

**High-Speed Low-Power Transmitter**  
**Using Tree-Type Serializer in 28-nm CMOS**

*Tongsung Kim*

**The Graduate School**  
**Yonsei University**  
**Department of Electrical and Electronic Engineering**

**High-Speed Low-Power Transmitter**  
**Using Tree-Type Serializer in 28-nm CMOS**

by

***Tongsung Kim***

Master's Thesis

Submitted to the Department of Electrical and Electronic Engineering  
and the Graduate School of Yonsei University  
in partial fulfillment of the requirements for the degree of

**Master of Science**

**June 2016**

This certifies that the master's thesis of Tongsung Kim is approved.

---

**Thesis Supervisor: Woo-Young Choi**

---

**Seong-Ook Jung**

---

**Duho Kim**

**The Graduate School**

**Yonsei University**

**August 2016**

# Table of Contents

<b>Table of Contents</b> .....	i
<b>List of Tables</b> .....	iii
<b>List of Figures</b> .....	iv
<b>Abstract</b> .....	x
<b>1. Introduction</b> .....	1
1.1. High-Speed Transmitter .....	1
1.2. Overview of Three Types of Serializer .....	4
1.3. Outline of Thesis .....	9
<b>2. Background &amp; Motivation</b> .....	10
2.1. Operation of Conventional 2-to-1 Multiplexer .....	10
2.2. Timing Constraints for Conventional Tree-Type Serializer ..	13
<b>3. Prototype Transmitter</b> .....	16
3.1. Multi-Phase Clock Using Tree-Type Serializer .....	16
3.2. Timing Constraint for Proposed Serializer .....	21
3.3. Phase Skew in Multi-Phase Clock Signals .....	26
3.2. Multi-Phase Divider and Phase-Aligner .....	31

3.2.1 Power Comparison between Conventional Divider and Multi-Phase Divider .....	31
3.2.2 Phase-Aligner for Multi-Phase Divider .....	34
<b>4. Implementation</b> .....	37
4.1. Overall Architecture .....	37
4.2. Latch-Less Serializer .....	38
4.3. Multi-Phase Divider .....	41
4.4. $2^{31}-1$ PRBS Pattern Generator .....	43
4.5. Voltage-Mode Output Driver .....	46
<b>5. Post-Layout Simulation Result</b> .....	51
<b>6. Summary</b> .....	60
<b>Bibliography</b> .....	61
<b>Abstract (In Korean)</b> .....	63
<b>List of Publications</b> .....	65

## List of Tables

<b>Table I .</b> Performance comparison with previous transmitters.....	59
---	----

## List of Figures

<b>Fig. 1.1.</b>	Block diagram of conventional transmitter for serial link. ....	3
<b>Fig. 1.2.</b>	Block diagram of shift register .....	6
<b>Fig. 1.3.</b>	Block diagram of large fan-in multiplexer .....	7
<b>Fig. 1.4.</b>	Block diagram of tree-type multiplexer .....	8
<b>Fig. 2.1.</b>	Conventional tree-type serializer topology .....	11
<b>Fig. 2.2.</b>	Operation of conventional 2-to-1 multiplexer. ....	12
<b>Fig. 2.3.</b>	Timing constraints for conventional tree-type serializer .....	15
<b>Fig. 3.1.</b>	Topology and timing diagram of the proposed tree-type serializer .....	19
<b>Fig. 3.2.</b>	16-to-1 proposed tree-type serializer topology .....	20
<b>Fig. 3.3.</b>	Timing constraint for proposed serializer .....	24
<b>Fig. 3.4.</b>	Setup time of D flip-flop, latch, and selector .....	25
<b>Fig. 3.5.</b>	Timing diagram of the last stage of serializer .....	28
<b>Fig. 3.6.</b>	Frequency divider topology and phase skew .....	29
<b>Fig. 3.7.</b>	Monte Carlo simulation of phase skew and buffering margin .....	30
<b>Fig. 3.8.</b>	Conventional divider and multi-phase divider.....	33
<b>Fig. 3.9.</b>	Phase aligner for multi-phase divider .....	36
<b>Fig. 4.1.</b>	Architecture of prototype transmitter .....	37
<b>Fig. 4.2.</b>	(a) conventional transmission-gate latch (b) clocked inverter latch.....	39

<b>Fig. 4.3.</b> Clocked inverter type latch simulation with frequency divider .....	40
<b>Fig. 4.4.</b> 4-phase latch-type frequency divider .....	42
<b>Fig. 4.5.</b> Schematic of series PRBS31 pattern generator.....	44
<b>Fig. 4.6.</b> Schematic of PRBS31 pattern generator.....	45
<b>Fig. 4.7.</b> Schematic of (a) a current-mode driver, and (b) a voltage- mode driver .....	48
<b>Fig. 4.8.</b> Operation of voltage-mode output driver.....	49
<b>Fig. 4.9.</b> Regulator, Replica driver, and VM driver topology.....	50
<b>Fig. 5.1.</b> Layout topology of prototype transmitter .....	53
<b>Fig. 5.2.</b> Eye diagram of serializer output data.....	54
<b>Fig. 5.3.</b> Energy-efficiency of serializer and clock distribution. ....	55
<b>Fig. 5.4.</b> Eye diagram of transmitter output data.....	56
<b>Fig. 5.5.</b> Energy-efficiency of transmitter .....	57
<b>Fig. 5.6.</b> Power comparison between conventional transmitter and proposed transmitter.....	58

## **Abstract**

# **High-Speed Low-Power Transmitter Using Tree-Type Serializer in 28nm CMOS**

*Tongsung Kim*

Dept. of Electrical and Electronic Engineering

The Graduate School

Yonsei University

Due to the increase in the demand for massive data in internet networking traffic, high-speed serial link I/O transceiver is essential. Since operating high speed data rate demands significantly large power, high speed and low power transceiver is desired.

In this thesis, a high-speed, low-power, and small-area transmitter is proposed with an optimized tree-type serializer. To achieve these targets, multi-phase clock signals are used to reduce latches in tree-type

serializer. Without extra cost, multi-phase clock generating dividers are implemented and a large number of latches are removed without performance degradation. With proposed tree-type serializer, voltage-mode driver is implemented instead of current-mode driver for low power consumption.

The prototype design is based on 28-nm CMOS technology. The prototype is operating at 25 Gb/s data rate while achieving low power and small area. The operation is verified with  $2^{31}-1$  PRBS pattern generator. Total power consumption at 25 Gb/s is 7.1mW.

---

**Keywords:** serializer, latch, multi-phase clock signals, voltage-mode driver

# **1. Introduction**

## **1.1. High-Speed Transmitter**

Internet networking traffic has been rapidly grown in recent years due to the explosive internet access, cloud computing, social media, etc. To transmit this massive data, there are two ways: parallel link, and serial link. In the parallel method, each data bit is transmitted through each channel, and all data should be transmitted at the same time. Even though this method is very simple, the number of I/O pin is very large. This results in big package costs due to large area and number of pins and produces problems such as data skew, clock skew, etc.

In the serial link, on the other hand, all the parallel data are converted to serial data, and this serial data is transmitted by one lane. Thus, problems mentioned in the parallel link can be solved, leading to this serial link method be widely used in diverse applications: PCI(Peripheral Component Interconnect) express, USB(Universal Serial Bus), SATA(Serial Advanced Technology Attachment), and HDMI(High-Definition Multi-media Interface). However, this method has design complexity due to muxing and demuxing process.

In the figure 1.1, block diagram of conventional transmitter is

described. Transmitter consists of serializer, clock source, frequency dividers, output driver, and PRBS pattern generator. For clock source, PLL(Phase-Locked Loop) is usually employed to make high frequency clock signals. This high frequency clock signals are then used for the last stage of serializer and frequency divider.

As can be seen in the figure, conventional transmitter receives parallel data with low speed data rate. Then these parallel data go to serializer to be converted into serial data. Each serializer needs sampling clock signals generated by frequency dividers for muxing process. After serialization, the data is aligned sequentially and has high speed data rate. Then, this data goes to output driver which has 50-ohm impedance matching.

With the increase in per-lane data rate, power and area burden for transmitter is a very big problem in serial link. Especially for high-speed data rate, serialization process demands high power consumption and large area, which should be solved.

## Transmitter

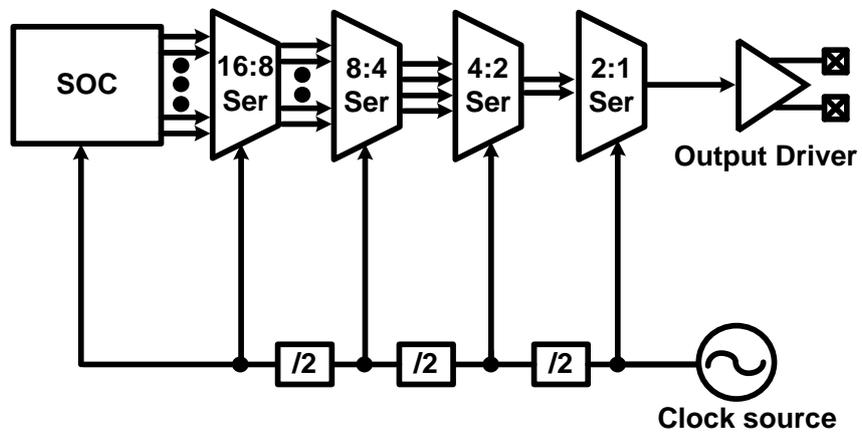


Figure 1.1: Block diagram of conventional transmitter for serial link

## 1.2. Overview of Conventional Serializers

Serialization process requires high power consumption and large area in high-speed data processing. For low power and small area design, several types of serializer have been developed and researched for a few decades: shift register [1], large fan-in multiplexer [2], and tree-type serializer [1].

The operation of shift register is described in Fig. 1.2. 2-to-1 multiplexer receives data, and moves these data to D flip-flops by pulse signal (when  $/N \text{ PULSE}=1$ ). Then, these data move to the next D flip-flop by full-rate clock signal (when  $/N \text{ PULSE}=0$ ). Though this method is straightforward, this block needs extreme high speed and low jitter clock signal for shifting input data. Thus, shift register consumes significantly large power [3]. Also, the maximum operating data rate is limited by device performance [3].

Large fan-in multiplexer serializes more than two input data by pulse signals generated by multi-phase clock signals as shown in Fig. 1.3. Since clock phase mismatch directly affects to output data jitter, each multi-phase clock should be well matched, which needs extra blocks [2], unless retiming D flip-flop exists at the last stage. Furthermore, this type of serializer has large parasitic capacitance at

output node and limits the operating speed [3], [4].

Tree-type serializer consists of multiple 2-to-1 multiplexers [1], which consist of five latches and selector as shown in Fig. 1.4. Each stage of multiplexers operates with clocks as fast as input data rate of multiplexer. Thus, front and middle parts of the serializer utilize low speed frequency clock signals, leading to small amount of power which is less than that of shift register. Also, each multiplexer has less parasitic capacitance at output node compared to large fan-in multiplexer, which is more suitable to high-speed design [4]. Even when analyzing timing constraints, it can be derived that tree-type serializer has more timing margin (2UI) than shift register (1UI) or large fan-in multiplexer (1UI). In conclusion, tree-type serializer is appropriate to high-speed and low-power serialization [4].

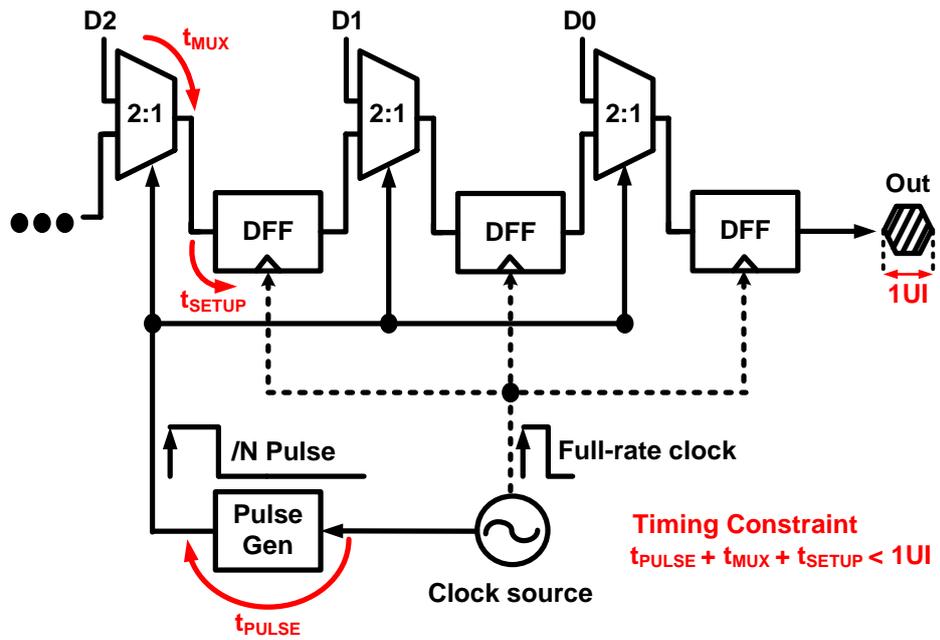


Figure 1.2: Block diagram of shift register

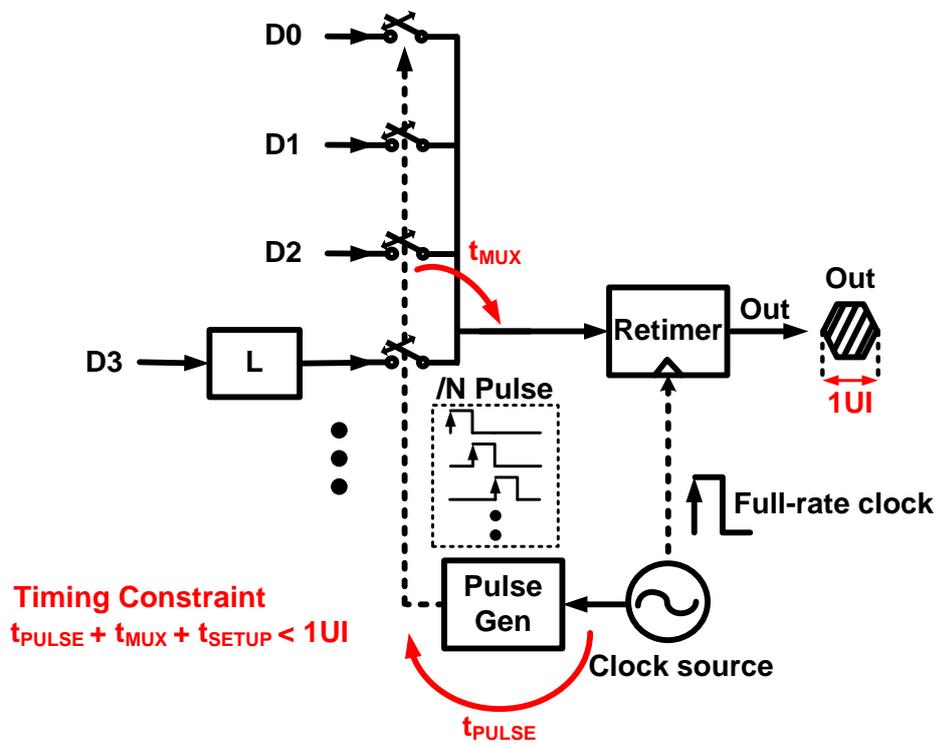


Figure 1.3: Block diagram of large fan-in multiplexer

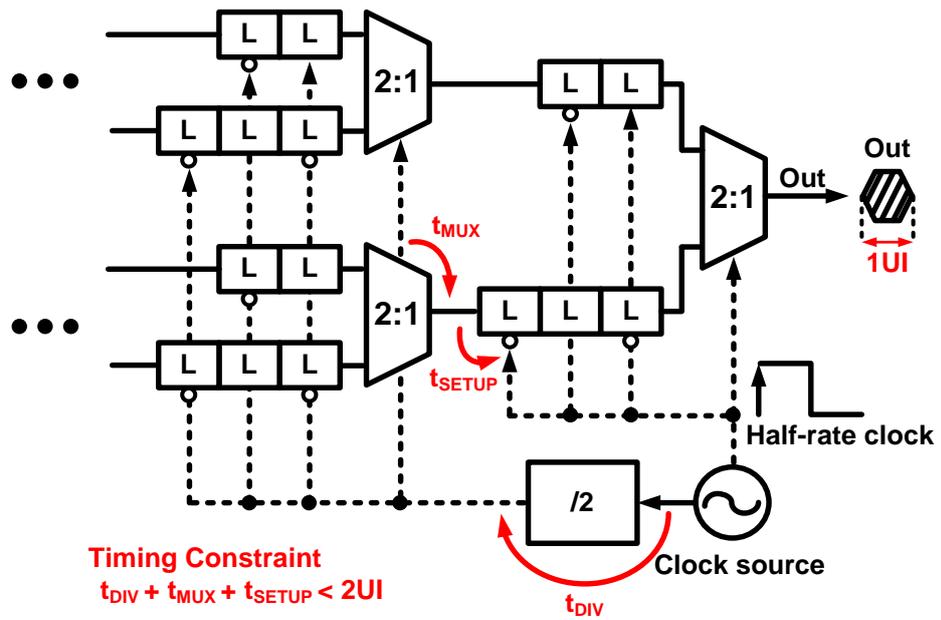


Figure 1.4: Block diagram of tree-type multiplexer

### **1.3. Outline of Thesis**

This thesis focuses on low-power and small-area design for transmitter with tree-type serializer. For this purpose, tree-type serializer with multi-phase clock signals is proposed and voltage-mode output driver is adopted. The proposed serializer with multi-phase clock signals reduces significantly large amount of power and area compared to conventional tree-type serializer. Also, voltage-mode output driver consumes lower power than current-mode output driver.

In chapter 2, timing constraints for conventional serializers and usage for latches in serializer will be explained. Then the operational principle and timing analysis of the proposed serializer and multi-phase frequency divider with divider initializer will be introduced in chapter 3. In chapter 4, the detailed schematic-level circuits for the prototype transmitter will be described. In chapter 5, post-layout simulation results will be shown. Finally, in chapter 6, summary will be given.

## **2. Background & Motivation**

### **2.1. Operation of Conventional Tree-Type Serializer**

Tree type serializer consists of multiple 2-to-1 multiplexers as shown in the figure 2.1. The idea is to group the input data in pairs and multiplex each pair, reducing the number by a factor of 2 after each stage [1].

The conventional 2-to-1 multiplexer consists of five-latch and selector [1]. This structure combines two parallel input data into serial output data stream. For a selector, its role is to enable one input data path and disable the other at the same time, and this operation is controlled by clock signals. To operate selector, thus, differential clock phases( $0^\circ$  and  $180^\circ$ ) are needed as each clock phase controls each data path.

When it comes to latches, each five latch has its role: retiming and phase-shifting data. Retiming latch functions as synchronizing incoming data with sampling clock to make them aligned. As shown in the figure 2.2, L1 and L2 latches align data A with clock signal, and L4 and L5 latches align data B with clock signal. By synchronizing incoming data with clock signals, data skew is removed by sampling

process.

On the other hand, the role of phase-shifting latch is to delay one input data to make half clock period offset with respect to the other data. In the figure 2.2, latch L3 shifts output data of latch L2, resulting half clock period offset with respect to B`. Then, phase shifted data A` is aligned with clock 0° signal, while data B` is aligned with clock 180° signal. After aligning each data with each clock signal, output data can be serialized in the selector without glitch problem.

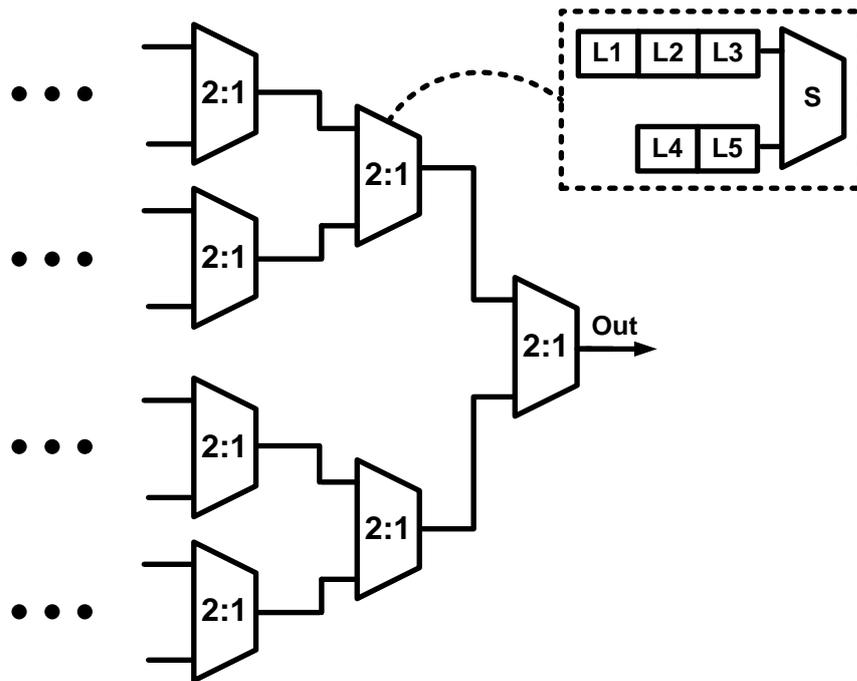


Figure 2.1: Conventional tree-type serializer topology

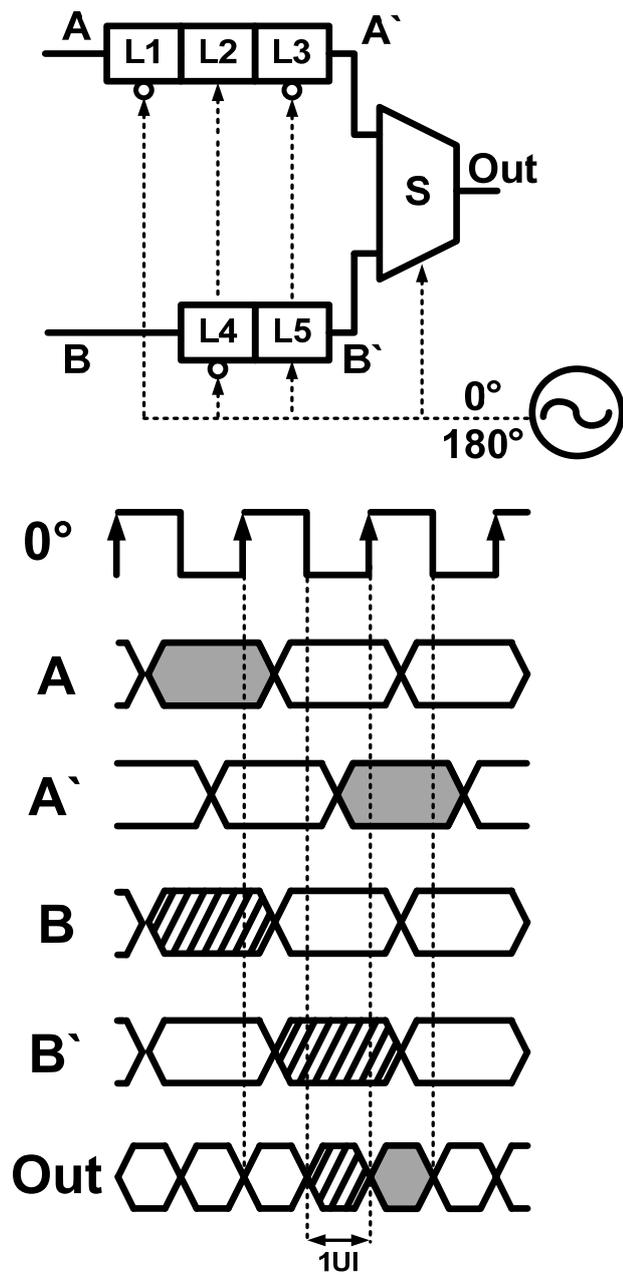


Figure 2.2: Operation of conventional 2-to-1 multiplexer

## 2.2. Timing Constraints for Conventional Tree-Type Serializer

Timing analysis for conventional tree-type serializer is shown in the figure 2.3. Since the timing margin is the least at the last stage in tree-type serializer, critical timing is placed at the last stage of serializer.

In the figure, there are pathA and pathB. The operation of pathA is as follows.  $CK_{/2}$  is divided into  $CK_{/4}$ , and  $CK_{/4}$  generates data stream (at node A), then this data stream is sampled by  $CK_{/2}$ . Thus, the clock period of  $CK_{/2}$  is timing margin of pathA. This is described as follows.

$$\text{PathA: } t_{\text{DIV}} + t_{\text{MUX}} + t_{\text{SU(DF)}} < 4UI \quad (2.1)$$

$t_{\text{DIV}}$  is clock frequency dividing delay,  $t_{\text{MUX}}$  is propagation delay at the selector,  $t_{\text{SU(DF)}}$  is setup time for D flip-flop, and  $1UI$  is output data window of the last stage.

On the other hand, timing constraint of pathB is as follows.

$$\text{PathB: } t_{\text{DIV}} + t_{\text{MUX}} + t_{\text{SU(DF)}} < 2UI \quad (2.2)$$

Timing margin is reduced to half of pathA. That is, pathB is more critical than pathA.

This implies that in the  $N:1$  tree-type serializer, the important timing constraint is placed at the last stage, so we do not have to concern other stages except the last one. Thus, to reduce power and

area of the conventional tree-type serializer, we can sacrifice timing margins of all the multiplexers except the last stage, which is a concept of our proposed design.

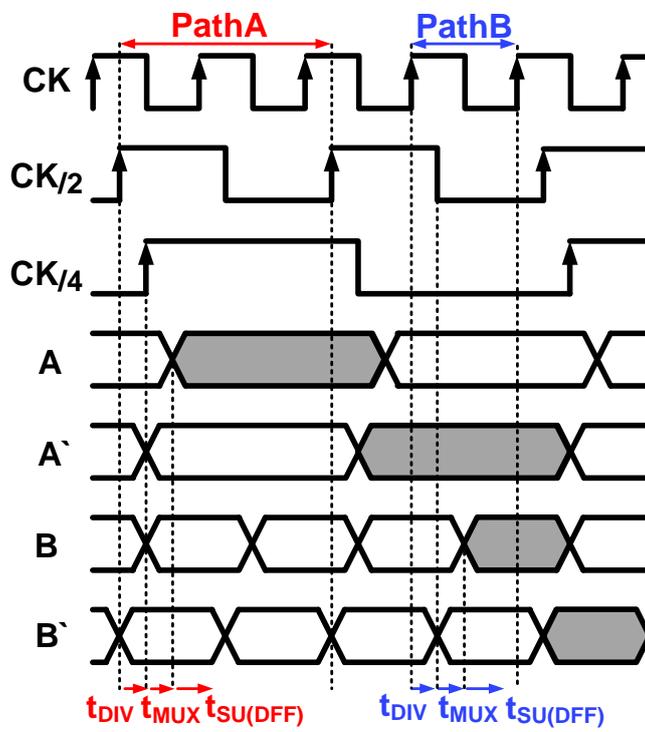
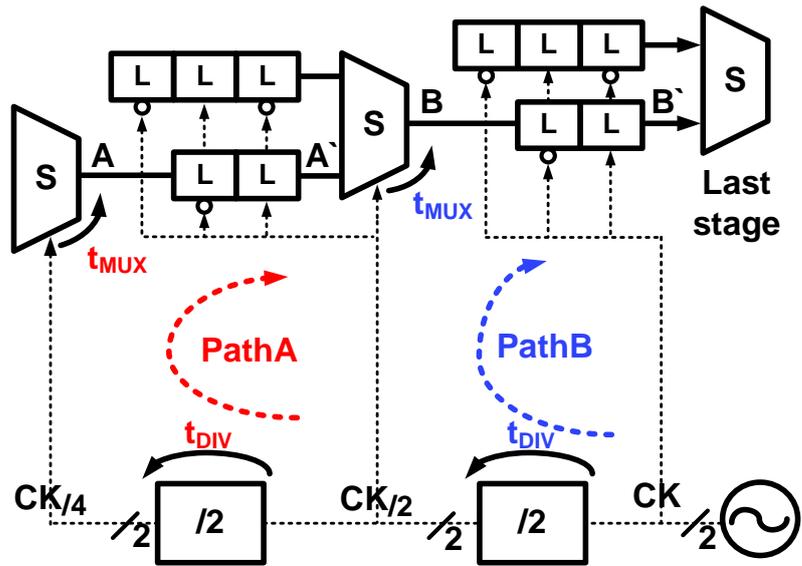


Figure 2.3: Timing constraints for conventional tree-type serializer

### 3. Proposed Serializer

#### 3.1. Multi-Phase Clock Using Tree-Type Serializer

As mentioned in previous chapter, conventional tree-type serializer consists of multiple 2-to-1 multiplexers, which consist of five latches and a selector. When the number of serializing data increases, the area and the power of the serializer increase exponentially. Especially, the increase in the number of latches is a big burden to clock signals, which drives all latches and selectors. Thus, reducing latches is a way to reduce power and area of the serializer. However, since all the latches have their roles, retiming and phase-shifting data, it is very difficult to reduce latches in the serializer without performance degradation.

In the figure 3.1, the proposed serializer is shown. At first, a frequency divider ( $/2$ ) generates four phases of divided clock signals. With these clock signals, data bits are sampled at the selectors, and the selectors generate serialized data stream (at node A and B). Since data at node A is sampled by clock signal  $0^\circ/2$  and  $180^\circ/2$ , this data stream is aligned with clock signal  $0^\circ/2$  and  $180^\circ/2$ . On the other hand, data stream at node B is sampled by clock signal  $90^\circ/2$  and  $270^\circ/2$ , and data at node

B is aligned to these clock signals. Thus, data stream at node A and at node B have phase offset which is as same as the offset between divided clock signals  $0^\circ/2$  and  $90^\circ/2$ , and this offset is also as same as the offset between clock signals  $0^\circ$  and  $180^\circ$ . This offset, therefore, is as same as the timing delay that phase-shifting latch generates, which means that we can reduce phase-shifting latch in multiplexer.

Furthermore, selector itself can be used as a retimer [5], which can reduce retiming-latches. That is, without five latches, retiming and phase-shifting latches, 2-to-1 multiplexer is capable of converting parallel data into serial data stream. With this method, serializer can reduce large amount of power and area.

This method can be enlarged to higher number of parallel data serializing operation. In our proposed design, 16-to-1 serializer is described in the figure 3.2. It can be seen that each stage of multiplexer uses multi-phase clock signals: 16-to-8 multiplexers use 16 phase clock signals, 8-to-4 multiplexers use 8 phase clock signals, and 4-to-2 multiplexers use 4 phase clock signals. That is, two clock phases drive only one multiplexer. For example, in the 16-to-8 stage, there are 8 multiplexers and 16 phase clock signals, and two clock phases drive each multiplexer.

By using multi-phase clock signals, latches at the middle stage

multiplexers can be removed as can be seen in the figure 3.2. Only the first stage and the last stage of the multiplexers have latches. The latches of the first stage multiplexers are for data aligning, and those of the last stage multiplexer are for increasing timing margin. The details will be explained in the next chapter. This method can be enlarged to higher number of data serialization, and the power and area reduction will be much larger.

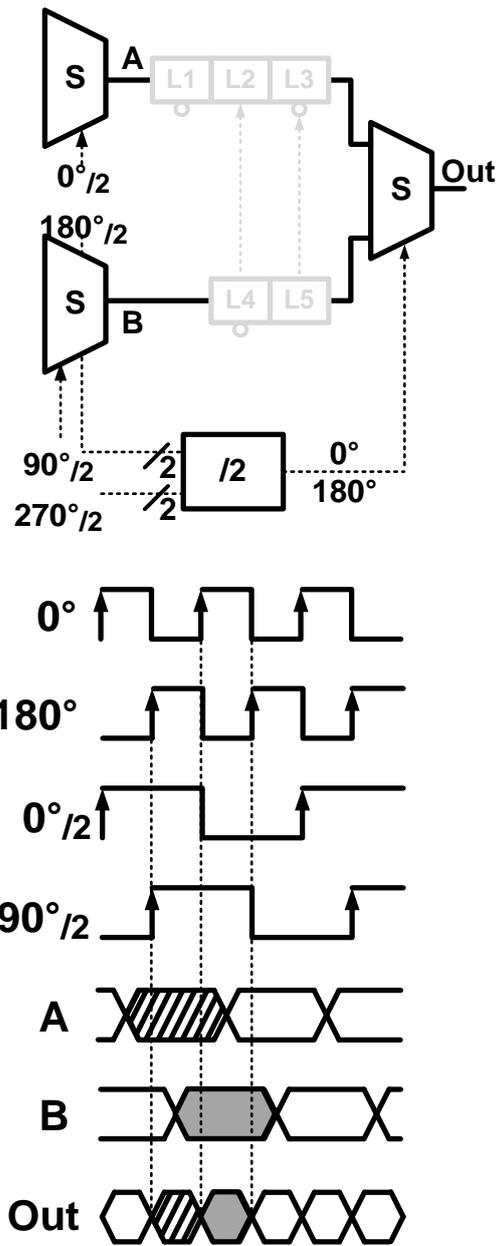


Figure 3.1: Topology and timing diagram of proposed tree-type serializer



### 3.2. Timing Constraint For Proposed Serializer

2-to-1 multiplexer without latches can operate as well as conventional 2-to-1 multiplexer with reducing power and area. However, as a trade-off, there is a timing margin reduction in multiplexer without latches, which is shown in the figure 3.3. PathA shows timing constraint of 2-to-1 multiplexer without latches.

The operation of pathA is as follows.  $CK_{/2}$  is divided into  $CK_{/4}$ , and  $CK_{/4}$  generates data stream (at node A), then this data stream is sampled at the selector by clock  $CK_{/2}$ . This is described as follows.

$$\text{PathA: } t_{\text{DIV}} + t_{\text{MUX}} < 2UI \quad (3.1)$$

$t_{\text{DIV}}$  is clock frequency dividing delay,  $t_{\text{MUX}}$  is propagation delay at the selector, and  $1UI$  is a half period of  $CK$  (=output data window of the last stage selector). Timing margin of pathA is  $2UI$ , while the timing margin of conventional serializer, Eq. (2.1), is  $4UI$ .

The differences between Eq. (3.1) and Eq. (2.1) are the existence of  $t_{\text{SU(DFD)}}$  (D flip-flop setup time) and the length of timing margin. First, when it comes to D flip-flop setup time, it is described in the figure 3.4. Sampling process of D flip-flop is as follows. First, master latch tracks the input data, and then holds this data. When master latch is holding data, slave latch tracks this data at the same time.

The timing margin for master latch to track input data is setup time for D flip-flop. On the other hand, since latch or selector track input data by themselves, these blocks need very small timing margin, and this timing margin is relatively negligible compared to setup time for D flip-flop. Thus, we omit setup time for latch or selector in our analysis.

Timing margin difference can be explained in the figure 3.3. When there is no latch between selectors, timing margin is from one rising time to the next falling time or from one falling time to the next rising time, which is a half of clock period ( $CK/2$ ). On the other hand, when there are more than 2 latches between selectors, shown in the figure 2.3, timing margin is from one rising time to the next rising time, which is 1 clock period ( $CK/2$ ). Thus, the margin in Eq. (2.1) is 2 times longer than that in Eq. (3.1).

In short, reducing latches in multiplexer suffers timing margin, leading to performance degradation. However, as mentioned in chapter 2.2, the most critical timing constraint in tree-type serializer is placed at the last stage. Thus, by relieving timing constraint of the last stage of serializer, we can avoid performance degradation. A way to relieve timing constraint is to place retiming latches as shown in the figure 3.3. In the figure, the last stage of serializer has retiming latches, and the timing constraint (pathB) is as follows.

$$\text{PathB}(\text{with latches}): t_{\text{DIV}} + t_{\text{MUX}} + t_{\text{SU(DF)}} < 2UI \quad (3.2)$$

This can be compared with the case when there is no retiming latch at the last stage of serializer as follows.

$$\text{PathB}(\text{without latches}): t_{\text{DIV}} + t_{\text{MUX}} < 1UI \quad (3.3)$$

The difference between Eq. (3.2) and (3.3) shows the effect of retiming latch, increasing timing margin. With retiming latches, we can compare critical timing of our proposed serializer, Eq. (3.2), with that of conventional serializer, Eq. (2.2). Then, it can be derived that critical timing constraint of our proposed serializer is as same as that of conventional serializer. That is, by having retiming latches only at the last stage of serializer, we can avoid performance degradation with eliminating a lot of latches in other stages. This leads to reducing large amount of power and area.

To compare power and area of our proposed serializer with conventional serializer, we can compare the number of latches and selectors. In conventional one, 75EA latches and 15 selectors are implemented in 16-to-1 serializer, while our proposed one has 18EA latches and 15 selectors. Therefore, it can be derived that a lot of power and area are saved by removing latches.

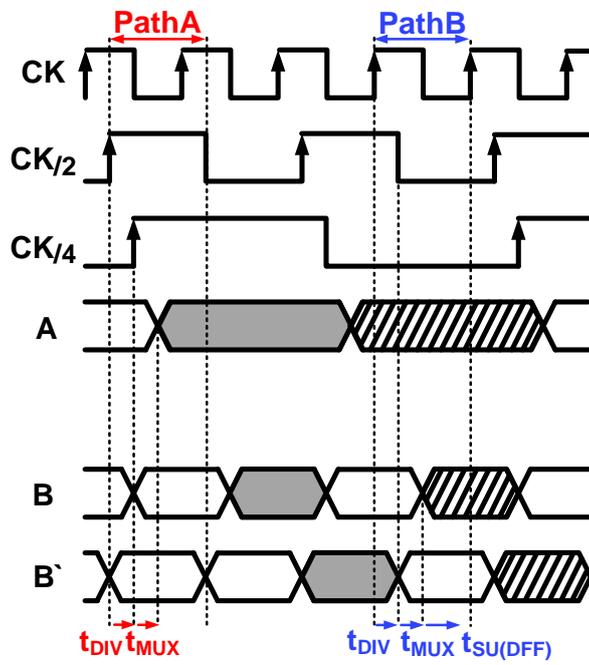
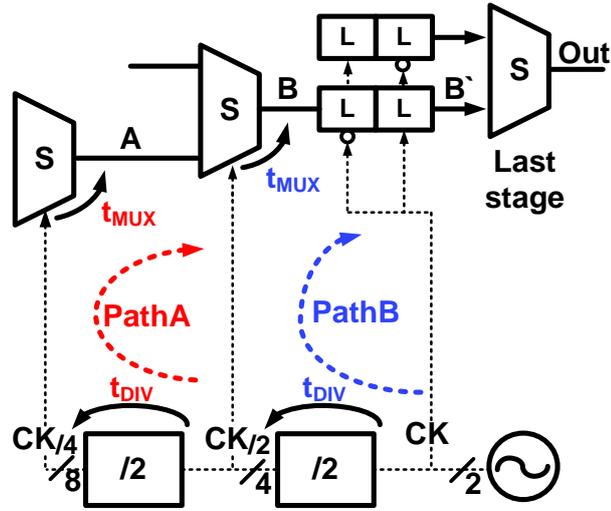


Figure 3.3: Timing constraint for proposed serializier

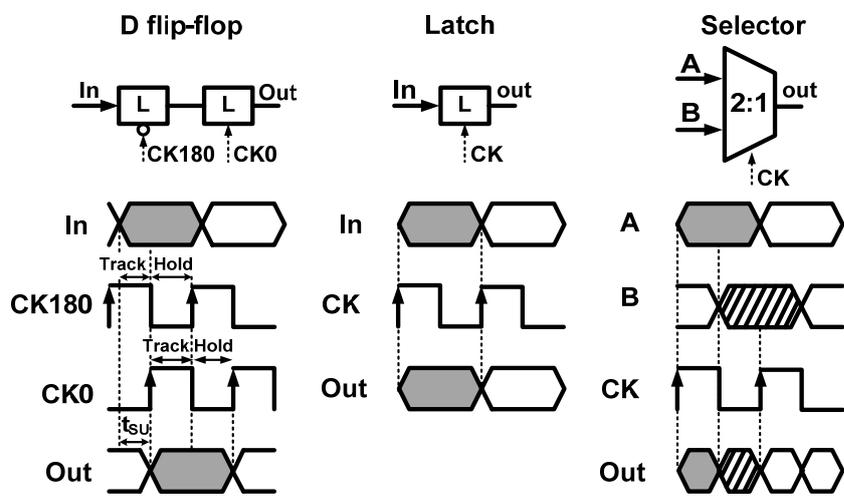


Figure 3.4: Setup time of D flip-flop, latch, and selector

### 3.3. Phase Skew In Multi-Phase Clock Signals

As explained previously, our proposed serializer uses multi-phase clock signals to reduce latches in serializer, and there are phase skews among multi-phase clock signals. Generally, phase skews in clock signals can generate output data jitter or influence timing constraint of serializer.

When it comes to output data jitter problem, this can be solved by characteristic of tree-type serializer. In the figure 3.5, data at node A has jitter due to phase skew of multi-phase clock signals, but this data is then sampled by clean differential clock signals at the last stage of serializer. That is, even though there are phase skews in multi-phase clock signals, tree-type serializer can avoid this by sampling data with differential clock signals at the last stage of serializer.

To see how phase skew influence timing constraint, the structure of frequency divider should be considered. In the figure 3.6, the frequency divider topology is described. To minimize power consumption of frequency divider, 4 phase clock signals are generated by using only 1 conventional latch-type frequency divider. As a trade-off, phase skew exists between  $0^\circ/2$  and  $180^\circ/2$  clock signals, which is 1 inverter delay. Also, phase skew between  $0^\circ/2$  and  $90^\circ/2$  exists due to

fan-out difference. These phase skews lead to decrease in critical timing margin.

However, there is other parameter that relieves critical timing margin. As explained previously, retiming latches are placed at the last stage of serializer to increase timing margin. To drive these latches, buffering inverter chains are essential, and delay from this inverter chain increases critical timing margin as shown in the figure 3.5. In the figure, timing diagram of the last stage of serializer is described. The parameter  $t_{\text{BUF}}$  (buffering inverter delay) increases timing margin in the timing diagram. Thus, the critical timing constraint of the proposed serializer, Eq. (3.2) can be modified as follows.

$$t_{\text{DIV}} + t_{\text{MUX}} + t_{\text{SU(DF)}} + t_{\text{SKEW}} < 2\text{UI} + t_{\text{BUF}} \quad (3.4)$$

$t_{\text{SKEW}}$ , clock phase skew, and  $t_{\text{BUF}}$  are added.

To compare  $t_{\text{SKEW}}$  and  $t_{\text{BUF}}$ , we did Monte Carlo simulation as shown in the figure 3.7. In the simulation, it can be derived that even in the worst case when phase skews are maximum and buffering delay (margin) is minimum,  $t_{\text{BUF}}$  is longer than  $t_{\text{SKEW}}$ . Thus, it can be seen that decrease in timing margin due to clock phase skew can be solved by buffering inverter chain that is essential for driving latches in the last stage of serializer.

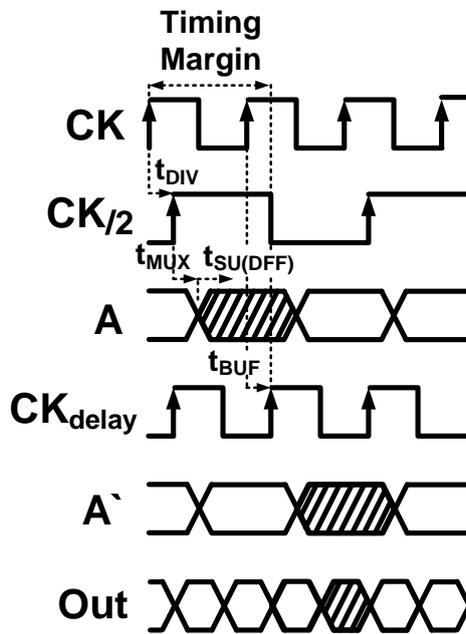
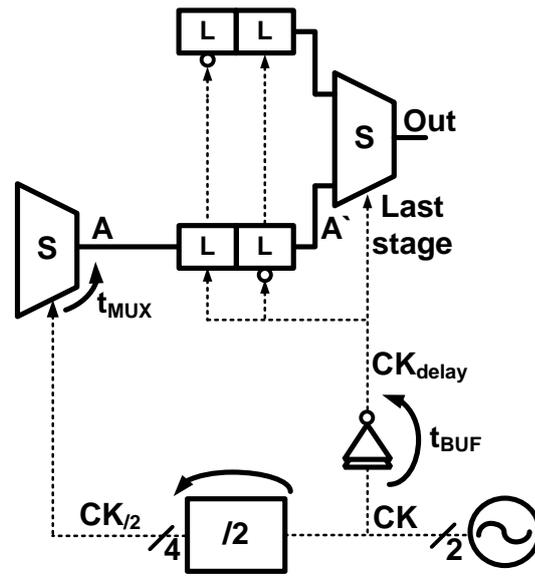


Figure 3.5: Timing diagram of the last stage of serializer

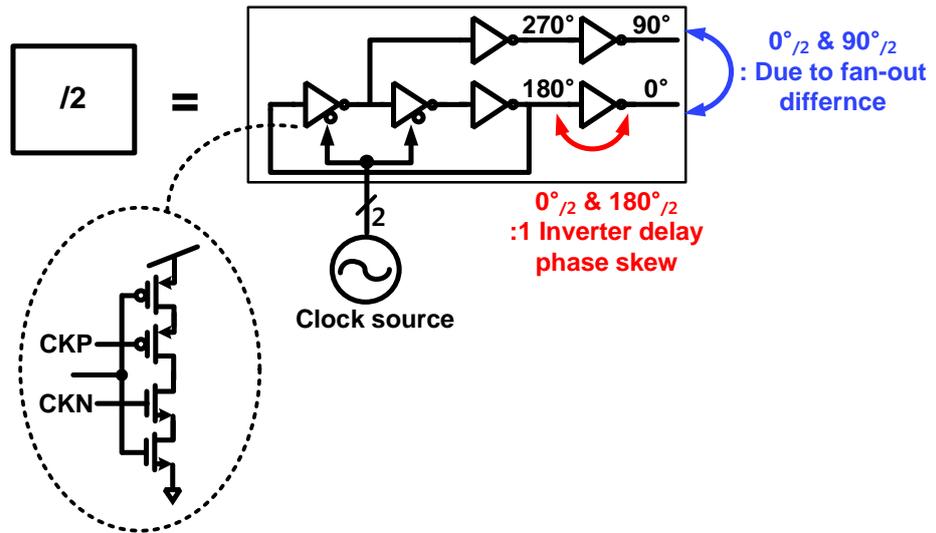


Figure 3.6: Frequency divider topology and phase skew

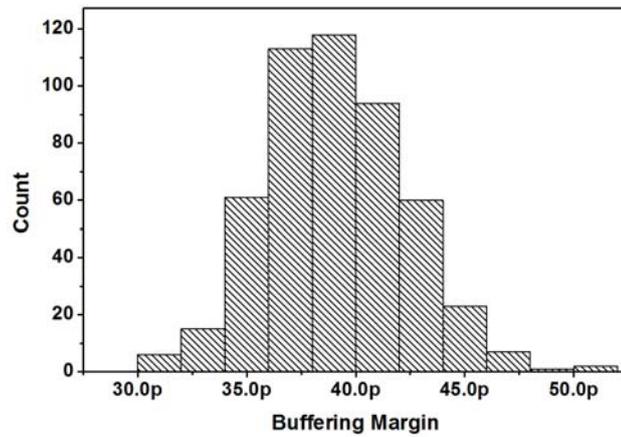
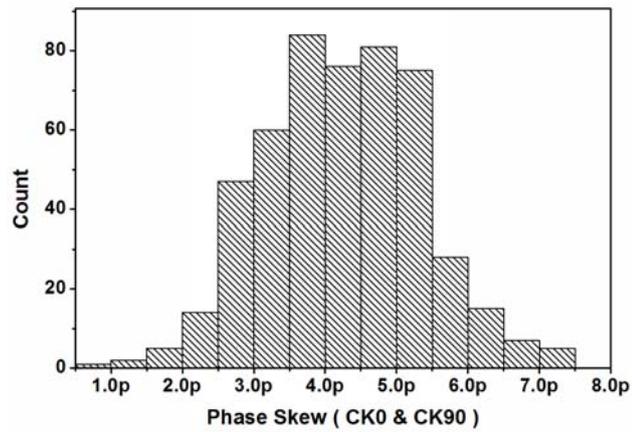
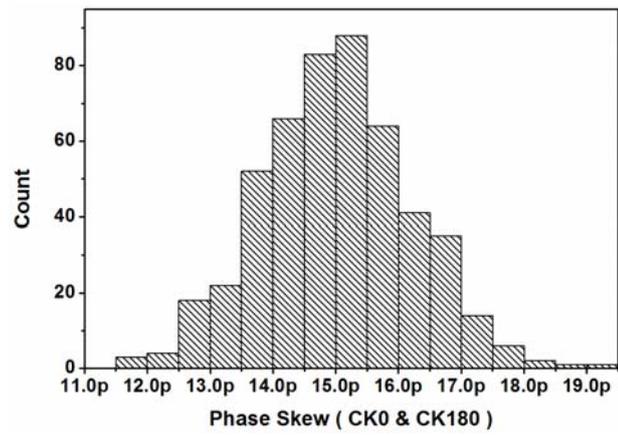


Figure 3.7: Monte Carlo simulation of phase skew and buffering margin

### **3.4. Multi-Phase Divider and Phase-Aligner**

#### **3.4.1. Power Comparison Between Conventional Divider and Multi-Phase Divider**

Our proposed tree-type serializer operates with multi-phase clock signals. Thus, multi-phase frequency divider should be designed not to consume large power compared to conventional divider. This is analyzed, and is described in the figure 3.8. The analysis assumes that the size of divider is determined by its driving blocks (multiplexers) without concerning clock tree. With this assumption, it can be seen that size of each conventional divider is large enough to drive many multiplexers. That is, 1 divider drives several multiplexers at any stage: 1 divider drives 8 multiplexers at 16-to-8 stage, 4 multiplexers at 8-to-4 stage, and 2 multiplexers at 4-to-2 stage.

On the other hand, in our proposed tree-type serializer, multi-phase clock signals drive 1 multiplexer. As can be seen in the figure, 1 divider drives 2 multiplexers at any stage.

Thus, the size of a divider of conventional tree-type serializer is 4 times larger than that of our proposed tree-type serializer at 16-to-8 stage, and the size of a divider of conventional one is 2 times larger

than that of our proposed one at 8-to-4 stage.

When comparing the total divider size, conventional dividers and multi-phase dividers are same. A 4x-size divider (conventional) equals to 4EA 1x-size dividers (multi-phase) at 16-to-8 stage, and a 2x-size divider (conventional) equals to 2EA 1x-size dividers (multi-phase) at 8-to-4 stage. Also both of conventional and multi-phase dividers drive same amount of multiplexers, which have same amount of parasitic capacitance. Since dynamic power is proportional to capacitance, total power consumption of both conventional and multi-phase dividers is same. Thus, there is no extra power consumption when we use multi-phase dividers instead of conventional dividers.

In addition, clock loading in proposed serializer is much less than that in conventional serializer due to reduced latches in proposed serializer. This factor comes with even smaller size of multi-phase divider. Therefore, with this analysis, multi-phase divider itself does not consume more power than conventional divider. Furthermore, with proposed serializer, more power and area can be saved.

This analysis is based on the assumption that divider size is determined by its driving blocks instead of concerning clock tree. If clock tree is concerned, power comparison is more complex. We have to concern divider type, latch type, reduced latches, etc. Furthermore,

we have to consider timing margin reduction due to having clock tree.

Thus, we made the assumption for simple power analysis.

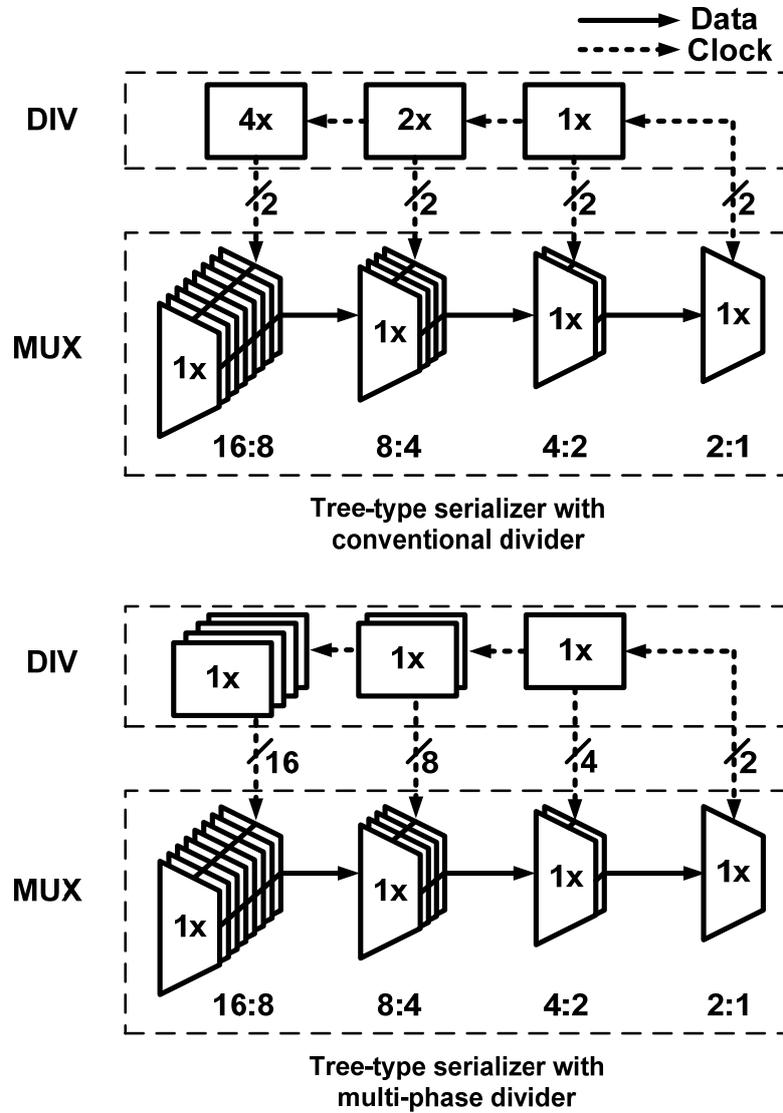


Figure 3.8: Conventional divider and multi-phase divider

### 3.4.2. Divider Initializer For Multi-Phase Divider

To make multi-phase clock signals, multi-phase divider is used. However, there is phase ambiguity problem especially in 8 phase generation or 16 phase generation. That is, 8-phase generating dividers and 16-phase generating dividers suffer from ambiguity in the phase relationship, which can lead to clock signal phases to be out of order. For example,  $45^\circ/4$  clock signal generating node can be inverted, which results in generating  $135^\circ/4$  clock signal. This problem is due to independent initial condition in each frequency divider.

To correct this ambiguity, divider initializer is essential to control initial condition which is shown in figure 3.9. The operation of the divider initializer is as follows. First, each block of 8-phase generating dividers should be triggered with 0 (trigger=0). In this state, every node of 8-phase generating dividers is fixed with initial value. That is, the value is not changing whether input clock signals are operating or not. Second, reset signal changes from 0 to 1. Then this reset value is sampled by  $90^\circ/2$  and  $270^\circ/2$  clock phases, and this sampled value goes to trigger signal. That is, trigger signal changes from 0 to 1. When trigger signal is 1, every node of 8-phase dividers begins to generate divided 8-phase clock signals. As shown in the

timing diagram,  $0^\circ/2$  clock signal samples trigger signal earlier than  $90^\circ/2$  clock signal. Thus, a 8-phase divider with having input clock phases  $0^\circ/2$  and  $180^\circ/2$  generates divided clock signal earlier than the other divider. That is, 8-phase divided clock signals are aligned in sequence. The divider initializer is also used in 16-phase dividers as well.

After reset signal changes from 0 to 1, this signal does not change. Thus, divider initializer's input signal does not change. Since D flip-flop in divider initializer consists of CMOS type latches and input does not change, the dynamic power is not consumed after setting initial condition.

Furthermore, lock detector from PLL can be used as a reset signal of this divider initializer, though PLL is not implemented in this design.

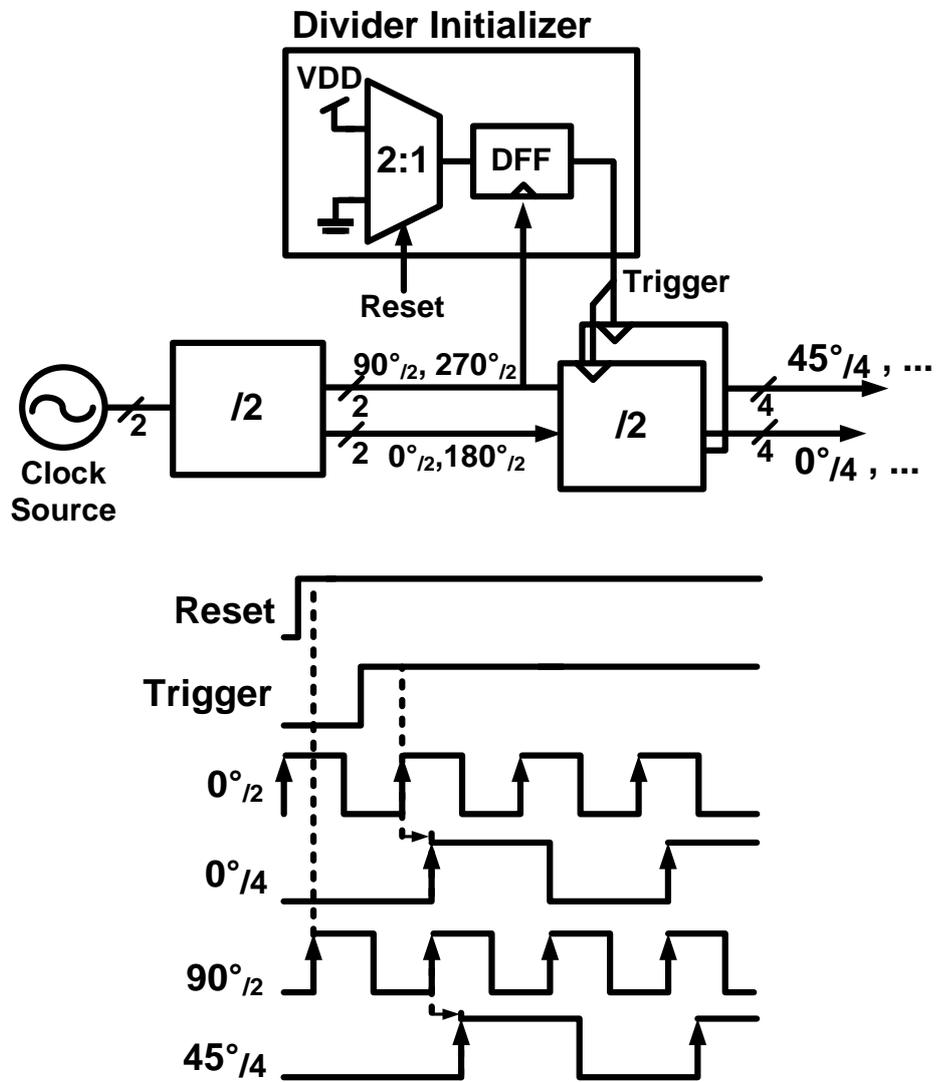


Figure 3.9: Phase aligner for multi-phase divider

## 4. Implementation

### 4.1. Overall Architecture

Figure 4.1 shows the block diagram of the prototype transmitter. For test purpose, 16 bit parallel PRBS31 pattern generator is implemented. Then, latches are used at the first multiplexer to align the incoming input data, and at the last multiplexer. For sampling clock signals, 3 stage frequency multi-phase dividers are implemented. Also, external clock source is employed instead of PLL. After serialization, voltage-mode output driver is used instead of current-mode driver for low power consumption.

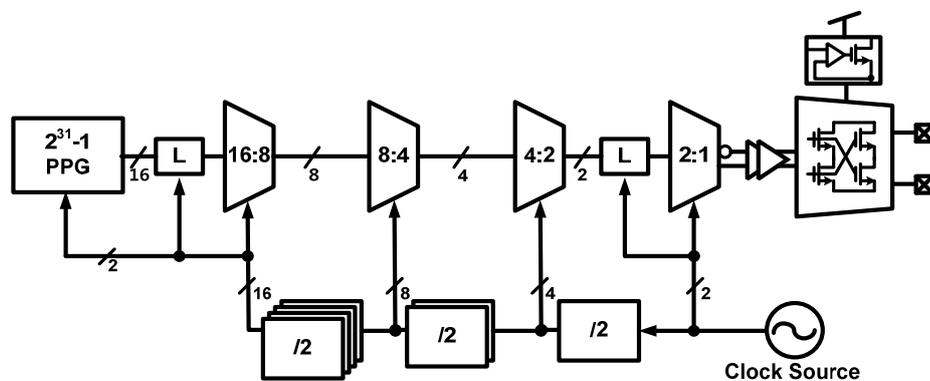


Figure 4.1: Architecture of prototype transmitter

## 4.2. Proposed Serializer

Our proposed tree-type 16-to-1 serializer is shown in the figure 3.2. This consists of 18EA latches and 15EA selectors.

For low power consumption, CMOS type latches and CMOS type selectors (transmission type selector) are used, because CMOS type consumes negligible amount of static power compared to CML type.

For latch, we adopted clocked inverter type for small power consumption, and this type can be compared to conventional transmission-gate type [6], [7]. The schematic topology is shown in Fig. 4.2. In simulation, 12.5-Gb/s input data is adapted to compare 2 types of latches. 22- $\mu$ W is consumed in conventional latch, and 20- $\mu$ W in clocked inverter latch. Furthermore, conventional latch has 14-ps of CK-to-Q delay, and clocked inverter has 6-ps of it. Thus, clocked inverter has lower power consumption with smaller CK-to-Q delay.

However, clocked inverter type latch cannot operate well in low-speed input data due to floating node in the structure. Clocked inverter itself does not have latching block but floating node, which results in vulnerable characteristic from leakage. This characteristic is simulated with frequency divider. In the simulation, input clock

frequency is applied from high-frequency (12.5-GHz) to low-frequency (under 2-MHz) with 1-V swing. The result is shown in Fig. 4.3. When input frequency is lowered to 2-MHz, the eye-diagram of output voltage shows that clocked inverter cannot maintain its voltage level, and frequency divider does not work in under 2-MHz of input clock frequency. Thus, we found that clocked inverter cannot work under 2-MHz in our design, but there is no problem operating 25-Gb/s data rate which is our target data-rate.

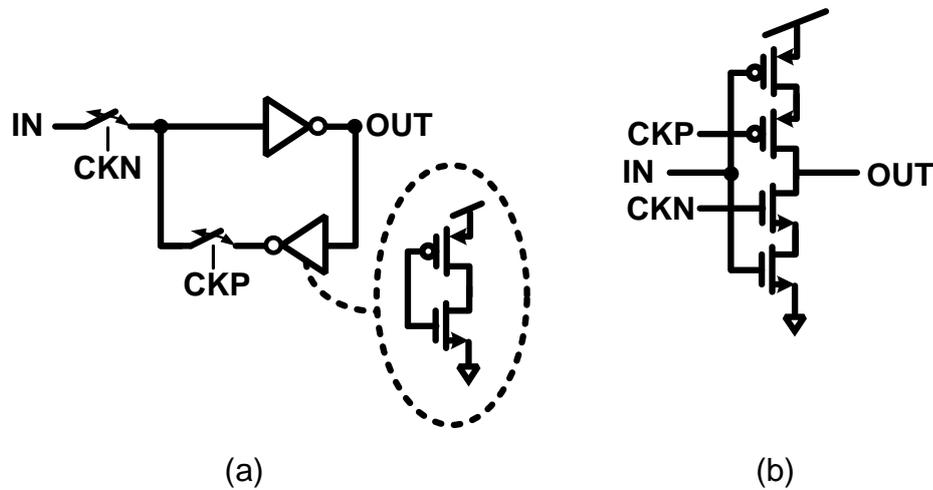


Figure 4.2: (a) conventional transmission-gate latch (b) clocked inverter latch

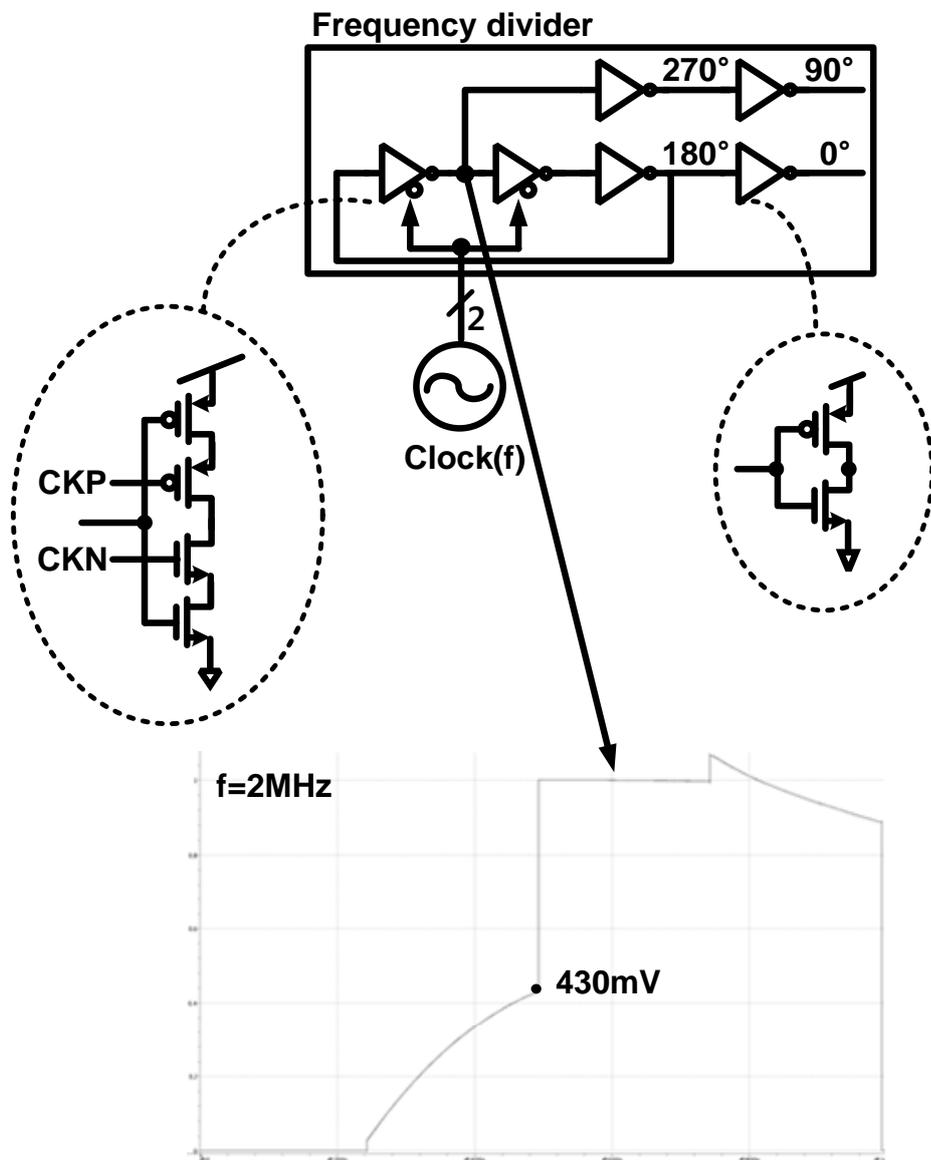


Figure 4.3: Clocked inverter type latch simulation with frequency divider

### 4.3. Multi-Phase Divider

In our proposed design, multi-phase clock signals are essential, so multi-phase generating frequency divider should be implemented. The operation is explained previously, and structure itself is described in figure 4.4. As can be seen in the figure, we adopted latch-type frequency divider for robust, simple, and broadband characteristics.

Furthermore, we used 1EA D flip-flop structure for generating 4 phase clock signals in our frequency divider, which can reduce power and area. On the other hand, there are clock phase skews between clock phases. As explained previously, clock signals  $0^\circ$  and  $180^\circ$  have 1EA inverter delay phase skew as well as clock signals  $90^\circ$  and  $270^\circ$ , and clock signals  $0^\circ$  and  $90^\circ$  has phase skew due to fan-out difference.

With the help of characteristics of tree-type serializer, problems due to clock phase skews are solved as explained in chapter 3.3. Thus, phase calibration circuit is not needed for generating multi-phase clock signals.

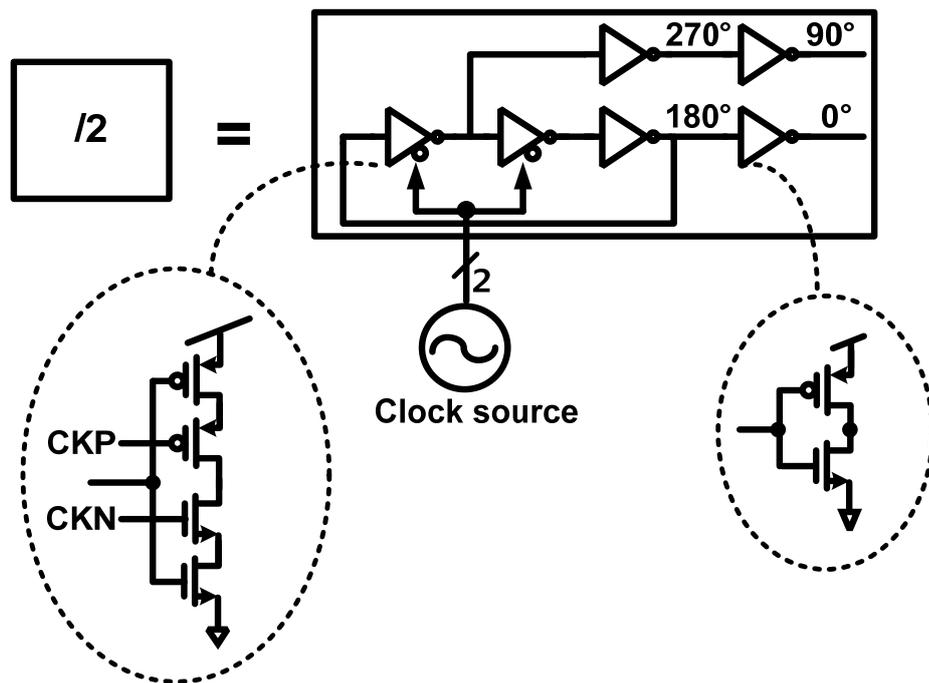


Figure 4.4: 4-phase latch-type frequency divider

#### 4.4. $2^{31}-1$ PRBS Pattern Generator

To verify whether the prototype transmitter is operating without any errors, pseudo-random binary sequence (PRBS) test pattern can be used. PRBS is a repetition of a pattern that itself consists of a random sequence of a number of bits. Since generating completely random binary waveforms is difficult, it is common to employ “pseudo-random” binary sequence [1].

A PRBS pattern is mostly generated by a connected feedback shift register as shown in the figure 4.5, and this consists of a cascaded string of binary storage elements. The contents are shifted simultaneously along the register with an external clock [8].

The prototype transmitter needs 16 bit parallel PRBS pattern, but it is very difficult to generate several parallel PRBS data with testing equipment. Thus, parallel PRBS pattern generator should be implemented inside the chip for testing simplicity. In the prototype transmitter, therefore, parallel PRBS31 pattern generator is implemented.

To generate parallel PRBS pattern generator, a polynomial format of PRBS31 should be considered, and it is shown as follows.

$$p(y) = y^{31} + y^{28} + 1 \quad (4.1)$$

This produces the sequence of  $2^{31} - 1$  bits. With satisfying this polynomial format, parallel PRBS31 pattern generator is described as shown in the figure 4.6. 31EA D flip-flops and 16EA XOR gates are employed. The reason for having 16EA XOR gates is to generate 16 parallel XOR output data. For example, the output node of DFF16 generates 16<sup>th</sup> data, 32<sup>nd</sup> data and so on, and 32<sup>nd</sup> data is  $D1 \oplus D4$ . Likewise, other D flip-flops (DFF1~DFF15) generate XOR outputs ( $D2 \oplus D5$ ,  $D3 \oplus D6$ , and etc) as shown in the figure 4.6. The detail of parallel PRBS pattern generator is described in [8].

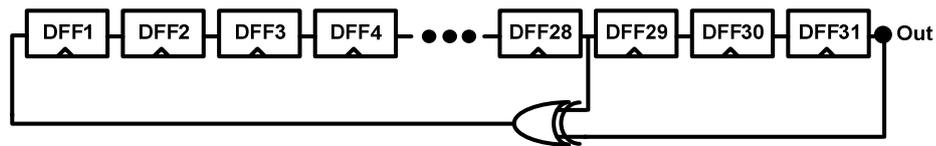


Figure 4.5: Schematic of series PRBS31 pattern generator

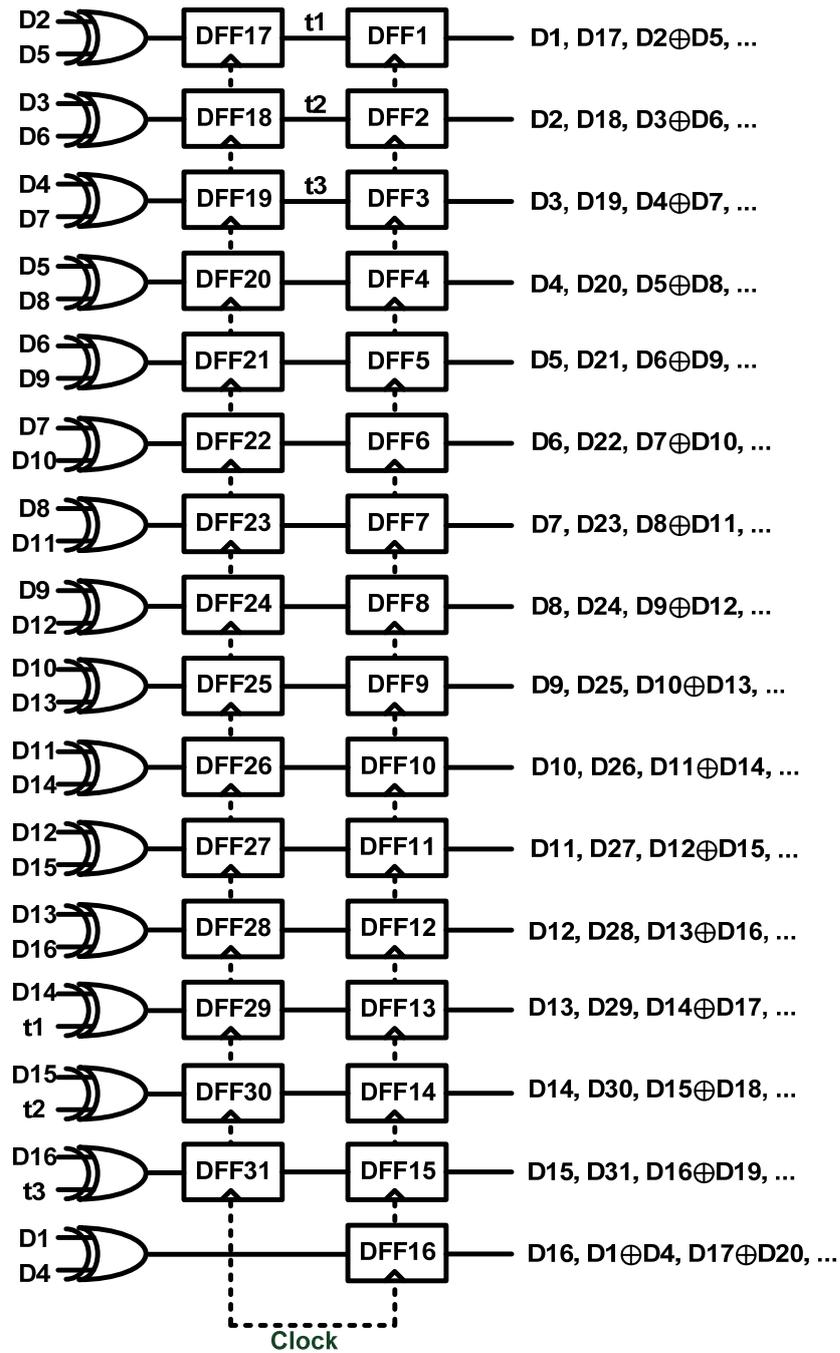


Figure 4.6: Schematic of PRBS31 pattern generator

## 4.5. Voltage-Mode Output Driver

In high-speed serial-link, a current-mode-logic (CML) type output driver is usually employed, because they support high data rates and low susceptibility to power supply noise. Furthermore, CML type output driver can adjust signal swing with simple control and can easily achieve impedance matching. These advantages, however, come with large power consumption.

On the other hand, voltage-mode (VM) type output driver consumes low power consumption. Also, with the help of technology development, 25 Gb/s data rate can be employed with this driver. Figure 4.7 shows basic schematic of CML output driver and VM output driver. When output swing is fixed with  $V_{\text{swing}}$  in both drivers, the current in CML driver is  $(V_{\text{swing}})/(R/2)$ , while the current in VM driver is  $(V_{\text{swing}})/(2R)$ . Thus, one fourth of the current in CML driver is used in VM driver.

The operation of VM driver is described in the figure 4.8. In the figure, 4EA MOSFETs are always ON regardless of input data level. Thus, current is always stable.

When it comes to impedance matching, CML type driver matches it with resistor, while VM driver matches it with MOSFETs

operating in triode region. Since impedance in VM driver is determined by the size of MOSFETs and gate voltage, impedance matching is very difficult in VM driver compared to CML type driver. Thus, as can be seen in the figure 4.9, replica driver is employed for impedance matching. Replica driver controls bias voltages ( $V_{ZP}$ ,  $V_{ZN}$ ) of VM driver, so impedance of VM driver maintains.

Furthermore, regulator is employed for controlling output swing without using additional power supply. With regulator, output swing of driver can be controlled, because VDRV of regulator is used as supply voltage of VM driver and replica driver. Thus, low swing output can be designed by using regulator. In our design, peak-to-peak output voltage swing is set by 150mV.

Even though, VM driver needs regulator and replica driver for adjusting voltage swing and impedance matching, a VM driver consumes much smaller power than CML driver. Therefore, we adopted VM driver as output driver.

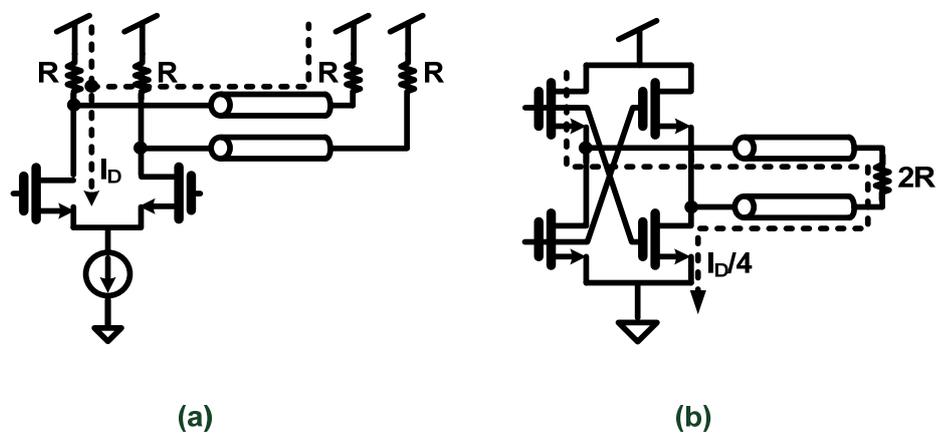


Figure 4.7: Schematic of (a) a current-mode driver, and (b) a voltage-mode driver

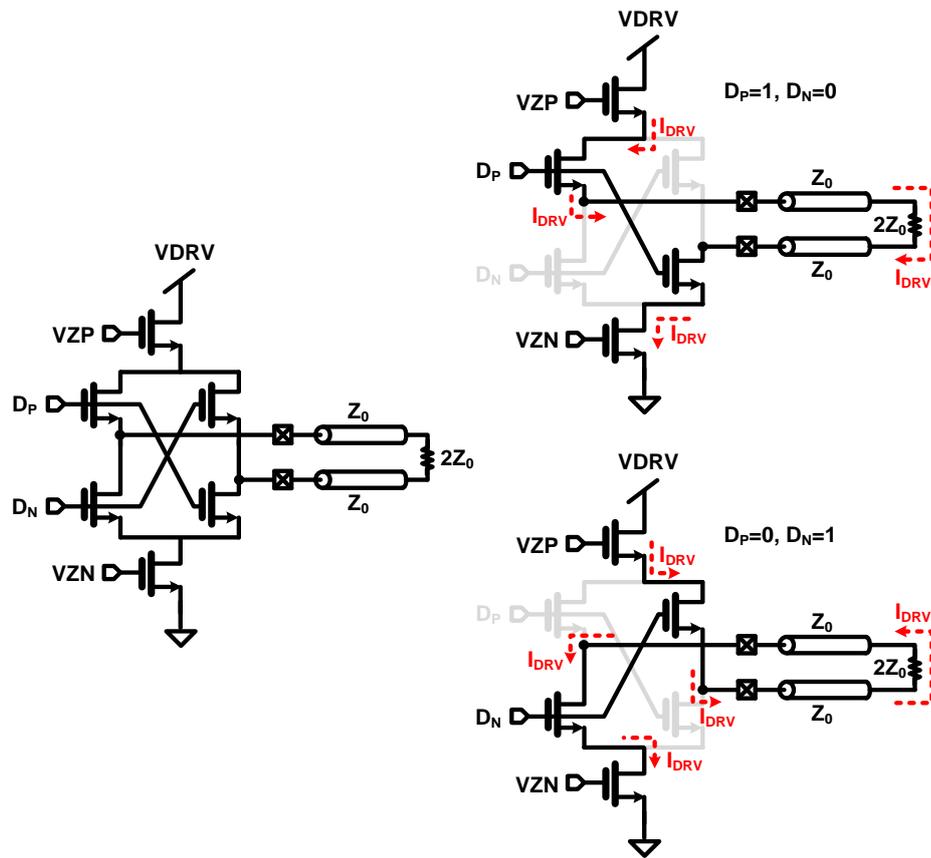


Figure 4.8: Operation of voltage-mode output driver

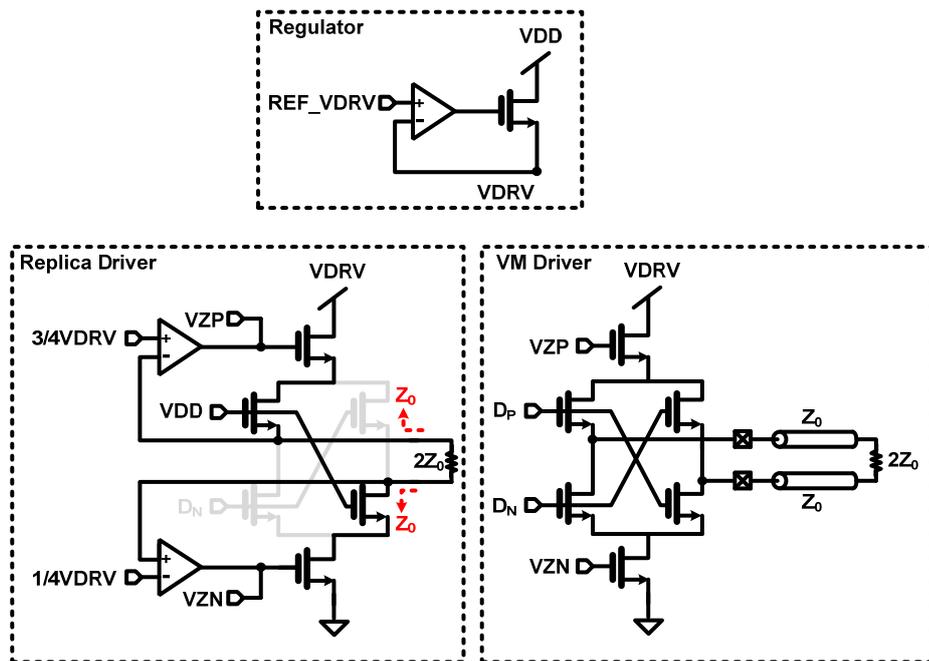


Figure 4.9: Regulator, Replica driver, and VM driver topology

## 5. Post-Layout Simulation Result

A prototype chip is designed in 28-nm CMOS technology. Figure 5.1 shows chip core layout. The circuit used 1-V supply at 25-Gb/s data rate and it occupies area of  $2160\mu\text{m}^2$ .

For energy efficiency of serializer and clock distribution, supply voltage is lowered while satisfying critical timing constraint, Eq. (3.2). Thus, as can be seen in the figure 5.2, low supply voltage is applied to low data rate, and the eye diagrams of serializer output data are shown. With this condition, energy efficiency of serializer and clock distribution is described in figure 5.3. Energy efficiency increases with increase in data rate, and energy efficiency is 0.184-pJ/b at 25-Gb/s data rate.

With same scheme, energy efficiency of transmitter is obtained. The eye diagrams are shown in the figure 5.4, and energy efficiency of transmitter is shown in the figure 5.5. Since output driver is not designed in the condition of low-swing input data, supply voltage of transmitter cannot be lowered under 0.85V in our design. Thus, energy efficiency of prototype transmitter is described from 0.85V to 1V of supply voltage. When it comes to power consumption of output driver, it depends on voltage swing not on data rate, so power consumption is

constant regardless of data rate, so energy efficiency decreases with increase in data rate. This tendency of energy efficiency for output driver is absolutely different from that for serializer and clock distribution. Thus, energy efficiency of transmitter has curve shape as can be seen in the figure 5.5, and has the lowest energy efficiency at 20-Gb/s data rate.

The power comparison between conventional 16-to-1 transmitter with 16-to-1 proposed transmitter is shown in the figure 5.6. We referenced [9] as a criteria of power breakdown. Total power consumption reduced from 9mW to 7.1mW, 21% power reduction. Power consumption of clock distribution reduced from 3.28mW to 2.21mW (33% reduction), and that of serializer reduced from 0.94mW to 0.51mW (46% reduction). These reductions come from removed latches and inverter chain in clock tree.

Table I summarizes this work along with recently reported low-power transmitters [2], [10], [11], and [12]. This work achieves the lowest energy efficiency among compared references.

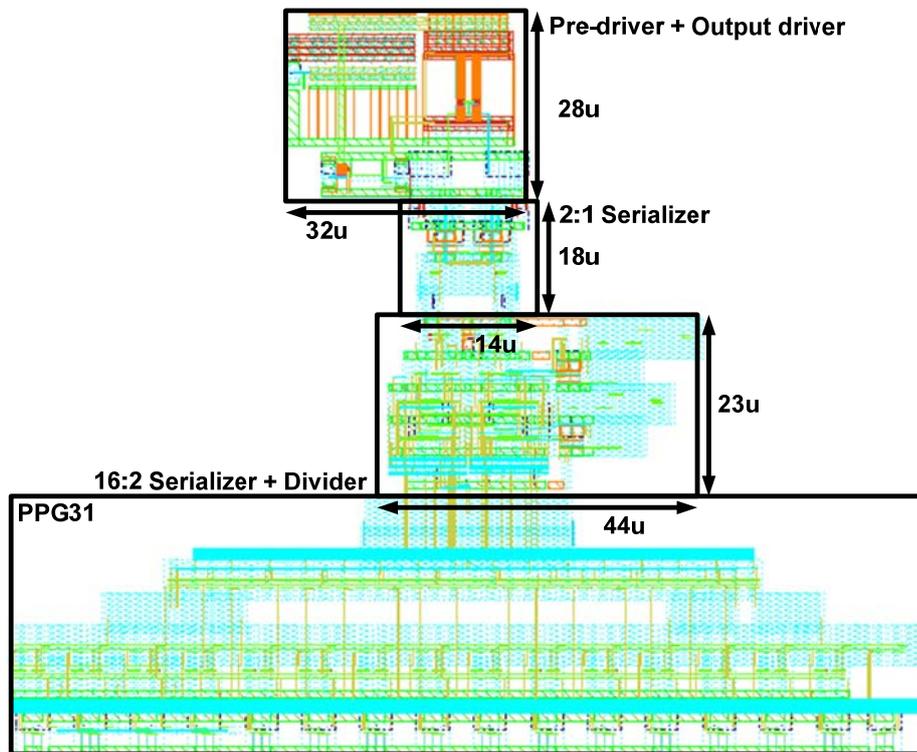


Figure 5.1: Layout topology of prototype transmitter

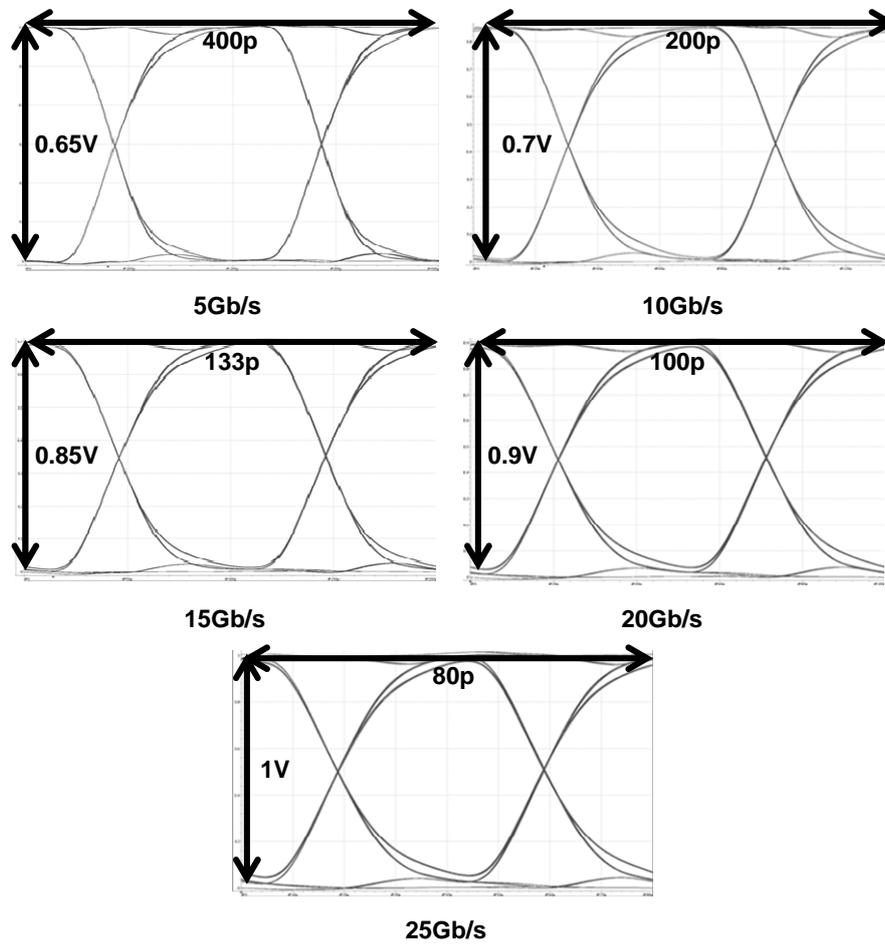


Figure 5.2: Eye diagram of serializer output data.

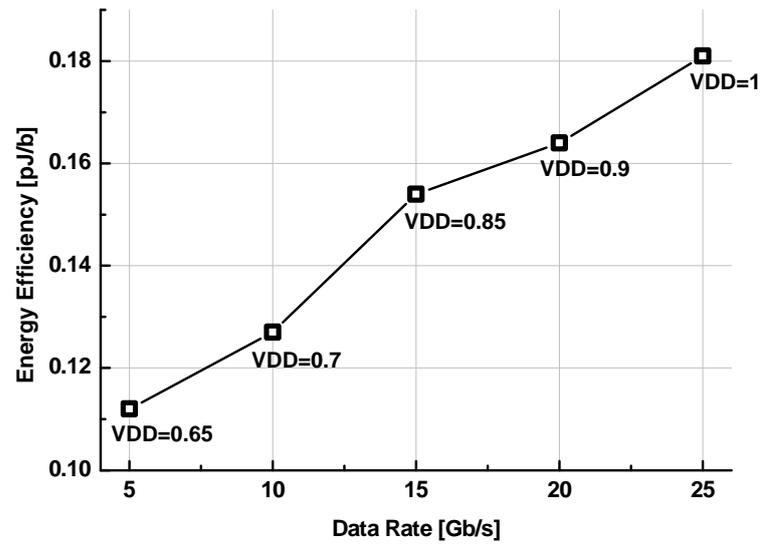


Figure 5.3: Energy-efficiency of serializer and clock distribution

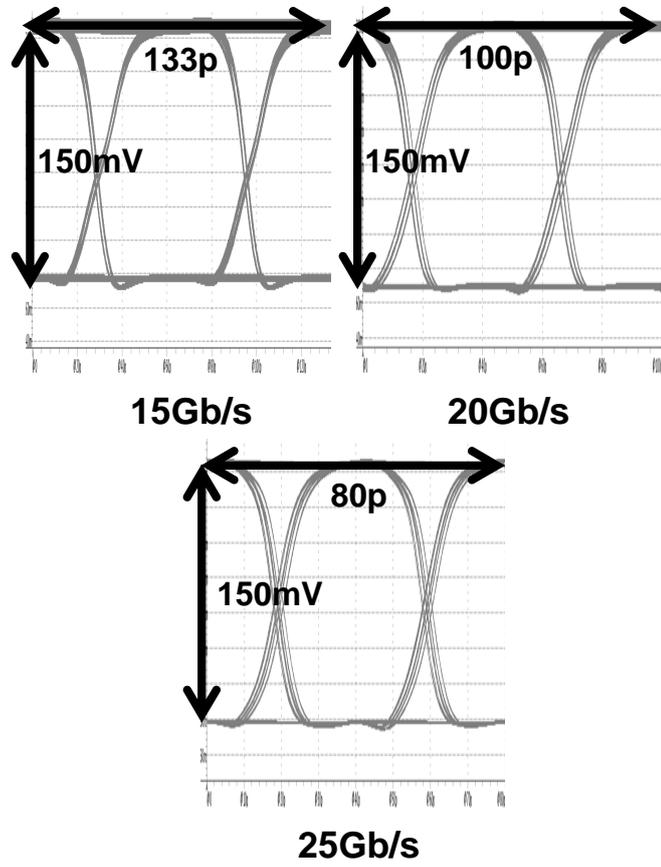


Figure 5.4: Eye diagram of transmitter output data

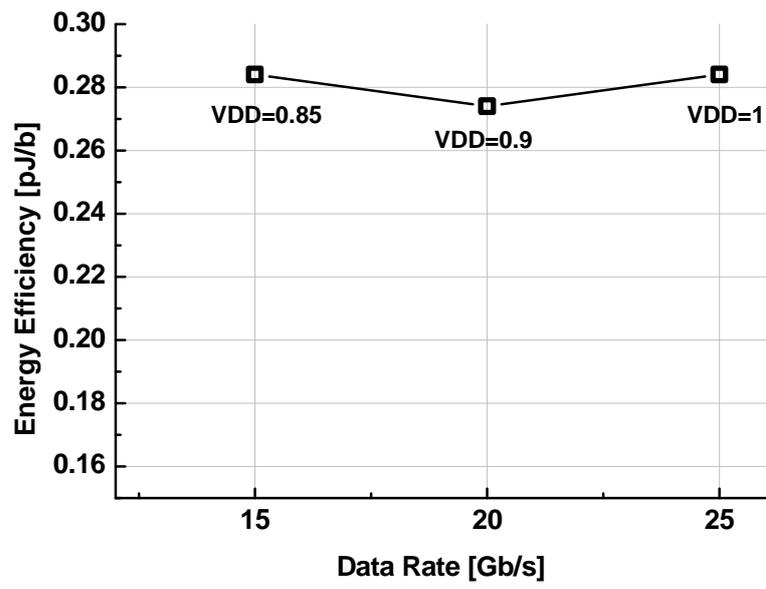
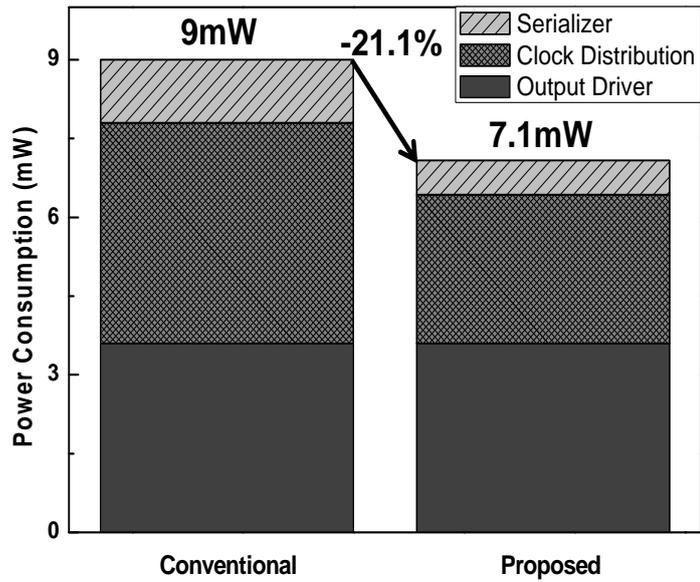


Figure 5.5: Energy-efficiency of transmitter



(b)

Figure 5.6: Power comparison between conventional transmitter and proposed transmitter

Table I

Performance Comparison With Previous Transmitters

	[2]	[6]	[7]	[8]	This work
CMOS technology	32nm SOI	65nm	65nm	65nm	28nm
Supply voltage	1	0.45-0.7	1	0.6-0.8	1
Data rate[Gb/s]	16	1-6	12.5	4.8-8	25
Serializer type	8:1	8:1	16:1	8:1	16:1
Driver	CML	VM	VM	VM	VM
Output swing	100mVppd	200mVpp	150mVppd	100-200mVppd	300mVppd
Power consumption[mW]	8.8	1.86	5.43	1.92	7.3
Area[mm <sup>2</sup> ]	-	-	-	0.022	0.0037
Energy efficiency[pJ/b]	0.55	0.31 (@6Gbps)	0.434	0.3 (@6.4Gb/s)	0.29 (@25Gbps)

## 6. Summary

In this paper, we propose a novel serializer without latches in the middle stage of tree-type serializer. Only the first and the last serializer have latches, which results in large reduction in power and area. By using multi-phase clock signals, a lot of latches are reduced without extra hardware cost. Furthermore, multi-phase clock signals are generated without extra power or area cost compared to conventional frequency divider. Also, in our design there is no need for phase calibration circuit that can cost a lot of power and area.

The prototype chip is designed with 28-nm CMOS technology. The overall transmitter architecture achieves low power and small area. PRBS31 pattern generator is implemented, and voltage-mode driver is adopted instead of current-mode driver due to its low power consumption. Our transmitter operates 25-Gb/s data rate with 7.1mW, which has low power consumption on high data rate.

This work can be more beneficial if the number of serializing data increases, because higher number of latches can be reduced.

## Bibliography

- [1] Behzad Razavi, Design of integrated circuits for optical communications, 2nd ed. New York, NY, USA: Wiley, 2012.
- [2] Timothy O. Dickson *et al.*, “A 1.8-pJ/bit 16x16-Gb/s source synchronous parallel interface in 32nm SOI CMOS with receiver redundancy for link recalibration” *IEEE CICC*, pp. 1-4, Sept 2015.
- [3] Hungwen Lu, and Chauchin Su, “A 1.25 to 5Gbps LVDS Transmitter with a Novel Multi-Phase Tree-Type Multiplexer” in *proc. IEEE A-SSCC*, Nov. 2008, pp. 389-392.
- [4] Wei-Yu Tsai, Ching-Te Chiu, Jen-Ming Wu, Shawn S. H. Hsu, and Yar-sun Hsu, “A novel low gate-count pipeline topology with multiplexer-flip-flops for serial link” *IEEE Trans. Circuits Syst. I*, Reg. Papers, vol. 59, no. 11, pp. 2600-2610, Nov. 2012.
- [5] C. J. Lombard, “Low power serializer circuit and method,” U.S. Patent 70060 21, Feb. 28, 2006.
- [6] Uming Ko and Poras T. Balsara., “High-Performance Energy-Efficient D-Flip-Flop Circuits” in *Symp. VLSI Circuits Dig. Tech. Papers*, 2000, pp 94-98.
- [7] Natsumi Kawai *et al.*, “A Fully Static Topologically-Compressed 21-Transistor Flip-Flop With 75% Power Saving” in *IEEE J. Solid-State Circuits*, Nov. 2014.
- [8] J. J. O’Reily, “Series-parallel generation of m-sequences” in *Radio and Electronic Engineer*, 1975, pp. 171-176.
- [9] John Poulton *et al.*, “A 14-mW 6.25-Gb/s Transceiver in 90-nm CMOS” in *IEEE J. Solid-State Circuits*, Dec. 2007.

- [10] Hiroshi Kimura *et al.*, “A 28Gb/s 560mW multi-standard SerDes with single-stage analog front-end and 14-tap decision feedback equalizer in 28nm CMOS” *IEEE ISSCC Dig. Tech. Papers*, 2014, pp. 38 – 39.
- [11] Timothy O. Dickson *et al.*, “A 1.4 pJ/bit, power-scalable 16x12 Gb/s source-synchronous I/O with DFE receiver in 32nm SOI CMOS technology” in *IEEE J. Solid-State Circuits*, Aug. 2015.
- [12] Woo-Seok Choi *et al.*, “A 0.45-to-0.7V 1-to-6Gb/s 0.29-to-0.58pJ/b source-synchronous transceiver using automatic phase calibration in 65nm CMOS” in *IEEE ISSCC Dig. Tech. Papers*, 2015, pp 1 – 3.

## Abstract (In Korean)

# Tree 구조 직렬 변환기를 사용한 고속 저전력 송신기

인터넷 통신망에서, 데이터 요구량의 증가로 인해, 고속의 송수신기가 필요하다. 고속의 데이터를 처리하는 데에는 큰 전력이 소모되므로, 고속 및 저전력의 송수신기가 요구된다.

이 논문에서는, 고속으로 동작하면서 저전력 저 면적의 송신기가 tree구조의 직렬 변환기와 함께 제안되었다. 이러한 목표를 위해 다중 위상의 클럭 신호들이 사용되어 tree 구조의 직렬 변환기 안의 래치들을 줄일 수 있었다. 추가적인 전력소모 없이, 다중 위상 클럭 생성 주파수 분할기를 내재하였고, 성능 저하 없이 많은 래치들을 줄일 수 있었다. 또한, 제안된 tree구조 직렬 변환기와 함께 전류 모드 드라이버 대신 전압 모드 드라이버를 사용하여 저전력을 구현하였다.

28 나노 공정으로 만들어 졌으며, 제안된 송신기는 25Gb/s의 데이터 속도를 가지면서 저전력과 저 면적으로 설계되었다. 동작은  $2^{31}-1$  PRBS 패턴 제너레이터를 통해 확인을 하였다. 레이아웃 후 시뮬레이션을 통해 25Gb/s 데이터 속도를 동작시키며 7.1mW의 전력을

소모하였다.

---

**핵심 단어:** 직렬 변환기, 래치, 다중 위상 클럭 신호, 전압 모드  
드라이버

## List of Publications

### *Domestic Conference Presentations*

- [1] **Tongsung Kim** and Woo-Young Choi, “Power-Optimized Design of N:1 Serializer in 65-nm CMOS,” 제 23 회 반도체 학술대회, Feb. 2016, 정선.