A Practical Quaternary FPGA Architecture Using Floating Gate Memories

by

Ayokunle Fadamiro

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
May 2, 2020

Keywords: FPGA, LUT, multi-valued logic, floating gates, MVL memory element, FinFET

Approved by

Spencer Millican, Chair, Assistant Professor of Electrical and Computer Engineering
Christopher Harris Co-Chair, Assistant Professor of Electrical and Computer Engineering
Adit Singh, James B. Davis Professor of Electrical and Computer Engineering

Abstract

A new quaternary FPGA (QFPGA) architecture based on floating-gate memory elements is presented and analyzed. While technology scaling has delivered substantial FPGA performance, consumer demands grow beyond what binary circuits can deliver. FPGAs implementing multi-valued logic (MVL) have been explored, but previously proposed architectures rely on non-standard fabrication techniques and optimistic performance analysis. Results show the proposed QFPGA implementation based on floating-gate memories has competitive delay and power performance compared to equivalent binary implementations and previous QFPGA architectures from literature when simulated in FinFET technology.

Acknowledgments

I take this opportunity to express sincere thanks and heartfelt appreciation to my academic supervisors. Dr. Spencer Millican, Dr. Christopher Harris, and Dr. Adit Singh have been a continual source of support, guidance, and encouragement throughout my graduate program. It has been a great privilege to work with these dedicated academics and innovative engineers. I especially appreciate Dr. Millican for his patience; his impact on my growth as a researcher cannot be overstated. I acknowledge with much gratitude the financial and career support of Dr. Jason Clark who really cares and gave me the opportunity to begin this exciting research journey.

My uncle Dr. Henry Fadamiro has always been a pillar of support in all aspects of my life and I would not have considered completing a Masters program at this time if not for his timely advice and encouragement. For his unending guidance especially in tough emotional times, I will always be grateful.

I appreciate all the unique love and care from my immediate family. My dear wife and daughter are my biggest support, anchor, and inspiration. All the special love they shower on me makes everything worth it. With their prayers, words, and ever so optimistic perspective, my mum and dad give me wings, grace to become anything I dream of. I continue to benefit from their continuous investment in my emotional, educational well being. My dear brother and sister always found a way to remind me that they have my back. I continue to strive to be the best I can be to inspire the best from them.

To conclude, I wish to extend my thanks to my class instructors, department professor, my colleagues in the Samuel Ginn College of Engineering, and many friends through this journey; for the hours of education, play, insight, and everyday learning. God bless you all.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Growing technology demands in novel computing architectures, like multi-valued logic (MVL), necessitate programmable quaternary MVL architectures compatible with modern circuit fabrication technology. MVL designs reduce circuit area and interconnection lengths in integrated circuits (ICs) and therefore reduces power dissipation and circuit delay. When applied to FPGAs, MVL can reduce the number of interconnects by transferring more information compared to traditional binary logic, which decreases the delay and area of FPGA [6–8]. Additionally, MVL allows more complex circuits to be implemented with faster speeds and less power, thus MVL becomes more relevant as transistor scaling continues.

Numerous studies proposed MVL implementations, [9–12] such as ternary (3-value logic) [13, 14] and quaternary (4-value logic), with the latter receiving more focus [1, 3, 15]. These designs improve performance and reduce power consumption. Previous studies implemented programmable quaternary logic [1, 16] (or presumed quaternary logic was programmed from another source [3]) to create a "quaternary LUT" (QLUT): when implemented in a quaternary FPGA (QFPGA), these QLUTs can be controlled with quaternary inputs and can output/store quaternary values.

While the aforementioned designs are sound in theory, their analysis and practicality are questionable. Previous designs made use of transistors with different threshold voltages and require numerous voltage supply rails [1] and thus pose manufacturability challenges. Additionally, some designs did not consider the source of quaternary values in memory [3], and optimistic timing and power simulations were used [1, 3].

This thesis proposes a new QFPGA BLE architecture based on practical fabrication techniques (i.e., floating-gate transistors) and compares its performance against contemporaries from literature [1, 3]. To avoid favoritism to any architecture, simulatable circuit schematics and circuit layouts for all compared FPGA architectures (the proposed architecture, the architecture from [1], and an equivalent binary) are implemented in the same technology [17]. Comparisons consider maximum transition delays, dynamic and static power consumption, and power density with respect to circuit areas. Unlike previous studies on QFPGA BLE performance [1], this study considers the manufacturability of FPGA architectures by implementing circuit layouts and pessimistically simulating circuit performance.

The specific contributions of this thesis are as follows:

- A discussion of the motivation for MVL and FPGA architectures in digital/analog circuit designs.
- A description of a unique MVL memory element based on floating-gate transistors which is compatible with modern FinFET fabrication technology.
- A description of a QFPGA basic logic element (BLE) which can be implemented with modern FinFET fabrication technology.
- A fair analysis of binary and quaternary FPGA architectures and a demonstration of the utility of quaternary logic for increasing FPGA performance.

The remainder of this thesis is organized as follows: Chapter 2 motivates the need for superior quaternary architectures while Chapter 3 discusses previous FPGA architectures. A practical QFPGA architecture using a unique MVL memory element is presented in Chapter 4. Chapter 5 describes how the proposed architecture and quaternary/binary equivalents are fairly simulated and evaluated. Chapter 6 presents detailed results and analysis, and this thesis concludes with Chapter 7.

Chapter 2

Motivation

It is widely believed that without new design techniques, the explosive performance benefits of Moore's law in sub-micron process technology nodes cannot be sustained. For the past six decades, Moore's law has been the driving force behind fabrication technology node scaling. Moore's law predicted the doubling of chip transistor density (i.e., the number of transistors in a single chip) every 18 months as a direct effect of the continuous shortening of transistor gate lengths. Very large scale integrated circuit (VLSI) fabrication enables packing up to billions of transistors in a single chip and accelerating performance of digital circuits in an unprecedented manner. These shrunk transistors have reduced node capacitance and can employ lower supply voltages to save power consumption while packing more functionality in the same chip.

However, as transistors shrink into smaller technology nodes, fabrication companies like Intel struggle to resolve transistor gate fabrication challenges, thus leading to increased power consumption in sub-micron technologies that is magnified by increased gate leakage current. At the $45nm$ process technology node, Intel hit the limit for transistor gate length that was possible for normal transistor operation [18]. Subsequent shrinkage of transistor dimensions did not include the gate length and power consumption remained high [19]. When Intel introduced FinFET (in the $22nm$ node), a type of transistor with better gate control, gate leakage current was driven down while the gate length remained more or less constant. Additionally, with massive and complex transistors circuits jammed into a chip, power consumption due to interconnections (as a result of routing wire capacitance) exceeded that of the transistors and became a major challenge.

Modern VLSI technology increasingly leans towards creative digital circuit techniques as alternative means of decreasing power dissipation in a chip. From a circuit design perspective, the major factors influencing power and speed performance are transistor supply voltage, circuit capacitance, and the number of active transistors required for a given circuit operation. Since transistor supply voltage is dictated by process technology and is inversely proportional to static energy consumption: (i.e., a reduction in supply voltage increases leakage current), the potential for energy efficient designs lies in the other two factors. A new circuit design that reduces transistor count and capacitance due to interconnection between transistors would achieve the desired reduction in power dissipation and pave the way for more complex circuits while allowing more transistors to be packed on a single chip. Additionally, reducing wiring complexity means less circuit delay and faster speed performance.

2.1 Multi-Valued Logic (MVL) and Performance Benefits

Multi-valued logic (MVL) also called many-valued logic, is a digital logic paradigm that extends the scope of classical logic by considering more than two values [20]. Classical logic is based on the principle of bivalence: that there can be only two truth values, "true" and "false" (also known as binary logic). MVL, like binary logic, accepts the use of truth functionality to define states of a system, but deviates from the principle of bivalence to allow more than two truth degrees [21]. These truth values define the radix of the system: the radix is the number of unique digits used to represent the total number of truth values. In digital logic design, MVL provides a broadened use of voltage or current differences between circuit nodes. When implemented with voltages, for example, binary logic with logic levels 0 and 1 represented by the ground voltage level of 0 V and V_{DD} respectively, can be extended to N logic levels that exist between 0 V and V_{DD}.

There are proven arithmetic advantages of MVL systems [22, 23] and physical benefits relating to dynamic energy consumption [24]. [22] showed that the use of a small swing voltage in MVL allows circuit operations to be performed at on-average lower voltages. Because

voltages representing MVL values are closer together compared to binary equivalents, the average energy consumption for MVL transitions is less than binary circuits [1]. An MVL circuit with transistors that use voltage levels closer to one another will also have faster transistor switching and therefore better transition speeds.

Related to routing complexity, [22] showed significant savings in hardware when MVL circuits are implemented. These savings are related to a reduction in the routing complexity in MVL because less logic elements are required for the same circuit implementation. In VLSI, interconnections occupy the majority of a chip. The number of interconnections required for a $N-valued$ logic can be reduced to $1/(\log_b \mathrm{N})$ (the interconnection equation) [25], which expresses the reduction of interconnections when a $b-valued$ logic system is implemented as a $N-valued$ logic system. When applied to a two dimensional silicon chip with b as 2 (for binary logic), it gives a reduction ratio of $1/(\log_2 \mathrm{N})^2$. This clearly shows potential for substantial interconnection area reduction for $(K > 2)$-valued logic.

Delays in digital circuits can also be reduced by considering less interconnections. The complexity, length, and area of interconnections introduce additional resistance and capacitance and limit the scaling of digital circuits in VLSI. Delay induced by interconnection resistance and capacitance are related by $t\ \alpha\ (R \cdot C \cdot L^2)$ (from [26]) where L is the interconnection length, R and C are the resistance and capacitance per unit length respectively. With MVL, there is less interconnection numbers and length required for digital implementations, thus faster circuits can be designed. With shorter interconnection lengths, there is smaller capacitance between wires and this leads to reductions in dynamic power dissipation [27] and cross talk noise.

Quaternary logic is a MVL of radix four that is proven to be more efficient than a radix logic of five, six, etc., and is preferred for its easy transition between the two classical (binary) states by adding two intermediate states between the low and high binary states. It has been shown that quaternary logic only requires small changes in fabrication technology to modify standard process voltage levels [25]. While quaternary logic inherits the benefits

of MVL, [25] mentions specific performance advantages of quaternary logic over binary logic. The benefit of quaternary logic implementation in a case where b is 2 (binary logic) and N is 4 for quaternary logic, using the aforementioned interconnection equation becomes $1/(\log_4 2) = 2$ and demonstrates that quaternary logic would require at least half as many components than binary logic for building the same digital system.

Quaternary logic will have less average dynamic power consumption compared to binary logic due to the distance between respective voltage transitions. This can be shown with a quick comparison of sample logic case considerations. Dynamic power consumption is related to the node capacitance C and power supply voltage V_{DD}. Consider binary logic levels represented by 0 V and V_{DD} (as 0 and 1) while $0 \cdot V_{DD}$, $1/3 \cdot V_{DD}$, $2/3 \cdot V_{DD}$, and V_{DD} correspond to 0, 1, 2, and 3 quaternary logic representations, respectively. Dynamic switching power (P_d) for transistors using either logic can be simplified as $P_d = C \cdot {V_{DD}}^2$ where C and V_{DD} are the same for both quaternary and binary logic cases and $t_{i \to j}$ represents transition from level i to j. In Table 2.1 below, calculations for average power consumption in both cases shows quaternary logic having about 31% less ($0.38 \cdot P_d$) average power consumption to binary logic's ($0.5 \cdot P_d$).

Table 2.1: Average power calculations for transitions in binary and quaternary logic

	Binary logic		**Quaternary logic**	
	Transitions	**Power (W)**	**Transitions**	**Power (W)**
	$t_{0\to0}$, $t_{1\to1}$	0	$t_{0\to0}$, $t_{1\to1}$, $t_{2\to2}$, $t_{3\to3}$	0
	$t_{0\to1}$, $t_{1\to0}$	C $\cdot {V_{DD}}^2$	$t_{0\to1}$, $t_{1\to2}$, $t_{2\to3}$, $t_{3\to3}$,$t_{2\to1}$, $t_{1\to0}$	$C \cdot (1/3 \cdot V_{DD})^2$
	-	-	$t_{0\to2}$, $t_{2\to0}$, $t_{1\to3}$, $t_{3\to1}$	$C \cdot (2/3 \cdot V_{DD})^2$
	-	-	$t_{0\to3}$, $t_{3\to0}$	C $\cdot {V_{DD}}^2$
Avg. power	-	$0.5 \cdot C \cdot {V_{DD}}^2$	-	$0.39 \cdot C \cdot {V_{DD}}^2$

While the MVL benefits discussed have been known in literature for while, the requirement of non-standard CMOS techniques for implementing proposed MVL architectures have slowed its path to practicality and competitiveness. To experience MVL's benefits in real-world applications, MVL designs that are compatible with current manufacture technologies

are needed. Research has shown that digital logic systems such as FPGAs can achieve significant performance improvements with MVL [3].

2.2 Field-Programmable Gate Arrays (FPGAs)

Since their commercial introduction in 1985 by Xilinx co-founders Ross Freeman and Bernard Vonderschmitt [28], FPGAs have grown to become one of the most widely used devices for digital applications. Propelled by advancements in process technology, FPGAs are currently applied in every major industrial, automotive, consumer application, and even in sensitive applications such as in high end data centers [29–31]. FPGAs are collections of many programmable logic elements without defined interconnections. Unlike application specific integrated circuits (ASICs) with defined interconnections and logic gates fabricated on silicon, interconnections and memory values in FPGAs can be re-programmed post-fabrication to model any digital circuit.

The programming technology used in a FPGA determines its how many times it can be programmed. Antifuse-based FPGAs have open switches in their interconnection structure: when these switches are blown, a permanent connection is made. Since this process cannot be reversed, this type of FPGA can only be programmed once. SRAM-based FPGAs store configuration data (as voltage) for the interconnections in the two cross-coupled inverters and can be re-programmed infinite number of times. However, SRAM memories are volatile, and the configuration data must applied every time the FPGA is turned on. Although SRAM-based FPGAs consume more power than antifuse-based FPGAs, they are preferred for their ability to be re-programmed more than once.

With FPGAs, faster verification of hardware designs can be achieved by reprogramming configuration bits: FPGAs do not incur the high manufacturing cost associated with prototyping ASICs. This advantage makes FPGAs the preferred choice for low volume prototyping of digital systems. Initially, FPGAs were solely used to prototype digital implementations intended for ASICs [32], but future FPGAs provided very short time-to-market, and low

non-recurring costs compared to ASIC implementations. With recent process technology developments, the speed of FPGAs has increased, power consumption reduced, and prices decreased. Additionally, FPGA computer aided design (CAD) tools that are cheaper than full fabrication CAD tools for ASIC implementations have been introduced. To an extent, FPGAs have become more competitive compared to ASICs because of these improvements.

However, complex FPGA implementations are larger and slower than comparable ASIC versions and high volume production of FPGAs is expensive. This is due to the reconfigurability cost: FPGAs have predefined layouts that contain multiple routing paths which aid flexibility but hamper optimization. The programmable blocks in FPGAs and the interconnections between them use more layout area. This introduces larger delays in FPGAs since FPGAs cannot be optimized to reduce interconnection delays like ASICs can. This also increases the cost of layout material required to implement a comparably larger FPGA system.

For FPGAs to remain competitive, FPGA architecture performance and logic density must improve. This paper proposes such improvements to FPGA architectures using MVL.

2.3 Conventional FPGA Architectures

A conventional FPGA architecture, developed by Betz et. al. [2], is the basis of many research studies and tools on FPGA architectures. The structure of this architecture (Figure 2.1) consists of two-dimensional matrix of configurable logic blocks (CLBs) organized in rows and columns.

2.3.1 Routing Interconnections

Two categories of routing interconnections, routing within each CLBs and routing connecting CLBs together, consume most of a FPGA's chip area. Figure 2.1 shows rows of CLBs connected with horizontal routing interconnections and columns of CLBs with vertical routing interconnections. Switch boxes at the intersection of horizontal and vertical

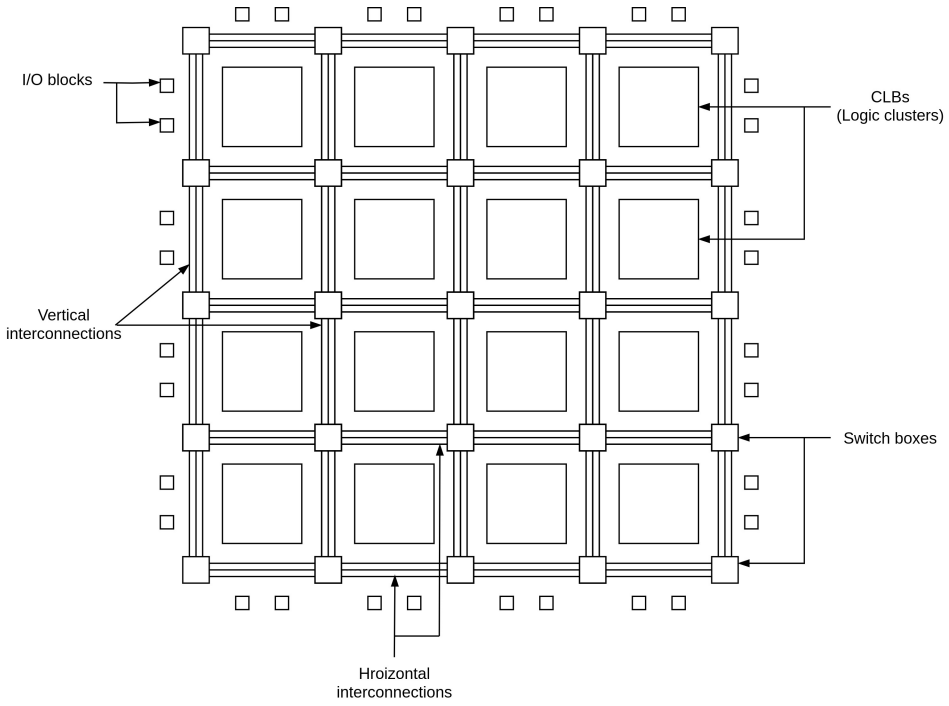

Figure 2.1: FPGA architecture overview described in [2].

interconnections allow programmable connectivity between the interconnections. In practical CLB implementations, 2-to-1, 4-to-1 multiplexers route logic among external resources and within the blocks. Circuit delay from FPGA routing architecture dominates delay and determines the logic density in the chip. With more advanced technology processes, the impact of the interconnections become more crucial [33]. For instance, interconnection power losses have a greater effect on FPGAs where 60% of the chip area is dedicated to routing resources [34].

2.3.2 Input and Output (I/O) Blocks

I/O blocks are included at the borders of the architecture to allow sending and receiving signals between the FPGA structure and external systems. The I/O block also controls programmable drivability and allows impedance matching to different I/O standards. There are as many as 40 different I/O standards that can be supported using dedicated transceiver

circuits [35]. Characteristics of these I/O standards range from analog to digital conversion and support of a variety of supply voltages.

2.3.3 Configurable Logic Blocks (CLBs) & Basic Logic Elements (BLEs)

The logic used for digital processing operations in a FPGA architecture is contained in CLBs. CLBs are responsible for implementing the stored logic to achieve a desired function. A CLB consists of a set of inputs, a set of outputs, and several connected basic logic elements (BLEs) with an input crossbar interconnecting all BLEs within the CLB. The input crossbar

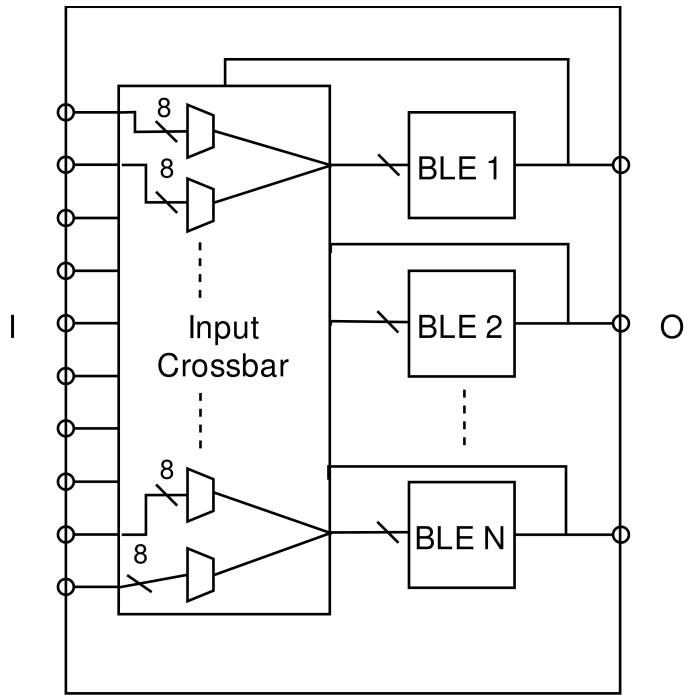

Figure 2.2: Configurable Logic Element (CLB) with N BLEs, I inputs and O outputs.

is made of 2-to-1 multiplexers or more complex multiplexers. The size of the input crossbar is determined by the number of CLB inputs to be selected from (by each multiplexer) and the number of BLEs. The multiplexers select from all possible CLB inputs to multiple BLE inputs and are controlled by programmable logic. Figure 2.2 shows a basic CLB structure with I inputs, O outputs, and N BLEs. Feedback from the BLE outputs to the input crossbar allows execution of more complex functions. For this thesis, a simple CLB with 8 BLEs is

selected for compatibility with 4-input and 2-input BLEs. Within the input crossbar, 8-to-1 multiplexers select from 8 inputs to the BLEs.

The focus and contribution of this thesis is the basic BLE (selected for simplicity) with details shown in Figure 2.3, which consists of a LUT connected to a simple 2-to-1 multiplexer: this multiplexer selects the FPGA output, which is either from the LUT or from a flip-flop driven by the LUT (with clock input).

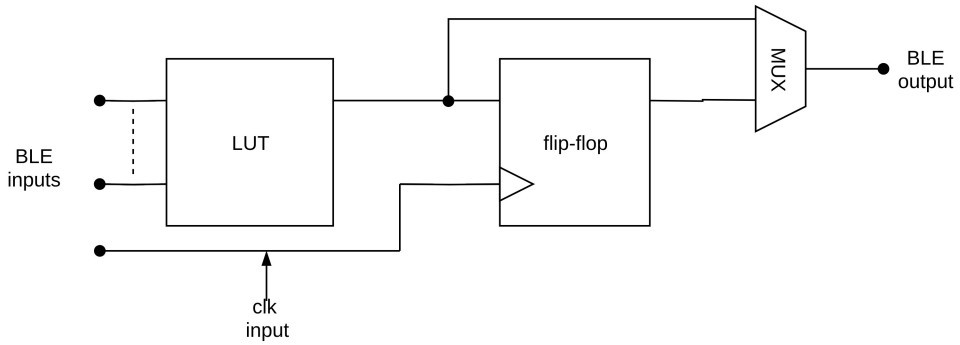

Figure 2.3: Basic Logic Element (BLE)

2.4 Look-Up Table (LUT) Performance

Besides FPGA BLE interconnections, LUT complexity significantly impacts the performance characteristics of FPGAs. LUTs are memory elements which store and implement the logic functions programmed in an FPGA. A LUT contains an array of memory elements which store programmed values and a selected memory element is driven to the LUT output. Figure 2.4 illustrates a LUT with K inputs and one output where K is a specified parameter of the LUT architecture. A programmed LUT controls a selection system using a number of K-inputs to choose from a set of memory elements to the output. The number of functions a LUT can implement is given by the LUT equation: $|F| = b^{(n \cdot b^k)}$ (from [3]) where n represents the number of outputs and can be discarded since only single output LUTs are considered for simplicity. K is the number of LUT inputs and b represents the number of logic levels (e.g.

2 for binary logic). Using the aforementioned equation, the number of different functions that can be implemented by a 4-input BLUT is $|F| = 2^{(1 \cdot 2^4)} = 65,536$, and for a 2-input QLUT (using quaternary logic) is $|F| = 4^{(1 \cdot 4^2)} \approx 4.3 \times 10^9$. A single QLUT (with half the

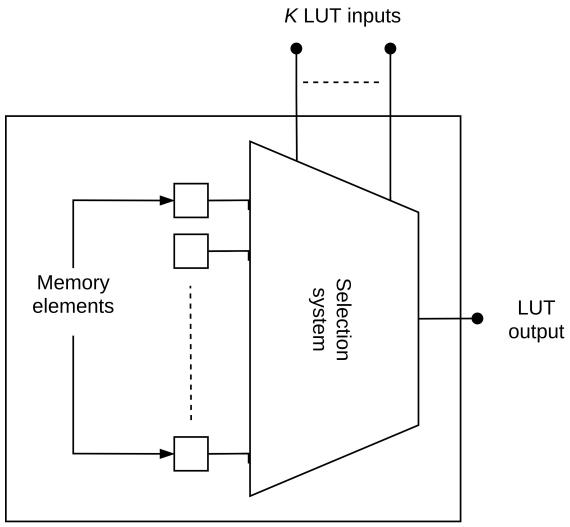

Figure 2.4: Look-Up Table with K inputs and one output.

number of inputs of a binary LUT) is functionally equivalently to two BLUTs because with each of the two QLUT quaternary (0, 1, 2, and 3) inputs represent two binary bits (i.e., 00, 01, 10, and 11), therefore a quaternary logic system holds twice as much information (presuming the same number of logic entries). When comparing FPGA BLEs, this study uses functionally-equivalent 4-input binary LUTs and 2-input QLUTs, both with single outputs.

Because the memory elements in a LUT hold the implementation of logic as discrete voltage values, their impact in the design of a QLUT is crucial. Binary LUTs use standard static random access memories (SRAMs) to store binary values (or bits) that dictate the function of the LUT block (and in turn the function of the FPGA BLE). To fully harness the benefits of quaternary logic, a QLUT requires an alternative circuit design to serve as memory elements. This alternative circuit design should be able to store quaternary logic as discrete voltage values or convert between quaternary logic and binary logic. The proposal of a unique quaternary logic-based memory element circuit for a QLUT is one of the important contributions of this thesis.

Chapter 3

FPGA Architectures From Literature

Quaternary architectures have been proposed in literature: some use current-based values [11,36,37] while others use voltage-based values [1,3], but voltage-based values are more prominent due to significantly reduced power consumption [38]. In current-based circuits, logic levels are defined by multiples of a reference current value. While current-based MVL circuits show better performance compared to corresponding binary circuits, better power performance is achievable in voltage-based MVL circuits (compared to binary equivalent circuits) that use multiples of a reference voltage (V_{DD}). The use of smaller voltage swings in a MVL circuit has a favorable impact on power dissipation compared to current swings, and this makes voltage-mode quaternary architectures more suitable for saving power consumption in a digital implementation.

This chapter studies one binary and two quaternary architectures from literature. Section 3.1 examines a complete 2-input QFPGA architecture and Section 3.2 focuses on the function of a Qdecoder circuit used in a QLUT architecture. The binary architecture used for comparison with the quaternary architectures is briefly detailed in Section 3.3. While this chapter discusses the relevant aspects of these architectures, Chapter 5 discusses the simulation of complete binary and quaternary BLEs required for performance analysis.

3.1 QFPGA Architecture from Literature

Of all literature on QFPGA architectures, a recent study [1] provides the most complete QFPGA BLE description. This architecture will be used as a comparison against the proposed architecture.

[1] implemented QLUTs using transistors with one of several threshold voltages. The study presumed the threshold voltage of transistors can be controlled by applying a voltage to the body terminal of transistors in forward body bias and reverse body bias configurations. Pairs of transistors (in a CMOS inverter configuration) with modified threshold voltages, called down literal converters (DLCs), function as the basic elements for translating between binary and quaternary logic. Three DLCs that are each able to categorize varying groups of quaternary logic levels are used. Table 3.1 shows the output values of the three DLCs where DLC0 identifies level {0} from levels {1,2,3}, DLC1 recognizes between two quaternary levels {0,1} and {2,3}, while DLC2 identifies level {3} from {0,1,2}.

Table 3.1: Identification of quaternary level groups with DLCs

Quaternary input	**DLC0**	**DLC1**	**DLC2**
$\mathbf{0_q}$	3_q	3_q	3_q
$\mathbf{1_q}$	0_q	3_q	3_q
$\mathbf{2_q}$	0_q	0_q	3_q
$\mathbf{3_q}$	0_q	0_q	0_q

Combinations of the DLCs created a quaternary repeater circuit (QRC) (analogous to a binary buffer), a quaternary multiplexer (QMUX), and a quaternary flip-flop (QFF) made with QMUXs, all of which rely heavily on a diverse set of transistors with individual threshold voltages and multiple voltage rails. Sections 3.1.1 and 3.1.2 briefly details the nuances of some components of this architecture.

3.1.1 Quaternary Multiplexer (QMUX)

The QMUX is made with the three DLCs expressed in Table 3.1 and inverters in the configuration as shown in Figure 3.1. Separate VDD and VSS supplies for the inverter-like DLCs are depicted. Four quaternary inputs $IN0$, $IN1$, $IN2$, and $IN3$ representing four quaternary inputs are connected to the output by transmission gates. A quaternary *Select* input supplies the three DLCs while the transmission gates are controlled by voltages from the output of the DLCs and inverters. Depending on the voltage value of the *Select* input, a

corresponding input from $IN0$, $IN1$, $IN2$, and $IN3$ is conducted to the output by enabling the relevant transmission gate (while the rest are disabled).

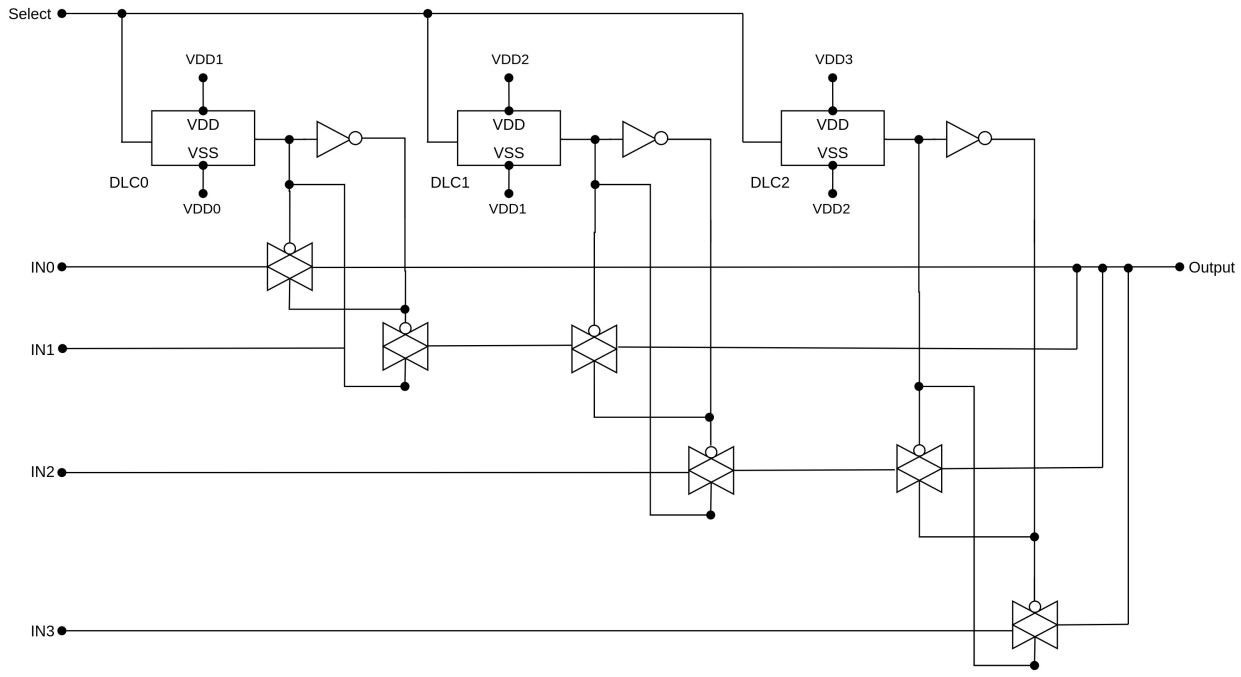

Figure 3.1: QMUX circuit from [1]

3.1.2 Binary-to-quaternary (B2Q) Converter

Four pairs of transistors and two inverters make up the B2Q converter. Figure 3.2 shows the B2Q circuit with two binary inputs $S0$ and $S1$ and respective inverted values $\overline{S0}$ and $\overline{S1}$ (from inverters) that control the gates of the transistors. Four quaternary voltages inputs $VDD0$, $VDD1$, $VDD2$, and $VDD3$ serve as supplies for the four transistor pairs. Depending on the combination of inputs $S0$ and $S1$, a transistor pair is activated when transmitting a corresponding quaternary voltage input to the *Output*.

3.1.3 Quaternary Repeater Circuit (QRC)

The QRC was made from six pairs of different DLCs connected as inverters to achieve a buffer-like operation for quaternary voltage inputs. The input of the QRC supplies three

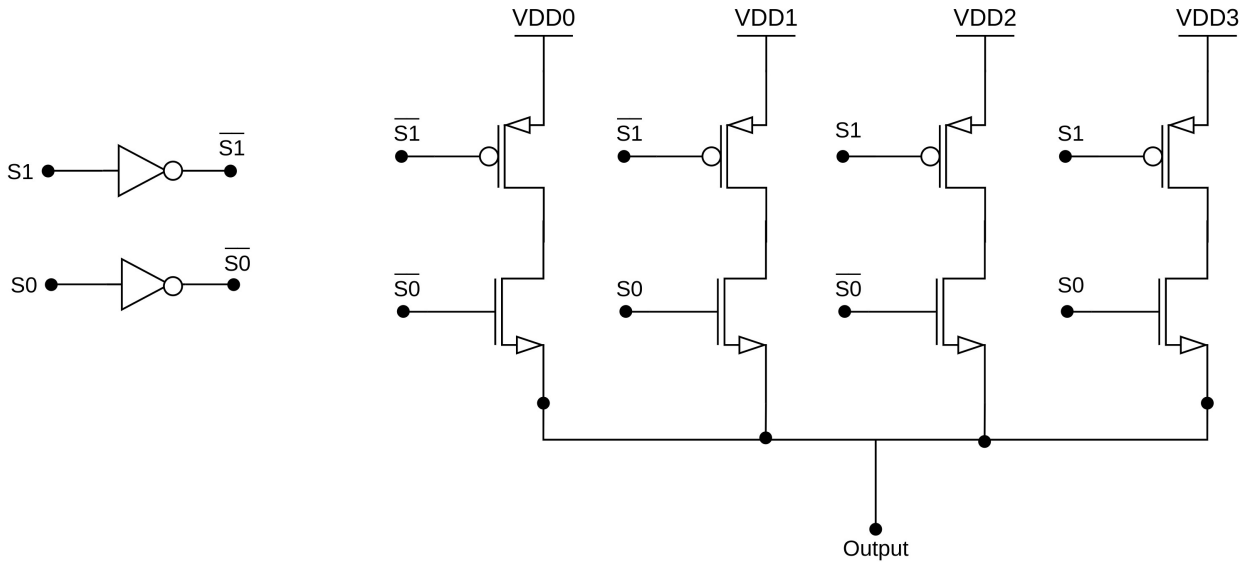

Figure 3.2: B2Q Converter circuit from [1]

inner DLC inverters whose outputs separately feed three outer DLC inverters. One of the outer DLC inverters supplies the QRC output and has its inverter drain and source voltages fed from outputs of the other two outer DLC inverters. These configuration allows the nodes of the QRC circuit modify the drain and source voltages of the outer DLC inverter to produce a QRC output similar to the input.

3.1.4 Quaternary Flip Flop (QFF)

The QFF is made of two parts connected in a master and slave configuration. In the master circuit, a QMUX is connected to a QRC which acts as a buffer. For the input master QMUX; two of its inputs are connected as feedback from its QRC, one input is grounded, and the last input acts as the active input. The slave circuit is similar to the master's; two inputs of its QMUX are connected as as feedback from its QRC, one input is ground and the last input is connected to the master QMUX's output. The QMUXs for the master and slave circuits have additional clock and reset inputs but the clock input of the slave QMUX is an inversion of the master QMUX.

Figure 3.3 shows the complete 2-input QFPGA BLE with two QLUT inputs: $Input1$ and $Input2$, a clock input (Clk) for the QFF, a *Select* input for the quaternary-based selector. [1] used binary SRAMs as LUT memory elements (i.e., two binary SRAMs per quaternary memory element) which were converted into quaternary voltage values using binary-to-quaternary (B2Q) converters. QLUT entries were selected with two layers of 4-to-1 QMUXs: this created a 16-entry ($0 \rightarrow 15$) QLUT selected with the two quaternary inputs.

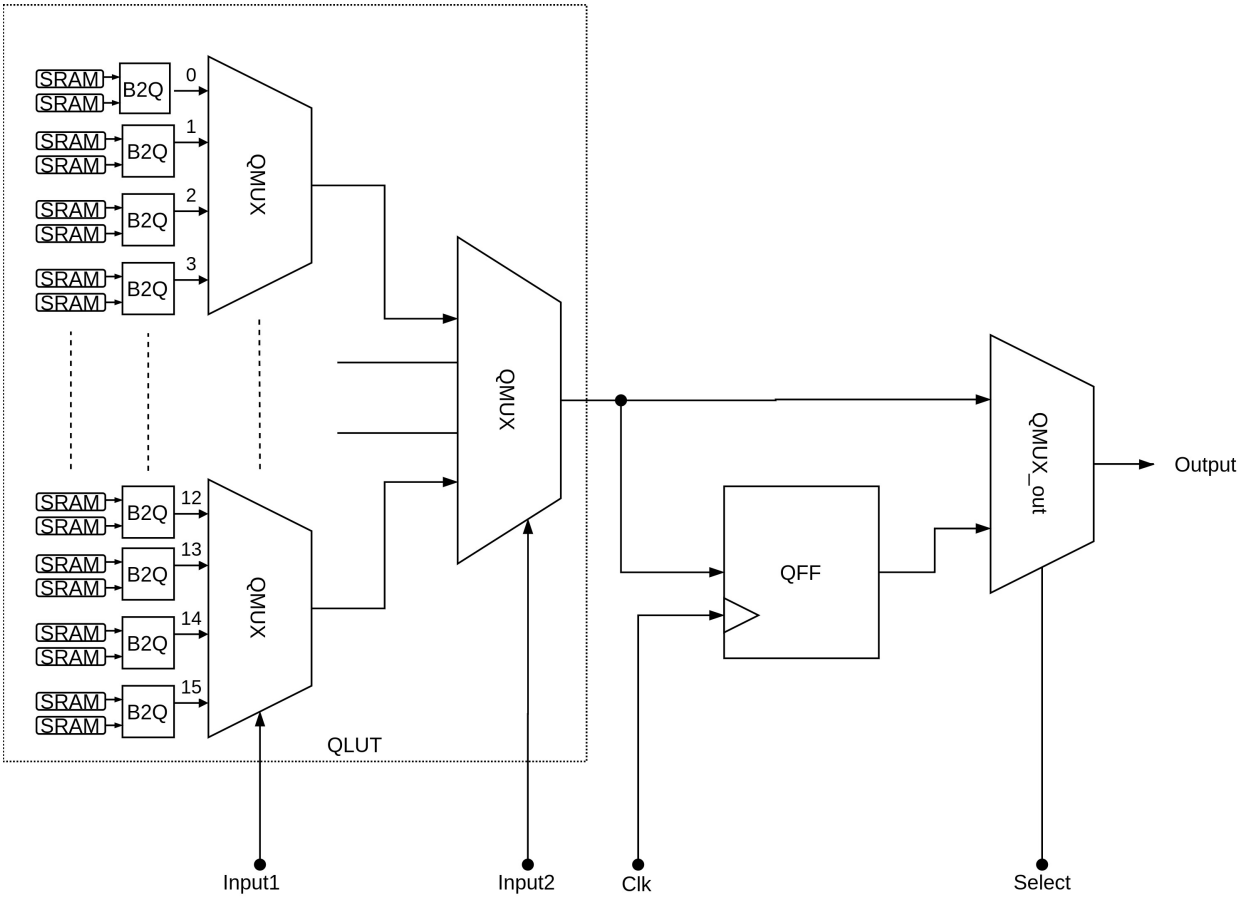

Figure 3.3: Overview of 2-input QFPGA architecture from [1]

To the authors' knowledge, [1] is the most recent full QFPGA BLE, but its conclusion on the BLE outperforming binary equivalent is questionable given the technology requirements to fabricate it. These requirements include body biasing connections which are incompatible with current manufacture techniques [39]. Additionally, the multiple voltage rails needed

to distribute four discrete voltage values in components such as those described in Sections 3.1.1 to 3.1.4 may be problematic. Whether the claims of the original study hold true with custom layouts and accurate SPICE simulations will be found in Chapter 6.

3.2 Practical Quaternary Multiplexer

[3] took advantage of voltage-adjusted comparators to create a transmission-gate multiplexer with quaternary selection signals as a QLUT. These comparators (a unique contribution of [3]) were used to create a Qdecoder circuit. Binary signals from a Qdecoder output select appropriate transmission gates which allow logic values from a quaternary memory to propagate to the LUT output. Unlike the method from [1], this architecture does not consider the memory element of the QLUT, nor does it propose a method of capturing quaternary values in an FPGA BLE. For these reasons, this study does not implement this architecture for comparison but will reuse the Qdecoder structure for the proposed QFPGA BLE architecture.

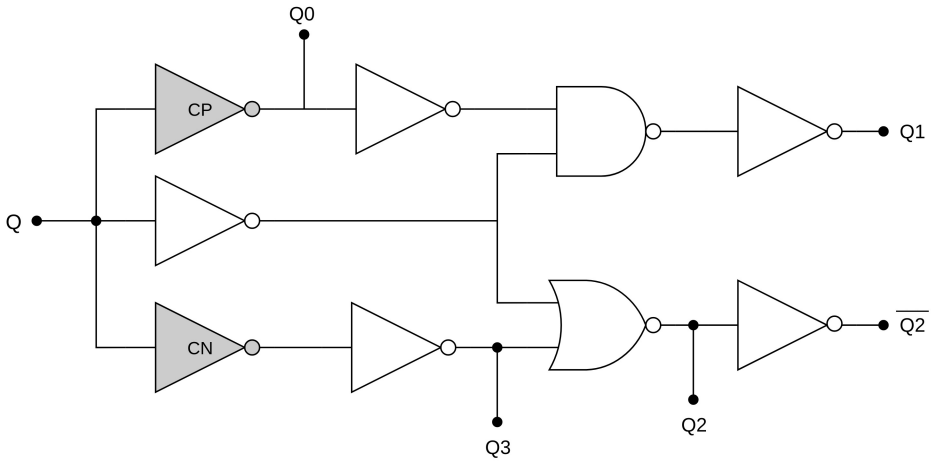

Figure 3.4: QDecoder logic structure (from [3]). Unlike typical inverters, comparators CP and CN do not invert their binary output until the required quaternary input voltage is observed.

It is noteworthy that the Qdecoder and multiplexer structure from [3] avoids manufacturability challenges observed in [1] since does not use individual threshold voltage manipulation techniques. The Qdecoder structure is made of standard NAND gates, NOR gates, and

Table 3.2: Qdecoder Function

Q	**Q_0**	**Q_1**	**Q_2**	**Q_3**
0_q	1_b	0_b	0_b	0_b
1_q	0_b	1_b	0_b	0_b
2_q	0_b	0_b	1_b	0_b
3_q	0_b	0_b	0_b	1_b

inverter-like comparators, as is illustrated in Figure 3.4. An inverter compares the voltage applied at its input with its internal threshold voltages, and it outputs a binary value that represents an inversion of the applied input voltage. Applying this principle, the function of a comparator is like an inverter, except the input voltage which triggers the output is shifted by inserting suitable diode transistors, as shown in Figure 3.5(a) (for CP) and (b) (for CN). Using diode transistors avoids large capacitance cost that would be incurred if transistors with unbalanced widths are used to achieve the same effect [40]. In these circuits, the reference voltage is shifted to 1/6 V_{DD} or 5/6 V_{DD} (see Figure 3.5(c)). Using these CP and CN comparators, the function from Table 3.2 is made with the circuit in Figure 3.4 without the penalty of individual transistor threshold modification. Table 3.2 shows how the Qdecoder translates quaternary input logic 0_q, 1_q, 2_q, & 3_q to binary output logic values Q_0, Q_1, Q_2, Q_3.

The Qdecoder is directly adapted from [3] for the proposed architecture: this is not an original contribution of this thesis's architecture, but its use in the proposed QFPGA is substantial enough to warrant a detailed explanation of its function.

3.3 Binary FPGA Architecture

A 4-input binary FPGA made with a LUT structure from [3] as shown in Figure 3.6, is used to compare with the quaternary architectures. A binary FPGA can consist of any selection structure capable of selecting from multiple memory entries. This selection system can be decoder-based or transmission gate-based but a transmission gate selection system is

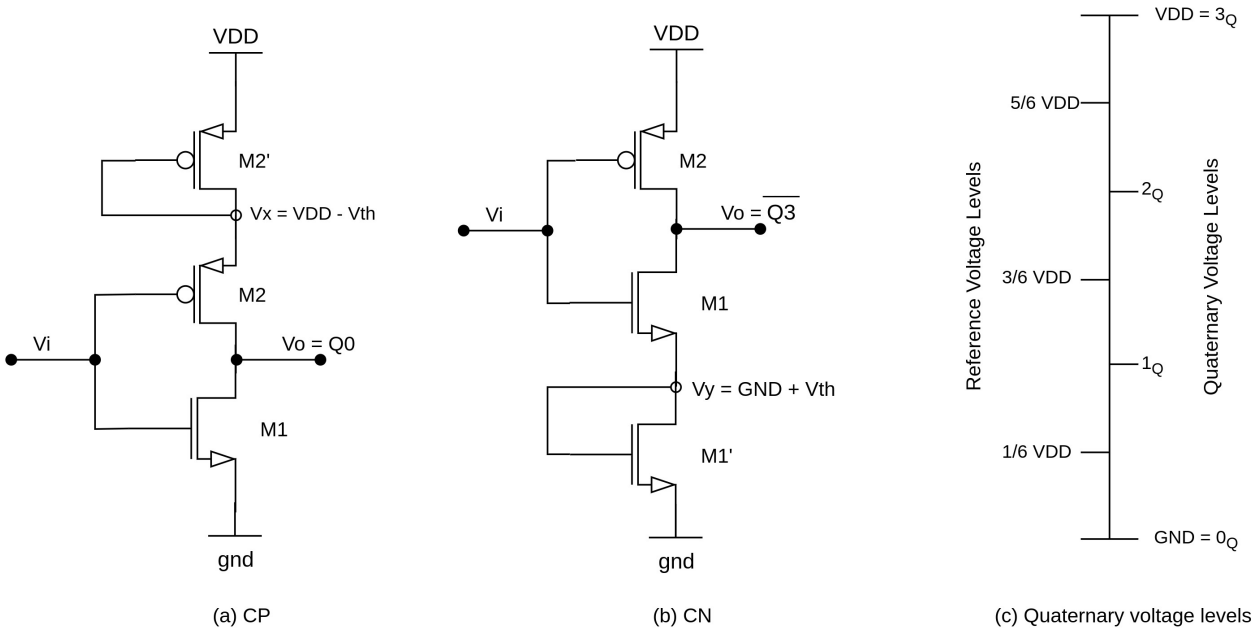

Figure 3.5: The quaternary comparators CN and CN (from [3]) function like inverters, except an additional diode shifts the triggering voltage of the inverters.

preferred for its efficiency over a decoder-based system. For the selected binary implementation, a tree of transmission gates serves as the LUT selection system while SRAMs are the memory elements. The four-stage selection system is controlled by the four LUT inputs ($X0$, $X1$, $X2$, $X3$) and respective inversions. A single SRAM memory is selected and transmitted over transmission gates to the output.

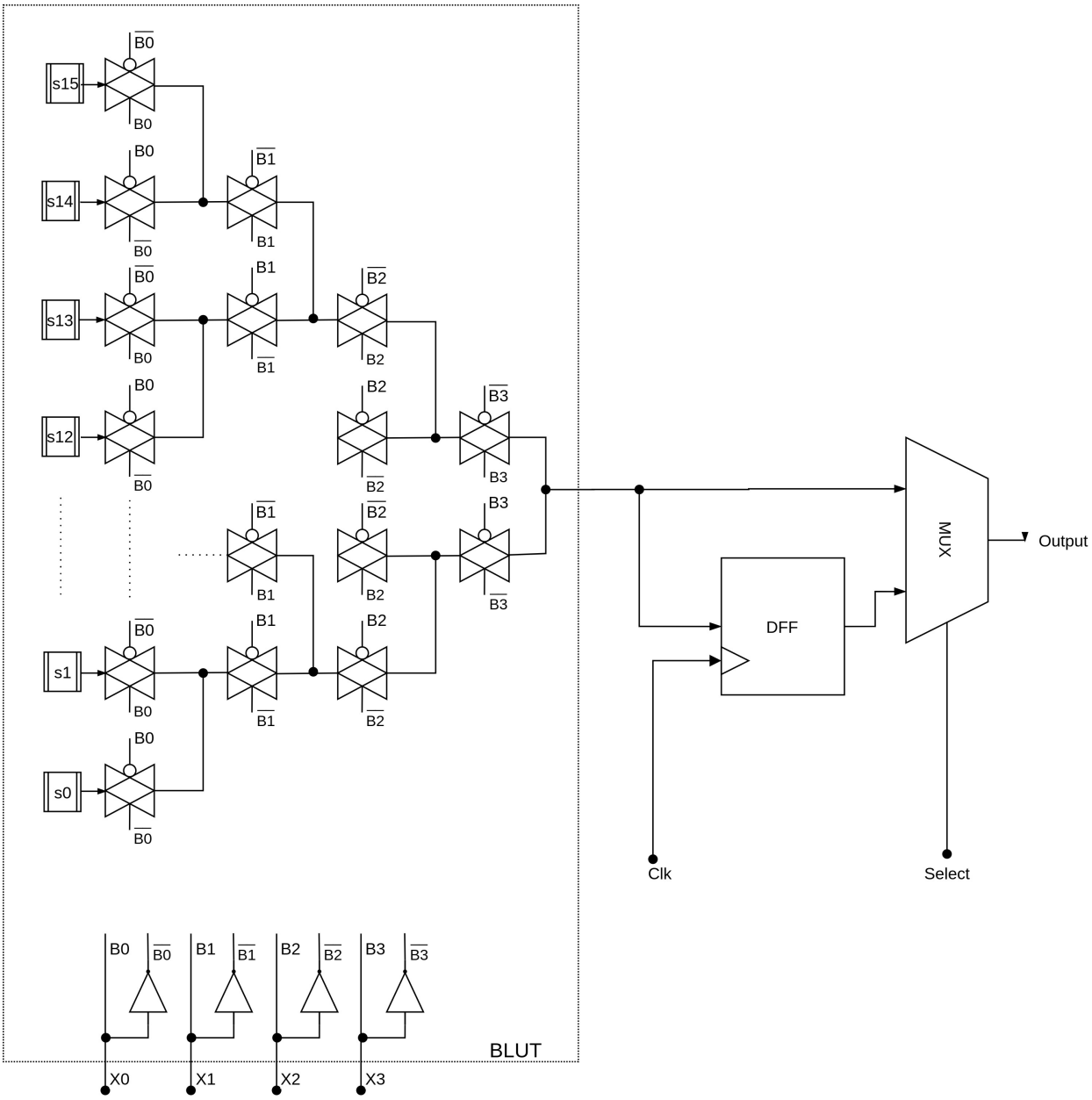

Figure 3.6: Overview of the 4-input binary FPGA architecture used for this study.

Chapter 4

A New Quaternary FPGA Cell With Floating-gate Memory Elements

This chapter covers the design of components employed in the proposed QFPGA architecture. Elements which make up the QLUT structure are detailed, including circuit descriptions of the multi-level memory element, followed by miscellaneous parts which make up the rest of the QFPGA BLE.

4.1 Floating-gate Transistors

The QLUT of the proposed QFPGA BLE uses floating-gate (FG) transistors to construct a MVL memory. The proposed QLUT will use FG transistors to store quaternary values by programming a floating gate with a desired charge. Simulation of a floating gate is covered in Chapter 5 while power and delay performance of implementing a quaternary values using a voltage divider will be explored in Chapter 6.

First presented in 1967 by Kahng and Sze [41], FG transistors have been used for various non-volatile memories in digital domain applications such as EPROMs, EEPROMS, flash memories, as well as in programmable and adaptive analog circuits [42, 43]. A FG transistor is a standard MOS transistors with an extra electrically-isolated floating gate between the transistor gate and channel. Unlike regular transistors, the floating gate is made of polysilicon surrounded by a SiO_2 insulator which enables long-term charge storage. The cross section of a FG transistor (Figure 4.1) has the floating gate and a control (regular) gate separated by a gate oxide layer which isolates charge (electrons) forced into it. The charges trapped in the gate oxide layer cannot leave without the application of an external force or potential. The amount of trapped charges modify the FG transistor's threshold

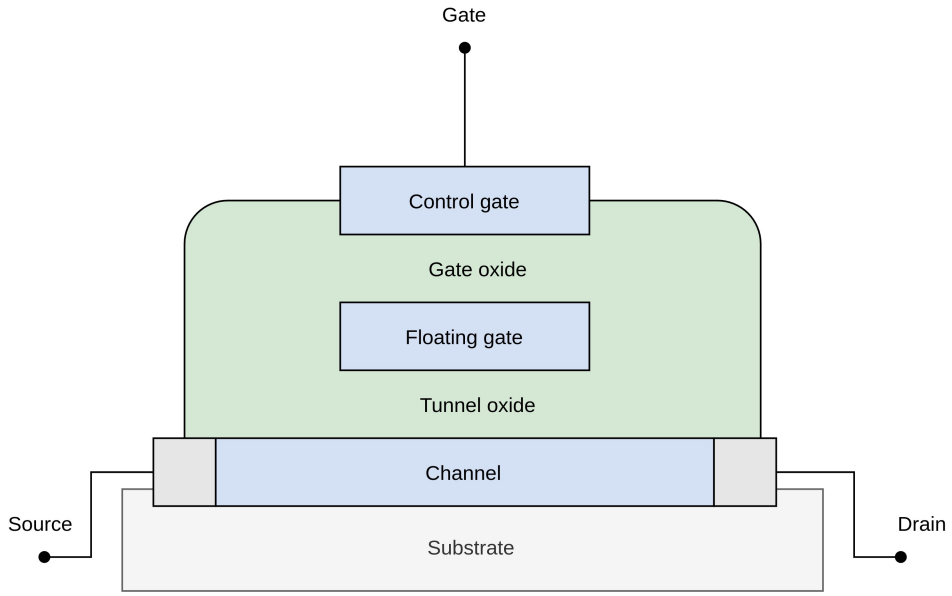

Figure 4.1: Cross section of a floating-gate transistor.

voltage. A FG transistor's properties can therefore be harnessed using its charge-storing capabilities and its I-V relationship to design and implement digital and analog circuits.

The effect of charges trapped on the floating gate is different when the FG transistor is made from a NMOS versus a PMOS transistor. A NMOS-based FG transistor with charges on its floating gate has an increased threshold voltage and would require more control gate voltage to activate the FG transistor. With a PMOS-based FG transistor, increasing the charge on its floating gate decreases its threshold voltage requiring a more negative voltage to switch on the FG transistor. In essence, programming a floating-gate transistor involves adding and removing charges on it. When a NMOS-based or PMOS-based FG transistor has a maximum charge trapped in it, it is said to be fully programmed. In this state, no amount of control gate voltage will switch on the FG transistor.

4.1.1 Programming Floating-gate Transistors

Charge addition/removal operations determine the specific I-V characteristics of a floating-gate transistor. Charges are added into a floating-gate transistor using hot-carrier injection techniques. Hot-carrier injection involves energizing carriers in the silicon with enough energy

to enter the gate oxide's conduction region (Figure 4.1) typically using UV light exposure or drain avalanche hot-carrier (DAHC) method [44]. Using the principle of hot-carrier injection, a high gate and drain potential relative to the source draws minority carriers from the source towards the drain. However, energy absorbed by the carriers is so high that they overcome the tunnel oxide barrier and get trapped in the floating gate's conduction band instead. The DAHC process is slightly different: high energy causes minority carriers to move from source to drain which in turn impact minority carriers in the drain (like an avalanche effect) and with suitable high field conditions enter into the floating gate.

Removing electrons from a floating-gate transistor using electron tunneling requires a MOS capacitor, as this is more reliable than a polysilicon-polysilicon capacitor. A tunneling junction exists between the floating gate and the substrate. Electrons can tunnel through this junction when the barrier is thin enough. Additional optimal tunneling conditions are created by an electric field between the tunnel voltage ($V\ tun$) and floating gate voltage ($V\ fg$) (due to voltage difference) [45]. Equation 4.1 represents the classical electron tunneling current ($Itun$) model where t_{ox} is the gate oxide thickness ε_0 is a dive parameter of around 25.6 /nm [46]. Both input voltage and charge on the floating gate determine the tunneling rate.

$$Itun = I_o \cdot \exp\left(-\frac{t_{ox} \cdot \varepsilon_0}{Vtun - Vfg}\right) \tag{4.1}$$

4.1.2 Floating-gate Transistor Modelling

The isolated floating gate of a floating-gate transistor poses circuit simulation challenges that compact models (CMs) of floating-gate devices seek to resolve. These CMs, such as [4], attempt to accurately simulate the effects and interactions of floating-gate operations in a circuit. In circuit design, floating gates require a node with a capacitive connection and no direct current (DC) path. [4] resolves this atypical circuit condition by using a floating-gate node (FGn) to both isolate the control gate potential from the other terminals, and

to calculate the floating gate potential. Figure 4.2 shows the cross section of the [4] model applied to a floating-gate transistor. The *Cfg* capacitor isolates the gate node from other nodes. The floating-gate node potential is a sum of potentials from all contributing nodes of the floating-gate transistor. With this arrangement from Figure 4.2, charge adding/removing operations can be modelled with suitable voltage generators between the transistor terminals.

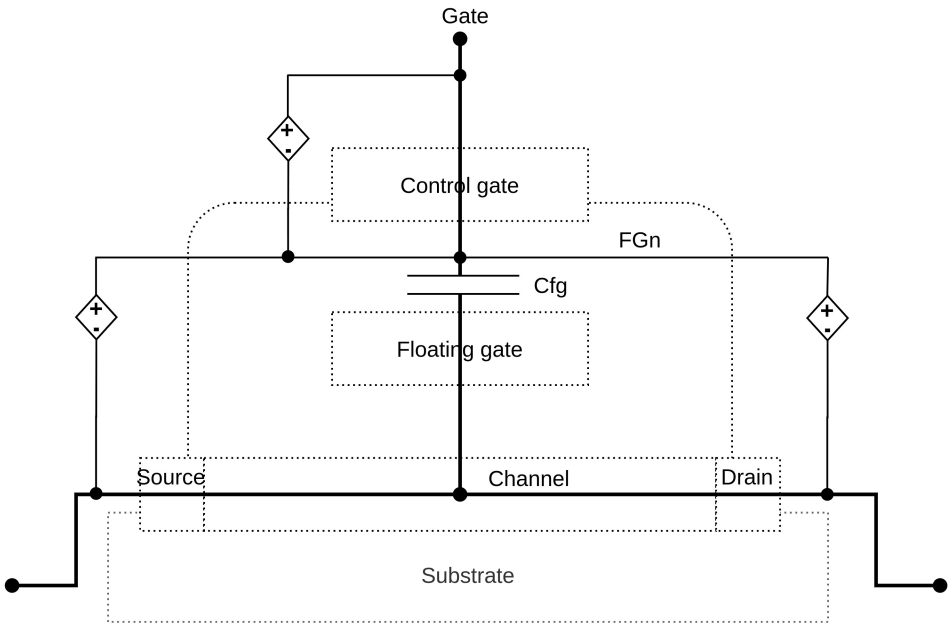

Figure 4.2: Basic schematic of a CM model [4] with a Floating-gate transistor.

A SPICE-compatible model [5] is selected among other models [4, 47, 48] to model the function of the floating-gate transistor because it considers floating-gate capacitive coupling and is suitable for DC and transient simulations. The model is adaptable to various hot-carrier injection and electron tunneling models for programming a floating-gate transistor. For this thesis, the model from [5] is adapted and using VCVSs makes it easy to program the floating-node voltage. As depicted in Figure 4.3, the transistor's "effective" gate voltage (the "effective-node" voltage) is a function of the applied control-gate voltage ("control gate-node") and the voltage programmed into the floating gate ("floating gate-node"). In effect, the charge programmed in the floating gate-node changes how much control-gate voltage is needed to turn on the floating-gate transistor (i.e. it alters the transistor's threshold).

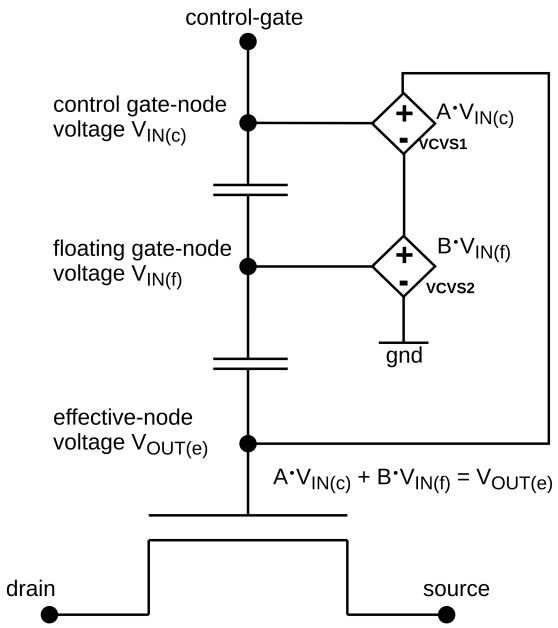

Figure 4.3: An adapted SPICE-compatible model (from [5]) of a floating gate modifies gate voltage of a transistor with programmed voltages.

Charge addition on the floating gate is modeled by voltage-controlled voltage source (VCVS) elements. The floating-gate transistor control is the input $V_{IN(c)}$ which controls VCVS1, while the floating gate-node voltage is the input $V_{IN(f)}$ which controls VCVS2. The output voltage for VCVS1 is $A \cdot V_{IN(c)}$ and for VCVS2 is $B \cdot V_{IN(f)}$ where A and B are constant values representing programmed charges. The effective-node voltage $V_{OUT(e)}$ is a summation of both VCVS outputs, as illustrated in Figure 4.3.

4.2 MVL Memory Element and Drain Control Circuit

The MVL memory element (the lower part) and its control circuit (the upper part) are depicted in Figure 4.4. The MVL memory element is made of two floating-gate transistors connected together as a voltage division circuit: the right-hand floating-gate transistor is made with an NMOS base (FG_N), and the other with a PMOS base (FG_P). The gate inputs for both floating-gate transistors, $V_{IN(c1)}$ and $V_{IN(c2)}$, require a higher gate voltage than V_{DD} for programming the floating-gate transistor. $V_{IN(f1)}$ and $V_{IN(f2)}$ represent the respective

programmed floating-gate floating node voltages. To regulate the drain voltages for the floating-gate transistors, a control circuit consisting of a PMOS transistor for FG_P and NMOS transistor for FG_N is used. $V_{d,1}$ is V_{SS} and $V_{d,2}$ is V_{DD} during normal device operation. The switching of the control transistors is handled by control signal c and its inverse $\overline{c}$. When the MVL memory element is provided with $V_{IN(c1)}$, $V_{IN(c2)}$ voltages, programmed with specific $V_{IN(f1)}$ and $V_{IN(f2)}$ voltages, and the drain control transistors are both switched on to supply $V_{d,1}$ and $V_{d,2}$, the voltage division circuit gives a quaternary voltage as V_{out}.

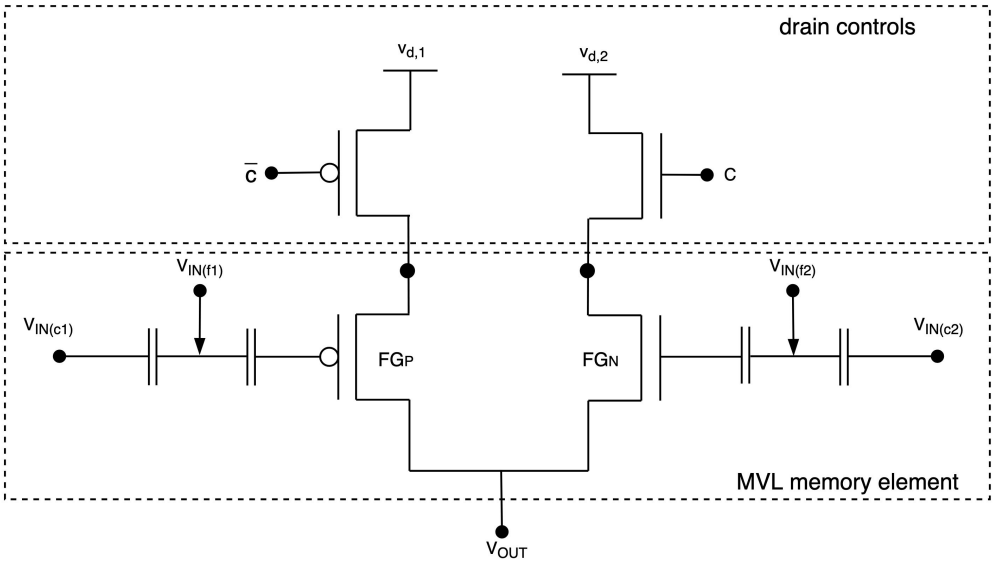

Figure 4.4: MVL memory element made of two connected floating gates (from Figure 4.3) with gate controls.

4.3 Proposed QFPGA Architecture

The proposed QLUT consists of 16 MVL memory elements, their respective gate controls, a selection structure (made with 3-input NAND gates and inverters), and two Qdecoders (from Section 3.2). To combine memory elements into a QLUT, control signals $C0$ to $C15$ and respective inversions $\overline{C0}$ to $\overline{C15}$ activate memory elements $M0$ to $M15$. 3-input NAND gates and inverters, as illustrated in Figure 4.5, supply these control signals and inversions based on the output of the two Qdecoders. The third input for the NAND gates

is an enable signal EN which can be used for auxiliary purposes (e.g., programming a BLE). When EN and the two inputs to a NAND gate are active, a single memory element and its output transmission gate are activated and connected to the QLUT output, W. Transmission gates at the output of each memory element disconnect deselected memories and reduce output capacitance. The use of a transmission tree structure from [3] was explored (and controlled with the existing Qdecoder outputs), but performance (delay and power) compared to using single transmission gates was worse.

An additional transmission-gate based 2-to-1 multiplexer and a QFF (made with a Qdecoder-based latches and 2-to-1 multiplexer) make up the proposed QFPGA BLE. The architecture of the QFF in the proposed BLE differs from the compared architecture [1] in that the multiplexers are two-to-one transmission-gate multiplexers as opposed to the original four-to-one multiplexers. However, replacing the original multiplexers did not significantly impact delay, power, or area results.

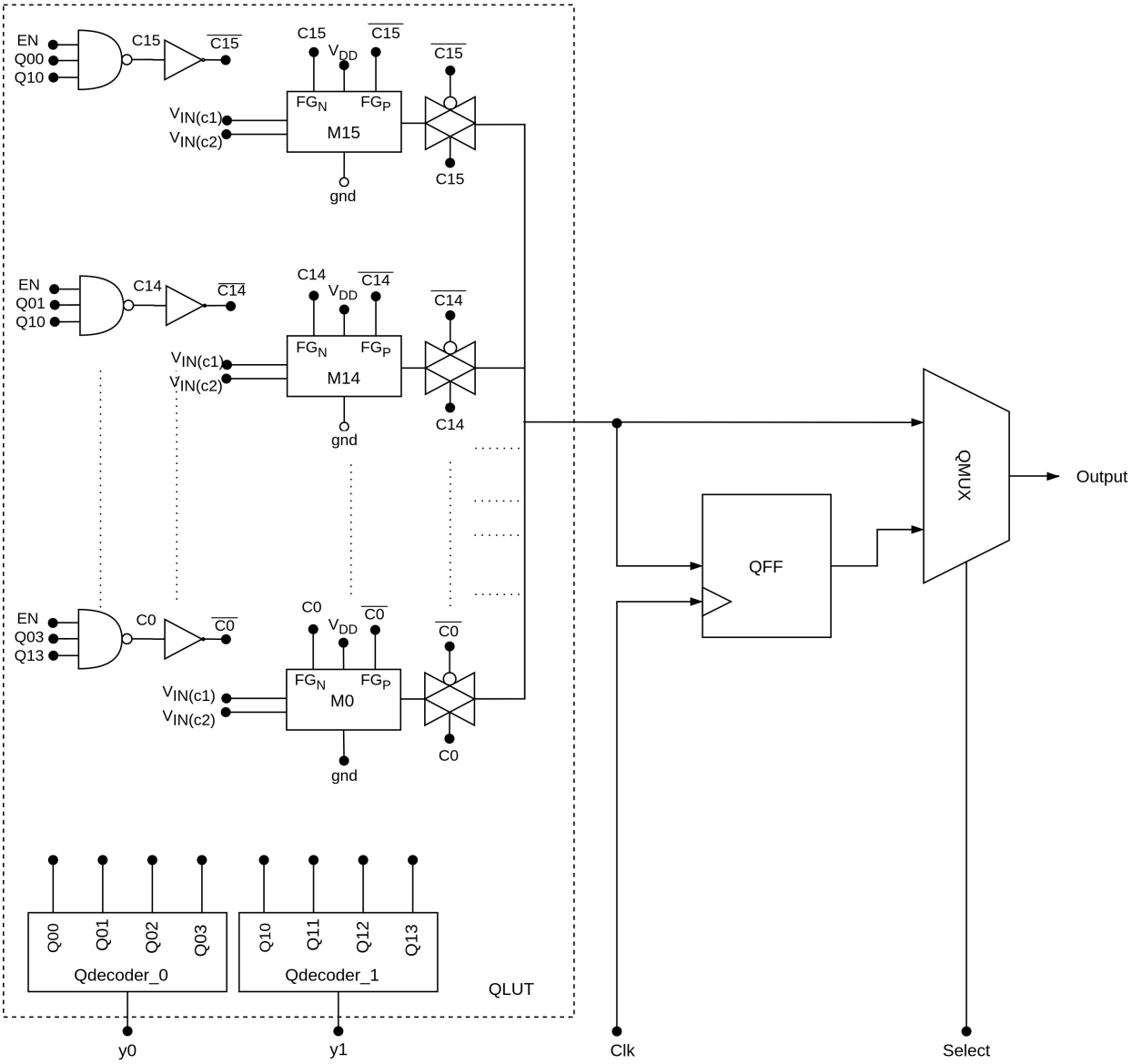

Figure 4.5: Implementation of Proposed QFPGA.

Chapter 5

Simulation And Evaluation of Architectures

5.1 Simulation and Design Process Using Cadence Tools

Simulation in this thesis refers to circuit-level simulations using SPICE simulators to model and capture the behaviour of a circuit described in terms of transistors, wires, resistors, capacitors, and their respective connections to one another. Schematic capture provides a means of placing circuit components in hierarchical arrangements and defining connection nodes among the components. This circuit models include wire resistance, changing voltage potentials, geometric properties of transistor models, etc. The aim of this circuit-level simulation is to provide detailed analog waveforms that accurately describes how the designed circuit would operate in the real world. In a circuit-level simulation, the first stage called *node-extraction* performs a static analysis of the circuit description. The next stage combines the information generated in the *node-extraction* stage with *device models* that mathematically describe the behaviour of circuit devices. A circuit simulator then solves a system of differential linear equations, or non-linear first-order algebraic equations, derived from all the circuit information to accurately model the circuit's operation. The complete process is complex, detailed, intensive, and utilizes several reliable numerical integration methods [49], and therefore circuit-level simulations can generate very accurate results for smaller designs such as those in this thesis, but these simulations can be slow with larger designs due to computation intensity.

The Cadence Virtuoso analog design environment (ADE) is a circuit simulator designed for use on Unix operating systems and contains the necessary tools for accurate simulation of complex circuits. It is built on a design framework environment that can be integrated with third-party tools and supports features such detailed annotation, interactive simulation, and

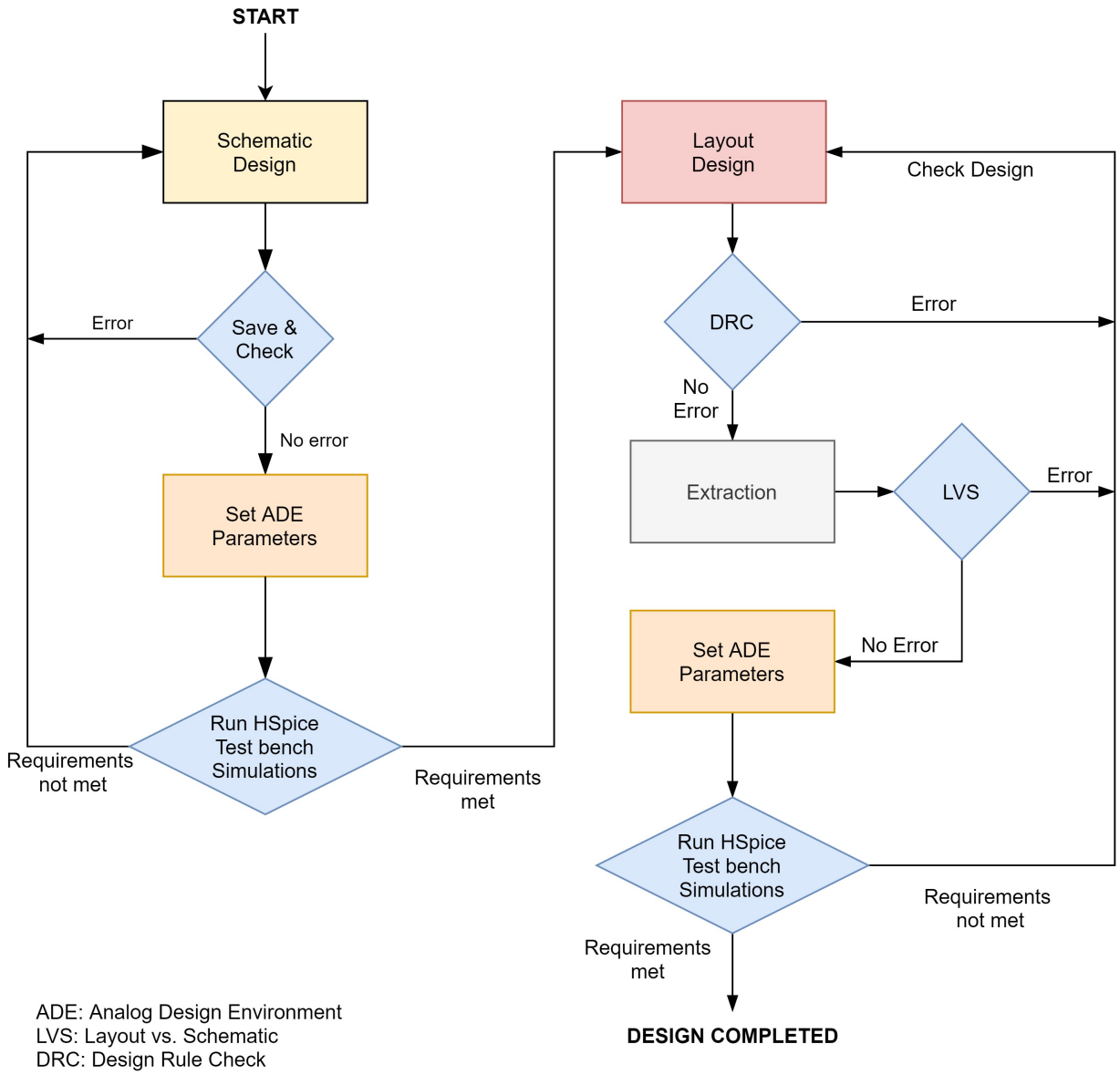

Figure 5.1: Overview of the Design Process.

simple to advanced output analysis. These features make it a good selection for interactive schematic editing and performing parametric analysis under several circuit conditions. With Virtuoso ADE's integration with third party tools, it can select HSpice as a simulator: this important feature allow the use of the FreePDK15 library HSpice transistors models.

FreePDK15 is an open-source predictive process design kit (PDK) library for the 15nm technology node [17]. The library specifies a V_{DD} supply voltage of 0.8 V. Binary logic levels are represented by 0 V and V_{DD} while $0 \cdot V_{DD}$, $1/3 \cdot V_{DD}$, $2/3 \cdot V_{DD}$, and V_{DD} correspond to

0, 1, 2, and 3 quaternary logic representations, respectively. The FreePDK15 library models FinFET technology, but this can still be used to model quaternary logic since floating gates can be implemented using FinFETs [5]. FinFETs operate faster with reduced leakage current [50] compared to planar MOSFET transistors, and resolve MOSFET transistor scaling challenges [51, 52]. Cadence Virtuoso Calibre DRC, LVS tools were utilized for creating layouts for the architectures.

The process of creating and generating circuit schematics and layout results is shown by Figure 5.1, which summarizes the steps that permit easy identification of design bottlenecks and troubleshooting.

5.1.1 Simulating the Floating-gate Transistor Model

The floating-gate transistor model from Figure 4.3 is simulated using the following test bench parameters: control gate voltages of 0 V and 3 V, and floating-gate node voltages between 0 V to 2 V. Simulation results of the parametric dc analysis for a floating-gate transistor (PMOS in this case) is shown in Figure 5.2 where the y-axis is the drain current and x-axis is control gate voltage. The drain current remains zero when the floating-gate node voltage is zero (equivalent to a stored charge of zero) regardless of the control gate voltage applied. As the floating-gate node voltage is swept for $V_{IN(f)}$: 0 V to 2 V and control gate voltage swept for $V_{IN(c)}$: 0 V to 3 V, the the drain current increases accordingly to the impact of both voltage sources (through the VCVSs) on the effective-node voltage. This simulation approach is similarly applied to a floating-gate NMOS transistor.

5.1.2 Simulating the MVL Memory Element

The MVL memory element from Section 4.2 is simulated using a test bench with appropriately programmed voltages for the floating nodes ($V_{IN(f1)}$, $V_{IN(f2)}$). The initial condition feature in Cadence Virtuoso ADE allows easy programming of the floating-gate floating nodes using a simulation file with specified node voltages for all 16 MVL elements at once.

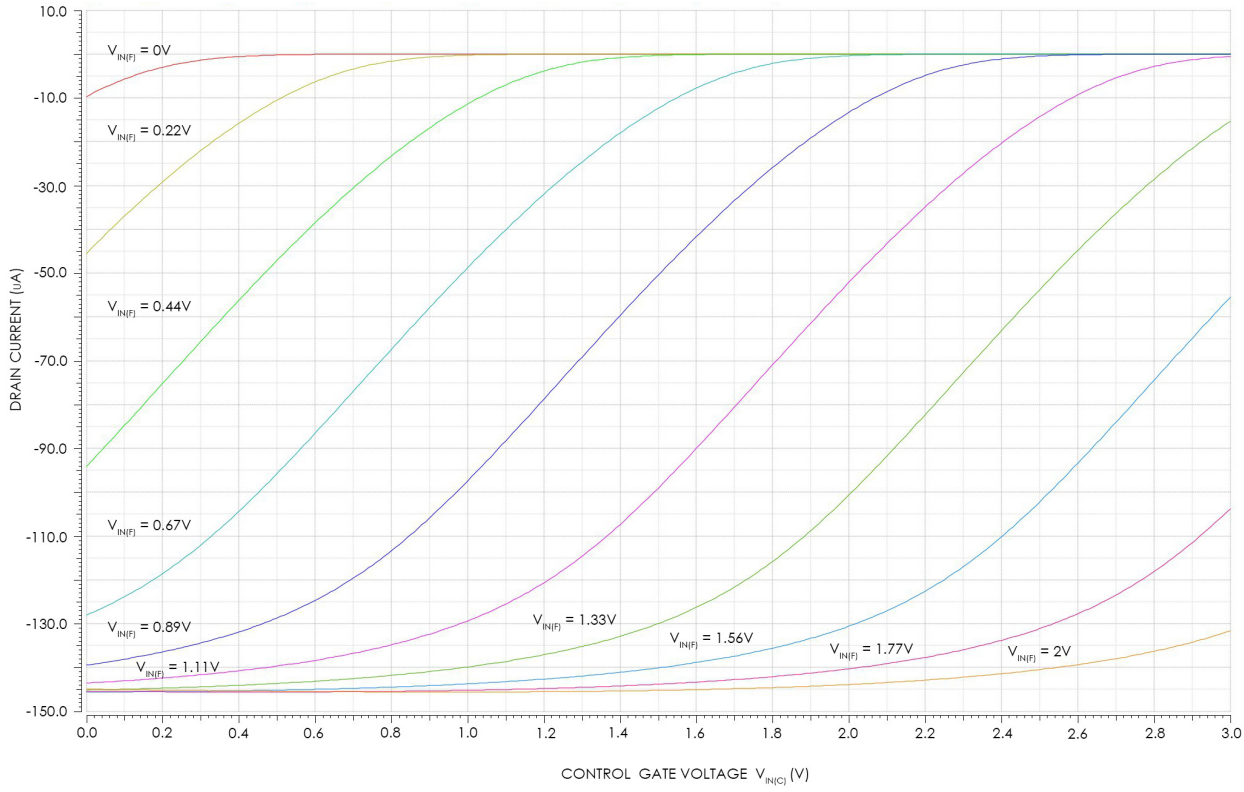

Figure 5.2: Simulation of Floating-gate Transistor Model.

A transient test bench simulation supplies control signals to operate one MVL element per specified transient time. Table 5.1 shows the set of floating-gate node voltages programmed to obtain the required quaternary levels with control node voltages $V_{IN(c1)} = V_{IN(c2)} = 2.5$ V, $V_{d,1} = 0$ V, and $V_{d,1} = 0.8$ V.

Table 5.1: Programmed floating node voltages for quaternary outputs.

$\mathbf{V_{IN(f1)}}$ (V)	$\mathbf{V_{IN(f2)}}$ (V)	**output voltage (V)**
0	0	0
1.2	0.25	0.27
0.87	0	0.53
0.44	1.11	0.8

5.1.3 Simulating Architectures

This thesis simulated the architecture from [1] using an openly available FinFET technology library [17], but the author is convinced body biasing individual transistors is impractical because multiple body biasing voltage rails would be required in fabrication [53], therefore threshold voltages are modified by changing individual transistor work-functions. This technique can model either individual body biasing (which is optimistic given the required body bias control routing) or individual transistor doping [54, 55].

Functional verification of the binary BLE and QFPGA BLE from [1] performed as expected. The test bench for the binary BLE selected from the 16 SRAM memories with random binary voltages programmed used the initial condition feature of the simulator. For the [1] QFPGA test bench, a memory unit (SRAM and B2Q) is selected per transient simulation period to obtain a quaternary output pattern similar to that in Figure 5.3. This output pattern is the same for the proposed QFPGA and allows the observation of all possible quaternary transition sets.

Simulation waveforms for the proposed QFPGA BLE is represented in Figure 5.3: the enable EN input to the 3-input NAND gates (Figure 4.5) is active throughout the simulation (not shown) while the different Q_i inputs (as other inputs) to the 3-input NAND gates activate MVL memories programmed with quaternary logic. The output pattern (OUT signal in Figure 5.3) covers all possible transitions between the quaternary logic levels.

Spikes observed at certain points in the simulation waveform results above can be attributed to the unstable interactions of the mathematical descriptions of circuit components during simulation time steps. Since the interactions attempts to model real conditions, it provides reliable information on the stability of the circuits.

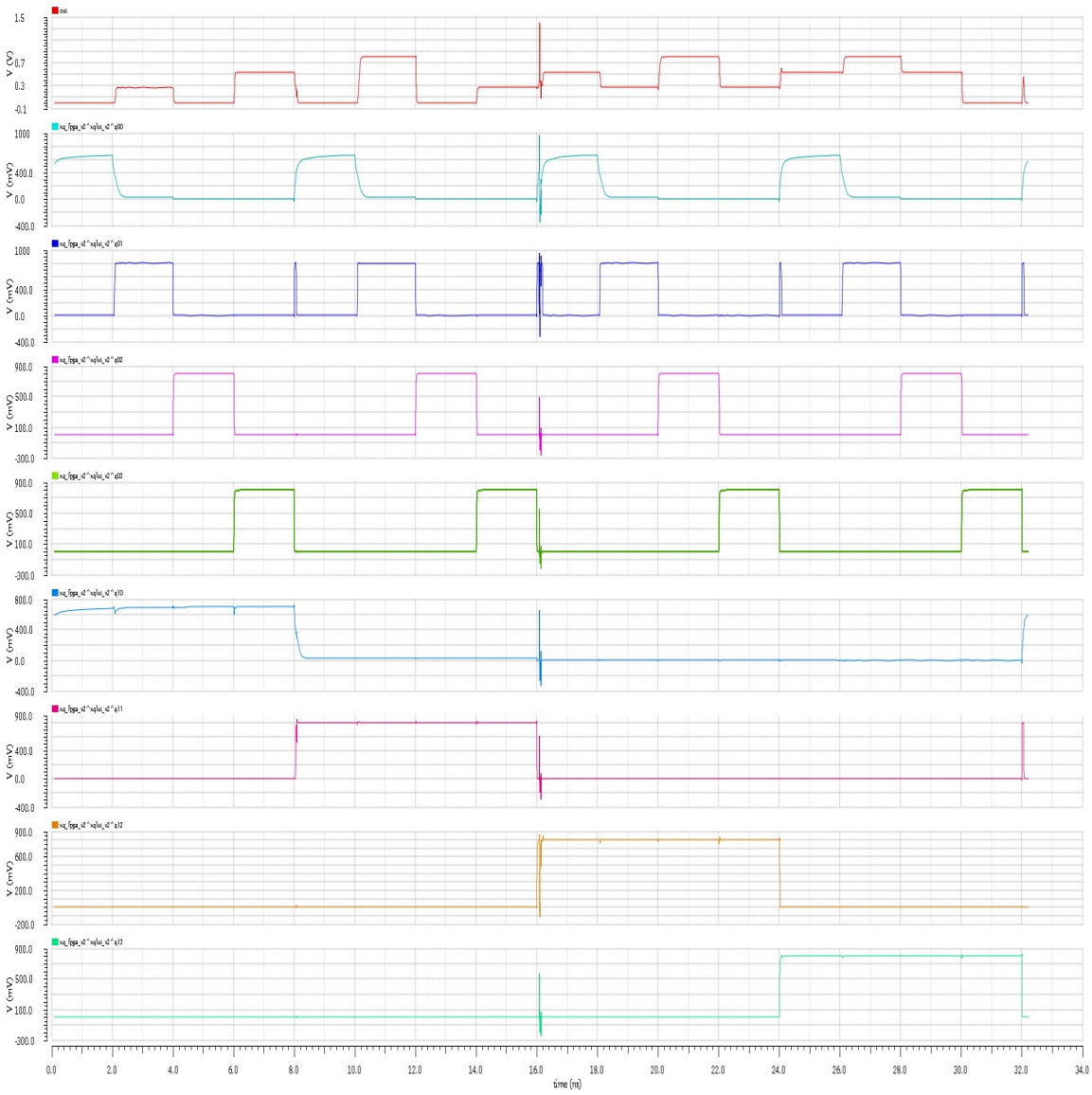

Figure 5.3: Simulation Results of the Proposed QFPGA BLE.

5.2 Evaluation of the Architectures

Test benches for the three architectures observe worst-case timing (from the QLUT inputs through the FPGA BLE multiplexer) and average/maximum circuit power. Appropriate V_{DD}, V_{SS}, programmed memory values, and other inputs select a single memory entry per clock cycle.

5.2.1 Delay Analysis

Propagation delays between LUT inputs and FPGA BLE outputs (by bypassing the (Q)FF) for all transitions were collected. Delay was measured as the time to reach 90% of the final target voltage.

5.2.2 Power Analysis

Total power consumption is given as energy consumed by a BLE over simulation time, divided by that time given by the relation

$$\left(\int_0^t I \cdot V_{DD} \cdot dt \right) / t \tag{5.1}$$

where I is the current, V_{DD} is the drain voltage, and t is simulation time. To observe the maximum performance of the BLEs, transient simulations are run at the highest clock cycle possible without degrading the outputs to unusable levels.

5.2.3 Power Density Analysis

Power density gives the relation of power consumption and BLE area as an effective measurement parameter and it is calculated as total power consumed over area.

5.2.4 Area Measurements via Layout

Area measurements for both binary and quaternary architectures were extracted by layout measurements. Binary and quaternary BLE layouts used the same technology library and the same binary digital BLEs for a fair comparison. Unlike other studies which perform layout automation, elements of the proposed (especially the novel MVL memories) and previous QFPGA architectures used a custom layout process. All BLEs were manually made to minimize area as much as possible with adjustments to address layout design-rule nuances. Floating-gate transistors which are not distributed as part of FreePDK15 are implemented as

black boxes: based on FinFET floating-gate transistor studies [56], floating-gate transistor widths were modelled as 40% wider than equivalent traditional transistors widths.

Chapter 6

Evaluation Results

6.1 Previous QFPGA Architecture [1] Single BLE Evaluation

Table 6.1 shows the performance of a single binary FPGA BLE and a quaternary FPGA BLE from literature [1], both of which are implemented using the evaluation process detailed in Section 5.

Table 6.1: Single BLE Delay, Power, Area, & Power Density

	Binary FPGA	**[1] QFPGA**	**Proposed QFPGA**	**% Difference of proposed to binary**	**% Difference of proposed to [1] QFPGA**
Delay (ps)	128	129	177	38.28%	37.21%
Power (W)	4.82E-05	9.90E-04	1.50E-04	210.77%	-84.87%
Area (mm^2)	3.49E-05	1.88E-04	8.47E-05	142.49%	-54.89%
Power Density (W/mm^2)	1.38	5.27	1.77	28.16%	-66.47%

The single QFPGA BLE from [1] shows practically identical delay performance compared to the binary BLE, but its power density is four times more the binary BLE. Unlike the original article [1] which modelled design nuances optimistically (e.g., multiple voltage rails, the effect of body biasing, etc.), this study models these nuances more accurately using full-custom HSpice simulations.

6.2 Proposed QFPGA Single BLE Evaluation

The propagation delay, power, and power density measurements for the proposed QFPGA cell are shown in Table 6.1. The binary BLE has better delay performance compared to the proposed QFPGA BLE (38.28% more delay), but this is acceptable given substantially fewer BLEs will be needed to implement a full circuit. The binary BLE also performs better in power density since the proposed QFPGA is a larger BLE (has more logic elements), but this is acceptable given substantially fewer BLEs will be needed to implement a full circuit.

When the area and power density of proposed BLE is compared against the equivalent previous architecture [1], the utility of the proposed architecture is revealed. The area required to implement the previous architecture is significantly larger (2.2 times), and the power density of the previous architecture is substantially larger (3 times).

6.3 Projecting Architectures with Arithmetic Benchmarks

Using analysis done on single BLEs, FPGA benchmark performance can be projected using the comparative QFPGA's synthesized results [1] on configurable logic block (CLB) performance. [1] presumed multiple BLEs connected to a reconfigurable multiplexer structure from [57] was used for sizing input crossbars. This input crossbar's size is based on the number of concatenated 8-to-1 multiplexers required to connect the BLEs (as discussed in Section 2.3.3). This study will presume the proposed and the cooperative QFPGA architecture use the same input crossbar structure of the same size. The configuration of these CLBs is given in Table 6.2, and the size of the CLBs' can be re-calculated. Original size calculations [1] were estimated as minimum width transistor area (MWTA) [58], but this measurement for BLE size is unnecessary given LUT areas were created using custom layouts. The size of LUTs in the CLBs are larger for benchmark evaluation (5-input vs. 4-input for binary and 3-input vs 2-input for quaternary), so layouts were expanded appropriately to calculate LUT areas. The required crossbar for binary and quaternary circuits is also made with a custom layout to minimize circuit area.

Table 6.2: Area calculations for configurable logic blocks

	Binary		**[1] QFPGA**		**Proposed QFPGA**	
Parameter	**Value**	**Area** (mm^2)	**Value**	**Area** (mm^2)	**Value**	**Area** (mm^2)
LUT Input Size	5	5.35E-5	3	1.76E-4	3	1.32E-4
No. of BLEs	10	5.35E-4	10	1.76E-3	10	1.32E-3
Input crossbar	50	1.75E-4	30	1.05E-4	30	1.05E-4
Total	-	7.10E-4	-	1.86E-3	-	1.43E-3

Table 6.3: Area Evaluation of Arithmetic benchmarks

	Binary			[1] QFPGA			Proposed QFPGA				
	Area (m^2)			**Area (m^2)**			**Area (m^2)**			**Difference of proposed to binary** (%)	**Difference of proposed to [1] QFPGA** (%)
Benchmarks	**Routing**	**Logic**	**Total**	**Routing**	**Logic**	**Total**	**Routing**	**Logic**	**Total**		
Adder32	3.6E-09	4.2E-09	7.8E-09	2.27E-09	8.43E-09	1.1E-08	2.3E-09	6.7E-09	8.9E-09	15.1	-16.3
Adder64	1.1E-08	9.5E-09	2.1E-08	5.94E-09	1.90E-08	2.5E-08	5.9E-09	1.5E-08	2.1E-08	0.0	-15.8
Mult32	4.5E-08	1.3E-07	1.7E-07	1.8E-08	1.3E-07	1.5E-07	1.8E-08	1.1E-07	1.2E-07	-27.6	-18.3
Mult64	1.6E-07	5.1E-07	6.7E-07	5.7E-08	4.7E-07	5.3E-07	5.7E-08	3.8E-07	4.3E-07	-35.2	-18.5
Mult128	7.8E-07	1.8E-06	2.6E-06	2.4E-07	1.8E-06	2.0E-06	2.4E-07	1.4E-06	1.6E-06	-35.7	-18.3

Using the projected sizes of 5-input binary and 3-input quaternary CLBs from Table 6.2, the projected performance of arithmetic benchmarks from [1] is given in Table 6.3. The logic area is found by multiplying the number of required CLBs (given in [1]) by the respective CLB sizes. The routing area from the original study is kept for this study (with the proposed and comparative QFPGA pessimistically having the same routing area), except MWTA units are translated to physical silicon area.

These projected benchmarks show the proposed QFPGA architecture substantially reduces the area needed to implement the benchmark, with area reductions up to 18% (*-18.3 %*). This can be attributed to both smaller logic and BLE routing areas for the proposed QFPGA implementations while the same number of QFPGA BLEs are needed. The only exception is the smallest of benchmarks (**Adder32**) where binary circuits are more efficient. The quality of the proposed benchmark is accentuated by a plot of benchmark areas, given in Fig. 6.1. Considering the plot trend in Fig. 6.1, it is predicted that this study's QFPGA BLE can achieve more competitive area gains with more complex benchmark implementations. More area gains are further expected when actual routing areas are considered for the proposed benchmarks (as opposed to the pessimistic identical routing areas), since smaller CLB areas will translate to less routing area.

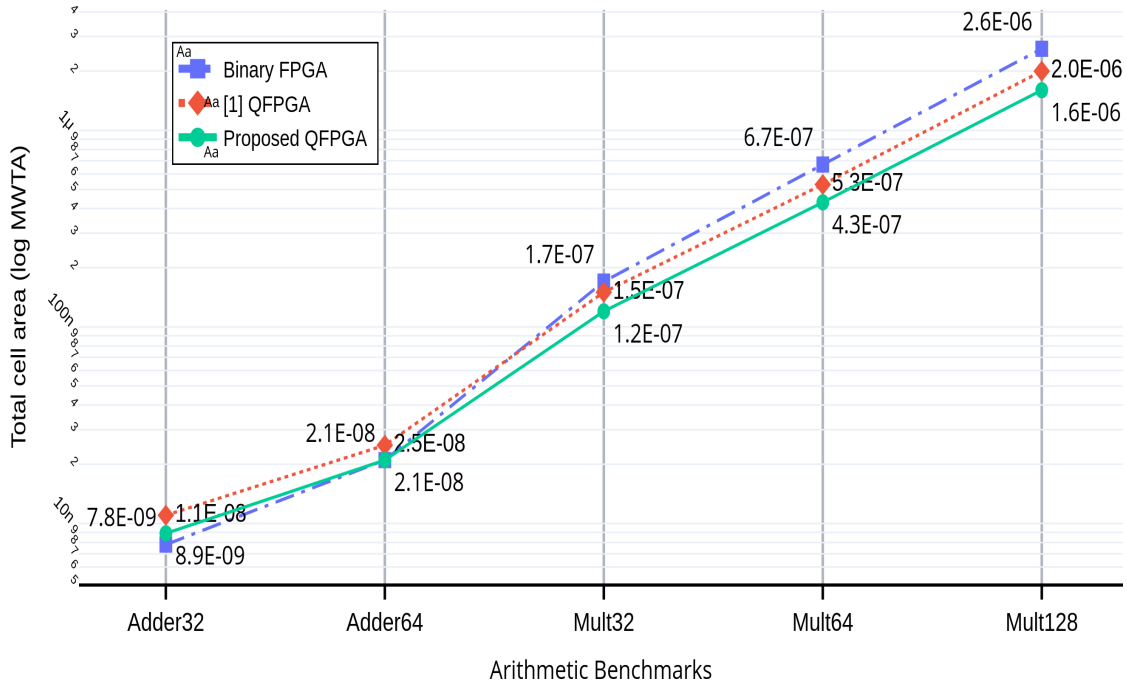

Figure 6.1: Plots of benchmark implementations reveal a trend of area gains for the proposed QFPGA BLE-based benchmarks compared with binary and [1] QFPGA implementations.

Chapter 7

Conclusions and Future Work

Unique MVL memories were implemented with floating-gate transistors and were used in a novel QFPGA architecture that is a competitive, viable alternative to an equivalent binary FPGA and previously proposed QFPGA architectures [1], both in terms of single-BLE delay and power density as well as benchmark-level area. The proposed QFPGA BLE architecture is compatible with modern FinFET logic technology and can be fabricated in processes allowing floating gates and digital logic.

Future work will investigate modelling interconnects as overhead delays to incorporate its impact on the performance of logic implementations (using binary and quaternary BLEs). Such a study will further validate the proposed quaternary BLE's performance.

Another area which will be addressed in future studies is novel architectures required to efficiently program the MVL memory elements. Although this work presumed that floating-gate transistors can be initialized in the circuit simulator, a more accurate model that can be verified and applied in hardware is required.

Furthermore, future research work should also focus on the challenges of implementing non-binary structures in a binary world. Given that we live in a binary world, there are currently no functional MVL or quaternary-logic synthesis tools or compilers. This presents a critical hurdle to validating MVL designs for use in real-world applications.

Bibliography

[1] S. Chaudhuri, "Beyond Bits: A Quaternary FPGA Architecture Using Multi-Vt Multi-Vdd FDSOI Devices," in *IEEE 48th International Symposium on Multiple-Valued Logic (ISMVL)*, Linz, Austria, May 2018, pp. 38–43.

[2] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Publishers, 1999.

[3] C. Lazzari, J. Fernandes, P. Flores, and J. Monteiro, "An Efficient Low Power Multiple-Value Look-up Table Targeting Quaternary FPGAs," in *Proceedings of the 20th International Conference on Integrated Circuit and System Design: Power and Timing Modeling, Optimization and Simulation*, ser. PATMOS'10. Berlin, Heidelberg: Springer-Verlag, pp. 84–93.

[4] P. Pavan, L. Larcher, and A. Marmiroli, "Floating Gate Devices: Operation and Compact Modeling," *NSTI Nanotechnology Conference and Trade Show - NSTI Nanotech*, vol. 2, Jan. 2004.

[5] S. J. Rapp, K. R. Mcmillan, and D. W. Graham, "SPICE-compatible modelling technique for simulating floating-gate transistors," *Electronics Letters*, vol. 47, no. 8, pp. 483 –485, April 2011.

[6] X. Chen and Y. Ha, "The Optimization of Interconnection Networks in FPGAs," in *Dynamically Reconfigurable Architectures*, ser. Dagstuhl Seminar Proceedings, P. M. Athanas, J. Becker, J. Teich, and I. Verbauwhede, Eds., no. 10281, Dagstuhl, Germany, 2010.

[7] Z. Marrakchi, H. Mrabet, U. Farooq, and H. Mehrez, "FPGA Interconnect Topologies Exploration," *International Journal of Reconfigurable Computing*, vol. 1, no. Article 6, pp. 1687–7195, 2009.

[8] A. DeHon, "Balancing Interconnect and Computation in a Reconfigurable Computing Array (or, Why You Don't Really Want 100% LUT Utilization)," in *Proceedings of ACM/SIGDA Seventh International Symposium on Field Programmable Gate Arrays*. Association for Computing Machinery, 1999, pp. 69–78.

[9] Smith, K.C., "The Prospects for Multivalued Logic: A Technology and Applications View," *IEEE Transactions on Computers*, vol. C-30, no. 9, pp. 619–634, Sep. 1981.

[10] E. Dubrova, "Multiple-Valued Logic in VLSI: Challenges and Opportunities," *Proceedings of NORCHIP'99*, Nov. 1999.

[11] A. Sheikholeslami, R. Yoshimura, and P. G. Gulak, "Look-up tables (LUTs) for multiple-valued, combinational logic," in *28th IEEE International Symposium on Multiple-Valued Logic*. USA: IEEE Computer Society, May 1998, pp. 264–269.

[12] P. M. Kelly, T. M. McGinnity, L. P. Maguire, and L. McDaid, "Exploiting binary functionality in quaternary look-up tables for increased functional density in multiple-valued logic FPGAs," *Electronics Letters*, vol. 41, no. 6, pp. 300–302, March 2005.

[13] P. Beckett, "Towards a balanced ternary FPGA," in *International Conference on Field-Programmable Technology*, Sydney, Australia, Dec. 2009, pp. 46–53.

[14] T. Felicijan and S. B. Furber, "An asynchronous ternary logic signaling system," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 6, pp. 1114–1119, Dec. 2003.

[15] H. Shirahama and T. Hanyu, "Design of High-Performance Quaternary Adders Based on Output-Generator Sharing," in *38th International Symposium on Multiple Valued Logic*. USA: IEEE Computer Society, May 2008, pp. 8–13.

[16] R. Cunha, H. Boudinov, and L. Carro, "Quaternary Look-Up Tables Using Voltage-Mode CMOS Logic Design," in *37th International Symposium on Multiple-Valued Logic*. USA: IEEE Computer Society, May 2007, p. 56.

[17] NCSU Electronic Design Automation (EDA). (2017) FreePDK15 wiki page. [Online]. Available: https://www.eda.ncsu.edu/wiki/FreePDK15:Contents

[18] H. . Lee, S. Rami, S. Ravikumar, V. Neeli, K. Phoa, B. Sell, and Y. Zhang, "Intel 22nm FinFET (22FFL) Process Technology for RF and mm Wave Applications and Circuit Design Optimization for FinFET Technology," in *IEEE International Electron Devices Meeting (IEDM)*, San Francisco, USA, 2018.

[19] C. Auth, "45nm high-k + metal gate strain-enhanced CMOS transistors," in *IEEE Custom Integrated Circuits Conference*, California, USA, Oct. 2008, pp. 379–386.

[20] G. Panti, "MULTI-VALUED LOGICS," *Quantified Representation of Uncertainty and Imprecision*, vol. 1, pp. 25–26, 2013.

[21] S. Gottwald, "A Treatise on Many-Valued Logics." England: Research Studies Press, Jan. 2001, pp. 3–5.

[22] T. Hanyu and M. Kameyama, "A 200 MHz pipelined multiplier using 1.5 V-supply multiple-valued MOS current-mode circuits with dual-rail source-coupled logic," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 11, pp. 1239–1245, Nov. 1995.

[23] D. Brito, T. Rabuske, J. Fernandes, P. Flores, and J. Monteiro, "Quaternary Logic Lookup Table in Standard CMOS," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 2, pp. 306–316, Jan. 2014.

[24] E. Ozer, R. Sendag, and D. Gregg, "Multiple-valued logic buses for reducing bus energy in low-power systems," *IEEE Proceedings - Computers and Digital Techniques*, vol. 153, no. 4, pp. 270–282, July 2006.

[25] A. N. Gupte and A. K. Goel, "Study of quaternary logic versus binary logic," in *First Great Lakes Symposium on VLSI*. Los Alamitos, CA, USA: IEEE Computer Society, March 1991, pp. 336–337.

[26] V. P. K.S. and K. S. Gurumurthy, "Quaternary CMOS Combinational Logic Circuits," in *International Conference on Information and Multimedia Technology*. USA: IEEE Computer Society, Dec. 2009, pp. 538–542.

[27] D. Etiemble and M. Israël, "Comparison of Binary and Multivalued ICs According to VLSI Criteria," *Computer*, vol. 21, no. 4, pp. 28—-42, Apr. 1988.

[28] R. H. Freeman, "Configurable Electrical Circuit Having Configurable Logic Elements And Configurable Interconnects," Granted Patent US 4 870 302 A, Sept., 1989.

[29] J. Rodriguez-Andina, M. Valdés, and M. Moure, "Advanced Features and Industrial Applications of FPGAs - A Review," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 4, pp. 853–864, Sept. 2015.

[30] E. Monmasson and M. Cirstea, "FPGA Design Methodology for Industrial Control Systems—A Review," *Industrial Electronics, IEEE Transactions on*, vol. 54, no. 4, pp. 1824–1842, Sept. 2007.

[31] J. Weerasinghe, R. Polig, F. Abel, and C. Hagleitner, "Network-Attached FPGAs for Data Center Applications," in *International Conference on Field-Programmable Technology (FPT)*, Dec. 2016, pp. 36–43.

[32] U. R. Khan, H. L. Owen, and J. L. A. Hughes, "FPGA architectures for ASIC hardware emulators," in *Sixth Annual IEEE International ASIC Conference and Exhibit*, Newyork, USA, Sep. 1993, pp. 336–340.

[33] J. Rose and D. Hill, "Architectural and Physical Design Challenges for One-Million Gate FPGAs and Beyond," in *Proceedings of the ACM Fifth International Symposium on Field-Programmable Gate Arrays*. New York, NY, USA: Association for Computing Machinery, 1997, pp. 129–132.

[34] L. Shang, A. S. Kaviani, and K. Bathala, "Dynamic Power Consumption in VirtexTM-II FPGA Family," in *Proceedings of ACM/SIGDA Tenth International Symposium on Field-Programmable Gate Arrays*. New York, NY, USA: Association for Computing Machinery, 2002, pp. 157–164.

[35] N. Zhang, X. Wang, H. Tang, A. Wang, Z. Wang, and B. Chi, "Low-voltage and high-speed FPGA I/O cell design in 90nm CMOS," in *IEEE 8th International Conference on ASIC*, Changsha, China, Oct. 2009, pp. 533–536.

[36] Z. Zilic and Z. G. Vranesic, "Multiple-valued logic in FPGAs," in *36th Midwest Symposium on Circuits and Systems*, Detroit, MI, USA, Aug. 1993, pp. 1553–1556 vol.2.

[37] K. W. Current, "Current-mode CMOS multiple-valued logic circuits," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 2, pp. 95–107, Feb. 1994.

[38] R. Silva, C. Lazzari, H. Boudinov, and L. Carro, "CMOS voltage-mode quaternary look-up tables for multi-valued FPGAs," *Microelectronics Journal*, vol. 40, pp. 1466–1470, Oct. 2009.

[39] X. Guo, V. Verma, P. Gonzalez, S. Mosanu, and M. Stan, "Back to the Future: Digital Circuit Design in the FinFET Era," *Journal of Low Power Electronics*, vol. 13, pp. 338–355, Sept. 2017.

[40] D. Brito, T. Rabuske, J. Fernandes, P. Flores, and J. Monteiro, "Quaternary Logic Lookup Table in Standard CMOS," vol. 23, no. 2. Los Alamitos, CA, USA: IEEE Computer Society, Feb. 2015.

[41] D. Kahng and S. M. Sze, "A floating gate and its application to memory devices," *The Bell System Technical Journal*, vol. 46, no. 6, pp. 1288–1295, July 1967.

[42] Y. Fujisaki, "Review of Emerging New Solid-State Non-Volatile Memories," *Japanese Journal of Applied Physics*, vol. 52, pp. 040 001–040 002, Apr. 2013.

[43] S. Lai, "Flash memories: where we were and where we are going," in *International Electron Devices Meeting Technical Digest (Cat. No.98CH36217)*, Dec. 1998, pp. 971–973.

[44] E. Takeda, A. Shimizu, and T. Hagiwara, "Role of hot-hole injection in hot-carrier effects and the small degraded channel region in MOSFET's," *IEEE Electron Device Letters*, vol. 4, no. 9, pp. 329–331, Sep. 1983.

[45] P. Hasler, B. A. Minch, and C. Diorio, "Adaptive circuits using pFET floating-gate devices," in *Proceedings of the 20th Anniversary Conference on Advanced Research in VLSI*. USA: IEEE Computer Society, Mar. 1999, pp. 215–229.

[46] C. A. Mead, "Scaling of MOS Technology to Submicrometer Feature Sizes," in *Feynman and Computation: Exploring the Limits of Computers*. USA: Perseus Books, 1999, pp. 93–115.

[47] K. Rahimi, C. Diorio, C. Hernandez, and M. D. Brockhausen, "A simulation model for floating-gate MOS synapse transistors," in *IEEE International Symposium on Circuits and Systems. Proceedings (Cat. No.02CH37353)*, vol. 2, May 2002, pp. II–II.

[48] J. L. Gray, R. Robucci, and P. Hasler, "The design and simulation model of an analog floating-gate computational element for use in large-scale analog reconfigurable systems," in *51st Midwest Symposium on Circuits and Systems*, Aug. 2008, pp. 253–256.

[49] K. G. Nichols, T. J. Kazmierski, M. Zwolinski, and A. D. Brown, "Overview of SPICE-like circuit simulation algorithms," *IEEE Proceedings - Circuits, Devices and Systems*, vol. 141, no. 4, pp. 242–250, Aug. 1994.

[50] B. Swahn and S. Hassoun, "Gate sizing: FinFETs vs 32nm bulk MOSFETs," in *Proceedings of the 43rd ACM/IEEE Design Automation Conference*. San Francisco, CA, USA: Association for Computing Machinery, July 2006, pp. 528–531.

[51] W. P. Maszara and M. Lin, "FinFETs - Technology and circuit design challenges," in *Proceedings of the European Solid-State Device Research Conference*, Bucharest, Romania, Sep. 2013, pp. 3–8.

[52] P. Mishra, A. Muttreja, and N. Jha, "FinFET circuit design," in *Nanoelectronic Circuit Design*. United States: Springer New York, Dec. 2011, pp. 23–54.

[53] V. Kursun and E. Friedman, "Supply and Threshold Voltage Scaling Techniques," in *Multi-Voltage CMOS Circuit Design*. John Wiley Sons, Ltd, 2006, pp. 45–84.

[54] M. Mustafa, "Threshold Voltage Sensitivity to Metal Gate Work-Function Based Performance Evaluation of Double-Gate n-FinFET Structures for LSTP Technology," *World Journal of Nano Science and Engineering*, vol. 3, pp. 17–22, Jan. 2013.

[55] M. Rostami and K. Mohanram, "Novel dual-V_{th} independent-gate FinFET circuits," in *Proceedings of the Asia and South Pacific Design Automation Conference*. Taipei, Taiwan: IEEE Press, 2010, pp. 867–872.

[56] Y. Liu, T. Kamei, T. Matsukawa, K. Endo, S. O'uchi, J. Tsukada, H. Yamauchi, Y. Ishikawa, T. Hayashida, K. Sakamoto, A. Ogura, and M. Masahara, "FinFET flash memory technology," in *ECS Transactions*, vol. 45, Apr. 2012, pp. 289–310.

[57] S. Young, P. Alfke, C. Fewer, S. McMillan, B. Blodget, and D. Levi, "A high I/O reconfigurable crossbar switch," in *Proceedings of the 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, Napa, CA, USA, April 2003, pp. 3–10.

[58] F. F. Khan and A. Ye, "Measuring the Accuracy of Minimum Width Transistor Area in Estimating FPGA Layout Area," in *Proceedings of the IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines*, Vancouver, British Columbia, Canada, May 2015, pp. 223–226.