A CMOS flip-flop featuring embedded Threshold logic functions

Marius Padure, Sorin Cotofana, and Stamatis Vassiliadis
Computer Engineering Laboratory
Delft University of Technology
Mekelweg 4, 2628 CD, Delft, The Netherlands
Email: {marius,sorin,stamatis}@ce.et.tudelft.nl

Abstract— **This paper describes a semi-dynamic CMOS flip-flop family featuring embedded Threshold Logic functions. First, we describe the new Threshold Logic flip-flop concept and circuit operation. Second, we present the concepts of embedded Threshold logic and run-time reprogrammability. Finally, it is proved by Spice simulation results that wide (up to 8 inputs) AND/OR Boolean functions can be embedded in the newly proposed Threshold Logic flip-flop with up to 54% less total latency when compared with the conventional flip-flop featuring the same embedded Boolean functions. Therefore proposed flip-flop is very attractive for high-performance pipelined arithmetic units.**

Keywords—**CMOS digital design, flip-flops, Threshold logic, computer arithmetic**

I. Introduction

The continual push for higher clock rates and higher performance has led microprocessor designers in recent years to design super-pipelined machines with multiple functional units that can execute operations concurrently. High clock rates in these machines are often achieved with high granularity pipelining, for which there are relatively few levels of logic gates per pipeline stage. One direct consequence of this design trend is that pipeline overhead is becoming more significant. This pipeline overhead is primarily due to the latency of the flip-flop or latch used and the clock skew of the system. While the clock skew varies, the latency of the flip-flop cannot be hidden. As an example, assuming that a flip-flop latency is four gates delay and that the clock cycle of a state-of-the-art microprocessor is 20 gates delay, the flip-flop overhead amounts 20% of the cycle time. This is a substantial penalty that degrades the overall performance of the system, since no useful logic operation is performed on the data when is being latched.

The idea of incorporating logic functions into storage elements to improve the critical path latency have emerged in the last decade as a potential alternative for meeting the cycle time goal of processors [5], [1]. The challenge has been to develop latch structures that can embed logic functions efficiently, in terms of both total latency (defined as the sum of setup time and clock-to-output latency) and area. While previously published flip-flops have embedded simple Boolean functions (AND/OR of few inputs), no attempt has been done to incorporate Threshold Logic functions into the storage elements.

It is well known that TL is fundamentally more powerful that Boolean logic since the TL gate (when envisioned as combinatorial element) can perform more complex and wider functions than the usual Boolean CMOS gates (e.g., NAND, OR) can. More formally, a Threshold Logic Gate (TLG) is defined as an n-input processing element such that its output performs the following Boolean function[1]:

$$F(X) = sgn\{\mathcal{F}(X)\} = \begin{cases} 1, & if \ \ \mathcal{F}(X) \geq 0 \\ 0, & if \ \ \mathcal{F}(X) < 0 \end{cases} \quad (1)$$

$$\mathcal{F}(X) = \sum_{i=0}^{n-1} \omega_i \cdot x_i - T \quad (2)$$

where X=$[x_0, x_1, \ldots, x_{n-1}]$, Ω=$[\omega_0, \omega_1, \ldots, \omega_{n-1}]$ and T are the set of Boolean input variables, the set of fixed signed integer weights associated with data inputs, and the fixed signed integer threshold, respectively [2].

Several recent theoretical investigations [6], [7] have indicated that computer arithmetic building blocks (e.g., adders and multipliers) can be implemented in TL with smaller number of logic gates and fewer logic stages when compared with traditional Boolean logic counterparts. Therefore, embedding TL functions in the storage elements may have a direct impact over the pipeline overhead.

In this paper we present a new class of flip-flop featuring embedded Threshold Logic functions to reduce the pipeline overhead. The main features of the basic design are short latency and a single phase clock scheme. Furthermore, this flip-flop has the capability of incorporating reconfigurable Threshold Logic functions with a small total latency when compared with conventional flip-flops featuring embedded functions capability. This feature greatly reduces the pipeline overhead, since each flip-flop can be

[1] All the operators are algebraic.

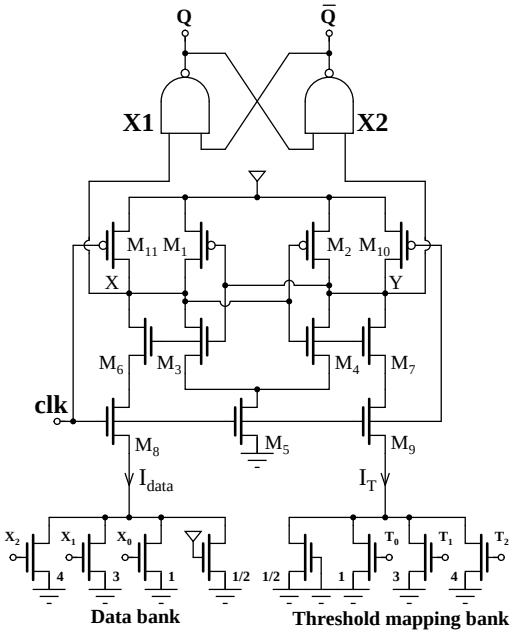

Fig. 1. Flip-flop with embedded Threshold functions

viewed as a special logic gate that serves as a synchronization element as well. Taken together, these features make the flip-flop presented in this paper well suited for high-performance microprocessor designs (e.g., computer arithmetic building blocks). It is proved in this paper by Spice simulation results that wide (up to 16 inputs) AND/OR Boolean functions can be embedded in the newly proposed Threshold Logic flip-flop with up to 53% less total latency when compared with the conventional flip-flop featuring embedded Boolean functions.

The paper is organized as follows: Section 2 explains the basic operation of the proposed Threshold Logic flip-flop. The concept of run-time reprogrammability is presented and supported by Spice simulations. Section 3 presents the simulation results and comparisons; Section 4 presents some concluding remarks.

II. Threshold Logic flip-flop

A. Basic operation

A schematic diagram of the Threshold Logic flip-flop (TLFF) is presented in Figure 1. The circuit is composed of a semi-dynamic front-end comprising a differential current-switch Threshold Logic gate (DCSTL) [4] followed by a static back-end comprising an RS latch. DCSTL front-end comprises a fast latched comparator and two parallel-connected sets of unit nMOS transistors, referenced herein as input data bank and threshold mapping bank. The nMOS transistors from the threshold mapping bank have the gates hardwired to ground or power supply. With respect to the circuit from Figure 1, the TLFF has 3 data inputs and 3 threshold mapping inputs. The data inputs, X_0, X_1, X_2, and the threshold mapping inputs, T_0, T_1, T_2, have the weights 1, 3, 4 respectively. The weights are implemented using parallel connected sets of 1, 3 and 4 unit transistors respectively.

The total conductances of the transistor banks are compared each other by the latched comparator and therefore the node X is logic zero if the current generated by the data bank is greater than the current generated by the threshold mapping bank and logic one otherwise. Please note that, by design, the data bank is prevented to have similar conductance with the threshold mapping bank, when the threshold is reached, since an nMOS transistor with weight 0.5 is always on. This prevents the latch comparator entering in a metastable state.

The circuit in Figure 1 operates as follows. On the falling edge of the clock, the flip-flop enters in *precharge phase*. Therefore, M_{10}, M_{11} are on, nodes X and Y are precharged high and the outputs Q and $\overline{Q}$ hold their previous evaluation values; since X and Y are high, M_6, M_7 are on pulling their sources to weak high level. On the rising edge of the clock, the flip-flop enters the *evaluation phase*. Therefore, $M_5, M_{8,9}$ are on and M_6, M_7 (shutoff devices) start drawing currents from nodes X and Y. If $I_{data} \geq I_T$ then the voltage at node X will start to drop faster than than the voltage at node Y. Therefore, X crosses first the latch switching threshold which regenerates rapidly to X low and Y high, causing Q high. Conversely, if $I_{data} < I_T$ then Y low and X high, causing Q low. At the end of the evaluation phase, the high-rising node among X and Y will be decoupled from being connected to ground by one of the shutoff transistors M_6, M_7 going off. Therefore no DC power is dissipated at the end of the evaluation phase. Additionally, any change on the inputs after the gate has ended the evaluation will not affect nodes X and Y and consequently TLFF is an edge-triggered flip-flop.

B. Embedding Threshold Logic functions

One distinctive advantage of the proposed TLFF is that complex TL functions can be embedded easily. Indeed, most logic functions available in Domino logic, such as OR/AND functions can be embedded in TLFF. Additionally, in comparison with Domino logic, wide OR/AND and their complements can be incorporated with no prohibitive latency. While latency is increased, the merger allows the elimination of one or more levels of logic from the path leading to the flip-flop. The result is a reduction of the overall latency of the circuit employing such a flip-flop.

With regard to Figure 1, an 8-input AND function can be implemented in TLFF by mapping all threshold mapping

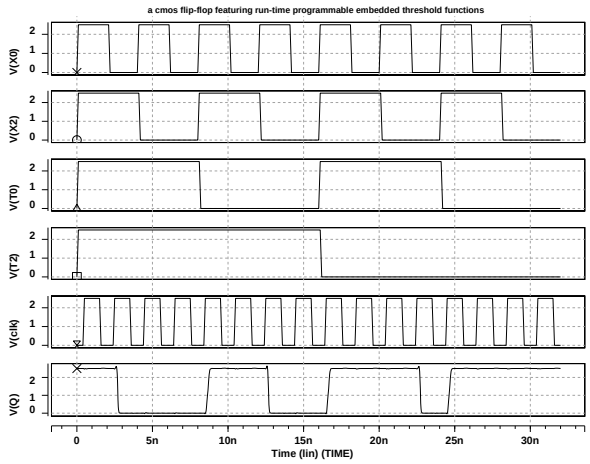

Fig. 2. Spice waveforms showing a TLFF having a run-time reconfigurable TL function

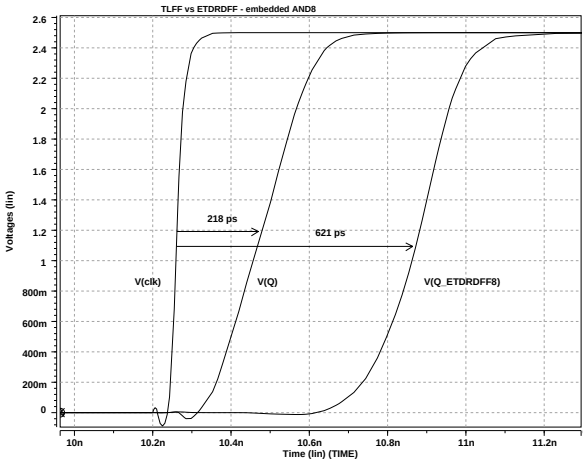

Fig. 3. Spice waveforms for TLFF with embedded 8-input AND function

inputs to V_{dd}. Therefore, $T = 8$ and all data inputs have to be logic one (see Equation 1) in order to have a logic one output. An 8-input OR function can be implemented with $T = 1$ and consequently, only one data input is necessary to be logic one in order to have a logic one output.

C. Run-time reconfigurability

Another attractive advantage of TLFF is the ability to change between two evaluations the TL function embedded in TLFF. This property comes from the fact that, in contrast with other TL gates (e.g., [3]), threshold mapping inputs, T_0, T_1, T_2 are accessible externally and can be treated as data inputs with negative weights. In Figure 2 there are presented the Spice waveforms of a reconfigurable TLFF as in Figure 1 having applied the following set of input vectors: $[X_0, X_1, X_2] = \{[1,1,1], [0,1,1], [1,1,0], [0,1,0]\}$ while threshold T is reprogrammed each four clock cycles as follows: $T = 8 \rightarrow 7 \rightarrow 5 \rightarrow 3$. Please note, that TLFF from Figure 1 has Ω=$[1,3,4]$ and $T \in \{0,1,3,4,5,7,8\}$.

III. Simulation results and comparisons

To evaluate the potential performance and cost of TLFF, an 8-input TLFF, as shown in Figure 1, was designed and simulated in $0.25\mu m$ feature size CMOS technology. For comparison purposes, the edge triggered dual-rail dynamic flip-flop (ETDRDFF) employed in UltraSPARC-III microprocessor [1] was simulated in similar conditions with Threshold Logic flip-flop. The ETDRDFF having no embedded logic function has $200ps$ delay.

In order to evaluate the potential impact of embedding Boolean functions over the total latency, the following simulation setups were considered:

1. an AND function with 2, 4, and 8-inputs respectively. The simulated TLFF had $T = 2, 4, 8$ and was compared with an ETDRDFF having embedded 2, 4, and 8-input AND functions respectively. Figure 3 presents the Spice waveforms for the TLFF and the ETDRDFF outputs having embedded an 8-input AND function. Please note that the $clk - Q$ delay of the TLFF is only 218 ps, compared with 621 ps for ETDRDFF.
2. an OR function with 2, 4, and 8-inputs. The simulated TLFF had $T = 1$ and was compared with an ETDRDFF having incorporated 2, 4, and 8-input AND functions respectively.

To emphasize the advantage of incorporating logic functions into the storage elements in Tables 1 and 2 there are analyzed also the discrete versions (a static gate followed by an ETDRDFF) of the ETDRDFF having incorporated AND/ORs of 2, 4, and 8 inputs.

Tables 1 and 2 show the total latency, normalized latency and dynamic power dissipation ($@100MHz$) figures respectively for the previous two simulation setups. The results are presented for typical corner, $@V_{dd} = 2.5V$, 27°C.

Table 1, shows that TLFF with embedded 8-input AND/NAND function is 65% faster but dissipates 46% more dynamic power. When comparing a TLFF having embedded an 8-input OR/NOR function ($T = 1$) with a similar ETDRDFF, Table 2 shows that TLFF is 53% faster but dissipates 51% more dynamic power. Since AND/OR functions embedded in ETDRDFF require long chains of nMOS transistors, the delay is seriously degraded. In contrast, TLFF implements AND/OR Boolean functions by parallel connected sets of unit nMOS transistors and therefore is faster.

TABLE I
SPEED COMPARISON OF TLFF VERSUS ETDRDFF (EMBEDDED AND DISCRETE) - AND FUNCTION

Flip-flop Type	Latency $[ps]$	Norm. latency	Power $[\mu W]$
TLFF-AND2	260	2.38	105
TLFF-AND4	227	2.73	106
TLFF-AND8	218	2.84	108
ETDRDe-AND2	250	2.48	41
ETDRDe-AND4	366	1.69	46
ETDRDe-AND8	621	1.00	58
ETDRDd-AND2	320	1.94	42
ETDRDd-AND4	421	1.47	44
ETDRDd-AND8	756	0.82	47

TABLE II
SPEED COMPARISON OF TLFF VERSUS ETDRDFF (EMBEDDED AND DISCRETE) - OR FUNCTION

Flip-flop Type	Latency $[ps]$	Norm. latency	Power $[\mu W]$
TLFF-OR2	310	2.13	109
TLFF-OR4	314	2.11	112
TLFF-OR8	316	2.09	114
ETDRDe-OR2	253	2.62	41
ETDRDe-OR4	366	1.81	46
ETDRDe-OR8	663	1.00	56
ETDRDd-OR2	334	1.98	42
ETDRDd-OR4	437	1.11	43
ETDRDd-OR8	627	1.05	45

Tables 1 and 2 show also that the TLFF latency depends mainly on the value of T, for the same fan-in. For example, an 8-input TLFF with $T = 1$ is with 44% slower than a the TLFF with $T = 8$.

With regard to Table 1, an ETDRDFF with embedded 2-input AND is with between 32% and 60% faster than the same flip-flop having embedded a 4-input and 8-input AND respectively. Table 2 shows that an ETDRDFF having incorporated a 2-input OR has between 45% and 62% less total latency than the same flip-flop having embedded a 4-input and 8-input OR respectively. Since a TLFF implementing an 8-input AND has $T = 8$, more current is drawn by the nMOS transistors from the threshold mapping bank and therefore a TLFF with 8-input AND is faster than a TLFF with 2-input AND. Indeed, the simulation results from Table 1 show that an 8-input AND TLFF is with 17% faster than a 2-input TLFF and with 5% faster than the 4-input TLFF. In contrast, a 2-input OR TLFF is marginally faster than the 4-input OR and 8-input OR respectively since the same amount of current is drawn from the power supply ($T = 1$).

Table 1 shows that the 2-input AND discrete version is with 28% slower when compared with the embedded version while the 8-input AND discrete version is with more than 21% slower. With regard to Table 2, a 2-input OR discrete version is with 32% slower than the embedded version while the 4-input OR discrete version is with more than 15% slower.

IV. CONCLUSIONS

A semi-dynamic CMOS flip-flop family featuring embedded Threshold Logic functions was presented. First, we described the new Threshold Logic flip-flop operation and the concepts of embedded Threshold logic and run-time reprogrammability. Subsequently, was proved by Spice simulation results that wide (up to 8 inputs) AND/OR Boolean functions can be embedded in the newly proposed Threshold Logic flip-flop with up to 54% less total latency when compared with the conventional flip-flop featuring embedded Boolean functions. Therefore proposed flip-flop is very attractive for high-performance pipelined arithmetic units.

REFERENCES

[1] F. Klass. Semi-dynamic and dynamic flip-flops with embedded logic. *Symposium on VLSI Circuits*, pages 108–109, 1998.

[2] S. Muroga. *Threshold logic and its applications*. Wiley and Sons Inc., 1971.

[3] H. Özdemir, A. Kepkep, Y. Leblebici, and U. Çilingiroglu. A capacitive Threshold logic gate. *IEEE Journal of Solid-State Circuits*, 31(8):1141–1150, August 1996.

[4] M. Padure, S. Cotofana, C. Dan, M. Bodea, and S. Vassiliadis. A new latch-based Threshold logic family. *Proceedings of International Semiconductor Conference, CAS2001*, 2:531–534, October 2001.

[5] H. Partovi, R. Burd, U. Salim, F. Weber, L. DiGregorio, and D. Draper. Flow-through latch and edge-triggered flip-flop hybrid elements. *IEEE International Solid-State Circuits Conference*, pages 138–139, 1996.

[6] S. Vassiliadis and S. Cotofana. 7|2 counters and multiplication with Threshold logic. *Proceedings of* 30^{th} *Asilomar Conference on Signals, Systems and Computers*, pages 192–196, November 1996.

[7] S. Vassiliadis, S. Cotofana, and K. Bertels. 2-1 addition and related operations with Threshold logic. *IEEE Transactions on Computers*, 45(9):1062–1068, September 1996.