Low Power, Area Efficient FinFET Circuit Design

Michael C. Wang, Princeton University

***Abstract*—FinFET, which is a double-gate field effect transistor (DGFET), is more versatile than traditional single-gate field effect transistors because it has two gates that can be controlled independently. Usually, the second gate of FinFET transistors is used to dynamically control the threshold voltage of the first gate in order to improve the performance and reduce leakage power. However, we can also utilize FinFET's second gate to implement circuits with fewer transistors. This is important since area efficiency is one of the main concerns in circuit design. In this paper, a novel scheme of implementing a majority gate and a 2-1 MUX by using both gates of FinFET transistors as inputs is presented. Simulation results show that FinFET logic implementation has significant advantages over static CMOS logic and pass transistor logic in terms of power consumption and cell area.**

***Index Terms*— FinFET, low power circuit, logic synthesis, combinational logic, independent gate.**

I. INTRODUCTION

As the size of transistors has scaled down, so have many digital applications. Cell phones, laptops, sensors, and many other applications all shrunk in size over the last few decades and they are more and more portable. For this to happen, chips in these digital applications have to be designed to optimize the number of transistors used, the fewer the better. In this case, pass transistor logic is an attractive solution because a circuit can usually be implemented in pass transistor logics with around half of the number of transistors required for static CMOS implementation. However, pass transistor logic allows inputs to be tied to the source and the drain of a transistor, thus create possible situations where NMOS has to drive a logic 1 and PMOS has to drive a logic 0. Since NMOS is not a good pull up device, the output of a pass transistor circuit will suffer from a voltage drop V_{th} and never achieve a full voltage swing to V_{DD}. With the continuing scaling of supply voltage, this voltage drop cannot be tolerated.

The additional back gate of a FinFET gives circuit designers many options. The back gate can serve as a secondary gate that enhances the performances of the front (primary) gate. For example, if the front gate voltage is V_{DD} (transistor is ON) the back gate can be biased to V_{DD} to provide bigger current drive, which reduces transistor delay. If the front gate voltage is 0 (transistor is OFF), the back gate can be biased to 0, which raises the threshold voltage of the front gate and reduces the leakage current. Most recent FinFET circuit researches, such as FinFET SRAM [1], focus on utilizing the back gate to improve circuit performance. On the other hand, the back gate can also be used to reduce the number of transistors needed to implement many logic functions. For a NFinFET, the transistor turns on if either the front gate or the back gate is V_{DD} – this is equivalent to two NMOS transistors in parallel. Recent researches, such as a 3-transistor FinFET NAND gate [2], utilize this property. However, we have not seen any research that utilizes this property beyond a simple logic gate such as a NAND gate. The focus of this paper is to expand the idea of using both gates of FinFET as inputs to more complicated logic circuits, and provide insight on how to design a FinFET-based circuit with independent inputs for any logic function.

In Section II and Section III, we will propose a novel FinFET majority gate and a 2-1 MUX. In Section IV, we will present simulation results. We conclude the paper in Section V.

Manuscript received July 3, 2009.

Michael C. Wang was a graduate student at Princeton University, Princeton, NJ 08544 USA. (e-mail: mcwang@alumni.princeton.edu).

Table 1. Truth table of a 3-input majority gate

A	B	C	Out
0	0	0	**0**
0	0	1	**0**
0	1	0	**0**
0	1	1	**1**
1	0	0	**0**
1	0	1	**1**
1	1	0	**1**
1	1	1	**1**

II. FINFET MAJORITY GATE

A majority gate is commonly used in a full adder. A typical majority gate has three inputs and one output. If more than half of the inputs are 1, it returns a 1 on the output, otherwise it returns a 0. Table 1 shows the truth table of a 3-input majority gate. Schematics of three different implementations of a majority gate are shown in Fig. 1 (from left to right:

Table 2. Truth table of a 2-1 MUX

A	B	S	Out
0	0	0	**0**
0	0	1	**0**
0	1	0	**0**
0	1	1	**1**
1	0	0	**1**
1	0	1	**0**
1	1	0	**1**
1	1	1	**1**

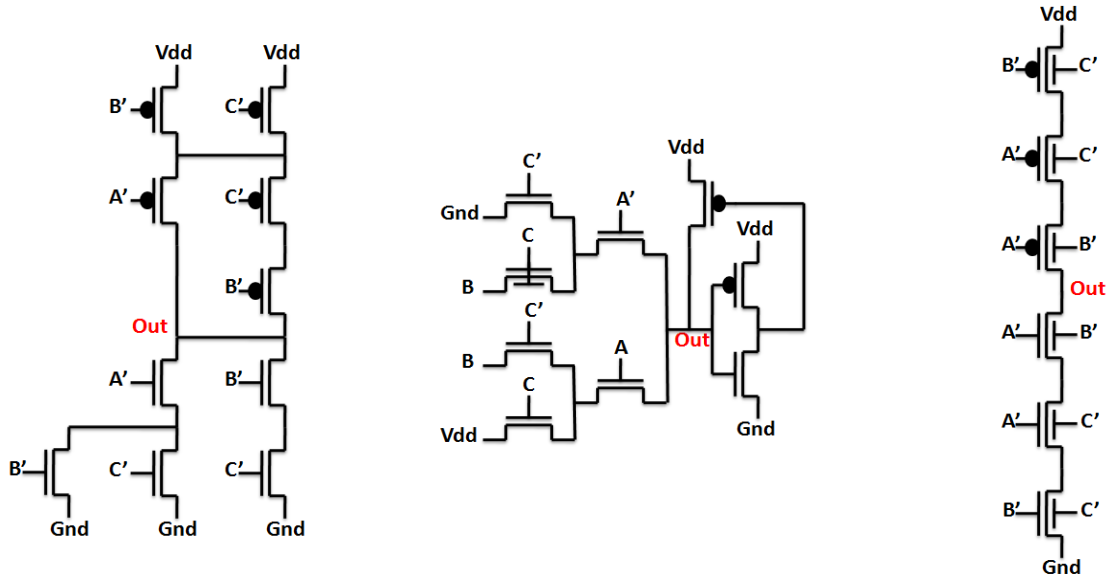

Fig. 1. Schematics of a majority gate in CMOS logic, pass transistor logic, and FinFET logic

CMOS logic, pass transistor logic, FinFET logic).

In a CMOS logic circuit, the output is connected to either V_{DD} or 0 through pull-up or pull-down network so that it is never floating. Since NMOS are good at passing a 0 while PMOS are good at passing the V_{DD}, NMOS transistors are used in the pull-down network while PMOS transistors are used in the pull-up network.

A pass transistor logic circuit can be designed by first generating a binary decision diagram, and then mapping nodes to transistors and branches to wires [3]. In a pass transistor logic circuit, a transistor can pass 0 and V_{DD}. In other words, it "does more work" than a transistor in CMOS logic circuit that only passes either 0 (NMOS) or V_{DD} (PMOS). Intuitively, the number of transistor needed in pass transistor logic circuit is generally smaller than in CMOS logic circuit (thus smaller cell area). For the same reason, pass transistor logic circuit does not achieve full voltage swing at the output node because transistors have to pass 0 and V_{DD}, and as we mentioned earlier, each type of transistor is only good at passing one of 0 or V_{DD}. To restore output voltage, a three-transistor level restorer circuit is appended at the output node.

Both CMOS logic and pass transistor logic were developed for conventional NMOS and PMOS transistor. If FinFET technology is available, we can easily adapt both circuit design methodology by replacing NMOS with NFinFET and PMOS with PFinFET, then, tie both gates of FinFET together. By using this approach, we can design a FinFET version of a CMOS logic circuit or a pass transistor logic circuit that retains the same functionalities as the MOSFET version. In the mean time, FinFET provides better circuit performances and reduces leakage current through effective suppression of short-channel effect and near-ideal subthreshold swing. For this reason, we will use FinFET for all layout designs and simulations in this paper. This way, we will be able to show that the impacts of using different logic styles come from the difference in architecture, not from the difference in transistors used.

The schematic of FinFET logic majority gate, as shown in the last schematic of Fig. 1, contains just 6 transistors. There

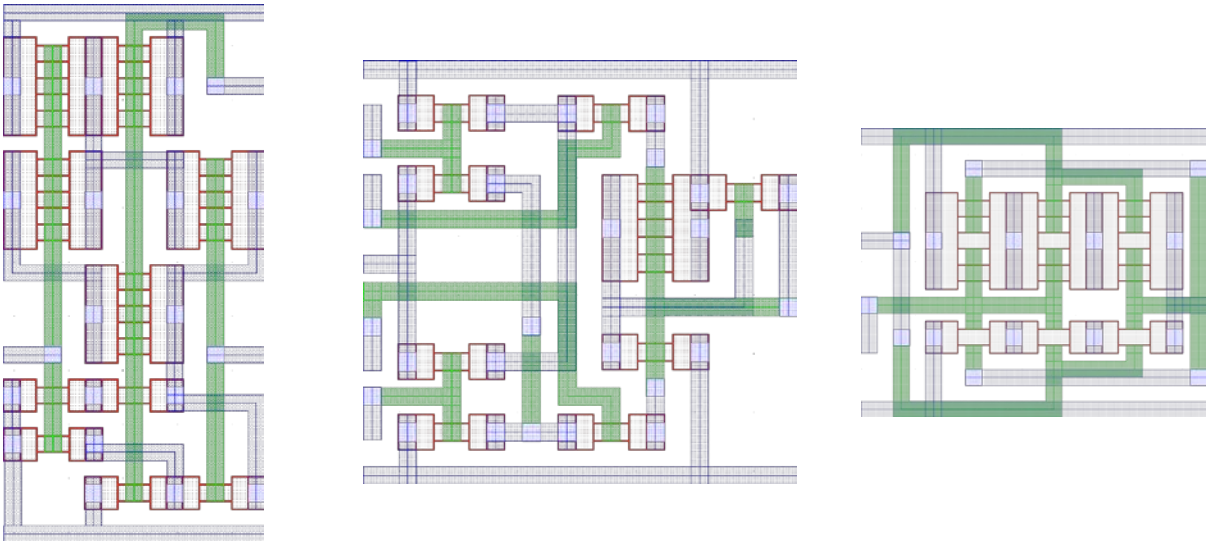

Fig. 2. Layouts of a 2-1 MUX in CMOS logic, pass transistor logic, and FinFET logic

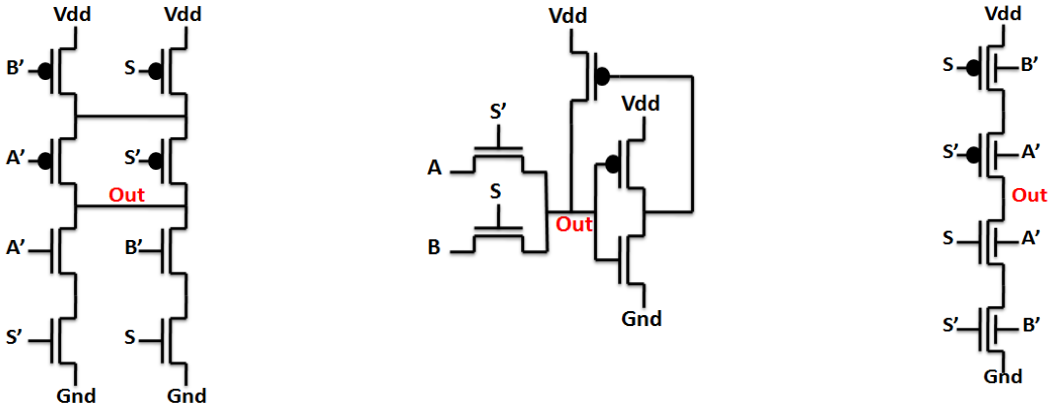

Fig. 3. Schematics of a majority gate in CMOS logic, pass transistor logic, and FinFET logic

is a current path from the output to either 0 or V_{DD} all the time, which is similar to CMOS logic. The difference is that each gate of FinFET transistors is driven by different inputs. This is called "independent mode" of FinFET.

From logic perspective, a FinFET operating in "independent mode" is equivalent to two MOSFET operating in parallel. By examining the circuit, we can obtain the output expression, out' = (A'+B')*(A'+C')*(B'+C'), which is actually in the OR-AND logic form (in contrast with the more intuitive AND-OR logic form). To construct the pull-down network, we want (A'+B')*(A'+C')*(B'+C') = 1 (so that out = 0, which means pull-down network is triggered to pass 0). Implementation each of A'+B', A'+C', and B'+C' requires two NMOS transistors, but since both transistors share the same source and drain, they can be merged into one single NFinFET. Next, we place these NFinFETs in series as implied by the product sign of the OR-AND logic expression. Similarly, we construct the pull-up network to complete the circuit shown in Fig. 1. Layouts of each implementation of majority gate are shown in Fig. 2. They are built using a hypothetical 30nm technology.

III. FINFET 2-1 MUX

2-1 MUX is another widely used 3-input function. It has applications in both combination logic circuit and sequential logic circuit. Table 2 shows the truth table of 2-1 MUX function. Schematics of three different implementations of a majority gate are shown in Fig. 3 (from left to right: CMOS logic, pass transistor logic, FinFET logic). A, B are the primary input bits, S is the select bit. The output bit is equal to A if S=0, and B if S=1. In other word, the output function is out = S'*A + S*B, where S' is the complement of S.

FinFET logic implementation of a 2-1 MUX can be constructed similar to FinFET majority gate described in Section II. The OR-AND logic expression is out' = (S+A')*(S'+B'), which can be mapped as two NFinFET in series in the pull-down network and two PFinFET in series in the pull-up network. The resulting circuit, which is shown as the last schematic in Fig. 3, contains 4 transistors.

Layouts of each implementation of majority gate are shown in Fig. 4. Note that pass transistor logic implementation, which is the second schematic in Fig. 2, actually achieves the smallest area among all three logic styles, even though it uses one more transistor (include level-restorer circuit) than FinFET logic implementation. However, its high power consumption (almost twice as much as FinFET logic implementation) makes it less attractive for low-power embedded applications.

IV. SIMULATION RESULTS

In this section, we present the simulation results generated for 30nm technology from Synopsys Sentaurus, which is a device level simulator [4], for all 6 circuits mentioned in this paper. Schematics of these circuits can be found in Figures 1

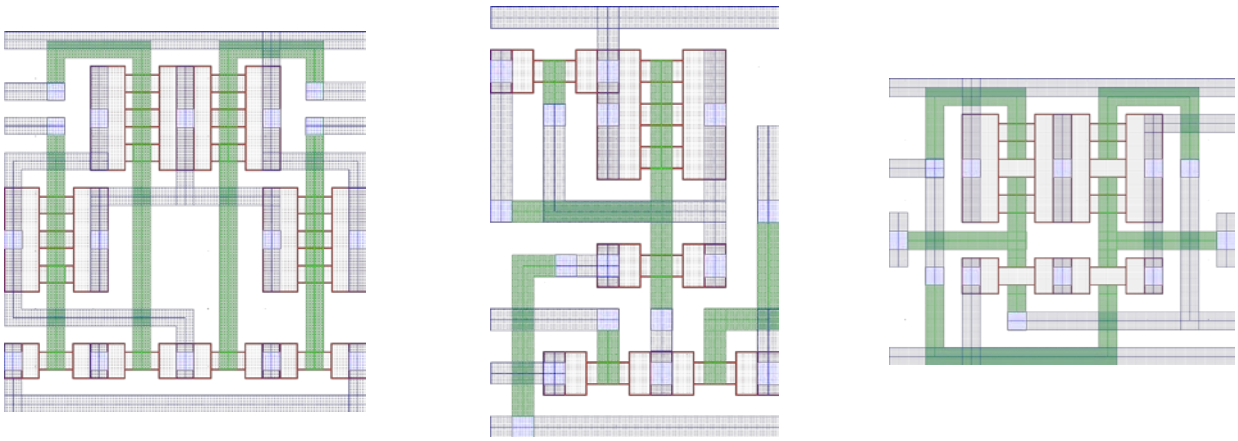

Fig. 4. Layouts of a 2-1 MUX in CMOS logic, pass transistor logic, and FinFET logic

Table 3. Summary of majority gate implementations

Designs →	CMOS	PTL	FinFET
# Transistor	10	9	6
Area (nm^2)	475200	529200	348300
F. Time (ps)	10.913	31.227	24.270
R. Time (ps)	14.204	31.883	70.314
Power (uW)	44.7	57.5	28.0

Table 4. Summary of 2-1 MUX function implementations

Designs →	CMOS	PTL	FinFET
# Transistor	8	5	4
Area (nm^2)	415800	243000	273600
F. Time (ps)	15.249	20.148	28.132
R. Time (ps)	9.079	28.604	49.769
Power (uW)	31.5	41.4	22.0

and 2. Before we present the results, we will explain how we perform simulations and extract data listed in Tables 3 and 4.

A. Layout Consideration

In FinFET technology, "device widths are dispensed in units of whole fins only." [5] This is known as device width quantization, which limits our ability to size transistors effectively in FinFET circuit. On top of that, there is also problem with FinFETs with even number of fins operating in "independent mode," because of the difficulty in inputs. However, it is shown in [6] that both inputs can be easily routed in a FinFET with 3 fins. For this reason, we chose 3 as the number of fins of PFinFET to the number of fins of NFinFET (3 fins PFinFET to 1 fin NFinFET). As shown in Tables 3 and 4, FinFET logic implementation achieves on average 25% reduction in cell area.

B. Delay Extraction

Delay is measured as the time difference between 10% and 90% of the voltage swing. For example, if we are trying to bring node A from 0V to 1V, then the delay is the time it takes for node A to go from 0.1V to 0.9V. In our simulation, we assumed that all the intermediate nodes have 1fF capacitance while the output nodes and input buses have 5fF capacitance.

The transient analysis plots for each circuit are shown in Figures 5 and 6. The input sequence is the same as reading from row 1 to row 8 of Tables 1 and 2. Of course, this sequence does not account for every possible transition, but it should give us fairly accurate estimates.

Simulation results show that CMOS logic circuits are approximately 2.2 times faster than pass transistor logic circuit and 3.5 times faster than FinFET logic circuit.

C. Power Extraction

We will not consider dynamic power consumption on output node and input buses because in our simulation, each circuit implementation switches same number of times on these nodes. The sole exception is pass transistor logic implementation for majority gate, which has 5 inputs (A, A', B, C, C') compare with 3 inputs (A', B', C') of the other two circuit implementations. The extra inputs lead to more dynamic power consumption for pass transistor logic implementation, but since it already has the highest active and leakage power consumption, omitting dynamic power consumption calculation will not change the fact that pass transistor logic circuits are usually inferior in term of power consumption.

Active power is the power consumed when both pull-up and pull-down network are active, creating a direct current path from V_{DD} to ground, while leakage power is the power consumed when charges "leak" through a transistor that is off. Calculating the active and leakage components of power consumption separately is very difficult. Therefore, we will calculate the aggregated power consumption by first calculating instantaneous power P(t) = V(t)*I(t), then sum up P(t) for all times (in our simulation, t: 0~1000ps), which will give us the total energy consumed for this operation. This is easy to do since Sentaurus can provide information about voltage and current across each transistor at any time. Finally, sum up the energy consumption for all transistors and divide by clock period to obtain the active and leakage power consumption, which is listed in Table 3 for majority gate and

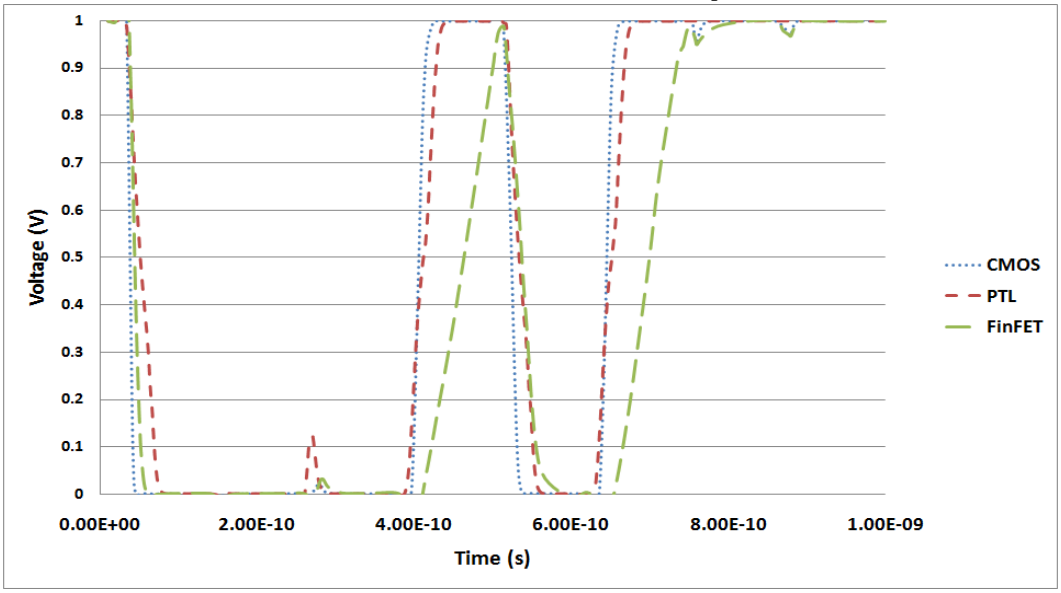

Fig. 5. Transient Analysis plot of a majority gate for different implementations

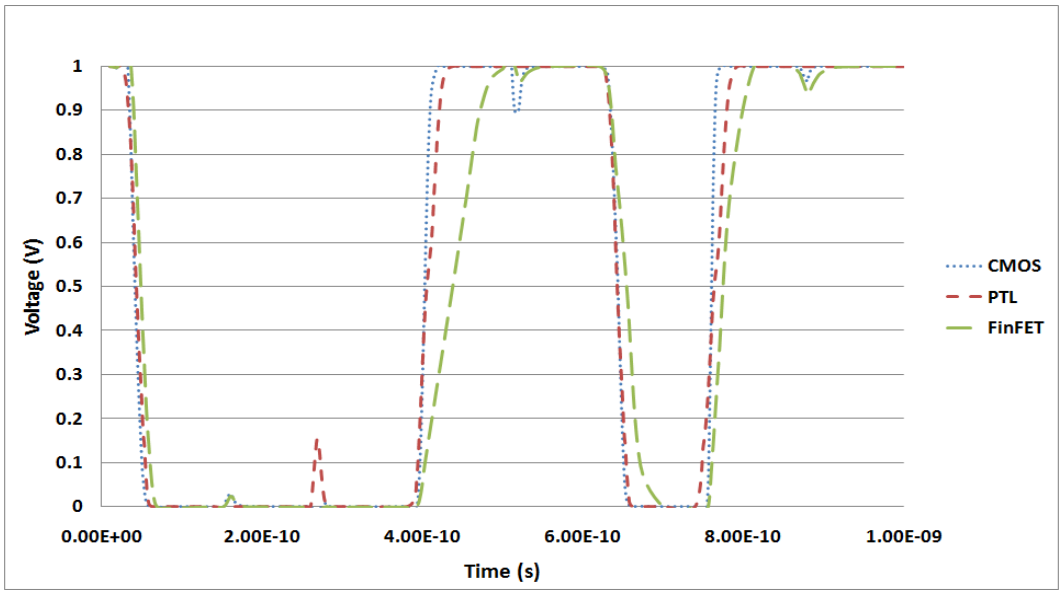

Fig. 6. Transient Analysis plot of a 2-1 MUX for different implementations

Table 4 for 2-1 MUX function.

Note that the input sequence in our simulation implies that input C (for majority gate) or input S (for 2-1 MUX) switches most frequently, while input A switches least frequently. The placement of inputs in a circuit has some impacts on power consumption [7]. For consistency, we place the least frequent switching input closest to the output and most frequent switching input closest to supply rails.

Simulation results show that FinFET logic circuits consume least amount of power. In comparison, CMOS logic circuits consume about 52% more power and pass transistor logic circuits consume about 98% more power.

V. Conclusion

FinFET not only has superior performance over bulk silicon MOSFET, but is primed to take over bulk silicon MOSFET as the dominant transistor choice for sub-45nm technology. FinFET also has potential for reductions of the required number of transistors and chip area in circuits, which is crucial for many digital applications. For a majority gate and a 2-1 MUX function, we were able to use both gates of FinFET as inputs by writing the logic expression in OR-AND form and using the fact that from logic perspective, FinFET is equivalent to two MOSFETs operating in parallel. Simulation results show that compared to other logic implementations, FinFET logic circuits achieve significant area and power reduction without voltage or transistor scaling, even though they suffer greatly in circuit speed.

References

[1] A. Carlson, Z. Guo, S. Balasubramanian, L.-T. Pang, T.-J. King Liu, and B. Nikolic, “FinFET SRAM with Enhanced Read / Write Margins,” SOI Conference, 2006.

[2] A. Muttreja, N. Agarwal, N.K. Jha, N.K., “CMOS logic design with independent-gate FinFETs”, ICCD 25th International Conference on, 2007, pp. 560-567.

[3] V. Bertacco, “Decision Diagrams and Pass Transistor Logic Synthesis,” Proc. of the ACM/IEEE International Workshop on Logic Synthesis, May 1997, pp. 1-5.

[4] http://www.synopsys.com/Tools/TCAD/DeviceSimulation, accessed June 1, 2009

[5] K. Bernstein, C. T. Chuang, R. Joshi and R. Puri, “Design and CAD challenges in sub-90 nm CMOS technologies,” ICCAD, Nov.2003, pp. 129-136.

[6] S. A. Tawfik and V. Kursun, "Finfet domino logic with independent gate keepers," Microelectronics Journal, March 2009.

[7] L. Benini, G.D., Micheli, "Dynamic Power Management," 1st ed. Springer, Nov 1997, pp. 24-25.