UNIVERSITY OF WATERLOO | FACULTY OF ENGINEERING
Department of Electrical and Computer Engineering

aspIR

Final Report

Group 34:

- Zayd Abbas – 20888149
- Anderson Hsieh – 20875033
- Andy Lang – 20902905
- Justin Llanos – 20819233

Project Consultant: Prof. Shiyu Su

Submitted: Feb 24th, 2025

Abstract

In high EMI environments, such as industrial settings with high-power machinery or medical settings with sensitive test equipment, traditional wireless communication systems like WiFi or Bluetooth can be prone to interference. A common solution for this is to use fiber optics to transmit data between sensors. This solution requires laying fiber which can introduce its own costs and technical effort. Our project seeks to provide an alternative by making use of a much more resilient part of the spectrum - infrared. By assuming changes in the ambient infrared are slower than several kilohertz we can emit a modulated signal directed at a receiving photo-detector. We apply the modulation schemes of traditional RF communication but replacing the antennas with an LED and photodiode to add the benefits of EMI resilience and line of sight security. Key features of the project include an FPGA-based modem, a transconductance amplifier to drive the LED at high frequency, and a receiving transimpedance amplifier to read the signal out of the photodiode.

Acknowledgments

We would like to express our deepest gratitude to Prof. Shiyu Su for his exceptional guidance and unwavering support as our project supervisor. His invaluable insights and expertise have been instrumental in the success of this work. We would also like to extend our heartfelt thanks to Xiaofei Zhang from the Emerging Radio Systems Group for his assistance in setting up our FPGA PLL configuration. His technical proficiency and dedication were crucial in overcoming the challenges we faced during this project. This accomplishment would not have been possible without their contributions.

Contents

List of Figures

List of Tables

1 High-Level Project Description

1.1 Motivation

In high EMI environments, such as industrial settings with high-power machinery or medical settings with sensitive test equipment, traditional wireless communication systems like WiFi or Bluetooth can be prone to interference. A common solution for this is to use fiber optics to transmit data between sensors. This solution requires laying fiber which can introduce its own costs and technical effort. We seek to provide an alternative solution using line-of-sight optical communication.

1.2 Project Objective

We propose a one-chip solution for an over-the-air infra-red transceiver. In addition to the chip, the potential customer's BOM would only include the IR photodiode and LED, a power supply and their peripheral sensors. To demonstrate proof of concept, we implement the analog interface to the diodes on a PCB using op-amps; the digital portion including the modem are implemented on FPGA; and off-the-shelf data converters are used.

1.3 Block Diagram

Figure 1.1 shows the top-level system block diagram. The digital block includes the DSP and the Host Communication subsystem, whereas the analog block includes all the analog subsystems. The blocks that are our custom designs are highlighted by the red boxes.

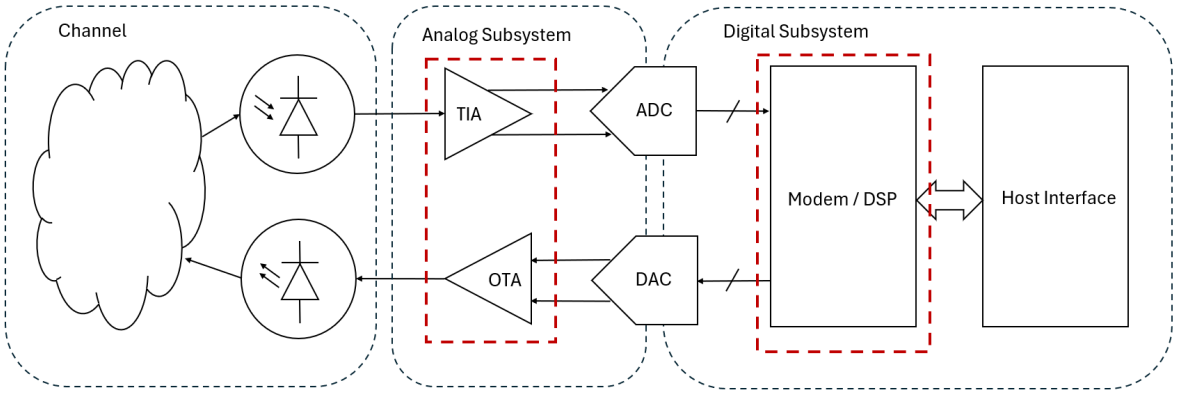

Figure 1.1: System Block Diagram

1.3.1 Analog Subsystem

The analog subsystem comprises the transmit and receive paths as well as the channel. The transmit path from the DAC comprises an OTA with integrated lead compensation. The receive path implements a TIA and single-ended to differential conversion to drive the ADC. The channel is a combination of LED, photodiode, and air.

1.3.2 Digital Subsystem

The digital subsystem can be separated into the DSP modem and the communication interface with the host PC. The DSP modem is implemented inside of the programmable logic in the Zynq-7000 SoC. The modulator is responsible for converting digital signals into phase-shifting key modulated signals, and vice versa for the demodulator. The host communication interface enables the DSP modem to communicate with the host computer through an Ethernet connection. The main driver of this interface is the Arm Cortex-A9 processors inside of Zynq-7000 SoC. The PZ7Z020-SOM also supports Ethernet connection which comes in handy for our first prototype testing. Since we have no degrees of freedom on the channel once the diodes are selected, we are required to compensate the channel non-linearities with digital predistortion.

2 Project Specifications

The project specifications are divided into functional and non-functional specifications and each

2.1 Functional Specifications

	Specification	Importance
Data Rate	$> 50\,\mathrm{Mbit\,s^{-1}}$	Critical
Range	0.5 m to 10 m	Critical
Signal Center Frequency	50 MHz	Flexible

Table 2.1: Functional Specifications

	Specification	Importance
Modulation	N-PSK	Flexible
Host Interface	Ethernet	Flexible
Bit Error Rate	$< 10^{-5}$	Flexible

Table 2.2: Non-Functional Specifications

2.2 Non-Functional Specifications

3 Detailed Design

3.1 TIA & ADC

The receive (Rx) path converts the photo-generated current of the photodiode to a differential voltage and outputs to the ADC. The TIA and ADC driver are implemented using op-amps (LTC6268-10 [1] and LTC6400-26 [2] respectively). A hazardous non-ideality for the TIA op-amp is it's input impedance. LTC6268-10 has an input resistance on the order of TΩ [1] so we consider it ideal. Figure 3.1 shows the Rx path including a first TIA stage around LTC6268-10, single to differential conversion and alias filter around LTC6400-26 and the ADC. The ADC output is passed to the FPGA for de-emphasis and demodulation.

Equation 3.1 is obtained from hand analysis of the TIA stage where the LTC6268-10 is modeled having a finite gain,A_0 and finite bandwidth, ω_0.

$$A(s) = -\frac{R_F}{\left(1 + \frac{1}{A_0}\right) + \frac{s}{A_0 \omega_p}} \tag{3.1}$$

Thus we control gain via the feedback resistance R_F.

A naive approach to current-voltage conversion is to use a resistor, R, the trans-impedance of such a circuit would be R, the same as the proposed design. In order for the input current to the Rx path to be as close to the diode's short circuit current as possible, an important parameter is the TIA's input resistance. The input resistance of the proposed design follows from Miller's Theorem in equation 3.2.

$$R_{in} = \frac{R_F}{1 + A_0} \tag{3.2}$$

That is the design provides the advantage of attenuating the input resistance by the op-amp's open loop gain at the cost of bandwidth.

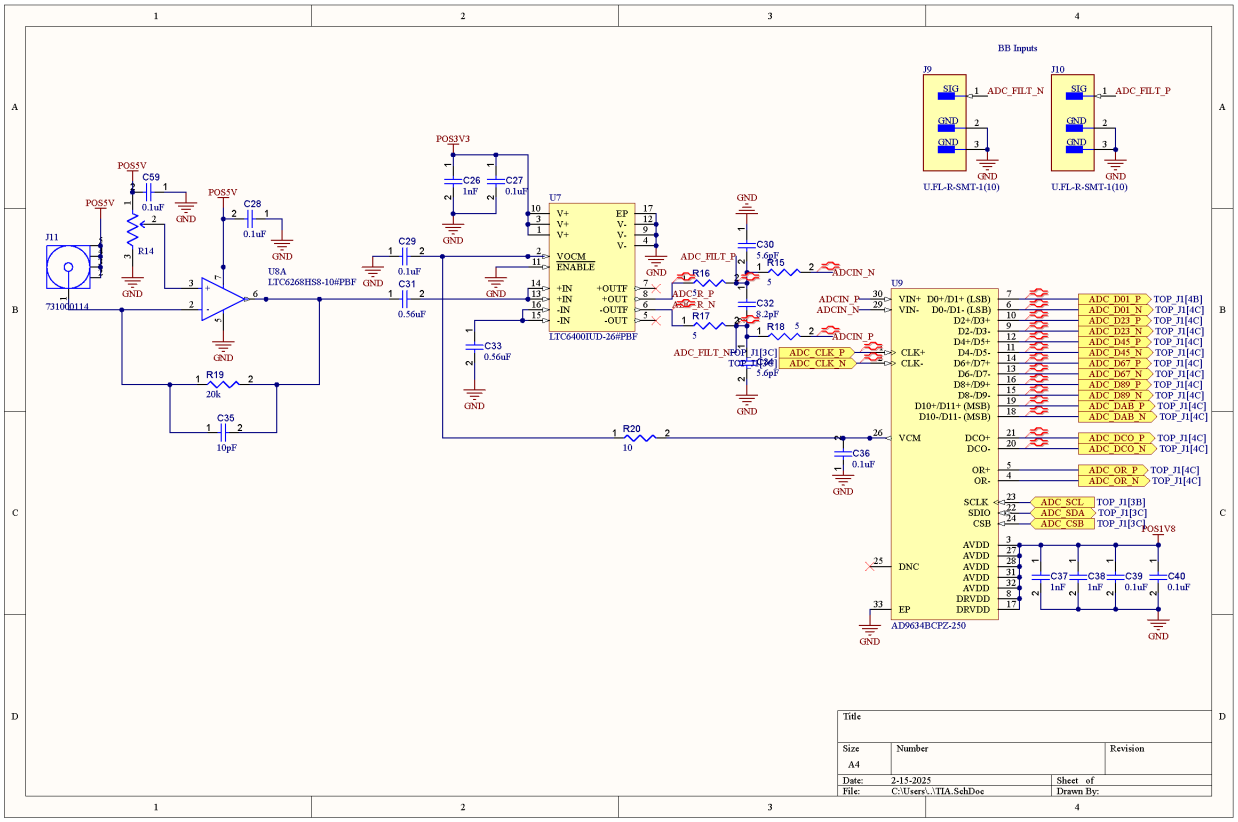

Figure 3.1: TIA Circuit

Since the input magnitude is a function of separation from the transmitting device, the design faces the hazard of exceeding the ADC's dynamic range. Since the selected ADC driver has a fixed gain to simplify the design and reduce the BOM, the only degree of freedom to control gain is the resistance, R_F in the feedback path of the TIA stage. The feedback resistor is calibrated based on a maximum transmitter swing at minimum distance to ensure the input swing to the ADC does not go out of range.

Another crucial parameter for the functional TIA is stability. In accordance with the datasheet [1], the required stabilization capacitance is on the order of 0.1 pF which can be realized by stray capacitances on the PCB and components.

3.2 OTA (LED Driver)

In its essence, the LED is a current-to-light converter with performance characteristics mainly limited due to the LED junction capacitance. To compensate for this, the LED driver consists of a transimpedance amplifier to convert input current from the DAC to a voltage, and then

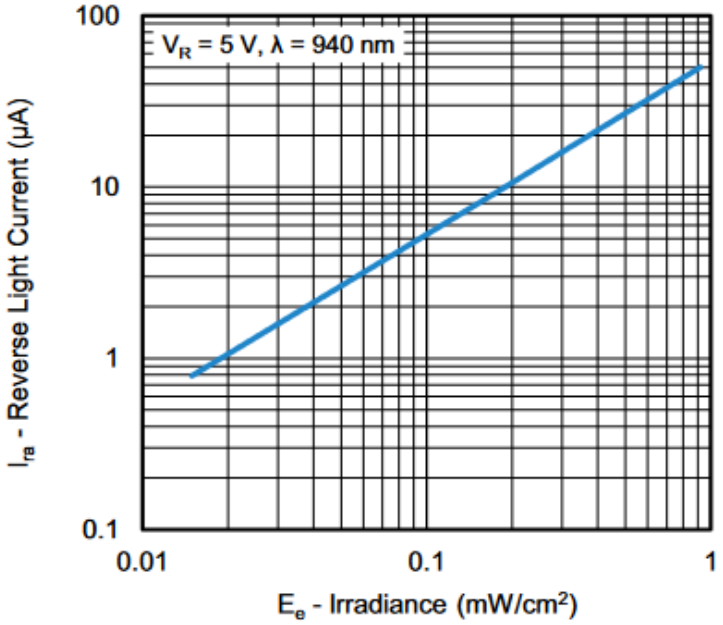

Figure 3.2: Photo-diode Responsivity [3]

a PD controller, where the differentiator compensates for the pole produced by the capacitor. Because the PD controller is voltage-based, the final stage drives the gate of a MOSFET, which acts as an operational transconductance amplifier (OTA), to turn our filtered voltage back into current to drive the LED.

The MOSFET's current is precisely controlled by a voltage setting resistor in the feedback loop of the previous stage.

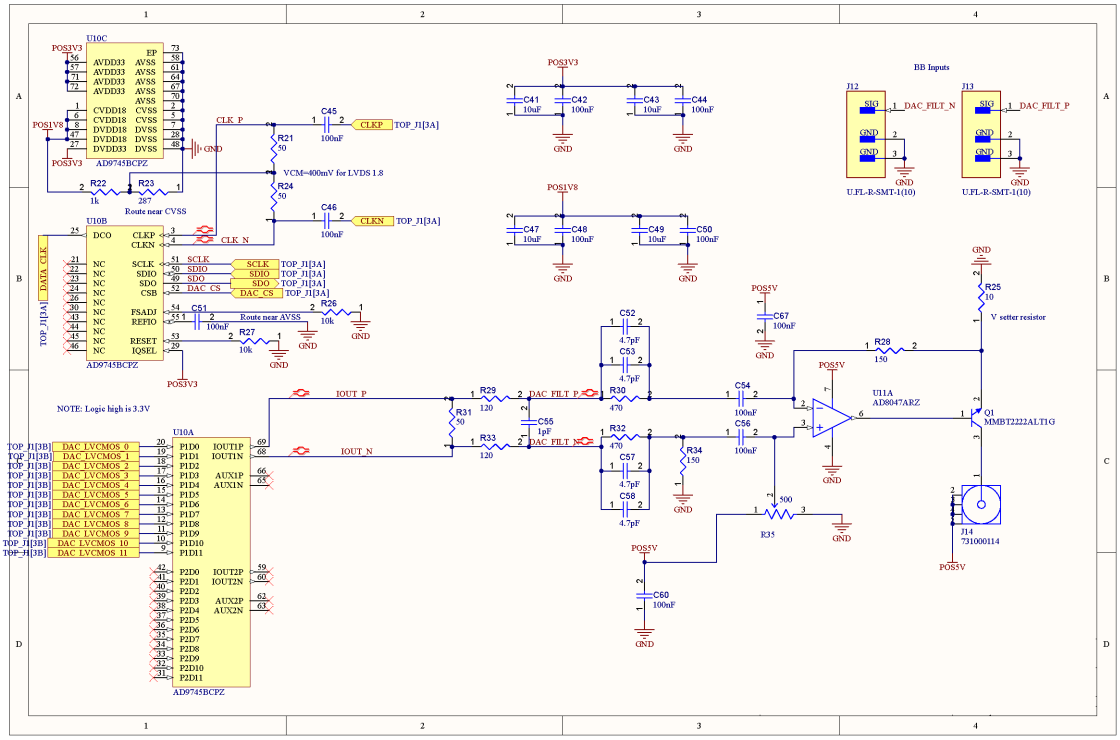

Figure 3.3: OTA Circuit

The differentiator's voltage transfer function is as follows:

$$A_v = \frac{-s \cdot \frac{R_f}{R_1}}{1 + \frac{1}{s \cdot R1 \cdot C1}} \tag{3.3}$$

Overall, the DAC to LED chain was able to exceed our desired specification of 50MHz, while maintaining a correct DC bias for the LED.

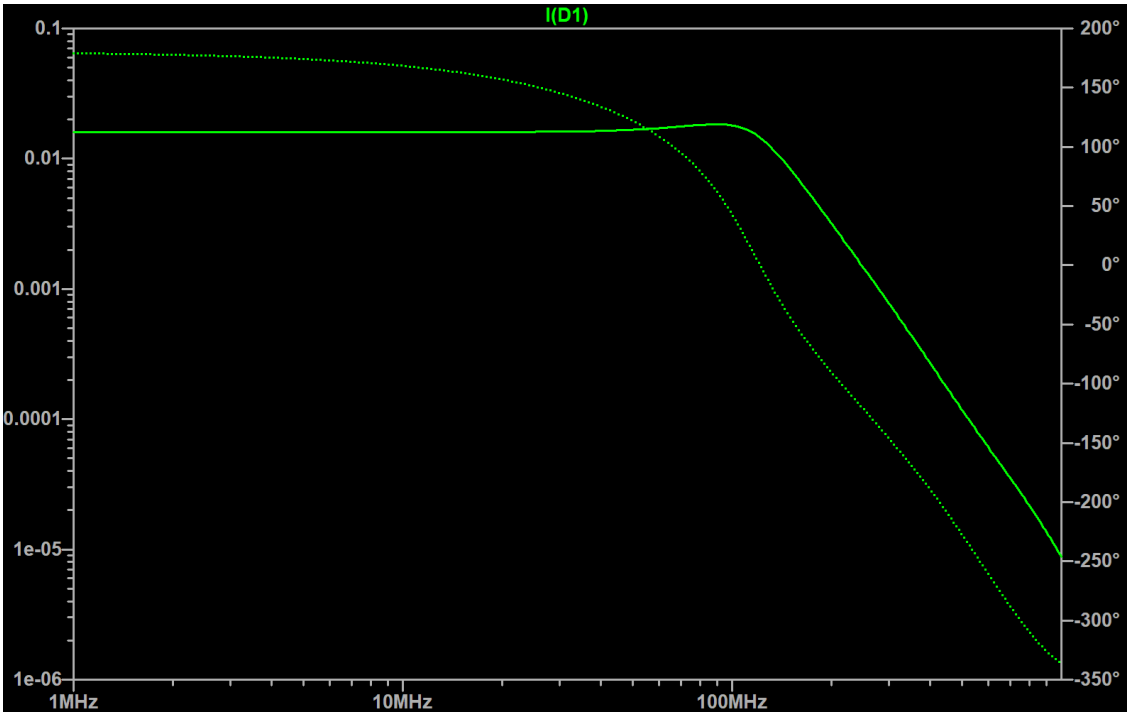

Figure 3.4: OTA Transfer Function

3.3 Channel Modelling & Compensation

A major challenge in the transmission of data in any medium is dealing with nonlinearities, noise, and scattering effects of the channel. Traditionally, infra-red communication systems are either operated with virtually no channel (as an optocoupler), or at extremely low speeds with full-swing PWM (such as in TV remotes). This project seeks to transmit data over infra-red with analog modulation in order to increase data rates. Along with this comes all the nonidealities of a traditional RF communication system. Because of this, a key part of this project includes compensation for these nonidealities. Split into two parts, firstly, compensation is designed for the LED and photodiode themselves; this seeks to neutralize the internal capacitance of the diodes, pushing their pole to a higher frequency, thus increasing the bandwidth and therefore maximum data rate. Secondly, compensation is done to flatten (pre-distort) the power linearity of the diodes as well. By increasing the power linearity, the diodes are able to be driven with higher power levels, directly increasing transmission gain and reducing harmonic distortion, which implicitly reduces overall noise, desensitization (increases dynamic range), and inter-symbol interference.

Nonlinear amplifier output can be modelled with the following equation:

$$y(t) = \alpha_1 \cdot x(t) + \alpha_2 \cdot x^2(t) + \alpha_3 \cdot x^3(t) \tag{3.4}$$

where $x(t) = \cos(t)$, it follows that

$$y(t) = \alpha_1 \cdot \cos(t) + \frac{\alpha_2}{2} \cdot (1 + \cos(2t)) + \frac{\alpha_3}{4} \cdot (3\cos(t) + \cos(3t)) \tag{3.5}$$

This shows the production of higher-order harmonics (and DC offset) with nonlinear gain.

3.4 Choosing the FPGA Devboard

Evaluation Board	FPGA	Cost(cad)	Differential Pairs I/O exposed on board	Ethernet on-board	LUTs	BRAM
Arty s7-50	Spartan 7 XC7S50-1CSGA324C	295	19 pairs	No	32,600	2700 kbits
Butterstick-fpga	Lattice ECP5 LFE5UM5G-85	278	10 pairs	Yes	84,000	3744 kbits
PZ7Z020 SOM	Zynq 7000 XC7Z020-2CLG400I	258.73	52 pairs	Yes	53,200	4.9 MB

Table 3.1: FPGA spec comparison

When evaluating the feasibility of FPGA development board, we took into account connectivity options, integration, cost, and features. All of the above FPGAs have more than enough LUTs and BRAM for our needs. Our ADC and DAC have 12 bits of resolution each and use differential signaling, so we wanted an FPGA with at least 24 pairs of differential I/Os. Additionally, due to our high bandwidth requirement, we decided to use Ethernet, which requires a physical layer on-board. For prototyping, the connectivity overhead when connecting our FPGA to the PCB could introduce limitations on speed and accuracy, and so we chose to use a SOM that slots directly on the PCB. Overall, we found the PZ7020 SOM to be our best option, it is also the most cost-efficient option.

3.5 DSP / Modem

DSP modulation schemes can generally be categorized into Amplitude shift keying(ASK), frequency shift keying(FSK), and phase shift keying(PSK). Their advantages and disadvan-

tages are listed in table 3.2.

Modulation Scheme	Pros	Cons
ASK	Bandwidth efficient Simple implementation	Noise prone High error rate Sensitive to data converter's accuracy
FSK	Immunity to amplitude noise	Lower spectral efficiency Low bit rate
PSK	Simple implementation Bandwidth efficient Immunity to amplitude noise	Susceptibility to phase errors

Table 3.2: Modulation Scheme Comparisons

Due to the above reasons, we've decided to start with PSK for our initial prototype. Within the category of PSK, there are BPSK, QPSK, 8PSK, etc, which implement 2, 4, 8 available symbols to be transmitted per cycle, respectively. We've decided to start with BPSK due to its simple implementation.

Implementing the modulator is relatively simple, because we are doing the mixing in software(in FPGA), we are aiming for the functionality of a hardware mutiplier. See Figure 3.5 for BPSK modulation.

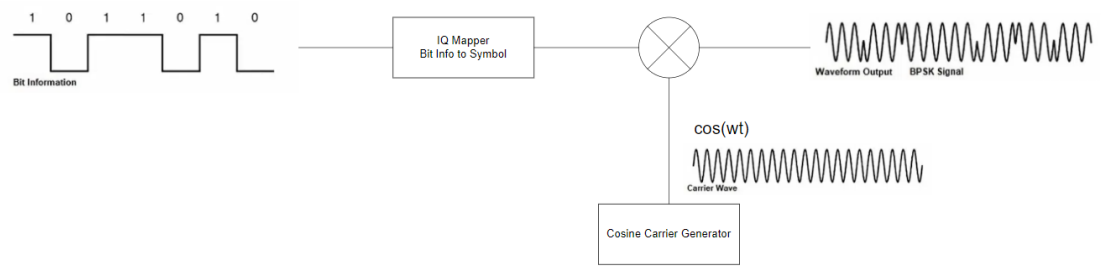

Figure 3.5: BPSK Modulation Block Diagram

The IQ Mapper in Figure 3.5 maps bit information to the constellation represented by a complex number, which is then multiplied with the carrier signal. Note that multiplying the imaginary part simply represents a phase shift. For BPSK, the symbols on the constellation is 1 and -1, exactly 180 degrees difference. A hardware multiplier is a deep combinational logic, which introduces timing risk as we increase the FPGA's operating frequency. Our implemnetation simply uses a counter and a look up table to generate the cosine wave, which

avoids using a hardware multiplier since the symbols are only different in phase. Later on as we implement higher-M PSK (QPSK, 8-PSK), the phase shift can simply be realized by offsetting the counter value. Below is a snippet of the modulator design in Verilog, and Figure 3.6 shows simulated waveform in Vivado

```
`include params.svh
`default_nettype none
module bpsk_modulator_top (
    input  wire                         clk,
    input  wire                         rst,
    input  wire                         en,
    input  wire                         in,
    output reg [`FIXDT_64_A_WIDTH-1:0] out
);

    wire unsigned [$clog2(`CARRIER_SAMPLES_PER_PERIOD)-1:0] count;
    wire signed  [`FIXDT_64_A_WIDTH-1:0] wave_out [0:0];

    assign out = en ? (in ? wave_out[0] : ~wave_out[0]) : 0;

    counter counter_inst (
        .clk(clk),
        .rst(rst),
        .out(count)
    );

    cosine_lut #(
        .READ_PORTS(1)
    ) cosine_lut_inst (
        .in({count}),
        .out(wave_out)
    );

endmodule
```

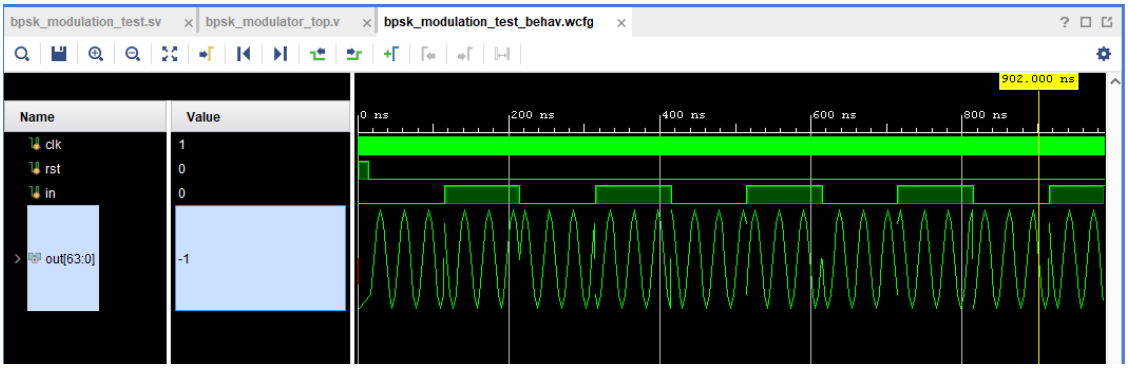

Figure 3.6: BPSK Modulation Simulation

For the demodulator, we implemented a costas loop for the demodulator with a PI filter to close the loop. To make sure the model works, we first started with a Matlab model. The matlab model is a loopback that first generates random bit data, modulates with carrier, demodulates and then compare the demodulated bit data. This helps to tune the PI filter coefficients. Some parameters set in the matlab model are shown below

```
symbol_rate = 2500000;                    % symbol rate (symbol/second)
bits_per_symbol = 1;                      % bits/symbol, 1 for bpsk
bit_rate = bits_per_symbol*symbol_rate; % bits per second
fs = 200000000;                           % sampling frequency
fc = 25000000;                            % carrier frequency
sps = fs / bit_rate;                      % samples per symbol
lpf_taps = 10;                            % depth of our integrator LPF

% PI Filter Parameters
Kp = 0.001;  % Proportional gain
Ki = 0.0001; % Integral gain
```

See Figure 4.3 for the simulated waveforms in the Matlab. Once we've proven that the proposed model works, we implemented the design with verilog and represented all data in fixed point data type with 43 fraction bits. Figure 3.7 for a detailed implementation of the Costas loop. The low-pass filters are integrators(10-tap accumulators). And the PI filter is simply 2 hardware multipliers and an accumulator inside, mutliplying the input phase error and the accumulated value with the coefficients, and readd them into the accumulator.

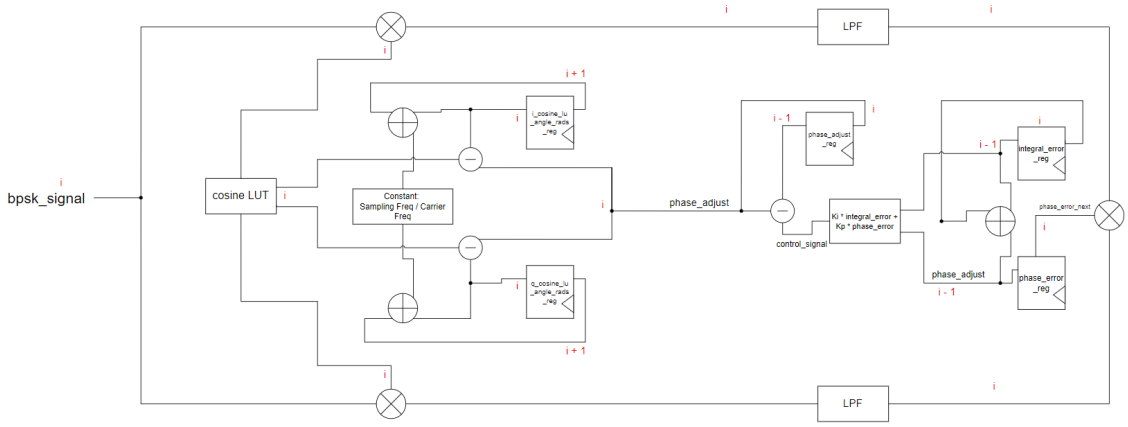

Figure 3.7: Costas Loop Simulation

Figure 3.8 shows the Verilog code simulated in Vivado. We can see the phase adjustment variable locks nicely.

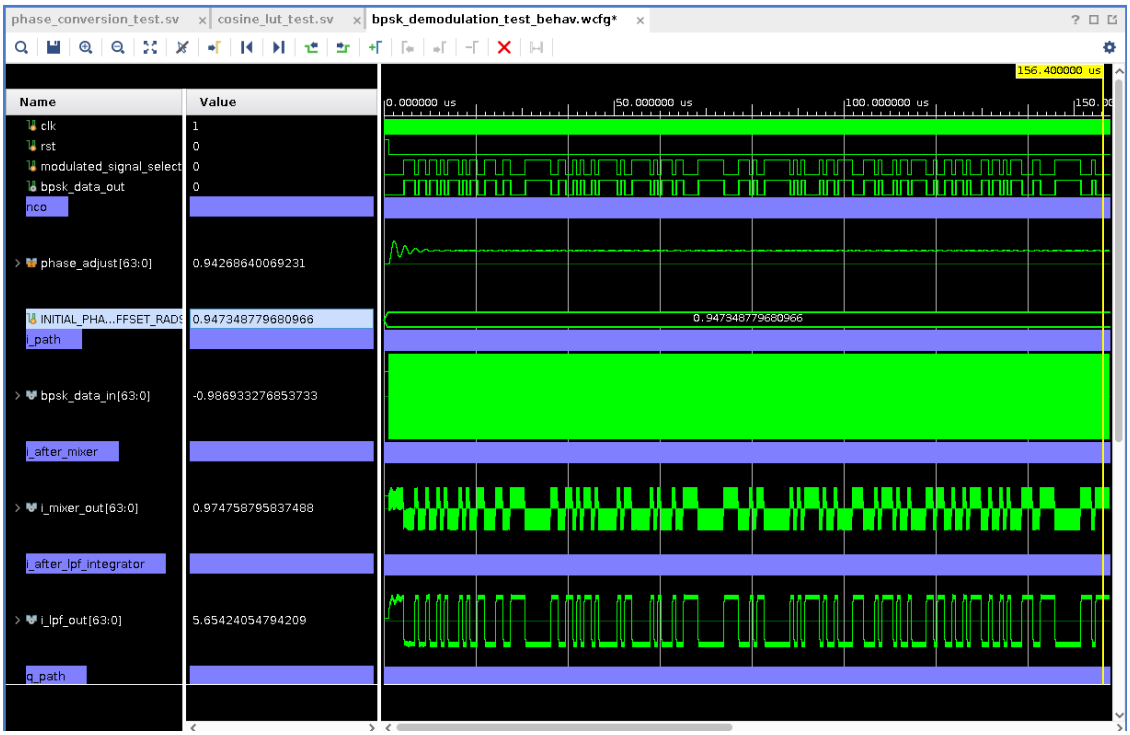

Figure 3.8: BPSK Demodulation Simulation

3.6 Internal Data Type and Precision

When doing arithmetic in FPGAs, it is a common practice to use fixed point data type to represent numbers, since representing data in IEEE 754 double precision floating point

numbers would introduce unnecessary complexity and hardware logic. Our design uses a 64-bit fixed point data type with 43 fraction bits, 20-bit integer bits, and 1 sign bit. Such datatype can cover the range of [-1048576,1048575.9999999999]. This is chosen carefully since the demodulation costas loop to a lot of precision to detect phase error. One of the challenges that took a long time to debug was not having enough precision in the whole demodulation system, causing the Costas loop to not be able to lock to the correct phase. This includes not enough bits for the internal datatype(students used 16 bits to start), and not enough entries in the cosine look-up table. 64 bits represent data with lots of precision, however, it could be a little bit of an overkill and introduce unnecessary levels of logic in the hardware arithmetic components. As an enhancement for the project in the future, this data will be shrunk down to a minimum width.

3.7 Digital Sinusoidal Wave Generator

Inside the Modulator/Demodulator, we use a digital sine wave to perform modulation and demodulation. Generating this sine wave digitally is challenging, and we decided to go with a look-up table implementation. A counter will increment at regular intervals, and every time the sine or phase-shifted sine wave must be used, we will index the look-up table to determine the value at that time step. A pitfall of this approach is resource usage, as implementing a digital sine with 12 bits (ADC/DAC) of resolution may take up a lot of LUTs in the PL. In order to have enough precision for the costas loop to lock to the right frequency, we ended up having to use 8192 entries, which is 13 bits for the look-up table address.

A snippet of the lookup table in Verilog is shown below.

```
module Lookup_Table3 (
  input               clk,
  input               rst,
  input        [4:0]  in1,  // address
  output signed [11:0] out // output voltage level
);
...
  (* ram_style = block *)
  wire signed [11:0] Lookup_Table3_table_data [0:19]; // sfix12_En10 [20]
...
  assign Lookup_Table3_table_data[0] = 12'sb010000000000;
  assign Lookup_Table3_table_data[1] = 12'sb001111001110;
  assign Lookup_Table3_table_data[2] = 12'sb001100111100;
  assign Lookup_Table3_table_data[3] = 12'sb001001011010;
```

```
  assign Lookup_Table3_table_data[4] = 12'sb000100111100;
  assign Lookup_Table3_table_data[5] = 12'sb000000000000;
  assign Lookup_Table3_table_data[6] = 12'sb111011000100;
...
always @(posedge clk) begin
    if (rst) begin
        out <= {12{1'b0}};
    end else begin
        out <= Lookup_Table3_table_data[in];
    end
end

endmodule
```

When we are modulating the signal, the smaller the sample time step the more entries we need on our look-up table. In the example above `Lookup_Table3_table_data` is defined to be an unpacked array of length 12'b X 20. 12'b because our DAC has 12-bit inputs, and 20 because each period is split into 20 steps. This number is going to go up significantly once we obtain more information and lock into our carrier frequency. With that being said, this is a relatively big look-up table with constant values hard-coded into memory. Using LUTs and distributed RAM would but a waste of hardware programmable resources(LUTs). Thus the `(* ram_style = block *)` is used to make sure Vivado instantiates block ram during synthesis. Note that block ram requires synchronous read, which means there is a one-cycle delay on lookup. This is fine because during modulation the time step that we input to the lookup table is always fixed and repeated over each period.

3.8 Zynq SoC with ADC/DAC Interface

The Zynq-7000 SoC we are using has 4 on-chip phase lock loops(PLL) for generating clocks 800 Mhz clock signal [4], table 73. However, the maximum I/O output frequency of the Zynq-7000 is 680 Mhz [4], table 68, we would have to pay extra attention in routing the clock signal on the custom PCB. The ADC has a maximum clock input frequency of 625 Mhz [5] and the DAC is 250 Mhz [6], if we make any changes from now on to the end of the project, we will still be able to operate the data converters at their highest speed.

3.9 Host PC Interface

Because of the high bandwidth transmission target we have for ourselves, we explored the options for different common communication protocols and their

Protocol	Speed	Solution in the FPGA/SoC	Physical interface
USB 2.0	480 Mbps	Free AXI USB 2.0 Device Controller IP + MicroBlaze softcore processor	PHY on custom PCB prototype
USB 3.0	5 Gbps	Implement FIFO Controller in Verilog	FTDI FT601x USB 3.0 to FIFO bridge chip
Fast Ethernet	100 Mbps	Open-source Ethernet IP non-processor integration OR MicroBlaze softcore processor + 1G/2.5G AXI Ethernet Subsystem IP.	PHY on custom PCB prototype
Gigabit Ethernet	1000 Mbps	Cortex ARM Ethernet controller	already built PHY on PZ7Z020-SOM
2.5Gb Ethernet	2.5 Gbps	Free Vivado 1G/2.5G AXI Ethernet Subsystem IP, non-processor mode	PHY on custom PCB prototype

Table 3.3: Communication Protocol Comparisons

We plan to use 50 Mhz sinusoidal wave as our carrier signal. Nyquist theorem states that a signal can be reconstructed if the sampling rate is at least two times faster than the signal frequency. Our ADC and DAC sampling rate is 250 Msps maximum, which allows us to use 125 Mhz carrier sinusoidal, 50 Mhz is well below the threshold. Note that the bottleneck here is the infrared LED switching frequency, based on the datasheets of the LED [7] and the photodiode [3] we are estimating a 50 Mhz switching frequency, however, the actual number is obtained from lab characterization. Out simulation on the TIA shows that the gain is around 107 dB at 50Mhz. Figure 3.9 shows that we have -3dB frequency at 87 Mhz.

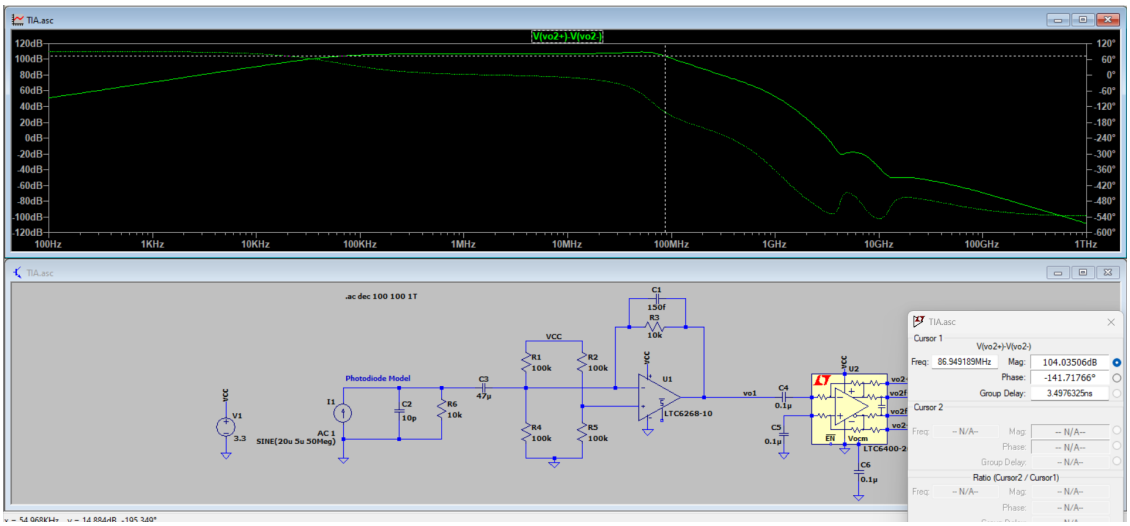

Figure 3.9: TIA frequency at -3dB

The relationship between bandwidth and bit duration is shown in equation 3.6.

$$\begin{aligned} T_b &= \text{bit duration} \\ R_b &= \text{bit rate} = f_b = \text{frequency of baseband signal} = 1/T_b \\ f_b &= \text{frequency of baseband signal} = 1/T_b \\ f_o &= \text{carrier frequency} = 50Mhz \\ BW &= \text{bandwidth of BPSK} = (f_o + f_b) - (f_o - f_b) = 2f_b \end{aligned} \tag{3.6}$$

Note that because BPSK transmits one bit per symbol, the bit rate is 1 / bit duration. Plugging the numbers in from the TIA simulation and previous analysis

$$\begin{aligned} f_b &= 87\,\text{MHz} - 50\,\text{MHz} = 37\,\text{MHz} \\ T_b &= 1/37\,\text{MHz} = 27\,\text{ns} \\ R_b &= 1/27\,\text{ns} = 37\,\text{Mbit}\,\text{s}^{-1} \end{aligned} \tag{3.7}$$

Equation 3.7 shows that we need a communication protocol that supports at least 37 Mbps. However, we plan on playing around with different modulation schemes in the future. Once our first prototype custom PCB is built, we only need to change the Verilog implementation in the Zynq-7000 in order to change the modulation scheme. No hardware reiterations are needed, so we like to keep the transmission speed high for future expansion. For example, if we go to 8PSK, for each bit duration we send 3 bits. The bit rate becomes 3.8.

$$R_b = (1/27ns) \cdot 3 = 111Mbps \tag{3.8}$$

This eliminates 100 Mbps fast Ethernet from our options. It also suggests that we probably don't need over 2.5 Gbps data rate since that is way out of the range that our photodiodes

and TIA can support.

Another factor to consider here is testing and project timeline. A physical PHY is needed to communicate between the FPGA and the host computer. If we choose to go with Ethernet, We would need to implement the Ethernet PHY circuit on our prototype PCB to test the modulation and demodulation of our DSP block. If we choose to go with USB 3.0 with the FTDI 60x FIFO to USB 3.0 bridge, we need to lay the chip out on our prototype PCB, which means we can't test it until the prototype PCB is done. Or we can also buy an UMFT60x Evaluation board and connect our FPGA dev board of choice to it, which mean extra cost and effort to make sure the connect types match.

Last but not least, we also considered the software side effort. USB 2.0 and 2.5Gb Ethernet solutions require configuring the MicroBlaze softcore processor in our FPGA using Vivado and integrating a free IP from Xilinx into our system. Fast Ethernet(100 Mbps) requires integrating an opensource Ethernet IP without a softcore processor into our system. USB 3.0 requires designing a FIFO controller in Verilog. All of the above will take more effort than the Gigabit Ethernet solution where we only need to program the hardcore ARM processor in Zynq-7000 in C.

After considering all of the above factors, we decided to go with Gigabit Ethernet. This gives us room to expand if we need a higher data transfer rate in the future, as well as fast efficient testing since our FPGA board of choice(The PZ7Z020-SOM) already has Ethernet PHY on it.

4 Prototype Data

4.1 PCB Design

To integrate our subsystems, a high-speed mixed-signal PCB was required. This PCB serves as the foundation of our design, connecting the various physical components while ensuring power and signal integrity. Given the complexity of our system, major design considerations included managing high-speed digital and analog signals, reducing noise and coupling, and ensuring robust power distribution. Several techniques were employed to optimize signal integrity, mitigate EMI, and provide reliable operation.

Additionally, the PCB required impedance controlled manufacturing due to the strict requirements of transmission lines on the board. The board employed a signal-ground-power-signal

stackup, chosen to provide the best possible quazi-TEM transmission lines for the high-speed transmission lines placed on the top of the board.

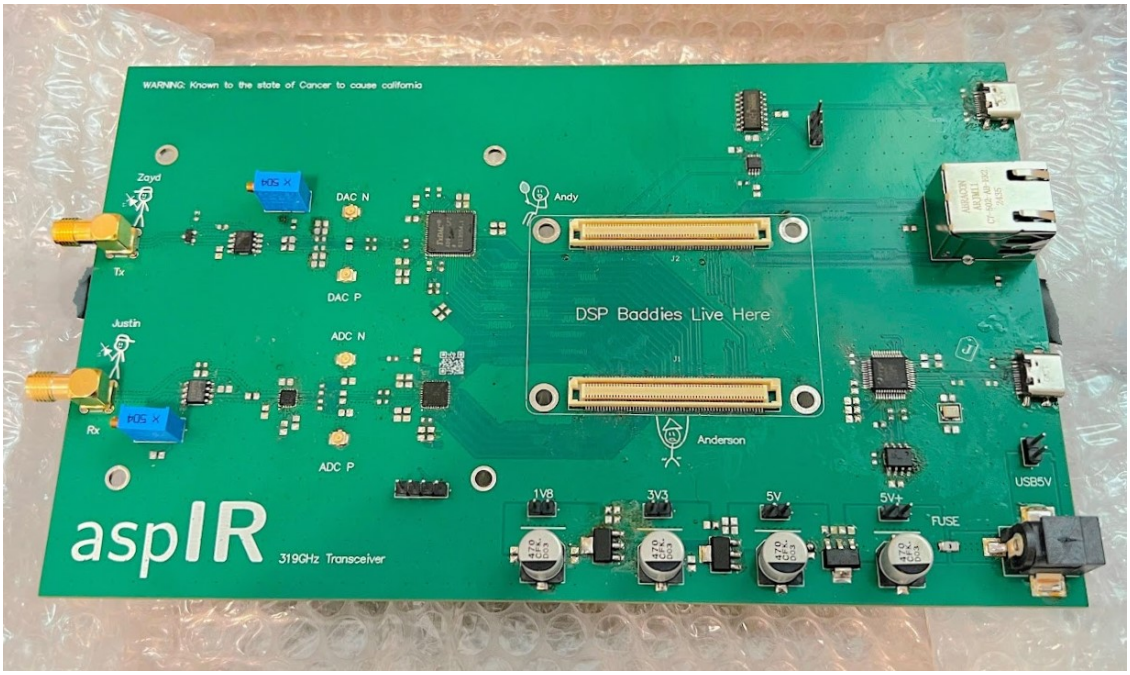

Figure 4.1: Assembled PCB

4.1.1 Power

The power distribution network was designed to minimize transients and provide low-noise power delivery to all subsystems, while budgeting enough power for the maximum load case.

Based on the supply voltages required by the FPGA, analog section biasing, and peripherals, there are 4 main voltage rails on the PCB: 5V+, 5V, 3.3V, and 1.8V. As we are dealing with high-gain amplifiers that are very susceptible to noise, switching regulators were ruled out in favour of LDOs. 5V+ is the fused input of the entire PCB; 6 volts is chosen to balance between a sufficient buffer for dropout and to reduce thermal loss. Because we are using LDO regulators, adequate electrolytic capacitors were added to stabilize voltage rails. Of course, decoupling capacitors were also added close to every load, minimizing trace inductance and allowing for stable power during transients such as startup and state switching. Finally, a split power plane was employed to distribute power across the entire PCB, and a ground plane and stitched pours were used to minimize return path.

4.1.2 Analog Section

In the analog section, reducing noise and coupling was critical. Analog traces were carefully routed, including the use of coplanar waveguide transmission lines that don't have any sharp corners. The entire analog section was also well-shielded using vias and ground pours.

4.1.3 Digital Section

High-speed digital signals required impedance-controlled traces and transmission line techniques to minimize reflections and signal degradation. Differential pairs were required for the clock and ADC LVDS lines, and parallel lines were meandered to ensure minimum skew between bits.

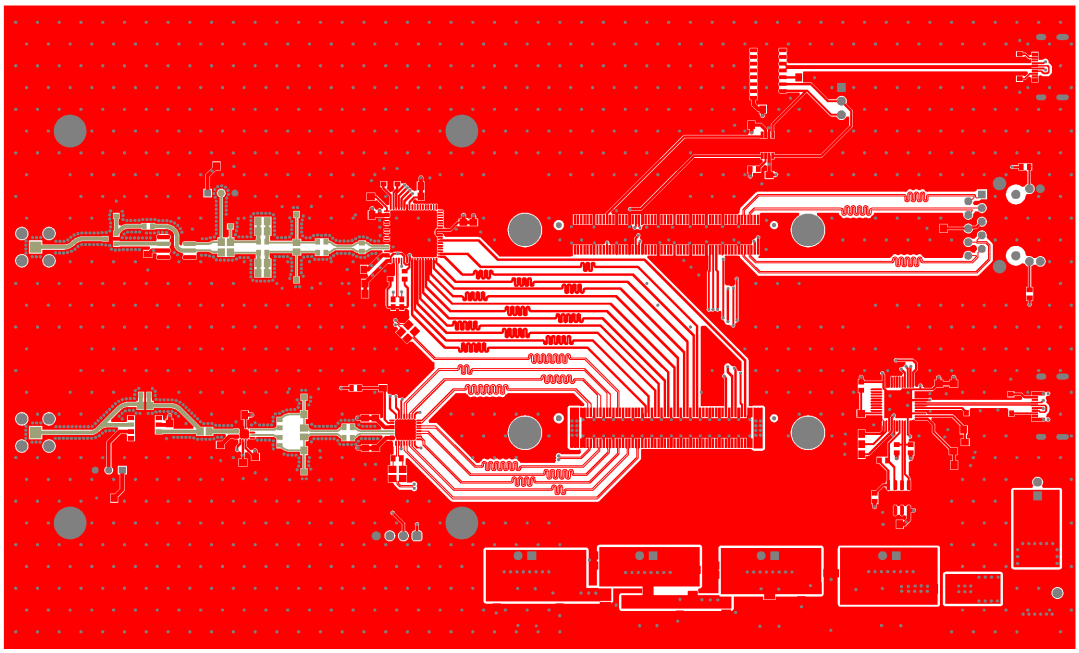

Figure 4.2: PCB top layer layout

4.1.4 Peripherals

Peripherals had their own special requirements, with ethernet and USB requiring differential traces and board-edge connectors.

4.2 BPSK Modeling

The matlab model shows the demodulation performance under the parameters listed in the detailed design section of the report.

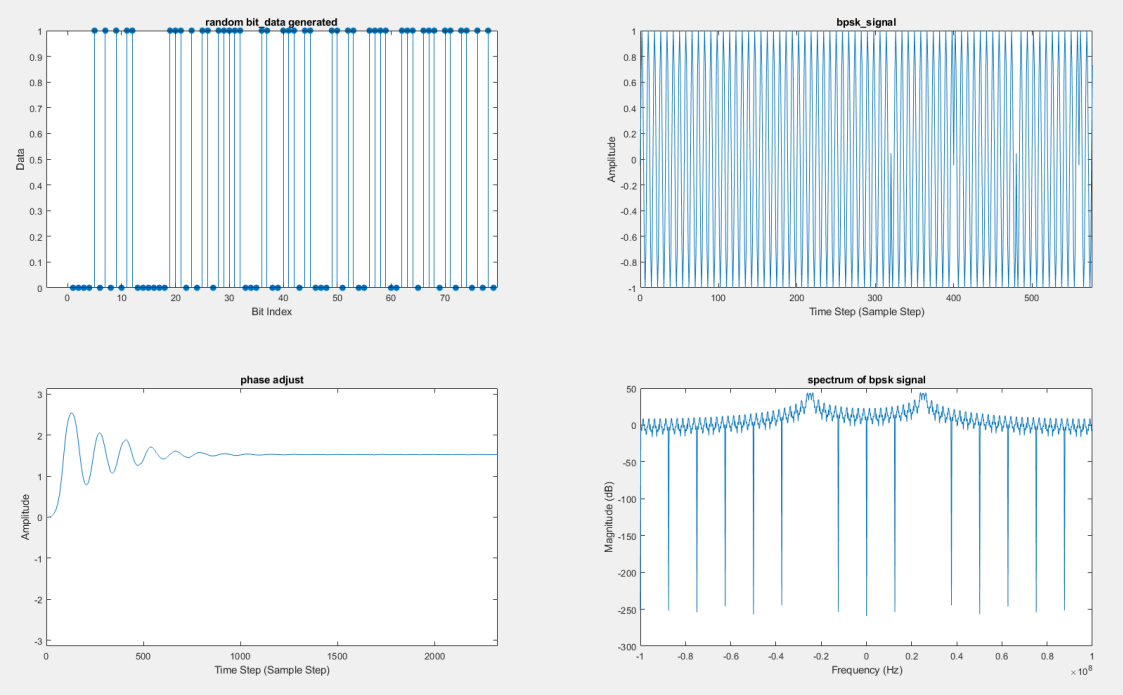

Figure 4.3: BPSK Matlab Simulated Waveform

5 Discussion and Conclusions

We can observe that the average time to for the Costas loop to settle and lock phase is about 1000 sample steps. The actual time it takes to lock phase would depend on our system operating frequency, given that we're operating our ADC/DAC at 250 Msps (period 4ns), it would take the Costas loop about 4000ns to lock phase.

5.1 Evaluation of Final Design

The objective of this project was to build a full IR transceiver capable of operating at 50MHz over 10m in the presence of noise and interference. The prototype meets this objective if we can send data under these conditions and meet speed requirements. The final product consists of one transceiver, connected to a host computer. The computer streams a video file through the transmitter and it is reflected by a mirror to the receiver and displayed on

screen to complete a loop-back test. To show that the data is being transmitted through light we can block the line-of-sight and the received video stops playing.

5.2 Use of Advanced Knowledge

Overall, the project incorporates advanced knowledge from every step of wireless communication. From device physics within the characterization of photodiodes and amplifiers (ECE 331) to analog circuits in the design of the analog signal processing block, OTA, TIA, and ADC driver (ECE 432). Board-level electronics knowledge is required in order to design the PCB with proper power supplies, and advanced EM knowledge is required to ensure signal integrity while operating at specified speeds (ECE475).

System-level knowledge of communication systems is required for the simulation, where every part of the transmit-receive chain will be simulated in MATLAB, including the modem (signal modulation), ADC and DAC, analog portions, and nonidealities found in our transmission channel.

In addition to hardware, A large portion of the project focuses on the modem, utilizing advanced modulation techniques (such as M-PSK) requiring digital signal processing knowledge, as well as this, digital hardware design knowledge from ECE327 is required to implement such a modem on an FPGA. This uses knowledge from ECE 313 Digital Signal Processing and ECE 318 Communication Systems.

Finally, in order to utilize the device for tests, computer networks and software knowledge are required to interface with the device over Ethernet to transmit and receive data.

5.3 Creativity, Novelty, Elegance

Traditional IR communication is low-speed and suffers from interference from other IR sources like the sun. Our design makes use of analog and digital signal processing techniques like amplification, differential signaling, and BPSK to reduce the effects of interference. We are also targeting a much higher bandwidth of 50 MHz. Our design is also low power, incorporating a wake-up feature so that the device isn't consuming power when it is not in use.

For our demo, we are to create a custom PCB and solder all of our components onto it. Since we prioritize speed and error rate, we want to reduce connection overhead as much. We selected an FPGA that is a SOM, which will allow us to test individually, and then when the time comes to build the prototype, we can insert the FPGA directly onto the PCB.

We are are using ethernet to interface between the host computer and our design. Ethernet was chosen as a solution over other interconnect methods like USB-C and UART, because it satisfies our speed requirements while being relatively easy to implement since our FPGA supports ethernet (but not USB-C).

5.4 Quality of Risk Assessment

During the design and manufacture of our prototype, we encountered two key risks.

We needed to ensure that the maximum swing on the ADC input did not exceed the safe operating range since the ADC is an expensive component. We calibrated the TIA gain to ensure at minimum range and maximum transmitted signal swing the ADC input was safe. This method was not fool-proof but we decided to omit a clamping circuit for performance reasons since that could introduce unwanted bandwidth limitations and distortion.

Secondly, to ensure that we reduce the risk of burning during soldering, components were placed entirely on one side, and a hot plate was used for soldering with low-temperature solder paste.

Since we use an LED for the transmission instead of a laser diode we can assure that the transmitted IR power is at a safe level for humans. It is as safe as a TV remote.

5.5 Student Workload

The workload was shared fairly equally within minor limitations of outside commitments.

Student	Contribution Pertcentage
Zayd Abbas	26 %
Anderson Hsieh	26 %
Andy Lang	22 %
Justin Llanos	26 %

Table 5.1: Estimated Workload Division

References

[1] Analog Devices, *4GHz Ultra-Low Bias Current FET Input Op Amp*, LTC6268-10/LTC6269-10.

[2] Analog Devices, *1.9GHz Low Noise, Low Distortion Differential ADC Driver for DC-300MHz*, LTC6400-26.

[3] Vishay Intertechnology, *Silicon PIN Photodiode: BPV10NF*, BPV10NF, Rev 2.1, Jul. 2021.

[4] Xilinx, Inc, *Zynq-7000 SoC (Z-7007S, Z-7012S, Z-7014S, Z-7010, Z-7015, and Z-7020): DC and AC Switching Characteristics*, Z-7020-DC-AC-switching, Dec. 2020.

[5] Analog Devices, *12-Bit, 170 MSPS/210 MSPS/250 MSPS, 1.8 V Analog-to-Digital Converter*, AD9634.

[6] Analog Devices, *Dual 8-/10-/12-/14-/16-Bit 250 MSPS Digital-to-Analog Converters*, AD9745.

[7] Vishay Intertechnology, "High Speed Infrared Emitting Diode, 940 nm, Surface Emitter Technology," Tech. Rep. VSLY5940, May 2015, Rev. 1.2.