|  |
| --- |
| Capstone |
| [Type the document subtitle] |
|  |
| **Travis Berger and Luis Santiago**  **Corporate Sponsor: Tony Mulienberg Intel**  **Academic Advisor: Dr. James Morris PSU** |
|  |

Acknowledgements

Abstract

Table of Contents

##### Introduction

This project was a collaboration between Intel and Portland State University. Intel developed a validation board called the Education Engagement Electrical Validation Board (E3VB). The purpose of this board was to teach engineering students and members of industry about signal integrity issues through hands on experiments. The goal of this project was to improve upon the E3VB but designing two additional experiments that can be added to the board in a future revision. Intersymbol interference, crosstalk, and proper mixed signal return path techniques were the signal integrity issues were the main focus of these experiments. To best highlight these issues two separate experiment boards were chosen one for ISI and Crosstalk and the other board for the proper mixed signal return path techniques. Due to the chosen audience a requirement for the experiments was to only utilize basic equipment available to electrical engineering students and industry members. This equipment included a function generator, oscilloscope, and a power supply. Knowledge of how to use this equipment is assumed. Both experiments have onboard clocks and have the option of being powered by micro USB or external power supply. Since the intended audience includes college students cost was a factor when designing this experiment, so it was decided that each experiment board could not exceed $100.00 in cost.

##### Background

###### ISI background

Intersymbol interference is a signal integrity issue that occurs in dispersive channels. Intersymbol interference occurs when a signal affects subsequent signals making the communication less reliable. Factors that contribute to ISI are jitter, crosstalk, simultaneous switching, reflections and more. When designing a communication medium it is important to take into consideration the properties of the transmission line and the component used to transmit the signal. The compounding effects of ISI over time can   
  
Crosstalk Background

###### MSG background

When dealing with mixed signal applications the routing of return paths becomes really important. There isn’t a very clear method on how to handle the return paths for mixed signal devices. Some people recommend that a separate ground and power supply are usually required when dealing with sensitive analog circuits [ref num staying well grounded]. Other engineers prefer to partially isolate the analog circuity from the flow of the return path from the digital circuity. In order to properly design mixed signal applications the designer needs to understand where all the current paths flow through the respective ground planes [ref num2 partioning and layout of mixed sig pcb]. For the inexperienced designer this can be a very difficult problem to overcome.

# **Timeline**

This project started with a projected timeline of ten weeks to deliver the agreed upon deliverables. This time period was broken down into four main phases. The phases were design, implementation, testing, and pass down preparation. Design was allotted three weeks, implementation was allotted one week, testing was allotted three weeks, and pass down preparation was allotted one week. Due to obligations outside of the project each week accounts for roughly twenty – thirty engineering hours per week. Over the course of the project these estimated deadlines shifted around due problems that arose.

**Project Management**

The project started without roles clearly defined. The group would meet twice weekly once with our advisor and once just as a team. At team meetings we would propose ideas and assign tasks to each team member to have done by specific deadlines. Over the course of the project team members fell into natural roles based on their proficiency at certain tasks. For example due to Luis’ background with software he wrote Verilog code to test the ISI experiment. This task given to another teammate would have been much more difficult and time consuming. Similarly most of the documentation was done by Travis because English is his first language and he can write a document much faster than Luis. With the loss of two team members over the course of this project, assigning tasks based on proficiency was the only way to meet key deadlines. Once one team member finished a task the other team member would look over the completed work and help catch errors and make changes.

##### **Design**

The problem this project attempted to solve was pretty open ended, so the design phase started with brainstorming possible signal integrity experiments. Two experiments were chosen, mixed signal ground techniques and an experiment dealing with intersymbol interference and crosstalk.   
  
Mixed Signal Return Path Techniques (Solidify name at some point)

This experiment consists of both digital and analog circuity. The point of this experiment was to highlight signal integrity issues that occur due to improper return paths in mixed signal applications. To best illustrate this the digital portion of the circuit needed to be noisy. The analog portion needed to represent sensitive analog circuity but still be easy and intuitive to use with the constraint that the whole board be less than $100.00. For the digital portion the design consisted of a 10 MHz clock and three Hex inverter packages. Each package contained six inverters driving a resistive load. The idea behind this design was the inverters in parallel would cause a big spike when switching. The resitive load was determined to be 1350 Ω because this caused the inverter to drive max current at about 3 mA. For the analog portion of the circuit a basic NE5532 operation amplifier was chosen. This was due to familiarity with intended audience and flexibility to create several different circuit topologies. Jumpers with different component values were used in place of stationary components. This allows the user to select what type of operation the op amp will perform. Figure 1 shows the two different return paths for the digital circuit. Return Path A goes straight back to the power supply. Return Path B goes directly through the analog portion of the board. Figure 2 shows the schematic for the experiment The schematic was created in CadSoft EAGLE software. This software was also used to create the board layout.

Digital Circuity With

Semi-Isolated Ground Plane

Return Path A

Return Path B

Analog Circuity

Source

Isolation

Figure 1: Diagram showing different return paths for mixed signal ground experiment.

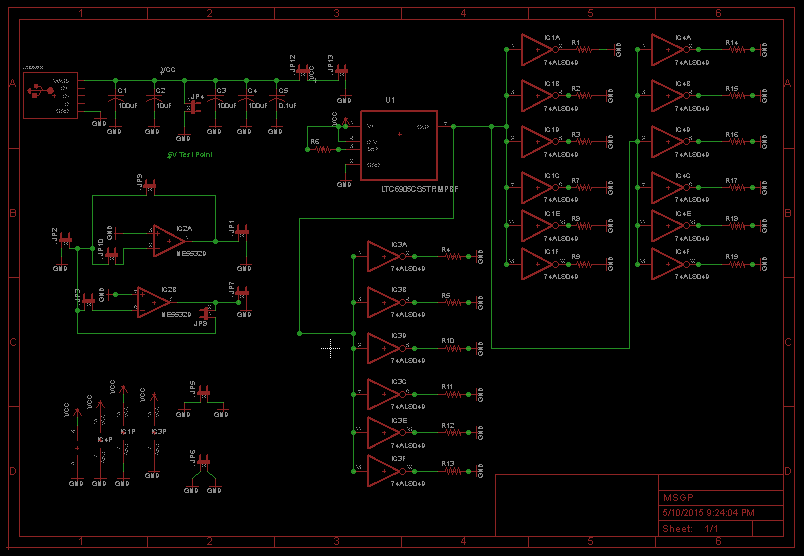


Figure 2: CadSoft EAGLE schematic for mixed signal ground technique experiment.

###### ISI/Crosstalk experiment:

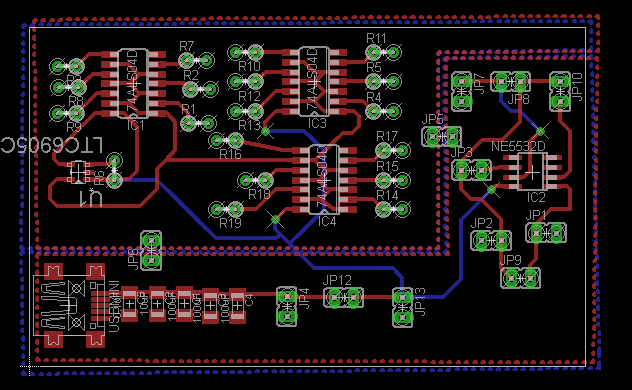
The ISI experiment with the intention to incorporate some of the concepts form both phenomena into one single experiment. Our sponsor wanted to be able to measure crosstalk at more than one line away. We began by identifying the characteristics that the user can easily change in order to she characteristic of the signal on the transmission line. We decided to use five traces to measure the crosstalk. One of the the traces is design with variable passive components capacitance and inductance can be changed to observe the change on the signal

In order to make the experiment more realistic and we added five Linear feedback shift register (LFSR) using XNOR to create pseudo random pattern where the user may need to observe the signal for a longer period of time in order to find any issues.

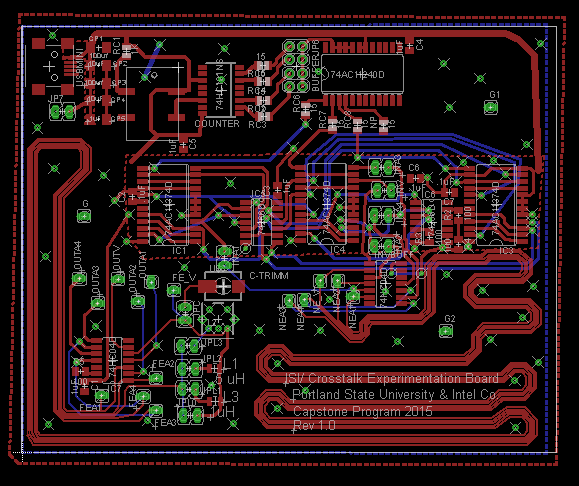
In order to create crosstalk we are using a clock with a frequency between 1-8MHz. All five traces were design to be .6 mil width and .6 mil apart each one with an approximate length of 26 inches to maximize the effect

##### **Implementation**

After the design phase was complete the design was implemented by creating board layouts to have fabricated at OSH Park. CadSoft EAGLE was also used to create the layout for both boards. The layout for the ISI experiment was particularly challenging given the need to route five <insert length> traces with 6 mil of separation between them. This combined with intentionally breaking DRC rules presented some obstacles. Figure 3(a) shows the layout for the mixed signal ground techniques experiment and Figure 3(b) shows the layout for the ISI/Crosstalk experiment.



(a)



(b)

Figure 3: (a) CadSoft EAGLE layout for the mixed signal ground techniques experiment. (b) CadSoft EAGLE layout for the ISI/Crosstalk experiment.

When the boards came back from OSH Park the majority of the components were placed using solder paste and a reflow oven. All the through hole parts were soldered by hand using a soldering iron. Once the boards were completed testing of the integrated boards started immediately.

Bill of Materials





##### **Testing**

The testing of these experiments was broken down into stages. The first stage of testing was module testing. For module testing each individual module was soldered to a breakout board and tested individually. Once modules were found to be working correctly they were then integrated together for integration testing. In this phase of testing the modules were connected together and tested to determine performance. <rephrase to represent test plan but not like it was actually done>. Unfortunately testing was pushed back due to an unexpected change in resources. This resulted in a few errors slipping by and not being caught until the layouts were already sent to fabrication. On the mixed signal ground techniques experiment the clock was found to work at 3.3 V instead of 5 V and no external resistance was needed to set the divisions. This error couldn’t be fixed on the fabricated board so the clock was left out of this experiment and an external clock was used as proof of concept. On the ISI/Crosstalk experiment the XNOR<double check?> needed a pull up resistor and a capacitor to ground. Another problem that was encountered was the wrong type of mini usb connector was ordered for the power supply. A Mini USB A was ordered which is really hard to find a connector for. Most Mini USB connectors are Mini USB B. To overcome this problem an external power supply was connected to the board.

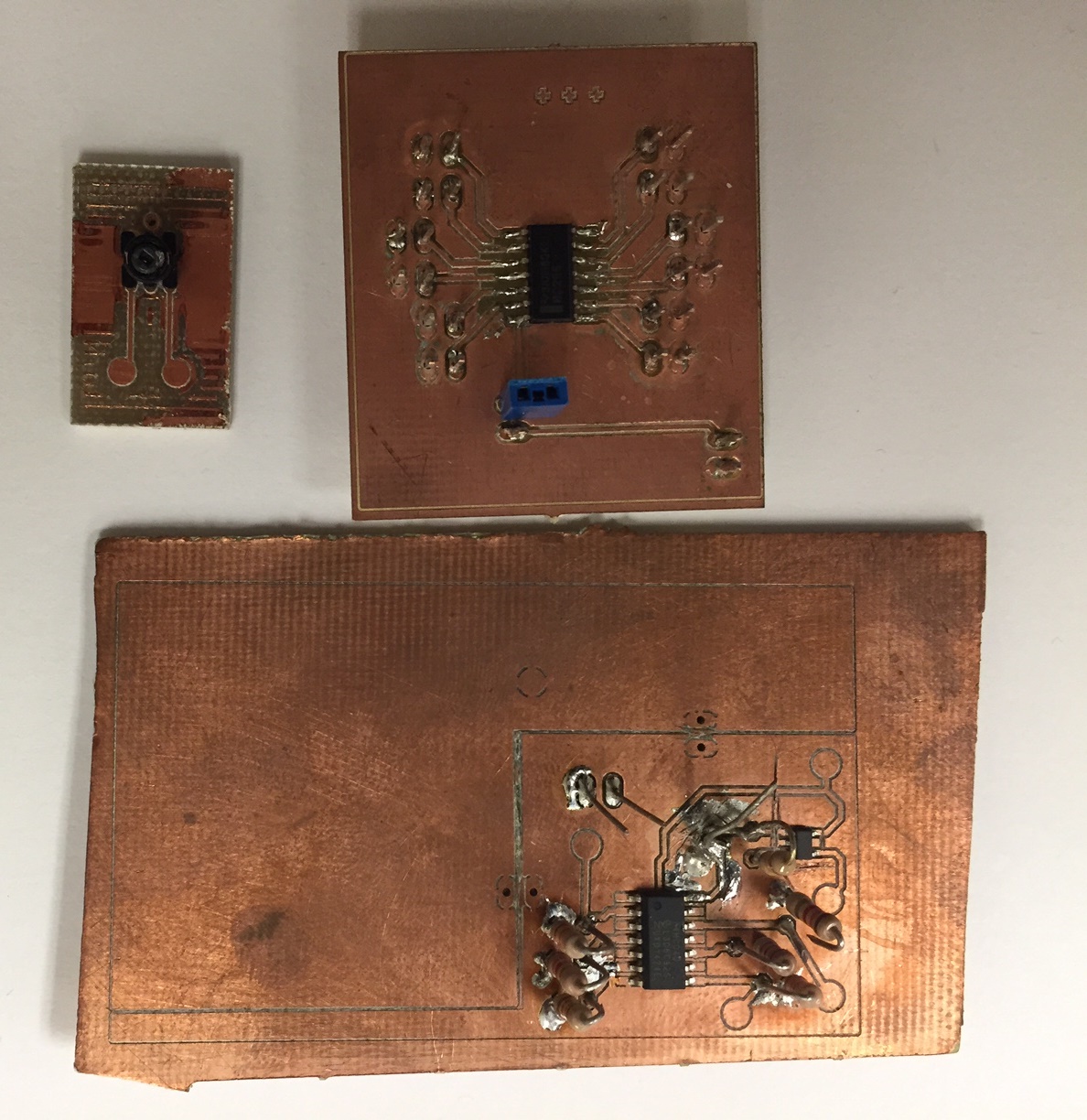
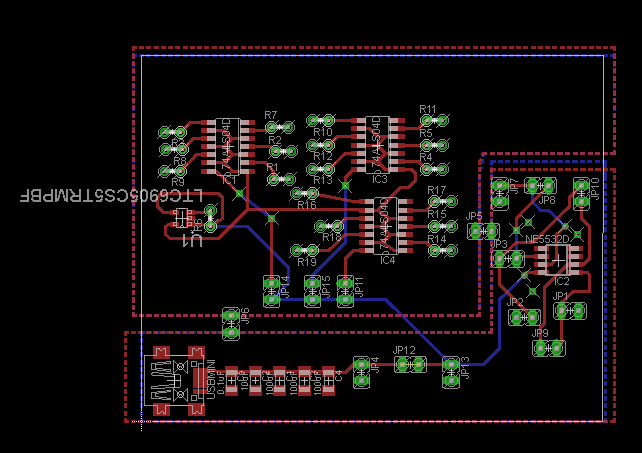


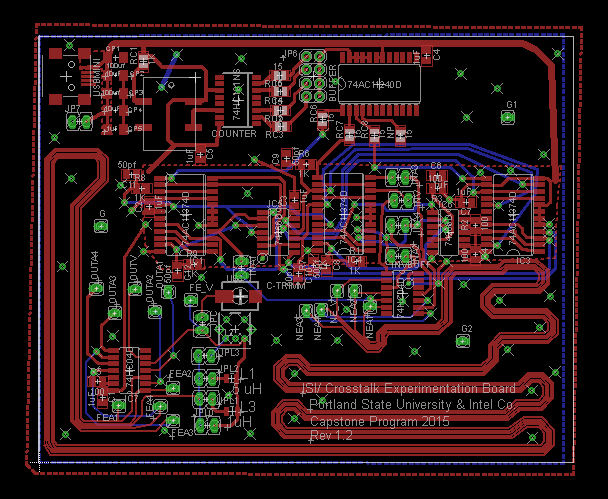
Figure <> : Breakout boards used for module and integration testing.

##### **Pass Down Preparation**

Since this project will be carried on by a different group of students and/or engineers it was important to prepare the project to be passed down successfully. This preparation included updating all schematics and layouts to reflect final design, fixing all the errors found during testing, and updating documentation. The main changes to the MSGT experiment were making the board larger, allowing the ability to turn on each individual inverter to see the effects of one, two, or three inverters on analog signal, updated clock to work with 5 V, and changed the topologies of one of the op amps to allow for more diverse operations. The main changes in the ISI experiment were to add the pull up resistor and bypass capacitor for the XNOR in the layout. The pass down documentation consisted of setup and instructions for experiments, noticeable design decisions, and improvements that still need to be made to the experiments.



(a)

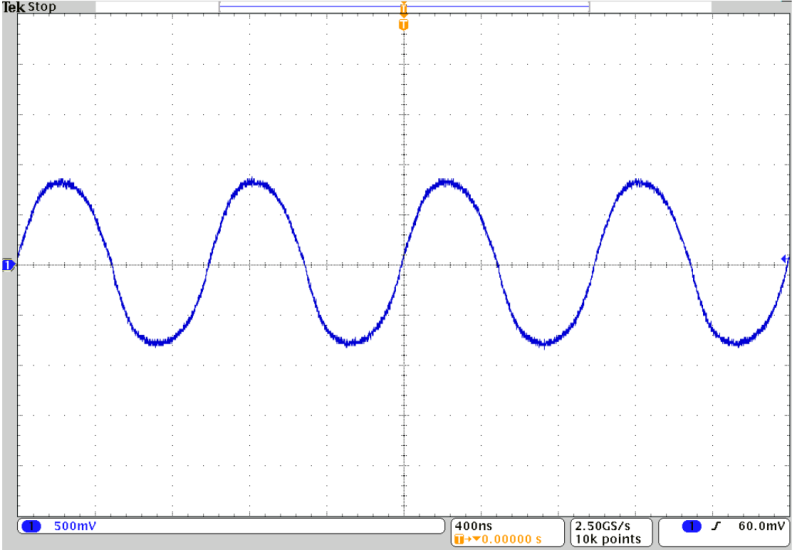


(b)

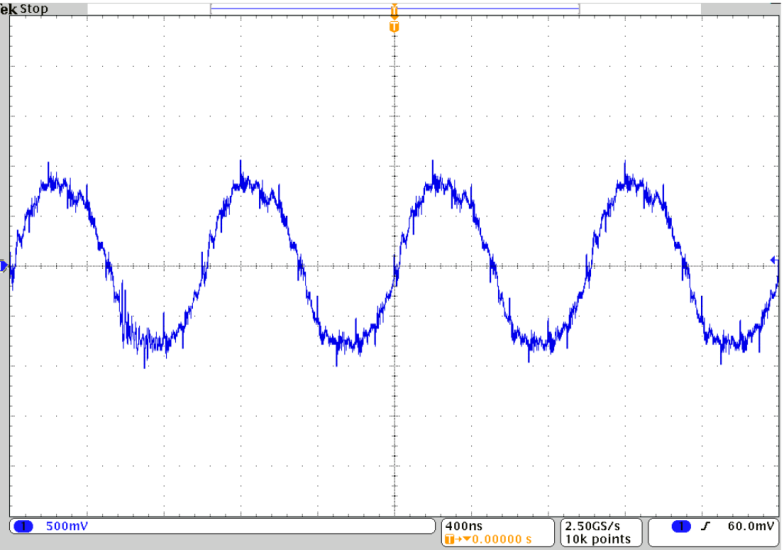
Figure <>: (a) the updated MSG experiment layout included in the pass down documents. (b) updated ISI experiment layout included in the pass down documents.

**Results**

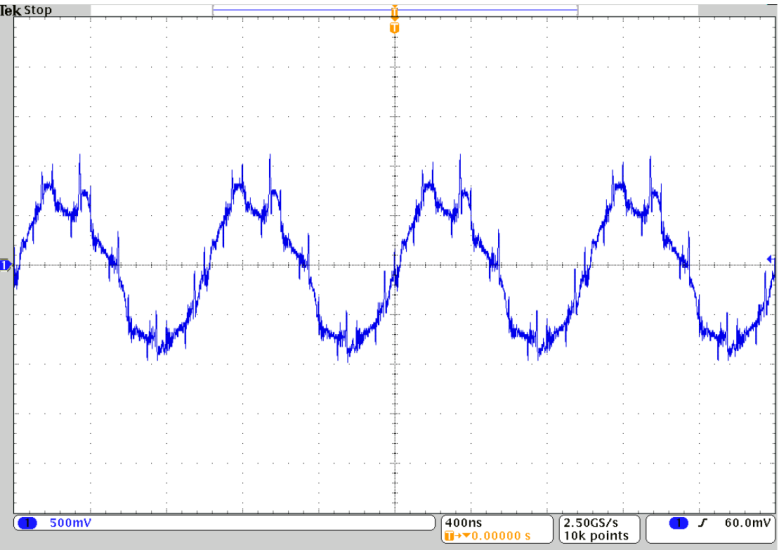
After testing and modifying the experiments both boards were successful in generating the results that were designed for. The MSG experiment was setup with an external signal generator driving a 5 MHz square wave with 5V amplitude in place of the clock. The analog signal was also provided by an external signal generator with a 2V amplitude at 1 MHz. Figure <> (a) shows the output of the analog signal without any interference from the digital signal. Figure <> (b) shows the output of the analog signal while totally isolated from the digital signal. Despite being isolated from the digital signal the effect of the two planes coupling distorts the signal. Figure<> (c) shows the signal with the return path of the digital circuit running through the analog circuit. This heavily distorts the analog output which shows the importance of correct ground techniques.



(a)

****

(b)

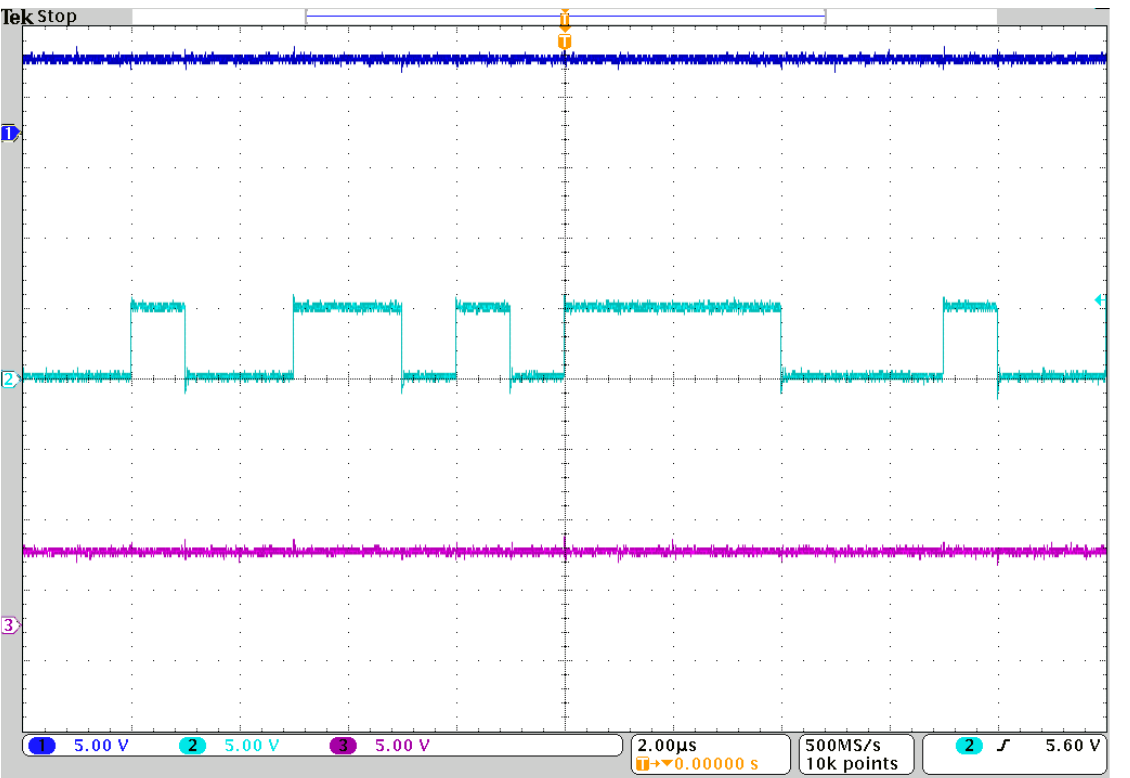
****

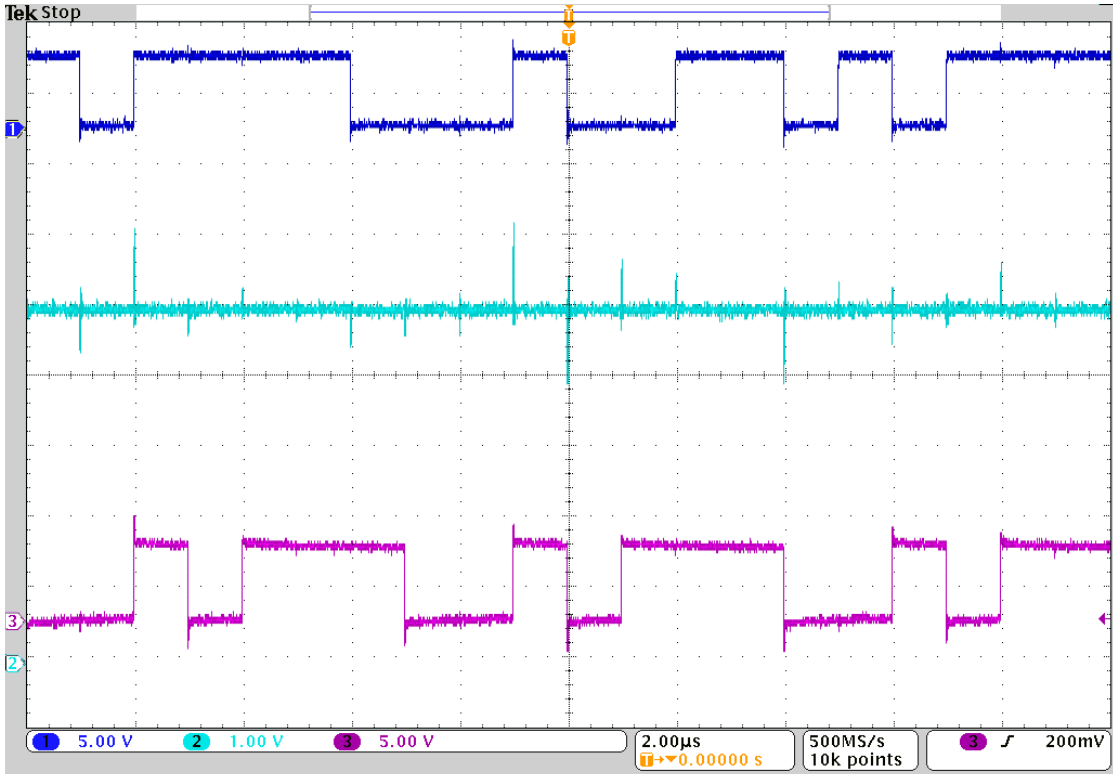
(c)

Figure <> (a) Analog signal without digital interference. (b) Analog signal with isolated ground planes. (c) Analog signal with return path routed through analog ground plane.

**ISI (Sub header under Results)**

ISI RESULTS

****

****

**Conclusion**

The goals of this project were met by designing and implementing a mixed signal ground techniques experiment and a new ISI/Crosstalk experiment. With modifications both experiment boards worked and the team was able to demonstrate these working experiments to the corporate sponsor. The pass down preparation was essential to ensuring that this project will continue with future engineers and students in the future.

Throughout this process many lessons were learned. The first lesson learned was that you can’t possibly plan for everything so your time schedule needs to account for these unplanned surprises. This was seen numerous times throughout the project most prominently in the change of scope in week 9, losing two team members, and a team member coming down with pneumonia. The second lesson learned was not to ship layouts out to the manufacturer until all testing is complete. This was a tough decision for us because we wanted to get the boards back in time to test and document, but ultimately we spent just as much time debugging and correcting the errors then we would have if we stuck to our test plan and sent the boards out a week late. Despite these obstacles the team was able to keep moving forward and made progress on the project. The most important lesson learned by this group was how to fail and keep moving forward. The final team members have a history of doing well in school especially excelling in academic projects. This project while successful did not win any awards at the poster ceremony. This was a hard lesson for the group but an important one because in life you can’t always be the best but you need to stand by your work and be proud of the accomplishments of your peers.

**References**

Appendix A

Verilog code

// Capstone Project ISI

// generates the LFSR signal for

// all five lanes used and compare output

// to maximize patterns

module sig\_gen (

lout , // Output of the counter

count0 , // lfsr 0 count

count1 ,

count2 ,

count3 ,

count4 ,

enable , // Enable for counter

clock , // clock input

reset , // reset input

INIT\_0,

INIT\_1,

INIT\_2,

INIT\_3,

INIT\_4

);

//------------- Parameter -----------

parameter LANES =5,

WIDTH= 4,

WIDTH2=3;

//-------------Input ----------------

input clock, enable, reset;

input [WIDTH-1:0] INIT\_0;

input [WIDTH-1:0] INIT\_1;

input [WIDTH-1:0] INIT\_2;

input [WIDTH2-1:0] INIT\_3;

input [WIDTH2-1:0] INIT\_4;

//------------OUTPUT -----------------

output [LANES-1 :0] lout; // store single

output [WIDTH-1:0] count0;

output [WIDTH-1:0] count1;

output [WIDTH-1:0] count2;

output [WIDTH2-1:0] count3;

output [WIDTH2-1:0] count4;

//--------- internal variables -------

wire [LANES-1 :0] pattern\_out;

wire [WIDTH-1:0] value0;

wire [WIDTH-1:0] value1;

wire [WIDTH-1:0] value2;

wire [WIDTH2-1:0] value3;

wire [WIDTH2-1:0] value4;

// varables for DFF delays

wire q1,q2\_1,q2\_2,q3\_1,q3\_2,q3\_3,q4\_1,q4\_2,q4\_3,q4\_4;

// ---------- assignments --------------

// connects out and lout (single bit current value)

assign lout[0]= pattern\_out[0] , // 0 delay LSFR0

lout[1]= q1 , // delayed by 1 on LSFR1

lout[2]= q2\_2 , // delayed LSFR2 by 2 LSFR3

lout[3]= q3\_2 ,

lout[4]= pattern\_out[4] ;

assign count0=value0,

count1=value1,

count2=value2,

count3=value3,

count4=value4;

// LSFR instantiations

// out = 4 bit lsfr value forwarded to testbench

// init inital value forwarded to tesbench ( assign on reset=1)

// line out is sinble bit value assign to

lfsr L0(.clk( clock),.reset( reset), .enable( enable),.lineout( pattern\_out[0]),.in(INIT\_0),.out(value0));

lfsr L1(.clk( clock),.reset( reset), .enable( enable),.lineout( pattern\_out[1]),.in(INIT\_1),.out(value1));

lfsr L2(.clk( clock),.reset( reset), .enable( enable),.lineout( pattern\_out[2]),.in(INIT\_2),.out(value2));

lfsr3b L3(.clk( clock),.reset( reset), .enable( enable),.lineout( pattern\_out[3]),.in(INIT\_3),.out(value3));

lfsr3b L4(.clk( clock),.reset( reset), .enable( enable),.lineout( pattern\_out[4]),.in(INIT\_4),.out(value4));

// D-FF instantiation to create delays

DFF dff1 (.d(pattern\_out[1]),.q(q1),.clock(clock),.enable(enable),.reset(reset));

DFF dff2\_1 (.d(pattern\_out[2]),.q(q2\_1),.clock(clock),.enable(enable),.reset(reset)); // get out 2 and delay by 1

DFF dff2\_2 (.d(q2\_1),.q(q2\_2),.clock(clock),.enable(enable),.reset(reset)); // delay out by 1 again

DFF dff3\_1 (.d(pattern\_out[3]),.q(q3\_1),.clock(clock),.enable(enable),.reset(reset)); // delay out 3

DFF dff3\_2 (.d(q3\_1),.q(q3\_2),.clock(clock),.enable(enable),.reset(reset));

//DFF dff3\_3 (.d(q3\_2),.q(q3\_3),.clock(clock),.enable(enable),.reset(reset));

DFF dff4\_1 (.d(pattern\_out[4]),.q(q4\_1),.clock(clock),.enable(enable),.reset(reset)); // delay #4

/\*DFF dff4\_2 (.d(q4\_1),.q(q4\_2),.clock(clock),.enable(enable),.reset(reset));

DFF dff4\_3 (.d(q4\_2),.q(q4\_3),.clock(clock),.enable(enable),.reset(reset));

DFF dff4\_4 (.d(q4\_3),.q(q4\_4),.clock(clock),.enable(enable),.reset(reset));

\*/

endmodule

//-----------------------------------------------------

// Design Name : TOP

// File Name : lsfr\_testbench.v

// Function : Find input to LSFR that provides

// max out combination

//-----------------------------------------------------

module top();

// ------- variables --------------

parameter WIDTH= 4,

WIDTH2=3,

LANES=5,

LOOP\_MAX=6'b100000;

reg clk;

reg reset;

reg enable;

integer j;

integer combinations;

//-------- initial values for each LSFR

reg [WIDTH:0] INIT\_0;

reg [WIDTH:0] INIT\_1;

reg [WIDTH:0] INIT\_2;

reg [WIDTH:0] INIT\_3;

reg [WIDTH:0] INIT\_4;

// store number of time a case happend

reg [LANES:0] ocurrence [(2\*\*LANES)-1:0];

// store count of LSFR only for visual purpose not really needed

wire [WIDTH-1 : 0] lsfr1;

wire [WIDTH-1 : 0] lsfr2;

wire [WIDTH-1 : 0] lsfr3;

wire [WIDTH2-1 : 0] lsfr4;

wire [WIDTH2-1 : 0] lsfr5;

// Stores the concatenated value of the

// single bit output of

wire [LANES-1 : 0] pattern;

initial begin

clk = 0;

// ---------7 lines immediately below: only used if for loops are commented to assign preset initial values

reset = 1;

enable = 0;

INIT\_0=4'b0000;

INIT\_1=4'b0000;

INIT\_2=4'b0000;

INIT\_3=3'b0000;

INIT\_4=3'b000;

//------------------------ Code to generate unique input values-------------------------------

#1 reset = 1; // reset ensures init value is stored

#1 enable = 0; // ensure it wont change util initialization is complete

// set all occurence to 0

for (j=0; j< 2\*\*LANES ; j=j+1) begin

ocurrence[j]=0;

end //end for loop

// enables signals

#1 combinations=0;

#1 reset = 0;

#1 enable = 1;

// once done check number of unique combinations genarated

// Note: clock period is 2

#1000 for (j=0; j< 2\*\*LANES ; j=j+1) begin

if (ocurrence[j]>0)

combinations=combinations+1;

end //end for loop

// display only if it found more than 24

#1 if (combinations>=24)begin

$display ("\n %d different combinations with %d, %d, %d, %d, %d\n\n",combinations,INIT\_0,INIT\_1,INIT\_2,INIT\_3,INIT\_4);

end //else begin

for (j=0; j< 2\*\*LANES ; j=j+1) begin

$display("pattern %d \tappeared %d \n",j, ocurrence[j] );

end //end for loop

#1 $finish;

end

//---------------------------- clock ---------------------------

always #1 clk = ~clk;

//--------------------- module instantiation -------------------

sig\_gen U1(

.lout (pattern) , // Output of the counter

.enable (enable) , // Enable for counter

.clock (clk) , // clock input

.reset (reset) , // reset input

.count0( lsfr1) ,

.count1(lsfr2 ) ,

.count2(lsfr3 ) ,

.count3(lsfr4 ) ,

.count4(lsfr5 ),

.INIT\_0(INIT\_0[WIDTH-1:0]),

.INIT\_1(INIT\_1[WIDTH-1:0]),

.INIT\_2(INIT\_2[WIDTH-1:0]),

.INIT\_3(INIT\_3[WIDTH2-1:0]),

.INIT\_4(INIT\_4[WIDTH2-1:0])

);

//------------------------ keeps track of times a pattern appears ------------

// at neg edge keeps track of how many

// times each pattern occured

always @(negedge clk) begin

if(pattern<32)

ocurrence[pattern]= ocurrence[pattern]+1;

end //end of always statement

endmodule

//-----------------------------------------------------

// Design Name : lfsr

// File Name : lfsr.v

// Function : Linear feedback shift register

//-----------------------------------------------------

module lfsr (

in,

out , // Output of the counter

lineout , // output to line

enable , // Enable for counter

clk , // clock input

reset // reset input

);

parameter WIDTH= 4;

//----------Output Ports--------------

output [WIDTH-1:0] out;

output lineout;

//------------Input Ports--------------

input [WIDTH-1:0] in;

input enable, clk, reset;

//------------Internal Variables--------

reg [WIDTH-1:0] out;

reg lineout;

wire linear\_feedback;

//-------------Code Starts Here-------

assign linear\_feedback = !(out[0] ^ out[1]);

always @(posedge clk)begin

if (reset) begin // active high reset

out <= in ;

end else if (enable) begin

out <= {linear\_feedback,out[3]

,out[2], out[1]};

// stores single bit output

// from LSFR

lineout <= out[1];

end

end

endmodule // End Of Module counter

module lfsr3b (

in,

out , // Output of the counter

lineout , // output to line

enable , // Enable for counter

clk , // clock input

reset // reset input

);

parameter WIDTH= 3;

//----------Output Ports--------------

output [WIDTH-1:0] out;

output lineout;

//------------Input Ports--------------

input [WIDTH-1:0] in;

input enable, clk, reset;

//------------Internal Variables--------

reg [WIDTH-1:0] out;

reg lineout;

wire linear\_feedback;

//-------------Code Starts Here-------

assign linear\_feedback = !(out[0] ^ out[1]);

always @(posedge clk)begin

if (reset) begin // active high reset

out <= in ;

end else if (enable) begin

out <= {linear\_feedback ,out[2], out[1]};

// stores single bit output

// from LSFR

lineout <= out[1];

end

end

endmodule // End Of Module counter

// D-FF

module DFF (d,q,clock,enable,reset);

//----------Output Ports--------------

output q;

//------------Input Ports-------------

input d,enable, clock, reset;

//------------Internal Variables--------

reg q;

always @(posedge clock) begin

if (reset) begin

q <= 0;

end else if (enable) begin

q<=d;

end

end // end always block

endmodule // end DFF modul

Bill of Material



