HDMI and HD SDI input card was chosen to interface common video equipment with the host PC and the related software. This offered high frame rates and provided a fluid SDK to streamline the video input stream into the PC software discussed in section 4.5. Below is an image of the product.

Figure 4 - 3: BlackMagic SDI Input Card. From [20]

4.2 Display Controller and Shield Implementation

4.2.1 Display Controller Implementation

To simplify the development of the LED panel, a commercial off-the-shelf (COTS) microcontroller development board was chosen to be used. The STM32F4 Discovery, from STMicroelectronics, pictured below in Figure X.Y, features STMicroelectronics's STM32F4VGT6, a deluxe, 168-MHz system clock-capable, ARM Cortex-M4-based microcontroller. It was chosen as it had more than every feature needed, including a built-in Ethernet MAC, nearly 200 kilobytes of SRAM, and numerous serial interfaces required, while being only $15 each.

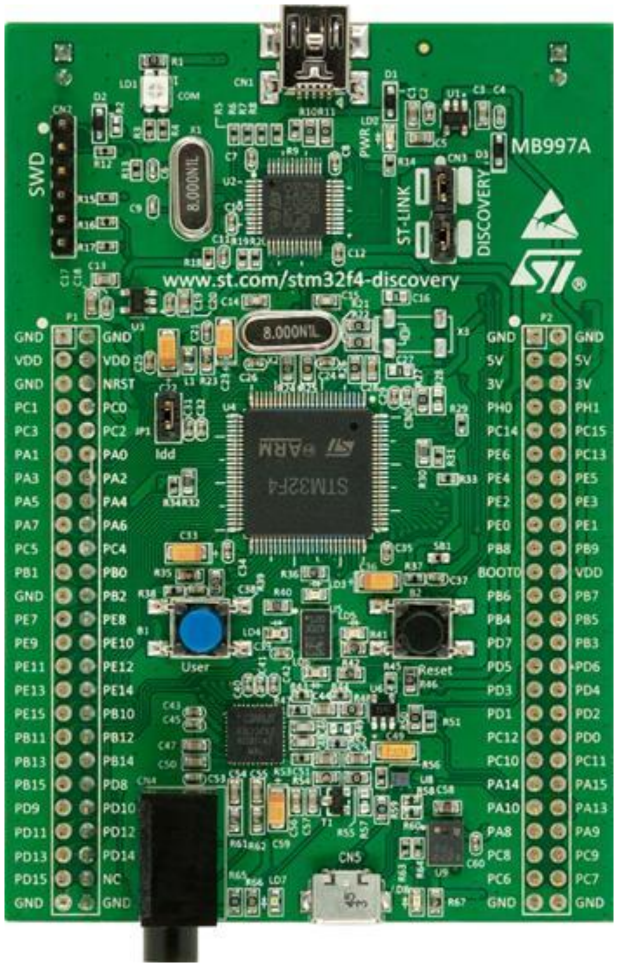


Figure 4 - 4: STMF4 Discovery Development Board. From [21]

To make use of the Ethernet MAC, the Discovery was paired with Embest's STM32F4-BaseBoard, a plug-in expansion PCB that provides an Ethernet PHY and

MagJack, plus an SD Card slot, RS-232 port and expansion headers. The decision to use the baseboard was based on the desire to remove risk in the implementation of the Ethernet section, which was considered risky due to the complex interface involved between the MAC and the PHY.

4.2.2 Display Controller Shield Implementation

As neither the Discovery nor the expansion baseboard had the desired connectors to be used to interface it with the LED panel itself, the concept of using a small shield board to bridge the two was followed through. The shield board attaches to one of the available 40-pin expansion headers that are on the baseboard. The full setup is shown in Figure X.Y. It encompasses five signaling LED's to be used for diagnostics, four isolated, optically-coupled RJ11 jacks used for direct UART-based inter-LED-panel communication, and a Flexible Flat Cable (FFC) connector to connect the controller.
The complete setup is shown in Figure 4.30 in Section 4.7.

4.3 Multiplexing Panel Architecture

4.3.1 Multiplexing Panel Design Overview

The multiplexing LED panel design was the design that was implemented and primarily tested for the operation of the LED display as a whole, while also being the cheapest and simplest to build as it requires the least number of components to operate. A rendering of the style of panel envisioned is shown in Figure 4-5.

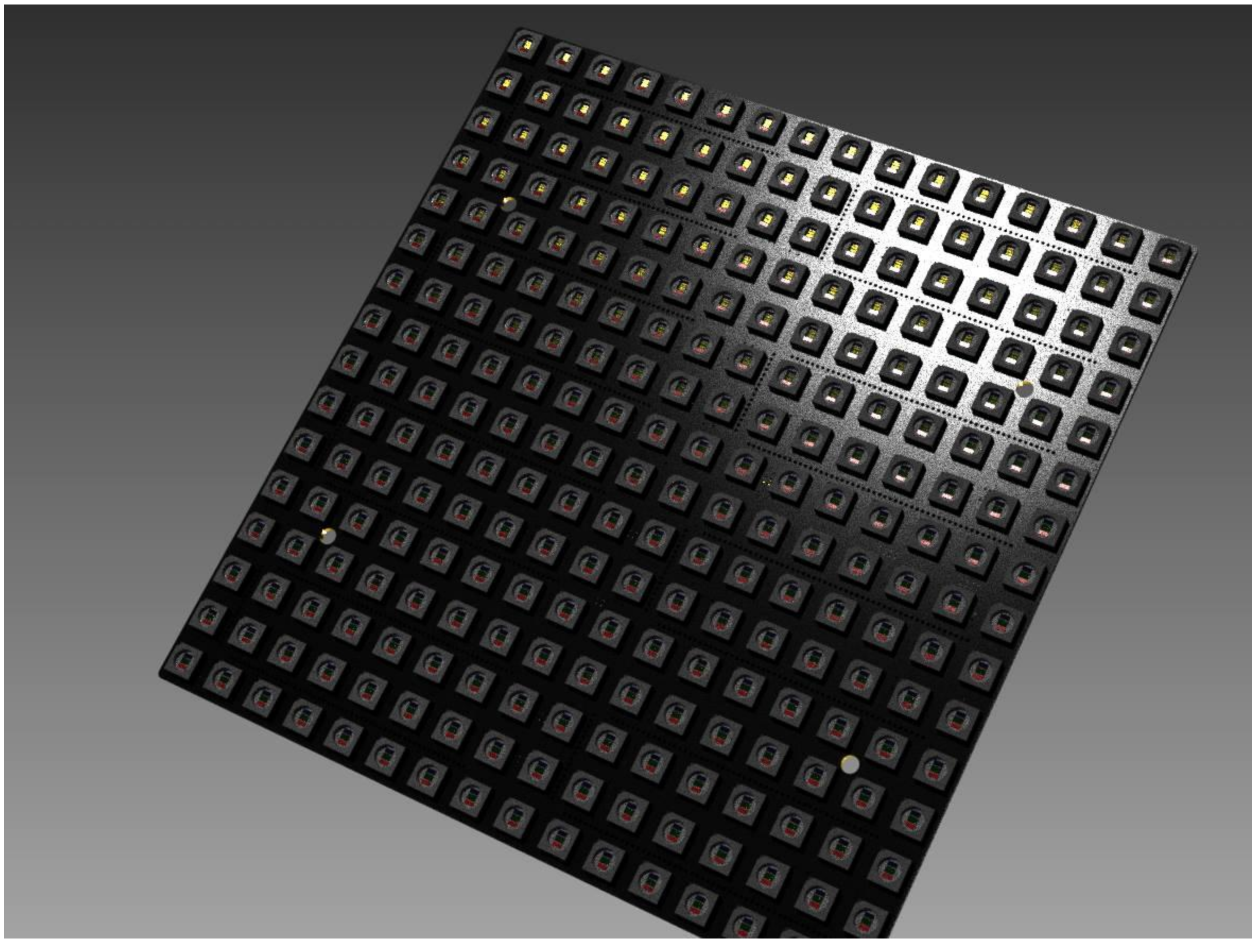


Figure 4 - 5: A 3D concept of the output matrix face of the multiplexing and direct drive PCBs. It should be noted that the mounting holes position on the multiplexing design differs slightly from that of the direct-drive design discussed in later sections.

4.3.2 Multiplexing Panel Functional Block Diagram

Figure 4-6 depicts the functional block diagram for a proposed, 32x32 multiplexing LED panel design. The design consists of X separate sections, namely: the Tri-Color LED Matrix, the LED Column Drivers, the Row-Select Switch composed of MOSFETs, gate drivers and the driver decoder, the Power Regulator, and the PWM Clock Generator. The actual constructed panel is 16x16, but other-wise identical.

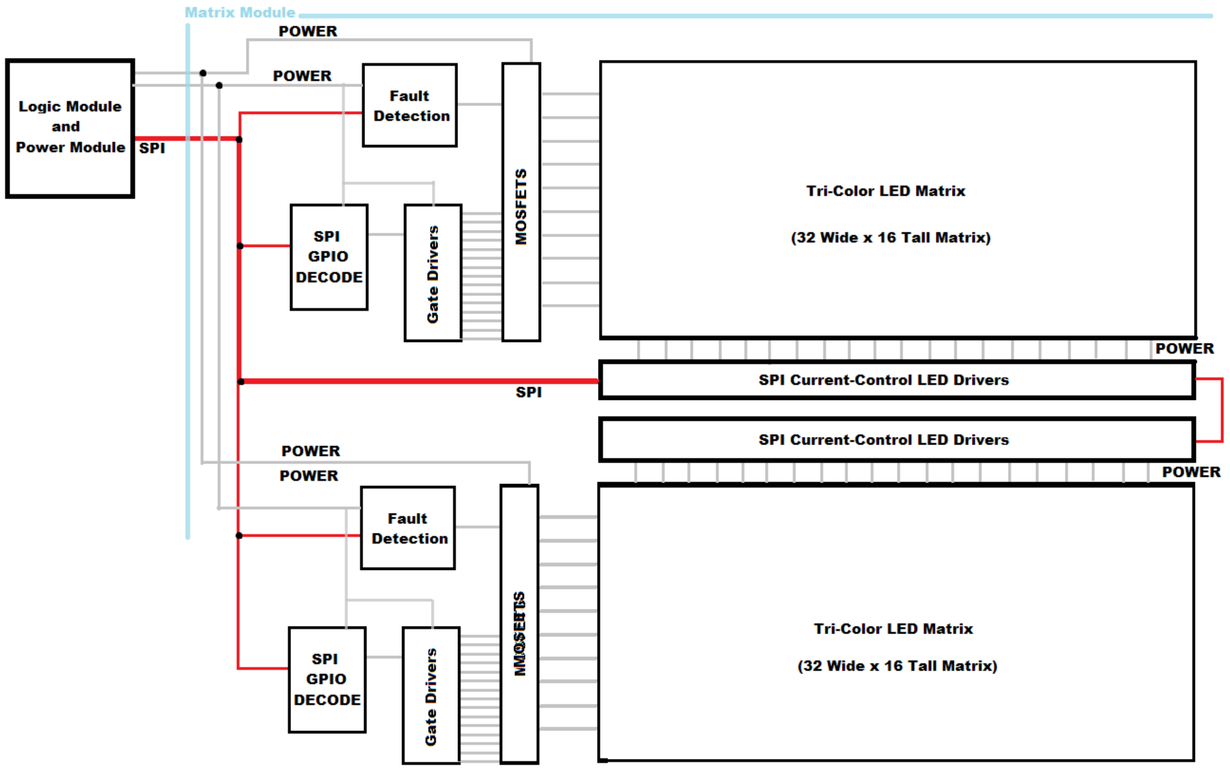


Figure 4 - 6: Functional Block Diagram of a Proposed 32x32 LED multiplexing panel.

4.3.3 Multiplexing Panel Theory of Operation

The multiplexing LED panel display, which would be used for most of the software testing, is based on a switch-matrix-like design. It has sixteen (16) rows of sixteen (16) columns of tri-color RGB LED's. Color control is provided via pulse-width modulation (PWM), theoretically providing up to 24-bit color control for each LED.

For LED control on this design, each LED driver integrated circuit (IC) had a single color to control. For each of the sixteen (16) *columns* of LED's, the cathodes (negative terminal) of each color are tied together, and connected to the current-sink outputs of the corresponding driver, one per column. For example, each red LED in column one is connected to output one on the LED driver controlling the red color. The anodes (positive terminal) of each *row* of LEDs are tied together and connected to discrete power MOSFETs, which are connected to the main power rail. Turning on a MOSFET allows current to flow from the main power rail to all sixteen tri-color LED's in a single row, through them to each driver which sinks the current to ground. After each MOSFET, and thus a row, is on for a set time, the “on-period”, the entire row is turned off. This is followed by the next MOSFET in order being turned on, until each MOSFET row has been scanned through, similar to television sets. The process is repeated at a high enough frequency that the scanning can’t be perceived by the human eye.

During the on-period, each LED driver current-sink output can either be sinking current or turned off, reducing the current through that specific columns LED to near-zero. Thus, the effective on-time of each LED can be modulated from a relative zero to one hundred (0 to 100) percent.

One key drawback of this design, naturally, is that only one given row can be turned on at any point in time. This reduces the effective average current that each LED can have through it at any point, as compared to each LED having its own independent control signal like in the direct-drive architecture. The maximum on-time percentage for each row is given in (4-1) below:

$$\mathrm{On-time_{Max}} = \frac{100\ \%}{\#\ \mathrm{of\ rows}} \tag{4 - 1}$$

This equation does not account for the time lost due to the required blanking period, whose percentage dependence depends on the scanning frequency. For the 16-row design used in this project, the maximum on-time calculates out to be approximately 6.25%. This percentage, multiplied by the maximum full-scale current output of the LED driver, results in the maximum effective (average) current through the LED, given in (4-2). Assuming a full-scale current of 20mA, the maximum effective current per LED for this design is 1.25mA.

$$\mathrm{I_{Effective-max}} = \mathrm{On-time_{Max}} \times \mathrm{I_{Full-scale}} \tag{4 - 2}$$

As a result, one of the key limitations of the multiplexing display architecture is the limited effective current that can flow through each LED, and thus limits the maximum brightness of the display. This can be compensated for by increasing the full-scale, instant current that flows through each LED during its on-period. However, an instant-current beyond the absolute maximum rating (AMR), typically 30 mA, can significantly reduce the lifespan of the LED, which would greatly reduce the value of the displays because they would need constant replacement or rework, and so there aren't many options to increase the average current short of reducing the number of scanned rows by adding more drivers.

The maximum effective current through the LED can be adjusted with PWM control. For the multiplexing design, the software has 8-bit resolution, although the hardware supports up to 12-bit resolution. Thus, the software provides an effective current resolution of 4.88 microamperes.

Due to nonlinearities between the instantaneous current flowing through LED and the absolute brightness of the output, the PWM resolution does not practically give close to the full 24-bit theoretical color control that is possible with linear LED's. An example current-brightness curve provided by the manufacturer of the LEDs used is given in Figure 4-7. Although the graph is linear, it doesn't account for the non-linear AC effects that are present in switching LEDs at a high rate. It's also been empirically noted to be inaccurate, as LED tolerances and other variations also result in changes between each LED for this curve as well.

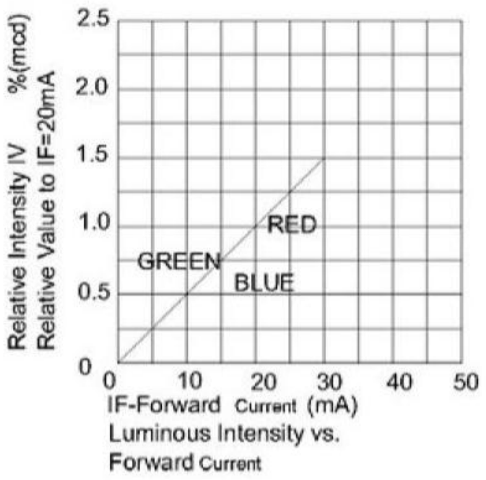


Figure 4 - 7: Led brightness curve. From [21]

4.3.4 Multiplexing Panel Component Selection and Description

4.3.4.1 LED Drivers

Texas Instruments (TI) TLC5490 16-channel LED drivers were chosen to be used for the multiplexing design based on past experiences with them in a proof-of-concept LED panel that was created in early 2013 for an Independent Study Project (ISP) at WPI separate from this MQP. They're also very affordable, at approximately $1.20 in 1000 unit volume. They feature a simple serial interface, as shown in Figure 4.6, which is compatible with most microcontrollers, including the hardware SPI modules that are included in STM32F4 microcontroller that we had chosen. By utilizing the chaining feature of their serial interface, the microcontroller was able to control all three drivers with a single serial bus, reducing the number of pins required for this task.

The TLC5940 contains 16 constant-current sinking output channels. A reference current for the output channels is created via an on-chip voltage reference and a supplied reference resistor. This reference current, multiplied by 31.5 internally, becomes the full-scale current output for each channel. Each channel can be individually programmed via the serial interface with a 6-bit word to sink between 0 and 100% of the value of the full-scale output current. The individual channel current programming feature is currently unused in the implementation, with each channel set to the same current. This is because the multiplexing architecture prevents individual LED's having their instantaneous current tuned to provide uniform colors and brightness for a given PWM value, because 16 LED's are connected in parallel to each channel's output, and the constant-current programming takes several milliseconds. This is too slow to adjust on-the-fly to each scan line, and as a result, color calibration is done through PWM duty cycle modification only.

Based on the device recommendations, a 1% tolerance reference resistor of 2-kOhms was used – this allows a full-scale output of 19.5 mA, with a settable 6-bit instantaneous current resolution of 305 microamperes. The intrinsic error in the LED driver output current accuracy prevented the team from choosing a higher-tolerance resistor and getting any more meaningful current accuracy.

Actual color control for video and image display is achieved through the grayscale PWM feature of the TLC5940. A PWM clock is provided to the driver, which it uses to increment a 12-bit counter. This counter is reset to and held at zero during the blanking period with a dedicated blanking input pin. Once the blanking pin is disabled, the counter begins incrementing again, and each channel's output is enabled. The counter is compared to a 12-bit programmed value for each channel – once a specific channels value is less than the current counter value, the channel is shut off and current ceases to flow through the output pin (and consequently, the LED for that column).

4.3.4.2 Grayscale PWM Clock Generation

To make use of the PWM color control feature of the chosen LED driver, a CMOS-level clocking source had to be provided. Two solutions were settled upon – one involved adding a single common CMOS oscillator to the LED panel, as suggested by TI. It drives the clock inputs of each driver. A 24-MHz version, the CB3LV-3I-24M0000 from CTS-Frequency Controls, was selected. It provided a stable, always-on, 50-ppm accurate clock source for color control. This was selected as allowing the operation of the 12-bit counter to reach its maximum value between each blanking period resulted in a scan rate per row of approximately 5.86-kHz, or 366-Hz for the entire display. This is more than fast enough for the human eye to not notice any refreshing of the display, and was hoped to be fast enough as to not be picked up by any digital cameras observing the display work.

The second option was to provide the clock source from the display controller directly over the controller-to-panel cable. This option would allow the software on the controller to adjust the clocking frequency to the ideal value, which could have been some other value than 24-MHz, or deal with any issues caused by using a dedicated clock source. An optional resistor bridged the clock line and the cable, allowing the team to choose which source to populate, and thus use.

4.3.4.3 LEDs

The LED’s selected for use are standard RGB tri-color, 5050 PLCC-6 packaged SMD LEDs. The LED’s are mounted flush to the LED panel on the front-side, enabling the use of a reflow soldering techniques and a pick-and-place machine assistance for manufacturing. Three LED’s are housed in the package, with each LEDs cathode and anode attached to its own pin. The panel connects each anode together to form a row, and each colors cathode into a column, connecting to the LED driver. The rated forward current for each LED is 20 mA, with a 30 mA peak. The forward voltage on the red LED typically is between 1.8 and 2.2 volts at the rated current, and is 3.0 to 3.4 volts for the green and blue LEDs.

4.3.4.4 Power MOSFETs

In order to properly turn on and off the rows of LED’s, a high-side, low on-resistance switch was required. To accomplish this, the Vishay SI2301 P-Channel MOSFET was selected. It features a -0.45 V logic-level threshold and a tiny SOT-23 package. The low threshold voltage allows the switch to be fully turned on with a wide range of supply voltages that would be used. This MOSFET also features a very low on-resistance, with only 130 milliohms of resistance at a gate-source potential of -4.5 volts.

The exact MOSFET used is not particularly critical, and the chosen model can be swapped out fairly easily if it has the same package and pin-out. The only key parameters that needed to be observed, were that it was able to be turned on with the fairly low voltage of the system, and that it’s resistance at the overdrive voltage applied (when going from the power rail to ground) is low enough to not cause a considerable voltage drop to the LED’s in each row. The instantaneous maximum current of up to 936 mA $(3 \times 16 \times 19.5 \text{ mA})$ could have been of concern as it may drop the rail voltage enough to prevent the LED drivers from having enough voltage overhead to regulate current properly. Power dissipation was never a concern due to the low duty cycle of each MOSFET.

4.3.4.5 MOSFET Driver / Row Decoder

In order to minimize the number of pins used between the multiplexing LED panel and the shield, as well as providing the required voltage level-shifting, the team opted for using a 4-to-16 decoder. The TI CD74HC154M was chosen for this task due to its high speed and operating voltage. Its use made possible to control all sixteen MOSFETs in a deterministic fashion whilst using only six control lines. Four row-select lines were used to select the power MOSFET to enable/disable, while two additional control lines were used to disable all rows independent of the four *row-select* lines, turning off any image being displayed. To allow for the level-shifting required to turn off the logic-level threshold MOSFETs, the decoder was powered directly from the main 5V power rail. Figure 4.8

depicts the connections used for this part, while Figure 4.9 depicts the truth table for operation of the decoder.

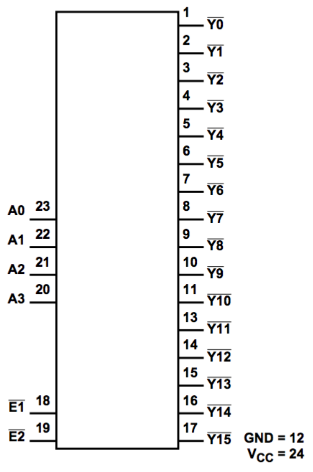
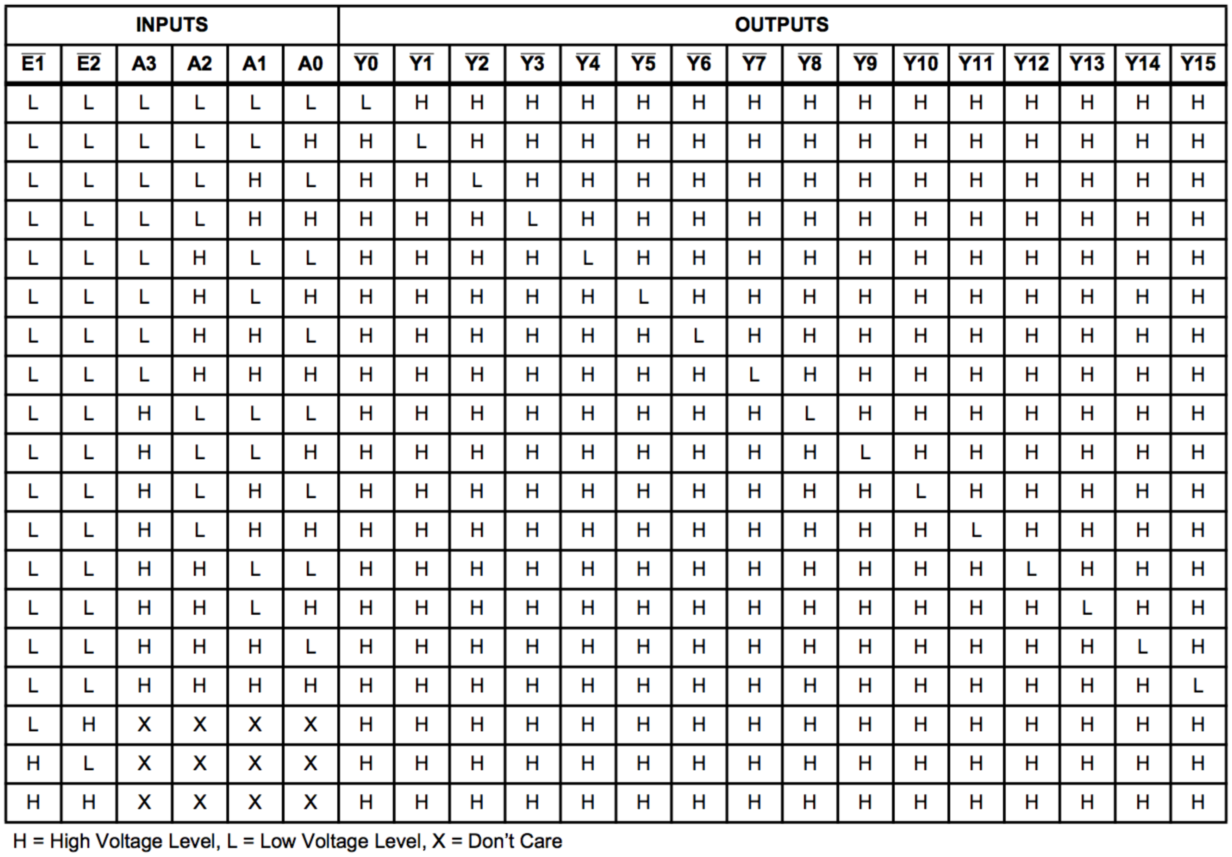


Figure 4 - 8: Pin Connections for the decoder used. From [23]



INPUTS						OUTPUTS															
$\overline{E1}$	$\overline{E2}$	A3	A2	A1	A0	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$	$\overline{Y4}$	$\overline{Y5}$	$\overline{Y6}$	$\overline{Y7}$	$\overline{Y8}$	$\overline{Y9}$	$\overline{Y10}$	$\overline{Y11}$	$\overline{Y12}$	$\overline{Y13}$	$\overline{Y14}$	$\overline{Y15}$
L	L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H
L	L	H	L	L	L	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H
L	L	H	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H
L	L	H	H	L	L	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H
L	L	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	L	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H

H = High Voltage Level, L = Low Voltage Level, X = Don't Care

Figure 4 - 9: Truth table for input states of CD74HC154M decoder. From [23].

4.3.4.6 Display Controller Connector

The connection between the multiplexing LED panel and the display controller was made using a 0.5-mm pitch, 30-pin flat flexible cable (FFC). This type of cable, and it's corresponding connector, was chosen as its size was small enough to be placed on the backside of the panel where it not impede the placement of the LED's unlike other, larger connectors such as standard through-hole or surface mount 100-mil pitch headers. The cable also is extremely small and flexible. One downside of this choice, though, is the relatively high cost compared to standard headers. Comparisons of these connectors can be seen in Figure 4.10. The connector had a pin-out that was chosen to be fully functional independent of the type of panel attached and could be used in future revisions of the system without requiring changes to the shield adapter for the display controller.

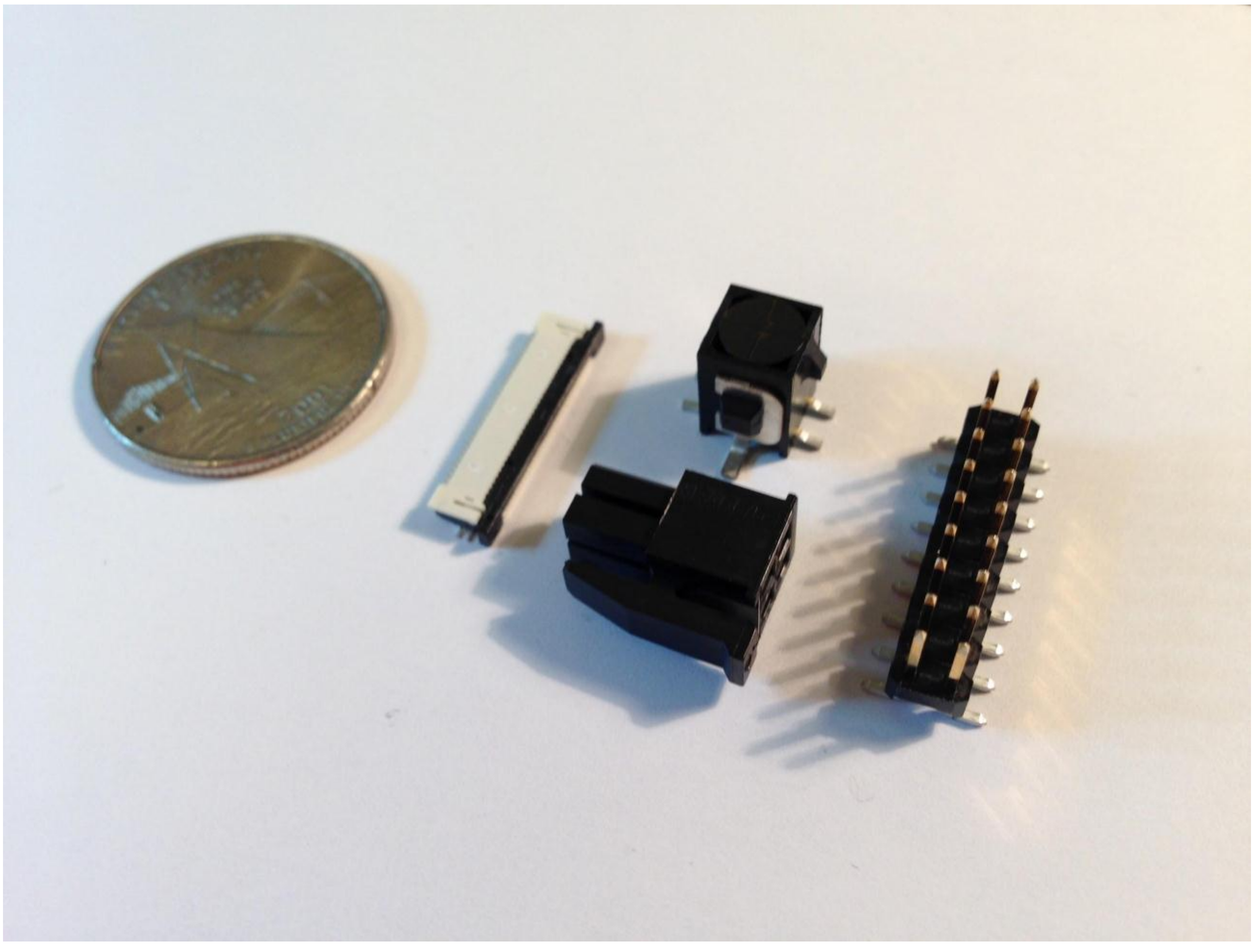


Figure 4 - 10: Reference scale for SMT connectors selected for the final design.

Figure 4-10, shown above, depicts the size differences between the many connectors that were used on the multiplexing LED panel, with a quarter for reference. The connector closest to the quarter is the 30-pin FFC connector, which handles the primary data interface between the display controller and the LED panel. It also provides the limited power that the display controller requires. The top-middle connector is a 4-pin Molex power connector, which is used to connect the main 5V power source to the panel. Beneath it is the mating 4-pin connector which is inserted into the Molex connector. On the right is a standard, 2x9 0.100" pitch header. As can be seen, the headers volume is much larger than the FFC connectors, and the actual board area required by the pads on the PCB is large enough to cause potential issues with a denser design, as was found with the direct-drive LED panel.

4.3.4.7 Power Regulation

To regulate the single input power rail down to 3.3V for the display controller logic circuits and the LED drivers, the LT1963-3.3 from Linear Technology was used. It provides a fixed, ultra-reliable 3.3V rail with low noise, low dropout and a fast response to load current spikes. The primary reason for this choice of regulator is due to prior positive experiences with it by the team members - as the power regulation can be a critical point in any system, the price premium for an LT part was considered worth it to reduce the total development risk to the key features of the LED panel from something as trivial as a power supply not working.

4.3.5 Multiplexing Panel Power Consumption

An analysis was performed to determine how much power was expected to be drawn by each source, and from what supply to determine the power source requirements. Table 4-2 contains the summarized results.

Table 4 - 2: Multiplexing Panel Power Consumption by Component

Component	Power Source	Current Consumption	Total Power Dissipated	Notes
LT1963-3.3 Regulator	5V Main Rail	1 mA	3.3 mW + Load	Quiescent
Tri-Color LED (Red, Green, Blue)	5V Main Rail	58.6 mA (19.5 mA + 19.5 mA + 19.5 mA), 936 mA	176 mW, 2.17 W	Instantaneous Per Individual LED package (RGB) max, (X16 per row)
TLC5940 LED Driver (Digital Interface and Bias)	3.3V Rail (LDO)	≈25 mA max	80 mW	Does not consider power due to LED current driver contribution
TLC5940 LED Driver (LED's)	5V Main Rail	Up to 312 mA each	Up to 620 mW to 1 W max	Maximum load implies all white output
CD74HC Decoder	5V Main Rail	≈100 uA max, plus MOSFET gate load	Variable	Power is nearly linear with scan-speed
SI2301 MOSFET	5V Main Rail	Up to 936 mA	Up to 114 mW at 6% duty cycle	Depends on total LED current

4.4 Direct Drive Panel Architecture

4.4.1 Direct-Drive Panel Overview

The direct-drive LED panel was one concept that came to mind almost immediately when brainstorming designs for the LED panel. The primary motivation in its design is to reduce the impact of the low effective LED current (see Eq. (X.Y) for more information) that the multiplexing architecture suffers from, and the correspondingly limited brightness. Essentially, it decreases the number of scan-lines for the display from 16 down to 1, by attaching a *unique LED driver output* to each and every individual LED on the panel (all 256 packages, or 768 LED's total). As a result, the number of drivers on the panel has increased from 3 to 32 (with a different model). With the increase in LED drivers comes a massive increase in power consumption and maximum brightness that is possible with the display. The engineering design complexity also increased significantly, requiring nearly two terms (four *months*) to design and create the direct-drive LED panel, as opposed to a single *week* to design and create the multiplexing LED panel. Due to thermal issues with the design, the LED drivers had to be mounted on a separate PCB and attached with headers to provide better thermal performance. The system is designed to handle a single 5V rail input, similar to the multiplexing panel, but can draw up to 16 amps of current. Figure 4.11 below shows a partially-assembled prototype, with a single daughter-board lacking heat sinks attached.

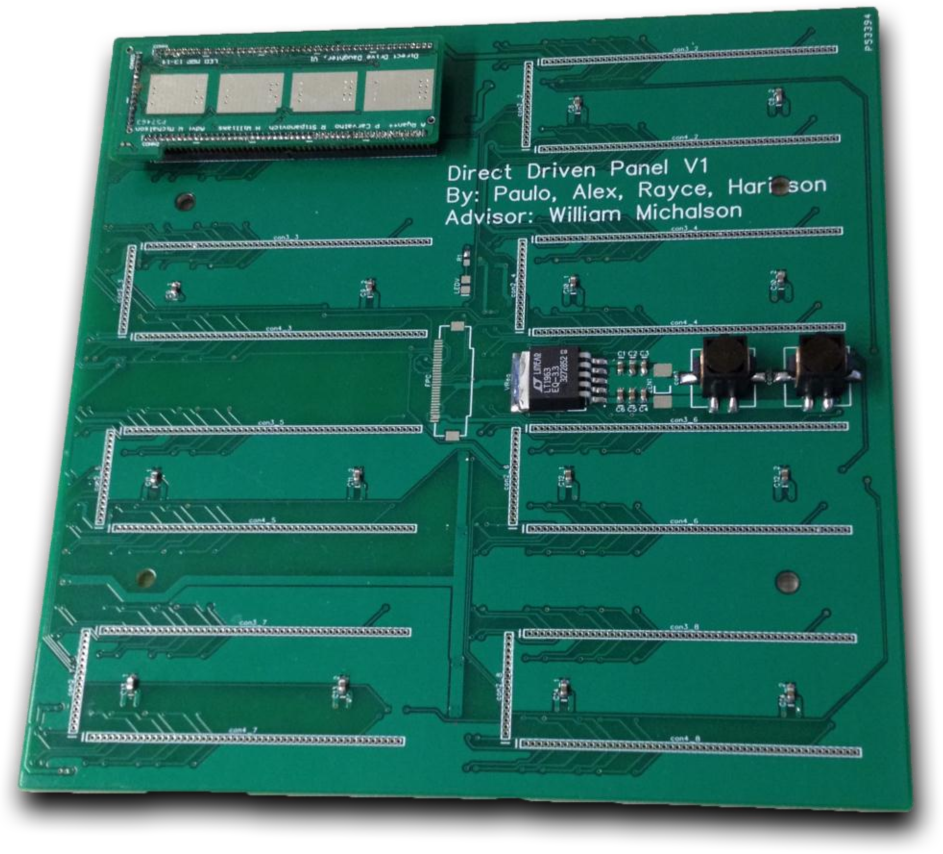


Figure 4 - 11: Final implementation of the direct-drive board design.

4.4.2 Direct-Drive Panel Functional Block Diagram

Figures X.Y and X.Z detail the functional components of the direct-drive LED panel. It is actually constructed of two separate systems due to technical engineering issues discovered during the design, encompassing nine (9) separate PCBs. In Figure 4.12, the main direct-drive LED panel functional diagram is shown -- it consists of the display controller-panel data interface connector, the power input connectors and regulation circuitry, 4x8 LED rectangular blocks, and the LED driving systems on eight separate PCB's, which are documented in Figure 4.13. Each LED driving system consists of the LED and data connectors, the actual LED driving system, the heat sinking system and a clocking system.

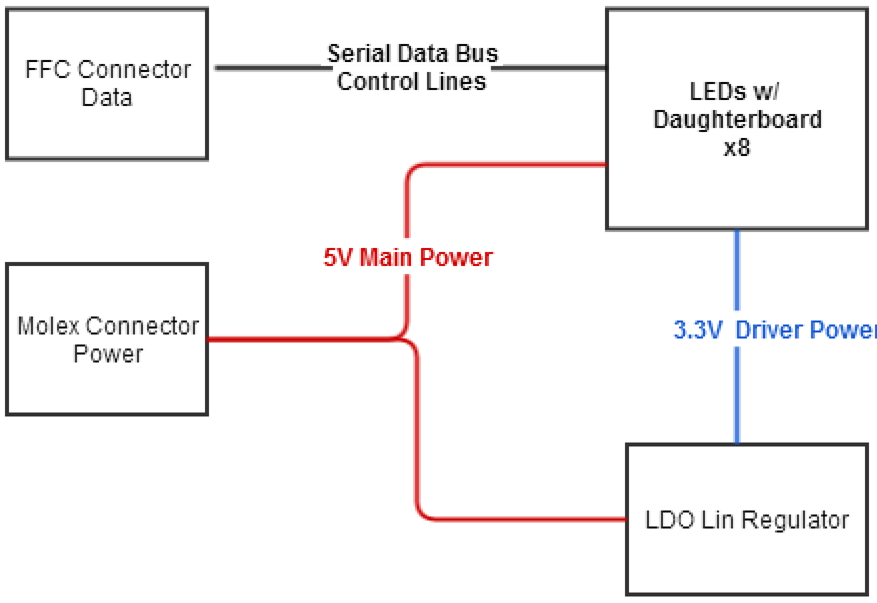


Figure 4 - 12: Direct-Drive LED Panel Functional Block Diagram

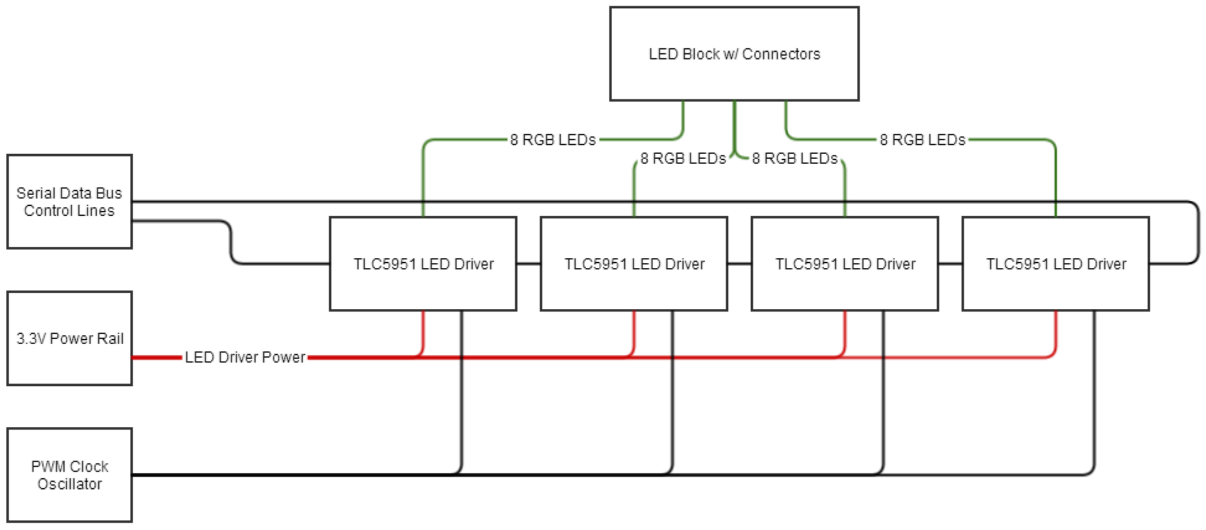


Figure 4 - 13: Direct-Drive Daughter Board Functional Block Diagram

4.4.3 Direct-Drive Panel Theory of Operation

The direct-drive LED panel design operates by directly connecting each and every LED to its own unique driver. There is no switch matrix in the design, and as such there are no row or column distinctions, or scanning of the display. Thirty-two separate 24-

channel LED drivers with programmable current-sink outputs are used to individually control each LED with no dependence on neighboring LEDs. A serial interface is used to program each new video frame to the panel, while real-time control pins refresh the LED display drivers at the appropriate rate to make them work. Because of the lack of scan lines, a new video frame only is loaded at approximately 60-Hz maximum because any higher refresh rate is wasted. All circuitry relating to the scanning operation has been removed as a result, including the power MOSFETs and the decoder.

To actually generate an image, the current-sink outputs of the drivers are switched on and off with a pulse-width modulation (PWM) control scheme - the duty cycle is what is programmed via the serial interface. It is very similar to the interface described in section 4.3.3. The blanking period does occur approximately every 410 microseconds to refresh the internal PWM counter, but due to the large number of drivers and the correspondingly large time it takes to upload new image data, coupled with the lack of need to constantly upload new row data, the video image is only latched into the internal driver PWM counters every 60-Hz maximum, not every blanking cycle. An example timing diagram of this is shown in figure 4.14.

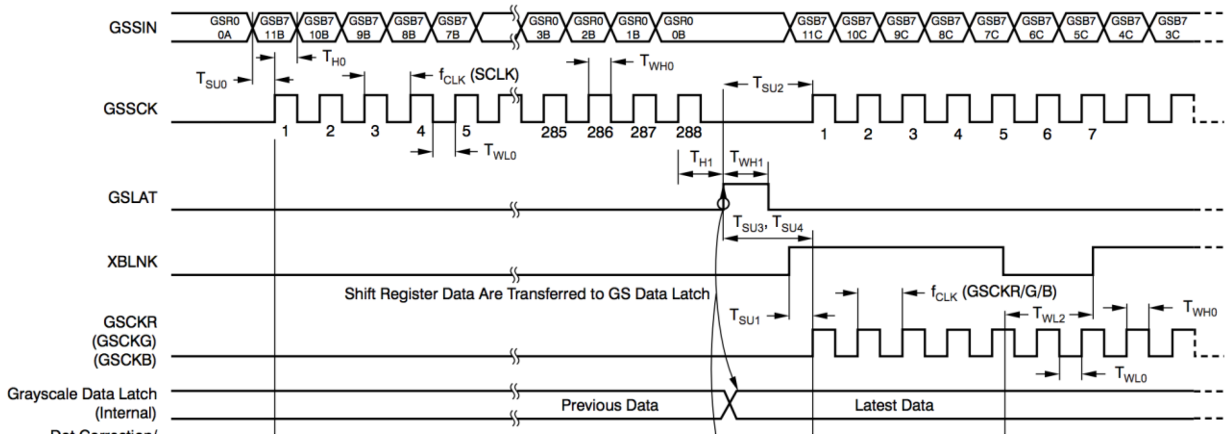


Figure 4 - 14: Timing diagram of counter, blanking and serial operation from [22].

4.4.4 Direct-Drive Panel Component Selection and Description

<u>LED Drivers</u>

For the Direct-Drive panel, the team opted to use the TLC5951 24-channel LED Driver from Texas Instruments. Similar to the TLC5940 LED Driver found on the multiplexing LED panel, it has a simple, cascadable serial interface combined with several real-time control pins. Twenty-four programmable constant-current sink output pins, divided into three banks of eight pins each, are connected to eight tri-color LEDs. Its main advantage over the TLC5940 is that the price-per-output channel is lower, $0.075 per channel at 10,000 units for the TLC5940 vs. $0.064 per channel at 10,000 units for the TLC5951 [22]. The price savings of the TLC5951 is highly advantageous as there are 768 individual LED channels required for the direct-drive architecture, and so more expensive channels quickly add up to increase the total cost. In addition, its interface provides near-complete compatibility with the TLC5940, allowing the display controller to work with either panel design with only a software change, and no hardware changes.

The TLC5951 drivers are split into eight banks of four drivers, as shown in the Figures 4.12, and 4.13. The serial interface is chained between each driver in the bank on the daughter boards, and between each daughterboard on the LED panel itself. All eight daughter boards (of 32 drivers) form a complete chain which can connect back to the display controller via the serial interface for fault monitoring, either from over-temperature warnings or the LED failure detection built-in to the TLC5951.

PWM Clock Generation

Due to the large of number of drivers on-board the panel, it was opted to use a single CMOS oscillator located on each daughter-board to drive the four local drivers. This prevented the issues of having to route a high-speed clock signal with minimal ground plane between daughter-boards, simplifying signal integrity problems, electromagnetic interference (EMI) noise and routing difficulties.

A 10-MHz, HCMOS 3.3V Oscillator from CTS-Frequency-Controls, the same manufacturer of the oscillator on the multiplexing board. The lower frequency version is identical, in that it provides similar noise and stability and accuracy performance as the 24-MHz version, but the lower frequency cuts down on the blanking period frequency required, reducing EMI and current spikes. The higher frequency is also not needed due to the architecture, as the absolute minimum frequency to maintain the full 12-bit resolution is only 245-kHz, or 12-bits worth of PWM counting at a 60-Hz image refresh rate.

Voltage Regulator

The LT1963-3.3 3.3V linear voltage regulator from Linear Technology Corporation was chosen to be used to generate the local 3.3V rail for each of the LED drivers. This is the same part as used for the 3.3V rail on the multiplexing panel, and the reasons for choosing it are similar - it is extremely reliable, can handle the large amount of current required by the total of the 32 LED drivers, plus the display controller, and primarily, to not have to purchase additional part models that are different from the multiplexing panel. This allows the team to cut down on excess stock of spare parts for both panels while the prototypes were being constructed, saving scarce prototyping money. The package for the regulator, a 5-lead D2PAK, offered superior thermal performance in removing heat from the regulator as well and transferring it to the PCB.

Power and Signal Connectors

The power connectors chosen to be used are the same Molex-style 4-pin connectors as used on the multiplexing LED panel - this version, though, uses two side-by-side due to the large current demands. A picture of these connectors can be seen in Figure 4.10. The primary connection to the display controller is also the same, a 30-pin 0.5-mm pitch FFC connector, chosen for compatibility reasons with the first shield board design, negating the need for a re-spin with a different connector.

To attach each and every daughter-board to the LED panel, 1-mm pitch Samtec header pins were chosen due to their ultra-high density and natural mounting abilities. Female-variants are mounted on the LED panel itself, while the male counterparts are attached to the daughter boards - the pins natural resistance to insertion and removal are relied on to keep each daughter-board in place. Two 50-pin headers are on the long-side of the daughter board, as can been seen in Figure 4.15, while a single 16-pin header is perpendicular to both on the “top” of the board. The 50-pin headers carry the LED signal current and attach to the LED drivers, as well as the 3.3V rail for the drivers on two pins per header. The 16-pin header contains several ground connections, as it is the ground return path for each LED on the daughter-board, and the serial and real-time control pin data lines.

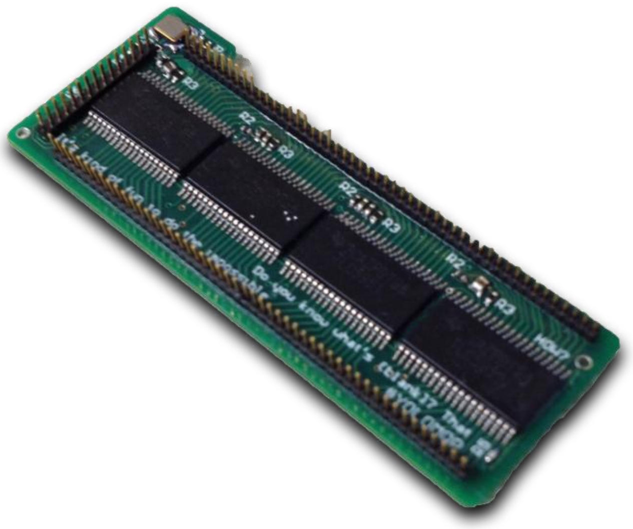


Figure 4 - 15: Picture of a finished daughter board. Note the size of the LED drivers and headers.

4.4.5 Direct-Drive Panel Thermal Design Problem and Solution

The original design concept for the direct-drive panel was to have each of the LED drivers attached to the backside of the direct-drive LED panel, with no daughter-boards involved. One critical issue that arose during the design-phase of the direct-drive panel is that the amount of power consumed by the panel, and by extension the heat produced, is enormous. As the design originally had no external heat sinking or airflow, this was found to be a problem, and initial analysis found there to be no simple solution. The results of the analysis gave the design that is the current state of the direct-drive panel, one with eight separate daughter-boards with back-plane heat sinking.

The problem discovered is that, worst-case with a 6-volt supply voltage and minimal voltage drop across the LEDs, each LED driver could potentially be dissipating up to 1.6 watts of power - while normally this isn't considered a lot of power, the fact that there would have been 32 drivers in extremely close proximity, with another 40 watts of power dissipated total on the front-side due to the LEDs with minimal air-cooling or heat sinking available is a cause for concern. This, coupled with the requirement in the original proposal to have the panels operate in an environment up to 70 degrees Celsius ambient, resulted in the expectation that the LED drivers would vastly exceed their rated junction temperature, and go into thermal shutdown protection. Figure 4.16 shows a work-in-progress of the original, proposed copper layout of the first revision of the direct-drive panel that was not finished due to the aforementioned issue.

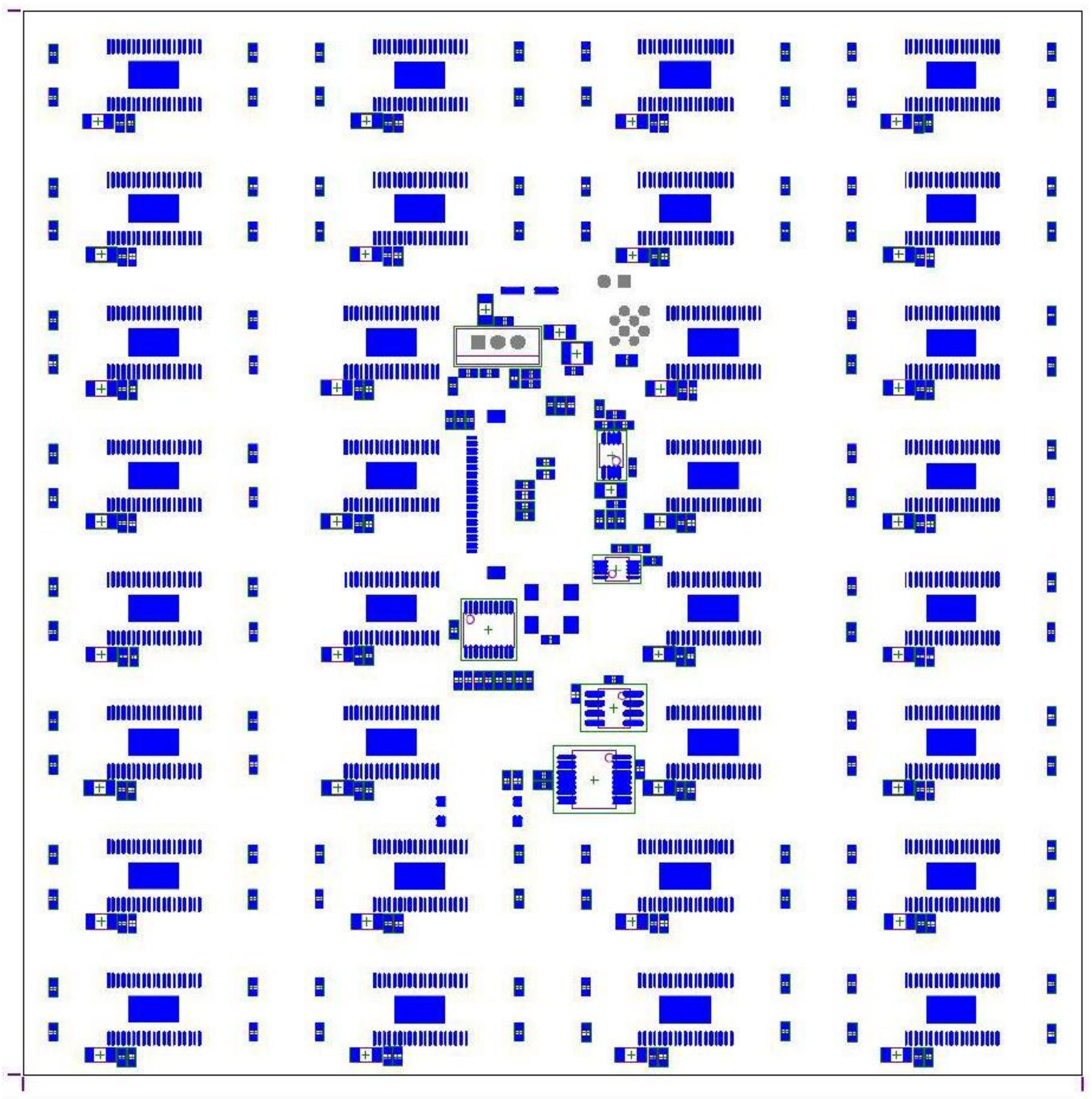


Figure 4 - 16: Proposed Bottom Copper Layout and Placement for rejected direct-drive LED panel

The maximum rated operating junction temperature of the TLC5951 is 150 degrees Celsius. As the design goal was to operate in an environment up to 70 degrees Celsius, it was determined that a thermal resistance from junction to ambient of 37.5 degrees per watt of power for each LED Driver was desired - this provided some safety margin during operation and would not be too difficult to design for. Although the original proposed design was a 4-layer PCB with dedicated ground and power planes, instead of the 2-layer PCB as it is now, the density of the drivers resulted in an effective PCB heat radiating area of 5.7 cm^2, or 0.9 square inches. The IC package, and TI's design recommendations, expect the majority of the heat sinking to occur with the PCB drawing away all of the heat. Almost no real amount of heat (less than 10% of total) was expected to be drawn away either by convection or radiation from the IC body itself. Due to the extremely low area per driver, though, the effective thermal resistance for each driver was *much* higher at approximately *170 degrees per watt* than the estimated figure of 25 degrees per watt as found in TI's design application note [23] for a similar, ideal configuration.

It was considered a possibility to add external heat sinks to each driver on the top of the package to reduce the thermal resistance. However, due to the small size of the driver, and that the plastic package of the IC already had a significant thermal resistance, the effect of any heat sink would have been marginal. It was calculated that the effect on the thermal resistance would be in the range of approximately 70 degrees per watt after a 10 by 10 millimeter heat sink was added to every driver. This would have enabled the LED panel to operate reasonably well in a room-temperature test setting during development, but would not satisfy the design criteria established. Additional reduction in the resistance would be possible with the addition of forced airflow, but as the original vision also had the LED panel in a near-air-tight case, this would have marginal benefit.

The solution found to the problem was to separate the drivers from the LED panel onto a separate breakout daughter PCB, invert them, and attach them directly to the backside of the LED panel, as described and shown in the overview. This solution works as the thermal resistance of the junction to the heat sink is greatly reduced - all heat flows from the IC junction, to the PCB through the low-resistance thermal-pad, through the thin PCB through a large number of heavy-copper thermal vias, and to the backside heat sink, which can be much larger and effective due to the heavy copper and close arrangement of each driver. IT's estimated that this could provide a thermal-resistance of less than 30 degrees per watt with the heat sink, but that needs to be tested fully. Figure 4.17 shows this current design approach on a built-up daughterboard with the back-side thermal pads for the heat sinks exposed.

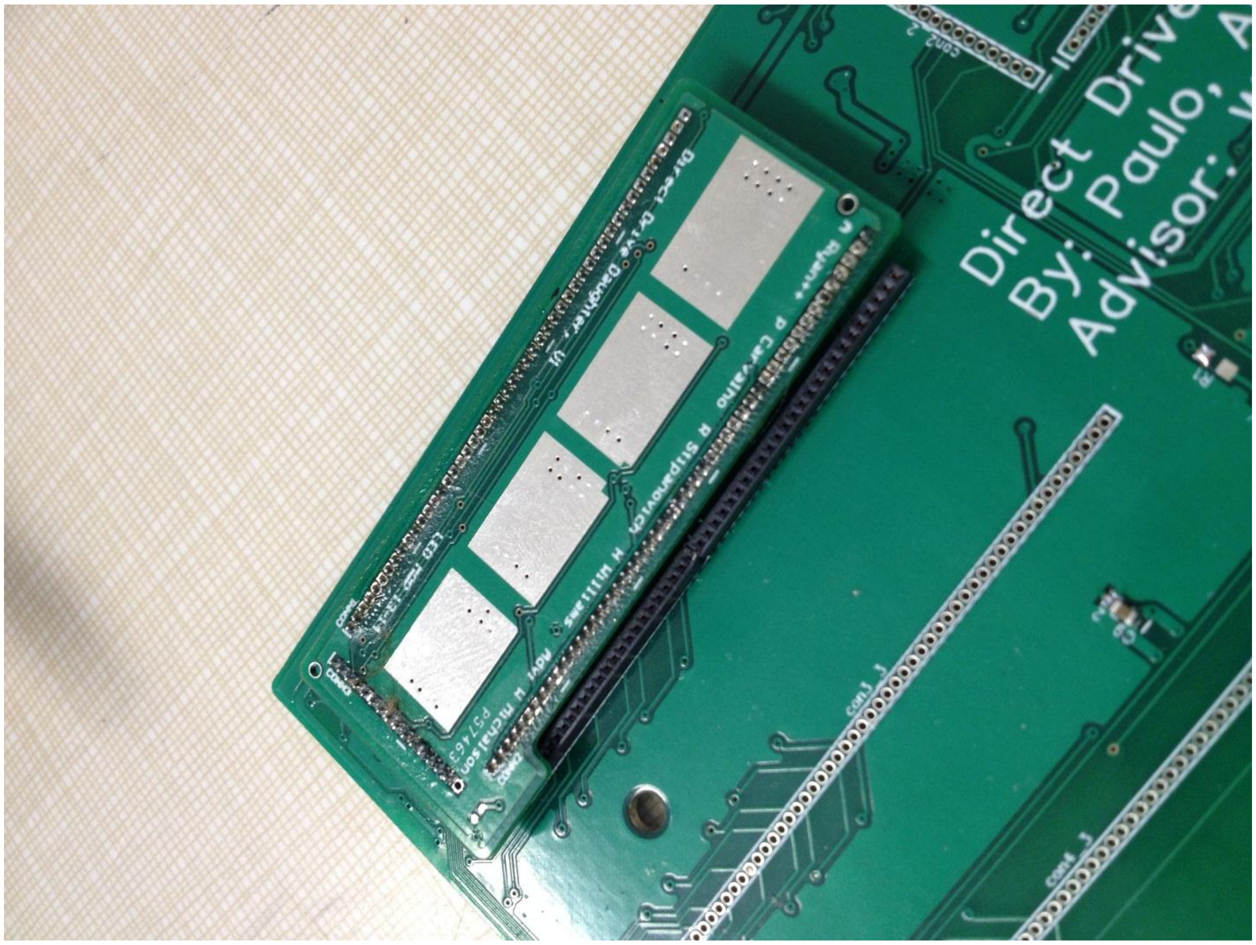


Figure 4 - 17: Image of the exposed copper pads on the opposite side of the LED driver IC's to be used for heat sink attachment.

4.5 LED Panel Embedded Software Design

This section details the design and operation of the embedded software created to run the STM32F4 microcontroller, which is the core part of the LED panel display control system. It handles the Ethernet communications stack, the processing and signals to drive the LED display, auxiliary LED panel communications interface, and system monitoring.

4.5.1 Embedded Software Overview

The embedded software was written using IAR Embedded Workbench for ARM, an Integrated Development Environment from IAR Systems. A starting point for the software was taken from provided examples for the combined STM32F4 development board and the complementary “shield”, which provided the Ethernet PHY and connector. The example software simplified development effort required to get the Ethernet connection set up and running, and allowed the team to focus its efforts on working on the networking and video display software.

As the tasks that the display controller would be required to handle are very complex (including a real-time combination of video display updating, inter-panel auxiliary port communications and Ethernet UDP networking), it was decided that a real-time operating system (RTOS) was to be used. Past experience and the compatibility with the example software led to the choice of selecting FreeRTOS, an open-source RTOS that has multiple ports and examples available for it, and so the kernel has been extensively tested and support. FreeRTOS features nearly all standard RTOS components, including task creation, deletion and preemption, mailboxes, mutexes and semaphores, and software timers. Version 6.1.0 was used for compatibility reasons.

To handle the Ethernet stack and provide a simple network programming API, the LwIP (Lightweight Internet Protocol) stack was integrated into the software. The key benefits of using this software are that it is free and open-source, it is stable, and it provides the common Berkeley Sockets API that is a standard for networking on UNIX-based computer systems. As such, numerous references existed to be consulted and used without issue due to the compatibility. Version 1.3.2 of the stack was utilized, again for compatibility reasons with other pieces of software.

4.5.2 Embedded Software Operating System Development

Development of the embedded software with the RTOS was realized by separating all major functions into either separate *tasks* or *interrupt routines (ISRs)*. A task is a major block of code with a priority handled whose operation is handled by the RTOS software. The RTOS determines which blocks are required to be run at certain points in time, and executes the block (as a lesson to the other blocks). It supports prioritization, so certain blocks can be interrupted if some other software loop is determined by the RTOS to require more immediate attention. An ISR is a block of code whose execution is controlled by the hardware itself, above the RTOS’ control. The result of this is that the execution time of the ISR is nominally deterministic and repeatable, as opposed to tasks which may be preempted and switched out of execution by the RTOS at any given time. The software was decided to be written as either a task or ISR depending on how reliable its operation was required to be.

4.5.3 Embedded Software Operation Description

On boot-up, the RTOS is disabled and standard initialization functions are executed that set up the STM32F4 microcontrollers' pins and internal components for operation of the display. This is followed by the initialization of the LwIP and FreeRTOS system components for use, as well as the task generation setup sequence, and ISR setup. Once the setup is complete, the FreeRTOS scheduler starts. The Idle Task is started immediately, along with the DHCP Task. The DHCP Task primarily has control of the system, while the MCU waits to establish a connection with the network switch and DHCP server and obtain an IP address. Once this is done, the DHCP Task spawns the UDP Reception Task, the UDP Transmission Task, the Interpanel Communications Task, the Panel Update Task, and the Alive Task. The DHCP Task then kills itself once DHCP has finished. The UDP Reception Task takes priority control, listening for broadcast transmissions from the host PC asking which panels are currently connected on the network while it establishes the LED display.

After commanding the UDP Transmission Task to respond with an, "I'm Alive", packet after the host PC broadcast query, the panel waits for a response of one of two packets from the host PC, in order to set up the displays coordinate system. One command from the PC is to act as the LED display origin point, and start performing spanning to establish what the physical hookup of the display is, through the auxiliary UART ports and the Interpanel Communications Task. The other possible command is to act as a spanning slave, and wait for a query response from neighboring panels over the UART interface. The origin panel will query all surrounding panels, asking them their IP address and assigning them a coordinate point, and communicates this information to the PC host. It will then move to nominal operation. After a non-origin panel is queried, and it responds with the requested information, the panel joins the querying process, looking for non-queried neighbor panels, after which it too goes into nominal operation. This process continues for all panels until the display is complete.

The LED Panel is effectively operating nominally at this point, with the self-configuring display setup and displaying received video frames, while also time-syncing with each neighboring panel using the UART ports.

4.5.4 Embedded Software Task and ISR Details

4.5.4.1 Idle Task

The Idle Task is entirely optional to run - it handles measurement functions of the system CPU utilization and how much RAM each task has consumed from its stack, and reports the data over a spare UART port available on the microcontroller. Disabling the task would have no effect on the system.

4.5.4.2 DHCP Task

The DHCP Task's operation is the initial focus of the system after the scheduler starts, and auto-kills itself upon completing. It works to establish a DHCP lease for the microcontroller display controller on the Ethernet network. It will loop its control code infinitely until the DHCP lease is established. After establishing the connection, it will spawn and start the rest of the system control tasks that are dependent on having a working IP address, followed by its death.

4.5.4.3 UDP Reception Task

The UDP Reception Task handles the parsing of incoming Ethernet packets, and sends messages to other tasks if it determines action needs to be taken. The task expects all messages to conform to the UDP communication specification developed, documented in section 4.5. Non-conforming messages are discarded. The primary loop checks the LwIP Ethernet frame buffer to see if there is data available. If there is data, it parses the data and possibly posts a message to another tasks mailbox. Otherwise, the task sleeps for a short while and repeats this process.

4.5.4.4 UDP Transmission Task

The UDP Transmission Task acts as a control mechanism for all outgoing UDP messages to the PC host, conforming to the UDP communications specification. It queries its own mailbox continuously, checking for any posted messages from other tasks, otherwise sleeping. After receiving a message and determining that a UDP packet must be sent, it makes a function call to the LwIP stack to transmit correct data.

4.5.4.5 Interpanel Communications Task

The Interpanel Communications Task manages the transmission and reception of data on the auxiliary UART communication ports, which establish a direct link to neighboring LED panels in the entire LED display. Transmissions in the communication protocol are delineated by each bytes most-significant bit; bytes with a set most-significant bit are used to indicate control messages, such as indicators of start-of-frames or end-of-frames, or the type of message being sent. Bytes with a reset most-significant bit are used to indicate data messages - these contain the actual content of the message.

During setup, this task manages the coordinate setting and discovery of neighboring panels. It sends out and receives messages asking neighbors if they have a valid coordinate point yet, and possibly assigning one if the neighbor lacks one.

During operation, the responsibility of this task is to check neighbors for operational faults by querying their status. Any detected faults, or lack of a response, is considered an error, and will be reported to the host PC via the UDP Transmission Task. It also handles the frame update sync, by which the task synchronizes the moment when the video array will update the displayed image to the next one stored in the buffer. Although the UART line does have some delay between the start of a transmission and the finish when the receiving UART decodes the information, the delay is predictable and has the ability to be compensated for. Sending a time-compensated special message is used to indicate the exact time the panel wishes to update to the next image. With enough compensated messages spanning through the network, the entire display synchronizes in several displayed images.

4.5.4.6 Alive Task

The Alive Task's purpose and operation is extremely simple. It is used to ensure that the microcontroller software has not undergone any soft- or hard-faults, and that it is still operational. It operates by toggling a status LED on the backside of the display at roughly a 2-Hz rate, otherwise sleeping due to the RTOS.

4.5.4.7 Panel Refresh ISR

The Panel Refresh ISR is automatically called by a hardware timer inside the microcontroller to update the LED display and perform display blanking at the appropriate intervals, either 2.44-kHz for the direct-drive panel or 5.86-kHz for the multiplexing panel. Each version of the ISR for each panel architecture starts by blanking the display to refresh the PWM counter. This is followed by potentially latching in new serial data, either every blanking period for the multiplexing architecture, or at a 60-Hz rate for the direct-drive architecture. If serial data is latched in either version, a new image stream is started streaming immediately before the blanking period ends. The multiplexing version also updates the selected row via the decoder control pins.

4.5.4.8 UART Communication ISRs

The UART Communication ISRs handles the placement of received UART bytes from the hardware UART module into the mailboxes checked by the Interpanel Communications Task. A different ISR exists for each UART port on the shield. The ISR is triggered by the hardware UART module

4.6 Video Processing Software Design

4.6.1 Overview

Qt was selected to develop the video processing and panel addressing host PC application as it offered significant speed improvements by enabling the team to quickly and effectively develop a complicated application using C++. This also provided the added benefit of being able to interface natively with frameworks such as OpenCV and the NVIDIA CUDA API set to enable faster video processing. Qt also provides a robust network framework as well as a flexible threading system that enabled the application to run in multiple threads, each allocated to a specific task. This ultimately improved performance and reduced the overall latency time from input source to output on the panels. A final additional benefit of designing the application with Qt was the ability to develop one set of source code that could be compiled for any operating system or architecture.

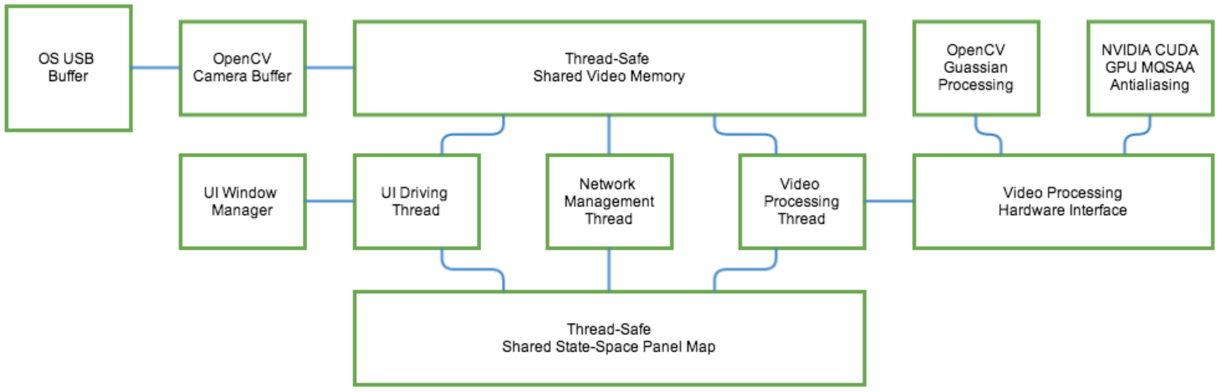

OS USB Buffer
OpenCV Camera Buffer
Thread-Safe Shared Video Memory
OpenCV Guassian Processing
NVIDIA CUDA GPU MQSAA Antialiasing
UI Window Manager
UI Driving Thread
Network Management Thread
Video Processing Thread
Video Processing Hardware Interface
Thread-Safe Shared State-Space Panel Map

Figure 4 - 18: PC Software Functional Diagram

Above in Figure 4.18 is a functional diagram of the software. The primary tasks of the software are divided into three separate threads. The UI Manager is responsible for updating the UI to reflect the current state of the array as well as handling any user input that would affect video output. The advantage of threading UI updating is that is prevents application lockups when scheduling demands become tight. In addition to this, there is a Network Management Thread. This is responsible for handling the self-addressing algorithm as well as detecting panel faults and notifying the UI thread of any changes. Finally, the video processing thread, which is set to a higher priority than the other two, is responsible for buffering a video stream, processing it, and outputting that data to the panels via its own network interface. As this ultimately is the core functionality of the panels, it effectively possesses its own set of resources, but still polls from a global state-space representation of the panel array to gather information on where to route video packets.

4.6.2 Pixel Sampling Theory

In most cases, the pixel count of an LED video wall does not match that of the content to be displayed. Because of this, down sampling algorithms must be used to translate the video content to a smaller pixel count. Issues arise when the input material has noisy content. If a single pixel is chosen to represent an area of video content, any noise occurring at that specific pixel location is directly translated. This ultimately generates a sporadic output image unsuitable for viewing in some cases. A better approach is to average the area around the pixel to be mapped, sampling from a region. This greatly removes noise in a signal and assigns the output pixel a color value more representative of the video content at hand.

Down sampling an image can lead to aliasing, which can itself generate many visually upsetting artifacts in an image. A method of resolving this issue is to blur an image and then down sample it, thus removing the high-frequency content that is the primary cause of aliasing. One of the most effective blurring techniques is a Gaussian Blur filter, which consists of convolving an input image with the Gaussian function:

$$f(x) = a \exp\left(-\frac{(x-b)^2}{2c^2}\right) + d \tag{4-3}$$

When convolved with an image, it generates an analog optical-modeling blur filter that improves visual clarity when resizing an image. This effectively assigns each pixel value a weighted average of its neighbors, making this filter ideal for creating a smaller image that is representative of a much larger one.

4.6.3 Video Processing Overview

As the panels will most likely not be able to reproduce the video content with a one-to-one pixel ratio, image resampling must occur. Alone, this is relatively simple. However, simple image resampling does not produce an accurate representation of the image, as image noise would lead to inaccurate sampling. Anti-Aliasing was required to be performed on the source material to ensure an accurate sample was taken. This was initially achieved in Java with Kernel-based Gaussian blurs. This then was greatly be improved in OpenCV using a low pass filter on a frequency transform of the image. This was experimentally improved further still by offloading the anti-aliasing to a GPU which can offer fast standards such as 16x MQSAA through the use of the NVIDIA CUDA architecture. However, in the final implementation of the software, a combination of Gaussian blur algorithms and Fourier domain low pass filtering were used. These were both implemented using tool chains provide through OpenCV. On larger images, a noise-reducing filter was used, as its compute time was significantly smaller than the traditional kernel-based Gaussian convolution performed on images with lower pixel-counts. The Gaussian convolution was used at lower resolutions, because it provided a more visually-pleasing output to the panels due to a larger amount of averaging required to accurately represent the video content. The Gaussian filter also more accurately replicates the properties of optical methods for image manipulation, and because of such, generally produces a more natural-looking blur result. This ultimately translates to a more subjectively accurate representation of the down sampled video content. Additional trivial processing was done to convert between formats of Qt and OpenCV. The flowchart below describes the process of importing and processing video frames to the panels.

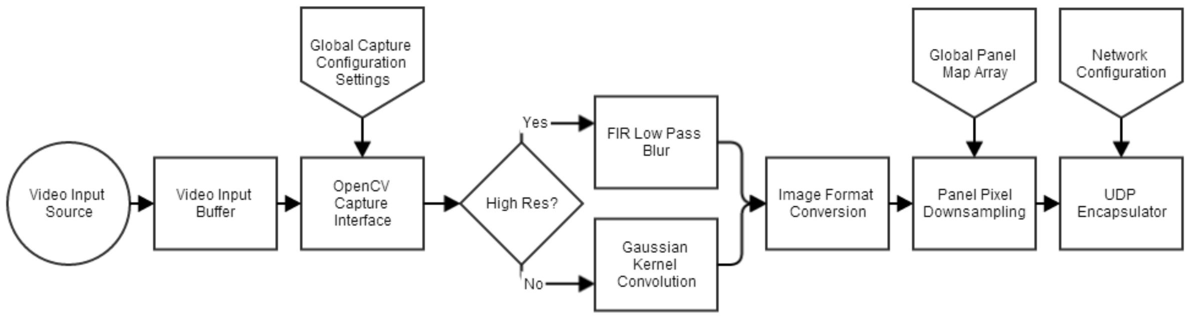
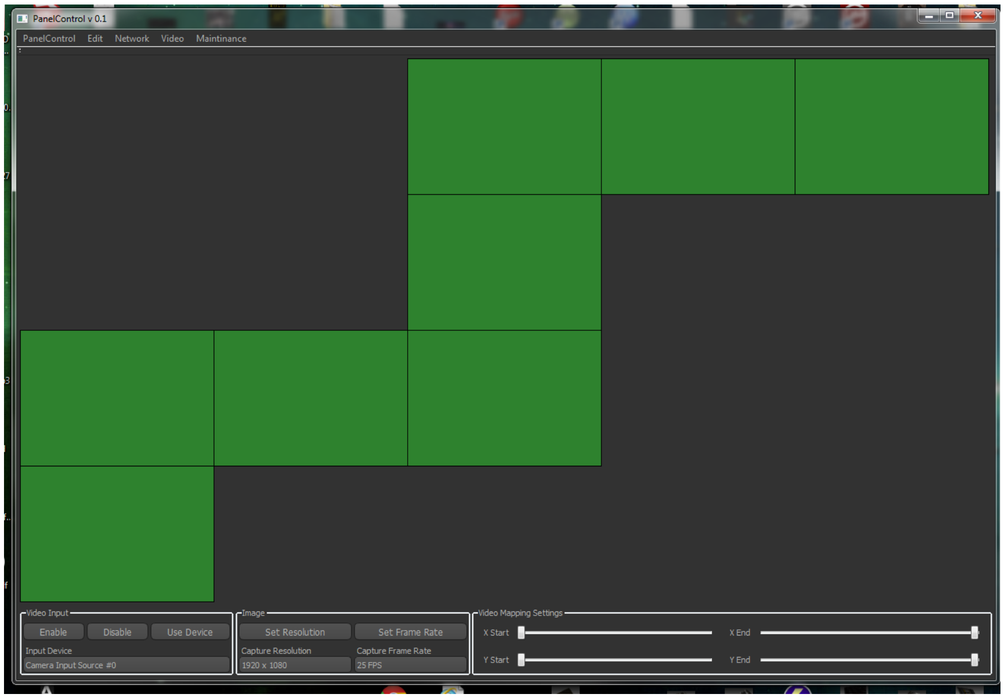


Figure 4 - 19: Flow diagram for a single video frame.

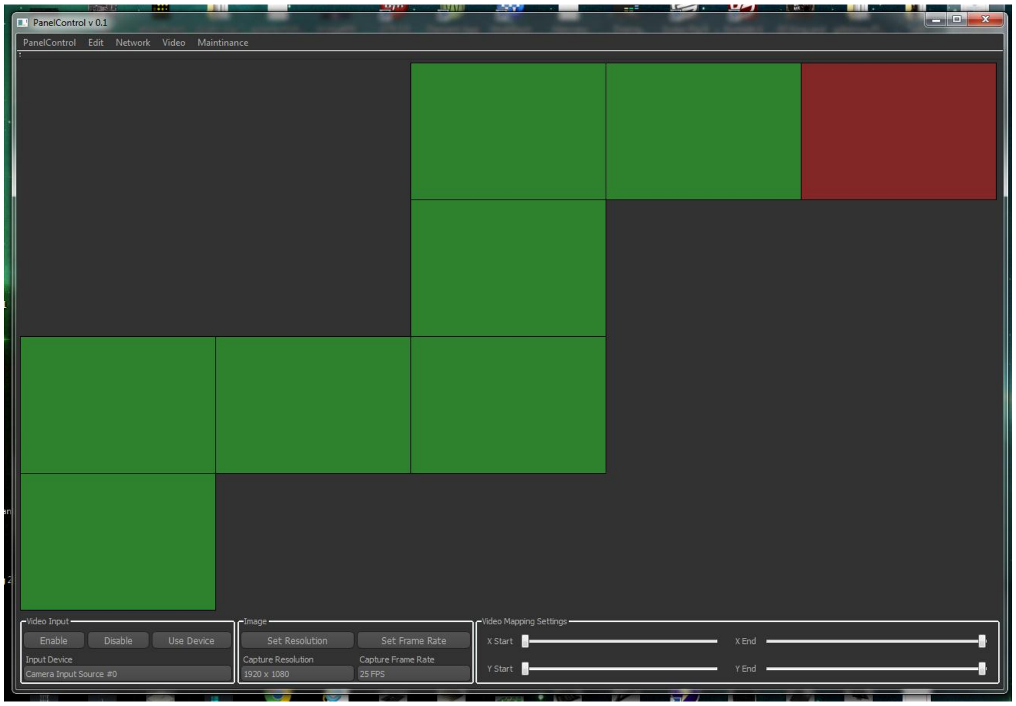
4.6.4 User Interface Design

As Qt provided a method for rapid UI development, a basic, multi-windowed user interface was developed to provide an intuitive method for video content mapping and array status verification in a real-time context. The core of the UI consists of a video displaying the video content being processed. Below are several setting for configuring basic video settings such as resolution and the specific input device to process video from. Overlaid on top of the video content is a contextual representation of the physical dimensions and configuration of the panel array. This overlay updates to reflect the position of panels, as well as their general status such as fault or timeout indication. This interface enables the user to easily determine what content is being mapped to what panel. The UI also offers the ability to scale and crop the video content to the array, although the default behavior scales the content to fit the entirety of the array. In this usage case, if a column or row of panels on the periphery of the array fails, the video content dynamically rescales to ignore the fault. On large scale video walls, this change would be unperceivable to the audience from far distances, but would allow a video director to still display all of the required content until a repair occurred. Below, in figures 4.20 and 4.21, are two representations of mapped arrays - one showing a healthy array, and one showing the location of a panel indicating a fault.



Input Device: Camera Input Source #0
Capture Resolution: 1920 x 1080
Capture Frame Rate: 25 FPS

Figure 4 - 20: Healthy array representation over video UI.



Input Device: Camera Input Source #0
Capture Resolution: 1920 x 1080
Capture Frame Rate: 25 FPS

Figure 4 - 21: UI indicating that a fault is occurring at a specific panel.

4.6.5 Self-Addressing Protocol Implementation

The self-addressing protocol was implemented through a combination of packet-based UDP traffic over the panel's 100 Base-TX Ethernet PHYs and the proprietary optically-isolated UART links. The two peripheral communication schemes were separated in functionality. The Ethernet port was strictly used for panel-host communication. This was primarily used to receive video data from the Host PC application, but also received and sent intermittent control signals described in a proprietary protocol formed for the project. It should be noted that the theoretical maximum array dimensions are 60x60 32x32 pixel panels. This would adequately cover the range of a full 1080p signal with a one-to-one pixel mapping. This theoretical hard limit on array dimensions drove the coordinate system to be positioned relative to a virtual center coordinate of 0, with a maximum translational distance from the array center of 60 panels. This is observed in the values chosen for the UDP protocol implementation below.

Table 4 - 3: UDP Protocol Specification Table

Part [bytes]	Size	Description	Expected Values
0..1	2x8-bit	**Source Address** The destination coordinate of the panel (-60 - 60 on each axis) represented in binary. This serves to indicate any address conflicts between the host and the panel. If this mismatches the L3 record of the host, the host can update its local records.	(0x00, 0x00) - (0x79, 0x79) for panels (0xFF, 0xFF) for host PC.
2..3	2x8-bit	**Destination Address** The destination coordinate of the panel (-60 - 60 on each axis) represented in binary. This serves to indicate any address conflicts between the host and the panel. If this mismatches the L3 record of the host, the host can update its local records.	(0x00, 0x00) - (0x79, 0x79) for panels (0xFF, 0xFF) for host PC.
4	8-bit	**Payload Type Enumeration** This indicates what type of data the rest of the payload will contain. Video payloads are 665 bytes long. Control signals and status updates are 1 byte long.	0x00 Reserved 0x01 Video Frame 0x02 Status Update 0x03 Control Signal 0x04 - 0x0FF Reserved

Below are possible payloads defined by the Payload Type Enumerations:

Table 4 - 4: Video Frame Payload Enumerations

5	8-bit	**Image Chunk Indicator** Numerical representation of what 16x16 chunk of the output image this frame represents	0x00-0xFF
6..670	664x 8-bit	**Individual Pixel Data for Chunk** Contains raw pixel data to be translated to video buffer via DMA at chunk address prefix.	(0x00-0xFF R, 0x00-0xFF G, 0x00-0xFF B)

Table 4 - 5: Status Update Payload Enumerations

5	8-bit	**State Change Enumeration** Used to update the controller of the panel's status.	0x00 Reserved 0x01 Idle 0x02 Pend Addr 0x03 Addressing 0x04 Keep Alive 0x05 Neighbor Error Left 0x06 Neighbor Error Top 0x07 Neighbor Error Right 0x08 Neighbor Error Bottom 0x09 Dropped Neighbor (Require re-address) 0x0A LED Error 0x0B Power Error 0x0C Network Error 0x0D Panel Link Error 0x0E - 0xFF Reserved

Table 4 - 6: Control Message Payload Enumerations

5	8-bits	**Control Message Enumeration** This enumeration contains non-video control signals to be sent between panels and the host computer. These serve as both debugging commands and as general control signals for the self-addressing algorithm, output control, and other features.	0x00 Reserved 0x01 Set as (0,0) and begin re-addressing 0x02 Panel Awareness broadcast 0x03 Reset Network Connection 0x04 Suppress Error 0x05 Enable Video Output 0x06 Disable Video Output 0x07 Show Diagnostic on LED 0x08 Identify panel rear LED 0x09 Clear Identification 0x0A - Shutdown 0x0B - 0xFF Reserved

Using a combination of this protocol, a procedural method for array self-addressing was developed. This method uses a combination of the UART and Ethernet connections to acquire information from neighboring panels. Based on the information received, if any, the panel then either addresses itself relative to the other panels. In the presence of an unaddressed array, or if the panel does not detect any neighbors, it will send a query to the host controller to send a broadcast message to begin array readdressing. The addressing protocol is outlined sequentially below. The processes below occur once per initialization of the network. Panels may be added and removed from the network without the need for initialization after the Layer 3 infrastructure is established.

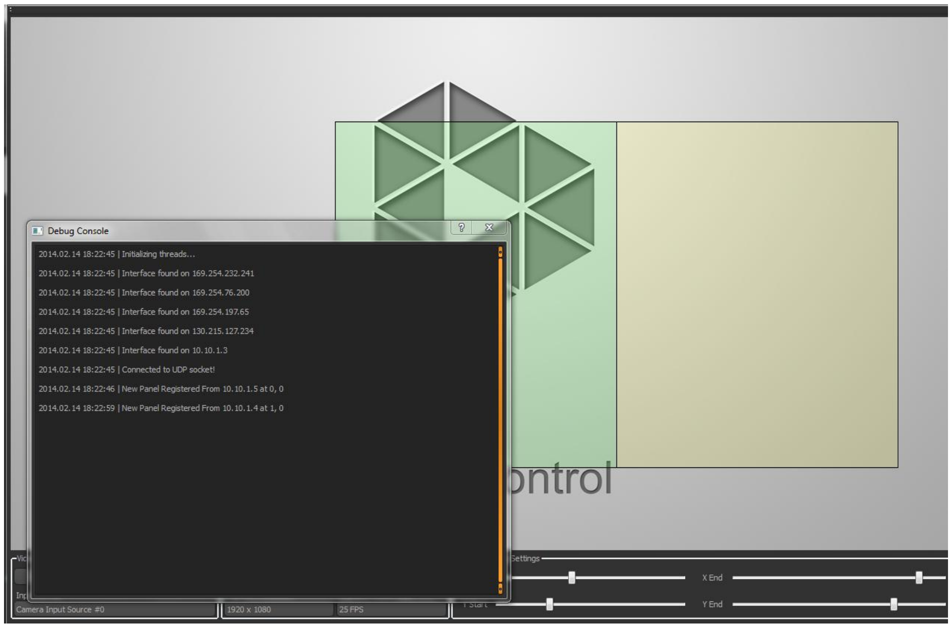
Network Initialization

1. Power is provided to panels and MCU's initialize.
2. Layer 2 switches initialize on panels.
3. TCP/Layer begins Root War to determine Root Bridge with a convergence time of 5 seconds.
4. Root Bridge determined and path optimization begins.
5. L2/L3 networks established.

Panel Self-Addressing Process

1. Host video processor begins by choosing first panel connected to it and designates it as coordinate (0,0). This will be further referred to as the Origin Panel (OP). Note: This may be the same or different than the Root Bridge panel as the L2 RSTP and L7 networks are independent.
2. Host processor sends broadcast packet over UDP to tell panels to accept coordinates.
3. OP sends its local coordinates and update signal over custom PHY to adjacent panels.
4. Adjacent panels receive this signal, update their coordinates by incrementing or decrementing values based on what port original update signal was received on.
5. Adjacent panels then send TCP packet to host video controller with their updated position.
6. Adjacent panels then repeat step 9 with their neighboring panels and process repeats.
7. Process repeats (9-11) until host video controller receives coordinates from all panels (Same number as original client list).
8. Host video controller sends broadcast STOP packet and all panels enter Idle Poll mode.
9. IDLE POLL Mode: Panels poll custom PHY periodically. If they detect a missing neighbor or addition of a neighbor, the panel notifies the host video controller via TCP and the host video controller re-initiates steps 2-8.

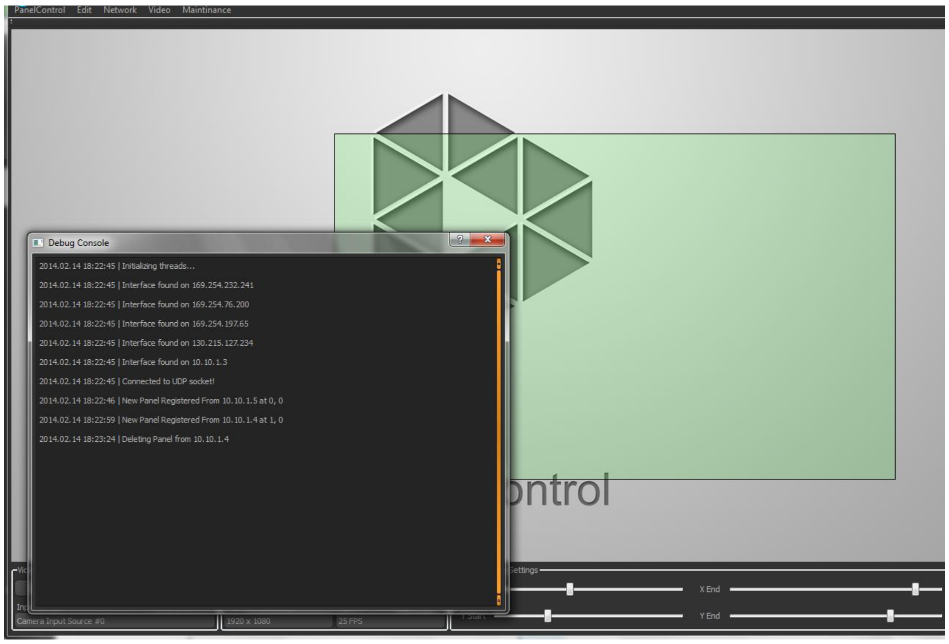
To maintain an accurate record of the array, the host program now initiates a watchdog timer for each panel. If the system requests data from a panel, or the panel has not sent a keep alive packet within a predetermined timeout period, the panel is marked as having network issues. If this persists, the panel is removed from the array. During the period when the panel is in a state of timeout, no video content is sent to the panel. This is done to optimize network allocation and potentially assist in packet delivery from the failing panel. Below, in figure 4.22, is a set of screenshots depicting an example scenario as expressed through the user interface.



```
2014.02.14 18:22:45 | Initializing threads...
2014.02.14 18:22:45 | Interface found on 169.254.232.241
2014.02.14 18:22:45 | Interface found on 169.254.76.200
2014.02.14 18:22:45 | Interface found on 169.254.197.65
2014.02.14 18:22:45 | Interface found on 130.215.127.234
2014.02.14 18:22:45 | Interface found on 10.10.1.3
2014.02.14 18:22:45 | Connected to UDP socket!
2014.02.14 18:22:46 | New Panel Registered From 10.10.1.5 at 0, 0
2014.02.14 18:22:59 | New Panel Registered From 10.10.1.4 at 1, 0
```

Camera Input Source #0 | 1920 x 1080 | 25 FPS

Figure 4 - 22: A display of a panel that is disconnected from the network



```
2014.02.14 18:22:45 | Initializing threads...
2014.02.14 18:22:45 | Interface found on 169.254.232.241
2014.02.14 18:22:45 | Interface found on 169.254.76.200
2014.02.14 18:22:45 | Interface found on 169.254.197.65
2014.02.14 18:22:45 | Interface found on 130.215.127.234
2014.02.14 18:22:45 | Interface found on 10.10.1.3
2014.02.14 18:22:45 | Connected to UDP socket!
2014.02.14 18:22:46 | New Panel Registered From 10.10.1.5 at 0, 0
2014.02.14 18:22:59 | New Panel Registered From 10.10.1.4 at 1, 0
2014.02.14 18:23:24 | Deleting Panel from 10.10.1.4
```

Figure 4 - 23: Example of the PC software removing a panel from a network if no response is received

4.7 Implementation Summary

4.7.1 Physical Design Verification and Manufacturability Analysis

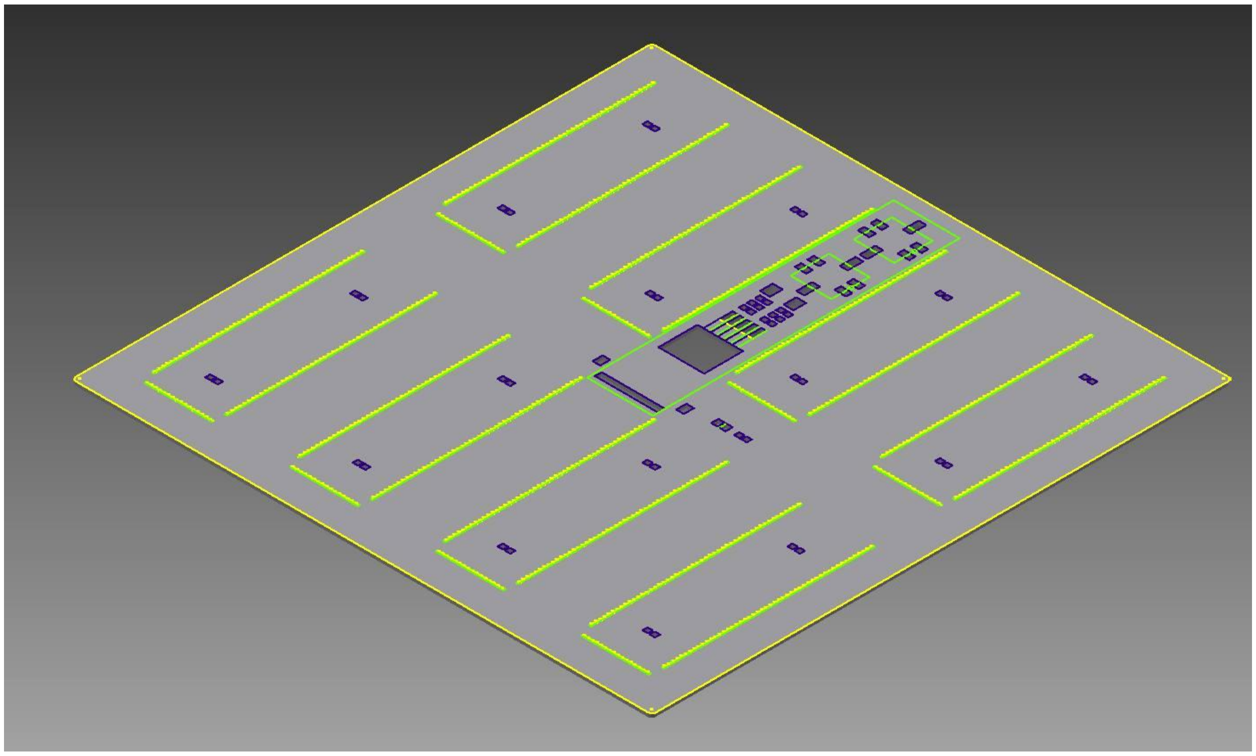


Figure 4 - 24: Intermediate step of board extrusion for design verification.

Before some of the panels were sent to manufacturing, physical tolerances of the panels were verified to ensure that the final design was able to perform as desired. This was conducted by using AutoCAD Inventor to extrude 3D representations of each PCB and virtually assemble the multiple-board assemblies to verify tolerances and component footprints. This was conducted in addition to the PCB manufacturer's minimum specification design rule checks required for board fabrication. This inter-board tolerance verification proved to be critical, as it indicated a spatial conflict between one of the LED driver daughter boards and a MOLEX connector on the multiplexing board. Making note of this, the design of the LED driver boards was adjusted to compensate for such tolerance issues. This is illustrated in the initial board design in figure 4.25 as compared to the revised design shown in figure 4.26.

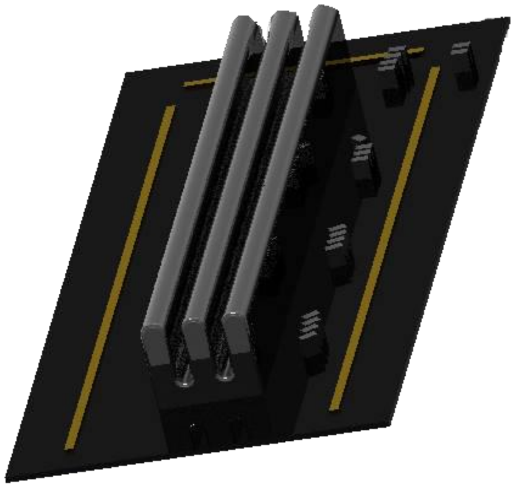


Figure 4 - 25: Original LED Direct-Drive Daughter Board Design

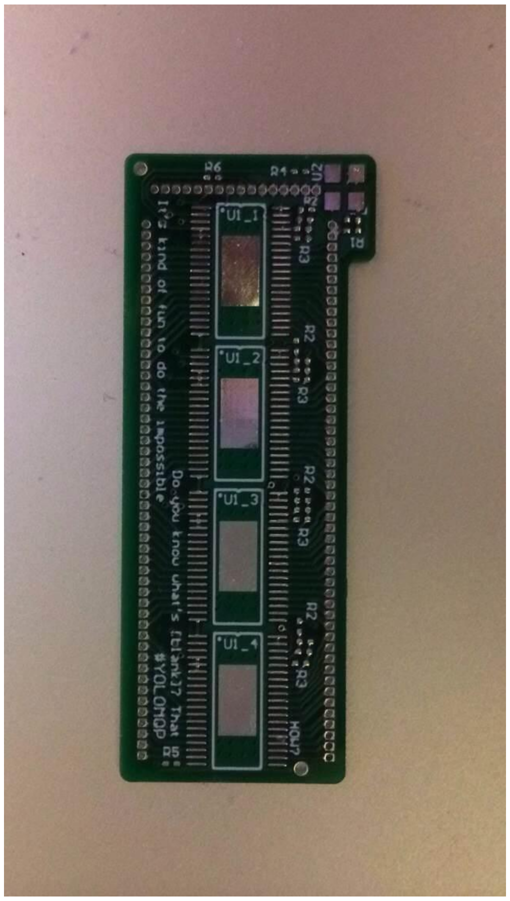


Figure 4 - 26: Modified Final LED Direct-Drive Daughter-Board Design

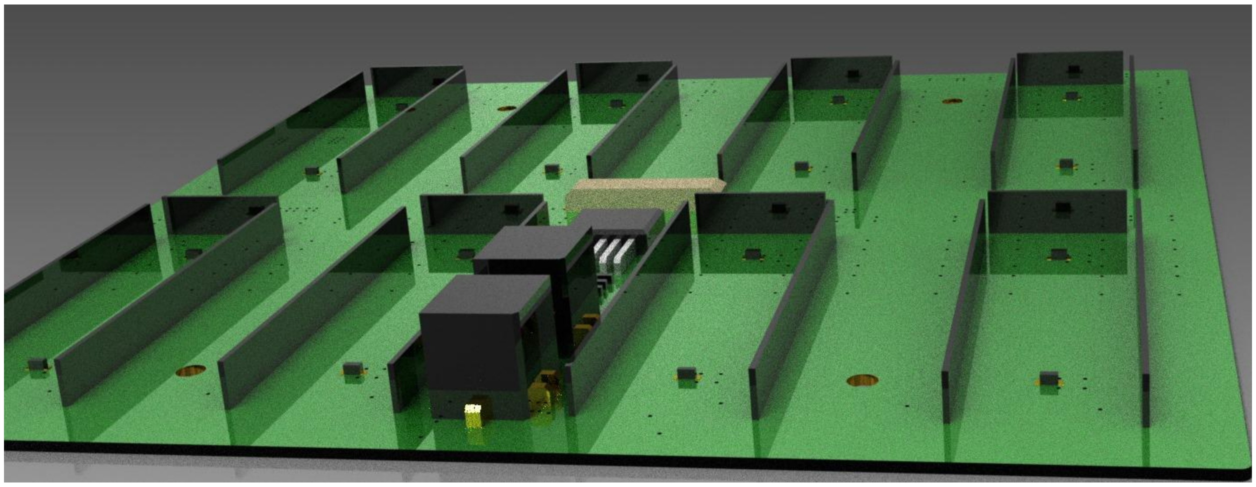


Figure 4 - 27: Direct-Drive LED Panel design shown in 3D for verification purposes

4.7.2 Chassis Design and Manufacturing

Development Chassis Design

Due to the implementation of development boards to improve the time-to-market of the product, each finished panel would be constructed with a minimum of 4 PCBs. These consisted of the LED Matrix panel, being either multiplexing or direct-drive, the Ethernet shield, ARM Cortex-M4 development board, and an optically-isolated UART transceiver board. As the Ethernet shield, ARM development board, and transceiver board were all coupled electrically with substantial amounts of 0.1 inch header pins, the mechanical connection inherent in the connectivity scheme proved to provide enough physical support to negate the need for additional reinforcement. As it was desirable to have a single assembly for each panel, the three boards joined by 0.1 inch header pins remained to be mechanically connected with the LED Matrix panel.

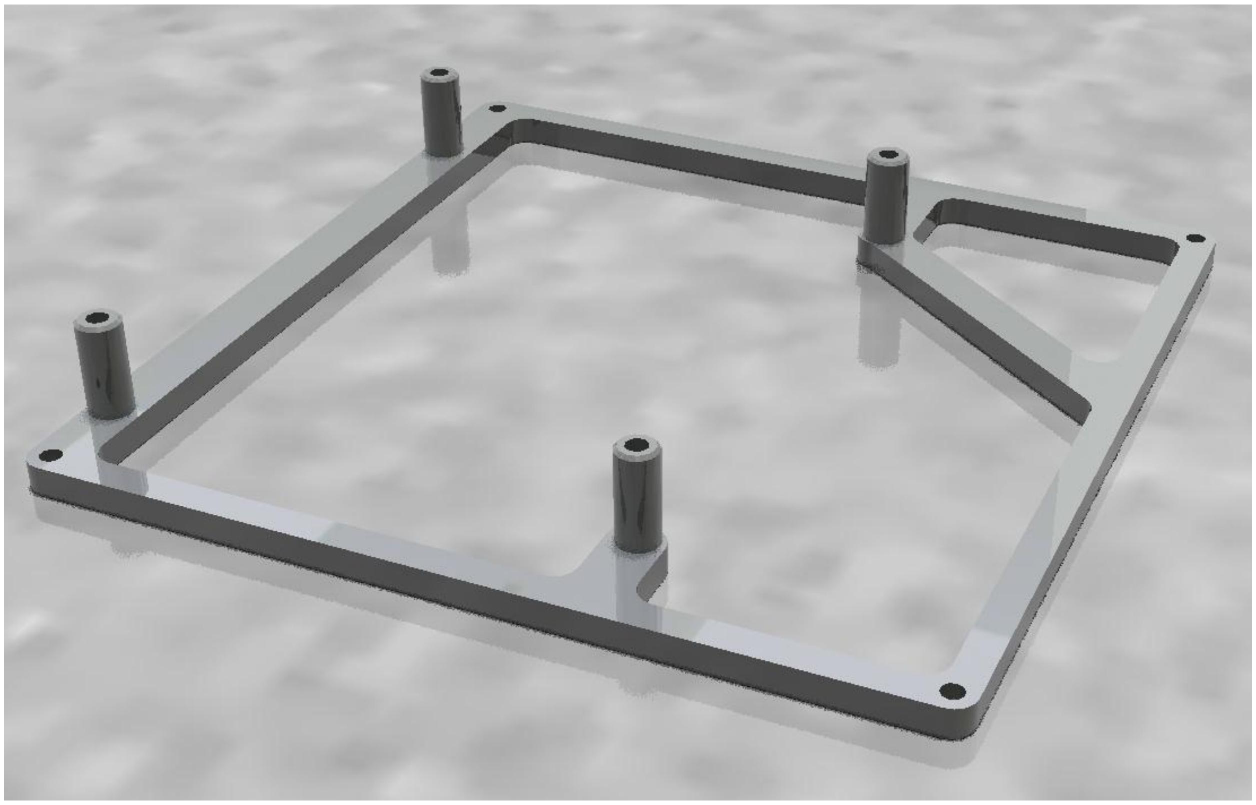


Figure 4 - 28: 3D rendering of standoff development frame

As the multiplexing LED Matrix boards were the first to be received from manufacturing, a series of chassis were designed to accommodate their specific mounting hole configuration first. This design also didn't require support of LED driver daughter boards like those in the direct-drive design, simplifying the chassis design further. To assist in transport and debugging of the panels, a small standoff frame was designed to secure all of the boards together. All of the original connections to the multiplexing matrix board and the development board were left open for development purposes. The resultant form was a series of lateral structural supports and four standoffs. The lateral support served to remove stress from the 4-layer design of the multiplexing board. This served to increase the lifespan of the panels, as it dampened any torsional forces incurred during transport. This frame provided a means to carry and stand a test panel and its supporting logic hardware. Above is a rendering of one of such frames seen in figure 4.28. This design was then realized through the use of rapid prototyping technologies. The lateral support design and development implementation can be seen in figures 4.29 and 4.30, respectively.

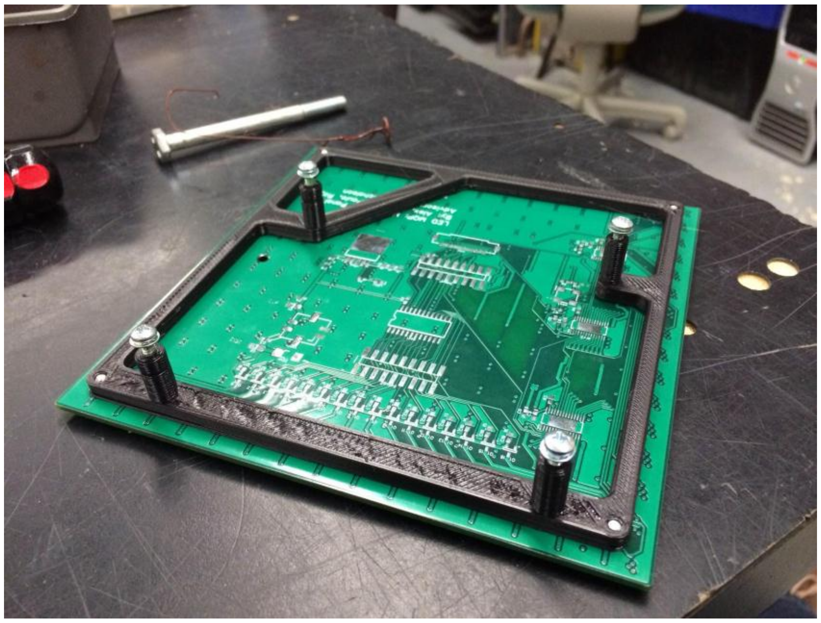


Figure 4 - 29: Standoff positioning

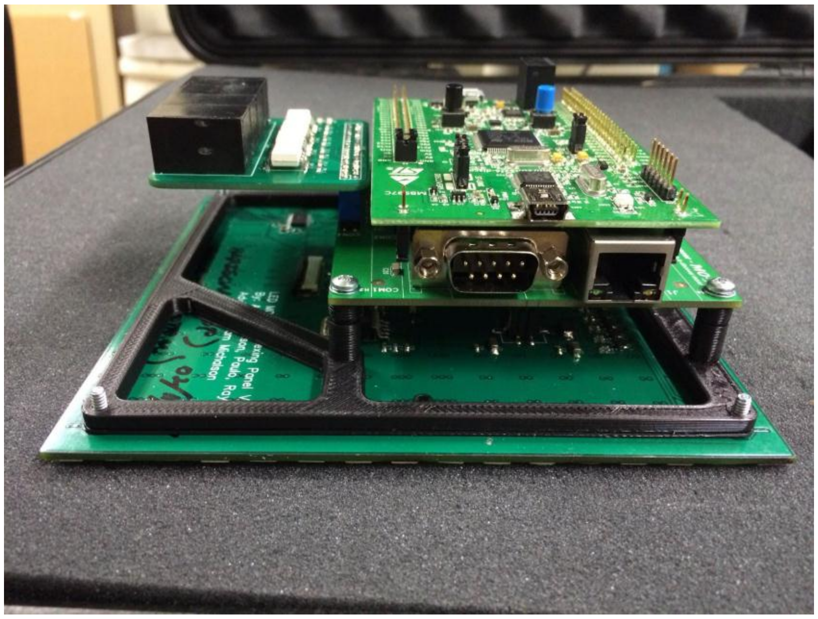


Figure 4 - 30: Development Chassis Implementation

Multiplexing Panel Enclosure

To provide a finished enclosure for the multiplexing panels, a chassis was designed to contain all of the required hardware for the panels and provide a means of easy tessellation for array formation. As the LED matrix panels were designed to effectively tessellate with no gap in between their respective edges, a design that mounted the panel as the front most component was chosen. This design also was self-contained within the dimensions of the panel, to enable edge-to-edge tessellation. As the scale of the project would only allow for a small number of panels to be constructed, it was determined to be unnecessary to integrate mechanical linking hardware into the panel enclosure. Since the array would likely be rapidly dynamically reconfigured and stacked, there would be little or no need to secure the development array. However, in the event that the panels would need to be installed, a mounting system was prototyped and secured on the rear of the panels. This would secure the panels to a singular master frame which would also serve to mount any power or network distribution peripherals. Below is a rendering of the multiplexing panel chassis and images of the final implementation, respectively.

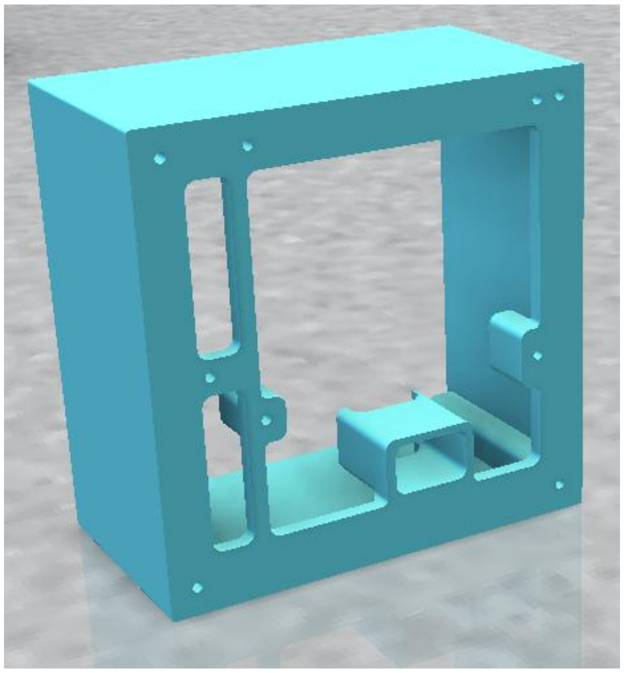


Figure 4 - 31: 3D rendering of final panel chassis.

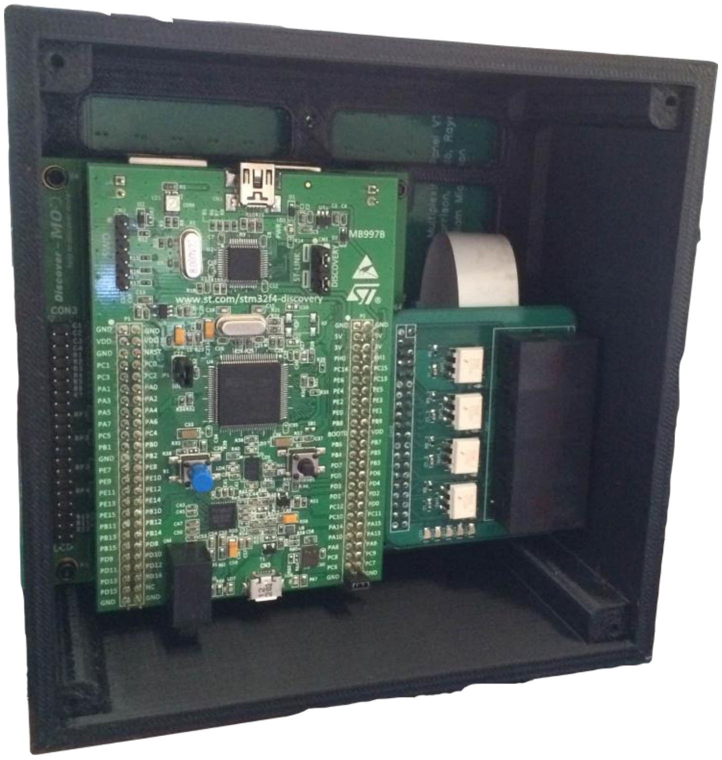


Figure 4 - 32: Chassis populated with logic electronics and development board.

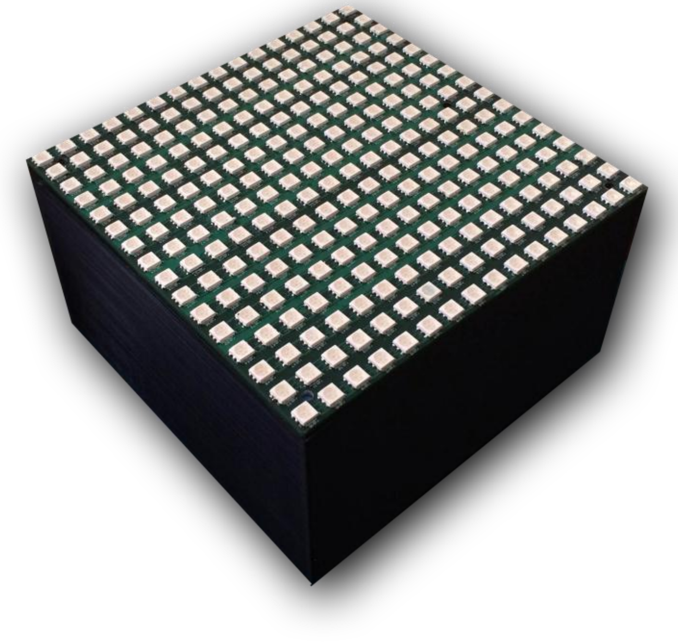


Figure 4 - 33: Chassis with multiplexing panel installed and sealed on front face.

Connector Selection

As one of the goals of the project was to design a ruggedized system, weatherproof, locking connectors were chosen to secure the data and power connections for the panels. Neutrik's series of locking connectors were chosen as they offer IP65 and IP67 certifications when connected. This would effectively seal the electronics from any harsh conditions that the panels might encounter. As Neutrik currently does not offer an RJ11-format locking connector, RJ45 connectors were chosen to be the exterior connectors for both the Ethernet link and UART link between panels. To avoid confusion, these connections would be labeled and color coded. However, as both the Ethernet connection and the four UART connections are electrically isolated, no damage would be incurred to the device, provided that Power over Ethernet (PoE) was not provided to the panels. The sealing connectors chosen for the rear data interface of the panels are shown below in Figure 4.34.

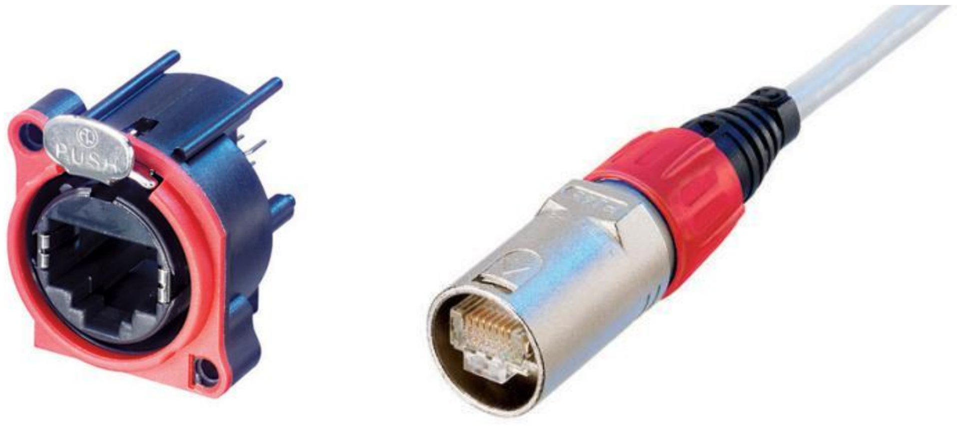


Figure 4 - 34: Neutrik Data connectors chosen to improve ruggedness.

As the panels were configured to run off of a DC power supply, a standard of using a 4-pin XLR connector was chosen to follow the conventions within the industry. This provided further intrusion protection on the rear of the panels while allowing for power to be easily distributed to panels. As the function of the panels produced in the project was to prove the viability of a self-addressing system, there was little need for daisy-chainable power pass-through connections. In most commercial product, there is some similar connection offering, allowing for power to be daisy chained through panels. This, however, potentially removes some of the redundancy in the array, as the panels then become primarily susceptible to power distribution failures over network data loss. For this reason, and to better suit the context of the project, only a single power input was used for each panel, eliminating the ability to daisy-chain power, improving the overall redundancy of the system. The power connectors chosen can be seen below.

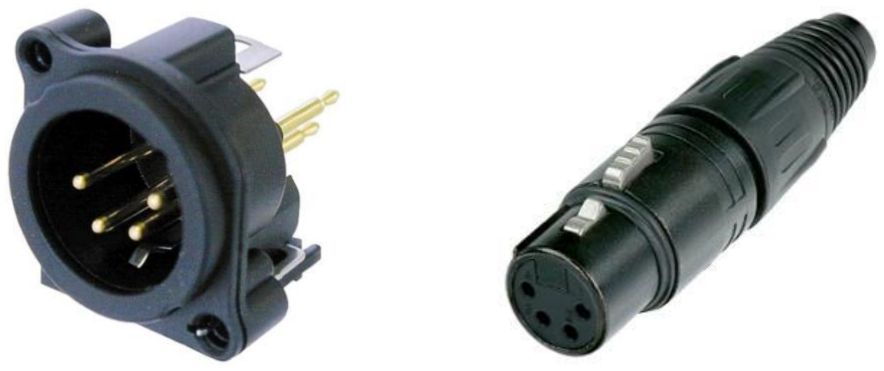


Figure 4 - 35: Locking power connectors chosen to fulfil industry conventions.

These connectors were then secured to the rear of the chassis through a custom-cut acrylic panel and sealed with silicon sealant. This served to seal the electronics from any form of dust or water intrusion, maintaining the IP65 certification, while enabling the electronics to remain accessible by removing the rear acrylic plate, shown below.

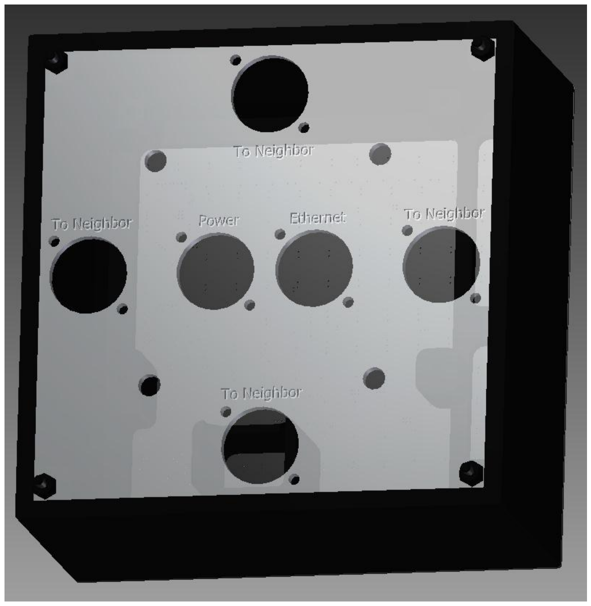


Figure 4 - 36: Rear connector housing and rigging points.

Chapter 5: Results

5.1 Still Image Calibration

Calibration of each of the panels to a uniform color was not completed. The proper general operation was verified on every panel however, and this ensured that there was some color uniformity. Defects in the actual LED manufacturing process lead to differences in color temperature and brightness between neighboring LEDs. These manufacturing errors can be corrected for by using small changes in drive current that can be controlled in the actual panel firmware. In an idealized final product, higher-quality LEDs would be selected to minimize lumen variance per LED package lot. As the generic LEDs selected for this project were primarily selected to minimize the overall cost of prototyping, individual component quality-control suffered, accounting for the majority of the brightness variances between individual pixels. While correction schemes were not implemented, implementing such in software would be a trivial task. In addition to this, simply improving the quality control of the LEDs used in manufacturing would vastly minimize such variances without the need to modify the existing panel firmware.

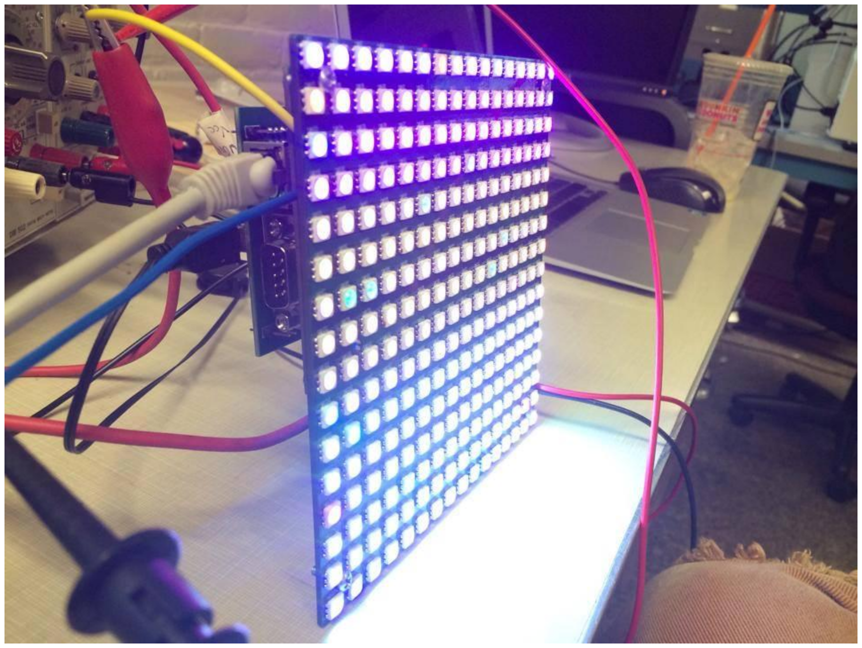


Figure 5 - 1: Testing color variations of the individual LEDs of a multiplexing panel.

5.2 Video Frame rate

Initial testing with the preliminary embedded and PC video processing software on a single panel (no frame syncing between panels) was measured to update the displayed video frame between approximately 21 to 25-Hz. This limitation was attributed to the delay in the video interface buffer provided by OpenCV at high resolutions. The idealized implementation of the software would use an architecture-dependent library that would be able to buffer video directly from memory as opposed to the slower alternative of waiting for the OS to grant OpenCV access to shared memory. As traditional film frame rates are at 24 fps, this generally is not an issue, as the LEDs of the display offer a much better control over persistence of vision than a typical LCD. These results were obtained by having an I/O pin toggle on the development board every time the panel received a frame. This pin was monitored with an oscilloscope, and the pulse intervals were averaged. The tests showed that at full 1080p resolutions and 90% simulated network utilization, the array was still capable of reproducing video content at acceptable frame rates. These frame rates were significantly improved when viewing 720p content or lower. A potential solution for this issue would to increase the clock speed of the CPU of the host PC or to ideally offload the image processing to a GPU. However, as a goal of the project was to make a standalone application that was platform-independent, all of the video processing was performed on the CPU.

5.3 Self-Addressing Stability

The self-addressing protocol was successfully implemented on a small scale. This physical small scale proved to possess predictable behavior shown in large-scale software simulations. Small test cases of two or three panels were shown to rapidly self-address. In addition to this, the software successfully handled the dynamic remapping of video content. Examples of this occurring were referenced in section 4.5. Due to the small size of the testing, it was impractical to include a small switch on each panel. Because of this, full physical testing of RSTP convergence when paired with the self-addressing protocol was not conducted. However, as extensive testing and simulation was done to verify the feasibility of using such technologies in a final implementation, it is believed that incorporating both technologies in one physical system would not yield and negative effects.

Chapter 6: Conclusions and Future Work

6.1 Project Outcomes

The project started out with the goals of developing a market ready modular LED display system that would be easy to install and reliable for use in live video applications. Throughout the course of three academic terms, the team tackled the four areas of panel configuration, powering topology, firmware and high level software concurrently, which made possible for the objectives to be achieved. The final project offers a strong demonstration of the viability for use of self-addressing algorithms to ease display installation while also permitting more flexibility in its installation and tolerance to failure. While time-to-market and budget constraints prevented the team from successfully implementing a market-ready product, most of the overall design concepts can be directly incorporated into a commercial product.

6.2 Future Design Suggestions

Even though the project succeeded in demonstrating the usefulness of self-addressing and redundancy networking schemes for large display installations, it is far from being market ready. Further work in the project should be conducted in three main areas: Hardware powering design; explore alternative communication methods; and, solidify product into less but custom made parts.

The first area that warrants more exploration is the hardware powering design which includes both the LED driving mechanism as well as the overall powering of the module. Although a multiplexing and direct driving topology were explored, due to time constraints, several methods within each remain untouched. For multiplexing these include: Segmenting the display into multiple separately driven displays; random scanning of the rows to reduce flickering and increase camera compatibility; as well as performing other methods of scanning. The direct drive in inherently a more complex but promising design and it would be interesting to determine if great power enhancement couldn't be achieved by blanking different regions of the display cyclically to reduce power consumption. Although the hardware is capable of such, there was no time to develop software for it. Finally, in the case of using Ethernet and multiplexing technology it may be possible to power the panels directly from the switches using PoE which would further reduce the complexity of installation. It is clear that although a lot was fitted within the scope of the project areas for continued exploration are many.

Use of different communication schemes have been proposed early in the design stages of the project but the team quickly converged on Ethernet for reasons mentioned in chapter 2 with an additional serial link for inter panel communication. Nonetheless, exploration of a high speed serial link by itself would be an interesting option for reducing the overall latency involved with Ethernet. In contrast, the use of an Ethernet link alone with a custom made Ethernet controller capable of reporting the MAC of the neighboring panel and their respective port would allow for reducing the overall cable count while maintaining the simplicity and reliability of a widely used protocol such as Ethernet.

The last but probably the most important are that is left to be completed in the project is the consolidation of the prototypes into fewer and custom made parts that perform their function optimally. Currently each of the panels consists of prototyping board for the microcontroller, a STMicroelectronics development board with the Ethernet transceiver, a custom made shield board, and the panel. This design should be reduced into a single custom made logic board that replaces the microcontroller break out, the shield and the Ethernet board with a separate board for the panel. A custom power supply with power factor correction should also be developed and included in the design.

Although the higher level goals of developing a self-addressing scheme for an easy to use and robust modular LED display technology was accomplished much work can be done to further advance the display and make into a market ready solution. Among these further experimentation into the driving of the LEDs, the communication schemes and consolidating the design into single custom made parts is warranted.

References

[1] http://pixled.com/product/21/F-11

[2] http://www.digiled.com/digiLED_IT_Tile_MR.pdf

[3]http://www.qs-tech.com/suggestion-details-20.html

[4]http://www.elationlighting.com/pdffiles/elation_ept9ip_specification_sheet.pdf

[5]http://www.digiled.com/Navigator.pdf

[6] http://www.elationlighting.com/ProductDetails.aspx?ItemNumber=1834&MainId=1&Category=27

[7]http://en.wikipedia.org/wiki/File:BNC_connector_(male).jpg

[8]http://en.wikipedia.org/wiki/File:HDMI-Connector.jpg

[9]http://www.escotal.com/osilayer.html

[10]http://www.cisco.com/c/en/us/support/docs/lan-switching/spanning-tree-protocol/5234-5.html

[11]http://www.routeralley.com/ra/docs/stp.pdf

[12]http://en.wikipedia.org/wiki/Times_Square_Studios

[13]http://en.wikipedia.org/wiki/Yas_Viceroy_Abu_Dhabi_Hotel

[14]http://en.wikipedia.org/wiki/File:Yas_Marina_Hotel_by_Rob_Alter.jpg

[15]http://www.itechdigitalproduction.com/wp-content/uploads/2013/11/LED-Wall.jpg

[16]http://www.elationlighting.com/ProductDetails.aspx?ItemNumber=1834&MainId=1&Category=27

[17]http://www.lumascape.com/html/IP_IK_rating.htm

[18]http://standards.ieee.org/getieee802/download/802.1D-2004.pdf

[19]http://www.costcentral.com/product-images-new/nortel-al2515e11-e6.jpg

[20]blackmagicdesign.com

[21]http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/PF252419

[22] http://www.noodlehed.com/ebay/datasheets/plcc6rgbled.pdf

[23]http://www.ti.com/lit/ds/symlink/cd74hc154.pdf

[24] http://www.ti.com/lit/ds/symlink/cd74hc154app.pdf

[25] http://www.ti.com/product/tlc5951

[26] http://www.ti.com/lit/an/slma002g/slma002g.pdf

Appendix A: Video Processing Top-Level UML

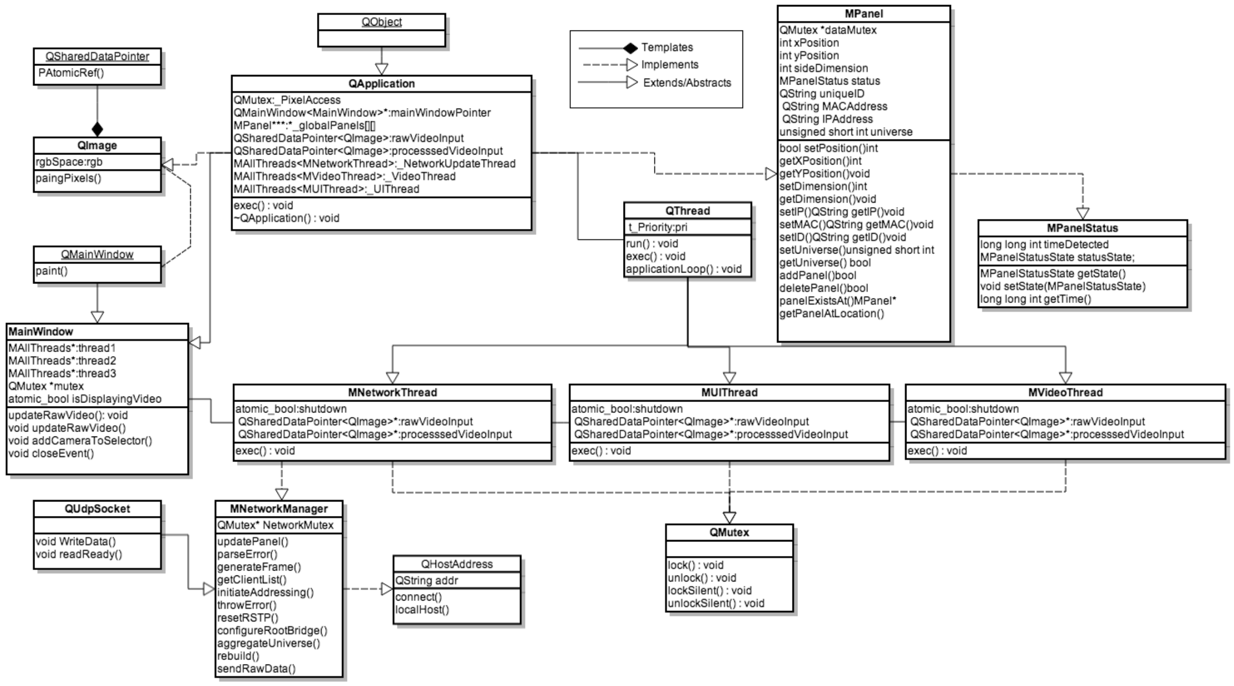


Figure A - 1: Video Processing Top-Level UML Diagram

Appendix B: Shield Board Reference Manual Content

During development, reference manuals were made of all created systems to aid in the programming and software development. The following sections is content taken directly from each manual. Appendix B contains the Shield Board reference manual content.

B-1 Pin Mapping

B-1.1 InterBoard FPC Connector

Connector bridges the shield board to the led panel. Currently using a 30 pin Molex FPC connector model 528933095.

Table B - 1: Shield FFC Connector Pin-out

Pin #	Pin Name	Connected To	Pin Description
1	RES	--	--
2	RES	--	--
3	GND	GND	Common Ground
4	GSSCK	CON2 P17	Gray Scale Clock for LED Drivers
5	GSMOSI	CON2 P19	Gray Scale Master Out for LED Drivers
6	GSMISO	CON2 P16	Gray Scale Slave Out for LED Drivers
7	GSLAT	CON2 P18	Gray Scale Latch
8	GND	GND	Common Ground
9	DCMOSI	CON2 P24	Dot Correction Master Out for LED Drivers
10	DCSCK	CON2 P14	Dot Correction Clock Line for LED Drivers
11	DCMISO	Con2 P25	Dot Correction Slave our for LED Drivers
12	GND	GND	Common Ground
13	XBLNK4	Con2 P9	Blank Signal for LED Drivers Group 4
14	XBLNK3	CON2 P8	Blank Signal for LED Drivers Group 3
15	XBLNK2	CON2 P7	Blank Signal for LED Drivers Group 2
16	XBLNK1	Con2 P6	Blank Signal for LED Drivers Group 1
17	GND	GND	Common Ground
18	RES	--	--
19	RES	--	--
20	RES	--	--
21	RES	--	--
22	RES	--	--
23	RES	--	--
24	GND	GND	Common Ground
25	I2C_SCL	CON2 P30	I2C Serial Clock Line
26	I2C_SDA	CON2 P29	I2C Serial Data Line
27	GND	GND	Common Ground
28	3V3	3V3	3V3 Voltage Source

29	3V3	3V3	3V3 Voltage Source
30	GND	GND	Common Ground

B-1.2 InterPanel RJ45 Connectors

These connectors make an isolated bridge between each complete panel (LED + Logic + Power); providing two way opto-coupled serial communication. In order to avoid the need to use cross over cable each 2 connectors have a different pin assignment that create a mating pair.

Type 1: Connectors 1/2

Table B - 2: RJ11 UART Connector Pin-out, Connectors 1-2

Pin #	Pin Name	Connected To	Pin Description
1	nDin	CON2 P3/P33	Shied Board Sends to Other Board
2	GND	GND	Common Ground (Drives Optocoupler LED)
3	nS+	Optocoupler	Shield Board Receives from Other Board
4	nS-	Optocoupler	Shield Board Received from Other Board
5	RES	--	--
6	RES	--	--

Type 2: Connectors 3/4

Table B - 3: RJ11 UART Connector Pin-Out, Connectors 3-4

Pin #	Pin Name	Connected To	Pin Description
1	nS+	Optocoupler	Shield Board Receives from Other Board
2	nS-	Optocoupler	Shield Board Received from Other Board
3	nDin	CON2 P40/P35	Shied Board Sends to Other Board
4	GND	GND	Common Ground (Drives Optocoupler LED)
5	RES	--	--
6	RES	--	--

B-1.3 Shield Board to Ethernet Shield Connector: CON2

Connectors the shield board to the discovery Ethernet shield which In turn connects to the Discovery board for the STM32F407VG Microcontroller. This connector serves as both a data interface between the two boards and provides power to the discovery assembly when in operation.

Table B - 4: STM32F4-BaseBoard to Shield Pin-Out

Pin #	Pin Name	Connected To (Shield)	Connected To (Discovery)	Pin Description
1	--	--	--	--
2	--	--	--	--
3	1Din	Con1 P6	PD5/TX2	Data Out to Neighboring Panel
4	1Do	Opt1_4	PD6/RX2	Data in from Neighboring Panel
5	GND	GND	GND	GND
6	XBLINK1	FPC1 P16	PD2/TX2	Digital Line to Panel
7	XBLINK2	FPC1 P15	PC12/TX5	Digital Line to Panel
8	XBLINK3	FPC1 P14	PA8/I2C3_SCL	Digital Line to Panel
9	XBLINK4	FPC1 P13	PA10/RX1	Digital Line to Panel
10	GND	GND	GND	GND
11	--	--	--	--
12	--	--	--	--
13	--	--	--	--
14	DCSCK	FPC1 P10	PA5/SPI1_SCK	SPI Line to Panel
15	GND	GND	GND	GND
16	GSMISO	FPC1 P6	PB14/SPI2_MISO	SPI Line to Panel
17	GSSCK	FPC1 P4	PB10/SPI2_SCK	SPI Line to Panel
18	GSLAT	FPC1 P7	PC2	Digital Line to Panel
19	GSMOSI	FPC1 P5	PC3/SPI2_MOSI	SPI Line to Panel
20	3V3	3V3	VDD3	Regulated Voltage in from Panel
21	3V3	3V3	VDD5	Regulated Voltage in from Panel
22	LED1	LED1	PB1	Indicator LED 1
23	LED2	LED2	PB0	Indicator LED 2
24	DCMOSI	FPC1 P9	PB5/SPI1_MOSI	SPI Line to Panel
25	DCMISO	FPC1 P11	PB4/SPI1_MISO	SPI Line to Panel
26	GND	GND	GND	GND
27	LED3	LED3	PD1	Indicator LED 3
28	LED4	LED4	PD0	Indicator LED 4
29	I2C_SDA	FPC1 P26	PB9/I2C1_SDA	Generic I2C Line to Panel
30	I2C_SCL	FPC1 P25	PB8_I2C1_SCL	Generic I2C Line to Panel
31	GND	GND	GND	GND
32	2Do	Opt1_3	PC11/RX3/RX4	Data in from Neighboring Panel
33	2Din	Con1 P12	PC10/TX3/TX4	Data Out to Neighboring Panel

34	3Do	Opt1_2	PD9/RX3	Data in from Neighboring Panel
35	3Din	Con1 P16	PD8/TX3	Data Out to Neighboring Panel
36	GND	GND	GND	GND
37	--	--	--	--
38	--	--	--	--
39	4Do	Opt1_1	PC7/RX6	Data in from Neighboring Panel
40	4Din	Con1 P22	PC6/TX6	Data Out to Neighboring Panel

B-2 Schematics, Copper and Diagrams

B-2.1 High Level Block Diagram

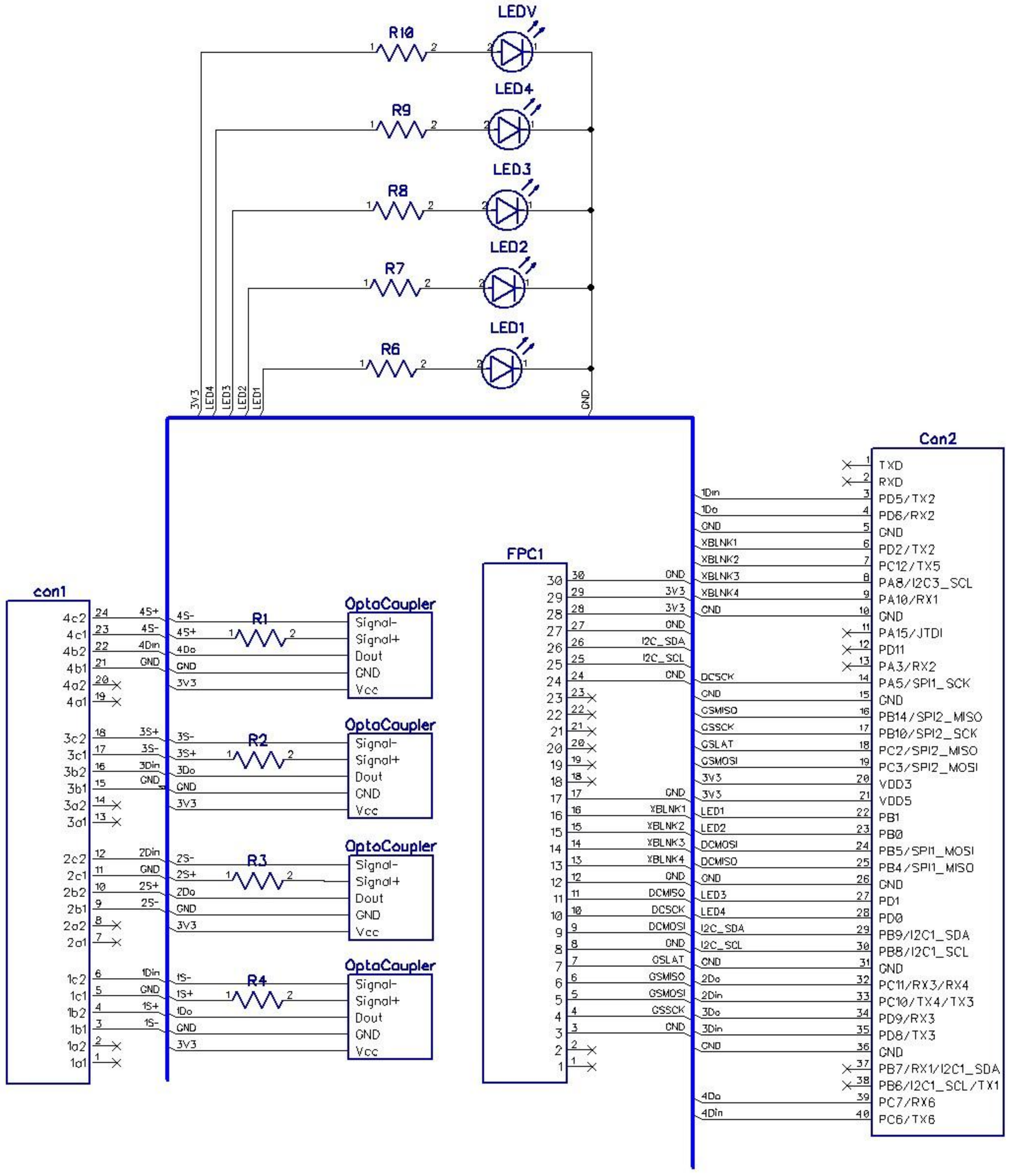


Figure B - 1: Shield Board High Level Diagram

B-2.2 Optocoupler Schematic

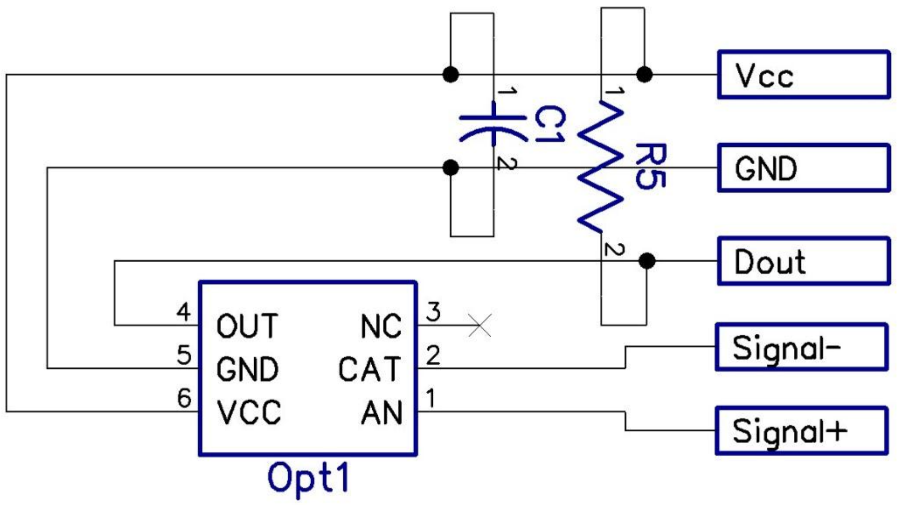


Figure B - 2: Optocoupler Schematic

B-2.3 Copper Top Layer

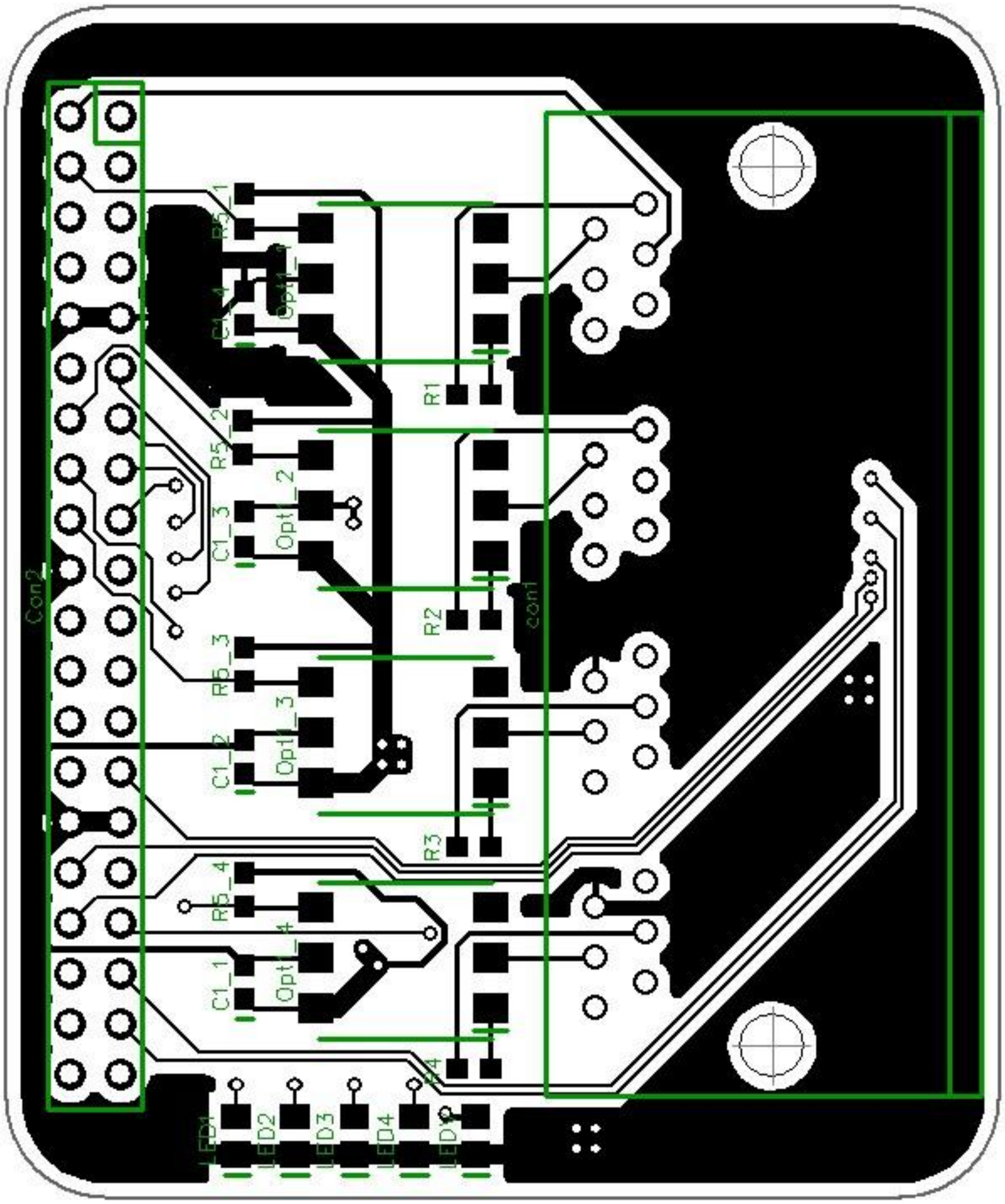


Figure B - 3: Top view of copper artwork with silkscreen.

B-2.4 Copper Bottom Layer

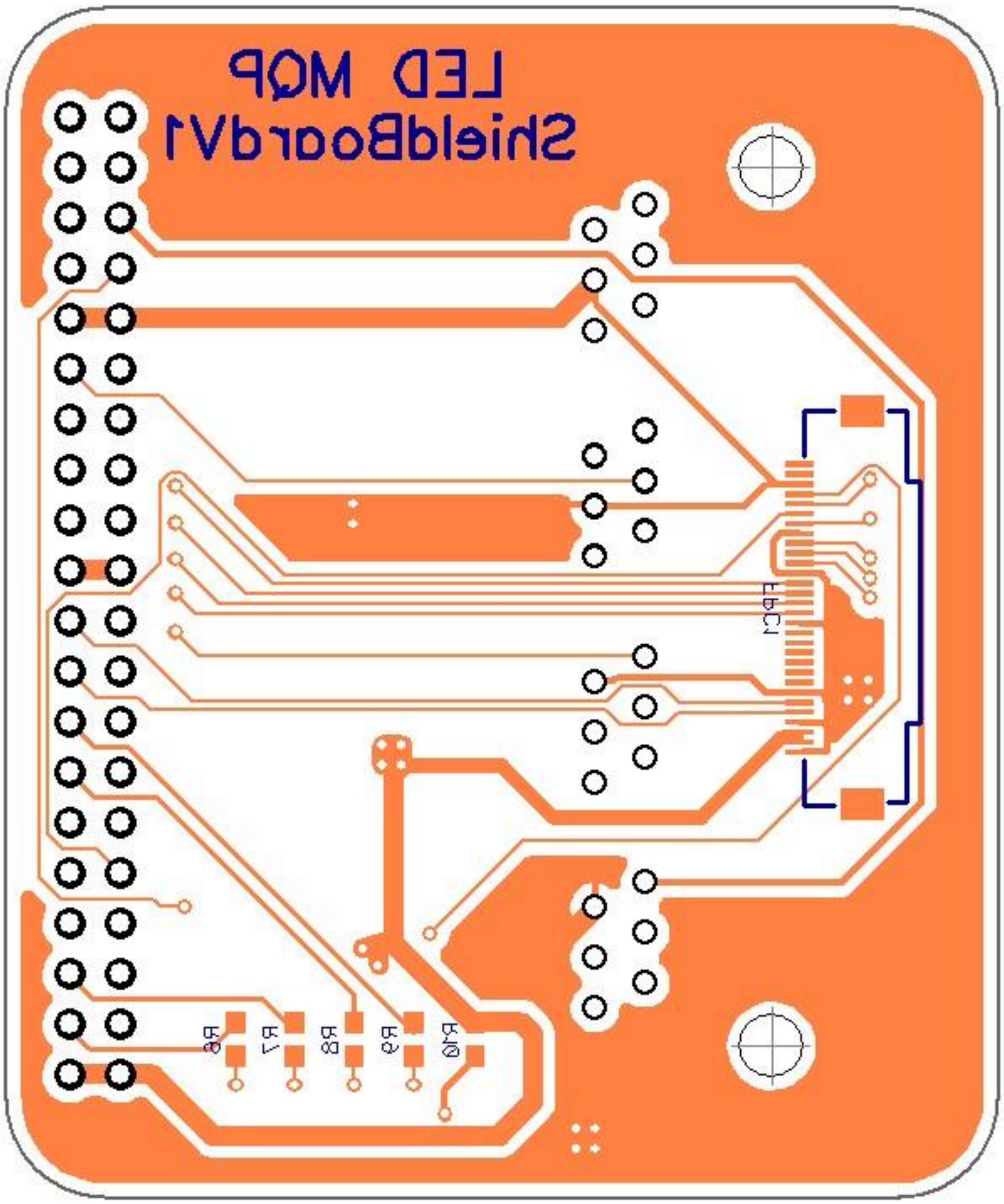


Figure B - 4: Bottom view of copper artwork with silkscreen.

B-3 Component List

Table B - 5: Component BOM for the Shield Board

RefDes	Type	Value	Description
C1_n	Cap C 0603	100 nF	Decoupling Capacitors for Opto-couplers.
Con1	Connector	SS-666604-NF	4X Rj45 Connector for conection between panels.
Con2	Connector	2X20 Female 2.54mm	Connector for attaching to Ethernet Shield
FPC1	Connector	52893-3095	FPC Connector for Attaching Shield Board to LED Board
LED1-LED4, LEDV	LED 0805	--	Indicator LEDs
Opt1_n	Optocoupler	H11L1SVM	Optocoupler for isolating boards
R1-R4	Res 0603	100 OHM	Current Limiter for Optocoupler LEDs
R6-R10	Res 0603	100 OHM	Current Limter for Indicator LEDs
R5_n	Res 0603	100 KOHM	Pull-Up for Optocoupler Data Out Lines

B-4 Errata

Table B - 6: Shield Manual Errata List

Error / Correction	Description	Date
Part Type CON2	Changed from Male to Female	2/10/14
Part Number OPT1	Changed to SMD Package	2/10/14

Appendix C: Multiplexing LED Panel Reference Manual

The following are contents scraped directly from the reference manual created to aid in the development of the software for the multiplexing LED panel.

C-1 Component/Connector Pin Out and Description

C-1-1 LED Drivers: TLC5940

The 16-channel sinking LED drivers are used in synchrony with the MOSFETs to systematically create a 2D image through persistence of vision. Table C-1 outlines the data connections of the drivers.

Table C - 1: Multiplexing LED Driver Pin-out

Pin	Connected To	Description
2: Blank	FPC25	When Blank is HIGH all outputs are OFF
3: XLAT	FPC24	When HIGH latches data to output registers
4: SCLK	FPC27	Serial Clock In
5: SIN	FPC26 or drv(n-1)	Serial Data In
6: VPRG	FPC20	When LOW device in GS mode, else device in DC mode
23: XERR	Debug1 15-17	Error output pin. Open drain no pull resistors on board
24: SOUT	drv(n+1)	Serial data out to next driver
25: GSCLK	Osc1 / FPC21	Reference clock for PWM
26: DCPRG	FPC22	DC Data In

For operational Specifications see the datasheet below: http://www.ti.com/lit/ds/symlink/tlc5940.pdf

C-1-2 Decoder: CD74HC154M

This component is used to sequentially power the MOSFETs responsible for selecting the LED line that will be powered by the drivers. Table C-2 outlines the principal data lines to the component.

Table C - 2: Multiplexing Decoder Pin-out

Pin	Connected To	Description
18: E1	FPC5	Active-Low Decoder Enable Pin
19: E2	FPC6	Active-Low Decoder Enable Pin
20: A3	FPC15	Address Selector Pin
21: A2	FPC16	Address Selector Pin
22: A1	FPC17	Address Selector Pin
23: A0	FPC18	Address Selector Pin

For operational specifications see the datasheet below: http://www.ti.com/lit/ds/symlink/cd74hc154.pdf

C-1-3 FPC Connector

This connector interfaces the multiplexing panel with the controller board. Both data and low power lines are available in the connector as specified in Table C-3.

Table C - 3: FFC Connector Pin-out

Pin	Connected To	Description
1: GND	GND	Power Supply
2: 3V3	3V3	Power Supply
3: 3V3	3V3	Power Supply
4: GND	GND	Power Supply
5: MUX_E1	DEC18: E1	MUX Enable Pin Active-Low
6: MUX_E2	DEC19: E2	MUX Enable Pin Active-Low
7: GND	GND	Power Supply
14: GND	GND	Power Supply
15: MUX_A3	DEC20: A3	Mux Address Pin
16: MUX_A2	DEC21: A2	Mux Address Pin
17: MUX_A1	DEC22: A1	Mux Address Pin
18: MUX_A0	DEC23: A0	Mux Address Pin
19: GND	GND	Power Supply
20: VPRG	DRVn6: VPRG	LED Driver Mode Selector
21: GSCLK	DRVn25: GSCLK	LED Driver PWM Clock Reference
22: DCPRG	DRVn26: DCPRG	LED Driver Dot Correction Serial In
23: GND	GND	Power Supply
24: XLAT	DRVn3: XLAT	LED Driver Latch (Active-High)
25: BLANK	DRVn2: BLANK	LED Driver Blank Pin (Active-Low)
26: DRV_SIN	DRV1(5): SIN	Serial Data to LED Drivers
27: SCLK	DRVn4: SCLK	Serial Clock for LED Drivers
28: GND	GND	Power Supply

Note that pins that are not mentioned in table C-3 are not connected and are reserved for future expansion.

C-1-4 Debug1 Connector

Debugging header 1. Saleae Logic-Compatible. This header debugs major signal lines and makes voltage measurement points available. See Table C-4.

Table C - 4: Debug Header #1 Pin-out

Pin	Connected To	Description
1: GND	GND	Common Point
2: Blank	FPC25, DRVn2	LED Driver Blank Signal
3: DCPRG	FPC22, DRVn3	LED Driver Programming Pin
4: DRV_SIN	FPC26, DRV1(5)	LED Driver Serial In
5: GSCLK	FPC21 or OSC	Led Driver PWM Reference Clock
6: SCLK	FPC27, DRVn4	LED Driver Serial Clock
7: VPRG	FPC20, DRVn6	LED Driver Mode Selector
8: XLAT	FPC24, DRVn3	LED Driver Latch Pin
9: 3V3	3V3	Voltage Supply
10: MUX_A0	FPC18, DEC23	Decoder Address Selector
11: MUX_A1	FPC17, DEC22	Decoder Address Selector
12: MUX_A2	FPC16, DEC21	Decoder Address Selector
13: MUX_A3	FPC15, DEC20	Decoder Address Selector
14: 5V	5V	Voltage Supply
15: XERR1	Drv1(23)	Error Detected Pin. Open-Drain No-Pull
16: XERR2	Drv2(23)	Error Detected Pin. Open-Drain No-Pull
17: XERR3	Drv3(23)	Error Detected Pin. Open-Drain No-Pull
18: GND	GND	Common Point

C-1-5 Debug2 Connector

Debugging header. Saleae Logic-Compatible. Makes available MOSFET gates for sequencing analysis. See Table C-5.

Table C - 5: Debugging Header #2 Pin-out

Pin	Connected To	Description
1, 18: GND	GND	Common Point
2-9: Mn	Q1_n: Gate	Connect to Even MOSFETs 0,2,4…, 14
10-17: Mn	Q1_n: Gate	Connect to Odd MOSFETS 15, 13, 11…, 1

C-2 Schematic and Description

C-2-1 High Level Diagram

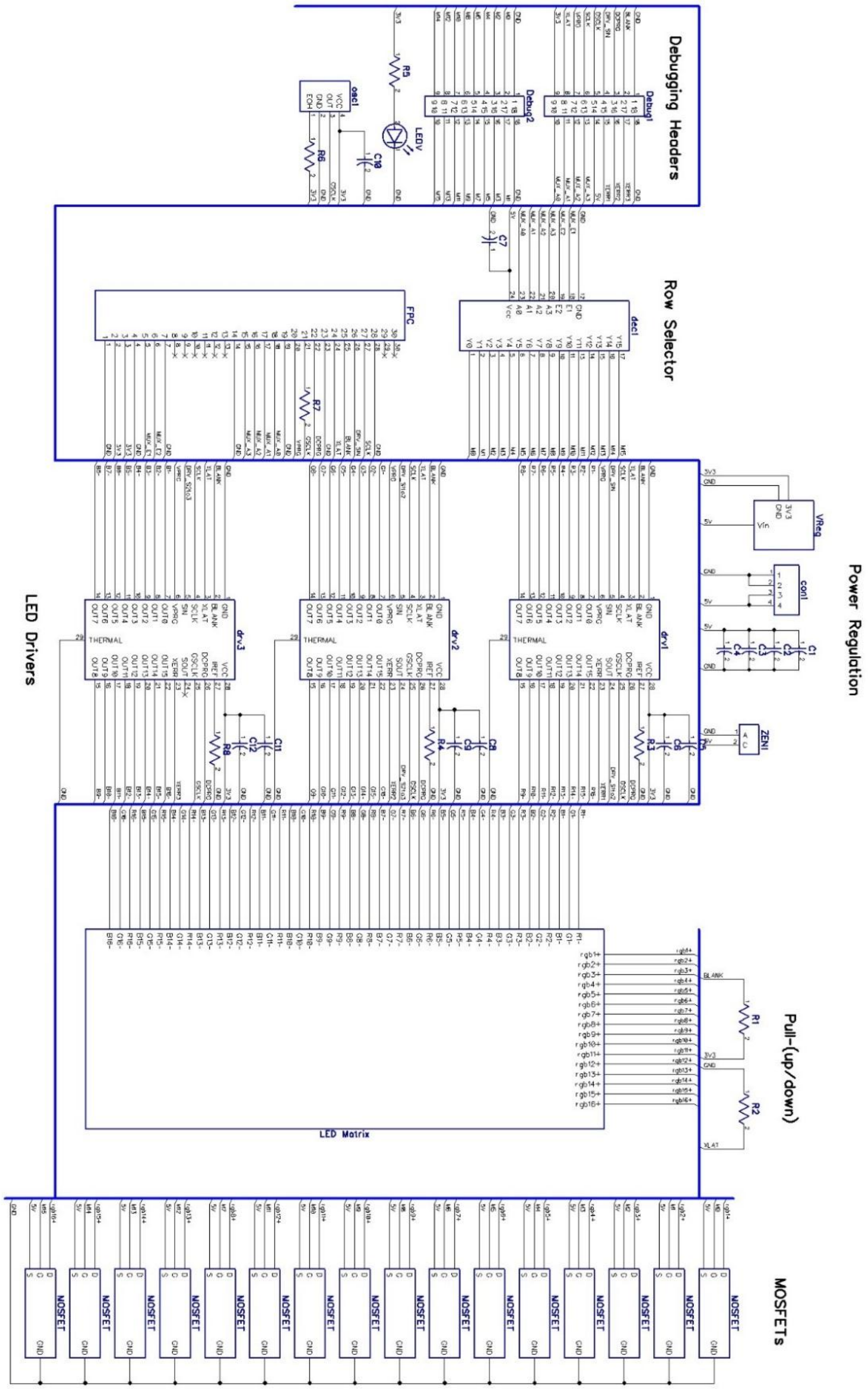


Figure C - 1: Multiplexing LED Panel High-Level Schematic

C-2-2 Voltage Regulator

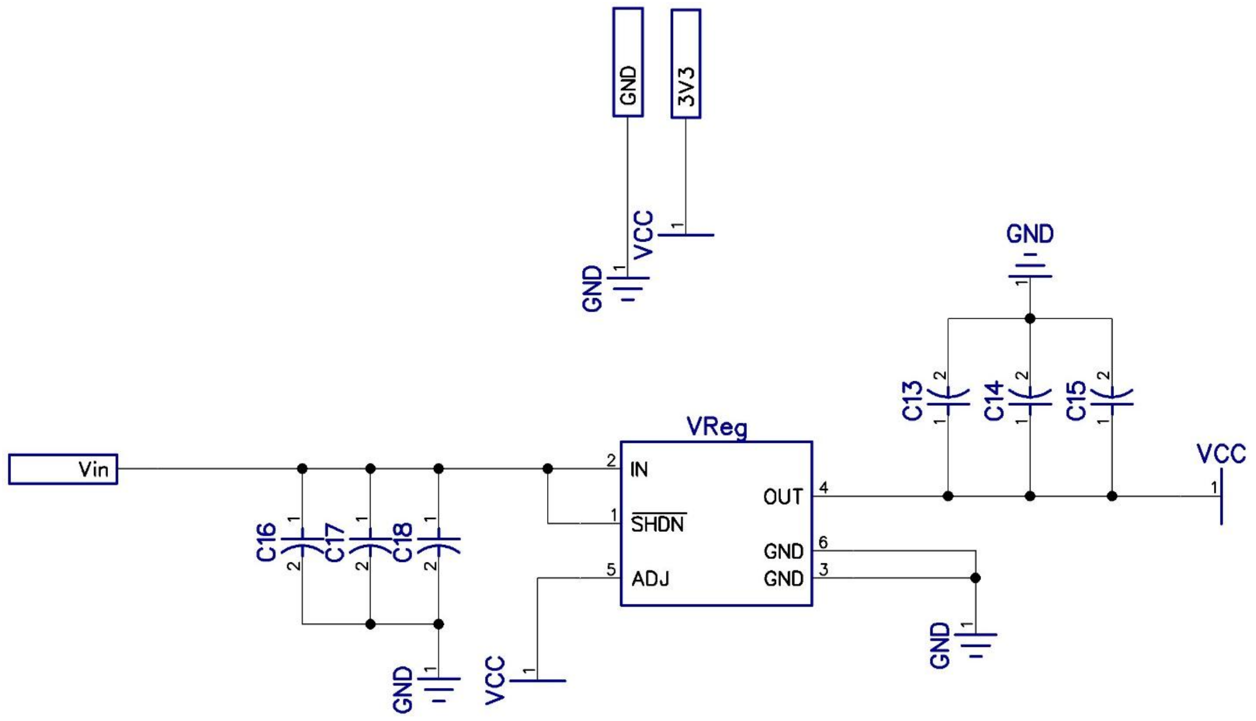


Figure C - 2: Multiplexing LED Panel Voltage Regulator Schematic

C-2-3 LED Matrix

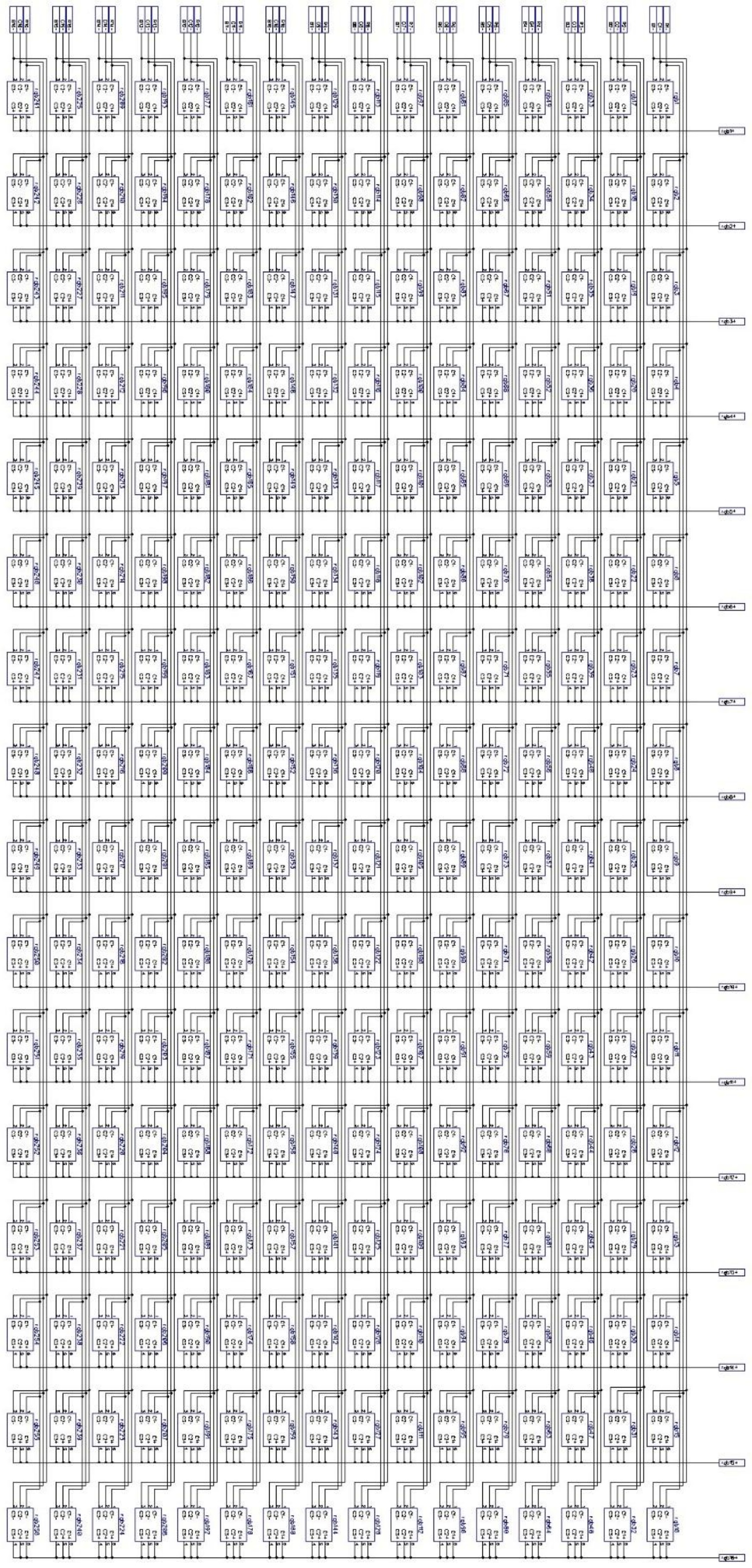


Figure C - 3: Multiplexing LED Panel LED Matrix Schematic

C-3 Copper Artwork

C-3-1 Top Layer

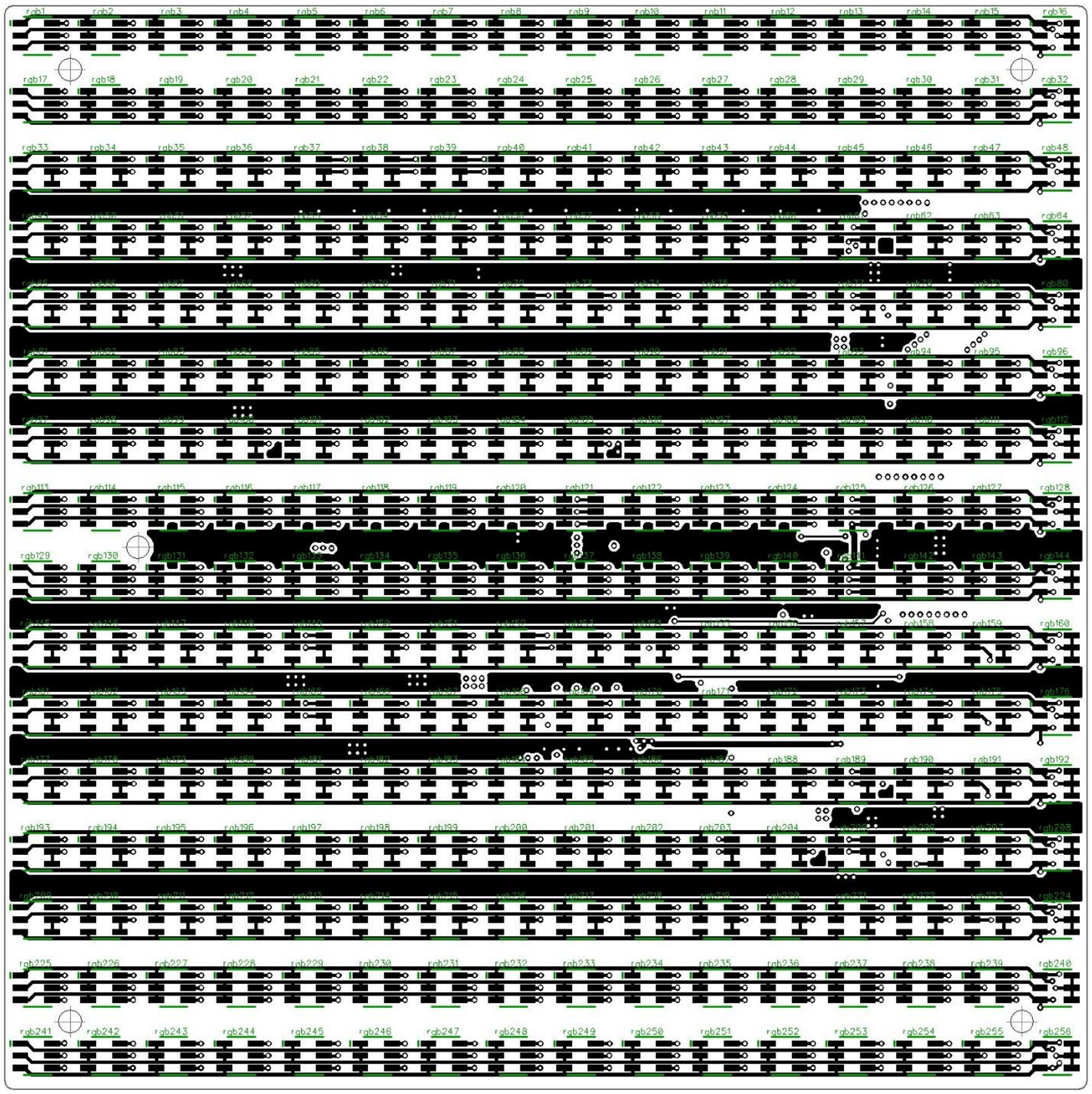


Figure C - 4: Multiplexing LED Panel Top-layer Copper and Silkscreen

C-3-2 Inner 1

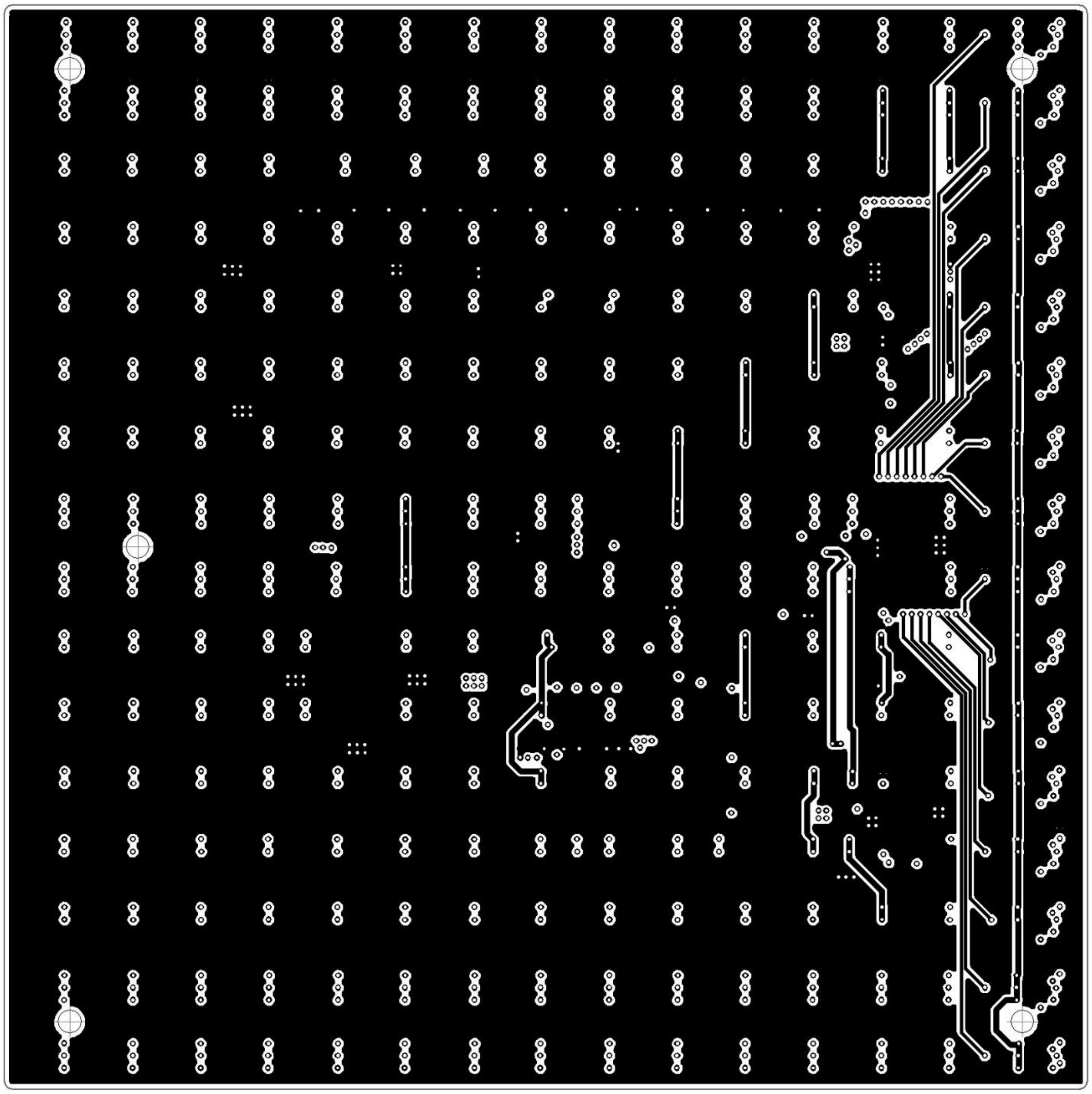


Figure C - 5: Inner Layer 1 Copper

C-3-3 Inner 2

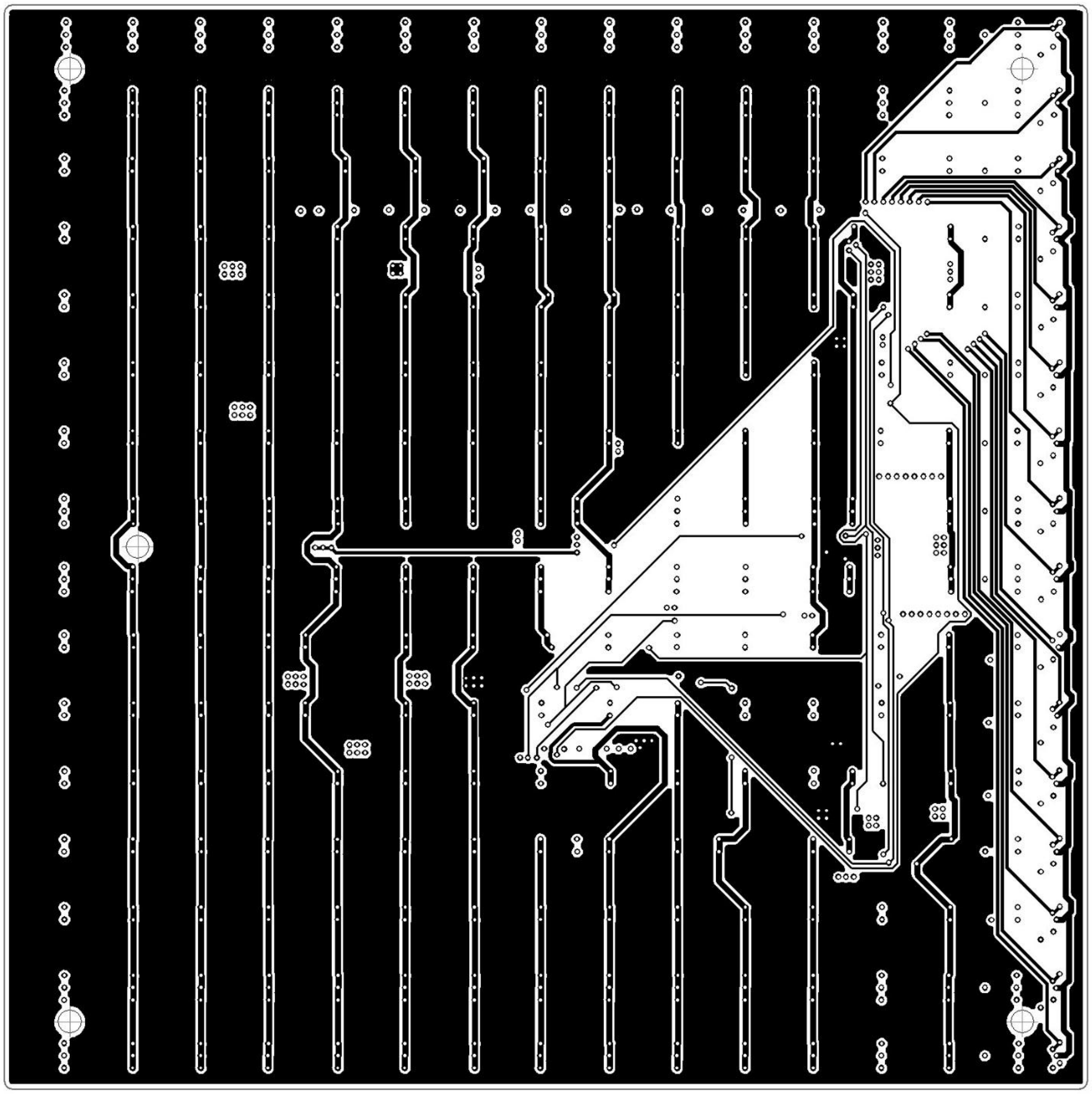


Figure C - 6: Inner Layer 2 Copper

C-3-4 Bottom Layer

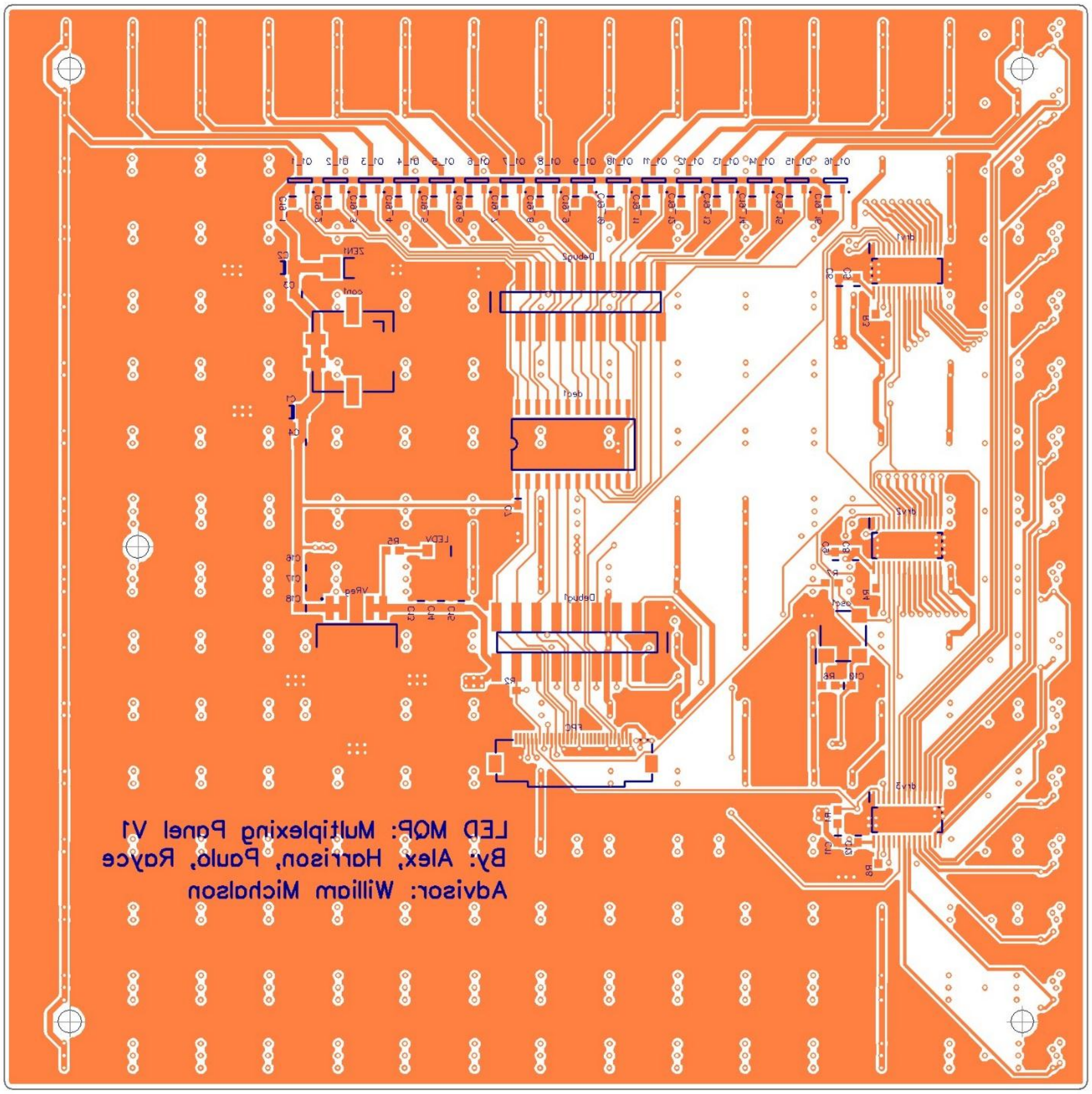


Figure C - 7: Bottom Layer Copper and Silksscreen

C-4 Component List

Table C - 6: Multiplexing LED Panel BOM

RefDes	Type	Value	Description
C1, C2	Cap C 1206	47 uF	Main Decoupling Capacitor
C3, C4	Cap C 0603	100 nF	Main Decoupling Capacitor
C5	Cap C 0603	100 nF	
C5, C9, C11	Cap C 0603	1 uF	Driver Decoupling Capacitor (Outer)
C6, C8, C12	Cap C 0603	100 nF	Driver Decoupling Capacitor (Inner)
C7	Cap C 0603	100 nF	Decoder Decoupling Capacitor
C10	Cap C 0603	100 nF	Crystal Decoupling Capacitor
C13	Cap C 0603	10 uF	Regulator Decoupling Capacitor
C14 – C17	Cap C 0603	100 nF	Regulator Decoupling Capacitor
C18	Cap C 0603	10 uF	Regulator Decoupling Capacitor
C19_n	Cap C 0603	2.2 uF	MOSFET Decoupling Capacitor
Con1	MOLEX 4POS	3-794638-4	4 Position Power Molex Connector
Debug1, Debug2	Header 2X9	961218-6300-AR-PR	Debugging Headers (Saleae Compatible)
Dec1	Decoder	CD74HC154M	16 Channel Decoder
Drv1-Drv3	LED Driver	TLC5940	16 Channel PWM LED Driver
FPC	Connector	52893-3095	30 Position FPC Connector
LEDV	LED 0805	--	Power On Indicator LED
Osc1	Oscillator	CB3LV-3I-24M0000	Oscillator for LED Driver GSCLK
Q1_n	MOSFET P	SI2301CDS	Line Selector MOSFET
R1	Res 0603	100 KOHM	Blank Pull-Up Resistor
R3, R4, R8	Res 0603	2 KOHM 1%	Current Reference Resistor for LED Driver
R2	Res 0603	100 KOHM	XLAT Pull Down
R5	Res 0603	100 OHM	LEDV Current Limiter
R6	Res 0603	100 KOHM	Oscillator Power On
R7	Res 0603	0 OHM or NC	If Soldered, GSCLK Comes from FPC; Else, Oscillator
Rgb(n)	RGB LED	PLCC6 LED	RGB LEDs in Front of Panel
vreg	V Regulator 3V3	LT1963EQ	3V3 Voltage Regulator
ZEN1	Zener 5V1	SMBJ5338B-TP	5V1 Protection Zener Diode

Appendix D: Direct-Drive LED Panel Reference Manual

The following are contents taken directly from the Direct-Drive LED Reference Manual, created to aid in the software development for the panel and as a resource for future development.

D-1 Component/Connector Pin-Out and Description

D-1-1 FPC Connector

This connector is used for connecting the direct drive panel to the logic boards. Both power and data lines are made available and are specified in Table 1.

Table D - 1: Direct-Drive LED Panel FFC Connector Pin-out

Pin	Connected To	Description
1:GND	GND	Power Supply
2:3V3	3V3	Power Supply
3:3V3	3V3	Power Supply
4:GND	GND	Power Supply
7:GND	GND	Power Supply
14:GND	GND	Power Supply
15:BLANK4	CON2_8, CON2_7	LED Driver Blank (Active LOW)
16:BLANK3	CON2_6, CON2_5	LED Driver Blank (Active LOW)
17:BLANK2	CON2_4, CON2_3	LED Driver Blank (Active LOW)
18:BLANK1	CON2_2, CON2_1	LED Driver Blank (Active LOW)
19:GND	GND	Power Supply
20:VPRG	CON2_n VPRG	Depends on daughter board
21:XLAT	CON2_n XLAT	Depends on daughter board
22:DCPRG	CON2_n DCPRG	Depends on daughter board
26:Ser0to1	CON2_1 Serial In	Main Serial In
27:SCK	CON2_n SCK	Serial Clock Line (All)
28:GND	GND	Power Supply

Note that pins that are not mentioned in table 1 are not connected and are reserved for future expansion.

D-1-2 Daughter Board Connectors CON2_n

These 1mm pitch connectors attach the daughter boards to the direct drive LED panel. This connector specifically serves the data lines and ground.

Table D - 2: CON2_n Headers Pin-out

Pin	Connected To	Description
3:Sout	Con2_(n+1)	Serial Line to Next Panel
4:VPRG	FPC20	Depends on daughter board
5:DCPRG	FPC22	Depends on daughter board
6:XLAT	FPC21	Depends on daughter board
7:BLANK	FPC(15|16|17|18)	Depends on daughter board
8:Sin	FPC26 or CON2_(n-1)	Serial In
(9,11-16):GND	GND	Power Supply
10:SCK	FPC27	Serial Clock Line (All)

Note that pins that are not mentioned in table 1 are not connected and are reserved for future expansion.

D-2 Schematics and Description

D-2-1 High Level Diagram

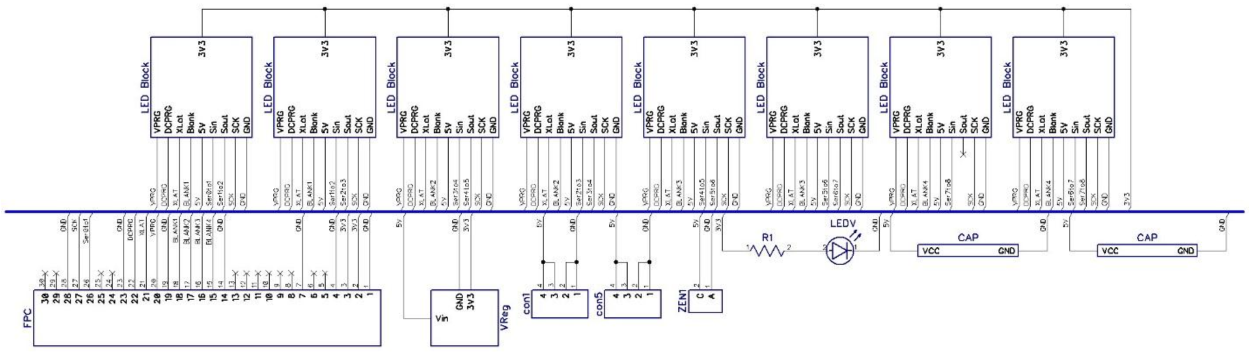


Figure D - 1: Direct-Drive LED Panel High-Level Schematic

D-2-2 Voltage Regulator

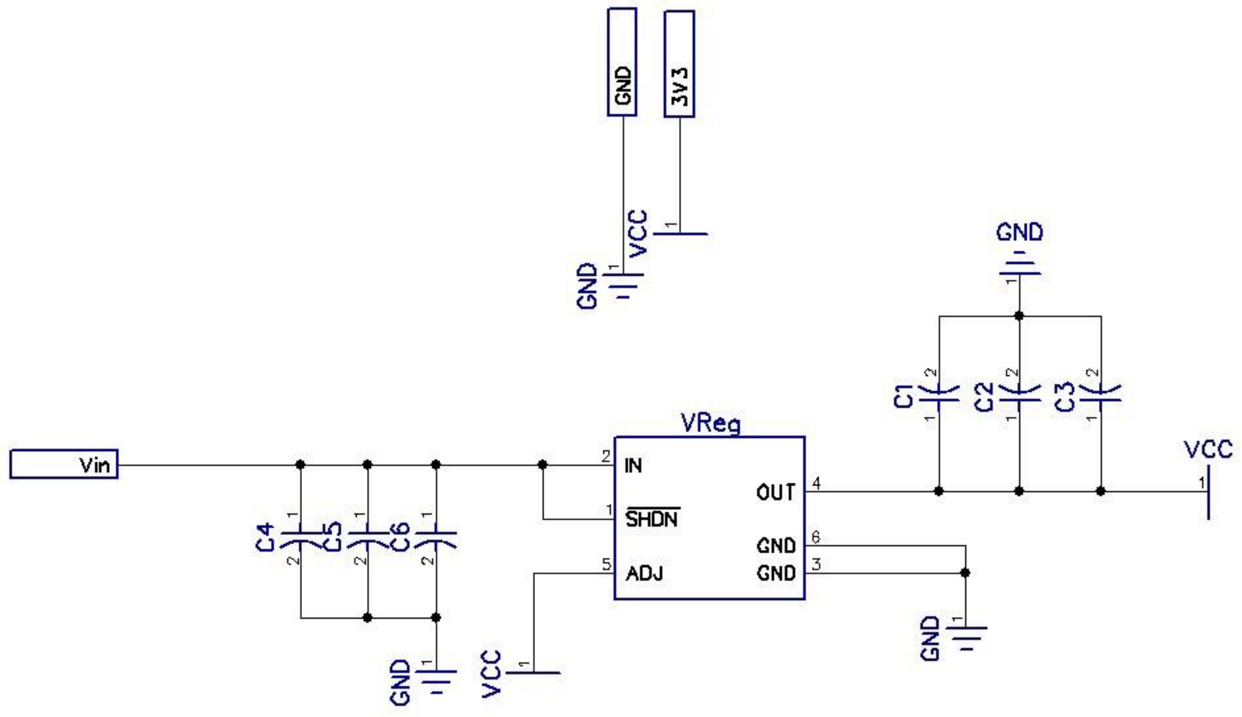


Figure D - 2: Direct-Drive LED Panel Voltage Regulator Schematic

D-2-3 Capacitor Block

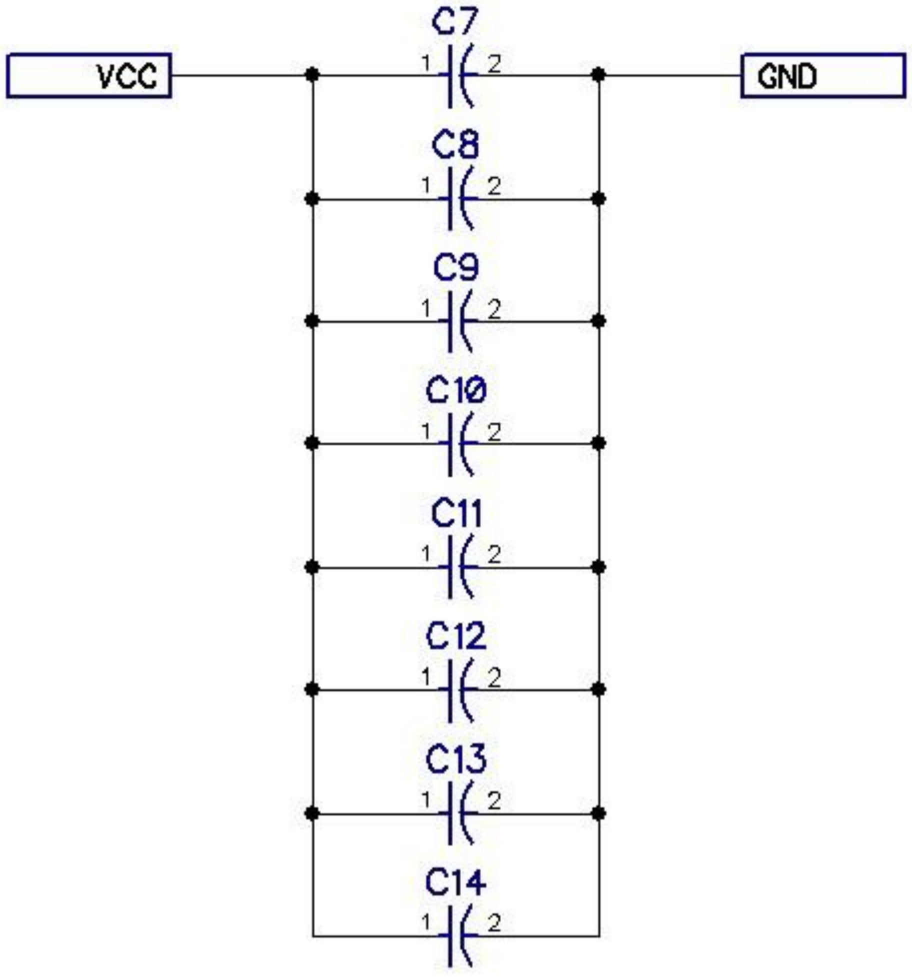


Figure D - 3: Direct-Drive LED Panel Decoupling Capacitor Array Schematic

D-2-4 LED Block

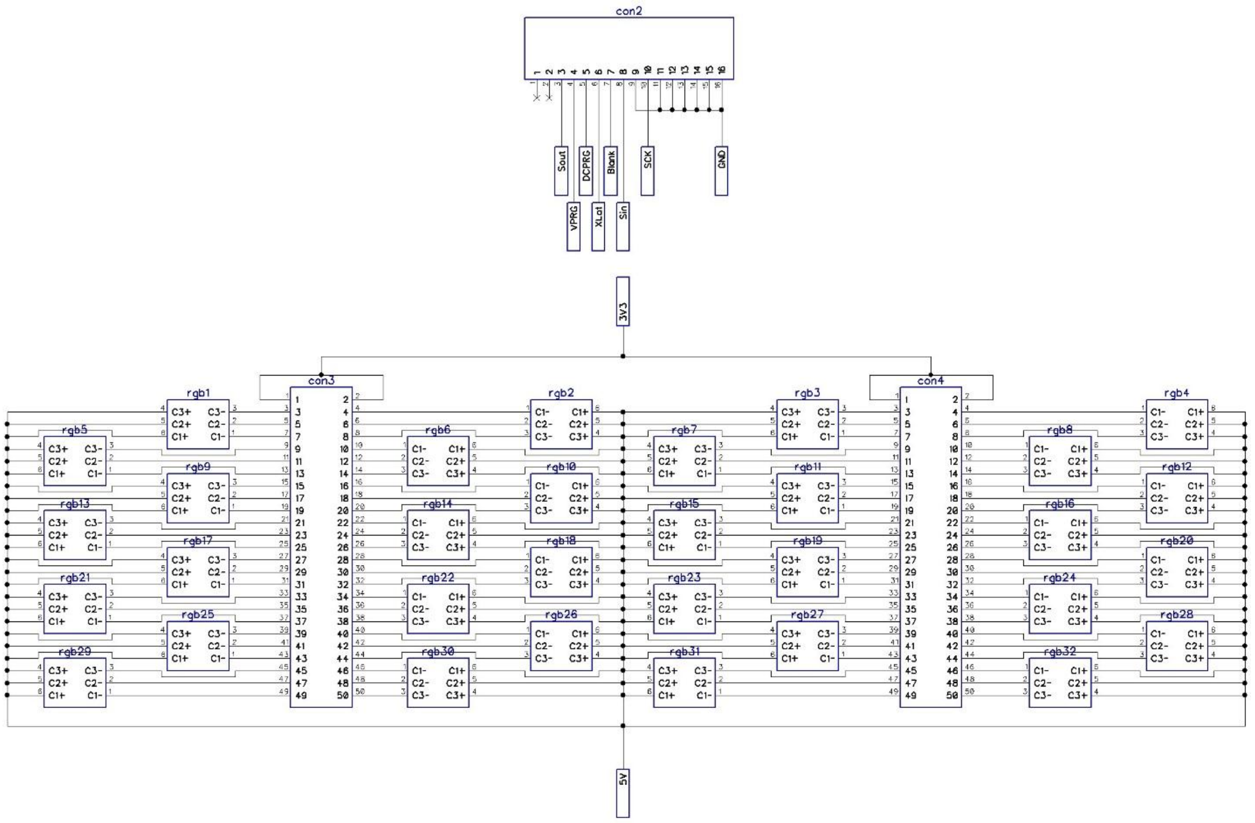


Figure D - 4: Direct-Drive LED Panel LED Block Schematic

D-3 Copper Artwork

D-3-1 Top Layer

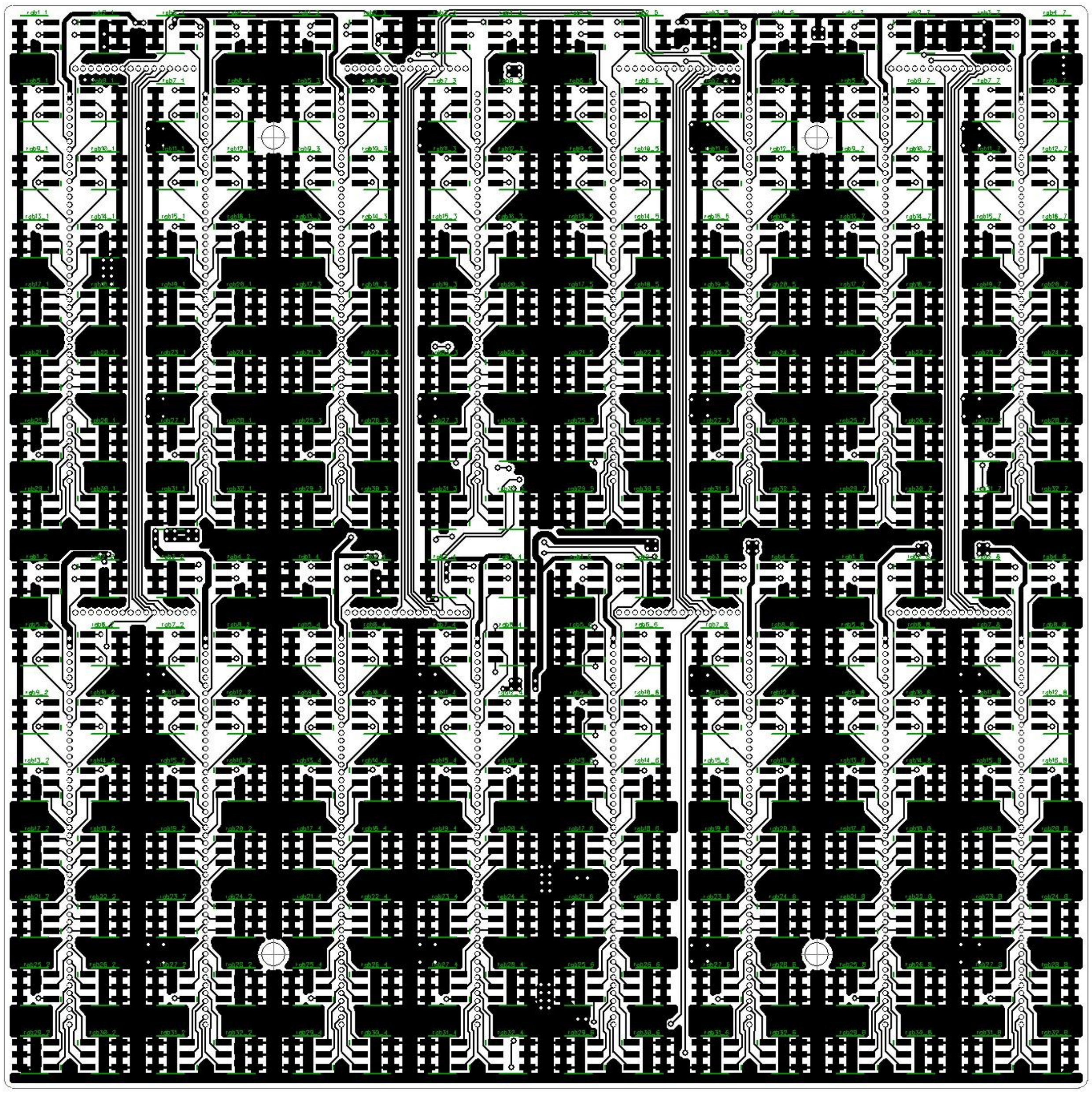


Figure D - 5: Direct-Drive LED Panel Top Layer Copper and Silkscreen

D-3-2 Bottom Layer

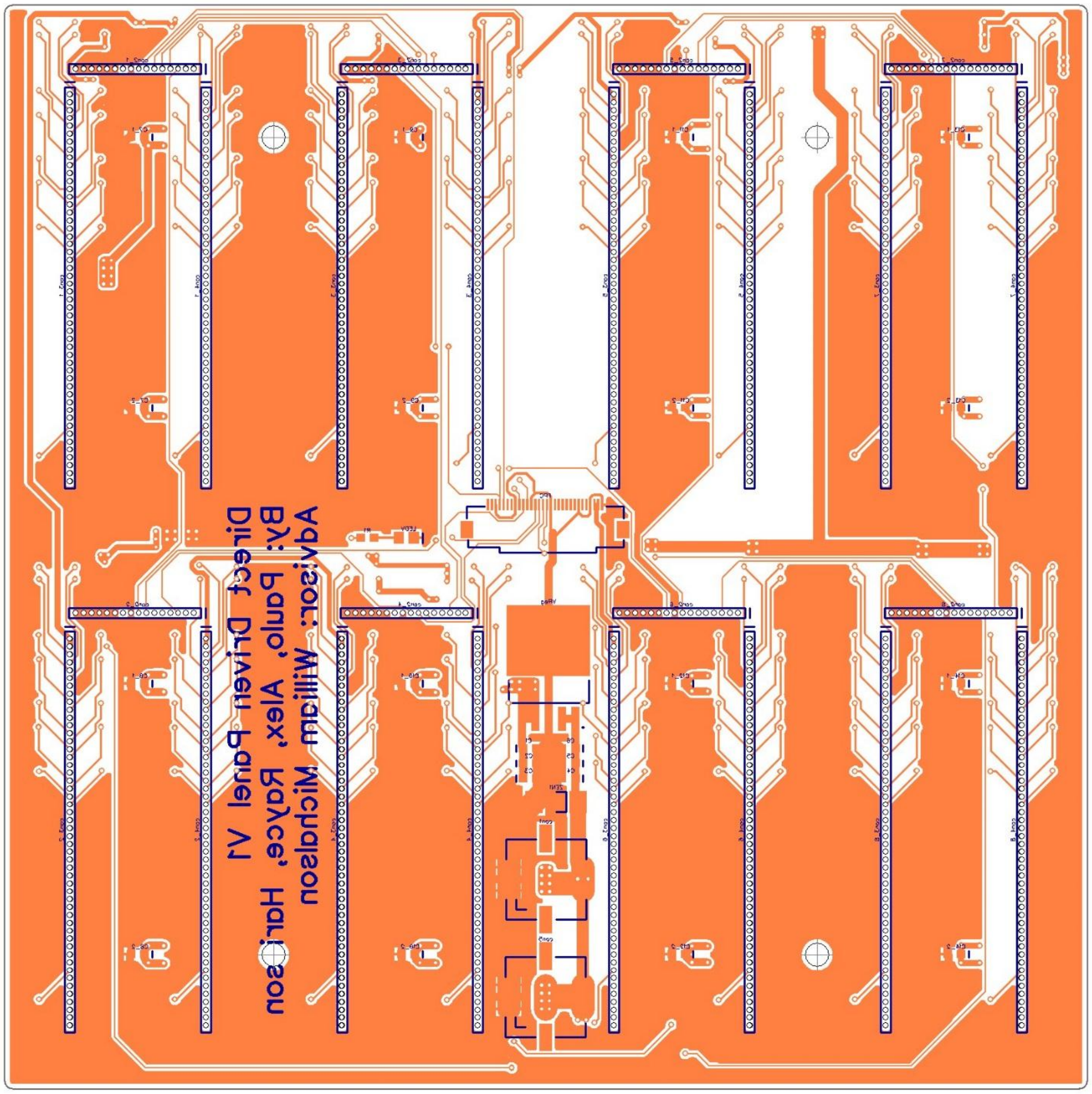


Figure D - 6: Direct-Drive LED Panel Bottom Layer Copper and Silkscreen

D-4 Component List

Table D - 3: Direct-Drive LED Panel BOM

RefDes	Type	Value	Description
C1,C2,C5,C6	CAP C 0603	100 nF	Main Decoupling Capacitor
C3,C4	CAP C 0603	47 uF	Main Decoupling Capacitor
C7_n-C14_n	CAP C 0603	10 uF	LED Decoupling Capacitor
Con1,Con5	MOLEX 4POS	3-794638-4	4 Position Molex Connector
Con2_n	CON 16POS	SMH100-LPSE-S20-ST-BK	Header for Connecting Daughter Board
Con3_n	CON 50POS	SMH100-LPSE-S20-ST-BK	Header for Connecting Daughter Board
Con4_n	CON 50POS	SMH100-LPSE-S20-ST-BK	Header for Connecting Daughter Board
LEDV	LED 0805	SMD LED	Power On Indicator
R1	Res 0603	100 OHM	LED Current Limiter
Zen1	Zener 5V6	SMBJ5338B-TP	Over Voltage Protection
Vreg	V Reg	LT1963EQ	3V3 Voltage Regulator

Appendix E: Direct Drive Daughterboard Reference Documents

The following are raw documents taken from the manual for the direct-drive daughterboard. They were intended to be used as a programming aid – the figure and table numbers have been update.

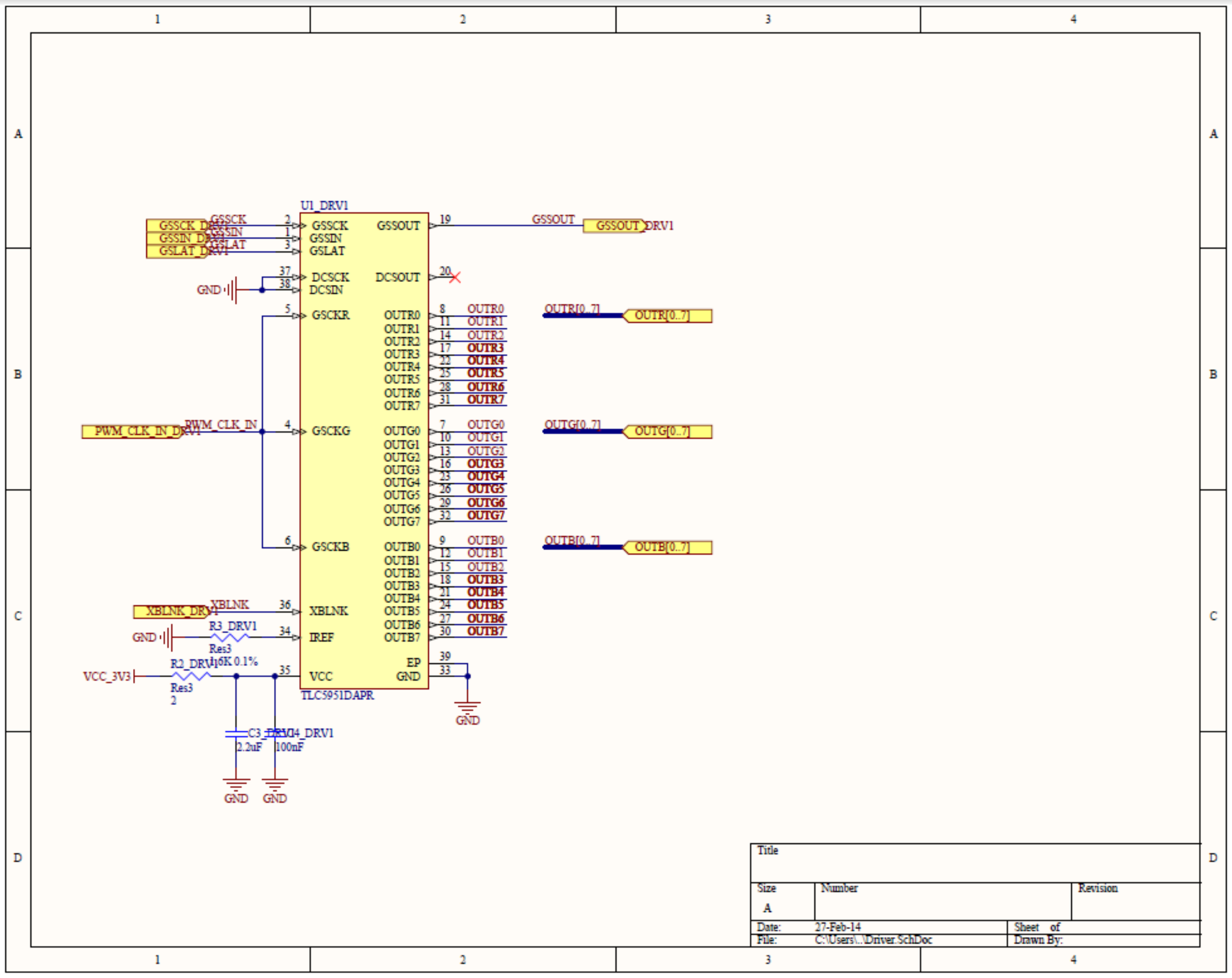


Figure E - 1: Direct Drive Daughterboard Driver Connection Schematic

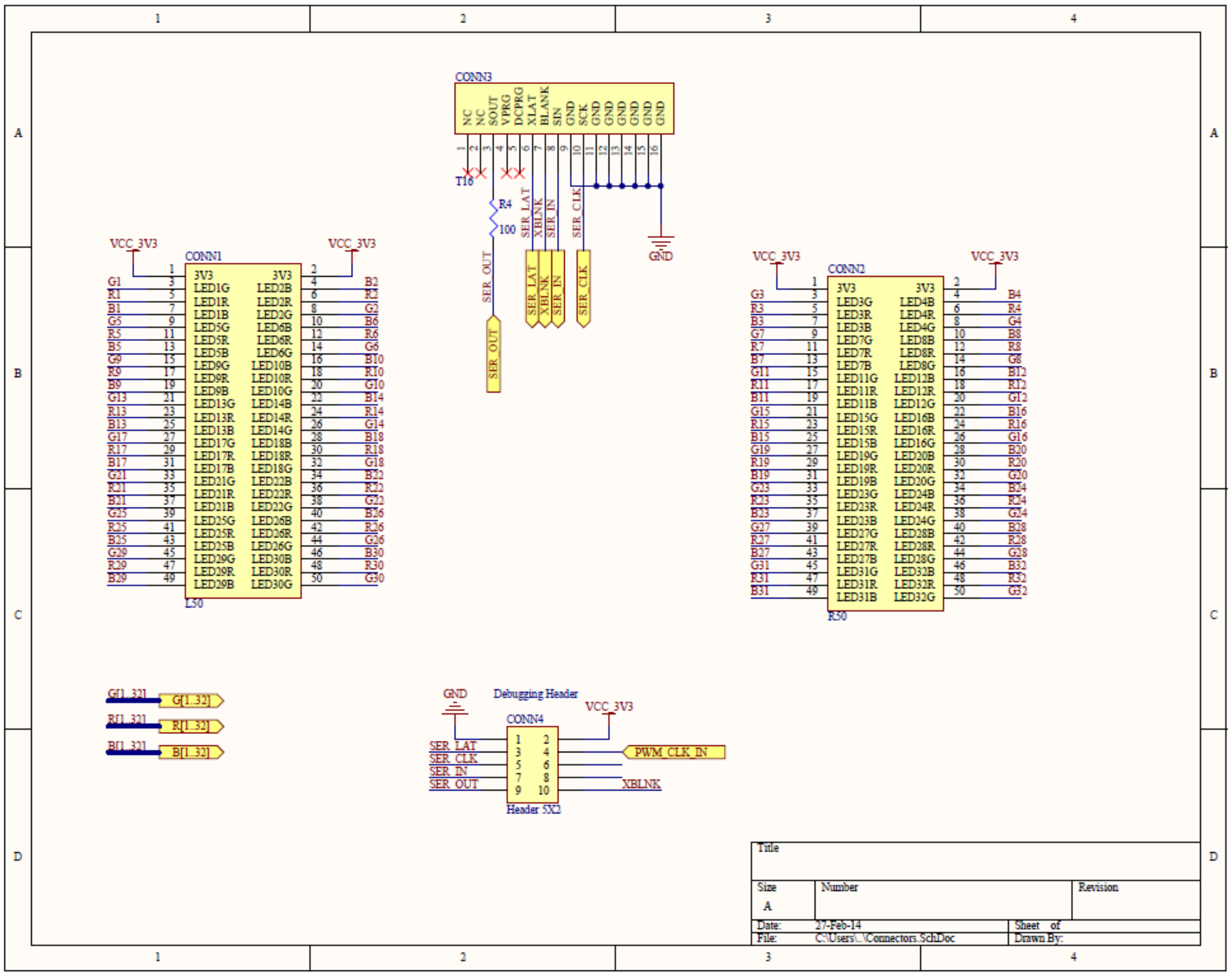


Figure E - 2: Direct Drive Daughterboard Pin Connections Schematic

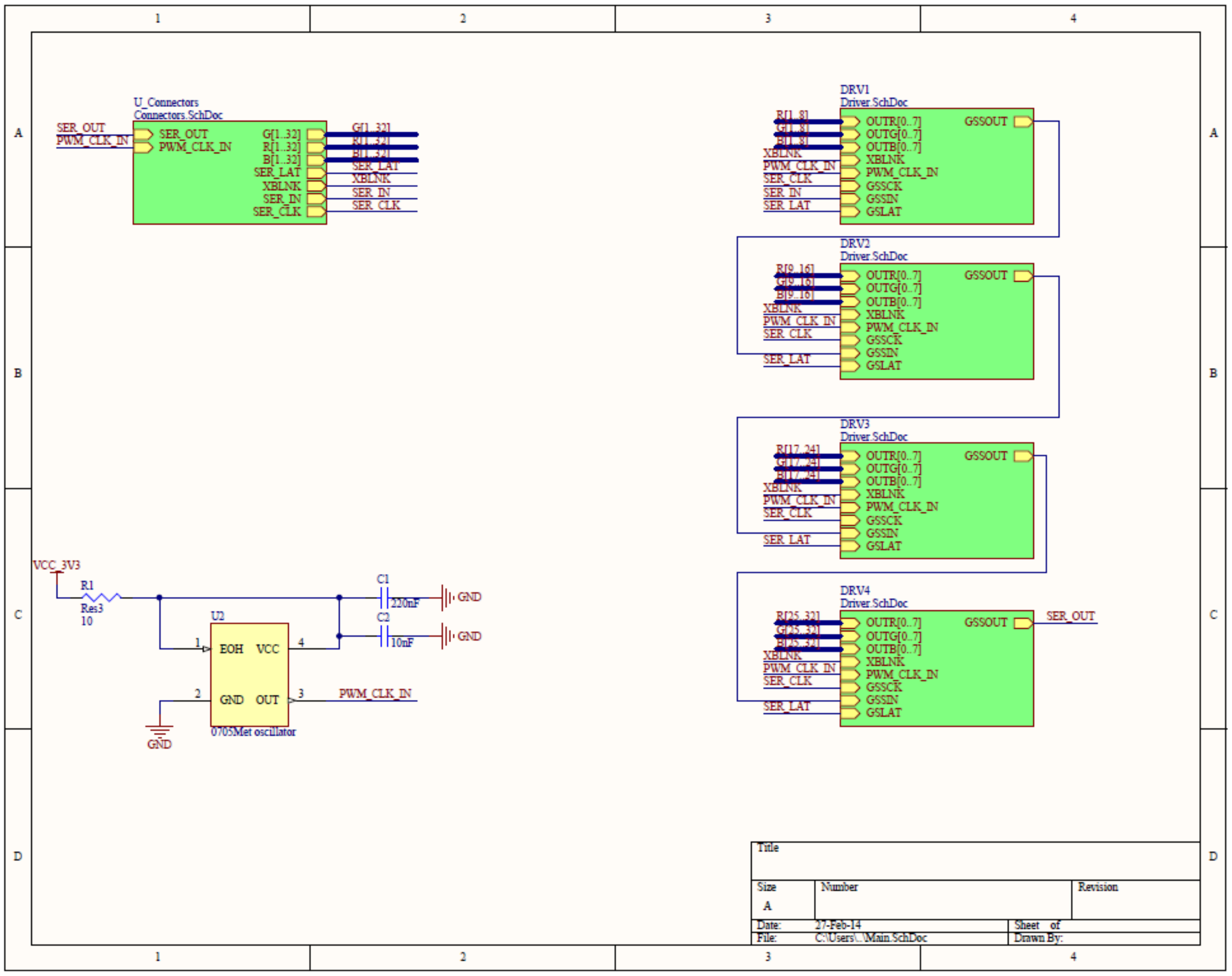


Figure E - 3: Direct Drive Daughterboard Main Overview Schematic

E-2 PCB Layer Prints

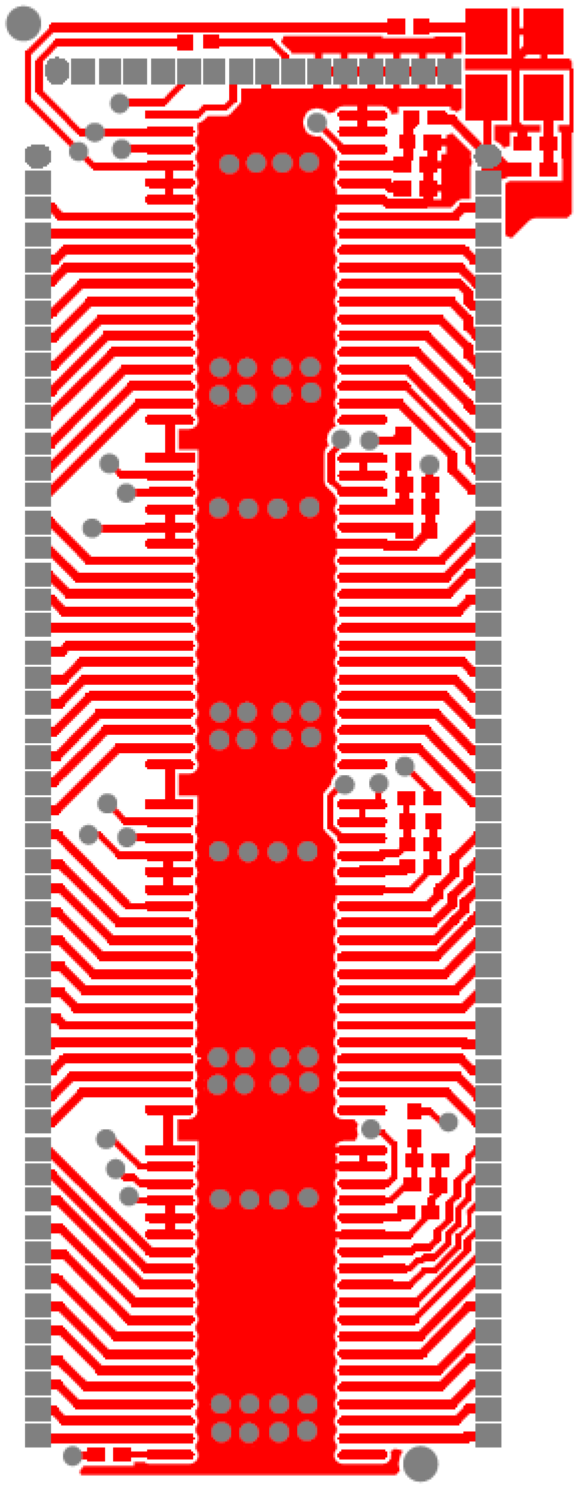


Figure E - 4: Direct Drive Daughterboard Top Copper

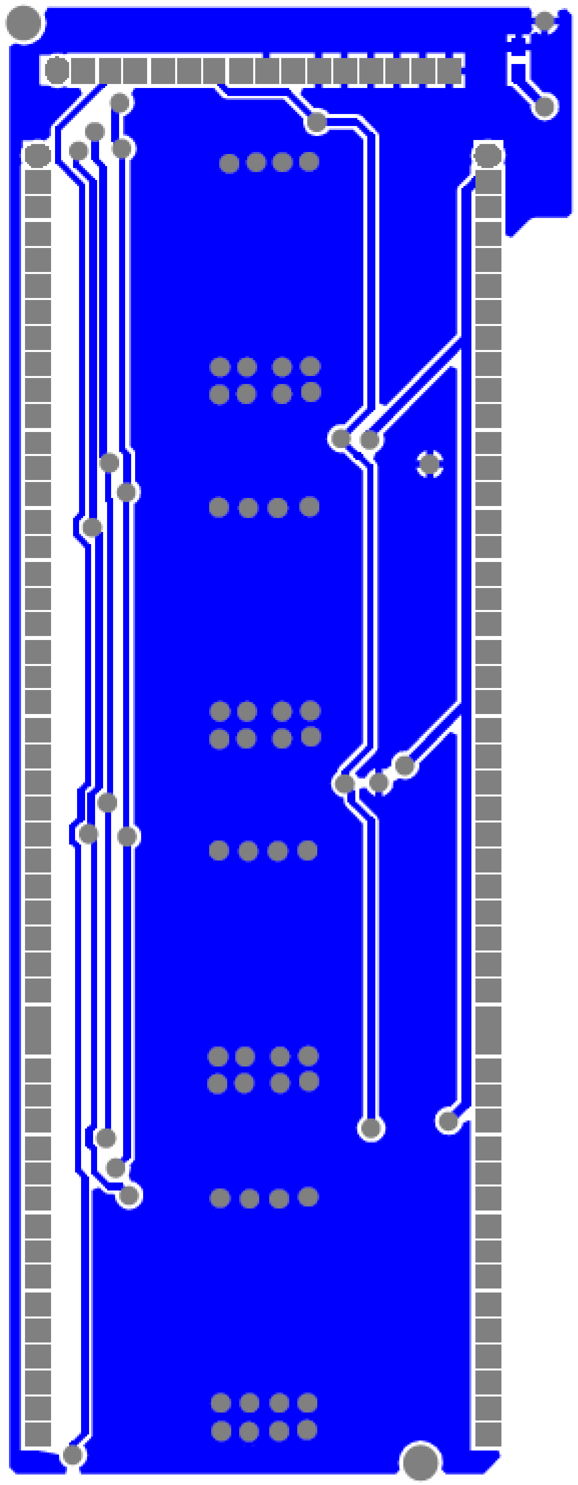


Figure E - 5: Direct Drive Daughterboard Bottom Copper

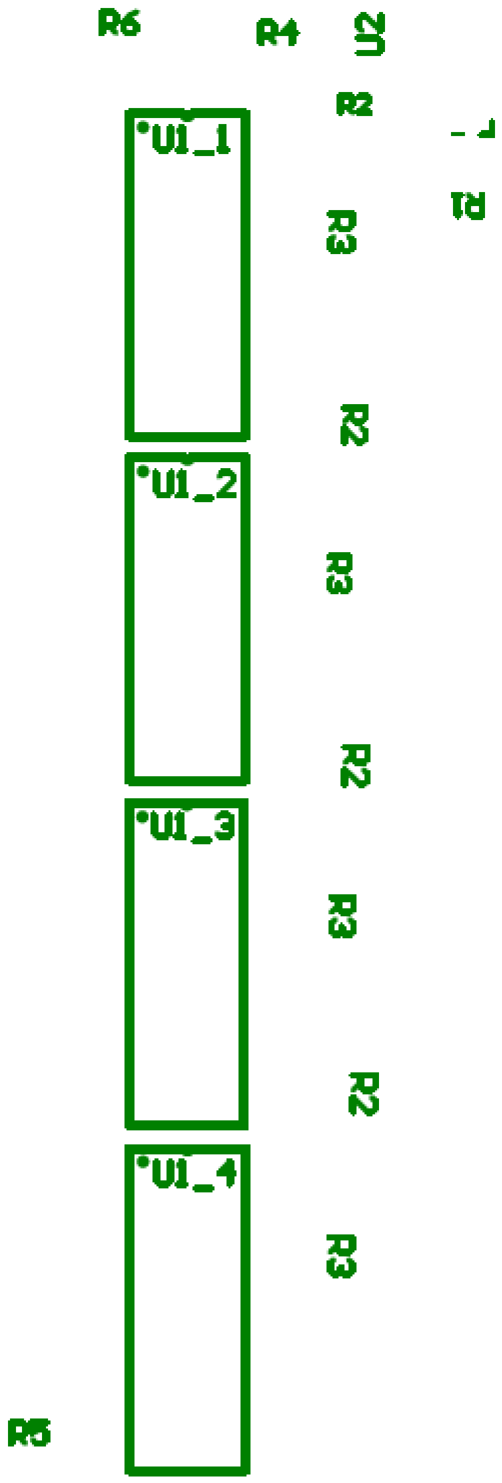


Figure E - 6: Direct Drive Daughterboard Top Silkscreen

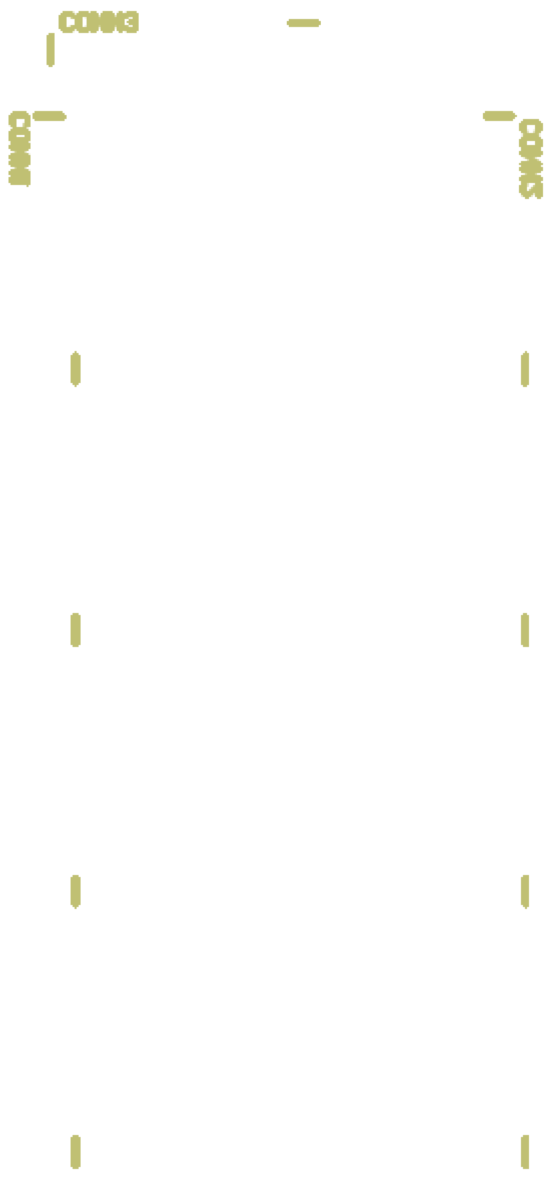


Figure E - 7: Direct Drive Daughterboard Bottom Silkscreen

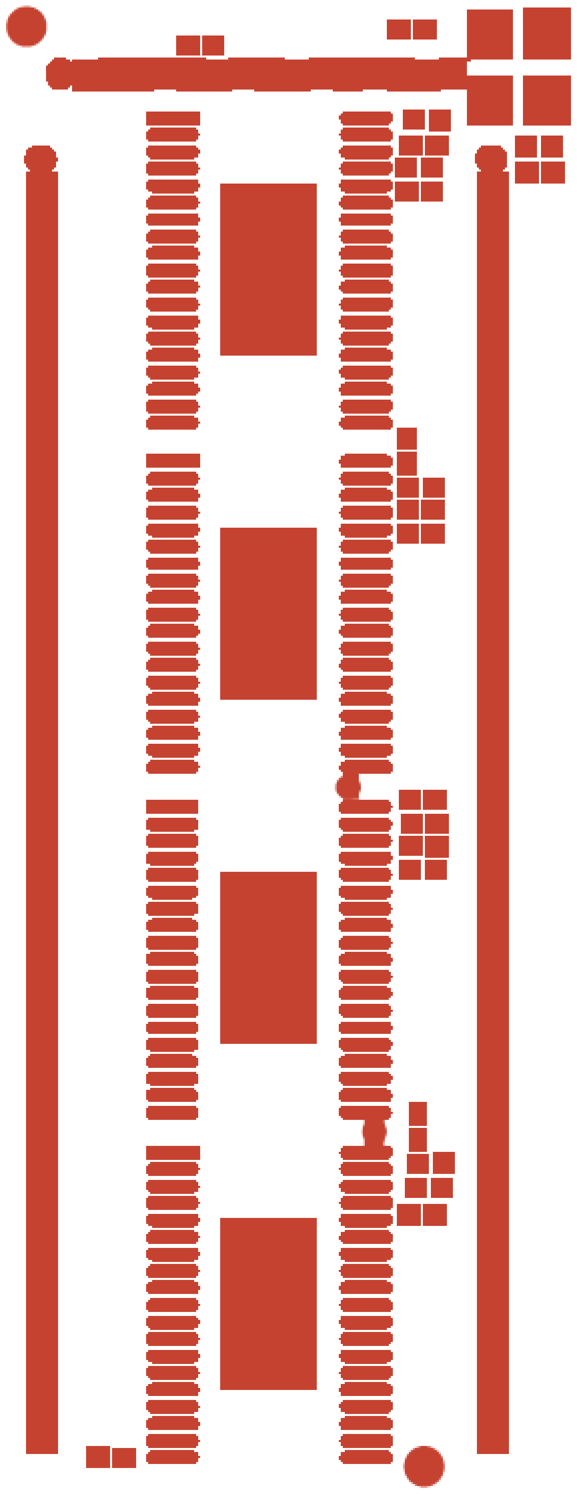


Figure E - 8: Direct Drive Daughterboard Top Soldermask

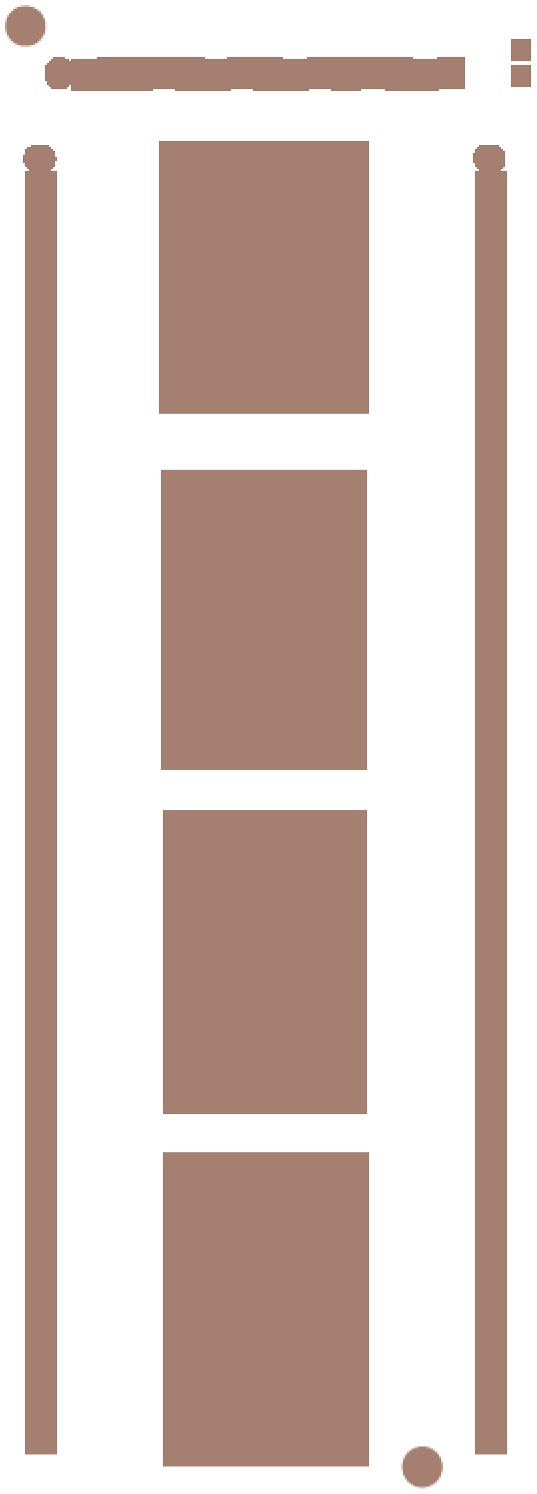


Figure E - 9: Direct Drive Daughterboard Bottom Soldermask

E-3 Daughterboard Render

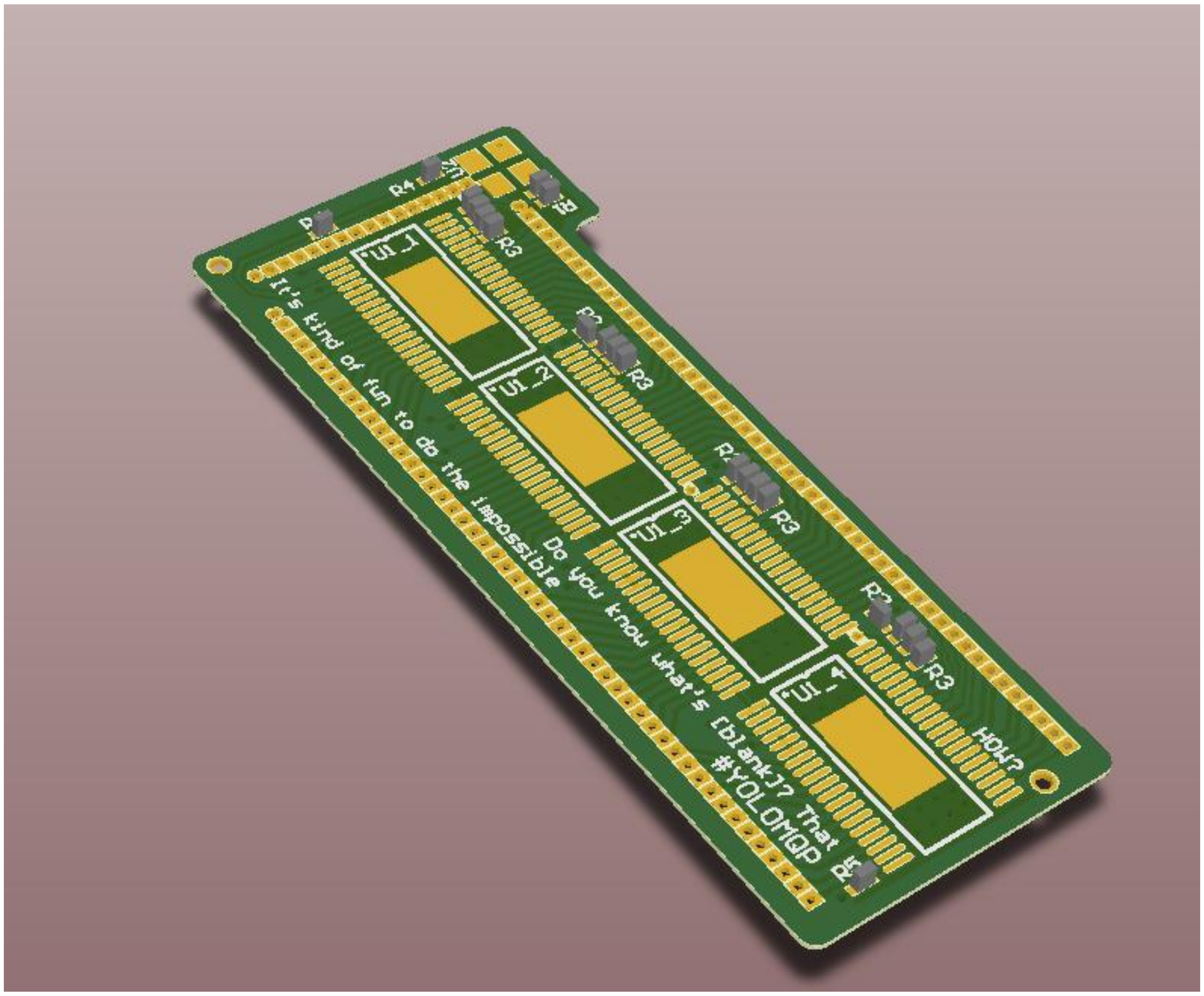

U1_1
U1_2
U1_3
U1_4
R3
R4
It's kind of fun to do the impossible
Do you know what's [blank]? That
#YOLOMQP
HOW?

Figure E - 10: Daughterboard Altium 3D Render

E-4 Component List

Table E - 1: Component List of Direct Drive Daughterboard

Part	Designations	Package	Quantity	Comments
4.2K Res	R3	0402	4	Full-Scale Current Set
0.1uF 25V	C1, C3	0402	5	Bypass Caps
1uF 25V	C2, C4	0402	5	Bypass Caps
900mOhm 120Ohm@100MHz Ferrite Chip	R2	0402	5	Noise Reduction to Drivers and Oscillator
TLC5951	U1_X	HTSSOP-38	4	LED Drivers
6-MHz Oscillator (Generic)	U2	3225	1	PWM Clock Generation
1-mm Pitch Headers, 50-pin	CONN1, CONN2	50x1-mm	2	LED Contacts and Vcc to LED Drivers
1-mm Pitch Headers, 16-pin	CONN3	16x1-mm	1	Control Signals