A METHODOLOGY FOR OPTIMUM DELAY, SKEW, AND POWER PERFORMANCES IN AN FPGA CLOCK NETWORK

Mohd S Sulaiman

Faculty of Engineering, Multimedia University, Selangor, Malaysia

Key words: FPGA clock network, High performance, IC design, Low power design, CMOS

Abstract: A methodology for FPGA clock network optimisation is presented. The algorithms for optimisation of clock skew, delay, and power considering slew rate constraint for an FPGA fixed-clock network are implemented and verified on SX 32 FPGA chip. Measurements indicated a 60% reduction in clock slew rate and a 22% improvement in power dissipation when compared to the results of the initial, un-optimised chip.

Metodologija za doseganje optimalne zakasnitve, porazdelitve signala in porabe moči urinega omrežja vezij FPGA

Kjučne besede: CMOS, FPGA, urina vezja, načrtovanje integriranih vezij, načrtovanje vezij z majhno porabo, CMOS

Izvleček: V prispevku predstavljamo metodologijo za doseganje optimalnega delovanja urinega omrežja znotraj vezij FPGA. Algoritmi za optimizacijo zakasnitve, popačenja signala in porabe moči so bili uvedeni in preverjeni na vezju SX 32 FPGA. Meritve pokažejo 60% zmanjšanje v popačenju signala in 22% zmanjšanje porabe moči v primerjavi z rezultati pred optimizacijo.

1 Introduction

Modern high performance VLSI systems are designed to work at a specific maximum clock frequency depending on their applications and the process technology used. One of the constraints in achieving maximum clock frequency is clock rise time (*slew rate*). The longest rise time in the clocking network limits the clock frequency (clock period) /1/.

A clock network is responsible for distributing clock signal from an input pad to the clock input of each block in an IC (sink). The clock net should be able to maintain the clock signal integrity. The distribution of clock signal on the chip must be done while minimizing the clock delay, clock skew, and slew rate /3/. *Clock delay* is defined as the maximum delay from the clock source to the input of any logic block. *Maximum clock skew* is defined as the difference between the longest clock delay and the shortest clock delay in the system.

To achieve minimum slew rate while maintaining clock signal integrity, minimum number of buffers are added into the clock tree. This technique also helps to reduce clock delay and clock skew. Proper buffer placement and buffer sizing minimizes the slew rate. Capacitive load of the driven gates/logic blocks (gate load) and parasitic loads of signal line (wiring load) are among the factors that affect clock slew rate, delay and power dissipation. Since power consumed by the clock network contributes a major portion of the total chip's power consumption /4/, reducing the clock net power consumption will have tremendous effect in system's overall power consumption.

Research works in the buffered clock network mainly focused on the minimization of clock delay and clock skew. As far as this work is concerned, previous works have not addressed the problem of clock delay, skew, and power optimisation with slew rate constraint. Work in /1/ emphasized on generating a clock network and optimising clock delay and skew without considering power performance while work in /3/ focused on inserting minimum number of buffers in clock trees with skew and slew rate constraints. The latter assumes that the clock tree will be buffered by a single type CMOS buffer. No doubt, this strategy helps to reduce clock skew and skew sensitivity to process variation. It, however, requires a balanced tree network, which is not applicable to FPGAs. The simultaneous change of buffer size and wire width to optimise performance and power in /5/ assumes that buffer locations are already given; i.e. clock tree is already buffered. This technique is useful in minimizing delay and power but it does not consider any slew rate constraint. Although the work done in /6/ can significantly reduce the clock delay and clock skew, it does not consider the effect buffer insertion has on slew rates and power dissipation. Additional capacitive loading imposed by adding buffers into the clock tree will increase the slew rates and power dissipation, which is not desirable, especially for high-speed mobile applications.

Based on these observations, this paper proposes an optimisation methodology for optimum clock delay, skew, and power performances for a given slew rate constraint.

For this work, several constraints are considered and a few assumptions are made:

- Buffers can be inserted at tree nodes only due to the FPGA physical layout constraint.

- The maximum clock slew rate is 0.5 ns, and the maximum allowable clock delay is 2.5 ns for CMOS 0.35-um technology.
- The clock tree will be buffered by buffer of different sizes due to loading considerations.

The outline for the remainder of this paper is as follows. Problem formulation is discussed in Section 2. In Section 3, the algorithm that solves the initial buffer insertion problem is presented. Algorithm for delay and slew rate optimisation by changing buffer position is discussed in Section 4. Section 5 discusses the buffer sizing strategy for simultaneous clock delay, skew, slew rate, and power optimisation. Section 6 explains the wire width sizing technique for delay reduction. Section 7 contains the simulation results and comparisons. Conclusions are presented in Section 8.

2.1 Definitions

The definitions of the terms that will be used in the later sections are as follows:

- *Unbuffered Clock Tree* (UBT): a clock tree T(V,E) consisting of wires (edges) E and nodes V with no buffers between the source and sink nodes (initial clock tree).
- *Buffered Clock Tree* (BFT): a clock tree T(V,E) after buffer insertion.
- *Wire* (E): an internal signal line connecting logic block's input to its output.
- *Node (V)*: a point that connects two logic blocks together

An FPGA is made up of z number of logic blocks (**x** rows x **y** columns, where **x** x **y** = **z**, input-output (I/O) blocks, and programmable interconnects (see Fig. 1). Logic blocks can be either logic modules or flip-flops (see Fig. 2 (a) and (b), respectively). The circuit models for this work are as shown in Fig. 3 and Fig. 4. Each clock tree branch (vertical column) consists of logic blocks and a driver to drive the clock signal to all the logic blocks in the column (see Fig. 4). Logic blocks are modelled as a series RC-circuit while the vertical wires are modelled as a π-RC circuit (see Fig. 3). The resistance R for the two models (series π-RC and RC) is given by the following formula:

$$R = \left(\frac{L}{W}\right)\rho \qquad (1)$$

where L = length of Logic Module

W= width of vertical track for the path of clock signal inside the logic block

r = sheet resistance of signal path

The capacitance in the RC-circuit that models the logic block for CMOS 0.35-um technology is 21.2 fF (calculated based on the 3D modelling technique described in /7/).

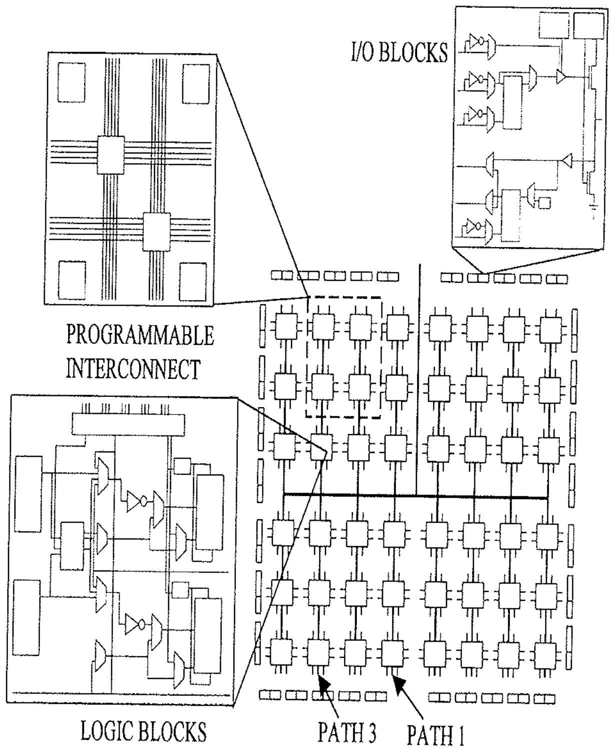


Fig. 1. FPGA Building Blocks

Horizontal wires are modelled as a PI-RC circuit (see Fig. 3). Wire resistance is calculated as follows:

$$R = \frac{L_h}{W_h}\rho + R_{\text{interconnect}} \qquad (2)$$

where L_h = horizontal length of logic module

W_h = horizontal width of logic module

r = sheet resistance of signal line

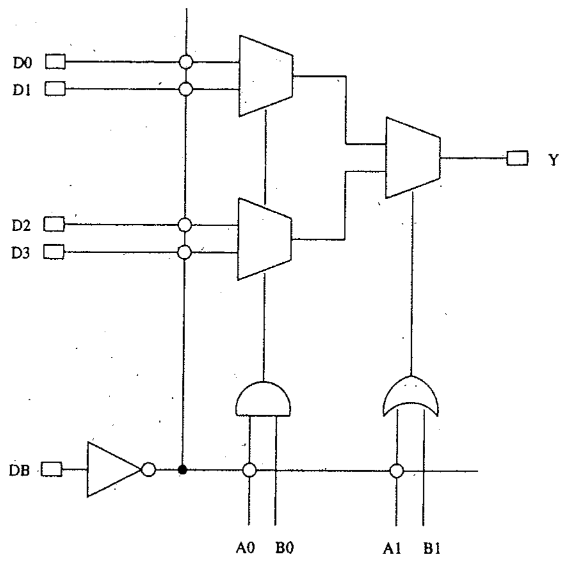
$R_{interconnect}$= resistance of interconnect

The capacitance is found to be 23 fF (based on the method described in /7/).

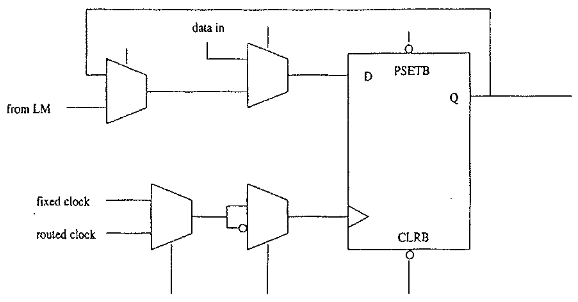
Figure 5 shows the sketch of the proposed technique for simultaneous optimisation of clock delay, skew, and power with slew rate constraint. The overall algorithm is shown in TABLE 1.

i. Initial Buffer Insertion:

Inserts different number of buffers in each source-to-sink for a UBT depending on the slew rates of that path.



(a)



(b)

Fig. 2. FPGA Logic Blocks: (a) Logic Module; (b) Flip-flop

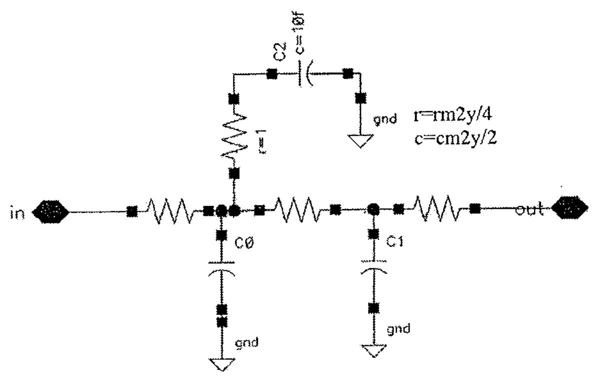


Fig. 3. RC Model for Logic blocks and wire

ii. Delay & Power Optimisation (Slew Rate Constraint) - modified /6/

Buffer locations are optimised to minimize clock delay. Buffer sizes in the BFT are changed accordingly to minimize slew rate, should there be a need to do so.

iii. Delay & Skew Optimisation by Buffer Sizing - proposed by /5/ and /6/, modified to consider slew rate constraint

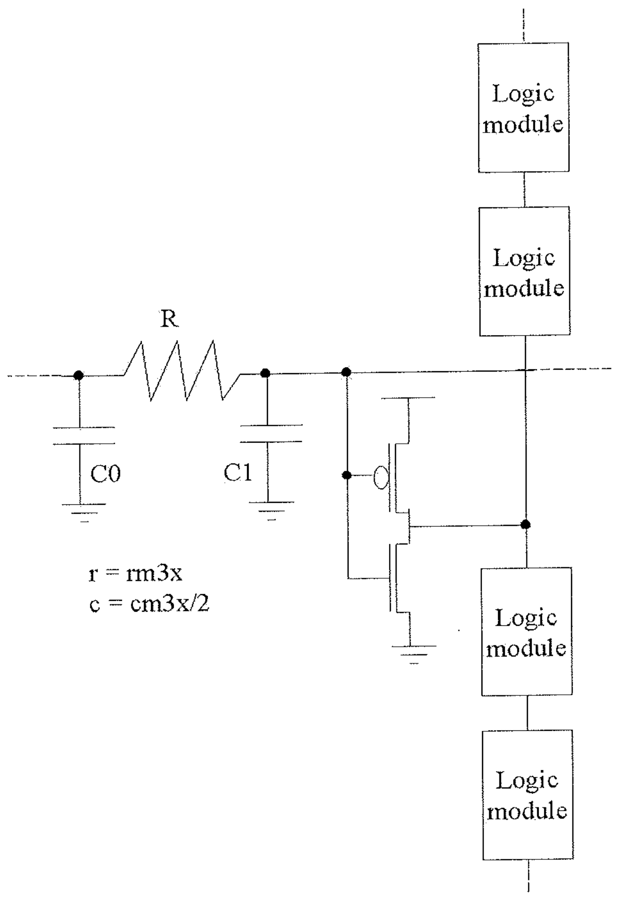


Fig. 4. Column model

Path by path, buffer sizes are changed to change the delay of each path/column.

iv. Wire sizing approach to optimise power, skew, and delay

Wire widths are changed to see the effect on clock delay, skew, and optimisation.

Table 1 - Overall optimization algorithm

```
Get slew rates and delay of every path
Do Initial Buffer Insertion

while flag = TRUE
  Do Delay and Power Optimization
  (Buffer Positions)

  if slew rates < $t_{max}$
    Do Delay & Skew Optimization (Buffer Sizing)
    flag = FALSE
  else
    flag = TRUE
    Do Slew Rate Minimization (change size of $B_{k-1}$)
  end
end
Do Skew & Power Optimization (Wire Sizing)
```

In this section, Initial Buffer Insertion problem is discussed.

Initial Buffer Insertion Problem: Given a UBT and a maximum clock delay, clock skew, and slew rates, determine the number of buffers to be inserted in each path (source to sink) such that slew rate is less than t_{max} of 0.5 ns. That is, number of buffers in each path = f(slew rate)

To solve the initial buffer insertion problem, given a slew rate constraint, the paths are first sorted in ascending order according to the maximum path clock delay, i.e. t_{delay_P1} (t_{path_min}) < t_{delay_P2} < ... < t_{delay_P20} < t_{delay_P21} (t_{path_max}). Next, the size of column driver is reduced by a factor of two.

We then measure the maximum slew rate of every path, starting with the shortest delay path, i.e. path 1. If the slew rate is less than t_{max}, no buffer is inserted in that path.

However, if the slew rate is greater than t_{max}, one buffer is inserted right in the middle of the clock path. The maximum slew rate for the path is then measured. If it is still greater than t_{max} or if the rise time improvement is less than 15%, we remove the buffer and start increasing the size of column driver until the goal is achieved.

If there is at least 15% improvement in path slew rate but it is still greater than t_{max},, another buffer is inserted. This time the two buffers are placed such that they divide the clock path in three equal-length sections. The buffer insertion step is repeated until the slew rates for all paths are less than t_{max}. In general, if we have k buffers in a path, they should be arranged such that they divide the path in k+1 equal sections.

For instance, consider three paths in the FPGA chip shown in Fig. 1. Initially, the slew rates for paths 1, 3, and 21 are 482 ps, 583 ps, and 940 ps, respectively. After the size of column driver has been reduced by a factor of two and one buffer has been inserted in path one, the slew rate is reduced by only 5%. Therefore, we remove the buffer in path 1 and start increasing the size of column driver (see Fig. 4). The final result for path 1 is that the slew rate is less than 0.5 ns. Path 3 needs one buffer while path 21 (the longest path, not shown in Fig. 1) needs two buffers.

We, then, continue the optimisation process with the simultaneous optimisation of clock delay, slew rate, and power by changing buffer positions.

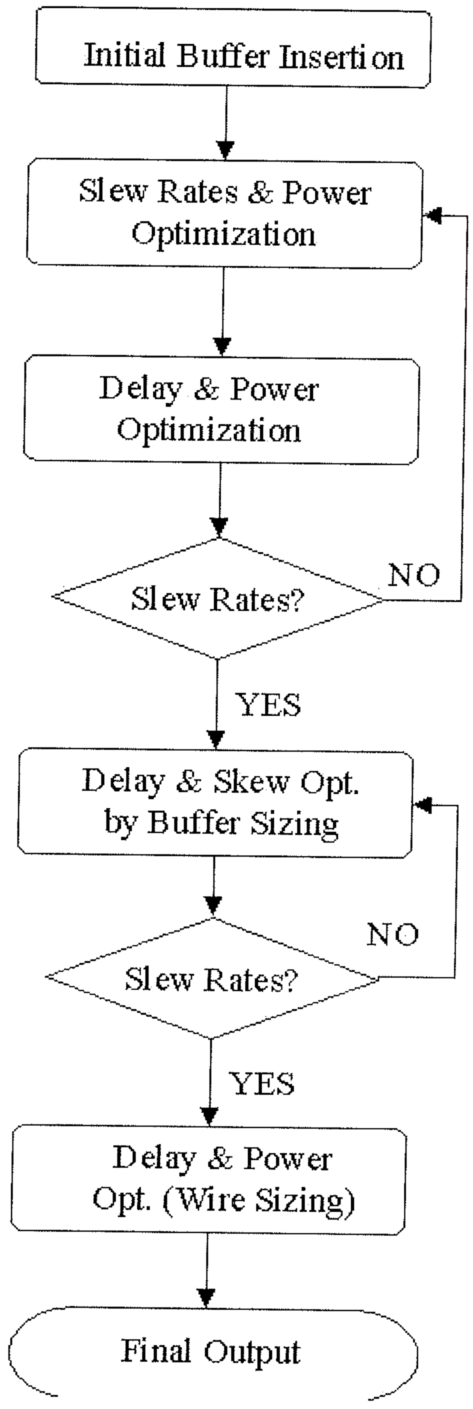

Initial Buffer Insertion
Slew Rates & Power Optimization
Delay & Power Optimization
Slew Rates?
NO
YES
Delay & Skew Opt. by Buffer Sizing
Slew Rates?
NO
YES
Delay & Power Opt. (Wire Sizing)
Final Output

Fig. 5. Optimisation Approach

The algorithm selects a buffer from source to sink (depth-first) and move its position to reduce the clock delay. If the delay is reduced, the buffer is moved to the new location, otherwise it stays at its original position. After all buffer positions have been optimised, we check the slew rate. A buffer is selected from sink to source (bottom-up). If the slew rate at input of buffer B_k is greater than 0.5 ns, we increase the size of buffer B_{k-1} until the slew rate is less than t_{max}.

TABLE III displays the Buffer Positioning Algorithm for simultaneous optimisation of clock delay and power with slew rate constraint. The original algorithm developed by /6/ was modified to consider the slew rate constraint.

Table 5 displays the algorithm used for optimising clock delay and skew. This original technique was developed by /6/, and then was modified to take into account the slew rate constraint.

Step 1: The algorithm arranges the path according to its delay in ascending order; t_{delay_P1} (t_{path_min}) < t_{delay_P2} < … < t_{delay_P20} < t_{delay_P21} (t_{path_max}).

Step 2: Starting with t_{path_min}, it selects a buffer from source to sink (depth-first).

Step 3: The buffer size is changed by Δs. If the delay is reduced, we then check the slew rate. We stick with the new size should the slew rate be less than t_{max},. Otherwise, the buffer size is not changed.

Step 4: If delay is reduced and slew rate is less than t_{max}, repeat step 3 until there is no further improvement.

Wire width sizing strategy for delay, skew and power optimisation is as described below:

Step 1: Given a maximum wire width due to chip area constraint, vertical and horizontal wire widths are increased.

Step 2: If clock delay, and skew are reduced, then change the widths.

Step 3: Otherwise, reduce wire widths. If delay and skew is decreased, keep reducing the wire widths. Else stop.

The results of the algorithms developed were verified on an FPGA chip where the clock tree is neither balanced nor of equal length. All the logic blocks in the FPGA chip have fan-out of one for worst-case capacitive loading imposed on the clock tree.

Table 4 presents the results of clock delay, skew, slew rates, and power before and after the clock tree is optimised. Chip area savings of 765 μm^2 is achieved with the new optimised clock tree. Power dissipation is improved by 22%. For all paths, the slew rates are reduced to within the scope of 500 ps (t_{max}). These values range from 366 ps to 463 ps.

Table 2 - Delay & Power optimization algorithm

```
Input: Buffered Clock Tree (BFT) with n
       paths and k_i level of buffers in path i.
       n = no. of paths
       Δx: buffer moving step (single-node step)

Output: Optimized clock path (position-wise)

Procedure: PathDelayMinimization (Path i, k_I, Δx, n)
for path = 1 to n
  // optimize delay (move buffer position)
  for buffer level i = 1 to k (depth-first approach)
    move buffer i, B_i up Δx
    check path clock delay

    if delay is reduced
      stay at new location
          keep moving up Δx until no further
      improvement
    else
      go back to old location
      move buffer i, B_i down Δx
      check path clock delay
      if delay is reduced
        stay at new location
              keep moving down Δx
        until no improvement
      else
        back to old location
      end
    end

end

  // minimize slew rate
  for buffer j = k to 1 (bottom-up)
    if slew rate at input of B_j > 500 ps
      increase size of B_j-1
    end
  end
end
```

Table 3 - Delay and skew minimization with slew rate constraint

```
Input: BFT with optimized buffer position
       $\Delta s$: buffer size increase
       n = no. of paths

Output: Optimized buffer sizes for delay, skew,
slew rate and power.

Procedure: BufferSizing (Path i, $k_i$, $\Delta s$, n)

for path = 1 to n (i.e. $\tau_{path\_min}$ to $\tau_{path\_max}$)
  for buffer level i = 1 to k

    // increase buffer size
    while increase_buffer_size = TRUE
      increase size of buffer i, $B_i$ by $\Delta s$

      if delay is reduced
        if slew rate < $t_{max}$
        buffer size = new size
        increase_buffer_size = TRUE
      else
        buffer size = old size
        increase_buffer_size = FALSE
      end
    end

    // reduce buffer size
    while reduce_buffer_size = TRUE
      reduce size of buffer i, $B_i$ by $\Delta s$

      if delay is reduced
        if slew rate < $t_{max}$
        buffer size = new size
        reduce_buffer_size = TRUE
      else
        buffer size = old size
        reduce_buffer_size = FALSE
      end
    end

  end                    // end of 2nd for loop

// need additional buffers?
  if delay > 2.5 ns ($t_{max}$)
    add another buffer in the path
    go to PathDelayMinimization
  end

  end           // end of main for loop
```

In this paper, a methodology for optimisation of clock delay, clock skew and power with slew rate constraint is presented. This method is effective especially when dealing with trade-offs among delay, skew, power, and slew rate for an FPGA chip.

The results presented in this paper have shown convincingly that the method developed yields sharper rise and fall edges and reduces power dissipation with practically no penalty in the clock delay.

/1/ I-Min Liu, T.L. Chou, A. Aziz, and D.F. Wong, "Zero-Skew Clock Tree Construction by Simultaneous Routing, Wire Sizing and Buffer Insertion", **Proc. 2000 Int'l Symposium on Physical design, pp. 33-38, 2000.**

/2/ M.Afghani and C.Svensson, "Performance of synchronous and asynchronous schemes for VLSI systems", *IEEE Trans. Comput.*, vol. 41, no. 7, pp. 858-872, 1992.

/3/ G.E. Tellez, "Minimal Buffer Insertion in Clock Trees with Skew and Slew Rate Constraints", *IEEE Transactions on CAD of IC and Systems*, vol. 16, pp. 333 - 342, April 1997.

/4/ J.W. Chung, D.Y. Kao, C.K. Cheng, and T.T. Lin, "Optimization of Power Dissipation and Skew Sensitivity in Clock Buffer Synthesis", *ISLPED 95*, pp. 179 - 184, 1995.

/5/ J.Cong, C.K.Koh, and K.S.Leung, "Simultaneous Buffer and Wire Sizing for Performance and Power Optimization", *ISPLED 96*, pp. 271 - 276, 1996.

/6/ X. Zheng, D. Zhou, and Wei Li, "Buffer Insertion for Clock Delay and Skew Minimization", *ISPD 99*, pp. 36 - 41, April 1999.

/7/ T.Stohr, et al, "Analysis, Reduction and Avoidance of Crosstalk on VLSI Chips", *ISPD 98*, pp. 211 - 218, 1998.

Mohd S Sulaiman
Faculty of Engineering, Multimedia University, 63100 Cyberjaya, Selangor, Malaysia
E-mail: shahiman@mmu.edu.my

Prispelo (Arrived): 30. 01. 2006; Sprejeto (Accepted): 29. 05. 2006

Table 4 - Comparison of clock delay, skew, slew rate, power dissipation, and buffer area between the unoptimised design and the optimized design

	Initial Clock Tree		Optimised Clock Tree		% Improvement
	Shortest Path (1)	Longest Path (21)	Shortest Path (1)	Longest Path (21)	Overall (Path 21)
Rise Time (ps)	481.1	939.5	463.5	362.8	61.4
Fall Time (ps)	493.0	842.9	456.3	385.7	54.2
Clock Delay (ns)	1.43	2.35	1.38	2.26	3.8
Maximum Clock Skew (ns)		0.92		0.88	4.3
Power (mW)		112.7		87.4	22.4
Area occupied by Buffers & Column Drivers (μm^2)		5145		4380	14.9