AN ABSTRACT OF THE THESIS OF

Steven K. Hsu for the degree of Master of Science in Electrical and Computer Engineering presented on February 2, 2001. Title: High-Performance Low-Power Microprocessor Circuits.

Redacted for Privacy

Abstract approved: _____

Shih-Lien Lu

This thesis will discuss two critical components of a digital system -- domino logic styles and flip-flops. In today's microprocessors, both domino logic and flip-flops are essential to high-performance and low-power design. Two new domino logic styles are presented and analyzed, Double Edge Triggered (DET) and Double Data Rate (DDR) Domino. Using a CMOS 0.25μm MOSIS model, HSPICE simulations show that a DET & DDR 8-bit adder has a maximum throughput of 1.6Gops (Giga Operations per second) and 2Gops, respectively, while a conventional domino adder has only 1Gops. In addition, this thesis proposes a novel master-slave flip-flop. This flip-flop is compared with other known flip-flop structures. Using a CMOS 0.35μm MOSIS model, the new flip-flop was simulated to have an optimal power-delay product. The proposed flip-flop consumes very low power, while having a very small clock load and data load.

High-Performance Low-Power Microprocessor Circuits

By

Steven K. Hsu

A Thesis submitted

To

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented February 2, 2001
Commencement June 2001

Master of Science thesis of Steven K. Hsu presented on February 2, 2001.

APPROVED:

Redacted for Privacy

Major Professor, representing Electrical and Computer Engineering

Redacted for Privacy

Head of Department of Electrical and Computer Engineering

Redacted for Privacy

Dean of Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for Privacy

Steven K. Hsu, Author

ACKNOWLEDGEMENT

I would first of all like to thank my advisor, Dr. Shih-Lien Lu, for his support and encouragement during my graudate years.

In addtion, I would like to thank Dr. Ram Krishnamurthy for his fruitful comments, motivation, and suggestions. I would also like to thank the rest of my graduate committee, which included Dr. John F. Wager and Dr. Bruce D'Ambrosio for their time.

Lastly, I would like to thank my parents for their advice and encouragement in my education.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF FIGURES

LIST OF FIGURES (continued)

LIST OF TABLES

LIST OF APPENDICES

High-Performance Low-Power Microprocessor Circuits

1. INTRODUCTION

1.1 Background

Power dissipation in a microprocessor can be divided into three components – static, short-circuit, and dynamic power. Currently, the major component of the power dissipation in a CMOS logic based digital system is the charging and discharging the load capacitance of circuit nodes, otherwise known as dynamic power [1]. This dynamic power can be expressed by the well-known equation:

$$Power = \alpha * f * C * V_{dd} * V_{swing} \qquad (1)$$

In this equation α is the activity factor, f is the frequency, C is the total node capacitance, V_{dd} is the supply voltage, and V_{swing} is the switching voltage. A constant, direct path between the supply voltage and ground causes static power. Another component of static power that is becoming more important is transistor leakage. Leakage, or standby current, is power consumed even when the transistors are cut-off. Short-circuit current is caused when CMOS logic stage transistor are in the linear region because of slow rise and fall times of the input. This short-circuit current occurs for a short amount of time and caused from a resistive path from power to ground. Today, decreasing power is equally important as increasing performance. One simple solution to decrease the power is to decrease the supply

voltage. The power is reduced quadtratically by reducing the supply voltage. From *equation (2)* we can see that if we solve for *dt* and set $dQ=C*dV$ to solve for *equation (3)*. However, by decreasing the supply voltage, a designer sacrifices performance thus causing longer delay; this relation is shown in *equation (4)*.

$$dQ / dt = I \qquad (2)$$

$$dt=C*dV / I \qquad (3)$$

$$Delay = C*Vdd / I \qquad (4)$$

In *equation (4)*, *C* is the total switched node capacitance, *Vdd* is the supply voltage, and *I* is the current. Because power and performance are both critical, it is crucial to design the two most important parts of a digital system, the logic and flip-flops, with high-performance and low-power techniques. Delay is not the only merit for performance. In a critical path, the delay may be the crucial factor; however, in some applications, the throughput of a circuit is the real performance metric. Throughput can be defined as the amount of work done in a given amount of time [2]. From *equation (5)*, we see that as frequency increases, the throughput increases. Typically, the faster the frequency a digital system can be clocked, the shorter the delay between the pipelines.

$$Throughput = \#\ of\ operations * frequency \qquad (5)$$

Pipelining is a technique that allows higher throughput, at the expense of latency. Increasing the number of pipeline stages also increase the maximum frequency of a circuits. To compare circuits, this thesis will use two types of metrics. The first

metric will be a power throughput product (PTP) and will be used to compare domino logic styles. This metric is defined in *equation (6).*

$$PTP = Power*(1/throughput) \qquad (6)$$

The second metric is the power delay product (PDP) and will be used to compare flip-flops. This metric is defined by *equation (7).*

$$PDP = Power*delay \qquad (7)$$

Using these two metrics, this thesis will be able to quantify the performance and power trade offs of logic styles and flip-flops.

Dynamic logic has been used mainly for critical paths and functional unit blocks where the throughput is essential. In these high throughput applications, domino has been the logic of choice since it has 30% or more raw performance improvement over static CMOS, especially in high fan-in gates. However, in certain transistor processes, logic only continues at maximum performance until increasing transistor widths becomes ineffective. In addition, due to higher leakage and an increase in keeper size, domino logic performance has been a diminishing benefit over static every process generation [3]. Using a dual threshold process helps alleviate the problem temporarily, however, does not address the long term performance issue -- since both the low and high V_t must scale (thus increasing the leakage) [4]. Thus, designers need to use high-performance logic families or find alternative circuit styles or micro-architectural speed-ups. In all domino schemes, one of the phases is used for pre-charge while the other phase is used for evaluation. This thesis will explore and discuss alternative circuit logic techniques

to exploit both phases of the clock in domino circuit design. Two new domino logic styles will be introduced, double edge triggered (DET) domino and double data rate (DDR) domino. DET and DDR domino yield higher maximum throughput, approximately 38% and 50%, respectively, than conventional domino. In addition, DDR and DET domino enable lower clock frequencies for the same throughput as domino. The need for a much lower frequency clock for the same throughput allows a flexible clock distribution to help the clock distribution design issues with today's microprocessors [5, 6].

For high-performance designs where high throughput is needed, logic is not the only important factor. Flip-flops account for a large percentage of the area as well. Many master-slave structures have been reported, yet due to higher demands for performance and power, new flip-flop structures must be realized. This thesis will also compare the conventional flip-flop, modified C^{2}MOS flip-flop [7], PowerPC 603 flip-flop [8], and a proposed flip-flop [9] in terms of delay and power.

1.2 Scope

This thesis will address circuit design issues and will show simulation results. The scope of this thesis is not to cover new architectures, but just to showcase the new circuit techniques introduced. Chapter 2 covers domino logic, where this thesis covers conventional, DET, and DDR domino. Comparisons and simulation results are also covered in chapter 2. In chapter 3, flip-flops are compared and a novel flip-flop is proposed. Simulation results are presented for the flip-flops in chapter 3. Chapter 4 presents conclusions and recommendations for the proposed circuits.

2. DOMINO LOGIC

2.1 Introduction

There have been many techniques used to increase frequency and throughput in CMOS circuit design. Many dynamic logic styles will improve the performance of the circuit. Dynamic logic replaces the slow PMOS logic transistors with a single clocked PMOS transistor and incorporates a pre-charge and evaluate stage. Domino logic was first introduced by [10] as an alternative dynamic logic style with a simplified clocking scheme and the ability to cascade gates. Today, domino logic is a standard in high-performance microprocessor design. Domino has distinct characteristics -- it is a non-inverting type and can only make a low to high transition on the output. Another type of dynamic logic includes NORA (No Race)[11], which prevents race conditions. This technique also improves upon conventional dynamic logic, but it needs PMOS transistors in some logic stages. Due to the elimination of the static stage in NORA, the performance increases. However, this performance comes at the expense of additional noise and reduced robustness. Reduced robustness causes false discharges that would be amplified through a direct concactinantion of dynamic stages.

Domino also has many different clocking schemes. Skew-tolerant domino has been proposed by [12]. It also removes latch delays by staggering the clocks of domino. In industry design, it is common to see a traditional 2 phase or a higher

performance 4 phase clocking scheme for domino design. Clock-Delayed Domino [13] has also been introduced. In this type of domino, the clock is delayed using a delay element. All of these high-performance techniques are useful in domino design.

2.2 Conventional Domino Logic

In this section, the operation of the conventional domino logic will be described. This thesis will use the term conventional domino for 4 phase clocking. The thesis will not address 2 phase textbook domino.

In a conventional high-performance domino logic gate, there is an NMOS logic part and a PMOS pre-charge part, which is a single clocked PMOS. The output also consists of keeper to reduce noise, charge sharing, and leakage.

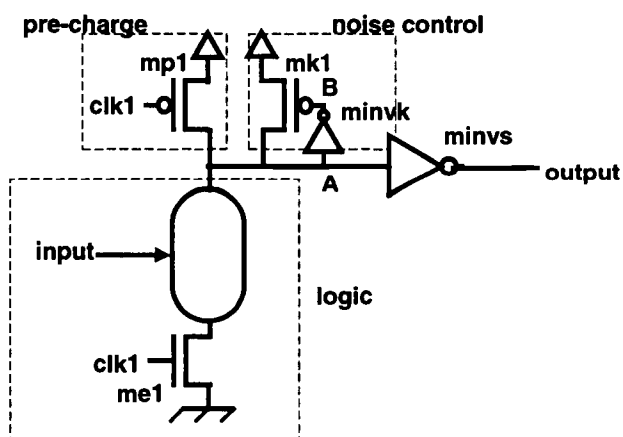

Figure 1. Conventional Domino CMOS Logic

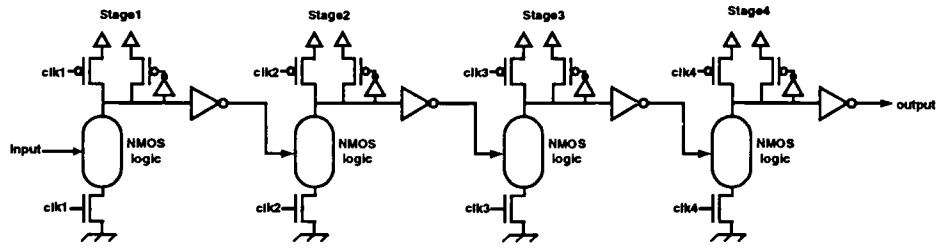

Figure 2. Conventional Domino Logic Pipeline

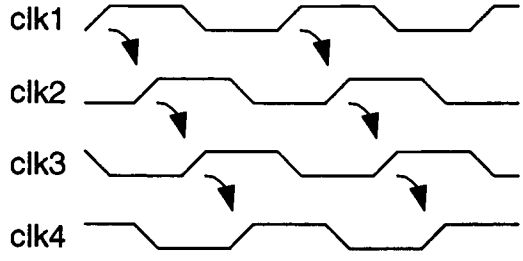

Figure 3. Domino Timing Diagram

For clarity, conventional domino operation is briefly described. As shown in Figure 1, conventional domino consists of an NMOS logic stage and a precharge/keeper output. Domino gates primarily have a speed advantage, as well as an area advantage. The speed advantage is due to the low input capacitance. All of the logic is evaluated with an NMOS pull down network so there is no need to drive additional PMOS gate capacitance on the input contrary to static CMOS. Another advantage of domino is that the area is small compared to an equivalent static logic function since the PMOS stage is replaced with a precharge PMOS. In

conventional domino, there is a precharge phase and then an evaluate phase. The output of a domino gate can make only a low to high transition during the evaluation phase (monotonic). During the precharge phase, typically all inputs must be at low to prevent precharge contention.

In Figure 2, a conventional domino logic pipeline is shown. The clocks are delayed phases of each other. The foot transistors can be removed aggressively in stages 2-4 if the static input node does not transition from a low to high before evaluation occurs. If the low to high transition occurs during the precharge phase, then there will be static power consumption. Figure 3 shows an example timing diagram for 4-phase domino clocking with 50% duty cycle. The term domino is used because as stage 1 evaluates, the result ripples through stage 2, and then stage 3 and stage 4. This clocking scheme consists of a clock and delayed versions of the clock. Clk2 is a delayed version of clk1, and clk3 is actually the complement of clk1. Since there are 4 phases of the clock, there is no need for intermediate latches and the data can ripple through, eliminating clock skew entirely from the cycle time. In this timing scheme, a stage is always evaluating when the data arrives. As shown, when clk1 is low, mp1 is on and precharges node A to the supply voltage in stage 1. When clk1 switches high, me1 switches on to allow the NMOS logic section to evaluate. Mk1, minvk are part of the keeper circuitry. If node A needs to be kept high during evaluation, node B is held low, and mk1 switches and sources current to keep node A high. If the leakage is too great, such as in wide ORs, the keeper can be replaced with a conditional keeper [14] to allow a fast

recovery time without fighting the pull down logic. Although domino provides many advantages, it has its disadvantages. Since there is a precharge phase, domino logic consumes a substantial amount of power. Also explained above, domino logic needs complex clocking schemes and circuitry to correctly function. Also shown in Figure 1, minvs is the static output inverter that can be replaced with just a static logic stage. Typically, this inverter is skewed high to increase performance. Usually, a static inverter is sized with a typical ratio of PMOS transistor width = 2.5 * NMOS transistor width. If a inverter is skewed high, the PMOS transistor width is increased, thus moving the low input trip point higher. However, due to noise issues in domino design, skewing this inverter trades off with noise tolerance of node A that can be amplified on to the output.

2.3 Double Edge Triggered Domino Logic

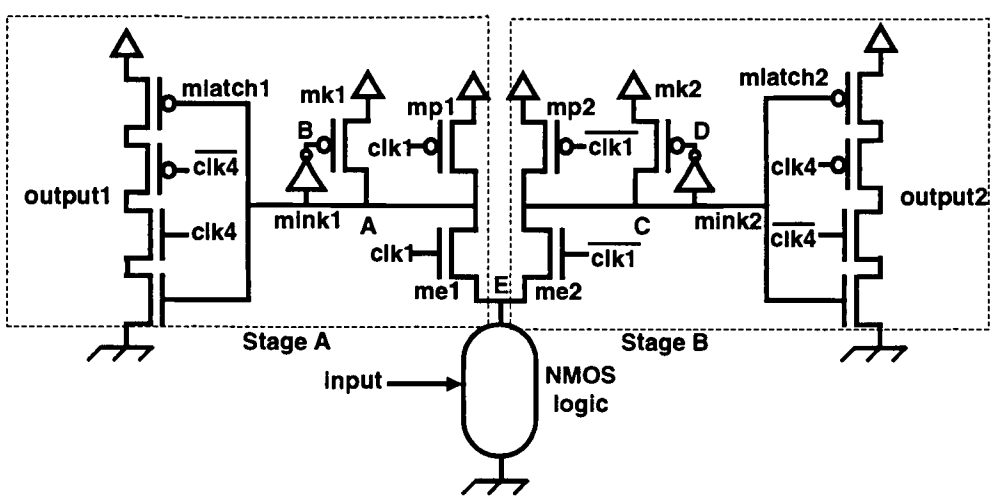

Figure 4. DET Domino Logic Block

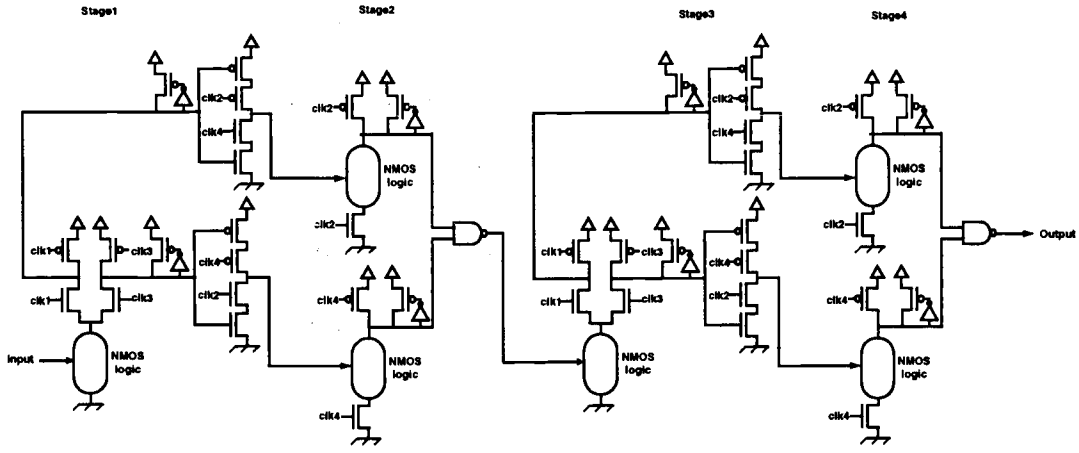

Figure 5. DET Domino Logic Pipeline

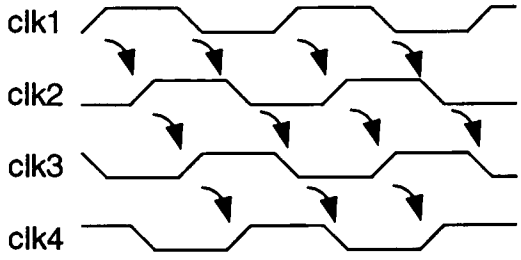

Figure 6. DET Domino Timing Diagram

DET domino logic style consists of an NMOS logic stage and a multiple output latch. A high level diagram of DET domino logic is shown in Figure 4. As shown, it is very similar to conventional domino. DET domino has many of the same advantages as domino. DET domino provides low input capacitance for high performance. However, since there is a complementary block on the output stage,

there is less of an area win over static. The main advantage of DET domino is that the logic can evaluate every phase of the clock. The NMOS evaluate transistors are moved to the top of the NMOS logic stack creating a mirror of conventional precharge/keeper outputs. By moving this NMOS evaluation transistor, the circuit trades performance for noise such as charge sharing. The circuit must be sized appropriately or use a dual rail domino style with cross-couple PMOS's to reduce this charge sharing noise. Stage A and B are mirrors of each other and consist of the precharge/keeper output. Also shown in Figure 4, instead of an inverter, the circuit operates with a latch (mlatch) to create a latch the output. The latch prevents race through conditions from occuring when the next phase of data rolls into the logic gate. DET domino also requires the complement of the clock and establishes this through a local inverter or a separate clock if necessary. Instead of having only an evaluation phase or only a precharge phase like conventional domino, DET domino will always have one stage evaluating while the other stage is precharging. For example, when clk1 is high, stage B can be precharing while stage A is evaluating. Stage B is precharged by mp2, and me2 turns off to isolate the logic from the precharge. When this happens, me1 is on and allows the logic to evaluate the logic function (assuming that stage A was precharged in the previous clock cycle). When the next phase switches and clk1 is low, stage B is evaluating (since it precharged last phase) and stage A is precharging for when clk1 switches high next. All stages of a DET domino pipeline evaluate on both phases of clock. In this circuit configuration, DET domino uses both edges of the clock to evaluate

and uses only one NMOS evaluation stack. However, like latched domino logic, the data inputs must be set-up before evaluation occurs. This is a disadvantage since it does not allow time-borrowing. Unlike conventional domino, the inputs do not have to be low when the logic is precharging since they are isolated.

In Figure 5, a DET domino logic pipeline shows the interface between stage 4 to stage 1. Each alternating stage uses a DET domino logic block. Stage 2 is just a DDR domino logic block. The reason that a designer must alternate between the two types of blocks is that there is a potential race through condition when two DET domino stages are cascaded. In Figure 6, the clk timing diagram is shown. As shown, a 4 phase clk timing sequence is similar to domino. Although it is 4 phase, the local inverters in each logic section create 2x the phases making the total 8. This is because each logic block needs clk signal and its complement.

2.4 Double Data Rate Domino Logic

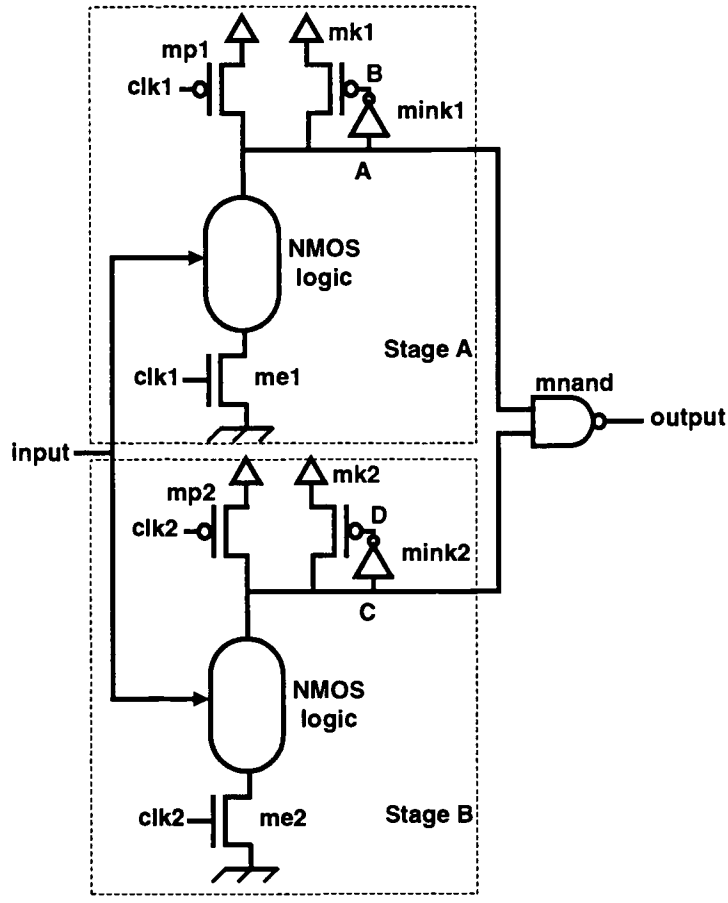

Figure 7. DDR Domino CMOS Logic

Double Data Rate Domino (DDR) Logic is also very similar to conventional domino. The concept includes having a duplicate logic function that can be used in parallel. The output can be used as a NAND to be latched, or can be kept separate with just inverters feeding into another domino stage. A circuit diagram of DDR domino is shown in Figure 7. The input is fed into both logic functions.

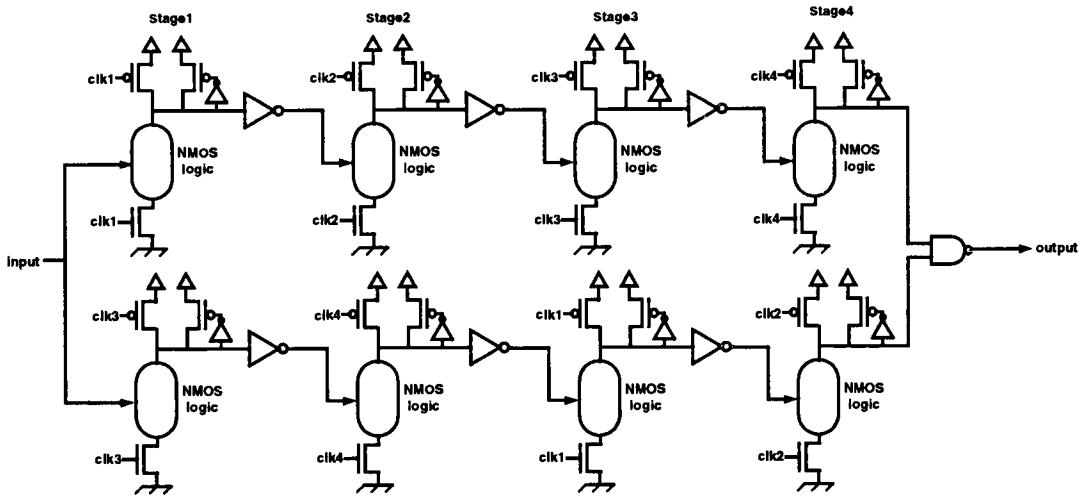

Figure 8. DDR Domino Logic Pipeline

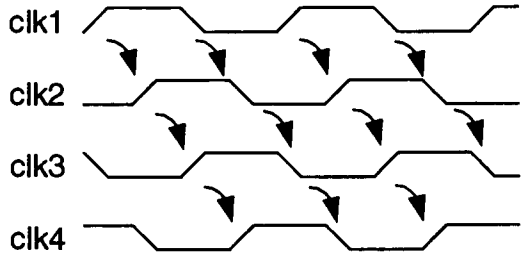

Figure 9. DDR Domino Timing Diagram

Figure 8 shows a logic pipeline for DDR domino. As one can see, it too is very similar to conventional domino. If an output is needed, then the two sides can be NANDed together as in stage 4. By keeping multiple outputs, this technique reduces the output capacitance. Having less output capacitance allows for higher frequencies. This also allows for time borrowing between stages if needed. In this

case, shown in Figure 9, clk1 and clk3 are non-overlapping 50% duty cycle clocks. The input will see twice the capacitance, however, if the output and input are kept separate the input capacitance will be the same as conventional domino. To describe the operation, as shown in Figure 3, assume that node A has precharged already in the previous clk1 phase. When data arrives, stage A is ready and evaluates when clk1 goes high. However, during this same phase, clk3 is low and node C is precharged. Stages A and B are NANDed and the correct output is propagated.

The NAND logic is summarized in Table 1. When clk1 goes low to precharge node A, clk3 goes high and evaluates stage C. Therefore, during every phase logic is done and propagated through. There is no wasted precharge time. The throughput is effectively doubled; however, the critical path delay still will be the same or slightly slower since the static inverter is now replaced with a static NAND. The throughput increase must be justified over the area and power impact that DDR domino brings. Case 1 is not applicable since nodes A or C will be precharged during a phase of the clock. Cases 2 & 3 are the same situation. In this situation, node A or C is evaluated as 0 while the opposite node is precharged. In either case, the node that evaluates to 0 controls the output to be a 1. In the final case 4, both node A & C are 1. In this situation, either nodes A or C is evaluated to 1, and the opposite node is precharged. Since the evaluation is at 1, the output is driven to 0 to produce the correct result.

NAND Output Truth Table				
Case	Input 1 (node A)	Input 2 (node C)	Output	Comment
1	0	0	1	This case is not applicable since one of the nodes will always be precharged.
2	0	1	1	Node A is evaluated as 0 and node C is precharged.
3	1	0	1	Node C is evaluated as 0 and node A is precharged.
4	1	1	0	Node C or A is evaluated as 1 and the other node is precharged.

Table 1. NAND Precharge Buffer Truth Table

Since domino logic discharges the output node (while in precharge phase), the activity factor for the output node is very high. For high fan out gates with an activity facto of 1, there is an excess of power dissipation in conventional domino. In order to increase the throughput, the frequency also must be increased.

2.5 Qualitative Comparison

Overall, Table 2 qualitatively covers conventional, DET, and DDR domino. In terms of area, conventional domino is less constraining. Both DET and DDR domino require approximately double the amount of transistors. Power consumption is typically larger if area is larger. DDR domino consumes the largest amount of power since the dynamic precharge dominates in domino and is twice the area. Domino and DDR domino both enable very high frequencies. DET

domino is slightly slower because latches are necessary at the outputs. Although clock frequencies are approximately within range of each other, DET and DDR domino throughput is far higher since evaluation occurs every phase of the clock.

Qualitative Comparison						
Logic Style	Area	Power	Frequency	Throughput	Complexity	Robustness
Conventional	*Low*	*Low*	*High*	*Low*	*Low*	*High*
DET	*High*	*Medium*	*Medium*	*Medium*	*Medium*	*Medium*
DDR	*High*	*High*	*High*	*High*	*Medium*	*High*

Table 2. Qualitative Comparison

Adding more transistors only creates more complexity; therefore, conventional domino is the simplest. Robustness of conventional domino and DDR domino is approximately the same. However, in DDR and DET domino, strict 50% duty cycles must be obeyed and any overlapping clocks (which should otherwise be non-overlapping) will create contention. This is typical in any double-edge triggered timing such as double-edge triggered flip-flops. The robustness in DET domino is slightly less; due to the charge sharing that occurs during evaluation.

2.6 Simulation

For the simulation, a TSMC 0.25μm CMOS modified process model from MOSIS [15] was used. A small modification to the process file included the introduction of the ACM=3 and HDIF=0.35μm to allow the simulation to calculate the source and drain area automatically. This allowed a reduction in time writing the spice deck because of AD, AS, PD, and PS are removed from the spice input file and are replaced with just a single GEO parameter. To exploit this ACM parameter, GEO could be set to values 0 through 3. The following are the associated meanings:

GEO = 0 (default) drain and source are not shared

GEO = 1 drain of device is shared with another device

GEO = 2 source of device is shared with another device

GEO =3 source and drain are shared with another device

The 0.25µm process contained a single Vt, the NMOS Vt was 0.44V, while the PMOS Vt was –0.66V. The voltage supply used was 2.5V. All simulations were done at a nominal 25°C. Table 3 shows the breakdown of the process model and simulation parameters.

Technology	TSMC CMOS
Channel Length	0.25µm
Model Level	49
NMOS V*t*	0.44
PMOS V*t*	-0.66
Supply Voltage	2.5V
Temperature	25°C

Table 3. HSPICE Simulation Model

2.7 Full Adder Circuits

The circuit used to test the logic styles was a full adder. Full adders are the principal building block for arithmetic units in microprocessors. A full adder is composed of a carry and a sum stage. *Equations 8* and *9* show the logic of a full adder.

$$Sum = A \oplus B \oplus C \qquad (8)$$

$$Carry = (A \wedge B) \vee (A \wedge C) \vee (B \wedge C) \qquad (9)$$

The logic has three inputs -- A, B, and C -- and the truth table of this full adder is shown in Table 4. In conventional domino logic, the output is low during precharge, therefore it is infinitely skewed low when evaluation occurs. In DDR & DET domino, the output is skewed infinitely towards the previous value of the output.

Conventional and DET domino full adders were designed and sized for performance, power, and noise. Figure 10 shows the design of the conventional domino full adder. The conventional domino full adder is composed of 26 transistors. In Figure 11, the DET domino full adder is shown. The DET domino full adder is composed of 55 transistors, whereas the DDR domino full adder consists of 52. The DDR domino full adder is shown in Figure 12. The carry logic creates part of the sum logic with carry_bar. Therefore, the sum transistor logic can be simplified as shown in *equation 10.*

$$Sum = (Carry\_bar \wedge (A \vee B \vee C)) \vee (A \wedge B \wedge C) \qquad (10)$$

Full Adder Truth Table				
Ain	Bin	Cin	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table 4. Full Adder Truth Table

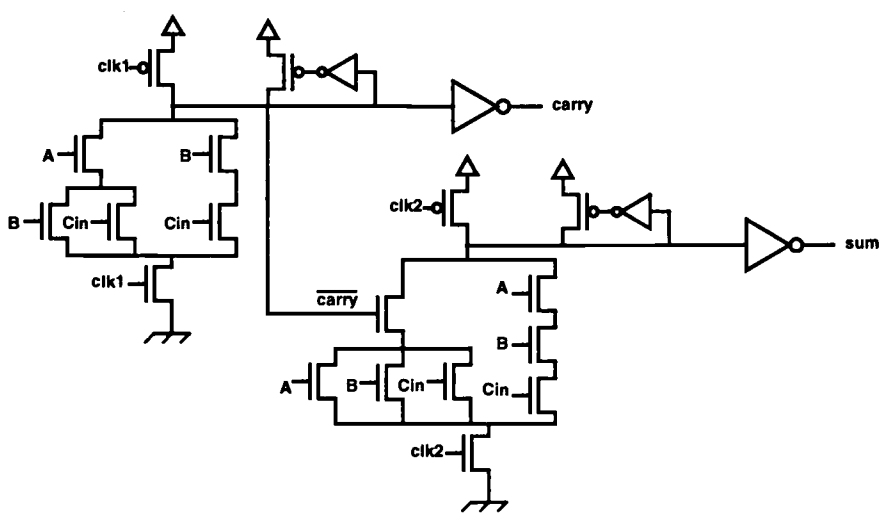

Figure 10. Conventional Domino Full Adder

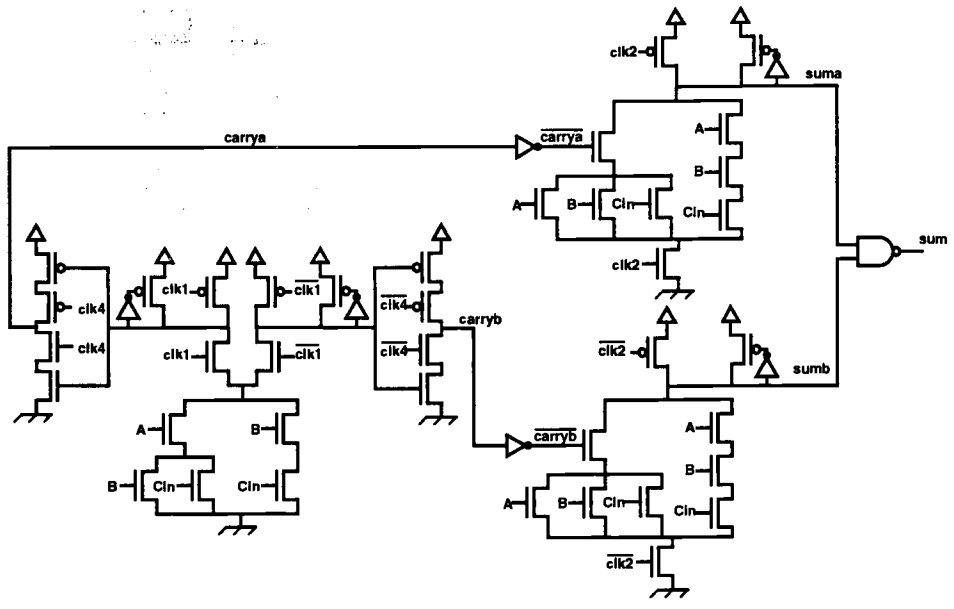

Figure 11. DET Domino Full Adder

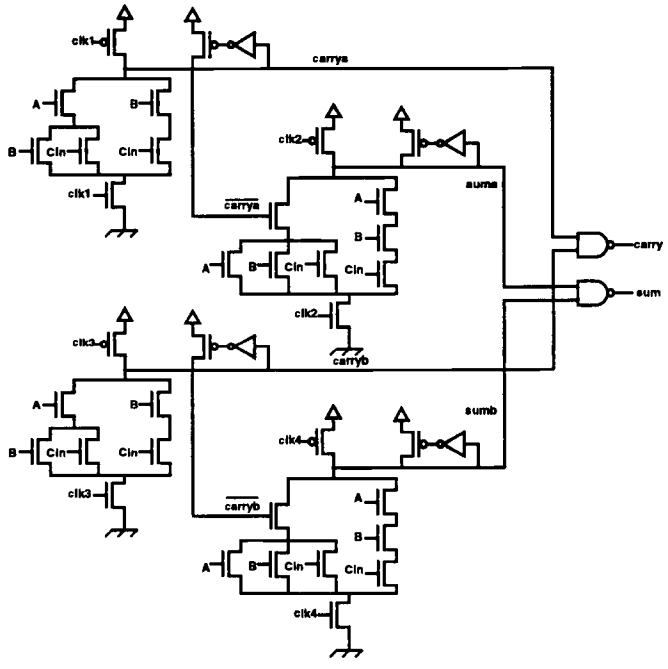

Figure 12. DDR Domino Full Adder

2.9 Full Adder Comparison

Each full adder was simulated in a small test bench to validate functionality and performance. All three inputs to the full adder were toggled so that all possible inputs are exercised. Therefore, all inputs have an equal probability of switching. The carry output and sum outputs have a fan out of 5μm PMOS capacitance. This setup allowed the full adder to be optimized for the 8-bit ripple carry adder. Power was measured by taking the average current consumption for one clock cycle. The table shows the clock power and logic power for a fixed throughput of 800Mops (Mega operations per second). In Table 5, there is a comparison of power for conventional, DET, and DDR domino full adder. As shown, the DET clock power is slightly larger than conventional and DDR domino clock power, due to the extra load capacitance associated with the latches. Since the comparison is a fixed throughput, the DET and DDR domino full adders are operating at half the frequency of the conventional full adder. Since this is the case, the clock power of conventional and DDR are approximately equal since conventional is 2x the frequency while DDR is double the load capacitance. The overall power consumption shows the conventional to be the lowest and the DDR domino to be the highest.

Full Adder Power Comparison (Fixed Throughput)			
Type	Clock Power (μW)	Logic Power (μW)	Total Power (μW)
Domino (Fig. 10)	31.6	118.3	149.9
DET Domino (Fig. 11)	41.8	132.5	174.3
DDR Domino (Fig. 12)	32.5	183.2	215.7

Table 5. Full Adder Power Comparison

Table 6 shows a comparison of total power and PTP for conventional, DET, and DDR domino. As shown in Figure 13, the power consumption of DDR domino is the highest among all three styles. DET domino power consumption falls between the two. Figure 14 shows that for a fixed throughput of 800Mops, conventional domino is sufficient in terms of power and throughput.

Full Adder Overall Comparison (Fixed Throughput)				
Type	# of transistors	Frequency (MHz)	Total Power (μW)	PTP (fJ)
Domino	26	800	149.9	187.4
DET Domino	55	400	174.3	217.9
DDR Domino	52	400	215.7	269.6

Table 6. Full Adder PTP Comparison

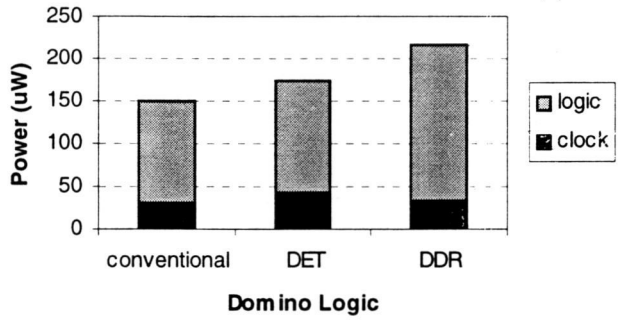

Figure 13. Full Adder Total Power Comparison

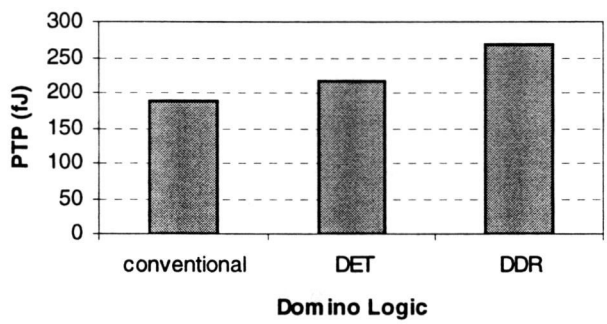

Figure 14. Full Adder PTP Comparison

2.10 8-bit Ripple Carry Adders

The architecture was composed of a simple 8-bit pipelined ripple carry adder. In a ripple carry adder, the carry of the first full adder becomes the input of the second full adder. The sum output is ripped off the ripple carry adder and de-skewed until all sums are available at the same time. Figure 15 shows the architecture for the conventional domino 8-bit adder. Figure 16 shows the architecture of the DET domino 8-bit adder. Figure 17 shows the architecture of the DDR domino 8-bit adder. As shown with the DDR adder, the NAND gates siphon the sum outputs. These architectures are very similar since they all use 4 phases.

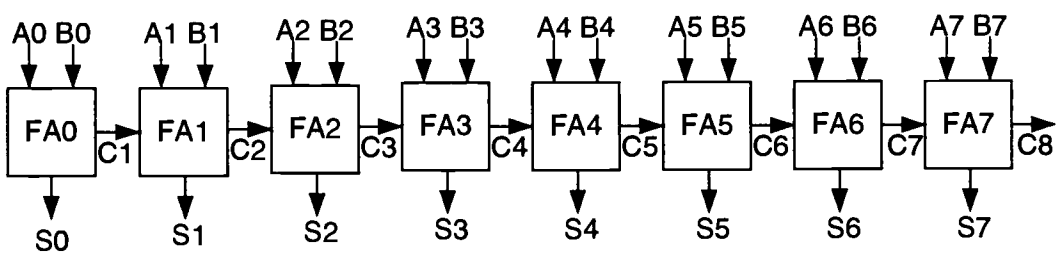

Figure 15. Conventional Ripple Carry Adder

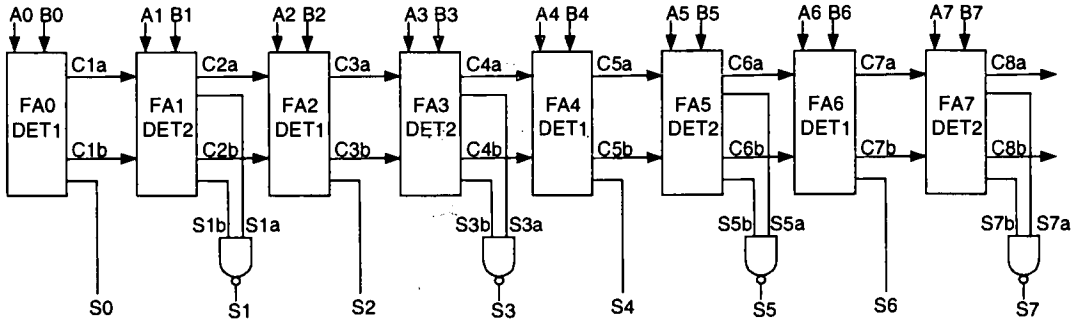

Figure 16. DET Domino Ripple Carry Adder

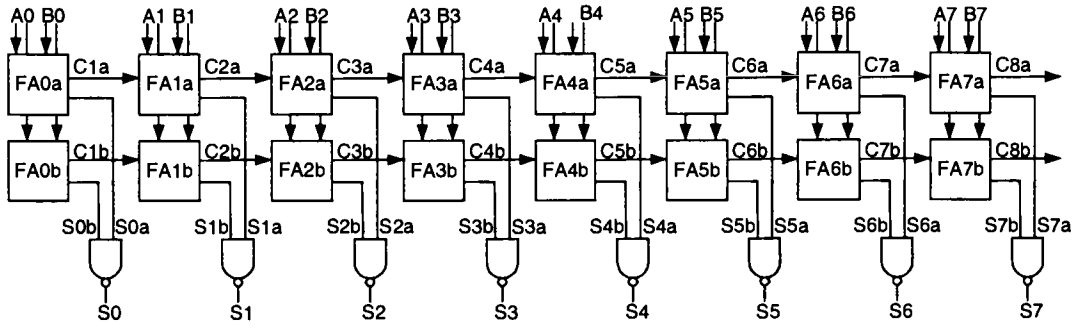

Figure 17. DDR Domino Ripple Carry Adder

In Table 7, the ripple carry adder characteristics are compared. Conventional domino adder was composed of the least amount of area with only 208 transistors. DDR and DET domino were approximately twice the transistor count with 416 and 452, respectively. The number of full adders refers to the number of physical full adders (FA). In DET domino's case, a full adder is counted as 1.5 FA since either two carry stages are needed or two sum stages are needed. Figure 18 summarizes

the transistor count of each type of adder. Although DET domino was composed of less full adder cells, the extra latch circuitry dominated the NMOS pulldown logic savings.

Ripple Carry Adder Comparison			
Circuit Style	Total # of transistors	# of phases	# of full adders
Domino	208	4	8
DET Domino	452	4	12
DDR Domino	416	4	16

Table 7. Comparison of 8-bit Adders

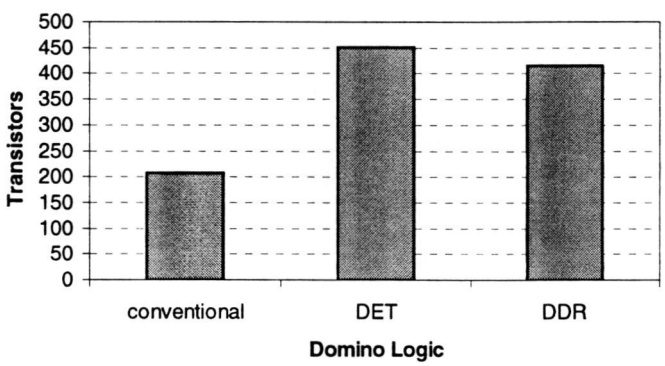

Figure 18. Total Transistor Count

In Table 8, the clock and logic power are shown. Logic power is measured for an activity of 1 and an activity of 0. With an activity of 0, power is measured for both cases where the output is always 1 or always 0. In conventional domino,

the worst-case power occurs when the output is always 1, since in every cycle the gate precharges, forcing the output to be 0. When the evaluation phase occurs, the domino node is discharged, forcing the output to be 1. If the fan out dominates the power, this could force conventional domino to consume excess power. In DDR domino, the output of the NAND will always stay at the correct value, thus saving some additional load power. Figure 20 shows the worst-case power consumption for all adders. The DET domino adder has the greatest power consumption when activity is 1, while conventional and DDR domino have the most power consumption when activity is 0 and always 1 (since they both always discharge).

However, in DET domino, since there is shared logic, the range of power consumption with activity is less than conventional and DDR domino. When the domino gate does not evaluate every clock cycle, the minimum amount of power is consumed because the domino node is not discharged.

In Table 9, the ripple carry adder simulation results are shown for the maximum clocking frequency. Figure 21 show that DDR domino has the highest maximum throughput of 2Gops (Giga operations per second) over and beyond conventional domino. Since conventional domino evaluates only once per clock cycle, the maximum throughput is half that of DDR domino. DET domino falls at 1.6Gops since the maximum clocking frequency is less than that of DDR and conventional. PTP is a valuable metric because DDR and DET domino consume approximately twice the transistor count. As shown in Figure 22, DET domino has a PTP almost independent of activity factor and value. DDR domino operates less

than conventional domino in terms of PTP. The highest points on the graph refer to the maximum power consumed, while the lowest points are where the lowest power is consumed.

In a highly pipelined design, where domino is not sufficient, a designer may choose to use DDR domino if the area and energy is not a concern. However, DET domino could be used if the energy needs to be reduced and the data activity has a high probability of having a value of 1.

Power Simulation (Max. Throughput)				
Nominal Conditions	Clock Power (mW)	Logic Power α=0 always 1 (mW)	Logic Power α=1 (mW)	Logic Power α=0 always 0 (mW)
Domino	0.81	3.66	3.19	2.56
DET Domino	1.63	3.41	4.63	3.15
DDR Domino	1.61	7.81	6.16	6.10

Table 8. Ripple Carry Adder Power Analysis

Ripple Carry Adder Simulation (Max. Throughput)					
Nominal Conditions	Max. Frequency (MHz)	Latency (ns)	Max. Throughput (Gops)	Worst Case Total Power (mW)	PTP (pJ)
Domino	1000	2.44	1.0	4.47	4.47
DET Domino	800	3.31	1.6	6.26	4.14
DDR Domino	1000	2.51	2.0	9.42	4.71

Table 9. 8-bit Ripple Carry Adder Simulation

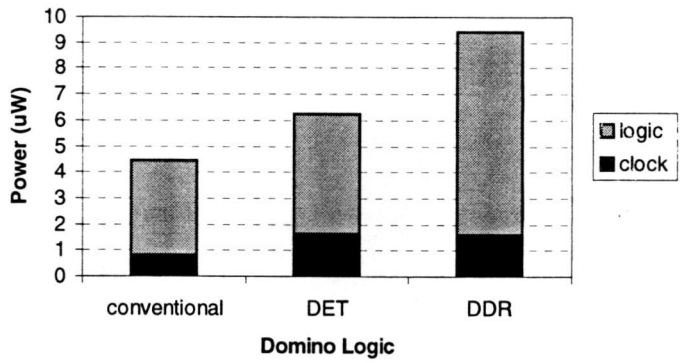

Figure 19. Ripple Carry Worst Case Power Comparison

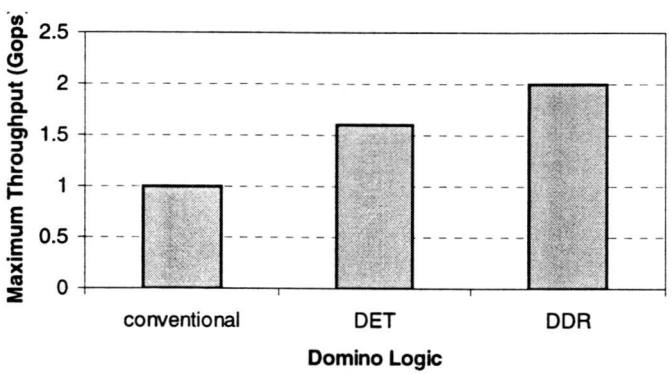

Figure 20. Ripple Carry Maximum Throughput Comparison

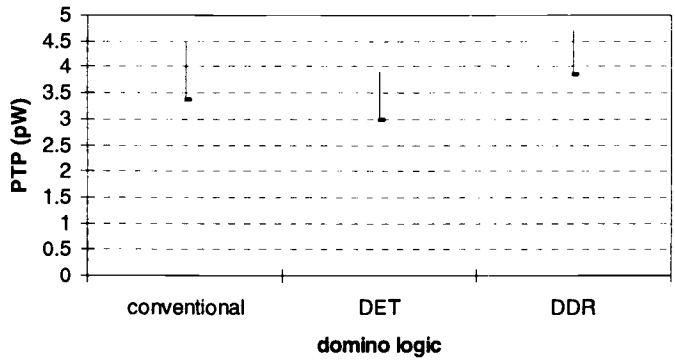

Figure 21. Ripple Carry PTP Comparison

3. FLIP-FLOPS

3.1 Introduction

In today's clocked synchronous circuits, a large percentage (up to 30-40%) of the power loss is due to the clock load. Therefore, it is very important to design the registers (flip-flops and latches) in a synchronous digital system for optimal performance and power consumption. These flip-flops need to have a small sampling window, consume low power, and have a small clock load. A novel high-performance low-power CMOS master-slave flip-flop is proposed here. The proposed flip-flop consumes very low power, while having a very small clock load and data load. In the HSPICE simulations, the new flip-flop has an optimal power-delay product better than other reported master-slave structures. The proposed flip-flop is compared to other reported master-slave flip-flops.

A flip-flop has a total power composed of internal power, data power, and clock power. As mentioned before, a flip-flop's dynamic power dissipation is dependent upon the switching activity. For data power, an α of 0 could equate to data signals of all 0s or all 1s for the period of time power is measured. Similarly an α of 1 equates to maximum switching activity, where data input switches from 0 to 1 alternating every cycle. On the other hand clock power always has an activity factor of 1, where there is no clock gating. Therefore, clock power has become a more important issue in designing an efficient flip-flop. In addition, from the power equation we know that lowering supply voltage is the most effective way to

reduce the power consumption [16; 17] of a CMOS circuit. Any new flip-flop must work well at low supply voltages and will need to be able to complement new logic styles, incorporating logic into them without penalty.

3.2 Flip-Flop Circuits

The previously reported flip-flops are shown in Figures 22-25. The conventional flip-flop is a standard 16-transistor structure. This robust flip-flop can be seen in many textbooks and is the most popular master-slave structure. The modified C²MOS flip-flop did not include a local clock buffer, contrary to [7]. We removed this to see the "real" clock power, because any of these master-slave flip-flop structures could use the same technique. However, the modified C²MOS flip-flop is suitable for low-power applications with a local clock buffer, due to its small internal power dissipation. The PowerPC 603 flip-flop contains 18 transistors and consumes very low power.

The proposed flip-flop, shown in Figure 26, has many interesting properties. To lower clock power consumption, the flip-flop uses only NMOS clocked transistors. The gate of NMOS transistors effectively has less capacitance than a PMOS, yielding faster switching and lower power for the clock circuitry. Although the proposed flip-flop uses only NMOS pass transistors, we remove the threshold drop and static power consumption by the cross-coupled PMOS [18] that is formed in the latch circuitry. The latch circuitry is created by two cross-coupled True-

Single Phase Clocking (TSPC) SN stages [19]; however, our structure is pseudo-static and needs the complement of the clock. The proposed flip-flop is a SN-SN configuration, and a race limited SP-SN flip-flop can be formed. There is no fighting within the internal circuitry, and the flip-flop can be optimized if the complement of the signal is available.

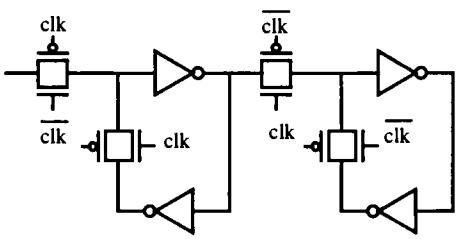

Figure 22. Conventional Master Slave Flip-Flop

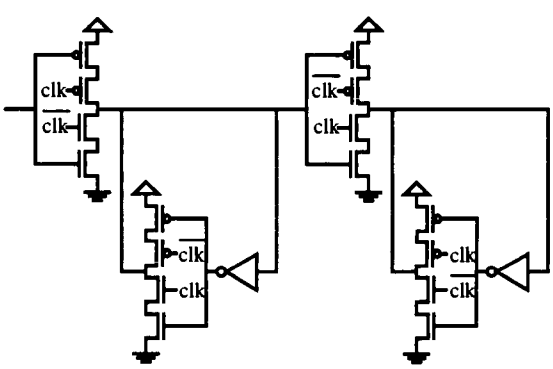

Figure 23. Modified C2MOS Flip-Flop

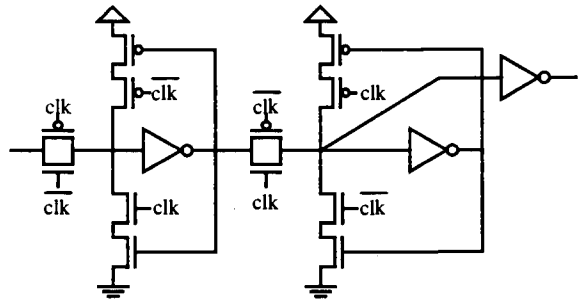

Figure 24. PowerPC Flip-Flop

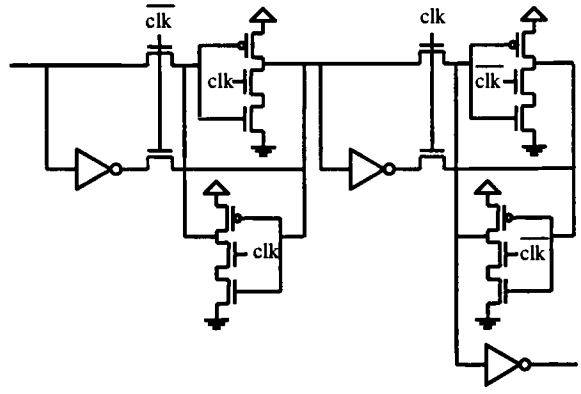

Figure 25. Proposed Flip-Flop

3.3 Simulation

The simulation methodology was very similar to that of the proposed method in [7]. As seen in Table 10, the technology and the HSPICE simulation are shown. The simulations used a 0.35μm MOSIS [15] BSIM model Level 49 at a voltage of 2.0V. Each flip-flop is optimized in HSPICE with the embedded optimizer for total average power and delay.

For clarity, this thesis will review the definitions of internal power, clock power, and data power. The internal power is the internal circuitry power consumption only and does not include the output load capacitance. The clock power is the power consumed by only the clocked transistors in the circuit. The data power is the power consumed by driving the flip-flop. The simulation measured the average power using a similar test bench as in [7] at 100MHz, an α of 0.5 for 16 clock cycles, and a load of 200fF.

3.4 Flip-Flop Test Bench

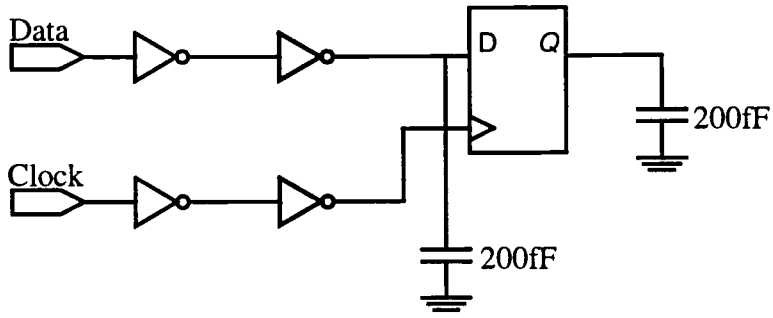

Figure 26. Flip-Flop Test Bench

To measure the performance, a different metric as proposed in [7]. Although an optimal data-output delay is a very suitable metric, our delay metric was the maximum clocking frequency of a ring oscillator composed of one inverter and one

flip-flop [20]. We used the same inverter size for each flip-flop, and as long as the oscillation was twice the clock frequency, the circuit was operational.

Simulation Parameters	
Technology: Channel Length 0.35μm V_{tn} = 0.5V V_{tp} = -0.7V MOSFET Model MOSIS Level 49 V_{dd} = 2.0V T = 25°C	Simulation: (1) Data/Clock slopes of ideal signals: 100ps (2) Clock duty-cycle: 50% (3) Delay Calculation: between 50% points (4) Data sequences: 16 clock cycles (5) Clock Frequency: 100MHz

Table 10. Flip-Flop Simulation Parameters

Upon meeting the oscillation requirements, we assumed:

$$1 / \text{Max. Freq.} = T_{delay} = T_{setup} + T_{clkQ} + T_{inv} + T_{skew}. \quad (11)$$

This T_{delay} metric, in (11), shows the performance of the flip-flop in normal operation, although includes the delay of the inverter.

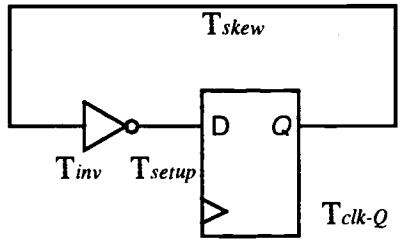

Figure 27. Flip-Flop Maximum Clocking Frequency

3.5 Results

In Table 11, the simulation results are shown for an activity factor of 0.5. One can see that the proposed flip-flop consumes very low power and has a suitable delay. The delay is also shown in Figure 28. In Figure 29, we can see the total power consumed. The conventional flip-flop shows robust results in terms of speed and power. Although the modified C²MOS consumes the most power, it is larger due to the clock power, where no local clock generator is used. However, it consumes very low internal power compared to the other structures. The PowerPC 603 consumes low power, but has limited performance. The proposed flip-flop consumes the lowest average power, while having a small clock load and a very small data load. The data load reduction from the other circuits is due to the parallel driving property used by the proposed flip-flop.

General Characteristics (activity =0.5)								
Nominal Conditions	# of transistors	Total transistor width (μm)	Internal Power (μW)	Clock Power (μW)	Data Power (μW)	Total Power (μW)	Delay (ps)	PDP (fJ)
Conventional	16	40.6	13.1	6.4	4.6	24.1	877.2	21.1
Modified C^2MOS	20	62.4	10.6	15.3	2.3	28.2	909.1	25.6
PowerPC 603	18	45.6	13.3	5.1	4.1	22.5	1063.8	23.9
Proposed (this work)	22	39.7	12.6	5.6	1.6	19.8	1000.0	19.8

Table 11. Flip-Flop Characteristics

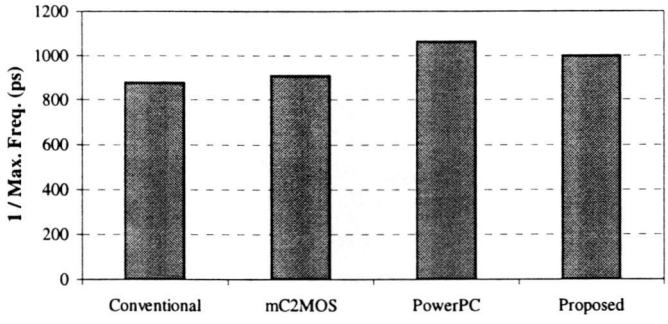

Figure 28. Overall Delay Comparison

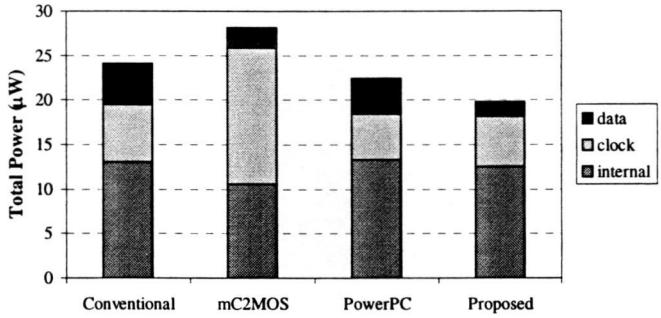

Figure 29. Total Power Consumption

From Figure 30, one can see the total power vs. delay for each optimized flip-flop. This graph shows the power for different activity factors ranging from 0 to 1. The marked areas represent $\alpha = 0.5$. Figure 6 shows the power-delay product for each flip-flop. From Figure 31, the proposed flip-flop has an optimal power-delay product better than the other previous structures.

In Figure 32, constant voltage scaling was performed on the conventional and the proposed flip-flop. Although the proposed flip-flop consumes low power, the proposed flip-flop does not operate for the following condition $V_t > V_{dd}-V_t$. To obtain operation at extremely low voltages, low-threshold pass transistors must be used. However, our proposed flip-flop does considerably well with the conventional in terms of power and delay.

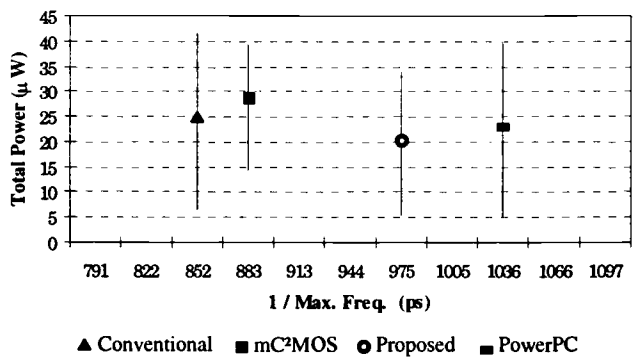

Figure 30. Total Power Range vs. delay

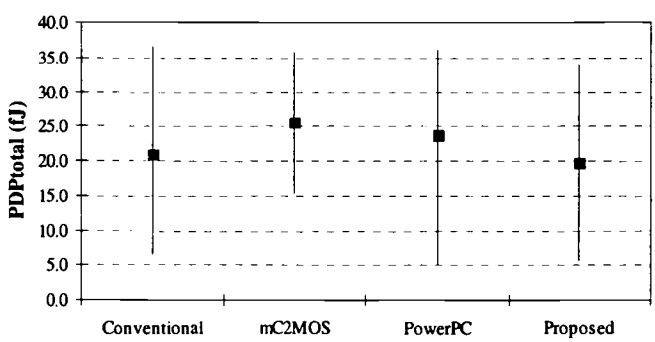

Figure 31. Ranges of PDPtotal

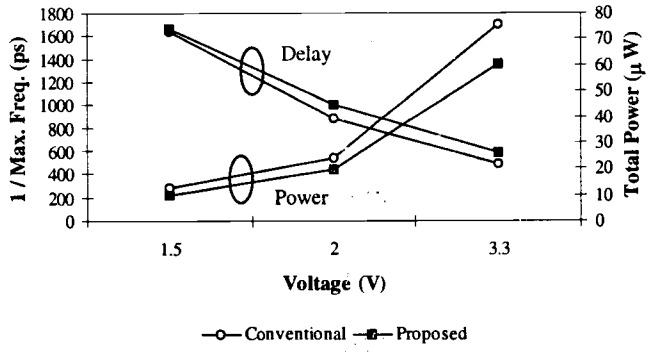

Figure 32. Constant Voltage Scaling

4. RECOMMENDATIONS & CONCLUSION

4.1 Recommendations

Although this thesis has showcased DDR and DET logic as domino logic solutions, they are both general-purpose solutions to any dynamic logic. Such examples could include Current Sensing Differential Logic [21], Differential Current Switch Logic [22], Pre-Charged Pass Transistor Logic [23], and N-Logic True Single Phase Logic [24]. These dynamic logic families are some candidates to have double data rate and double edge triggered concepts applied. This would need further investigation and experimentation to see how much impact these techniques would have. DET domino chage sharing could be removed by using DET Pre-Charged Pass Transistor Logic. In addition, another appropriate solution would be dual-rail DET domino with cross-coupled PMOS's for charge sharing and leakage control.

The proposed flip-flop could work very well with Complementary Pass Logic (CPL) styles [25, 26, 27]. Since the complement of the data input is produced and needed, the extra input inverter can be moved to the output to produce the complement. Typically, in order to have low voltage operation, CPL must have low-threshold NMOS pass-transistors. With this type of process, the proposed flip-flop also increases in performance as well. A very typical application could be CPL with dual-V_t NMOS only. The second V_t for NMOS is used for the pass transistors, and is achieved via substrate bias. Dual-V_t CMOS process yields more

performance, however, it is more expensive since it requires a twin tub. Either multiple-V_t schemes will work well with the proposed flip-flop. However, we must point out that the other flip-flops do not benefit as much as the proposed with the dual-V_t NMOS only scheme.

4.2 Conclusion

High-performance low-power CMOS circuits were presented. DET and DDR domino were presented and compared with conventional domino. Using a 0.25μm CMOS MOSIS model, HSPICE simulation confirmed that DET and DDR domino yield higher maximum throughput, approximately 38% and 50%, respectively, than conventional domino and thus enable lower clock frequencies for a given throughput. Domino logic styles were presented, and other parts of a digital system were investigated.

Logic is not the only critical design issue; flip-flops also need to be investigated. A novel CMOS master-slave flip-flop was presented. The proposed flip-flop consumes very low power, while having a very small clock load and data load. The proposed flip-flop was compared to other reported master-slave flip-flops. Using a 0.35μm CMOS MOSIS model, the new flip-flop was simulated to have an optimal power-delay product better than other reported master-slave structures.

REFERENCES

[1] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*. Menlo Park, California: Addison Wesley Publishing Company, 1993.

[2] J. Hennessy and D. Patterson, *Computer Architecture: A Quantitative Approach*. San Francisco, California: Morgan Kaufmann, 1996.

[3] S. Thompson, I. Young, J. Greason, and M. Bohr, "Dual Threshold Voltages and Substrate Bias: Keys to High Performance, Low Power, 0.1μm Logic Desgins," *1997 Symposium on VLSI Technology Digest of Technical Papers*, pp. 69-70.

[4] V. De and S. Borkar, "Technology and Design Challenges for Low Power and High Performance," *1999 International Symposium of Low Power Electronic Devices*, pp. 163-168.

[5] Y. Liu, S. Nassif, L. Pileggi, and A. Strojwas, "Impact of Interconnect Variation on the Clock Skew of a Gigahertz Microprocessor," *Design Automation Conference 2000*, pp. 168-171.

[6] S. Rusu and S. Tam, "Clock Generation and Distribution for the First IA-64 Microprocessor," *International Solid State Circuits Conference 2000*, pp. 176-177.

[7] V. Stojanovic and V. Oklobdzija, "Comparative Analysis of Master-Slave Latches and Flip-Flops for High-performance and Low-power Systems," *IEEE J. Solid State Circuits*, vol. 34, no. 4, pp. 536-548, April 1999.

[8] G. Gerosa, S. Gary, C. Dietz, D. Pham, K. Hoover, J. Alvarez, H. Sanchez, P. Ippolito, T. Ngo, S. Litch, J. Eno, J. Golab, N. Vanderschaaf, and J. Kathle, "2.2W, 80MHz Superscalar RISC processor," *IEEE J. Solid State Circuits*, vol. 29, no. 12, pp. 1440-1454, Dec. 1994.

[9] S. Hsu and S.L. Lu, "A Novel High-Performance Low Power CMOS Master-Slave Flip-Flop," *Proceedings of 12th Annual IEEE International ASIC/SOC Conference*, September 1999.

[10] R. H. Krambeck, C. Lee, and H. Law, "High-Speed Compact Circuits with CMOS," *IEEE J. Solid State Circuits*, vol. 17, no. 3, pp. 614-619, June 1982.

REFERENCES (continued)

[11] N. Goncalves and H. De Man, "NORA: A Racefree Dynamic CMOS Technique for Pipelined Logic Structures," *IEEE J. Solid State Circuits*, vol. SC-18, no. 3, pp. 261-266, June 1983.

[12] D. Harris and M. Horowitz, "Skew-Tolerant Domino Circuits," *IEEE J. Solid State Circuits*, vol. 32, no. 11, pp. 1072-1711, November 1997.

[13] G. Yee, et al., "Clock-Delayed Domino for Adder and Combinational Logic Design," *Proceedings of the 1996 International Conference on Computer Design*, Austin, Texas, pp. 332-337.

[14] A. Alvandpour, P. Larsson-Edefors, and C. Svensson, "A Leakage Tolerant Multi-Phase Keeper for Wide Domino OR's," *The 6th IEEE International Conference on Electronics, Circuits, and Systems 1999*, vol. 1, pp. 209-212.

[15] <http//:www.mosis.edu>.

[16] S. Mutoh, S. Shigematsu, Y. Matsuya, H. Fukuda, and J. Yamada, "A 1V Multi-threshold Voltage CMOS DSP with an Efficient Power Management Technique for Mobile Phone Application," *ISSCC Digest of Technical Papers*, FA 10.4, Feb. 1996.

[17] R. Gonzalez, B. Gordon, and M. Horowitz, "Supply and Threshold Voltage Scaling for Low-power CMOS, " *IEEE J. Solid State Circuits*, vol. 32, no. 8, pp. 1210-1216, August 1997.

[18] M. Afghahi, "A Robust Single Phase Clocking for Low-power, High-speed VLSI Applications," *IEEE J. Solid State Circuits*, vol. 31, no. 2, pp. 247-254, Feb. 1996.

[19] J. Yuan and C. Svensson, "New Single-clock CMOS Latches and Flip-flops with Improved Speed and Power Savings," *IEEE J. Solid State Circuits*, vol. 32, no. 1, pp. 62-69, Jan. 1997.

[20] M. Afghahi and J. Yuan, "Double Edge Triggered D Flip-flops for High Speed Low-power Applications," *IEEE J. Solid State Circuits*, vol. 26, no. 8, pp. 1168-1170, August 1991.

REFERENCES (continued)

[21] J. Park, J. Lee, and W. Kim, "Current Sensing Differential Logic: A CMOS Logic for High Reliability and Flexibility," *IEEE J. Solid State Circuits*, vol. 34, no. 6, pp. 904-908, June 1999.

[22] D. Somasekhar and K. Roy, "Differential Current Switch Logic: A Low Power DCVS Logic Family," *IEEE J. Solid State Circuits*, vol. 31, no. 7, pp. 981-991, July 1996.

[23] M. Hanawa, K. Kaneko, T. Kawashimo, and H. Maruyama, "A 4.3ns 0.3μm CMOS 54x54b Multiplier Using Precharged Pass-Transistor Logic," 1996 *IEEE International Solid-State Circuits Conference*, pp. 364-365.

[24] R. Gu and M. Elmasry, "All N-Logic High-Speed True-Single-Phase Dynamic CMOS Logic," *IEEE J. Solid State Circuits*, vol. 31, no. 2, pp. 221-229, February 1996.

[25] A. Parameswar, H. Hara, and T. Sakurai, "A Swing Restored Pass-transistor Logic Based Multiply and Accumulate Circuit for Multimedia Applications," *IEEE J. Solid State Circuits*, vol. 31, no. 6, pp. 804-809, June 1996.

[26] K. Yano, T. Yamanaka, T. Nishida, M. Saito, K. Shimohigashi, and A. Shimizu, "A 3.8-ns CMOS 16x16-b Multiplier using Complementary Pass-Transistor Logic," *IEEE J. Solid State Circuits*, vol. 25, no. 2, pp. 388-395, April 1990.

[27] K. Yano, Y. Sasaki, K. Rikino, and K. Seki, "Top-down Pass-transistor Design," *IEEE J. Solid State Circuits*, vol. 31, no. 6, pp. 792-803, June 1996.

APPENDICES

Appendix A: 0.35μm CMOS MOSIS HSPICE Model

```
* DATE: Oct 30/98
* LOT: n88y                    WAF: 11
* Temperature_parameters=Default
.MODEL CMOSN NMOS (                                LEVEL   = 49
+VERSION = 3.1            TNOM    = 27             TOX     = 7.6E-9
+XJ      = 1.5E-7         NCH     = 1.7E17         VTH0    = 0.4964448
+K1      = 0.5307769      K2      = 0.0199705      K3      = 0.2963637
+K3B     = 0.2012165      W0      = 2.836319E-6    NLX     = 2.894802E-7
+DVT0W   = 0              DVT1W   = 5.3E6          DVT2W   = -0.032
+DVT0    = 0.112017       DVT1    = 0.2453972      DVT2    = -0.171915
+U0      = 444.9381976    UA      = 2.921284E-10   UB      = 1.773281E-18
+UC      = 7.067896E-11   VSAT    = 1.130785E5     A0      = 1.1356246
+AGS     = 0.2810374      B0      = 2.844393E-7    B1      = 5E-6
+KETA    = -7.8181E-3     A1      = 0              A2      = 1
+RDSW    = 925.2701982    PRWG    = -1E-3          PRWB    = -1E-3
+WR      = 1              WINT    = 7.186965E-8    LINT    = 1.735515E-9
+XL      = 0              XW      = 0              DWG     = -1.712973E-8
+DWB     = 5.851691E-9    VOFF    = -0.132935      NFACTOR = 0.5710974
+CIT     = 0              CDSC    = 8.607229E-4    CDSCD   = 0
+CDSCB   = 0              ETA0    = 2.128321E-3    ETAB    = 0
+DSUB    = 0.0257957      PCLM    = 0.6766314      PDIBLC1 = 1
+PDIBLC2 = 1.787424E-3    PDIBLCB = 0              DROUT   = 0.7873539
+PSCBE1  = 6.973485E9     PSCBE2  = 1.46235E-7     PVAG    = 0.05
+DELTA   = 0.01           MOBMOD  = 1              PRT     = 0
+UTE     = -1.5           KT1     = -0.11          KT1L    = 0
+KT2     = 0.022          UA1     = 4.31E-9        UB1     = -7.61E-18
+UC1     = -5.6E-11       AT      = 3.3E4          WL      = 0
+WLN     = 1              WW      = 0              WWN     = 1
+WWL     = 0              LL      = 0              LLN     = 1
```

```
+LW       = 0              LWN      = 1              LWL      = 0
+CAPMOD   = 2              CGDO     = 1.96E-10       CGSO     = 1.96E-10
+CGBO     = 0              CJ       = 9.276962E-4    PB       = 0.8157962
+MJ       = 0.3557696      CJSW     = 3.181055E-10   PBSW     = 0.6869149
+MJSW     = 0.1            PVTH0    = -0.0252481     PRDSW    = -96.4502805
+PK2      = -4.805372E-3   WKETA    = -7.643187E-4   LKETA    = -0.0129496
)
*
.MODEL CMOSP PMOS (                                  LEVEL    = 49
+VERSION = 3.1             TNOM     = 27             TOX      = 7.6E-9
+XJ       = 1.5E-7         NCH      = 1.7E17         VTH0     = -0.6636594
+K1       = 0.4564781      K2       = -0.019447      K3       = 39.382919
+K3B      = -2.8930965     W0       = 2.655585E-6    NLX      = 1.51028E-7
+DVT0W    = 0              DVT1W    = 5.3E6          DVT2W    = -0.032
+DVT0     = 1.1744581      DVT1     = 0.7631128      DVT2     = -0.1035171
+U0       = 151.3305606    UA       = 2.061211E-10   UB       = 1.823477E-18
+UC       = -8.97321E-12   VSAT     = 9.915604E4     A0       = 1.1210053
+AGS      = 0.3961954      B0       = 6.493139E-7    B1       = 4.273215E-6
+KETA     = -9.27E-3       A1       = 0              A2       = 1
+RDSW     = 2.30725E3      PRWG     = -1E-3          PRWB     = 0
+WR       = 1              WINT     = 5.962233E-8    LINT     = 4.30928E-9
+XL       = 0              XW       = 0              DWG      = -1.596201E-8
+DWB      = 1.378919E-8    VOFF     = -0.15          NFACTOR  = 2
+CIT      = 0              CDSC     = 6.593084E-4    CDSCD    = 0
+CDSCB    = 0              ETA0     = 0.0286461      ETAB     = 0
+DSUB     = 0.2436027      PCLM     = 4.3597508      PDIBLC1  = 7.447024E-4
+PDIBLC2  = 4.256073E-3    PDIBLCB  = 0              DROUT    = 0.0120292
+PSCBE1   = 1.347622E10    PSCBE2   = 5E-9           PVAG     = 3.669793
+DELTA    = 0.01           MOBMOD   = 1              PRT      = 0
+UTE      = -1.5           KT1      = -0.11          KT1L     = 0
+KT2      = 0.022          UA1      = 4.31E-9        UB1      = -7.61E-18
+UC1      = -5.6E-11       AT       = 3.3E4          WL       = 0
+WLN      = 1              WW       = 0              WWN      = 1
```

```
+WWL      = 0              LL       = 0              LLN      = 1
+LW       = 0              LWN      = 1              LWL      = 0
+CAPMOD   = 2              CGDO     = 2.307E-10      CGSO     = 2.307E-10
+CGBO     = 0              CJ       = 1.420282E-3    PB       = 0.99
+MJ       = 0.5490877      CJSW     = 4.773605E-10   PBSW     = 0.99
+MJSW     = 0.1997417      PVTH0    = 6.58707E-3     PRDSW    = -93.5582228
+PK2      = 1.011593E-3    WKETA    = -0.0101398     LKETA    = 6.027967E-3
)
```

Appendix B: 0.25μm CMOS MOSIS HSPICE Model

```
* DATE: Dec  6/99
* LOT: n99y                    WAF: 10
* Temperature_parameters=Default
.MODEL CMOSN NMOS (
+LEVEL   = 49              acm     = 3              hdif    = 0.35e-6
+VERSION = 3.1             TNOM    = 27             TOX     = 5.7E-9
+XJ      = 1E-7            NCH     = 2.3549E17      VTH0    = 0.4365497
+K1      = 0.3915623       K2      = 0.0175145      K3      = 1E-3
+K3B     = 2.6588343       W0      = 1E-7           NLX     = 1.111465E-7
+DVT0W   = 0               DVT1W   = 0              DVT2W   = 0
+DVT0    = -0.0408321      DVT1    = 0.0746768      DVT2    = 0.307109
+U0      = 407.1177485     UA      = 9.442714E-11   UB      = 1.092986E-18
+UC      = 1.63196E-11     VSAT    = 1.365087E5     A0      = 1.3189329
+AGS     = 0.2711719       B0      = 3.291713E-8    B1      = -1E-7
+KETA    = 4.645753E-3     A1      = 0              A2      = 1
+RDSW    = 439.9558234     PRWG    = 0.0345487      PRWB    = -0.0441065
+WR      = 1               WINT    = 4.645705E-9    LINT    = 1.116516E-9
+XL      = 3E-8            XW      = 0              DWG     = -1.494138E-9
+DWB     = 1.459097E-8     VOFF    = -0.1026054     NFACTOR = 0.1344887
+CIT     = 0               CDSC    = 1.527511E-3    CDSCD   = 0
+CDSCB   = 0               ETA0    = 1.930311E-3    ETAB    = 2.946158E-4
+DSUB    = 0.0214865       PCLM    = 1.3387947      PDIBLC1 = 0.480652
+PDIBLC2 = 9.034986E-3     PDIBLCB = -1E-3          DROUT   = 0.5593223
+PSCBE1  = 9.843289E9      PSCBE2  = 2.10878E-9     PVAG    = 1.0033136
+DELTA   = 0.01            MOBMOD  = 1              PRT     = 0
+UTE     = -1.5            KT1     = -0.11          KT1L    = 0
+KT2     = 0.022           UA1     = 4.31E-9        UB1     = -7.61E-18
+UC1     = -5.6E-11        AT      = 3.3E4          WL      = 0
+WLN     = 1               WW      = -1.22182E-16   WWN     = 1.2127
+WWL     = 0               LL      = 0              LLN     = 1
+LW      = 0               LWN     = 1              LWL     = 0
```

```
+CAPMOD  = 2              XPART   = 0.4            CGDO    = 3.11E-10
+CGSO    = 3.11E-10       CGBO    = 1E-11          CJ      = 1.758521E-3
+PB      = 0.99           MJ      = 0.457547       CJSW    = 4.085057E-10
+PBSW    = 0.8507757      MJSW    = 0.3374073      PVTH0   = 7.147521E-5
+PRDSW   = -67.2161633    PK2     = -1.344599E-3   WKETA   = 3.035972E-3
+LKETA   = -9.0406E-3     LAGS    = -0.3012
)
*
.MODEL CMOSP PMOS (
+LEVEL   = 49             acm     = 3              hdif    = 0.35e-6
+VERSION = 3.1            TNOM    = 27             TOX     = 5.7E-9
+XJ      = 1E-7           NCH     = 4.1589E17      VTH0    = -0.6586391
+K1      = 0.5199897      K2      = 0.0357513      K3      = 0
+K3B     = 15.5613889     W0      = 1E-6           NLX     = 1E-9
+DVT0W   = 0              DVT1W   = 0              DVT2W   = 0
+DVT0    = 2.6100181      DVT1    = 0.4363142      DVT2    = -0.042436
+U0      = 196.024903     UA      = 2.767112E-9    UB      = 1.90709E-18
+UC      = 6.166867E-11   VSAT    = 1.975064E5     A0      = 0.2398712
+AGS     = 0.0943234      B0      = 3.21184E-6     B1      = 5E-6
+KETA    = 0.0312217      A1      = 0              A2      = 1
+RDSW    = 997.072701     PRWG    = -0.1916111     PRWB    = -0.495
+WR      = 1              WINT    = 2.527293E-9    LINT    = 1.254514E-8
+XL      = 3E-8           XW      = 0              DWG     = -3.253948E-8
+DWB     = 4.92072E-8     VOFF    = -0.15          NFACTOR = 1.5460516
+CIT     = 0              CDSC    = 1.413317E-4    CDSCD   = 0
+CDSCB   = 0              ETA0    = 0.7241245      ETAB    = -0.240523
+DSUB    = 1.0813613      PCLM    = 2.0772083      PDIBLC1 = 4.31459E-4
+PDIBLC2 = 0.0252121      PDIBLCB = -9.960722E-4   DROUT   = 0.0432774
+PSCBE1  = 3.191047E10    PSCBE2  = 1.323218E-8    PVAG    = 0.0420525
+DELTA   = 0.01           MOBMOD  = 1              PRT     = 0
+UTE     = -1.5           KT1     = -0.11          KT1L    = 0
+KT2     = 0.022          UA1     = 4.31E-9        UB1     = -7.61E-18
+UC1     = -5.6E-11       AT      = 3.3E4          WL      = 0
```

```
+WLN      = 1              WW       = 0              WWN      = 1
+WWL      = 0              LL       = 0              LLN      = 1
+LW       = 0              LWN      = 1              LWL      = 0
+CAPMOD   = 2              XPART    = 0.4            CGDO     = 2.68E-10
+CGSO     = 2.68E-10       CGBO     = 1E-11          CJ       = 1.902493E-3
+PB       = 0.9810285      MJ       = 0.4644362      CJSW     = 3.142741E-10
+PBSW     = 0.9048624      MJSW     = 0.3304452      PVTH0    = 4.952976E-3
+PRDSW    = 29.8169373     PK2      = 3.383373E-3    WKETA    = -7.913501E-3
+LKETA    = -0.0208318
)
```

Appendix C: Conventional Domino 8-bit Ripple Carry Adder HSPICE Netlist

```
* Conventional Domino CMOS Logic * Loaded Full Adder

.options node list post=2

.param mL = 0.25u

X0 Vdd in 0 clk1 C1 C1_ FAC
X1 Vdd 0 C1 clk2 C2 C2_ FAC
X2 Vdd 0 C2 clk3 C3 C3_ FAC
X3 Vdd 0 C3 clk4 C4 C4_ FAC
X4 Vdd 0 C4 clk1 C5 C5_ FAC
X5 Vdd 0 C5 clk2 C6 C6_ FAC
X6 Vdd 0 C6 clk3 C7 C7_ FAC
X7 Vdd 0 C7 clk4 C8 C8_ FAC

X8 Vdd Vdd 0 C1_ clk2 S1 FAS
X9 Vdd Vdd 0 C2_ clk3 S2 FAS
X10 Vdd Vdd 0 C3_ clk4 S3 FAS
X11 Vdd Vdd 0 C4_ clk1 S4 FAS
X12 Vdd Vdd 0 C5_ clk2 S5 FAS
X13 Vdd Vdd 0 C6_ clk3 S6 FAS
X14 Vdd Vdd 0 C7_ clk4 S7 FAS
X15 Vdd Vdd 0 C8_ clk1 S8 FAS

Mns1 0 S1 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns2 0 S2 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns3 0 S3 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns4 0 S4 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns5 0 S5 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns6 0 S6 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns7 0 S7 0 Vss CMOSN L='mL' W='5u' GEO=3
```

```
Mns8 0 S8 0 Vss CMOSN L='mL' W='5u' GEO=3

** Double Pumped Domino Full Adder

.SUBCKT FAC Ain Bin Cin clk Cout L

* Carry Logic
Mn10 L Ain J Vss CMOSN L='mL' W='4u' GEO=3
Mn11 L Bin K Vss CMOSN L='mL' W='4u' GEO=3
Mn12 J Cin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn13 J Bin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn14 K Cin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn15 M clk Vss Vss CMOSN L='mL' W='5.76u' GEO=1
XC L Cout clk OUTC

.ic V(L)=0

.ENDS FAC


.SUBCKT FAS Ain Bin Cin Cout_bar clk Sum_out

* Sum Logic
Mn16 W Cout_bar T Vss CMOSN L='mL' W='4u' GEO=3
Mn17 T Ain X Vss CMOSN L='mL' W='4.8u' GEO=3
Mn18 T Bin X Vss CMOSN L='mL' W='4.8u' GEO=3
Mn19 T Cin X Vss CMOSN L='mL' W='4.8u' GEO=3
Mn20 W Ain V Vss CMOSN L='mL' W='4u' GEO=3
Mn21 V Bin U Vss CMOSN L='mL' W='4.8u' GEO=3
Mn22 U Cin X Vss CMOSN L='mL' W='5.76u' GEO=3
Mn24 X clk Vss Vss CMOSN L='mL' W='6.912u' GEO=1
XS W Sum_out clk OUTS
```

```
.ic V(W)=0

.ENDS FAS

* Double Pumped Output Logic

.SUBCKT OUTC M O clk
Mp9 M clk Vdd Vdd CMOSP L='mL' W='7u' GEO=1

Mp10 N M Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn25 N M Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp11 M N Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1

Mp12 O M Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mn23 O M Vss Vss CMOSN L='mL' W='2.5u' GEO=1
.ENDS OUTC

.SUBCKT OUTS M O clk
Mp13 M clk Vdd Vdd CMOSP L='mL' W='7u' GEO=1

Mp14 N M Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn26 N M Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp15 M N Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1

Mp16 O M Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mn27 O M Vss Vss CMOSN L='mL' W='2.5u' GEO=1
.ENDS OUTS

.SUBCKT CLKINV A B
Mp1 B A Vee Vee CMOSP L='mL' W='40u' GEO=1
Mn1 B A Vss Vss CMOSN L='mL' W='20u' GEO=1
.ENDS CLKINV
```

```
.SUBCKT CLKINVd A B
Mp1 B A Vff Vff CMOSP L='mL' W='40u' GEO=1
Mn1 B A Vss Vss CMOSN L='mL' W='20u' GEO=1
.ENDS CLKINVd

* Clock Buffers
Xclk1 clk1_ clk1 clkinv
Xclk2 clk2_ clk2 clkinv
Xclk3 clk3_ clk3 clkinv
Xclk4 clk4_ clk4 clkinv
Xclk1d clk1_ clk1d clkinvd
Xclk2d clk2_ clk2d clkinvd
Xclk3d clk3_ clk3d clkinvd
Xclk4d clk4_ clk4d clkinvd
```

Appendix D: DET Domino 8-bit Ripple Carry Adder HSPICE Netlist

```
* Double Data Rate Domino CMOS Logic w/ NAND
* Loaded Full Adder

.options node list post=2

.param mL = 0.25u

X0 Vdd Vss Vss clk1 clk3 clk4 clk2 C1a C1b FAC

X1a C1a Vdd Vss clk2 C2a_ FACD
X1b C1b Vdd Vss clk4 C2b_ FACD
X1 C2a_ C2b_ C2 NAND

X2 C2 Vdd Vss clk1 clk3 clk4 clk2 C3a C3b FAC

X3a C3a Vdd Vss clk2 C4a_ FACD
X3b C3b Vdd Vss clk4 C4b_ FACD
X3 C4a_ C4b_ C4 NAND

X4 C4 Vdd Vss clk1 clk3 clk4 clk2 C5a C5b FAC

X5a C5a Vdd Vss clk2 C6a_ FACD
X5b C5b Vdd Vss clk4 C6b_ FACD
X5 C6a_ C6b_ C6 NAND

X6 C6 Vdd Vss clk1 clk3 clk4 clk2 C7a C7b FAC

X7a C7a Vdd Vss clk2 C8a_ FACD
X7b C7b Vdd Vss clk4 C8b_ FACD
X7 C8a_ C8b_ C8 NAND
```

```
X8a Vdd Vss Vss C1a clk2 S1a_ FASD
X8b Vdd Vss Vss C1b clk4 S1b_ FASD
X8 S1a_ S1b_ S1 NAND

X9 Vdd Vss Vss C2 clk1 clk3 clk4 clk2 S2a S2b FAS

X10a Vdd Vss Vss C3a clk2 S3a_ FASD
X10b Vdd Vss Vss C3b clk4 S3b_ FASD
X10 S3a_ S3b_ S3 NAND

X11 Vdd Vss Vss C4 clk1 clk3 clk4 clk2 S4a S4b FAS

X12a Vdd Vss Vss C5a clk2 S5a_ FASD
X12b Vdd Vss Vss C5b clk4 S5b_ FASD
X12 S5a_ S5b_ S5 NAND

X13 Vdd Vss Vss C6 clk1 clk3 clk4 clk2 S6a S6b FAS

X14a Vdd Vss Vss C7a clk2 S7a_ FASD
X14b Vdd Vss Vss C7b clk4 S7b_ FASD
X14 S7a_ S7b_ S7 NAND

X15 Vdd Vss Vss C8 clk1 clk3 clk4 clk2 S8a S8b FAS

Mns2a 0 S2a 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns4a 0 S4a 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns6a 0 S6a 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns8a 0 S8a 0 Vss CMOSN L='mL' W='5u' GEO=3

Mns2b 0 S2b 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns4b 0 S4b 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns6b 0 S6b 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns8b 0 S8b 0 Vss CMOSN L='mL' W='5u' GEO=3
```

```
Mns1 0 S1 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns3 0 S3 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns5 0 S5 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns7 0 S7 0 Vss CMOSN L='mL' W='5u' GEO=3

** Double Pumped Domino Full Adder

.SUBCKT FAC Ain Bin Cin c1k clk_bar c1k1 c1k1_ Couta Coutb

* Carry Logic
Mn12 L Cin J Vss CMOSN L='mL' W='6u' GEO=3
Mn13 L Bin K Vss CMOSN L='mL' W='6u' GEO=3
Mn14 J Ain Vss Vss CMOSN L='mL' W='7.2u' GEO=1
Mn15 J Bin Vss Vss CMOSN L='mL' W='7.2u' GEO=1
Mn16 K Ain Vss Vss CMOSN L='mL' W='7.2u' GEO=1
XC L Couta Coutb c1k clk_bar c1k1 c1k1_  DPOUTC

.ic V(L)=0 V(Cout_bar)=0

.ENDS FAC

.SUBCKT FAS Ain Bin Cin Cout c1k clk_bar c1k1 c1k1_ Suma Sumb

* Sum Logic
Mn17 W Ain V Vss CMOSN L='mL' W='6u' GEO=3
Mn18 V Bin U Vss CMOSN L='mL' W='7.2u' GEO=3
Mn19 U Cin Vss Vss CMOSN L='mL' W='8.64u' GEO=1
Mn20 T Ain Vss Vss CMOSN L='mL' W='7.2u' GEO=1
Mn21 T Bin Vss Vss CMOSN L='mL' W='7.2' GEO=1
Mn22 T Cin Vss Vss CMOSN L='mL' W='7.2u' GEO=1
Mn23 W Cout_bar T Vss CMOSN L='mL' W='6u' GEO=3
XS W Suma Sumb  c1k clk_bar c1k1 c1k1_ DPOUTS
```

```
Mp24 Cout_bar Cout Vdd Vdd CMOSP L='mL' W='2.5u' GEO=1
Mn25 Cout_bar Cout Vss Vss CMOSN L='mL' W='1u' GEO=1

.ENDS FAS

.SUBCKT FACD Ain Bin Cin clk L

* Carry Logic
Mn10 L Ain J Vss CMOSN L='mL' W='4u' GEO=3
Mn11 L Bin K Vss CMOSN L='mL' W='4u' GEO=3
Mn12 J Cin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn13 J Bin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn14 K Cin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn15 M clk Vss Vss CMOSN L='mL' W='5.76u' GEO=1

Mp9 L clk Vdd Vdd CMOSP L='mL' W='7u' GEO=1
Mp10 N L Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn25 N L Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp11 L N Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1

.ic V(L)=0 V(Cout_bar)=0

.ENDS FACD


.SUBCKT FASD Ain Bin Cin Cout clk W

* Sum Logic
Mn16 W Cout_bar T Vss CMOSN L='mL' W='4u' GEO=3
Mn17 T Ain X Vss CMOSN L='mL' W='4.8u' GEO=3
Mn18 T Bin X Vss CMOSN L='mL' W='4.8u' GEO=3
Mn19 T Cin X Vss CMOSN L='mL' W='4.8u' GEO=3
```

```
Mn20 W Ain V Vss CMOSN L='mL' W='4u' GEO=3
Mn21 V Bin U Vss CMOSN L='mL' W='4.8u' GEO=3
Mn22 U Cin X Vss CMOSN L='mL' W='5.76u' GEO=3
Mn24 X clk Vss Vss CMOSN L='mL' W='6.912u' GEO=1

Mp13 W clk Vdd Vdd CMOSP L='mL' W='7u' GEO=1
Mp14 N W Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn26 N W Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp15 W N Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1

Mp30 Cout_bar Cout Vdd Vdd CMOSP L='mL' W='2.5u' GEO=1
Mn31 Cout_bar Cout Vss Vss CMOSN L='mL' W='1u' GEO=1


.ic V(W)=0

.ENDS FASD

* Double Pumped Output Logic

.SUBCKT DPOUTC X P D clk clk_bar clk1 clk1_
Mn24 X clk_bar M Vss CMOSN L='mL' W='5u' GEO=2
Mp12 M clk_bar Vdd Vdd CMOSP L='mL' W='7u' GEO=1

Mp25 N M Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn25 N M Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp26 M N Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1

Mn26 X clk Q Vss CMOSN L='mL' W='5u' GEO=2
Mp27 Q clk Vdd Vdd CMOSP L='mL' W='7u' GEO=1

Mp28 R Q Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn27 R Q Vss Vss CMOSN L='mL' W='0.25u' GEO=1
```

```
Mp29 Q R Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1

Mp30 G Q Vdd Vdd CMOSP L='mL' W='12u' GEO=1
Mpc1 P clkl_ G Vdd CMOSP L='mL' W='10u' GEO=2
Mnc1 P clkl Z Vss CMOSN L='mL' W='5u' GEO=2
Mn51 Z Q Vss Vss CMOSN L='mL' W='6u' GEO=1

Mp31 H M Vdd Vdd CMOSP L='mL' W='12u' GEO=1
Mpc2 D clkl H Vdd CMOSP L='mL' W='10u' GEO=2
Mnc2 D clkl_ Y Vss CMOSN L='mL' W='5u' GEO=2
Mn50 Y M Vss Vss CMOSN L='mL' W='6u' GEO=1

*Mp32 N clk_bar Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1
*Mp33 N M Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1
*Mn30 N clk_bar S Vss CMOSN L='mL' W='1.0u' GEO=2
*Mn31 S M Vss Vss CMOSN L='mL' W='1.0u' GEO=1
*Mp34 M N Vdd Vdd CMOSP L='mL' W='3.0u' GEO=1

*Mp35 R clk Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1
*Mp36 R Q Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1
*Mn32 R clk T Vss CMOSN L='mL' W='1.0u' GEO=2
*Mn33 T Q Vss Vss CMOSN L='mL' W='1.0u' GEO=1
*Mp37 Q R Vdd Vdd CMOSP L='mL' W='3.0u' GEO=1

.ic V(X)=0 V(D)=0 V(P)=0

.ENDS DPOUTC

* Double Pumped Output Logic

.SUBCKT DPOUTS X P D clk clk_bar clkl clkl_
Mn24 X clk_bar M Vss CMOSN L='mL' W='5u' GEO=2
Mp12 M clk_bar Vdd Vdd CMOSP L='mL' W='7u' GEO=1
```

```
Mp25 N M Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn25 N M Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp26 M N Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1

Mn26 X clk Q Vss CMOSN L='mL' W='5u' GEO=2
Mp27 Q clk Vdd Vdd CMOSP L='mL' W='7u' GEO=1

Mp28 R Q Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn27 R Q Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp29 Q R Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1

Mp40 E Q Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mp41 P clkl_ E Vdd CMOSP L='mL' W='10u' GEO=2
Mn43 P clkl H Vss CMOSN L='mL' W='7u' GEO=2
Mn54 H Q Vss Vss CMOSN L='mL' W='7u' GEO=1

Mp44 F M Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mp45 D clkl F Vdd CMOSP L='mL' W='10u' GEO=2
Mn47 D clkl_ I Vss CMOSN L='mL' W='7u' GEO=2
Mn55 I M Vss Vss CMOSN L='mL' W='7u' GEO=1

*Mp32 N clk_bar Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1
*Mp33 N M Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1
*Mn30 N clk_bar S Vss CMOSN L='mL' W='1.0u' GEO=2
*Mn31 S M Vss Vss CMOSN L='mL' W='1.0u' GEO=1
*Mp34 M N Vdd Vdd CMOSP L='mL' W='3.0u' GEO=1

*Mp35 R clk Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1
*Mp36 R Q Vdd Vdd CMOSP L='mL' W='1.0u' GEO=1
*Mn32 R clk T Vss CMOSN L='mL' W='1.0u' GEO=2
*Mn33 T Q Vss Vss CMOSN L='mL' W='1.0u' GEO=1
*Mp37 Q R Vdd Vdd CMOSP L='mL' W='3.0u' GEO=1
```

```
.ic V(X)=0 V(D)=0 V(P)=0

.ENDS DPOUTS


.SUBCKT NAND A B D
Mp1 D A Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mp2 D B Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mn1 D A C Vss CMOSN L='mL' W='5u' GEO=1
Mn2 C B Vss Vss CMOSN L='mL' W='5u' GEO=1
.ENDS NAND

.SUBCKT CLKINV A B
Mp1 B A Vee Vee CMOSP L='mL' W='40u' GEO=1
Mn1 B A Vss Vss CMOSN L='mL' W='20u' GEO=1
.ENDS CLKINV

.SUBCKT CLKINVd A B
Mp1 B A Vff Vff CMOSP L='mL' W='40u' GEO=1
Mn1 B A Vss Vss CMOSN L='mL' W='20u' GEO=1
.ENDS CLKINVd

* Clock Buffers
Xclk1 clk1_ clk1 clkinv
Xclk2 clk2_ clk2 clkinv
Xclk3 clk3_ clk3 clkinv
Xclk4 clk4_ clk4 clkinv
Xclk1d clk1_ clk1d clkinvd
Xclk2d clk2_ clk2d clkinvd
Xclk3d clk3_ clk3d clkinvd
Xclk4d clk4_ clk4d clkinvd
```

Appendix E: DDR Domino 8-bit Ripple Carry HSPICE Netlist

```
* DDR Conventional Domino CMOS Logic * Loaded Full Adder

.options node list post=2

.param mL = 0.25u

X0a Vdd in 0 clk1 C1a C1a_ FACout
X1a Vdd 0 C1a clk2 C2a C2a_ FACout
X2a Vdd 0 C2a clk3 C3a C3a_ FACout
X3a Vdd 0 C3a clk4 C4a C4a_ FACout
X4a Vdd 0 C4a clk1 C5a C5a_ FACout
X5a Vdd 0 C5a clk2 C6a C6a_ FACout
X6a Vdd 0 C6a clk3 C7a C7a_ FACout
X7a Vdd 0 C7a clk4 C8a C8a_ FACout

X0b Vdd in 0 clk3 C1b C1b_ FACout
X1b Vdd 0 C1b clk4 C2b C2b_ FACout
X2b Vdd 0 C2b clk1 C3b C3b_ FACout
X3b Vdd 0 C3b clk2 C4b C4b_ FACout
X4b Vdd 0 C4b clk3 C5b C5b_ FACout
X5b Vdd 0 C5b clk4 C6b C6b_ FACout
X6b Vdd 0 C6b clk1 C7b C7b_ FACout
X7b Vdd 0 C7b clk2 C8b C8b_ FACout

X8a Vdd 0 0 C1a_ clk2 S1a FASout
X9a Vdd 0 0 C2a_ clk3 S2a FASout
X10a Vdd 0 0 C3a_ clk4 S3a FASout
X11a Vdd 0 0 C4a_ clk1 S4a FASout
X12a Vdd 0 0 C5a_ clk2 S5a FASout
X13a Vdd 0 0 C6a_ clk3 S6a FASout
X14a Vdd 0 0 C7a_ clk4 S7a FASout
```

```
X15a Vdd 0 0 C8a_ clk1 S8a FASout

X8b Vdd 0 0 C1b_ clk4 S1b FASout
X9b Vdd 0 0 C2b_ clk1 S2b FASout
X10b Vdd 0 0 C3b_ clk2 S3b FASout
X11b Vdd 0 0 C4b_ clk3 S4b FASout
X12b Vdd 0 0 C5b_ clk4 S5b FASout
X13b Vdd 0 0 C6b_ clk1 S6b FASout
X14b Vdd 0 0 C7b_ clk2 S7b FASout
X15b Vdd 0 0 C8b_ clk3 S8b FASout

XNAND1 S1a S1b S1 NAND
XNAND2 S2a S2b S2 NAND
XNAND3 S3a S3b S3 NAND
XNAND4 S4a S4b S4 NAND
XNAND5 S5a S5b S5 NAND
XNAND6 S6a S6b S6 NAND
XNAND7 S7a S7b S7 NAND
XNAND8 S8a S8b S8 NAND

Mns1 0 S1 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns2 0 S2 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns3 0 S3 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns4 0 S4 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns5 0 S5 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns6 0 S6 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns7 0 S7 0 Vss CMOSN L='mL' W='5u' GEO=3
Mns8 0 S8 0 Vss CMOSN L='mL' W='5u' GEO=3

* Carry Logic

.SUBCKT FACout Ain Bin Cin clk 0 L
```

```
Mn10 L Ain J Vss CMOSN L='mL' W='4u' GEO=3
Mn11 L Bin K Vss CMOSN L='mL' W='4u' GEO=3
Mn12 J Cin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn13 J Bin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn14 K Cin M Vss CMOSN L='mL' W='4.8u' GEO=3
Mn15 M clk Vss Vss CMOSN L='mL' W='5.76u' GEO=1
Mp13a L clk Vdd Vdd CMOSP L='mL' W='7u' GEO=1
Mp14a N L Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn26a N L Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp15a L N Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mp27 O L Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mn27 O L Vss Vss CMOSN L='mL' W='5u' GEO=1

.ic V(L)=0 V(P)=volt V(O)=volt

.ENDS FACout

.SUBCKT FASout Ain Bin Cin Cout_bar clk W

* Sum Logic
Mn16 W Cout_bar T Vss CMOSN L='mL' W='4u' GEO=3
Mn17 T Ain X Vss CMOSN L='mL' W='4.8u' GEO=3
Mn18 T Bin X Vss CMOSN L='mL' W='4.8u' GEO=3
Mn19 T Cin X Vss CMOSN L='mL' W='4.8u' GEO=3
Mn20 W Ain V Vss CMOSN L='mL' W='4u' GEO=3
Mn21 V Bin U Vss CMOSN L='mL' W='4.8u' GEO=3
Mn22 U Cin X Vss CMOSN L='mL' W='5.76u' GEO=3
Mn24 X clk Vss Vss CMOSN L='mL' W='6.912u' GEO=1
Mp13a W clk Vdd Vdd CMOSP L='mL' W='7u' GEO=1
Mp14a N W Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
Mn26a N W Vss Vss CMOSN L='mL' W='0.25u' GEO=1
Mp15a W N Vdd Vdd CMOSP L='mL' W='0.25u' GEO=1
```

```
.ic V(W)=0 V(P)=volt V(0)=volt

.ENDS FASout

.SUBCKT NAND A B C
Mp1 C A Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mp2 C B Vdd Vdd CMOSP L='mL' W='10u' GEO=1
Mn3 C A D Vss CMOSN L='mL' W='5u' GEO=2
Mn4 D B Vss Vss CMOSN L='mL' W='5u' GEO=1
.ENDS NAND

.SUBCKT CLKINV A B
Mp1 B A Vee Vee CMOSP L='mL' W='80u' GEO=1
Mn1 B A Vss Vss CMOSN L='mL' W='40u' GEO=1
.ENDS CLKINV

.SUBCKT CLKINVd A B
Mp1 B A Vff Vff CMOSP L='mL' W='80u' GEO=1
Mn1 B A Vss Vss CMOSN L='mL' W='40u' GEO=1
.ENDS CLKINVd

* Clock Buffers
Xclk1 clk1_ clk1 clkinv
Xclk2 clk2_ clk2 clkinv
Xclk3 clk3_ clk3 clkinv
Xclk4 clk4_ clk4 clkinv

* Clock Buffers
Xclk1d clk1_ clk1d clkinvd
Xclk2d clk2_ clk2d clkinvd
Xclk3d clk3_ clk3d clkinvd
Xclk4d clk4_ clk4d clkinvd
```

Appendix F: Example HSPICE Code for Conventional Domino Mesasurements

```
sources

.param volt=2.5
Vcc Vdd 0 DC volt
.global vdd
Vee Vee 0 DC volt
.global vee
Vff Vff 0 DC volt
.global vff

.param gnd=0
Vgnd Vss 0 DC gnd
.global vss

.param nw=0.5u

*Clock Parameters

.param frequency=1000X
.param duty=.5
.param rise=.05ns
.param fall=.05ns
.param level='((1/frequency)*duty)-rise'
.param clk_period='(1/frequency)'

Vclk1 clk1_ 0 pulse gnd volt delay1 rise fall level clk_period
Vclk2 clk2_ 0 pulse gnd volt delay2 rise fall level clk_period
Vclk3 clk3_ 0 pulse gnd volt delay3 rise fall level clk_period
Vclk4 clk4_ 0 pulse gnd volt delay4 rise fall level clk_period

Vin in 0 pulse gnd volt delay1 rise fall '2*level' '2*clk_period'
```

```
.param length='level'
.param delay1='(clk_period/4)'
.param delay2='(clk_period/4)*2'
.param delay3='(clk_period/4)*3'
.param delay4='(clk_period/4)*4'
.param data_period='clk_period'

.param tran_end='(5*clk_period)'
.meas tran avg_curr_logic avg I(Vcc) from 'clk_period' to '5.0*clk_period'
.meas avg_power_logic param='(-avg_curr_logic*2.5)/4'
.meas tran avg_curr_clk avg I(Vee) from 'clk_period' to '5.0*clk_period'
.meas avg_power_clk param='(-avg_curr_clk*2.5)/4'
.meas tran avg_curr_clkd avg I(Vff) from 'clk_period' to '5.0*clk_period'
.meas avg_power_clkd param='(-avg_curr_clkd*2.5)/4'
.TRAN 0.1ns tran_end
```

Appendix G: Example HSPICE Code for DDR/DET Domino Measurements

```
* sources

.param volt=2.5
Vcc Vdd 0 DC volt
.global vdd
Vee Vee 0 DC volt
.global vee
Vff Vff 0 DC volt
.global vff

.param gnd=0
Vgnd Vss 0 DC gnd
.global vss

*Clock Parameters

.param frequency=800X
.param duty=.5
.param rise=.05ns
.param fall=.05ns
.param level='((1/frequency)*duty)-rise'
.param clk_period='(1/frequency)'
.param level2='level*2'
.param clk_period2='clk_period*2'

Vclk1 clk1_ 0 pulse gnd volt delay1 rise fall level clk_period
Vclk2 clk2_ 0 pulse gnd volt delay2 rise fall level clk_period
Vclk3 clk3_ 0 pulse gnd volt delay3 rise fall level clk_period
Vclk4 clk4_ 0 pulse gnd volt delay4 rise fall level clk_period
```

```
Vin in 0 pulse gnd volt delay4 rise fall '1*length' '1*data_period'

VinA0 A0 0 pulse gnd volt delay4 rise fall'4*length+4*rise' '4*data_period'
VinB0 B0 0 pulse gnd volt delay4 rise fall'2*length+2*rise' '2*data_period'
VinC0 C0 0 pulse gnd volt delay4 rise fall'1*length' '1*data_period'

VinA0d A0d 0 pulse gnd volt delay1 rise fall'4*length+4*rise' '4*data_period'
VinB0d B0d 0 pulse gnd volt delay1 rise fall'2*length+2*rise' '2*data_period'
VinC0d C0d 0 pulse gnd volt delay1 rise fall'1*length' '1*data_period'


.param length='level'
.param delay1='(clk_period/4)'
.param delay2='(clk_period/4)*2'
.param delay3='(clk_period/4)*3'
.param delay4='(clk_period/4)*4'
.param delayd='(clk_period/4)*4'
.param data_period='clk_period'

.param tran_end='(5*clk_period)'
*.param tran_end='(14*clk_period)'

.meas tran avg_curr_logic avg I(Vcc) from '1*clk_period' to '5*clk_period'
.meas avg_power_logic param='(-avg_curr_logic*2.5)/4'
.meas tran avg_curr_clk avg I(Vee) from '1*clk_period' to '5*clk_period'
.meas avg_power_clk param='(-avg_curr_clk*2.5)/4'
.meas tran avg_curr_clkd avg I(Vff) from '1*clk_period' to '5*clk_period'
.meas avg_power_clkd param='(-avg_curr_clkd*2.5)/4'

.TRAN 0.1ns tran_end
```

Appendix H: Example HSPICE Netlist Conventional Flip-Flop

```
* Standard DQ Flip Flop

.options node list post=2 relv=1e-4 absvar=.3 relvar=0.1
.param mL = 0.35u

* Flip Flop Net List
Mp1 D clk A Vdd CMOSP L='mL' W='wp1' AD='wp1*(mL/2)*6' AS='wp1*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn1 D clk_bar A Vss CMOSN L='mL' W='wn1' AD='wn1*(mL/2)*6' AS='wn1*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn2 B A Vss Vss CMOSN L='mL' W='wn2' AD='wn2*(mL/2)*6' AS='wn2*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp2 B A Vdd Vdd CMOSP L='mL' W='wp2' AD='wp2*(mL/2)*6' AS='wp2*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn3 B clk C Vss CMOSN L='mL' W='wn3' AD='wn3*(mL/2)*6' AS='wn3*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp3 B clk_bar C Vdd CMOSP L='mL' W='wp3' AD='wp3*(mL/2)*6' AS='wp3*(mL/2)*6' PD='6*mL' PS='6*mL'

Mn4 Q C Vss Vss CMOSN L='mL' W='wn4' AD='wn4*(mL/2)*5' AS='wn4*(mL/2)*5' PD='5*mL+wn4' PS='5*mL+wn4'
Mp4 Q C Vdd Vdd CMOSP L='mL' W='wp4' AD='wp4*(mL/2)*5' AS='wp4*(mL/2)*5' PD='5*mL+wp4' PS='5*mL+wp4'

Mp5 E Q Vdd Vdd CMOSP L='mL' W='wp5' AD='wp5*(mL/2)*6' AS='wp5*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn5 E Q Vss Vss CMOSN L='mL' W='wn5' AD='wn5*(mL/2)*6' AS='wn5*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn6 E clk_bar C Vss CMOSN L='mL' W='wn6' AD='wn6*(mL/2)*6' AS='wn6*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp6 E clk C Vdd CMOSP L='mL' W='wp6' AD='wp6*(mL/2)*6' AS='wp6*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn7 F clk A Vss CMOSN L='mL' W='wn7' AD='wn7*(mL/2)*6' AS='wn7*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp7 F clk_bar A Vdd CMOSP L='mL' W='wp7' AD='wp7*(mL/2)*6' AS='wp7*(mL/2)*5' PD='6*mL' PS='6*mL'
Mp8 F B Vdd Vdd CMOSP L='mL' W='wp8' AD='wp8*(mL/2)*6' AS='wp8*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn8 F B Vss Vss CMOSN L='mL' W='wn8' AD='wn8*(mL/2)*6' AS='wn8*(mL/2)*6' PD='6*mL' PS='6*mL'


C1 Q 0 200fF
C2 D 0 200fF
C3 test_D 0 200fF
```

Appendix I: Example HSPICE Netlist PowerPC Flip-Flop

```
* Power PC 603 Flip Flop

.options node list post=2 relv=1e-4 absvar=.3 relvar=0.1
.param mL = 0.35u

* Flip Flop Net List
Mp1 D clk A Vdd CMOSP L='mL' W='wp1' AD='wp1*(mL/2)*6' AS='wp1*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn1 D clk_bar A Vss CMOSN L='mL' W='wn1' AD='wn1*(mL/2)*6' AS='wn1*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn2 B A Vss Vss CMOSN L='mL' W='wn2' AD='wn2*(mL/2)*6' AS='wn2*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp2 B A Vdd Vdd CMOSP L='mL' W='wp2' AD='wp2*(mL/2)*6' AS='wp2*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn3 B clk C Vss CMOSN L='mL' W='wn3' AD='wn3*(mL/2)*6' AS='wn3*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp3 B clk_bar C Vdd CMOSP L='mL' W='wp3' AD='wp3*(mL/2)*6' AS='wp3*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn4 I C Vss Vss CMOSN L='mL' W='wn4' AD='wn4*(mL/2)*5' AS='wn4*(mL/2)*5' PD='5*mL+wn4' PS='5*mL+wn4'
Mp4 I C Vdd Vdd CMOSP L='mL' W='wp4' AD='wp4*(mL/2)*5' AS='wp4*(mL/2)*5' PD='5*mL+wp4' PS='5*mL+wp4'
Mp5 H I Vdd Vdd CMOSP L='mL' W='wp5' AD='wp5*mL' AS='wp5*mL' PD='2*mL' PS='2*mL'
Mn5 G I Vss Vss CMOSN L='mL' W='wn5' AD='wn5*mL' AS='wn5*mL' PD='2*mL' PS='2*mL'
Mn6 C clk_bar G Vss CMOSN L='mL' W='wn6' AD='wn6*(mL/2)*6' AS='wn6*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp6 C clk H Vdd CMOSP L='mL' W='wp6' AD='wp6*(mL/2)*6' AS='wp6*(mL/2)*6' PD='6*mL' PS='6*mL'
Mn7 A clk E Vss CMOSN L='mL' W='wn7' AD='wn7*(mL/2)*6' AS='wn7*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp7 A clk_bar F Vdd CMOSP L='mL' W='wp7' AD='wp7*(mL/2)*6' AS='wp7*(mL/2)*6' PD='6*mL' PS='6*mL'
Mp8 F B Vdd Vdd CMOSP L='mL' W='wp8' AD='wp8*mL' AS='wp8*mL' PD='2*mL' PS='2*mL'
Mn8 E B Vss Vss CMOSN L='mL' W='wn8' AD='wn8*mL' AS='wn8*mL' PD='2*mL' PS='2*mL'

Mp9 Q C Vdd Vdd CMOSP L='mL' W='wp9' AD='wp9*(mL/2)*5' AS='wp9*(mL/2)*5' PD='5*mL+wp9' PS='5*mL+wp9'
Mn9 Q C Vss Vss CMOSN L='mL' W='wn9' AD='wp9*(mL/2)*5' AS='wn9*(mL/2)*5' PD='5*mL+wp9' PS='5*mL+wn9'

C1 Q 0 200fF
C2 D 0 200fF
C3 test_D 0 200fF
```

Appendix J: Example HSPICE Netlist C2MOS Flop-Flop

```
* Modified C2MOS Flip Flop

.options node list post=2 relv=1e-4 absvar=.3 relvar=0.1
.param mL = 0.35u

* Flip Flop Net List

Mp1 A clk C Vdd CMOSP L='mL' W='wp1' AD='wp1*mL' AS='wp1*mL' PD='2*mL' PS='2*mL'
Mn1 A clk_bar B Vss CMOSN L='mL' W='wn1' AD='wn1*mL' AS='wn1*mL' PD='2*mL' PS='2*mL'
Mp2 C D Vdd Vdd CMOSP L='mL' W='wp2' AD='wp2*mL' AS='wp2*mL' PD='2*mL' PS='2*mL'
Mn2 B D Vss Vss CMOSN L='mL' W='wn2' AD='wn2*mL' AS='wn2*mL' PD='2*mL' PS='2*mL'
Mp3 A clk_bar G Vdd CMOSP L='mL' W='wp3' AD='wp3*mL' AS='wp3*mL' PD='2*mL' PS='2*mL'
Mn3 A clk F Vss CMOSN L='mL' W='wn3' AD='wn3*mL' AS='wn3*mL' PD='2*mL' PS='2*mL'
Mp4 G E Vdd Vdd CMOSP L='mL' W='wp4' AD='wp4*mL' AS='wp4*mL' PD='2*mL' PS='2*mL'
Mn4 F E Vss Vss CMOSN L='mL' W='wn4' AD='wn4*mL' AS='wn4*mL' PD='2*mL' PS='2*mL'
Mp5 E A Vdd Vdd CMOSP L='mL' W='wp5' AD='wp5*mL' AS='wp5*mL' PD='2*mL' PS='2*mL'
Mn5 E A Vss Vss CMOSN L='mL' W='wn5' AD='wn5*mL' AS='wn5*mL' PD='2*mL' PS='2*mL'
Mp6 Q clk_bar H Vdd CMOSP L='mL' W='wp6' AD='wp6*mL' AS='wp6*mL' PD='2*mL' PS='2*mL'
Mn6 Q clk I Vss CMOSN L='mL' W='wn6' AD='wn6*mL' AS='wn6*mL' PD='2*mL' PS='2*mL'
Mp7 H A Vdd Vdd CMOSP L='mL' W='wp7' AD='wp7*mL' AS='wp7*mL' PD='2*mL' PS='2*mL'
Mn7 I A Vss Vss CMOSN L='mL' W='wn7' AD='wn7*mL' AS='wn7*mL' PD='2*mL' PS='2*mL'
Mp8 Q clk K Vdd CMOSP L='mL' W='wp8' AD='wp8*mL' AS='wp8*mL' PD='2*mL' PS='2*mL'
Mn8 Q clk_bar L Vss CMOSN L='mL' W='wn8' AD='wn8*mL' AS='wn8*mL' PD='2*mL' PS='2*mL'
Mp9 K J Vdd Vdd CMOSP L='mL' W='wp9' AD='wp9*mL' AS='wp9*mL' PD='2*mL' PS='2*mL'
Mn9 L J Vss Vss CMOSN L='mL' W='wn9' AD='wn9*mL' AS='wn9*mL' PD='2*mL' PS='2*mL'
Mp10 J Q Vdd Vdd CMOSP L='mL' W='wp10' AD='wp10*mL' AS='wp10*mL' PD='2*mL' PS='2*mL'
Mn10 J Q Vss Vss CMOSN L='mL' W='wn10' AD='wn10*mL' AS='wn10*mL' PD='2*mL' PS='2*mL'

C1 Q 0 200fF
C2 D 0 200fF
C3 test_D 0 200fF
```

Appendix K: Example HSPICE Netlist SN-SN Flip-Flop

```
*SN-SN Flip Flop

.options node list post=2 relv=1e-4 absvar=.3 relvar=0.1
.param mL = 0.35u

* Flip Flop Net List
Mn1 D clk_bar A Vss CMOSN L='mL' W='wn1' AD='wn1*mL' AS='wn1*mL' PD='2*mL' PS='2*mL'
Mn2 D_bar clk_bar B Vss CMOSN L='mL' W='wn2' AD='wn2*mL' AS='wn2*mL' PD='2*mL' PS='2*mL'

Mp1 B A Vdd Vdd CMOSP L='mL' W='wp1' AD='wp1*(mL/2)*5' AS='wp1*(mL/2)*5' PD='5*mL+wp1' PS='5*mL+wp1'
Mn3 B clk C Vss CMOSN L='mL' W='wn3' AD='wn3*mL' AS='wn3*mL' PD='2*mL' PS='2*mL'
Mn4 C A Vss Vss CMOSN L='mL' W='wn4' AD='wn4*mL' AS='wn4*mL' PD='2*mL' PS='2*mL'
Mp2 A B Vdd Vdd CMOSP L='mL' W='wp2' AD='wp2*(mL/2)*5' AS='wp2*(mL/2)*5' PD='5*mL+wp2' PS='5*mL+wp2'
Mn5 A clk E Vss CMOSN L='mL' W='wn5' AD='wn5*mL' AS='wn5*mL' PD='2*mL' PS='2*mL'
Mn6 E B Vss Vss CMOSN L='mL' W='wn6' AD='wn6*mL' AS='wn6*mL' PD='2*mL' PS='2*mL'

Mn7 B_bar clk T Vss CMOSN L='mL' W='wn7' AD='wn7*mL' AS='wn7*mL' PD='2*mL' PS='2*mL'
Mn8 B clk Q_bar Vss CMOSN L='mL' W='wn8' AD='wn8*mL' AS='wn8*mL' PD='2*mL' PS='2*mL'

Mp3 T Q_bar Vdd Vdd CMOSP L='mL' W='wp3' AD='wp3*(mL/2)*5' AS='wp3*(mL/2)*5' PD='5*mL+wp3'
PS='5*mL+wp3'
Mn9 T clk_bar F Vss CMOSN L='mL' W='wn9' AD='wn9*mL' AS='wn9*mL' PD='2*mL' PS='2*mL'
Mn10 F Q_bar Vss Vss CMOSN L='mL' W='wn10' AD='wn10*mL' AS='wn10*mL' PD='2*mL' PS='2*mL'
Mp4 Q_bar T Vdd Vdd CMOSP L='mL' W='wp4' AD='wp4*(mL/2)*5' AS='wp4*(mL/2)*5' PD='5*mL+wp4'
PS='5*mL+wp4'
Mn11 Q_bar clk_bar G Vss CMOSN L='mL' W='wn11' AD='wn11*mL' AS='wn11*mL' PD='2*mL' PS='2*mL'
Mn12 G T Vss Vss CMOSN L='mL' W='wn12' AD='wn12*mL' AS='wn12*mL' PD='2*mL' PS='2*mL'

Mp13 D_bar D Vdd Vdd CMOSP L='mL' W='wp13' AD='wp13*(mL/2)*5' AS='wp13*(mL/2)*5' PD='5*mL+wp13'
PS='5*mL+wp13'
Mn13 D_bar D Vss Vss CMOSN L='mL' W='wn13' AD='wn13*mL' AS='wn13*mL' PD='2*mL' PS='2*mL'
```

```
Mp14 B_bar B Vdd Vdd CMOSP L='mL' W='wp14' AD='wp14*(mL/2)*5' AS='wp14*(mL/2)*5' PD='5*mL+wp14'
PS='5*mL+wp14'
Mn14 B_bar B Vss Vss CMOSN L='mL' W='wn14' AD='wn14*mL' AS='wn14*mL' PD='2*mL' PS='2*mL'

Mp15 Q Q_bar Vdd Vdd CMOSP L='mL' W='wp15' AD='wp15*(mL/2)*5' AS='wp15*(mL/2)*5' PD='5*mL+wp15'
PS='5*mL+wp15'
Mn15 Q Q_bar Vss Vss CMOSN L='mL' W='wn15' AD='wn15*(mL/2)*5' AS='wn15*(mL/2)*5' PD='5*mL+wn15'
PS='5*mL+wn15'

C1 Q 0 200fF
C2 D 0 200fF
C3 test_D 0 200fF
```

Appendix L: Example HSPICE Flip-Flop Optimization Code

```
* sources
.param volt=2
Vcc Vdd 0 DC volt
Vxx Vxx 0 DC volt
.global vdd
.param gnd=0
Vgnd Vss 0 DC gnd
.global vss

*Clock Parameters

.param frequency=100X
.param duty=.5
.param rise=.1ns
.param fall=.1ns
.param level='((1/frequency)*duty)-rise'
.param clk_period='(1/frequency)'

Vclk clk_in 0 pulse gnd volt level rise fall level clk_period
Vclk_test test_clk_in 0 pulse gnd volt level rise fall level clk_period

Vclk_blk Vdd_blk 0 volt
Vdata_gry Vdd_gry 0 volt
Vclk_blk_test Vdd_blk_test 0 DC volt
Vdata_gry_test Vdd_gry_test 0 DC volt

* Loaded Inverters actual inverters

Mp1data data_in_bar data_in Vxx Vxx CMOSP L='mL' W='wp1data' AD='wp1data*(mL/2)*5'
AS='wp1data*(mL/2)*5' PD='5*mL+wp1data' PS='5*mL+wp1data'
```

```
Mn1data data_in_bar data_in Vss Vss CMOSN L='mL' W='wn1data' AD='wn1data*(mL/2)*5'
AS='wn1data*(mL/2)*5' PD='5*mL+wn1data' PS='5*mL+wn1data'
Mp2data D data_in_bar Vdd_gry Vdd_gry CMOSP L='mL' W='wp2data' AD='wp2data*(mL/2)*5'
AS='wp2data*(mL/2)*5' PD='5*mL+wp2data' PS='5*mL+wp2data'
Mn2data D data_in_bar Vss Vss CMOSN L='mL' W='wn2data' AD='wn2data*(mL/2)*5' AS='wn2data*(mL/2)*5'
PD='5*mL+wn2data' PS='5*mL+wn2data'

Mp1clk clk_bar clk_in Vdd_blk Vdd_blk CMOSP L='mL' W='wp1clk' AD='wp1clk*(mL/2)*5' AS='wp1clk*(mL/2)*5'
PD='5*mL+wp1clk' PS='5*mL+wp1clk'
Mn1clk clk_bar clk_in Vss Vss CMOSN L='mL' W='wn1clk' AD='wn1clk*(mL/2)*5' AS='wn1clk*(mL/2)*5'
PD='5*mL+wn1clk' PS='5*mL+wn1clk'
Mp2clk clk clk_bar Vdd_blk Vdd_blk CMOSP L='mL' W='wp2clk' AD='wp2clk*(mL/2)*5' AS='wp2clk*(mL/2)*5'
PD='5*mL+wp2clk' PS='5*mL+wp2clk'
Mn2clk clk clk_bar Vss Vss CMOSN L='mL' W='wn2clk' AD='wn2clk*(mL/2)*5' AS='wn2clk*(mL/2)*5'
PD='5*mL+wn2clk' PS='5*mL+wn2clk'

* Unloaded Inverters dummy inverters

Mp1data_test test_data_in_bar test_data_in Vxx Vxx CMOSP L='mL' W='wp1data' AD='wp1data*(mL/2)*5'
AS='wp1data*(mL/2)*5' PD='5*mL+wp1data' PS='5*mL+wp1data'
Mn1data_test test_data_in_bar test_data_in Vss Vss CMOSN L='mL' W='wn1data' AD='wn1data*(mL/2)*5'
AS='wn1data*(mL/2)*5' PD='5*mL+wn1data' PS='5*mL+wn1data'
Mp2data_test test_D test_data_in_bar Vdd_gry_test Vdd_gry_test CMOSP L='mL' W='wp2data'
AD='wp2data*(mL/2)*5' AS='wp2data*(mL/2)*5' PD='5*mL+wp2data' PS='5*mL+wp2data'
Mn2data_test test_D test_data_in_bar Vss Vss CMOSN L='mL' W='wn2data' AD='wn2data*(mL/2)*5'
AS='wn2data*(mL/2)*5' PD='5*mL+wn2data' PS='5*mL+wn2data'

Mp1clk_test test_clk_bar test_clk_in Vdd_blk_test Vdd_blk_test CMOSP L='mL' W='wp1clk'
AD='wp1clk*(mL/2)*5' AS='wp1clk*(mL/2)*5' PD='5*mL+wp1clk' PS='5*mL+wp1clk'
Mn1clk_test test_clk_bar test_clk_in Vss Vss CMOSN L='mL' W='wn1clk' AD='wn1clk*(mL/2)*5'
AS='wn1clk*(mL/2)*5' PD='5*mL+wn1clk' PS='5*mL+wn1clk'
Mp2clk_test test_clk test_clk_bar Vdd_blk_test Vdd_blk_test CMOSP L='mL' W='wp2clk'
AD='wp2clk*(mL/2)*5' AS='wp2clk*(mL/2)*5' PD='5*mL+wp2clk' PS='5*mL+wp2clk'
```

```
Mn2clk_test test_clk test_clk_bar Vss Vss CMOSN L='mL' W='wn2clk' AD='wn2clk*(mL/2)*5'
AS='wn2clk*(mL/2)*5' PD='5*mL+wn2clk' PS='5*mL+wn2clk'

.param length=4ns
.param delay='((clk_period-4n)/2)'
.param data_period='2*clk_period'

* Data Activity of 1 1010101...
Vin data_in 0 pulse gnd volt delay rise fall length data_period
Vin_test test_data_in 0 pulse gnd volt delay rise fall length data_period

.param
+wp1=opt1(1.2u, 1.2u, 22u)
+wn1=opt1(1.2u, 1.2u, 22u)
+wp2=opt1(1.2u, 1.2u, 22u)
+wn2=opt1(1.2u, 1.2u, 22u)
+wp3=opt1(1.2u, 1.2u, 22u)
+wn3=opt1(1.2u, 1.2u, 22u)
+wp4=opt1(1.2u, 1.2u, 22u)
+wn4=opt1(1.2u, 1.2u, 22u)
+wn5=opt1(1.2u, 1.2u, 22u)
+wn6=opt1(1.2u, 1.2u, 22u)
+wn7=opt1(1.2u, 1.2u, 22u)
+wn8=opt1(1.2u, 1.2u, 22u)
+wn9=opt1(1.2u, 1.2u, 22u)
+wn10=opt1(1.2u, 1.2u, 22u)
+wn11=opt1(1.2u, 1.2u, 22u)
+wn12=opt1(1.6u, 1.2u, 22u)
+wp13=opt1(1.2u, 1.2u, 22u)
+wn13=opt1(1.2u, 1.2u, 22u)
+wp14=opt1(1.2u, 1.2u, 22u)
+wn14=opt1(1.2u, 1.2u, 22u)
+wp15=opt1(5.6u, 1.2u, 22u)
```

```
+wn15=opt1(5.6u, 1.2u, 22u)
+wp1data=opt1(20.4u, 1.2u, 22u)
+wn1data=opt1(9.7u, 1.2u, 22u)
+wp2data=opt1(11.0u, 1.2u, 22u)
+wn2data=opt1(12.7u, 1.2u, 22u)
+wp1clk=opt1(11.4u, 1.2u, 22u)
+wn1clk=opt1(8.1u, 1.2u, 22u)
+wp2clk=opt1(13.7u, 1.2u, 22u)
+wn2clk=opt1(17.8u, 1.2u, 22u)

.param tran_end='(8*clk_period)'
.TRAN 0.1ns tran_end
.TRAN 0.05ns tran_end sweep optimize=opt1 results=tpopt,tpclkopt,tpclk_baropt,tpdataopt,pdp,
model=optmod

.model optmod opt itropt=30 max=1e+5

.ic D=0 A=0 B=volt C=volt Q=0 E=0 data_in=0 data_in_bar=volt

*
* Measure Power
*
.meas tran avg_curr avg I(Vcc) from='0.5*tran_end' to='tran_end'
.meas tran avg_curr_clk avg I(Vclk_blk) from='0.5*tran_end' to='tran_end'
.meas tran avg_curr_data avg I(Vdata_gry) from='0.5*tran_end' to='tran_end'
.meas tran avg_curr_clk_test avg I(Vclk_blk_test) from='.5*tran_end' to='tran_end'
.meas tran avg_curr_data_test avg I(Vdata_gry_test) from='.5*tran_end' to='tran_end'

.meas avg_power param = '-avg_curr*volt'
.meas avg_power_clk param = '-avg_curr_clk*volt'
.meas avg_power_data param = '-avg_curr_data*volt'
.meas avg_power_clk_test param = '-avg_curr_clk_test*volt'
.meas avg_power_data_test param = '-avg_curr_data_test*volt'
```

```
.meas tran tropt trig v(clk) val='volt/2' rise=3 targ v(Q) val='volt/2' rise=2
.meas tran tfopt trig v(clk) val='volt/2' rise=4 targ v(Q) val='volt/2' fall=2
.meas tran trclkopt trig v(clk) val='.1*volt' rise=1 targ v(clk) val='.9*volt' rise=1
.meas tran tfclkopt trig v(clk) val='.9*volt' fall=1 targ v(clk) val='.1*volt' fall=1
.meas tran trclk_baropt trig v(clk_bar) val='.1*volt' rise=1 targ v(clk_bar) val='.9*volt' rise=1
.meas tran tfclk_baropt trig v(clk_bar) val='.9*volt' fall=1 targ v(clk_bar) val='.1*volt' fall=1
.meas tran trdataopt trig v(D) val='.1*volt' rise=2 targ v(D) val='.9*volt' rise=2
.meas tran tfdataopt trig v(D) val='.9*volt' fall=2 targ v(D) val='.1*volt' fall=2

.meas tpopt param='(tropt+tfopt)/2' goal=100ps
.meas tpclkopt param='(trclkopt+tfclkopt)/2' goal=100ps
.meas tpclk_baropt param='(trclk_baropt+tfclk_baropt)/2' goal=100ps
.meas tpdataopt param='(trdataopt+tfdataopt)/2' goal=100ps

.meas avg_power_eff param='avg_power-40u'
.meas avg_power_clk_eff param='avg_power_clk-avg_power_clk_test'
.meas avg_power_data_eff param='avg_power_data-avg_power_data_test'

.meas PDP param='(avg_power_eff+avg_power_clk_eff+avg_power_data_eff)*tpopt' goal=10f minval=50f

.model

.option post
.END
```

Appendix M: Example HSPICE Code for Flip-Flop Power Measurements

```
* sources
.param volt=2
Vcc Vdd 0 DC volt
Vxx Vxx 0 DC volt
.global vdd
.param gnd=0
Vgnd Vss 0 DC gnd
.global vss

*Clock Parameters

.param frequency=100X
.param duty=.5
.param rise=.1ns
.param fall=.1ns
.param level='((1/frequency)*duty)-rise'
.param clk_period='(1/frequency)'

Vclk clk_in 0 pulse gnd volt level rise fall level clk_period
Vclk_test test_clk_in 0 pulse gnd volt level rise fall level clk_period

Vclk_blk Vdd_blk 0 DC volt
Vdata_gry Vdd_gry 0 DC volt
Vclk_blk_test Vdd_blk_test 0 DC volt
Vdata_gry_test Vdd_gry_test 0 DC volt

* Loaded Inverters actual inverters

Mp1data data_in_bar data_in Vxx Vxx CMOSP L='mL' W='wp1data' AD='wp1data*(mL/2)*5'
AS='wp1data*(mL/2)*5' PD='5*mL+wp1data' PS='5*mL+wp1data'
```

```
Mn1data data_in_bar data_in Vss Vss CMOSN L='mL' W='wn1data' AD='wn1data*(mL/2)*5'
AS='wn1data*(mL/2)*5' PD='5*mL+wn1data' PS='5*mL+wn1data'
Mp2data D data_in_bar Vdd_gry Vdd_gry CMOSP L='mL' W='wp2data' AD='wp2data*(mL/2)*5'
AS='wp2data*(mL/2)*5' PD='5*mL+wp2data' PS='5*mL+wp2data'
Mn2data D data_in_bar Vss Vss CMOSN L='mL' W='wn2data' AD='wn2data*(mL/2)*5' AS='wn2data*(mL/2)*5'
PD='5*mL+wn2data' PS='5*mL+wn2data'

Mp1clk clk_bar clk_in Vdd_blk Vdd_blk CMOSP L='mL' W='wp1clk' AD='wp1clk*(mL/2)*5' AS='wp1clk*(mL/2)*5'
PD='5*mL+wp1clk' PS='5*mL+wp1clk'
Mn1clk clk_bar clk_in Vss Vss CMOSN L='mL' W='wn1clk' AD='wn1clk*(mL/2)*5' AS='wn1clk*(mL/2)*5'
PD='5*mL+wn1clk' PS='5*mL+wn1clk'
Mp2clk clk clk_bar Vdd_blk Vdd_blk CMOSP L='mL' W='wp2clk' AD='wp2clk*(mL/2)*5' AS='wp2clk*(mL/2)*5'
PD='5*mL+wp2clk' PS='5*mL+wp2clk'
Mn2clk clk clk_bar Vss Vss CMOSN L='mL' W='wn2clk' AD='wn2clk*(mL/2)*5' AS='wn2clk*(mL/2)*5'
PD='5*mL+wn2clk' PS='5*mL+wn2clk'

* Unloaded Inverters dummy inverters

Mp1data_test test_data_in_bar test_data_in Vxx Vxx CMOSP L='mL' W='wp1data' AD='wp1data*(mL/2)*5'
AS='wp1data*(mL/2)*5' PD='5*mL+wp1data' PS='5*mL+wp1data'
Mn1data_test test_data_in_bar test_data_in Vss Vss CMOSN L='mL' W='wn1data' AD='wn1data*(mL/2)*5'
AS='wn1data*(mL/2)*5' PD='5*mL+wn1data' PS='5*mL+wn1data'
Mp2data_test test_D test_data_in_bar Vdd_gry_test Vdd_gry_test CMOSP L='mL' W='wp2data'
AD='wp2data*(mL/2)*5' AS='wp2data*(mL/2)*5' PD='5*mL+wp2data' PS='5*mL+wp2data'
Mn2data_test test_D test_data_in_bar Vss Vss CMOSN L='mL' W='wn2data' AD='wn2data*(mL/2)*5'
AS='wn2data*(mL/2)*5' PD='5*mL+wn2data' PS='5*mL+wn2data'

Mp1clk_test test_clk_bar test_clk_in Vdd_blk_test Vdd_blk_test CMOSP L='mL' W='wp1clk'
AD='wp1clk*(mL/2)*5' AS='wp1clk*(mL/2)*5' PD='5*mL+wp1clk' PS='5*mL+wp1clk'
Mn1clk_test test_clk_bar test_clk_in Vss Vss CMOSN L='mL' W='wn1clk' AD='wn1clk*(mL/2)*5'
AS='wn1clk*(mL/2)*5' PD='5*mL+wn1clk' PS='5*mL+wn1clk'
Mp2clk_test test_clk test_clk_bar Vdd_blk_test Vdd_blk_test CMOSP L='mL' W='wp2clk'
AD='wp2clk*(mL/2)*5' AS='wp2clk*(mL/2)*5' PD='5*mL+wp2clk' PS='5*mL+wp2clk'
```

```
Mn2clk_test test_clk test_clk_bar Vss Vss CMOSN L='mL' W='wn2clk' AD='wn2clk*(mL/2)*5'
AS='wn2clk*(mL/2)*5' PD='5*mL+wn2clk' PS='5*mL+wn2clk'


.param length=4ns
.param delay='((clk_period-4n)/2)'
.param data_period='2*clk_period'

* Data Activity of 1 1010101...
.param activity = 1
Vin data_in 0 pulse gnd volt delay rise fall length data_period
Vin_test test_data_in 0 pulse gnd volt delay rise fall length data_period

* Data Activity of 0.5 1011001...
*.param activity = 0.5
*Vin data_in 0 pwl 0 volt '4.25*clk_period-rise' volt '4.25*clk_period' 0 '5.25*clk_period-rise' 0
'5.25*clk_period' volt '6.25*clk_period-rise' volt '6.25*clk_period' 0 '7.25*clk_period-rise' 0
'7.25*clk_period' volt '8.75*clk_period-rise' volt '8.75*clk_period' 0 '11.25*clk_period-rise' 0
'11.25*clk_period' volt '11.75*clk_period-rise' volt '11.75*clk_period' 0 '16*clk_period-rise' 0
'16*clk_period' volt '20.25*clk_period-rise' volt '20.25*clk_period' 0 '21.25*clk_period-rise' 0
'21.25*clk_period' volt '22.25*clk_period-rise' volt '22.25*clk_period' 0 '23.25*clk_period-rise' 0
'23.25*clk_period' volt '24.75*clk_period-rise' volt '24.75*clk_period' 0 '27.25*clk_period-rise' 0
'27.25*clk_period' volt '27.75*clk_period-rise' volt '27.75*clk_period' 0
*Vin_test test_data_in 0 pwl 0 volt '4.25*clk_period-rise' volt '4.25*clk_period' 0 '5.25*clk_period-
rise' 0 '5.25*clk_period' volt '6.25*clk_period-rise' volt '6.25*clk_period' 0 '7.25*clk_period-rise' 0
'7.25*clk_period' volt '8.75*clk_period-rise' volt '8.75*clk_period' 0 '11.25*clk_period-rise' 0
'11.25*clk_period' volt '11.75*clk_period-rise' volt '11.75*clk_period' 0 '16*clk_period-rise' 0
'16*clk_period' volt '20.25*clk_period-rise' volt '20.25*clk_period' 0 '21.25*clk_period-rise' 0
'21.25*clk_period' volt '22.25*clk_period-rise' volt '22.25*clk_period' 0 '23.25*clk_period-rise' 0
'23.25*clk_period' volt '24.75*clk_period-rise' volt '24.75*clk_period' 0 '27.25*clk_period-rise' 0
'27.25*clk_period' volt '27.75*clk_period-rise' volt '27.75*clk_period' 0
```

```
.param
+wp1=1.6u
+wn1=1.8u
+wp2=1.2u
+wn2=1.3u
+wp3=1.2u
+wn3=1.2u
+wp4=1.6u
+wn4=1.2u
+wn5=1.2u
+wn6=1.2u
+wn7=1.8u
+wn8=1.3u
+wn9=1.2u
+wn10=1.2u
+wn11=1.2u
+wn12=1.2u
+wp13=2.4u
+wn13=1.4u
+wp14=2.4u
+wn14=1.4u
+wp15=6.2u
+wn15=4.5u
+wp1data=22u
+wn1data=22u
+wp2data=22u
+wn2data=22u
+wp1clk=22u
+wn1clk=22u
+wp2clk=22u
+wn2clk=22u

.param tran_end='(32*clk_period)'
```

```
.TRAN 0.1ns tran_end

.model optmod opt itropt=30 max=1e+5

.ic D=0 A=0 B=volt F=0 C=volt Q=0 E=0 H=volt I=0

*
* Measure Power
*
.meas tran avg_curr avg I(Vcc) from='.5*tran_end' to='tran_end'
.meas tran avg_curr_clk avg I(Vclk_blk) from='.5*tran_end' to='tran_end'
.meas tran avg_curr_data avg I(Vdata_gry) from='.5*tran_end' to='tran_end'
.meas tran avg_curr_clk_test avg I(Vclk_blk_test) from='.5*tran_end' to='tran_end'
.meas tran avg_curr_data_test avg I(Vdata_gry_test) from='.5*tran_end' to='tran_end'

.meas avg_power param = '-avg_curr*volt'
.meas avg_power_clk param = '-avg_curr_clk*volt'
.meas avg_power_data param = '-avg_curr_data*volt'
.meas avg_power_clk_test param = '-avg_curr_clk_test*volt'
.meas avg_power_data_test param = '-avg_curr_data_test*volt'

.meas tran tropt trig v(clk) val='volt/2' rise=3 targ v(Q) val='volt/2' rise=2
.meas tran tfopt trig v(clk) val='volt/2' rise=4 targ v(Q) val='volt/2' fall=2
.meas tran trclkopt trig v(clk) val='.1*volt' rise=1 targ v(clk) val='.9*volt' rise=1
.meas tran tfclkopt trig v(clk) val='.9*volt' fall=1 targ v(clk) val='.1*volt' fall=1
.meas tran trclk_baropt trig v(clk_bar) val='.1*volt' rise=1 targ v(clk_bar) val='.9*volt' rise=1
.meas tran tfclk_baropt trig v(clk_bar) val='.9*volt' fall=1 targ v(clk_bar) val='.1*volt' fall=1
.meas tran trdataopt trig v(D) val='.1*volt' rise=2 targ v(D) val='.9*volt' rise=2
.meas tran tfdataopt trig v(D) val='.9*volt' fall=2 targ v(D) val='.1*volt' fall=2
.meas tpopt param='(tropt+tfopt)/2' goal=100ps
.meas tpclkopt param='(trclkopt+tfclkopt)/2' goal=100ps
.meas tpclk_baropt param='(trclk_baropt+tfclk_baropt)/2' goal=100ps
.meas tpdataopt param='(trdataopt+tfdataopt)/2' goal=100ps
```

```
.param activity_power = '(frequency*200f*volt*volt*activity)/2'

.meas avg_power_eff param='avg_power-activity_power'
.meas avg_power_clk_eff param='avg_power_clk-avg_power_clk_test'
.meas avg_power_data_eff param='avg_power_data-avg_power_data_test'
.meas avg_power_total_eff param='(avg_power_eff+avg_power_clk_eff+avg_power_data_eff)'

.meas PDP param='avg_power_total_eff*tpopt' goal=20f
```

Appendix N: Example HSPICE Code for Maximum Frequency

```
* sources
.param volt=3.3
Vcc Vdd 0 DC volt
Vxx Vxx 0 DC volt
.global vdd
.param gnd=0
Vgnd Vss 0 DC gnd
.global vss

*Clock Parameters

.param frequency=1700X *1700X for 3.3V; 1000X for 2V; 610X for 1.5V
.param duty=.5
.param rise=.1ns
.param fall=.1ns
.param level='((1/frequency)*duty)-rise'
.param clk_period='(1/frequency)'

Vclk clk_in 0 pulse gnd volt level rise fall level clk_period
*Vclk_bar clk_bar 0 pulse volt gnd level rise fall level clk_period

Vclk_blk Vdd_blk 0 volt
Vdata_gry Vdd_gry 0 volt

*Mp1data data_in_bar data_in Vxx Vxx CMOSP L='mL' W='wp1data' AD='wp1data*(mL/2)*5'
AS='wp1data*(mL/2)*5' PD='5*mL+wp1data' PS='5*mL+wp1data'
*Mn1data data_in_bar data_in Vss Vss CMOSN L='mL' W='wn1data' AD='wn1data*(mL/2)*5'
AS='wn1data*(mL/2)*5' PD='5*mL+wn1data' PS='5*mL+wn1data'
*Mp2data D data_in_bar Vdd_gry Vdd_gry CMOSP L='mL' W='wp2data' AD='wp2data*(mL/2)*5'
AS='wp2data*(mL/2)*5' PD='5*mL+wp2data' PS='5*mL+wp2data'
```

```
*Mn2data D data_in_bar Vss Vss CMOSN L='mL' W='wn2data' AD='wn2data*(mL/2)*5' AS='wn2data*(mL/2)*5'
PD='5*mL+wn2data' PS='5*mL+wn2data'

Mp1clk clk_bar clk_in Vdd_blk Vdd_blk CMOSP L='mL' W='wp1clk' AD='wp1clk*(mL/2)*5' AS='wp1clk*(mL/2)*5'
PD='5*mL+wp1clk' PS='5*mL+wp1clk'
Mn1clk clk_bar clk_in Vss Vss CMOSN L='mL' W='wn1clk' AD='wn1clk*(mL/2)*5' AS='wn1clk*(mL/2)*5'
PD='5*mL+wn1clk' PS='5*mL+wn1clk'
Mp2clk clk clk_bar Vdd_blk Vdd_blk CMOSP L='mL' W='wp2clk' AD='wp2clk*(mL/2)*5' AS='wp2clk*(mL/2)*5'
PD='5*mL+wp2clk' PS='5*mL+wp2clk'
Mn2clk clk clk_bar Vss Vss CMOSN L='mL' W='wn2clk' AD='wn2clk*(mL/2)*5' AS='wn2clk*(mL/2)*5'
PD='5*mL+wn2clk' PS='5*mL+wn2clk'


.param length=4ns
.param delay='((clk_period-4n)/2)'
.param data_period='2*clk_period'

Vin data_in 0 pulse gnd volt delay rise fall length data_period

.param
+wp1=1.6u
+wn1=1.8u
+wp2=1.2u
+wn2=1.3u
+wp3=1.2u
+wn3=1.2u
+wp4=1.6u
+wn4=1.2u
+wn5=1.2u
+wn6=1.2u
+wn7=1.8u
+wn8=1.3u
+wn9=1.2u
```

```
+wn10=1.2u
+wn11=1.2u
+wn12=1.2u
+wp13=2.4u
+wn13=1.4u
+wp14=2.4u
+wn14=1.4u
+wp15=6.2u
+wn15=4.5u
+wp1data=22u
+wn1data=22u
+wp2data=22u
+wn2data=22u
+wp1clk=22u
+wn1clk=22u
+wp2clk=22u
+wn2clk=22u
+wpinv=9u
+wninv=9u


.param tran_end='(20*clk_period)'
.TRAN 0.1ns tran_end

.model optmod opt itropt=30 max=1e+5

.ic D=0

*
* Measure Power
*
.meas tran avg_curr avg I(Vcc) from='.5*tran_end' to='tran_end'
.meas tran avg_curr_clk avg I(Vclk_blk) from='.5*tran_end' to='tran_end'
```

```
.meas tran avg_curr_data avg I(Vdata_gry) from='.5*tran_end' to='tran_end'
.meas avg_power param = '-avg_curr*volt'
.meas avg_power_clk param = '-avg_curr_clk*volt'
.meas avg_power_data param = '-avg_curr_data*volt'
.meas tran tropt trig v(clk) val='volt/2' rise=3 targ v(Q) val='volt/2' rise=3
.meas tran tfopt trig v(clk) val='volt/2' rise=4 targ v(Q) val='volt/2' fall=4
.meas tran trclkopt trig v(clk) val='.1*volt' rise=1 targ v(clk) val='.9*volt' rise=1
.meas tran tfclkopt trig v(clk) val='.9*volt' fall=1 targ v(clk) val='.1*volt' fall=1
.meas tran trclk_baropt trig v(clk_bar) val='.1*volt' rise=1 targ v(clk_bar) val='.9*volt' rise=1
.meas tran tfclk_baropt trig v(clk_bar) val='.9*volt' fall=1 targ v(clk_bar) val='.1*volt' fall=1
.meas tran trdataopt trig v(D) val='.1*volt' rise=2 targ v(D) val='.9*volt' rise=2
.meas tran tfdataopt trig v(D) val='.9*volt' fall=2 targ v(D) val='.1*volt' fall=2
.meas tpopt param='(tropt+tfopt)/2' goal=100ps
.meas tpclkopt param='(trclkopt+tfclkopt)/2' goal=100ps
.meas tpclk_baropt param='(trclk_baropt+tfclk_baropt)/2' goal=100ps
.meas tpdataopt param='(trdataopt+tfdataopt)/2' goal=100ps
.meas PDP param='(avg_power+avg_power_clk+avg_power_data)*tpopt' goal=20f
```