

# Clocking for Correct Functionality on Wave Pipelined Circuits

Woo Jin Kim, Yong-Bin Kim

Electrical and Computer Engineering Department, Northeastern University

**Abstract**—Wave pipelining is a method of speeding up the circuit design without the insertion of registers. This reduces the overhead related to clocking significantly and allows for better temporal logic utilization. However, because of the fact there are no intermediate registers, the clocking of wave pipelined circuits becomes more delicate and very susceptible to delay changes due to process, voltage and temperature (PVT) variations. With traditional method of pipelining, it becomes just a problem of frequency adjustment, but, with wave pipelining, it becomes a more difficult problem due to more complicated constraints involving clock frequency and skew. In this paper, we look at some of the prior works done in this area, and introduce a localized clock circuitry scheme to address this issue.

## I. INTRODUCTION

Pipelining can be divided into two techniques: conventional pipelining and wave pipelining. The conventional pipeline method partitions the combinational logic into smaller chunks and inserts registers at the boundaries. Wave pipeline method, on the other hand, does not physically partition the logic or insert registers. Instead it utilizes the fact that, if the logic path is long enough and the data dispersion can be reasonably small, multiple sets of data can be sent through the logic at a faster clock rate and without latching the data on the way.

The wave pipeline enjoys several advantages over the conventional technique. It can achieve high clock rate without partitioning the logic and adding synchronization elements thereby reducing quantization overhead associated with the conventional pipeline as well as the clock buffering and routing which reduce the layout congestion.

However, Wave pipelining does require tighter timing constraints as there are no intermediate registers to store the intermediate data. It is especially vulnerable to delay changes due to process and operating environment. This is due to the fact, although conventional pipeline only depends on max delay, wave pipeline gets constrained by minimum delay as well; thus getting double the impact in the worst case.

This paper looks at some of the previous research done in this area and also presents a simple circuitry scheme to address the issue.

## II. BACKGROUND

To a first order, it can be generalized that wave pipelining achieves a speed-up of  $N$ . For the wave pipeline scheme,  $N$  represents the number of data "waves" that propagate simultaneously through the design. This is similar to the traditional pipeline scheme where  $N$  would be the number of pipe stages.

### A. Basics of Wave Pipeline

The clocking of the traditional pipeline is constrained by the worst delay among the pipe stages and the setup and hold times of the registers at the design boundary. Similar constraint exists for the wave pipeline design as well. However, it is constrained by both the worst and best delays as the clocking is a function of the difference of the two. Simple clocking constraint can be expressed as follows.

$$T_{CLK} > T_{DELAY(Max)} - T_{DELAY(Min)} + T_{SETUP} + T_{HOLD} + 2 * T_{SKEW} \quad (1)$$

where  $T_{CLK}$  is the clock cycle,  $T_{DELAY(Max)}$  is the maximum propagation delay of the design,  $T_{DELAY(Min)}$  is the minimum propagation delay,  $T_{SETUP}$  and  $T_{HOLD}$  are the setup and hold times of the register and  $T_{SKEW}$  is the clock skew. The sum of setup and hold times of the registers and the clock skew constitute the clocking overhead. Since, there are no additional registers and their inherent delays, latency is reduced when compared to conventional pipelining.

This clocking scheme increases the percentage of logic utilization and, for properly designed circuits, can result in a clock rate faster than what can be achieved by the conventional pipeline. Figure 1 illustrates this.

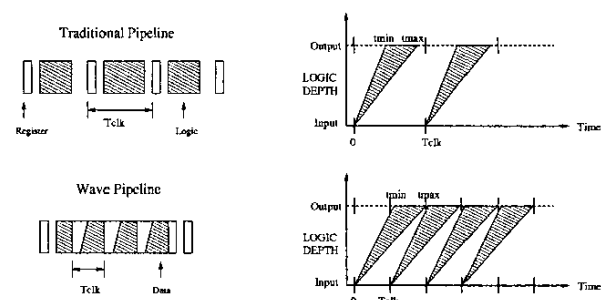


Fig. 1. Wave Pipeline Concept

In the figure, it can be seen that the delay from the inputs to the outputs of the blocks tend to spread out. This is due to the topology difference of the paths. The leftmost and rightmost points correspond to the  $T_{DELAY(Min)}$  and  $T_{DELAY(Max)}$  of the design. The spread of the data is where the data is still unstable, and the data should not be latched by the registers at this point. The clocking scheme should be such that the clock trigger falls between

successive data waves with the setup time, hold time and the clock skew accounted for.

At any given instance for a wave pipelined design, there can be many waves of data flowing through the design at the same time. In Figure 1, it can be seen that there are two waves going through the design at the same time. If we look at the  $T_{clk}$  point which is the second trigger point, the first data wave is almost at the output while a second wave is being inserted into the design. For this design,  $N$  is equal to 2.

Similar constraints exist for the internal nodes of the design as theoretically the internal nodes can have more tighter constraint. This so called signal separation constraint or the internal node constraints can be written in the following form

$$T_{CLK} > T_{nodeX}(Max) - T_{nodeX}(Min) + T_{Stable} + T_{SKEW} \quad (2)$$

where, again,  $T_{CLK}$  and  $T_{SKEW}$  are the clock period and the clock skew.  $T_{nodeX}(Max)$  and  $T_{nodeX}(Min)$  are the largest and smallest delay from the input to the internal node,  $nodeX$ .  $T_{Stable}$  denotes the time for which  $nodeX$  must hold the data stable in order to propagate the data properly to the next gate. Equation 2 has been derived in many papers [1], [2], [3], [4], [5] in slightly different forms and the parameters used are quite straight forward. The clock skew is accounted for only once in the internal node because only the input register is in play.

### B. Wave Pipeline Clocking

Initially, it has been thought that above the minimum at which wave pipeline constraints are satisfied, the clock period can be of any value [6],[4]. However, it has been shown by Lam [7] and Gray [8] that valid region for the clock period is discontinuous.

For simplicity, let us assume that the constraint at the output register is the determining constraint and that  $T_{SKEW_{OOD}} = 0$ . Then from Equations ?? and ??, we can derive the following equation.

$$MaxTime < N * T_{CLK} < MinTime + T_{CLK} \quad (3)$$

Equation 4 can be modified to the following form which is applicable for  $N \geq 1$ .

$$\frac{MaxTime}{N} < T_{CLK} < \frac{MinTime}{N-1} \quad (4)$$

If  $N = 1$ , it reflects a traditional scheme of pipeline where the clock period,  $T_{CLK}$  depends on  $MaxTime$  and is unbound above. The design becomes bounded below and above,  $[\frac{MaxDelay}{N}, \frac{MinDelay}{N-1}]$ , once  $N \geq 2$ .

It is easy to see from the above equations that unless the circuit is perfectly balanced,  $MaxTime = MinTime$ , the valid clock ranges will not be continuous.

The interest of this paper is the aspect of process and environment variation and their impact on delay. It has been reported in [9] that wave pipelining is up to 6-times more sensitive than conventional pipelining. In this paper, a relatively simple scheme is proposed to minimize the effects on functionality.

## III. PRIOR RESEARCH

To reduce the performance impact of the process and environmental variations, several ideas have been suggested.

The use of dynamically adaptive clock generator addresses the data delay variations by adjusting the clock rate [11]. The use of adaptive clock delay or skew introduces and adjusts the "constructive" skew between the input and the output registers of the wave pipelined logic to allow for proper clocking of the logic [11]. Another method makes use of the adaptive power supply [9]. To compensate for the propagation delay variation, the supply voltage of the wave pipelined logic is adjusted to maintain the specified delay. Biased logic gates can be used to compensate for the delay variation due to process variations [11], [9]. Extra biasing transistors are added in series to the existing transistors in both the NMOS and PMOS networks. Examples of biased logic are shown in Figure 2.

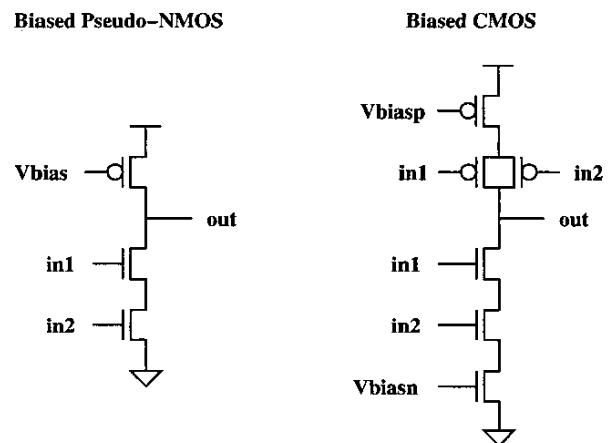


Fig. 2. Biased Logic NAND2

Approach similar to biased logic but which is applied in more macro scale is the driver current starving method [12]. It is similar in that biased voltage is applied to the design. However, only the output driver of the design is biased. Another similar approach is using a special output driver which has controlled loading [13]. Extra loading is added through shunted transistors which can be controlled by the bias voltage applied to the gate of the transistors (Figure 3).

While the adaptive clock generator approach allows for altering clock frequencies, it assumes spatial uniformity of PVT variations which does not reflect real situation. Other adaptive approaches and biased logic require manual control by the designer and/or extra voltage connections which can complicate the design.

This research proposes a simple mechanism that accounts for possible effects of PVT without the complications mentioned.

## IV. SELF DELAY CHECKING SCHEME

The proposed mechanism takes into account the effects of process and environmental variations and guarantees that

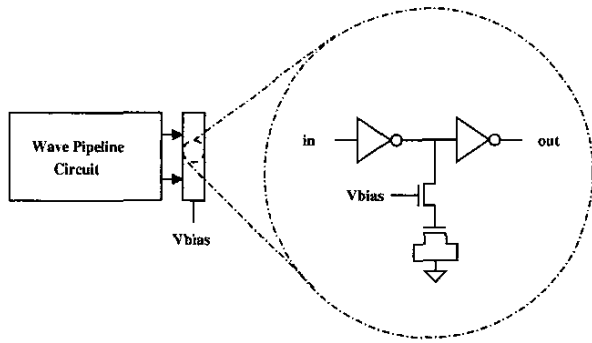


Fig. 3. Delay Compensation Using Current Starving

the chip will work even when the wave pipelined circuit gets to be slower or faster and the design is no longer aligned with the design clock.

The basic idea of this self delay checking scheme is to skew the clock of the output registers and work the wave pipeline circuit every other clock if the delta delay gets to be larger than the clock period.

#### A. Path Mirroring

In order to for this scheme to work, it is necessary to know in advance the worst and best timing paths as the delta of the two, plus some overhead, will basically determine the operating frequency of the wave pipeline design.

It should be possible to know the worst and best timing paths from the timing analysis done on the circuit during the process of delay balancing. Once the paths are singled out, the exact same path for best and worst timing should be duplicated. The pin connections of these paths are the controlling pins. These controlling pins of the gates in the paths are connected the same way as the original paths, but the leftover input pins of the gates in the paths are connected to the appropriate power rail - vdd or vss - to make them uncontrollable and thus reduce the extra wiring. Figure 4 illustrates.

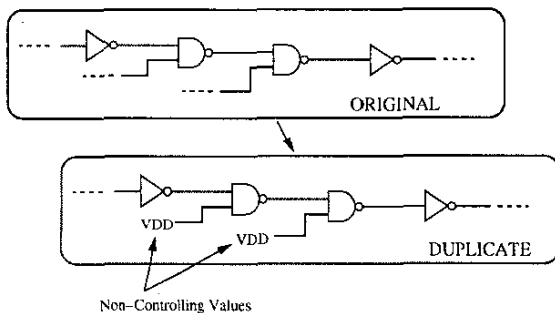


Fig. 4. Duplication of MAX/MIN Paths

Although these delay determining paths have been duplicated, it will not be very effective if they are not affected the same way by the process and environmental variations. This means that the original and the duplicate paths have to be in close proximity of their other self. This can be re-

alized by grouping the cell twins and placing them together during the layout phase.

As the cells would be sitting right next to each other, any variations that occur would theoretically impact them in the same way. The routes of these paths can be pre-routed to further mirror their topology.

#### B. Clocking Interface

Now a circuit interface with the system clock and the rest of the chip is needed to determine whether the wave pipeline circuit can operate at the system frequency or the clock needs to be tweaked to accommodate the design.

First, it needs to be determined if the path delay variation has changed such that the clocking at the output registers does not latch stable data. Figure 5 shows how this is implemented. In the figure, *SCLK* is the system clock, *LCLK* is the skewed version of the clock with system clock frequency and *WP\_CLOCK* is the locally generated clock for the wave pipelined design. Implementations of *LCLK* and *WP\_CLOCK* can be seen in Figures 6 and 7.

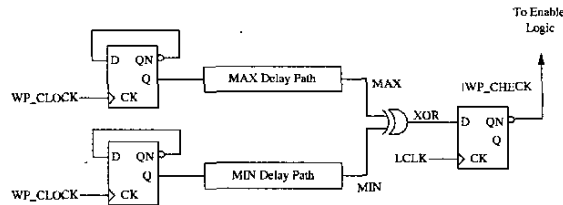


Fig. 5. Generation of Self Check Signal

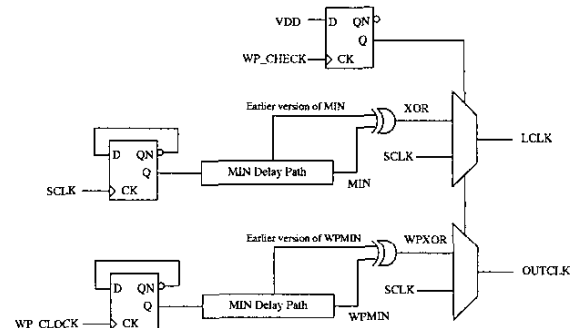


Fig. 6. Skew Generator

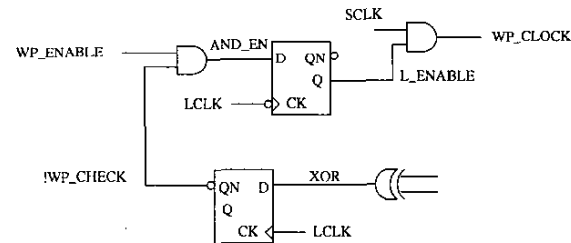


Fig. 7. Generation of WP Clock

MIN and MAX paths are the duplicates of the actual min-delay and max-delay paths. The output of these

paths are XOR-ed to create the XOR signal which gets latched on the rising edge of the system clock to generate WP\_CHECK. The initial values can be VDD or VSS as long as the two registers keep the same values to each other.

WP\_CHECK is the indicator whether the design accommodates the original system clock. If the design is still workable with the system clock WP\_CHECK will remain low. If not, the signal will start toggling at every system clock cycle.

Once the WP\_CHECK toggles, Figure 6 which is the skew generator comes into play. This circuit applies what is constituted as "constructive skew" concept to generate a skewed clock for the output registers in order to properly latch the data. WP\_CHECK is used to select between the skewed clock and the system clock.

Figure 5 shows how the MAX and MIN paths are connected and Figure 7 illustrates how the gating of the clock which goes to the wave pipeline circuit works.

Figure 8 shows how the clock is slowed by two when the design is too slow due to increased delay from process and environmental variations.

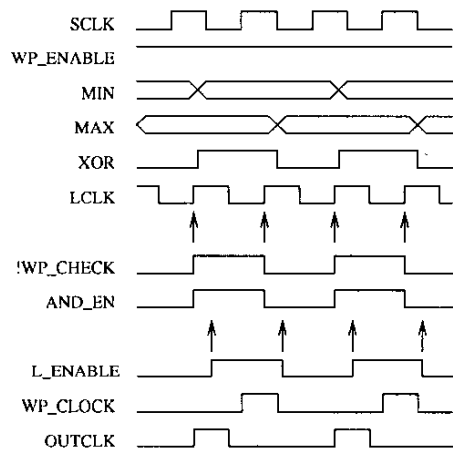


Fig. 8. Timing Waveform When WP Design is Slow

## V. RESULTS

Figure 9 shows the hspice results for the case where difference between the maximum delay path and minimum delay path have grown too large for the system clock to handle. As mentioned, the proposed circuitry recognizes that the clock frequency must be slowed down which results in WP\_CLOCK triggering every other system clock cycle.

## VI. CONCLUSION

It has been shown that, with the proposed circuitry, it is possible to adjust the local clock of the wave pipelined logic to guarantee the proper operation of the adder. The circuitry itself is simple, and with the mirroring and grouping of the routes and gates of the determinant timing paths, process and/or environmental variations are sure to be realistically reflected. A big concern of wave pipelined circuit

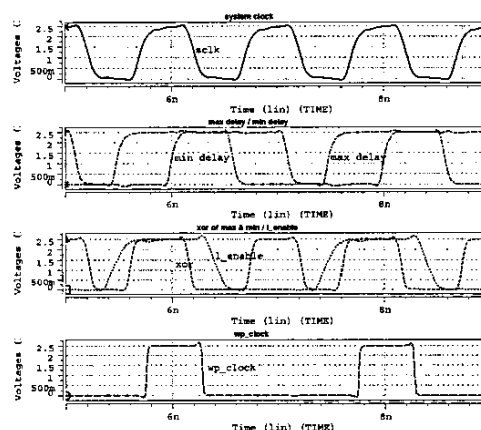


Fig. 9. Simulation Result for Large Delay Mismatch

is its proper operation in face of all the design related variations and this circuit addresses this aspect without the overhead of extra voltage sources.

## REFERENCES

- [1] W. Burleson, F. Klass and W. Liu, "Wave pipelining: A tutorial and survey of recent research", Technical Report, Stanford University, 1998.
- [2] B. Ekroot, "Optimization of Pipelined Processors by Insertion of Combinational Logic Delay", PhD Dissertation, Department of Electrical Engineering, Stanford University, 1987.
- [3] D. Wong, G. De Micheli and M. Flynn, "Designing High-Performance Digital Circuits Using Wave Pipelining: Algorithms and Practical Experiences", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol.12, No.1, January 1993, pp.25-46.
- [4] D. A. Joy and M. J. Ciesielski, "Clock Period Minimization with Wave Pipelining", IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, Vol.12, No.4, April 1993, pp.461-472.
- [5] C. T. Gray, W. Liu, W. Van Noije, T. Hughes and R. K. Cavin III, "A sampling technique and its CMOS implementation with 1 Gb/s bandwidth and 25ps resolution", IEEE Journal of Solid State Circuits, March, 1994, pp.340-349.
- [6] J. P. Fishburn, "Clock Skew Optimization", IEEE Transactions on Computers, Vol.C-29, No.5, May 1980, pp.945-951.
- [7] W. Lam, R. Brayton and A. Sangiovanni-Vincentelli, "Valid Clocking in Wavepipelined Circuits", Proceedings of IEEE Conference on Integrated Circuits Computer Aided Design, 1992, pp.518-525.
- [8] C. T. Gray, T. Hughes and R. Cavin III, "Timing Constraints for Wave-Pipelined Systems", IEEE Transactions on Computer-Aided Design, Vol.13, No.8, Aug. 1994, pp.987-1004.
- [9] K. Nowka, "High-Performance CMOS System Design Using Wave Pipelining", PhD Dissertation, Department of Electrical Engineering, Stanford University, 1995.
- [10] F. Klass, "Wave Pipelining: Theoretical and Practical Issues in CMOS", PhD Dissertation, Department of Electrical Engineering, Delft University, 1995.
- [11] D. Fan, C. T. Gray, W. Farlow, T. Hughes, W. Liu and R. Cavin, "A CMOS Parallel Adder Using Wave Pipelining", MIT Advanced Research in VLSI and Parallel Systems, 1992, pp.147-164.
- [12] D. Jeong, G. Borriello, D. Hodges and R. Katz, "Design of pll-based clock generation circuits", IEEE Journal of Solid-State Circuits, April 1987, pp.255-261.
- [13] M. Johnson and E. Hudson, "A variable delay line pll for cpu-processor synchronization", IEEE Journal of Solid-State Circuits, October 1988, pp.1218-1223.