FPGA Interconnection Networks with Capacitive Boosting in Strong and Weak Inversion

by

Fatemeh Eslami
B.Sc., Shahid Beheshti University, Tehran, Iran, 2009

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

© Fatemeh Eslami, 2012
University of Victoria

All rights reserved. This thesis may not be reproduced in whole or in part, by photocopying or other means, without the permission of the author.

FPGA Interconnection Networks with Capacitive Boosting in Strong and Weak Inversion

by

Fatemeh Eslami
B.Sc., Shahid Beheshti University, Tehran, Iran, 2009

Supervisory Committee

Dr. Mihai Sima, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Michael McGuire, Departmental Member
(Department of Electrical and Computer Engineering)

Dr. Daniela Constantinescu, Outside Member
(Department of Mechanical Engineering)

Supervisory Committee

Dr. Mihai Sima, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Michael McGuire, Departmental Member
(Department of Electrical and Computer Engineering)

Dr. Daniela Constantinescu, Outside Member
(Department of Mechanical Engineering)

ABSTRACT

Designers of Field-Programmable Gate Arrays (FPGAs) are always striving to improve the speed of their designs. The propagation delay of FPGA interconnection networks is a major challenge and continues to grow with newer technologies. FPGAs interconnection networks are implemented using NMOS pass transistor based multiplexers followed by buffers. The threshold voltage drop across an NMOS device degrades the high logic value, and results in unbalanced rising and falling edges, static power consumption due to the crowbar currents, and reduced noise margins. In this work, circuit design techniques to construct interconnection circuit with *capacitive boosting* are proposed. By using capacitive boosting in FPGAs interconnection networks, the signal transitions are accelerated and the crowbar currents of downstream buffers are reduced. In addition, buffers can be non-skewed or slightly skewed to improve noise immunity of the interconnection network. Results indicate that by using the presented circuit design technique, the propagation delay can be reduced by at least 10% versus prior art at the expense of a slight increase in silicon area.

In addition, in a bid to reduce power consumption in reconfigurable arrays, operation in weak inversion region has been suggested. Current programmable interconnections cannot be directly used in this region due to a very poor propagation delay and sensitivity to *Process-Voltage-Temperature (PVT)* variations. This work also

focuses on designing a common structure for FPGAs interconnection networks that can operate in both strong and weak inversion. We propose to use capacitive boosting together with a new circuit design technique, called *Twins transmission gates* in implementing FPGA interconnect multiplexers. We also propose to use capacitive boosting in designing buffers. This way, the operation region of the interconnection circuitry is shifted away from weak inversion toward strong inversion resulting in improved speed and enhanced tolerance to PVT variations. Simulation results indicate using capacitive boosting to implement the interconnection network can have a significant influence on delay and tolerance to variations. The interconnection network with capacitive boosting is at least 34% faster than prior art in weak inversion.

Contents

List of Tables

List of Figures

List of Abbreviations

ADP	Area-Delay Product
ASICs	Application-Specic Integrated Circuits
CB	Connection Box
CLB	Configurable Logic Block
CMC	Canadian Microelectronics Corporation
CMOS	Complementary Metal Oxide Semiconductor
Cu	Copper
DIBL	Drain-Induced Barrier Lowering
FF	Flip-flop
FPGA	Feild Programmable Gate Array
LE	Logic Element
LUT	Lookup Table
MOM	Metal-Over-Metal
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
MOSIS	Metal Oxide Semiconductor Implementation Service
MWCNT	Multi-Walled Carbon Nanotube
NMOS	n-channel MOSFET
PDP	Power-Delay Product

PMOS	p-channel MOSFET
PTL	Pass Transistor Logic
PVT	Process, Voltage and Temperature
SB	Switch Box
SOI	Silicon on Insulator
SRAM	Static Random Access Memory
TSMC	Taiwan Semiconductor Manufacturing Company
ULP	Ultra-low Power

ACKNOWLEDGEMENTS

My foremost gratitude goes to my academic supervisor, Dr. Mihai Sima. Throughout my studies at Uvic, Dr. Sima provided me with a tremendous amount of guidance, encouragement, support, and friendship. All I can say in this small amount of space is that I consider myself extremely fortunate to have had the opportunity to work under his supervision. Thank you, Dr. Sima, for your always strong support and guidance.

I am eternally grateful to my thesis committee members, Dr. McGuire and Dr. Constantinescu for their feedback, which has always aimed to make this thesis better.

Last but not least, I wish to thank my family for their unwavering support and encouragment. Without them, I would never go this far.

Chapter 1

Introduction

1.1 Motivations and Objectives

Field-Programmable Gate Arrays (FPGAs) are integrated circuits that can be programmed to implement any digital circuit subject to available logic capacity. FPGAs are used in a wide variety of applications such as communications, digital signal processing, cryptography, and bioinformatics [5]. The primary advantages of FPGAs are that they are flexible, and thus can be (re)configured to implement application-specific computation. This flexibility results in a shorter time to market than designing an Application-Specic Integrated Circuits (ASICs). However, this flexibility makes FPGAs significantly slower and less power-efficient than ASICs. The speed and power overhead of FPGAs limit the use of FPGAs for high-speed or low-power applications.

It has been identified that FPGA interconnection networks are the main contributor to the overall propagation delay of FPGAs [6, 7]. The effect of FPGA interconnection networks on FPGA performance motivates research of new more efficient and high performance FPGA interconnection networks. A significant number of studies have already focused on faster, more area efficient programmable routing resources in *strong inversion* (also known as super-threshold region) [8, 9, 10, 11, 3].

In energy-constrained applications, such as cellular phones, laptop computers, biomedical devices like hearing aid, and wireless receivers, power consumption is the primary requirement while speed is of secondary consideration [12, 13, 14]. In such applications, sub-threshold FPGA, where power consumption is reduced by order of magnitudes by operating in weak inversion) (also known as sub-threshold region), would be a promising option [7]. However, this power saving is not without chal-

lenges; in weak inversion, where the drain current depends exponentially on the gate-to-source voltage, the propagation delay becomes very large and continues to grow with supply voltage scaling. Furethermore, the leakage current integrates over the much longer delay until leakage energy exceeds the active energy; this has strong implications in the optimization of sub-threshold circuits. A high sensitivity to *Process-Voltage-Temperature (PVT)* variations is encountered, since in this region the drain current also depends exponentially on the threshold voltage [14, 7]. Therefore, the circuits used in commercial FPGAs in strong inversion can not be used in weak inversion without any adaptation. Only a few studies have investigated the optimization of circuit design for FPGA interconnection network in weak inversion [7, 15, 16].

This work focuses on circuit techniques that are compatible with and supportive of those approaches in terms of design architecture and structure, and particularly with regards to advanced technologies such as 90nm and 65nm process nodes. The main goal of this thesis is to analyse and augment circuit structures used to implement interconnection networks in deep-submicron FPGAs to make a tradeoff between delay, power, area, and reliability in both strong and weak inversion.

FPGA interconnection networks are implemented using multiplexeres followed by buffers. Multiplexers are implemented using NMOS pass transistor. The threshold voltage drop across an NMOS device degrades the high logic value resulting in unbalanced rising edge and falling edge and reduced noise margins. In addition, the problem of passing high logic value through NMOS device makes the buffers suffer from static power consumption due to the crowbar currents. This problem is more severe in sub-threshold since there is an exponential relationship between the drain current and gate-to source voltage, which translates into exponentially larger propagation delay, and the circuit is more sensitive to PVT variations. The long-term goal of this research is to design a common structure for FPGA interconnection networks that can operate in both strong and weak inversion.

1.2 Contributions

The contributions of this research can be organized into two categories: FPGA interconnection networks in strong inversion and FPGA interconnection networks in weak inversion. With respect to FPGA interconnection networks in strong inversion, our contributions are as follows:

1. Circuit techniques that use *capacitive boosting* for building programmable interconnection networks. An NMOS device called *Isolator* is incorporated into SRAM cells used in multiplexers to creat a capacitive boosting and provide full-swing signaling. A PMOS device *called Trimmer* is used to make balanced rising edge and falling edge. This results in designing non-skewed or at most slightly-skewed bufferes that have a good noise immunity.

2. The use of capacitive boosting in implementing interconnection networks were verified to improve the delay by at least 10% and 17% for short and long inteconnection, respectively.

3. Making tradeoffs between area, delay, and power consumption of FPGA interconnection networks in modern thechnologies. At least 17% and 12% improvement for *power-delay product (PDP)* shows a good tradeoff between the power and delay. In addition, a slight improvement in *area-delay product (ADP)* shows that there is a good balance between area and delay. These improvements are achieved without any use of low-threshold or zero-threshold transistors and dual-rail supply voltages to avoid additional large costs and steps during device fabrication.

Since the circuits used in commercial FPGAs in strong inversion can not be used in weak inversion without any adaptation, our contributions with respect to FPGA interconnection networks in weak inversion are as follows:

4. Circuit design techniques that use capacitive boosting in implementing FPGAs interconnection networks operating in weak inversion. Capacitive boosting is used to implement multiplexers and also bufferes to take advangae of the exponential dependency of the drain current to the gate-to-source voltage in weak inversion.

5. Circuit design techniques in implementing buffers with capacitive boosting to shift the operation region away from weak inversion toward strong inversion. Shifting the operation from weak inversion to strong inversion caused increased drive current and enhanced speed and tolerance to PVT variations.

6. Implementing multiplexeres with capacitive boosting and *Twins transmision gate*, which uses only NMOS devices to balance the falling edge and rising edges, resulting in enhanced speed and tolerance to PVT variations.

7. Using the circuit design techniques in implementing FPGAs interconnection network, it is shown that capacitive boosting is effective at reducing delay by at least 34% for interconnection networks.

8. Using the circuit design techniques, it is shown that capacitive boosting is effective at enhancing the interconnection networks tolerance at the present of PVT variations.

9. Making tradeoffs between area, delay, and power consumption of FPGA interconnection networks. At least 42% and 23% improvement has been achieved for *area-delay product (ADP)* and *power-delay product (PDP)*, respectively. These tradeoffs are achieved without any use of low-threshold or zero-threshold transistors or dual-rail supply voltages to avoid additional large costs and steps during device fabrication.

10. Showing that the circuit design techniques are independent of the technology and do not need redesign based on the technology.

1.3 Organization

This thesis is composed of 5 chapters. Chapter 2 provides related background information on the main components of FPGAs and challenges facing implementing these main components. It also presents metrics used in evaluating FPGAs. Chapter 3 presents the circuit design of FPGA interconnection networks in details. In addition, it provides circuit design techniques in implementing the interconnection networks in strong inversion. Chapter 4 focuses on weak inversion and challenges facing designing sub-threshold FPGAs. It also provides circuit design techniques in implementing the interconnection networks in weak inversion. Chapter 5 summarizes the conclusions drawn throughout the thesis and provides suggestions for future work.

Chapter 2

Background

The main goal of this thesis is to understand and augment the circuit structures used to implement the main components in deep-submicron FPGAs to make good tradeoffs between delay, power consumption, area and reliability in both super- and sub-threshold regions.

In this chapter, the conventional design approaches for the main components of FPGAs are summarized. Transistor-level design is a challenging task and the implementation affects the area and performance of an FPGA significantly. Previous attempts addressing transistor-level design difficulties will be reviewed. Issues that necessitated the reliable and high performance circuit technique performed in this thesis will be described. The standard metrics for evaluating an FPGA are also reviewed. Finally, a circuit design technique called *bootstrap technique* (also known as the *bootstrap effect*) is discussed. We believe that bootstrap technique can be used to improve the circuit design of FPGAs components.

2.1 FPGA Architecture and Circuit Structure

FPGAs have three primary components: *a programmable interconnection network (routing)* that connects various blocks by turning on and off appropriate switches; *Configurable Logic Blocks (CLBs)* which implement logic functions; *I/O blocks* that will not be explored in this thesis. Figure 2.1 shows this basic structure of an FPGA. *Connection boxes (CB)* exist on a CLBs four sides to allow input signals to be routed into the CLB. *Switch boxes (SB)* allow CLB output signals to be routed out and also provide connectivity between wire segments as shown in Figure 2.1. In this section,

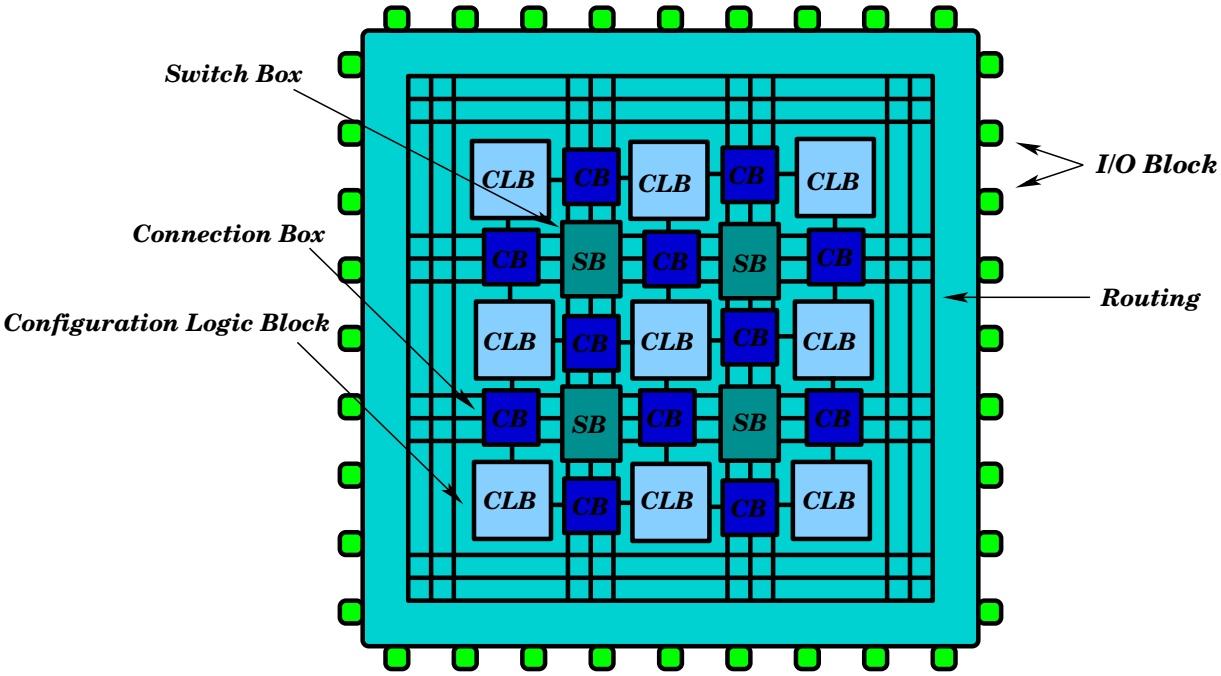

Figure 2.1: Basic architecture of an FPGA

the implementation of the interconnection network and the major components of CLBs are reviewed. Particular emphasis will be placed upon the routing related topics.

Interconnection Network

The interconnection network consumes the largest amount of area in an FPGA [17]. Connectivity between logic blocks is achieved through the wires and programmable interconnect resources. As shown in Figure 2.2, programmable interconnect circuits are composed of SRAM configuration memory, multiplexers, and drivers [5, 18, 3].

A multiplexer selects the signal to be passed to the output from a variety of inputs. Programmability is acheieved at boot time by uploading configuration information into SRAM memory cells. Since multiplexers are widely used in FPGAs interconnection network, their implementation affects the area and performance of an FPGA significantly. To reduce area and improve speed, multiplexers are generally implemented using NMOS pass gates. The drawback of NMOS pass gates is a threshold voltage drop when passing a logical high value, since an NMOS pass transistor with a gate voltage of V_{DD} is unable to pass a signal at V_{DD} from source to drain. The resulting weak logic high prevents the PMOS transistor of the downstream buffer

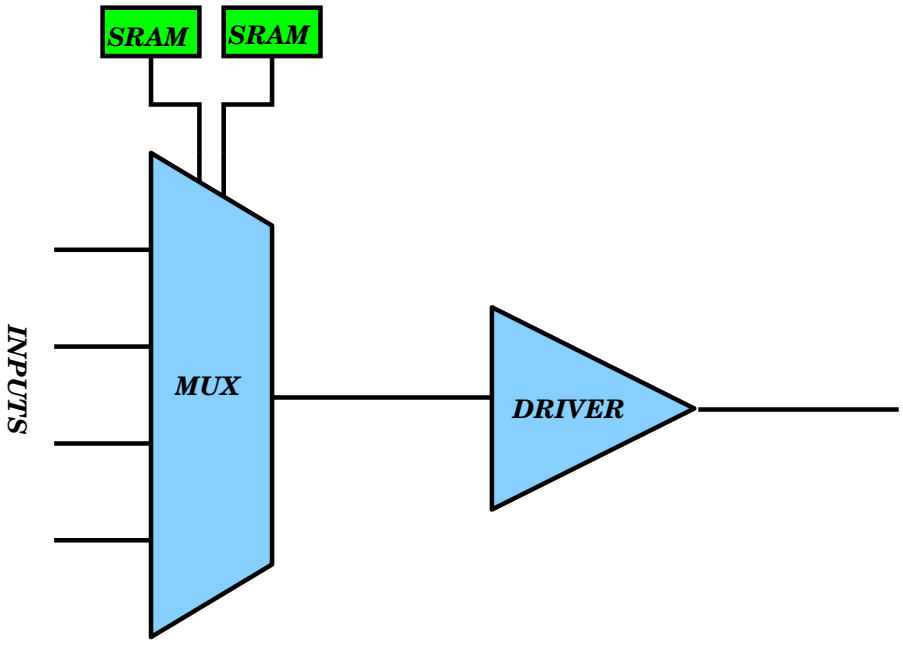

Figure 2.2: Programmable interconnect components

from turning fully off, generating static power consumption. This problem can be mitigated by the use of level restoring PMOS pull-up transistor, called a *keeper* as shown in Figure 2.3; however, since the circuit is now ratioed, this results in increased high_to_low transition time and/or dynamic power consumption [19].

Another less common alternative is to use transmission gates (with an NMOS and a PMOS) to construct the multiplexer tree; however, CMOS transmission gates require near three times the area of an NMOS passgate because of PMOS transistors. In addition, using a PMOS device in parallel with an NMOS device generates large leakage currents and adds parasitic capacitance on the signal propagation path [20, 21, 22, 11, 23]. Other solutions to this problem include:

1. using static gate-boosting method, in which the gate voltage is raised above the standard VDD, but raising concerns on gate oxide integrity and the device reliability as technology scales down [5, 8].

2. using low-threshold or even zero-threshold pass transistors, which eliminates the threshold drop, but increasing leakage current through the other off branches. Moreover, it requires additional steps during device fabrication and makes the solution more expensive and also technology dependent [24, 25].

3. unfolding the multiplexer in order to drive the level restoring buffer with pre-

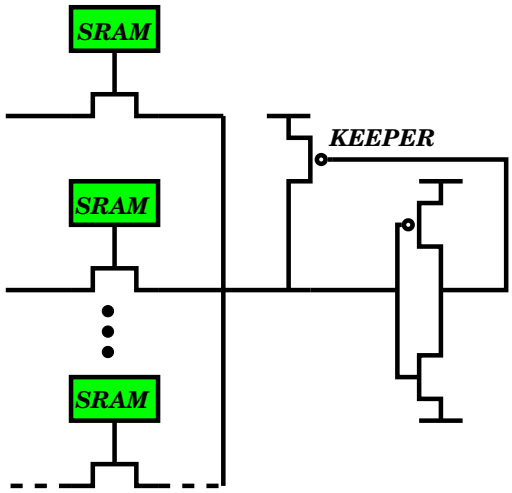

Figure 2.3: Transistor-level implementation of the multiplexer based on NMOS pass gates and Keeper

switch strong signals at the expenses of a larger silicon area. In addition, this technique can be only used for directional routing architectures [1].

These techniques are either technology dependent, require additional voltage supplies, increase the silicon area significantly, generate large leakage currents, add parasitic capacitance on the signal propagation path, or increase the design effort. Designing performant and viable multiplexer without additional supply voltages or low- or zero-threshold devices is one goal of this thesis.

There are two main routing architectures: *bidirectional* and *unidirectional*. In a bidirectional routing network as shown in Figure 2.4, a wire can transmit a signal in either direction. The drivers of the wires are tristate drivers and can be disabled when not being used. A common approach to build tristate buffers in FPGAs is to place an NMOS passgate at the output of the driver as shown in Figure 2.5. However, the output of the NMOS has a negative effect on speed because the NMOS produces a threshold voltage drop in the output signal swing which has to drive a long wire of the routing networks. Moreover, since only one of the two tristate drivers connected to each wire can be enabled after configuration, this approach causes a significant waste of area [8].

In a unidirectional routing network as shown in Figure 2.6, each wire transmits data in a single direction, where each wire is only driven by a single driver. This approach is known as single-driver wiring [9]. The circuit design of single-driver routing is shown in Figure 2.7.

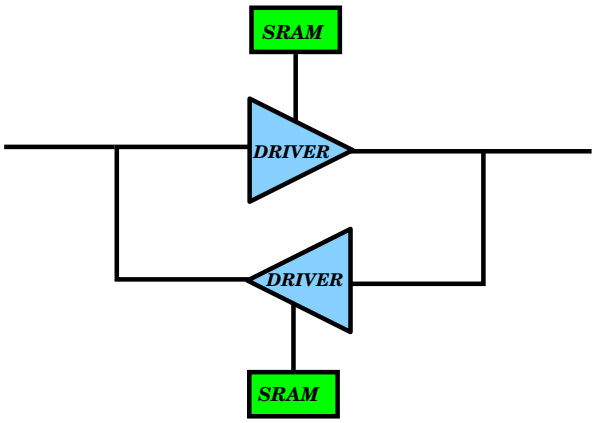

Figure 2.4: Bidirectional Wire

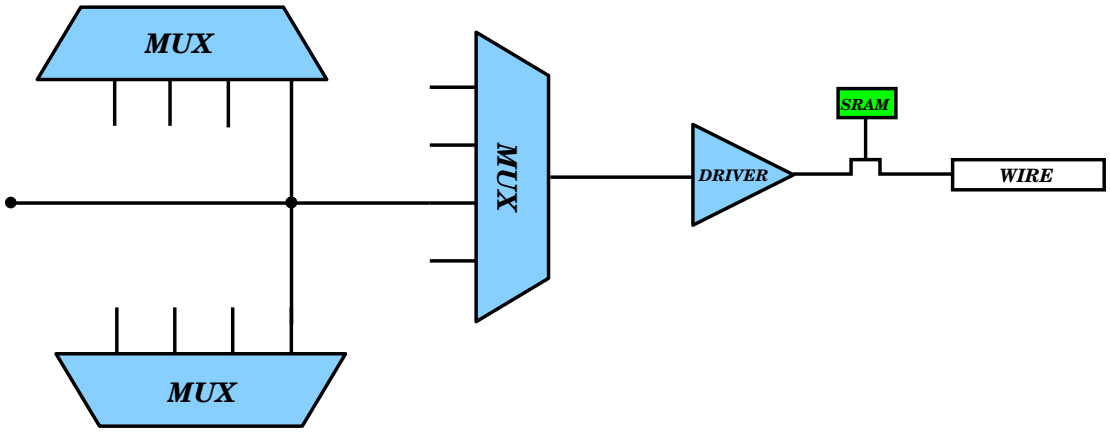

Figure 2.5: Circuit design of tri-state buffer

In this thesis, we consider single-driver routing architecture since this approach is more efficient in terms of area and provides shorter delay over the bidirectional architecture [9].

The driver following the multiplexer (Figure 2.7) strenghtenes the multiplexed signal which has to be transmitted through the wire. A driver is built with one or more inverters of increasing size and connected in series as shown in Figure 2.8. Since the distance between the inverters is very small, it is called a *lumped* driver design. An alternative approach is to space the buffers apart along the length of the wire that they must drive. This is referred to as a *distributed* driver design [10]. This work is focused on the transistor-level circuits of the multiplexers and the drivers inside the FPGAs interconnection network.

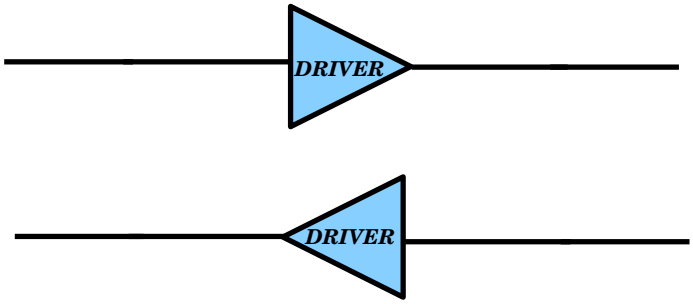

Figure 2.6: Unidirectional wire

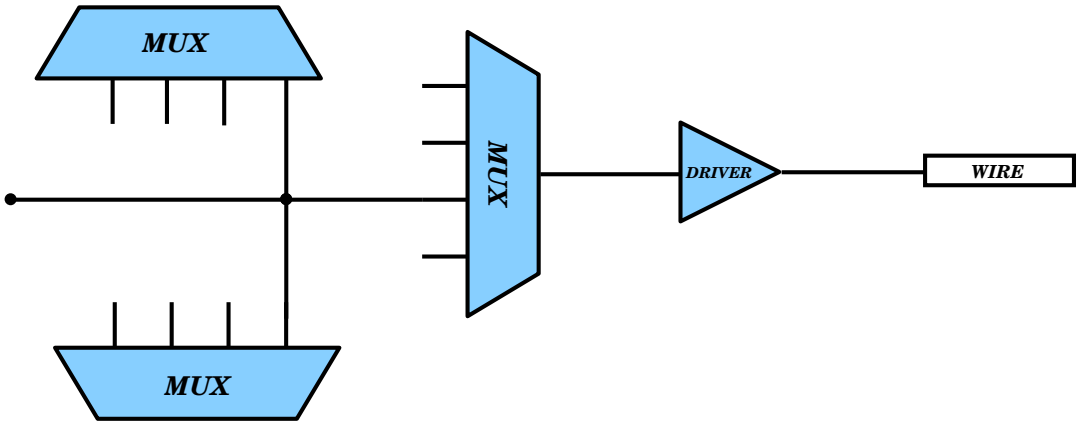

Figure 2.7: Circuit design of single driver

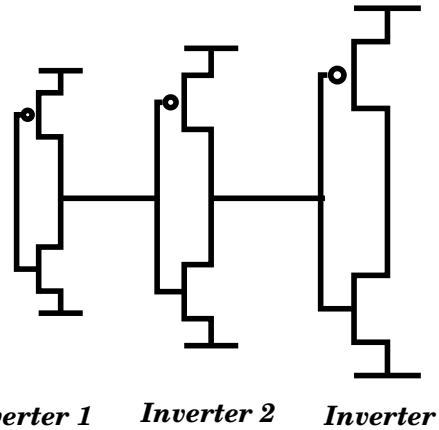

Figure 2.8: Transistor_level circuit of driver

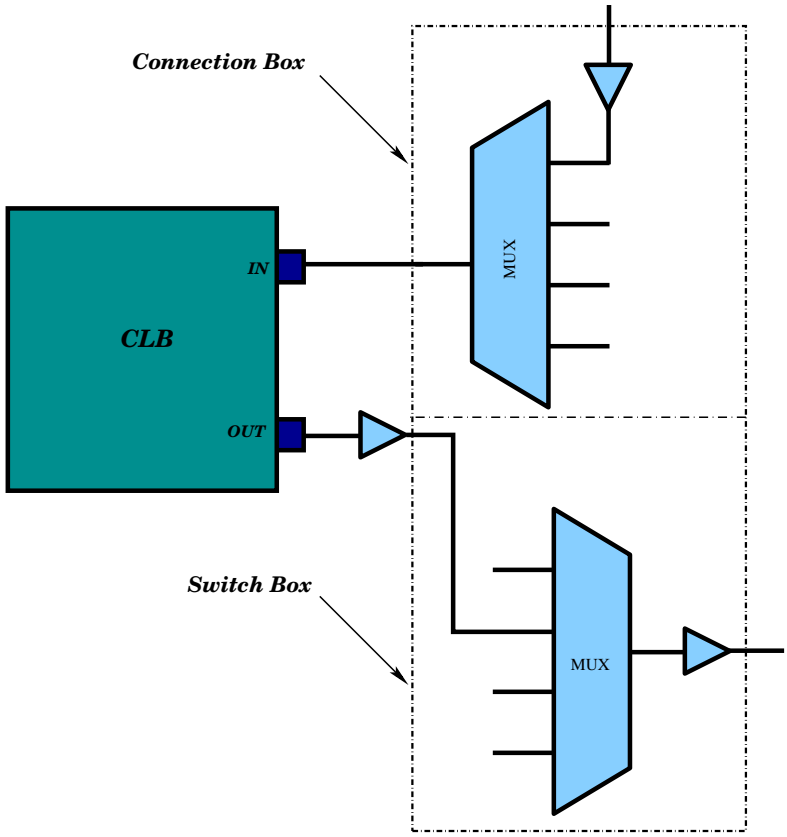

Figure 2.9: Connectivity between CLBs and programmable routing within FPGAs

Configurable Logic Blocks (CLBs)

The Configurable Logic Blocks (CLBs) are the main logic resources for implementing logic functions. Each CLB can be connected to the interconnection network by programmable switches as shown in Figure 2.9. Each CLB is composed of one or more *Logic Elements (LE)*. LEs are commonly built of a lookup table (LUT) with K inputs that can implement any logic function of K inputs, where typically K=4, 5, or 6, and one output. Each LUT is generally paired with a flip-flop to support sequential designs as shown in Figure 2.10. When CLBs contain more than one LE, local interconnect connects the CLB inputs and also outputs back to the inputs of each LE as shown in Figure 2.11 [26, 27]. As show in Figure 2.11, the output of the local interconnect is used as LUTs input in each LE; therefore, the performance of local interconnect has effect on the performance of CLBs.

Improving the circuit design of the local interconnect is part of the work in this thesis.

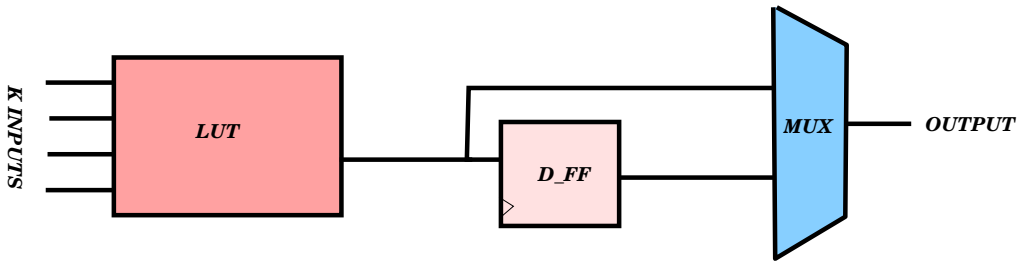

Figure 2.10: Circuit design of LE

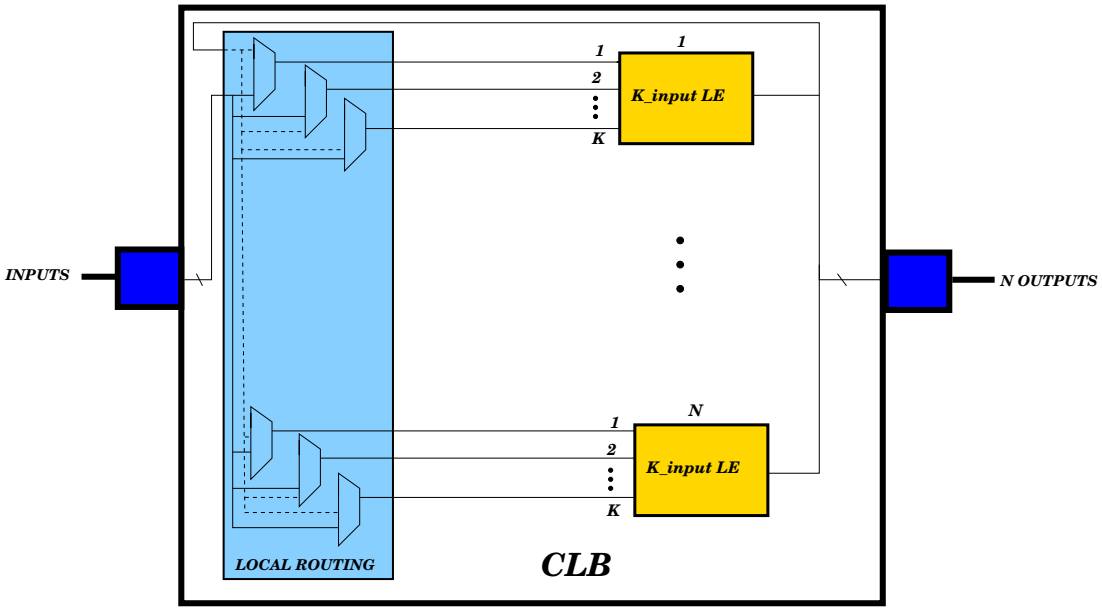

Figure 2.11: CLB circuit design contains of more than one LEs

Since LUTs are used to implement any logic function, their architecture and circuit significantly impacts the area, performance and power consumption of an FPGA. LUTs are commonly implemented by multiplexers [5, 28, 29, 4, 30] as shown in Figure 2.12. Note that this multiplexer differs from the previous routing multiplexer shown in Figure 2.2, since the inputs are now its select signals, and hence they drive the gates of each pass transistor. If multi-stage multiplexers are used to implement a LUT, buffers are typically inserted between every two stages of multiplexing for performance and reliability reasons [28] as shown in Figure 2.13. Modern FPGAs have LUTs that can be configured to be also used as memories or shift registers [27, 31]

At the circuit level, LUTs are often built using NMOS pass-transistor gates for speed and density reasons. Keepers are used to restore the threshold voltage drop across NMOS devices [5, 29, 4, 30]. The circuit design of a pass transistor based

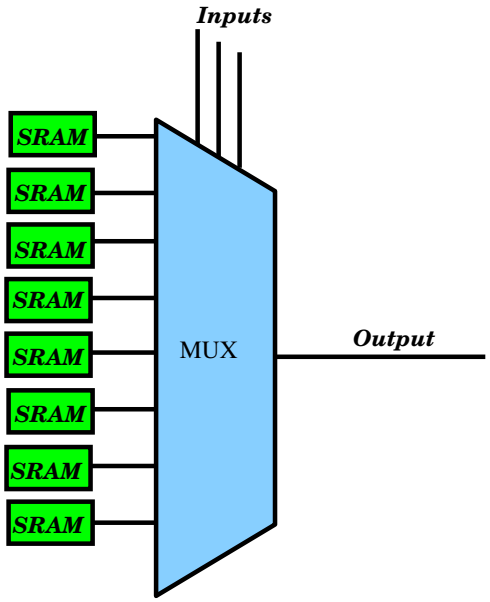

Figure 2.12: Multiplexer as a lookup table

LUT has been shown in Figure 2.14. Using gate-boosting techniques to mitigate the threshold voltage drop has device reliability problem in modern technologies that have thin gate oxides, and are prone to physical deterioration [8]. Anothor alternative is to use transmission-gate based implementation. However, it adds extra load through the signal propagation path and is expensive in terms of area.

The worse-case delay through the LUT occurs when the lefmost select signal toggles since the signal has to propagate from the gate of the transistor to its drain and then go all the way through the N-1 multiplexing stages, where N is the number of multiplexing stages. In addition, having unaligned arrival_time for the LUT input signals causes glitches and increases the power consumption. Designers of FPGAs are striving to improve the circuit design of the LUT in terms of propagation delay to be able to make a good tradeoff between area, delay, and power consumption.

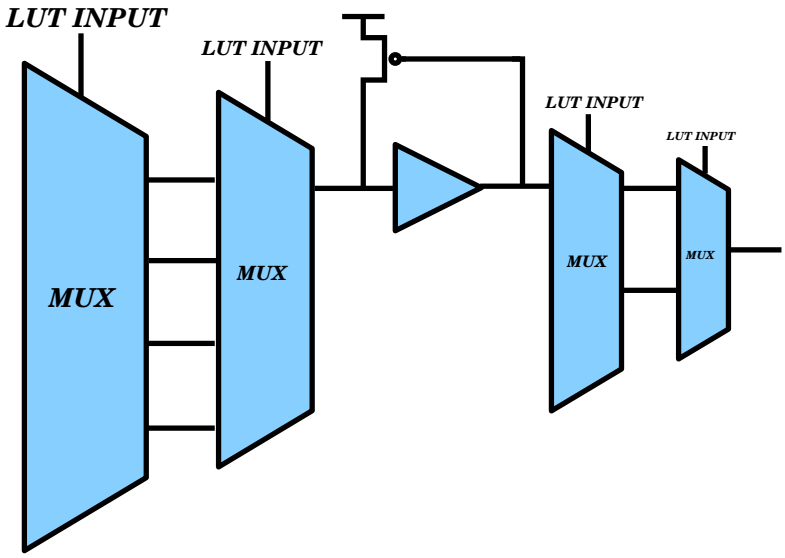

Figure 2.13: Multi-stage multiplexing with buffer insertion

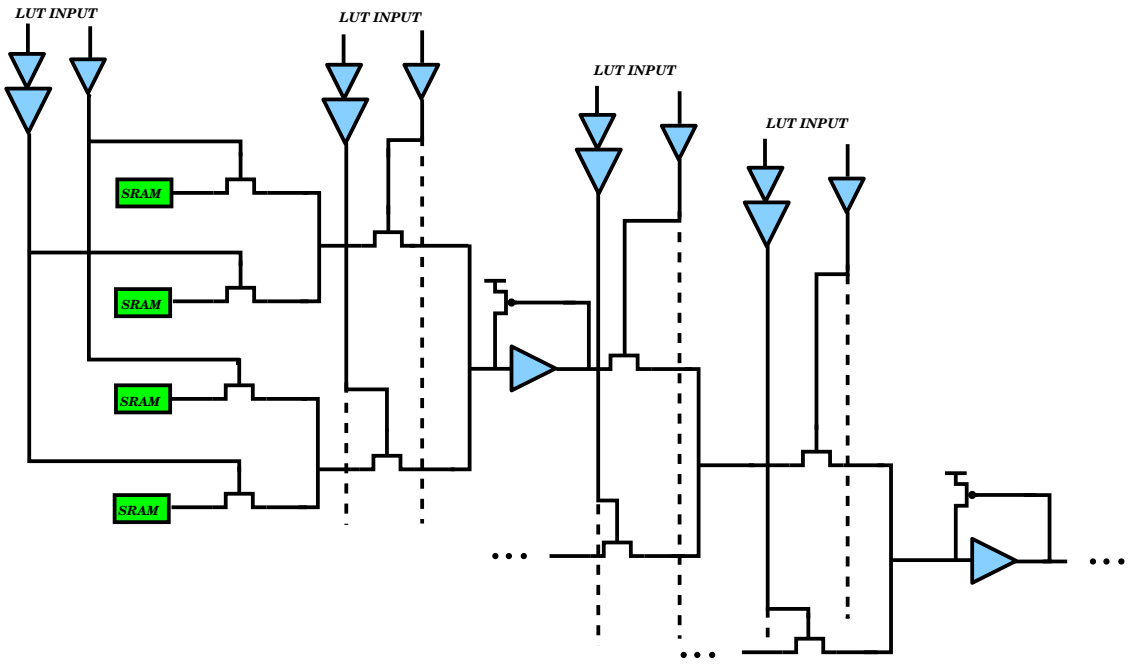

Figure 2.14: Transistor_level circuit of lookup table based on pass transistor

2.2 Evaluation Metrics

FPGAs are used in a wide range of markets with different cost, performance and power consumption requirements. Three metrics are commonly used to evaluate an FPGA implementation: silicon area, propagation delay, and power consumption. Two composite metrics are also widely used in FPGA chip design: the area-delay product (ADP) to ensure that the delay is not sacrificed to obtain the minimum area or vice-versa, and power-delay product (PDP) to ensure that the delay of the interconnect is not sacrificed to obtain the minimum power consumption [5, 1, 32, 29, 33].

All these previously described components of an FPGA can have a significant effect on the area, performance and power consumption. Hence, we believe there is a need to augment the circuit design of an FPGA that enables tradeoffs between area, speed, and power consumption. We consider these evaluating metrics to evaluate the effectiveness of our designs in this thesis.

2.3 Boostrap Pass-Transistor Logic

Bootstrap Pass-Transistor Logic (PTL) was proposed to overcome the output voltage loss and speed degradation of standard PTL. By boosting the gate voltage higher than V_{DD}, the drain voltage of the NMOS pass transistor is able to rise up to V_{DD} without using a Keeper transistor [34, 35, 36]. A Bootstrap configuration consists of a pass transistor for data propagation and an *Isolator* that ensures capacitive coupling in pass transistor between source and gate Figure 2.15. In other words, the Isolator isolates the boosting node (which is the NMOS pass transistor gate) from other potential nodes. The parasitic capacitance C_{gs} helps boost the gate voltage up higher than V_{DD} when the input signal rises, as shown in Figure 2.16. As a result, the voltage at the output node of the NMOS pass-transistor logic can now raise to V_{DD}. The Isolator turns *on*, supplying the pass transistor gate with charge, when the gate voltage drops below V_{DD} - V_P, and turns *off* when the gate voltage rises above V_{DD} -V_P.

The bootstrap technique has been used in DRAM logic and in adiabatic circuits [37]. Bootstrap technique has also been used to improve the general pass-transistor logic with a small voltage supply. They applied boot technique in an arithmetic logic unit (ALU) and XOR circuits at high performance and low power operation [35]. In addition, body-biasing has improved the capacitive coupling in circuits fabricated

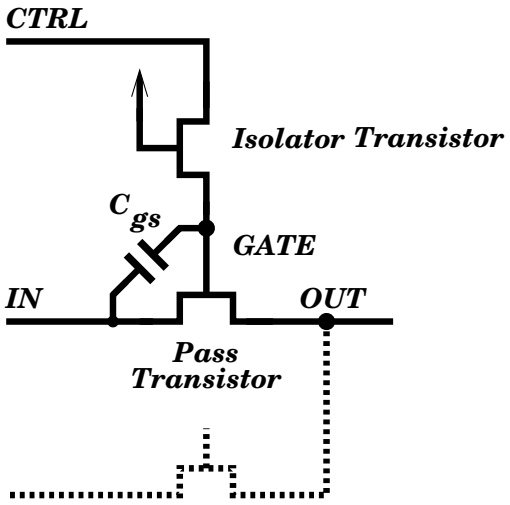

Figure 2.15: Bootstrap pass-transistor logic

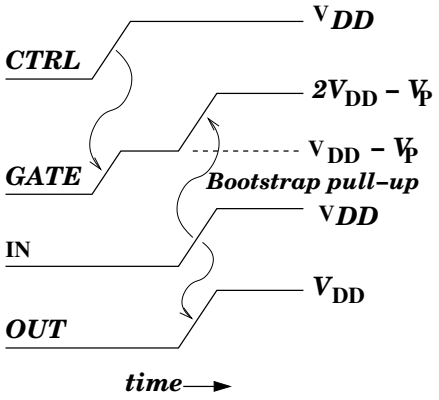

Figure 2.16: Bootstrap waveform

in silicon on insulator (SOI) technology [36]. More recently, capacitive boosting has been used in level shift and word line driving circuits [38], and in shift registers [39]. In these works, boosting is used to shift the voltage level higher than V_{DD}. However, to the best of our knowledge, the bootstrap technique in an FPGA environment has not been reported. This issue is addressed in the subsequent chapter.

Chapter 3

Capacitive Boosting in Strong Inversion for FPGA Interconnection Network

In the previous chapter, we described the prior art in cirucuit design of the FPGA components, with the main goal of having a good tradeoffs between delay, area and power consumption. Specially, we described the challenges that circuit designers face in designing programmable routing components such as multiplexers and drivers. The analysis in Chapter 2 points to the need to enhance the interconnection network since the interconnect delay is a significant problem in FPGAs. We also, explained a circuit technique called *capacitive boosting*, which has been used in pass transistor logice (PTL) in order to improve the circuit delay. This chapter proposes the use of capacitive boosting for an FPGA interconnection network. A key contribution of this dissertation is the design of high speed global and local interconnection networks using capacitive boosting.

This chapter reviews the implementation details of the global and local interconnection networks in FPGAs. Bootstrap technique is then proposed to augment the routing network. The penalties of the capacitive boosting technique are also discussed. Then, a solution to mitigate them in FPGAs environment is presented. Finally the measurements of the area, performance, power consumption, area_delay product (ADP) and power_delay product (PDP) of the global and local routing based on capacitive boosting are presented together with a comparison against prior art.

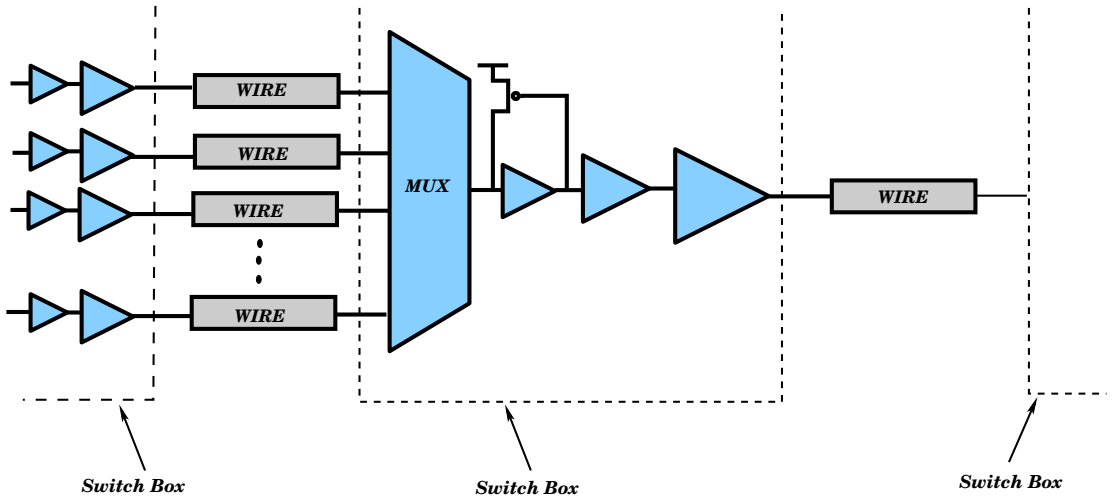

Figure 3.1: Global routing multiplexer and driver circuit [3, 4]

3.1 Circuit of Global and Local Interconnection Network

Based on single_driver routing architecture, which each wire transmits data in a single direction and also each wire is only driven by a single driver, multiplexers allow a variety of signals to access the routing driver. Multiplexers are typically build of NMOS pass gates followed by level restoring buffer and multi_stage drivers as mentioned in Section 2.1 [3]. In the global routing (switch box) as shown in Figure 3.1, the inputs of the multiplexers come from the other wire segments and the driver transmits the multiplexed signal through the wire to the other switch boxes. Figure 3.2 shows the transistor_level circuit of the multiplexer and the routing buffers in commercial FPGAs.

In the local routing, which is used within each CLB as mentioned in Section 2.1, the inputs of the multiplexers come directly from the CLBs input and/or output buffers and the multi_stage local routing driver transmits the multiplexed signal to the LUTs input inside each LEs as shown in Figure 3.3. The transistor_level circuit of the multiplexer and the local routing buffers of the local routing in commercial FPGAs is shown in Figure 3.4. In the local routing as it is obvious from the Figure 3.4, the wire length between the input drivers and the NMOS pass gates is short. Therefore, the wire does not introduce any significant loading and has no effect on the signal strength comeing out of the driver.

As mentioned in Section 2.1, in designing both global and local routing circuits,

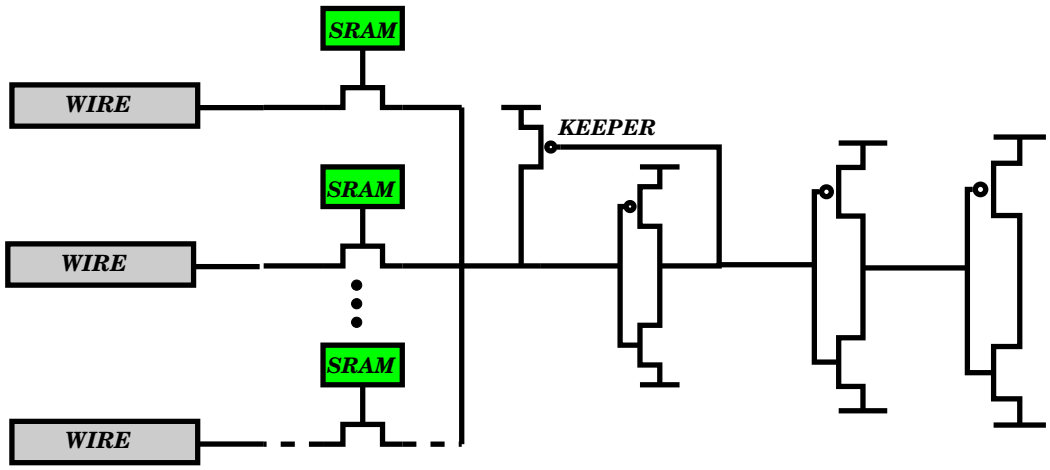

Figure 3.2: Transistor-level implementation of global routing multiplexer and driver

the use of NMOS pass gates in designing the multiplexers has the drawback of a threshold voltage drop when passing a high voltage. That is, an NMOS pass transistor with a gate voltage of V_{DD} is unable to pass a signal at V_{DD} from source to drain. The weak high voltage signal prevents the PMOS transistor of the downstream buffer from turning fully off, generating static power consumption. This problem can be mitigated by the use of a PMOS pull-up transistor, called a *Keeper* as shown in Figure 3.2 and 3.4. However, the circuit is ratioed and it adds the circuit complexity since the Keeper is fighting against the driver during a high_to_low transition. To circumvent this negative effect, the Keeper is made weak by increasing its length [19]. In fact, there is a trade-off in choosing the length of the Keeper; increasing the lenght of the Keeper also increases the time to restore the high voltage resulting more power consumption while decreasing its lenght increases the high_to_low transition time. Since in FPGAs the delay is mainly caused by the routing, designing a high performance routing network is one of the challenges facing integrated circuit designers [6].

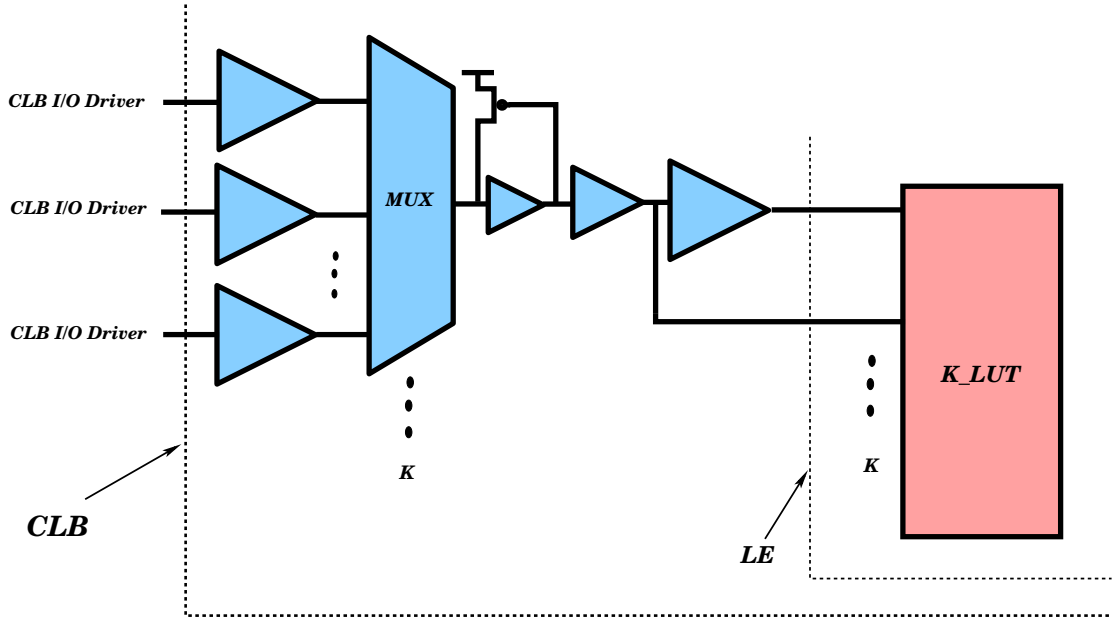

Figure 3.3: Local routing multiplexers and drivers circuit

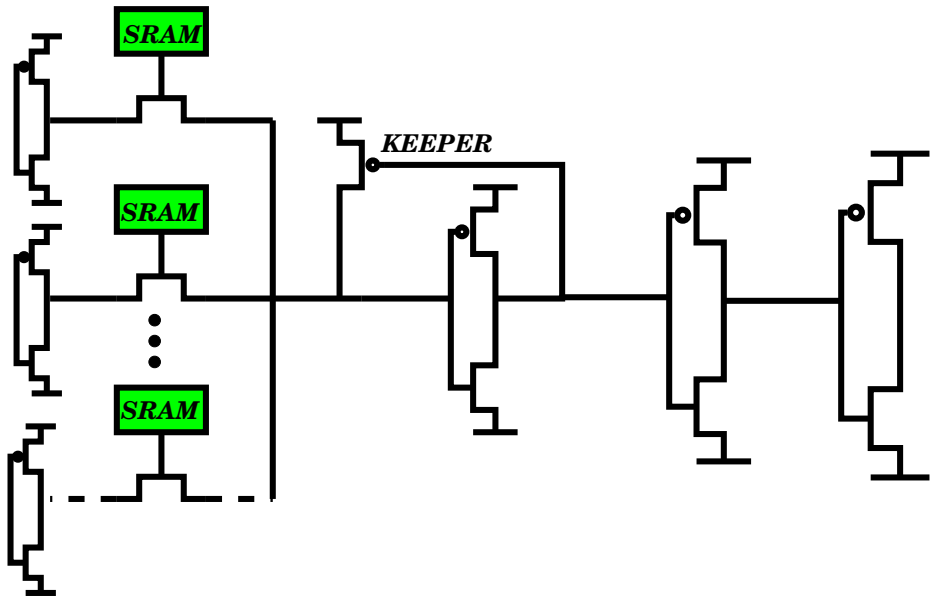

Figure 3.4: Transistor-level implementation of the local routing multiplexers and drivers

3.2 Capacitive Boosting for Global and Local Routing

To improve the performance of FPGA interconnection network, we propose to use capacitive boosting techniques rather than pure static solutions (such as, boosting the gate using dual V_{DD} [5, 8], or unfolding the multiplexer [1]). This is achieved by connecting a minimum-size Isolator, between the configuration memory cell and the pass transistor gate. The straightforward application of the bootstrap technique for FPGA routing is presented in Figure 3.5. When passing logic zero, the potential of the pass-transistor gate equals V_{DD}-V_P because of leakage and a threshold voltage loss across the Isolator transistor. When the Line Driver outputs a rising edge (also referred to as a pull-up transition), the pass-transistor gate rises above V_{DD}. Due to capacitive coupling, the gate potential immediately after a pull-up transition equals (assuming no parasitic capacitance connected to the pass-transistor gate node) $2V_{DD}$-V_P. This is because during rising edge the value of V_{DD} is coupled to the pass-transistor gate with the potential of V_{DD}-V_P resulting $2V_{DD}$-V_P. Depending on the operation history and leakage level, the pass-transistor gate potential can take any value from (V_{DD}-V_P) to ($2V_{DD}$-V_P). When the Line Driver outputs a falling edge (also referred to as a pull-down transition), the pass-transistor gate goes below V_{DD}-V_P also due to capacitive coupling. In this case, the Isolator turns *on*, supplies the pass-transistor gate with charge and brings the gate potential back to V_{DD}-V_P. Since the output of the pass-transistor multiplexer is driven to the full voltage swing level even before the Keeper transistor turns *on*, the signal transition is accelerated, and the short-circuit current of downstream gates is cut off.

There are two major problems with the Bootstrap configuration in Figure 3.5. First, the pull-up transition induces a large voltage on the gate, which may affect the gate oxide integrity. Second, depending on the leakage level, the pass-transistor gate voltage may drop back to V_{DD}-V_P. In this case, a pull-down transition induces a negative voltage spike on the gate, which slows down the pull-down transition. Although one can try to avoid pull-down transitions at a gate potential of V_{DD}-V_P [35], this might not always be possible on critical paths.

To limit these voltage overshoots and undershoots, we propose to incorporate the Isolator transistor into the SRAM cell, and deploy a PMOS Trimmer transistor in parallel with the Isolator as shown in Figure 3.6. A limitation in voltage variation occurs due to the parasitic capacitance at the gate node (C_{DB}, C_{SB}, and C_{GS} for

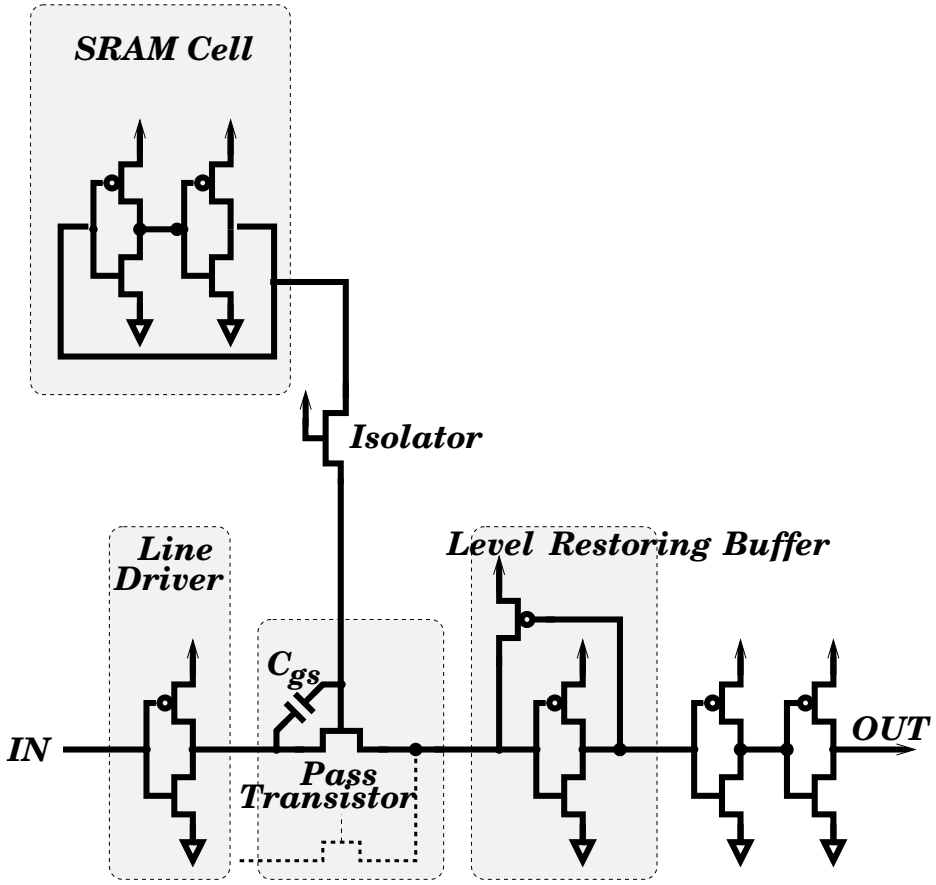

Figure 3.5: Straightforward application of the bootstrap technique to FPGA interconnect.

Isolator, Trimmer, and SRAM NMOS), which forces a charge redistribution in the pass transistor parasitic capacitance C_{GS} during pull-up and pull-down transitions. Having the Isolator incorporated into SRAM cell also reduces the positive spikes in gate voltage when the pass transistor is *off*. This beneficial effect is due to the fact that the SRAM NMOS drives the pass transistor gate directly and any positive spike will be absorbed by the SRAM NMOS. However, the designer has always got the option to reduce the parasitic capacitance at the gate node by extracting the Isolator transistor back from the SRAM cell, like in the original configuration shown in Figure 3.5.

Following a pull-up transition, the Trimmer restores the pass-transistor gate voltage back to V_{DD}, which definitely helps the incoming pull-down transition. The waveforms for the standard FPGA interconnect, the standard bootstrap interconnect, and the proposed bootstrap interconnect are presented in Figure 3.7. It is apparent that,

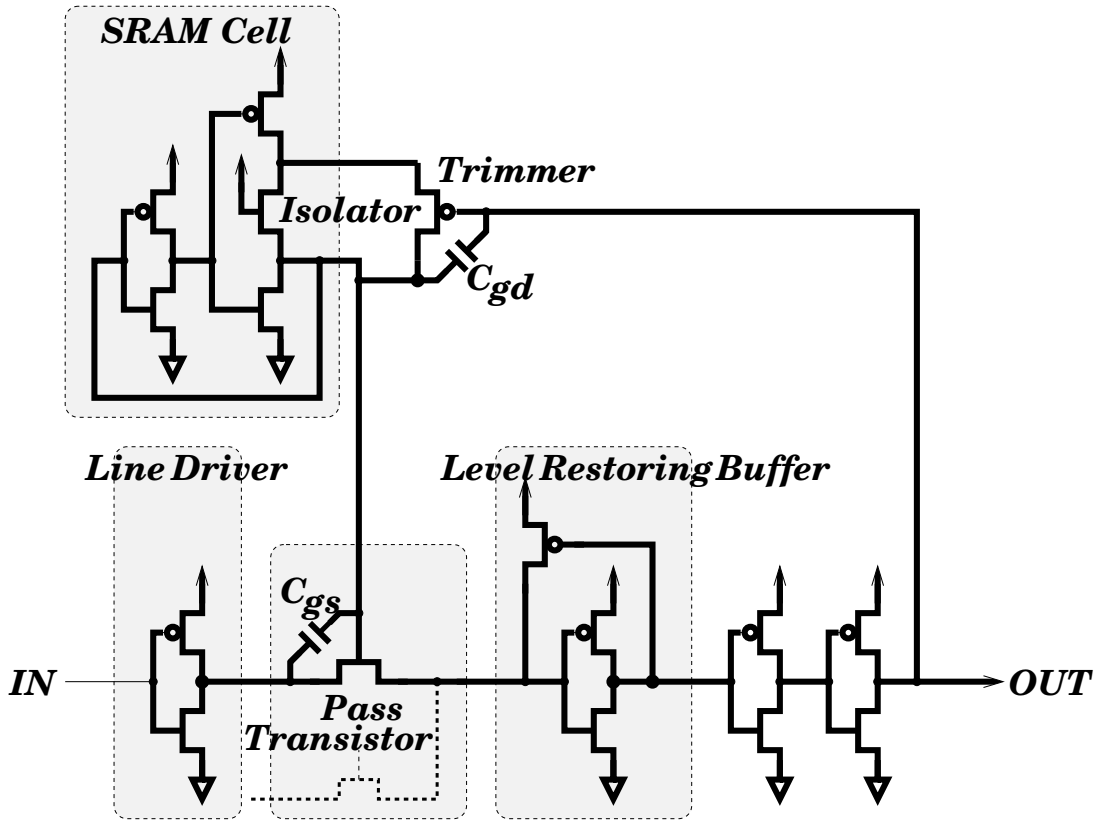

Figure 3.6: Bootsrap improved

due to the PMOS Trimmer, the gate voltage is restored back to V_{DD} after the high signal has propagated through the level-restoring buffer all the way to output. The Trimmer also limits the magnitude and the duration of the transitory increase of the pass transistor's gate voltage, with beneficial effects for the gate oxide integrity.

In addition, after the low signal level has fully propagated through the multiplexer and level-restoring buffer, the Trimmer starts turning *off*. Due to the capacitive coupling from the Trimmer's gate to the Trimmer's drain, a beneficial second bootstrap effect will push the restored gate voltage slightly above V_{DD}. It is worth emphasizing that such a voltage overshoot during a pull-down transition is never possible in the standard bootstrap configuration. Moreover, since the the Trimmer turns *off* after the '0' signal has propagated through the level-restoring buffer all the way to output, it has no longer an influence on the propagation time during a pull-up transition.

Our bootstrap configuration with an NMOS Isolator and a PMOS Trimmer resembles a CMOS transmission gate. Although transmission gates have been recently proposed [28], using transmission gates as switches increases the capacitance along

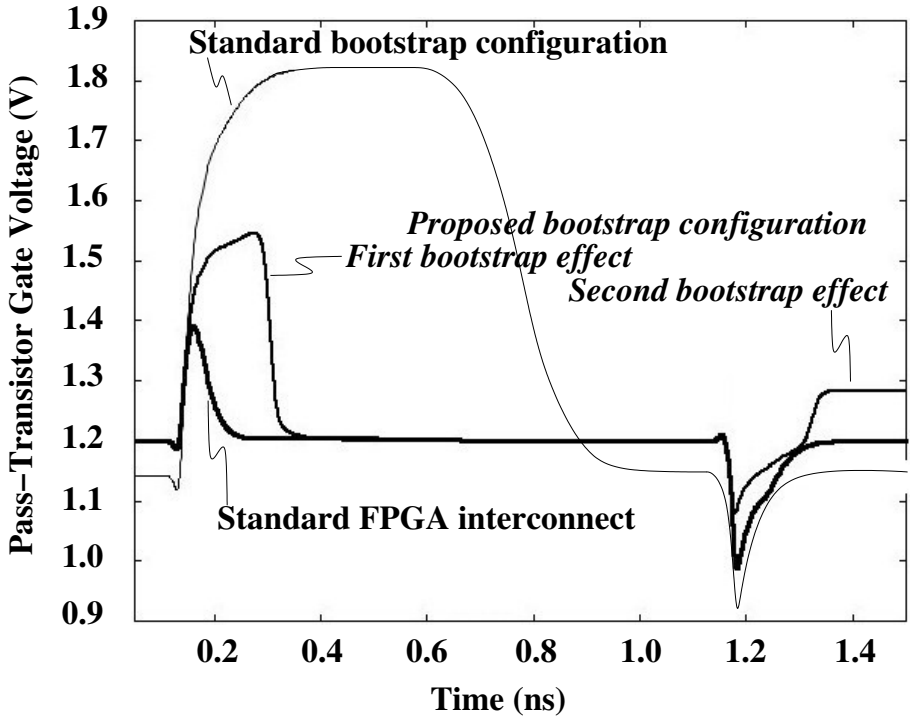

Figure 3.7: Signal waveforms at the pass-transistor gate.

the signal propagation path; thus, larger buffers must be provided for driving both the NMOS and PMOS transistors of the full transmission gate [40, 20, 23]. In the proposed circuit, however, the Isolator-Trimmer transmission gate is connected to the pass-transistor gate. As a result, our circuit technique introduces only a minimum parasitic capacitance to the signal propagation path. In addition, the Isolator and Trimmer are both minimum size transistors. At layout level, these transistors can share their sources with the SRAM cell; thus they introduce only a small additional silicon area.

3.3 Simulation Framework and Results

There are a number of attributes (such as the size of the multiplexer and wire drivers) in desigining routing networks. Lee et al. investigated a range of possible input numbers per single-driver routing switch in terms of delay and area-delay [9]. For delay optimization, it was determined that the fastest single_driver switch contains a 4:1 multiplexer. A single_driver switch with an 8:1 multiplexer was determined to be optimal for area_delay product (ADP). In [3], it was determined that a series of

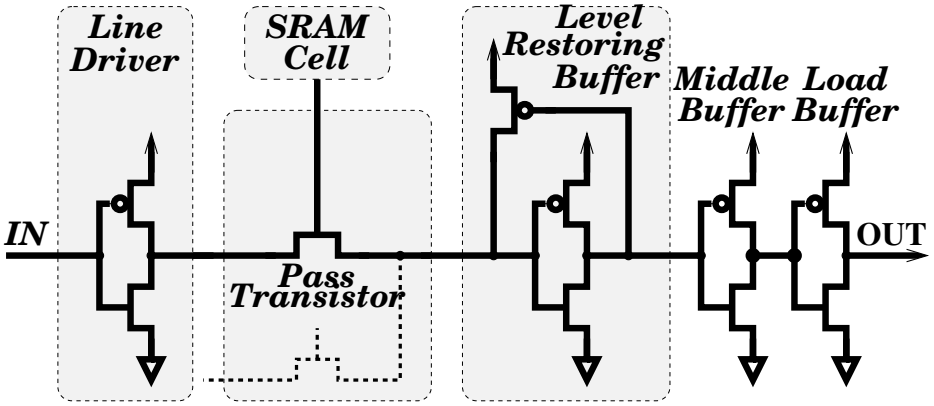

Figure 3.8: Basic components of the circuit under the test.

three inverters, rather than two or four, is desirable in terms of delay and area. In this work, we considered 8:1 multiplexers built with a single layer of NMOS pass transistors. Therefore, the circuit under test comprises one line driver, eight pass transistors together with eight configuration SRAM cells, one level-restoring buffer, one keeper, one intermediate buffer, and one end buffer as shown in Figure 3.8. The bootstrap circuit also includes eight Isolators and eight Trimmers. The transistor sizes are presented in Table 3.1 for both scenarios (pass transistors with minimum size and optimal size). In the standard circuit, the level restoring buffer is skewed to be able to switch earlier for rising transition because of voltage drop in NMOS device. The additional freedom degree that a designer has in a bootstrap configuration versus the standard configuration is the skew of the level-restoring buffer. In standard configuration, the level restoring buffer is skewed due to the need to balance the rising and falling transition delay times. A non-skew level restoring buffer has better noise immunity but unbalanced transitions. In a bootstrap configuration, the rising (pull-up) transition is significantly improved by the pass transistor gate voltage above V_{DD}. Therefor, the level restoring buffer is non-skewed or slightly skewed increasing noise immunity.

Table 3.1: Component sizes (nm). All transistors are of minimum length with the except of Keeper.

Module ↓ \ Technology →	180nm	130nm	90nm	65nm
Common figures to both minimum and optimum size pass transistors				
Line Driver (pMOS/nMOS)	3,000 / 1,500	2,000 / 1,000	1,500 / 750	1,500 / 750
Keeper (Width/Length)	220 / 360	160 / 240	120 / 180	120 / 120
Middle Buffer (pMOS/nMOS)	1,760 / 880	1,280 / 640	960 / 480	960 / 480
Load Buffer (pMOS/nMOS)	7,040 / 3,520	5,120 / 2,560	3,360 / 1,680	3,840 / 1,920
Isolator	220	160	120	120
Trimmer	220	160	120	120
Minimum size pass transistors				
Pass Transistor – standard and bootstrap	220	160	120	120
Level Restoring Buffer – standard (pMOS/nMOS)	440 / 440	320 / 320	240 / 240	240 / 240
Level Restoring Buffer – bootstrap (pMOS/nMOS)	660 / 260	400 / 160	360 / 160	300 / 150
ADP optimal size pass transistors				
Pass Transistor – standard	840	640	400	300
Pass Transistor – bootstrap	400	200	200	200
Level Restoring Buffer – standard (pMOS/nMOS)	440 / 340	320 / 200	240 / 150	240 / 240
Level Restoring Buffer – bootstrap (pMOS/nMOS)	600 / 300	408 / 180	360 / 160	360 / 160

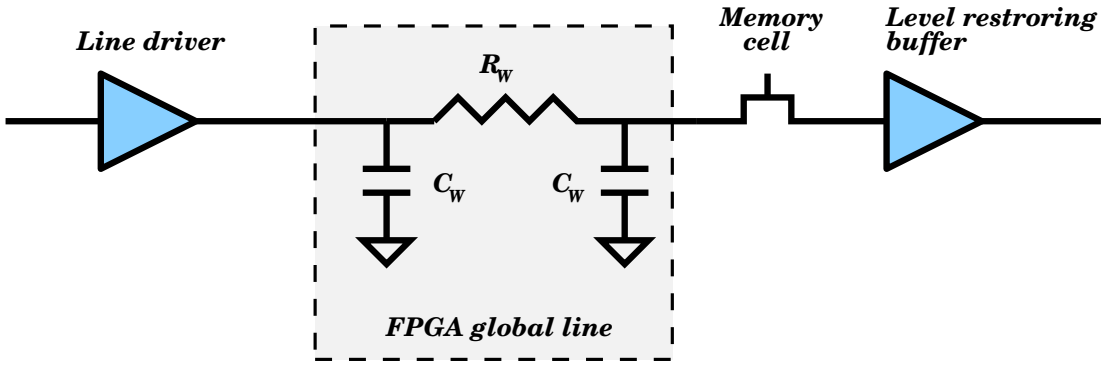

Figure 3.9: The long wire model of an FPGA line

We have created Spice netlists for both the standard and bootstrap level-restoring buffers and for local (that is, without a wire model) and global (that is, with a wire model) interconnects. Because the interconnection network is the dominant silicon area consumer, and current FPGA implementations are typically limited in use by their large propagation delay and power consumption [41], three common metrics are used to evaluate an FPGA implementation: silicon area, propagation delay, and power consumption. Two composite metrics are also widely used in FPGA chip design: the area-delay product [5, 8] to ensure that the delay of the interconnect is not sacrificed to obtain the minimum area or vice-versa, and power-delay product [32] to ensure that the delay of the interconnect is not sacrificed to obtain the minimum power consumption.

All circuit designs, netlists, and simulations were completed with the Cadence IC5.1.41 tool suite [42] including Analog Artist and Spectre for standard 180nm, 130nm, 90nm, and 65nmtechnologies provided by Canadian Microelectronics Corporation (CMC) and MOSIS with 1.8V, 1.2V, 1.2V, and 1.0V supply voltages, respectively. The same set of simulations were completed both with and without a long wire model to emulate a global interconnect, and a local interconnect, respectively. The long wire model of the FPGA line is shown in Figure 3.9 and the long wire model parameters are presented in Table 3.2. The capacitance and resistance numerical figures were obtained with the tools and methods we have previously used [1]. The results are reported according to area, propagation delay, and power consumption metrics, and also the area-delay and power-delay products composite metrics.

The delay is measured from the line driver input to the level-restoring buffer output with respect to the midpoint between supply and ground (1-to-0 and 0-to-1 in Tables 3.3a, 3.3b, 3.4a, and 3.4b refer to pull-up and pull-down transitions at

Table 3.2: FPGA wire model parameteres [1]

Technology	Rw (Ω)	Cw (fF)
180nm	144	15.0
130nm	180	10.7
90nm	154	8.4
65nm	154	8.4

the pass-transistor source/drain, respectively). The power consumption is calculated by integrating the voltage-current product over a normalized period of time. To estimate the area, we use the model proposed by Lemieux and Lewis [8]. For global interconnect, eight minimum size transistors (one per MUX input) are added to the total area as long wiring overhead. The resulting area figures are normalized to the minimum transistor area in each technology.

Lee et al. and Hung et al. consider all transistors in the routing multiplexers to be of minimim size [10, 4]. For this assumption, the simulation results are available in Tables 3.3a, and 3.3b. It is apparent that an improvement in propagation delay of 12-20% is achieved across all considered technologies for both local and global interconnects. With the exception of 180nm technology, the area-delay product slightly decreases; this shows that the area penalty is a good trade-off for the delay improvement. Similar to propagation delay, the power-delay product also exhibits a consistent improvement across all considered technologies, showing the effectiveness of the capacitive boosting in multiplexers built with minimim-size transistors.

Lemieux and Lewis indicate that a pass transistor larger than minimum size is needed if a level-restoring buffer is used [8]. Kuon and Rose analyze the situation when the size of the multiplexer transistors is optimized [30]. For this reason, the next set of simulations consider non-minimum size pass transistors that lead to an optimum area-delay product. In Tables 3.4a and 3.4b it is apparent that an improvement in propagation delay of 10-17% has been achieved across all four technologies and for both local and global interconnects. The area-delay product decreases slightly, which means the area penalty is acceptable for the obtained delay improvement. The power-delay product also exhibits a significant improvement of at least 17% for local interconnect, and 12% for global interconnect.

An interesting comparison can be made between the standard interconnection network built with optimum-size pass transistors and the bootstrap interconnection network built with minimum-size pass transistors. In Tables 3.3a, 3.3b, 3.4a, and 3.4b it is apparent that, with the exception of the 65nm technology, the bootstrap

configuration leads to a smaller area-delay product. The power-delay product is smaller in the bootstrap configuration in all considered technologies. As a result, using minimum-size switches with capacitive boosting rather than standard optimum-size switches is a serious circuit design technique that should be further investigated. Together with the previous results, this analysis indicates that capacitive boosting provides an improvement in the performance of the optimized standard interconnection network, making it a very promising circuit technique for FPGA design.

Since the voltage on the pass transistor gate is higher than nominal, the capacitive boosting approach can potentially affect the gate oxide integrity (especially for newer technologies), and, therefore, may reduce the device reliability. Due to the Trimmer, the overshoot impulse applied on the pass transistor gate has an amplitude of 0.4V and a short duration of 100 picoseconds, as shown in Figure 3.7. It is still an open question whether such an impulse stress can affect the device reliability. However, we are optimistic; for example, according to Mutlu and Aminzadeh, the DC equivalent voltage of a pulse train with an amplitude of 0.4V and frequency of 2GHz is equal to 50mV [43], a value that is well supported by all current technologies. Additionally, it is only 5% of 1V. It should also be noted that the magnitudes of all gate-to-source, drain-to-source, and gate-to-drain voltages across the pass transistor do not experience a level in excess of V_{DD} during the pull-up transition. This is beneficial for device reliability since normally each of the terminals of the pass transistor can support a maximum gate-to-source, drain-to-source, and gate-to-drain voltage of V_{DD} without breakdown or deterioration.

(a) Experimental figures for local interconnect (without line model).

Technology and Circuit	180nm			130nm			90nm			65nm		
	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)
0-to-1 (ps)	143	123	-14.0	131	97	-25.6	52	37	-28.8	51	35	-31.4
1-to-0 (ps)	153	133	-13.1	116	104	-10.3	53	47	-11.3	41	39	-4.9
Average Delay (ps)	148	128	-13.5	124	101	-18.5	53	42	-20.8	46	37	-19.6
Area (normalized)	87	103	+18.4	97	113	+16.5	97	113	+16.5	97	111	+14.4
Power (μW)	464.7	464.3	-0.1	102.4	102.3	-0.1	64.2	64.1	-0.2	40.7	40.6	-0.3
Area-Delay (ps)	12,876	13,184	+2.4	12,028	11,413	-5.1	5,141	4,746	-7.7	4,462	4,107	-8.0
Power-Delay (fJ)	68.8	59.4	-13.7	12.7	10.3	-18.9	3.4	2.7	-20.6	1.9	1.5	-21.1

(b) Experimental figures for global interconnect (with line model).

Technology and Circuit	180nm			130nm			90nm			65nm		
	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)
0-to-1 (ps)	151	132	-9.5	139	106	-23.7	56	42	-25.0	57	42	-26.3
1-to-0 (ps)	170	150	-11.8	131	119	-9.2	61	55	-9.8	50	47	-6.0
Average Delay (ps)	161	141	-12.4	135	113	-16.3	59	49	-16.9	54	45	-16.7
Area (normalized)	95	111	+16.8	105	121	+15.2	105	121	+15.2	105	119	+13.3
Power (μW)	796.5	795.5	-0.1	184.6	184.5	-0.1	120.2	120.0	-0.2	73.9	73.7	-0.3
Area-Delay (ps)	15,295	15,651	+2.3	14,175	13,673	-3.5	6,195	5,929	-4.3	5,670	5,355	-5.6
Power-Delay (fJ)	128.2	112.2	-12.5	24.9	20.8	-16.5	7.1	5.9	-16.9	4.0	3.3	-17.5

Table 3.3: Experimental figures for local (without line model) and global interconnect (with line model) using minimum-size pass transistors.

(a) Experimental figures for local interconnect (without line model).

Technology and Circuit	180nm			130nm			90nm			65nm		
	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)
0-to-1 (ps)	135	105	-22.2	116	92	-20.7	47	34	-27.7	41	31	-24.4
1-to-0 (ps)	139	129	-7.2	108	97	-10.2	46	43	-6.5	36	34	-5.6
Average Delay (ps)	137	117	-14.6	112	95	-15.2	47	39	-17.0	39	33	-15.4
Area (normalized)	109	117	+7.3	108	113	+4.6	105	115	+9.5	103	113	+9.7
Power (μW)	482.1	471.2	-2.3	108.9	103.6	-4.9	67.7	66.0	-2.5	41.6	40.0	-3.8
Area-Delay (ps)	14,933	13,689	-8.3	12,096	10,735	-11.3	4,935	4,485	-9.1	4,017	3,729	-7.2
Power-Delay (fJ)	66.0	55.1	-16.8	12.2	9.8	-19.7	3.2	2.6	-18.8	1.6	1.3	-18.8

(b) Experimental figures for global interconnect (with line model).

Technology and Circuit	180nm			130nm			90nm			65nm		
	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)	Standard	Bootstrap	Δ (%)
0-to-1 (ps)	142	114	-19.7	125	102	-18.4	52	39	-25.0	48	38	-20.8
1-to-0 (ps)	153	146	-4.6	120	112	-6.7	53	51	-3.8	44	43	-2.3
Average Delay (ps)	148	130	-12.2	123	107	-13.0	53	45	-15.1	46	41	-10.9
Area (normalized)	117	125	+6.8	116	121	+4.3	113	123	+8.8	111	121	+9.0
Power (μW)	810.6	803.2	-0.9	191.5	185.4	-3.2	122.3	120.0	-1.9	74.6	73.1	-2.0
Area-Delay (ps)	17,316	16,250	-6.2	14,268	12,947	-9.3	5,989	5,535	-7.6	5,106	4,961	-2.8
Power-Delay (fJ)	120.0	104.4	-13.0	23.6	19.8	-16.1	6.5	5.4	-16.9	3.4	3.0	-11.8

Table 3.4: Experimental figures for local interconnect (without line model) and global interconnect (with line model) using optimal ADP-size pass transistors

3.4 Summary

In this chapter, we first described the detailed design of an FPGA global and local interconnection network. We have proposed a level-restoring buffer based on the capacitive boosting effect. We used the *Isolator* between the configuration memory cell and the pass transistor gate to boost the gate voltage above V_{DD}. By deploying the *Trimmer*, the drawbacks of the conventional boosting technique were mitigated; the Trimmer limits the magnitude and the duration of the transitory increase of the pass transistor's gate voltage, with beneficial effects for the gate oxide integrity. The *Trimmer* also improves the pull-down transition delay by restoring the pass transistor's gate voltage back to V_{DD} when a pull-down transition imposes a nagative spike on the pass transistor's gate. The simulations indicate a reduction of at least 10% in propagation delay for the proposed circuit versus the standard one across 180nm, 130nm, 90nm, and 65nm technologies. It should be noted that these enhancements are obtained without additional supply voltages or low- or zero-threshold devices. As mentioned, the penalty of our approach is a slightly increased silicon area requirement for the circuitry. Given the fact that the area-delay product does not increase in the bootstrap circuit with respect to the standard one, we can say that the circuit design technique we propose is a viable alternative in building performant level-restoring buffers. Equally important is that the circuit technique we propose gives the designer a higher degree of flexibility in choosing the trade-off between area, propagation delay, and power consumption, that the prior level-restoring buffers do not provide.

Chapter 4

Capacitive Boosting in Weak Inversion for FPGA Interconnection Networks

The growth of portable applications such as cellular phones, laptop computers, biomedical devices like hearing aid, and wireless receivers has caused many efforts to decrease the energy and/or power consumption. In these energy-constrained applications, *ultra-low power (ULP)* consumption has been the primary requirement while speed is of secondary consideration. For these applications, operating in sub-threshold region has been proposed to save significant power consumption [12, 13, 14].

An Application-Specific Integrated Circuit (ASIC) is an energy efficient solution for low power circuits since it can be customized for a particular use. However, the inability to change the hardware makes it impossible to be reused for other applications. The flexibility of FPGAs together with sub-threshold operation would be an excellent synergy if it is technically possible. An FPGA offers hardware performance together with the flexibility to reconfigure it for other applications. The device's flexibility would reduce time to market for emerging ULP products. This is why sub-threshold FPGA, where power consumption is reduced by operating in sub-threshold region, would be a promising option for low power applications. However, sub-threshold FPGA design has major challenges. The speed of the FPGA in sub-threshold is important since it may limit the application areas of sub-threshold FPGAs. Since FPGAs routing components have a major effect on FPGAs speed [7], our main contribution in this chapter is to explore and propose a circuit design

technique to overcome the difficulties related to operating in sub-threshold FPGAs.

In this chapter, we first summarize the sub-threshold operation concepts. We also present the challenges that designers interested to work with sub-threshold circuits are involved. In addition, we next discuss the sub-threshold FPGA design and present the challenges facing sub-threshold FPGA design. Finally, we propose a sub-threshold FPGA interconnection network using capacitive boosting and present the simulation results.

4.1 Sub-threshold Operation

It is apparent in Equation (4.1) that in any digital circuit, switching (dynamic) energy scales quadratically with supply voltage [14]:

$$E_{\text{DYN}} = CV_{DD}^2 \tag{4.1}$$

where C is the total switched capacitance. Due to this quadratic dependence, by reducing the voltage supply below the threshold voltage (sub-threshold region), the switching energy/power consumption is significantly reduced.

In order to operate in the sub-threshold region, the circuit is supplied with a voltage, V_{DD}, that is less than the threshold voltage of the transistor, V_{th}. As a result $V_{GS} < V_{th}$. In this case, the channel is not strongly inverted but only weak inverted. Equation (4.2) gives a simple model for the sub-threshold drain current [14].

$$I_{D\text{sub-th}} = I_o \exp\left(\frac{V_{GS} - V_{th} + \eta V_{DS}}{nV_T}\right)\left(1 - \exp\frac{-V_{DS}}{V_T}\right) \tag{4.2}$$

where n is the sub-threshold slope factor, $V_T = kT/q$ is the thermal voltage, and I_o is the drain current when $V_{GS} = V_{th}$ and η represents the drain-induced barrier lowering (DIBL) coefficient [14].

Even small voltage difference between the drain and source of the transistor causes some of energetic carriers at source to enter the MOSFET channel and flow to the drain. As a result, the sub-threshold drain current, $I_{D\text{sub-th}}$, depends exponentially on the gate-to-source voltage drop, V_{GS}, and threshold voltage, V_{th}, as it is apparent in Equation (4.2) .

4.2 Challenges for Sub-threshold Circuit Design

There are a number of issues related to CMOS logic operating in sub-threshold (also referred to as weak inversion). Since the current is significantly reduced as compared to strong inversion, the speed is slow. For example, a minimum-size inverter has a delay in the scale of ns in 130nm technology. Therefore, this operation region is inappropriate for high speed applications. In addition, the propagation delay increases exponentially with additional supply voltage reduction. What is specific to sub-threshold operation is that the leakage current integrates over the longer delay until leakage energy per operation exceeds the active energy. This means that we design and optimize the circuit in sub-threshold in a different way from the super-threshold. Second, a sub-threshold CMOS digital circuit exhibits a reduced *ION/IOFF* ratio that translates functionality problems and lower immunity to the noise. Third, according to Equation (4.2), the transistor drain current and then its functionality are sensitive to *process-voltage-temperature (PVT)* variation. Variation in the sub-threshold can make *ION* so small and *IOFF* so large that operation failure of CMOS gates may occur [13, 14, 44]. In addition, in strong inversion NMOS devices are stronger than PMOS devices with the same size due to the higher mobility of electrons relative to holes. However, this is not the case for sub-threshold circuits and PMOS devices can be stronger than NMOS devices [45]. This means that the circuit designs must be independent of having stronger PMOS devices or stronger NMOS devices and do not need redesign based on the technology.

Given the above mentioned considerations, a number of design recommendations and warnings can be stated. First, circuits that use a topology that decreases *ION* compared to *IOFF* should be avoided [13, 14, 44]. For example, large transistor stacks and many parallel leakage paths decrease *ION* compared to *IOFF*. Also, in the presence of variation, sizing is a weak knob in sub-threshold; as a result, correct operation cannot be guaranteed by sizing [7]. This is because sizing has a linear effect on the drain current while the effect of PVT is exponential. As a result, ratioed circuits should be avoided. For example, in 6T SRAM cell, sizing is used to provide proper functionality in strong inversion while it presents functionality problems because of sensitivity to V_{th} variation in sub-threshold region [7]. Our effort instead will focus in increasing the gate voltage by capacitive boosting.

Previous works made attempts to mitigate the difficulties of designing sub-threshold circuits. In simple static CMOS gates, short stacks (e.g., less than four series transis-

tors) have been used [44, 7]. In addition, upsizing the stacked transistors is necessary to provide robustness at the expense of increased area and energy consumption [44]. To reduce the effect of variation on SRAM cell and therefore improve its robustness, different circuit topologies and design methodologies (e.g., using 8T or 10T SRAM cell, Schmitt trigger circuit, increasing the cell read current, and decreasing the leakage current) has been proposed [46, 47]. As mentioned by Calhoun [7], the supply voltage is a strong knob to combat variation because of exponential impact of V_{GS} on current (Equation (4.2)). Increasing V_{DD} slightly can have a significant effect on robustness and speed at the expense of an increase in energy consumption [14].

4.3 Sub-threshold FPGA Design Challenges

FPGA devices operating in weak inversion have recently emerged [7]. A sub-threshold FPGA design faces a combination of sub-threshold circuit challenges and problems inherent to FPGA architectures. There are three major challenges for sub-threshold FPGA design. First, variation is an issue that occures in any other sub-threshold circuit. Variation has effect on the functionality of logical elements in FPGA. Variation has also an effect on FPGAs interconnection structures. One solution to reduce the effect of variation on logic resources is to address this problem in place and route step. Such that, the synthesized design will avoid placing critical paths on slow gates affected by variation. However, this approach requires test data from the FPGA die to be available during the synthesis process, which is not always feasible [7].

Second, interconnect network form an important part of FPGA architecture in terms of delay and power consumption. In sub-threshold region, transistor drive current decreases whereas wire capacitance remains basically the same. These capacitors and the variation in interconnect drivers cause large variations in propagation delay which means that designing the interconnect network in sub-threshold FPGAs is a challenge. In addition, the leakage from off branches in switch boxes and connection boxes (Figure 3.2) increasing the IOFF and decreasing *ION/OFF* ratio. Since series-connected pass transistors lead to very poor output swing and speed in sub-threshold, repeaters should be deployed in every switch boxes. However, the delay and energy of the interconnection network are still an issue and dominate the FPGA metrics in sub-threshold region [7]. Therefore, designing a reliable, low power and fast interconnect is a major challenge in sub-threshold FPGA design. This issue is addressed in the next section. Our goal is to improve the FPGA interconnection network in terms

of *speed* and *area-delay product (ADP)* under sub-threshold conditions.

Additionally, clock network drives all of the registers and flip-flops across the entire FPGA fabric. Driving the large capacitive load of the clock network and the variation in the clock drivers can result in a large clock skew. A high-speed and power efficient method to reduce the clock skew is necessary in sub-threshold FPGA.

Third, SRAM memory is also a major design challenge for sub-threshold FPGA. The memory is used in registers and LUTs and consist of SRAM bit cells embedded into the CLBs. Moreover, a large amount of SRAM cells configure multiplexors inside switch boxes and connection boxes. These SRAM cells provide configurability and require write access only during configuration and can be performed in strong inversion. However, the read access is required during FPGA operation, which means that data retention and low leakage design are more important for them [7].

The standard circuits of the global and local routing for super-threshold operation (Figure 3.2 and 3.4) are not appropriate for sub-threshold operation. This is mainly because connecting NMOS passgates in series lead to very poor output swing and speed in sub-threshold. Having poor output swing makes the circuit asymptotically slow and sensitive to variation and noise. Therefore, the propagation delay has a long-tail distribution as shown in Figure 4.1 and therefore, the best that the designer can do in the design process is to consider the worst case scenario.

One option to improve signal swing and speed is to use transmission gates in implementing switch boxes, as proposed by Calhoun [7]. However, tristate buffers are required in every switch box to mitigate variation, and this approach generates a large area overhead in the switch boxes. In a second, a sub-threshold FPGA using a low swing, dual-VDD interconnect scheme has been proposed [15]. Delay has been reduced due to the use of higher V_{DD} for interconnect, and reduced energy has been achieved by using pass gates in switch boxes and low swing signals. To improve tolerance to the variations, a single ended asynchronous sense amplifier has been used for the low swing interconnect. However, it still needs to use dual-rail V_{DD} and also requires architectural level changes such as changing the size of the logic clusters.

In addition, in sub-threshold, the wire resistance is negligible compared to the driver resistance because of the small current. That is, the driver resistance dominates the routing delay. It has been shown that the delay and energy consumption of the sub-threshold FPGA interconnection can be improved by optimizing and operating interconnect driver transistors in the near-threshold operating region, while the signal swing on the wire remains in sub-threshold region [16]. This has been achieved by

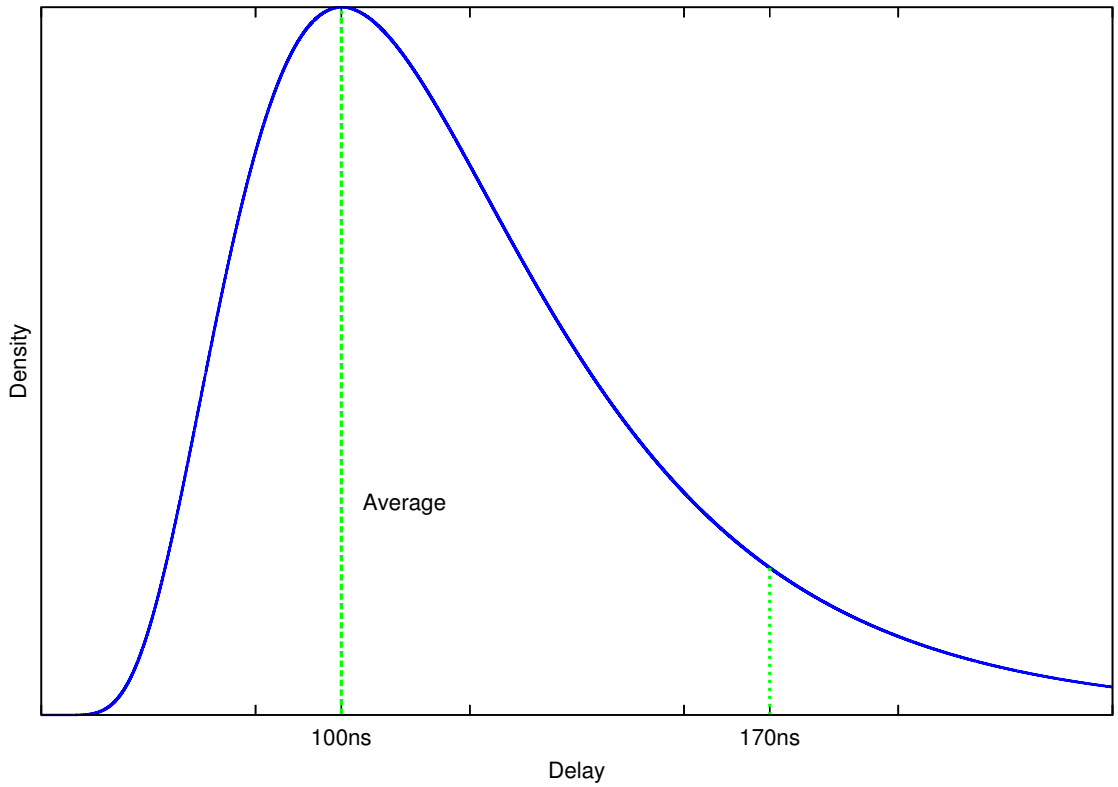

Figure 4.1: An example of the variations effect on the delay distribution

optimizing the threshold voltage and oxide thickness of driver transistors. Also, it has been observed that repeater insertion in long line and the use of multi-walled carbon nanotube (MWCNT) as an interconnect instead of the standard copper (Cu) will not provide any significant advantage in the delay or energy with respect to the super-threshold region [16].

Since the performance of the FPGA is dominated by the performance of its routing components, designing high speed and reliable line drivers and multiplexers used in switch boxes is necessary. In the next section, we propose the use of capacitive boosting in implementing the line drivers and multiplexers in sub-threshold FPGA interconnection network.

4.4 Capacitive Boosting for FPGA Interconnection Circuitry in Sub-threshold Region

Because of the exponential relationship between current and gate-to-source voltage in the sub-threshold region, capacitive boosting has been used to improve the circuit performance and robustness [2, 48, 49]. Capacitive boosting has also been used to design clock interconnect driver [2, 48]. The idea of boosting the gate voltage of the driver transistors is to shift the circuit operation region away from sub-threshold toward super-threshold resulting in improved speed and reliability. A negative capacitance effect has also been used to improve the boosting efficiency [50]. However, this requires a non-standard technology with ferroelectric insulator. Moreover, boosting technique has been used to improve driving ability of the read buffers for 6T SRAM cells [49]. To the best of our knowledge, capacitive boosting for sub-threshold FPGAs has never been attempted.

In this section, our intention is to improve the FPGA routing networks in terms of speed, area_delay product (ADP) and tolerance to PVT variations in sub-threshold region. We propose to use capacitive boosting in the design of switch box drivers and multiplexers.

4.4.1 Designing the Driver based on Capasitive Boosting

Since the driver delay is the main contributor to the interconnect delay in sub-threshold [7], and also due to the exponential relationship between the current and gate-to-source voltage in sub-threshold, we propose to use capacitive boosting in designing the line drivers in global routing. That way, we want to exploit the exponential dependence between the current and gate-to-source voltage to improve the speed and driving ability of the driver. The circuit is shown in Figure 4.2. By boosting the gate voltage of transistors in the *output stage* of the driver, operating region is shifted from sub-threshold to near-threshold region or super-threshold region. The proposed capacitive boosted driver improves the driving ability, and results an improvement in speed and enhancement in tolerance to PVT variations. When the input signal has a transition from the low_to_high, VL is boosted from 0 to -Δ V by the boosting circuit. N1 turns *on* and the gate of the output PMOS (P2) is driven by VL= $-\Delta$ V (instead of 0V). Hence, the drain current and driving ability of the P2 will increase to drive the long wire enhancing the speed and tolerance to variation. Similarly, when

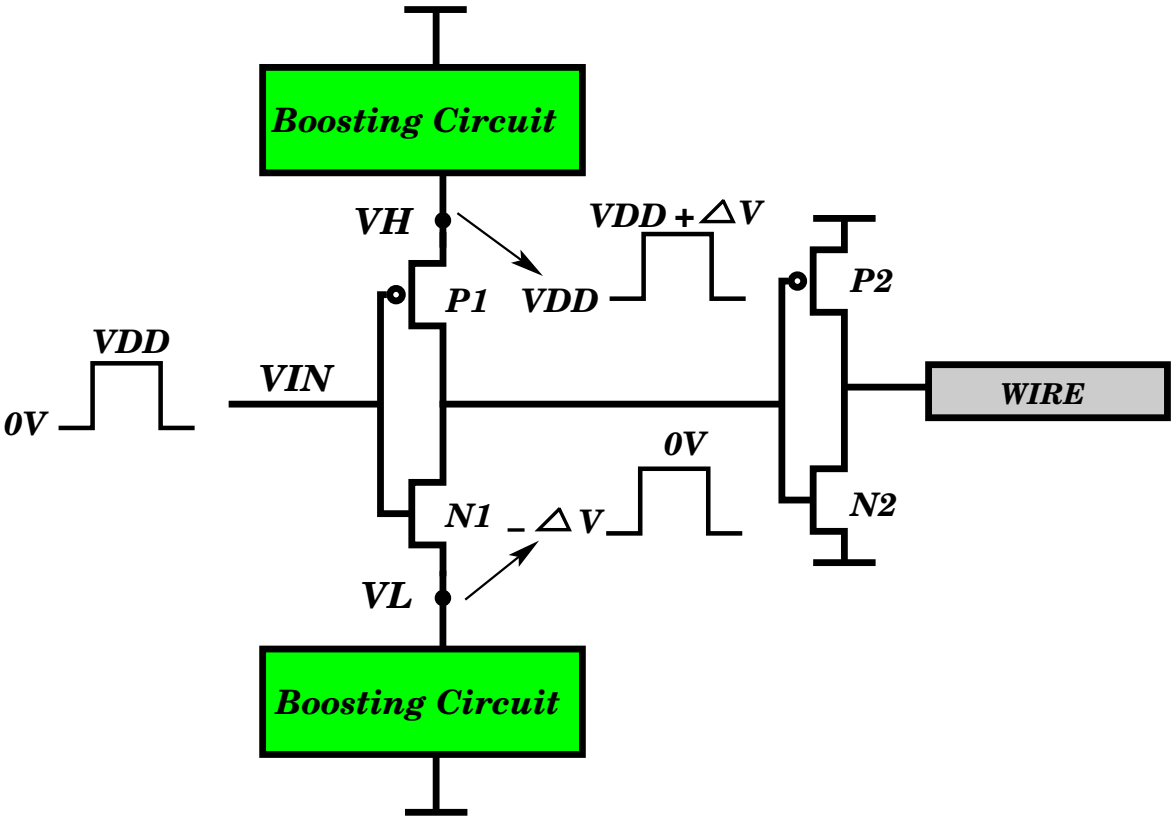

Figure 4.2: Operation principle of the proposed sub-threshold boosted driver

the input signal has a transition from the high_to_low, VH is boosted from V_{DD} to V_{DD}+Δ V by the boosting circuit. P1 turns *on* and the gate of the output NMOS (N2) is driven by VH= V_{DD}+Δ V (instead of V_{DD}). Therefore, N2 drives the long wire with increased current resulting improved speed and variation tolerance. Note that for the same amount of voltage boosted on P2 and N2, the increased current and driving ability is less in super-threshold. In fact, boosting has a significant effect on the current in sub-threshold due to exponential relationship between the current and gate-to-sournce voltage.

The circuit implementation of the boosted driver we propose is shown in Figure 4.3. If VIN is V_{DD}, N4 and P3 are *on* precharging VH to V_{DD}. When VIN has a high_to_low transition, VC1 has a low_to_high transition. Therefore, P3 turns *off* and VH is boosted from V_{DD} to V_{DD}+Δ V by C1. In this case, P1 is *on* and the gate of the N2 is driven by VH= V_{DD}+Δ V results in increased current and driving ability to drive the long wire. While passing zero, P5 and N3 are *on* and VL is precharged to 0V. When VIN has a low_to_high transition, VC2 has a high_to_low transition. As

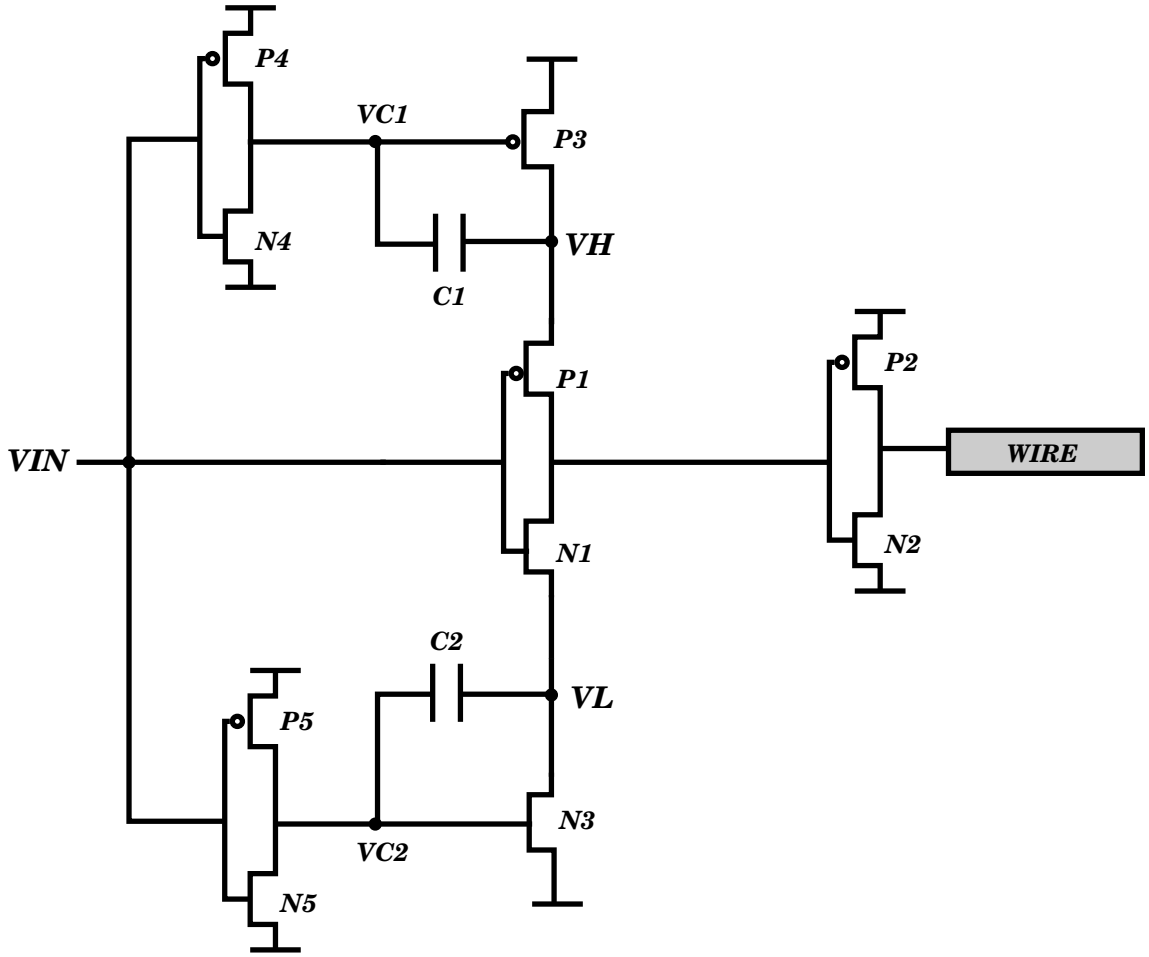

Figure 4.3: Circuit implementation of the proposed sub-threshold boosted driver

a result, N3 turns *off* and VL is boosted from 0 to -Δ V by C2. Because of VIN low_to_high transition, N1 turns *on* and P2 is driven by VL= $-\Delta$ V resulting improved current and speed to drive the long wire. In addition, in order to boost VC1 and VC2 as soon as possible when the transition comes, the inverter which drives VC1 is skewed to have a fast low_to_high transition. Similarly, the inverter which drives VC2 is skewed to have a fast high_to_low transition. The waveform of the proposed capacitive boosted driver is shown in Figure 4.4. Layout design of the proposed driver is shown in Figure 4.5. The layout was drawn using the MOSIS design rules [51, 52].

In order to have sufficient boosting on VH and VL, C1 and C2 need to be large enough to be able to maintain the boosted signal for the entire time duration of the signal transition. C1 and C2 will determine the current ability of the driver, and must be larger than the parasitic capacitance of the node VH/VL, which includes

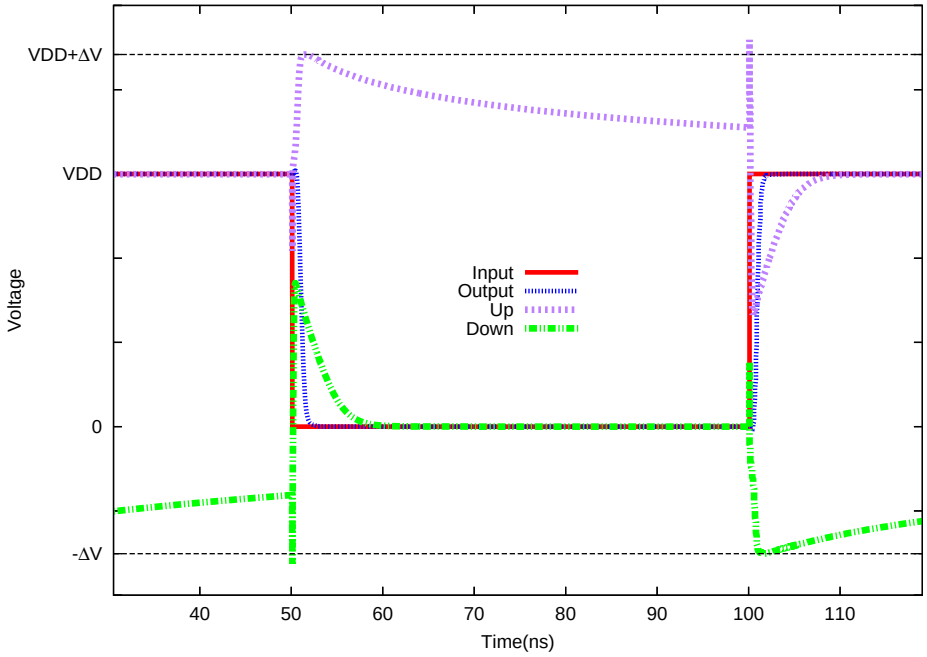

Figure 4.4: Operation waveform of the proposed sub-threshold boosted driver

the gate capacitance of the P3 and N3 together with the junction capacitances of all the other devices attached to the VH and VL nodes. There are a number of options to implement the boosting capacitors. One option would be to implement the capacitance using a MOS capacitor. However, a MOS capacitance is smaller in the sub-threshold region compared to the super-threshold since the depletion capacitance is in series with the oxide capacitance [2]. Consequently, a large area should be dedicated to the MOS capacitor in order to achieve enough boosting. Since the MOS capacitor is also sensistive to PVT variations, implementing a large capacitance with a MOS device is not a good option. Another option in implementing the boosting capacitors is to use metal-over-metal (MOM) capacitors [53]. A MOM device is based on the capacitance between parallel metal lines seperated by the inter-level dielectric. The device requires no additional masks and additional process cost [54, 55, 56]. As technology scales, reduced metal line width and spacing and also an increased number of available metal layers makes MOM a useful alternative in designing the boot capacitors. The MOM unit capacitance is determined by the number of metal

fingers, the width and space of the metal fingers, and the number of metal layers used. Figure 4.6 shows the top layout view of an MOM using 12×12 metal fingers. We have designed the MOM such that it occupies the same layout area of the boosted driver. We divided the area of the boosted driver in half. Each half is used to design boosting capacitors, C1 and C2 shown in Figure 4.3. The MOM device can be build upon the layers above metal 1 layer with the same area as the boosted driver. Since there are many metal layers in newer technologies, for example 9 metal layeres are available in TSMC 65nm technology, there are many unused metal layers that can be used to build the boosting MOM capacitor with no additionl cost and area penalty. It is worth mentioning that MOM has a good tolerance to process variation, temperature and voltage variation since it is a passive device built with only metal and dielectric [56]. Therefore, using MOM as C1 and C2 in implementing our boosted driver makes our design reliable.

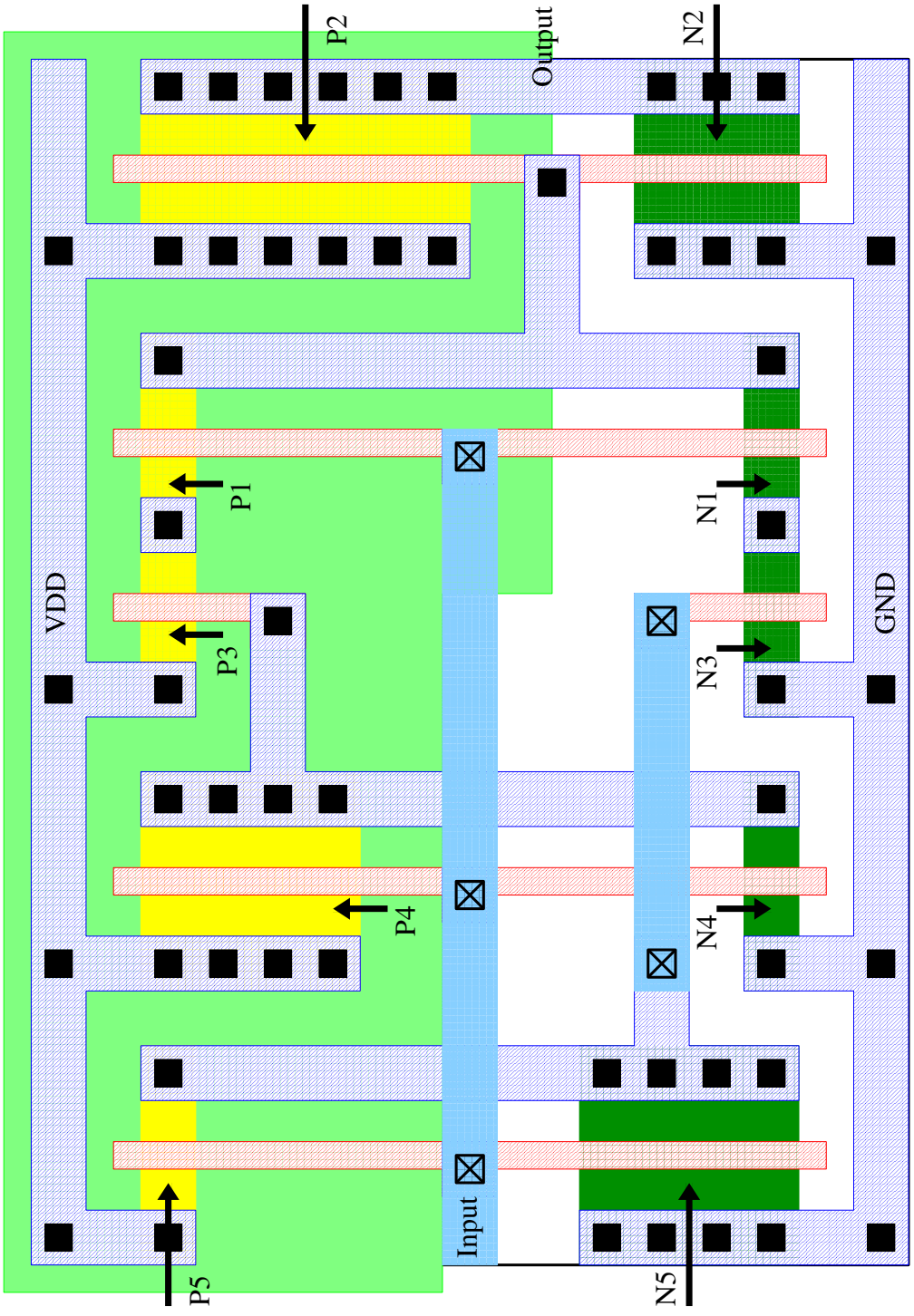

Figure 4.5: Layout of the proposed sub-threshold boosted driver

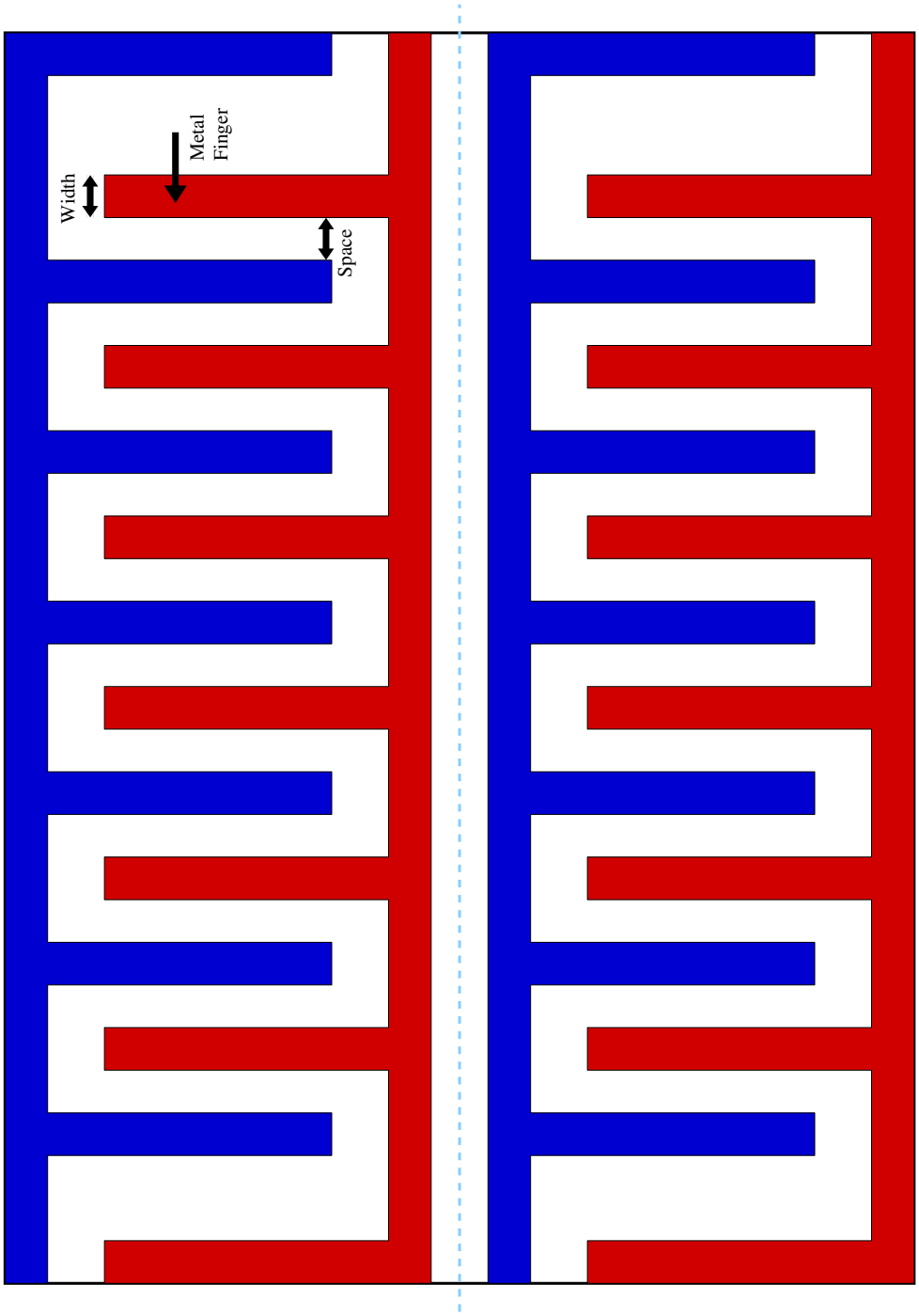

Figure 4.6: The top layout view of structure of a MOM capacitor

4.4.2 Designing the Multiplexer with Capacitive Boosting

Commercial FPGAs are operated in strong inversion, and multiplexers inside the routing network are mainly implemented using NMOS pass-transistor logic [8]. However, as we mentioned previously (in Section 2.1), the use of NMOS pass transistors lead to very poor output swing and makes the circuit asymptotically slow and sensitive to variations in the sub-threshold region. One way to improve the speed is to implement multiplexers using transmission gate [7]. However, even after using transmission gate and deploying buffers, the delay and energy consumption of the interconnection network is the main contributor to the overall delay of the FPGA in sub-threshold. In addition, this aproach increases the area of the switch boxes [7]. We have already shown (Section 3.2) that capacitive boosting provides performance improvement of FPGA interconnection networks operating in strong inversion [57]. Bootstrap pass-transistor logic consists of a *Pass Transistor* for data propagation and an *Isolator* that ensures capacitive coupling between the pass transistor's source and gate (Figure 3.5). The parasitic capacitance C_{gs} helps boost the gate voltage up higher than V_{DD} when the input signal rises. As a result, the voltage at the output node of the NMOS pass-transistor logic can now be raised to V_{DD} [57].

It is apparent that the capacitive boosting is a dynamic method to speed up the rising transition (the static method uses CMOS transmission gates [7]). Boosting with a single NMOS pass transistor exhibits a poor falling delay in sub-threshold and it will not be considered any further. To improve the falling transition, we propose to use a non-boosted NMOS transistor in parallel with the NMOS pass transistor with boosted gate. This results in an NMOS-*Twins* transmission gate, as shown in Figures 4.7. Since NMOS is a good transmitter of '0', the circuit has now a good falling edge. Also, the circuit has a good rising edge due to capacitive boosting on the gate of the boosted NMOS pass transistor. Our proposed circuit technique in implementing multiplexer resembles the CMOS transmission gate, where the PMOS transistor is replaced with an NMOS transistor; this technique requires less area in layout level and it is more robust against latch-up over charge injection due to a single type of transistors used. In addition, we use capacitive boosting to transmit logic '1' instead of PMOS since capacitive boosting is more effective in sub-threshold due to the exponential relationship of the current with the gate-to-source voltage.

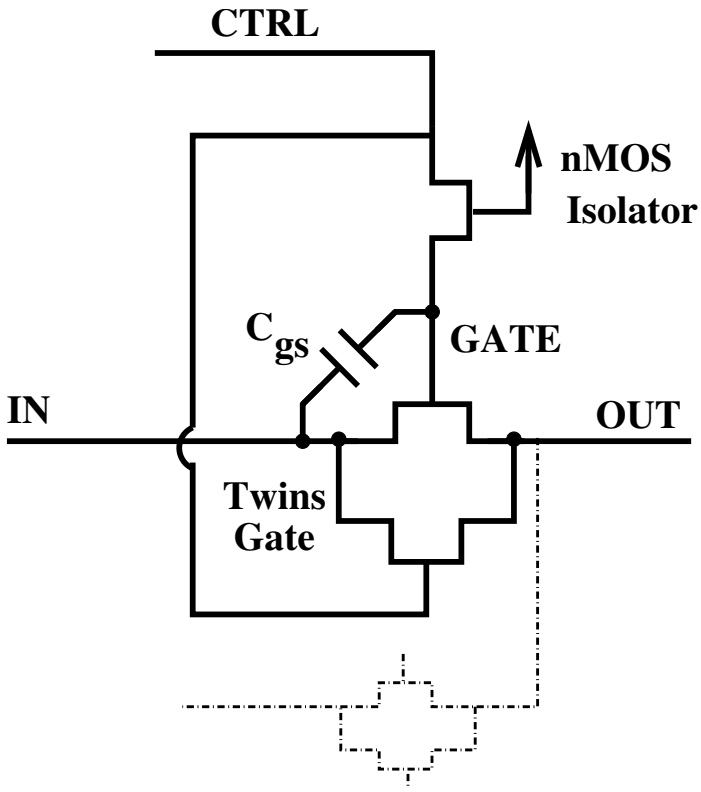

Figure 4.7: Circuit implementation of Twins transmission gate

4.4.3 Designing the Switch Box based on Capacitive Boosted Driver and Multiplexer

As we mentioned in Section 2.1, and Figure 2.7, a so-called single_driver switch is composed of a multiplexer and a driver that strengthenes the multiplexed signal to be transmitted through the wire. In prior art, the sub-threshold multiplexer is implemented using CMOS transmission gates; the driver has to be large enough to be able to drive the longe wire and mitigate the variation [7]. However, this approach has a large area penalty; the delay of the interconnection continues to dominate the overall delay of the FPGA. In the capacitive boosting approach, we implement the multiplexer using Twins transmission gates. We also use a boosted driver to drive the long wire as shown in Figure 4.8. This way, the operation region of the single_driver switch is shifted away from sub-threshold toward super-threshold, resulting an improved speed and enhanced tolerance to variations. The last inverter of the boosted driver is driven with the boosted nodes in the boosted inverter as shown in Figure 4.8. This insures that its gate drive is $V_{DD}+\Delta$ V ($-\Delta$ V) instead

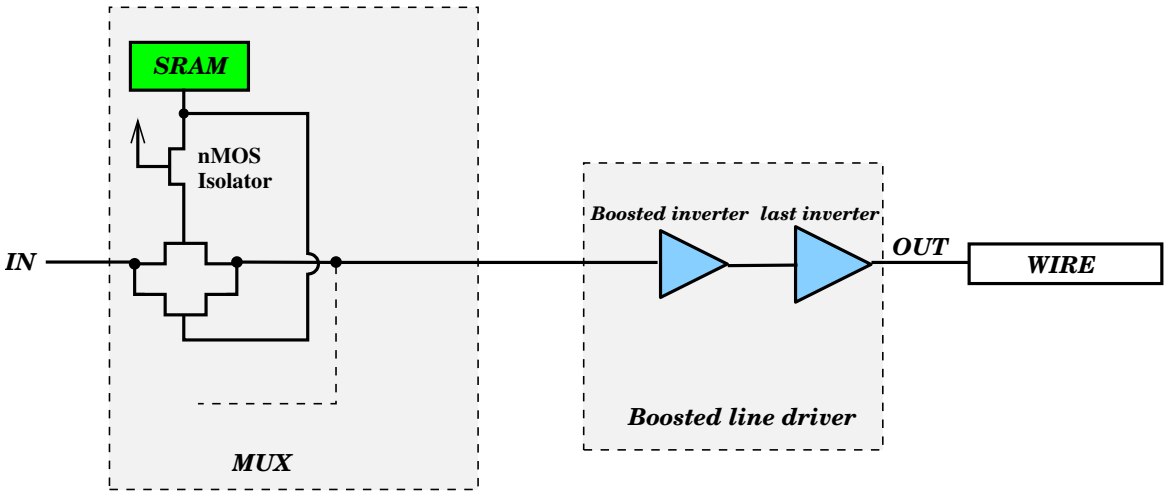

Figure 4.8: Implementation of the proposed single_driver switch based on Twins and boosted line driver

of V_{DD} (0V), and, since the last inverter operates in sub-threshold, its current and driving ability increases exponentially. As a result, the last inverter does not require to be large as in the prior art (non-boosted driver). This results in saving silicon area in implementing a switch box.

4.5 Simulation Framework and Results

As disscussed in Section 3.3, a single_driver switch with an 8:1 multiplexer was determined to be optimal for area_delay product (ADP) metric. As prior art, we considered 8:1 multiplexers built with a single layer of CMOS transmission gates. Therefore, the circuit under test comprises eight CMOS transmission gates together with eight configuration SRAM cells, line driver and level-restoring buffer. The experimental setup shown in Figure 4.9 is based on transmission gate and non-boosted driver. We put a long wire model after each wire driver to evaluate the driving ability and performance of each design under realistic circuit conditions. The bootstrap single_driver switch also includes 8:1 multiplexers built with a single layer of Twins transmission gates and eight Isolators together with eight configuration SRAM cells, boosted line driver and level-restoring buffer as shown in Figure 4.10.

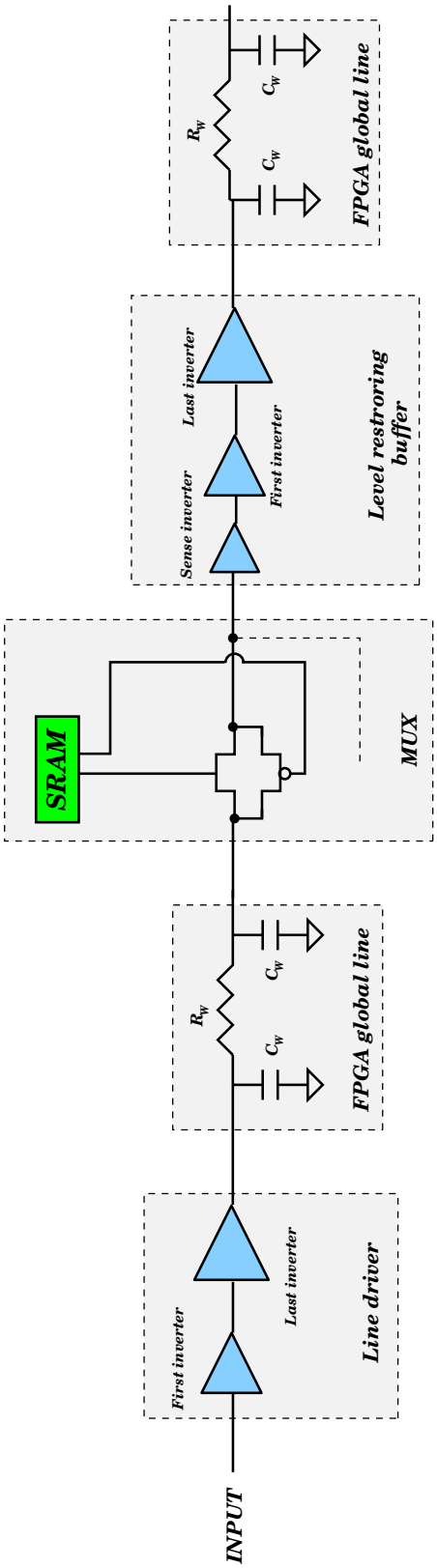

Figure 4.9: An example of the switch box driver based on transmission gate

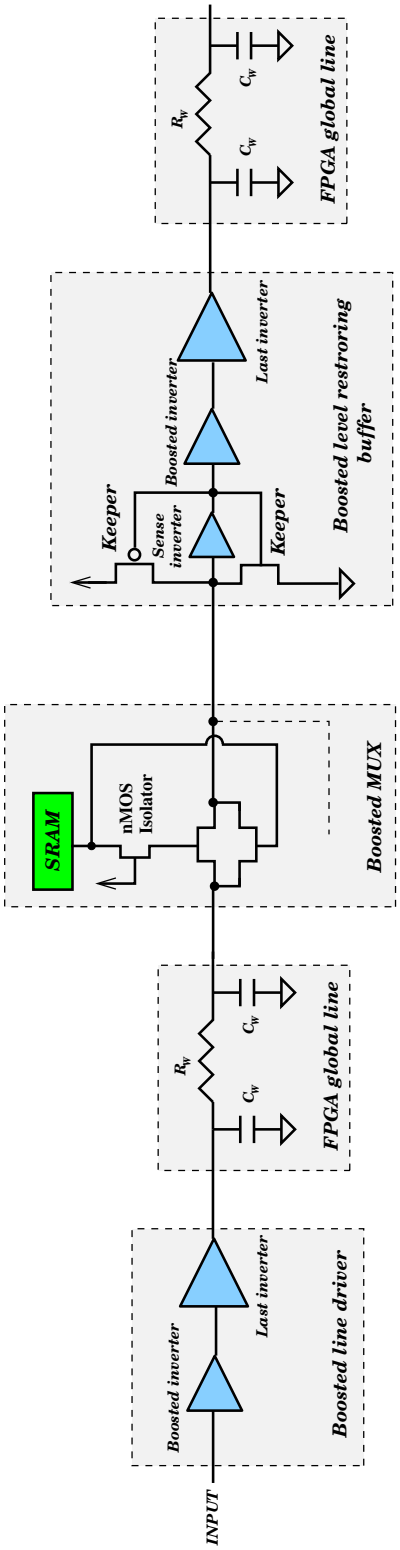

Figure 4.10: An example of the proposed switch box driver based on Twins and boosted line driver

During a long inactive periode when there is no transition in the input, the boosted voltages of the proposed driver cannot be sustained because of the parasitic leakage currents. To mitigate this issue, Keepers are also used in level-restoring buffers. This way, the data is locked locally within the level-restoring buffer after being passed through the Twins transmission gates. This feature provides a better noise immunity.

The transistor sizes of the routing network are presented in Tables 4.1 and 4.2. In reducing the rising delay, it is also apparent that a wider boosted NMOS pass transistor is preferable since it provides a larger capacitive coupling, and thus, a stronger boosting effect on the gate of the NMOS pass transistor. The transistor sizes presented in Tables 4.1 and 4.2 are for a driver which drives HEX-length segments (which means the wire spans 6 CLB tiles) [58]. Different length wire segments will typically have routing resources with different sizes. For example, larger boosting capacitors, C1 and C2 (shown in Figure 4.3), are required for a boosted driver which drives a longer segment than the HEX-length segments.

Most of the benefits of the proposed single_driver switch are the result of the boosted driver. According to Table 4.2, a boosting capacitor of 12fF is needed to be able to drive HEX-length segments. According to our simulation, it is possible to built two 26fF MOM capacitors with 12×12 metal fingers using 2-to-4 metal layers per each capacitor in TSMC 65nm technology. Note that these capacitors require no additional silicon area since they are deployed in unused metal layers situated above the area of the boosted driver (Figure 4.5).

To reduce the leakage current of the circuit in sub-threshold minimum-size transistors are used. However, upsizing of the transistor is needed to assure the circuit reliabilty and functionality [13, 14, 44]. We sized the circuits in a way to have balanced rising and falling edges to improve the design reliability and robustness [13, 14, 44, 45]. In sizing the circuits in 90nm, a minimum leakage current is observed at a width of W=180nm for NMOS device as shown in Table 4.3. Leakage current decreases initially with increasing size from Wmin=120nm to W=180nm, but then starts to increase with increasing size. The leakage current at W=180nm is about 21% less than the leakage current at minimum_width NMOS device. Therefore, following the recommendations in [45, 59], we considered W=180nm instead of Wmin=120nm when sizing the boosted driver as presented in Table 4.2. The leakage current of the PMOS device increases monotonically with increasing size.

Table 4.1: Component sizes (nm). All transistors are of minimum length.

Single_driver switch based on CMOS transmission gate and non-boosted driver		
	65nm	90nm
First Inverter (PMOS/NMOS)	1000/500	2000/1000
Last Inverter (PMOS/NMOS)	3000/1500	3000/1500
Sense Inverter (PMOS/NMOS)	360/180	360/180
Transmission Gate (PMOS,NMOS)	240, 240	240, 240

Table 4.2: Component sizes (nm). All transistors are of minimum length with the except of Keepers.

Single_driver switch based on Twins transmission gate and boosted driver			
		65nm	90nm
Boosted Inverter	C1, C2	12, 12	10, 10
	P4, N4	480, 120	360, 180
	P5, N5	120, 480	120, 360
	P3, N3	120, 120	120, 180
	P1, N1	120, 120	120, 180
Last Inverter (P/N)		2000/1000	2000/1000
Twins(boosted,non-boosted) (N, N)		360, 180	360, 180
Sense Inverter (P/N)		360, 180	360, 180
Keeper(NMOS) (W/L)		120/120	180/180
Keeper(PMOS) (W/L)		120/120	120/180

Table 4.3: NMOS device leakage current versus size in 90nm

NMOS Device Size(nm)	Drain Current(pA)
120	617
130	569
140	537
150	515
160	500
170	491
180	486
220	487
240	596

It should be noted that smaller boosting capacitors are needed in 90nm compared to 65nm. This is because in sub-threshold region, PMOS devices are stronger than NMOS devices in 65nm technology which makes the PMOS competitor in a full transmission gate much harder to defeat by boosting. Therefore, we need larger boosting capacitors to increase the drain current of NMOS devices in our proposed circuits in 65nm technology node.

We have created Spice netlists for both the boosted and non-boosted single_driver switches. All circuit designs, netlists, and simulations were completed with the Cadence IC5.1.41 tool suite [42] including Analog Artist and Spectre for standard 65nm and 90nm technology provided by Canadian Microelectronics Corporation (CMC) and MOSIS and for a number of sub-nominal supply voltages. The results are reported according to area, propagation delay, and power consumption metrics, and also the area-delay and power-delay products composite metrics. The delay is measured from line driver input to the output of the sense inverter of level-restoring buffer with respect to the midpoint between supply and ground. The results are presented in Tables 4.4 and 4.5 for 65nm and 90nm, respectively in sub-threshold region. Results in near-threshold and super-threshold region are presented in Tables 4.6 and 4.7 in 65nm and 90nm, respectively. Labels 1-to-0 and 0-to-1 in Tables 4.4, 4.5, 4.6 and 4.7 refer to pull-up and pull-down transitions at the pass-transistor source/drain, respectively. The power consumption is calculated by integrating the voltage-current product over a normalized period of time. To estimate the area, we use the model proposed by Lemieux and Lewis [8]. The resulting area figures are normalized to the minimum transistor area in each technology.

Table 4.4: Simulation results for single_driver switche in sub-threshold

Voltage	Technology and Circuits	65nm		
		Non-boosted switche	Bootsted switche	Δ (%)
0.3V	0-to-1 (ns)	3.53	2.4	-32
	1-to-0 (ns)	3.8	2.24	-41
	Average Delay (ns)	3.66	2.32	-36.6
	Area (normalized)	137.25	120.25	-12.3
	Power (μW)	0.367	0.427	16.3
	Area-Delay (ns)	502	278	-44.6
	Power-Delay (fJ)	1.34	0.99	-26.1
0.25V	0-to-1 (ns)	7.67	5.2	-32.2
	1-to-0 (ns)	8.5	4.91	-42.2
	Average Delay (ns)	8.08	5.05	-37.5
	Power (μW)	0.255	0.295	15.6
	Area-Delay (ns)	1,108	607	-45.2
	Power-Delay (fJ)	2.06	1.5	-27
0.2V	0-to-1 (ns)	18.15	12.94	-28.7
	1-to-0 (ns)	20.34	12.45	-38.7
	Average Delay (ns)	19.24	12.69	-34
	Power (μW)	0.163	0.188	15.3
	Area-Delay (ns)	2,640	1,525	-42.2
	Power-Delay (fJ)	3.13	2.38	-23.9

Table 4.5: Simulation results for single_driver switche in sub-threshold

Voltage	Technology and Circuits	90nm		
		Non-boosted switche	Bootsted switche	Δ (%)
0.4V	0-to-1 (ns)	3.5	1.92	-45
	1-to-0 (ns)	3.3	1.85	-43.9
	Average Delay (ns)	3.4	1.88	-44.7
	Area (normalized)	149.7	116.7	-22
	Power (μW)	1.52	1.63	7.2
	Area-Delay (ns)	509	219.4	-56.9
	Power-Delay (fJ)	5.17	3	-42
0.35V	0-to-1 (ns)	6.78	3.83	-43.5
	1-to-0 (ns)	6.9	3.47	-49.7
	Average Delay (ns)	6.84	3.65	-46.6
	Power (μW)	1.16	1.24	7
	Area-Delay (ns)	1023	426	-58.3
	Power-Delay (fJ)	7.9	4.5	-43
0.3V	0-to-1 (ns)	14.63	8.68	-40.6
	1-to-0 (ns)	16.8	7.63	-54
	Average Delay (ns)	15.7	8.1	-48.4
	Power (μW)	0.77	0.83	8
	Area-Delay (ns)	2350	945	-59.7
	Power-Delay (fJ)	12	6.7	-44

Table 4.6: Simulation results for single_driver switche in near-threshold and super-threshold

Voltage	Technology and Circuits	65nm		
		Non-boosted switche	Boosted switche	Δ (%)
0.4V	0-to-1 (ns)	1.04	0.748	-28
	1-to-0 (ns)	1.04	0.695	-33.1
	Average Delay (ns)	1.04	0.72	-30.7
	Power (μW)	0.653	0.764	16.9
	Area-Delay (ns)	142	86	-39.4
	Power-Delay (fJ)	0.67	0.54	-19.4
0.6V	0-to-1 (ps)	253	209	-17.3
	1-to-0 (ps)	232	204	-12
	Average Delay (ps)	242.5	206.5	-14.8
	Power (μW)	1.5	1.75	16.6
	Area-Delay (ps)	33,283	24,831	-25.3
	Power-Delay (fJ)	0.36	0.36	0
0.8V	0-to-1 (ps)	125	114	-8.8
	1-to-0 (ps)	111	117	5.4
	Average Delay (ps)	118	115.5	-2.1
	Power (μW)	2.76	3.23	17
	Area-Delay (ps)	16,195	13,888	-14.2
	Power-Delay (fJ)	0.32	0.37	15.6
1V	0-to-1 (ps)	84	85	1.1
	1-to-0 (ps)	76	86	13.1
	Average Delay (ps)	80	85.5	6.8
	Power (μW)	4.55	5.4	18.6
	Area-Delay (ps)	10,980	10,281	-6.3
	Power-Delay (fJ)	0.36	0.46	27.7

Table 4.7: Simulation results for single_driver switche in near-threshold and super-threshold

Voltage	Technology and Circuits	90nm		
		Non-boosted switche	Boosted switche	Δ (%)
0.6V	0-to-1 (ps)	642	428	-33.3
	1-to-0 (ps)	600	439	-26.8
	Average Delay (ps)	621	433.5	-30.1
	Power (μW)	3.43	3.72	8.4
	Area-Delay (ps)	92,963	50,589	-45.5
	Power-Delay (fJ)	2130	1612.6	-24.3
0.8V	0-to-1 (ps)	288	231	-19.8
	1-to-0 (ps)	274	246	-10.2
	Average Delay (ps)	281	238.5	-15.1
	Power (μW)	6.12	6.7	9.4
	Area-Delay (ps)	42,065	27,832	-33.8
	Power-Delay (fJ)	1719.7	1598	-7
1V	0-to-1 (ps)	182	184	1
	1-to-0 (ps)	181	180	-0.5
	Average Delay (ps)	181.5	182	-0.2
	Power (μW)	9.6	10.5	9.4
	Area-Delay (ps)	27,170	21,239	-21.8
	Power-Delay (fJ)	1742.4	1911	-1.7
1.2V	0-to-1 (ps)	137	147	7.3
	1-to-0 (ps)	142	151	6.3
	Average Delay (ps)	139.5	149	6.8
	Power (μW)	13.9	15.2	9.3
	Area-Delay (ps)	20,883	17,388	-16.7
	Power-Delay (fJ)	1939	2264	16.7

It is apparent that an improvement in propagation delay of 36% and 44% has been achieved at 0.3V in 65nm and 0.4V in 90nm, respectively, which are typical supply voltages used in sub-threshold region for these process nodes. The power-delay product decreases siginificantly in sub-threshold region, which means the power penalty is acceptable for the obtained delay improvement. The area-delay product decreases significantly in sub-threshold region by at least 42% across the both technologies. Owing to the exponential dependance of the sub-threshold current to gate-to-source voltage, a small amount of voltage boost results in a significant decrease in propagation delay. 12% and 22% area saving has been achieved in designing the boosted driver in 65nm and 90nm, respectively. Saving in area has been acheived because smaller drivers (Figure 4.10) are needed to drive the wire compared to the non-boosted driver (Figure 4.9).

According to Tables 4.4, 4.5, 4.6 and 4.7, the beneficial effects of the boosted single_driver switch on speed are stronger in sub-threshold region than the super-threshold region. As the supply voltage increase toward nominal voltage, which is 1V and 1.2V in 65nm and 90nm, respectively, the effect of the boosting on speed diminishes. This is because the relationship between the current and gate-to-source voltage in super-threshold is no longer exponential approaching a linear dependance. Also, the *last inverter* in the boosted driver (Figure 4.10) which is smaller than the non-boosted counterpart (Figure 4.9) has to fight agains the *Keepers* which makes the effect of boosted driver on speed in super-threshold (without an exponentially improved current) weaker than corresponding effect in sub-threshold (with an exponentially improved current). This confirms that the proposed boosted switch can be run in both strong and weak inversion, but becomes more efficient at low supply voltages due to the exponential current behavior in sub-threshold region. Our proposed circuit has better speed over all supply voltages ranging from 0.2V to 1V with the exception of the nominal 1V in 65nm. Our proposed circuit has also better speed over all supply voltages ranging from 0.3V to 1.2V with the exception of 1.2V in 90nm. It should be noted that in *ultra-low power (ULP)* applications, it is important to operate in sub-threshold to reduce the energy consumption, and the applications do not demand a maximum operation speed in super-threshold. Besides that, our boosted circuit has always a better *area-delay product (ADP)* across the all supply voltages, which is an important metric in evaluating an FPGA.

To quantify the sensitivity to process variations, a Monte Carlo simulation with 1,000 runs has been carried out. The 3σ deviation from the rising delay and falling

Table 4.8: Delay statistics for non-boosted and boosted single_driver switch in 65nm

Circuit	$\mu_{Delay(ns)}$		$\sigma_{Delay(ns)}$		μ/σ	
	HL	LH	HL	LH	HL	LH
Non-boosted	3.6	3.6	0.52	0.53	14 %	14 %
Boosted	2.1	2.2	0.13	0.17	6 %	7 %

Table 4.9: Delay statistics for non-boosted and boosted single_driver switch in 90nm

Circuit	$\mu_{Delay(ns)}$		$\sigma_{Delay(ns)}$		μ/σ	
	HL	LH	HL	LH	HL	LH
Non-boosted	3.3	3.4	0.27	0.25	12 %	13 %
Boosted	1.9	1.9	0.19	0.2	10 %	9.5 %

delay is presented in Tables 4.8 and 4.9 for 65nm and 90nm, respectively. It is apparent that the boosted circuitry has a smaller sensitivity than the non-boosted (the previous best) in both 65nm and 90nm process nodes. This can be explained by the fact that boosted driver transistors are shifted from sub-threshold to near-threshold or super-threshold increaesing the drive current which results in enhanced tolerance to variations. Moreover, the Twins based multiplexer has also boosting on the gate of the boosted NMOS pass transistor increasing the current and variation tolerance. It also uses a single type of devices (namely NMOS), whereas CMOS transmission gate uses both NMOS and PMOS devices; matching is always better when devices of a single type are used.

Variation to temperature is an important factor which has also to be considered in evaluating any circuit in sub-threshold circuit. Tables 4.10 and 4.11 present the rise delay and fall delay of the circuits under temperature ($0 \ldots 80°$C) variation at 0.3V and 0.4V supply voltage in 65nm and 90nm, respectively. Delay variation, which is defined as maximun delay to minimum delay ratio [2], is reduced from 2.82× to 1.67× in 65nm and from 1.9× to 1.5× in 90nm for the fall delay. Rise delay variation is also reduced from 2.73× to 1.97× in 65nm and from 1.97× to 1.66× in 90nm. Figure 4.11 shows the rise delay and Figure 4.12 shows the fall delay under (0.3V $\pm$5%) and ($20 \ldots 80°$C) temperature variation in 65nm. It verifies the reduced impact of environmental variations on proposed ciruit performance.

Table 4.10: Delay variation (i.e., maximun delay to minimum delay ratio [2]) under temprature variation at 0.3V V_{DD} in 65nm

Temperature (°C)	HL (ns)		LH (ns)	
	Non-boosted	Boosted	Non-boosted	Boosted
0	5.64	2.56	5.25	2.8
25	3.9	2.26	3.63	2.47
50	2.83	2	2.65	1.83
65	2.37	1.74	2.24	1.6
80	2	1.53	1.92	1.42
Delay Variation	2.82×	1.67×	2.73×	1.97×

Table 4.11: Delay variation (i.e., maximun delay to minimum delay ratio [2]) under temprature variation at 0.4V V_{DD} in 90nm

Temperature (°C)	HL (ns)		LH (ns)	
	Non-boosted	Boosted	Non-boosted	Boosted
0	4.23	2.3	4.53	2.5
25	3.36	1.87	3.55	1.97
50	2.74	1.63	2.87	1.61
65	2.46	1.56	2.55	1.57
80	2.23	1.53	2.29	1.5
Delay Variation	1.9×	1.5×	1.97×	1.66×

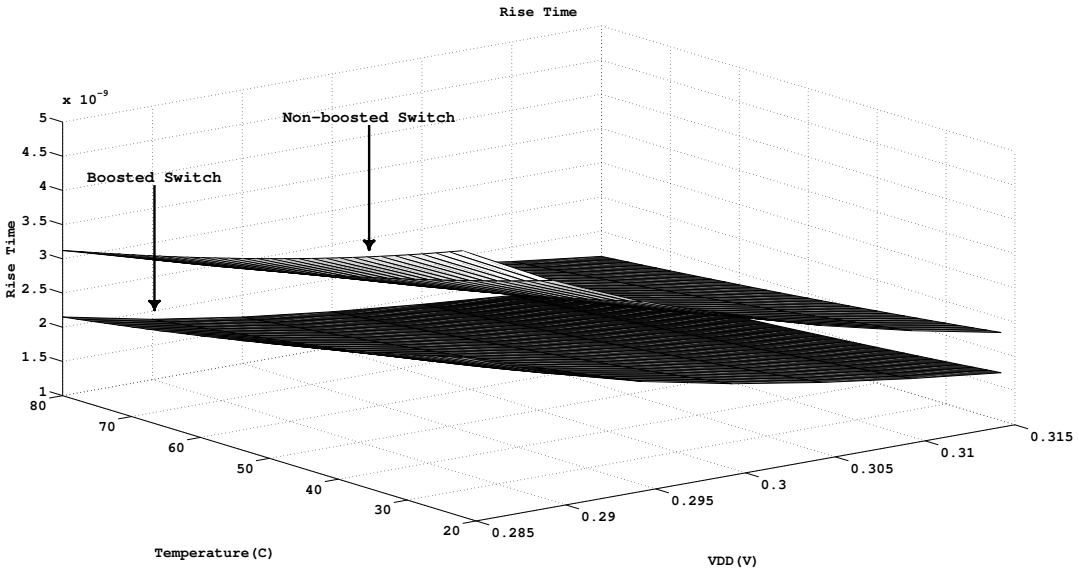

Figure 4.11: Rise delay with respect to temperature and supply voltage variations in 65nm

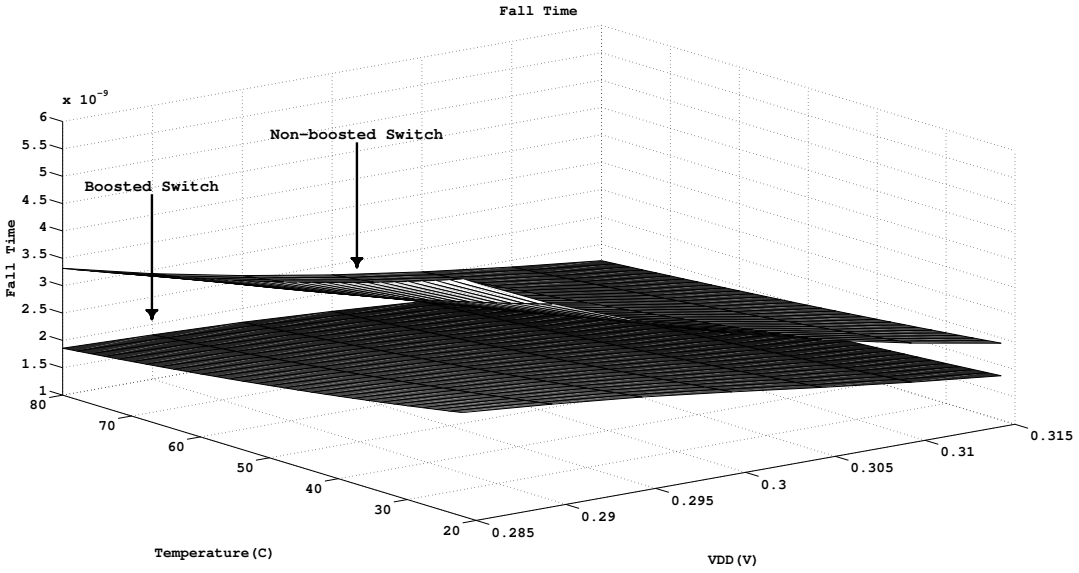

Figure 4.12: Fall delay with respect to temperature and supply voltage variations in 65nm

Table 4.12: Rise delay and fall delay for different wire length in 65nm

Wire Length (mm)	HL (ns)		LH (ns)	
	Non-boosted	Boosted	Non-boosted	Boosted
0.5	4.48	2.46	4.16	2.53
1	6.8	3.3	6.3	3.05
2	10.68	5.08	10.1	4.09
3	14.32	7	13.6	5.14
4	17.9	9.3	17.13	6.24

Table 4.13: Rise delay and fall delay for different wire length in 90nm

Wire Length (mm)	HL (ns)		LH (ns)	
	Non-boosted	Boosted	Non-boosted	Boosted
1	4.65	2.27	4.82	2.27
2	7.68	3.27	7.86	3.1
3	10.4	4.25	10.6	3.9
4	13.1	5.23	13.2	4.7
5	15.6	6.2	15.7	5.5

In order to evaluate the driving ability of our boosted single_driver switch, we considered different length wire segments as a load for the driver. Although, the drivers should be optimized for different length wire segments, we considered the driver optimized for HEX-length segments and only changed the wire length to compare the driving ability of boosted and non-boosted driver without loss of generality. Since it is unlikely that any wires as longer than 4-5mm will be considered [11], we assessed the rise delay and fall delay for wire lengths ranging from 0.5mm to 4mm in a 60nm process as presented in Table 4.12 and from 1mm to 5mm in a 90nm process as presented in Table 4.13 .

It is interesting to note that as in the delay results, boosted single_driver switch has better speed than non-boosted single_driver switch for increasing wire length. This is apparent from Tables 4.12 and 4.13, where the boosted design significantly outperforms the non-boosted design; it is confirmed that boosting the long wire driver transistors and using Twins transmission gate reduces the propagation delay of the single_driver switch.

4.6 Summary

In this chapter, we first described the sub-threshold operation concepts. We have also discussed the issues related to designing sub-threshold circuits. We have presented the sub-threshold FPGAs and challenges facing sub-threshold FPGA design. Since interconnection network is the main delay contributor in sub-threshold FPGAs, we proposed to use capacitive boosting to shift the driver transistors from sub-threshold to near-threshold or super-threshold increasing the drive current and robustness . We have shown that capacitive boosting is a powerful tool for improving the performance and enhancing variation tolerance of sub-threshold FPGA interconnection circuitry owing to the exponential relationship between the drain current and gate-to-source voltage in the sub-threshold region. In addition, we have shown that the synergy of capacitive boosting with NMOS-Twins transmission gates in implementing multiplexers provides a better speed and reliability over the prior art. The simulations indicate a reduction of at least 36% and 44% in propagation delay and ADP, repectively for the proposed circuit versus the prior art at 0.3V V_{DD} and 0.4V V_{DD} which are the typical sub-threshold supply voltages in 65nm and 90nm, respectively. In addition, the proposed boosted circuit has better tolerance to PVT variations and is more reliable under environmental variations. It should be noted that these enhancements are obtained without additional supply voltages or low- or zero-threshold devices. Given the fact that the area-delay product decreases in the bootstrap circuit with respect to the prior art, we can say that the circuit design technique we propose is a viable alternative in building performant and robust FPGA interconnection networks in weak inversion.

Chapter 5

Conclusions and Future Work

The focus of this thesis is on understanding and augmenting the circuit structures used to implement FPGA interconnection networks, in order to make tradeoff between delay, power consumption, area, and reliability in both strong and weak inversion. For FPGA designers, this is a significant concern as the delay of the interconnection network is the main contributor of the FPGA overall delay. In addition, it is necessary to ensure a reliable operation in weak inversion. The main challenge of operating in weak inversion is robustness in the presence of variations.

This dissertation described two contributions towards addressing problems related to FPGA interconnection networks. The first contribution addresses the performance of the FPGA global and local interconnection network operating in strong inversion, and the second contribution addresses the performance and robustness of the FPGA interconnection networks operating in weak inversion. These contributions and key results presented in each chapter of this dissertation are summarized below.

Chapter 3 addressed the interconnect delay problem in strong inversion by investigating the design of level restoring buffer. We deployed the basic bootstrap circuit in NMOS based multiplexers and modified it to have balanced rising and falling propagation delays in FPGA routing environment. Our resulting circuit improves both the global and local interconnect speed. By examining the power-delay product (PDP) and area-delay product (ADP) of our circuit, it can be seen that interconnection network with capacitive boosting can offer more to FPGAs than just improved delay. The use of our circuit provides good tradeoffs between the delay, area, and power consumption in designing the global and local routing network. Results show that using capacitive boosting can reduce delay by at least 10-17% across all considered technologies in our simulations. The power-delay product also exhibits a significant

improvement of at least 17% for local interconnect, and 12% for global interconnect. The area-delay product decreases slightly, which means the area penalty is acceptable for the obtained delay improvement. This work was published in [57].

The second contribution of this thesis, which is described in Chapter 4, is a new technique in designing the global routing resources in weak inversion FPGAs. The technique involves using capacitive boosting in designing the long wire drivers in global interconnection networks. Due to the exponential relationship between the current and gate-to-source voltage in weak inversion, the operation region of routing resources is shifted away from sub-threshold toward near-threshold or even super-threshold. This results in increased drive current and tolerance to variation. We also proposed a new circuit design technique, called *Twins transmission gate*, in implementing multiplexers used in single_driver switch . Our resulting boosted single_driver switch improves the speed at least 34% in weak inversion. In addition, a good tradeoff between area and delay has been achieved, since ADP decreases at least 42% for boosted switch box in weak inversion. The boosted circuit increases the power consumption of the switch box at most 16% in sub-threshold. However, a decrease of at least 23% in PDP shows a good tradeoff between the delay and power using our boosted switch box. It is also important to notice that our boosted single_driver switch decreases the area of the switch box by 12% and 22% in 65nm and 90nm process nodes, respectively since area is an important factor in evaluating an FPGA. Additionally, the results indicate that the proposed boosted circuit has better tolerance to PVT variations.

Future Work

This work has attempted to lay the preliminary ground work for further research into interconnect optimization for FPGAs operating in both strong and weak inversion. Since this research is mainly divided into two parts, recommendations are grouped into two categories: FPGA routing in strong inversion and FPGA routing in weak inversion.

As future work in strong inversion, we will investigate the bootstrap effectiveness on FPGA interconnection networks with two-level NMOS pass transistor trees. In this thesis, we only considered multiplexeres with a single layer of pass transistors. Capacitive boosting can be used in designing clock network for faster clock speeds and better skew. It would also be interesting to use capacitive boosting in designing

logic elements such as LUTs, FFs, and barrel shift units with improved delay. The gate-oxide integrity under impulse overshoot stress is also an interesting open research question.

As future work in weak inversion, we will investigate designing an FPGA that can be run in both strong and weak inversion, while the priority is given to weak inversion because of the higher sensitivity to variations. We will also apply our boosted driver in designing clock drivers in sub-threshold, since designing a fast and robust clock network is a major challenge in sub-threshold FPGAs.

Bibliography

[1] S. Miller, M. Sima, and M. McGUIRE, "Alternatives in designing level-restoring buffers for interconnection networks in field-programmable gate arrays," in *Digital System Design Architectures, Methods and Tools, 2007. DSD 2007. 10th Euromicro Conference on*, pp. 138–146, IEEE, 2007.

[2] J. Kil, J. Gu, and C. Kim, "A high-speed variation-tolerant interconnect technique for sub threshold circuits using capacitive boosting," pp. 67–72, 2006.

[3] G. Lemieux and D. Lewis, "Circuit design of routing switches," in *Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays*, pp. 19–28, ACM, 2002.

[4] E. Hung, S. Wilton, H. Yu, T. Chau, and P. Leong, "A detailed delay path model for fpgas," in *Field-Programmable Technology, 2009. FPT 2009. International Conference on*, pp. 96–103, IEEE, 2009.

[5] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for deep-submicron FPGAs*. Kluwer Academic Publishers, 1999.

[6] M. Sheng and J. Rose, "Mixing buffers and pass transistors in fpga routing architectures," in *Proceedings of the 2001 ACM/SIGDA ninth international symposium on Field programmable gate arrays*, pp. 75–84, ACM, 2001.

[7] B. Calhoun, J. Ryan, S. Khanna, M. Putic, and J. Lach, "Flexible circuits and architectures for ultralow power," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 267–282, 2010.

[8] G. Lemieux and D. Lewis, *Design of interconnection networks for programmable logic*. Springer, 2004.

[9] G. Lemieux, E. Lee, M. Tom, and A. Yu, "Directional and single-driver wires in fpga interconnect," in *Field-Programmable Technology, 2004. Proceedings. 2004 IEEE International Conference on*, pp. 41–48, IEEE, 2004.

[10] E. Lee, G. Lemieux, and S. Mirabbasi, "Interconnect driver design for long wires in field-programmable gate arrays," *Journal of Signal Processing Systems*, vol. 51, no. 1, pp. 57–76, 2008.

[11] E. Lee, G. Lemieux, and S. Mirabbasi, "Interconnect driver design for long wires in field-programmable gate arrays," *Journal of Signal Processing Systems*, vol. 51, no. 1, pp. 57–76, 2008.

[12] H. Soeleman, K. Roy, and B. Paul, "Robust subthreshold logic for ultra-low power operation," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 9, no. 1, pp. 90–99, 2001.

[13] A. Wang and A. Chandrakasan, "A 180-mv subthreshold fft processor using a minimum energy design methodology," *Solid-State Circuits, IEEE Journal of*, vol. 40, no. 1, pp. 310–319, 2005.

[14] A. Wang, B. Calhoun, and A. Chandrakasan, *Sub-threshold design for ultra low-power systems*. Springer Verlag, 2006.

[15] J. Ryan and B. Calhoun, "A sub-threshold fpga with low-swing dual-vdd interconnect in 90nm cmos," in *Custom Integrated Circuits Conference (CICC), 2010 IEEE*, pp. 1–4, IEEE, 2010.

[16] S. Pable and M. Hasan, "High speed interconnect through device optimization for subthreshold fpga," *Microelectronics Journal*, 2011.

[17] A. DeHon, *Reconfigurable architectures for general-purpose computing*. Massachusetts Institute of Technology, 1996.

[18] J. Huang, C. Sung, B. Wang, K. Nguyen, X. Wang, and R. Cliff, *High-speed programmable interconnect*. Google Patents, July 17 2001. US Patent 6,262,595.

[19] J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital integrated circuits*. Prentice-Hall, 1996.

[20] S. Young, M. Hart, V. Kondapalli, and M. Voogel, "Integrated circuit multiplexer including transistors of more than one oxide thickness," Sept. 27 2005. US Patent 6,949,951.

[21] E. Lee, G. Lemieux, and S. Mirabbasi, "Interconnect driver design for long wires in field-programmable gate arrays," in *Field Programmable Technology (FPT), International Conference on*, pp. 89 –96, IEEE, 2006.

[22] D. Lewis, "Merged logic element routing multiplexer," May 8 2007. US Patent 7,215,141.

[23] L. Ciccarelli, A. Lodi, R. Giansante, L. Magagni, R. Canegallo, R. Guerrieri, *et al.*, "Switch block for fpga architectures," Dec. 9 2008. US Patent 7,463,067.

[24] Y. Nakagome, K. Itoh, M. Isoda, K. Takeuchi, and M. Aoki, "Sub-1-v swing internal bus architecture for future low-power ulsis," *Solid-State Circuits, IEEE Journal of*, vol. 28, no. 4, pp. 414–419, 1993.

[25] S. Kulkarni and D. Sylvester, "High performance level conversion for dual vdd design," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 12, no. 9, pp. 926–936, 2004.

[26] D. L. J. Rose, R.J. Francis and P. Chow, "Architecture of eld-programmable gate arrays: the effect of logic block functionality on area effciency," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 5, p. 12171225, 1990.

[27] *Xilinx Incorporated., San Jose, California,*. http://www.xilinx.com/.

[28] T. Pi and P. Crotty, "Fpga lookup table with transmission gate structure for reliable low-voltage operation," Dec. 23 2003. US Patent 6,667,635.

[29] I. Kuon and J. Rose, "Area and delay trade-offs in the circuit and architecture design of fpgas," in *Proceedings of the 16th international ACM/SIGDA symposium on Field programmable gate arrays*, pp. 149–158, ACM, 2008.

[30] I. Kuon and J. Rose, "Exploring area and delay tradeoffs in fpgas with architecture and automated transistor design," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 19, no. 1, pp. 71–84, 2011.

[31] *Altera Incorporated., San Jose, California,*. http://www.altera.com/.

[32] V. George and J. Rabaey, *Low-energy FPGAs: architecture and design*. Springer, 2001.

[33] J. Luu, I. Kuon, P. Jamieson, T. Campbell, A. Ye, W. Fang, K. Kent, and J. Rose, "Vpr 5.0: Fpga cad and architecture exploration tools with single-driver routing, heterogeneity and process scaling," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 4, no. 4, p. 32, 2011.

[34] J. Koomen and J. van den AKKER, "A most invertor with improved switching speed," *Solid-State Circuits, IEEE Journal of*, vol. 7, no. 3, pp. 231–237, 1972.

[35] K. Fujii and T. Douseki, "A sub-1 v bootstrap pass-transistor logic," *IEICE transactions on electronics*, vol. 86, no. 4, pp. 604–611, 2003.

[36] M. Iijima, M. Kitamura, N. Masahiro, T. IPPOSHI, and S. MAEGAWA, "Boosted voltage scheme with active body-biasing control on pd-soi for ultra low voltage operation," *IEICE transactions on electronics*, vol. 90, no. 4, pp. 666–674, 2007.

[37] C. Seitz, A. Frey, S. Mattisson, S. Rabin, D. Speck, and J. Snepscheut, "Hot clock nmos," 1985.

[38] Y. Nonaka, "Bootstrap circuit," Mar. 3 2011. US Patent 20,110,050,317.

[39] S. Liu, C. Chen, and M. Lin, "Method and device for reducing voltage stress at bootstrap point in electronic circuits," Aug. 20 2007. US Patent App. 11/894,752.

[40] J. Anderson and F. Najm, "Active leakage power optimization for fpgas," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 25, no. 3, pp. 423–437, 2006.

[41] K. Poon, S. Wilton, and A. Yan, "A detailed power model for field-programmable gate arrays," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 10, no. 2, pp. 279–302, 2005.

[42] *Cadence Design Systems, Inc., San Jose, California*. http://www.cadence.com/.

[43] A. Mutlu and P. Aminzadeh, "A method to comprehend the impact of interconnect coupling effects on gate oxide reliability," in *Reliability Physics Symposium Proceedings, 2003. 41st Annual. 2003 IEEE International*, pp. 570–571, IEEE, 2003.

[44] J. Kwong and A. Chandrakasan, "Variation-driven device sizing for minimum energy sub-threshold circuits," in *Proceedings of the 2006 international symposium on Low power electronics and design*, pp. 8–13, ACM, 2006.

[45] J. Ryan, J. Wang, and B. Calhoun, "Analyzing and modeling process balance for sub-threshold circuit design," in *Proceedings of the 17th ACM Great Lakes symposium on VLSI*, pp. 275–280, ACM, 2007.

[46] N. Verma, J. Kwong, and A. Chandrakasan, "Nanometer mosfet variation in minimum energy subthreshold circuits," *Electron Devices, IEEE Transactions on*, vol. 55, no. 1, pp. 163–174, 2008.

[47] J. Kulkarni, K. Kim, and K. Roy, "A 160 mv robust schmitt trigger based subthreshold sram," *Solid-State Circuits, IEEE Journal of*, vol. 42, no. 10, pp. 2303–2313, 2007.

[48] S. Lin, Y. Wang, R. Luo, and H. Yang, "A capacitive boosted buffer technique for high-speed process-variation-tolerant interconnect in udvs application," pp. 304–309, 2008.

[49] N. Verma and A. Chandrakasan, "A 256 kb 65 nm 8t subthreshold sram employing sense-amplifier redundancy," *Solid-State Circuits, IEEE Journal of*, vol. 43, no. 1, pp. 141–149, 2008.

[50] M. Rahaman and M. Chowdhury, "Interconnect technique for sub-threshold circuits using negative capacitance effect," in *Circuits and Systems, 2009. MWSCAS'09. 52nd IEEE International Midwest Symposium on*, pp. 1122–1125, IEEE, 2009.

[51] C. Piña, "Evolution of the mosis vlsi educational program," in *Electronic Design, Test and Applications, 2002. Proceedings. The First IEEE International Workshop on*, pp. 187–191, IEEE, 2002.

[52] *Mosis Scalable CMOS (SCMOS) Design Rules,*. http://www.mosis.com/.

[53] N. Weste and K. Eshraghian, "Principles of cmos vlsi design: a systems perspective," *NASA STI/Recon Technical Report A*, vol. 854, p. 47028, 1985.

[54] K. Christensen, "Design and characterization of vertical mesh capacitors in standard cmos," in *VLSI Circuits, 2001. Digest of Technical Papers. 2001 Symposium on*, pp. 201–204, IEEE, 2001.

[55] R. Aparicio and A. Hajimiri, "Capacity limits and matching properties of integrated capacitors," *Solid-State Circuits, IEEE Journal of*, vol. 37, no. 3, pp. 384–393, 2002.

[56] C. Chang, C. Chao, J. Chern, and J. Sun, "Advanced cmos technology portfolio for rf ic applications," *Electron Devices, IEEE Transactions on*, vol. 52, no. 7, pp. 1324–1334, 2005.

[57] F. Eslami and M. Sima, "Capacitive boosting for fpga interconnection networks," in *Field Programmable Logic and Applications (FPL), 2011 International Conference on*, pp. 453–458, IEEE, 2011.

[58] J. Anderson and F. Najm, "Low-power programmable routing circuitry for fpgas," in *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*, pp. 602–609, IEEE Computer Society, 2004.

[59] B. Fu and P. Ampadu, "Leakage power minimization of nanoscale cmos circuits via non-critical path transistor sizing," in *Electronics, Circuits and Systems, 2006. ICECS'06. 13th IEEE International Conference on*, pp. 1101–1104, IEEE, 2006.