The Report Committee for Anuwat Saetow
Certifies that this is the approved version of the following report:

Application of Digital Calibration Technique on Global Bidirectional Interconnects in Integrated Circuit

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

David Zhigang Pan

Lizy Kurian John

Application of Digital Calibration Technique on Global Bidirectional Interconnects in Integrated Circuit

by

Anuwat Saetow, B.S.E.

Report

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

The University of Texas at Austin

December 2014

Dedicated to my parents, Thanapol and Vallaya.

Acknowledgements

I would like to thank my supervisor Professor David Z. Pan for all the advice and guidance. I would also like to thank Professor Lizy K. John for being my reader on this report. This work would not have been possible without their support and tutelage.

Application of Digital Calibration Technique on Global Bidirectional Interconnects in Integrated Circuit

Anuwat Saetow, M.S.E.

The University of Texas at Austin, 2014

Supervisor: David Z. Pan

The trend to integrate more and more processing cores and memory cores into a single module has increased the overall size of chips to the point where global interconnects between sub-units are becoming harder and harder to route and meet timing rules and requirements. The traditional way of routing interconnects and the use of uniform, unidirectional, point to point busses may no longer be optimal for certain designs where metal layers and chip area for interconnects are limited. The need for a more flexible routing methodology is necessary and can be achieved by using routing and calibration techniques currently being implemented at board level design. This report proposes the use of non-uniform, bidirectional, and possibly multi-point loads global interconnects within a single chip module through the use of on chip calibration techniques to compensate for less restrictive wiring rules for certain chip designs. This report will also apply a widely used digital calibration technique to simulate the implementation on a field programmable gate array.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

The trend to integrate more and more processing cores and memory cores into a single module has increased the overall size of chips to the point where global interconnects between sub-units are becoming harder and harder to route and meet timing rules and requirements. It has been projected that embedded memory, and inevitability, their buses, are going to dominate area usage on system on chip (SoC) within the next few years [15]. New challenges has come forth in the design of global interconnect infrastructures for these large and highly integrated SoCs [3]. The traditional way of routing interconnects and the use of uniform, unidirectional, point to point busses may no longer be optimal for certain designs where metal layers and chip area for interconnects are limited. The need for a more flexible routing methodology is necessary and can be achieved by using routing and calibration techniques currently being implemented at the board design level.

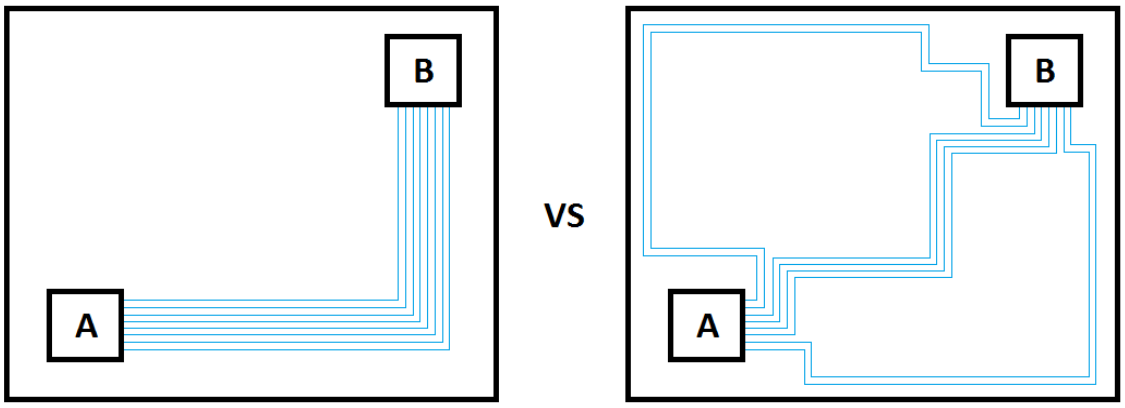

Figure 1: Flexibility of uniform (left) versus non-uniform (right) interconnects.

This report proposes the use of non-uniform, bidirectional, and possibly multi-point loads global interconnects within a single chip module through the use of on chip data lane

calibration techniques to allow for compensation of less restrictive wiring rules. This report will also apply a widely used digital calibration technique to simulate the implementation on a field programmable gate array (FPGA).

Chapter 2: Fundamentals

Integrated circuits (ICs) on a single monolithic die with over one billion transistors will put a tremendous strain on interconnection networks in chips. It has been estimated that modern monolithic dies will have over 10^{17} coupling inductances and capacitances across nine to ten metal layers. The interconnects on such a chip will be the limiting factor in semiconductor design and performance because, unlike transistors, on chip interconnects do not benefit from miniaturization the same way transistors do [11]. While logic gates delay time decreases with scaling, interconnect delays will actually increase [16]. On the bright side, there are some benefits for global interconnects from scaling. As transistors get smaller and smaller, the size of the same chip in the new technology also becomes smaller thus shortening wire lengths between logic blocks [16]. It has also been postulated, depending on assumptions used, that delay ratios are close to unity [16][29]. However, as we add more and more processing cores and memory cores into a single die, the overall size continues to increase and interconnects between sub-units are still long and remains an issue and poses routing challenges.

2.1 CURRENT SOLUTIONS

Designers have developed tools such as analytical wire RC delay models and simulations along with floor planning techniques as solutions to interconnect design issues. Distributed RC delay models are used in interconnect designs to attack topology and wire sizing issues [8]. Commonly used methods such as delay-driven segregation of local versus global routing and layout-driven synthesis all strive to keep symmetric uniform routing of groups of interconnect buses from sub-unit to sub-unit within an integrated chip [16]. This classical approach works well if resources can be allocated and planned early on to ensure stringent rules and routing requirement are met.

Other solutions have also been developed to solve global interconnect design issues, each with its own drawbacks. Networks on chip (NoC) proposes that designers "route packets, not wires" [10]. This solution places costly routers within each logic tile and connects them through a logic network shown below.

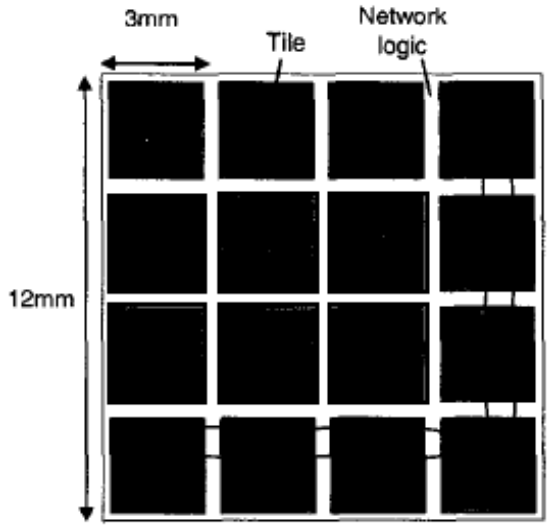

Figure 2: Partition die into tiles and network logic [10].

Exotic medium solutions such as photonic NoCs and wireless NoCs have also been proposed. Complex electronic to optical and optical to electronic conversion network gateways are needed [26]. Optical transceivers for photonic NoCs require costly laser sources and modulators. Routing paths thru the use of waveguides are also needed [17]. Wireless NoCs also entails additional complex hardware on chip. Wireless transceivers require high frequency voltage controlled oscillators (VCO), modulators, amplifiers, and on chip antennas [12]. A conceptual diagram of an intra-chip wireless interconnect system is shown below.

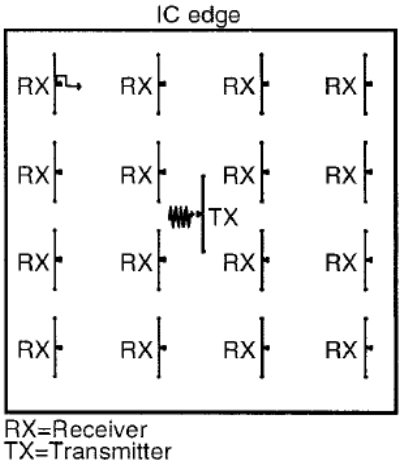

Figure 3: Conceptual illustration of intra-chip wireless interconnect [12].

The need for a more flexible routing methodology is becoming more and more essential to future global interconnect designs in integrated circuits. While other solutions exist, they come at a very high cost in terms of undue constraints in floor planning, requires exotic process technology to implement, and/or addition of complex transceivers on chip. This report will demonstrate how designers of global interconnects can have less restrictive routing and wiring rules with minimal floor plan constraints through the use of data lane calibration techniques. Through this methodology, asymmetric/non-uniform data paths can be utilized within a single interconnect bus group running from one sub-unit to another requiring only basic technology processes and medium already widely used in industry.

2.2 OBSTACLES

There are some key impediments that must be taken into consideration when designing global interconnects in an integrated circuit and still achieve proper functionality. Three main obstacles are: propagation delay, signal integrity (SI), and process variation. This section will provide a brief overview of each challenge.

2.2.1 Propagation Delay

As technology nodes scale to smaller and smaller dimensions, propagation delays through a fixed-length wire will increase, even with the use of low-k dielectrics and new copper technologies [16]. Most distributed RC delay models are based on the following simplified formula, $t = \sum R_k^* \cdot C_k^*$, where R_k^* is defined as the resistance between source and node k while C_k^* is defined as the capacitance at node k. This model, while simple, still retains an accurate delay upper bound and is comparable to the commonly used Elmore delay model [8].

2.2.2 Signal Integrity

Signal integrity has become a critical issue to proper interconnect operation in modern chip designs. It is commonly defined as "the ability of a signal to generate assured correct responses in a circuit" [4]. Signal shape of waveforms traversing through interconnects are influenced by activities in nearby sources and as a consequence appears as noise [28]. Sources of noise can come from multiple avenues and depends on electrical characteristics (resistance, capacitance, and inductance) [4][28]. Because of this, transmission lines must be shielded with insulators to reduce signal loss and minimize cross-talk [28]. Without proper signal integrity, receivers of sub-units within a chip will not be able to correctly receive data from other parts of the chip. The metric for good or adequate signal integrity characteristics are stable high logic state, stable low logic state, adequate slew rate with limited over/under shoot, and minimum skewing, and is illustrated in Figure 4 below [4].

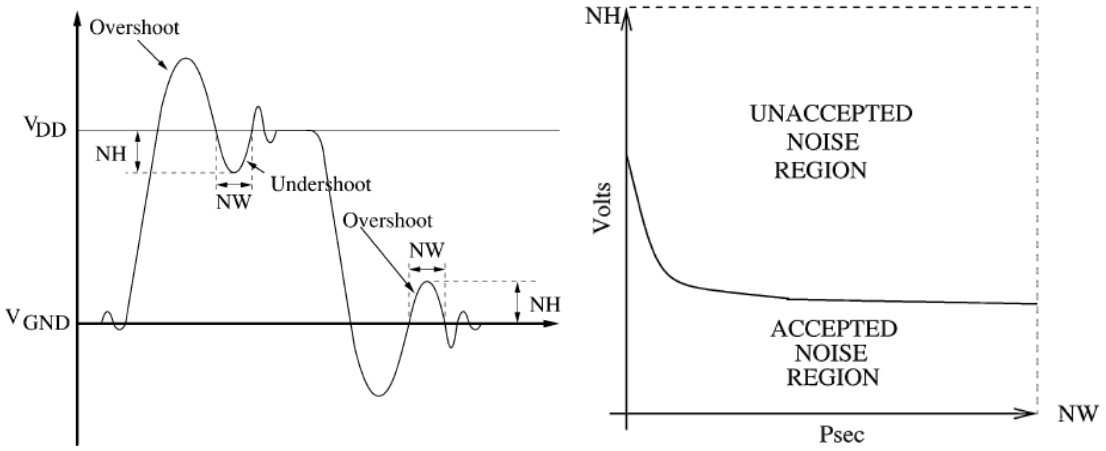

Figure 4: Example of overshoot and undershoot acceptability [4].

Through the use of simulation and adherence to layout, placement, and routing rules, signal integrity compliance can be met although that in and of itself is not a guarantee of proper operation. Present computer aided design (CAD) tools are unlikely to take all possible operating conditions into account and consequently signal integrity will always be a design issue in modern ICs [4].

2.2.3 Process Variation

Process variation is another key impediment to the design of global interconnects as smaller and smaller process nodes become available and more pervasive. Large variations, either systematic and/or random, in the process can undo results generated by analytical delay models and simulations which can impair proper operation. Ring oscillators can be used to detect and sort out issues in process variations but it does not remedy the situation [18]. Also, the question of localized versus generalized variation within a single chip can also impact global interconnect routing and make attempts to model for worst case variation more difficult. Different regions within the same chip can have widely different parametric characteristics which in turn can affect propagation

delay of certain data lanes within a given symmetric uniform interconnect group thus making them asymmetric [18].

2.3 CALIBRATION

2.3.1 Phase Rotators

Data lane wiring calibration used in board level design primarily revolves around the use of phase rotators or delay muxes that are capable of delaying early signals relative to late signals. Phase rotators, sometimes called phase shifters basically functions as adjustable delay elements and are usually implemented as analog circuits. They generally span 360˚ and come in 2^N evenly spaced discrete phases as seen below [30].

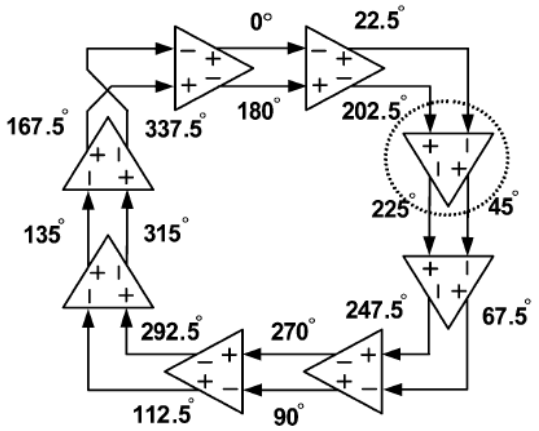

Figure 5: Conceptual illustration of a, 16 phase, phase rotator [14].

Since these phase rotators are discrete, they are susceptible to aliasing where the required solution falls between two adjacent 2^N settings. To reduce the chance of aliasing, N can be increased so that spaces between discrete phases are smaller. The down side to making N bigger is the extra cost in hardware and possible loss of linearity due to increase in error to phase step size ratio [30]. For proper digital calibration results to be achieved, phase rotators must be linear across the full 360˚.

2.3.2 Digital Calibration

Digital calibration technique on bidirectional interconnects aims to synchronize multiple uneven data lane wire groups running from one sub-unit to another within the same integrated circuit. This technique is already being used at the board level for chip to chip connections as shown in Figure 6 below [1].

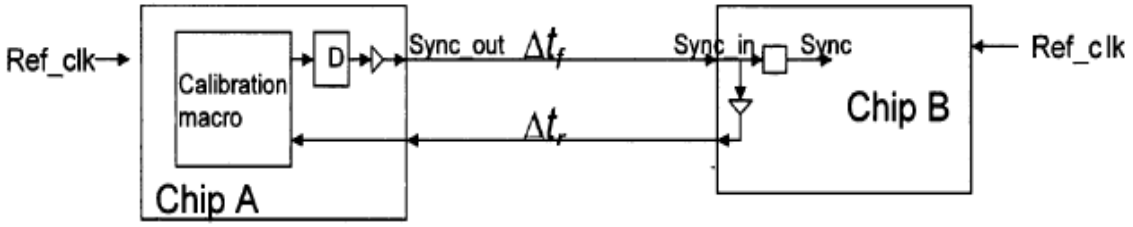

Figure 6: Illustration of digital calibration used between two chips [1].

Note that each data lane group are defined by having a minimum of two bidirectional data lanes, one for the sync signal and the other for the response signal, with roughly equal wire length and characteristics. More than two bidirectional data lanes with equal wire lengths can be grouped but only two will be actively involved in the calibration process, unless logic gates are used. Once a sync signal lane within a group has been assigned, if the remaining lanes are ORed together than preference will be given to the fastest arriving response signal lane. However, if the remaining lanes are ANDed together, than preference will be given to the slowest arriving response signal lane.

The first step to calibration is to measure the round trip delay between the two units on each data lane group. This is accomplished by having sub-unit A start its timer and send a sync signal to sub-unit B. When sub-unit B receives the signal, it sends its own response signal back to sub-unit A. When sub-unit A receives the response signal, it stops its timer and latches in the count. Since there are latches in each unit, the round trip

delay is the time it takes to prorogate the sync signal from A to B, plus the time it takes B to latch the signal and responds, plus the time it takes to propagate the response signal back from B to A [1]. By taking the round trip delay of all the data lane groups, simple arithmetic can be used to calculate what the phase rotator value should be for each lane. In this case, take the longest round trip delay of all the groups subtract each group's round trip delay from that value and divide it in half, we get each group's correct phase rotator offset value. Note that given this algorithm, the longest group's phase offset will always be zero since any number subtract by itself is zero, and dividing zero in half is still zero. Because of this, the longest lane's delay will be the upper bound of all lanes, plus or minus any aliasing from the phase rotators, the calibration engine, and any deviation in line characteristic within each data lane group. Note that any deviation in line characteristic within each data lane group will need to be accounted for using traditional global interconnect methods described in Section 2.1, "Current Solutions".

2.3.3 Fault Tolerance

Since we are relaxing the traditional rigid wiring rules and routing requirement, care should be taken to deploy redundant bidirectional data lane groups in cases where calibrate was unsuccessful. Spare data lane groups could be added along with steer muxes to shift data away from the defective data lane group [19]. Alternately, error correcting code (ECC) can be added to the interconnect bus logic block to tolerate minute errors on the bus.

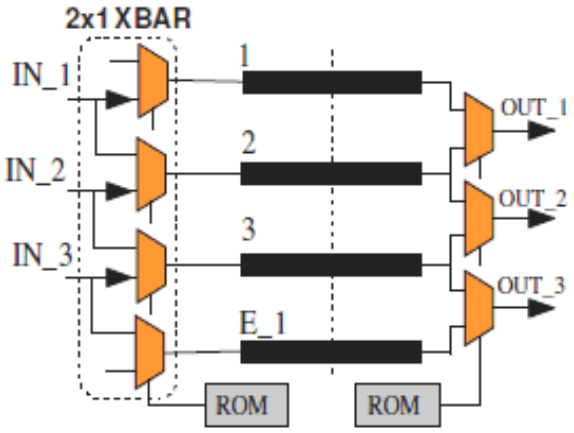

Figure 7: Example of redundant bit lanes and steer muxes [19].

Chapter 3: Implementation

3.1 SETUP

Application of the digital calibration technique on an FPGA first requires setup of the test environment. This was facilitated through the use of the following:

- 2ea, Digilent's Atlys FGPA Development Board (500MHz) [31]
- 2ea, 10-Pin (2x5) Connector Header 0.100"(2.54mm) Pitch
- 2ea, 50ohm Coax Shielded Cable, 131"(332.74cm) [DQS0/DQ0]
- 2ea, 50ohm Coax Shielded Cable, 65.5"(166.37cm) [DQS1/DQ1]
- 2ea, 50ohm Coax Shielded Cable, 10.875"(27.623cm) [NotUsed]
- 2ea, 50ohm Coax Shielded Cable, 5.531"(14.049cm) [DQS2/DQ2]

Each of the eight 50ohm coax cables had to be stripped, tinned, and had its ground (GND) shield twisted and extended away from the core at both ends. To create the non-uniform wire assembly, each pair of equal length coax cable (long, medium, short2, and short1) was attached to the 10-pin connector header assembly with the longest pairs of wires on the right and the shortest placed one position in from the left most pin to make room for the GND on the far left as seen in Figure 8.

Pmod5	Pmod4	Pmod3	Pmod2	Pmod1
GND	DQS2	-	DQS1	DQS0
Shield	Short1	Short2	Medium	Long
Pmod11	Pmod10	Pmod9	Pmod8	Pmod7
GND	DQ2	-	DQ1	DQ0
Shield	Short1	Short2	Medium	Long

Figure 8: Connector header assembly pin assignment.

Scrap 50ohm coax cores was used to connect all eight coax shied to the GND pin on the far left and also to bridge the top and bottom GND pins on the connector head assembly

itself. Hot glue was then added near the ends of the coax cable to hold everything in place so that undue strain would not put on the solder joints. Note that it was critical to get all the signals soldered and placed correctly, improper solder joints added signal integrity issues and caused the calibration engine to malfunction. Once complete, both connector head assembly should look like Figure 9 below.

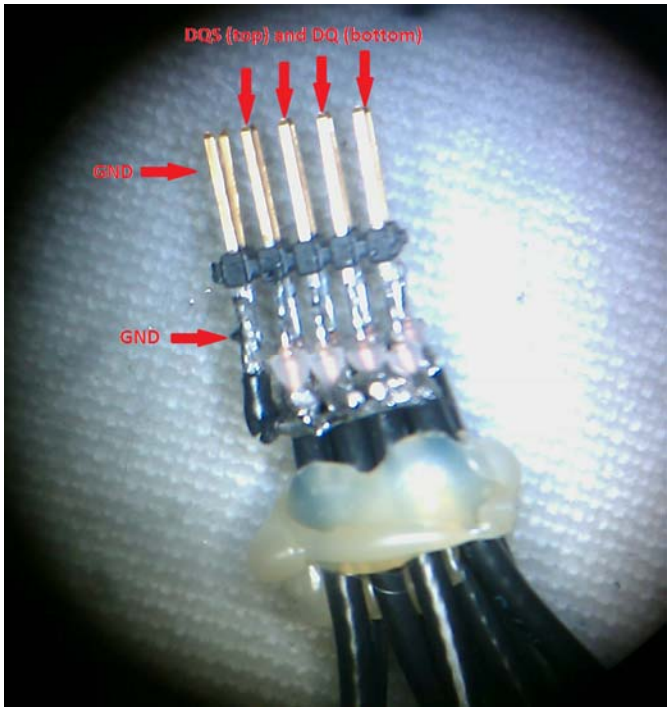

Figure 9: Connector header assembly.

The connector headers are then plugged into the Pmod connector of the Atlys FPGA boards with DQS0 using the longest cable inserted into Pmod pin 1. Note that Pmod pins 6 and 12 are not connected since they carry Vcc which were not required. Shielding was provided by the GND connection on Pmod pins 5 and 11. The completed connections are illustrated in Figure 10 below.

Figure 10: Variable length wire assembly connected to FPGA boards.

3.2 FPGA IMPLEMENTATION

The Atlys development board from Digilent uses Xilinx's Spartan-6 FPGA core. The two FPGA boards are simulating two sub-blocks within an integrated circuit and the uneven coax cables are simulating non-uniform global interconnect wires. The development board has a 100MHz reference clock (CLK) which can be bumped up internally to run the Spartan-6 FPGA at 500MHz. The Atlys board contains 22 basic input/output (I/O) used for the 8 light emitting diodes (LEDs), 8 switches, 5 generic buttons, and 1 reset button seen in Figure 11.

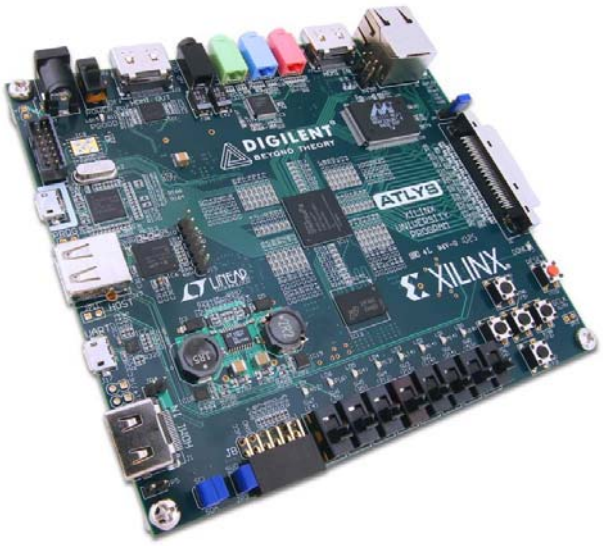

Figure 11: Atlys FPGA board [31].

These basic I/Os were utilized and defined based on three modes of operation plus a designation as master (send) or slave (receive) device. The three primary modes of operation are: standby/debug, calibration, or operation. It also has 8 bidirectional I/O assigned to the Pmod port which uses a standard rectangular connector [31]. Since a minimum of 6 bidirectional data I/O were required for this application, I opted to use the Pmod port connector because of its relative ease of use.

3.2.1 Limitations

Since there are no actual phase rotators on the Atlys development board, the delay capability of a, 16 phase, phase rotator were simulated by allowing each rising edge of the 500MHz clock to equal one increment of a phase rotator delay value. The down side of this was two fold. The equivalent functional clock frequency and thus the data strobe (DQS) and data (DQ) switch rate of the sub-block was reduced to 1/16 of 500MHz or 31.25MHz. A scope shot was taken to verify that the new functional clock frequency was close to 31.25MHz in Figure 12 below.

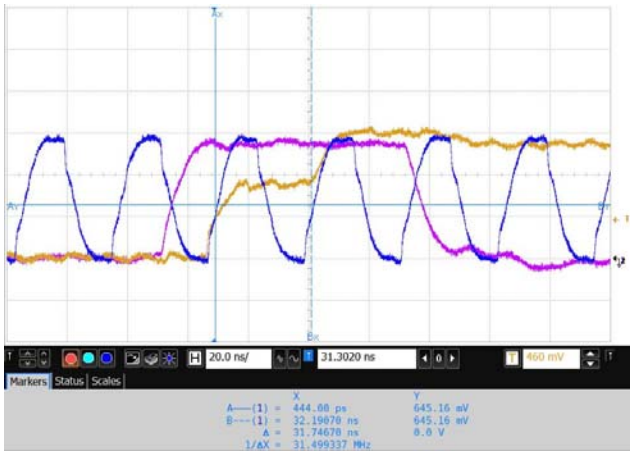

Figure 12: Functional clock (blue signal) frequency measured.

Also, given 31.25MHz as opposed to the original 500MHz, the ability to simulate a half cycle propagation delay at 31MHz requires that the longest wire length had to be almost 333cm or 3.33 meters long. Given a wire that was over 3 meters long and generically designed I/O drivers/receivers on an FPGA, obstacles previously stated in Section 2.2 was now a real issue and made the use of the 50ohm coax shielded cable a requirement for proper operation. Figure 13 below is a side by side comparison of a coax versus a 24 American wire gauge (AWG).

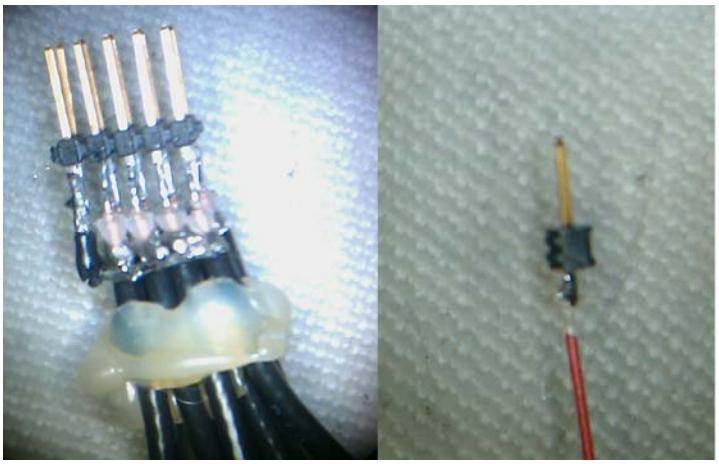

Figure 13: Shielded coax versus unshielded 24 gauge wire.

From Figure 14 below, we can see that even at just 1.66 meters, half the length of the long wire, DQS1's signal at the far-end using 24AWG takes over 150ns to get to stable high logic state. At 31.25MHz frequency or 32ns period, 150ns would be equivalent to 4.6875 clock cycles. If the calibration engine was attempting to calibrate that, the counters for the turn around time might overrun, assuming it detected the sync signal at all. Even if it didn't overrun, it would not have calibrated to a valid result. Looking at the 50ohm shielded coax cable however, we see its rise time was about 2ns, which is within 1/16 of a clock cycle and also happens to equal one click of the simulated phase rotator value. From the 24AWG near-end scope shot below, we also see that it has bad SI, with roughly 40% overshoot in the signal compared to the coax near-end.

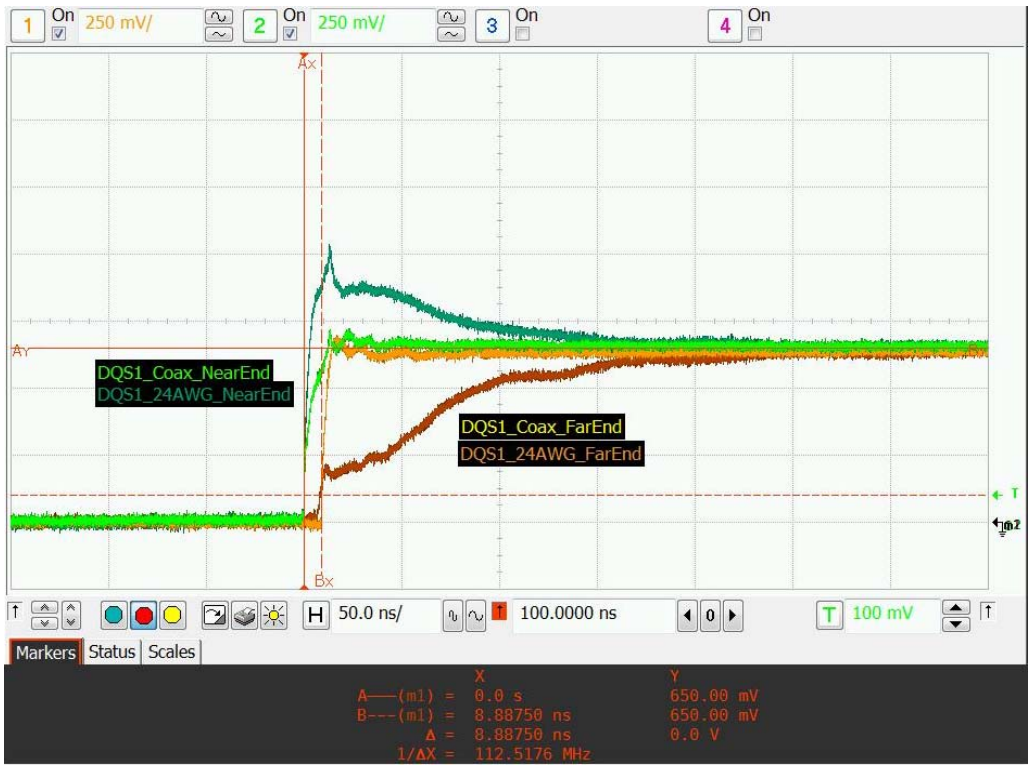

Figure 14: Scope shot of coax versus 24 gauge wire at 166cm.

3.2.2 Standby/Debug Mode

The FPGA was setup into three primary modes, one of which was standby/debug. The standby/debug mode consists of three sub-modes: standby, wire test (receive), and wire test (send). In standby, the input/output buses are put into a safe state where both FPGAs are tri-stated and both DQS/DQ bus control registers are kept low as seen in Table 1 below.

Table 1: Input and output bus/control for standby/debug mode.

	Mode=Switch[7:0]	Standby=0x00	WireTest(Receive)=0x01	WireTest(Send)=0x02
I/O Bus	BUS_DQS[2:0]	Tri-State	Input(dqs_in[2:0])	Output(dqs_out[2:0])
	BUS_DQ[2:0]	Tri-State	Input(dq_in[2:0])	Output(dq_out[2:0])
Bus Cntl	dqs_cntl	LOW	LOW	HIGH
	dq_cntl	LOW	LOW	HIGH

This mode was also used as a transition mode between other modes. This is needed to ensure that both FPGAs did not drive into the same bus at the same time which may damage the I/O of the FPGAs. In standby, all indicator LEDs are statically off and none of the buttons were utilized as seen in Table 2 below.

Table 2: Switch, button, and LED assignment for standby/debug mode.

	Mode=Switch[7:0]	Standby=0x00	WireTest(Receive)=0x01	WireTest(Send)=0x02
Indication(output)	LED[0]	Static OFF	DQS0 State	DQS0 State (Static ON)
	LED[1]	Static OFF	DQS1 State	DQS1 State
	LED[2]	Static OFF	DQS2 State	DQS2 State
	LED[3]	Static OFF	DQ0 State	DQ0 State
	LED[4]	Static OFF	DQ1 State	DQ1 State
	LED[5]	Static OFF	DQ2 State	DQ2 State
	LED[6]	Static OFF	Static ON	Static ON
	LED[7]	Static OFF	Static ON	Static ON
Function(input)	Button[Up]	Not Used	NotUsed	DQS1 HIGH
	Button[Left]	Not Used	NotUsed	DQS2 HIGH
	Button[Center]	Not Used	NotUsed	DQ0 HIGH
	Button[Right]	Not Used	NotUsed	DQ1 HIGH
	Button[Down]	Not Used	NotUsed	DQ2 HIGH

The wire test (send/receive) modes were used in conjunction with each other to debug any I/O or wiring issues. To properly operate this mode, one FPGA was places into receive mode while the other into send mode. Both FGPAs will indicate static on for LED[7:6] but the master FPGA used for the send function will also indicate a static on for LED[0] which corresponds to a high output on DQS[0] bus lane. DQS[0] was not connected to any of the five buttons and will always default to high in the wire test (send) mode. Upon receiving the high DQS[0] signal from the master FPGA, the slave FPGA which was set to wire test (receive) mode, will light LED[0]. If a high signal was not received properly, the designated LED will not be lit. The remaining data lanes DQS[2:1] and DQ[2:0] are mapped to the buttons and LEDs as indicated in Table 2 above.

Oscilloscope channel de-skew was accomplished, in wire test mode. Ground from channel 1 through 3 of the o-scope was attached to the slave FPGA which was set to wire test (receive) mode. Also, all three o-scope signal channels were attached to the slave FPGA at Pmod pin 1, DQS0 lane. The o-scope was set to default 0ps skew on all three channels and trigger at 118mV on channel 2 as seen in Figure 15 below.

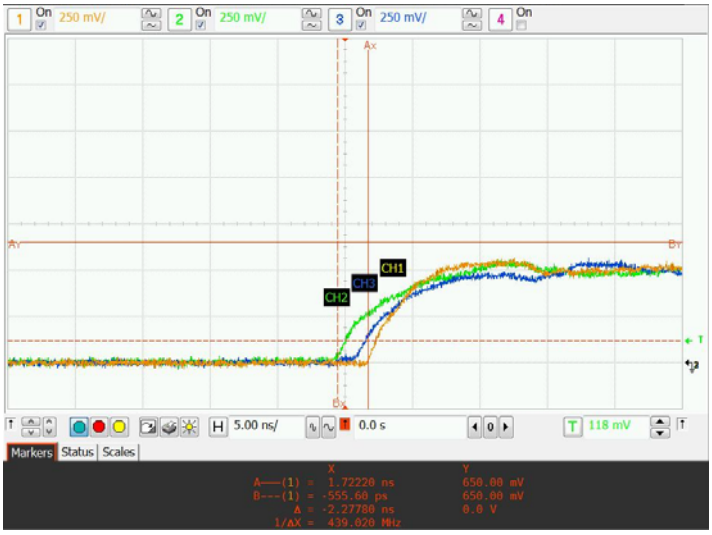

Figure 15: Oscilloscope channels before de-skew.

The DQS0 signal was set high as soon as the master FPGA's mode switch was set to wire test (send) mode with switch[7:0] set to 0x02. By adjusting the channel skew on each scope channel all three channels was de-skew to channel 1, as seen in Figure 16 below, with 0ps default for channel1, -2.25ns adjustment for channel 2, and -750ps adjustment for channel 3.

Figure 16: Oscilloscope channels after de-skew.

3.2.3 Calibration Mode

The calibration mode consists of four sub-modes: calibrate (receive), calibrate (send), calculate calibrated results, and display calculated value. In calibrate (receive) mode, dqs_in[2:0] is set to input from BUS_DQS[2:0] while dq_out[2:0] is outputted to BUS_DQ[2:0]. In calibrate (send) mode, dqs_out[2:0] is set to output to BUS_DQS[2:0] while dq_in[2:0] is inputted from BUS_DQ[2:0]. For calculate calibrated results mode and display calculated value mode, input/output buses are put into tri-stated and both DQS/DQ bus control registers are kept low as seen in Table 3 below.

Table 3: Input and output bus/control for calibration mode.

	Mode=Switch[7:0]	Calibrate(Receive)=0x04	Calibrate(Send)=0x08	CalculateCalResult=0x10	DisplayCalcValue=0x20
I/O Bus	BUS_DQS[2:0]	Input(dqs_in[2:0])	Output(dqs_out[2:0])	Tri-State	Tri-State
	BUS_DQ[2:0]	Output(dq_out[2:0])	Input(dq_in[2:0])	Tri-State	Tri-State
Bus Cntl	dqs_cntl	LOW	HIGH	LOW	LOW
	dq_cntl	HIGH	LOW	LOW	LOW

The calibrate (send/receive) modes were used in conjunction with each other to initiate calibration. To properly operate this mode, one FPGA was places into receive mode while the other into send mode, in that order. The slave FGPA in receive mode will indicate static off for all LED[7:0] and does not make use of any buttons. The master FPGA in send mode will make use of indication LED[7:0] and all buttons as seen in Table 4 below.

Table 4: Switch, button, and LED assignment for calibration mode.

	Mode=Switch[7:0]	Calibrate(Receive)=0x04	Calibrate(Send)=0x08	CalculateCalResult=0x10	DisplayCalcValue=0x20
Indication(output)	LED[0]	Static OFF	Calibration DQS/DQ 0 OK, Calibration Value[0]	Static OFF	Calculated Value[0]
	LED[1]	Static OFF	Calibration DQS/DQ 1 OK, Calibration Value[1]	Static OFF	Calculated Value[1]
	LED[2]	Static OFF	Calibration DQS/DQ 2 OK, Calibration Value[2]	Static OFF	Calculated Value[2]
	LED[3]	Static OFF	Calibration Value[3]	Static OFF	Calculated Value[3]
	LED[4]	Static OFF	Calibration Value[4]	Static ON	Calculated Value[4]
	LED[5]	Static OFF	Calibration Value[5]	Static OFF	Calculated Value[5]
	LED[6]	Static OFF	Counter Value Overflow ERROR	Static OFF	Static OFF
	LED[7]	Static OFF	Calibration COMPLETE	Static OFF	Static ON
Function(input)	Button[Up]	Not Used	Set LED[5:0]	Not Used	DQS/DQ 0 - 2 Calc Display LED[5:0]
	Button[Left]	Not Used	DQS/DQ 2 Delay Display LED[5:0]	Not Used	DQS/DQ 2 Delay Display LED[5:0]
	Button[Center]	Not Used	DQS/DQ 1 Delay Display LED[5:0]	Calculate	DQS/DQ 1 Delay Display LED[5:0]
	Button[Right]	Not Used	DQS/DQ 0 Delay Display LED[5:0]	Not Used	DQS/DQ 0 Delay Display LED[5:0]
	Button[Down]	Not Used	Clear LED[5:0]	Not Used	DQS/DQ 0 - 1 Calc Display LED[5:0]

To initiate calibration, the slave FPGA was set to calibrate receive mode with switch[7:0] set to 0x04. The DQS bus control signal (dqs_cntl) was set low to input and the DQ bus control signal (dq_cntl) was set high to output as seen in Table 3 above. Upon detection of a high signal on a DQS wire, the slave FPGA raises its corresponding DQ wire from low to high, with DQ0 responding to DQS0, DQ1 responding to DQS1, and DQ2 responding to DQS2. On the master FPGA side, cal_state[3:0] was used to keep track on different calibration states. Cal_state[3:0] was only updated during rising/falling

edge transitions of the 31.25MHz functional clock. At cal_state 0, the DQS bus control signal was set high to output, the DQ bus control signal was set low to input, per lane calibration done indicators cal_done[2:0] was reset to 0b000, and cal_state 0 was incremented to cal_state 1 on the falling edge of the functional clock. In cal_state 1, on the rising edge of the functional clock, cal_state is incremented to 2, all three DQS lanes are set to output high, and cal_count[7:0] used to count delay between DQS to DQ is set to reset value of 0x00. In cal_state 2, cal_count[7:0] is incremented on the rising edge of the 500MHz clock and each DQ input was monitored for a high response. An example of DQS0 going high on the master FPGA and then DQ0 going high as a response from the slave FPGA is illustrated in Figure 17 below.

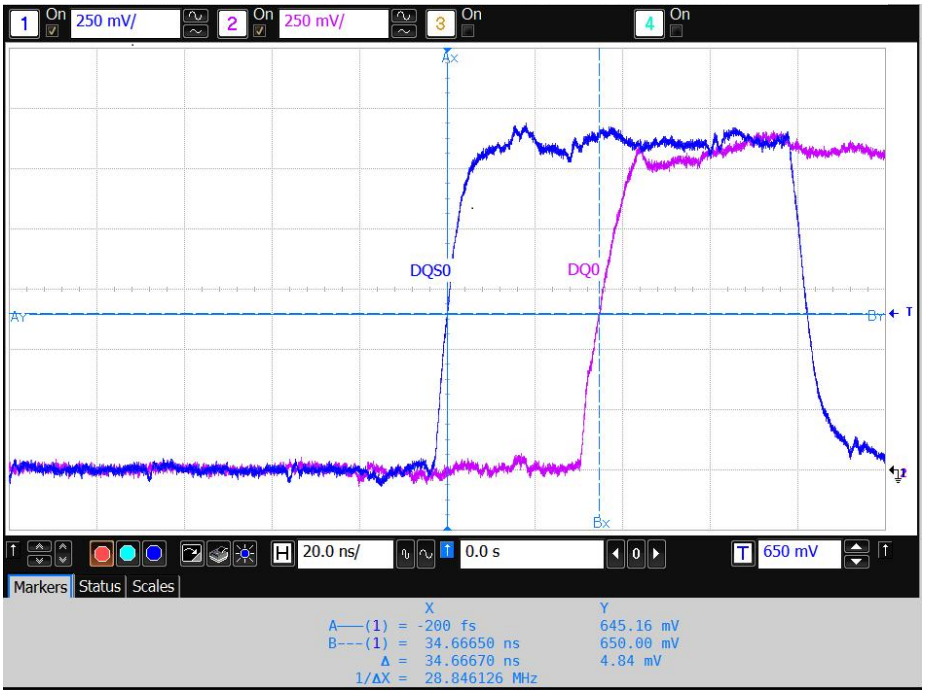

Figure 17: DQS0 far-end from master to DQ0 far-end from slave.

Note that Figure 17 was taken at the far-end of each source device, DQS0 from the far-end of the wire relative to the master FPGA at the slave FPGA input and DQ0 from the

far-end of the wire relative to the slave FPGA at the master FPGA input. If a response is received on a particular DQ, the cal_done bit corresponding to that lane will be set to high and cal_count[6:1] will be transferred to the corresponding raw delay register; with raw_delay_0[5:0] assigned to DQS0/DQ0, raw_delay_1[5:0] assigned to DQS1/DQ1, and raw_delay_0[5:0] assigned to DQS2/DQ2. Note that at this state, the cal_count was essentially shifted right by 1 thus dividing the result by 2; also, the shifted value will only be transferred to the raw delay register only once per lane since the cal_done is checked on a lane by lane bases. If cal_count[7] is set, a counter value overflow error will be indicated by LED[6], LED[2:0] will indicate which DQS/DQ lanes were calibrated properly and which were not, and cal_state will be incremented to 3 imediately. Else, if all 3 lanes were calibrated properly without an overflow error, cal_state will be incremented from 2 to 3, cal_done[2:0] equaling 0b111 will be transferred to LED[2:0], LED[5:3] will be off, and LED[7] will be on indicating a successfully completed calibration on the falling edge of the functional clock. In cal_state 3, dqs_cntl is set to 0 to tri-state the DQS bus, cal_count is reset to 0, and LED[5:0] are able to be assigned to read each raw delay registers seen in Table 4 above. Left button for raw_delay_2[5:0], center button for raw_delay_1[5:0], and right button for raw_delay_0[5:0] which corresponds to delays for each DQS/DQ pair.

Once calibration has been completed on one FPGA, the same steps were repeated on the other FPGA. Both FPGAs were then put into calculate calibrated result mode by setting switch[7:0] to equal 0x10. In this mode, dqs_cntl and dq_cntl are turned off putting the DQS bus and DQ bus into tri-state as indicated in Table 3 above. As indicated in Table 4, LED[4] comes on to indicate calculation ready after which the center button was depressed to load calc_delay_2[3:0] register with the value of raw_delay_0[5:0] minus raw_delay_2[5:0] and calc_delay_1[3:0] register with raw_delay_0[5:0] minus

raw_delay_1[5:0]. After the calculations were completed and calculated values stored in calc_delay_2[3:0] and calc_delay_1[3:0] for both FPGAs, both FPGAs were put into display calculated value mode with switch[7:0] equal 0x20. LED[7:6] was set to 0b10 to indicate that display mode was ready and each push button are able be depressed to display the desired register values on LED[5:0]. Note that for calc_delay_2[3:0] register (up button) and calc_delay_1[3:0] register (down button), LED[5:4] are padded to 0b00 for proper indication while raw_delay_2[5:0] (left button), raw_delay_1[5:0] (center button), and raw_delay_0[5:0] (right button) were used directly, as indicated in Table 4. By taking the mode of 3 calibration cycles for each FPGA, it was observed that the value for raw_delay_2[5:0] = 7, raw_delay_1[5:0] = 11, and raw_delay_0[5:0] = 16 resulting in the calculated value for calc_delay_2[3:0] = 9 and calc_delay_1[3:0] = 5.

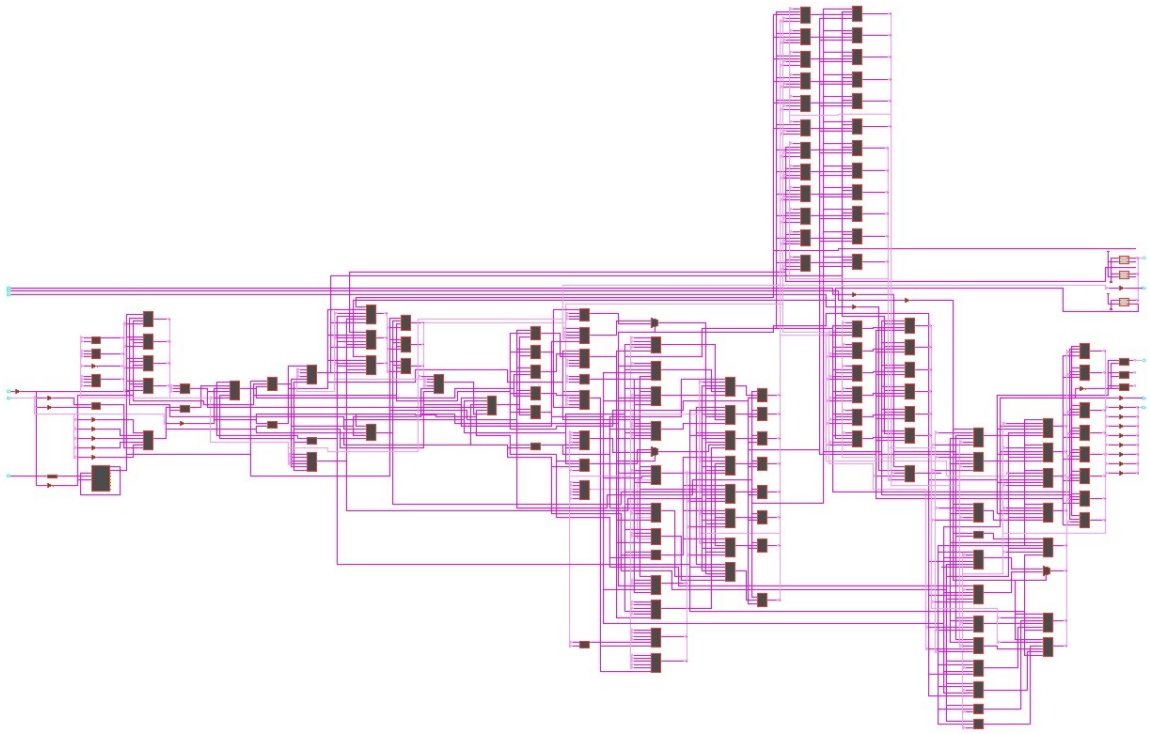

Figure 18: Calibrate send mode block diagram.

A block diagram of the calibrate (send) mode is provided in Figure 18 above. Calibrate (receive) mode block diagram is provided in Figure 19 below. Both diagrams were generated based on its respective verilog code section from Xilinx's PlanAhead v13.4 schematic generation tool.

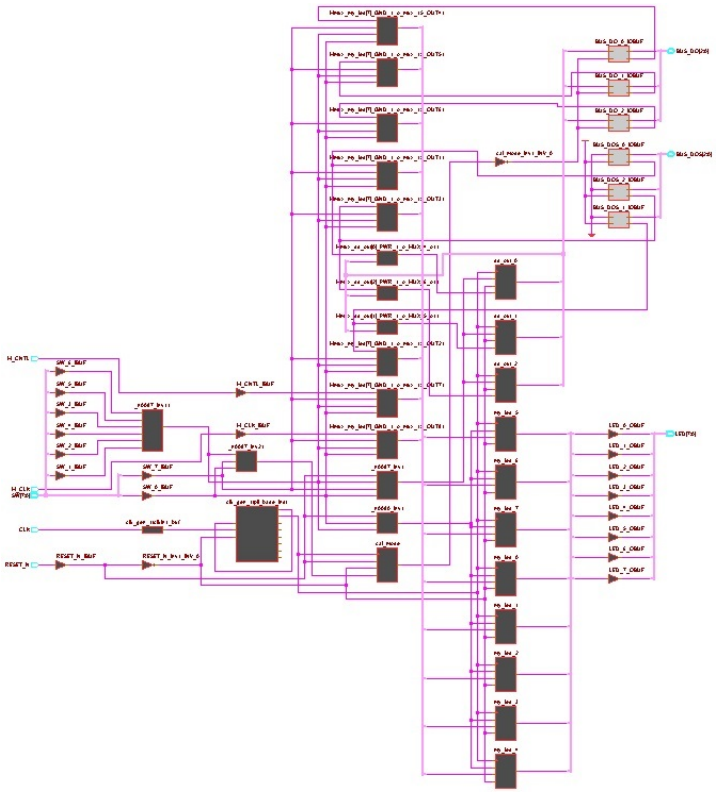

Figure 19: Calibrate receive mode block diagram.

3.2.4 Operation Mode

The operation mode consists of two sub-modes: receive data and send data. The receive data mode, switch[7:0] equal 0x40, and send data mode, switch[7:0] equal 0x80, were used in conjunction with each other to facilitate the sending of data from master FPGA to slave FPGA so that o-scope shots can be taken to evaluate the calibration results. To properly operate this mode, one FPGA was places into receive data mode

while the other into send date mode, in that order. The slave FGPA in receive data mode had its DQS and DQ bus control signal set low to input as seen in Table 5 below.

Table 5: Input and output bus/control for operation mode.

	Mode=Switch[7:0]	ReceiveData=0x40	SendData=0x80
I/O Bus	BUS_DQS[2:0]	Input(dqs_in[2:0])	Output(dqs_out[2:0])
	BUS_DQ[2:0]	Input(dq_in[2:0])	Output(dq_out[2:0])
Bus Cntl	dqs_cntl	LOW	HIGH
	dq_cntl	LOW	HIGH

The slave FGPA in receive data mode will indicate static off for all LED[7:0] and does not make use of any buttons. The master FGPA in send data mode had its DQS and DQ bus control signal set high to output as seen in Table 5 above. The master FGPA in send data mode will indicate static on for LED[7:4] and static off for LED[3:0]. To access three different data pattern across 4 beats of data strobe, the left, center, and right button can be depressed as indicated in Table 6 below.

Table 6: Switch, button, and LED assignment for operation mode.

	Mode=Switch[7:0]	ReceiveData=0x40	SendData=0x80
Indication(output)	LED[0]	Static OFF	Static OFF
	LED[1]	Static OFF	Static OFF
	LED[2]	Static OFF	Static OFF
	LED[3]	Static OFF	Static OFF
	LED[4]	Static OFF	Static ON
	LED[5]	Static OFF	Static ON
	LED[6]	Static OFF	Static ON
	LED[7]	Static OFF	Static ON
Function(input)	Button[Up]	Not Used	Not Used
	Button[Left]	Not Used	Send Pattern 1
	Button[Center]	Not Used	Send Pattern 2
	Button[Right]	Not Used	Send Pattern 3
	Button[Down]	Not Used	Not Used

Each press of the left, center, or right button will cause the strobe to be sent at the 31.25MHz functional clock frequency across 2 cycles as seen in Figure 20 below.

Figure 20: Operational mode scope shot of DQS0 and DQ0.

From Figure 20 above and Figure 21 below, taken with the infinite persistence display feature of the oscilloscope, it is noted that the DQ signal leads DQS by 4 phase rotator ticks or 90˚ of the 31.25MHz functional clock. Additionally, both the DQS and DQ of each respective wire lane was delayed by the calculated amount determined in Section 3.2.3 above, where DQS0/DQ0 was delayed by 0 phase rotator ticks (0ns), DQS1/DQ1 was delayed by calc_delay_1[3:0] = 5 phase rotator ticks (10ns), and DQS2/DQ2 was delayed by calc_delay_2[3:0] = 9 phase rotator ticks (18ns). The nano-second (ns) delay values were calculated based on 1 phase rotator tick equaling 1/16 of a 500MHz clock as previously described in Section 2.3 above. Also, the DQ was allowed to be sampled at both the rising and falling edge of DQS which allows for sampling data at twice the functional frequency. Setup and hold time of DQS to DQ, either on rising or falling edge of data strobe, was roughly 50% setup and 50% hold as measured in Figure 21 below.

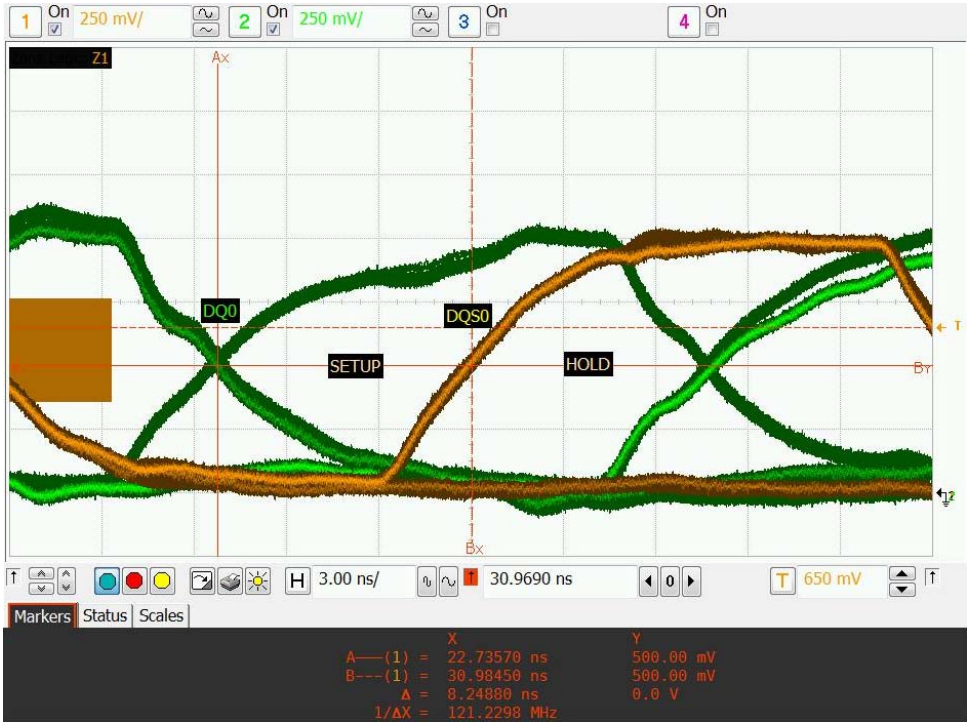

Figure 21: Operational mode scope shot of DQS0 and DQ0 zoomed in.

Chapter 4: Data Collection, Analysis, and Results

The oscilloscope was the primary method used to collect data for analysis. The initial measurement was to measure propagation delay time of each wire lane using the wire test modes. From Figures 22, 23, and 24 below, propagation delay for DQS0 on the long wire was measured at 17.333ns, DQS1 on the medium wire was 8.889ns, and DQS2 on the short wire was 1.111ns. Since the experiment was to have one wire delayed at approximately half, one at a quarter, and one edge aligned with the functional clock the measured value for each lane can be divided by 32ns to yield 54.2% for DQS0, 27.8% for DQS1, and 3.5% for DQS2 which was relatively close to what was required in terms of experiment setup.

Figure 22: Propagation delay of DQS0 on the long wire.

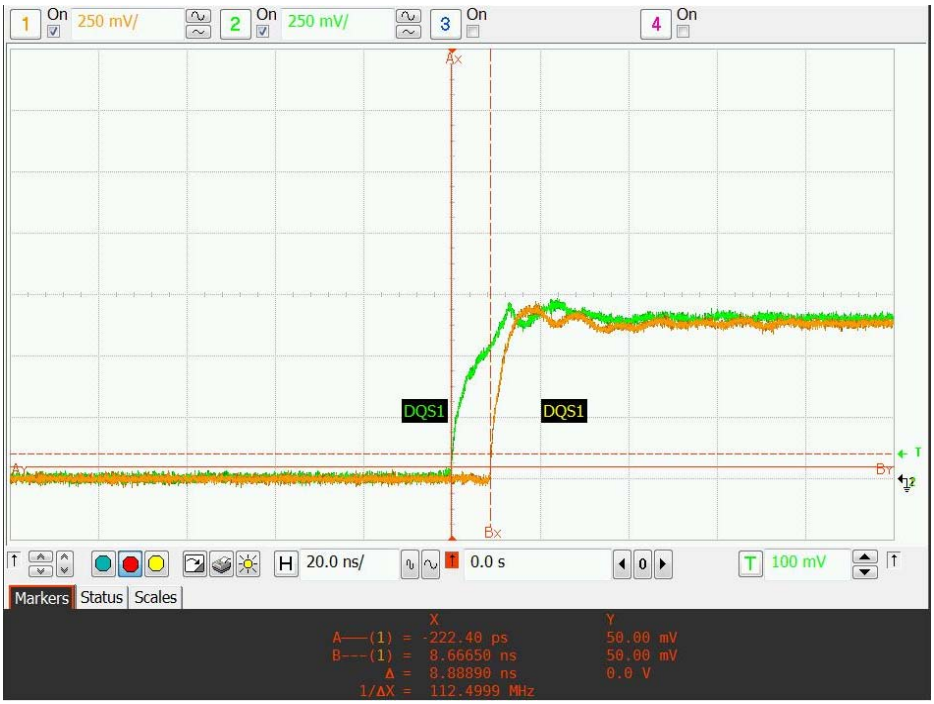

Figure 23: Propagation delay of DQS1 on the medium wire.

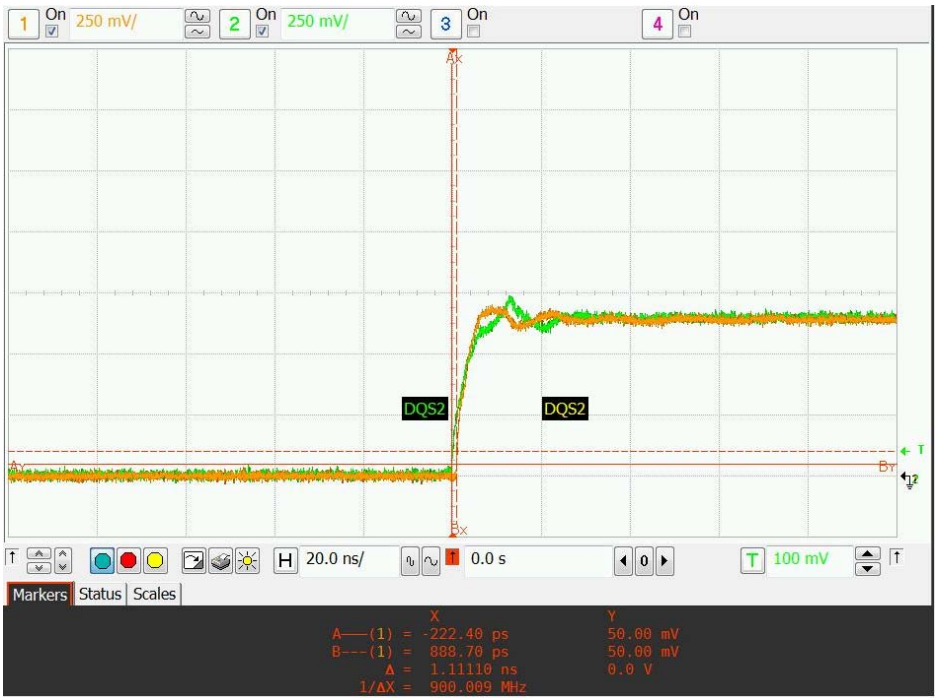

Figure 24: Propagation delay of DQS2 on the short wire.

The next measurement was to take the delta delay of DQS0 on the long wire minus the delay of DQS2 on the short wire to obtain the simulated phase rotator offset for DQS2 and DQ2. Based on the measurement taken from Figure 25 below, the offset delay value of 16.778ns was obtained which translates to roughly 8.4 ticks since each tick of the simulated phase rotator was worth 2ns from the 500MHz clock. Compared to the calibrated result stored in calc_delay_2[3:0] with the value of 9 ticks, the result was within the one tick granularity of the simulated phase rotator.

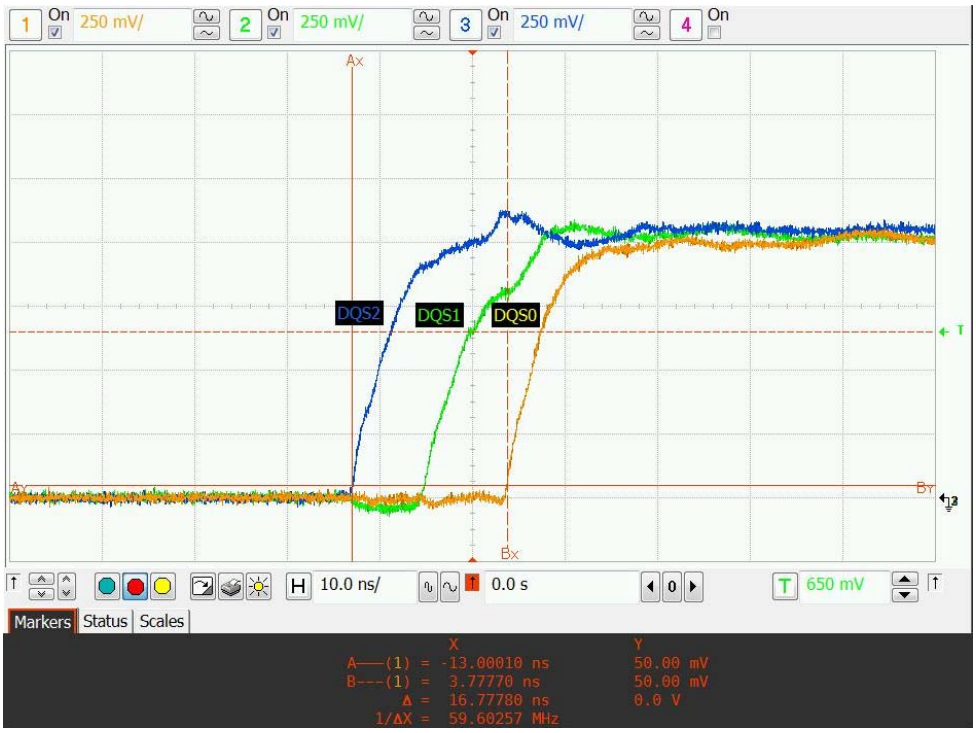

Figure 25: Measurement of expected delay offset for DQS2.

Taking the measurement from Figure 26 below, the delta delay of DQS1 on the medium wire minus the delay of DQS2 on the short wire gives 7.778ns. Subtracting the value of DQS1 minus DQS2 from the value of DQS0 minus DQS1, the value of DQS0 minus DQS1 can be obtained to be 9ns or 4.5 ticks. Compared to the calibrated result stored in

calc_delay_1[3:0] with the value of 5 ticks, this result was also within the one tick granularity of the simulated phase rotator.

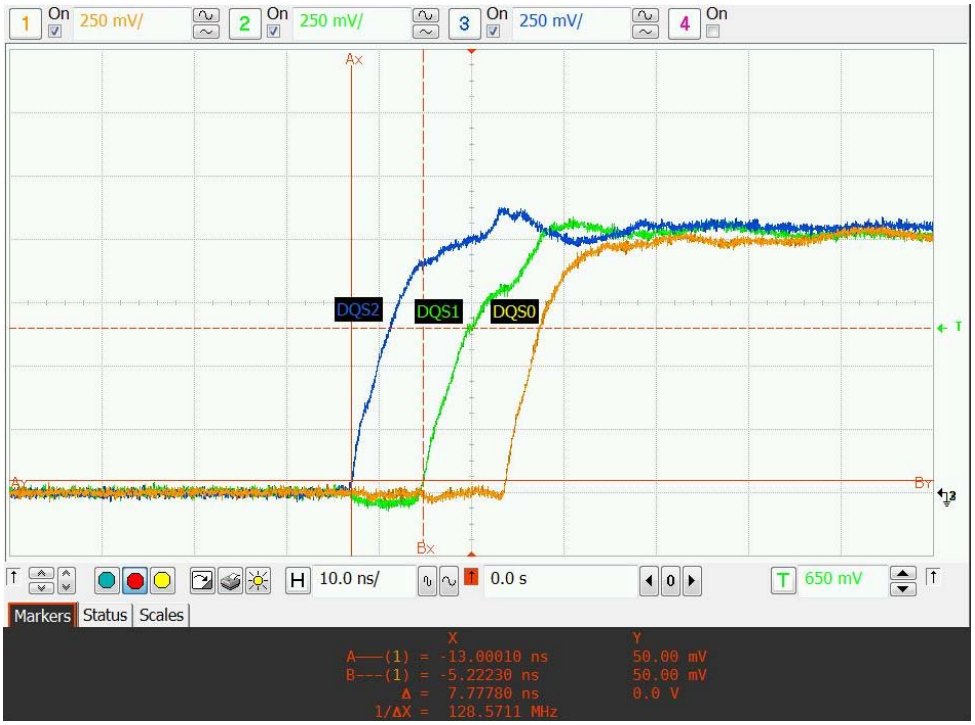

Figure 26: Measurement of expected delay offset for DQS1.

Another way to show the accuracy of the calibration technique was to take a before and after calibration scope shot in operations mode of all three DQS and do a comparison. Each data strobe sequence in operation mode was set to be four beats long, consisting of two rising DQS and two falling DQS edges across two full 31.25MHz functional clocks. This helps highlight the significant of the calibration result on the signal as seen in Figure 27 versus Figure 28 below.

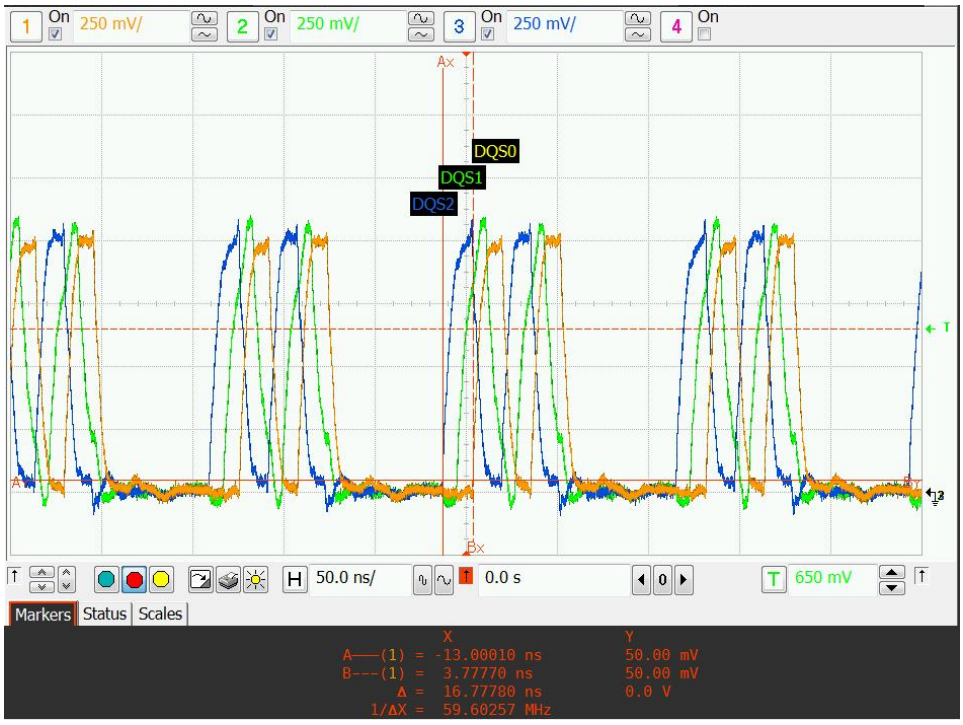

Figure 27: Operation mode before calibration on all DQS.

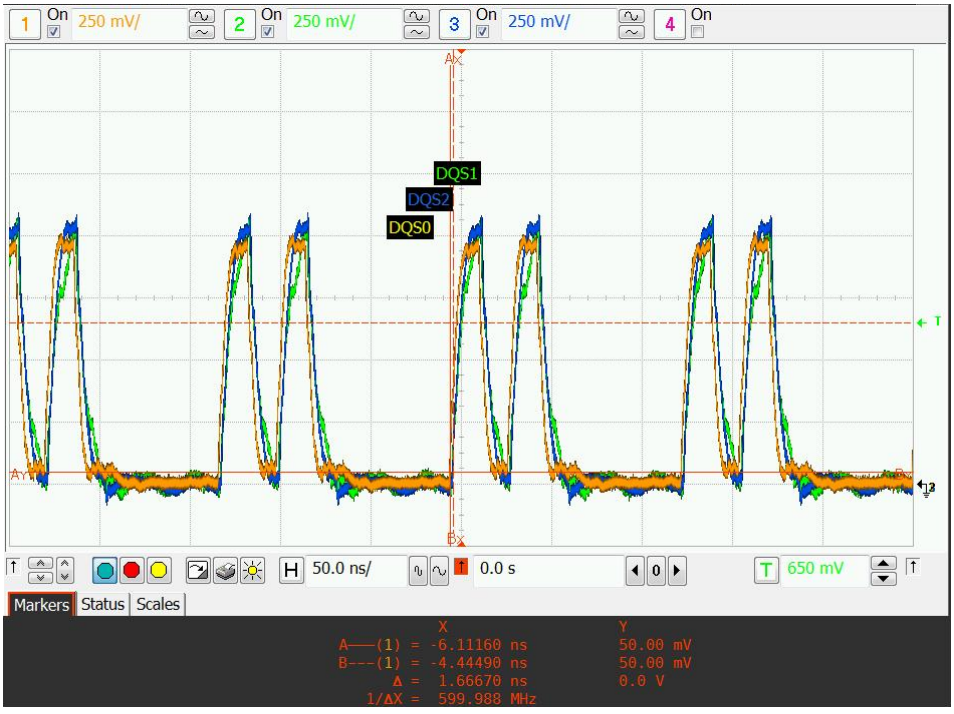

Figure 28: Operation mode after calibration on all DQS.

Measuring the largest delta of the first rising edge strobe in the sequence, we see that the worst delta before calibration to be over half a clock (16.778ns) and after calibration to be less than a sixteenth of a clock (1.667ns). Zooming in on the post calibration scope shot in Figure 29 below, it is noted that DQS0 seems to be about one tick earlier than DQS1 and DQS2.

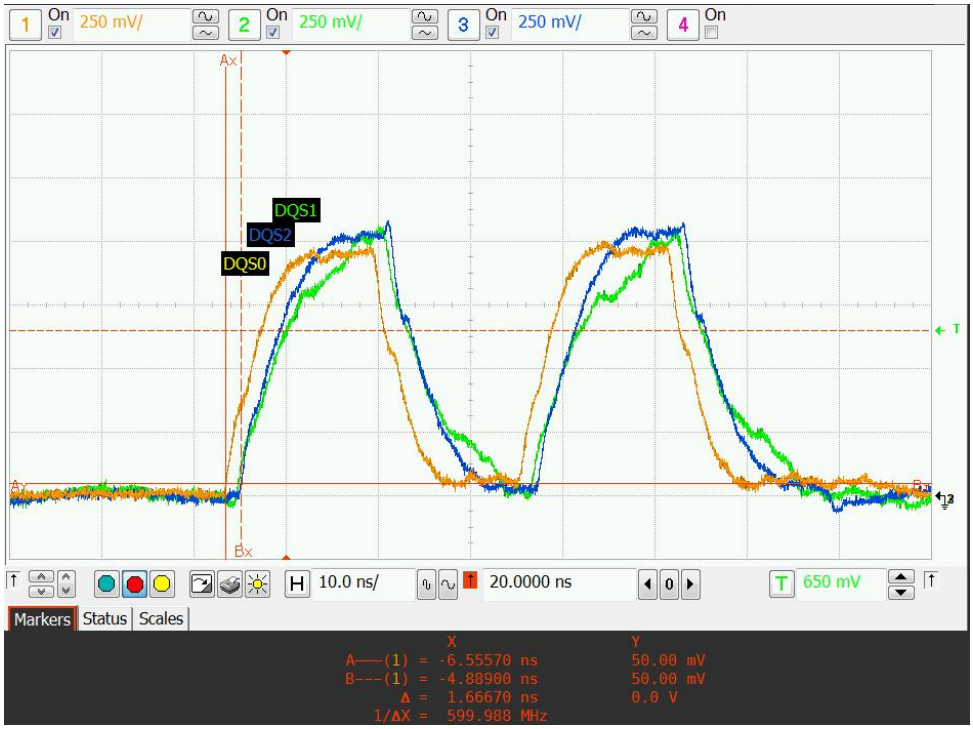

Figure 29: Measurement of largest DQS to DQS delay after calibration.

This can be explained as an aliasing of the calibration result from a combination of two possible factors. One is the rounding made during the divide by 2 in the calibration phase when the values from the cal_count[6:1] was transferred to each raw delay registers dropping the fractions in cal_count[0]. The other is the granularity of the phase rotator itself and its limitation to only be able to calibrate to the nearest 1/16 of one clock.

Based on these calibration analysis and results, it is apparent that by applying the digital calibration technique proposed in this report on global bidirectional interconnects of integrated circuits, overly restrictive routing rules and timing requirements can be relaxed. Variations in delay from one data lane group to another can be effectively calibrated out with the degree of accuracy roughly equal to the granularity of the selected phase rotator. Symmetric uniform global interconnects can be reconfigured with asymmetric non-uniform global interconnects, thus giving designers more freedom in routing.

Chapter 5: Conclusion

5.1 SUMMARY

The trend to integrate more processing cores and memory cores into a single module has increased the overall size of integrated circuits to the point where global interconnects between sub-units are becoming difficult to route and meet strict timing rules and requirements. New challenges has come forth in the design of global interconnect infrastructures for these large and highly integrated SoCs [3]. The traditional way of routing interconnects and the use of uniform point to point busses may no longer be optimal for certain designs where metal layers and chip area for interconnects are limited. Other solutions have also been developed to solve global interconnect design issues, most involves the use of complex technology and/or new hardware structure not widely available on current mass produced ICs. The need for a simpler, more flexible, routing methodology has become necessary and can be achieved by using calibration techniques currently being implemented at the board design level. The digital calibration technique applied in this report on a bidirectional interconnect has successfully synchronized multiple uneven data lanes running from one FPGA to another. It models sub-units within an IC, utilizing only very basic structures that are currently being implemented. During design space explorations for a particular system, the use of a non-uniform, bidirectional, global interconnects with data lane calibration technique may be the optimal solution.

5.2 FUTURE WORK

The digital calibration technique described and applied in this report concentrated on the timing aspects of propagation delay variation, data slew rate, and data skew differences between non-uniform data lane groups. A critical aspect not mentioned was

the voltage amplitude calibration technique. This technique, which can be used in conjunction with the digital timing calibration technique, seeks to optimize the voltage levels of a multi-load net through the capability of the driver/receiver circuit to adjust its variable pull-up resistances and would further improve the earlier proposed solution [7].

Bibliography

[1] Becker, Wiren Dale, et al. "Methods to self-synchronize clocks on multiple chips in a system." U.S. Patent No. 7,382,844. 3 Jun. 2008.

[2] Bernstein, Kerry, et al. "Interconnects in the third dimension: design challenges for 3D ICs." Proceedings of the 44th annual Design Automation Conference. ACM, 2007.

[3] Carloni, Luca P., Partha Pande, and Yuan Xie. "Networks-on-chip in emerging interconnect paradigms: Advantages and challenges." Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip. IEEE Computer Society, 2009.

[4] Champac, Victor, Victor Avendaño, and Joan Figueras. "Built-in sensor for signal integrity faults in digital interconnect signals." Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 18.2 (2010): 256-269.

[5] Chang, M-C. Frank, et al. "RF interconnects for communications on-chip." Proceedings of the 2008 international symposium on Physical design. ACM, 2008.

[6] Chen, Jonathan Y., Patrick J. Meaney, and William J. Scarpero Jr. "Digital system having a multiplicity of self-calibrating interfaces." U.S. Patent No. 6,934,867. 23 Aug. 2005.

[7] Cleitus, Antony, et al. "Bidirectional communication system and calibrator." U.S. Patent No. 7,643,954. 5 Jan. 2010.

[8] Cong, Jason, Kwok-Shing Leung, and Dian Zhou. "Performance-driven interconnect design based on distributed RC delay model." Design Automation, 1993. 30th Conference on. IEEE, 1993.

[9] Cordero, Edgar R., et al. "Implementing chip to chip calibration within a TSV stack." U.S. Patent No. 8,692,561. 8 Apr. 2014.

[10] Dally, William J., and Brian Towles. "Route packets, not wires: On-chip interconnection networks." Design Automation Conference, 2001. Proceedings. IEEE, 2001.

[11] Davis, Jeffrey A., et al. "Interconnect limits on gigascale integration (GSI) in the 21st century." Proceedings of the IEEE 89.3 (2001): 305-324.

[12] Floyd, Brian A., and Chih-Ming Hung. "Intra-chip wireless interconnect for clock distribution implemented with integrated antennas, receivers, and transmitters." Solid-State Circuits, IEEE Journal of 37.5 (2002): 543-552.

[13] Hajimiri, Ali. "Distributed integrated circuits: an alternative approach to high-frequency design." Communications Magazine, IEEE 40.2 (2002): 168-173.

[14] Hajimiri, Ali, et al. "Integrated phased array systems in silicon." Proceedings of the IEEE 93.9 (2005): 1637-1655.

[15] Hamdioui, Said, Georgi Gaydadjiev, and Ad J. Van de Goor. "The state-of-art and future trends in testing embedded memories." Memory Technology, Design and Testing, 2004. Records of the 2004 International Workshop on. IEEE, 2004.

[16] Ho, Ron, Kenneth W. Mai, and Mark A. Horowitz. "The future of wires." Proceedings of the IEEE 89.4 (2001): 490-504.

[17] Kırman, Nevin, et al. "On-chip optical technology in future bus-based multicore designs." IEEE micro 27.1 (2007): 56-66.

[18] Kuhn, Kelin J., et al. "Process technology variation." Electron Devices, IEEE Transactions on 58.8 (2011): 2197-2208.

[19] Loi, Igor, et al. "A low-overhead fault tolerance scheme for TSV-based 3D network on chip links." Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design. IEEE Press, 2008.

[20] Lorin, Harold, Leslie D. Ball, and Gilbert Eloy. "Interconnect technology as a management challenge." MIS Quarterly (1987): 433-435.

[21] Mizuno, Masayuki, William J. Dally, and Hideaki Onishi. "Elastic interconnects: Repeater-inserted long wiring capable of compressing and decompressing data." Solid-State Circuits Conference, 2001. Digest of Technical Papers. ISSCC. 2001 IEEE International. IEEE, 2001.

[22] Moore, Gordon E. "Cramming more components onto integrated circuits." Proceedings of the IEEE 86.1 (1998): 82-85.

[23] Nakahara, K., et al. "A novel three phase-states phase shifter." Microwave Symposium Digest, 1993., IEEE MTT-S International. IEEE, 1993.

[24] Owens, John D., et al. "Research challenges for on-chip interconnection networks." IEEE micro 27.5 (2007): 96.

[25] Peh, Li-Shiuan, and William J. Dally. "A delay model and speculative architecture for pipelined routers." High-Performance Computer Architecture, 2001. HPCA. The Seventh International Symposium on. IEEE, 2001.

[26] Shacham, Assaf, Keren Bergman, and Luca P. Carloni. "On the design of a photonic network-on-chip." Proceedings of the First International Symposium on Networks-on-Chip. IEEE Computer Society, 2007.

[27] Sidiropoulos, Stefanos. "Slave device with calibration signal generator for synchronous memory system." U.S. Patent No. 7,489,756. 10 Feb. 2009.

[28] Spencer, Todd J., Tyler Osborn, and Paul A. Kohl. "Materials science. High-frequency chip connections." Science (New York, NY) 320.5877 (2008): 756.

[29] Sylvester, Dennis, and Kurt Keutzer. "Getting to the bottom of deep submicron." Proceedings of the 1998 IEEE/ACM international conference on Computer-aided design. ACM, 1998.

[30] Wang, Hua, and Ali Hajimiri. "A wideband CMOS linear digital phase rotator." Custom Integrated Circuits Conference, 2007. CICC'07. IEEE. IEEE, 2007.

[31] "Atlys™ Board Reference Manual." Revision August 5, 2013, Digilent Inc., 2013.