A Scalable and High-Density FPGA Architecture with Multi-Level Phase Change Memory

Chunan Wei, Ashutosh Dhar, and Deming Chen

Dept. of Electrical and Computer Engineering, University of Illinois, Urbana-Champaign
albertcnw@gmail.com, {adhar2, dchen}@illinois.edu

***Abstract*—As CMOS technology is stretched to its limits it has become imperative to look to alternative solutions for the next generation of FPGAs. In particular, due to the configurable nature of FPGAs, on-chip memory remains to be a major concern for designers. In this work we explore the use of Phase-Change Memory (PCM). We exploit the ability of PCM to exist in multiple intermediate states to store 2 bits per cell and develop a new Look Up Table (LUT) architecture. The new LUT can either store two functions with shared inputs or a single function with an additional input. We also explore the use of PCM in local routing mechanisms and thus propose a new Configurable Logic Block (CLB) composed of CMOS and PCM. The new design promises significant improvements in logic density and performance with area improvements of over 40% for all LUT sizes and delay improvements of 7% to 13% on an average for LUTs of size 10 to 6 .**

I. Introduction

Field Programmable Gate Arrays have proven themselves to be a promising candidates for parallel, energy-efficient and reliable computing. With the rapid scaling of silicon technology, FPGAs have seen dramatic improvements in area-delay characteristics, as well as logic density. However, due to the limitation of CMOS technology, further improvement of SRAM based FPGA technology has become rather difficult. When coupled with the fact that in traditional SRAM based FPGAs, memory occupies over 40% of the area [1], which in turn impacts routing delay. Finding alternatives to SRAM based FPGAs has generated a great deal of interest. Recent work such as, 3D-NEMS FPGA [2], R-RAM FPGA [15] and PCM based FPGA [3] suggest that the leading trend for FPGA development is to introduce emerging Non-Volatile Memories (NVM) into the FPGA architecture. This paper takes a step towards a new generation of FPGAs based on Non-Volatile Memories (NVM).

Emerging NVMs have gained a significant amount of interest in recent years, with several types of NVMs being proposed and developed. Among them, Phase Change Memory (PCM) is considered to be one of the most promising candidates for future storage elements. This is because PCM has excellent scalability [4] with the potential of a single cell being scaled to a few nanometers in size. Most importantly, PCM is compatible with existing CMOS processes. It also has a moderate amount of read/write cycles and has a significantly better write cycles when compared to NAND Flash based alternatives [5]. Of particular interest, is the use of NVMs for building FPGAs from the reliability perspective: resistance to soft-errors due to Single Event Upsets (SEU). Due to their operating principle, resistive memory based NVMs such as PCM and R-RAMs are resilient to soft-errors caused by radiation when compared to SRAM based cells. This makes them very attractive for system critical applications, albeit at the cost of latency. In addition, the non-volatile nature of the cells eliminates the need to reconfigure the FPGA after power cycles, saving time as well as eliminating the need to add external flash memories or micro-controllers to save or reprogram the FPGA configuration.

With this in mind, we present this paper as an exploration into the novel use of PCM for building FPGAs. Incorporating PCM into FPGA designs has been attempted previously in a rather simplistic fashion; by replacing SRAM cells with 1T2R PCM cells [3]. In contrast, we explore the use of PCM to exist in multiple states and thus store multiple bits per cell. By utilizing a similar 1T2R structure we demonstrate the ability of the PCM cell to store 2 bits. The ability to store multiple bits per cell allows for increased logic density as well as presents very significant area savings. In this work we develop a new LUT architecture that is capable of exploiting this property and demonstrate how the new LUT saves area and provides potential delay improvements as well when compared to SRAM based approaches.

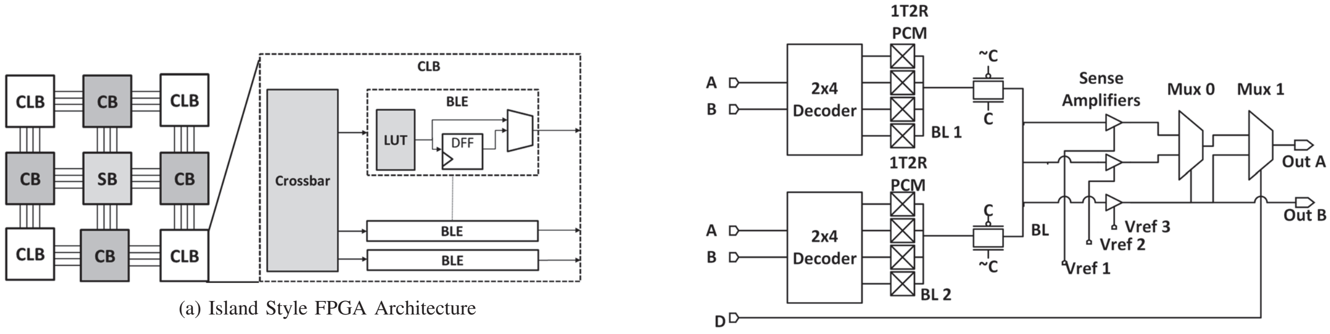
It is known that the majority of area and delay originates from the routing mechanisms. In this work we only focus on the logic clusters of the FPGA. There are two reasons for this; Firstly, the ability to store multiple bits per cell is best exploited from the logic perspective. Secondly, the unique architecture we propose along with the immediate area saving of PCM cells allows for logic cluster sizes to shrink in a fashion that reduces global routing delays.

We summarize our contributions as:

- We exploit the characteristics of Phase Change Memories (PCM) to store multiple bits per cell
- We present a new Look Up Table (LUT) architecture using multi-bit PCM cells. The area of the LUTs is half as that of comparable SRAM LUTs.
- We build SPICE models of our proposed LUT and compare the area and delay characteristics of it with respect to current SRAM based designs using standardized CAD flows and benchmarks.
- We explore the use of PCM as configuration memory for routing mechanisms.

Our paper is organized in five sections. Following this introductory section, Section II presents background on PCM and FPGA architectures. We then present and describe our new LUT architecture in Section III, followed by a discussion on our evaluation setup and methodology in Section IV. In Section V we discuss our results and findings and we then conclude with Section VI and highlight the features and results of our work as well as provide some insight into future explorations.

978-3-9815370-4-8/DATE15/©2015 EDAA



(a) Island Style FPGA Architecture

(b) Architecture of Proposed PCM LUT

Fig. 1: Overview of FPGA Architecture and Proposed LUT Architecture

II. Background

A. Phase Change Memory

Phase Change Memory is a class of non-volatile memories (NVM), that relies on the variance of electrical resistance of a Chalcogenide material to store data by changing its phase. The phase change material is capable of existing in two states; crystalline and amorphous. The amorphous state exhibits a high resistance and is considered to be the *RESET* state which can be described as a binary value of 0, while the crystalline state exhibits low resistance and is called the *SET* state which thus stores a binary value of 1. This change in state is achieved by heating and cooling of the material. In particular, the state of the material depends on the *quenching* phase [5], which in turn is controlled by the programming current applied via the heating element. Applying a large current to melt the crystal and then rapidly cooling by discontinuing the current results in the material transitioning to an amorphous state. Applying a medium current pulse over a relatively longer period of time results in the material transitioning to crystalline state. Each of the above two states exhibit radically different electrical resistance, which allows us to model a PCM cell as a resistor.

An interesting aspect of PCM cells is the ability of the Chalcogenide to exists in intermediate states i.e. exhibit multiple resistance levels. In theory, this allows for multiple bits to be stored per cell. Multi-level cells have been a subject of great interest in academia as well as industrial research [6] [7] [8]. We exploit this ability of PCM to store multiple bits in this work, although we limit it's use to store 2 bits only. Storing 2 bits per cell simplifies the design as only 4 distinct states need to be detected, which allows for a simple sense amp design.

A caveat in the use of PCM is the write stage. As of now, programming PCM requires a significant amount of energy to change the state of the material, which presents a challenge in design of PCM based devices. However, significant interest in PCM technology as well the shrinking size of PCM has reduced the amount of energy required towards programming. In this work we assume that the programming current is not overwhelmingly large to necessitate the need for extremely large MOSFETs, as well as that programming infrastructure can be shared by multiple cells [3].

B. FPGA Architecture

FPGA architectures have been studied quite well and for the most part island style architectures are the most prevalent [9]. In an island style architecture, arrays of configurable logic blocks (CLB) tiles are placed along with programming routing as shown in Fig. 1a. Each CLB connects to the global routing via connection boxes (CB) and switch boxes (SB). CLBs are composed of a group of basic logic elements (BLE) and local routing elements that route incoming global inputs as well outputs of the BLEs. At the lowest level of the hierarchy is a BLE which in turn is composed of a Look Up Table (LUT) and a flip-flop. These LUTs are responsible for implementing logic functions and are key to the basic functionality of FPGAs. Our work focuses on the design of LUTs for FPGAs as well as the internal routing (darker shaded blocks in the CLB in Fig. 1a).

III. Mutli-Bit PCM LUT

As mentioned in section II-A, PCM is capable of existing in multiple states, each of which corresponds to unique resistance values. Our novel LUT design relies on this very ability. Fig. 1b presents the overall architecture of the proposed LUT.

The coupling of 2^K cells with a K input decoder would effectively create a K-input LUT, similar to what conventional FPGAs use. In our design, we program the PCM to operate in any one of four states, allowing us to store two bits per PCM cell. Thus, we have doubled the logic density while requiring half as many physical memory cells. Since half the memory cells are required, then we would need half as many word-lines and half the decoder size. For our particular design, when provided with *K-1* inputs, two output bits can be obtained such that they can be used to store two different functions that share all or a subset of *K-1* inputs. Alternatively, an additional select input can be used to select a single bit from the two bits provided by the cell to form a K-input LUT. Thus, using the resources of a *K-1* LUT, we are capable of implementing a *true K input LUT* with almost half the area.

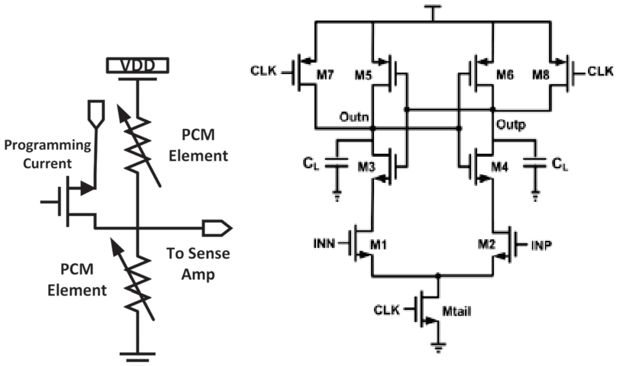
Thus, our proposed LUT can function in two modes; single-output *K LUT* and twin-output *K-1 LUT*. In both modes, *K-1* inputs are fed into the decoder, which provide select lines that allows the selected PCM cell to charge/discharge the shared bit-line. Next, three voltage comparators and a multiplexer are used to determine the two stored bits. When operating in the twin-output mode, first function is stored using the most significant bit (Out B in Fig. 1b) while the second function is stored using the least significant bit (Out A in Fig. 1b). Whilst operating in the single-output mode, the LUT utilizes an additional input signal to select either the most significant bit or the least significant bit as the output (Out A). Thus, the above described LUT can operate either as a

K-input LUT with a single output, or a single LUT with K-1 inputs with two outputs.

This hybrid architecture, is similar in functionality to the *Fractured LUT* design used by manufactures such as Altera and Xilinx. However, it should be noted that the operation and working of our design is fundamentally different. For example, Altera's Adaptive Logic Module, ALM, is composed of several smaller LUTs that can be either used to implement a large LUT or operate independently as two smaller LUTs [10]. In contrast, our design stores two bits per cell in a single LUT, rather than composing two smaller LUTs into a larger LUT; thus, our design can provide almost twice the logic density.

A. Memory Cell Design

The memory cell model for the PCM cell is shown in Fig. 2a. The design is a 1T2R design, similar to that described in [3]. The resistors in the schematic represent a single PCM. We select this design for two reasons. First, the 1T2R design allows us to control the relative resistance of the resistors and thus create an output voltage in between the values of VDD and GND. Second, the leakage current incurred while reading from the cells can be minimized with the 1T2R design by ensuring that the sum of resistances is large. Programming of the 1T2R cell can be performed by controlling the programming current magnitude, duration and polarity. This can be done by enabling the programming transistor shown in Fig 2a and poviding a path to VDD or GND in order to program one of the cells at a time [3].



(a) 1T2R PCM Cell

(b) Sense Amplifier

Fig. 2: Overview of 1T2R PCM Cell and Sense Amplifier Design

Traditional single-bit PCM cells, which store only one bit per cell, have their output swing from 0V to VDD. So, the two PCM cells are programmed to a *one-on one-off* state. For example; to store logic 1, the top memory cell in Fig. 2a is programmed to a *SET* state and the bottom memory cell to the *RESET* state. This way, not only have we created a low resistance between the output and the source but we've also created a high resistance across the memory cells between VDD and GND. This design has fast read speed and low leakage power [3].

For our design, we aim to store two bits in a single memory cell. Using the same memory cell design, we design the output voltages of the cell to be 0, VDD/3, 2VDD/3 and VDD representing the bit vectors 00, 01, 10 and 11 respectively. In order to reduce leakage power, the sum of the resistances from the two cells must be high. In order to acheive this, we rely on two intermediate states, apart from *SET* and *RESET*, such that their resistances are less than that of the *RESET* state. A combination of these intermediate states allows us to create the required output voltage for the bit vectors 01 and 10. However, since the resistance of each of these states is higher than that of the *SET* state, the read speed for the cases 01 and 10 is higher than that of 00 and 11. Thus, on an average, the overall read speed has been sacrificed.

B. Sense Amplifier

Since the output of our memory cell ranges between VDD and GND, we need a sense amplifier circuit to determine the actual outputs from the memory cell. We've employed a sense amplifier circuit design based on *latched voltage comparators* as shown in Fig. 2b.

We use three voltage comparators and provide three different reference voltages; VDD/2 (vref3), VDD/4 (vref1) and 3VDD/4 (vref2) as shown in Fig. 1b . Based on the relation between output voltage and VDD/2, we can directly determine the most significant bit of our output (Out B in Fig. 1b). For example, if the output voltage is greater than VDD/2, then the most significant bit of the output can only be 1 since the data stored is either 10 or 11, else it must be either 00 or 01. Then, based on the most significant output, we can choose which output to use from the other two sense amplifiers, since the reference voltages vref3 and vref1 will provide the least significant bit. This results in a small area overhead when compared to a traditional SRAM-based LUT. However, since every LUT memory cell can share all the sense amplifiers, the area overhead is diminished.

C. Decoder

Traditional SRAM based FPGAs use a *Mux Tree* structure to implement the entire LUT [9]. This usually consists of an array of SRAM cells whose outputs are routed to the final LUT output via a series of transmission gates or pass transistors. However, for our design we cannot rely on such traditional approaches. The *Mux Tree* based designs must resort to buffer insertion and level correcting circuits after every 2 or 3 stages in order to maintain drive strength, and compensate for any voltage drops through the pass gates. Since our 1T2R cells have very precise voltage outputs for different states, any sort of level conversion or two-stage buffer would result in the output swinging to 0 or VDD, effectively losing one bit. Thus, we must resort to individually selecting the appropriate cells and feeding their output to common bit-line. In doing so, all the cells can share the sense amplifiers since they have a common bit-line.

The decoder we use for our design is inspired by the SRAM LUT architecture and is also a tree structure based design. Larger decoders are built by cascading several trees together. As a consequence of using such a design, the decoder area grows exponentially and we see the same phenomenon in an SRAM based LUT as well since it is also a tree based design. However, the PCM based LUT only requires a *K-1* input decoder to implement a K-input LUT, which gives the PCM based LUT a significant area advantage as the LUT size increases.

The corollary to this is that we also note that the LUT delay is high due to the reduced read speed of the PCM cells and the large parasitic capacitance along the bit-line. This is particularly evident as the LUT size increases since the parasitic capacitance is proportional to the number of cells connected to the line.

The delay through the PCM LUT can be expressed as

$$T_{LUT} = T_{decoder} + T_{BitLine} + T_{SenseAmp} + T_{Mux}$$

$T_{SenseAmp}$ and T_{Mux} can be considered as constants and the of delay the decoder grows exponentially with the number of inputs. Out of all of these components, the parasitic capacitance on the bit-line is one of the major limiting factors. Thus, focus should be given to improving this delay. Our solution to this is relatively simple; fracture/split the bit-line, as shown in Fig. 1b. Through simulation, we found that the output voltage of the PCM cell can be passed through up to three transmission gates reliably and we leveraged this knowledge.

In Fig. 1b, a four input LUT is implemented by composing two smaller 2-input decoder-memory arrays into a larger LUT. Note that we show two 2x4 decoders in the figure since our PCM LUT requires $K - 1$ input decoders only. The output of each PCM row is then selected through a single level transmission-gate based mutiplexer. In doing so, the bit-line capacitance seen by the PCM cells has been halved. In practice, we will use up to three levels of mutiplexers which will reduce the capacitance seen by up to a factor of eight, approximately. This significantly improves the delay characteristics of the PCM LUT without any area overhead.

D. Crossbar Area Minimization

A large portion of the overall delay is caused by global routing rather than logic delay in the LUT. As we stated in the introduction, in this work we focus only on the logic clusters where the overall delay can be improved in two ways; reducing the delay through the LUT or shrinking the CLB size to reduce the impact of global routing.

We model the internal routing of the CLB via a fully connected crossbar where the configuration bits are stored in SRAM arrays that drive pass transistors. Once configured, routing paths do not change at run-time. Thus, the delay through the routing structure depends only on the length of interconnects, the size of the pass transistors and the line parasitics. The configuration memory doesn't directly contribute to the delay. However, the larger the memory cell, the longer the interconnect. Thus, if we reduce the area overhead of the configuration memory, we can reduce the delay through the crossbar as well as the overall size of the CLB which in turn would reduce global routing overheads.

In order to reduce the overhead of the crossbar configuration memory, we replace the SRAM configuration memory cells with 1T2R PCM cells, similar to what is done in related works for global routing [15]. We use these PCM cells to store a single bit only. No additional sense amplifiers or logic is required.

IV. Evaluation Setup

Fig. 3 describes our evaluation setup and methodology. We evaluate our LUT design using SPICE based simulation models of the circuits and following standardized CAD flows.

A. Circuit Design and SPICE Models

In order to derive accurate delay and area values, we use SPICE models simulated with Synopsys HSPICE. The PCM LUT as well as the reference SRAM LUT design were designed at the 22nm node based on the predictive technology model [12]. Rather than relying on automatic sizing tools, all the circuits have been designed and sized manually to ensure the smallest size. As we noted in Section III-A, the read resistance of the states 01 and 10 are higher than 11 and 00, thus the delay for 00 → 10 is the worst case delay, while 11 → 00 is the best case. We use the worst case value of both delays as the delay for the LUT.

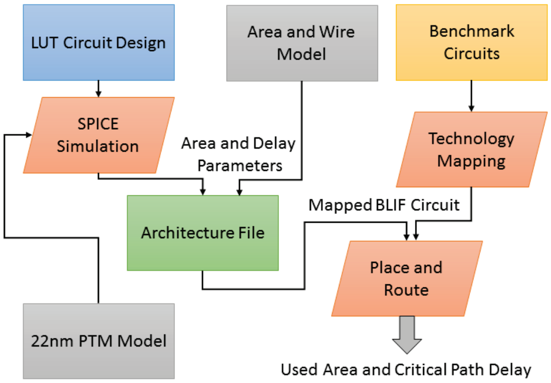


Fig. 3: Evaluation Setup and Methodology

As a reference for comparison, we compared our PCM LUT to an SRAM based LUT capable of operating in a fractured LUT mode. We modeled the fractured operation as two smaller sized LUTs along with an output mux, similar to the design described in [10]. The SRAM LUT was designed using the *Mux Tree* design with minimum sized transmission gates. SRAMs used in the LUT were designed to be as small as possible using a 6-T design. We estimate the size of a single PCM cell to be four times smaller than SRAM based on [3].

TABLE I: Architecture and Memory Cell Parameter Values

Parameter	Value
CLB Size (N)	10 LUTs
LUT Sizes (K)	6,8,10,12
CLB Input Size	$N * (K - 2)$
Routing Channel Width	300
Segment Length	4
Segment Direction	Unidirectional
Fc In	0.15
Fc Out	0.10
Routing Mux	Pass transistor based
SRAM Cell Area, Delay	14 MWTA,11.59ps
1T2R PCM Cell Area, Delay	7 MWTA, 18.44ps
PCM Cell Resistance Range	$10K\Omega$, $30K\Omega$, $60K\Omega$, $100K\Omega$

B. Benchmarks and Simulation

Having derived area and timing values for both the SRAM and the PCM LUTs we evaluate both of them via simulation of the placed and routed designs. In order to do this we use standardized benchmarks from the MCNC benchmark set as well as larger benchmarks from the IWLS 2005 benchmark set.

We performed technology mapping for the selected benchmarks using ABC [13] and to perform place and route, we used the VPR implementation provided in the VTR package [14]. The architecture file was developed based on the reference

40nm architecture file provided by VPR. For an accurate model, we scaled the resistance, capacitance and delay parameters specified from 40nm to 22nm. In particular, resistance and capacitance values for routing channels and devices were scaled, as well as delays for flip-flops, routing buffers and wire line delays. The metal resistance and capacitance values for routing tracks were scaled based on the lengths of the CLBs. The CLB area and delay parameters were created based on the area of the LUTs and our crossbar model. Delay values of the crossbar were scaled proportionally to the number of inputs, outputs and the length.

Table I provides additional details about the architecture under evaluation, the size and worst-case delay of the SRAM cell and the 1T2R PCM cell as well as the range of resistance values used for the four states of the PCM cells. Note, that the area is expressed in terms of *Minimum Width Transistor Area (MWTA)*.

V. Results

Having completed the evaluation flow described in Section IV, we now proceed to discuss the results of our study.

First, we compare the performance of our PCM LUTs with that of SRAM based LUTs. Table II presents the size and delay of a single PCM LUT with respect to that of an SRAM based LUT. It also shows the size of our architecture's CLB with respect to a full SRAM based implementation.

TABLE II: Area and Delay of PCM LUT vs SRAM LUT

LUT Size	LUT Area (MWTA)		LUT Delay (ps)		CLB Area(MWTA)	
	SRAM	PCM	SRAM	PCM	SRAM	PCM
K6	1884	1094	109	129.35	72840	28940
K8	7530	4214	125.5	214.43	171300	74140
K10	30120	16700	142.5	336.15	451200	217000
K12	120468	66614	150	844.032	1420680	738140

As expected, the area of our PCM LUTs reduces by nearly 50% which indicates that we have effectively doubled the logic density of the LUTs. These area savings are a combination of two factors; The use of PCM cells to store 2 bits per cell and the resulting halving of the decoding resources. In addition, the use of PCM as configuration memory in the crossbar has resulted in the CLB being half as small as well. However, the exponential rise in delay due to the parasitic capacitance becomes painfully evident as the size of the LUT increases. In particular, LUT size of 12 is over five times slower than it's SRAM counterpart. While this is certainly a setback, the overall performance is largely dominated by global routing which we can expect to reduce as the overall size of the CLB has shrunk as well.

Table III presents the results of our place and route simulations. We present the Area, Delay and the Delay-Area Product (DAP) of the optimized PCM cells with respect to the reference SRAM design. Note, that the area presented here is the sum of the area occupied by logic and routing resources, while the delay is the sum of the total logic delay and the net delay i.e. the critical path delay of the circuits. We also compare the area and delay of each PCM LUT with a K6 SRAM LUT as well. It is commonly accepted that a six input LUT is the most optimal solution. However, this theory is based on an SRAM LUT. We will explore whether this holds true for PCM LUTs as well.

The increased logic density is evident from the data presented in Table III where we can see more than 40% reduction in area on an average when compared to SRAM based LUTs. LUT sizes of 6, 8, 10 and 12 show 40.1%, 46.58%, 47.29% and 46.49% improvements in area. We also see that the larger benchmarks such as DSP, DMA and des_perf show almost identical area savings, indicating that larger real-world circuits will be able to exploit the area benefits of our architecture.

An important aspect of the results of our evaluation are with regard to delay and DAP. The most significant impact on the critical path delay stems from global routing, which in turn is a function of the length each segment must travel, and the RC parameters of the metal used in routing. By improving the logic density of the LUTs and the use of PCM cells in internal routing, the overall reduction in CLB area should provide scope to improve the delay characteristics.

As we see in Table III, on an average, 7% to 13% critical path delay improvement can be achieved by using the proposed LUT architecture. However, we note that as the LUT size increases, the overall delay improvement worsens. LUTs of size 6, 8, 10 and 12 have improvements of 13.46%, 8.81%, 7.06% and −3.72% respectively. Notably, we see that K12 is rather impractical due to it's worsened delay improvements. LUTs of size 6 and 8 have proven to be better than their SRAM counterparts. This is particularly important since K6 and K8 are the dominant LUT sizes in most commercial FPGAs today. On a more interesting note, the larger benchmarks such as DSP and DMA consistently show much higher delay improvements, in the order of 30% to 40% across all LUT sizes. This can be explained by the fact that the impact of global routing overheads is much larger than intrinsic LUT delay for these benchmarks due to their size. So, despite the inherently slower LUTs, real-world circuits will benefit from both area and delay savings.

The question that remains is whether K6 is indeed the most optimal selection. To better understand this, we shall look at the PCM LUTs of size 6 and 8. Larger LUTs are inherently slower and they tend to be more tedious to design and have larger leakage power losses, and our data suggests that K6 and K8 are the best performers. When compared to a K6 SRAM LUT, the PCM LUT's delay and area advantage for larger LUT sizes is diminished and the results suggest that when compared to the current optimal selection, K6 SRAM, the six input PCM LUT is the best choice with a DAP of close to 50% as compared to the K8 PCM LUT with a DAP improvement of just 18.81% over the K6 SRAM LUT. A stark contrast to these findings is the performance of the larger LUTs of size 10 and 12 with respect to the K6 SRAM. Both K10 and K12 show very poor performance in terms of area and delay.

Thus, we suggest that for FPGAs based on our PCM based architecture, K6 LUTs are still the best selection.

VI. Conclusion

In this work we have presented a new PCM based FPGA architecture. Our architecture utilizes PCM cells to store 2 bits per cell, and relies on PCM for configuration memory of internal CLB routing. The resulting architecture was evaluated for a variety of benchmarks and we demonstrated significant area savings, with over 40% reduction in area. The reduced size of the CLB resulted in overall delay reductions of 8% to 13%. We concluded that LUT sizes of 6 provided the best

TABLE III: Area, Delay and Delay-Area-Product of PCM LUT relative to SRAM LUT

$Baseline$	PCM LUT	$Parameter$	aes_core	des_perf	$diffeq$	DMA	DSP	$elliptic$	$ex1010$	$ex5p$	$misex3$	seq	$spla$	$Average$
One to One comparison of PCM LUTs with equivalent size SRAM LUT as baseline														
K6 SRAM	$K6$	$Area$	0.57	0.57	0.59	0.57	0.57	0.60	0.59	0.61	0.60	0.74	0.58	40.10%
		$Delay$	0.84	0.69	0.98	0.76	0.74	0.92	0.90	1.00	0.87	0.94	0.87	13.46%
		DAP	0.48	0.39	0.58	0.44	0.42	0.55	0.53	0.61	0.52	0.70	0.51	47.95%
K8 SRAM	$K8$	$Area$	0.53	0.52	0.54	0.52	0.52	0.54	0.55	0.55	0.54	0.54	0.53	46.58%
		$Delay$	0.73	0.71	1.09	0.58	0.67	1.04	1.20	1.00	0.94	1.06	1.01	8.81%
		DAP	0.38	0.37	0.58	0.30	0.35	0.56	0.66	0.55	0.51	0.57	0.54	51.13%
K10 SRAM	$K10$	$Area$	0.52	0.52	0.53	0.52	0.52	0.53	0.56	0.53	0.53	0.53	0.52	47.29%
		$Delay$	0.85	0.68	1.13	0.68	0.64	1.09	1.24	1.02	1.06	0.92	0.92	7.06%
		DAP	0.44	0.35	0.59	0.35	0.33	0.57	0.70	0.54	0.56	0.49	0.48	50.86%
K12 SRAM	$K12$	$Area$	0.53	0.53	0.53	0.53	0.53	0.54	0.55	0.54	0.54	0.53	0.53	46.49%
		$Delay$	0.62	0.61	1.13	0.63	0.54	1.36	1.76	1.15	1.29	1.20	1.12	−3.72%
		DAP	0.33	0.33	0.60	0.33	0.29	0.73	0.96	0.62	0.69	0.64	0.60	44.38%
One to One comparison of PCM LUTs with K6 SRAM LUT as Baseline														
K6 SRAM	$K8$	$Area$	1.04	1.05	0.80	1.04	1.04	0.73	0.35	0.60	0.73	0.80	0.73	18.86%
		$Delay$	0.83	0.93	1.13	1.00	1.11	1.09	0.84	0.69	1.00	1.20	1.09	0.95%
		DAP	0.87	0.97	0.90	1.04	1.15	0.80	0.30	0.41	0.73	0.96	0.79	18.81%
K6 SRAM	$K10$	$Area$	2.46	2.50	1.36	2.47	2.46	1.66	0.06	1.33	1.07	1.36	1.09	−62.07%
		$Delay$	1.48	1.62	1.46	1.82	1.90	1.26	0.38	0.92	1.24	1.50	1.26	−35.07%
		DAP	3.65	4.06	1.99	4.49	4.68	2.10	0.02	1.22	1.34	2.05	1.37	−145.16%
K6 SRAM	$K12$	$Area$	7.55	7.62	4.12	7.59	7.57	3.22	0.16	3.96	3.22	4.12	2.53	−369.52%
		$Delay$	3.06	3.95	2.54	4.15	3.81	2.36	0.69	1.77	2.25	3.35	2.72	−178.66%
		DAP	23.14	30.08	10.47	31.51	28.84	7.59	0.11	7.01	7.24	13.79	6.87	−1414.95%

area-delay trade-offs for our PCM based architecture.

In our introduction, we stated that we consider this work as a step towards scalable NVM based FPGAs. We believe that our architecture, based on the results shown, is indeed a step towards scalable and high density FPGAs. While, we have only concentrated on the development of a new CLB, our future work will also explore the use of PCM for improvements in global routing mechanisms such as the switch block and the connection block.

Acknowledgment

This work was supported in part by the Center for Future Architectures Research (C-FAR), one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA

References

[1] M. Lin, A. El Gamal, Y.-C. Lu, and S. Wong, “Performance benefits of monolithically stacked 3-d fpga,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, Feb 2007.

[2] C. Dong, C. Chen, S. Mitra, and D. Chen, “Architecture and performance evaluation of 3d cmos-nem fpga,” in *System Level Interconnect Prediction (SLIP), 2011 13th International Workshop on*, 2011.

[3] P.-E. Gaillardon, M. Ben-Jamaa, M. Reyboz, G. Beneventi, F. Clermidy, L. Perniola, and I. O’Connor, “Phase-change-memory-based storage elements for configurable logic,” in *Field-Programmable Technology (FPT), 2010 International Conference on*, Dec 2010, pp. 17–20.

[4] S. Raoux, G. Burr, M. Breitwisch, C. Rettner, Y. Chen, R. Shelby, M. Salinga, D. Krebs, S.-H. Chen, H. L. Lung, and C. Lam, “Phase-change random access memory: A scalable technology,” *IBM Journal of Research and Development*, vol. 52, no. 4.5, pp. 465–479, July 2008.

[5] H. S. P. Wong, S. Raoux, S. Kim, J. Liang, J. P. Reifenberg, B. Rajendran, M. Asheghi, and K. E. Goodson, “Phase change memory,” *Proceedings of the IEEE*, vol. 98, no. 12, pp. 2201–2227, Dec 2010.

[6] N. H. Seong, S. Yeo, and H.-H. S. Lee, “Tri-level-cell phase change memory: Toward an efficient and reliable memory system,” in *Proceedings of the 40th Annual International Symposium on Computer Architecture*, ser. ISCA ’13, 2013.

[7] N. Papandreou, H. Pozidis, A. Pantazi, A. Sebastian, M. Breitwisch, C. Lam, and E. Eleftheriou, “Programming algorithms for multilevel phase-change memory,” in *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, 2011.

[8] N. Papandreou, H. Pozidis, T. Mittelholzer, G. F. Close, M. Breitwisch, C. Lam, and E. Eleftheriou, “Drift-tolerant multilevel phase-change memory,” in *Memory Workshop (IMW), 2011 3rd IEEE International*, 2011.

[9] V. Betz, J. Rose, and A. Marquardt, Eds., *Architecture and CAD for Deep-Submicron FPGAs*. Norwell, MA, USA: Kluwer Academic Publishers, 1999.

[10] “Improving fpga performance and area using an adaptive logic module,” in *Field Programmable Logic and Application*, ser. Lecture Notes in Computer Science, J. Becker, M. Platzner, and S. Vernalde, Eds., 2004, vol. 3203.

[11] W. Zhang and T. Li, “Characterizing and mitigating the impact of process variations on phase change based memory systems,” in *Proceedings of the 42Nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO 42. New York, NY, USA: ACM, 2009, pp. 2–13.

[12] W. Zhao and Y. Cao, “New generation of predictive technology model for sub-45nm design exploration,” in *Quality Electronic Design, 2006. ISQED ’06. 7th International Symposium on*, 2006. [Online]. Available: http://ptm.asu.edu/

[13] “Berkeley logic synthesis and verification group, abc: A system for sequential synthesis and verification, release 1.01 131122. http://www.eecs.berkeley.edu/ alanmi/abc/.”

[14] J. Luu, J. Goeders, M. Wainberg, A. Somerville, T. Yu, K. Nasartschuk, M. Nasr, S. Wang, T. Liu, N. Ahmed, K. B. Kent, J. Anderson, J. Rose, and V. Betz, “VTR 7.0: Next Generation Architecture and CAD System for FPGAs,” vol. 7, no. 2, June 2014, pp. 6:1–6:30.

[15] Cong, J.; Bingjun Xiao, ”FPGA-RPI: A Novel FPGA Architecture With RRAM-Based Programmable Interconnects,” Very Large Scale Integration Systems, IEEE Transactions on , vol.22, no.4, pp.864,877, April 2014