Dynamic Flip-Flop Conversion to Tolerate Process Variation in Low Power Circuits

Mehrzad Nejat, Bijan Alizadeh, and Ali Afzali-Kusha
School of Electrical and Computer Engineering, College of Engineering, University of Tehran
Email: nejat.mehrzad@ut.ac.ir, b.alizadeh@ut.ac.ir, afzali@ut.ac.ir

***Abstract*—A novel time borrowing method called dynamic Flip-Flop conversion is presented in this paper. A timing violation predictor detects the violations halfway in the critical path and dynamically converts the critical Flip-Flop to a latch. This way, time borrowing benefits of latches are utilized in a Flip-Flop based design which is more adaptable with Computer-Aided-Design tools. The overhead of this method is smaller than that of similar methods due to the elimination of delay elements. According to the post-synthesis simulations and Monte-Carlo analysis of Spice simulations on some ITC'99 benchmark circuits, the power overhead of the proposed method is about 15% and 19% smaller than that of Soft-Edge-Flip-Flop and Dynamic-Clock-Stretching circuits respectively in a simple case of about 40% yield improvement. This overhead would be relatively even smaller for higher performance and yield improvements.**

***Keywords*—Time borrowing, Flip-Flop, Latch, Timing violation**

I. Introduction

Variation has become one of the main obstacles to performance improvement in today's nanometer technologies, especially in low power designs. Supply voltages have reduced to near and below the threshold voltage of the transistors in ultra-low-power systems. Performance variation due to global process variation alone increases by approximately 5X from nominal supply voltage to near-threshold region [1]. To avoid timing violations and achieve a specific yield in the presence of variations, a margin should be added to the minimum clock period and this stops the performance improvement. Worst-case design guarantees the correct operation of the system; but it is too conservative since the worst case does not happen most of the time. By using error detection and correction methods, higher performances are achievable without yield loss.

It is well-known that a digital circuit is usually constructed from sequential elements or Clocked-Storage-Elements (CSE) [2] and some combinational logics between them. This architecture is called pipeline which consists of some stages starting from a source CSE and ending to a destination CSE. These CSEs are either Flip-Flops (FF) or latches. There are two important timing constraints that should be satisfied in the pipeline stages: Minimum and maximum delay path constraints which are determined by the hold-time and setup-time of the CSEs respectively [2] and are formulated in the following inequalities:

$$d_i > t_{h,i} - t_{cq,(i-1)} \tag{1}$$

$$D_i < T - t_{s,i} - t_{cq,(i-1)} \tag{2}$$

978-3-9815370-2-4/DATE14/© 2014 EDAA

Where d_i and D_i denote the minimum and maximum path delays in stage i, respectively. T is the clock period and $t_{h,i}$, $t_{s,i}$ and $t_{cq,(i-1)}$ denote the hold time and setup time of destination and clock to output delay of source CSE in the stage i respectively.

Hold Time Violation (HTV) happens in minimum delay paths when the new output of the source CSE reaches the destination CSE before it can safely store the last data. Whereas, the Setup Time Violation (STV) occurs in the critical paths when the data arrives to the destination CSE so late that it cannot be stored in the corresponding clock period. This is the problem that limits the maximum clock frequency. Time borrowing is a commonly used technique for solving this problem. A critical stage can borrow some time from the following stage if it has enough timing slack. This is usually done by skewing the clock or by creating a transparency window in the destination CSE. The level sensitive behavior and small overhead of latches make them a good option for variation-tolerant systems. However, mostly because of the difficulties in timing analysis of latch based designs and more susceptibility to HTVs, the designers prefer to use FFs.

This paper proposes a novel method for time borrowing in which the critical FFs dynamically convert to latches only when a timing violation is detected. This way, by adding a simple and small circuit to critical paths and without using multiple clock signals or delay elements, time borrowing benefits of latch-based designs is utilized in a FF based design only when needed. Hence, the main contributions of this paper are as follows:

- The early timing violation predictor which is an improved version of the detection circuit represented in [3], detects the violations before the rising edge of the clock.
- Statistical analysis is done on the location of the middle node for timing violation predictor.
- A simple circuit is presented which dynamically converts the FFs in the critical path to latches so that dynamic time borrowing is done without using any delay element or multiple clock signals. That is why our method increases the performance with a small overhead.

The rest of the paper is organized as follows. Section II discusses some of the previous works on time borrowing. The Dynamic FF conversion (DFFC) method is presented in Section III. Experimental results are reported and discussed in Section IV and Section V concludes the paper.

II. Previous Works

Time borrowing has recently gained more attention due to the increased effects of variations in new technologies and extensive applications of ultra-low-power systems. Some works have focused on the use of latches in critical parts of the systems. The authors of [4] have improved the timing yield using an algorithm to replace some FFs with latches. According to the reported results, although the overhead of latches is smaller than that of FFs, the number of required resources has increased which results in an extra overhead. Pulsed latches have been used in [5] in a Field Programmable Gate Array (FPGA). By lowering the duty cycle of the clock in these latches, hold time violations has been reduced. The authors of [6] have reduced the mean critical delay by utilizing Soft Edge FFs (SEFF). They have shown that smaller delays can be achieved by increasing the softness, i.e., the transparency window size. But this results in more power and area overhead. An optimization method has been proposed in [7] that finds the optimal window sizes of SEFFs to get the minimum power-delay product (PDP). Clock skewing or clock stretching is another technique that has been used in [3], [8]. The authors of [3] have proposed a circuit that dynamically detects timing violations and feeds a delayed clock signal to the destination FF while [8] presents an optimization algorithm that finds the optimal delay values of each FF. It uses variable delay chains and assigns a particular clock signal to each FF so that slacks can be transferred between different stages. Razor is a hybrid technique for dynamic detection and correction of timing errors [9]–[11]. In this technique, timing errors are detected in FFs and corrected by micro-architectural recovery mechanisms.

III. Dynamic Flip-Flop Conversion (DFFC)

DFFC consists of two parts: violation prediction and FF conversion as shown in Fig. 1. When a timing violation is predicted, FF converter (FFC) converts the FF to a latch so that the data arriving after the edge of the clock can still pass the destination CSE and then converts it back to a FF when there is no violation. The timing diagram in Fig. 2 clarifies the behavior of this system. In this figure, the arrows indicate the propagation of data from source to the "Mid" node and destination CSE. The red arrows indicate a data arriving later than the clock period. The change of the "Mid" node data in low phase of the clock issues an error signal which converts the destination FF to a latch. In the other cases, the "Err" signal is low and the FF remains unchanged.

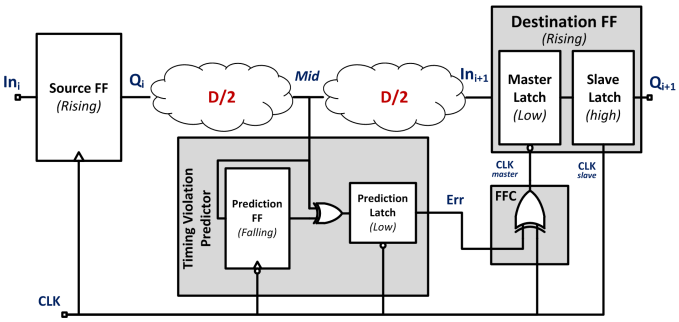


Fig. 1: The structure of the DFFC system

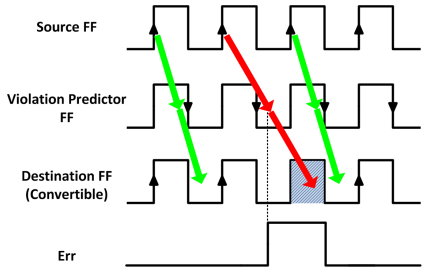


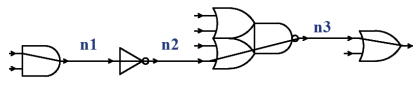
Fig. 2: DFFC behavior. Red arrows indicate a data arriving late and green ones represent a data which arrives in proper timing and the gray-dashed area depicts a transparency window.

A. Violation Prediction Circuit

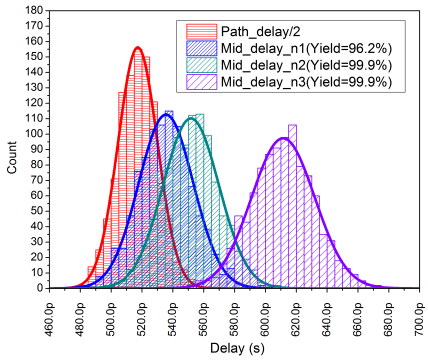
For the FFC to work properly, timing violation should be detected before the rising edge of the clock. Such detection circuit has been presented in [3]. This circuit was used here with some improvements. The Prediction FF captures the data on "Mid" node at the falling edge of the clock. When there is no timing violation, the data on "Mid" node should become stable before the falling edge. Otherwise, it would change after this edge and the captured data and the current data of "Mid" node would become different and the XOR output would become high which indicates a timing violation. Here, instead of an active high latch [3], a falling edge FF is used in the prediction circuit to keep the error signal until the next falling edge. Otherwise, this signal would be lost at the rising edge of the clock. An active low latch is also used to avoid the evaluation of "Err" signal in the high phase of the clock, so to prevent false errors.

B. Middle Node Selection

The timing violation prediction technique is based on locating the middle node in the critical path. Using timing analysis tools, like Synopsys Prime Time, the designer can calculate the delay values along the critical path, and the node to which the delay is almost half the total path delay can easily be located. Usually, there is no node with a delay exactly equal to "path-delay/2". If the "Mid" delay is smaller than this amount, the predictor might fail to detect some violations. On the other hand, if it is bigger, there could be false errors. False errors would just convert the critical FF unnecessarily and this would not cause any problem, while the undetected violations would result in STVs. So it is better to select a "Mid" node with a delay slightly bigger than "path-delay/2". The authors of [3] have made the assumption that there is spatial correlation in variations and the "Mid" delay always remains "path-delay/2". Here, Monte Carlo analysis has been done on the extracted Spice netlist of the critical path of ITC'99-b14 benchmark in the presence of both global and local random variations. The middle part of this path is shown in Fig. 3a and the delay profile to the three depicted nodes along with the total path delay divided by 2 is illustrated in Fig. 3b. The corresponding yields are also presented in this figure. It is clear that the probability of "n1" delay being smaller than "path-delay/2" is more than that of "n2" and "n3". So, choosing "n1" as the "Mid" node results in 96.2% yield, while this amount increases to 99.9% for "n2" and "n3". So in this case, "n2" would be the best choice for the "Mid" node. The designer should perform such analysis to achieve the highest yield.



(a)



(b)

Fig. 3: The middle part of the critical path of ITC'99-b14 circuit (a) and the delay profile of 3 middle nodes and the total path delay divided by 2 (b)

C. Flip-Flop Conversion (FFC) Circuit

Master-Slave FFs are widely used in digital circuits. These FFs are constructed from an active low and an active high latch. Here, a simple method is presented to convert a FF to a latch. By feeding the inverted clock signal to the master part of the FF, it would change to two consecutively connected similar latches which act as a single latch. As depicted in Fig. 1, the FFC circuit is simply a logical XOR gate. If a timing violation is predicted and the "Err" signal is high, the XOR would feed the inverted clock to the Master latch of the FF.

D. Advantages and Issues

To the best of our knowledge, in almost all of the time borrowing methods except those applied to latch-based designs, delay elements are used whether for creating a transparency window or for skewing the clock. Some of them set the delay values at design time [7] and some have used variable delay buffers with programming capabilities to program different delay values on the chip [8]. To achieve higher frequencies, bigger delays are required which adds a very large power and area overhead to the system. Delay chains, also consume a large leakage power which is very important specially in new technologies and in sub/near threshold designs [12]. Error detection techniques like Razor [9]–[11] also require some micro-architectural recovery overheads. DFFC On the other hand, only uses simple logical circuits and no delay element or multiple or nonstandard clock signals and it dynamically predicts the STVs and prevents them from happening.

HTV is a problem in all of the time borrowing methods and even error detection techniques like Razor [10]. This problem could be more serious in latch-based designs and DFFC because of the big transparency window. There are methods for addressing this problem like adding delay elements to minimum delay paths or reducing the clock duty-cycle [5], [7], [10]. The probability of HTVs is smaller in DFFC because the transparent window is created only if necessary. If a STV is happening, there is a high chance that all the paths ending to the critical FF are slow and thus HTV is less probable. However, the use of HTV prevention techniques is necessary with DFFC. Combinational feed-back loops is another problem in transparent time-borrowings and in DFFC which is treated exactly like HTVs.

IV. Experimental Results

To evaluate the efficiency of DFFC method, five benchmark circuits of ITC'99 [13] were synthesized using Nangate open cell library [14]. This library uses 45 nm, low threshold voltage, predictive models. After synthesis, the Spice netlist of the most critical path of each benchmark was extracted and Spice circuit simulation was performed on it. Both global and local random variations were modeled on the threshold voltage of the circuits which is the most important factor when analyzing delay variations. The results of Monte-Carlo analysis of these circuits are reported in Table I. Area overheads were measured by the synthesis tool and they are reported in this table as a percentage of the area of the whole benchmark circuit. Power overheads on the other hand, were measured by Spice simulations and they are reported as a percentage of the power of the extracted critical path. That is because Spice simulations give more precise results on power, but not area. According to these results, almost all of the STVs have been prevented by using DFFC.

To compare the DFFC with other similar methods, two structures of Dynamic Clock Stretching (DCS) [3] and SEFF [6], [7] was used on the critical path of b14 benchmark along with DFFC. For the DCS, the same detection circuit was used as DFFC (Fig. 1) and the only difference was the clock skewing circuit which was exactly as presented in [3]. SEFF was realized by adding a delay element before the clock of

TABLE I: Yield improvement of five ITC'99 benchmark circuits with the use of DFFC method and Power and Area overhead of each DFFC circuit

Benchmark	Functionality	Num. of FFs	Num. of Gates	Yield without DFFC	Yield with DFFC	Power Overhead of each DFFC*	Area Overhead of each DFFC**
b05	Elaborate the contents of a Memory	34	574	38.4%	100%	12.2%	1.871%
b12	1-player game (guess a sequence)	121	1006	84.2%	100%	32.2%	0.890%
b14	Viper processor (subset)	245	5678	60.5%	99.9%	19.6%	0.102%
b15	80386 processor (subset)	449	7577	71.0%	100%	4.8%	0.198%
b22	A copy of b14 and two modified versions of b14	703	18086	70.5%	100%	3.5%	0.070%

*DFFC power overhead is measured by Spice simulation and reported as a percentage of the critical path power
** DFFC area overhead is measured by synthesis tool and is reported as a percentage of the whole benchmark area.

TABLE II: Comparison of three different time borrowing methods applied to the critical path of ITC99-b14 benchmark circuit

Time Borrowing Method	No Time Borrowing	DFFC	DCS [3]	SEFF [7]
Yield	60.5%	99.9%	99.6%	96.7%
Power Overhead*	NA	19.6%	24.3%	23.1%
Area Overhead**	NA	0.102%	0.130%	0.070%
Delay element	NA	NA	75 ps	108 ps

*Power overhead is reported as a percentage of the critical path power.
** Area overhead is reported as a percentage of the whole benchmark area.
NA: Not Applicable

the master latch of the critical FF. The results are presented in Table II. The amount of delay used in SEFF and DCS is also reported. These results show 19% and 15% smaller power overhead of DFFC compared to DCS and SEFF respectively, in a simple case of about 40% yield improvement. DFFC also has 21% smaller area overhead compared to DCS while the area overhead of SEFF is 31% smaller than that of DFFC in this case. This is because of the lack of dynamic error detection circuit in SEFF.

It is obvious that for achieving higher frequencies, bigger delay values are required in techniques like DCS and SEFF. Fig. 4 shows the power and area overheads of the three time-borrowing methods for three different maximum achievable frequencies while preserving 99% yield. These methods are implemented on the most critical path of a b12 ITC'99 benchmark. The overhead of DFFC is constant since it does not use any delay element. According to this figure, the power and area overhead of DFFC is smaller than that of SEFF for maximum frequencies higher than 1.566 GHz and 1.765 GHz which are respectively equal to 13% and 27% improvement in the maximum frequency of the system without time borrowing (1.389 GHz). DFFC overheads are always smaller compared to DCS method.

V. Conclusion and Future Works

A novel time borrowing technique called DFFC was presented in this work. DFFC uses a timing violation predictor to convert the critical FFs to latches when a violation is about to happen. This way, the structure of the system is still based on FF which is more adaptable with Computer Aided Design (CAD) tools and the time borrowing benefits of latches are utilized only when needed. The overhead of this method is smaller compared to similar methods such as DCS and SEFF especially due to the elimination of delay elements and it would become relatively even smaller when reducing the clock period further for higher performance improvements. The probability of HTVs is smaller in DFFC compared to some of the other time borrowing methods due to its dynamic behavior. However, it is still a problem because of the big transparency window when the FF is converted to a latch. Currently, this problem should be addressed by adding delay elements to minimum paths or reducing the clock duty cycle like other time-borrowing methods.

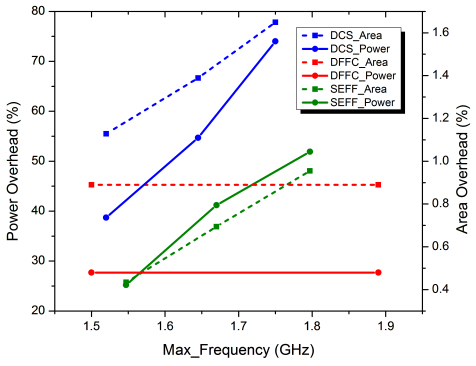


Fig. 4: Power and Area overhead of different time borrowing methods for different maximum achievable frequencies when preserving 99% yield in b12 benchmark

We plan to improve the structure of DFFC in our future works to alleviate the HTV problem. Furthermore, for achieving higher frequencies, more DFFC circuits should be added to the system because more paths become critical. Although DFFC doesn't use delay elements in its structure, they should be added to the minimum delay paths to prevent HTVs. An optimization algorithm can also be developed in our future works to find the optimal amount of frequency improvement using DFFC method.

References

[1] R. G. Dreslinski *et al.*, "Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 253–266, 2010.

[2] V. G. Oklobdzija *et al.*, *Digital system clocking: high-performance and low-power aspects*. Wiley-IEEE press, 2005.

[3] V. Mahalingam *et al.*, "Dynamic clock stretching for variation compensation in vlsi circuit design," *J. Emerg. Technol. Comput. Syst.*, vol. 8, no. 3, pp. 1–13, 2012.

[4] Y. Chen and Y. Xie, "Tolerating process variations in high-level synthesis using transparent latches," in *Proc. ASP-DAC*. IEEE Press, 2009, pp. 73–78.

[5] B. Teng and J. H. Anderson, "Latch-based performance optimization for field-programmable gate arrays," *IEEE Tran. Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 5, pp. 667–680, 2013.

[6] V. Joshi *et al.*, "Soft-edge flip-flops for improved timing yield: design and optimization," in *Proc. ICCAD*. 1326214: IEEE Press, 2007, pp. 667–673.

[7] M. Ghasemazar and M. Pedram, "Optimizing the power-delay product of a linear pipeline by opportunistic time borrowing," *IEEE Tran. Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 10, pp. 1493–1506, 2011.

[8] A. Tiwari *et al.*, "Recycle: pipeline adaptation to tolerate process variation," *SIGARCH Comput. Archit. News*, vol. 35, no. 2, pp. 323–334, 2007.

[9] D. Ernst *et al.*, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proc. 36th annual IEEE/ACM International Symposium on Microarchitecture*. 956571: IEEE Computer Society, 2003.

[10] S. Das, "Razor: A variability-tolerant design methodology for low-power and robust computing," Ph.D dissertation, 2009.

[11] D. Bull *et al.*, "A power-efficient 32 bit arm processor using timing-error detection and correction for transient-error tolerance and adaptation to pvt variation," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 18–31, 2011.

[12] H. Kaul *et al.*, "Near-threshold voltage (ntv) design: opportunities and challenges," in *Proc. 49th Annual Design Automation Conference*. 2228572: ACM, 2012, pp. 1153–1158.

[13] (2013) Itc'99 benchmarks. [Online]. Available: http://www.cad.polito.it/downloads/tools/itc99.html

[14] (2013) Nangate freepdk45 generic open cell library_v1.3_2010_12. [Online]. Available: http://www.si2.org/openeda.si2.org/projects/nangatelib/