A 12.5GHz High Speed Data Link

Glenn Kewley

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2018-63
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2018/EECS-2018-63.html

May 11, 2018

290C Final Project, 12.5Gbit/sec Serial Link

M. Bowers, X. Chen, G. Kewley, A. Khidre

1. Design Requirements

For this project, a high-speed serial link is designed to work over 5 different channels in a backplane environment.

Design Parameter	Min Spec	Designed For/ Met
Data Rate	12Gbit/sec	12.5Gbit/sec
Bit Error Rate (BER)	10^{-15}	10^{-20}
Supply Noise	+/-35mV	+/- 35mV
Termination Mismatch	2%	2%

2. Channel Response

We first look at the 5 channels over which the link will need to operate: 30" top level trace, 30" bottom trace, 20" middle trace, 10" middle trace, and 1.5" top trace. From a design perspective, the two most limiting channels are the 30" bottom due to its long delay, high signal attenuation, precursor, and reflections, and the 1.5" Top trace which is the least attenuated and has the shortest time delay with reflections that create a long tail of inter symbol interference(ISI).

In the last plot of *Figure 1* all 5 channel responses are overlaid for comparison. It can be seen that every channel has a precursor and several post cursor ISI symbols. The precursor indicates that there must be a transmit pre-emphasis to cancel the pre-cursor ISI, as this cannot be corrected for at the receive side. While the process variations can lead to termination resistance variation of up to 2%, this caused only 1-200uV differences in the channel responses. These differences can be easily accounted for in the channel equalization.

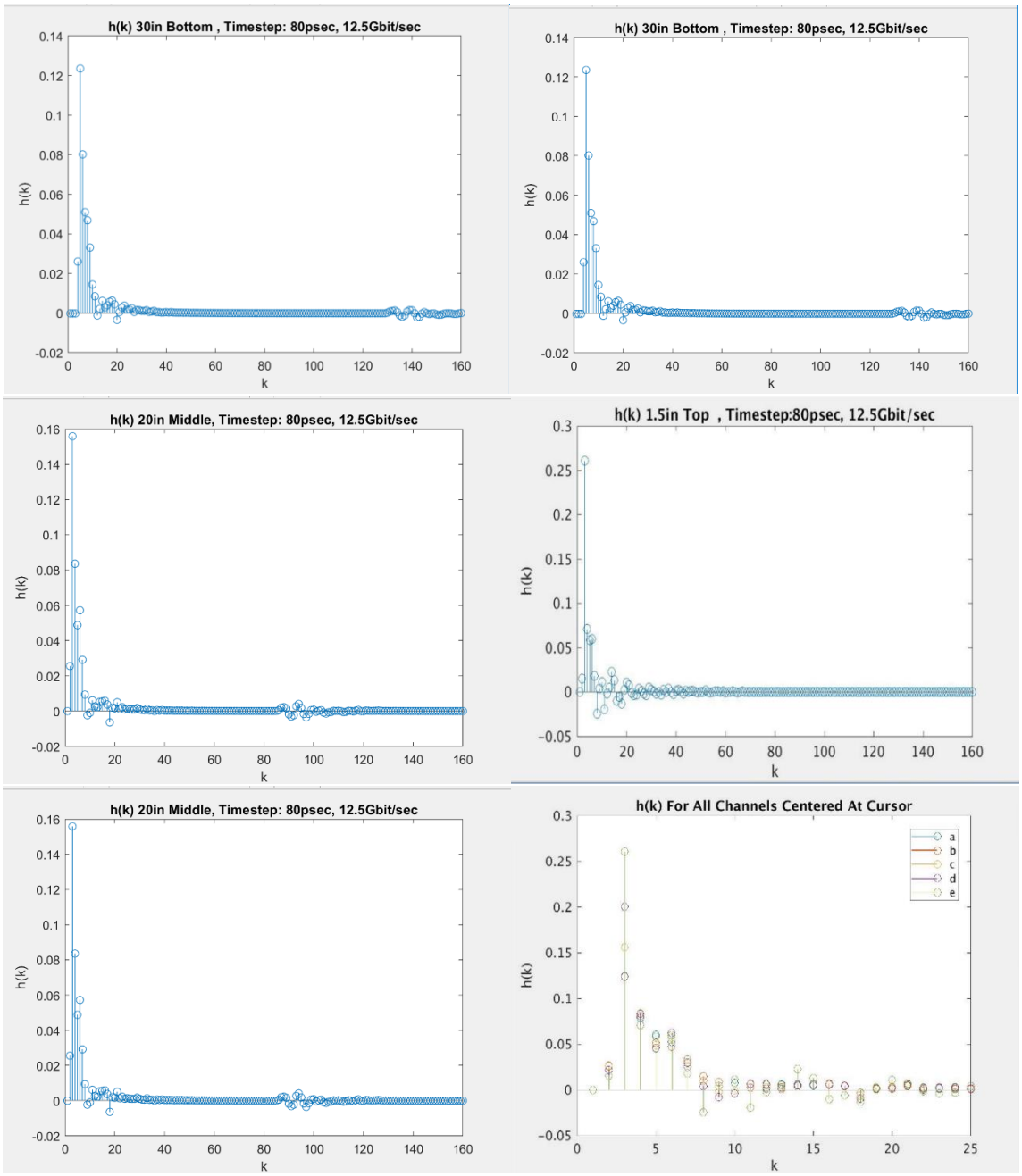


Figure 1

3. Equalization Architecture

Based on the channel response, and specifically the presence of a pre-cursor ISI on every channel, a transmit FIR is necessary and is realized with a fully differential current steering pair. On the receive side, a 6-tap decision feedback equalizer (DFE) is implemented also fully differential using similar architecture to the transmit FIR.

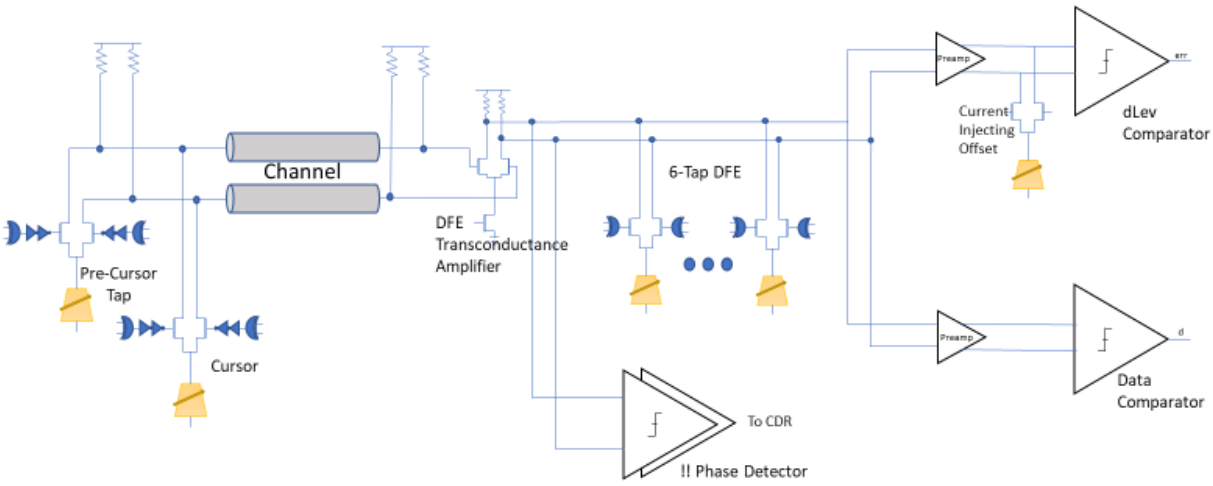


Figure 2

To realize a 1V differential transmission signal, and with the 32nm technology provided, an input and output common mode of 650mV is used for all differential lines, and a supply voltage of 900mV. All differential lines keep the transistors in saturation with input common mode of 900mV down to 400mV.

4. Equalization Sub-Blocks

a. Binary Weighted Current DAC

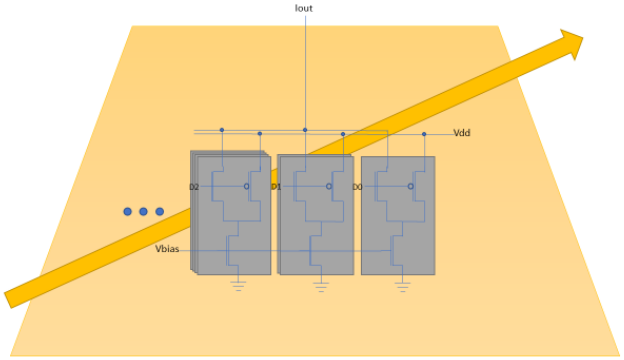


Figure 3

The Tx FIR, Rx DFE, and dLev Loop all use a current DAC. A multipurpose current DAC was created in cadence to meet the specs for all systems requiring its implementation. With an input parameter of only the maximum output current, the DAC can be placed in any system without needing re-design. There are two-versions of this DAC: an 8bit and a 5bit.

b. Current Steering Pairs

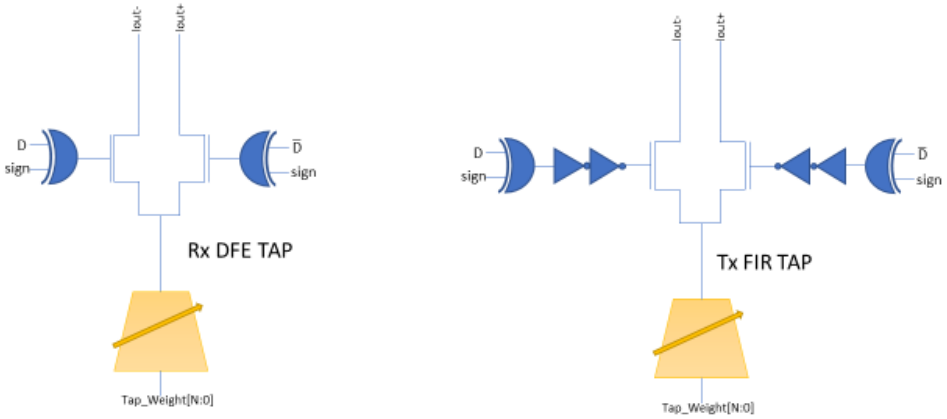


Figure 4

Two versions of the current steering pair were needed for the equalizer. The Tx-FIR version needs to drive 10mA to produce a +/- 1V swing across the 50Ω termination resistors. This increased size in transistors required the addition of fanout inverters to drive the switching elements at the data speed from the minimum sized digital logic. Incorporating the current DAC, the top-level block was created that can be adjusted with a single input parameter of the maximum current needed. All taps used a 5-bit DAC with the exception of the main cursor transmit tap which uses an 8bit. The reason for this is to keep the Δ_{ω}, or change in voltage per update, of the adaptation loops consistent to within 1-3mV.

c. Comparator Subsystem

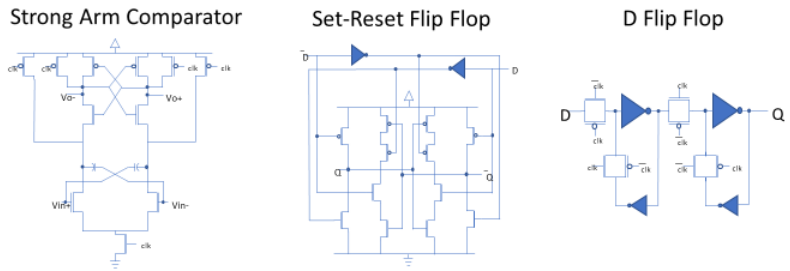


Figure 5

A strongarm latch was used in both the main data comparator and the dLev comparator. Both were followed by a S-R Flip Flop to maintain the data decision for the second half of the clock cycle when the strong arm goes into regeneration. The dLev comparator required an extra gain stage after the S-R Flip Flop which was implemented using a transmission gate D-Flip Flop. This was necessary because the dLev comparator is sampling lines that are constantly near-equivalent in voltage. Without this extra digital gain stage, the probability of getting an unresolved bit decision was too high for reliable operation.

d. DFE Transconductance Amplifier and Closing the First Tap

From the termination resistors on the receive-side of the link, a differential transconductance amplifier converts the signal into a differential current. On the same line, the DFE taps provide offset currents to cancel the ISI. The currents are summed and passed through a resistor to create a differential voltage for comparison and data decision. This line also drives the dLev, comparator and bang-bang phase detectors.

With several sub-systems on this line, there is a large amount of load capacitance in addition to the self-loading capacitance of the summing amplifier and DFE taps. For this reason, it is critical to ensure that the gm of the summing amplifier be high enough to ensure that the data decision can propagate through the digital logic to the first DFE-Tap, apply the ISI cancelling current and have the voltage on the line settle all before the next clock edge.

$$gm = \frac{(A_v/\tau_{sum}) \cdot (C_L + N_{taps} \cdot C_{parasitic})}{1 - \frac{A_v/\tau_{sum}}{\omega_{t,sum}} \cdot \left(1 + \frac{2 \cdot MDFE \cdot V_{cursor}}{V^*_{tap}} \cdot \frac{\omega_{t,sum}}{\omega_{t,tap}}\right)}$$

- $V^*_{tap} = 525.2mV$
- $f_{t,tap} = 324.2GHz$
- $V^*_{sum} = 282mV$
- $f_{t,sum} = 322GHz$
- $T_{bit} = 80psec$
- $T_{dig} = 25psec$
- $\tau_{sum} = \frac{T_{bit} - T_{dig}}{4_\tau} = 13.75psec$
- $MDFE = \frac{h_1 h_2 \cdots h_{N_{taps}}}{h_0} = 1.9$
- $C_{parasitic} = 500\,{}^{aF}/_{Tap}$
- $C_L = 12.22fF$
- $A_V = 1.2({}^{V}/_{V})$

Figure 6

The data in *figure 6* was gathered from the results of the final design. The initial estimates for some of these parameters were higher and the final design utilizes $gm_{simulated} = 4.7({}^{mA}/_{V})$ and this was used for the simulations. In a subsequent revision of the design, a $gm_{optimized} = 1.7\,{}^{mA}/_{V}$ could be utilized to reduce power.

e. Data Adaptation Dual Loop

The link utilizes a dual-loop adaptation architecture. A loop for finding the average data level, and a loop for adapting the Tx-FIR and Rx-DFE tap weights. While both require a bit decision from the main data comparator, the tap-loop requires the error signal produced by the dLev loop. The dLev error signal, in turn, changes as the tap weights approach their ideal value.

- $\omega_{n+1}^{k} = \omega_{n}^{k} + \Delta_{\omega} \cdot \mathrm{sgn}\left(e_n \cdot I(d_{n-k} == 1)\right)$ General Form
  - *k = tap index, n = sample index*

- $\omega_{n+1}^{-1} = \omega_{n}^{-1} + \Delta_{\omega} \cdot \mathrm{sgn}\left(e_n \cdot I(d_{n+1} == 1)\right)$ Pre-Cursor
- $\omega_{n+1}^{0} = \omega_{n}^{0} + \Delta_{\omega} \cdot \mathrm{sgn}\left(e_n \cdot I(d_{n} == 1)\right)$ Cursor
- $\omega_{n+1}^{1} = \omega_{n}^{1} + \Delta_{\omega} \cdot \mathrm{sgn}\left(e_n \cdot I(d_{n-1} == 1)\right)$ DFE Tap 1
- $\omega_{n+1}^{4} = \omega_{n}^{4} + \Delta_{\omega} \cdot \mathrm{sgn}\left(e_n \cdot I(d_{n-2} == 1)\right)$ DFE Tap 2

Figure 7

Figure 7 shows the general form of the tap update equation. Notice that the tap is only updated when the current data bit is a logic "1". While 6 DFE taps were used, only two are included for simplicity. With $k = 0$ being the cursor weight, all $k > 0$ represent the DFE taps. Each DFE- tap is updated based on the current error signal and a timed delayed data bit, and the current data bit.

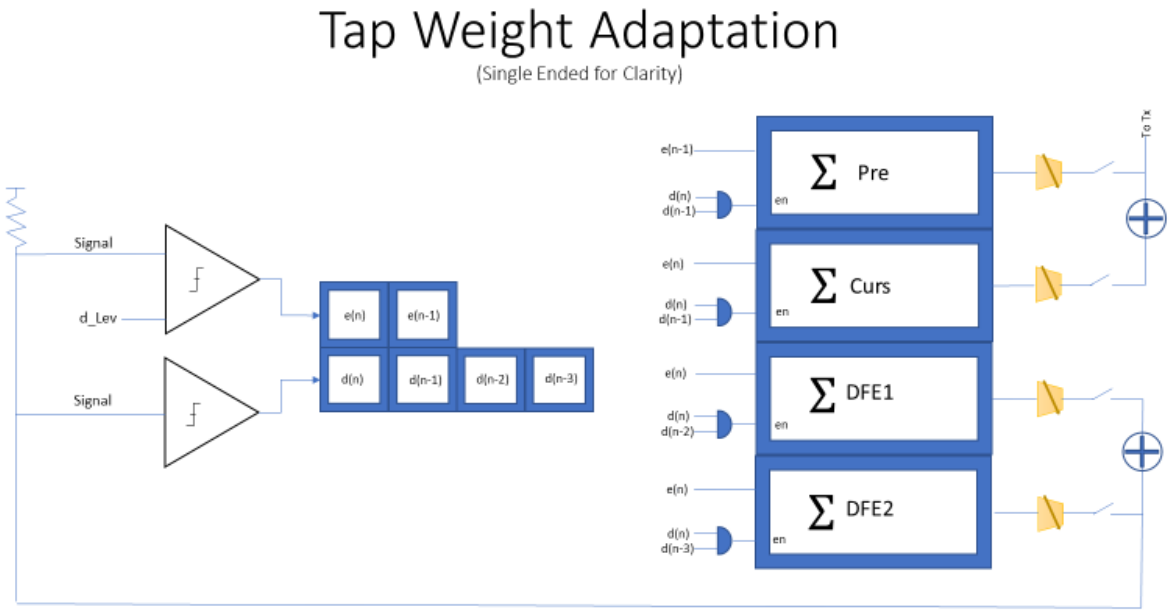


Figure 8

The dLev loop is similar in design, and uses a current steering pair to drive the preamplifier into the dLev comparator until they are equal in voltage. The loop then dithers about the optimal value once it is reached.

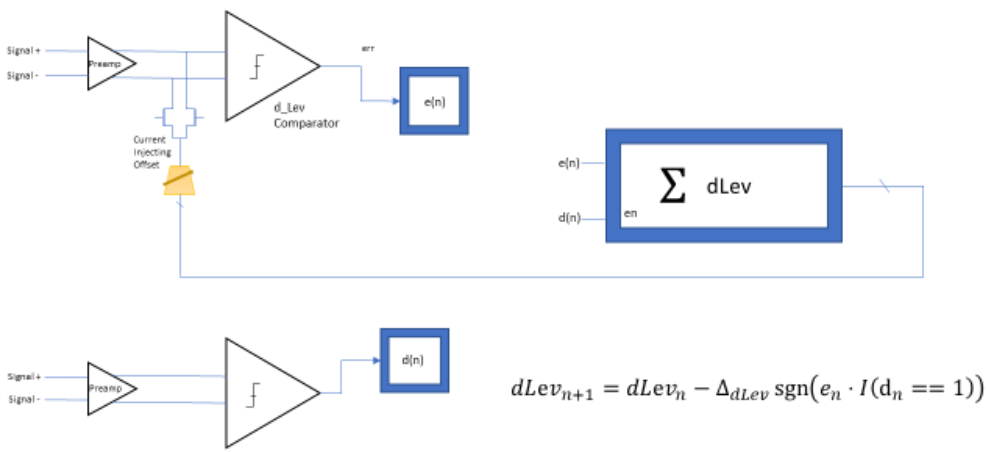


Figure 9

5. Simulation Environments

Two simulation environments were used: MATLAB Simulink and Cadence AMS. Simulink's simulation speed was necessary for functional verification of the link before full analog simulations were performed. In Cadence the link is designed entirely in a 32nm CMOS technology, with the exception of the adaptation digital blocks that are implemented in Verilog. While neither environment produces noteworthy schematic views, the two top level designs are displayed in *figure 10* to show the similarities.

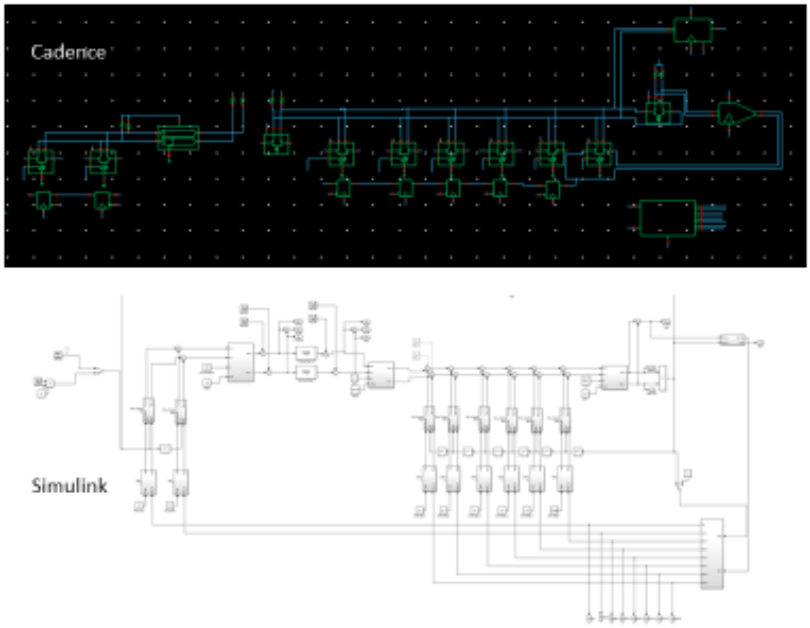


Figure 10

6. Behavioral Modeling

This section reviews the behavioral modeling of each component inside the CDR (Clock Data Recovery) unit. The top-level diagram for CDR is Figure 11. The reference clock is the clock generated from crystal and an output clock is created using the PLL loop. The output clock is 12.5 GHz since the target data rate is 12.5Gbit/s and Single Data Rate is employed in this project. The DLL, lower loop in figure 11, is implemented to correct the phase of the clock feeding into the Phase Interpolator so that the clock rising edge lines up with a stable data input and clock falling edge lines up with data falling edge.

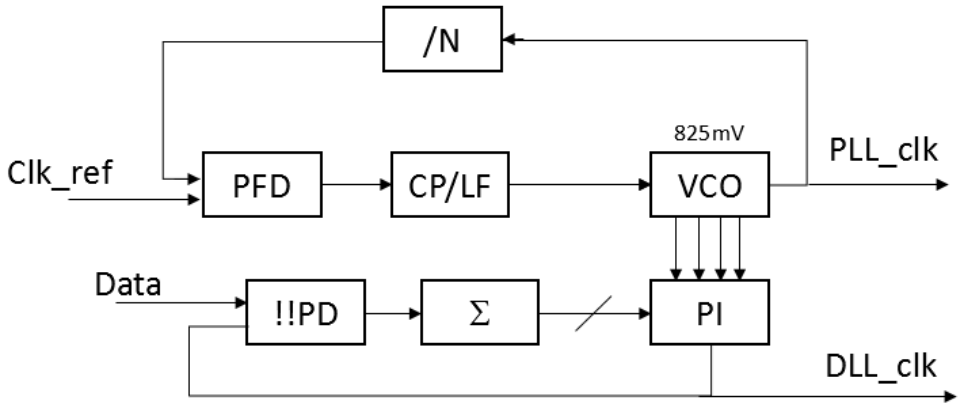


Figure 11: Top Level CDR Diagram

a. Phase Frequency Detector

The Phase Frequency Detector is supposed to not only detect phase differences, but also frequency differences. The purpose of phase-frequency detection instead of phase detection is to mitigate a harmonic wave reference frequency, then the error signal will be at a totally different frequency and result in Harmonic Locking. Therefore, the frequency and phase both need to be corrected. The following diagrams show the circuit for PFD.

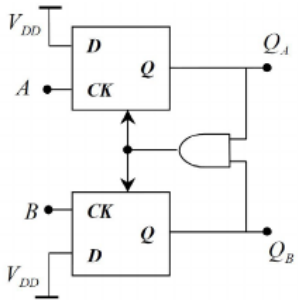


Figure 12. Phase Frequency Detector

b. Charge Pump

The purpose of Charge Pump is to send voltage (charge) to Voltage Controlled Oscillator. When the Charge Pump receives an “up” signal from PFD, it will pump charges to Loop Filter and VCO; when the Charge Pump receives a “down” signal from PFD, it will drain charges from Loop Filter so that the voltage is roughly constant since we only connect the positive out to VCO.

The following diagrams show the circuit for Charge Pump.

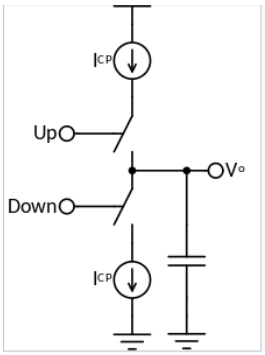


Figure 13. Charge Pump

c. Loop Filter

The Loop Filter Diagram is shown below and C1=9pF, C2=1pF, R=1.388KΩ.

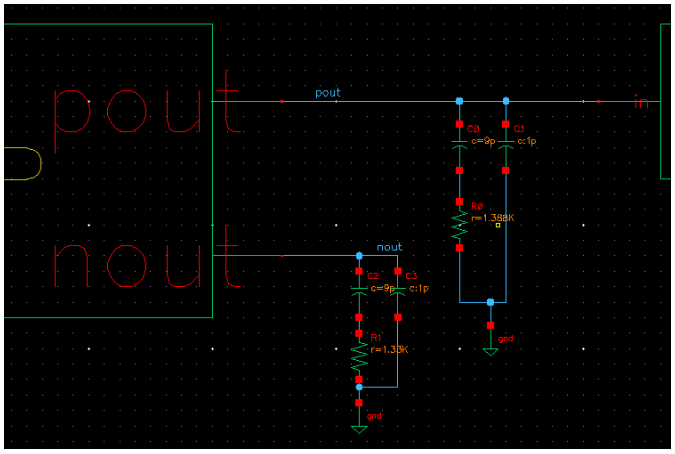


Figure 14 Loop Filter

d. Voltage Controlled Oscillator

The VCO is uses an input voltage to create an output signal with certain frequency/voltage (Hz/V) gain. It is realized by ring oscillators in transistor level. Since the behavioral modeling is only modeling the unit’s behavior, the HDL code is very straightforward and expressed the amplifying function in Hz/V and converts from an initial sine wave to square wave. As will be discussed later, the VCO will feed 4 clocks with 90° phase difference to Phase Interpolator, the VCO has 4 outputs but will only feedback the 0° phase clock back to divider and then back to PFD.

e. Divider

The Divider is dividing the VCO output signal's frequency by a certain number and feedback the divided clock to PFD and correct phase and frequency there. The project employs a divider-by-25 since our reference clock is 500 MHz and the target clock frequency is 12.5 GHz.

f. Bang Bang Phase Detector

The DLL loop starts with Bang Bang Phase Detector. The Bang Bang Phase Detector is supposed to detect the phase difference between data and clock. That clock is feedback from Phase Interpolator. And its outputs are T (Transition) and E (Early) signals to feed to accumulator. Whenever T is high, it tells the accumulator to update; and E high means early, E low means late. The accumulator needs to count up or count down accordingly.

g. Accumulator

The concept of Accumulator is straightforward. It is a 9 bit up-down counter based on the T and E signal from Bang Bang Phase Detector. 8 bits are for data and 1 bit for sign of the counter.

h. Phase Interpolator

The Phase Interpolator behavioral model is the most complicated one to understand and debug in CDR. It should take in the 8-bit data from Accumulator and convert that data into phase delay within 90°. As previously stated, the VCO provides 4 clocks with different phases to Phase Interpolator. They are 0°, 90°, 180°, 270°. The Phase Interpolator is supposed to choose 1 region out of 4, which are 0°-90°, 90°-180°, 180°-270°, 270°-0°. The PI needs to have its own mux select logic to rotate from region to region if the desired phase delay is not in the current region. 8-bit data is 256 in decimal so the phase delay resolution is 90°/256 = 0.351°.

Putting everything together, the recovered clock and data is shown in the diagram below. "Data" is coming from "rand_bit_stream" in Cadence "ahdlLib" library. The clock rising edge lines up with a stable data input and clock falling edge lines up with data falling edge. That guarantees the correct frequency and phase of receiver side clock to sample data and helps reduce BER.

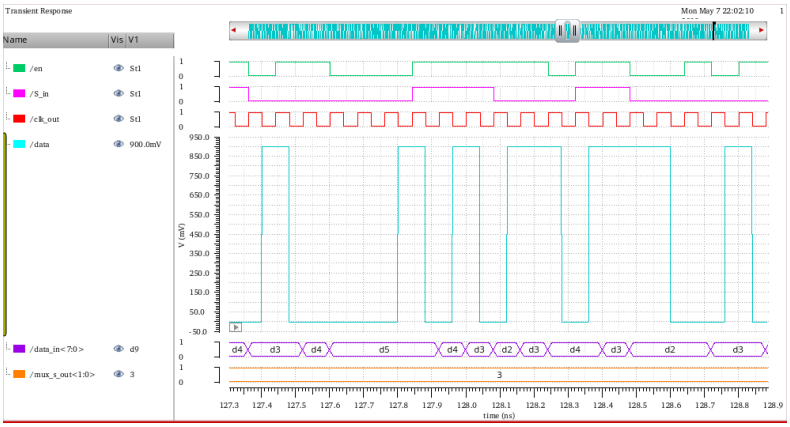


Figure 15. Clock recovered to rise and fall at the correct moment

Clock Data Recovery

The Clock Data Recovery (CDR) is realized using a dual loop architecture (PLL & DLL). The phase locked loop (PLL) consists of a VCO, divide by N, phase frequency detector, charge pump, and a loop filter. The DLL loop is comprised of a phase interpolator, phase detector, accumulator, and the associated control logic. All blocks have been implemented at the transistor level.

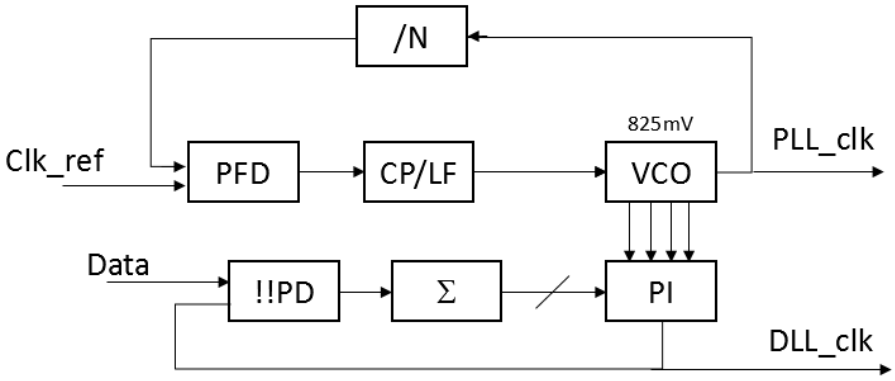


Figure 16 Clock Data Recovery block diagram

7. PLL Transistor Level

The VCO is a phase forward ring oscillator with level shifters to bring the output back to full scale voltage swing at 50% duty cycle. The generated oscillator frequency is a function of device sizing and power supply voltage. The inverters power supply is from the charge pump output Vcontrol. The unloaded VCO output frequency is 12.5GHz when the control voltage is 823mA.

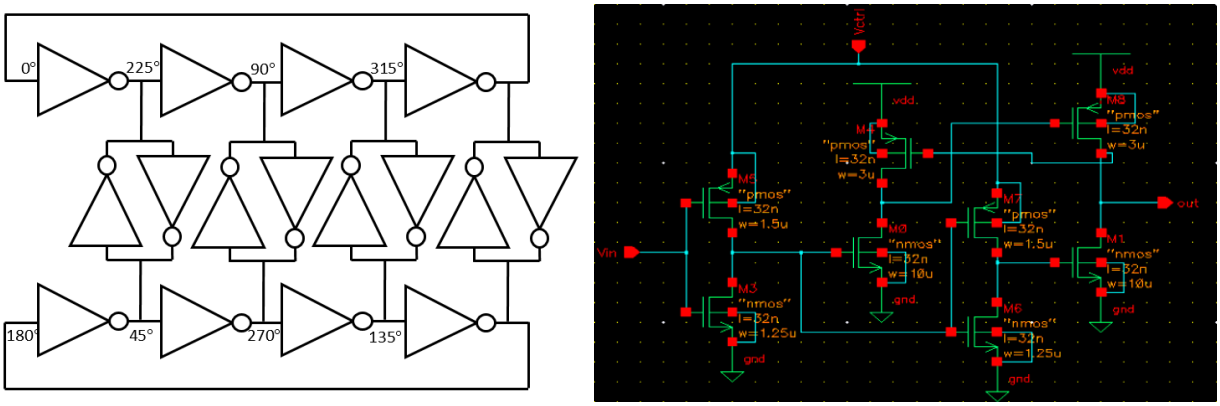


Figure 17 Phase Forward Ring Oscillator & Level Shifter

The frequency gain (K_{VCO}) is 34,906MHz/V. This value is used for setting the RC of the loop filter.

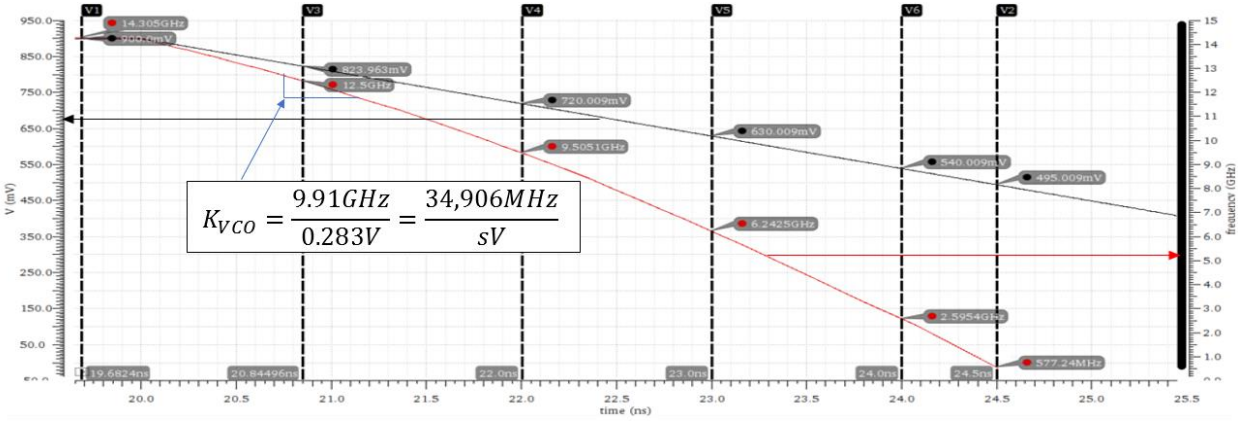


Figure 18 K_{VCO}

Jitter was measured after running the VCO for 500ns and plotting the eye diagram as shown below. There is a 4.5ps of jitter or 5.6%.

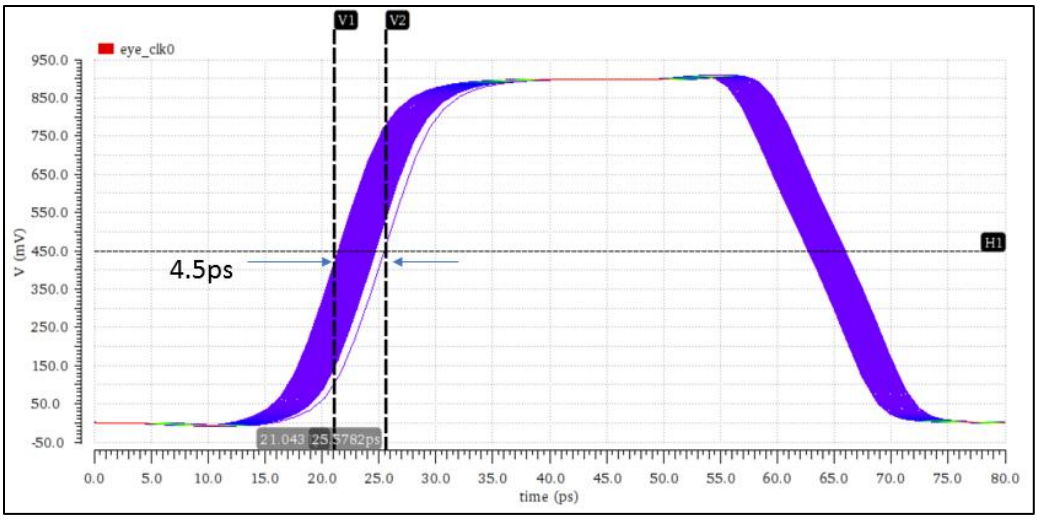


Figure 19 Unloaded VCO Jitter

With a target data rate of 12.5Gsps and a design criterion of a using a reference clock less than 500MHz we must divide the data rate clock down (VCO output) for comparison to this reference clock. Dividing down by multiples of 2x simplifies the hardware implementation. The divide by N is a constructed with a series of 5 D-FF's for a divide factor of 2^5. This will result in a clock period of 2.56ns (390.63MHz) which is compared to a reference clock in the phase detector.

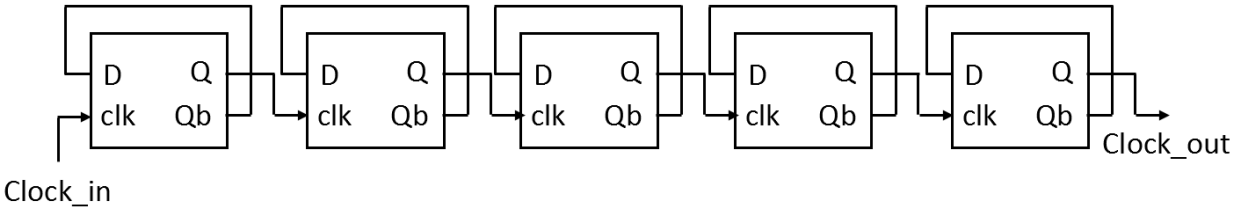


Figure20. Divide by N

The DFF's were constructed by transistor level NAND gates as shown below.

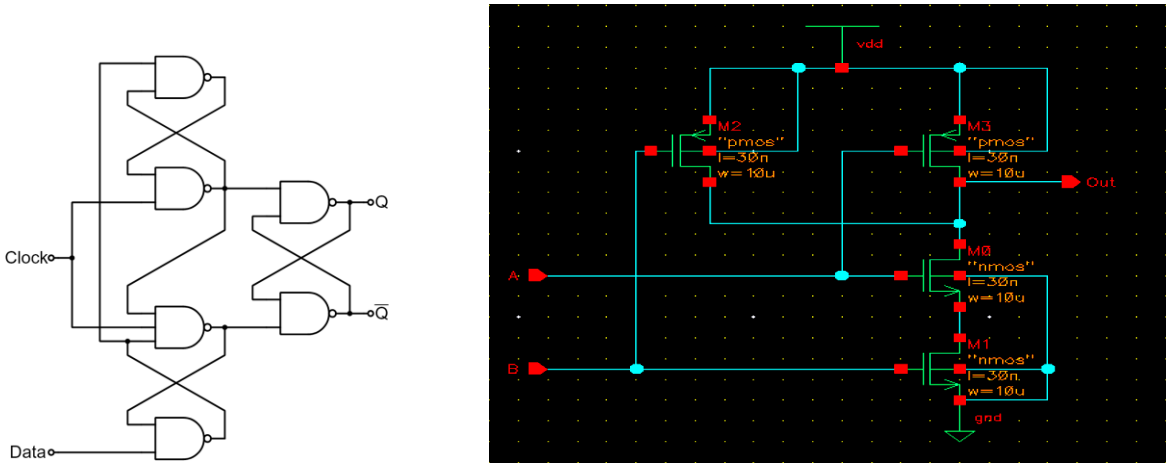


Figure21. DFF and Transistor level NAND

The divided data rate clock and our reference clock are inputs to the phase frequency divider. This functional block compares the rising edge of the ref clock to the rising edge off the feedback clock (divided data rate clock from VCO). The output is a pulse from either up_bar or down and the pulse width is proportional to the delay between the rising edges of RefClk and FbClk. Simulations of this block are shown in figures 23.

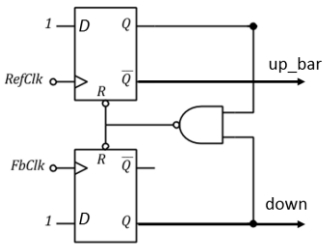
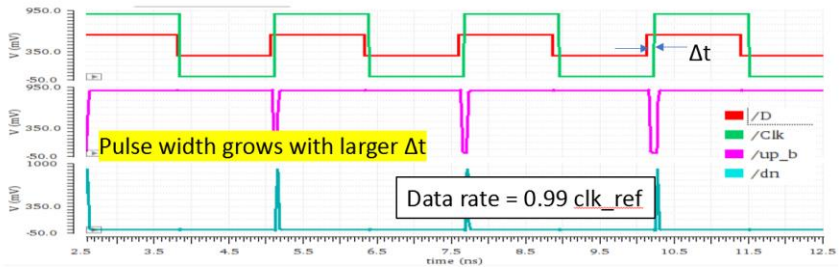
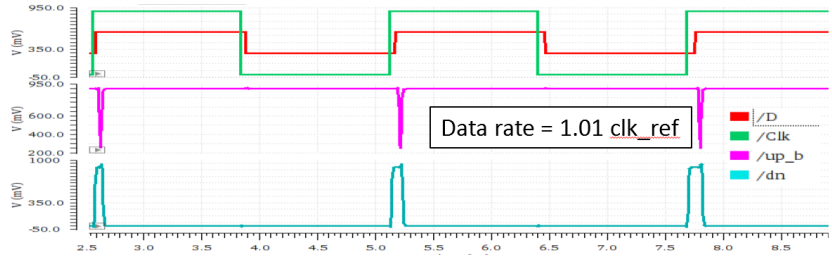


Figure 22. Phase Frequency Detector



a.



b.

Figure 23. Phase Frequency Detector Simulations

If the data rate clock is too slow the PFD will output pulses from the up-bar output. Conversely, if the data rate clock is too fast the PFD will output pulses from the down output. The pulse width will continue to increase with increasing Δt. These signals are used to drive the charge pump.

Charge Pump

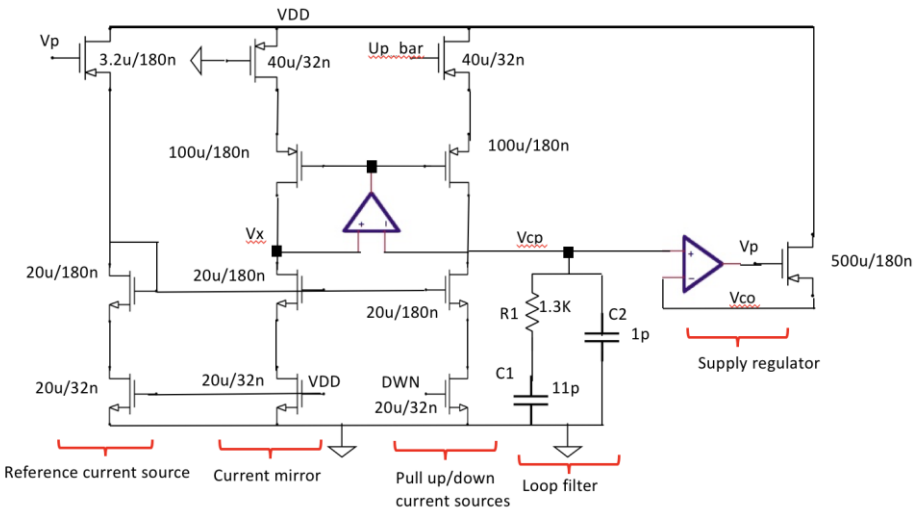


Figure 24

The architecture used for PLL charge pump is shown above. The pull up/down current sources are 1:1 mirrored from reference current source. The reference current source is however, mirrored from PMOS supplying current to the VCO. This is self-biasing technique I used to mitigate variation due to process and temperature. The charge pump Op-Amp is used to balance the currents between both branches.

Charge pump Op-amp

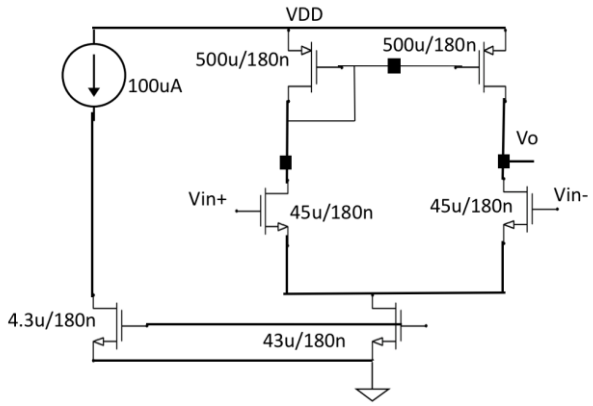


Figure 25

The charge pump Op-amp is shown above. It is single ended single stage differential amplifier. The tail DC current is 1.02mA, whereas reference current source is 100uA leading total I_{dc} = 1.12mA. The input differential pair are biased with V* = 0.12 V

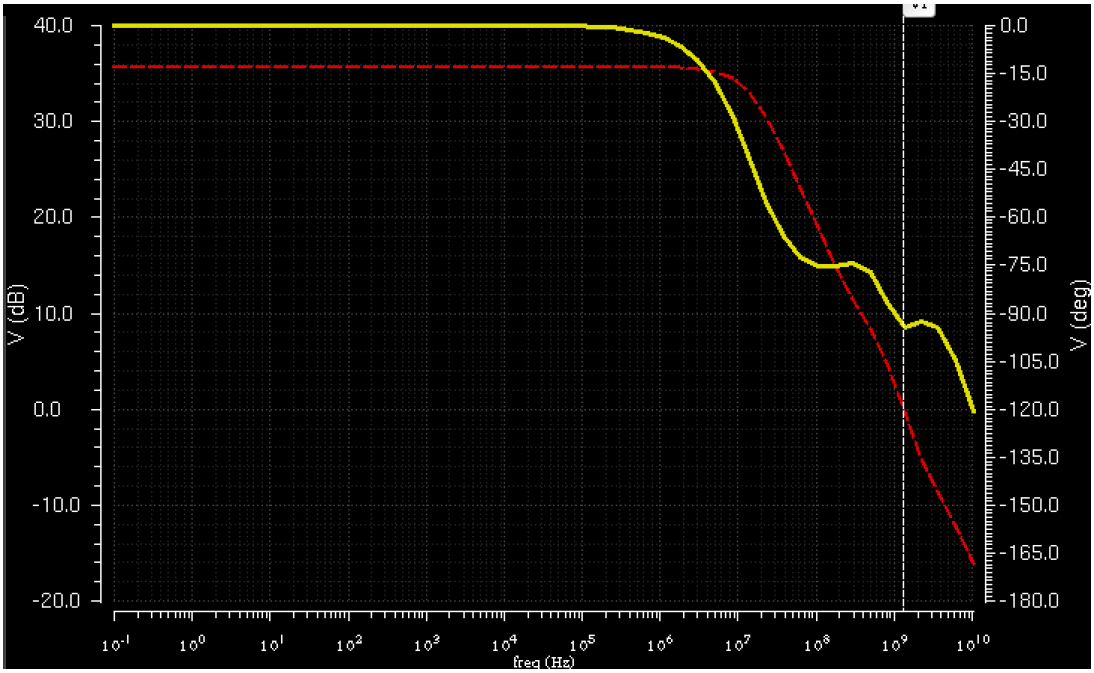


Figure 26

The performance of the op-amp is shown above. It has gain =36 dB with GBW = 1GHz. The Phase margin is > 90 degree which is enough to ensure stability inside charge pump feedback loops.

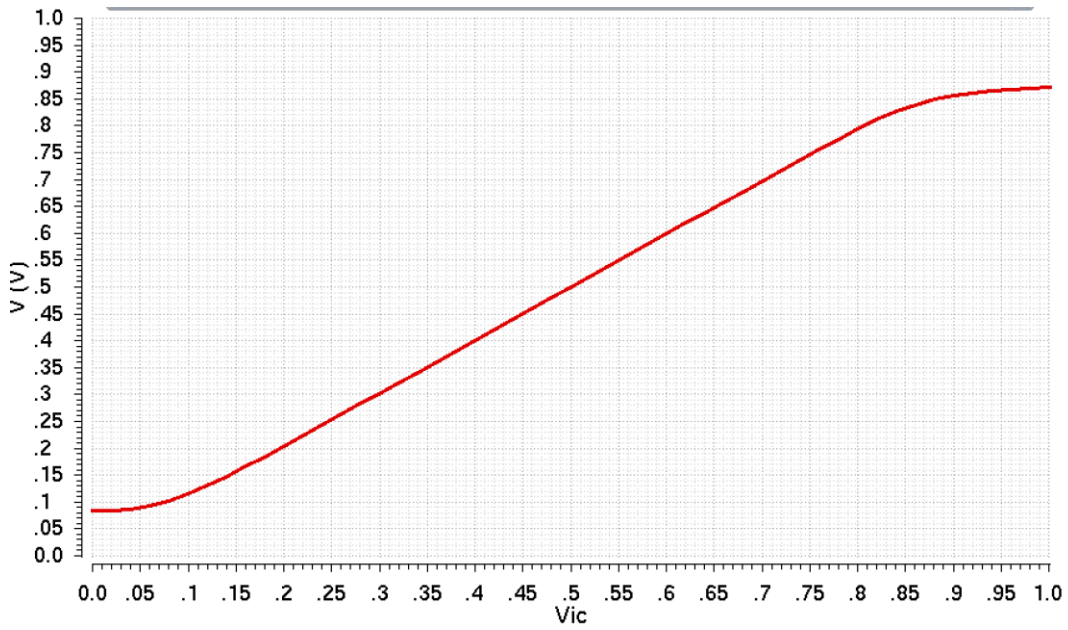


Figure 27

The input common mode range is shown above. As could be observed, the linear region ranges from 0.15 - 0.85 V which meets requirement for VCO control voltage to be 825mV for output clock 12.5 GHz at steady state (lock condition).

Loop Filter

The shown loop filter leads to 2nd order PLL. It components value could be estimated given:

1. $I_{cp} = 0.15$ mA
2. $PM = 60° \rightarrow \zeta \sim 0.6$
3. *Bandwidth = 39MHz*
4. $K_{vco} = 34906.65$

Therefore,

$$R_1 = 2\zeta \sqrt{\frac{N}{Icp.Kvco.C_1}}$$

$$C_1 = \frac{Icp.Kvco}{N\ x\ w_n^2}$$

$$C_2 \approx \frac{C_1}{10}$$

Supply regulator Op-amp

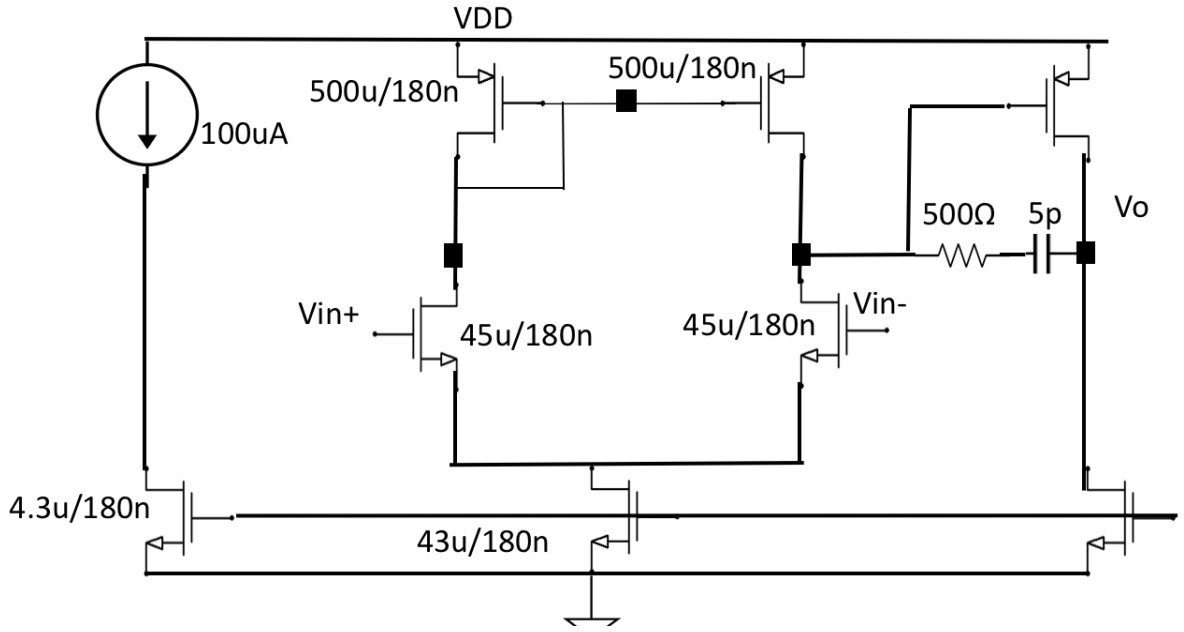


Figure 28

The supply regulator op-amp is similar to the charge pump op-amp, but a second stage is added with series RC feedback to improve phase margin (loop bandwidth stability)

Figure 29

The gain of this amplifier is 77 dB with PM =100 up to 100MHz which is enough for charge pump application, since loop filter voltage (input to this op-amp) is not supposed to change with frequency above 100MHz

PLL transient simulation Results

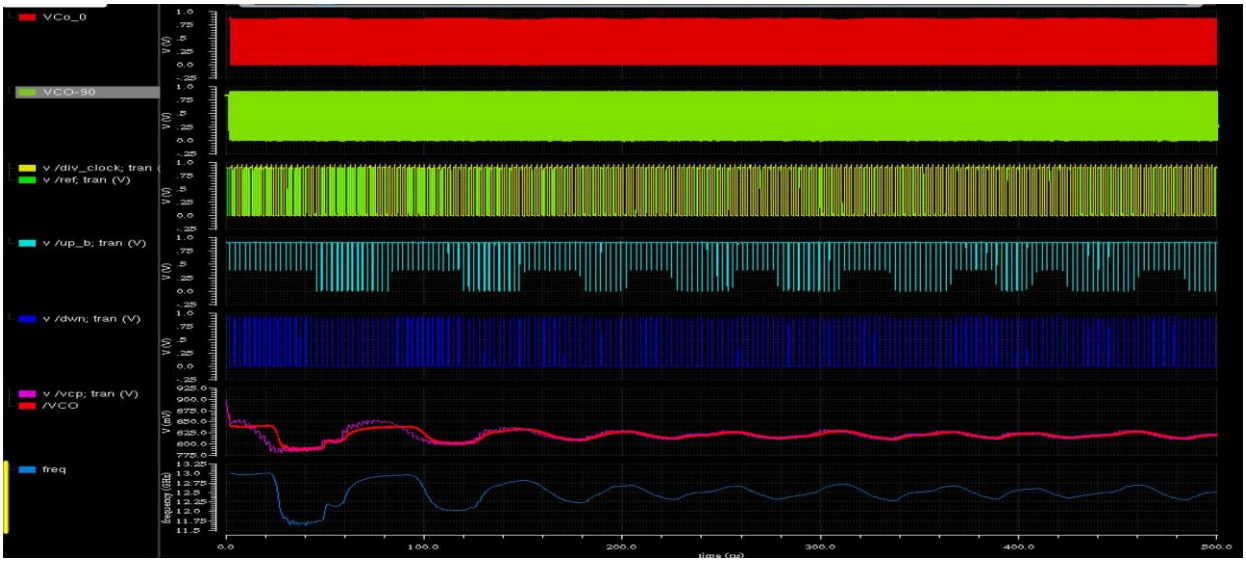


Figure 30

The transient simulation results for the whole PLL, with all blocks implemented at transistor level as presented in previous section, is shown in the above figure. The Loop filter is initialized at 0.73 V for quick convergence of the loop. The top curve in red is the VCO clock feedback to the PFD through the divider, whereas the curve below in green is the 90 degrees phase shifted VCO output clock unloaded. The following curves are for reference clock along with divided clock from VCO whereas outputs from PFD are plotted next. The loop filter voltage along with regulated VCO voltage are plotted in red and purple below PFD outputs. The bottom curve shows instantaneous frequency of VCO clock which as could be see, settles to 12.5 GHz.

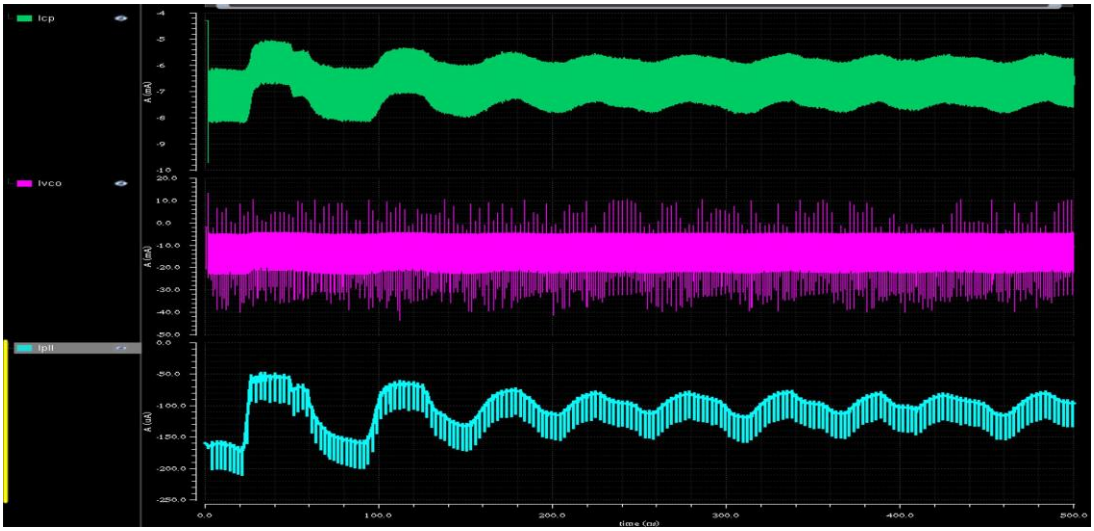


Figure 31

The above figure shows currents for charge pump (dark green), VCO (light green) , and whole PLL (purple). The charge pump is 150uA as per design, whereas VCO is drawing ~ 8mA maximum. The whole PLL is drawing close -15mA average

DLL

The Delay Locked Loop consists of a Phase Interpolator (PI), Phase Detector, and an Accumulator that implements control logic for the PI. The PLL loop sets the frequency of the data rate clock and the DLL corrects for phase error. The Phase Interpolator is constructed with two 8-bit current DAC's differentially tied to 50-ohm termination resistors. The differential summing node signal is converted to single ended with a differential pair.

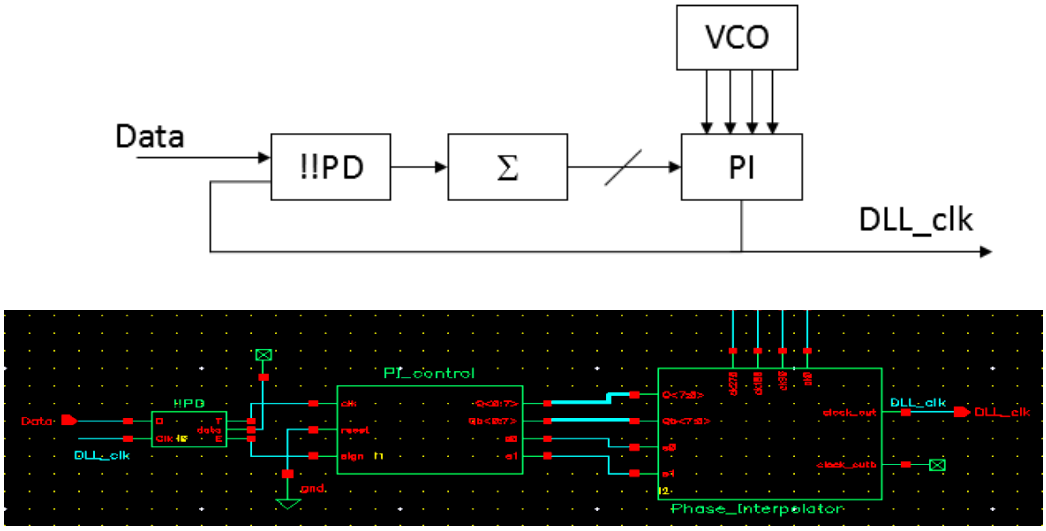


Figure 33 PLL block diagram and schematic

The Bang-Bang Phase Detector (Alexander or !!PD) compares the links received data to the negative edge of the clock signal from the VCO and the present data bit with the previous data bit. The output of the !!PD is three signals (T, Data, E). The Transition "T" signal indicates there has been a transition in the data stream while the Early "E" signal indicates the position of the rising edge VCO clock relative to the data stream rising edge.

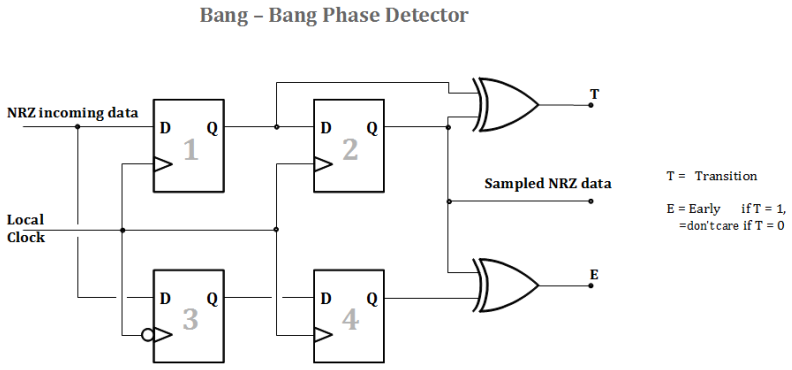


Figure 34 !!PD

Simulation results of the !!PD are shown below (figure 35). The transition output asserts when there has been a transition in the data stream. The "E" signal is 1 when the clock input occurs before the data transition and 0 when the clock signal occurs after the data transition.

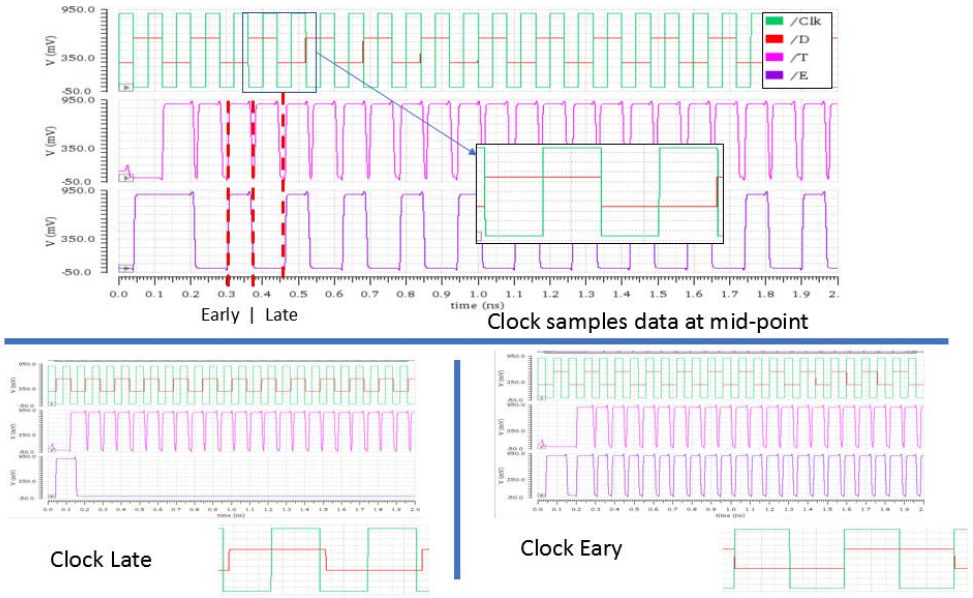
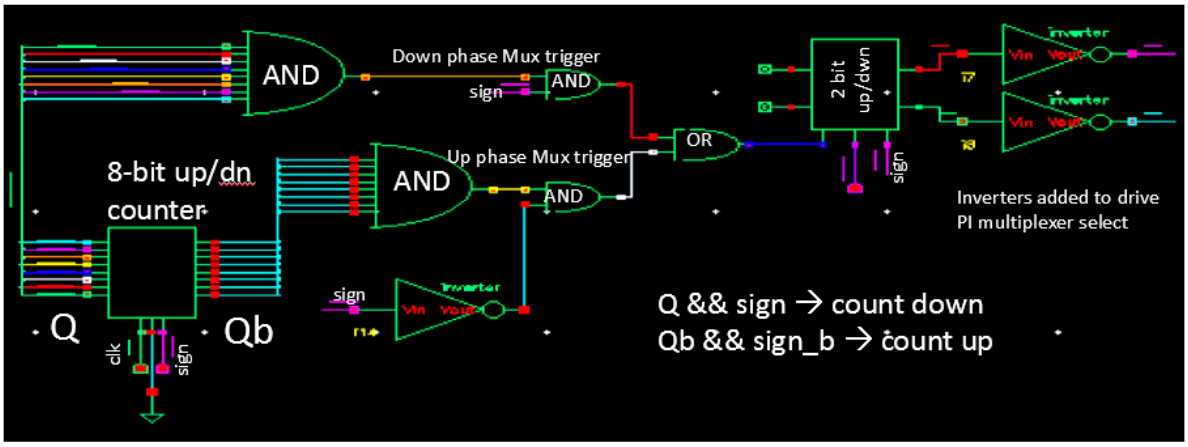


Figure 35 !!PD Steady State output, clock late, clock early simulations

The !!PD output drives the accumulator. The transition signal triggers the accumulator to count-up or down while the Early signal is used as the accumulator sign bit with a 1 signaling count-down (increase phase) and a 0 signaling count-up (decrease phase).



CLOCK: triggered from !!PD transition "T"
SIGN : !!PD early output "E"
Q & Qb drive Phase interpolator current DACs

Figure 36 Accumulator and PI control logic

The accumulator is comprised of an 8-bit up/down counter, two 8-bit AND, one OR, three inverters, and a 2-bit up/down counter. The 8-bit up/down counter is shown below. It is designed from transistor level JK-FF's and logic gates (AND, OR)

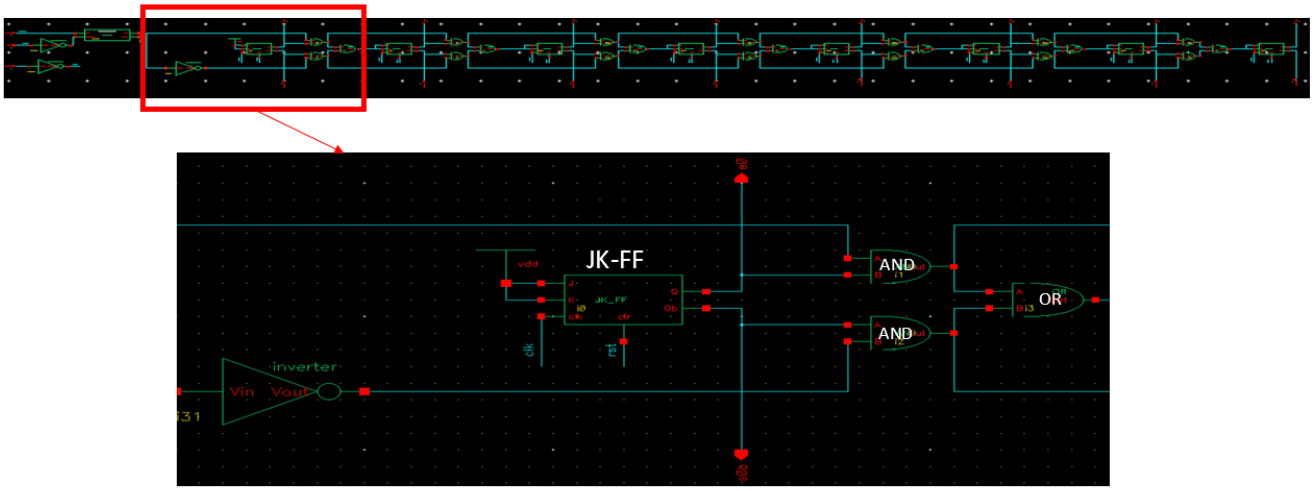
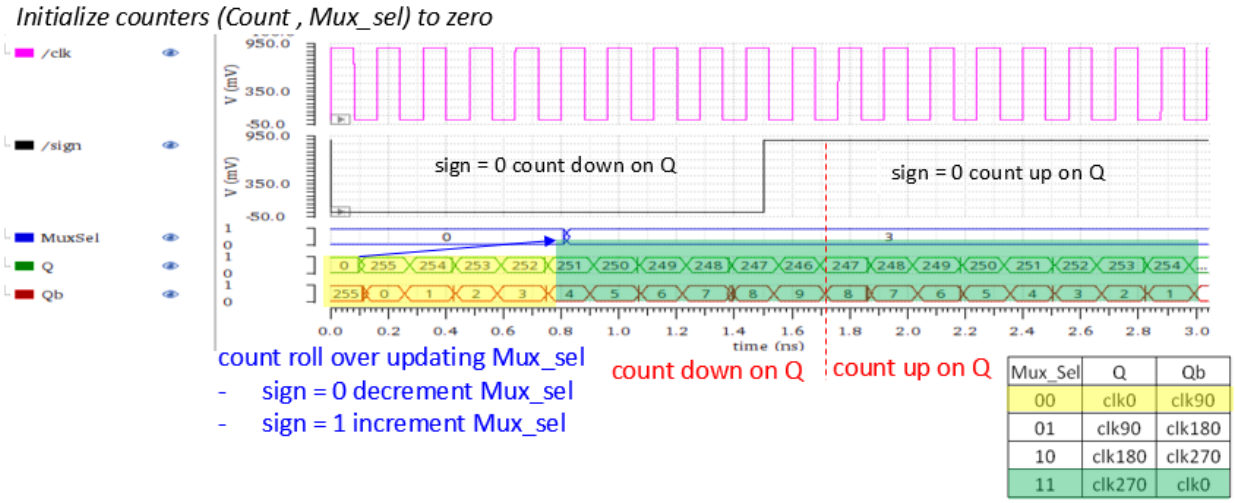


Figure 37. 8-bit up/down counter

Accumulator clock is triggered from the transition signal from the !!PD and the sign bit comes from the !!PD "E" signal. The accumulator and PI control logic simulation are show below in figure 38.



Mux_Sel	Q	Qb
00	clk0	clk90
01	clk90	clk180
10	clk180	clk270
11	clk270	clk0

Figure 38 Accumulator and PI control logic simulations

Q and Qbar signals define the tap weight of the current DAC's in the phase interpolator.

The phase interpolator consists of 2 current DAC's, comparator, output buffer, and clock select logic.

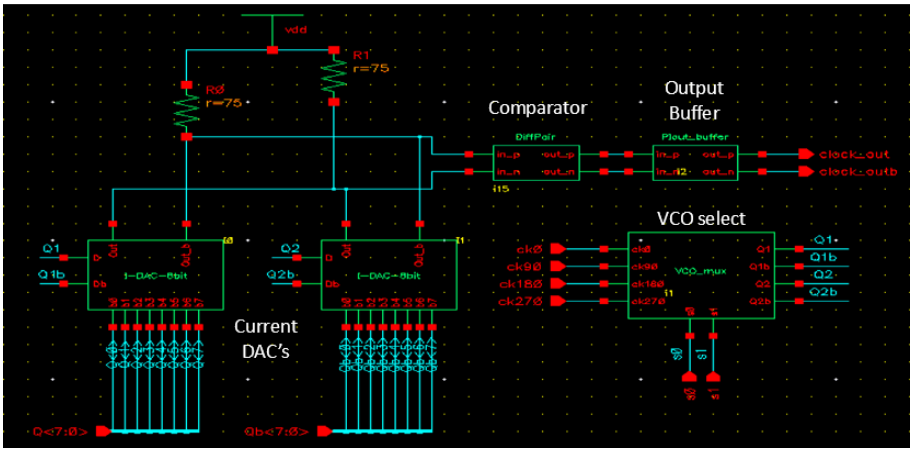


Figure 38 Phase Interpolator

The current DAC's are implemented at the transistor level and the schematics are shown below.

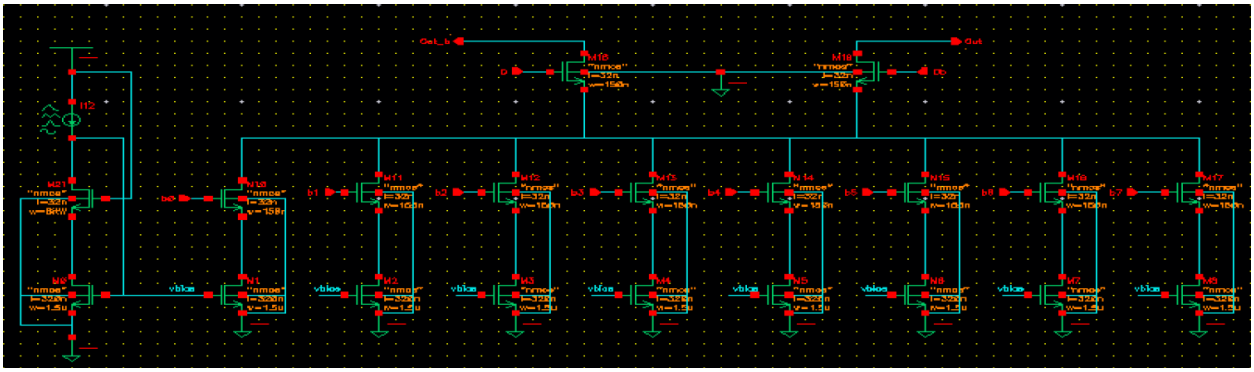


Figure 39. 8-bit current DAC

The DAC and associated current steering pair are designed to provide full rail differential swing. A comparator on the current summing node (figure 38) provide single ended output to an output buffer for driving the !!PD. The DLL loop simulation results are shown below.

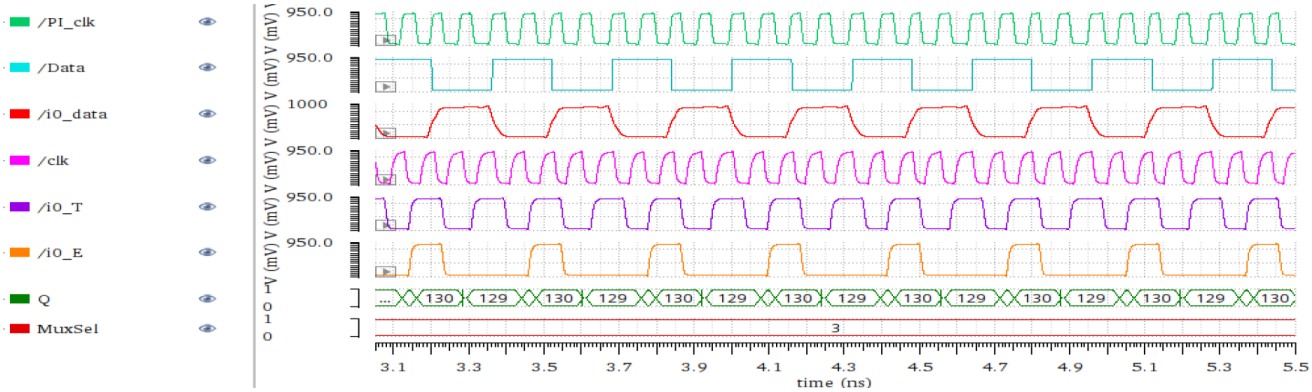
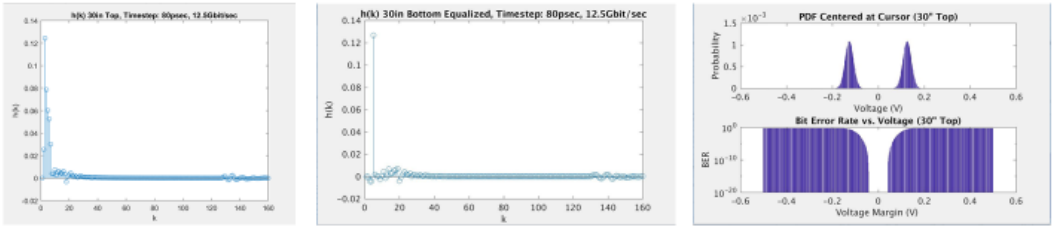


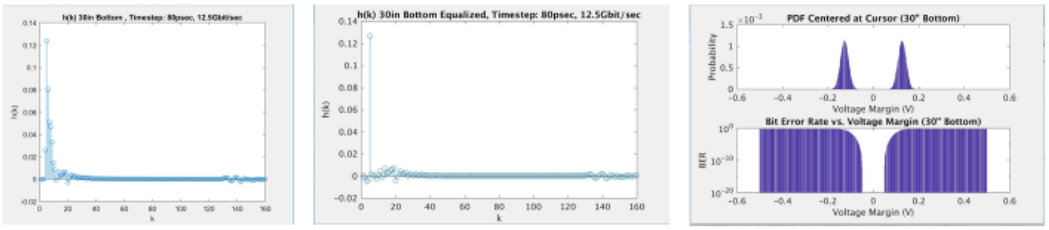
Figure 39 DLL Loop Simulation

8. Results

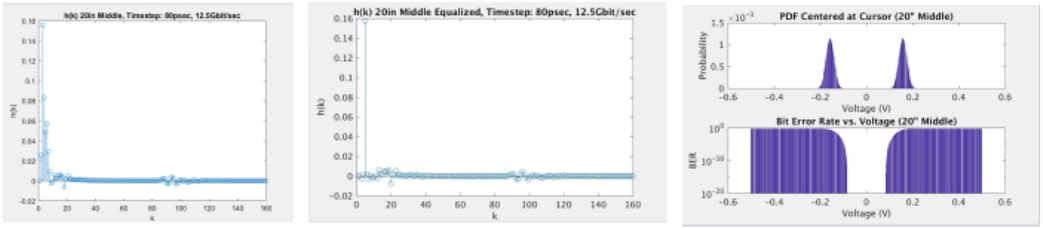
Results 30” Top



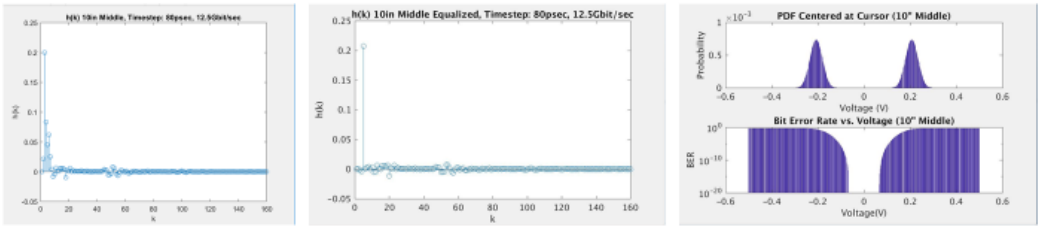
Results 30” Bottom



Results 20” Middle



Results 10” Middle



Results 1.5” Top

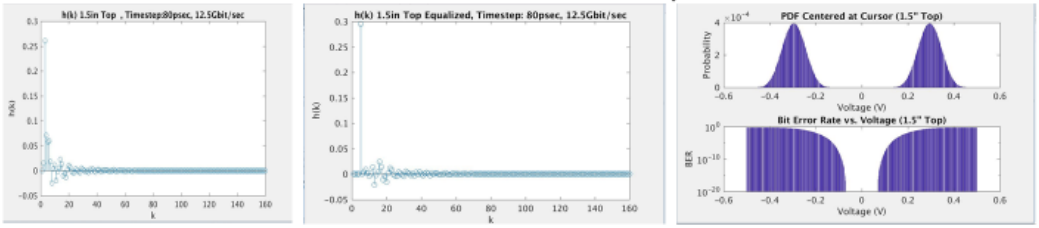
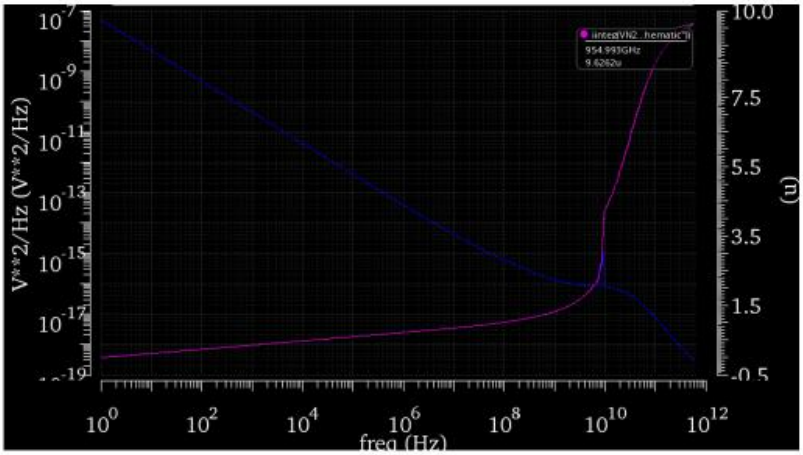


Figure 40

a. BER Calculation



$v_{n,th} = 3.1\ mVrms$

$v_{n,supRx} = 6.2\ mVrms$

$v_{n,supRx} = 7.8\ mVrms$

$v_{n,TOT} = 17.1\ mVrms$

Figure 41 Noise Analysis

Figure 40 shows a noise analysis of the entire link to determine the flicker and thermal noise contribution to at the input of the main data comparator. In addition, AC sweeps were performed on the Tx and Rx supply voltage lines to determine their gain to the differential input to the main data comparator. The contributions of the uncorrelated +/-35mV Tx and Rx supply noise were calculated assuming worst case supply gain. The total worst case random noise is 17.1 mV rms.

The bit error rate was calculated by convolving the statistical probability density function (PDF) of each precursor with every other pre-cursor to find the overall PDF of the sample spaced data post-equalization is shown in the second graph of each channel in *figure 39*. The Overall PDF can be seen in the top-right most graph of each channel in *figure 39,* and below that can be seen the BER plot. The BER is the integral from 0 to infinity of the overall channel PDF. Though the Y-axis of the graph is logarithmic and it cannot be easily seen, it is important to point out that the BER curve approaches exactly 0.5 going to positive and negative infinity, and that the total integral is equal to one.

Assuming the worst case total random noise of 17.1mVrms from above, each channel still has a BER less than 10^{-20}.

b. Power Dissipation

Sub System	Average Power Dissipation
Transmitter	9mW
Receiver	7.53mW

Table 1

The total simulated power dissipation for the channel equalization is 16.53mW. Initially, the adaptation loops are running at full data speed, but this is unnecessary after the channel has been adapted. The design includes a function to stop and lock the channel tap coefficients

after a certain time period, which would eliminate the power due to adaptation logic and dLev loop and significantly decrease the overall power dissipation of the link.

c. Adaptation Results

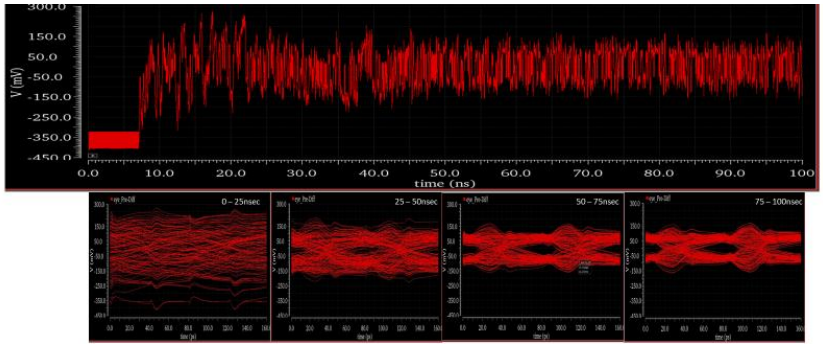


Figure 42

(Transient Simulation of Vrx for 30” Bottom Trace)

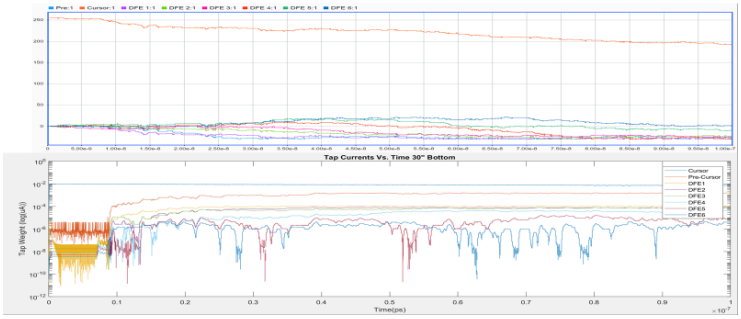


Figure 43

(Transient Simulation of Tap Weights 30” Bottom Trace) Top: Simulink, Bottom: Cadence

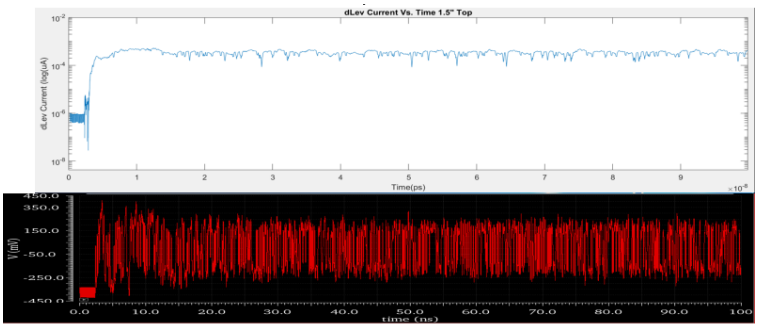


Figure 44

(Transient Simulation of dLev Current Weight) Top: dLev, Bottom: Vrx

Figures 42-44 show the evolution of the dual loop adaptation for the 30” bottom trace over a 100nsec transient simulation. Figure 42 shows 4 eye diagrams containing 25nsec of data spaced 25nsec apart during the transient below the transient voltage at the input to the main data comparator. Figure 43 shows the transient simulation of all taps in cadence and in Simulink (Note that the Simulink plots are signed integers while the cadence plots are in absolute tap current). Lastly, figure 44 shows the current source used to offset the Dlev comparator input line. Below it is Vrx to the main data comparator, which is the data level it is tracking.

d. Closing the First Tap Results

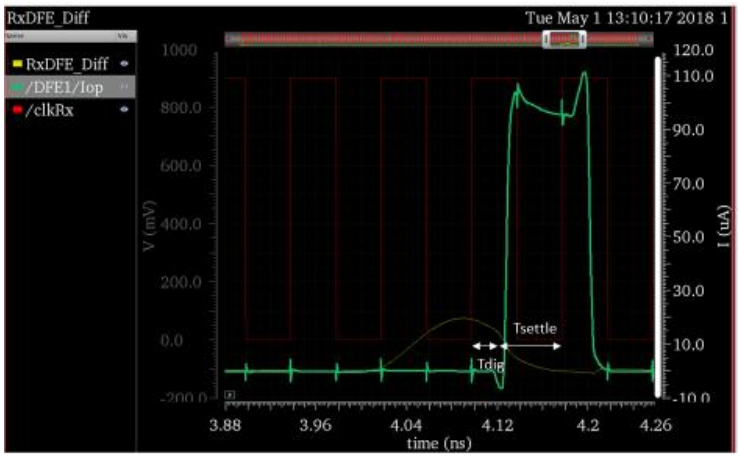


Figure 45

Figure 45 shows the digital delay time from the decision from the main data comparator to the first DFE tap. It takes approximately 30psec for the signal to reach the current switching pair of DFE1, which allows 50psec for the voltage on the summing node to settle. The yellow curve represents the voltage on the summing node, and it can be seen that it has settled appropriately at the edge of the next clock cycle.

9. Conclusion

While a second iteration of design reforms could further optimize the link and CDR, the initial estimates used in the design process brought the entire system very close to optimal size, power and performance.

10. References

1Alon, Elad “EE290C High-Speed Electrical Interface Circuit Design” Lecture, University of California, Berkeley. Spring 2018

2. Alon, Elad “EE240 Advanced Analog Integrated Circuits” Lecture, University of California, Berkeley. Spring 2009

3. V. Stojanovic *et al*., "Autonomous dual-mode (PAM2/4) serial link transceiver with adaptive equalization and data recovery," in *IEEE Journal of Solid-State Circuits*, vol. 40, no. 4, pp. 1012-1026, April 2005.
doi: 10.1109/JSSC.2004.842863

4. P. M. Figueiredo and J. C. Vital, "Kickback noise reduction techniques for CMOS latched comparators," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, no. 7, pp. 541-545, July 2006.
doi: 10.1109/TCSII.2006.875308

5. Palermo, Sam “ECEN689: Special Topics in High-Speed Links Circuits and Systems” Lecture, Texas A&M University, Spring 2010.

6. Lucio Rodoni, George von Büren, Alex Huber, Martin Schmatz, , and Heinz Jäckel; “A 5.75 to 44 Gb/s Quarter Rate CDR With Data Rate Selection in 90 nm Bulk CMOS”, IEEE JOURNAL OF SOLID-STATE CIRCUITS, VOL. 44, NO. 7, JULY 2009

7. Callender, Steven and Ali M. Niknejad. “A phase-adjustable Delay-Locked Loop utilizing embedded phase interpolation.” *2011 IEEE Radio Frequency Integrated Circuits Symposium* (2011): 1-4.

8. Lee, M.-J.E. & Dally, William & Poulton, J.W. & Chiang, Patrick & Greenwood, Stephen. (2001). An 84-mW 4-Gb/s clock and data recovery circuit for serial link applications. 149 - 152. 10.1109/VLSIC.2001.934223.

9. Sidiropoulos, Stefanos. HIGH PERFORMANCE INTER-CHIP SIGNALLING “Technical Report No. CSL-TR-98-760” Computer Systems Laboratory Departments of Electrical Engineering and Computer Science Stanford University, April 1998

10. Razavi, Behzad. (2009). Designing Bang-Bang PLLs for Clock and Data Recovery in Serial Data Transmission Systems. 34-45. 10.1109/9780470545492.ch4.

11. Walker, Richard C. “Designing Bang-Bang PLLs for Clock and Data Recovery in Serial Data Transmission Systems”

Appendix A

Verilog Code

PFD

```
module PFD ( dn, up, loc_clk, ref_clk );

  input ref_clk;
  input loc_clk;
  output up;
  output dn;
reg q1;
reg q2;
wire rst;
assign rst = up & dn;
assign up = q1;
assign dn = q2;

always @ (posedge ref_clk or posedge rst)
begin
    if (rst == 1'b1)
    begin
    q1 <= 0;
    end
    else
    begin
    q1 <= 1;
    end
end

always @ (posedge loc_clk or posedge rst)
begin
    if (rst == 1'b1)
    begin
    q2 <= 0;
    end
    else
    begin
    q2 <= 1;
    end
end
endmodule
```

Charge Pump

```
`include "constants.vams"
`include "disciplines.vams"
`timescale 10ps/1ps
module CP ( nout, pout, dn, up );

  output nout;
  input up;
  input dn;
  output pout;
parameter real cur = 1m;

    electrical pout, nout;
    real out;
analog begin
    @ (initial_step) out = 0.0;
        if (dn && !up)
            out = -cur;
        else if (!dn && up)
            out = cur;
        else out = 0;
            I(pout, nout) <+ -transition(out, 0.0, 5p, 5p);
    end
endmodule
```

VCO

```
module vco_update ( in, out, out0, out90, out180, out270 );

  output out0;
  input in;
  output out90;
  output out;
  output out270;
  output out180;
parameter real amplitude = 1.0,
        center_frequency = 500M,
        gain = 14.545G,
        tran_time = 10E-15;
parameter integer steps_per_cycle = 20;
localparam real omegac = 2*3.14*center_frequency,
        omega_gain = 2*3.14*gain;
electrical in, out, out0, out90, out180, out270, ninty, one_eighty, two_seventy;
integer n, m, x, y;
analog
begin: main
real vin, instantaneousFreq;
vin = V(in);
V(out) <+ amplitude*sin(idt(vin*omega_gain+omegac, 0.0));
//V(out0) <+ amplitude*sin(idt(vin*omega_gain+omegac, 0.0));
//V(out90) <+ amplitude*sin(idt(vin*omega_gain+omegac, 0.0)+1.57);
//V(out180) <+ amplitude*sin(idt(vin*omega_gain+omegac, 0.0)+3.14);
//V(out270) <+ amplitude*sin(idt(vin*omega_gain+omegac, 0.0)+4.71);
V(ninty) <+ amplitude*sin(idt(vin*omega_gain+omegac, 0.0)+1.57);
V(one_eighty) <+ amplitude*sin(idt(vin*omega_gain+omegac, 0.0)+3.14);
V(two_seventy) <+ amplitude*sin(idt(vin*omega_gain+omegac, 0.0)+4.71);
instantaneousFreq = center_frequency + gain*vin;
$bound_step(1.0 / instantaneousFreq / steps_per_cycle);
@(cross(V(out), +1))
begin
    n = 1;
end

@(cross(V(out), -1))
begin
    n = 0;
end
V(out0) <+ transition(n? amplitude:0, 0, tran_time);

@(cross(V(ninty), +1))
begin
    m = 1;
end
@(cross(V(ninty), -1))
begin
    m = 0;
end
V(out90) <+ transition(m? amplitude:0, 0, tran_time);

@(cross(V(one_eighty), +1))
begin
    x = 1;
end
@(cross(V(one_eighty), -1))
begin
    x = 0;
end
V(out180) <+ transition(x? amplitude:0, 0, tran_time);

@(cross(V(two_seventy), +1))
begin
    y = 1;
end
@(cross(V(two_seventy), -1))
begin
    y = 0;
end
V(out270) <+ transition(y? amplitude:0, 0, tran_time);
end
endmodule
```

Divider

```
module divider (
    input clk,
    input rst,
    output clk_div
    );

reg [4:0] count;
reg D;
parameter N=25;
assign clk_div=D;
initial D=0;
initial count = 5'b00000;
always @ (posedge clk or negedge clk)
begin
    if (rst == 1'b1)
    begin
        count <= 0;
    end
    else if (count < (N-1))
    begin
        count <= count + 1;
    end
    else
    begin
    D <= !D;
    count <= 5'b0;
    end
end
endmodule
```

Bang-Bang Phase Detector

```
module bang_bang (
    input in,
    input clk,
    output out,
    output T,
    output E
    );
reg q1, q2, q3, q4;
wire a, b;
always @ (posedge clk)
begin
    q1 <= in;
    q2 <= q1;
    q4 <= q3;
end

always @ (negedge clk)
begin
    q3 <= in;
end

assign a = q1;
assign out = q2;
assign c = q4;
//xor (T, a, out);
//xor (E, c, out);
assign T = a^out;
assign E = out^c;

endmodule
```

Accumulator

```
module accum (
    input clk,
    input en,
    input S_in ,
    output S_out,
    output [7:0]Dout
    );
reg [8:0] SUM;
wire [8:0] P_M;
assign P_M = (~S_in) ? 1:-1;
assign S_out = SUM[8];
assign Dout = (SUM[8]) ? ~SUM[7:0]:SUM[7:0];
//assign Dout = SUM[7:0];
initial SUM = 9'b0;
always @ (posedge clk)
    begin
    if (en)
    begin
        SUM <= SUM + P_M;
    end
    end
endmodule
```

Phase Interpolator

```
// Created by ihdl
`timescale 1fs/1fs

module PI_mux (
    input clk_0,
    input clk_90,
    input clk_180,
    input clk_270,
    input [7:0] data,
    input sign,
    output clk_out,
    output[1:0] mux_s_out,
    output use_clk
    );
reg[1:0] mux_s;
//wire use_clk, clk2;
reg clk_out;
reg[9:0] counter;
wire[9:0] tester;
//initial mux_s[0:1] = 2'b00;

assign use_clk = (mux_s == 2'b00)? clk_0: (mux_s == 2'b01)? clk_90: (mux_s == 2'b10)? clk_180: clk_270;
assign mux_s_out = mux_s;
assign tester = counter;
initial mux_s = 2'b00;
initial counter = 10'b0000000000;
always @ (posedge use_clk)
begin
if ((data == 8'b11111111) && (tester > 10))
    begin
    mux_s <= mux_s + (sign?-1:1);
    counter <= 0;
    end


else
    begin
    counter <= counter +1;
    end
end

always @ (posedge use_clk)
begin

clk_out <= #((data[0]?78:0) + (data[1]?156:0) + (data[2]?312:0) + (data[3]?624:0) + (data[4]?1248:0) + (data[5]?2496:0) + (data[6]?4992:0) + (data[7]?9984:0)) 1;

end
always @ (negedge use_clk)
begin

clk_out <= #((data[0]?78:0) + (data[1]?156:0) + (data[2]?312:0) + (data[3]?624:0) + (data[4]?1248:0) + (data[5]?2496:0) + (data[6]?4992:0) + (data[7]?9984:0)) 0;

end

endmodule
```