EE273 Final Project Report

Frisky Systems, Inc. F-40 Router

"Has your system been frisked today?"™

James Bonanno
Jeffrey Tyhach
Ui Sun Han

3/11/2002

1 Design Overview

We approached the problem of designing the system for an improved bit rate by:

1. Analyzing the existing backplane model, characterizing it, and also determining limiting issues preventing high-speed data transmission.
2. Modeling various options for the chip package and card traces.
3. Deciding to address speed-limiting characteristics in a minimalist manner – that is by adding as little complexity as possible, while achieving the target speed.
4. Implementing measures to reduce inter-symbol interference and crosstalk, and creating simulations of these strategies.

Characterizing the backplane allowed us to determine the characteristic impedance of the line. It also showed us that high-frequency attenuation and crosstalk would be major obstacles for transmission rates at and greater than 4 Gbps.

1.1 Signaling Design

After characterizing the backplane, a differential bipolar current-mode signaling convention was chosen, along with equalization to deal with high-frequency attenuation (causing ISI), and with cross-talk cancellation. Current mode signaling was chosen for the ease of on-chip design as compared with voltage-mode signaling. Differential signaling allows for each signal to be self-referenced, and it allows twice the signal swing of single-ended signaling. Bipolar was used so the return current would be 0, preventing any self-induced power supply noise. We evaluated other potential schemes, such as bi-directional signaling, and multiple signal levels, but determined that such approaches were not needed to meet the target transmission rate, nor were very feasible given the response of the backplane.

1.2 Timing Design

After simulating crosstalk control and equalization, and verifying that these measures were sufficient to achieve an acceptable noise margin, we determined how much timing margin we would have under various timing schemes, which is evaluated further in section 6. We chose an implementation that would send clock with the data, bundled closed loop timing.

The transmit clock, 2.25Ghz is generated with a 225Mhz reference clock oscillator and a phase lock loop (PLL). Our timing scheme is able to easily meet noise and timing margins at a bit rate of 4.5 Gb/s. Figure 1 shows the top-level system block diagram between a line card and a switch card.

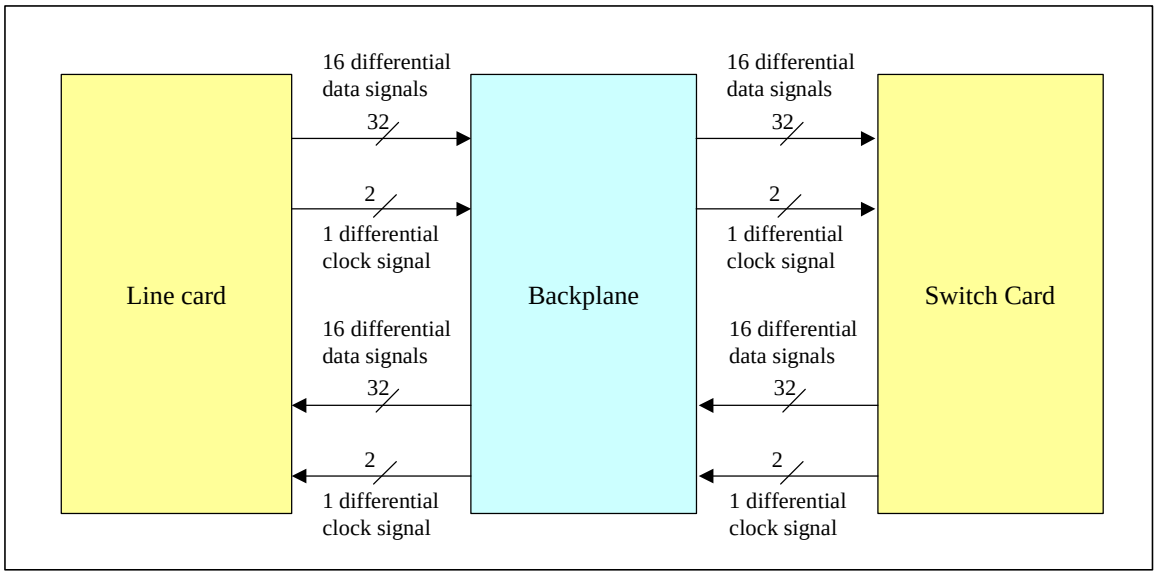

Figure 1 – Top Level System Block Diagram

1.3 Design Summary

Figure 2 is a block diagram of our signaling design. Our design work involved specifying all aspects shown on this diagram except for the backplane itself, which is fixed. Our design meets the signaling goals with acceptable noise and timing margins. The remainder of this report presents our design in detail, justifies our design decisions, and shows our calculations of timing a noise margin.

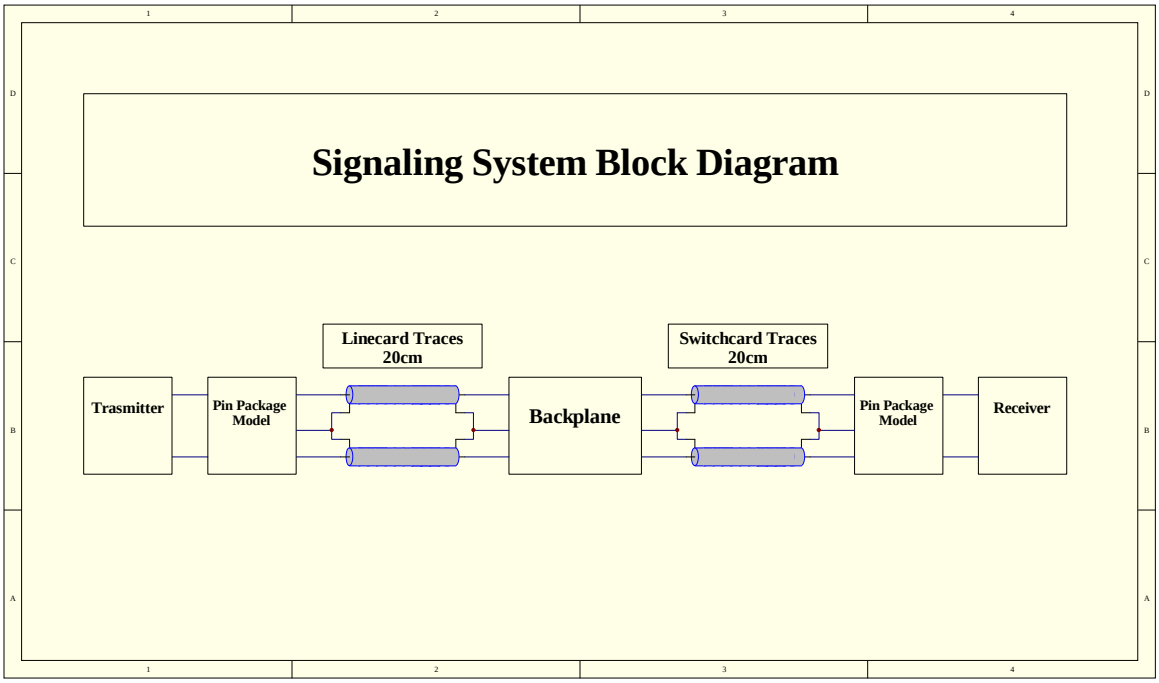

Figure 2 – Signaling System Block Diagram

2 Board Specification

This section describes the specification for the line card and switch card. The board stack-up and trace widths were carefully selected for impedance matching and low crosstalk.

2.1 Chip Package Model

We chose to use the flip-chip package for its low pin inductance and parasitic capacitance. The 1cm trace of transmission line was modeled as an ideal transmission line length of:

$$linelength = \frac{0.01m}{\frac{1}{2} * 3.0x10^{8} m/s} = 66.67\,ps$$

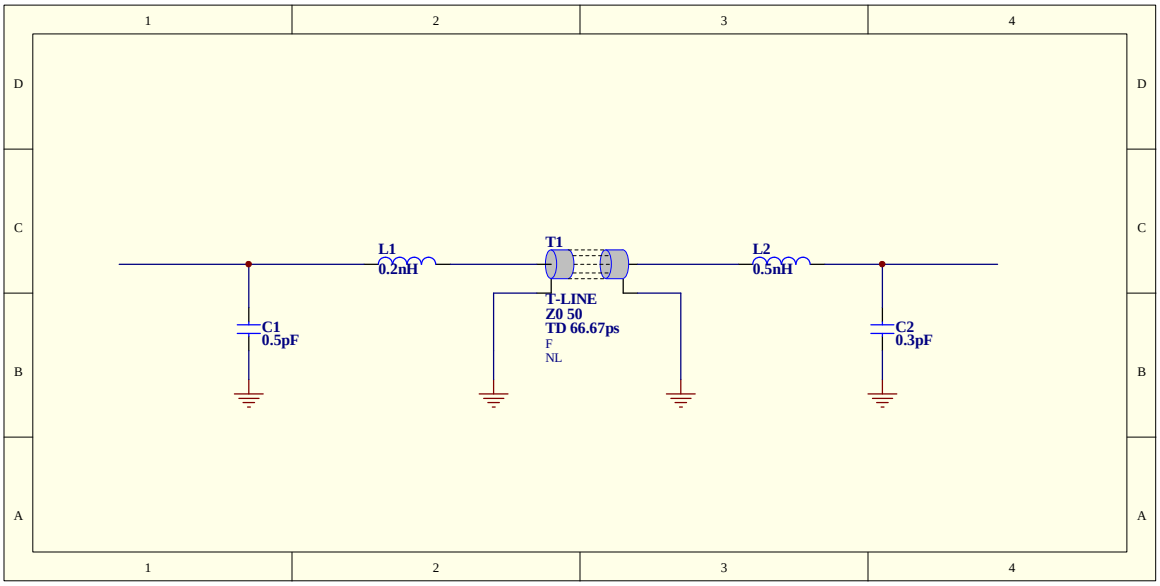

Figure 3 – Flip Chip Package Model

We determined that the flip-chip package with low inductance and capacitance reduces crosstalk and reflection noise at our target signaling rate of 4.5 Gbps at modest cost increase.

2.2 Line/Switch Card Board Stack-up

The board stack-up was designed so that each trace has a characteristic impedance of 50Ω. We used a minimum spacing of 4 mils between the pairs of a differential pair, and a maximum of 46 mils distance between two differential pairs for a low crosstalk coefficient. Also, a typical 10 mils width was used for high-speed signal traces. The ball pitch of 1mm limits the distance between the two differential pairs. The pitch of the

Teradyne VHDM-HDS connector is 2mm, which gives plenty of room to route the high-speed signals. Figure 4 shows the Teradyne connector footprint information.

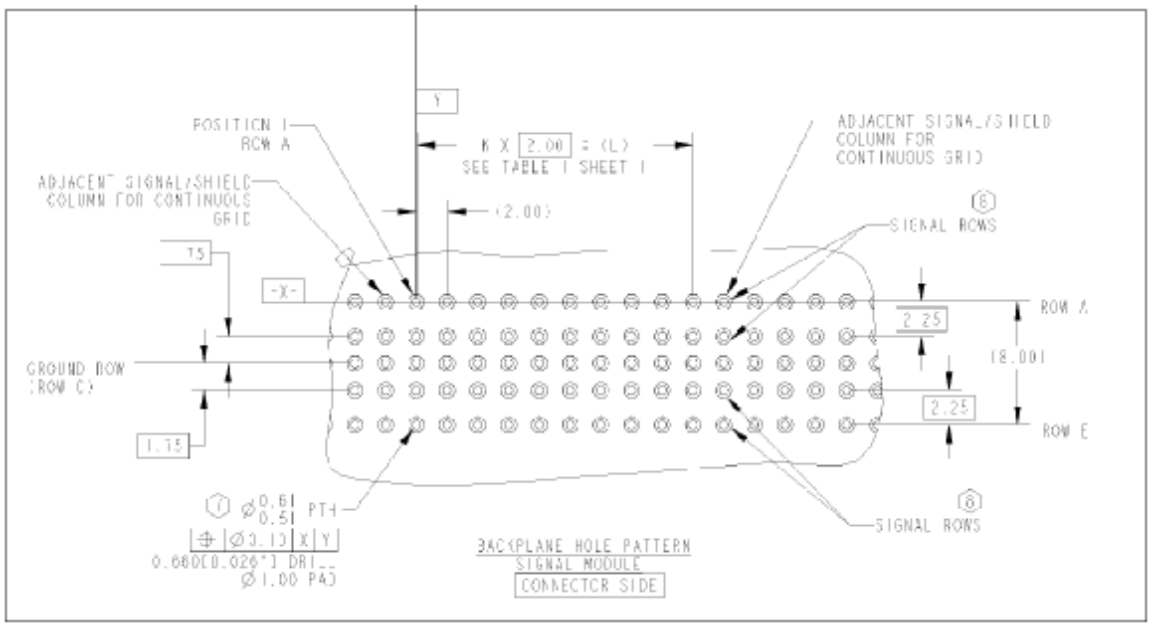

Figure 4 – Line/Switch Card Board Stack-up Dimensions

We selected Teflon glass for our dielectric material. Teflon glass has dielectric constant of 2.55 and dielectric loss tangent of 0.005. The low dielectric constant of the Teflon glass reduces crosstalk noise. A section of the board stack-up with dimensions is shown in Figure 5 below.

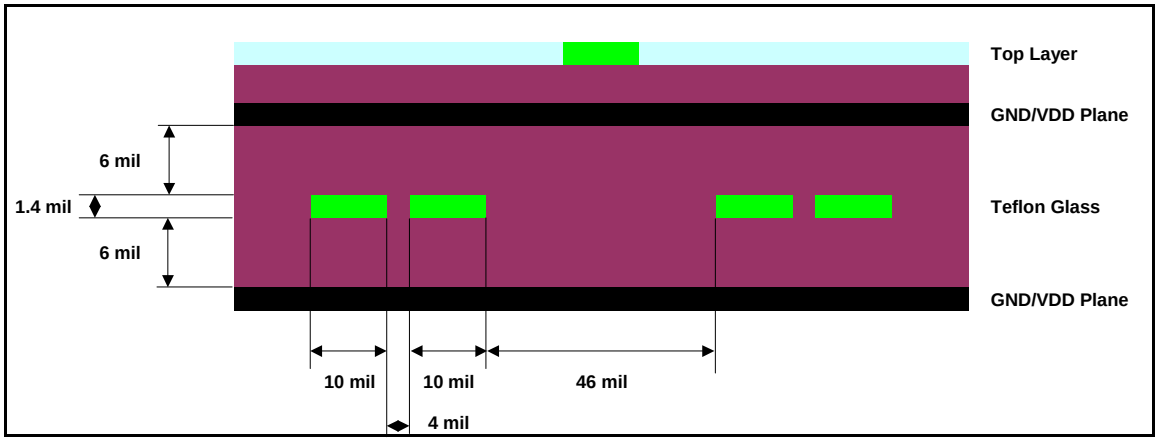

Figure 5 – Line/Switch Card Board Stack-up Dimensions

The full 8-layer line card stack-up is shown in Figure 6 below. For the line card, an 8-layer board should be sufficient to route all 68 impedance-matched traces(64 data and 4 clock) and allow some slow control signals on the top layer. However, this 8-layer structure is not ideal since the layers are not symmetric. It might cause the board to become warped during either the manufacturing process or operation.

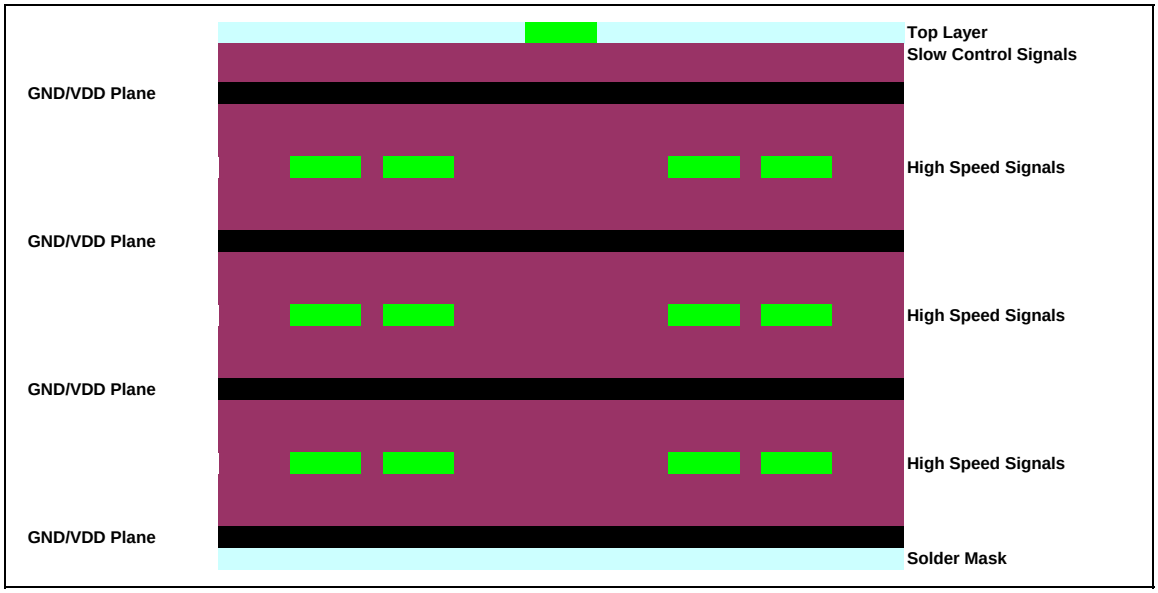

Figure 6 – Full 8-layer board stack-up

A 9-layer alternative, as in Figure 7, can be used to reduce the possibility of board warp issues. This design includes an additional signal layer at the bottom. This also allows some components to be placed on the bottom side, possibly allowing the board size to be reduced. In order to make a decision on the final board stack-up, a complete cost vs. benefit study must be done.

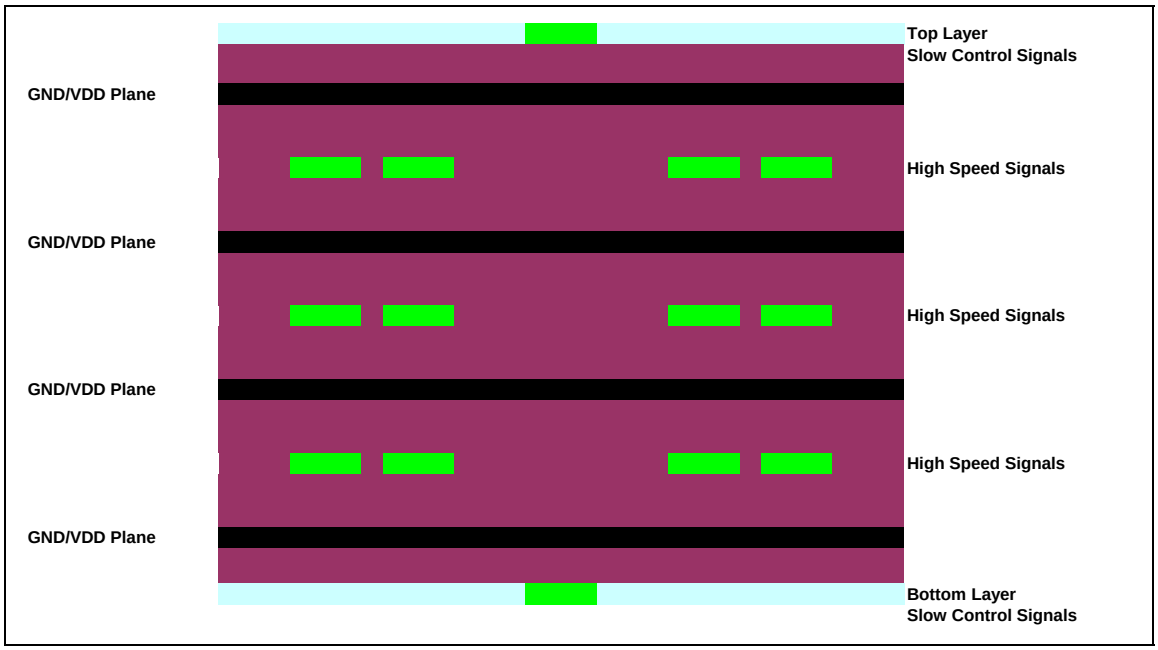

Figure 7 – Alternate 9-layer Board Stack-up

For the switch card, it must be able route 68 x 7 impedance matched traces. An 8-layer board might not be sufficient, and closer study, as well as actual board design, is required to decide the final board stack-up. Since we will be adding the high-speed signals as strip lines with two solid GND/VDD plane for shielding, our analysis of signal integrity will not change even if more layers are required for the switch cards.

2.3 Electrical Model

Using the dimensions discussed above, Linpar was used to generate RLC file. Using this RLC file, the aggressor and victim lines were implemented as a w-element in SPICE.

```
* N = number of lines
*********
2
* Lo = inductance matrix
***************
2.680e-7
1.358e-12   2.680e-7
* Co = Capacitance matrix
**************
1.059e-10
-5.362e-16  1.059e-10
************** 0 0 0
 Go = conductance matrix ******************* 0 0 0
* Rs = skin effect matrix
***************
0.004
0     0.004
* Gd = dielectric loss matrix
***************
0
0     0
```

In order to verify the impedance and to investigate crosstalk, the model was simulated using Spice. Figure 8 shows that the impedance of the trace is 50Ω.

```
* Line card trace simulation
.option post
* step source
v1 x gnd pulse(0 1 1ns 50ps 50ps 400ps 30ns)
* gnd is at near end of line
rx x xa 50
v2 y gnd 0
ry y ya 50
vg gnd 0 0
w2 N=2 xa ya gnd xb yb gnd rlgcfile=lossy_board.rlc l=0.2
*termination
rtx xb gnd 50
rty yb gnd 50

.tran 1ps 4ns

.end
```

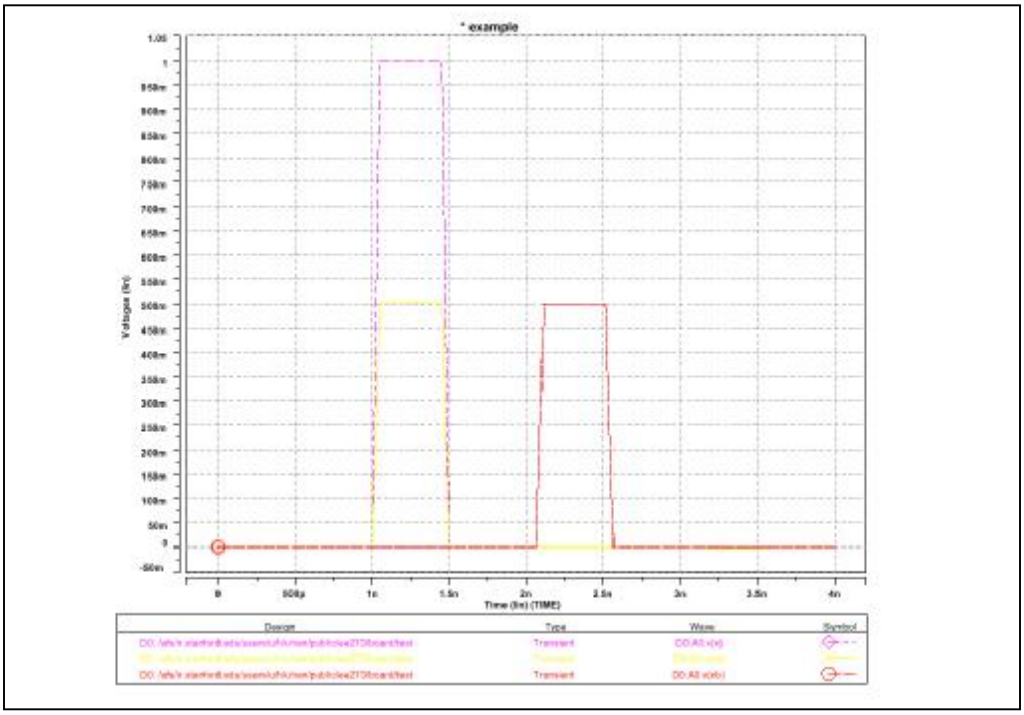

Figure 8 – Simple Spice Simulation of Line Card Traces

The crosstalk caused by the aggressor line is very small, as was expected. Figure 9 shows the 1V pulse created crosstalk of less than 1.3uV on the victim line.

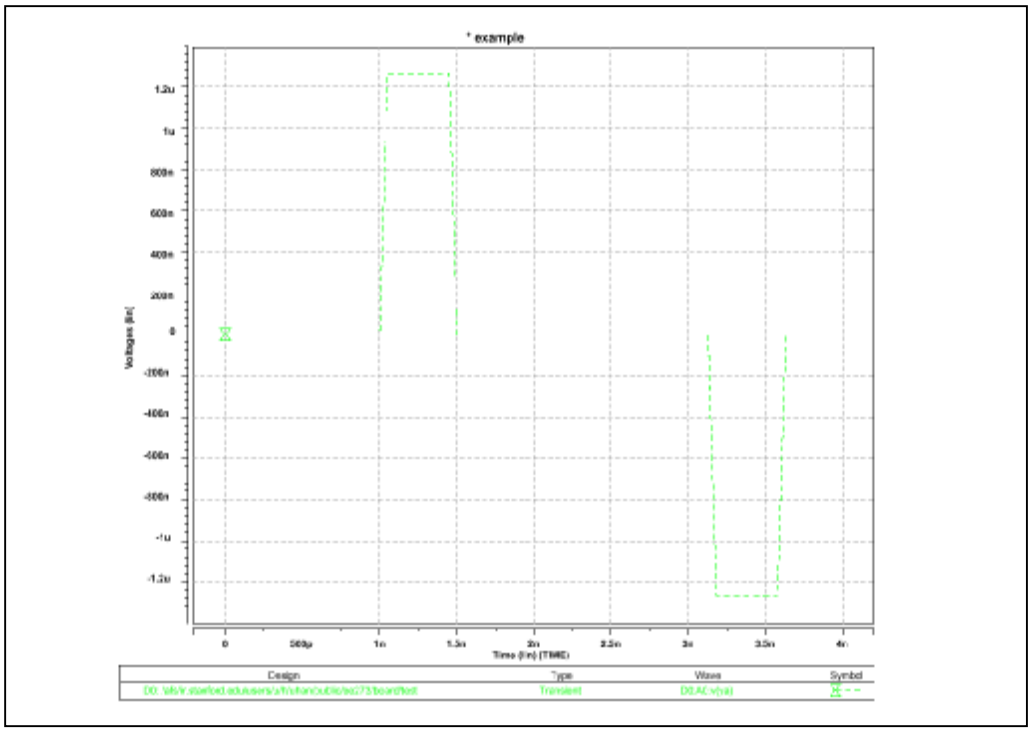

Figure 9 – Crosstalk simulation of Line Card Traces

3 Details of the Signaling Approach

This section describes the signaling convention chosen for our design. This signaling convention is a unidirectional, differential, bipolar, current-mode scheme. It includes equalization and crosstalk control in order to achieve a signaling rate of 4.5 Gbps.

3.1 Unidirectional vs. Bi-directional Signaling

We chose to use unidirectional signaling, which involves sending data in one direction over a wire at any particular time. This differs from bi-directional signaling which can simultaneously transmit signals in both directions.

While bi-directional signaling appears to be an easy way to increase the signaling rate, its benefits are mitigated by the introduction of reverse-channel crosstalk, which is not present in unidirectional signaling. It is also more complex (and costly) to implement because receiving a signal involves canceling out the signal that is simultaneously being transmitted.

Canceling the transmitted signal is not a perfect process. Due to parameter mismatches, a portion of the forward traveling wave is coupled in to the estimate of the reverse traveling wave. This is a source of noise. Additionally, bi-directional signaling generates return current, introducing another noise source.

In a differential simultaneous bi-directional signaling system, two distinct sources are required at each end of the line. One of these sources drives the line, while the other is used to create a local duplicate of the transmitted signal. This duplicated signal can then be subtracted from the signal on the line with the use of a differential amplifier, thereby extracting the reverse traveling wave (the received signal).

We determined that the benefit of using bi-directional signaling would not outweigh the drawbacks, especially given that the magnitude of the reverse crosstalk was on the order of the received signal itself. In addition, it is possible to achieve the target-signaling rate without the complexity and cost of bi-directional signaling.

3.2 Single-ended vs. Differential Signaling

We chose to use differential signaling as opposed to single-ended signaling, to minimize the sources of noise. With differential signaling, the signal itself serves as a reference. Therefore, a reference need not be generated at the receiver. There is also very little power supply noise because equal and opposite currents are drawn from the drivers for the signal and its complement. Also, using bipolar, differential signaling results in no return current. The advantage of using single-ended signaling is the fact that only one line is needed to transmit a single signal. Therefore, ignoring the number of required returns, twice the number of signals can be transmitted simultaneously. Unfortunately because of the additional noise, it would most likely be necessary to decrease the transmission rate significantly in order to achieve a reasonable noise margin. In addition,

ingle ended signaling requires one return for every n signals, forcing higher data rates on the remaining data lines, and therefore even further increasing the complexity of the design.

3.3 Number of signal levels

Our design uses two signal levels. Increasing the number of signal levels results in more energy expended for each transmitted bit. Also, additional signal levels would result in a more complex receiver, and also would decrease the noise margin. In addition, while increasing the number of signal levels is very attractive for increasing the data rate, since the transmission line has such a high attenuation coefficient, it would be extremely difficult to implement a robust system given the further reduced noise margin.

3.4 Signaling Rate

In order to meet the goal of 4Gbps per link our system transmits differential signals at a rate of 4.5Gbps per link.

3.5 Signaling Convention – differential current-mode

We decided to employ current-mode signaling rather than voltage-mode. Current-mode signaling allows a reduced signal swing and it isolates the receiver from power supply noise. In addition, current mode drivers are much easier to implement on chip and with greater precision than voltage mode drivers.

3.6 Equalization

Initial analysis of a single pulse being received at the far end of the transmission system showed that inter-symbol interference was a major concern. Without equalization, ISI contributes so much noise that the eye is completely closed. Figure 10 shows the waveform of a single pulse being received at the far end of the transmission system.

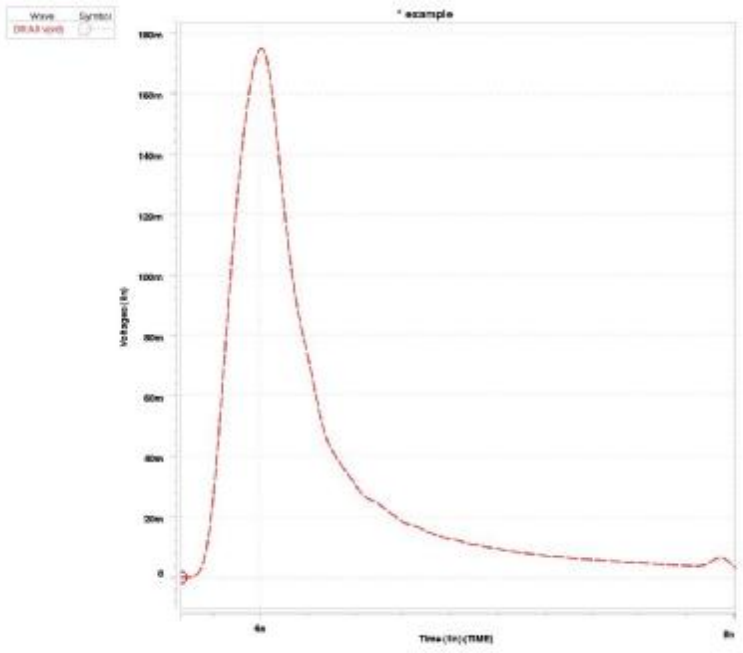

Figure 10 – Result of a single pulse at the near end, seen at the far end

As can be seen in Figure 10, the tail of a single pulse will greatly interfere with at least the next 5 pulses. A way to correct this is to use equalization.

Equalization allows for frequency-dependant attenuation to be cancelled. This is accomplished by pre-emphasizing the high frequency components of the signal where attenuation is the highest, and pre-attenuating the low frequency components. This is especially true for this channel, given the attenuation in the frequency range of interest, as can be seen in Figure 11.

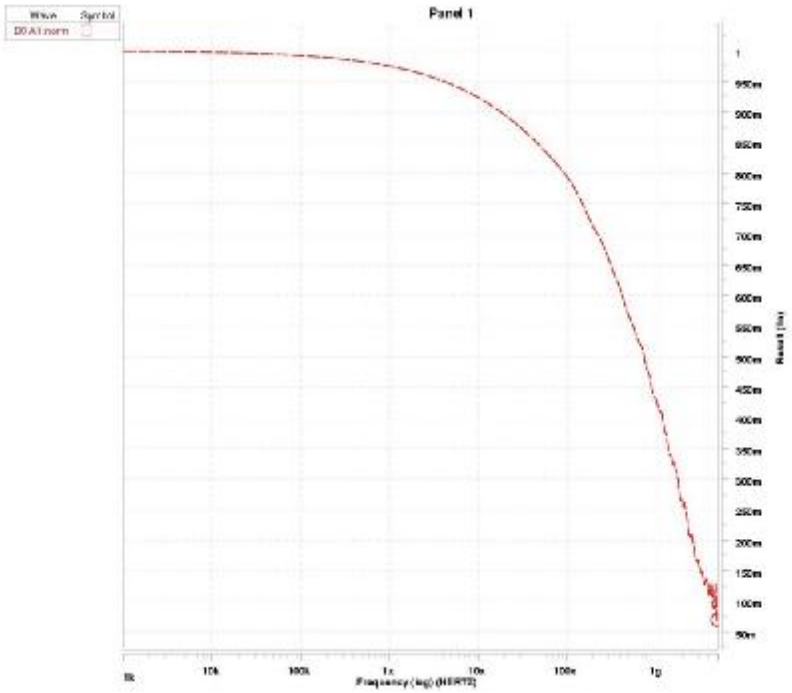

Figure 11 – Frequency Response of the Transmission Channel

As can be seen in the above figure, high frequencies above 1GHz (2Gbps), are highly attenuated, whereas very low frequency (DC) signals are not attenuated at all.

In designing the equalizer, a 5-tap design was selected, as the single pulse in Figure 10 will interfere significantly with the 5 subsequent pulses. Measurements of the received pulse were taken at bit-period intervals, starting at the pulse peak. These measurements, and their normalized values, can be seen in Table 1.

Time (s)	Voltage (V)	Normalized Voltage
5.74	0	0
6.00	0.154	1
6.22	0.0594	0.385714
6.44	0.0238	0.154545
6.67	0.0148	0.096104
6.89	0.00968	0.062857
7.11	0.00721	0.046818

Table 1: Measured and Normalized Voltages of the Received Single Pulse

To create the weights for each tap of the equalizer, equations were created to model an idealized received pulse. This pulse should have a value of 1 after one bit-period, and 0 thereafter. Using the normalized value of the voltage as the h_n term, the corresponding weight, w_n, was solved for using the following equations:

$$h1 * w1 = 1$$
$$h2 * w1 + h1 * w2 = 0$$
$$h3 * w1 + h2 * w2 + h1 * w3 = 0$$
$$h4 * w1 + h3 * w2 + h2 * w3 + h1 * w4 = 0$$
$$h5 * w1 + h4 * w2 + h3 * w3 + h2 * w4 + h1 * w5 = 0$$

The corresponding weights can be seen in Table 2.

Time (s)	Voltage (V)	Normalized Voltage	Weights
5.74	0	0	
6.00	0.154	1	1
6.22	0.0594	0.385714	-0.38571
6.44	0.0238	0.154545	-0.00577
6.67	0.0148	0.096104	-0.03427
6.89	0.00968	0.062857	-0.01168
7.11	0.00721	0.046818	-0.01222

Table 2: Resulting Equalizer Weights

An additional constraint of the design is that any current mode driver is limited to 20mA. Since in the worst case, the absolute values of the weights shown in Table 2 could sum together, the maximum current was limited to:

$$I = \frac{20mA}{\sum_{i=1}^{6} |wi|} = 13.79mA$$

Using the weights found in Table 2, a PWL source was created in spice, and the equalized pulse received at the far end can be seen in Figure 12.

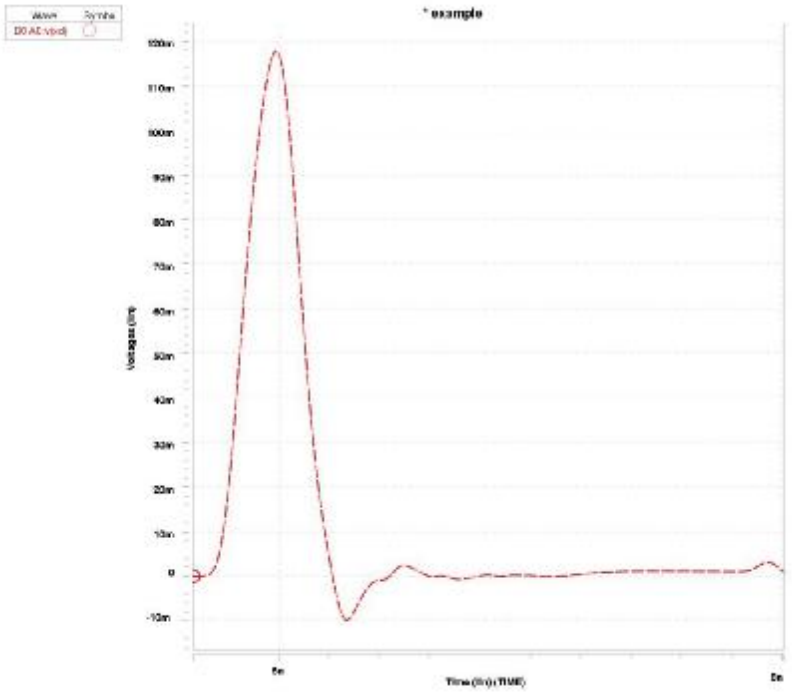

Figure 12 – Single Equalized Pulse as Seen by the Receiver

There is still a small amount of inter-symbol interference between subsequent pulses, but it is small compared to the original single pulse. Eye diagrams generated using the equalizer now show a well-defined eye. However, there is a significant reduction in the noise margin due to crosstalk.

Figure 13 is a high-level block diagram of the equalization filter implementation. In this figure, a unit delay is one bit-period ($1/4.5 \times 10^{9}$).

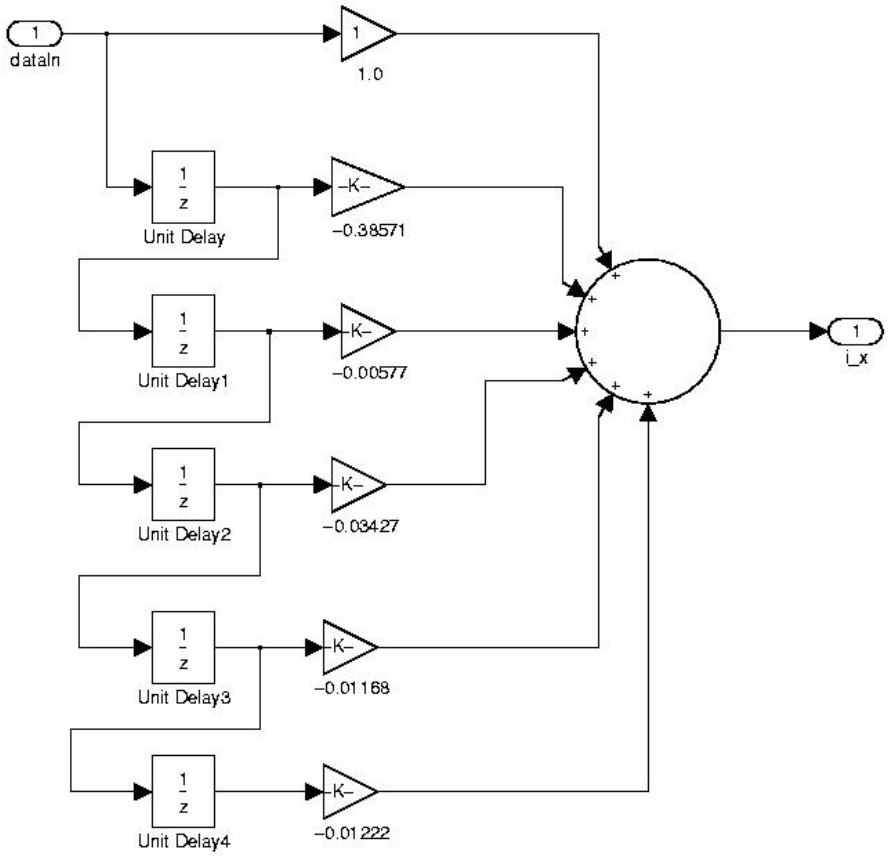

Figure 13 – Equalization Filter (1 unit delay = $1/(4.5 \times 10^9)$)

3.7 Crosstalk Control

A signal transitioning on a transmission line induces traveling waves on adjacent transmission lines due to the coupling capacitance as well as the mutual inductance between the neighboring lines.

In this system, crosstalk is a major issue. When transmitting a simple single unequalized pulse, the resulting crosstalk on the neighboring transmission line is quite large, as can be seen in Figure 14.

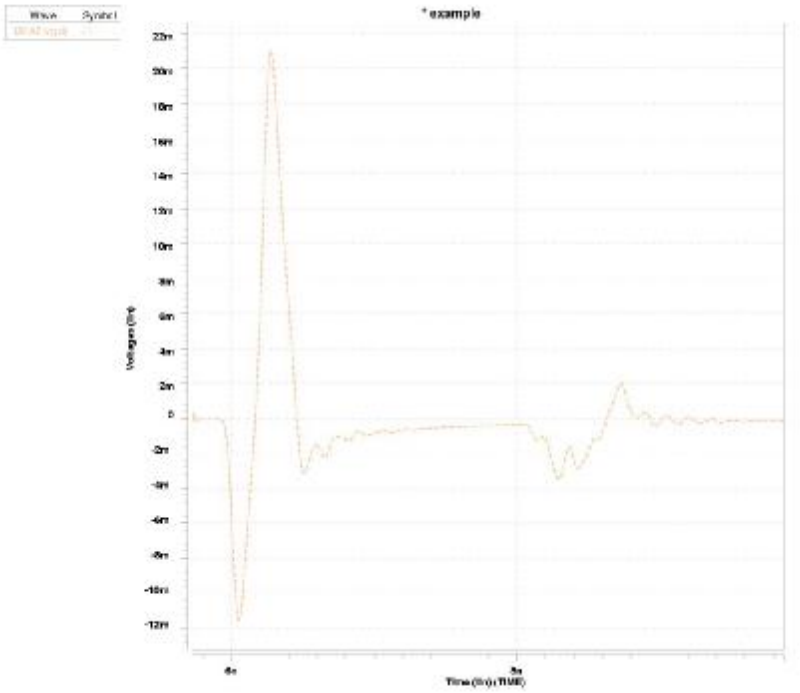

Figure 14 – Crosstalk Seen at Receiver From a Lone Pulse

In the figure above, an alternating sequence of ones and zeros causes the components of the crosstalk to line up in the worst possible case. Measuring the magnitude of the crosstalk at bit-period intervals gives the worst-case crosstalk, of 43mV.

This crosstalk can be theoretically cancelled exactly by transmitting a sequence of inverse pulses down the victim line. An important item to note is that the crosstalk seen is proportional to the derivative of the signal transmitted on the aggressor line. Therefore, when the aggressor line is reaching the maximum value of the pulse and its derivative is zero, the victim line is crossing zero. As a result, for every single pulse transmitted on the aggressor, two pulses are generated in the same bit-period on the victim. Therefore, in order to cancel the crosstalk, a cancellation pulse sequence must be transmitted at twice the rate of the aggressor line, and aligned with the aggressor.

After much analysis and simulation, a method was devised to cancel the crosstalk. Figure15 is a high-level block diagram of the crosstalk cancellation system. In this figure, a unit delay is one half of a bit-period since it operates at twice the data rate of the overall signaling system, and uses the data from nearby lines to determine what the change in transmitted current following equalization will be for any given bit-period.

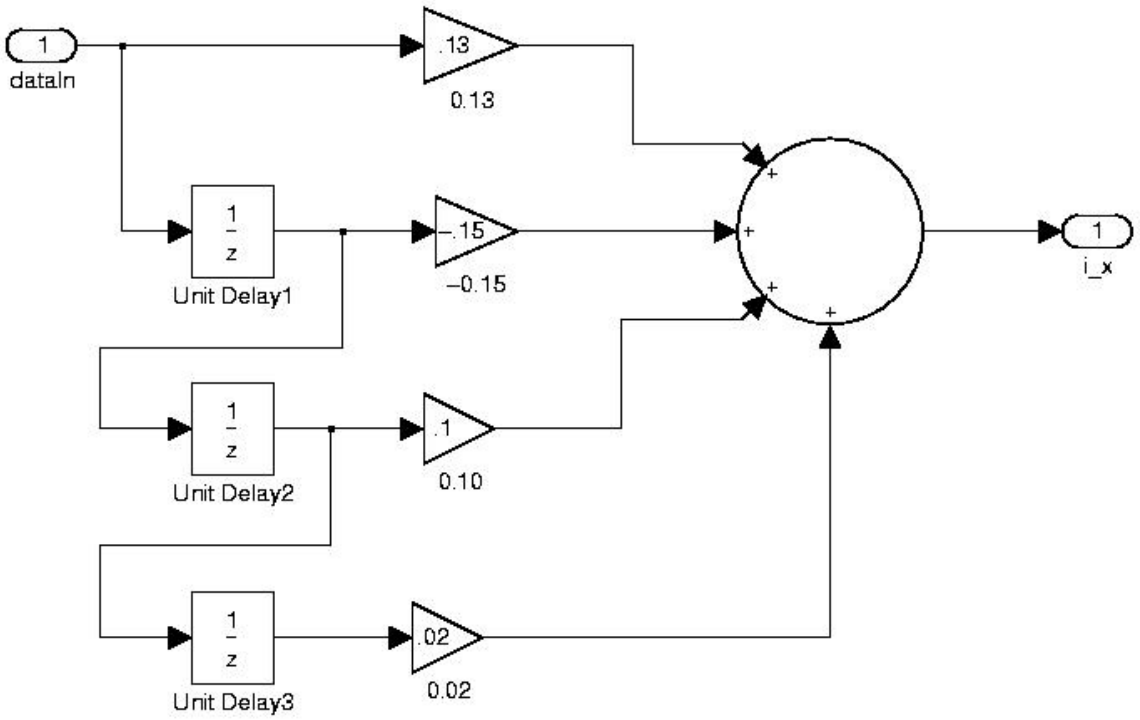

Figure 15: Crosstalk Cancellation Block Diagram (Delay = $1/(2*4.5x10^9)$)

Four pulses are required to cancel the crosstalk, spanning two data bit-periods. As a result, the cancellation system must remember what the change in current was from the last bit-period. A vector of 4 weights was chosen to scale the anticipated current change: (0.13, -0.15, -0.10, 0.02). For example, if a 0 to 1 transition were to occur, assuming a 20mA current change, the resulting currents generated would be:

$$20mA*(0.13,-0.15,-0.10,0.02) = (2.6mA,-3.0mA,-2mA,4mA)$$

In this example, the 2.6mA current would be transmitted at the same time as the 0 to 1 transition on the aggressor, followed by a –3.0mA pulse half a bit-period later. As the aggressor transmits the second bit, the victim line transmits the –2mA pulse and the first pulse to cancel the latest transition, and another half bit-period later, the 4mA pulse is sent, along with the second cancellation pulse of the next aggressor bit. The result of this method can be seen on the 0 to 1 transition in Figures 16 and 17.

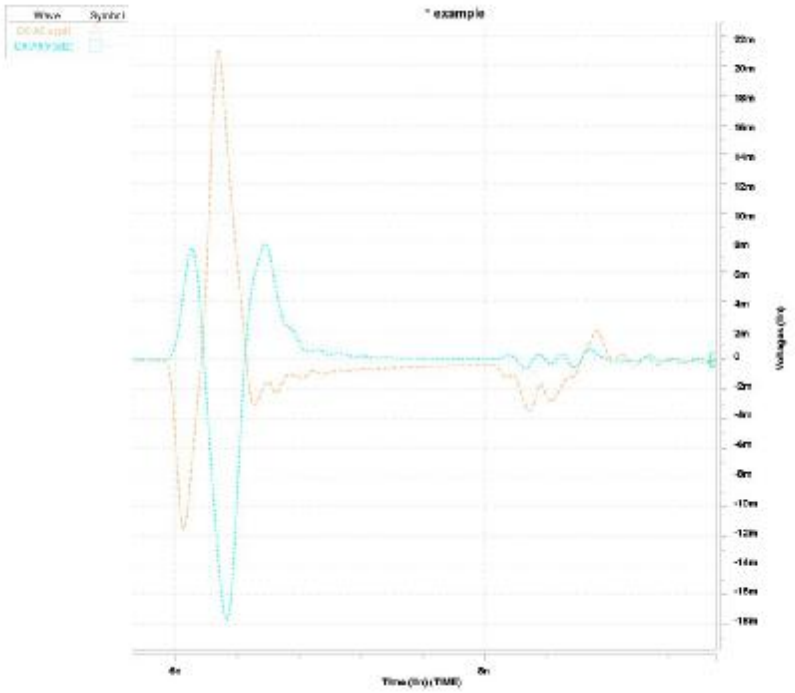

Figure 16 – Crosstalk and Crosstalk Cancellation Signals of a Lone Pulse

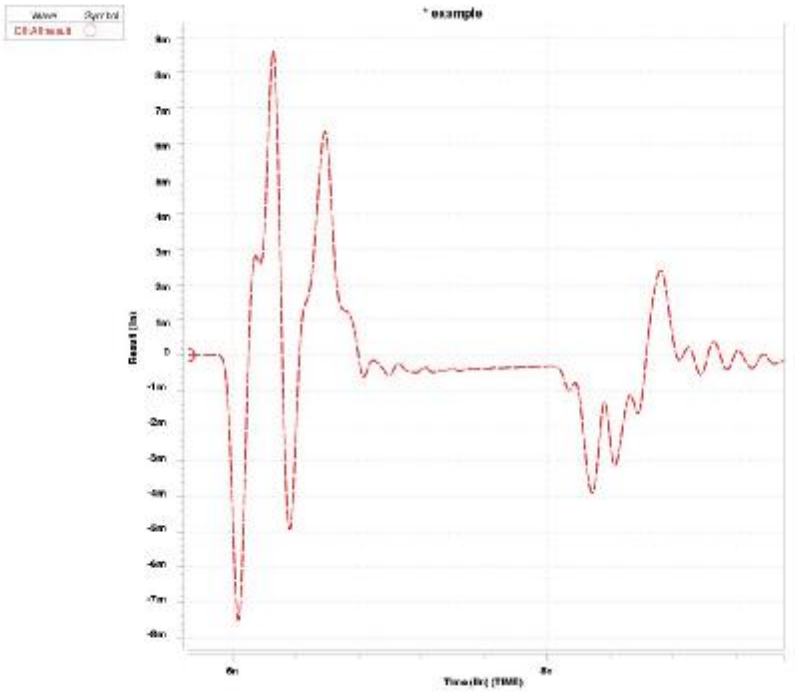

Figure 17 – Resulting Crosstalk Following Cancellation of a Single Pulse

From this waveform, the worst-case remaining crosstalk is the maximum sum of the absolute value of the waveform at 222ps intervals. Using this method, the worst-case crosstalk is now measured to be 35mV and exists for the alternating 1010 bit sequence.

Figures 18 through 21 show additional crosstalk scenarios and the remaining crosstalk on the lines following cancellation, with equalization active.

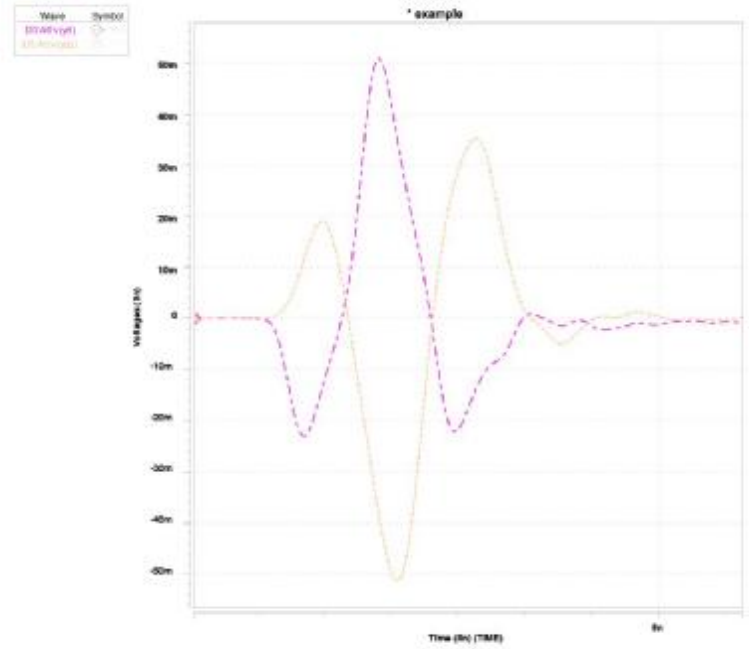

Figure 18 – Crosstalk and Crosstalk Cancellation Signals of a Lone One Pulse

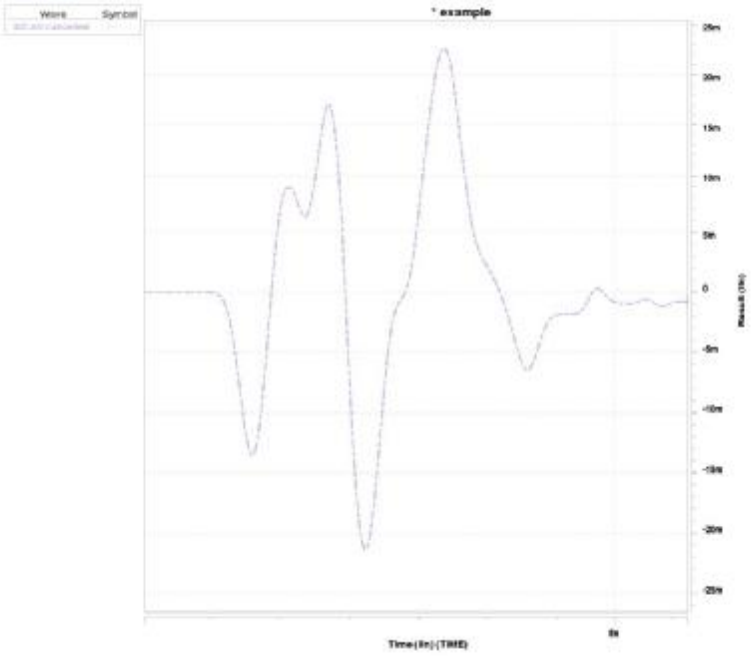

Figure 19 – Resulting Crosstalk Following Cancellation of a Lone One Pulse

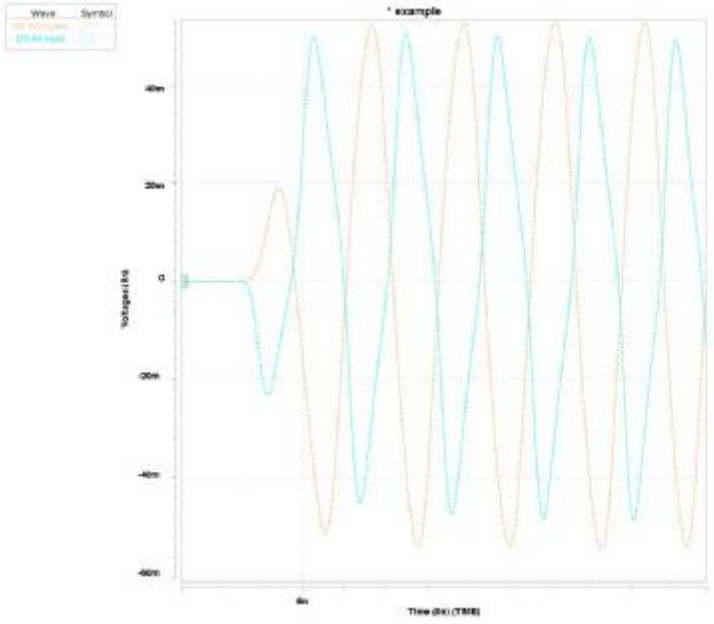

Figure 20 – Crosstalk and Crosstalk Cancellation Signals of
an Alternating 1010 Sequence

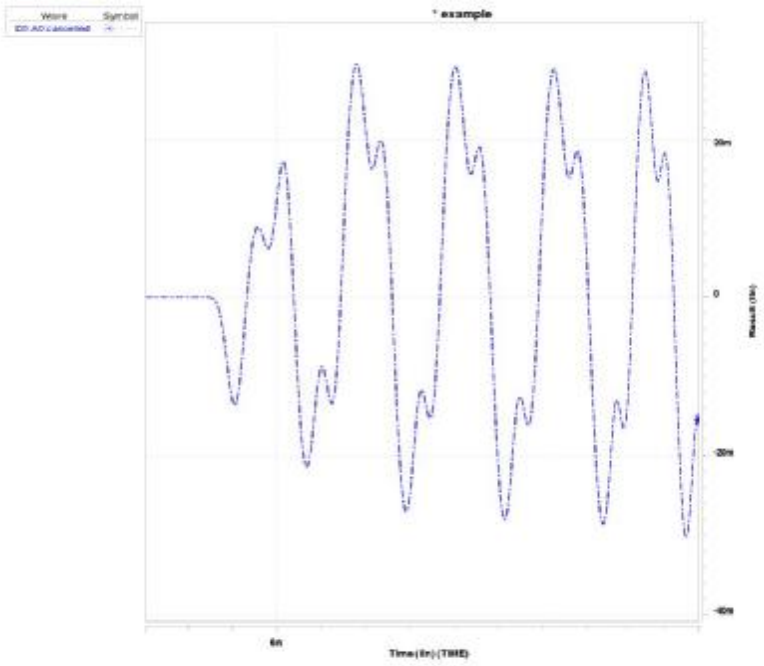

Figure 21 – Resulting Crosstalk Following Cancellation of an Alternating 1010 Sequence

With equalization and crosstalk control running on the same line, there is the possibility of going above the supply voltage. Currently, the most current the equalizer can supply is limited to 20mA. By itself, 20mA * 25Ω = 0.5V < 1.2V (since our design is source terminated as well as receiver terminated). However, crosstalk control must also be accounted for.

For crosstalk control, the worst case is for a maximum swing in current on adjacent lines simultaneously, by an alternating 1010 pattern. This causes a (0.23, 0.17) multiplier vector to be generated for crosstalk control. In this switching pattern, the equalizer generates +/- 18.53mA current pulses. The resulting currents generated by the equalizer are:

$$2*(18.53mA-(-18.53mA))*(0.23, 0.17) = (17.05mA, 12.6mA)$$

The maximum current the crosstalk controller will generate is 17.05mA. In the worst case, its equalizer is generating a 20mA source, so a total current of:

$$17.05mA + 20mA = 37.05mA$$

This current is fed into a 25Ω impedance, generating a voltage of:

$$37.05mA * 25\Omega = 0.926V < 1.2V$$

Even with the worst-case data sequence and both the equalizer and crosstalk cancellation active, the maximum voltage generated on the source end of the line with best-case termination is less than the supply voltage.

In the worst-case termination:

$$R_t = 60\Omega, \text{ so } R_{eff} = 27\Omega, \text{ and } V = 27\Omega * 37.05mA = 1.01V < 1.2V$$

Following the implementation of equalization and crosstalk control, the eye-diagram is now quite clean and crosstalk no longer consumes a large percentage of the noise margin. The resulting eye-diagrams can be seen in Figures 22 and 23.

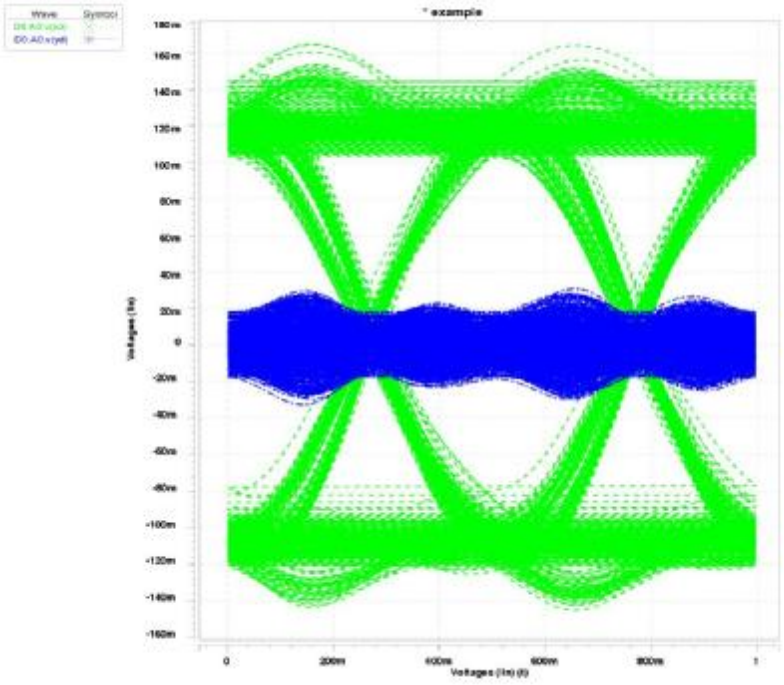

Figure 22 – Eye Diagram and Resulting Crosstalk for a Pseudo-Random Bit Sequence with Lone Ones and Zeros

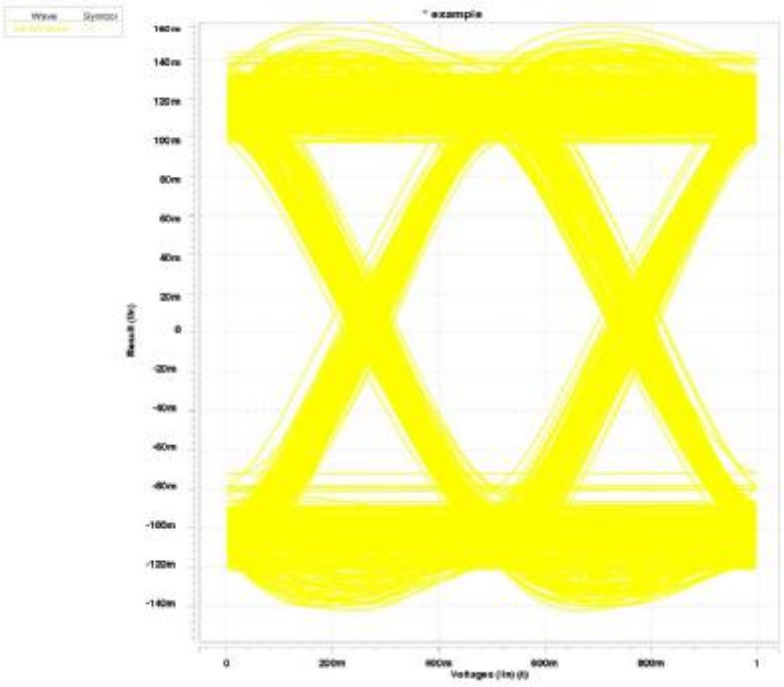

Figure 23 – Resulting Eye Diagram After Subtracting the
Worst Case Crosstalk from the Received Signal

3.8 Signaling Details – levels, rise-time, impedance

In our design, a 20mA current source is used to generate the transmitted signal. Because this is a differential system, a positive wave is induced on one of the signal lines, while a negative wave of equal magnitude is induced on the other signal line. As described above, equalization and crosstalk control are performed on the data before it is transmitted. Equalization adjusts the signal being sent on the aggressor line, while crosstalk control injects a current onto the victim line in order to cancel much of the crosstalk signal induced by the aggressor line.

Our signaling levels were chosen due to the available building blocks. The chosen system provides a signal that is powerful enough to overcome fixed noise sources, but it is not unnecessarily large as might be the case with a voltage signaling system referenced to the power supply. Therefore the energy/bit is kept at a level much less than that of a full-swing voltage-mode signaling system.

We chose a rise time of ¼ of the cycle time. This means that a signal is stable for at least ½ of the cycle time, and it can be transitioning the other ½ of the time. Therefore, the signal will be stable long enough to significantly exceed the aperture time requirement of 20ps. There are a few benefits to a slow rise time. First, it results in less cross talk. This is because cross talk on a victim line is proportional to the derivative of the signal on the aggressor line. Additionally, a slow rise time excites fewer parasitics of a transmission line.

We chose to terminate the system at both ends in order to minimize reflections. To determine the characteristic impedance of the line, we generated a TDR trace. This trace indicates that 50Ω termination resistors should be used, as can be seen in Figure 37.

3.9 Driver, Receiver, Termination Circuits

We chose to terminate both transmitter and receiver side to reduce reverse crosstalk and reflection noise. Double termination scheme consumes more power than single termination, but it is necessary to minimize the noise in high speed signaling.

Our driver is capable of providing up to 20mA of current. The current source is used to drive the differential signal. Figure 24 shows the high level diagram of the driver circuit.

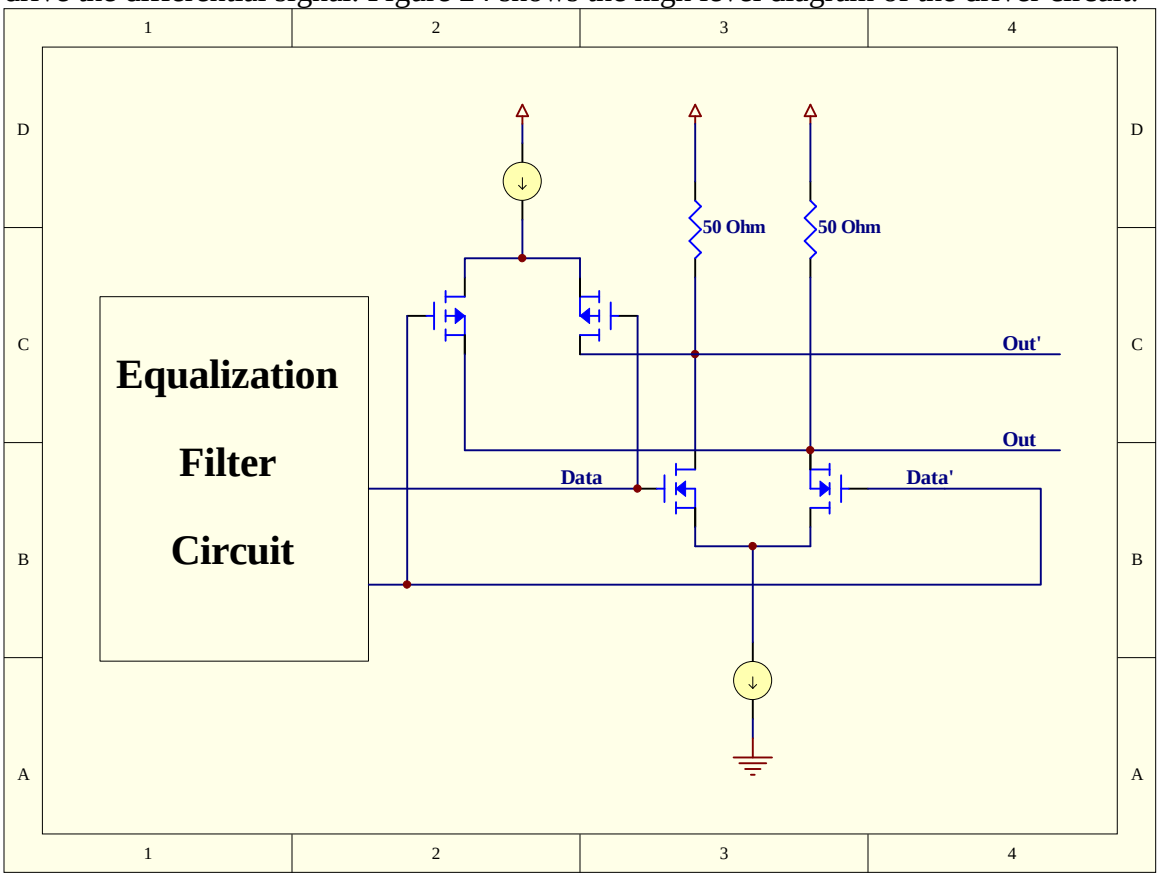

Figure 24 – Transmitter Circuit Model

The receiver circuit includes the 100-ohm differential on-chip termination resistors and offset compensation circuits. The receiver-offset compensation is discussed in section 5.2, Receiver offset and sensitivity.

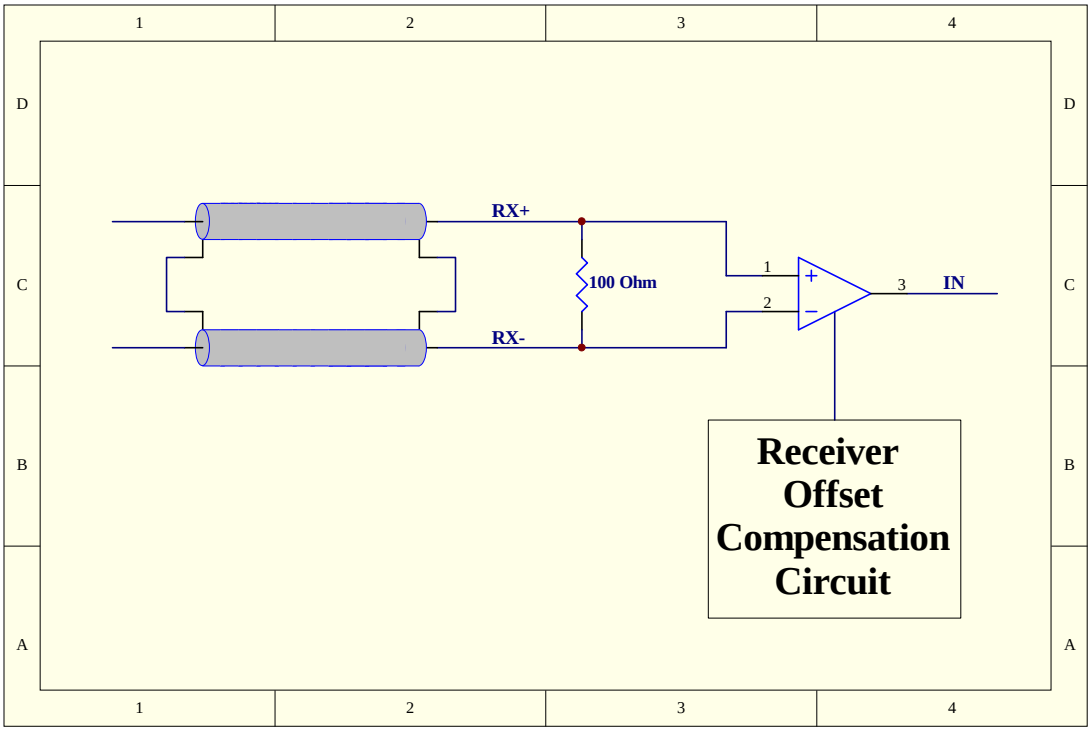

Figure 25 – Receiver Circuit Model

4 Details of Timing and Synchronization

This section describes the timing specification for the line card and switch card. Each line card and switch card includes one local 225Mhz crystal oscillator to generate the system, transmitter, and receiver clocks.

4.1 Overall Timing Concept

In our signaling system design, we decided to use bundled closed loop timing. Using bundled closed loop timing, we can cancel skew of the transmitter clock, the delay of the transmission line, the output delay of the transmitter and the aperture offset of the data path. Figure 26 shows overall view of how the clock is transmitted along with the data in our design.

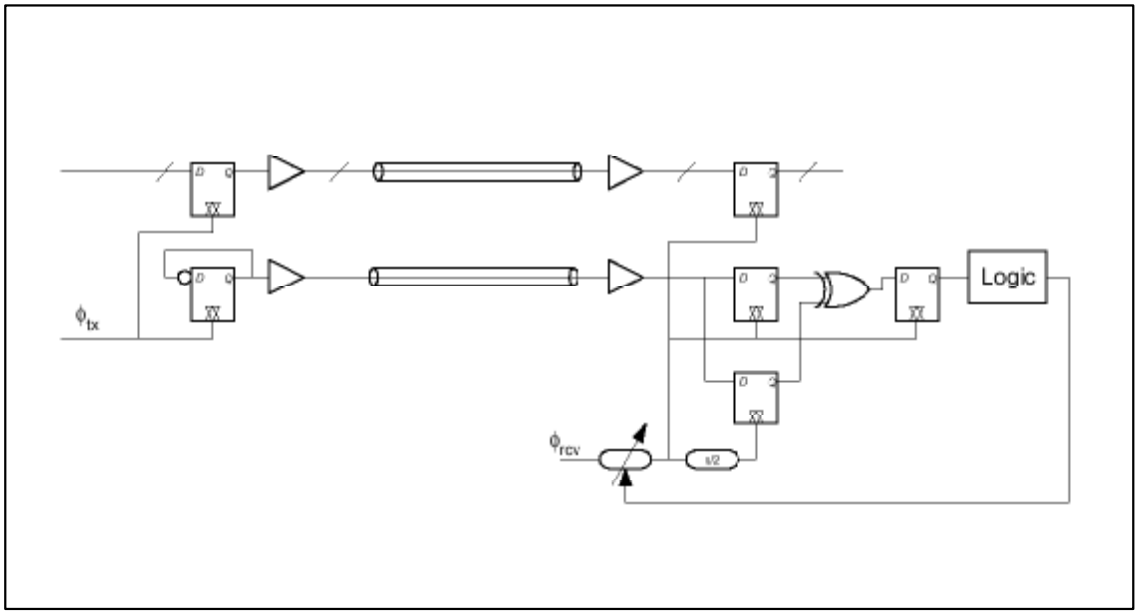

Figure 26 – Bundled Closed-Loop Timing

The chips in the line and switch cards run at a core frequency of 225MHz and at 2.25Ghz for transmitters and receivers. In order to generate the crosstalk cancellation pulses, the transmitter also uses a 4.5GHz clock. Both the 2.25GHz and 4.5GHz clocks are generated using the on-chip PLL.

Even though we are using 225MHz crystal oscillators in both line and switch card, we need to synchronize received data with local 2.25GHz clock. Our crystal oscillator has 20ppm accuracy. One clock can be slightly faster than the other and this mismatch must be compensated using a plesiochronous synchronizer.

Once the data is received with transmitted clock, it is synchronized with the local 2.25GHz clock using a plesiochronous FIFO synchronizer discussed in lecture. With a plesiochronous FIFO synchronizer, we need to send a periodic null character to

resynchronize. Figure 27 shows the simple block diagram implementation of synchronizer.

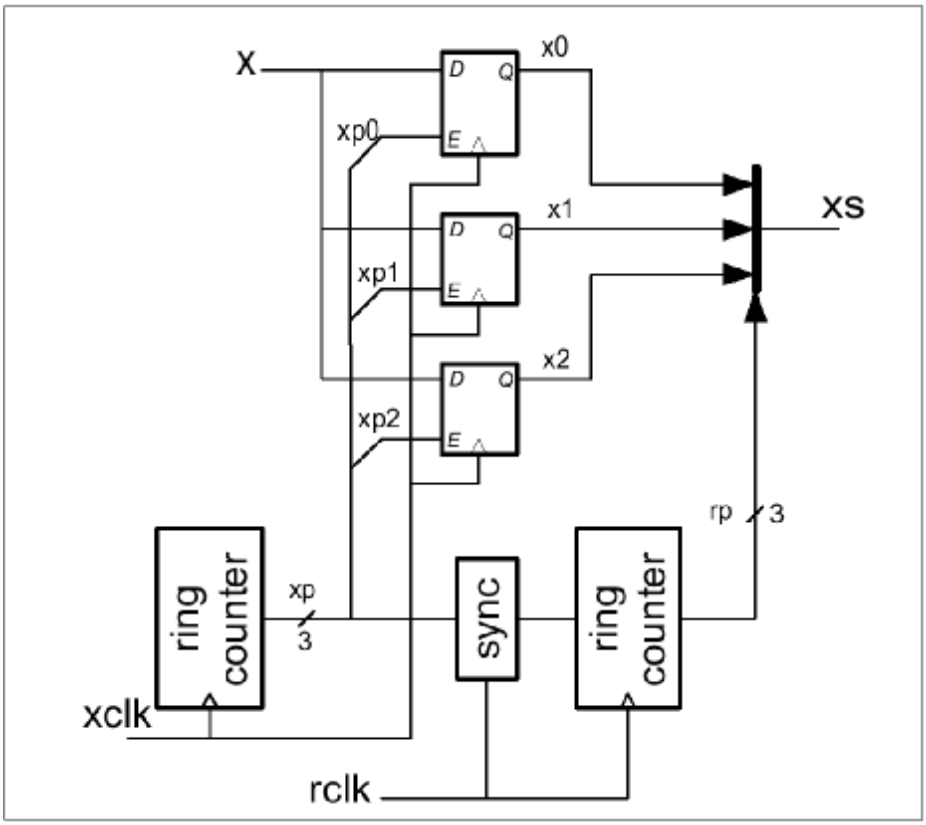

Figure 27 – Receiver Synchronizer Block Diagram

4.2 Timing Circuit

In our design, each line card has an independent crystal oscillator running at 225Mhz. Also, the switch cards include one crystal oscillator running at 225Mhz.

Figure 28 below shows one of the transmitters. The on-chip PLL multiplies the local reference clock oscillator running at 225Mhz by factor of 10. Using both edges of the 2.25Ghz clock, data is serialized and sent to the current mode transmitter at 4.5Gbps. Our transmitter implementation includes crosstalk cancellation that requires a 4.5GHz clock. This clock is also generated by the same PLL.

Figure 28 also shows the clock transmitter circuit. Our clock is transmitted differentially at a frequency of 2.25GHz.

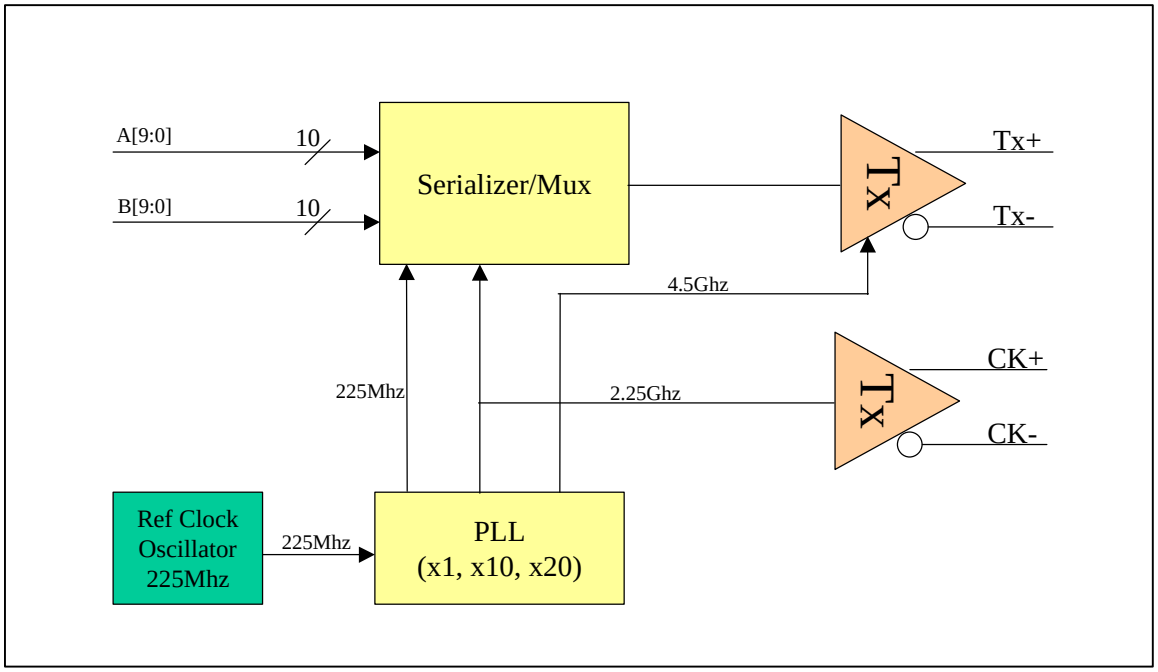

Figure 28 – Transmitter Timing Block Diagram

Figure 29 below shows the block diagram of PLL implementation including VCO, loop filter, charge pump, and phase comparator, as well as the divide down counters that generate the separate clocks.

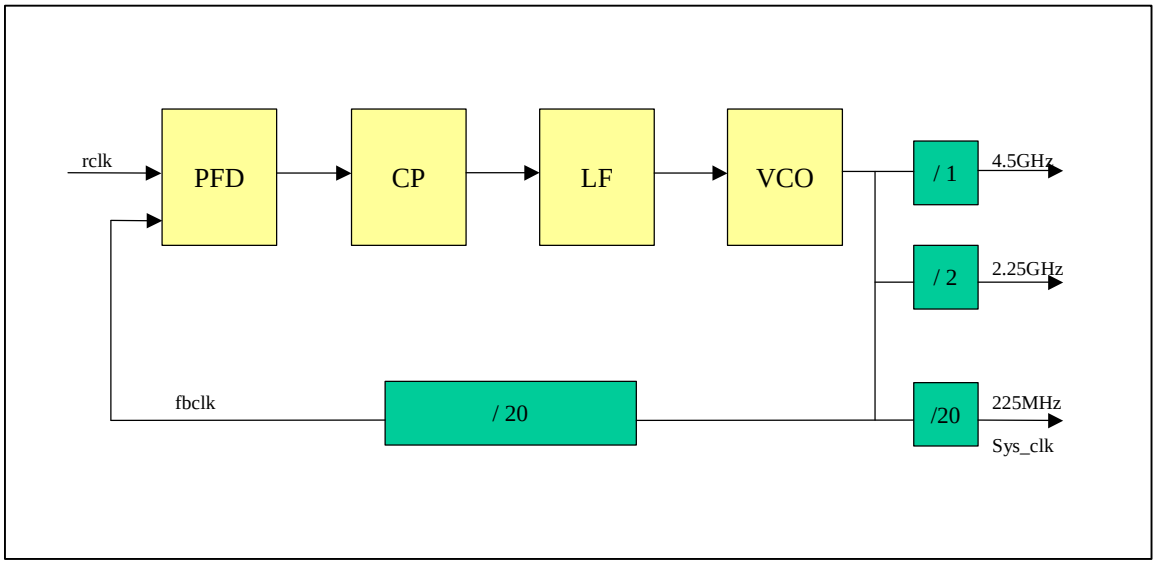

Figure 29 – PLL Block Diagram

The serializer has two 10-bit inputs to allow continuous 4.5Gbps transmission. While the first 10 bits are being transmitted, the second 10 data bits are latched into the serializer. This allows our transmitter circuit to operate without being limited by the 500ps flip-flop delay, as can be seen in Figure 30.

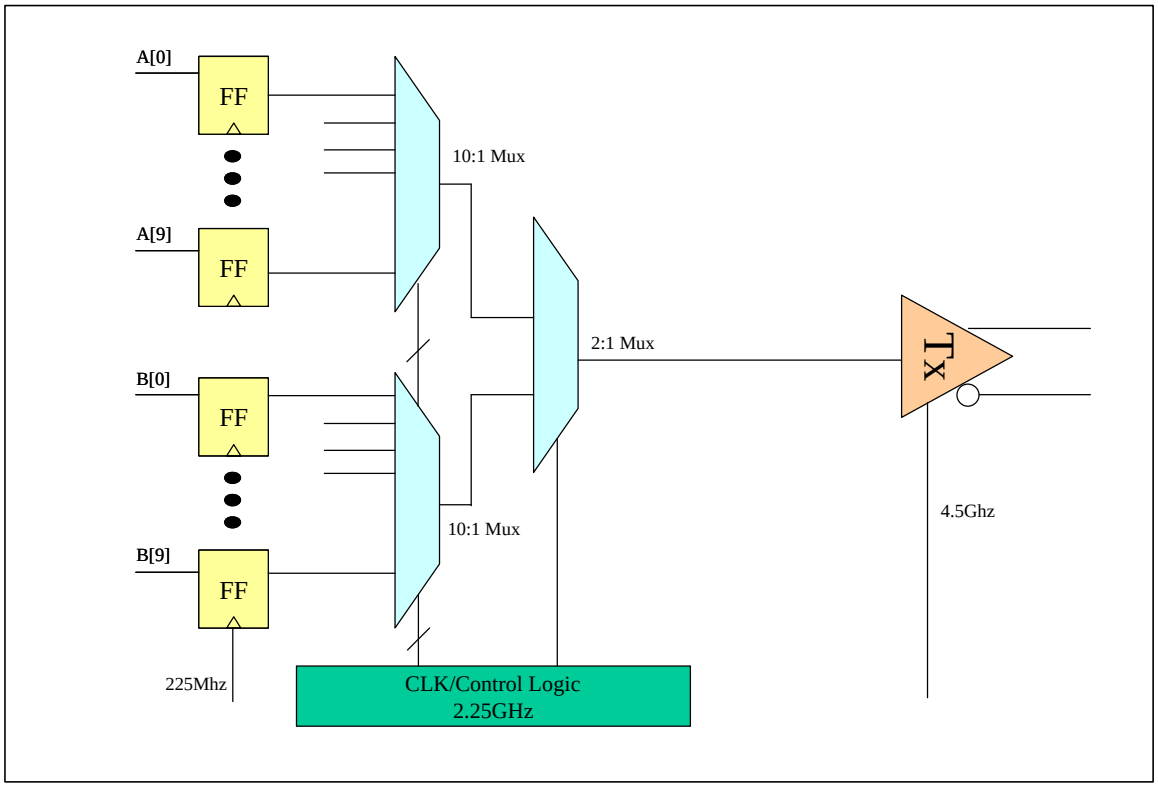

Figure 30 – Serializer Diagram

At the receiver side, the incoming data stream is sampled by the received clock and synchronized with the local clock. After synchronization, the data is de-serialized to be used at the core 225MHz clock frequency. Figure 31 shows the block diagram of the receiver circuit.

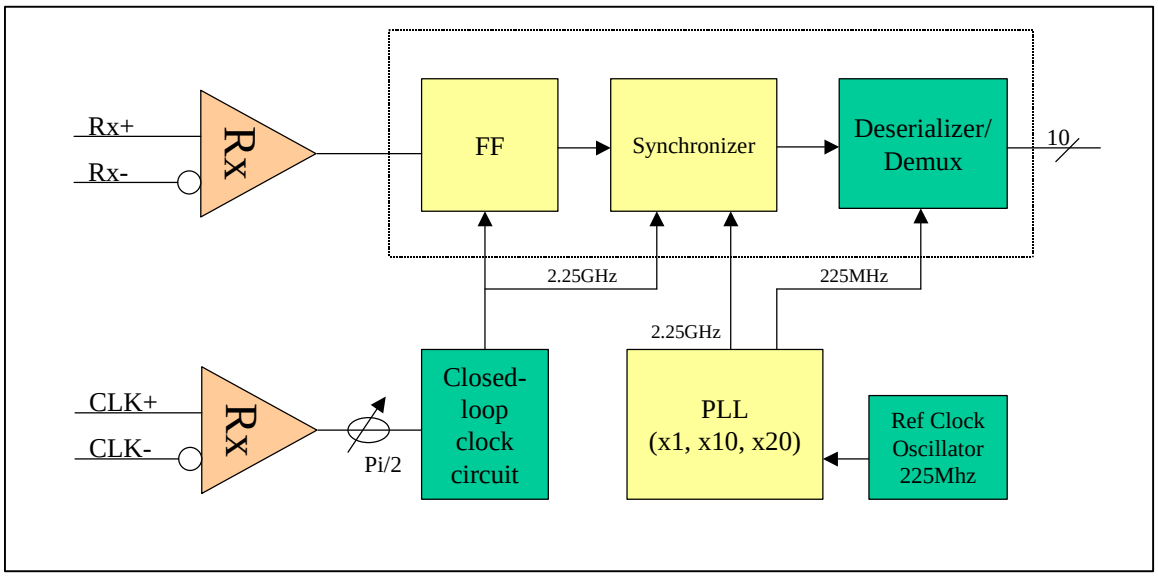

Figure 31 – Receiver Timing Block Diagram

Detailed receiver flip flops, synchronizer, and deserializer circuit diagram is shown in Figure 32 below.

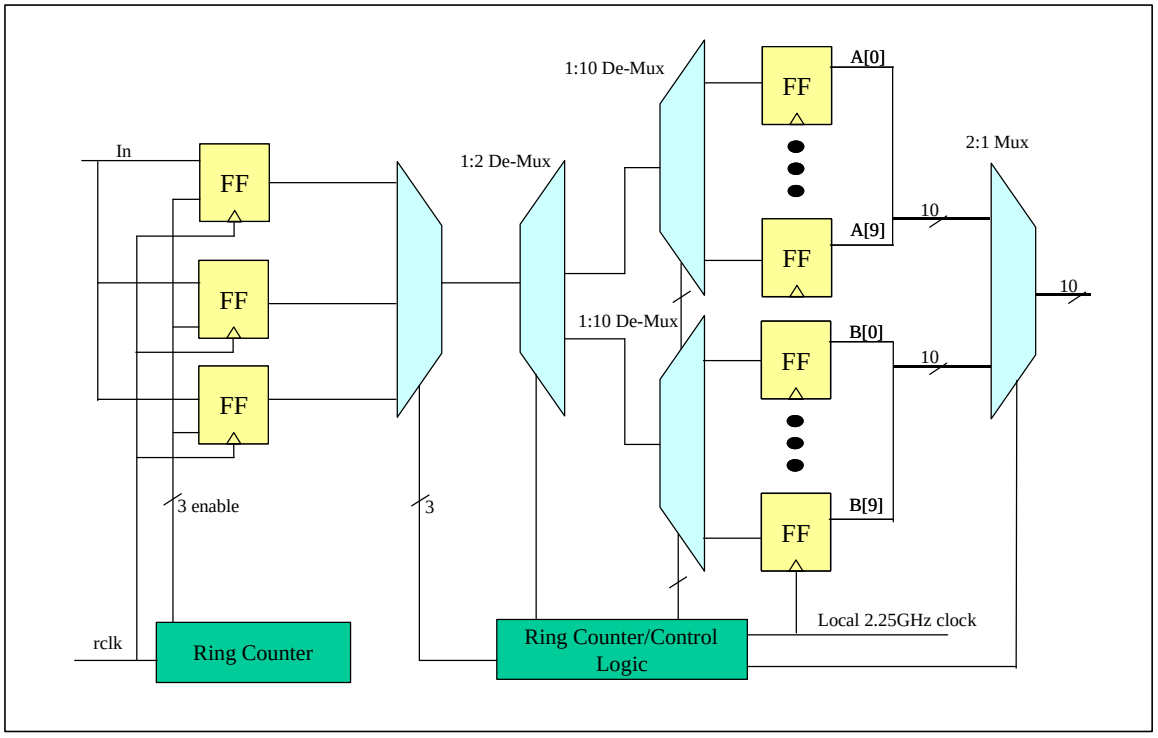

Figure 32 – Receiver Timing Block Diagram

5 Noise Budget

5.1 Transmitter Offset

The transmitter circuitry, while well designed, is still susceptible to process variations, causing the worst-case current drive to be within 5% of the nominal value. Since the transmission line attenuates the signal, the worst case is for the current source to be 95% of its nominal value. Therefore, k = 0.05.

5.2 Receiver offset and sensitivity

The receiver circuitry, since it is also susceptible to process variations, can have an offset of +/- 40mV, as well as a sensitivity of 10mV. The gain bandwidth product is 10GHz, giving 2.25GHz for the signal bandwidth, and the remainder for gain. While the sensitivity restriction must be accepted, the offset voltage can be cancelled by building additional circuitry around the receiver that measures the offset voltage during chip initialization and adjusts the inputs accordingly to achieve 0mV offset.

On a per-latch basis, a control register and a digital to analog converter can be used to statically trim the offset. The output of the comparator can be measured using an analog to digital converter and either sent directly to a set of output pins, or to an internal FSM that will automatically modify the control register settings until the static offset is 0. This ported option would allow a test engineer to characterize the offset the manufacturing process is generating, as well as to characterize the efficiency of the FSM in canceling the offset.

Another option this scheme can allow is for the control register to be loaded with a test vector through a boundary scan chain, such as JTAG, should the FSM fail, or for chip testing.

A block diagram of the process is shown in Figure 33.

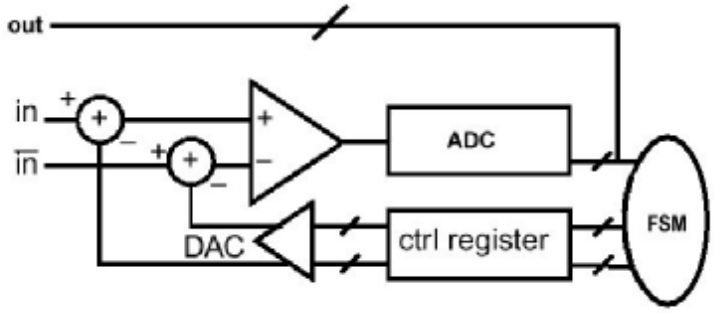

Figure 33: Input Offset Cancellation Block Diagram

After offset cancellation, the receiver offset is on the order of uV and is therefore negligible. Sensitivity however, is still an issue and must be included in the noise budget, at 10mV.

5.3 Crosstalk

As presented, the backplane model is highly limited by crosstalk. After building the complete model, crosstalk was found to be approximately 72mV for a 220mV receiver swing, giving a k of 0.33. Clearly, this is a major limitation to a robust system design. Therefore, a method of canceling the crosstalk as already discussed was designed. The resulting worst-case crosstalk was found to be 35mV for the same receiver swing, giving a much more manageable k of 0.16.

5.4 Inter-symbol Interference

Again, as the system was initially presented, with only a simple current source driver without equalization, intersymbol interference (ISI) was a major concern, preventing an open eye from being received. After equalization was implemented, an open eye was generated as can be seen in Figure 22. In addition, the resulting receiver end pulse response can be seen in Figure 12. In this figure, it can be clearly seen that a bit-period following the initial pulse, there is no residual voltage on the line, meaning that ISI is 0.

Termination mismatch is a problem in this process as resistors have a maximum range of +/- 20% of their nominal value. Again, as attenuation is a problem over the backplane transmission line, the worst-case termination is for a low resistance, in this case 40Ω. As a result, the received signal is attenuated even more. In addition, due to the termination mismatch, the incident signals will reflect off of the receiver. Since this design is also source terminated, and is unidirectional, for reflections to be an issue, they must reflect

off of both the receiver termination and the source termination. From the telegrapher's equation, for both the source and receiver,

$$krt = krs = \frac{40-50}{40+50} = -0.11$$

Therefore, the total interference is the result of the multiplication of krs and krt:

$$k = krt * krs = -0.11 * -0.11 = 0.012$$

5.5 Gaussian Noise

Gaussian noise comes from a few sources in this system. The first is thermal noise in the termination resistors.

$$Vjr = (4 * kb * T * R * B)^{1/2}$$
$$Vjr = (4 * 1.38X10^{-23} * 300K * 50\Omega * 2.25GHz)^{1/2}$$
$$Vjr = 43uV$$

Compared to other sources of gaussian noise, this is negligible.

Perpendicular crosstalk and crosstalk of any kind is not an issue in the line and switch card traces as they are shielded by Gnd and Vdd planes.

There is a source of gaussian noise in the backplane of 5mV.

Total gaussian noise in the system is 5mV.

5.6 Power supply noise

Since the system is a completely differential signaling system, there is very little AC power supply current and as a result there is very little self-induced power supply noise. As in any industrial design, the power supply for the I/O circuitry is completely isolated from the core supply through separate supply pins and a separate power and ground distribution network. In addition, a guard ring may be used to further isolate the I/O substrate from ground bounce. As a result, power supply noise is not an issue in this design.

5.7 BER Calculation and Summary

The signal swing value includes line attenuation as it was obtained from spice with both the equalizer and crosstalk cancellation active.

Signal Swing (dp-dn)	(288mv-(-288mV))	576mV
Gross Margin		288mV
Transmitter Offset	0.05	14.4mV
Termination Offset	0.11	31.7mV
Crosstalk	0.16	46mV
Reflections	0.012	3.5mV
KN	0.332	95.6mV
Receiver offset + sensitivity		10mV
Bounded Noise		105.6mV
Net Margin		182.4mV
Gaussian Noise		5mV
VSNR		36.5
BER		$1.0x10^{-289}$

Table 3: Peak Eye Opening BER Calculation

This analysis assumes that the signal will be sampled at the highest point in the eye opening. Following the timing analysis, the BER calculation will be revisited with a reduced signal swing based on the total jitter and skew in the system.

The BER limit along with the eye diagram can be used to generate a timing margin. (Note: the eye diagram simulation includes termination offset, attenuation, and transmitter offset, and as a result, the signal swing is reduced and the other k factors are not included in the table as they are already included in the analysis.)

Signal Swing	440mV
Gross Margin	220mV
Crosstalk	35mV
Reflections	2.6mV
Receiver offset + sensitivity	10mV
BER	1.0e-20
VSNR	9.597
Gaussian Noise	5mV
Net Margin	48.0mV
Bounded Noise	47.84mV
Required Opening (differential)	95.84mV
Required Opening (one-line)	47.9mV

Table 4: Reverse BER Calculation for Timing Margin Calculation

From the eye-diagram, this provides a 133ps sampling window that meets the BER.

6 Timing Budget

This section describes the timing budge of our system. All possible jitter and skew sources have been identified and included in our timing margin analysis.

6.1 Skew/Jitter Sources

The jitter and skew sources have been carefully analyzed in order to calculate timing margin. Table 5 shows the jitter and skew sources and their values at the transmitter side.

Transmitter side jitter	
Crystal Oscillator	20ps p-p
VCO	22.2ps p-p at 4.5GHz
Phase Comparator	20ps p-p
Flip Flop	50ps (10% of 500ps delay)
Transmitter side skew	
Flip Flop	50ps (10% of 500ps delay)

Table 5: Transmitter Side Skew/Jitter

Table 6 below shows the jitter and skew sources and their values at the receiver side.

Receiver side jitter source	
Crystal Oscillator	20ps p-p
VCO	22.2ps p-p at 4.5GHz
Phase Comparator	20ps p-p
Flip Flop	50ps (10% of 500ps delay)
Line variation	2.8ps
Pi/2 delay line	5.6ps
Flip Flop	50ps (10% of 500ps delay)
Receiver side skew	
Line variation	5.5ps
Pi/2 delay line	11.1ps
Flip Flop	50ps (10% of 500ps delay)

Table 6: Receiver Side Skew/Jitter

6.2 Timing Margin Analysis

From the transmitter, a data bit will experience 64ps of jitter from the transmitter clock, 50ps of jitter and 50ps of skew from the transmitter flip flop, 2.8ps of jitter and 5.6ps of skew from the transmission line, and 11.1ps of jitter from the pi/2 clock delay line. The clock signal transmitted along with the data bit will experience the same skew and jitter

sources, but will not see the jitter from the pi/2 delay line. To calculate the total variation in the data and clock, the sources that can be cancelled or neglected must be defined. Since the data and clock are being generated by the same clock, then any jitter that the clock line sees, so will the data line. This means that any jitter on the clock line is tracked by the data line. In addition, to match the delays of the lines for crosstalk cancellation, variable delay elements have been added at the receiver end of the line. Since skew is deterministic, these delays can be set to correct for any skew between flip flops at the transmitter side, and hence this 50ps of skew can be removed. This now means, that the total variation seen in the data line and the clock line at the receiver is:

Data variation=50ps + 2.8ps + 5.6ps + 11.1ps = 69.5ps

Clock variation = 50ps + 2.8ps + 5.6ps = 58ps.

To meet the BER timing constraint derived in section 5.7, the sum of the data and clock variations must be less than 133ps.

69.5ps + 58ps = 127.5ps < 133ps

Once this data has arrived at the receiver, it must be synchronized to the receiver clock domain. While transmitter clock jitter could be ignored before, now it must be included in the analysis. When the first flip-flop stage in the synchronizer samples the incoming data, it is using a clock that is varying by the total variation above, plus VCO jitter and phase detector error:

Clock variation = 58ps + 44.4ps + 20ps = 122.4ps

Any flip-flop using this clock will produce an output that varies by 122.4ps + 50ps jitter + 50ps skew = 222.4ps. In addition, there is 500ps of delay through a flip-flop. Therefore, at a worst case, the output is stable after 611.2ps. This constrains the synchronizer to have at least 611.2ps*4.5e9 = 2.75 or 3 stages in the FIFO.

Now, this data must be received into the receiver's clock domain. The plesiosynchronous synchronizer will handle the drift between chip clock frequencies, so timing must guarantee that jitter will not cause the FIFO stages to latch the incorrect data. Since there are 4 stages in the FIFO, this is an effective bit-period of 4/4.5e9 = 889ps. The data arriving from the FIFO has jitter of 222.4ps and a delay of 500ps. The clock driven on chip has 64.4 ps of jitter. Part of the delay can be handled by waiting an integer number of cycles until the data has settled. Two bit-periods after the data has been received is 444ps, and is the closest approximation to 500ps without using a delay line. This leaves 56ps of offset. So the sampling window is reduced by this offset and the sum of the data and receiver clock jitter:

Sampling window = 3/(4.5e9)-56ps-222.4ps-64ps=324ps > 0

Therefore, the deserializer will have a 324ps sampling window, and timing is not violated.

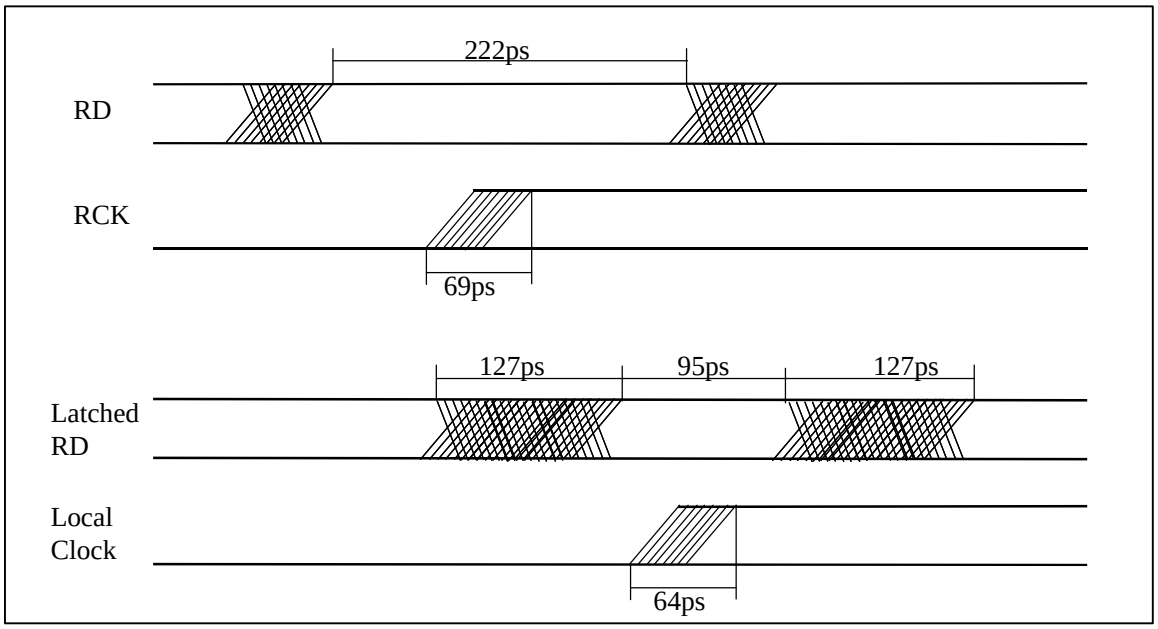

Figure 34: Timing Noise Diagram

6.3 BER Revisited

From the timing analysis, 127.5ps of the 133ps sampling window are required. Using the worst-case eye opening for this timing window, the BER is recalculated.

Signal Swing	440mV
Gross Margin	220mV
Crosstalk	35mV
Reflections	2.6mV
Receiver offset + sensitivity	10mV
BER	1.0e-20
VSNR	9.597
Gaussian Noise	5mV
Net Margin	48.0mV
Bounded Noise	47.84mV
Required Opening (differential)	95.84mV
Required Opening (one-line)	47.9mV
Actual worst-case opening	54.0mV
Net Margin	60.4mV
VSNR	12.08
BER	2.05×10^{-32}

Table 7: BER analysis revisited for worst-case timing margins

As can be seen in the above table, the BER surpasses the specification of 1×10^{-20}.

7 SPICE Simulations

This section contains various plots from SPICE simulations including eye diagrams of both random inputs and lone pulse. Also, TDR and TDT simulation results are included. From the TDT simulation, the length of the line was found to be approximately 5.53ns.

7.1 Eye Diagram – Random Inputs

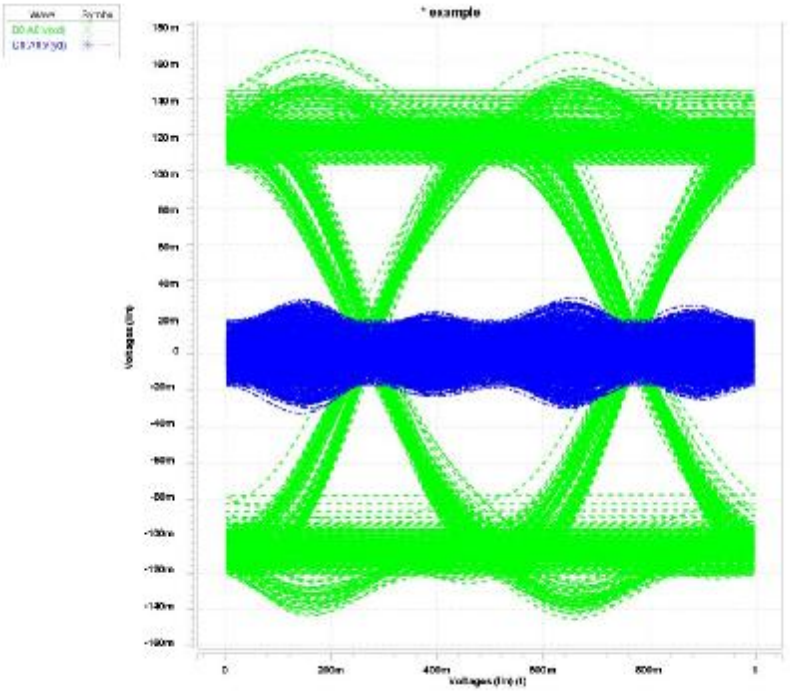

Figure 35: Eye Diagram of Random Inputs and Lone 1's and 0's
Reduced Crosstalk Shown

7.2 Eye Diagram – Lone Pulse

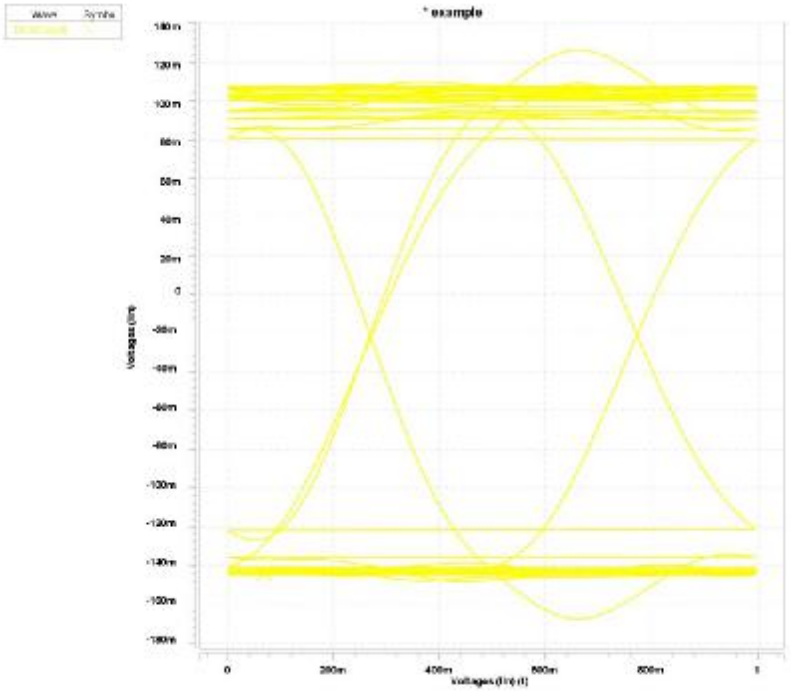

Figure 36: Eye Diagram of Lone 0 and 1 Pulses Only

7.3 TDR and TDT

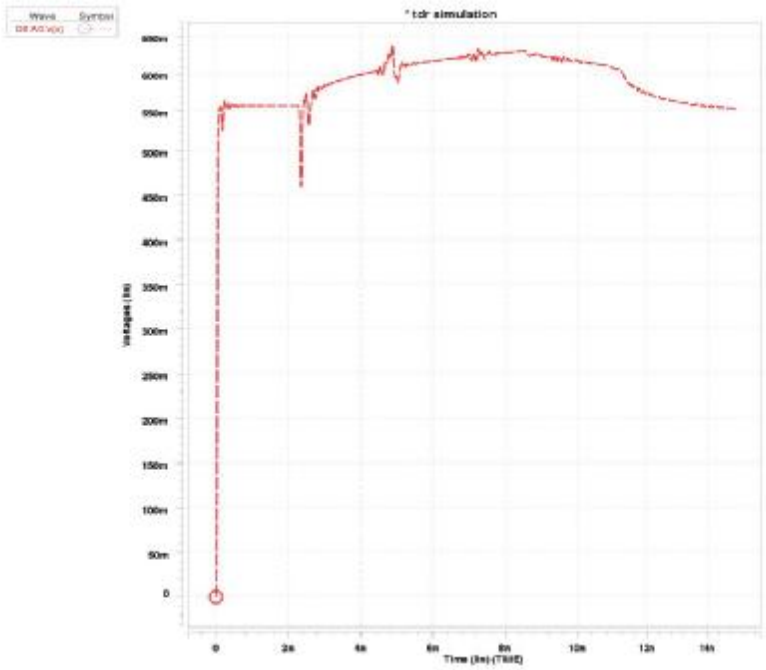

Figure 37: TDR Response of the Full Transmission Model (skeleton.sp)

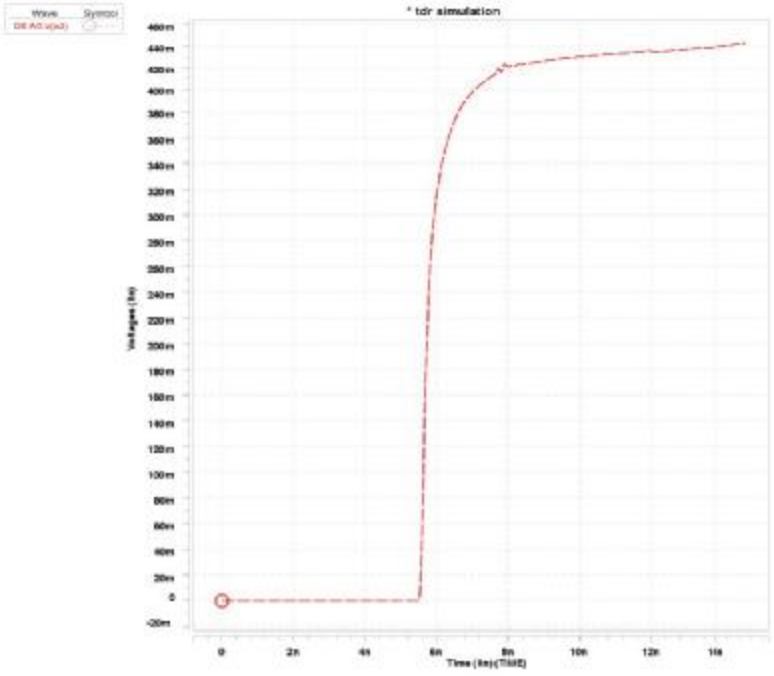

Figure 38: TDT Response of the Full Transmission Model (skeleton.sp)

8 Cost Analysis

The cost calculation of the each line card and the switch card is very difficult given following cost information:

1. Board cost - $0.1 per cm^2 if 8 or less number of layers are used, $0.2 per cm^2 if 9 or more layers are used.
2. Chip cost - $30 plus $0.05 per pin for flip chip package
3. Power cost - $20 per watt

With this information, we cannot accurately estimate the cost of each line card and switch card. First of all, we do not have sufficient information of any additional components that will contribute total size of the board. This also limits us to estimate total power of the system and cost of the power supply.

Typical crystal oscillator cost around $3 -$5. If we need a microprocessor, depending on the computational need, it could cost between $10 and $300. Also, the memory requirement determines the cost model. With the amount and type of memory required, the cost could range from tens of dollars to hundreds of dollars. Finally, an external connection via fiber cable requires a connector and a transceiver. The transceiver running at 1Gb/s – 10Gb/s can cost around $300 to $1000 per port depending on the laser wavelength. For example, short-wave 850nm 2Gb/s transceiver cost about $400 and long-wave 1380nm transceiver cost about $800.

In order to have accurate estimation of cost, we need more information on component requirements.

9 Appendix – Code Listing

9.1 Hardware Model – skeleton.sp

```
* Project Circuit file

.subckt fcpin in ga out gb
c0 in ga 100f
c1 in ga 0.5p
l1 in txin 0.2n
t1 txin ga txout gb z0=50 td=66.67p
l2 txout out 0.5n
c2 txout gb 0.3p
.ends

.subckt regpin in ga out gb
c0 in ga 100f
c1 in ga 1.0p
l1 in txin 3.0n
t1 txin ga txout gb z0=50 td=66.67p
l2 txout out 1.0n
c2 txout gb 0.3p
.ends

* Source Termination
R1 x gnd 40
R2 y gnd 40

* TX chip pin
xfcpin1 x gnd x1 gnd fcpin
xfcpin2 y gnd y1 gnd fcpin
* xregpin1 x gnd x1 gnd regpin
* xregpin2 y gnd y1 gnd regpin

* Line Card Transmission lines
*T1 x1 gnd xa gnd z0=50 td=1n
*T2 y1 gnd ya gnd z0=50 td=1n
w2 N=2 x1 y1 gnd xa ya gnd rlgcfile=lossy_board.rlc l=0.2


* Back Plane Model
xm1 xa ya gnd xb yb gb model

* Switch Card Transmission Lines
*T3 xb gb xc gb z0=50 td=1n
*T4 yb gb yc gb z0=50 td=1n
w3 N=2 xb yb gb xc yc gb rlgcfile=lossy_board.rlc l=0.2


* Receiver chip pin
xfcpin3 xc gb xd gb fcpin
xfcpin4 yc gb yd gb fcpin
* xregpin3 xc gb xd gb regpin
* xregpin4 yc gb yd gb regpin
```

```
* Receiver termination
R3 xd gb 40
R4 yd gb 40
```

9.2 Current Source Generation Perl Script (Equalization and Crosstalk Control)

```
#!/usr/local/bin/perl

print ("Making source.hsp.........");

open(SRC, ">source.hsp");


print SRC ".option post\n\n";

print SRC "*******************************\n";
print SRC "* Parameters                   *\n";
print SRC "*******************************\n";

print SRC ".param bitrate = 4.5G \n";
print SRC ".param per     = '1/bitrate'\n";
print SRC ".param tr      = 'per/4'\n";
print SRC ".param Is      = '13.8mA'\n";
print SRC ".param one     = '-Is'\n";
print SRC ".param zero    = 'Is'\n\n";


print SRC "*******************************\n";
print SRC "* Current Source              *\n";
print SRC "*******************************\n";

print SRC "i1 x gnd \n";

#print SRC "+ pwl   \n"; # 0    0  \n";

$i = 0;

@one_tap  = (-0.3857, -0.00577, -0.03427, -0.01168, -0.0122);
@zero_tap = ( 0.3857, 0.00577, 0.03427, 0.01168, 0.0122);

@current= (0, 0, 0, 0, 0);
@last_1 = (0, 0, 0, 0, 0);
@last_2 = (0, 0, 0, 0, 0);
@last_3 = (0, 0, 0, 0, 0);
@last_4 = (0, 0, 0, 0, 0);
@last_5 = (0, 0, 0, 0, 0);
@last_6 = (0, 0, 0, 0, 0);

$cur = 0.95*13.79;
$zero = $cur;
$one  = -$cur;
```

```
open(IN, "random_number");
$_ = <IN>;

chomp($_);
  if ($_ eq "0") {
    @current = @zero_tap;
    @last_1  = @zero_tap;
    @last_2  = @zero_tap;
    @last_3  = @zero_tap;
    @last_4  = @zero_tap;
    @last_5  = @zero_tap;
    $tmp      = ($zero - $cur*($last_1[0] + $last_2[1] + $last_3[2] +
$last_4[3] + $last_5[4] + $last_6[5]));
  }
  else {
    @current = @one_tap;
    @last_1  = @one_tap;
    @last_2  = @one_tap;
    @last_3  = @one_tap;
    @last_4  = @one_tap;
    @last_5  = @one_tap;
    $tmp      = ($one - $cur*($last_1[0] + $last_2[1] + $last_3[2] +
$last_4[3] + $last_5[4] + $last_6[5]));
  }
close(IN);
open(IN, "random_number");

print SRC "+ pwl   0    $tmp\mA  \n";

while(<IN>) {

  chomp ($_);

  if ($_ eq "0") {
    @current = @zero_tap;
  }
  else {
    @current = @one_tap;
  }

  print SRC "+'$i*per+tr'\t";

  if ($_ eq "0") {
    $tmp = $zero - $cur*($last_1[0] + $last_2[1] + $last_3[2] +
$last_4[3] + $last_5[4] + $last_6[5]);
  }
  else {
    $tmp = $one -  $cur*($last_1[0] + $last_2[1] + $last_3[2] +
$last_4[3] + $last_5[4] + $last_6[5]);
  }

  print SRC "$tmp\mA\t\t";

  $i = $i+1;

  print SRC "'$i*per'     ";
```

```
  print SRC "$tmp\mA\n";

  @last_6 = @last_5;
  @last_5 = @last_4;
  @last_4 = @last_3;
  @last_3 = @last_2;
  @last_2 = @last_1;
  @last_1 = @current;
}

#print SRC "+R \n\n\n\n";
close (IN);


print SRC "*******************************\n";
print SRC "* Current Source 2             *\n";
print SRC "*******************************\n";

print SRC "i2 y gnd \n";

#print SRC "+ pwl   \n"; # 0    0  \n";

$i = 0;

@one_tap  = (-0.3857, -0.00577, -0.03427, -0.01168, -0.0122);
@zero_tap = ( 0.3857, 0.00577, 0.03427, 0.01168, 0.0122);

@current= (0, 0, 0, 0, 0);
@last_1 = (0, 0, 0, 0, 0);
@last_2 = (0, 0, 0, 0, 0);
@last_3 = (0, 0, 0, 0, 0);
@last_4 = (0, 0, 0, 0, 0);
@last_5 = (0, 0, 0, 0, 0);
@last_6 = (0, 0, 0, 0, 0);

$cur = 0.95*13.79;
$zero = $cur;
$one  = -$cur;

$tmp = 0;
$tmp_last = 0;


open(IN, "random_number");
$_ = <IN>;

chomp($_);
  if ($_ eq "0") {
    @current = @zero_tap;
    @last_1  = @zero_tap;
    @last_2  = @zero_tap;
    @last_3  = @zero_tap;
    @last_4  = @zero_tap;
    @last_5  = @zero_tap;
    $tmp      = ($zero - $cur*($last_1[0] + $last_2[1] + $last_3[2] +
$last_4[3] + $last_5[4] + $last_6[5]));
```

```
  }
  else {
    @current = @one_tap;
    @last_1  = @one_tap;
    @last_2  = @one_tap;
    @last_3  = @one_tap;
    @last_4  = @one_tap;
    @last_5  = @one_tap;
    $tmp      = ($one - $cur*($last_1[0] + $last_2[1] + $last_3[2] +
$last_4[3] + $last_5[4] + $last_6[5]));
  }
close(IN);

print SRC "+ pwl   0    0  \n";

open(IN, "random_number");

$dIold=0;
$dIcur=0;
$cur1=0.13;
$cur2=-0.15;
$cur3=-0.1;
$cur4=0.02;

while(<IN>) {

  chomp ($_);

  if ($_ eq "0") {
    @current = @zero_tap;
  }
  else {
    @current = @one_tap;
  }


  if ($_ eq "0") {
    $dIold=$dIcur;
    $tmp_last = $tmp;
    $tmp      = ($zero - $cur*($last_1[0] + $last_2[1] + $last_3[2] +
$last_4[3] + $last_5[4] + $last_6[5]));
    $dIcur  = ($tmp - $tmp_last);
    $tmp_out = $cur1*$dIcur + $cur3*$dIold;

    print SRC "+'($i)*per+50ps'   ";
    print SRC "$tmp_out\mA\t";

    print SRC "'($i+1/2)*per'   ";
    print SRC "$tmp_out\mA\t";


    $tmp_out  = $cur2*$dIcur + $cur4*$dIold;

    print SRC "'($i+1/2)*per+50ps'  ";
    print SRC "$tmp_out\mA\t";
    print SRC "'($i + 1)*per'  ";
    print SRC "$tmp_out\mA\n";
```

```
  }
  else {
     $dIold=$dIcur;
    $tmp_last = $tmp;
    $tmp      = ($one - $cur*($last_1[0] + $last_2[1] + $last_3[2] +
$last_4[3] + $last_5[4] + $last_6[5]));
    $dIcur  = ($tmp - $tmp_last);
    $tmp_out = $cur1*$dIcur + $cur3*$dIold;

    print SRC "+'($i)*per+50ps'   ";
    print SRC "$tmp_out\mA\t";

    print SRC "'($i+1/2)*per'   ";
    print SRC "$tmp_out\mA\t";


    $tmp_out  = $cur2*$dIcur + $cur4*$dIold;

    print SRC "'($i+1/2)*per+50ps'  ";
    print SRC "$tmp_out\mA\t";
    print SRC "'($i + 1)*per'  ";
    print SRC "$tmp_out\mA\n";
   }


  $i = $i+1;

  @last_6 = @last_5;
  @last_5 = @last_4;
  @last_4 = @last_3;
  @last_3 = @last_2;
  @last_2 = @last_1;
  @last_1 = @current;
}

#print SRC "+R \n";

close (SRC);
close (IN);


print ("Completed.\n");
```