Spintronic Threshold Logic Array (STLA) - A Compact, Low Leakage, Non-volatile Gate Array Architecture

Nishant.S.Nukala, Niranjan Kulkarni and Sarma Vrudhula
School of Computing, Informatics and Decision Systems Engineering,
Arizona State University, Tempe, Arizona 85281, U.S.A.
{nishant.s.nukala, niranjan.kulkarni, vrudhula}@asu.edu

***Abstract*—This paper describes a novel, first of its kind architecture for a threshold logic gate using conventional MOSFETs and an STT-MTJ (Spin Torque Transfer-Magnetic Tunnelling Junction) device. The resulting cell, called STL which is extremely compact can be programmed to realize a large number of threshold functions, many of which would require a multilevel network of conventional CMOS logic gates. Next, we describe a novel array architecture consisting of STL cells onto which complex logic networks can be mapped. The resulting array, called STLA has several advantages not available with conventional logic. This type of logic (1) is non-volatile, (2) is structurally regular and operates like DRAM, (3) is fully observable and controllable, (4) has zero standby power. These advantages are demonstrated by implementing a 16-bit carry look-ahead adder and compared with two optimized conventional FPGA implementations (Carry Look Ahead Adder and Ripple Carry Adder). The STLA has 12X lower transistor count (compared to CLA-FPGA) and 10X reduction (compared to RCA-FPGA) with comparable energy which will continue to reduce as the STT-MTJ device technology matures.**

I. Introduction

A Boolean function $f(x_1, x_2, \cdots, x_n)$ is called a threshold function if there exist integer weights $w_1, w_2, \cdots, w_n$ and a fixed threshold T such that

$$f(x_1, x_2, \ldots, x_n) = \begin{cases} 1, & \text{if} \sum_{i=1}^{n} w_i x_i \geq T \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

A threshold function $f(x_1, x_2, \ldots, x_n)$ can be represented by $[w_1, w_2, \ldots, w_n; T]$. The following are two examples of threshold functions.

$$\begin{aligned} f(a,b,c) &= a \vee bc \\ &= 2a + b + c \geq 2 \\ &= [2,1,1;2] \quad (2) \\ g(a,b,c,d,e) &= ac(b \vee d \vee e) \vee de(a \vee bc) \vee ab(d \vee e) \\ &= 2a + b + c + d + e \geq 4 \\ &= [2,1,1,1,1;4]. \quad (3) \end{aligned}$$

$h(a, b, c, d) = ab \vee cd$ is not a threshold function.

This work was supported in part by the Science Foundation Arizona (SFAz) grant SRG 0211-07 and by the Stardust Foundation.

By a threshold *gate* we mean a non-decomposable primitive circuit that realizes a threshold function by comparing some physical quantity such as charge, voltage or current. This definition excludes implementations that realize threshold functions as simply logic functions using conventional logic primitives such as AND and OR.

As is evident from the example of the function $g(a, b, c, d, e)$ above, a single threshold gate can implement a complex logic function, which in conventional logic design might require a multilevel network of logic gates. Given that a single threshold gate can implement an otherwise complex logic function, with the availability of an efficient design of a threshold gate, a network of threshold gates can potentially lead to significant reduction in area, dynamic and leakage power, and increase in performance when compared to an equivalent logic network consisting of conventional logic primitives. This has been the main motivation for the study and development of threshold logic.

The advent of nanotechnology has renewed interest in threshold logic because nano-devices such as resonant tunnelling diodes (RTDs) and quantum cellular automata (QCAs) are inherently threshold gates [8] [5]. Apart from these, another important nano-device which is best suited for threshold logic is Single Electron Transistor (SET). [9] explores a novel design of an *n*-input linear threshold gate, requiring one tunnel junction and $n + 2$ capacitors. [10] describes an improved design of the SET based threshold logic gate and the design was demonstrated by implementing a full adder.

The survey in [4] details nearly 50 different implementations of threshold logic gates. Among the various types, those based on differential logic such as DCSTL and SCSDL have been the most promising because they employ the conventional CMOS devices, and require no special processing and have been shown to be relatively fast.

In this paper we introduce a completely new architecture for a threshold gate. It is based on the integration of conventional MOSFETs and a Spintronic device, known in the literature as *Spin Transfer Torque - Magnetic Tunnelling Junction* (STT-MTJ) device. The novel feature of this architecture is that the STT-MTJ device is intrinsically a primitive threshold device, i.e., it changes state when the magnitude of the current through the device exceeds some threshold. This simple property, when

exploited properly leads to a extraordinarily simple realization of a complex threshold gate, referred to here as an STL cell. Next, we present novel architecture consisting of a 2-D array of STL cells, each of which can be programmed to implement threshold functions. The array is referred to as a *Spintronic Threshold Logic Array* (STLA).

An STLA has several important features that are not present in any conventional logic array: (1) it is a *non-volatile* logic circuit, which means that it will retain the partial results of a computation even when the power is disconnected; (2) it resembles the structure and operation of a DRAM, and is very compact; (3) the inputs of every STL cell are fully controllable, and its outputs are fully observable, without having to include any scan capability; (4) it has zero standby power consumption.

Recent works such as [14], [6], [12] discuss the usage of STT-MTJ in logic computation. [14] describes the design of a non-volatile flip flop which can be used in FPGA and system on chip circuits. [6] explores the design of a magnetic full adder for low power applications. Logic operation performance with STT-MTJ was compared to their traditional methods of computation with separate logic and memory units in [12]. All prior work on the use of STT-MTJ for logic use it for storage (logic 0 or 1) or as resistive networks to realize single logic gates. In contrast, the method described here employs a single STT-MTJ device in conjunction with MOSFETs to realize a complex threshold function.

II. Preliminaries of STT-MTJ

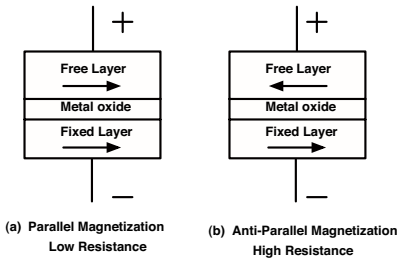

Fig. 1. Structure of STT device

Figure 1 illustrates the structure of an STT-MTJ device, which consists of three layers: (1) a *fixed* orientation magnetic layer, (2) a thin dielectric layer, and (3) a *free* orientation magnetic layer. When the electron spins in either the free or fixed layer are all aligned in one direction, the layer is magnetized. The fixed layer is magnetized in one direction and the direction of the free layer can be controlled to be either in parallel (low resistance R_{low}) or anti-parallel (high resistance R_{high}) with the fixed layer. The two distinguishable resistance values serve to represent binary states. The binary states are represented by magnetic state (no leakage and non-volatile) as opposed to charge, and the state is sensed by measuring the resistance. Of the several different ways that the magnetic state can be switched, the use of spin-torque transfer has the greatest advantages: scalability, low current consumption and very high density [11]. The switching from R_{high} to R_{low} takes place if a large enough ($> I_c$ for a fixed area) positive current flows through the device. A negative current will result in switching in the opposite direction. I_c depends on the area of the STT-MTJ device as well as the duration of the current flow. It is in this sense that an STT-MTJ device is viewed as a primitive threshold device. Moreover, integration of STT-MTJ with CMOS process is a very easy task. STT-MTJ like a via occupies no plane silicon area, as it as it fabricated above the CMOS circuits [15].

A. STT-MTJ parameters

The STT-MTJ device models used in this paper were developed by Zhao *et.al* [13] and the device parameters are explained in Table I.

TABLE I
Characteristics of STT-MTJ

Variable	Description	Default Value
T_{ox}	Tunnel Oxide Thickness	0.85nm
Area	STT-MTJ surface	$65nm \times 65nm \times \pi/4$
R_{low}	Parallel Resistance Value	$3K\Omega$
R_{high}	Anti-parallel Resistance Value	$6.6K\Omega$
TMR(0)	TMR ratio with zero bias	120%

- R_{low} is the value of the resistance when the magnetization of the free and fixed layers are in the same/parallel direction.
- Similarly, R_{high} is the value of the resistance when the magnetization of the free and fixed layers are in the opposite/anti-parallel direction.
- T_{ox} is the thickness of the dielectric material that separates the two ferromagnetic materials in the STT-MTJ device.
- The STT-MTJ device used here is a circular one, with length and width being the same i.e., 65nm. An elliptical STT-MTJ structure can be obtained if the length and the width are different. R_{high}, T_{ox} and Area are related as $R_{high} \propto \frac{exp(T_{ox})}{Area}$, which clearly shows that an elliptical structure has more R_{high} than a circular one [4].
- Tunneling Magneto Resistance (TMR) shown in Table I is defined as $(R_{high} - R_{low})/R_{low}$. This is a measure of how well the two resistance states can be distinguished. Higher the TMR, easier the distinguishability. Presently, the TMR is in the order of 50-200% [3]. TMR is also a strong function of the bias voltage applied across the STT-MTJ device, due to the hysteresis behaviour seen in the R-V characteristics of the device.

III. Architecture and Design of the Spintronic Threshold Logic (STL) -Cell

This section describes the architecture of the threshold logic cell using the STT-MTJ device. The STL threshold logic cell is shown in Fig. 2.

The cell operates in the following manner. The signals WR, RD_i, and PR are pairwise complementary i.e., no two of

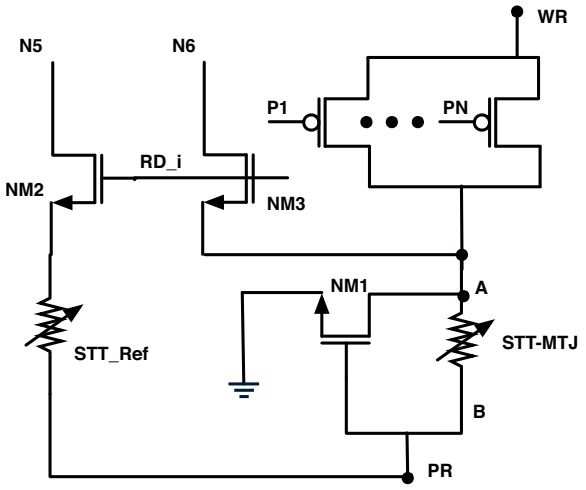

Fig. 2. STL-Cell

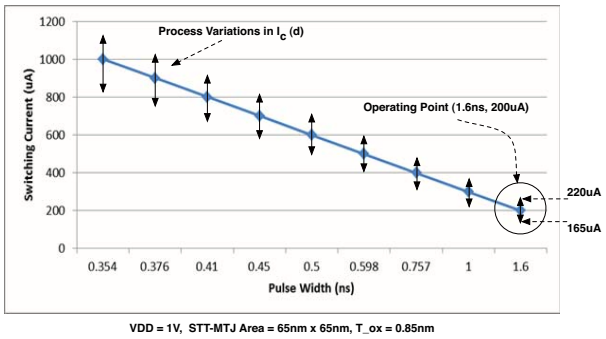

Fig. 3. Switching current vs pulse width of STT device

them are high at the same time. Initially, PR is asserted and the current flows into the NMOS transistor (NM1) through the STT-MTJ device from B to A. The flow of current brings the STT-MTJ in the anti-parallel or high resistance state. When the WR is asserted, certain amount of current (I), flows through the STT-MTJ device, depending on the number of ON PMOS transistors (P1 to PN shown in Fig 2). If I $\geq (I_c)$ switching current, then the STT-MTJ device switches to the low resistance state otherwise it remains in high resistance state. This is the *write* phase of the cell. When the WR pulse goes low, no current flows through the STT-MTJ device, and since the device is non-volatile, the state is maintained.

Reading the state of STT-MTJ is done by asserting the RD_i pulse. When RD_i goes high, transistor NM3, (whose source is connected to the STT-MTJ device) and transistor NM2 (connected to the STT-Ref) are enabled. The drains of these two transistors N5 and N6 are the two outputs of the STL cell and they are connected to the sense amplifier to evaluate the state of the STT-MTJ device. The design of sense amplifier is detailed in the section IV-A and the simulation results of the STL cell are shown in VI-A.

A. Design of STL cell

Fig. 3 shows a plot of the critical switching current I_c versus the pulse width d. In reality, I_c is exponentially related to d, for small values of d. However, this particular device was simulated for a limited range of d over which it is linear. The plot also emphasizes the fact that I_c is a distribution at each d due to variations in the STT-MTJ. Note that the smaller the pulse width d, the greater the variation.

Let $I_c^{min}(d)$ and $I_c^{max}(d)$ denote the values of I_c such that the probability of STT-MTJ switching from R_{high} to R_{low} with a pulse width of d is less than ϵ when $I_c \leq I_c^{min}(d)$, and is greater than $1-\epsilon$ when $I_c \geq I_c^{max}(d)$ for a suitably chosen small $\epsilon > 0$. In the present design, we have chosen d=1.6ns because the variation in I_c is expected to be the least at that point and then estimated empirically the values of $I_c^{max}(1.6ns) = 220uA$ and $I_c^{min}(1.6ns) = 165uA$. Let w denote the width of input transistors. Let N_{min} denote the minimum number of transistors of width w whose ON current exceeds $I_c^{max}(d)$. Similarly, let n denote the maximum number of transistors whose ON current is less than $I_c^{min}(d)$. For a given w, we search for suitable values of N_{min} and n by simulation. As expected, these parameters (I_c^{max}, I_c^{min}, d, w, N_{min}, n, ϵ) all affect the power, delay and robustness of the circuit. However, unique feature of the STL cell is that, these parameters also affect the feasibility of the threshold functions.

Given n and N_{min}, a threshold function $f = [w_1, w_2, \ldots, w_n; T]$ can be realised if in the function $\lceil N_{min}/T \rceil \times f$, the minimum total weight among the minterms in the onset is greater than or equal to N_{min}, and the maximum total weight among the minterms in the offset is less than or equal to n, and $\lceil N_{min}/T \rceil \times \sum_{i=1}^{n} w_i x_i \leq N$, where N is the total number of PMOS transistors.

For different values of n, N_{min} and N, all the realizable functions are easily enumerated. Table II shows the number of realizable functions. The shaded row shows the configuration used in our design.

TABLE II
NUMBER OF FUNCTIONS ENUMERATED

N	N_{min}	n	#Functions
20	9	6	15
25	9	6	28
25	12	9	34
30	9	6	50
40	9	6	99
40	12	9	158
50	12	8	83
50	9	5	173

As an example, consider the threshold function $f = ab + bc + ca = [1, 1, 1; 2]$. Suppose for some w, the search resulted in $N_{min} = 9$ and $n = 6$. Then $f' = \lceil 9/2 \rceil \times f = [5, 5, 5; 10]$. f' and f are the same function. In f', if any two variables are logic high, then the minimum total weight of an onset minterm is $10 > 9$, and the maximum total weight of the offset minterm (only one variable is high) is $5 < 6$. So f' is feasible with

these values of N_{min} and n and the total number of transistors required is $N = 15$.

Now, consider $f = [4, 2, 2, 1, 1; 6]$ and suppose $N_{min} = 12$, $n = 9$. Then $f' = \lceil 12/6 \rceil \times f = [8, 4, 4, 2, 2; 12]$. The minimum total weight of the onset minterm is $12 \geq N_{min}$ but the maximum total weight of the offset minterm is $10 \not\leq 9$. Hence f cannot be realized with the restriction $N_{min} = 12$ and $n = 9$ and the total number of transistors required to realize f is $N = 20$.

Several STL cells can be cascaded to form a gate array to realize larger functions. This architecture is detailed in the next section and a case study for realizing a 16-bit Carry Look Ahead Adder has been presented.

IV. STLA Gate Array Architecture

Fig. 4 shows a $2X2$ Spintronic Threshold Logic Array (STLA) architecture.

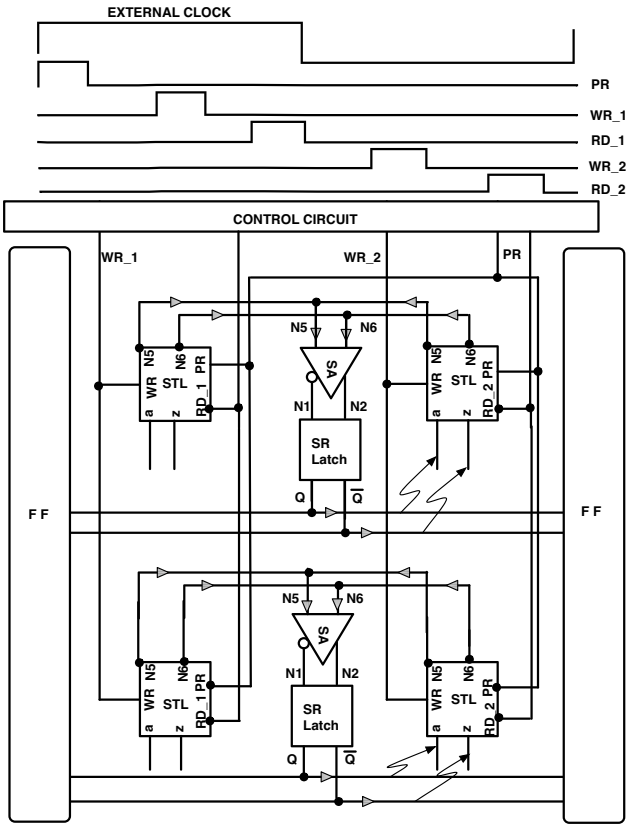

Fig. 4. STL Array

The STLA architecture, consists of 2-D array STL cells. Similar to DRAM architecture, each row shares a sense amplifier (SA) x and a SR latch (shown in Fig.4). The computation in the array is done column wise. Signals RD_i, WR_i, PR are generated by the control circuitry (details not shown here) for each column i. Initially all the STT-MTJs in all the cells are preset by asserting the PR signal. WR_1 signal is then asserted and each STL cell in the first column computes the function mapped on to it based on inputs a to z shown in the figure. It should be noted that these inputs are connected to the gates of the PMOS transistors of the STL cell as described in section III. The results of the computation are stored in their respective STT-MTJ devices. RD_1 (read) signal is then asserted and the value stored in the STT-MTJ is read by the sense amplifier and stored in the SR latch. One (WR_i, RD_i) signal pair is needed for each column to compute and read the STL cell. The output of the SR latch fans out to the STL cells in the next column, depending on the functions mapped on to the complete array. Repeaters can be provided to avoid fan out issues. WR_2 signal is then asserted and the STL cells in the second column compute and store their results into their respective STT-MTJ devices. RD_2 is then asserted to read the state of STT-MTJs. The previous value in the SR latch is lost, and the new values from the second column are now stored in the SR latch. (**NOTE: The values in the previous column are still available in STT-MTJ due to *non-volatility*.**) These act as inputs to the STL cells in the third column. This process repeats till the last column. External flipflops can be connected to the primary inputs and outputs of the array or *tile*). Fig. 4 shows the connections and the waveforms for each signal in the array. A case study showing the mapping of a 16-bit Carry Look Ahead Adder is shown in the section V.

A. Design of the Sense Amplifier

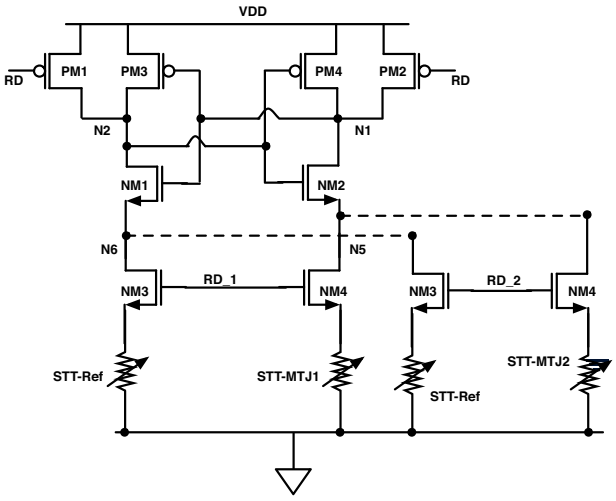

Fig. 5. Sense Amplifier

The sense amplifier used to read the state of STT-MTJ is shown in Fig. 5. It consists of two cross coupled inverters (PM3, NM1 and PM4, NM2) and two PMOS transistors (PM1 and PM2) to reset the nodes N1 and N2. Transistors (NM3, NM4) and STT-MTJ devices (STT-Ref and STT-MTJ1) are part of the STLA cell (shown as NM2, NM3 and STT-Ref, STT-MTJ in Fig. 2). These transistors and STT devices (NM3, NM4 and STT-Ref, STT-MTJs) of other STLA cells in the same row, are connected to the nodes N5 and N6 of the sense amplifier (shown as dotted lines in Fig. 5).

The operation of the sense amplifier is as follows. The RD signal driving the gates of PM1 and PM2, is high when any of the RD_i signals are high, and low when all of them are low. Whenever, RD is low (called the "reset" phase), PM1 and PM2 are ON and the nodes N1 and N2 are pre-charged to logic high. During this phase, N5 and N6 are both at *VDD-*V_{th}.

During the evaluation phase, ONLY ONE RD_i signal will be asserted (therefore *RD* signal is asserted), depending on which STL cell has to be read. Suppose RD_1 is asserted and STT-MTJ1 is in low resistance state, then NM3 and NM4 are ON and the STT devices (STT-Ref and STT-MTJ1) are connected to nodes N5 and N6 of the sense amplifier. Since RD signal is high, the transistors PM1 and PM2 are OFF. The resistance of "STT-Ref" device is $(R_{high}+R_{low})/2$. This value can be achieved either by changing the surface area of the STT-device or by connecting two series STT devices, each of whose resistances are R_{high} and R_{low}, in parallel. Here, the dimensions of the device have been changed to obtain a resistance value $(R_{high}+R_{low})/2$. Since STT-MTJ1 is in low resistance state, ($<$ resistance of STT-Ref), more current is drawn from node N1 than from node N2. Thus N1 discharges faster than N2 and when it becomes less than the switching threshold of the inverter PM3 and NM1, PM3 switches ON pulling N2 to logic high and N1 eventually goes down to logic low. Nodes N1 and N2 act as inputs to the SR latch.

B. S-R Latch

A symmetric SR latch has been designed to store the values evaluated by the sense amplifier and also to drive the inputs of the STL cells in the next column. Details of the SR latch can be found in [7].

V. Mapping of a 16-bit Carry Look Ahead Adder

First we describe the threshold gate representation of a two-bit carry look ahead adder, which is the basic building block of 16-bit CLA. (NOTE: Mapping was done manually, as the primary focus of the paper is the STLA architecture).

A. Two-bit-carry Look Ahead Adder

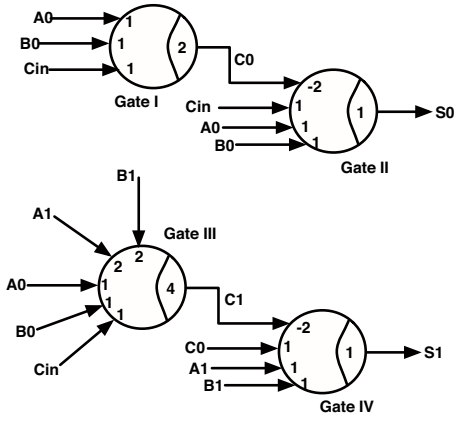

Fig. 6. Threshold representation of 2-bit Carry Look Ahead Adder

The structure of a two bit carry look-ahead (CLA) adder implemented using threshold logic is shown in Fig. 6. *Gate I* computes carry, whose inputs are $A0$, $B0$ and C_{in}. The well known representation of carry in a one-bit full adder is $carry = ab + bc + ca$, which is [1, 1, 1; 2]. *Gate II* computes sum, which is $abc + (a + b + c)\overline{carry}$. The weight vector for this function (sum) is [-2, 1, 1, 1; 1]. Threshold gates I and II form a one-bit full adder.

Gate III computes the $carry$ of the second stage (carry look ahead computation) and has $A0$, $B0$, $A1$, $B1$ and C_{in} as its inputs. Its threshold representation is [2,2,1,1,1; 4]. Carry generated in the second stage ($C1$), the previous stage carry ($C0$), $A1$ and $B1$ act as inputs to the *Gate IV*, which is second stage sum function. $S0$ and $S1$ are the sum outputs of a two-bit carry look ahead adder and $C1$ is the carry out.

B. Mapping of 2-bit Carry Look Ahead Adder

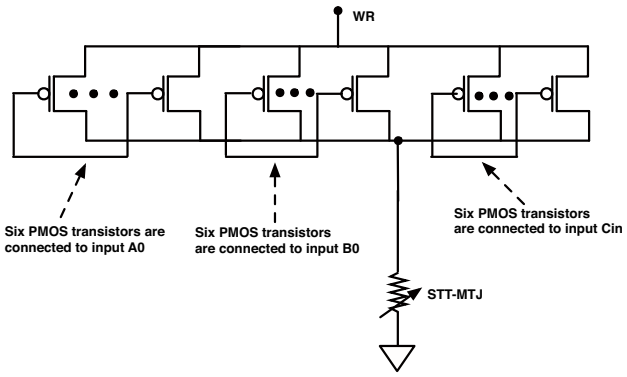

Fig. 7. Physical Implementation of $carry$ in STL cell

Each threshold function in Fig. 6 is mapped onto one STL cell. The first stage $carry = [1, 1, 1; 2]$ can be represented as [6, 6 ,6; 12]. Each of the three inputs of this function is connected to six PMOS transistors in the STL cell as shown in Fig.7. Hence, if any of the two inputs are ON or all inputs are ON, the STT switches to a low resistance state. All the other gates in Fig. 6 are also implemented in the same fashion by suitably modifying their weight vector as explained in section III-A.

If the gates can compute parallely, they are mapped into the same column of the STLA array. For e.g., *Gates I and III* can be mapped to the same column.

C. 16-bit-carry Look Ahead Adder

The mapping of the entire 16-bit carry look ahead adder is shown in Table III, where a cell in the table corresponds to an STL cell in the array.

Each shaded area in the table, corresponds to a 4-bit CLA consisting of two 2-bit adders. This pattern of blocks is repeated four times to construct a 16-bit CLA. Each carry and sum gates has been numbered and the correspondence is as follows. Gates $1, 5, 9, 13, 17, 21, 25, 29$ implement [1, 1, 1;

TABLE III
MAPPING OF 16-BIT CARRY LOOK AHEAD ADDER

Latch O/P	WR_8 RD_8	WR_7 RD_7	WR_6 RD_6	WR_5 RD_5	WR_4 RD_4	WR_3 RD_3	WR_2 RD_2	WR_1 RD_1	WR_0 RD_0
$Q_0 \rightarrow$	B	26	25	18	17	10	9	2	1
$Q_1 \rightarrow$	B	B	27	B	19	B	11	B	3
$Q_2 \rightarrow$	32	28	24	20	16	12	8	4	B
$Q_3 \rightarrow$	30	29	22	21	14	13	6	5	B
$Q_4 \rightarrow$	B	31	B	23	B	15	B	7	B

2] function (first stage carry). Gates $3, 7, 11, 15, 19, 23, 27, 31$ implement the second stage carry as [2, 2, 1, 1, 1 ; 4]. Remaining gates $2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32$ implement the sum as [-2, 1, 1, 1; 1]. The negative weights in the sum have been taken care through proper use of inverters. Note that no extra inverters are needed since latch produces complemented outputs. Parallelism among the blocks has been exploited to reduce the number of rows and columns. Each of the entries in Table III correspond to gates shown in Fig. 8, which also shows the interconnections among the gates.

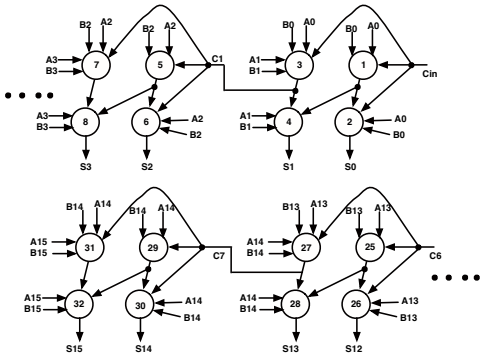

Fig. 8. Mapping of 16-bit Carry Look Ahead Adder

The latch outputs are named Q_0 , Q_1, Q_2, Q_3 and Q_4 (one SR latch per row as shown in Table III). It should be noted that the output sum bits of the 16-bit CLA are computed by different STL cells and they are available in their respective SR latches at different time instances. Table IV shows the time instance (in terms of read pulse number) and location (in terms of SR latch output) for each of the sum bits. Whenever an output bit is available, it can be stored in external flipflops. The final carry out S_{16} is available on Q_4 after RD_7 signal.

TABLE IV
SUM OUTPUT OF 16-BIT CARRY LOOK AHEAD ADDER

SR latch Output	Q_2	Q_3	Q_2	Q_0
Read Pulse	RD_8	RD_8	RD_7	RD_7
Sum Bit	S_{15}	S_{14}	S_{13}	S_{12}
STL cell#	32	30	28	26
Read Pulse	RD_6	RD_6	RD_5	RD_5
Sum Bit	S_{11}	S_{10}	S_9	S_8
STL cell#	24	22	20	18
Read Pulse	RD_4	RD_4	RD_3	RD_3
Sum Bit	S_7	S_6	S_5	S_4
STL cell#	16	14	12	10
Read Pulse	RD_2	RD_2	RD_1	RD_1
Sum Bit	S_3	S_2	S_1	S_0
STL cell#	8	6	4	2

An entry labelled B, in Table III corresponds to a buffer or a dead-cell. The function of the buffer is to store the value generated in the previous column, for its next column. For example, $carry$ generated by entry 3, in column 0 is stored in the buffer B (in column 1) and can be used by any of the gates in column 2. Therefore, feed forward is possible if an STL cell acts as a buffer, without changing the contents of the SR latch. An STL can be a dead cell instead of a buffer. The write and read signals feeding the STL can be disconnected, so the cell computes nothing. This behaviour does not alter the state of the SR latch. Therefore the dead cell logically acts as a buffer. In our implementation, STL cells are configured as dead cells instead of buffer.

VI. EXPERIMENTAL RESULTS

A. Functional Schematic Simulation

The simulation results of the STL cell (Fig. 7) implementing a $carry = [1, 1, 1; 2]$ function is shown in Fig. 9. Inputs given were A=0, B=0 and C=1 (NOTE: Signals A and B turn ON the PMOS devices) and the switching of STT-MTJ within the 1.6ns pulse width can be clearly seen for these particular inputs. The STT-MTJ device does not switch, if ONLY ONE of the inputs is 0.

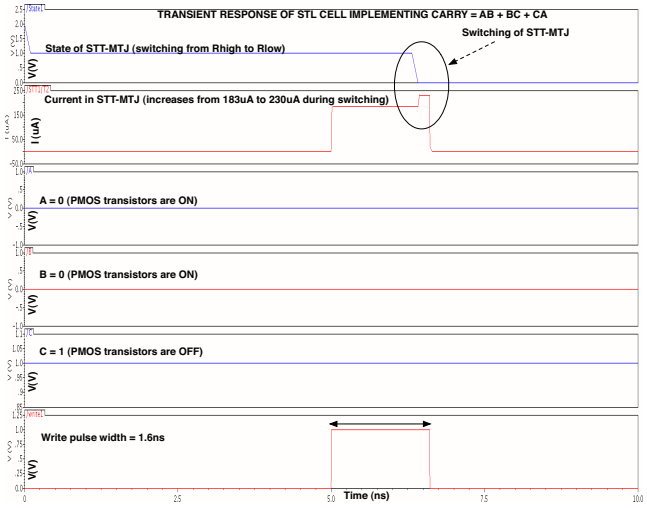

Fig. 9. Response of STL cell showing the implementation of $carry = [1, 1, 1; 2]$. (Magnitude of logic high = 1V)

The 16-bit Carry Look Ahead Adder was simulated using Cadence Spectre environment and the models used for MOSFETs were from commercial (65nm) design kit. The STT-MTJ models were from [13]. Fig. 10 shows the simulation results of 4-bit CLA. It shows six signals, five SR latch outputs and the RD signal (recall that RD signal is ON, when any of the RD_i is ON).

The intermediate spaces between the pulses of the RD signal are the WR_i signals (not shown). Note that, the WR_i and RD_i are spaced apart for the proper view of the functional simulation results. The inputs used in the simulation of the 4-bit CLA are $A_3A_2A_1A_0$ = 0011, $B_3B_2B_1B_0$ = 0011 and $C_{in} = 1$. Therefore the sum and $carry$ outputs are

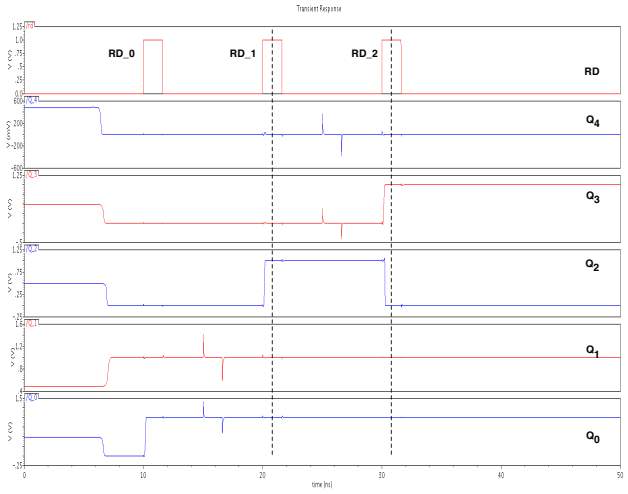

Fig. 10. Outputs of SR latches showing the computation for three read cycles in a 4-bit CLA

expected to be 0111 and 0 respectively. Initial pulse RD_0 is used to compute the *carry* bits of first and second stages. After RD_1 signal is asserted, the sum bits $S0$ and $S1$ are available at Q_0 and Q_2 respectively (Note that Q_0 and Q_2 go high after RD_1 signal in Fig. 10). Similarly, after RD_2 is asserted, sum bits $S2$, $S3$ and $S4$ (final carry out) are available at Q_3 (high), Q_2 (low) and Q_4 (low) respectively.

B. Comparison with FPGA

Two 16-bit adders [1] Carry look Ahead Adder (CLA) and [2] Ripple Carry Adder (RCA) have been implemented on Xilinx Virtex-5 FPGA and comparison with the STLA is discussed. Three metrics area, power, delay have been compared.

1) $12X$ Transistor Reduction: The number of LUTs (and the transistor count) used for implementing CLA (in FPGA) are shown in Table V.

TABLE V
NO. OF LUTS IN VIRTEX-5 USED FOR 16 BIT CARRY LOOK AHEAD ADDER

#LUTs	**#SRAM cells**	**#NMOS switches (excluding SRAM cells)**	**#Transistors (including 6T SRAM cell)**
$4 \times LUT2$	$4 \times 4 = 16$	$6 \times 4 = 24$	120
$2 \times LUT4$	$16 \times 2 = 32$	$30 \times 2 = 60$	252
$14 \times LUT5$	$32 \times 14 = 448$	$62 \times 14 = 868$	3556
$26 \times LUT6$	$64 \times 26 = 1664$	$126 \times 26 = 3276$	13260
		Total→ 4228	Total→ 17188

An LUT typically consists of SRAM cells and NMOS transistors/switches for lookup operation. Total number of SRAM cells is 2^n and the total number of NMOS switches is $2^1 + 2^2 + ...2^n$, where n is the number of inputs [15]. This gives a simple estimate of transistor count as *(# SRAM cells $\times$ 6) + (# NMOS switches)*. Additional inverters needed were ignored. This count itself grows exponentially as the number of inputs grow.

The transistor count for STLA architecture is $(45 \times 30) + (5 \times 6) + (5 \times 16) = 1460$, where each number is defined in Table VI.

TABLE VI
TRANSISTOR DETAILS OF STLA TILE

Number of STL cells in the entire STLA array	45
Transistors per STL cell	30
Number of sense amplifiers	5
Number of SR latches	5
Transistors in sense amplifier	6
Transistors in SR latch	16

It can be clearly seen that there is $12X$ *reduction in the transistor count* in STLA, compared to CLA-FPGA (shown in Table V $= 1460/17188$). This reduction directly translates to post layout silicon area. If the SRAM transistor count is not considered, this improvement reduces to $3X$. Similar results were observed when STLA was compared with RCA-FPGA. The reduction in transistor count was $10X$ $(= 1460/14918)$ and $2.5X$ without the SRAM transistor count $(= 1460/3662)$.

2) Zero Standby Power: Data volatility leads to significant standby power dissipation. This problem is more dominant in SRAM based FPGAs.

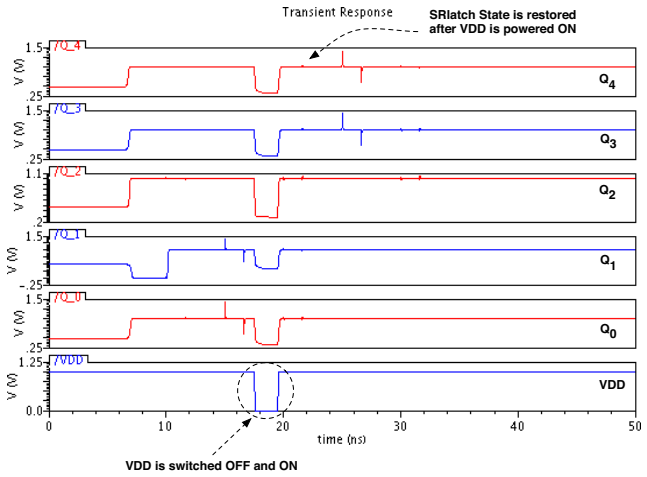

Fig. 11. Waveforms showing the restoration of latch outputs after the supply was turned OFF and then ON

A prominent feature is the non-volatile nature of STT-MTJ. Due to this feature, when power supply is switched OFF, the state of the SR latch can be restored without any significant delay. This leads to near zero standby power. Therefore in standby mode circuit supply can be turned OFF. Fig. 11 shows the waveforms of simulation demonstrating this fact. The first five waveforms correspond to the latch states during active mode. The last waveform shows supply turned OFF and then ON for certain duration. It is clearly seen that the state of the latch is restored without any significant delay.

3) Dynamic Power, Delay and Energy: The dynamic power, delay and energy, for FPGA and STLA are shown in Table VII.

TABLE VII
DYNAMIC POWER, DELAY AND ENERGY OF FPGA AND STLA

Metric	CLA-FPGA	STLA	Pipelined STLA
Power (mW)	3	0.6	0.64
Delay (ns)	3.1	20	10
Energy (pJ)	9.3	12	12.8
Average power (mW) (20ns clock period)	0.45	0.6	0.64

The power values shown in the Table VII for CLA based FPGA, is the average logic power (neglecting the signal power), with 30% toggle activity of the inputs. The average power (20ns clock period) of RCA-FPGA was $0.4mW$ for the same toggle activity. The power of STLA was calculated as the average of powers for best case and worst case vectors. Significant portion of power (and delay) of STL Array is mainly due to large switching current of STT-MTJ. A large part of the research in STT-MTJ is aimed at reducing the switching current. Grandis corporation, a pioneer in STT-RAM technology, claims that this device has an excellent scalability with technology. Switching current can scale down significantly below 100uA for 45nm and beyond, which leads to denser, less expensive and low power STT-MTJ architectures [1].

The delay of STLA is about 20ns. The delay comes from STT-MTJ, which needs 1.6ns write pulse width. The read time is typically 400ps. So, 2ns is needed for one write and read operation per stage/column in STLA. There are nine stages (in STLA for 16 bit CLA) and hence 20ns is the maximum time after which the final carry out is obtained. The nine stage STLA can be implemented as two small five stage cascaded STLAs in order to reduce the delay by half without considerable increase in power.

C. Complete Scan Capability without Scan

Every STLA cell in the STLA is completely scannable. The inputs and outputs of each STL cell are tied to the latches. Therefore the inputs can be externally controlled and its outputs are fully observable, akin to memory. No additional scan circuitry is needed to test the STL Array.

D. Robustness

Monte Carlo simulations have been done on one STLA cell (STL cell + Sense amplifier + SR latch) using **Synopsys HSPICE** [2]. It was observed that in 1000 simulations, there were no failures for about 8% variation (which is approximately 3 sigma) in the STT-MTJ dimensions. It should be noted that the variations in transistors are not restricted (6 sigma or more).

VII. CONCLUSION AND FUTURE WORK

In this work we proposed a novel gate array architecture consisting of spintronic based threshold logic cells. This architecture exhibits attractive features such as compactness, non-volatility, zero standby power. These have been demonstrated by mapping a 16-bit Carry Look Ahead adder on to the STLA. These results were compared with an FPGA implementation of 16-bit CLA. STLA is shown to have $12X$ reduction in the transistor count and zero standby power (as confirmed by the simulation results).

The optimal design of this STL cell requires statistical analysis by considering process variations both in the MOS transistors (V_{th}, L and W) and STT-MTJ device (oxide thickness t_{ox}, area A). I_c^{min} and I_c^{max} have only been estimated in this design, however the accuracy of these estimates (in contrast with foundry data) is to be investigated. Also, the design of the control circuitry used in the STLA should be studied for proper pulse generation.

ACKNOWLEDGEMENTS

We would like to thank Dr. Weisheng Zhao, CNRS Research Scientist, France, for providing us with the 65nm perpendicular STT-MTJ device models and helping us with his valuable suggestions about the device.

REFERENCES

[1] The future of scalable stt-ram as a universal embedded memory.
[2] Synopsys HSPICE- Accurate Circuit Simulation.
[3] C. Augustine, N. Mojumder, X. Fong, S. Choday, S. Park, and K. Roy. Spin-transfer torque mrams for low power memories: Perspective and prospective. *Sensors Journal, IEEE*, 12(4):756 –766, april 2012.
[4] V. Beiu, J. M. Quintana, and M. J. Avedillo. VLSI Implementations of Threshold Logic - A Comprehensive Survey. *IEEE Trans. Neural Networks*, 14:1217–1243, 2003.
[5] E. Blair and C. Lent. Quantum-Dot Cellular Automata: an Architecture for Molecular Computing. In *Simulation of Semiconductor Processes and Devices, 2003. SISPAD 2003. International Conference on*, pages 14 – 18, Sep 2003.
[6] Y. Gang, W. Zhao, J.-O. Klein, C. Chappert, and P. Mazoyer. A high-reliability, low-power magnetic full adder. *Magnetics, IEEE Transactions on*, 47(11):4611 –4616, nov. 2011.
[7] H. Gao, P. Baltus, and Q. Meng. Low voltage comparator for high speed adc. In *Signals Systems and Electronics (ISSSE), 2010 International Symposium on*, volume 1, pages 1–4, sept. 2010.
[8] P. S. Jian, G. Haddad, P. Mazumder, and J. Schulman. Resonant Tunneling Diodes: Models and Properties. *Proceedings of the IEEE*, 86(4):641 –660, Apr. 1998.
[9] C. R. Lageweg, S. D. Cotofana, and S. Vassiliadis. A linear threshold gate implementation in single electron technology. In *Proceedings. IEEE Computer Society Workshop on VLSI 2001: Emerging Technologies for VLSI Systems*, pages 93–98, April 2001.
[10] C. R. Lageweg, S. D. Cotofana, and S. Vassiliadis. A full adder implementation using set based linear threshold gates. In *Proceedings 9th IEEE International conference on electronics, circuits and systems - ICECS 2002*, pages 665–669, September 2002.
[11] J. Li, P. Ndai, A. Goel, S. Salahuddin, and K. Roy. Design paradigm for robust spin-torque transfer magnetic ram (stt mram) from circuit/architecture perspective. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 18(12):1710 –1723, dec. 2010.
[12] S. Patil, A. Lyle, J. Harms, D. Lilja, and J.-P. Wang. Spintronic logic gates for spintronic data using magnetic tunnel junctions. In *Computer Design (ICCD), 2010 IEEE International Conference on*, pages 125 – 131, oct. 2010.
[13] Y. Zhang, W. Zhao, Y. Lakys, J. Klein, J.-V. Kim, D. Ravelosona, and C. Chappert. Compact modeling of perpendicular-anisotropy cofeb/mgo magnetic tunnel junctions. *Electron Devices, IEEE Transactions on*, 59(3):819 –826, march 2012.
[14] W. Zhao, E. Belhaire, and C. Chappert. Spin-mtj based non-volatile flip-flop. In *Nanotechnology, 2007. IEEE-NANO 2007. 7th IEEE Conference on*, pages 399 –402, aug. 2007.
[15] W. Zhao, E. Belhaire, C. Chappert, and J. O. Klein. *Hybrid Spintronic/CMOS circuit design and analysis* . LAP LAMBERT Academic Publishing, 2010.