Delay Insensitive Ternary Logic Utilizing CMOS and CNTFET

Ravi Sankar Parameswaran Nair

University of Arkansas, Fayetteville

Follow this and additional works at: http://scholarworks.uark.edu/etd

Part of the Electrical and Electronics Commons, Electronic Devices and Semiconductor Manufacturing Commons, and the VLSI and Circuits, Embedded and Hardware Systems Commons

Recommended Citation

Parameswaran Nair, Ravi Sankar, "Delay Insensitive Ternary Logic Utilizing CMOS and CNTFET" (2012). Theses and Dissertations. 548.
http://scholarworks.uark.edu/etd/548

This Dissertation is brought to you for free and open access by ScholarWorks@UARK. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of ScholarWorks@UARK. For more information, please contact scholar@uark.edu, ccmiddle@uark.edu.
DELAY INSENSITIVE TERNARY LOGIC UTILIZING CMOS AND CNTFET
DELAY INSENSITIVE TERNARY LOGIC UTILIZING CMOS AND CNTFET

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Engineering

By

Ravi Sankar Parameswaran Nair
Missouri University of Science and Technology
Master of Science in Computer Engineering, 2007

August 2012
University of Arkansas
ABSTRACT

As digital circuit design continues to evolve due to progress of semiconductor processes well into the sub 100 nm range, clocked architectures face limitations in a number of cases where clockless asynchronous architectures require substantially less power, generate less noise, and produce less electro-magnetic interference (EMI). This dissertation develops the Delay-Insensitive Ternary Logic (DITL) asynchronous design paradigm that combines the design aspects of similar Dual-Rail asynchronous paradigms and Boolean logic to create a single wire per bit, three voltage signaling and logic scheme.

DITL is designed at the transistor level using multi-threshold CMOS and carbon nanotube (CNT) FETs to develop primitive logic gates, which are combined to design larger circuits, simulated at the transistor level, and compared with other paradigms for energy, timing, and area. DITL is applied to design secure hardware resistant to side-channel attacks and found to be more attack resistant than other methods.
This dissertation is approved for recommendation to the Graduate Council.

Dissertation Director:

____________________________________
Scott Smith, Ph.D.

Dissertation Committee:

____________________________________
Jia Di, Ph.D.

____________________________________
Alan Mantooth, Ph.D.

____________________________________
Simon Ang, Ph.D.

____________________________________
Randy Brown, Ph.D.
DISERTATION DUPLICATION RELEASE

I hereby authorize the University of Arkansas Libraries to duplicate this dissertation when needed for research and/or scholarship.

Agreed_____________________________________

Ravi Sankar Parameswaran Nair

Refused_____________________________________

Ravi Sankar Parameswaran Nair
ACKNOWLEDGEMENTS

I am honored to work in such a positive environment provided by the University of Arkansas and the students, faculty and staff of the Engineering Department.

I would like to offer my heartfelt gratitude to my dissertation director, Dr. Scott C. Smith for giving me the opportunity to work with him on various projects that gave me new insights and technical knowledge. He has been a great guide and mentor right from the time I worked on my Masters and throughout my Ph.D. program. Working with him was always a pleasant and positive experience and he always showed extreme patience with me and gave encouragement when needed.

I would like to offer my heartfelt gratitude to Dr. Jia Di, for letting me work on projects he conceived with Dr. Smith and for intervening at the right times in my research giving guidance and new ideas to try out which progressed my research further.

I would like to sincerely thank Dr. Alan Mantooth, Dr. Simon Ang, and Dr. Randy Brown for serving on my dissertation committee. I would like to thank my fellow students whom I worked with, at the ENGR Research Labs, for being great guys and providing me valuable help when I needed it.

Last but not least, I thank my loving family for their support in every way that held me strong in my quest for knowledge.
# TABLE OF CONTENTS

1. Introduction ................................................................................................................. 1

2. Previous work .............................................................................................................. 4
   2.1 NULL Convention Logic (NCL) ................................................................................. 4
   2.2 Pre-Charge Half-Buffer (PCHB) ............................................................................. 10
   2.3 Ternary Logic .......................................................................................................... 12

3. Delay-Insensitive Ternary Logic (DITL) .................................................................... 15
   3.1 Redesigning Ternary Voltage Detect Circuits ......................................................... 15
   3.2 DITL Gate Architecture ......................................................................................... 19
   3.3 Comparing DITL with PCHB and NCL .................................................................. 22

4. DITL Secure Hardware Application ........................................................................... 24
   4.1 Problems of Existing Security Solutions ............................................................... 25
      4.1.1 Inflexibility ........................................................................................................ 25
      4.1.2 High Overhead ................................................................................................. 25
   4.2 Circuit-Level Secure Hardware Design Plan ......................................................... 26
   4.3 Circuit-Level Side-Channel Attacks and Countermeasures .................................. 27
      4.3.1 Power-Based Attacks ....................................................................................... 27
      4.3.2 Timing-Based Attacks ...................................................................................... 28
      4.3.3 Electromagnetic-Based Attacks ....................................................................... 28
      4.3.4 Fault-Based Attacks ......................................................................................... 29
   4.4 Circuit-Level Side-Channel Attack Mitigation ....................................................... 30
      4.4.1 Side-channel Attack Mitigation Using DITL .................................................... 30
   4.5 Achieved Team Research Objectives .................................................................... 36
LIST OF FIGURES

Figure 1: THmn Gate ........................................................................................................5
Figure 2: NCL Gate Symbols: TH23 (left) TH22 (right) .................................................5
Figure 3: NCL TH23 circuit ............................................................................................6
Figure 4: NCL TH22 circuit ............................................................................................6
Figure 5: NCL system framework: Local handshaking instead of a global clock ..........7
Figure 6: NCL NAND2 registered design .......................................................................8
Figure 7: NCL Register ..................................................................................................8
Figure 8: NCL NAND2 Gate .........................................................................................9
Figure 9: NCL THand0 circuit ......................................................................................9
Figure 10: PCHB NAND2 circuit ..................................................................................10
Figure 11: NCL TH33 gate circuit ................................................................................11
Figure 12: Watchful timing diagram [19] .....................................................................13
Figure 13: Original ternary logic detect circuits [21] ....................................................13
Figure 14: 2-Transistor Detect Circuits with Reverse Body Bias .................................16
Figure 15: Final 3-Transistor Detect Circuits with RBB ................................................17
Figure 16: Is-DATA component ...................................................................................18
Figure 17: Version I of DITL NAND2 ..........................................................................20
Figure 18: Cadence Simulation of DITL NAND2 .........................................................21
Figure 19: Version II of DITL NAND2 ..........................................................................22
Figure 20: Secure Full Adder design .............................................................................31
Figure 21: DITL ALU Testbench ..................................................................................37
Figure 22: DITL ALU Ultrasim simulation ....................................................................38
LIST OF TABLES

Table I: Truth Table for Detect Circuits ..................................................................................14
Table II: Truth Table for Is-DATA Component ......................................................................19
Table III: Nand2 Comparison: DITL vs PCHB vs NCL .........................................................23
Table IV: Measurements from Balanced DITL gates .................................................................33
Table V: Measurements from Balanced DITL Full Adder .........................................................34
Table VI: Secure Full Adder comparison ..................................................................................35
Table VII: DITL ALU Simulation results .................................................................................39
Table VIII: Measurements for CNT DITL NAND2 Registers ...............................................59
Table IX: Measurements for CNT DITL XOR2 and 1Bit Registers .........................................60
Table X: Measurements for CNT DITL Gates ........................................................................61
Table XI: Measurements for CNT DITL Full Adders Type 1 and 2 .........................................64
Table XII: Measurements for CNT DITL Full Adders Type 3 and 4 .........................................65
1. INTRODUCTION

For the last three decades, the focus of digital design has been primarily on synchronous, clocked architectures. However, as clock rates have significantly increased while feature size has decreased, clock skew has become a major problem. High performance chips must dedicate increasingly larger portions of their area for clock drivers to achieve acceptable skew, causing these chips to dissipate increasingly higher power, especially at the clock edge, when switching is most prevalent. As these trends continue, the clock is becoming more and more difficult to manage, while clocked circuits’ inherent power inefficiencies are emerging as the dominant factor hindering increased performance. These issues have caused renewed interest in asynchronous digital design. Asynchronous, clockless circuits require less power, generate less noise, and produce less electro-magnetic interference (EMI), compared to their synchronous counterparts, without degrading performance. Furthermore, delay-insensitive asynchronous paradigms have a number of additional advantages, especially when designing complex circuits, like Systems-on-Chip (SoC), including substantially reduced crosstalk between analog and digital circuits, ease of integrating multi-rate circuits, and facilitation of component reuse.

As demand increases for designs with higher performance, greater complexity, and decreased feature size, asynchronous paradigms will become more prevalent in the multi-billion dollar semiconductor industry, as predicted by the International Technology Roadmap for Semiconductors (ITRS), which envisions a likely shift from synchronous to asynchronous design styles in order to increase circuit robustness, decrease power, and alleviate many clock-related issues [1, 2]. ITRS shows that asynchronous circuits currently account for approximately 20% of chip area, and estimates they will comprise 30% of chip area by 2017, and 45% of chip area by 2022 [3].
Asynchronous circuits can be grouped into two main categories: bounded-delay and delay-insensitive models. Bounded-delay models, such as Micropipelines [4], assume that delays in both gates and wires are bounded. Delays are added based on worse-case scenarios to avoid hazard conditions. This leads to extensive timing analysis of worse-case behavior to ensure correct circuit operation. On the other hand, delay-insensitive circuits, like NULL Convention Logic (NCL) [5] and Pre-Charge Half-Buffers (PCHB) [6], assume delays in both logic elements and interconnects to be unbounded, although they assume that wire forks within basic components, such as a full adder, are isochronic [7], meaning that the wire delays within a component are much less than the logic element delays within the component, which is a valid assumption even in future nanometer technologies. Wires connecting components do not have to adhere to the isochronic fork assumption. This implies the ability to operate in the presence of indefinite arrival times for the reception of inputs. Completion detection of the output signals allows for handshaking to control input wavefronts. Delay-insensitive design styles therefore require very little, if any, timing analysis to ensure correct operation (i.e., they are correct by construction), and also yield average-case performance rather than the worse-case performance of bounded-delay and traditional synchronous paradigms. Each data unit in a delay-insensitive system can take at least three values: logic 0, logic 1, and a spacer called NULL. NCL and PCHB delay-insensitive circuit methods have to use at least two binary rail signals to represent a single data unit. Therefore for every $N$ number of bits, at least $2N$ interconnect wires are needed. Each of these rails needs its own set of gates to evaluate logical values, and hence dual-rail circuits will have around twice the number of transistors compared to Boolean logic.

In this dissertation, a new method called Delay-Insensitive Ternary Logic (DITL) is introduced, which combines the design aspects of NCL, PCHB, and Boolean gates to form a
delay-insensitive paradigm that uses a single rail to represent a single data unit, which can have three distinct voltage levels corresponding to the three values of logic 0, logic 1, and NULL. Some advantages envisioned for DITL compared to NCL are half the number of interconnects, and fewer transistors and power dissipation due to a reduced voltage swing for each NULL to DATA transition. This dissertation will cover the following main topics:

1) Previous Work: An introductory view of Asynchronous paradigms such as NCL and PCHB, and also discusses previous work concerning Ternary Logic and its drawbacks;

2) Delay-Insensitive Ternary Logic: DITL is developed at the gate level using transistor simulations of a basic logic circuit like NAND2. A 1.2 V 130nm IBM 8rf-DM CMOS process is used to develop the basic DITL architecture through simulation and verification at the transistor level;

3) DITL Secure Hardware Application: One particular DITL application researched is creating Secure Hardware chips resistant to side channel attacks via statistical collection of timing and power data. The DITL architecture is modified and a basic functional unit such as a Full Adder is evaluated against NCL and Boolean methods towards its ability to resist side channel attacks. Necessary basic circuits are developed that helped towards the creation of a secure DITL gate library and design of an Arithmetic Logic Unit (ALU) utilizing this library. The DITL ALU simulation results are presented.

4) Carbon Nano Tube FET Based DITL Design: As an alternative approach, DITL circuits are reworked using a 0.9 V Carbon-Nano Tube (CNT) FET spice model and several design styles are researched.
2. PREVIOUS WORK

2.1 NULL Convention Logic (NCL)

NCL uses dual-rail signals to achieve delay-insensitive behavior. A dual-rail signal, $D$, consists of two wires, $D^0$ and $D^1$, which may assume any value from the set \{DATA0, DATA1, NULL\}. The DATA0 state ($D^0 = 1$, $D^1 = 0$) corresponds to a Boolean logic 0, the DATA1 state ($D^0 = 0$, $D^1 = 1$) corresponds to a Boolean logic 1, and the NULL state ($D^0 = 0$, $D^1 = 0$) corresponds to the empty set meaning that the value of $D$ is not yet available. The two rails are mutually exclusive, so that both rails can never be asserted simultaneously; this state is an illegal state.

NCL differs from other gate-level delay-insensitive paradigms [8-12] in that these other paradigms only utilize one type of state-holding gate, the C-element [13]. A C-element behaves as follows: when all inputs assume the same value then the output assumes this value, otherwise the output does not change. On the other hand, all NCL gates are state-holding. Thus, NCL circuits have a greater potential for optimization than other gate-level delay-insensitive paradigms [14].

NCL uses threshold gates for its basic logic elements [15]. The primary type of threshold gate is the THmn gate, where $1 \leq m \leq n$, as depicted in Figure 1. THmn gates have $n$ inputs. At least $m$ of the $n$ inputs must be asserted before the output will become asserted. Because NCL threshold gates are designed with hysteresis, all asserted inputs must be de-asserted before the output will be de-asserted. This ensures a complete transition of inputs back to NULL before asserting the output associated with the next wavefront of input DATA. Therefore, a THnn gate is equivalent to an $n$-input C-element and a TH1n gate is equivalent to an $n$-input OR gate. In the representation of a THmn gate, each of the $n$ inputs is connected to the rounded portion of the
gate; the output emanates from the pointed end of the gate; and the gate’s threshold value, \( m \), is written inside of the gate.

\[ \text{Figure 1: } \text{THmn gate} \]

Figure 2 shows the symbols for a TH23 gate and a TH22 gate. A TH23 has 3 inputs with a threshold value of 2. Hence, the output is asserted when at least two of the three inputs are asserted. The output is de-asserted only when all three inputs are de-asserted. The TH22 gate has a threshold of 2 and 2 inputs. The output is asserted/de-asserted only when both inputs are asserted/de-asserted. Figures 3 and 4 show the transistor level implementations of static versions of the TH23 and TH22 gates. The hysteresis state holding function is provided by the feedback path from the output of the inverter as seen in these figures.

\[ \text{Figure 2: } \text{NCL Gate Symbols: TH23 (left) TH22 (right)} \]

By employing threshold gates for each logic rail, NCL is able to determine the output status without referencing time. Delay-insensitive circuits communicate using request and acknowledge signals, \( K_i \) and \( K_o \), respectively, as shown in Figure 5, to prevent the current DATA wavefront from overwriting the previous DATA wavefront, by ensuring that the two DATA wavefronts are always separated by a NULL wavefront [5].
Figure 3: NCL TH23 circuit

Figure 4: NCL TH22 circuit
The acknowledge signal from the receiving circuit is the request signal to the sending circuit. When the receiver circuit latches the input DATA, the corresponding $K_o$ signal will be logic 0, indicating a request-for-NULL ($rfn$); and when it latches the input NULL, the corresponding $K_o$ signal will be logic 1, indicating a request-for-DATA ($rfd$). When the sending circuit receives a $rfd$/$rfn$ on its $K_i$ input, it will allow a DATA/NUL wavefront to be output, respectively. This delay-insensitive handshaking protocol coordinates delay-insensitive circuit behavior, analogous to coordination of synchronous circuits by a clock signal. Additionally, delay-insensitivity requires a circuit to be input-complete, which means that all outputs may not transition from NULL to DATA until all inputs have transitioned from NULL to DATA, and that all outputs may not transition from DATA to NULL until all inputs have transitioned from DATA to NULL [12]. In circuits with multiple outputs, it is acceptable according to Seitz’s “weak conditions” of delay-insensitive signaling [9], for some of the outputs to transition without having a complete input set present, as long as all outputs cannot transition before all inputs arrive.

An example of a design that implements an NCL system in practice is shown in Figure 6, consisting of two input NCL registers, followed by a NCL NAND2 function and followed by a single output NCL register. There is a TH22 gate used to combine the $K_o$ signals from the input registers to create a single $K_o$ output for the system. Figure 7 shows the internal circuit of the NCL Register. The NCL register is composed of NCL TH22n gates and a Boolean NOR2 gate.
The TH22n gate is a TH22 gate with a reset to logic 0 function included. Figure 8 shows the internal structure of the NCL NAND2 function, which is composed of a TH22 gate and a THand0 gate. The internal circuit of the THand0 gate is shown in Figure 9. The NCL NAND2 registered design is used for comparison with DITL in the Section 3.3.

**Figure 6: NCL NAND2 registered design**

**Figure 7: NCL Register**
Figure 8: NCL NAND2 Gate

Figure 9: NCL THand0 circuit
2.2 Pre-Charge Half-Buffer (PCHB)

PCHB circuits [6] are designed at the transistor level, utilizing dynamic CMOS logic, instead of targeting a predefined set of gates like the previously mentioned DI paradigms [5, 8-12]. PCHB circuits have dual-rail data inputs and outputs, and combine combinational logic and registration together into a single block, as shown in Figure 10, yielding a very fine-grain pipelined architecture. The dual-rail output is initially pre-charged to NULL.

Figure 10: PCHB NAND2 circuit
When request \((R_{ack})\) and acknowledgement \((L_{ack})\) are both \(rfd\), the specific function will evaluate when the inputs, \(X\) and/or \(Y\), become DATA, causing the output, \(F\), to become DATA. \(L_{ack}\) will then transition to \(rfn\) only after all inputs and the output are DATA. When \(R_{ack}\) is \(rfn\) and \(L_{ack}\) is \(rfd\), or vice versa, the output will be floating, so weak inverters must be used to hold the current output value. After both \(R_{ack}\) and \(L_{ack}\) are \(rfn\), the output will be pre-charged back to NULL. After all inputs become NULL and the output changes to NULL, \(L_{ack}\) will change back to \(rfd\), and the next DATA wavefront can evaluate after \(R_{ack}\) becomes \(rfd\). The PCHB circuit contains the gates such as a Boolean NOR2, strong-weak Inverter pair and an NCL TH33 gate. The NCL TH33 gate is shown in Figure 11 below and is used in the PCHB for generating the \(L_{ack}\) signal when both the inputs \(X\) and \(Y\) are in the same state as each other and as the output, \(F\).

Figure 11: NCL TH33 gate circuit
2.3 Ternary Logic

Ternary logic utilizes three distinct voltage values per wire, \(V_{dd}\), \(\frac{1}{2}V_{dd}\), and Gnd, whereas binary logic utilizes two distinct voltage values, \(V_{dd}\) and Gnd. Hence, ternary logic can be used as an alternative to dual-rail logic to represent the three logic states (i.e., DATA0, DATA1, and NULL), requiring only one wire per bit. \(V_{dd}\) is used to represent DATA1, Gnd to represent DATA0, and \(\frac{1}{2}V_{dd}\) to represent NULL, which yields maximum noise margin with minimum switching power dissipation, since each wire always switches to NULL between every two DATA states, such that the voltage swing is always \(\frac{1}{2}V_{dd}\).

[16, 17] develop a ternary logic completion detection circuit for use with a bounded-delay self-timed paradigm; and [18, 19] develops a ternary bounded-delay self-timed paradigm, which is similar to micropipelines [4]. However, as mentioned at the beginning of Section II, delay-insensitive paradigms have many more advantages compared to their bounded-delay counterparts. [20] develops a delay-insensitive ternary logic transmission system, called Asynchronous Ternary Logic Signaling (ATLS), which converts dual-rail signals into ternary logic for transmission over a bus, in order to decrease transmission area and power. However, all of the logic processing is still done using dual-rail logic. [21, 22] develop a circuit called a Watchful as part of their proposed delay-insensitive ternary logic paradigm. However, as shown in the timing diagram in Figure 12, their approach is not delay-insensitive because it assumes that the input \(in\) will transition to VI (NULL state) before the signal \(clear\) is asserted, causing the signal \(full\) to be de-asserted. In order to be delay-insensitive, \(full\) must not be de-asserted until both \(clear\) is asserted and \(in\) transitions to VI. Otherwise, if \(in\) remained at one DATA value (e.g., if no additional data needs to be processed at this time), this DATA value would continue to be utilized in subsequent operations instead of causing the system to become idle.
[23] utilizes diode-connected transistors to shift the threshold voltage in special inverters dedicated to detect the presence of only one input logic level. As shown in Figure 13, for the Detect logic 0 circuit, \( \text{in} \) must be lower than \( \text{Vdd}-2\text{Vt}_p \) for the PMOS transistors to turn ON and pull \( \text{out} \) to Vdd. Similarly, for the Detect logic 1 circuit, \( \text{in} \) must be higher than \( 2\text{Vt}_N \) for \( \text{out} \) to be pulled down to Gnd. The truth table for Detect0 and Detect1 is provided in Table I.

**Figure 12: Watchful timing diagram [19]**

**Figure 13: Original ternary logic detect circuits [21]**
### TABLE I: Truth Table for Detect Circuits

<table>
<thead>
<tr>
<th>Ternary Input</th>
<th>Detect0 output</th>
<th>Detect1 Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gnd or DATA0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>½ Vdd or NULL</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Vdd or DATA1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
3. DELAY-INSENSITIVE TERNARY LOGIC (DITL)

The Delay-Insensitive Ternary Logic (DITL) paradigm developed in this paper utilizes three distinct voltage levels, Vdd, ½ Vdd, and Gnd, to encode the three delay-insensitive logic states, DATA0, NULL, and DATA1, respectively, on a single wire. The motivations for utilizing ternary logic for delay-insensitive circuit design include reducing area (only half the number of wires are required for each bit compared to dual-rail logic) and reducing power/energy (since each transition i.e., NULL to DATA or vice-versa, only requires a ½ Vdd swing compared to a full Vdd swing for dual-rail logic).

3.1 Redesigning Ternary Voltage Detect Circuits

The Original detect circuits previously mentioned in Figure 13 were used as a basis for DITL design. From transistor level simulations in a 1.2V 130nm IBM 8rf-DM process, these circuits were found to consume significant static power because all transistors are partially turned ON for a NULL (½ Vdd) input, which consumes 31.8 nW for Detect0 and 5.5 nW for Detect1. Additionally, they require output inverters to properly shape the outputs; otherwise the output is only 1.07V instead of 1.2V for Detect0 with an input of 0V, and 0.17V instead of 0V for Detect1 with an input of 1.2V.

To decrease static power consumption, a method called Reverse Body Bias (RBB) [24-26] was used. In RBB, a voltage higher than Vdd is applied as the Pfet body bias and a voltage lower than Gnd is applied as the Nfet body bias so as to increase the threshold voltage, which results in less leakage and static power. New detect circuits were devised to incorporate RBB so that a high power reduction and improved ternary input detection were achieved while not sacrificing speed of operation. This set of Detect0 and Detect1 circuits are shown in Figure 14 and they
were found to perform well using an inverter like 2-transistor model. Using the following body biases: \( VBp0 = +4V; \ VBn0 = 0V; \ VBp1 = +1.5V; \) and \( VBn1 = -2.4V, \) the 2-transistor Detect0 consumed 1.13 nW and Detect1 consumed 0.98 nW. The 2-transistor detect circuits were also faster than their 3-transistor counterparts (i.e., average propagation delay of 0.37 ns vs. 0.45 ns for Detect0 and 0.33 ns vs. 0.65 ns for Detect1).

![Figure 14: 2-Transistor Detect Circuits with Reverse Body Bias](image)

Later it was noticed that the range of allowable bias voltages for the IBM 8rf-DM process transistors were much lower than the 2-transistor bias values given above. These detect circuits used multi-threshold transistors such as Low power (high threshold), High Speed (low threshold), and normal operation (regular threshold); and all of these had 2.2nm thin oxide dielectrics that were restricted to a maximum voltage of \( V_{dd_{max}} = 1.6 \) V. The maximum steady state voltage allowed between any two terminals (gate, source, drain, and body) of a FET cannot exceed \( V_{dd_{max}}. \) In addition to this, Thin Oxide FETs that are exposed to source to drain bias higher than 1.5 V under nominal, normal operating conditions must be longer than the minimum
channel length of 120 nm in design dimension and the circuits must be rigorously analyzed for all Hot Carrier stress types for all the devices. Even though the circuits were designed and verified to be working in transistor level simulation in Cadence virtuoso spectre simulator, it is expected that the simulation may fail at either the layout level or while testing a fabricated chip.

As Section 4.5.1 describes, a DITL ALU design was done to be utilized to fabricate a chip; and this design needed new detect circuits that conformed to the IBM PDK 1.2V 130nm process constraints. For this the original 3-transistor detect circuits were used in combination with RBB to produce a compromise. This new and final design is shown in Figure 15 where all of the Pfets' body biases are set to 1.6V and Nfets' body biases to -0.4V. This would make a Detect0 with steady state power of 8nW and propagation delay of 0.55ns; and a Detect1 with 0.75nW and 0.65ns.

![Figure 15: Final 3-Transistor Detect Circuits with RBB](image-url)
Using the Ternary Detect0 and Detect1 circuits, a component called Is-DATA, shown in Figure 16, was designed to distinguish the three input voltages and encode the input states into binary outputs. Is-DATA has a $IsD$ output that is logic 1 when the input is either DATA0 or DATA1, and is logic 0 when the input is NULL; $Is0$ is logic 1 when the input is DATA0 and logic 0 when the input is either NULL or DATA1; and $Is1$ is logic 1 when the input is DATA1 and logic 0 when the input is either NULL or DATA0, as summarized in Table II.

![Figure 16: Is-DATA component](image-url)
TABLE II: Truth Table for Is-DATA Component

<table>
<thead>
<tr>
<th>Ternary Input</th>
<th>IsD</th>
<th>Is1</th>
<th>Is0</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA0 (Gnd)</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>NULL (½ Vdd)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DATA1 (Vdd)</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

3.2 DITL Gate Architecture

The PCHB paradigm is shown in Figure 10, where each component is designed at the transistor level, and consists of dual-rail data inputs and outputs, with registration included in every combinational logic component. Like PCHB, DITL circuits are designed at the transistor level, but consist of ternary data inputs and outputs and binary handshaking signals. For Version I of DITL, shown in Figure 17, primary inputs X and Y are directly connected to Is-DATA components as well as the Specific Function. When Rack and Lack are both rfd and the inputs, X and Y, are both DATA, the specific function will evaluate, causing the output, F, to become DATA, which will then transition Lack to rfn. When Lack is rfn and Rack is still rfd, the specific function is floating, so the output needs to be held at its proper DATA value, either DATA0 or DATA1, which is done by the Hold 0 and Hold 1 circuitry, respectively. After Rack changes to rfn, the output will be pre-charged to NULL (i.e., ½ Vdd), through N-fets for increased speed. After all inputs become NULL and the output changes to NULL, Lack will change back to rfd, and the next DATA wavefront can evaluate after Rack becomes rfd and the inputs change to DATA. If Rack changes to rfd before the inputs become NULL, if the inputs become NULL
before Rack changes to rfd, or if both Rack and Lack are rfd but the inputs are still NULL, the
pre-charge to NULL logic will no longer be conducting, so the NULL output must be maintained
through the Hold NULL circuitry.

Figure 18 shows the Cadence simulation of the DITL NAND2 function, using the 1.2V,
130nm IBM 8RF-DM process. As can be seen from the waveform, output $F$ will transition to
DATA only when both Rack and Lack are rfd and both inputs $X$ and $Y$ are DATA. The output $F$
can transition to NULL as soon as both Lack and Rack are simultaneously rfn.

![Figure 17: Version I of DITL Nand2](image-url)
Version II of the DITL architecture is shown in Figure 19, where the Specific Function inputs come from the input Is-DATA components instead of the external inputs, X and Y. Version II requires one additional inverter inside the Is-DATA component for the isI output corresponding to each data input, but the advantage is that each data input drives exactly one Is-DATA component for each DITL circuit to which it is an input, such that the capacitance driven by a particular signal only depends on the number of circuits to which the signal is an input, and not on the type of circuits it drives (e.g., if signal A is an input to an XOR2 and NOR3 circuit and signal B is an input to a NAND4 and OR2 circuit, both drive the same amount of capacitance.
because they both drive two Is-DATA components). The use of Version II DITL circuits in a secure hardware application is discussed in detail in Section 4.4.

![Figure 19: Version II of DITL Nand2](image)

3.3 Comparing DITL with PCHB and NCL

NAND2 circuits previously discussed for NCL in Figure 6, PCHB in Figure 10, and DITL in Figures 17 and 19, were simulated in Cadence and the results are tabulated in Table III. DITL Version I is slightly slower, but requires slightly less area and energy compared to Version II. Compared to PCHB, DITL is 21% slower, 74% larger, but requires 68% less energy. Compared to NCL, DITL is 50% slower, but requires 38% less energy and is 89% smaller.
Therefore, DITL has a significant energy advantage compared to PCHB and NCL, and is also more area efficient than NCL. Additionally, as circuit size increases, DITL and PCHB circuits increase at a much smaller rate than NCL circuits (e.g., for a NAND2 vs. a NAND4 circuit, the area increase is 42% for DITL, 70% for PCHB, and 94% for NCL).

Comparing the Average static power, DITL consumes 150% more power than PCHB and 18% more power than NCL. DITL peak dynamic power is 7% less than NCL and 123% more than PCHB.

### TABLE III: NAND2 Comparison: DITL vs PCHB vs NCL

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DITL V1</td>
<td>5.43</td>
<td>50.3</td>
<td>78</td>
<td>6.9</td>
<td>60</td>
</tr>
<tr>
<td>DITL V2</td>
<td>5.40</td>
<td>52.3</td>
<td>82</td>
<td>7.5</td>
<td>67</td>
</tr>
<tr>
<td>PCHB</td>
<td>4.49</td>
<td>86.3</td>
<td>46</td>
<td>3</td>
<td>35</td>
</tr>
<tr>
<td>NCL</td>
<td>3.61</td>
<td>70.8</td>
<td>151</td>
<td>6.35</td>
<td>72</td>
</tr>
</tbody>
</table>
4. DITL SECURE HARDWARE APPLICATION

The increasingly pervasive use of digital information storage and processing devices largely facilitates societal activities, ranging from people’s everyday life to government and military missions. The demands of storing and processing sensitive information, e.g., passwords, messages, personnel records, have resulted in incorporating strong cryptographic algorithms inside these devices. Sensitive information is first encrypted by the host device and becomes cipher text before it is transferred to another device, where the cipher text is decrypted into plaintext for processing. Since most pervasive data storage and processing devices use one or more Integrated Circuits (ICs) as core component(s), incorporating cryptography on-chip significantly enhances the security of the information being stored/processed due to the fact that modern cryptographic algorithms, e.g., Advanced Encryption Standard (AES), RSA, are very difficult to break in a brute-force way. However, attackers have switched their targets from the cryptographic algorithms themselves to the implementations of these algorithms. In particular, attackers have been trying to exploit on-chip security information, e.g., cryptographic keys, through “side-channel” measurements, including power consumption, timing delay, and electromagnetic (EM) emissions.

From a hardware perspective, such side-channel attacks can be implemented at both the circuit- and architecture-level. At the circuit-level, due to CMOS circuit characteristics, a digital CMOS IC exhibits fluctuations in these side-channel measurements while processing different data, causing information leakage. By applying statistical algorithms to the measured transient side-channel data, attackers are able to decipher the secure information stored on-chip. In addition to these passive side-channel attacks, there is a type of semi-active attack, named fault
injection attack, where attackers intentionally induce faulty behaviors of the target IC and monitor the circuit outputs and/or side-channel characteristics.

### 4.1 Problems of Existing Security Solutions

Much research has been performed in mitigating side-channel attacks, and a number of solutions have been proposed. Unfortunately, these solutions have one or more weaknesses/limitations as discussed below.

#### 4.1.1 Inflexibility

Most solutions are developed to protect one cryptographic algorithm (e.g., AES, RSA) in mitigating one or two side-channel attacks (e.g., power, timing). Therefore, the flexibility of these solutions is severely limited. This inflexibility is two-fold: 1) there is no universal solution to all four major categories of side-channel attacks, i.e., power-, timing-, EM-, and fault-based attacks; and 2) there is a lack of general side-channel mitigation techniques that can be adopted by all prevailing cryptographic algorithms, such that when the user switches to another algorithm, there will be no major changes in the hardware design methodology for increased security.

#### 4.1.2 High Overhead

Almost all existing solutions add significant overhead to the original implementation. Such overhead includes more power consumption, longer time delay, larger chip area, reduced circuit reliability, higher design complexity, and incompatibility with commercial digital IC design flow. For example, dual-rail asynchronous logic for mitigating power-based attacks causes considerable timing and area overhead and requires a customized design flow; various
pre-charge based dynamic logic paradigms introduce additional power consumption, increased design complexity, and reliability degradation; fault-tolerant techniques usually incur severe penalties in power, timing, and area. Such overhead hinders the wide adoption of these side-channel attack countermeasures in commercial products.

4.2 Circuit-Level Secure Hardware Design Plan

At the circuit level, Delay-Insensitive Ternary Logic (DITL) is to be utilized for designing logic circuits with properly sized transistors. While maintaining the advantages of asynchronous logic in mitigating side-channel attacks, e.g., distributed and balanced switching activities, DITL eliminates the drawbacks such as high area overhead, average performance that facilitates timing-based attacks, and imbalanced load capacitance between the two rails. In addition, DITL offers a number of benefits including lower power, higher performance, and commercial design flow compatibility.

This research has only one assumption: physical/invasive attacks are excluded. Such attacks require de-packaging the target IC to expose the internal circuits, and using special equipment to monitor/modify circuit elements or stored data. Examples include micro-probing attacks, chip rewriting attacks, and memory remanence attacks. There are two reasons for making this assumption: 1) such attacks require special equipment, much longer time, and highly skilled attackers to perform, which significantly limits the numbers of ICs/applications that require this type of protection; and 2) a number of physical and even destructive protection mechanisms have been developed, such as inserting pressure sensors on-chip to sense de-packaging behaviors. If needed, these mechanisms can be included with the proposed research to achieve an even higher level of security.
As proof of concept, a series of full adders were designed in Boolean, NCL, and DITL at the transistor level, using the IBM 8RF-DM 0.13µm process and compared for different power and timing parameters as explained in Section 4.4.1; and DITL is shown to be the best option for designing secure functional units. Afterward, a DITL library for secure gates was created and a DITL ALU circuit employing these gates was designed and simulated.

4.3 Circuit-Level Side-Channel Attacks and Countermeasures

4.3.1 Power-Based Attacks

Most electronic devices running cryptographic algorithms are implemented in CMOS technology, where transistors act as voltage-controlled switches. While a circuit node is switching, electrons flow across the corresponding transistors to charge/discharge its load capacitance, thereby consuming power. Due to the fact that different transistors will be turned ON/OFF while processing different data, causing different power consumption, side-channel attacks in this category are implemented using the IC’s transient power data. The theory of power-based attacks, e.g., Differential Power Analysis (DPA), was introduced in [327-28]. In general, these attacks require the transient power data while the target IC performs encryption/decryption on different texts, and then use statistical algorithms to derive the key. Power-based attacks are the most powerful and prevalently implemented side-channel attacks, which have been successfully implemented to crack almost all cryptographic algorithms on different platforms, including DES [29], Elliptic Curve Cryptosystems [30], RSA [31-32], AES [33-34], and all AES candidates [35], implemented on FPGAs [36] and as ASICs [37].

A number of methods have been proposed for mitigating power-based attacks by decoupling transient power consumption from the data being processed. Techniques based on
balancing power fluctuation include new CMOS logic gates [38-53], which go through a full charge/discharge cycle for each data processed. Asynchronous circuits, especially dual-rail encoded logic, have been well studied for anti-DPA because of the fixed switching activities during each DATA-Spacer cycle [54-72]. Other power balancing methods include modifying the algorithm execution [73-76], compensating current at the power supply node [77-80], and using subthreshold operation [81]. Additionally, many techniques for randomizing power data have been proposed [82-93].

4.3.2 Timing-Based Attacks

The principle of timing-based attacks is very similar to power-based ones except these attacks rely on timing fluctuations of the target circuit while processing different data patterns. Depending on the load capacitance and driving strength, the charge/discharge process during the switching activities at an internal circuit node will take different amounts of time to finish, which in turn causes different timing delays. First introduced in [94], Timing Analysis (TA) attacks have demonstrated their success on RSA [95], DES [96], AES [97], RSA with Montgomery multiplications [98], and GPS systems [99]. Existing countermeasures include inserting dummy operations [100], using redundant representation [101], and unifying the multiplication operands [102].

4.3.3 Electromagnetic-Based Attacks

Due to the inevitable existence of parasitic reactance, electrical current flowing through a switching CMOS gate causes a variation in the electromagnetic (EM) field surrounding the chip, which can be monitored by antennas particularly sensitive to the related impulse [103]. Similar statistical analysis methods can be applied utilizing EM variances while the target chip is
processing different data. Simple and Differential Electromagnetic Attacks (SEMA and DEMA) have been successfully implemented to crack DES [104-105], RC4 [106], AES [107], and Ellipse Curve Cryptosystems [108-109], on both FPGAs [108-109] and Smart-Cards [110]. Although some power-balancing methods also reduce EM fluctuations, masking EM variance is more difficult due to increased difficulty in matching parasitic reactance. EM attack countermeasures include signal strength reduction and signal information reduction [111].

4.3.4 Fault-Based Attacks

Unlike the previous three passive attacks, fault-based attacks are semi-active in that attackers need to perform certain unusual operations to induce faults inside the target circuit. During the existence of faults, the circuit outputs as well as the side-channel information will be monitored and Differential Fault Analysis (DFA) will be applied to perform the attack, the effectiveness of which has been demonstrated on DES [112], RSA [113-117], Ellipse Curve Cryptosystems [118-120], AES [121-125], Common Scrambling Algorithm [126-127], and RC4 [128]. Fault-injection methods can be classified as non-invasive (variations in supply voltage, external clock, and/or temperature), semi-invasive (exposure to white light, lasers, X-rays, and EM fields), and invasive (ion beams, active probes, and circuit modification) [129]. In general, most fault-tolerant design techniques, such as temporal and spatial redundancy, can be applied to mitigate certain types of faults. These techniques include Concurrent Error Detection (CED) [130-139], error detection/correction code [140-152], modular redundancy [153-154], Built-In Self-Test (BIST) [155], and algorithm modification [156-162]. In addition, the use of dual-rail encoding and its fault analysis can be found in [163-168].
4.4 Circuit-Level Side-Channel Attack Mitigation

Delay-insensitive (DI) asynchronous (clockless) design styles, such as NULL Convention Logic (NCL) [169] and others [170-175], require very little, if any, timing analysis to ensure correct operation. Most DI paradigms [169-175] utilize multi-rail signals, such as dual-rail logic, to achieve delay-insensitivity. For Delay-Insensitive (DI) methods, separating two adjacent DATA wavefronts by a NULL waveform [176] guarantees that there are always two switching events for each dual-rail signal for every DATA processed, thereby decoupling the total number of switching events from the data being processed. However, as pointed out in previous work [177], the imbalanced load between the two rails still causes considerable power/timing/EM fluctuations among different data patterns. Using the single rail per bit DITL methodology at the circuit level has the advantage that power, timing, and emissions can be more easily balanced to prevent attacks compared to dual-rail delay-insensitive methods.

4.4.1 Side-channel Attack Mitigation Using DITL

Since DITL only has one output wire, timing, power, and EM can be more easily balanced because each signal will only drive a single capacitance as in the case of DITL Version II architecture discussed earlier (Figure 19); and a gate’s output will always make a $\frac{1}{2}$ Vdd transition every DATA and NULL cycle, regardless of the DATA value (i.e., $\frac{1}{2}$ Vdd $\rightarrow$ Vdd $\rightarrow$ $\frac{1}{2}$ Vdd for a N$\rightarrow$D1$\rightarrow$N transition and $\frac{1}{2}$ Vdd $\rightarrow$ 0 $\rightarrow$ $\frac{1}{2}$ Vdd for a N$\rightarrow$D0$\rightarrow$N transition). Since each DITL Version II gate input always drives exactly one Is-DATA component, the type of gate being driven will not affect the load capacitance, such that the output driving strength selection of a DITL gate only depends on the number of gates it drives, which substantially reduces the number of balanced gates needed for a chip design library.
As proof of concept, a series of full adders were designed in Boolean, NCL, and DITL, using the IBM 8RF-DM 0.13µm process. The Boolean Full Adder (FA) is a standard gate-level design consisting of five logic gates, as shown in Figure 20. For the NCL FA, two versions have been designed: one is a 10-threshold-gate design based on utilizing complete logic functions to directly implement Figure 20, denoted as NCL-10G; the other is an optimized 4-threshold-gate design, denoted as NCL-4G. Being compatible with its Boolean counterpart, the DITL FA also consists of five gates with different driving strengths, balanced for timing/power through proper transistor sizing. To balance timing and power, transistors were sized to yield similar output→DATA and output→NULL times, propagation delays, peak current spike during transitions, and energy, for all possible transitions. Note that the first two NAND2 gates in Figure 20, denoted by “1”, are sized with driving strength of one gate, while the last NAND2 gate, denoted by “2”, has a driving strength of two gates, since it will be used to drive the $C_{in}$ input of a subsequent FA, connected in ripple-carry fashion.

![Figure 20: Secure Full Adder design](image)

Simulations of Balanced DITL NAND2(1), NAND2(2), and XOR2 gates yielded the results shown in Table IV. Explaining the case of one of the DITL gates in the Table, the output→DATA0/1 times were made to be as close as possible to each other when all of the four input patterns possible for the 2-input gate were applied to the gate. This would require
appropriately sizing and balancing the Pfet network that sets the output to DATA1 and the Nfet network that resets the output to DATA0 in the DITL Version II gate shown previously in Figure 19. Likewise, the output→NULL times were made similar to each other over all four input patterns by sizing and balancing the network for $\frac{1}{2}$ Vdd. It was found that using pass transistor gates instead of two Nfets in series to channel $\frac{1}{2}$ Vdd would be best suited to yield better balanced times.

The Energy over an entire operation, where each NULL→DATA0/1→NULL would qualify as a single operation, was made to be as close as possible to each other over all four possible operations. To do this without changing the time balanced transistors of the DITL gate pull up and pull down networks, extra inverter like small circuits were introduced which will dissipate power selectively. These circuits will be controlled by outputs of the Is-DATA component such that for some selected input patterns these circuits will turn ON to dissipate power while not serving any logical function. The final result is that all input patterns produce almost the same energy consumption for the DITL gate over an entire operation. To match peak current spikes for each operation, extra inverters that serve no logical function can be added that turn ON for selected input patterns and create a similar reading for every input pattern. In short for Table IV, all values that appear in a single row need to be as close as possible to each other and the individual DITL gates were modified with this objective in mind.

After balancing the NAND2 and XOR2 gates, they were put together as shown in the previous Figure 20 to form a DITL 5-gate balanced Full Adder (FA). The simulations of the DITL FA yielded the results summarized in Table V. No further balancing or transistor sizing was done on the Full Adder. In Table V, as expected, the values in a single row are very close to each other over all the eight possible FA input patterns.
TABLE IV: Measurements from Balanced DITL gates

<table>
<thead>
<tr>
<th>Input Pattern</th>
<th>N→00→N</th>
<th>N→01→N</th>
<th>N→10→N</th>
<th>N→11→N</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DITL Nand2</strong> (1)</td>
<td>Output→DATA (ps)</td>
<td>519.9</td>
<td>546.1</td>
<td>529.5</td>
</tr>
<tr>
<td></td>
<td>Output→NULL (ps)</td>
<td>565.5</td>
<td>574.5</td>
<td>552.5</td>
</tr>
<tr>
<td></td>
<td>Energy (J)</td>
<td>3.63E-14</td>
<td>3.76E-14</td>
<td>3.75E-14</td>
</tr>
<tr>
<td></td>
<td>Current Spike (uA)</td>
<td>55</td>
<td>54</td>
<td>55</td>
</tr>
<tr>
<td><strong>DITL Nand2</strong> (2)</td>
<td>Output→DATA (ps)</td>
<td>659.5</td>
<td>691.4</td>
<td>670.7</td>
</tr>
<tr>
<td></td>
<td>Output→NULL (ps)</td>
<td>682.2</td>
<td>678</td>
<td>660.2</td>
</tr>
<tr>
<td></td>
<td>Energy (J)</td>
<td>3.84E-14</td>
<td>3.94E-14</td>
<td>3.93E-14</td>
</tr>
<tr>
<td></td>
<td>Current Spike (uA)</td>
<td>55</td>
<td>54</td>
<td>54</td>
</tr>
<tr>
<td><strong>DITL Xor2</strong></td>
<td>Output→DATA (ps)</td>
<td>783.9</td>
<td>780.7</td>
<td>779.7</td>
</tr>
<tr>
<td></td>
<td>Output→NULL (ps)</td>
<td>636.8</td>
<td>617.6</td>
<td>625.5</td>
</tr>
<tr>
<td></td>
<td>Energy (J)</td>
<td>4.46E-14</td>
<td>4.41E-14</td>
<td>4.42E-14</td>
</tr>
<tr>
<td></td>
<td>Current Spike (uA)</td>
<td>71</td>
<td>58</td>
<td>59</td>
</tr>
</tbody>
</table>
TABLE V: Measurements from Balanced DITL Full Adder

<table>
<thead>
<tr>
<th>Input Pattern</th>
<th>N→000→N</th>
<th>N→001→N</th>
<th>N→010→N</th>
<th>N→011→N</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DITL FA Sum output</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>→DATA (ps)</td>
<td>645.1</td>
<td>665</td>
<td>665.8</td>
<td>665.4</td>
</tr>
<tr>
<td>→NULL (ps)</td>
<td>487.9</td>
<td>488.3</td>
<td>539.2</td>
<td>484.7</td>
</tr>
<tr>
<td><strong>DITL FA Carry output</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>→DATA (ps)</td>
<td>587.1</td>
<td>587.1</td>
<td>589.9</td>
<td>606.1</td>
</tr>
<tr>
<td>→NULL (ps)</td>
<td>562</td>
<td>566.2</td>
<td>562.4</td>
<td>606.1</td>
</tr>
<tr>
<td><strong>Energy (J)</strong></td>
<td>3.05E-13</td>
<td>3.03E-13</td>
<td>3.10E-13</td>
<td>2.90E-13</td>
</tr>
<tr>
<td><strong>Current Spike (uA)</strong></td>
<td>301</td>
<td>277</td>
<td>289</td>
<td>283</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Pattern</th>
<th>N→100→N</th>
<th>N→101→N</th>
<th>N→110→N</th>
<th>N→111→N</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DITL FA Sum output</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>→DATA (ps)</td>
<td>677.9</td>
<td>657.8</td>
<td>636.7</td>
<td>665.8</td>
</tr>
<tr>
<td>→NULL (ps)</td>
<td>537.8</td>
<td>491.6</td>
<td>483.8</td>
<td>487.1</td>
</tr>
<tr>
<td><strong>DITL FA Carry output</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>→DATA (ps)</td>
<td>585.6</td>
<td>600</td>
<td>596.1</td>
<td>592.2</td>
</tr>
<tr>
<td>→NULL (ps)</td>
<td>557.1</td>
<td>605.5</td>
<td>601.5</td>
<td>602.3</td>
</tr>
<tr>
<td><strong>Energy (J)</strong></td>
<td>3.11E-13</td>
<td>2.91E-13</td>
<td>2.87E-13</td>
<td>2.84E-13</td>
</tr>
<tr>
<td><strong>Current Spike (uA)</strong></td>
<td>290</td>
<td>284</td>
<td>277</td>
<td>290</td>
</tr>
</tbody>
</table>
Table VI shows the maximum variance percentage of each parameter among all possible input combinations, and compares the DITL FA to the NCL and Boolean Full Adders. These four Full Adders are simulated in Cadence Spectre and are compared in five categories: “Sum/C\textsubscript{out} transition slope” is the combined rise/fall time during each transition for Sum and C\textsubscript{out} outputs, respectively; “delay” is the total time for a N→D→N cycle; “peak current spike” is the magnitude of the supply voltage current spike during each transition; and “energy” is the total energy consumed during each transition.

**TABLE VI: Secure Full Adder comparison**

<table>
<thead>
<tr>
<th>Full Adder</th>
<th>Maximum Variance Percentage</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Sum Transition Slope</strong></td>
<td><strong>C\textsubscript{out} Transition Slope</strong></td>
<td><strong>Delay</strong></td>
<td><strong>Peak Current Spike</strong></td>
<td><strong>Energy</strong></td>
</tr>
<tr>
<td>Boolean</td>
<td>27.8%</td>
<td>11.4%</td>
<td>93.6%</td>
<td>221.4%</td>
<td>313.4%</td>
</tr>
<tr>
<td>NCL-4G</td>
<td>21.0%</td>
<td>13.0%</td>
<td>105.3%</td>
<td>51.0%</td>
<td>32.0%</td>
</tr>
<tr>
<td>NCL-10G</td>
<td>12.9%</td>
<td>58.4%</td>
<td>19.0%</td>
<td>47.2%</td>
<td>10.4%</td>
</tr>
<tr>
<td>DITL</td>
<td>8.5%</td>
<td>5.6%</td>
<td>13.8%</td>
<td>18.1%</td>
<td>7.4%</td>
</tr>
</tbody>
</table>

Although NCL as a dual-rail asynchronous logic is well-known to be more side-channel attack resistant compared to Boolean logic, the DITL design exhibits the least variations in all parameters, as shown in Table VI. Since power (energy and current spike) and timing (slope and delay) are significantly more balanced for DITL, DPA and TA will be much more difficult to
succeed. This demonstrates DITL’s capability in balancing power and timing with different driving strengths in a multi-gate circuit, which validates the DITL cell library development strategy undertaken.

4.5 Achieved Team Research Objectives

4.5.1 DITL Secure ALU Design

Utilizing the methods to develop the DITL secure Full Adder discussed in Section 4.4.1, a DITL balanced gate library was created to be used to design a DITL Secure 8051 Arithmetic Logic Unit (ALU). The gate library consisted of timing and power balanced DITL circuits for Half Adder, Full Adder, 2:1 Multiplexers, up to 4-input versions of NAND and NOR gates, XOR2 and XNOR2 gates, and several inverters and buffers with a variety of drive strengths. The developed library also contained C-elements [13] for use in the completion circuitry to conjoin multiple $L_{ack}$ signals together and Ternary buffer circuits to increase drive strength of Ternary signals. All the DITL gates were created at the transistor level and simulated using the IBM PDK 1.2V 130nm 8rfDM process and then layouts were created. The Boolean 8051 ALU was designed in HDL and all of the Boolean gates were replaced by DITL equivalents in the netlist with connections for $Rack$ and $Lack$ handshaking signals included.

4.5.2 DITL ALU Simulation and Results

The DITL ALU verilog netlist was imported into Cadence as a transistor level design and simulations were done. The DITL ALU schematic is not included here as it is much too large to be intelligible, but Figure 21 shows the test bench used to simulate the design. It contains the symbol for ALU and a VerilogA controller, which gives inputs to the ALU. Figure 22 shows the
Ultrasim simulation waveform, which gives the supply current, handshaking signals \textit{Rack} and \textit{Lack}, one of the many inputs \textit{Tmp1bus<0>} and one of the many outputs \textit{resultH<0>}. The simulation is shown for eight different DATA-NUL input patterns.

The ALU simulation results are tabulated in Table VII. The simulation time was calculated for each DATA wavefront to produce an output DATA and each NULL wavefront to produce an output NULL, for all of the input cases and the total simulation time was summed and averaged over 8 different operations, giving the Average delay of the DITL ALU per operation, also called TDD or DATA-To-DATA cycle time.

![Figure 21: DITL ALU Testbench](image)

37
The total dynamic energy was found by integrating the current waveform for both Vdd and $\frac{1}{2}$ Vdd (not shown in Figure 22 but similar to I(Vdd)) over the entire simulation time, multiplying it by the values of 1.2 V and 0.6V respectively, averaging this over the number of operations and taking the sum to obtain Dynamic Energy per operation. The $\frac{1}{2}$ Vdd dynamic energy value was negligible compared to Vdd.

The static power readings during the time outputs of the ALU are all DATA and all NULL were found separately by obtaining the current values for different cases from Vdd and $\frac{1}{2}$ Vdd current waveforms, averaging these, and multiplying by the 1.2V and 0.6V values and summing them.

Figure 22: DITL ALU Ultrasim simulation
Once the simulation was successful, a layout was created for the DITL ALU using Virtuoso Layout editor. Figure 23 shows the DITL ALU Layout, which is ready to be integrated into a Layout plan with pads and be taped out for fabrication.

<table>
<thead>
<tr>
<th></th>
<th>Average Delay/Operation TDD (nS)</th>
<th>Average Dynamic Energy/Operation (nJ)</th>
<th>Static Power for DATA (uW)</th>
<th>Static Power for NULL (uW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DITL ALU</td>
<td>136</td>
<td>0.178</td>
<td>137</td>
<td>87.5</td>
</tr>
</tbody>
</table>

**Figure 23: DITL ALU Layout**
5. CARBON NANO TUBE FET BASED DITL DESIGN

Carbon Nano-Tube (CNT) FET is a possible alternative to Multi-Threshold CMOS process for designing circuits that require a range of threshold voltages. The threshold voltage of a CNT FET can be finely adjusted by changing its diameter. Therefore, a multi-threshold design can be implemented using CNT FETs of different diameters. DITL circuits need multi-threshold FET circuits to distinguish DATA0 and DATA1 values from NULL as previously discussed in Section 3.1. A CNT FET is like a normal FET such that it has gate, source, drain, and body terminals, and has a threshold voltage that determines when the FET turns ON and OFF and acts like a signal switch. CNT FETs are composed of Carbon Nano-Tubes. A CNT is characterized by its diameter or Chirality factor, which is defined as the angle of atom arrangement along the tube determined by the integer pair \((n, m)\). To use a Single Wall CNT as a semi-conductor FET, values for \(n\) and \(m\) should be chosen such that \(n /= m\) and the difference \(n-m /= \) integer multiple of 3, otherwise the CNT will act as a metallic conductor.

The diameter of the CNT is given as a function of integers \(n\) and \(m\) as
\[
D_{\text{CNT}} = 0.078328*(n^2 + m^2 + nm)^{1/2} \text{ nanometers}
\]

The Threshold voltage of a CNT FET with diameter \(D_{\text{CNT}}\) in \(\text{nanometers}\) is
\[
V_t = (0.43602/D_{\text{CNT}}) \text{ Volts}
\]

The CNT FET spice models were obtained from Stanford University [178] and they use a 0.9V power supply. The basic model of a CNT FET can be easily modified for \((n,m)\) values to obtain a custom threshold voltage. This chapter discusses some of the methods by which DITL circuits can be implemented using CNT FETs. The CNT Spice models from Stanford were used to simulate the CNT FET circuits in Hspice. The simulations yielded Energy dissipation values.
and output waveform files which were analyzed in Saber Cosmoscope to measure rise and fall times.

5.1 Previous Work using CNT FET and Ternary Logic

The basis of creating a new architecture for Ternary voltage circuits with CNT FETs starts with the consideration of a Single Ternary Inverter explained in [179] and shown in Figure 24 below, which uses CNT FETs to invert a DATA0 to DATA1, DATA1 to DATA0, and NULL to NULL.

![Figure 24: CNT FET Single Ternary Inverter](image)

The Single Ternary Inverter employs CNT FETs with different threshold voltages and their corresponding (n,m) Chirality values. When the input IN is 0.9V (DATA1), Nfets with (19,0) and (10,0) turn ON and pull down output OUT to 0V (DATA0). When IN is 0V, Pfets
with (19,0) and (10,0) turn ON and pull up OUT to 0.9V. When IN is 0.45V (NULL), Nfets and Pfets with (19,0) and (13,0) will turn ON creating a diode connected FET path from Vdd to Gnd, so that OUT is held at 0.45V.

The team that designed the Single Ternary Inverter went on to design logic gates and arithmetic circuits utilizing Ternary Logic and CNT FETs [180], where a Ternary encoding scheme is used instead of a Binary one to do arithmetic logic. However, their design is intended to be used as a replacement for Binary CMOS logic in Synchronous systems and does not implement a delay-insensitive asynchronous system, such as DITL.

5.2 New CNT DITL Architecture using Diode Connected CNT FETs

All of the previously discussed DITL gate architectures used a ½ Vdd supply for generating the output state of NULL. Based on the diode connected CNT FET mechanism used in the Single Ternary Inverter in Figure 24, a new DITL gate architecture called DITL-TI was designed that used a single Vdd supply and specially designed multi-threshold CNT FETs.

A DITL-TI NAND2 gate is shown in Figure 25. The circuit is divided into groups of FETs as shown by shaded areas. The un-shaded parts of the circuit show the diode connected FETs (which will hold their drains on node Z at NULL when their respective sources are simultaneously connected to Vdd and Gnd) and also Ternary Inverters for primary inputs X and Y and output Zb. Since no separate circuits are used to differentiate NULL from DATA, this objective is achieved by pairs of series connected FETs having 0.2 Vt. For example, for the Hold NULL network, series connected Pfet pair for X and Xb signals will simultaneously turn ON when both X and Xb are NULL and not for any other combination. Similarly, series connected 0.2 Vt FETs for pairs (X, Xb), (Y, Yb), and (Z, Zb) in both Pfet and Nfet networks will only turn ON simultaneously when corresponding signals in the pair are both NULL.
For the Set NULL network, ideally the Pfet and Nfet networks each should have had both 
(X, Xb) and (Y, Yb) pairs controlling four FETs to detect an input complete NULL that will set Z 
to NULL. But in the case of NAND2, the number of FETs required can be reduced to two per Pfet and Nfet Set NULL networks. The reason is because when Xb and Yb are both DATA0 or DATA1, only one of either the Pfet or Nfet network will turn ON and this would not result in a NULL at Z, instead it will provide an alternate route for the Set DATA And Logic. Thus, while

Figure 25: DITL-TI NAND2 Gate
minimizing the number of Set NULL FETs, care should be taken to check for input completeness
and also make sure the minimized network will either work with the Set DATA Logic or does
not turn ON in any case except when Z should be NULL.

The And Logic Section of the DITL-TI NAND2 gate sets the node Z as per AND2 logic,
which will create an inverted NAND2 function at Zb. The And Logic uses CNT FETs with 0.55
Vt that will only turn ON for either DATA1 or DATA0, and stay OFF for NULL. Hence, the
ternary signals can be directly connected to these FETs. To ensure input completeness for the set
DATA network, both the inputs must be represented in each pull up and pull down path. Thus,
the NAND2 Set Logic for an AND2 gate would need 8 FETs instead of the Boolean 4 FET
design. The Hold DATA network also uses 0.55 Vt CNT FETs. The Hold DATA network
implementation depends on the cases where one input is DATA0/1 while another is NULL. This
includes cases such as when the NAND2 output node Zb has become DATA1, one of the inputs
X and Y can be DATA 0/1 while other has transitioned to NULL, and when Zb has become
DATA0, one of X or Y can stay at DATA1 while the other transitions to NULL.

Similar to the NAND2 gate, a DITL-TI XOR2 gate design is shown in Figure 26. The
Set NULL network for the XOR2 gate has three Nfets to ensure input completeness. The Hold
NULL network is the same as the NAND2 gate. The XNOR2 Set Logic is used to set Z, which
is inverted as XOR2 at Output Zb. The Hold DATA network has two additional Nfets compared
to the NAND2 gate.

The Hspice simulation of the NAND2 and XOR2 DITL-TI gates yielded the waveforms
shown in Figures 27 and 28, respectively. Both simulation waveforms show hysteresis
functionality of output Zb, state holding for NULL at time 40 ns where the output remains at
NULL after Y changes to DATA while X remains NULL. A similar case is at 70 ns where Zb
remains DATA after $Y$ changes to NULL while $X$ remains DATA. It should be noted that both the NAND2 and XOR2 gates discussed here are pure combinational gates with no embedded registration. These gates can be used with a separate register circuit discussed later to form sequential DITL logic. Embedded registration can be included into the DITL-TI gates to create the NAND2 Register shown in Figure 29 and XOR2 Register in Figure 30. The additional handshaking signals $K_i$ and $K_o$ are added to the circuit and the gate functions the same way as the original CMOS DITL gates discussed in Section 3.2.

Figure 26: DITL-TI XOR2 Gate

45
Figure 27: Output waveform for DITL-TI NAND2 Gate

Figure 28: Output waveform for DITL-TI XOR2 Gate
The general architecture of the DITL-TI Register is similar to the DITL-TI Gate with FETs for handshaking signal $K_i$ added in Set NULL, Hold NULL, Set Logic DATA, and Hold DATA networks, and generation of the $K_o$ signal from the output using an Is-NULL component. In this case the Set NULL network will only be enabled if there is a $K_i$ request-for-NULL and the Hold NULL network will be kept ON if the $K_i$ request-for-NULL is still active. Similarly, the Set Logic DATA network will only be enabled if there is a $K_i$ request-for-DATA and the

**Figure 29: DITL-TI NAND2 Register**

The general architecture of the DITL-TI Register is similar to the DITL-TI Gate with FETs for handshaking signal $K_i$ added in Set NULL, Hold NULL, Set Logic DATA, and Hold DATA networks, and generation of the $K_o$ signal from the output using an Is-NULL component. In this case the Set NULL network will only be enabled if there is a $K_i$ request-for-NULL and the Hold NULL network will be kept ON if the $K_i$ request-for-NULL is still active. Similarly, the Set Logic DATA network will only be enabled if there is a $K_i$ request-for-DATA and the
Hold DATA network will be kept ON if $K_i$ request for DATA is still active. Note that the $K_i$ signal needs a Boolean inverter.

![DIagram of DITL-TI XOR2 Register](image)

**Figure 30: DITL-TI XOR2 Register**

discussed in Section 3.1 but uses a NOR2 Boolean gate in place of NAND2 to create a $K_o$ of logic 1 or request-for-DATA for an output $Z_b$ of NULL and $K_o$ of logic 0 or request-for-NULL for an output $Z_b$ of DATA0/1. The Hspice simulations of DITL-TI NAND2 and XOR2 registers yielded the waveforms shown in Figures 31 and 32, respectively.
Figure 31: Output waveform for DITL-TI NAND2 Register

Figure 32: Output waveform for DITL-TI XOR2 Register
Based on the DITL-TI register design, a Single Bit register shown in Figure 33 was created to be used with combinational DITL-TI gates. The Hspice simulation yielded the waveform shown in Figure 34.

Figure 33: DITL-TI Single Bit Register
5.3 CNT DITL using Detect Circuits

Similar to the Detect0 and Detect1 circuits discussed in Section 3.1, CNT Detect circuits can be created as shown in Figure 35. The Detect0 and Detect1 circuits were implemented using CNT FETs with threshold voltages adjusted such that they yield minimum static and leakage power dissipation with acceptable switching speed. With Vdd at 0.9 V, the Detect0 outputs a logic 1 only when its input is logic 0 and Detect1 outputs a logic 0 only for an input of logic 1. The Detect0 FETs will be turned OFF when IN is between 0.25 V to 0.35 V. The Detect1 FETs will be turned OFF when IN is between 0.55 V and 0.65 V. Therefore, dynamic power dissipation is minimal for both circuits.

Figure 34: Output waveform for DITL-TI Single Bit Register
The CNT Detect0 and Detect1 circuits can be used to detect ternary voltages at the input; hence, a CNT DITL architecture similar to CMOS DITL registers discussed in Section 3.2 can be designed. The Version I and II of the Original DITL NAND2 shown previously in Figures 17 and 19 can be directly implemented using CNT FETs. The Specific functions in Version I of DITL NAND2 would need to use 0.55 Vt CNT FETs since they are directly connected to ternary voltage inputs. The FETs in pre-charge to NULL and the evaluate function networks can use 0.2 Vt CNT FETs for faster speed. All other FETs in the DITL circuit including inside the TH22, inverter, and Is-DATA components can use normal 0.25 Vt CNT FETs. For the Version II DITL NAND2, since Is-DATA components provide the signals, the specific function network can use 0.2 Vt CNT FETs for even faster performance. Figure 36 shows the simulation waveform for a CNT DITL NAND2 version I and II, which is directly based on the original CMOS design.
Similar to the original CMOS DITL architecture established in Section 3.2, modifications in the evaluate and hold data networks would result in a DITL-New Version I NAND2 register shown in Figure 37 and Version II NAND2 register shown in Figure 38. Here also, Version II used the outputs of Is-DATA components to replace the primary inputs for the specific function networks. The modifications compared to the original circuit include the absence of an output completion TH22 gate, change in complexity of the Hold DATA1/0 networks, and difference in the way the Set NULL network is implemented. The output waveforms for the DITL-New NAND2 registers match the waveform in Figure 31 shown previously for the DITL-TI NAND2 register.

Figure 36: Output Waveform for Original CNT DITL NAND2 Register
Comparing the waveforms in Figures 31 and 36, it can be seen that both the Original DITL NAND2 and DITL-New (and DITL-TI) NAND2 register circuits behave the same way except when the output is holding NULL in the time range of 65 ns to 80 ns. For the original DITL circuit, the output can become NULL as soon as $Ki$ and $Ko$ are requesting NULL. But the DITL-new registers and DITL-TI registers implement it such that both inputs have to become NULL first before the output becomes NULL.

Figure 37: DITL-New Version I NAND2 Register
The registration part of the CNT DITL architecture can be removed to yield the CNT DITL NAND2 combinational gates Version I and II, shown in Figures 39 and 40, respectively. The simulation of these gates will yield a similar waveform as previously shown in Figure 27. To be used with these combinational gates, a DITL-New Version I and II single bit register can be designed as shown in Figure 41 and 42, respectively. Its simulation will yield a similar waveform as previously shown in Figure 34 for DITL-TI single bit register.

Figure 38: DITL-New Version II NAND2 Register

The registration part of the CNT DITL architecture can be removed to yield the CNT DITL NAND2 combinational gates Version I and II, shown in Figures 39 and 40, respectively. The simulation of these gates will yield a similar waveform as previously shown in Figure 27. To be used with these combinational gates, a DITL-New Version I and II single bit register can be designed as shown in Figure 41 and 42, respectively. Its simulation will yield a similar waveform as previously shown in Figure 34 for DITL-TI single bit register.
Figure 39: DITL Version I NAND2 Combinational Gate

Figure 40: DITL Version II NAND2 Combinational Gate
Figure 41: DITL-New Version I Single Bit Register

Figure 42: DITL-New Version II Single Bit Register
5.4 CNT DITL Simulation Results and Commentary

The CNT DITL circuits mentioned in Sections 5.2 and 5.3 were simulated in Hspice and results for dynamic energy dissipation and rise and fall times tabulated in Tables VIII, IX, and X for a preliminary comparison.

Comparing between the different DITL register methods in Tables VIII and IX, it can be seen that DITL-New circuits perform the best in terms of power and delay while DITL-TI circuits perform worst. The DITL-TI circuits all have significant static power dissipation due to the method of diode connecting FETs to achieve a NULL at the output. Figure 43 shows the Vdd source current waveform and primary output for a DITL-TI NAND2 register. It can be seen that whenever the output is at NULL, the static current is at least 5uA. In Tables VIII and IX, comparing between best cases on DITL-New Version I and II circuits, Version I circuits consume less power because of using 0.55 Vt CNT FETs for the Logic network and Version II circuits are faster but more power consuming because of using 0.2Vt CNT FETs.

Among the designs shown in Table X, a similar comparison can be made that DITL-TI gates are worst performers due to significant static power consumption. DITL Version I gates perform best in terms of energy while DITL Version II gates perform best in terms of speed. As a demonstration of CNT DITL used in a bigger circuit, a set of Full Adders were simulated using the methods and gates described previously. Four kinds of Full Adder (FA) circuits were simulated:

(1) The Five gate FA example shown in Figure 20 using CNT DITL NAND2 and XOR2 gates and 1Bit registers as discussed in Tables IX and X
TABLE VIII: Measurements for CNT DITL NAND2 Registers

<table>
<thead>
<tr>
<th>Gate Type</th>
<th>Dynamic Energy (fJ)</th>
<th>Static Power DATA (nW)</th>
<th>Static Power NULL (nW)</th>
<th>Rise and Fall Times in (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N→Vdd</td>
</tr>
<tr>
<td>DITL-TI NAND2 register</td>
<td>153</td>
<td>24.1</td>
<td>3730</td>
<td>12.8</td>
</tr>
<tr>
<td>DITL V1 NAND2 register (Original)</td>
<td>7.77</td>
<td>58.1</td>
<td>14.9</td>
<td>18.2</td>
</tr>
<tr>
<td>DITL V2 NAND2 register (Original)</td>
<td>9.26</td>
<td>19.7</td>
<td>21.4</td>
<td>15.2</td>
</tr>
<tr>
<td>DITL V1 NAND2 register (New)</td>
<td>6.66</td>
<td>7.99</td>
<td>32.2</td>
<td>14.9</td>
</tr>
<tr>
<td>DITL V2 NAND2 register (New)</td>
<td>8.23</td>
<td>11.1</td>
<td>11.5</td>
<td>11.6</td>
</tr>
</tbody>
</table>
### TABLE IX: Measurements for CNT DITL XOR2 and 1Bit Registers

<table>
<thead>
<tr>
<th>Gate Type</th>
<th>Dynamic Energy (fJ)</th>
<th>Static Power DATA (nW)</th>
<th>Static Power NULL (nW)</th>
<th>Rise and Fall Times in (pS)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N→Vdd</td>
</tr>
<tr>
<td>DITL-TI Xor2 register</td>
<td>138</td>
<td>16.2</td>
<td>3360</td>
<td>14.7</td>
</tr>
<tr>
<td>DITL V1 Xor2 register (Original)</td>
<td>8.35</td>
<td>21.2</td>
<td>36.8</td>
<td>23.1</td>
</tr>
<tr>
<td>DITL V2 Xor2 register (Original)</td>
<td>9.85</td>
<td>10.6</td>
<td>15.8</td>
<td>20.0</td>
</tr>
<tr>
<td>DITL V1 Xor2 register (New)</td>
<td>7.20</td>
<td>24.8</td>
<td>27.9</td>
<td>19.9</td>
</tr>
<tr>
<td>DITL V2 Xor2 register (New)</td>
<td>8.88</td>
<td>9.9</td>
<td>10.4</td>
<td>16.7</td>
</tr>
<tr>
<td>DITL-TI 1Bit register</td>
<td>105</td>
<td>11.9</td>
<td>2550</td>
<td>12.3</td>
</tr>
<tr>
<td>DITL V1 1Bit register (New)</td>
<td>3.76</td>
<td>22.7</td>
<td>36.9</td>
<td>15.8</td>
</tr>
<tr>
<td>DITL V2 1Bit register (New)</td>
<td>4.64</td>
<td>6.98</td>
<td>12.6</td>
<td>12.1</td>
</tr>
</tbody>
</table>
2) The 5-gate FA using CNT DITL NAND2 and XOR2 registers as discussed in Tables VIII and IX.

3) Full Adder using 1Bit registers and FA-Sum and FA-Carry custom logic DITL gates based on the DITL gates in Tables IX and X.

4) Full Adder using FA-Sum and FA-carry custom logic DITL registers based on the DITL registers in Tables VIII and IX.
The simulation waveform for a CNT DITL Full Adder circuit is shown in Figure 44 and it looks the same for all four types of FA circuits. It can be seen that the Full Adder displays hysteresis by state holding NULL at 40 ns and DATA at 70 ns. Tables XI and XII shows results from simulations of CNT DITL Full Adder circuits.
Considering Table X, the low energy FA circuits used DITL Version I gates from Table X and DITL-New Version I registers from Table IX, while the speedier FA circuits used DITL Version II gates from Table X and DITL-New Version II registers from Table IX. For Table XI, DITL FA SC gates were based on DITL Version I and Version II gates from Table X for energy and speed cases respectively, and DITL FA SC registers were based on DITL-New Version I and Version II gates from Table VIII for energy and speed cases respectively.

The results in Tables X and XI show that the DITL FA 5 gates + 1Bit register circuits have best cases of Rise and Fall times for NULL to DATA1/0 back to NULL, due to the smallest
series transistor depth in its final stage, which is a maximum of four series FETs, but they also have very high dynamic energy and power consumption.

**TABLE XI: Measurements for CNT DITL Full Adders Type 1 and 2**

<table>
<thead>
<tr>
<th>Gate Type</th>
<th>Dynamic Energy (fJ)</th>
<th>Rise and Fall Times in (pS)</th>
<th>Delay (pS)</th>
<th>Power (nW)</th>
<th>Power (nW)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>N→Vdd</td>
<td>Vdd→N</td>
<td>N→Gnd</td>
<td>Gnd→N</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DITL FA 5 gates + 1Bit Reg (Energy)</td>
<td>77.6</td>
<td>s</td>
<td>13.8</td>
<td>2.6</td>
<td>15.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c</td>
<td>9.6</td>
<td>2.7</td>
<td>15.9</td>
</tr>
<tr>
<td>DITL FA 5 gates + 1Bit Reg (Speed)</td>
<td>93.5</td>
<td>s</td>
<td>10.0</td>
<td>2.6</td>
<td>10.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c</td>
<td>6.8</td>
<td>2.6</td>
<td>9.5</td>
</tr>
<tr>
<td>DITL FA 5 Registers (Energy)</td>
<td>76.9</td>
<td>s</td>
<td>19.9</td>
<td>6.8</td>
<td>19.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c</td>
<td>16.1</td>
<td>6.8</td>
<td>22.0</td>
</tr>
<tr>
<td>DITL FA 5 Registers (Speed)</td>
<td>92.9</td>
<td>s</td>
<td>16.6</td>
<td>6.7</td>
<td>15.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c</td>
<td>12.3</td>
<td>6.8</td>
<td>14.4</td>
</tr>
</tbody>
</table>
**TABLE XII: Measurements for CNT DITL Full Adders Type 3 and 4**

<table>
<thead>
<tr>
<th>Gate Type</th>
<th>Dynamic Energy (fJ)</th>
<th>Rise and Fall Times in (pS)</th>
<th>Delay (pS)</th>
<th>Power (nW) NULL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>N→Vdd</td>
<td>Vdd→N</td>
<td>N→Gnd</td>
</tr>
<tr>
<td>DITL FA SC gates + 1Bit Reg (Energy)</td>
<td>59.9</td>
<td>s</td>
<td>19.5</td>
<td>2.6</td>
</tr>
<tr>
<td></td>
<td>c</td>
<td>15.5</td>
<td>2.6</td>
<td>17.7</td>
</tr>
<tr>
<td>DITL FA SC gates + 1Bit Reg (Speed)</td>
<td>69.3</td>
<td>s</td>
<td>15.2</td>
<td>2.7</td>
</tr>
<tr>
<td></td>
<td>c</td>
<td>14.8</td>
<td>2.7</td>
<td>14.7</td>
</tr>
<tr>
<td>DITL FA SC Registers (Energy)</td>
<td>39.8</td>
<td>s</td>
<td>27.6</td>
<td>7.0</td>
</tr>
<tr>
<td></td>
<td>c</td>
<td>23.2</td>
<td>7.1</td>
<td>25.0</td>
</tr>
<tr>
<td>DITL FA SC Registers (Speed)</td>
<td>50.0</td>
<td>s</td>
<td>22.4</td>
<td>8.8</td>
</tr>
<tr>
<td></td>
<td>c</td>
<td>22.4</td>
<td>7.1</td>
<td>21.7</td>
</tr>
</tbody>
</table>

DITL FA SC Register circuits have the lowest energy/static power consumption as well as lowest input to output propagation delay. This can be attributed to the single stage design where sum and carry are evaluated in the first stage itself. But since the Sum and Carry custom logic functions have at least eight FETs in series in both Pfet and Nfet Set Data networks, the NULL to DATA1/0 times are highest for DITL FA SC Register circuits.
6. CONCLUSION

6.1 Summary

A new asynchronous paradigm called Delay-Insensitive Ternary Logic (DITL) combining the design aspects of NCL, PCHB, and Boolean gates was developed at the transistor level. DITL uses a single wire per bit, three voltage scheme to represent the three delay-insensitive states of DATA0, DATA1, and NULL. DITL is found to be more energy efficient when compared to similar paradigms like PCHB and NCL when simulated using the IBM 8rf-DM 1.2V 130nm CMOS process. An application for DITL is discussed in Secure Hardware design where circuit level side channel attacks based on measurement of timing, power, and EM signatures of the secure chip can be thwarted by the use of special DITL circuits. These special DITL circuits are designed such that each DITL component of the system is balanced in terms of energy per operation, timing for NULL to DATA to NULL transitions, and EM signatures so that the attacker cannot distinguish between different cases of input patterns. As a proof of concept of Secure DITL design, a 5-gate Full Adder was designed using balanced DITL gates and compared for variance in measurements against NCL and Boolean Full Adders, showing that DITL outperforms the others in resisting side-channel attacks. Using the same methodology, a gate library of balanced DITL gates was created towards the transistor level design and layout of a DITL Secure ALU ready to be fabricated into a chip. The DITL ALU design was implemented at the transistor level and simulated for several input cases, which yielded timing, energy, and power measurements.

Carbon Nano-Tubes that work as FETs were used to build DITL architectures previously implemented using CMOS. By carefully choosing the CNT diameter, easy threshold modification of the FETs was achieved, which made it easier to design ternary detect circuits.
CNT DITL circuits were able to obtain either significant energy savings or increased speed up by employing CNT FETs with carefully crafted threshold voltages, unlike CMOS design where the choice of thresholds is limited. An alternate DITL architecture was also developed that makes use of diode connected CNT FETs to create an output of NULL without a $\frac{1}{2}$ Vdd supply, but it suffered from large static power consumption. A comparison was drawn in terms of energy and timing parameters for the different types of CNT DITL architectures by building a series of Full Adders. Among the full adder designs, the full custom single stage register full adders using a modified version of the original DITL architecture were found to be the best performers in case of dynamic switching energy as well as throughput, which is an indication that with better control of threshold voltages using CNT FETs, complex gates can be made to have both energy savings and speed.

6.2 Future Work

DITL is fundamentally different from the prevailing Boolean logic at the physical-level; therefore, no DITL gate libraries exist in the industry-standard CAD tools. Hence, a full set of DITL libraries at VHDL-, transistor-, and layout-level, offering multiple driving strengths for each gate should be developed. The VHDL-level library will contain the behavioral description of each DITL gate, and will be used for functional simulation. The transistor-level library consists of the transistor schematic of each DITL gate using the IBM 8RF-DM 130nm CMOS process. In addition to functionality, the most important consideration is transistor sizing, which has two purposes: 1) for achieving multiple output driving strengths; and 2) for balancing power and timing during gate switching while driving different fan-outs. The currently available balanced DITL ALU gate library can be expanded as needed for gates with different drive
strengths and to cater to other/larger DITL designs. These three libraries set the foundation of the subsequent tasks.

As stated before, one of the advantages of DITL compared to other asynchronous paradigms is the compatibility with the synchronous circuit design flow due to the fact that Boolean and DITL paradigms utilize the same set of logic functions. Since the size and complexity of modern digital ICs keep growing, an automated DITL design flow using commercial CAD tools is critical for wide adoption of DITL.
REFERENCES


73


[121] C. Giraud, “DFA on AES,” 4th International Conference on AES, 2004


[145] M. Karpovsky, K. Kulikowski, and A. Taubin, “Robust Protection Against Fault-Injection Attacks on Smart Cards Implementing the Advanced Encryption Standard,” pp.93, 2004 International Conference on Dependable Systems and Networks (DSN'04), 2004


