An Asynchronous FPGA TH*x*2 Programmable Cell for Mitigating Side-Channel Attacks

John M. Emmert
Dept. of EECS
University of Cincinnati
Cincinnati, OH, USA
john.emmert@uc.edu

Anvesh Perumalla
Dept. of EECS
University of Cincinnati
Cincinnati, OH, USA
perumaak@mail.uc.edu

Luis Concha
Dept. of EECS
University of Cincinnati
Cincinnati, OH, USA
luis.concha@uc.edu

***Abstract*—One approach to mitigate side-channel attacks (SCAs) is to use clockless, asynchronous digital logic. To simplify this process, we propose a unique asynchronous FPGA based on a new TH*x*2 programmable threshold cell. At a minimum, FPGAs require a programmable logic cell that can implement a complete set of logic so that it can be connected through the programmable interconnect network to form any digital system. To meet that criteria, we take advantage of CMOS transistors to implement a programmable TH*x*2 threshold cell capable of performing both TH12 and TH22 asynchronous operations. Our complete sixteen transistor FPGA cell includes eight transistors to implement the base TH*x*2 threshold operation, three transistors to switch between the TH12 and TH22 modes, and five memory cell transistors for mode storage. Our unique minimal transistor, programmable TH*x*2 implementation enables formation of a complete set of asynchronous threshold gates and a complete set of standard combinational logic functions. The symmetric nature of the FPGA cell, in regard to the number of transistors (eight NMOS and eight PMOS), makes it ideal for a four row by four column transistor grid with a nearly square, easily array-able layout. It should be noted our TH*x*2 cell is highly compact and suitable for implementing a clockless, asynchronous FPGA.**

***Index Terms*—asynchronous, security, assurance, trust, side-channel attacks, FPGA, field programmable gate array**

I. INTRODUCTION

Synchronous or clocked integrated circuit (IC) based systems are susceptible to side-channel attacks (SCAs), especially if fabricated at an untrusted foundry where a malicious Trojan circuit could be inserted. By leveraging Trojan circuits and monitoring power consumption, electromagnetic radiation, or other IC characteristics, a malicious, untrusted agent or entity can compromise security and steal sensitive information like credit card numbers and secret keys [1].

One approach to mitigate SCAs is **clockless asynchronous** digital design [2]. Clockless asynchronous logic in the form of a Field Programmable Gate Array (FPGA) technology offers a synergistic set of defenses against SCA attacks to include distributed, difficult to monitor internal switching and very regular structures that make it difficult to add Trojans during IC fabrication. To simplify asynchronous design and improve capabilities, we propose a unique FPGA architecture based on our TH*x*2 cell shown in Fig. 1 [3]. In Fig. 1, the dotted lines represent the connections that enable the TH12 and TH22 modes of the TH*x*2 cell.

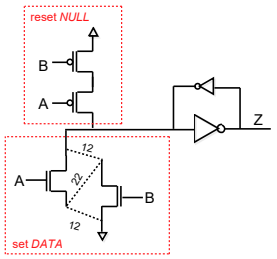


Fig. 1: TH12 and TH22 modes of the TH*x*2 cell [3].

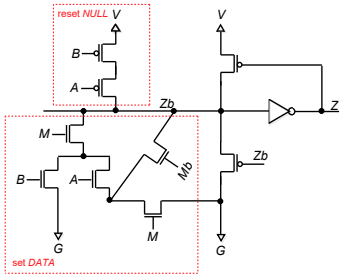


Fig. 2: Transistor diagram of the TH*x*2 base cell.

In this paper, we leverage our TH*x*2 cell architecture and **describe** the development and operation of the compact multi-mode programmable FPGA TH*x*2 cell shown (Fig. 2). The value stored in the programming memory cell (not shown) sets the value of *M* and *Mb*. A value of *M* = '1' (*Mb* = '0') sets the mode of the TH*x*2 cell to TH12, and a value of *M* = '0' sets the mode to TH22. It should be **noted** that our programmable TH*x*2 cell includes mode storage, is highly compact and suitable for tight 2-D arraying required for FPGAs.

II. BACKGROUND

In this section we provide a brief overview of SCAs and asynchronous circuits.

A. Side-Channel Attacks

One way to describe a Trojan circuit is an *extra circuit added by the manufacturer during the IC fabrication process*. Trojan circuits can be used in a positive way to provide feedback to the manufacturer and designer on the manufacturing

process, but they can also be used maliciously to leak private or secret information during normal IC operation. Several types of Trojans have been developed that require minimal area overhead and are very difficult to detect either during regular IC testing or IC reverse engineering [4].

SCAs don't directly monitor signal values. Instead, they use indirect measures to exploit existing or added (Trojan) circuitry to steal secret or private information. SCAs can leverage electromagnetic radiation, temperature variations, power use, or other characteristics to indirectly back out data during normal IC operation. A common example used to steal secret keys is a power based SCA that uses a Malicious Off-chip Leakage Enabled Side-channel (MOLES) Trojan Circuit [5].

The MOLES circuit consists of a pseudorandom number generator (PRNG), some XOR gates, and some added capacitive loads. The low area overhead PRNG can be added using a simple linear feedback shift register, or often the MOLES circuit can take advantage of existing PRNG found on most digital processors. The extra capacitive loads and XOR gates are virtually unnoticeable, and their operation looks like noise during normal system testing. A weakness often exploited during a power based SCA is synchronized power spikes caused by combinational switching during synchronized clock cycles. An untrusted agent or entity can collect synchronized sets of transient power readings over long periods of time, and use common signal processing techniques to detect secret keys, $K = k_1 k_2 k_3 ... k_n$, stored in the firmware [5]. The covertly inserted Trojan circuit and the exploitation of the synchronized power readings are one example of a power based SCA.

B. Asynchronous Logic

There are many types of asynchronous design techniques ranging from locally clocked to completely clockless, and each has its own advantages and disadvantages [6]. One type of clockless logic circuit is Null Convention Logic (NCL) [7]. The NCL circuit works well for data flow designs because data flows through NCL networks in waves. A data wave is only processed when all input data is available, so it is self-timed. Since processing only occurs when data is available, there are no timing assumptions, and thus this guarantees data sequencing and correct data arrival at the receiver under varying gate, process and wire delays [7]. The NCL circuit scheme uses multi-wire encoding. One wire represents a logic '1' and one wire a logic '0.' For example, a dual rail connecting signal *A* has a logic '1', *A*_1, rail and a logic '0', *A*_0, rail.

The backbone of NCL asynchronous circuits is the threshold gate [7]. The threshold gate has the property of hysteresis, and is denoted by TH*mn*, where the output of the gate is asserted (set) if the gate has a valid '*DATA*' value on *m* (threshold) of its *n* inputs. In other words when its threshold is met, its output is asserted. The output stays asserted (hysteresis) until all inputs go back to '*NULL*' in its reset phase [7]. For practical implementation using only CMOS transistors, logic '1' represents a '*DATA*' value and logic '0' represents a '*NULL*' value. Our TH*x*2 is capable of implementing both TH12 and TH22 gates, thus forming a complete set of logic [3]. It should be noted that NCL asynchronous circuits offer advantages to SCA avoidance that include **distributed** (unsynchronized in time) and **low power consumption**.

III. OPERATION

The core of our asynchronous FPGA architecture is our programmable TH*x*2 cell. In this section, we describe the operation and design of the cell.

The block diagram of the asynchronous TH*x*2 Field Programmable Gate Array (FPGA) cell is shown in Fig. 3. The *V* and *G* are the positive and negative supply voltages, *Vdd* and *Vss*, respectively. The input *I* is for the value to be stored in the programming memory cell (MC), and the input *Wb* is the *active low* write enable signal to program the MC. The inputs *A* and *B* are the TH*x*2 threshold function inputs, and the output *Z* is the TH*x*2 threshold function output. The internal signal *M* and *Mb* (*Mb* is the logic inverse of *M*), control the mode of the FPGA TH*x*2 cell. The FPGA cell in Fig. 3 is composed of two primary subcomponents: the eleven transistor TH*x*2 base cell and the five transistor programming MC. The two subcomponents form a programmable, asynchronous cell that when arranged in a two-dimensional (2D) array can be used to implement any digital system.

The transistor diagram in Fig. 2 shows the connectivity of the TH*x*2 base cell. The "reset *NULL*" subblock is standard for a two-input threshold gate. When *NULL* values ('0's) are applied to the *A* and *B* inputs, the two PMOS transistors turn "ON," and the *Zb* wire is pulled up to *V* (logic '1'), and the output *Z* wire is reset to *NULL* ('0') by the standard CMOS inverter.

In Fig. 2, the "set *DATA*" block is a key component of the programmable FPGA cell. The mode, *M*, of this block is controlled by the value stored in the programming MC (Fig. 4). A value of **'1' on *M*** ('0' on *Mb*) puts the cell in the **TH12** mode. With *M* = '1,' both NMOS transistors with *M* on the *gates* will be turned "ON" and the diagonal NMOS transistor with *Mb* on the *gate* will be turned "OFF," and the NMOS transistors with *A* and *B* on the *gates* will be in a parallel, TH12 configuration (*drain* nodes connected to *Zb* and *source* nodes connected to *G* or *Vss*).

In the "set *DATA*" block of Fig. 2, a value of **'0' on *M*** ('1' on *Mb*) puts the cell in the **TH22** mode. With *M* = '0,' both NMOS transistors with *M* on the *gates* will be turned "OFF" and the diagonal NMOS transistor with *Mb* on the *gate* will be turned "ON," and the NMOS transistors with *A* and *B* on the *gates* will be in a series, TH22 configuration (*drain* node on the *A* transistor connected to *Zb* through the diagonal *Mb* NMOS transistor).

The rest of the Fig. 2, the part not in the dotted lined borders, forms the output inverter and "*HOLD*" network for the TH*x*2 base cell. This minimal transistor circuit uses a unique modification from a standard implementation to i) keep the overall number of transistors for the programmable FPGA cell to a minimum (sixteen) and ii) keep the number of NMOS and PMOS transistors even (eight). Fig. 5 shows the

standard and modified output inverter and "*HOLD*" networks. Fig. 5a shows the standard implementation that uses two cross-coupled inverter circuits (two NMOS transistors and two PMOS transistors) to output and "*HOLD*" the value of the TH12 and TH22 cells. This would not work for the minimum area FPGA cell because it would result in an imbalanced number of NMOS (nine) and PMOS (seven) transistors. Fig. 5b shows the modified output inverter and "*HOLD*" circuit that uses one NMOS and three PMOS transistors. The transistors in Fig. 5b when combined with the transistors in the "reset *NULL* (Fig. 2)," "set *DATA* (Fig. 2)," and "Memory Cell (Fig. 4)" blocks make a total of sixteen transistors, eight NMOS and eight PMOS. The even number of NMOS and PMOS transistors makes this implementation ideal for forming a nearly square, easily array-able layout.

Some additional explanation of Fig. 5a and Fig. 5b may be necessary to understand and compare the two networks. The standard output inverter and "*HOLD*" transistors in Fig. 5a form two cross-coupled inverters. The larger, output inverter offers a strong signal for the *Z* output. The "*HOLD*" transistors drive the *Zb* signal and enable the hysteresis effect that holds the output value at *DATA* during transition of the *A* and *B* inputs from *DATA* back to *NULL* values. The widths of the transistors in the "*HOLD*" inverter are minimally sized so the "reset *NULL*" and "set *DATA*" networks can overwrite the value held by the "*HOLD*" inverter. In Fig. 5b, the NMOS transistor of the standard "*HOLD*" inverter is replaced by a PMOS transistor. Details of how this works can be understood by considering the four states of the FPGA cell in the TH12 (*M* = '1') and TH22 (*M* = '0') modes: reset *NULL* active, set *DATA* active, *HOLD Z* = *DATA* ('1') active, and *HOLD Z* = *NULL* ('0') active.

1) In the reset *NULL* active state: Regardless of the value on *M*, both *A* and *B* inputs are *NULL* ('0'). The two PMOS transistors in the reset *NULL* network are turned "ON," so the value on *Zb* is driven hard to a '1.' In Fig. 5b, with *Zb* = '1' and *Z* driven to '0,' the *HOLD* PMOS transistor with *Z* on the *gate* is "ON," and it reinforces the *Zb* node to '1.' The PMOS transistor with *Zb* on the *gate* is "OFF" with *Zb* = '1,' and its *source* and *drain* nodes are in an open circuit configuration, not affecting *Zb* or *Z*. This gives the same logical results as the circuit in Fig. 5a.
2) In the set *DATA* active state: Some combination (depending on the mode *M* of TH*x*2) of the *A* and *B* inputs are set to *DATA* ('1') in order to pull *Zb* down to '0' and set the output *Z* to *DATA* ('1'). A combination of NMOS transistors in the set *DATA* network, are turned "ON," so the value on *Zb* is driven hard to a '0'. In Fig. 5b, with *Zb* = '0' and *Z* driven to '1,' the *HOLD* PMOS transistor with *Z* on the *gate* is "OFF," and its *source* and *drain* nodes are in an open circuit configuration, not affecting *Zb* or *Z*. However, the PMOS transistor with *Zb* = '0' on the *gate* is "ON," reinforcing the *Zb* node to '0.' This gives the same logical results as the circuit in Fig. 5a.
3) In the *HOLD Z* = *DATA* ('1') active state: Neither the reset *NULL* or set *DATA* networks are active regardless of the value on *M*. With a '1' on *Z*, the *HOLD* PMOS transistor with *Z* on its *gate* is "OFF." Its *source* and *drain* nodes are in an open circuit configuration, and also have no logical effect on *Zb*. The *HOLD* PMOS transistor with *Zb* = '0' on its *gate* will be "ON," and provide a *weak* '0' to *Zb* which is enough to *HOLD* the output to '1' since there is nothing else driving *Zb*. In addition, since all other paths to a *source* are open, there is minimal dynamic current draw.
4) In the *HOLD Z* = *NULL* ('0') active state: Neither the reset *NULL* or set *DATA* networks are active regardless of the value on *M*. With a '0' on *Z*, the *HOLD* PMOS transistor with *Z* on its *gate* is "ON." Its *source* and *drain* nodes provide a path from *Zb* to *Vdd*, and *HOLD Zb* at '1.' The *HOLD* PMOS transistor with *Zb* = '1,' will be "OFF," and not affect the logical output *Z*.

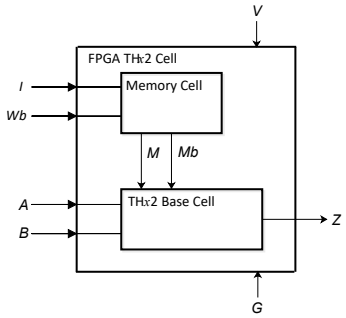


Fig. 3: Block diagram of the FPGA TH*x*2 cell.

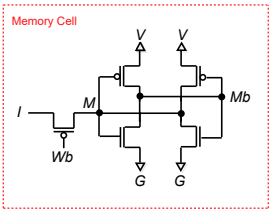
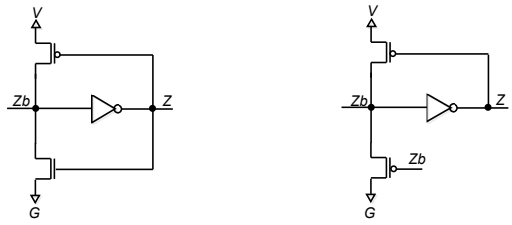


Fig. 4: Transistor diagram of the memory cell for the FPGA TH*x*2 base cell.

The last element of the FPGA TH*x*2 cell is the MC shown in Fig. 4. This is a standard five transistor MC built on a set of minimum sized cross-coupled inverters except that the *WRITE* transistor is a PMOS instead of an NMOS transistor. Again, this is so there are the same number of NMOS and PMOS transistors. Since the *WRITE* transistor is a PMOS transistor, the actual *WRITE* signal is *active low*, *Wb*.

IV. TEST DATA AND ANALYSIS

For the FPGA TH*x*2 cell to work properly, the size of the output inverter and *HOLD* network transistors need to be carefully determined. Then, the widths of the devices in the set to *DATA* and reset to *NULL* sub-circuits (Fig. 2) need to be sized large enough to overpower the *HOLD* network



(a) (b)

Fig. 5: (a) Traditional and (b) modified output and "*HOLD*" networks for the TH*x*2 base cell.

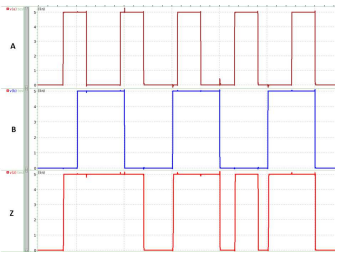


Fig. 6: FPGA TH*x*2 cell simulation in TH12 mode (M = '1').

transistors. Otherwise, the size can be varied to provide more delay (less area) or vice versa. To test the FPGA TH*x*2 cell, it was implemented and then a spice model was extracted from the cell layout. It was tested in both the TH12 and TH22 modes by first writing a logic '1' to the MC and simulating the TH12 mode, and then writing a logic '0' to the MC to simulate the TH22 mode. The spice simulations in Fig. 6 and Fig. 7 show the output waveforms for the TH12 (M = '1') and TH22 (M = '0') modes respectively.

For the TH12 mode (Fig. 6), we set the inputs, A and B, to *NULL* ('0') to reset the output, Z, to *NULL* ('0'). Then we cycle through all combinations of the inputs, A and B, set to *DATA* and *NULL*. The output verifies that with any input value set to *DATA*, the output is also *DATA*. Also, when both inputs are reset to *NULL*, the output is also reset to *NULL*. Thus, the TH12 mode $A + B = Z$ of the FPGA TH*x*2 cell is functioning as expected. Similarly, for the TH22 mode (Fig. 7), we set the inputs, A and B, to *NULL* to reset the output, Z, to *NULL*. Then we cycle through all combinations of the inputs set to *DATA* and *NULL*. The output verifies that both input values must be set to *DATA* for the output to be set to *DATA*. For the TH22 mode, we also see that once the output is set to *DATA*, it stays *DATA* until all inputs go back to *NULL*. This is the hysteresis effect required for proper operation of the TH22 function and the $A \bullet B = Z$ form of the TH22 mode of the FPGA TH*x*2 cell.

V. SUMMARY AND CONCLUSIONS

There are several advantages to clockless asynchronous digital design [7]. Examples include: 1) the asynchronous nature of logic switching minimizes opportunities for power, electromagnetic radiation, temperature and other SCAs, and digital noise reduction for sensitive, mixed-signal ICs; 2) data is processed at average speed versus worst case for synchronous sequential circuits; and 3) the difficult clock-routing step is eliminated from the IC design flow. Some common drawbacks include logic area increase, dual rail wires for all signal nets, and lack of CAD tools for optimizing asynchronous circuits. To make asynchronous design more acceptable and common place, the drawbacks need to be improved, and it needs to be easier to implement asynchronous logic technologies like NCL.

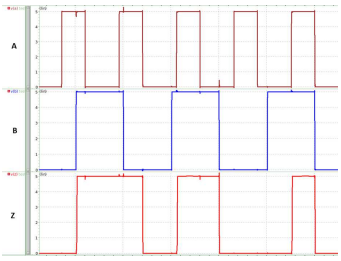


Fig. 7: FPGA TH*x*2 cell simulation in TH22 mode (M = '0').

One easy-to-use digital implementation technology is the FPGA. When compared to other digital circuit implementation technologies, the FPGA can quickly support and satisfy most needs. When you combine FPGA technology with the protection provided by asynchronous circuits against malicious attacks and low power advantages, the asynchronous FPGA becomes a good choice for many applications.

In this paper we presented an asynchronous FPGA technology that can mitigate SCAs and is suitable for NCL asynchronous digital designs. We showed how the asynchronous FPGA cell is based on a minimum sized, multi-mode TH*x*2 threshold gate, and we showed how it can be implemented using a symmetric number of NMOS and PMOS transistors, making it a good candidate for 2-D array structures. It is also compact and includes local storage for the mode select. Lastly, the programmable TH*x*2 cell forms a **complete set** of logic that can be used to implement any asynchronous digital circuit.

REFERENCES

[1] P. Kocher, J. Jaffe, B. Jun, "Differential power analysis," in Annual International Cryptology Conference, Springer, 1999, pp. 388–397.

[2] S. Moore, R. Anderson, P. Cunningham, R. Mullins, and G. Taylor, "Improving smart card security using self-time circuits," in Proceedings of the Eighth International Symposium on Asynchronous Circuits and Systems, IEEE Computer Society, Silver Spring, MD, 2002, pp. 211–218.

[3] J. Emmert and A. Perumalla, "An Asynchronous MPGA TH*x*2 Cell and Architecture for Mitigating Side-Channel Attacks," in IEEE National Aerospace & Electronics Conference, 2019.

[4] M. Tehranipoor and F. Koushanfar, "A survey of HW Trojan taxonomy and detection." IEEE Des. Test Comput. Vol. 27, 2010, pp. 10–25.

[5] L. Lin and W. Burleson, and C. Parr, "MOLES: malicious off-chip leakage enabled by side-channels," in IEEE/ACM International Conference on CAD(ICCAD), November, 2009, pp. 117–122.

[6] R. Sridhar, "Asynchronous design techniques," in Proceedings of Fifth Annual IEEE International ASIC Conference, September 1992, pp. 296-300.

[7] K. Fant and S. Brandt, "NULL convention logic: a complete and consistent logic for asynchronous digital circuit sythesis," in Proceedings of the International Conference on Application Specific Systems, Architectures and Processors, August 1996, pp. 261-273.