100 Hz 512x512 SLEDS SYSTEM DESIGN

by

Kassem Nabha

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Master of Science in Electrical and Computer Engineering

Summer 2014

100 Hz 512x512 SLEDS SYSTEM DESIGN

by

Kassem Nabha

Approved: _____

Fouad E. Kiamilev, Ph.D.
Professor in charge of thesis on behalf of the Advisory Committee

Approved: _____

Kenneth E. Barner, Ph.D.
Chair of the Department of Electrical and Computer Engineering

Approved: _____

Babatunde A. Ogunnaike, Ph.D.
Dean of the College of Engineering

Approved: _____

James G. Richards, Ph.D.
Vice Provost for Graduate and Professional Education

ACKNOWLEDGMENTS

I would first like to thank God for giving me the chance and providing me with the knowledge and ability to be here today. Without his help, this would not have been possible.

Next I am truly grateful for everything my advisor Dr. Fouad Kiamilev has done for me. He has become my friend as well as an advisor and thanks to his belief in me, my experience has been amazing. Without his guidance and constant encouragement, I would not be as successful as I am today.

Rodney McGee and Nicholas Waite are two very knowledgeable individuals that have been a substantial asset to CVORG and my learning experience. Rodney McGee is one of the best engineers and friends I have ever had the privilege of working with. With his guidance and knowledge, I've learned new ways to think about issues and solve problems.

When it comes to design, Nick immediately comes to mind, as he is the mastermind behind this highly complex computer architecture. With his outstanding skills in design and programming, he has taught me a great deal on how to improve my code and my thought process of how to develop new things.

My experiences in CVORG and the University of Delaware during the years of my undergraduate and masters programs would not have been the same without other lab members. They have all at times helped me with SLEDs and made my academic experience all the better. I would like to extend a personal thank you to: Hamzah Ahmed, Jacob Benedict, Tyler Browning, Stephen Cockerill, Jon Dickason, Garrett Ejzak, Miguel Hernandez, Josh Marks, Michelle Pettinella, and Ron Reisor.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Infrared (IR) detectors applications are widely used across fields from scientific and military to medical and industrial. IR can detect information that human eyes cannot perceive allowing the advancement in science and technology.

Due to the importance of the infrared detectors in today's world, it is necessary to accurately test and characterize these detectors with a frame of reference related to the application. IR projection systems are a great way to characterize the detectors because they can be used as high accuracy reference.

Described in this work is a 512x512 super-latticed light emitting diode system (SLEDS) operating at a frame rate of 100Hz. This system has been fully developed, tested, and corrected for non-uniformity (NUC). Further work will be done to achieve higher frame rates and two frequencies of emission (colors) instead of one.

Chapter 1

INTRODUCTION

1.1 SLEDS Background

Infrared (IR) detectors have applications across fields from scientific and military to medical and industrial. IR can detect information that human eyes cannot perceive allowing the advancement in science and technology. Some applications in which IR imaging is used are finding stars millions of light-years away, finding shorts in electric circuits, finding gas leaks, and thermal night vision.

In order to test IR detectors, it is necessary to have testing equipment that is able to project realistic IR images at a large range of intensities. Existing technologies use methods such as blackbody radiation and resistive arrays to test IR detectors. The super-latticed light emitting diode system (SLEDS) is a new IR projector technology that is able to project images at a much higher level of precision and at much higher intensities than projectors that are currently on the market. This is achieved by using a 512x512 array of light emitting diodes (LED's) that is driven by various electrical components that are run by an FPGA. With a 512x512 resolution, images projected by SLEDS are exceedingly accurate portrayals of the input image. The LED's allow the projector to run at significantly higher intensities than the resistor arrays without damaging the system, by just supplying a relatively current to the LED's. The system

is able to take in any DVI input image and output that image on the SLEDS array. The user is also able to use a software interface to program the projector to display desired images.

This work is a detailed description of the SLEDS project, in which a 512x512 array of infrared light emitting diodes operates at a frequency of 100Hz. This system has been fully developed, tested, and the LED pixels have been corrected for non-uniformity. There are a number of components in this system, which together form the fully developed projector. Each of these components will be discussed, as well as an explanation as to how they interact with one another to run the system. A goal of SLEDS is to reach a frame rate of 1 kHz and two frequencies of emission (colors). This paper will also give details regarding the future of SLEDS and how these goals will be obtained.

1.2 Motivations

State-of-the-art Infrared Scene projectors are based on Thermal Pixel Arrays (TPA) provide apparent temperatures in the mid-wave infrared up to only ~ 400°C at maximum frame rates of 200Hz. The goals of this project are to move beyond the constraints of TPAs using IR light-emitting diode (LED) technology. LEDs are inherently fast, relying on electronic transitions between energy bands in a semiconductor, rather than thermally generated quasi-blackbody radiation from TPAs. Frame rates of the LED arrays are effectively limited only by the read-in integrated circuit (RIIC) and drive electronics. The electronic nature of IR LED emission allows

the IR LED to be significantly more efficient than emission from TPAs [3]. The results are far higher apparent temperatures for IR LEDs. LEDs also exhibit narrower bandwidth emission compared to blackbody radiation. However, by fabricating an emitter array from two or more distinct LEDs in each pixel, an LED array can emulate subtleties of a broader band or multiband source[5].

The current fully developed and usable system is the result of many years of SLEDS development. This previous work includes the development of a 64x64-pixel array described in Rodney McGee master's thesis [1].

The current system is built on lessons learned from previous systems and the need for new features. Completely redesigned drive electronics have been developed and multiple versions of code and hardware have been implemented to reach a fully operating system. In the past year the SLEDs system has undergone numerous new developments and is continuing to evolve. Some of these designs will be discussed later in this paper.

Chapter 2

SYSTEM OVERVIEW

2.1 3x3 System Example

This section will explain, in a simple fashion, how the major components of how SLEDS work together to form a complete system. Starting with the read-in integrated circuit (RIIC) and the LEDs, and ending with the main driver; an overview of the system will be given.

Each pixel in the SLEDS array is a light emitting diode. As with light emitting diodes, the more voltage across it the brighter it gets. The RIIC is simply many pixels grouped together. For simplicity, this explanation will discuss a 3x3 pixel RIIC. The concepts discussed in this example remain valid for any size array. To turn a single pixel on, the pixel must be addressed with the appropriate voltage to yield the desired amount of emitted light.

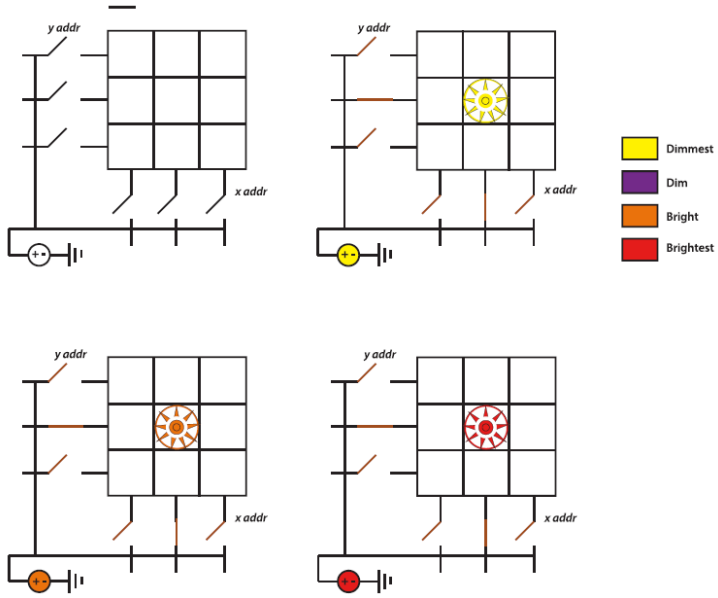

Figure 2.1 : 3x3 pixel RIIC with addressing and voltage in

Figure 2.1 illustrates how to operate a single pixel at different brightness levels. Each pixel has an x and y address corresponding to its position within the RIIC. As seen in Figure 2.1, once the x and y addressing switches are closed, then the appropriate voltage to produce the desired LED intensity is supplied to the pixel; the higher the voltage the brighter the LED.

The varying voltage levels that change the brightness of the LED are supplied by an amplifier which converts an analog current signal to a voltage signal at a desired range; this current signal changes as the digital input of the digital to analog converter (DAC) changes. This process allows the brightness of the pixels to be controlled digitally. Figure 2.2 below shows how changing the digital signal on the input side of the DAC changes the voltage sent to the LED. When a digital input of zero is supplied

to the DAC, zero volts are sent to the pixel, causing the LED to turn off. When a digital input of all ones (the maximum binary value) is supplied to the DAC, the maximum voltage is sent to the pixel, causing the LED to shine with its maximum intensity. As the digital input is incremented from zero to all ones, the voltage output is also incremented linearly from zero to maximum voltage.

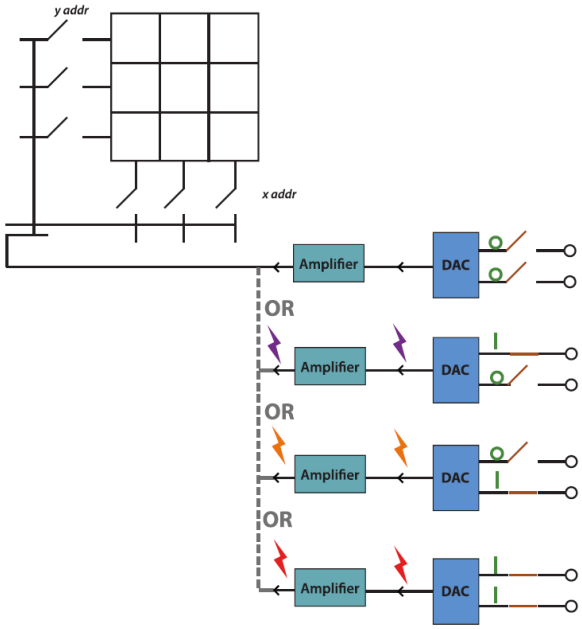

Figure 2.2 : DAC, amplifier, and the RIIC with a changing digital input

The question now becomes where this binary signal and the address of the pixel come from. The answer is a Xilinx ML605 FPGA board, which can control the addressing and digital signals with a hardware description language (HDL). The DAC, amplifier, and RIIC are connected to the output of the FPGA board using a SLEDS interface board; the board is used as link between all the components in the system.

The ML605 is equipped with Microblaze, which allows a user to program it using the C programming language. By using C, the user can program exactly which pixel to turn on and at what intensity. Figure 2.3 below graphically shows the connection between the FPGA board, the DAC, the amplifier, and the RIIC.

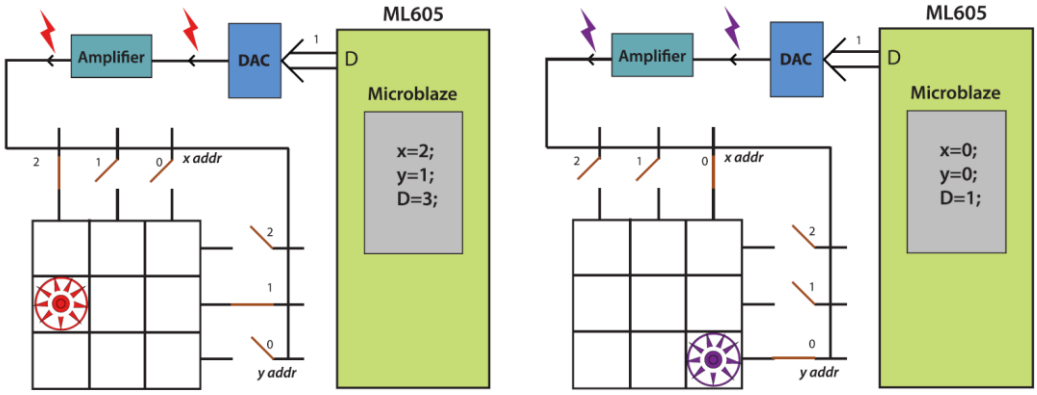

Figure 2.3 : ML605 driving the system

Another way to address the pixels, rather than writing C code to program the FPGA, is through a DVI input to the FPGA. The user simply needs to input the desired image to be projected through to the FPGA board, and then the image will be projected by the array of pixels. A simple example of the DVI mode setup is shown in Figure 2.4 below.

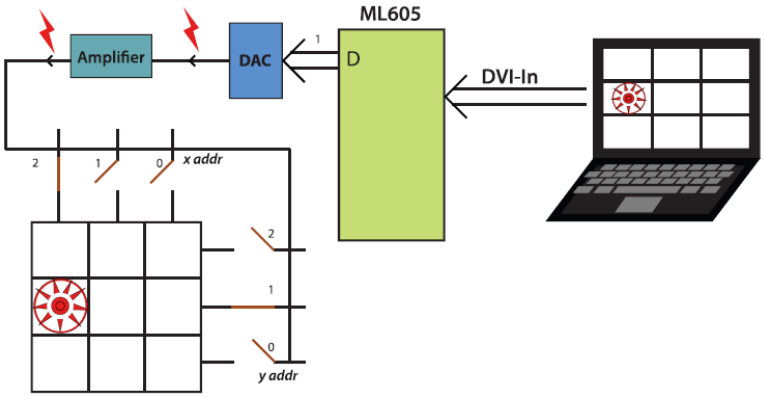

Figure 2.4 : DVI input to ML605 than the SLEDS system

Chapter 3

SYSTEM COMPONENTS

3.1 Pixel

Each LED consists of 10V CMOS pass transistor pairs, which are used to pass the analog input to the drive transistor as shown in Figure 3.1 below.

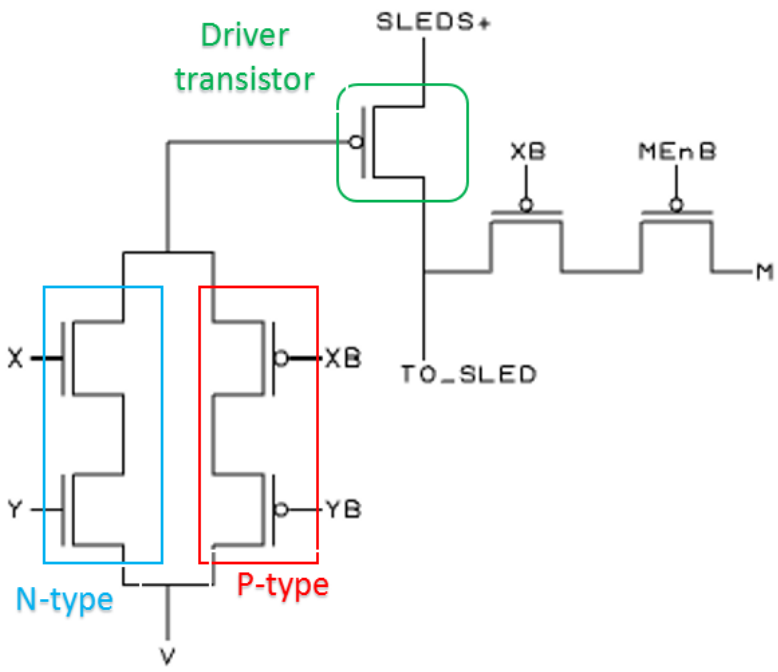

Figure 3.1 : Single pixel schematic

A pass transistor pair (both N_type and a P_type MOSFET in parallel) are used to pass both low and high voltages, as N_type MOSFET's will only pass low voltages and the P_type MOSFET will only pass high voltages. For example to turn a pixel on

at high light intensity we turn on both X and Y of that pixel and to turn the pixel on at low light intensity we turn on both XB and YB of the pixel. The other two transistors XB and MEnB allow the LED to be tested.

3.2 RIIC

The 512 x 512 driver RIIC for the single-color array is a complementary metal-oxide-semiconductor (CMOS) integrated chip that simultaneously drives each emitter with an externally-controllable constant current. This driver was fabricated using On Semiconductor's C5N process [7]; a three metal, 0.5μm CMOS process with thick oxide, and lateral high-voltage transistors. Each individual LED driver circuit occupies an area of 48 x 48 μm^2, matching the SLEDS array, and connects to the LED anode and cathode terminals using 15μm high contact pads to minimize thermal and electrical impedance between the SLEDS and RIIC [4]. Each driver cell contains a CMOS circuit that stores an analog voltage and converts it to a current at its bonded emitter, and is capable of driving more than 20mA of current with a 10V swing. The drive characteristics were determined from individual SLEDS device testing. The layout of the entire RIIC and a single RIIC pixel are shown in Figure 3.2.

The RIIC driver chip uses a reticle-stitching technique to split the 512x512 array into four 256x256 sub arrays that are aligned and stitched together to form a complete chip because the maximum reticle size permitted is about 2.1cm x 2.1cm [8]. The resultant size of each 512x512 RIIC is 3.3x3.3 cm^2. The periphery of each RIIC

driver chip is occupied by independent address decoding and low-to-high voltage translational circuitry. The RIICs were processed on 8” silicon wafers with 19 complete RIICs per wafer

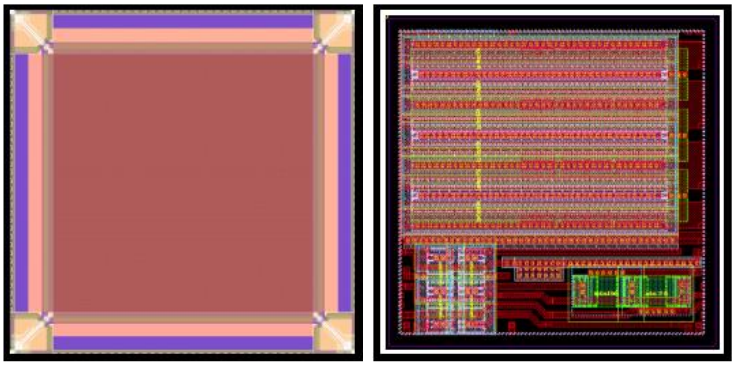

Figure 3.2 : Overall layout of the 512x512 RIIC (left) and layout of a single pixel.

The 512x512 RIIC is mounted to a PCB board as shown in Figure 3.3. The board has 20 digital lines and 4 analog lines other than the power lines. Those are composed of 16 bit addressing, a 4 bit pre-charge, and 4 analog channels [6]. Due to the RIIC being 512x512, we would have needed 36-bit address lines; therefore, we introduced four pre-charge lines that would control each quadrant independently and decrease the address lines to 16-bits. For a pixel to turn on both its address lines and pre-charge line has to be high. Since the system is split into four quadrants, each quadrant shares the same address lines, producing a four pixel write at the same time.

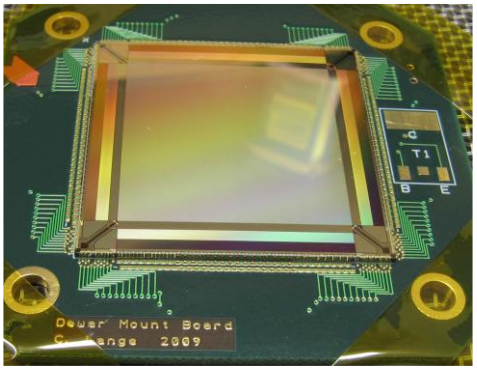

Figure 3.3 : Integrated RIIC and PCB

3.3 SLEDS Interface Board

The SLEDS interface board works as a link between the FPGA, the DAC, and the SLEDS RIIC to facilitate functions such as current measurement and a programmable circuit breaker to prevent damage to the SLEDS array.

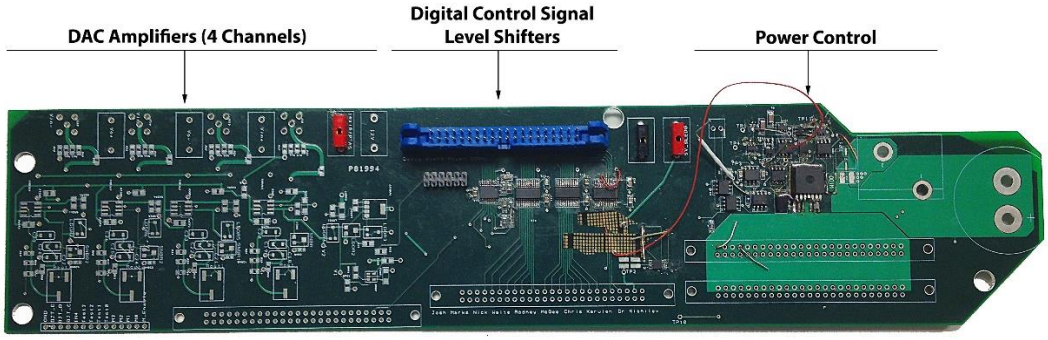

Figure 3.4 : Generation one SLEDS interface board

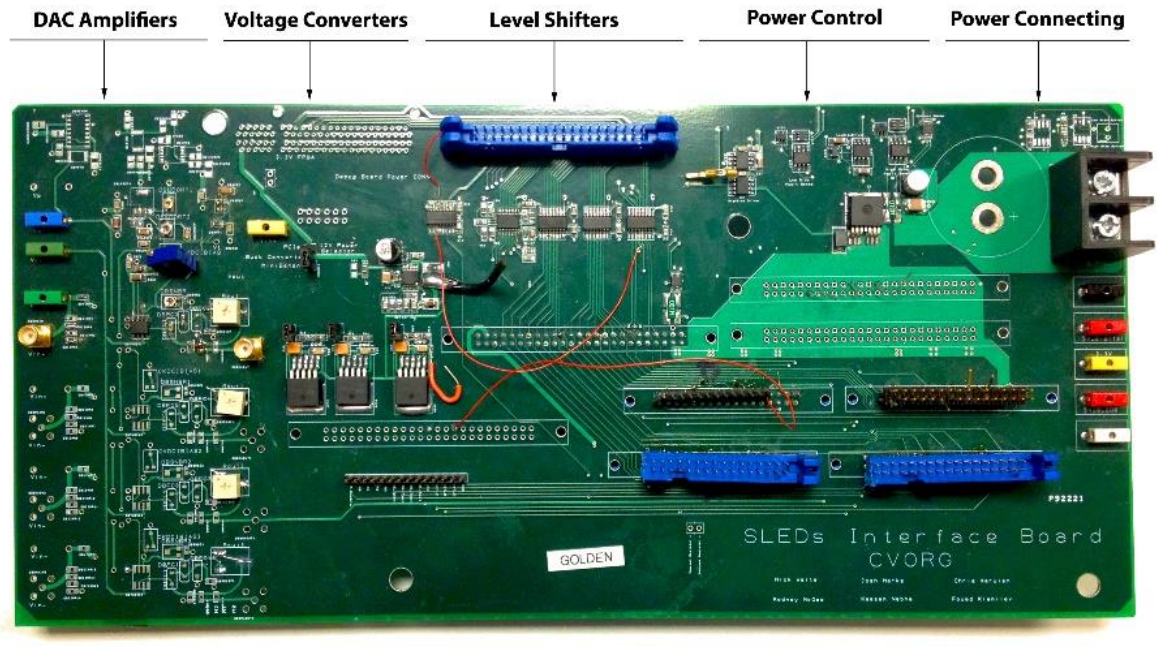

Figure 3.5 : Generation two SLEDS interface board

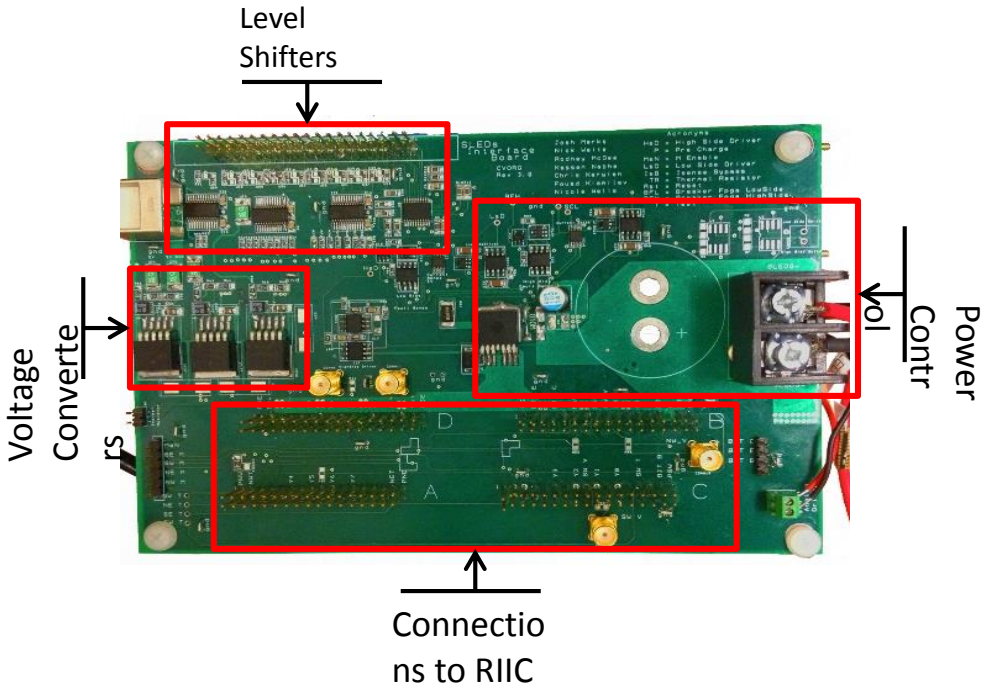

Figure 3.6 : Generation three SLEDS interface board

The Figures above show three generations of custom digital interface boards that have been developed. The latest interface board shown in Figure 3.6 was developed due to two major problems found in both previous generations. Both boards had a lot of signal integrity problem on the digital lines causing miss pixel write. It was also determined that the ground-return strategy was causing "ground bounce" on the boards. Both of these issues were negatively affecting the refresh speed and quality of the imagery that we were able to display on the SLEDS projector. To correct these issues, we redesigned the digital interface board. The redesign uses a new power supply routing strategy, which dramatically reduces ground bounce effects. We have also terminated the signal lines going to the SLEDS RIIC.

3.4 DAC and Amplifier

In this system we use an off the shelf digital to analog converter (DAC) show in Figure 3.6. The DAC used is the analog device AD9747. This particular DAC has a high dynamic range of 96 dB and has programmable current outputs. Since the LEDS control circuit takes a voltage and the DAC outputs current it was necessary to use a trans-impedance amplifier (TIA) to convert the current input into a voltage output.

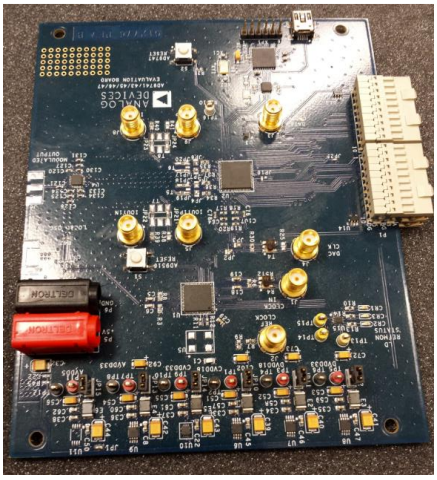

Figure 3.7 : DAC evaluation board

After measuring the capacitance and inductance of the analog line that drives the sleds chip we found that the impedance was much higher than anticipated. This issue prevented us from driving the SLEDS array at high refresh rates because of the resulting slow signal rise times. Correcting this required a re-design of the analog interface PCB. The new analog interface PCB design is shown in Figure 3.8 below. The new board design uses a Direct Service Line (DSL) operational amplifier instead of more conventional operational amplifier to drive the RIIC. This amplifier has approximately 100 times the drive strength of the previous amplifier that was used. In addition the new board has several potentiometers to adjust ringing, stability, and rise time signal characteristics. Additional information about the SLEDS previous amplifiers can be found in Chris Kerwien's thesis [2] about analog interface electronics for SLEDS.

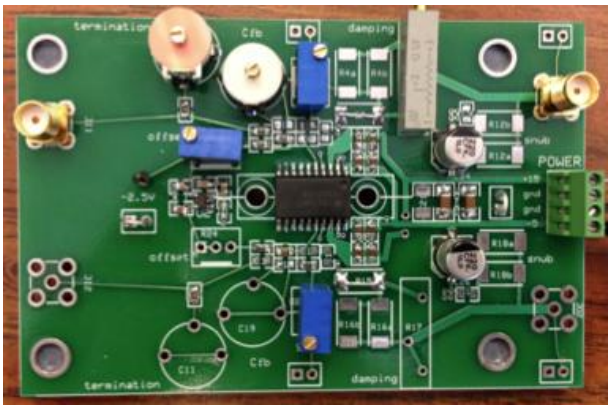

Figure 3.8 : Amplifier board

3.5 ML605 FPGA

Figure 3.9 shows an ML605, which is a field programmable gate array board. The ML605 is a very important piece in the system; it allows us to programmatically drive signals depending on the desired output. We use EDK (Embedded Development Kit) to design our VHDL peripherals, and SDK (Software Development Kit) to develop the firmware which drives the VHDL peripherals. The ML605 has a high and a low pin count FMC slots that are heavily used by our system connecting to a DVI FMC card and to the SLEDS interface board. Below is a Figure of the ML605.

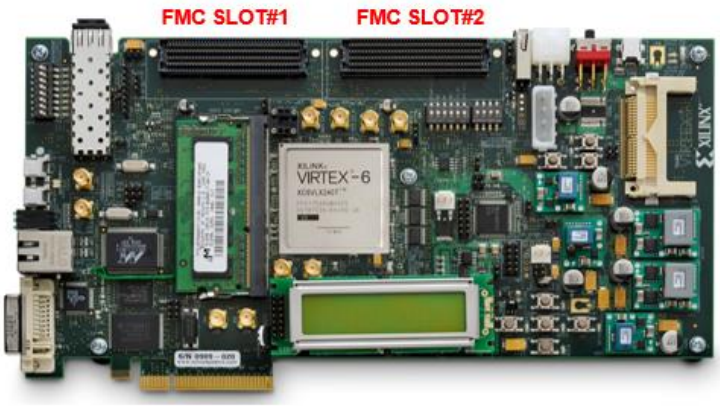

Figure 3.9 : ML605 FPGA

3.6 Packaging

Figures 3.10 below shows our final closed system electronics (CSE) hardware. This system operates from a standard 120V AC power supply and uses USB connection to a PC for controlling CSE. It has standard DVI input and DVI output to monitor the images being displayed on the projector.

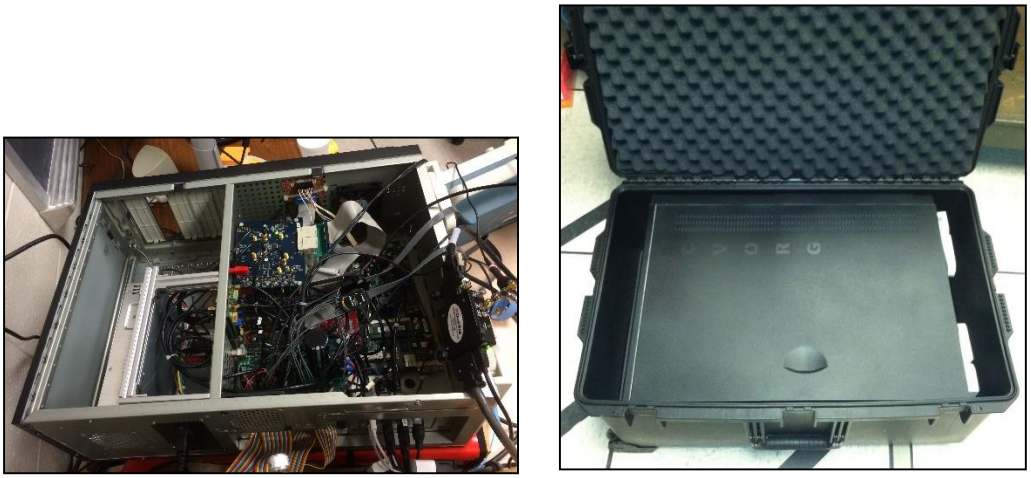

Figure 3.10: Close System Electronics (CSE)

Chapter 4

OVERVIEW OF SLEDS OLD AND NEW

4.1 25 Hz SLEDS Hardware Overview

The 25Hz SLEDS projector design uses is a single-color 512x512 SLEDS array and an ML605 FPGA motherboard that has 2 FMC expansion slots. In expansion slot one is a DVI input-output card, and in expansion slot 2 is a debug board that interfaces with custom digital and analog circuit boards that drive the SLEDS RIIC as shown in Figure 4.1 below.

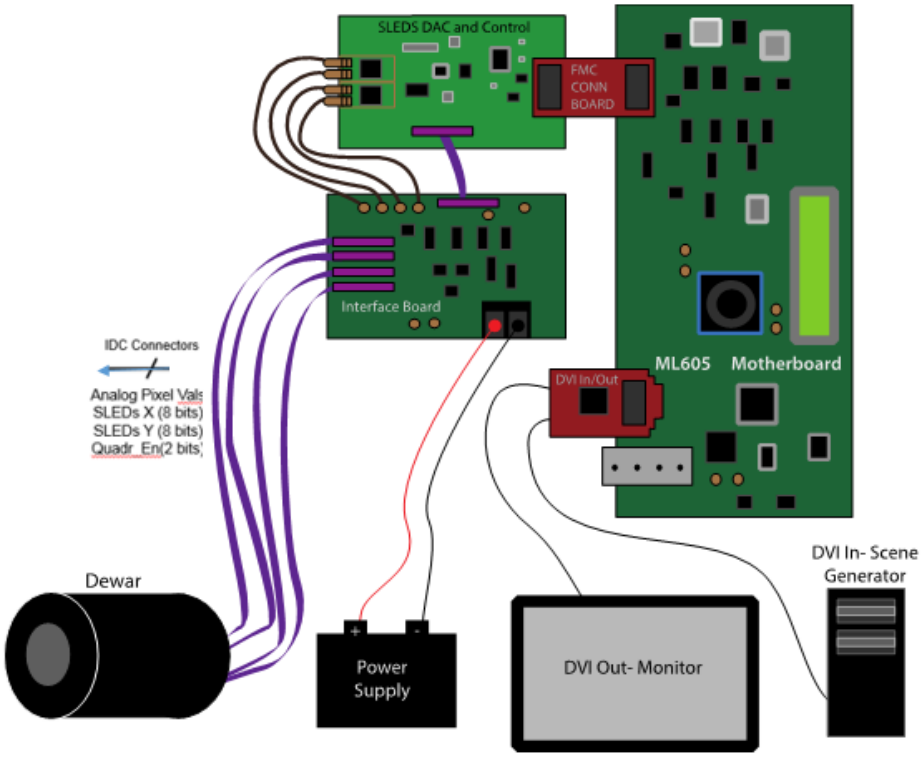

Figure 4.1 : High-level 25 Hz SLEDS

4.2 25 Hz System Overview

Figure 4.2 shows an architectural overview of the system inside the ML605. There are two input methods that can be used to run SLEDS. The first is a DVI input the second is a Python user interface. When operating the system in DVI input mode, the user can simply input the desired image through a DVI port to SLEDS. After a series of system operations, the IR representation of that image will be displayed and can be captured by an IR camera. In the Python user interface mode, instead of inputting an actual image, the user has a number of commands to choose from, such as "draw grid" or "draw square", that tell the system to run certain configurations. When a Python command is chosen a series of data bytes is sent to the ML605 through a serial communication between the computer that holds the Python code and the serial port on the ML605. This data is sent to a MicroBlaze processor that is programmed with C using a Software Development Kit (SDK). The C code receives these data bytes and initializes the system. The data bytes received by the C code from the serial communication gets decoded and upon the decoded result a command from C to VHD is activated, causing the system to behave as desired by the user. For each Python instruction, a different C function exists to execute the command in the system.

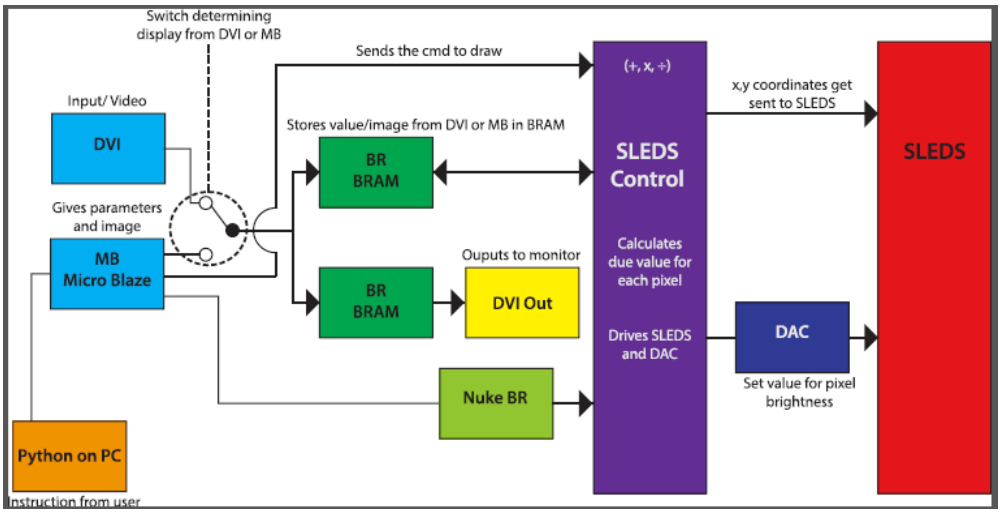

Figure 4.2 : 25 Hz SLEDS system architecture

Once the input is given to the system, whether it is through DVI or the Python interface, the values of each pixel must be stored. For this purpose, there are two BRAMs where the inputs get stored. Both BRAMs consist of 512x512 registers, each with 16-bit values. Each 16-bit value represents the intensity value of a pixel with respect to its location on the SLEDS array. Besides the input BRAMs, there is another BRAM referred to as the Non-Uniformity Correction (NUC) BRAM. The NUC BRAM has 512x512 registers, each with 20-bit values. Regardless of which type of input mode is used, the Microblaze first loads the NUC BRAM with the NUC parameters for each pixel.

Each of the input BRAMs store exactly the same data, but the data is used differently in each one. One of the BRAMs simply outputs the information to a DVI port so that the user can see what image is being sent to the system. Upon request from the SLEDS controller peripheral, the input BRAM and NUC BRAM sends their

values for a particular pixel location. These values are combined in a mathematical operation shown in Figure 4.3. This results in a non-uniformity correction for that particular pixel. Further description will be discussed in section 5.1.

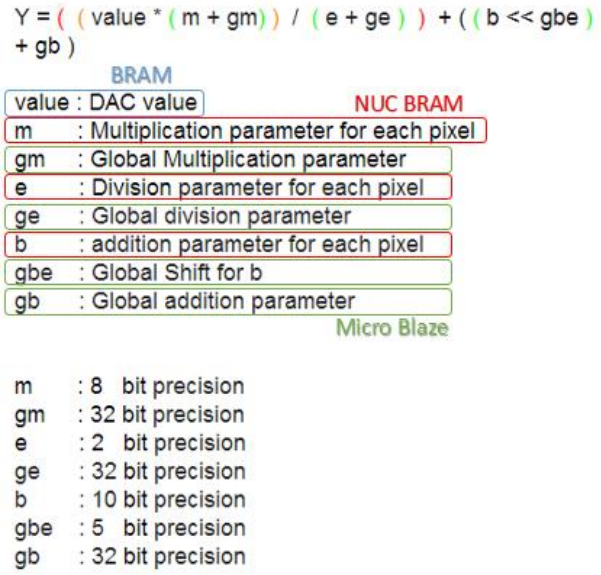

Figure 4.3 : NUC components and Mathematical operation

The SLEDS controller peripheral runs on a 100 MHz input clock producing a tick of 10ns. The peripheral contains two counters simultaneously running at 10ns per tick; the internal counter and the addressing counter. The internal counter uses three variables to determine when to do certain tasks. These variables are called A_tick, E_tick, and P_tick. A_tick is used to handle the analog signal intensity, E_tick is used to handle the pre-charge signal, and P_tick is the time it takes for the LED to physically turn on. The addressing counter simply counts from zero to 262144 (512x512) in order to address each pixel.

The system starts addressing a pixel at the A_tick variable. When the internal pixel counter reaches A_tick, an address is sent to the BRAM containing the input data as well as the NUC BRAM. At A_tick plus one, the input data BRAM sends the 16-bit intensity value for the pixel at that particular address to the SLEDS controller peripheral. At the same time, the NUC BRAM sends the 20-bit NUC parameters at that address to the SLEDS controller peripheral.

At A_tick plus two, the address of the pixel is sent to the RIIC. By A_tick plus three, the SLEDS controller peripheral has both the pixel intensity value, and the NUC parameters for that pixel. From A_tick plus three to A_tick plus five, the system performs the mathematical operations that apply the NUC parameters to the intensity value coming from the BRAM; a graphical representation of the operation is shown in Figure 4.4. This is a necessary step in the process because it ensures that all pixels are uniformly corrected at every given input value, regardless of the inevitable physical differences from one pixel to another. At A_tick plus six, the 16-bit pixel value, corrected for non-uniformity, is then sent to the DAC where it is converted to an analog signal and amplified to an appropriate voltage. From the DAC, the voltage signal is sent to the RIIC to be projected by the SLEDS array. This process, from sending an address to the BRAMs to sending an analog voltage signal to the RIIC, is repeated over and over for each pixel in the array.

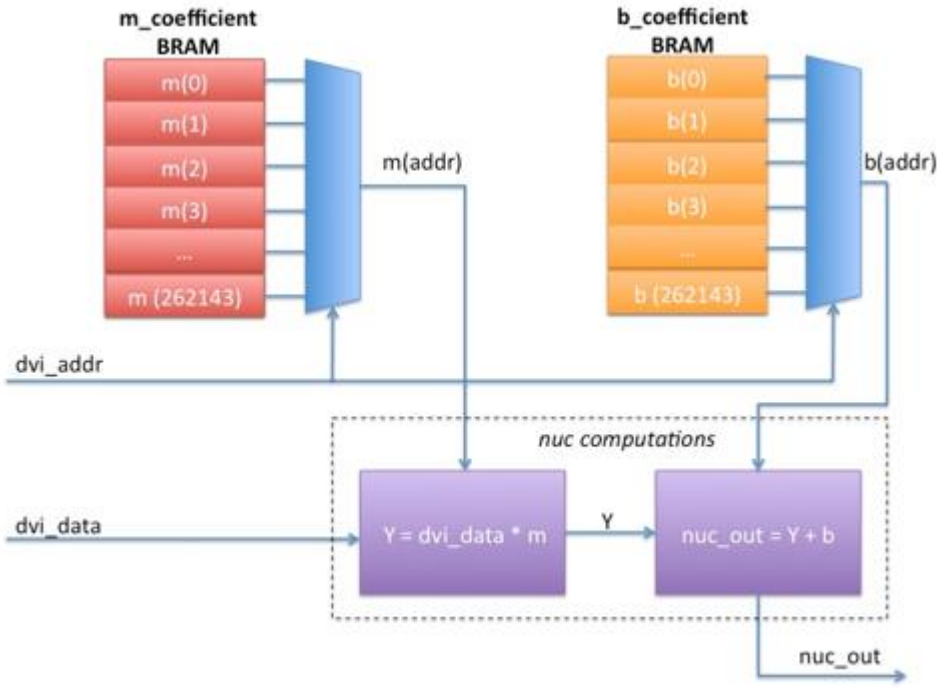

Figure 4.4 : Architectural process of NUC

If P_tick is set to 15 ticks, then a single pixel takes 150ns to write out to the SLEDS array. For a 512x512 array of pixels, simple math would give us 512x512 = 262144 pixels. At 150ns per pixel, that equals 39ms, or 25Hz, to write to the entire SLEDS array.

4.3 100Hz System Overview

The current 100Hz system is very similar to the 25Hz system, except for a few differences that allow the system to run four times faster. The main differences between the two systems are the storage of the input data; the method used to NUC the pixel values, and the number of DACs.

Just as in the 25Hz system, there is a 512x512 NUC BRAM that is initially loaded by the Microblaze with the NUC parameters for each pixel. There is still a 512x512 BRAM that takes in the input data, and sends it to a DVI output port to be viewed by the user. However, in the 100Hz system, the address and intensity value for each pixel is sent to a four-channel input peripheral instead of directly to another BRAM. From the four-channel input peripheral, the address for the pixel is sent to the NUC BRAM, and the NUC parameters for that address are sent back to the four-channel input peripheral. The NUC parameters and the pixel value are then combined through a pipelining technique shown in Figure 4.5. This method actually performs the NUC as the input data comes in, rather than NUCing the pixel values later when they are sent to the SLEDS controller peripheral, as is the case with the 25Hz system.

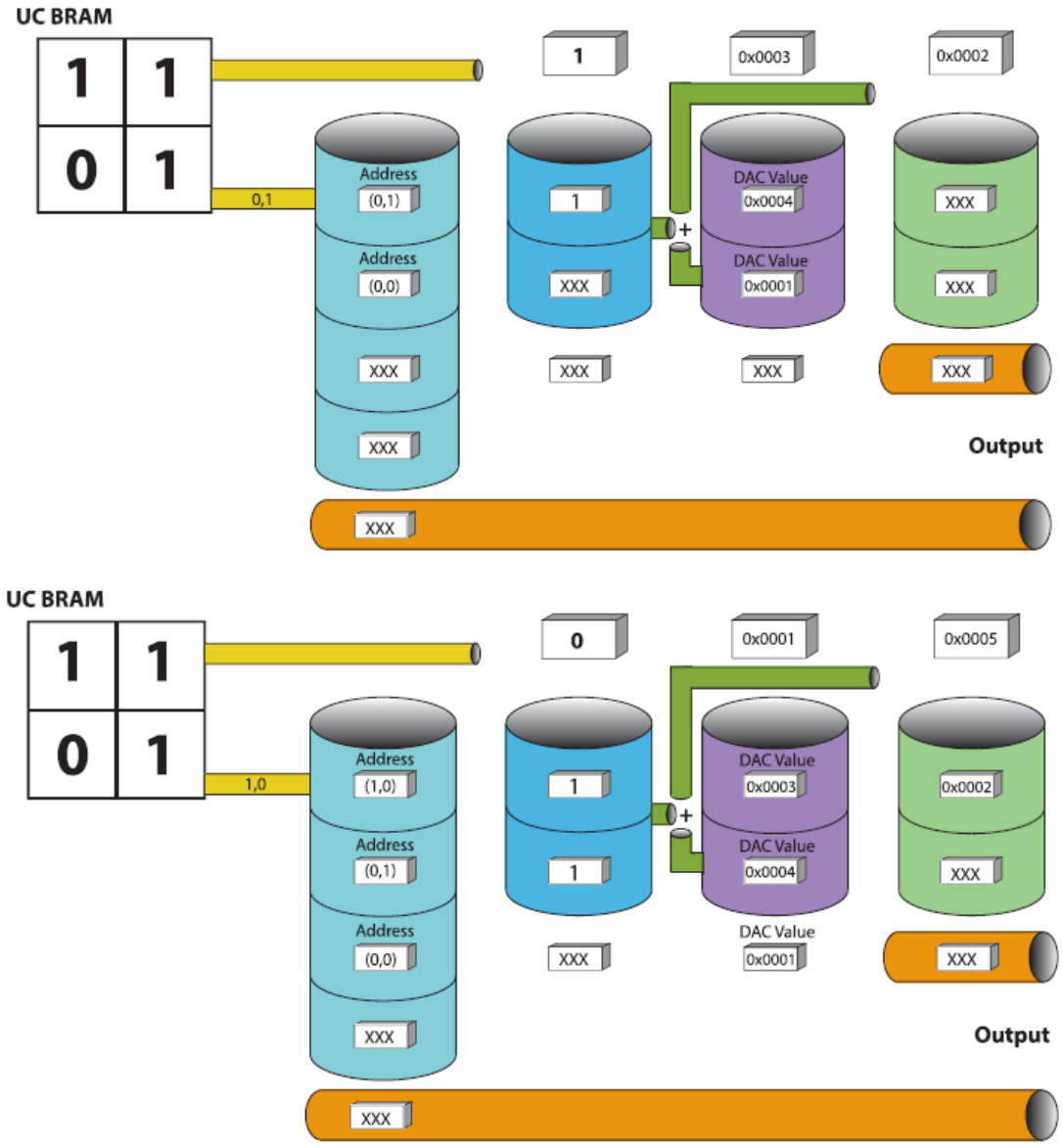

Figure 4.5 : NUC pipelining operation

Instead of having one 512x512 BRAM to hold the input data, the data is now stored in four 256x256 BRAMs as shown in Figure 4.6. The four-channel input board takes the NUCed pixel value and uses the address and a demultiplexer to choose the

correct BRAM, according to which quadrant of the 512x512 SLEDS array the corresponding pixel lies. Therefore it is called the "four-channel input peripheral" because it inputs the data to the BRAMs and outputs four channels from the demultiplexer, one for each BRAM.

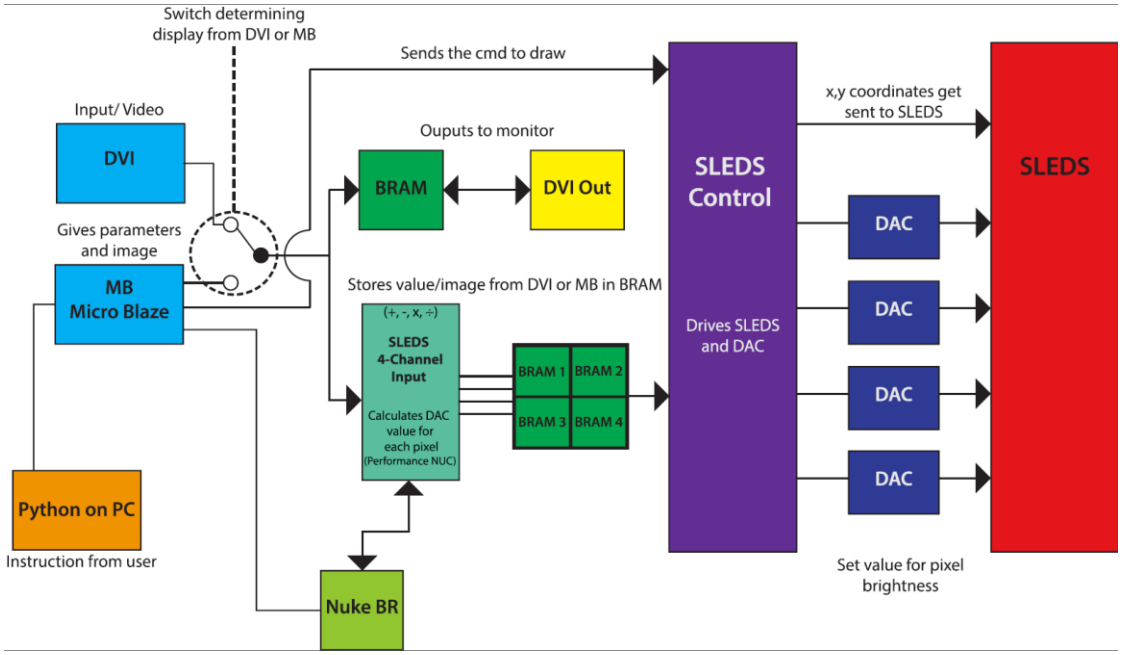

Figure 4.6 : 100 Hz SLEDS system architecture

In this system, when the internal counter reaches A_tick, one address is sent out to all four BRAMs. At A_tick plus one, the intensity values at that address in each BRAM are simultaneously sent back to the SLEDS controller peripheral. In the 100Hz system, there are now four DACs to handle each of the intensity values. At A_tick plus two, all four intensity values are sent to the DACs to be converted to analog signals and amplified to the correct voltages. This change to four BRAMs and four

DACs allows four pixels to be written to at the same time, as compared to writing to each pixel in the entire array one at a time. As a result, this system is four times faster and runs at 100Hz.

4.4 100 Hz SLEDS Hardware Overview

The block diagram below shows the CSE we built that supports four drive channels and operates at 100Hz frame rates. This is one-step in an effort to eventually implement a 1 KHz frame rate CSE. To move toward 1 KHz, we will need to migrate the design to a new motherboard that supports 10 FMC expansion slots. This approach will enable us to have multiple four-channel drivers in the system, in which each analog channel increases the frame rate by a factor of two.

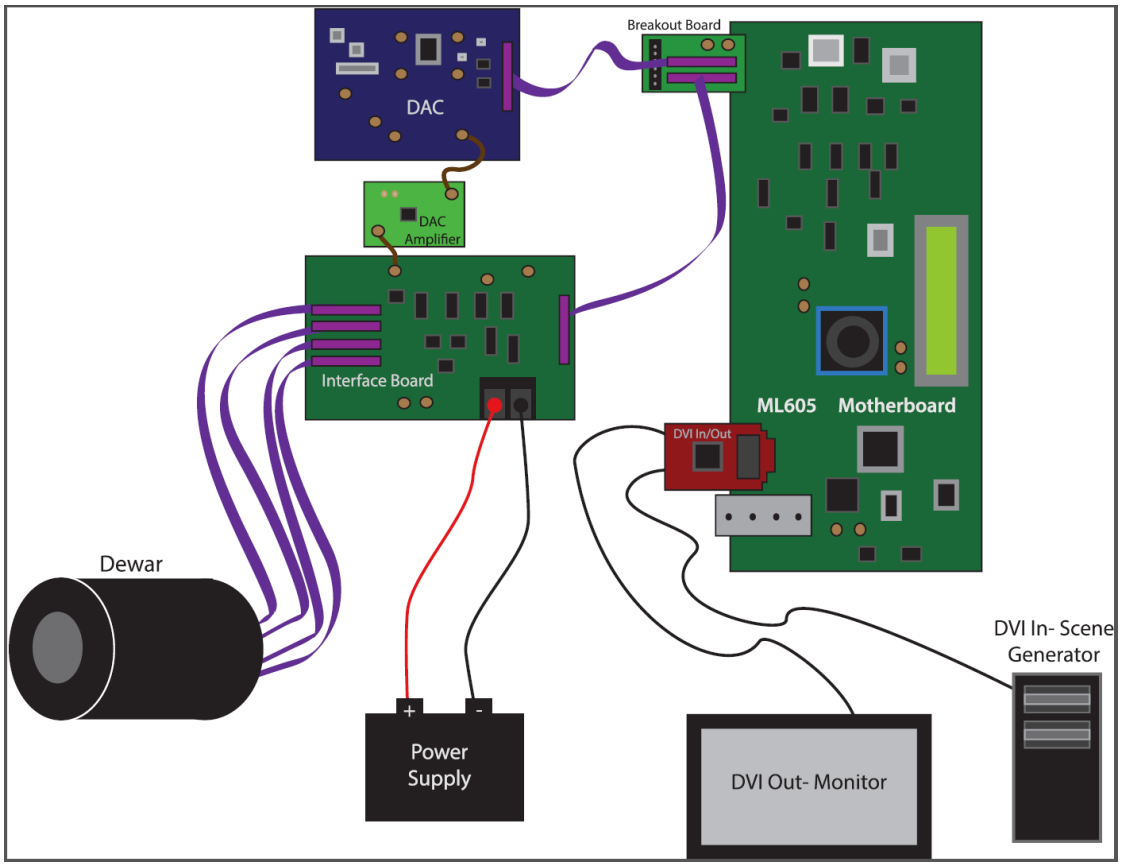

Figure 4.7 : High-level 100 Hz SLEDS

4.5 Future Revision

We had originally planned to continue using our ML605 platform to reach 1 KHz with the SLEDS system. However, upon making our calculations, we determined that the ML605 would only be able to support frame rate of up to 950Hz; falling short of the 1 KHz design goal. Furthermore, we wanted a design that would scale to larger array sizes and higher frame rates. Based on that we have revised our design to use a much more powerful FPGA motherboard – the TB-6V-LX760-LSI. This motherboard

has 10 FMC expansion slots and is designed so that multiple motherboards can be stacked horizontally or vertically. This motherboard will enable us to design a CSE package that will be scalable to meet future SLEDS projector needs. Using this motherboard, we can deliver a 1 KHz single-color 512x512 array or a 1 KHz two-color 512x512 array. Below are two options for building this system one option uses a single motherboard and the second option uses two motherboards.

The 1 KHz single-color or dual-color 512x512 system will use a 16-channel SLEDS DAC-Amplifier system. We will make a custom LPC (low pin count) FMC card that integrates the 4-analog drive channel solution that was made for the 100HZ SLEDS system. We will make a custom LPC FMC card, called FMC-SLEDS-JOSHIF, that interfaces with the new SLEDS interface board. The number of DVI FMC cards would scale depending on the input format. Each DVI channel has a limit of 4Gbit/sec; therefore, it is possible to achieve a 16Gbit/sec DVI input stream bandwidth if all four DVI FMC cards are in use.

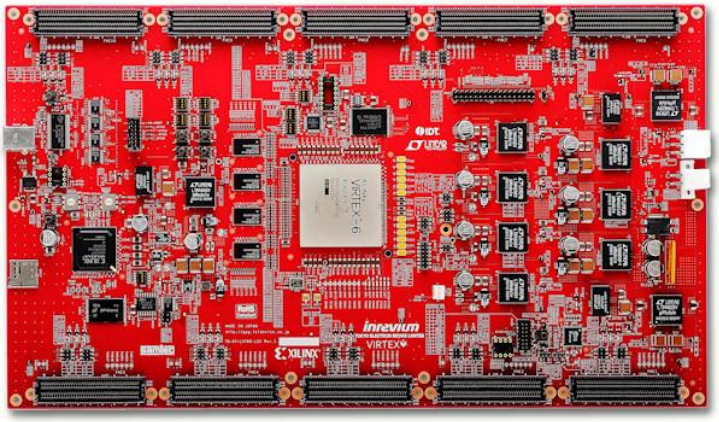

Figure 4.8 : TB-6V-LX760-LSI Board

The math below shows the analog sample rate that we need to achieve a 1 KHz frame rate. In summary, for 1-color, the sample rate is 17MHZ and for 2-color the sample rate is 33MHZ.

1-color system :(1000Hz / 16 channels) x (512 x 512 pixels/frame) = 16,384,000 pixel/second ~ 17 MHz

2-color system :(1000Hz / 16 channels) x (512 x 512 pixels/frame) x 2 colors = 32,768,000 pixel/second ~ 33 MHz

An alternate CSE design is one using a 32-channel SLEDS DAC-Amplifier system. This system would be for 2-color SLEDS or for 1Kx1K and beyond systems. This would be achieved by stacking two motherboards using this connector board:

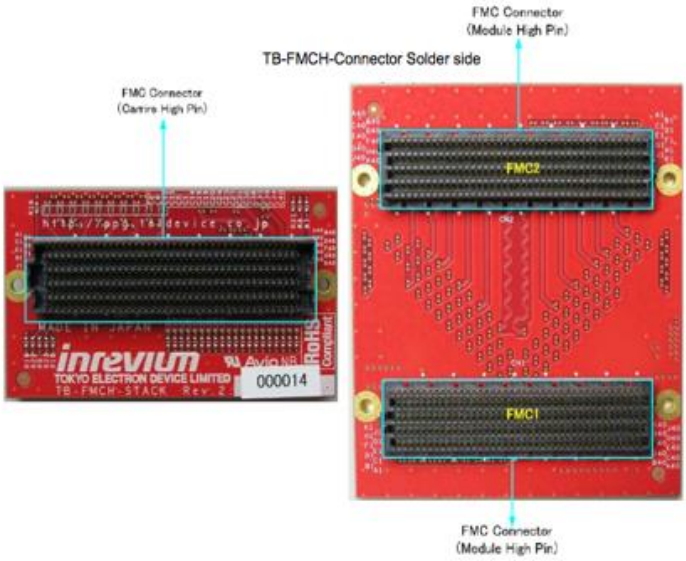

Figure 4.9 : TB-FMCH-STACK and TB-FMCH-CONNECTOR

Each motherboard is designed to handle one color in the 2-color projector system. As a result the sample rate would be smaller: 9MHz for 1 color and 17MHz

for two color.

1-color system: (1000Hz / 32 channels) x (512 x 512 pixels/frame) = 8,192,000 pixel/second ~ 9 MHz

2-color system: (1000Hz / 32 channels) x (512 x 512 pixels/frame) x 2 colors = 16,384,000 pixel/second ~ 17 MHz

Figure 4.10 bellow shows an example architecture of the system using one mother board and 16 DACs, and Figure 4.11 shows an example architecture of the system using Two mother board and 32 DACs.

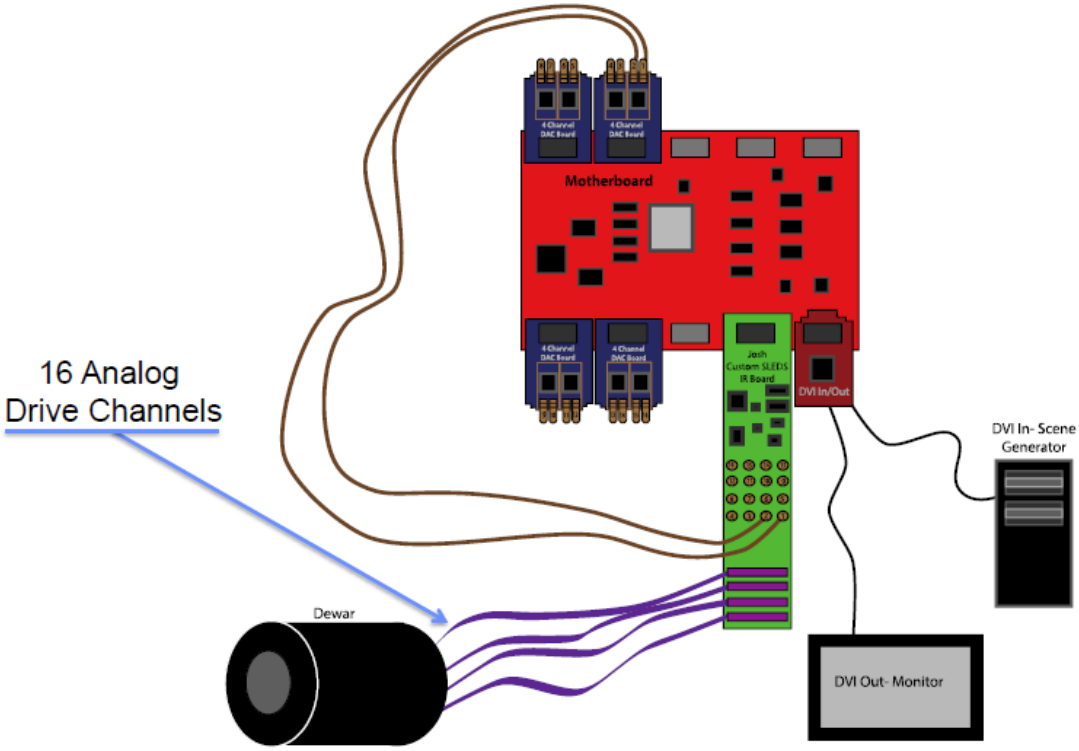

Figure 4.10: One color 1K Hz SLEDS system

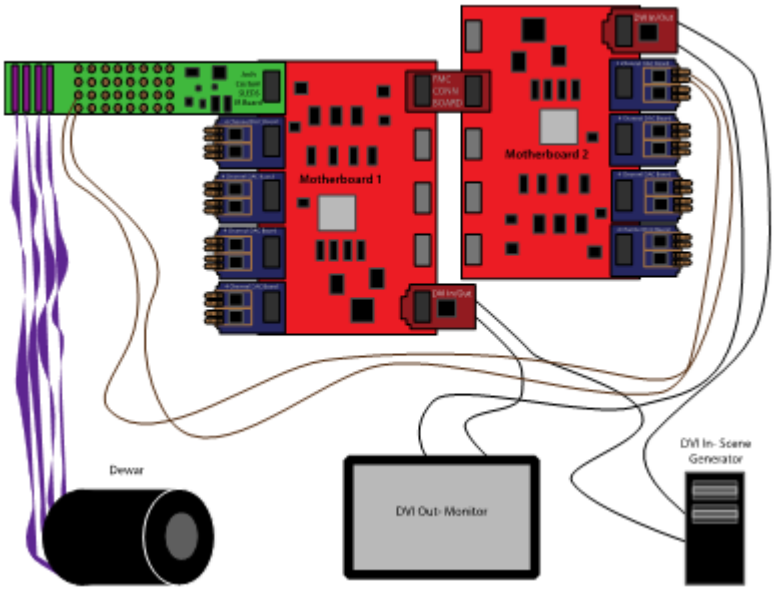

Figure 4.11: Two color 1 KHz SLEDS system

Chapter 5

NON UNIFORMITY CORRECTION

5.1 Light Based NUC

LED's tend to have a different behavior from one another. For example, giving two LED's the same current might result in one being brighter than the other. To display a uniform projection, all LEDS in the array have to be at the same brightness at a specific voltage. To correct the issue we had to do a Non Uniformity Correction (NUC). The process of NUCing the array using light observation is called L_NUC.

To perform L_NUC on the array the first thing we have to do is collect light data for each pixel. This data will contain how bright each pixel is at a given voltage. After sweeping each pixels input voltage from 0 to 10v and recording its brightness, we extract this data and plot it in a curve as shown is Figure 5.1.

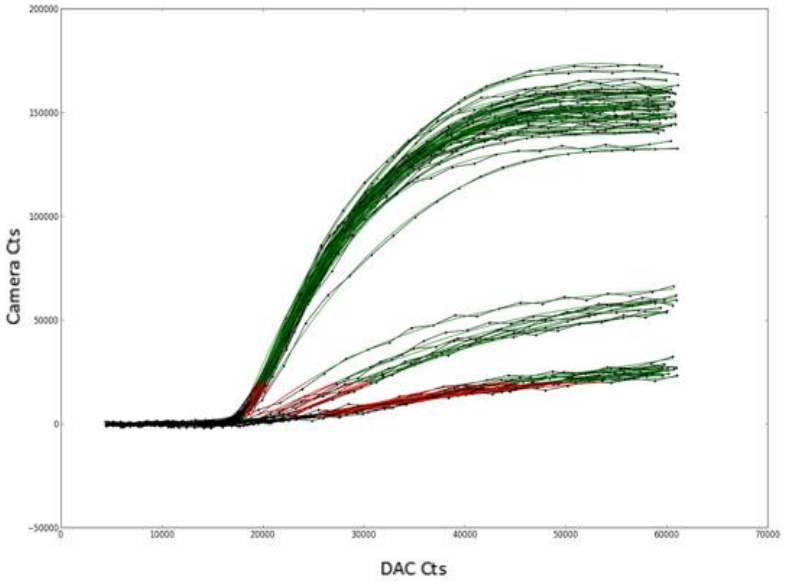

Figure 5.1 : 64 Pixels Light curves

Once the extraction of the curves for each pixel has been done, a high order polynomial equation is generated to fit all the curves together. For each pixel we have unique variables that plug in to the high order polynomial equations that would represent the fit. Those unique variables for each pixel are saved in a table called the ideal NUC table. Due to the lack of memory on the ML605, it's difficult to store all the parameters generated from the polynomial equation for each pixel and implement the polynomial solution in the FPGA. Instead, we take the ideal NUC table and convert it to a real NUC table by converting the higher order polynomial equation into a function $y = (M / E) x + B$ for each pixel. Now we only have to store three values with a maximum total size of 20 bits per pixel, which is a total size of 5242880 (512 * 512 * 20) bits for the whole array. One way to improve the efficiency of the NUC is to make sure all four DAC's are at the same amplified level.

Once the NUC table is generated it gets loaded into the ML605 via the python interface. The table gets stored in the NUC BRAM, and is used by the SLEDS controller peripheral as mentioned in previous chapters.

5.2 Current Based NUC

After various experimental results it was confirmed that current measurements can be used instead of light output measurements to identify good and bad pixels in the projector. This was a significant finding because current testing is much easier than light testing. We refer to this as I_NUC because it is based on measurements of the current. Figures below show comparison between L_NUC and I_NUC data.

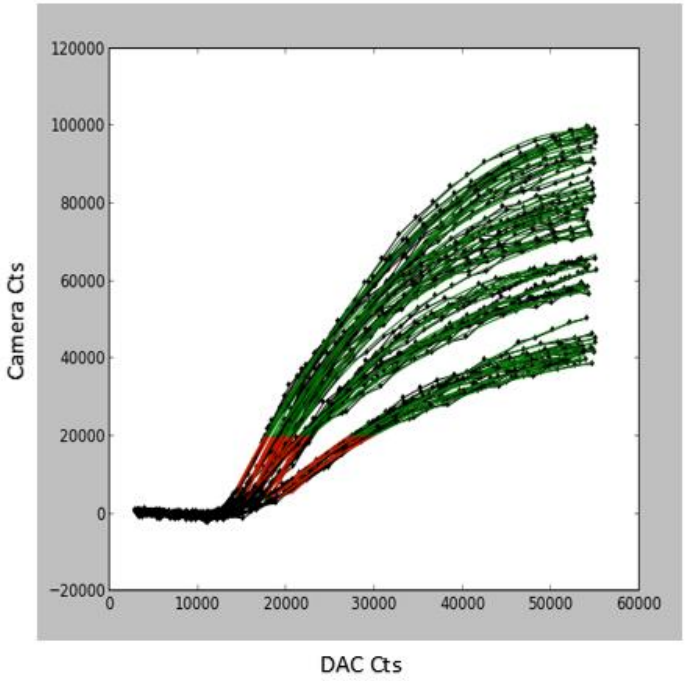

Figure 5.2 : light output (IR camera counts) vs. DAC value for 64 pixels.

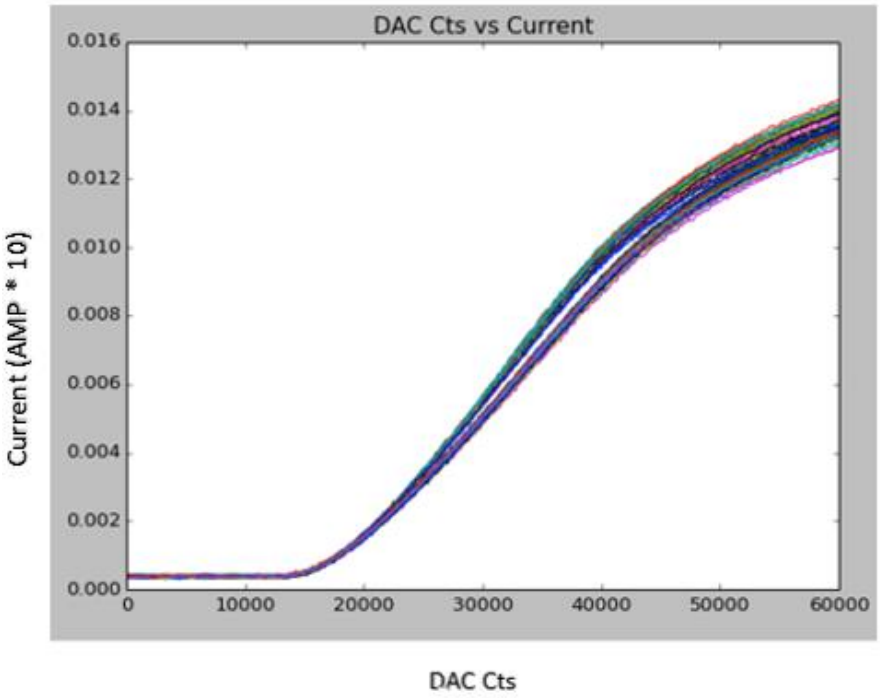

Figure 5.3 : SLEDS current (Amperes) vs. DAC value for same 64 pixels.

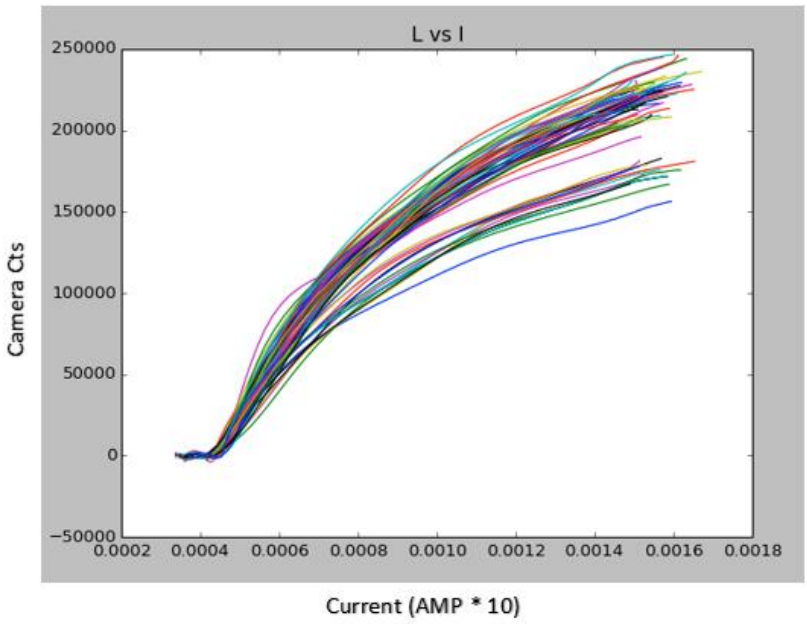

Figure 5.4 : Combining results from Fig.1 and Fig.2

The current measurement was possible due to a current monitoring chip located on the SLEDS interface board. The chip monitors the current going into the SLEDS RIIC. We transmit the information between the chip and the ML605 using I2C communication, a two-wire serial bus that provides a link between circuits. After reading the data through the I2C peripheral provided in the ML605 the data gets sent to the python interface for data processing.

To get a current curve for each pixel, we have to sweep it from 0 to 10v just like the L_NUC process. But unlike L_NUC, instead of getting light readings we get current readings. Since current, voltage, and light are related, we can produce the same NUC table. Unfortunately, it takes a longer time to collect current data than light data

due settling time in the pixel and the fact that we have to measurement each pixel at a time.

Future work will be done to gather more information at a much faster speed to allow do a full NUC on the system using only current data collected by the current monitor.

5.3 Combining both NUC methods

As mentioned in section 5.1, to have the most dynamic range its best to have all the amplifiers at the same voltage output. In order to do so the amplifiers would need to be tuned by using the tunable resistors on the amplifier board. But since the system is packaged, it's difficult to reach these resistors. Since I_NUC requires no external equipment, it's possible to get reading of one voltage value of all the pixels in the array in a short period of time and use that data to correct the uniformity across quadrants instead of pixels. After doing that a regular L_NUC can be used to generate the NUC table.

5.4 NUC Summary

The left Figure 5.5 below shows the characteristics of the SLEDS array after it has been processed through our FPGA-based NUC. This process has been done by collecting camera counts and processing them. From this left figure we can conclude that this array has a very good yield except for the upper right corner and the bottom right side.

The right Figure 5.5 below is an image of array yield that was obtained using I_NUC. This method uses the actual LED power consumption to evaluate the pixels yield and uniformity. Measuring current going to each pixel at a given voltage allows us to calculate the LED power consumption.

One can see both images look fairly similar, which shows that I_NUC is just as effective as L_NUC in determining the arrays uniformity. One major difference between the two figures is that the left figure has been NUCed while the right figure is just raw data. The difference in colors between the two is due to the difference in scale. In Figure 6.5, the image is computed by voltage vs light, where in Figure 6.6, its current vs voltage. Each quadrant in the right image is a different shade because the amplifiers for each quadrant had not been tuned yet.

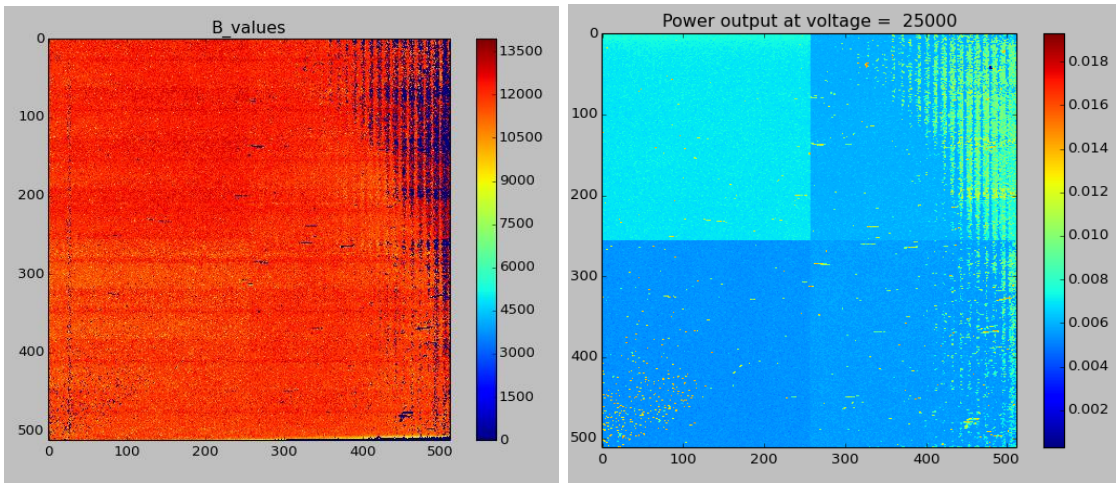

Figure 5.5 : L_NUC (left), I_NUC (right) array characteristics

5.5 Future NUC

Figure 5.6 below shows the yield of the latest array we have. Of the 262,144 pixels in the array, 20 pixels were completely dead and 92% of the pixels were NUCable with our current FPGA-base NUC algorithm. The remaining 8% of the pixels are functioning but their characteristics are non-linear and require a more sophisticated NUC algorithm to correct and operate. In the current NUC approach, these pixels are turned off during projector operation.

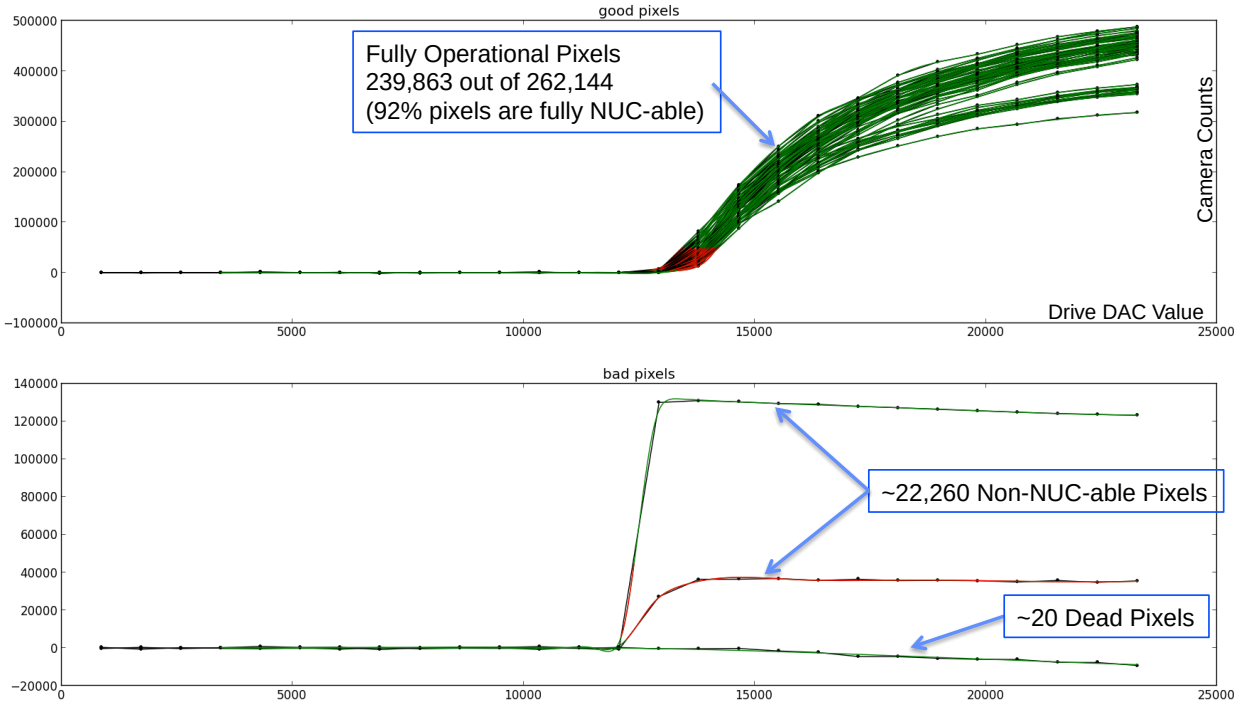

Figure 5.6 : Yield of the SLEDS array

To address the issue of non-NUCable pixels, we have built a computer to perform more complicated NUC processing using NVIDIA GPU. The Figure 5.7

below shows the operation of this computer. It has a DVI input that takes a DVI stream. Then a sophisticated NUC algorithm is applied to the pixels using GPU parallel processing and finally the resulting image is output to the FPGA-based CSE. This hardware setup enables much more sophisticated NUCing algorithms to be implemented.

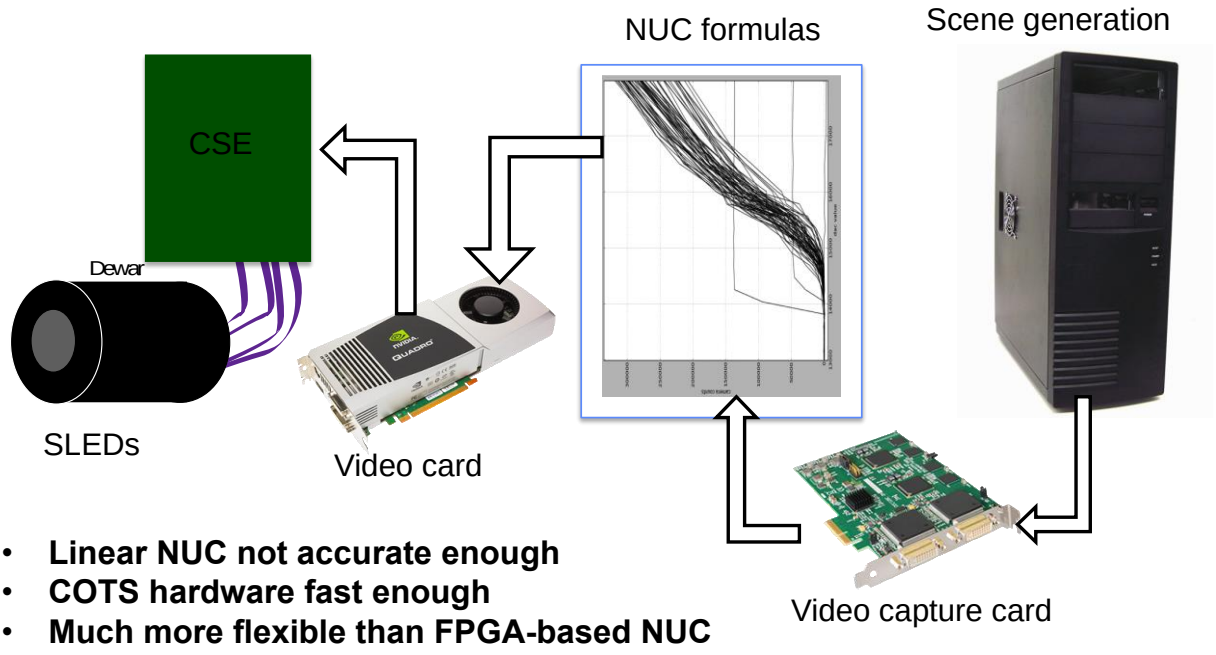

Figure 5.7 : Future NUC computer in operation

Chapter 6

RESULTS

6.1 Donald Duck

The projector consists of a CSE electronics box powered using AC power, a double walled flask with a vacuum between its walls (DEWAR) with active SLEDS RIIC inside of it, and a laptop computer to load NUC tables into the CSE during system initialization. All of the components are packaged in a rugged Pelican military-grade cases suitable for air transport. The system comes with an operation manual. The system has fail-safe checks built-in, such as overcurrent protection, which shuts down the projector if the current into the RIIC exceeds a certain thermally safe limit. All the wiring harnesses have been tied down and pull-tested for rigidity and stability. All efforts have been made to make the projector operation user friendly – so during normal operation, two front panel buttons allow the projector to be reset and powered on or off. The projector operation is very simple: first the power is turned on to the FPGA board, then power is turned on to the CSE, then the laptop computer is used to load CSE firmware and NUC tables, and finally the projector starts running until the CSE is powered off. When the projector is running, it takes DVI input and projects a 512x512 image from the upper left corner (if the incoming image has higher resolution) onto the RIIC. A copy of what is being displayed on the array is also sent

to a DVI-output port for the purpose of viewing the projector output on a computer monitor. Figure 7.1 below shows the final system completely connected.

Figure 6.1 : SLEDS completely connected

When it comes to testing or demonstrating the SLEDs system, often dots are used as shown in Figure 61. The reason being that it allows us to test each individual pixel. Once we have made sure that our NUC table is efficiently correcting the brightness of the pixels, as shown in Figure 6.2 a, we then display a video through SLEDS to test the projector, and visualize the system operating. In Figure 6.2 b, we show a snapshot of a short clip from a Donald Duck cartoon show. From the image, one can see how close SLEDS gets to showing the actual image. All the dark areas are not visible because it is below the cameras threshold, while anything above that is displayed on the screen.

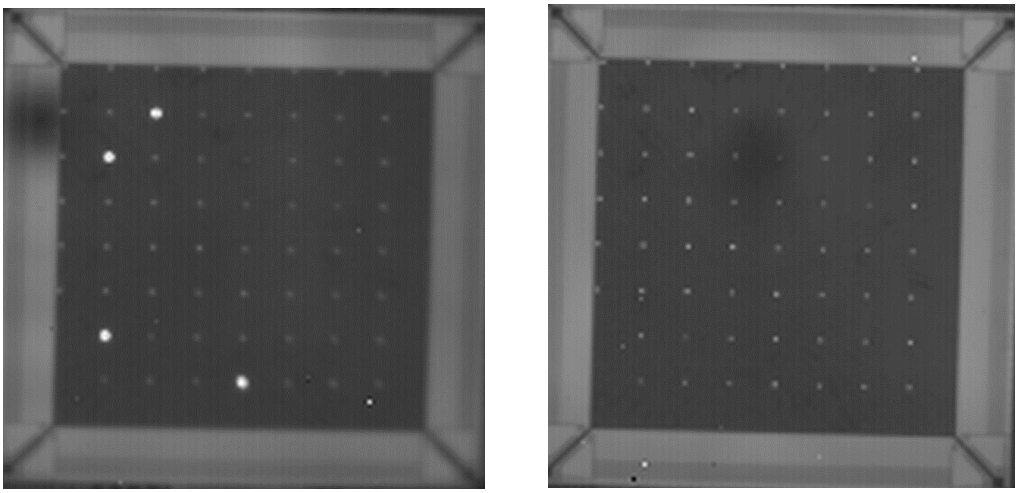

Figure 6.2 : SLEDS before NUC (Left) SLEDS after NUC (Right)

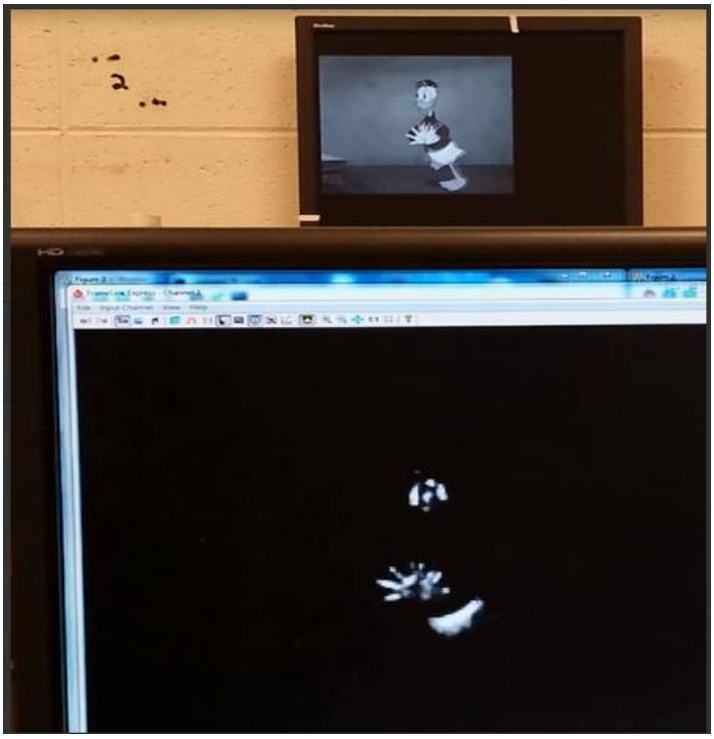

Figure 6.3 : Snap shot of a Donald Duck clip

The SLEDS has been sent to a testing facility where it was thoroughly examined. Below the results are shown, as well as plans for the future system to fix the issues currently facing SLEDS.

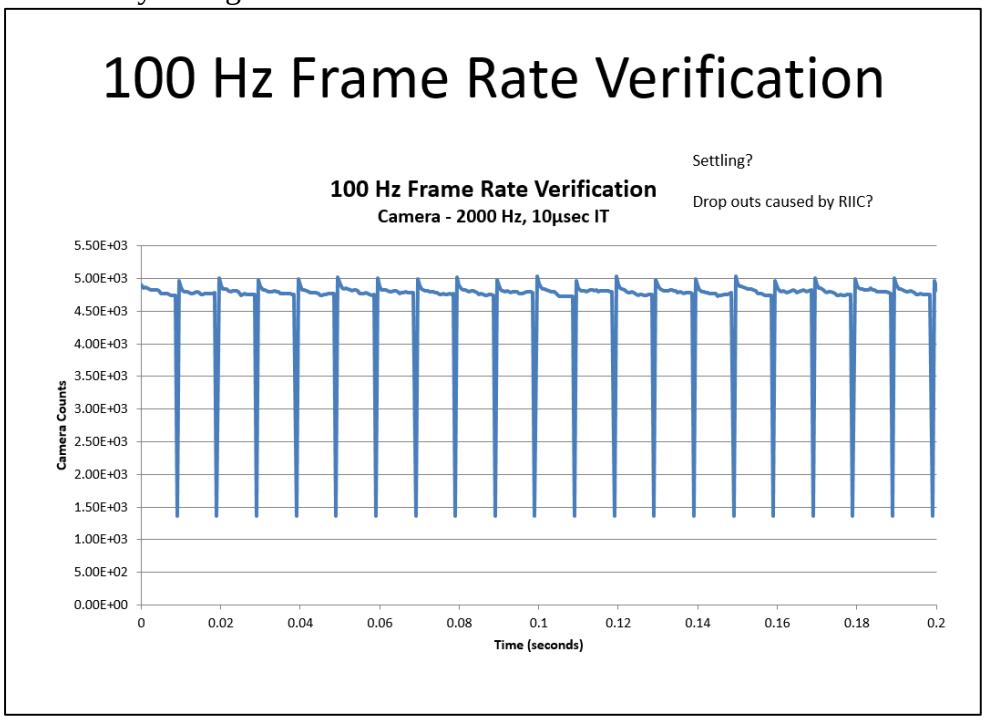

Figure 6.4 : 100 Hz frame rate

Figure 6.4 above shows the SLEDS projector imagery as it is being reset using a 100HZ sync signal. The camera in this test is operating at 2000HZ therefore it is able to see the reset being performed at the end of every frame.

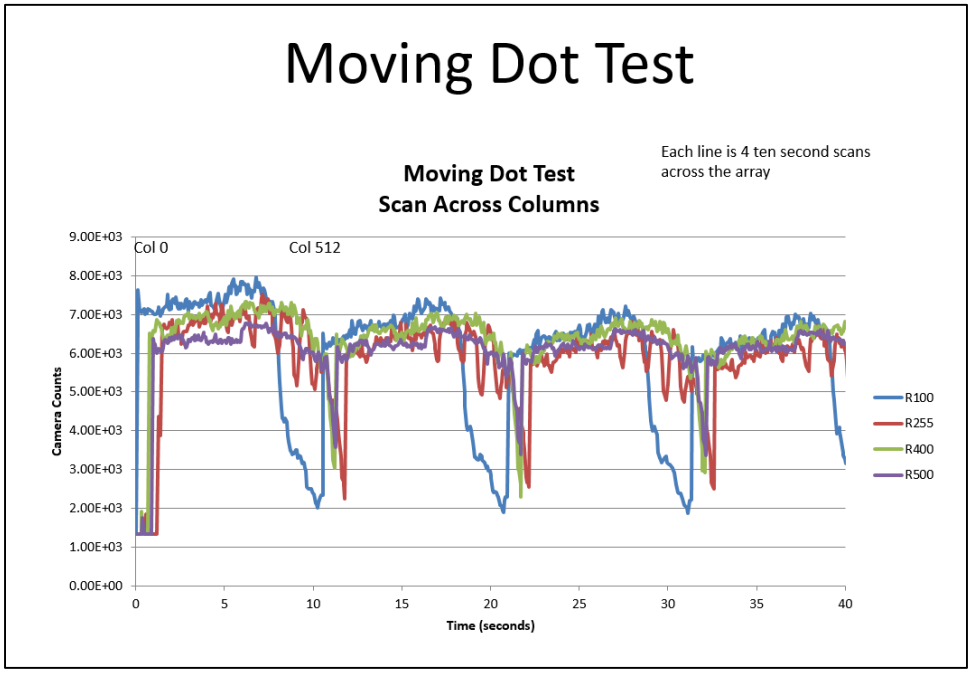

Figure 6.5 : Uniformity test with four moving dotes

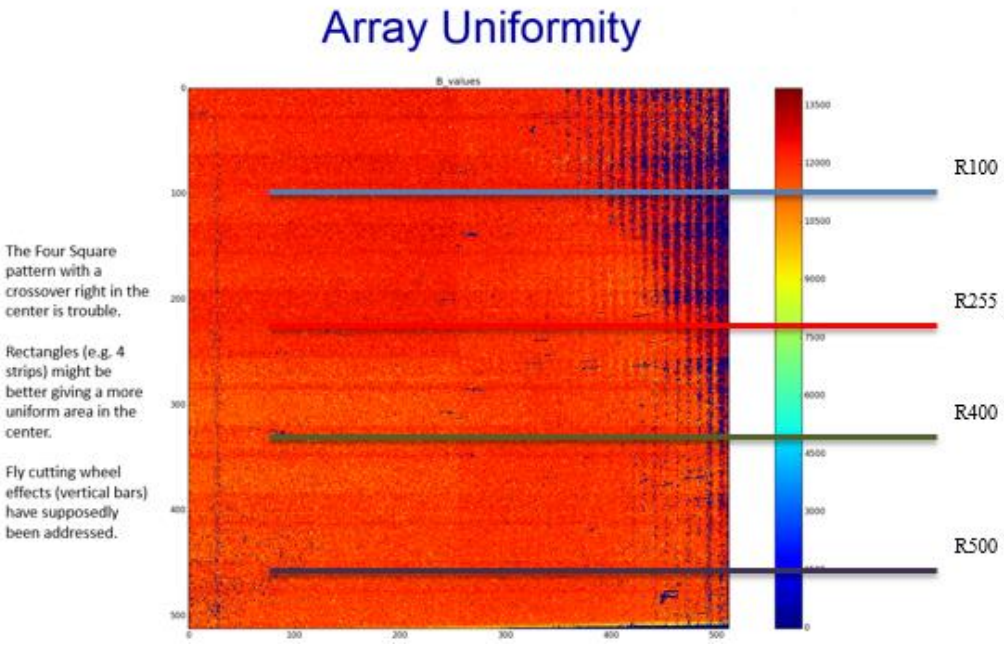

Figure 6.6 : SLEDS array uniformity

Figure 6.5 shows the yield of the SLEDS array across four arbitrary rows. As can be seen from the measured SLEDS array yield, shown in Figure 6.6 – the results found in Figure 6.5 are strongly correlated with the SLEDS array yield.

6.2 RIIC and CSE Improvements

- **50-Ohm termination of analog and digital lines**

In SLEDS projector, the CSE and RIIC communicate digital and analog signals over un-terminated signal lines. Using un-terminated signal lines causes ringing and signal settling delays, which would be problematic for future SLEDS goals of 16-bit analog signal resolution and 1KHZ refresh rate (e.g. the analog signaling rate to achieve 1KHZ frame rate is many MHZ requiring fast analog settling time).

In future SLEDS projector, the analog signal lines are double terminated. This is accomplished using 50-ohm external termination resistors both on CSE and RIIC side. The digital lines are 50-ohm terminated on the RIIC side.

- **Dual transistor Drive**

In SLEDS RIIC, one large PMOS transistor drives the SLEDS device. This transistor converts analog voltage into SLEDS current. It has been designed to "push" tens of milliamps into SLEDS – which corresponds to a huge stepping size in brightness causing a high apparent temperature light output. It turns out that for low apparent temperature light output (such as 300K) SLEDS needs only a few microamperes – this is an output current that SLEDS RIIC cannot drive because the

PMOS drive transistor is too big. Figure 6.7 below shows the noise at the low temperature caused by have a big driving transistor.

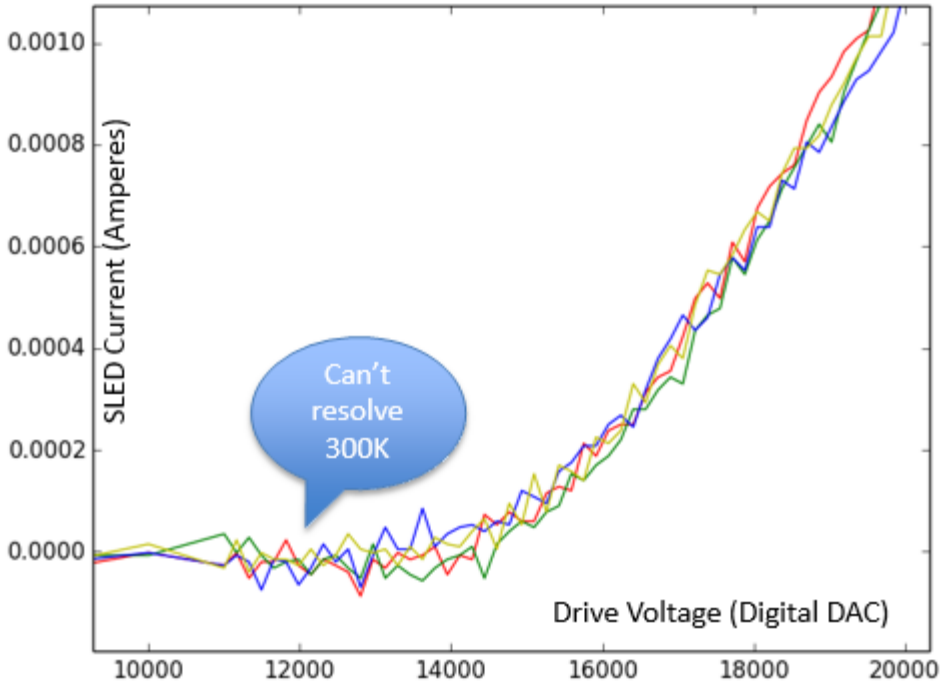

Figure 6.7 : Noise at low temperatures

In future SLEDS RIIC, driving SLEDS is done using two parallel-connected transistors: a large transistor for milliamp drive currents and a small transistor for micro-amp drive currents. We have added one digital bit for each analog input that signals whether the analog signal is meant for high- or low- transistor. This enables future SLEDS RIIC to drive SLEDS to both low and high apparent temperatures.

An added benefit of dual transistor drive is that we can turn on the entire SLEDS array when display low-temperature scenes. This is because at current of few

microamperes, all 512x512 pixels can be driven and not exceed the thermal limit of our package.

- **Snapshot mode**

In SLEDS RIIC, each pixel in the 512x512 array stored one analog value. That required SLEDS frame refresh to be done progressively (e.g. one pixel at a time). On the other hand, our applications require "snapshot" mode whereby the entire frame changes simultaneously.

In future SLEDS RIIC, each pixel stores four analog values and two digital bits. Two analog values store the two colors currently being displayed and another two analog values store the two colors currently being loaded. The two digital bits decide whether the high- or low-current transistor is used to drive the SLEDS device.

An added benefit of adding snapshot mode in future SLEDS is reduction in the time when the projector is displaying "invalid" or "partially-formed" images. In future SLEDS, this "invalid" time is limited to the brief period when LOAD signal is asserted to load a fully formed frame image into all 512x512 SLEDS devices.

- **In-pixel CMOS address logic**

In SLEDS RIIC, address logic in the pixel was implemented using pass transistors. Although this approach required a small number of transistors (e.g. 2 PMOS and 2 NMOS transistors), it created an unintended "charge-pump" problem, whereby pixels in the row that was being "written-to" would over time change their stored value (and hence the pixel light output would also change).

In future SLEDS RIIC, the address logic in the pixel uses CMOS logic. CMOS logic completely eliminates “charge pump” problem because it is static logic rather than dynamic logic.

- **More analog channels**

In SLEDS RIIC, there are four analog drive channels. In future SLEDS RIIC, the number of analog drive channels is increased to 32. Having more analog channels enables us to achieve 1KHZ refresh rate.

- **CMOS technology upgrade**

SLEDS RIIC was built using 3-level metal 0.5 micron high-voltage CMOS process from ON Semiconductor. Future SLEDS RIIC uses the same technology but adds two additional metal layers. Based on actual pixel layout, this upgrade doubles the width of metal buses to bring power to SLEDS devices (from 75 microns in SLEDS to 150 microns in future SLEDS). This doubles the power bus capability in future SLEDS RIIC over SLEDS RIIC.

- **CSE board upgrades**

The SLEDS projector uses an ML605 FPGA board with two FMC expansion slots. One of these slots connects to half-dozen discrete boards that together produce four analog drive channels for driving the SLEDS RIIC. The other slot connects to custom digital board that produces dozen digital drive channels for driving the SLEDS RIIC.

In future SLEDS, the ML605 board is being replaced with SUPER-FPGA board (also known as TB-6V-LX760-LSI). The SUPER-FPGA board has ten FMC

expansion slots and can be scaled up to support larger SLEDS arrays, higher refresh rates or more colors by connecting multiple SUPER-FPGA boards. The future SLEDS board uses a larger FPGA with double memory and logic capacity than that the FPGA used in SLEDS projector.

Also in future SLEDS, the half-dozen discrete boards that produce four analog drive channels are being replaced with a single compact board that connects into an FMC slot. With this approach, one SUPER FPGA has the potential to drive 40 analog drive channels at pixel rates of up to 50MHz.

An added benefit of new CSE approach is higher integration of components and reduced reliance on discrete cabling and avoidance of ground loops. This should result in less power supply noise issues that were observed with SLEDS projector.

- **NUCPUTER**

In SLEDS projector, NUC operation is done in the FPGA. On the other hand, we have experimentally determined that SLEDS devices need a more sophisticated NUC than a linear fit approach that FPGA circuits have the ability to implement. In SLEDS projector, this limitation required us to “disable” approximately 10 percent of the SLEDS array because although the pixels were operational, we were not able to properly NUC them.

In future SLEDS projector, NUC operation is being moved to a separate computer and implemented using an NVIDIA GPU to allow the full use of the array.

BIBLIOGRAPHY

[1] Rodney McGee. Design, fabrication, iteration, and testing of the 68x68 superlattice led infrared projections system and associated electronics. Master's

[2] Christopher Kerwien. Design and characterization of 12V analog transimpedance amplifier for superlattice LED projector. Bachelor's

[3] Norton, D. T., Olesberg, J. T., McGee, R. T., Waite, N. A., Dickason, J., Goossen, K. W., Kiamilev, F. ... & Boggess, T. F. (2013). 512 512 Individually Addressable MWIR LED Arrays Based on Type-II InAs/GaSb Superlattices. *Quantum Electronics, IEEE Journal of*, *49*(9), 753-759.

[4] Lange, C., McGee, R., Waite, N., Haislip, R., & **Kiamilev, F. E.** (2011, May). System for driving 2D infrared emitter arrays at cryogenic temperatures. In *SPIE Defense, Security, and Sensing* (pp. 801507-801507). International Society for Optics and Photonics.

[5] Flip Chip bonding of 68 x 68 MWIR LED Arrays, Naresh Das, Monica Taysing-Lara, Kimberley Olver, Fouad Kiamilev, J. Prineas, J. Olsberg, E. Kroepeck, L. Murray and T. Boggess, IEEE Trans. on Elect. Packaging Manufacturing, Vol. 32, No. 1 (January 2009)

[6] John P. Uyemura. *CMOS Logic Circuit Design*. Kluwer Academic Publishers, 2002.

[7] C5: 0.5um Proces Technology. http://www.onsemi.com/PowerSolutions/content.do?id=16693, February 2011.

[8] The MOSIS Service Website. http://www.mosis.com/on_semi/c5/, February 2011.