Asynchronous circuits have recently been a breakthrough in many high performance computers. The concept of asynchronous circuits which started a long time ago has slowly grasped the attention of many designers. The Muller-C-element is an important control block in many asynchronous designs and therefore it is important to understand some of the possible failures that might occur in this circuit. The timing and behavior of this element will have an important effect on the overall performance of the system. The purpose of this research is to study some of the common failures that exist in synchronous logic and find out if these failures can also happen in the C-element.

Condition for a failure must be present in order for it to occur. Understanding the conditions required for a circuit failure to occur, we will show realistic examples in the applications of C-element in which such similar conditions will also happen. In this thesis, we are interested in analyzing the circuit failures in C-element due to different logic threshold voltages of different devices, problem of charge-sharing and metastability
characteristic of circuit. Simulations results will show such failures does occur in the C-element when the conditions were met.
FAILURE ANALYSIS OF MULLER-C-ELEMENT

by

OONPIN CHEW

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed August 5th, 1996
Commencement June 1997
Master of Science thesis of Oonpin Chew presented on August 5th, 1996

APPROVED:

Redacted for Privacy
Major Professor, representing Electrical and Computer Engineering

Redacted for Privacy
Chair 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

/ Oonpin Chew
ACKNOWLEDGMENTS

All glory and honor to my Lord Jesus Christ for His peace, wisdom and blessings.

Many thanks to my parents, Mr. and Mrs. Lim Theam Chew for sponsoring my education here in Oregon State University (United States of America). Special thanks to Dr. Shih-Lien Lu for his kind and patience guidance in my studies and research work. A great appreciate to friends that have given me great support throughout this whole process of getting this master thesis completed. This thesis is also dedicated to my dear friend YeeLing Soh for her constant encouragement and motivation. Special thanks also to Kris and Dennis Moder for their effort in proofreading this thesis.
# TABLE OF CONTENTS

1. INTRODUCTION ......................................................... 1
   1.1 Asynchronous and Synchronous Digital System ................. 1
   1.2 Micropipeline and Muller-C-element .......................... 4
   1.3 Micropipeline ..................................................... 5
   1.4 Hazards and Failures ............................................. 10
   1.5 Purpose of This Thesis .......................................... 12
   1.6 Organization of This Document ................................. 13

2. DESIGN OF MULLER-C-ELEMENT .................................... 14
   2.1 Muller-C-element ................................................ 14
   2.2 Dynamic Implementation of Muller-C-element ............... 17
   2.3 Static Implementations of Muller-C-element .................. 18

3. CIRCUIT FAILURES .................................................. 22
   3.1 Failure due to Different Logic Threshold Voltages .......... 22
      3.1.1 Failure due to Different Logic Threshold Voltages in C-element .... 24
      3.1.2 Analysis due to Different Logic Threshold Voltages ............... 27
   3.2 Charge-Sharing .................................................. 28
   3.3 Failure due to Metastable State ............................... 31
      3.3.1 Study of Metastable State in a Latch ....... 33
   3.4 Failure due to Metastable State in C-element ................ 37
   3.5 Observations and Summary ..................................... 44
<table>
<thead>
<tr>
<th>TABLE OF CONTENTS (Continued)</th>
</tr>
</thead>
</table>

4. PROPOSE IMPROVEMENTS ................................................................. 47
  4.1 Output-Controlled Muller-C-element .............................................. 47
  4.2 Other Possible Improvements ....................................................... 53
    4.2.1 State Diagram ........................................................................ 54
    4.2.2 Analysis of State Diagram ...................................................... 57
    4.2.3 Timing Requirement and Logic Threshold Voltages ...................... 58
    4.2.4 Device Sizing and Layout ....................................................... 60

5. CONCLUSION ....................................................................................... 62
  5.1 Muller-C-element in Self-timed Circuits .......................................... 62
  5.2 Findings and Result Summary ....................................................... 63
  5.3 Future Work .................................................................................. 65

BIBLIOGRAPHY ......................................................................................... 66

APPENDICES ............................................................................................. 68

  Appendix A Hspice netlist for Ivan Sutherland’s Muller-C-element ......... 69
  Appendix B Hspice output file for Ivan Sutherland’s Muller-C-element .... 71
  Appendix C Hspice netlist for output-controlled Muller-C-element ......... 77
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Symbolic representation of Muller-C-element</td>
<td>4</td>
</tr>
<tr>
<td>2.</td>
<td>A simple example of a pipeline interconnection circuit that controls data transfer between blocks A and B</td>
<td>6</td>
</tr>
<tr>
<td>3.</td>
<td>The structure of micropipeline</td>
<td>7</td>
</tr>
<tr>
<td>4.</td>
<td>Event control for 4-cycle pipeline</td>
<td>9</td>
</tr>
<tr>
<td>5.</td>
<td>The structure of 4-cycle FIFO</td>
<td>10</td>
</tr>
<tr>
<td>6.</td>
<td>Hspice simulation result for muller-C-element</td>
<td>15</td>
</tr>
<tr>
<td>7.</td>
<td>Muller-C-elements with inverters</td>
<td>16</td>
</tr>
<tr>
<td>8.</td>
<td>Dynamic implementation of Muller-C-element using electrical capacitor as internal storage element</td>
<td>18</td>
</tr>
<tr>
<td>9.</td>
<td>CMOS static implementation of Muller-C-element</td>
<td>19</td>
</tr>
<tr>
<td>10.</td>
<td>Output-controlled Muller-C-element</td>
<td>20</td>
</tr>
<tr>
<td>11.</td>
<td>Simple example of a system</td>
<td>23</td>
</tr>
<tr>
<td>12.</td>
<td>Micropipeline with feedback environment</td>
<td>24</td>
</tr>
<tr>
<td>13.</td>
<td>Failure in the C-element due to different logic threshold voltages</td>
<td>26</td>
</tr>
<tr>
<td>14.</td>
<td>Input structure of C-element and charge-sharing, guide for analysis</td>
<td>29</td>
</tr>
<tr>
<td>15.</td>
<td>Dynamic implementation of C-element showing charge-sharing mechanism</td>
<td>30</td>
</tr>
<tr>
<td>16.</td>
<td>Metastable state</td>
<td>33</td>
</tr>
<tr>
<td>17.</td>
<td>Definition of setup time</td>
<td>33</td>
</tr>
<tr>
<td>18.</td>
<td>Small signal model for a latch</td>
<td>35</td>
</tr>
<tr>
<td>19.</td>
<td>Hspice simulation showing possible metastable state in C-element</td>
<td>38</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES (continue)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.</td>
<td>Modeling the C-element with series of input transistors and a cascaded inverter</td>
</tr>
<tr>
<td>21.</td>
<td>Spectrum of output by varying Tss (setup time)</td>
</tr>
<tr>
<td>22.</td>
<td>Simulation showing relationship between voltage at node 4 ($\Delta V_4$) and output node ($\Delta V_8$) which might cause a failure (metastable state or functional failure (curve3) in the C-element</td>
</tr>
<tr>
<td>23.</td>
<td>Simulation showing functional correctness of output-controlled C-element</td>
</tr>
<tr>
<td>24.</td>
<td>Simulation showing better tolerance level for output-controlled C-element</td>
</tr>
<tr>
<td>25.</td>
<td>Simulation showing output-controlled C-element producing a correct response to the inputs as compared to a failure using the classical implementation of the C-element</td>
</tr>
<tr>
<td>26.</td>
<td>Simulations showing correct response using output-controlled C-element</td>
</tr>
<tr>
<td>27.</td>
<td>Schematic of classical C-element showing the internal nodes</td>
</tr>
<tr>
<td>28.</td>
<td>State diagram of Muller-C-element</td>
</tr>
</tbody>
</table>
FAILURE ANALYSIS IN MULLER-C-ELEMENT

CHAPTER 1

INTRODUCTION

As computers have evolved rapidly over the years, much attention has been focused on synchronous circuits. However, designs using a synchronous methodology have many constraints that designers are not able to address. The main constraint is timing. This has resulted in the re-emergence of asynchronous (self-timed) computers. In many synchronous designs, a master clock is used to "synchronize" the activities on a chip, and this clock speed has become the distinguishing difference in many seemingly identical systems.

The distribution of this master clock (global clock) has become an important speed limiting factor in VLSI design today. The switching behavior in synchronous circuits also creates large current spikes in the substrate, resulting in switching noise being introduced to the system. All of these factors become more obvious with the rapid shrinking of feature sizes and the corresponding increase in integration at the chip level.

1.1 Asynchronous and Synchronous Digital System

Asynchronous, or self-timed logic, systems run at their own speed. The circuit generates a completion signal when it is ready to accept new input data. This completion
signal then can be used to trigger further processing of the resulting data. The performance of the logic is solely based on local timers and the availability of data to work on. Thus, by having the data available at a higher speed, the operations can be completed at a higher speed rate without restriction from the clock edges or levels that govern passing of data from one state of a machine to the other.

Asynchronous logic is not a new area. Many digital communication systems which require high-speed clocking use some ingenious ways to maintain the correct sequence of operation. There are many reasons that motivate the study of asynchronous logic, many are important issues that pose problems in many VLSI clocked systems. Power factor is the key reason for asynchronous logic. Power consumption has become an important issue recently. Enormous efforts have been put into designing low-power, high-speed VLSI circuits. This interest is partly due to the increase in battery-powered computing devices.

As VLSI feature sizes shrink to 0.5-micron level and below, products are getting smaller and chip density and the number of transistors increase. The heat that is given out by the system then becomes an important issue. Synchronous logic makes all the modules on a chip to function even though it might not be executing anything. This forcing is due to the CPU clock that forces the unit to run in place because a missed synchronization will result in a “failure” in the system.

As mentioned earlier, VLSI logic is getting more and more complex in order to incorporate numerous functions on a chip. The global clock that synchronizes the activities on a chip must be connected to many more components. This physical constraint results in the problem of clock skew. The demand for a more a precise clock signal also creates a problem in synchronous circuits.
To contrast between synchronous and asynchronous timing disciplines, we present an example which will illustrate the difference [1,2,3]. Pictorially, asynchronous system work like a scheduled train line. At every designated interval, there will be a train taking off from the station whether there is a full load of cargo or none at all. A particular passenger has to synchronize his/her travel itinerary with the schedule of the train. In contrast, an asynchronous system is like travelling in your own car. There is no fear of missing a scheduled departure time. You may visit a new place whenever you have finished visiting an old location.

In asynchronous logic, we are interested in the sequence of events. We no longer allow the different computation blocks to relinquish their synchronization responsibility to the global clock. Because of this separation from the global clock, the problems found in synchronous logic are less prominent, or virtually insignificant in asynchronous circuits because the unit is functioning based on the completion of other units that synchronize the whole data flow in the processor. Also, the replacement of a clock signal by event completion signals and data signals greatly reduces the problem of sharp transient switching. There are also other advantages in designing asynchronous logic. These include scalability, reduction in ground bounce problems, and the ability to design for testability due to the high degree of controllability and observability on asynchronous circuits. All these are possible because each individual module of the chip can be designed and built separately. Thus, the testing and verification of each chip can also be done independently. With each module or chip verified to be functionally correct, they can be assembled on a single chip if the area permits with no extra timing constraint needing to be satisfied.
1.2 Micropipelines and Muller-C-element

To understand how an asynchronous or clockless machine works, we have to first look at Ivan Sutherland's micropipeline[4]. Sutherland proposed a different timing discipline which uses the transition signals to control the flow of complex operations and data flow in the system. This concept frees up the constraint of lockstep synchronous execution where the clock period must accommodate the longest delay path. As mentioned earlier, the control signals are transition signals or data that control the flow of operations and data. Control circuits for transition signalling are built out of modules that form various logical combinations of events. The exclusive OR (XOR) circuit acts as an OR element for events. When either input of an XOR circuit changes states, its output also changes states.

The Muller-C-element acts as the AND element for events. When both inputs of a Muller-C-element are in the same logical state, the Muller-C-element's output takes the same state as the inputs. When the two inputs differ, the Muller-C-element uses the internal storage element to maintain the previous state and hold its output. Therefore only after an event takes place on both of its inputs will the Muller-C-element produce an event at its output. The logical symbol for the Muller-C-element is as shown in Figure 1.

![Figure 1: Symbolic Representation of Muller-C-element.](image)
The control of Sutherland's micropipeline is composed of a string of Muller-C-elements. Therefore the accuracy and performance of this gate is a critical requirement for the performance of the micropipeline.

1.3 Micropipeline

As mentioned above, one of the applications of Muller-C-elements can be found in the Sutherland micropipeline. However, before we look at the actual structure of the micropipeline, we would first need to develop a basic model for the request-acknowledge protocol[5] which is used in the implementation of micropipeline. The simplest interconnection circuit and pipelining handshake circuit is shown in Figure 2. The signal Rin (the completion signal of computation of Block A) is used to see if the output datum of block A is valid. It then checks the feedback acknowledge Ain to see if block B is ready for a new input. This is important since block A might take a longer time to finish its computation than block B. This is important because the input of block B will be overwritten if insufficient time is given for the completion of its computation. Rout controls the request signal to block B, indicating when block B should start the evaluation. Aout controls the acknowledge signal to the interconnection block preceding block A, notifying block A when its output datum is transferred to block B.
Having the basic concept of an event-driven pipeline, we can now study the structure of the micropipeline. A string of Muller-C-elements interspersed with inverters is the only logic requirement to control the pipeline. Let us first try to understand how a pipeline without any processing, (i.e. an FIFO) would work, and how it can be constructed using simple control logic which is made up of a string of Muller-C-elements. This is shown in Figure 3.
The pipeline is first initialized at logic low. Therefore, one of the input states of the C-element is at high due to the inversion of signals in this specific type of C-element. Rin is asserted high when the data for Din is ready to be captured by the first stage of the pipe.
As the other input to the C-element is ready to accept the data, Din will be captured in the register and send a signal to the second stage of the pipe after a time delay. The C-element of the second stage, upon receiving this signal, will start to capture the data from the first stage register and, at the same time, pass a control signal to the first stage, showing that data is in the process of transferring.

The control signal that starts off the second stage (Aout) is also sent to the preceding stage of the pipe which will switch Rin back to a logic low state waiting for the next ready signal to come.

This event-controlled storage element is required to respond to both the rising and falling transitions. This can be achieved by using two latches side by side, one controlled by a control wire called “captured”, and the other by another wire called “pass”, which are activated alternately. The reason the registers are arranged to be driven from one end while their control signals are being sensed from the other is because the control signals for the register must be amplified to drive all the switches in the many storage elements involved. Since the wires that carry control signals are long, there is always some delay in controlling the register.

We can also use a 4-cycle request-acknowledge protocol for the functionality of the micropipeline. Assume that the four signals Rin, Rout, Ain and Aout are initially at logic level 0 (Rin-, Rout-,Ain-,Aout-). When block A finishes its computation, it raises Rin(Rin+) to request for a data transfer to block B. Since Ain is initially low, meaning that block B is ready to accept a new input, the handshake circuit raises Aout(Aout+) to tell block A that its output datum has been accepted. Rin can then be reset(Rin-). The
handshake circuit then raises Rout(Rout+) to initiate the computation in block B. Eventually block B will complete its task and output a completion signal. This information is fed back through Ain(Ain+) so that Rout(Rout-) gets a reset which will in turn reset Ain(Ain-) and complete the four-phase handshake loop. The four-phase handshake protocol always uses the rising transitions to initiate operation and the falling transitions to reset. The four-phase handshake protocol dictates that the sequence of signal transitions on the right-hand side of the handshake circuit in Figure 4 is always the iterative Rout+ -> Ain+ -> Rout- -> Ain-, and on the left-hand side, Rin+ -> Aout+ -> Rin- -> Aout-.

With this protocol and the basic control structure that we have adopted in the 2-phase handshake, we can obtain a very simple circuit for the event control of a 4-cycle pipeline/FIFO, as shown in Figure 5.

![Figure 4: Event control for 4-cycle pipeline](image-url)
1.4 Hazards And Failures

Without careful design of any circuit, there will be a chance for hazardous conditions to happen. These problems include metastable state, unwanted short pluses, or
incorrect signals from the circuit. Because hazards will create a failure to the circuit, therefore it is necessary to carefully implement the circuit, considering all areas of constraints and requirements needed to construct a stable circuit.

Any cross-coupled gate in a circuit potentially exhibits metastability. In a synchronous system, the clock rates are so tailored that the clock signal always trails the data. The setup and hold-time constraints are requirements in the synchronous circuits to prevent metastable states from happening. Metastability occurs due to the inherent analog nature of the storage elements used in all electronic circuits. A bi-stable element has two stable states: namely the “zero” and “one”. Under the right conditions, the element may enter a metastable state where the output is an undetermined state between 0 and 1. Another place where metastability is noticed is in cases where mutual exclusion is usually handled by an arbiter. These circuits are prominent in both synchronous and asynchronous circuits. Ensuring that the data and the clock bear the correct relation with each other is easier in synchronous systems and thus makes this problem of metastability easier to tackle. However, this problem is more prominent in the asynchronous systems since there is no master clock to control the data flow and the operation of the logical or functional units.

The difference between the threshold voltages of different devices can also cause potential problem to the circuit[6]. Interpretation of logic level for different devices will be different depending on its threshold voltage. This situation is more prominent in asynchronous logic as individual modules or devices are independent. Therefore, even with feedback circuits like in the case of micropipeline that uses the request-acknowledge protocol, the environment that reads the acknowledge signal might have a different
interpretation of the logic state because of its threshold voltage, thereby resulting in generating a false signal to the control. We will show that such a situation is possible in the real example of the micropipeline which we studied in 1.2.

Some research has focused on the occurrence of metastable states in digital CMOS circuits like latched [7] flip-flops [8]. In asynchronous logic, the control of data flow are event driven which is the primary function of the Muller-C-element. Thus it is essential that the Muller-C-element be stable. The occurrence of a metastable state will result in a misinterpretation of data which will cause a failure in the control of data flow in the system, thus resulting in a failure which is undesirable in any circuit or system.

1.5 Purpose of This Thesis

Successful design of digital systems with asynchronous inputs requires good management and understanding of timing relations. Improper timing inputs will result in a failure to the system. As mentioned in section 1.3, failures might occur due to differences in the threshold voltage for different devices and therefore, interpretation of the logic state will differ from one device to another. Also, the timing constraint problems that will cause a failure to the circuit will also be introduced. The purpose of this thesis is to first study and analyze the potential failures that might occur in the C-element and the conditions that are needed to be present for such a failure to occur. The conditions might occur in the asynchronous system with the usage of the C-element as its control, for example the micropipeline. Some circuit design problems include charge-sharing and the occurrence of metastable state. These conditions will cause the circuit to fail.
1.6 **Organization of This Document**

In the next chapter, we will study the different implementations of the Muller-C-element. Failures will be introduced in Chapter 3, along with the difference in threshold voltages for different devices, a problem which creates a potential failure condition to the system due to misinterpretation of transition. Other circuit problems, including charge-sharing and the metastable state, will also be discussed. Chapter 4 will include some other improvements we can make to the design of C-element. Output-controlled C-element will also be introduced in Chapter 4 and we will show that this implementation of C-element will have a higher tolerance and will improve the performance of the C-element under the same conditions where failures might have occurred. We will conclude and hint at future directions of research in Chapter 5.
CHAPTER 2

DESIGN OF MULLER-C-ELEMENT

CMOS (Complimentary Metal Oxide Silicon) technology is recognized as the preferred manufacturing process technology for VLSI systems. CMOS provides an inherently low power static circuit technology which has the capability of providing a lower power-delay product than comparable design-rule bipolar, nMOS or GaAs technologies. The Muller-C-element also uses CMOS technology for its implementation. For these reasons, the CMOS implementation of the Muller C-element will be studied.

2.1 Muller-C-element

As mentioned in the introduction, the Muller-C-element is an event-driven AND gate used in asynchronous logic. The simple truth table for the Muller-C-element (non-inverting inputs) is shown in Table 1.

Table 1: Truth Table for Muller-C-element (non-inverting inputs)

<table>
<thead>
<tr>
<th>Muller-C-element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inputs</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>0 0</td>
</tr>
<tr>
<td>0 1</td>
</tr>
<tr>
<td>1 0</td>
</tr>
<tr>
<td>1 1</td>
</tr>
</tbody>
</table>

Qn is the previous output state of the Muller-C-element. An English description for the Muller-C-element is shown in Figure 7.
An example of the behavior for the Muller-C-element with respect to the inputs can be seen from the Hspice simulation result shown in Figure 6. We can see that the output of the Muller-C-element maintains a logic low initially (initial condition is low), and switches to a logic high only when both of the inputs are at a logic high state. The output is maintained at logic high until both inputs are in the same state again (in this case, a logic low). The output of the Muller-C-element will then switch to low following the state of the inputs.

Figure 6: Hspice simulation result for muller-C-element
Although the absolute state of a transition signal might not matter much, it does matter when the state of this transition signal is relative to the other signals. Sometimes a transition signal is relatively related to the inversion of another signal. Thus, it is sometimes important to invert the signals or produce an inversion signal.

The symbolic representation for the inversion is similar to those in standard digital logic symbol representation. A "bubble" represents an inverted signal such as illustrated in Figure 7. Later in the chapter, we will introduce the different implementations of the Muller-C-element. In this thesis, we will only consider the non-inverting input and output signal to the Muller-C-element for simplification.

![Diagram of Muller-C-elements with inverters](image-url)

Figure 7: Muller-C-elements with inverters.
2.2 **Dynamic Implementation of Muller-C-element**

A simple dynamic implementation of the Muller-C-element is shown in Figure 8. This circuit uses the electrical capacitance as an internal storage element. A and B are the inputs of the Muller-C-element. If the inputs are in the same states, the capacitance stores the state of the inputs. For example, when inputs A and B are low, the capacitor will be charged to VDD through the PMOS transistors. Similarly when both inputs are high. The charge in the capacitor is discharged to VSS through the NMOS transistors, thereby changing the output to a logic “1” state through the output inverter. When input A is at logic “0” and input B at logic “1”, charges are shared between two nodes. Since charges in node 4 control the output of the Muller-C-element, deterioration of charges at node 4 will result in an unstable state at the output of the Muller-C-element.

Like all dynamic memories, this dynamic implementation of the Muller-C-element requires memory refresh. The charges on the electrical capacitance are subjected to charge decay. Without refreshing, the memory stored in the Muller-C-element might get corrupted due to charge decay, and output an incorrect state through the element causing a failure in the system. Note that the transistor to initialize the Muller-C-element during master clear is not shown in the circuit.
2.3 Static implementations of Muller-C-element

By replacing the memory capacitor with static logic, we have the static CMOS Muller-C-element, shown in Figure 9. The sizes of the static logic that replace the electrical capacitance in the dynamic implementation can be smaller in size since they are just used to retain the value which has already been established. We can also see that this general circuit can be expanded to three or more inputs.

Figure 8: Dynamic implementation of Muller-C-element using electrical capacitor as internal storage element
Figure 9: CMOS static implementation of Muller-C-element

The static logic that replaces the memory capacitor controls the internal storage memory of the Muller-C-element. From the circuit, we can see that the inputs to the Muller-C-element control the source and drain transistors (M9 and M10 respectively) of the internal storage static logic. Referring to Figure 9, when both inputs X and Y are at logic "0", M1 and M2 are turned on, sourcing node 4 to VDD through transistors M7 and M8. The voltage at node 4 drives the inverter and produces a logic "0" state at the output. The
output of the Muller-C-element drives the input of the cross-coupled inverter, which in this case turns on M9, storing the present state of the Muller-C-element in the internal static logic. If input changes to a logic “1”, this will switch M2 off and M3 on. Since input Y is still at “0” and the output is also at “0”, it will keep M9 and M8 on and thus source node 4 to VDD, maintaining a logic low state at the output.

The same analysis can be done for both input at a logic “1” state. Notice that when both inputs are different, there will not be a change of potential at node 4; thus, the element maintains the same state as the stored internal memory of the Muller-C-element.

Figure 10: Output-controlled Muller-C-element
Another static implementation of the Muller-C-element can be constructed by re-arranging the transistor of the Muller-C-element as shown in Figure 10. This implementation, which will be referred to as output-controlled Muller-C-element throughout this thesis, creates a larger internal capacitance at node 4 which will increase the tolerance level of the C-element. This will be shown in Chapter 4. The functionality is the same as the classical implementation.
CHAPTER 3  

CIRCUIT FAILURES

There are many factors that will contribute to a failure in a circuit. This chapter will study two specific failures that might occur in both the synchronous and asynchronous logic. The first potential problem to the system is the difference in threshold voltages for different devices. This problem is even more prominent in an asynchronous circuit because individual blocks are designed independently. This problem is aggravated when the load of the driver is large. This will be examined later in the chapter. Because it is asynchronous, the frequencies of the inputs would be different. This would also cause a lot of problems to the circuit. Metastability is another phenomena that will cause a failure to the system. Although it is quite difficult to understand this phenomena of metastability, we will show that such failure does occur in the C-element.

3.1 Failure due to Different in Logic Threshold Voltages

Effective methods for asynchronous VLSI circuits require a simple abstraction of the VLSI medium. An attractive abstraction has been introduced by Martin[9,10]. He views a VLSI circuit as a network of so-called VLSI operators connected by wires. VLSI operators form a generalization of well-known (combinational) gates, such as AND and NOR, but also include sequential components such as the Muller-C-elements and various latches. Therefore, we can see that there are a lot potential problems in the circuit due to
this difference in logic threshold voltages for different VLSI operators. A simple example of such a problem can be seen in Figure 11.

Block A, Block B, and the control module are different VLSI operators. The logical threshold voltages for both Block A and Block B are different. When a VLSI operator responds to a productive transition, it does so at a particular voltage level at the relevant input: logic threshold (voltage). If Block A is given an artificially low logic threshold voltage compared to Block B, Block A will respond to a transition from the feedback signal indicating that Block B has completed the task and is therefore sending a signal to the control unit, turning Block B off. This is a failure to the system since Block B hasn’t yet completed its task. This effect is even more prominent if the load of Block B is large and therefore requires a longer time to switch to either a strong high or low signal. During this slow switching time, a transition signal from the Block A might occur anytime, even between the two logical states of the device, causing a failure to the system.
3.1.1 Failure due to Different Logic Threshold in C-element

We can see that the above example is very similar to Sutherland’s micropipeline as discussed in Chapter 1. A recapture of the micropipeline with only one pipeline stage and a feedback environment is shown in Figure 12.

Figure 12: Micropipeline with feedback environment
We can see the potential problem in Figure 12 if the logic threshold for the feedback environment is different from that of the C-element. This is a possible situation since the feedback environment may be constructed separately. As mentioned in the beginning of the chapter, a large capacitive load will aggravate this problem. In the case of the micropipeline, the load seen by the C-element can be very large when it is driving a large number of registers. The large capacitive load creates a long rising or falling time for the C-element and if the feedback environment has a low logic threshold, it will respond to the productive transition, giving a false result to the C-element. Simulation is shown in Figure 13. These simulations show the output of the C-element with two different feedback environments, of which one is of a lower logic threshold compared to the other. Initial conditions for the micropipeline is at logic “0”. The C-element used in the micropipeline has one of the inputs inverted and therefore, what the C-element sees is a logic “1” at that input. This is simulated by asserting input 2 to high. When the first block is ready to accept the data, Rin will be asserted. This is simulated by asserting input 1 to high (shown in Figure 13). Because the load to the C-element is large, it raises slowly (Aout). Knowing how a micropipeline function (as discussed in Chapter 1), the acknowledgment of Aout will switch Rin to Rin-. Therefore, input 1 (Rin) is reset to a logic “0” state.

Plot 1 show the set of inputs that have a set of inputs that the feedback logic have a higher voltage threshold interpretation as the set of input shown in plot 3. Plot 2 shows the response to the input set of plot 1 and plot 4 shows the response to the input set of plot 3. The different response by the different feedback environment is simulated by shortening the pulse width of input 1 (Rin). The feedback environment that has a smaller logic threshold
voltage will respond faster, thus producing a shorter pulse. The difference in pulse width for this simulation is 0.1ns. With such a small difference in reacting time, one case produces a correct answer after a period of time, entering into a metastable state (Plot2) and recovers, producing the final correct response to the inputs. The second simulation caused a failure in the circuit since the response to the inputs is incorrect. (Plot 4).

Figure 13: Failure in the C-element due to different logic threshold voltages
Note that the potential at node 4 in plot 4 raises much faster than the output and reaches the threshold voltage for one of the transistors in the output driver (M11) before the output (node8) reaches the threshold voltage for the feedback transistor (M10). If M10 is turned on, it will drain the potential at node 4 to VSS, driving the output of the C-element to a correct logic "1" state. The reverse is true for Plot 2. Transistors are the same size for simple circuit analysis (PMOS: W=8um, NMOS: W=4um, L=2u). For circuit schematics, refer to Figure 9.

3.1.2 Analysis of Failure due to Different Logic Threshold Voltages

This problem of different logic threshold voltages is inevitable in asynchronous logic. Even if we matched the logic threshold voltages for all devices (uniform logic threshold voltages)[6], there is still a possibility for this condition to happen. An example of this is mentioned in [6] section 3.3. The difference in capacitance of wires connecting the VLSI operators and the use of asymmetric and symmetric forks (isochronic forks)[6] which are necessary in some cases would also create such problems.

Because the inputs of the C-element can come from different environments and different frequencies (since it is asynchronous), the scenario shown in the simulation is real. With the difference in logic threshold voltages between devices, we have also shown that even with feedback, this condition can also occur. Therefore, knowing that there might be a failure if such a condition occurs, we would need to understand how the C-element reacts to such conditions. From the simulation, the potential at node 4 and output node (node 8)
are important factors that determine the final result of the C-element. Depending on the time needed for node 4 to charge or discharge compared to the time needed for the output node (node 8), different outcomes can result. The time needed to charge or discharge the potential at a node is directly related to the capacitance as seen by that node. We need to obtain a relationship between the capacitance at these two nodes, in order to determine the behavior of the circuit.

The possible voltage supply to node 4, the capacitance at node 4, the load capacitance, and the sequence of input transitions to the C-element are all contributing factors to the failure we have shown in the simulation. The next few sections will discuss these factors, showing some simple analysis on a simpler circuit, and we will conclude this chapter by relating these concepts to the composition of the C-element.

3.2 Charge-Sharing

We will be able to see that there are possibilities for charge-sharing to occur in the Muller-C-element. This mechanism of charge-sharing will occur in the C-element if a sequence of transitions takes place and creates the proper conditions for this to happen. The input structure to the C-element is shown in Figure 14(a).

Taking the sequence of input A and B as (0 0) -> (0 1) -> (1 0) -> (1 1) -> (0 1), we can see that node 3 and 4 are first charged to a potential of VDD through M1 and M2. The assertion of B will discharge the potential at node 2 to VSS, and the subsequent assertion of signal A will make both node 3 and 4 at a potential of VSS.
Figure 14(a)(b): Input structure of C-element and charge-sharing, guide for analysis

Note that at this time, node 3 is still at a potential of VDD. With input A going to a logic “0” state, it will turn on M2. By turning off M3, this will create a potential difference between node 2 and 4. The mechanism of charge-sharing occurred as there is no path to VDD or VSS, and charges are shared among these two nodes. Modeling a transistor as a switch, we can simplify to the circuit as shown in Figure 14(b).

Assume that the capacitance at node 2 is equivalent to the capacitance at node 4 i.e C0 = C1. When node 2 is charged to VDD (take VDD as 5V), using the quantitative charge equation,

\[ Q = CV \]  \hspace{1cm} (3.1)

the charge at node 2 will be 5C. When the switch is closed, the total equivalent capacitance of the circuit will be:
\[ C_{total} = C_0 + C_1 = 2C \text{ (capacitors in parallel, } C_0 = C_1) \]

Using the theory of charge conservation,

\[ Q = C_{total} \cdot V \]

\[ \Rightarrow V = Q/(2C) \]

\[ V = 2.5V \text{ (for a } 5V \text{ VDD)} \]

This charge-sharing problem is more prominent in the dynamic implementation of the C-element as there is no feedback signal to recharge the internal memory logic as compared to the static CMOS implementation of the C-element. Simulation of the dynamic C-element using the same transitions as described above is shown in Figure 15.

Figure 15: Dynamic implementation of C-element showing charge-sharing mechanism
However, under certain conditions, this problem of charge-sharing cannot be neglected even in the static CMOS implementation of the C-element. The shared potentials contribute or reduce the potential at node 4. As we have studied in the previous section, if the internal capacitance is small and the load is large, the internal capacitance may be charged or discharged at a much faster rate than the load which switches the output inverter to a different state, causing a failure to the system.

3.3 Failure due to Metastable State

As mentioned earlier, any cross-coupled inverters will present a potential problem for a metastable state to occur. The metastable state takes place if the effect of the input signal to the latch is too close to the decision threshold. Because of the insufficient information retained in the latch, the latch is unable to decide which way to settle for an extended period of time. The metastable state is a peculiar state of data storage circuits from which it takes an unusually long time to settle to the final state. If the latch circuit is more complex, node voltages may even oscillate before settling at the final state.

In a completely synchronous logic circuit, we can reduce the clock frequency so we can increase the setup time of the latch. Then the latch acquires decisive information before the latch's input circuit turns off, and the metastable state does not occur. The metastable state matters essentially in asynchronous logic circuits; the output driver of the subsystem that sends data out and the input latch that receives that data have different clocks. With reference to Figure 16(a) and (b), the clock edge of CKR (curve 1 of Figure 16(b)), we can
have any relation with the clock edge of CKD (curve 2-4) since both of them are operating at different frequencies. Curve 2 shows that CKD is much earlier than CKR; curve 3 shows that both clock edges are happening at almost the same time; and curve 4 shows that CKD happens much later than CKR. If CKD is earlier than CKR, the latch captures stable low logic level data. If CKD is earlier than CKR, the latch captures stable high logic level data. However, if the two clocks are almost synchronous as shown by curve 3, the input data waveform 3' provides confusing information to the latch of chip B. This is the case of metastability[8] and is an example of failure in the synchronous circuits. The occurrence of such a condition is common with MCM (multi-chip modules) where a system is built from many ASIC (application specific) chips. Many of these chips are operating at their own frequency and hence the system would have to handle the problems of different frequencies. When placed together, a metastable state may occur. Furthermore the timing relation may also depend on environmental conditions like temperature. The metastable state makes the assembly of large, high-speed systems quite difficult. Some solutions include:

1. Assembling systems compactly and trimming transmission delays accurately using some kind of advanced system assembly technology
2. Sending clock and data together, using matched transmission line (isochronic fork)
3. Receiving clock from signal

All the above are possible solutions to metastable state in synchronous logics, unfortunately, they are all very costly.
3.3.1 Study of Metastable State in a Latch

A latch's response to clock and data is characterized by setup time. Sufficient time must be given to the data to be stable before the latch opens to capture the data. Figure 17(a) shows a realistic input circuit.
Figure 17(b) shows $T_s$, which is the setup time for the latch to capture a stable logic state. Therefore the latch will fail if the setup time requirement is not met, which may result in a metastable state.

Sufficient time must also be given to hold the data valid before the latch closes ($T_h$). This is known as hold time or data valid time. The latch captures the correct data only if the setup and hold time requirements are met. The requirement that the input data must have been stabilized at least by the setup time before the clock edge is equivalent to saying that the latch circuit must have acquired sufficient internal information to determine the future state when the connection of the data input path is cut off.

To analyze this problem of metastability, many researchers have approached this problem by using the ac small-signal frequency domain analysis[7][8]. We can also analyze this problem in the time domain[9]. In the time domain approach, the latch (usually a consisting of a back to back inverter) is set at a marginal triggering condition for the metastable state by adjusting the setup time for both inputs. The voltage difference grows exponentially with respect to the time.

The error rate due to the metastable state is exponentially proportional to the resolving time constant $\tau$. This resolving time constant is inversely proportional to the gain-bandwidth ($gm$) of the closed-loop positive feedback system. When a latch is in a metastable state, then it can be viewed as a differential amplifier that is biased at an operating voltage $V_{GSW}$, where the input and the output of the inverter are the same value, which is exactly the metastable state. There are many factors that govern the occurrence of
the metastable state in latch or flip flops. From the above statement we can see that \( g_m \) is one of the parameters that will affect the error rate due to metastability. However, there are also parameters that will contribute to the occurrence of the metastable state. Capacitance for the cross-coupled inverters, Miller effects, sizes of device, temperature, threshold potential, substrate doping, channel-length modulation, and static noise margin[7] are all other possible factors that will affect the error rate due to metastability. In this thesis, we will analyze this problem of metastability of a latch with relation to its gain-bandwidth \( g_m \) and capacitance.

In order to analyze this quantitatively, it is necessary to capture the cross-coupled inverters using the linear model of MOS devices. This modeling is necessary in order to simplify the analysis needed to be done to characterize the metastable behavior of this device.

Using the small signal analysis, back to back inverter can be modeled as shown in Figure 18.

![Small signal model for a latch](image)

Figure 18: Small signal model for a latch
Using KCL, we can write the equations to describe this circuit:

\[ \begin{align*}
    gm_1(V_1) + C_1 \frac{d}{dt}(V_1) - \frac{V_1}{R_1} &= 0 \\
    gm_2(V_2) + C_2 \frac{d}{dt}(V_2) - \frac{V_2}{R_2} &= 0
\end{align*} \] (3.2)

As the first order model, the PMOS is modeled as a load resistance \( R \) because it is in the saturation region and the NMOS is modeled as a voltage-controlled current source \( gm \).

By solving the equations, we can see that the solution of the equation is in the form of exponential \( e^{at} \)

\[ v(t) = e^{\left(-\frac{1-gmR}{RC}\right)t} \] (3.3)

\[ a = -\left(\frac{1-gmR}{RC}\right)t \]

\( \tau \) can be seen in the form of

\[ e^{\frac{t}{\tau}} \]

therefore

\[ \tau = \frac{RC}{1-gmR} \] (3.4)

From equation (3.4), we can see that the larger the capacitance at the node, the longer the time needed. The time constant will also be dependent of the \( gm \) of the inverter. \( \tau \) can be seen as the time for the system to recover from metastable state. The values of \( \tau \) for one node as compare to the other in the back-to-back will determine the outcome of the system.
The concept of metastability might appear to be easy to comprehend; however, to completely understand this problem of metastability, the calculations can be very complex, since it involves many other factors including the non-linearity properties of MOS transistors. There are many published papers that addresses this phenomena of a metastable state in a latch using both small and large signal models [11][12].

3.4 Failure due to Metastable State in C-element

Understanding how a metastable state can happen to a latch, we can now study this type of failure in the C-element. As discussed in the previous section, it is necessary to have sufficient setup and hold time for the latch in order to capture the correct data. The same requirement must be met for the Muller-C-element. The setup time here is the relative time between the two signal edges to the C-element. Simulation of these conditions not being met is shown in Figure 19.

The example shown in Figure 16(a)(b) is realistic in the context of asynchronous logic since the frequencies of which data is sent from one point to another are different. In the case of the micropipeline (refer to Figure 12), if the pipeline is first initialized at a logic low state, when the first pipe is ready to accept the data, it will assert a high signal (simulated by input 2 going high). Note that the C-element used in the micropipeline has one of the inputs inverted and hence the supply to the C-element is at a logic low state (simulated by input 1 at logic low state, also assuming that the inverter has a zero delay). These two transitions will generate a control signal to the next stage. Considering that the
delay for the feedback environment is slower than that of the second stage, the signal from the second stage might come in faster than that from the feedback environment, forcing the other input to a logic low state (simulated by input 2 going low and input 1 rises before falling of input 1).

![Diagram of C-element CMOS circuit (metastability)](image)

Figure 19: Hspice simulation showing possible metastable state in C-element

As there is no clock to synchronize the flow of inputs to the C-element, the input in this case (input 2) shuts off much earlier with respect to the rising edge of input 1. Thus, there is not enough internal information for the C-element to make a decision like in the case of a latch as studied in the previous section.

Blocks are connected together by using wire, and some signals have to travel a longer path, thereby creating a longer delay. Others might be able to travel a shorter path
and therefore the relationship between both inputs is extremely difficult to predict in the case of asynchronous circuits.

Referring back to the schematic of the Muller-C-element, we can see that there is a pair of cascaded inverters (modeling of C-element with cascaded inverters is shown in Figure 20). Therefore, we would expect the possibility of a metastable state to occur in the circuit. Treating the C-element like a latch, we can do a complete analysis of the metastability characteristics of the C-element. The main concern in this thesis is to find out if this form of failure does occur in the C-element without going into detailed analysis for the circuit.

A metastable state occurs when the system is not driven. This means that the system is at a point where it needs to decide which direction to switch. Using the small signal analysis to determine the parameter $\tau$ is insufficient [9]. From the schematic of the C-element (Figure 9), we can see that the duration for which the element is not being driven is very short because the feedback from the output will switch the necessary transistor "on" and drive the C-element to a known state.

The relationship between the two voltage potentials at this decision voltage, which is also known as unstable bias voltage, will determine whether the system will function correctly, fail, or enter into a metastable state. Therefore, if the potential voltages of both points reached the unstable bias voltage at the same instant of time $t$, we would expect a metastable state to occur. We describe this condition as $\Delta V_0 = \Delta V_1 = V_{GSW}$ where $\Delta V_0$ and $\Delta V_1$ are the change in potential voltage at each end of the crossed-coupled inverters at time $t$, where the system is in the process of determining the state of the system (metastable state).
and $V_{GSW}$ is the unstable bias voltage. Depending on the relationship between the gain and the capacitance at the node, the time constants $\tau$ for the system to recover itself (without external driven force or static noise) differ. Calculations for these parameters can be found in [13-15].

$\Delta V_i(t)$ is the change of voltage at node $i$ at time $t$, and therefore, we can view $\Delta V_4$ as the change of voltage at node 4 and $\Delta V_8$ as the change of voltage at the output node of the C-element. If $\Delta V_4(t)$ is greater than $\Delta V_8(t)$ at any instances during the duration of time that the system is determining its state, and $V_4(t)$ passes the decision threshold of the transistor of the output driver (in the case of the C-element), it will drive the output to a wrong state causing a failure to the system. On the other hand, if $\Delta V_4(t)$ is smaller than $\Delta V_8(t)$, the C-element will respond to the stable state, switching on the necessary feedback transistors, driving the output to a correct state and producing a correct response. The time of consideration will be the recovery time needed for the system to stabilize its output after entering into a metastable state. This parameter will be the time constant $\tau$ or $T$ as mentioned earlier. Looking at equation (3.1), we know the current equation is

$$I = \frac{dQ}{dt}$$

(3.2)

using Equation (3.1)

$$\therefore \Delta V = \frac{I_d \Delta t}{C}$$

(3.3)

re-arranging (3.7)

$$\Delta t = \frac{C \Delta V}{I_d}$$

(3.4)
Figure 20: Modeling the C-element with a series of input transistor and a cascaded inverter

In the simulation as shown in Figure 19, the correct response for the output should be at logic “1” state when both the inputs are high. But if the C-element is driving a large load, it will take a longer time to rise to a logic “1” state. If the change at node 4 is large enough, it will switch the output back to a logic “0” state, causing a failure to the system. If the capacitive load at node 4 is smaller as compared to the load, we would expect it to charge or discharge faster than the output node which has a large capacitive load.

Taking the load of the C-element to be large which is common in the application of the micropipeline and looking at the classical implementation of Muller-C-element as shown in Figure 9, we can see that by raising input 2 as shown in Figure 17, this transition will turn both transistors M3 M4 on, and node 4 will be drained to a logical low state.
However, from Equation (3.3), the large load at the output will need to take a longer time for the C-element to charge or discharge. This slow rising of the output voltage (from the simulation) didn’t shut off the PMOS completely (changing from a non-sat to sat condition), and therefore, there is still a current flow from VDD to node 4. The capacitance at node 4 is much smaller than that of the load. Taking the same transitions as shown in the simulation, we can see that the mechanism of charge-sharing will occur in the circuit while M10 is off because the output feedback signals have not reached the decision threshold voltage of the NMOS.

With the additional supply of voltage from the charge-sharing mechanism, plus the voltage supply due to the current flow through M9, which is the dominating factor, the small capacitance at node 4 might be charged-up to the decision threshold of the transistor (in this simulation, M12) much faster than at the output. Therefore, we would expect that by using the same scenario, by varying the parameter of Tss, we would have a spectrum of possible outputs ranging from producing a correct output response, output entering into a metastable state, and incorrect output response after a certain range of time. The increase in the parameter Tss would also mean that the output response with the same load would start to rise at an earlier time and reach the decision threshold voltage (in this simulation, it is the decision threshold of M10) faster as compared to that of M12, which is determined by the potential at node 4. The spectrum of output by varying the parameter Tss with constant load is shown in Figure 21.
The simulation is run on level 2 model and the Vth (threshold voltage) is approximately 1.7V. Therefore between 1.7V and 3.3V, it is a “no man’s land” and a rough estimation of the unstable bias voltage will be around 2.5V. From the simulations, we can see that the metastable state occurs at approximately 2.5V as estimated. This voltage is the unstable bias voltages (V_{GSW}) that we described earlier. In the case of the C-element, ΔV₀ is equivalent to ΔV₄ and ΔV₁ is equivalent to ΔV₈. Looking at the relationship between the potential voltage at node 4 (ΔV₄) and output node (ΔV₈) that produces curves 1, 2 and 3 as shown in Figure 21, we can show that this analysis is correct through simulations. Figure 22 shows these relationships. Note that the comparison is done throughout the time duration where the C-element is in a metastable state.
Figure 22: Simulation showing relationship between voltage at node 4 ($\Delta V_4$) and output node ($\Delta V_8$) which might cause a failure (metastable state or functional failure (curve3)) in the C-element

3.5 Observations and Summary

We have studied several of the failures that could happen in a circuit and have related it back to the Muller-C-element. The difference in logic threshold interpretation between logic blocks will create conditions that might cause the C-element to fail. Such failure happens when the C-element is driving a heavy load and signals from the feedback
environment due to misinterpretation of the logic state switches and resets one of the inputs while the output of the C-element is yet to be stabilized. The C-element can be used in any manner in the asynchronous design. The wide scope of its applications increases the possibility for such conditions to occur that could result in failure. Such hazardous conditions can even occur in systems that have feedback signals to ensure proper timing requirements are met. One example we have mentioned is the micropipeline, and we have cited examples that hazardous conditions can happen in the application of the C-element in the micropipeline. Simulation results have shown that with a misinterpretation of logic state that causes the signal to reset with a small differential of 0.2ns, the response result may have different outcomes: either a correct response or a failure in the circuit may occur.

In section 3.3, we discussed the failure due to timing problems in the latch. In particular, if the setup and hold time for the latch is not met, there will be a condition for which the potential changes between the two nodes are approximately equal to the unstable bias voltage at a certain time and this will cause a metastable state to occur. This is directly related to the changes of voltage at one end of the inverter to the other end in a cross-coupled inverter. The potential at these nodes are highly dependent on the capacitance seen at these nodes. We can see from Equation (3.3) that a larger capacitance at the node will require a longer time to charge and discharge, so the sizes of the capacitance at a particular node will determine the changes in voltage with respect to time for that node.

Adding on to this problem under a certain sequence of transitions, there is a possibility of charge sharing to occur. Though the charge shared among the nodes are not much contribution to the potential at that node, it will nevertheless create some problems
and in some case aggravate the problem as we have discussed in this chapter.

The behavior of the C-element that reacts to the situation caused by different logic threshold voltages that result in a failure is the same reaction we saw when the inputs to the C-element are not meeting the timing requirements. The latter caused the C-element to enter into a metastable state and break after a duration of time. The relationship between the changes in potential voltages between the internal node and output node is the same reason that would cause a failure to the circuit.
CHAPTER 4

PROPOSED IMPROVEMENTS

The relationship between the capacitance of the internal node (node4) and the load driven by the C-element is directly related to the change in potential voltages at these nodes with time. Under certain conditions, a failure might occur. The difference in logic threshold voltages between devices will also create conditions that will create a circuit failure to the C-element. Realistic examples of possible hazardous conditions which occur due to not meeting the timing requirement were described in the previous chapter. These conditions will create possible circuit failures in the C-element. The reasons behind these failures are the same for both cases.

4.1 Output-Controlled C-element

The capacitance seen at the internal node (node4) is dependent on the composition of the C-element. Therefore it is necessary to study the capacitance seen at node 4 in different compositions of the C-element. We will only compare the classical C-element used in Sutherland’s micropipeline and the output-controlled C-element which is a rearrangement of transistors from the classical implementation.

As we can see from the simulations as shown in Figure 21, the relationship between the changes of potential voltage at node 4 to the potential changes at node 8 with time. If
the change of voltage at node 4 is the same as the voltage that change in node 8 at a specific
time equal to the unstable bias voltage, the C-element goes into the metastable state. This
decondition is described in section 3.4. Therefore, it is necessary (using the condition as
shown in the simulation) to make sure that the voltage potential at node 4 does not reach
the decision threshold voltage at a shorter time as compared to the output voltage which is
used to switch the correct transistor “on”, driving the device into a correct stable state.
Therefore, with larger capacitance at node 4, the more tolerant the circuit will be under
certain conditions. We need to balance the capacitive load between these two crucial nodes
to make sure that its functionality is not compromised.

By increasing the capacitance of node 4, it will require a longer time to charge to
the decision voltage for the NMOS (M12 in Figure 9). When the output reaches the unstable
bias voltage, the change in potential at the output node should be much higher than at node
4. This will drive the correct transistor “on” and produce a correct response. We can very
easily take the reverse of this scenario and analyze the circuit which results in the same
conclusion.

The dynamic response of MOS systems are strongly dependent on the parasitic
capacitances associated with the MOS device and interconnection capacitances that are
formed by other materials that connect the transistors together. The total load capacitance
on the output of a CMOS gate is the sum of

1. Gate capacitance (of other inputs connected to the output of the gate)
2. Diffusion capacitance (of the drain regions connected to the output)
3. Routing capacitance (of connections between the output and the other inputs)
Estimating the capacitance of a CMOS circuit, we would need to identify the parasitic capacitance of a MOS transistor. They are:

1. $C_{gs}, C_{gd} = \text{gate-to-channel capacitances, which are lumped at the source and}$
   \hspace{1cm} \text{the drain regions of the channel, respectively.}$
2. $C_{sb}, C_{db} = \text{source and drain-diffusion capacitance to bulk or substrate; and}$
3. $C_{gb} = \text{Gate-to-bulk capacitances}$

The total gate capacitance can be seen as $C_g = C_{gb} + C_{gs} + C_{gd}$

Now looking at the composition of the Muller-C-element, we will start to examine the capacitance of the classical C-element as shown in Figure 9. The capacitance at node 4 will be a contribution of two $C_g$ from M11 and M12, four $C_{\text{drain}}$ of M2, M3, M9 and M10. Therefore, the total capacitance at node 4 is $C_{\text{node4}} = 2C_g + 4C_{\text{drain}}$.

Referring to the schematic of output-controlled Muller-C-element shown in Figure 10, this is just a re-arrangement of transistors from the classical implementation of the C-element. There is no change in the logical function of the C-element using this method of implementation. This re-arrangement of transistors in the circuits increases the internal capacitance of the circuit at node 4. Node 4 now sees the parasitic capacitance of an additional inverter. This will contribute two more $C_{\text{drain}}$ to the total capacitance at node 4. Therefore the total capacitance now at node 4 is $C_{\text{node4}} = 2C_g + 6C_{\text{drain}}$. This increase in capacitance at node 4 without addition of extra devices causes the C-element to have a higher tolerance level than the classical implementation. Therefore, we would expect that under specific conditions which might create a failure in the classical C-element, a lesser
problem would be posed using this implementation. Using the same three scenarios shown in Figure 22, Figure 23 to 25 will compare the potential at the output and node 4 of both the implementation of C-element. Figure 23 shows the functional correctness of the output-controlled C-element. Figure 24 shows that the output of the output-controlled C-element gives a correct response without going into the metastable state. Figure 25 shows that the output-controlled C-element, because of its better tolerance level, gives a correct response to the input while the classical implementation gives a wrong response resulting in a failure in the circuit. To distinguish between the waveform for different implementations of the C-element, node 4 is mapped to node 3 and node 8 is mapped to node 7 for the output-controlled C-element.

Figure 23: Simulation showing functional correctness of output-controlled C-element
Figure 24: Simulation showing better tolerance level for output-controlled C-element

Figure 25: Simulation showing output-controlled C-element producing a correct response to the inputs as compared to a failure using the classical implementation of the C-element.
From all three simulation results, we can see that the output-controlled C-element has a higher tolerance level compared to the classical implementation due to the increase in capacitance of the internal node 4. This is achieved by simply rearranging the transistors of the circuit without introducing additional devices and not compromising on any changes in the functionality of the C-element.

Simulation is also done using the conditions that create a failure in the C-element due to different logic threshold voltages of devices. This is shown in Figure 26(a)(b).

We can see from the simulation results that the output-controlled implementation of C-element is of higher tolerance level compared to that of classical implementation. This increase of the internal capacitance is "free".

(a)
4.2 Other Possible Improvements

We have introduced the problem of the charge-sharing mechanism in Chapter 3. Under certain conditions, the occurrence of this mechanism will aggravate the problem of potential voltage changes at the internal node with the voltage changes at the output node with respect to time, causing a failure to the system. As discussed in section 3.2, note that it is a sequence of inputs to the C-element that causes the mechanism of charge-sharing to occur. Therefore, it would be helpful to have the ability to determine the possible sequence that will cause a potential charge-sharing problem in the circuit. We will use the state diagram.
4.2.1 State Diagram

A state diagram traces all the possible inputs to the Muller-C-element and determines the state of internal nodes (node 3, 4, 5 recaptured of the C-element as shown in Figure 27). From these state diagrams, we will be able to identify transitions of data flow that might cause a potential charge-sharing problem in the Muller-C-element. In the case of dynamic implementation of the C-element where there is no feedback as we have shown in Figure 15, this charge-sharing problem will cause a circuit failure to the circuit. The charges shared between node 4 and other nodes will be necessary to look at since it will either speed up or slow the voltage changes at node 4 compared to the output voltage with time. For example, if the output is charging from a logic “0” to a logic “1” state as shown in Figure 26(a)(b) and if the changes in voltage at node 4 is much faster than the change of voltage at the output node, the potential at node 4 will reach the decisive threshold voltage of the NMOS (M12 in Figure 27) much earlier than the output reaching a decisive potential that can turn M10 “on”, thus resulting in a failure to the system. The charges shared between node 3 and node 4 (from the sequence as simulated, there is a potential difference between this two nodes before M2 is turned “on” by switching X(input 1) low) will contribute to a faster voltage change at node 4.
However, it is the slow-rising edge at the output which is the dominating factor for the faster voltage change at node 4. This slow edge is also slow in shutting off M9 and thus there is a current flow from VDD through M7 charging up the potential at node 4.

Figure 28 shows the state diagram of the Muller-C-element.
Figure 28: State diagram of Muller-C-element
From the initial state, inputs to the Muller-C-element are changed. Note that the inputs change in the following manner: One of its inputs will maintain its original input state while the other changes its state. For example, if inputs X and Y are originally at logic "0" state, this will create two cases where one will be input X becoming logic "1" and input Y remaining unchanged. Therefore, the final state of XY will be "10". The other case will be that Y becomes "1" while X remains unchanged, causing the input of XY to be "01" ("x" denotes an unknown state). Having known the input to the circuit and by doing some circuit analysis, we would be able to determine the state of internal nodes 3, 4, and 5.

When there is a change in the input signals, it is first captured and placed in a dotted box, and if the change of the input signal can result in a stable state or can finally be determined, it will be placed in a solid box with the present known state. Each state can also be alias to a tag which is placed inside a circle. If the next state, due to the change in input transition, is the same as the one of the other states with a tag, it will be pointed to by the tag. This same analysis was done for all three different implementations of the Muller-C-element.

4.2.2 Analysis of State Diagram

From the state diagram, we are able to study and predict the outcome of the results by carefully determining the three states of the internal nodes which we have shown earlier are factors that might cause a potential charge-sharing problem in the Muller-C-element.
The results from each transition are dependent on the previous state of the Muller-C-element and therefore, we will be able to identify the thread of events that might lead to an unknown state in the device. Note that there is no unknown state shown on the output node, because we are not sure if the failure will occur. Figure 28 shows the state diagram for the original design of the Muller-C-element used in Ivan Sutherland's pipeline. We can see that there are two possible problem threads from the state diagram: these transitions are the logic states of Y and X which changes from “1 1” to “1 0” and the transition from ”0 0” to ”0 1” (shown by the thicker transition arrows and boxes). Taking the input logic state from “1 1” to “1 0”, initial inputs to the Muller-C-element is both at logic “1” which causes the potential at node 3, 5 and 4 as “17070” respectively. Now as the state of input X changes to a logic “0” state, referring back to the schematic of original Muller-C-element, we can see that there is a closed path between node 4 and node 3. Both nodes are at a different potential and therefore create a charge-sharing problem. Using the analysis method as studied in section 3.2, we can determine the potential shared among these two nodes. The parasitic capacitance at node 3 is much smaller than that at node 4 (approximately 2 C_{drain}). Therefore, we would expect the charge to be discharged quickly at node 3. The same analysis can be applied to the other problem thread, resulting in the same conclusion.

4.2.3 Timing Requirement and Logic Threshold Voltages

The reason for metastable state to occur, as discussed in Chapter 3, is that the data does not meet the setup and hold time for the latch. In asynchronous logic, using the C-
element as an example, there is no clock to synchronize the signals since it is event driven. The inputs to the C-element can come at different frequencies having different relationships between the signal edge. The setup time as defined in Chapter 3 will be the time relationship between the 2 input signal edges to the C-element. Sufficient time must be allocated to make sure that the C-element has enough internal information to make the decision. From Equation (3.8), we can determine the minimum time required for the C-element to drive the load (Cload), thereby avoiding a possible failure to the system.

There are also other factors to consider, especially in the asynchronous circuit. Some signals would have to travel a longer path relative to the other signals and hence would therefore have a longer delay. Wires are of different capacitance depending on the characteristic of the material the wire is made of. All these capacitances will also contribute to the delay of the signals which need to be taken into consideration while calculating the setup and hold time for the circuit in order to avoid failure. Because a larger load will also require a longer time to charge or discharge, the setup and hold time should also consider the load that the C-element is driving.

Some research has proposed the use of isochronic forks[6], which means that the time taken for two signals to be sent from point A to point B will be of the same delay. This will relax some constraints of the timing issue. However, the same paper also gave warnings on certain potential problems with the usage of isochronic forks. One of the problems is the different logic threshold voltages for different devices as we have discussed in this thesis.

We have seen the potential problem of this property in different devices and have shown realistic conditions for which such property of the devices can create hazardous
conditions to the system. Making sure that the logic threshold voltages are the same for
devices that need to communicate with each other is one solution to relax this problem
(uniform logic thresholds voltages). Again, there are also other problems relating to the
usage of uniform logic thresholds voltages for different devices.

4.2.4 Device Sizing and Layout

Many examples that were shown in this thesis deal with a large capacitive load. This
large load is common in the application of the C-element. In the example of the
micropipeline, the C-element may be driving a large number of registers and thus the load
will be big. Knowing the size of the load for which the device must drive, we can size the
transistors for the output driver. A larger size transistor (large W) implies that a larger
current can flow through the device, resulting in a faster charging and discharging process
for the output driver.

In the case of the C-element, we can also size the transistors properly so that the
time required to charge and discharge the load will be small. The fast rise and fall time for
the output will relax the chances for which data can switch halfway through the process
before the feedback can occur to turn on the appropriate transistor in the internal CMOS
memory logic, driving the output to a correct state. This fast switching time will also tighten
the probability for the voltage at the internal node to change much faster than the change in
the potential voltage at the output. Therefore the output will reach the decision threshold
voltage first before the voltage on the internal node can drive it in the opposite direction.
Faster switching, as mentioned in section 4.2.2 would also relax the setup and hold time requirement. However this constraint of setup and hold time must always be met in order to avoid any failure in the circuit. Knowing the relationship between the capacitance on the internal node to that of the load, proper layout is necessary to make sure that there is no extra parasitic capacitance to be added to the output node while making sure that the internal node has enough capacitance in order to have a higher tolerance level. However, too large a capacitance at the internal node will also cause problems to the circuit, resulting in a circuit failure.
CHAPTER 5

CONCLUSION

5.1 Muller-C-element In Self-Timed Circuits

As discussed and introduced in Chapter 1, self-timed circuits use events to drive the next transition in a circuit as there is no global clock to synchronize the data flow and the execution of an event in the system. Therefore, the synchronization in asynchronous logic (self-timed) would have to be very dependent on events that govern the data flow in the system.

The Muller-C-element changes its output state only when both the inputs are at the same state (non-inverted Muller-C-element), and the input to the Muller-C-element is the control of an event. Hence it is obvious that the Muller-C-element is very important to asynchronous logic. The Muller-C-element is the key device for the Ivan Sutherland's micropipeline and also in many data flow processors.

The Muller-C-element is also a storage element in asynchronous circuits in that the previous state remains unchanged if and only if the condition for the input events is met. If these conditions are not true, the Muller-C-element will remain at its previous state that is stored in the internal static memory logic of the element.
5.2 Findings And Results Summary

We have looked at several failures that might occur in the C-element, namely the problem of charge-sharing, metastability, and the different interpretation of logic states due to the difference in logic threshold voltages for different devices. These failures are real and will cause the system to break or malfunction. The differences in threshold voltages between devices will create a realistic situation in the micropipeline that will cause a failure in the system. Like synchronous latches, the C-element also exhibits the potential for a metastable state to occur. Again, we can relate these conditions using the micropipeline as an example. This scenario is described in section 3.4. The problem of charge-sharing is also discussed and the simulation of the dynamic implementation of the C-element shows that such a mechanism can create a failure in the system.

From the analysis, we can see the relationship between the change of potential voltage at the internal node compared with the external load with time. Using Hspice simulations, we simulated several situations that showed the correctness of this analysis. If the change of potential voltage is larger than the changes of potential at the output node at a certain time \( t \) within the range of unstable bias voltage of the circuit, the circuit might switch and cause a failure. If within the range of unstable bias voltage where changes of potential voltages are approximately equal, the device is in a metastable state since there is no potential voltage at that moment which is high or low enough to cause a switch in the output state (insufficient internal information). Obviously, if the output feedback voltage is larger than the internal node, it will switch the output to the correct state (as in the case
shown in the simulations). The change of potential voltages at any node is directly related to its capacitance at that node. With a larger capacitance, it will require a longer time for the capacitance to charge or discharge. By rearranging the transistors in the classical implementation of the Muller-C-element, we have the output-controlled C-element. Node 4 (node 3 in the simulation) has an additional two drain capacitance as it sees an additional inverter. This slight increase in the capacitance at this node makes this implementation of Muller-C-element have a higher tolerance level. Simulation results have shown that under actual conditions that cause a failure using the classical implementation of C-element didn’t cause a failure in the output-controlled C-element. The additional capacitance at this node can be classified as “free” since there are no additional devices added onto the circuits.

The changes in the potential voltage at the internal node also dependent on the supply of charges that is sourcing this node. We have shown that this mechanism of charge-sharing does occur in the C-element. Despite this, the effect of the charge-sharing problem is not as significant as the slow rising or falling edge of the output signal due to a high load which is slow to switch the transistor “on” in the circuit. Therefore, having a current charging or discharging the internal node, the charge-shared among the nodes does aggravate this situation. From the composition of the C-element, we can see that there is a sequence of events that is needed in order to create a charge-sharing problem in the circuit. Hence, we use the state diagram that can help to identify such transitions. Metastability has always been ways a problem for designers. There are many factors that govern the timing in which a signal is sent from one module to another. This problem of different frequencies within the system is an even more prominent problem in the asynchronous world since
signals can come at any instance of time. Making sure that there is a proper timing relationship between the two signals in the C-element will relax the constraints of metastability occurring. However, as mentioned earlier, there are many other factors that govern the timing issue in a system. Wire capacitance, long path, module delays, a larger load, and other factors all need to be taken into consideration while calculating the timing requirements so as to avoid the occurrence of a metastable state. The best circuit design strategy would not be to solve, but to get around metastability problems.

5.3 Future Work

The purpose of this thesis is to analyze the possible failures that could happen to the C-element - in particular, the problem of different logic threshold voltages, charge-sharing, and metastability.

Future work would include continuing the study of possible failures that might occur in the C-element. Also, efforts should be put into designing a better C-element. As shown in this thesis and many other works, the C-element is an important control block in many asynchronous designs.

A detailed study of metastability characteristics of the C-element would be an interesting topic to do for further research.
BIBLIOGRAPHY


APPENDICES
Appendix A Hspice netlist for Ivan’s Sutherland Muller-C-element

* C-element CMOS circuit (METASTABILITY)

* .option post

.MODEL NTRAN NMOS LEVEL=2 LD=0.458388U TOX=280.000000E-10
+ NSUB=3.496105E+16 VTO=0.930261 KP=4.464000E-05 GAMMA=0.8759
+ PHI=0.6 UO=362.246 UEXP=9.499054E-02 UCRIT=130365
+ DELTA=1.000000E-06 VMAX=100000 XJ=0.250000U LAMBDA=1.446553E-02
+ NFS=1.245818E+12 NEFF=1 NSS=1.000000E+12 TPG=1.000000
+ RSH=20.580001 CGDO=5.652906E-10 CGSO=5.652906E-10 CGBO=4.291585E-10
+ CJ=4.015000E-04 MJ=0.446500 CJSW=5.023000E-10 MJSW=0.270500 PB=0.750000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is 0.35 um

.MODEL PTRAN PMOS LEVEL=2 LD=0.355084U TOX=280.000000E-10
+ NSUB=1.443000E+16 VTO=-0.712799 KP=2.528866E-05 GAMMA=0.563231
+ PHI=0.6 UO=205.063 UEXP=0.357053 UCRIT=60449.2
+ DELTA=1.000000E-06 VMAX=23204.3 XJ=0.250000U LAMBDA=6.567991E-02
+ NFS=8.419363E+11 NEFF=1 NSS=1.000000E+12 TPG=-1.000000
+ RSH=77.339999 CGDO=4.378947E-10 CGSO=4.378947E-10 CGBO=4.687447E-10
+ CJ=2.156000E-04 MJ=0.396400 CJSW=2.663000E-10 MJSW=0.083900 PB=0.530000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is 0.38 um

Vdd 9 0 5

***********
*changing constant

.param V1stimes = 2n
.param V1rtimes = 'V1stimes+1n'
.param V1data_valid = 'V1rtimes+4n'
.param V1ftimes = 'V1data_valid+1n'
.param V2stimes = 3n
.param V2rtimes = 'V2stimes+0.5n'
.param V2data_valid = 'V2rtimes+0.955n'
.param V2ftimes = 'V2data_valid+0.5n'
.param WvalP=8u
.param WvalN=4u
.param Lval=2u
Ctest 4 0 50f

*Inputs

Vx 1 0 pw1(0 0 V1stimes 0 V1rtimes 5 V1data_valid 5 V1ftimes 0 12n 0)
Vy 2 0 pw1(0 0 V2stimes 0 V2rtimes 5 V2data_valid 5 V2ftimes 0 12n 0) ac 1
*circuit
MP1 3 2 9 9 PTRAN w=WvalP l=Lval
+ AD='WvalP*7u' AS='WvalP*7u'
+ PD='WvalP*2+14u' PS='WvalP*2+14u'

MP2 4 1 3 9 PTRAN w=WvalP l=Lval
+ AD='WvalP*7u' AS='WvalP*7u'
+ PD='WvalP*2+14u' PS='WvalP*2+14u'

MP3 6 2 9 9 PTRAN w=WvalP l=Lval
+ AD='WvalP*7u' AS='WvalP*7u'
+ PD='WvalP*2+14u' PS='WvalP*2+14u'

MP4 6 1 9 9 PTRAN w=WvalP l=Lval
+ AD='WvalP*7u' AS='WvalP*7u'
+ PD='WvalP*2+14u' PS='WvalP*2+14u'

MP5 4 8 6 9 PTRAN w=WvalP l=Lval
+ AD='WvalP*7u' AS='WvalP*7u'
+ PD='WvalP*2+14u' PS='WvalP*2+14u'

MP6 8 4 9 9 PTRAN w=WvalP l=Lval
+ AD='WvalP*7u' AS='WvalP*7u'
+ PD='WvalP*2+14u' PS='WvalP*2+14u'

MN1 4 1 5 0 NTRAN w=WvalN l=Lval
+ AD='WvalN*7u' AS='WvalN*7u'
+ PD='WvalN*2+14u' PS='WvalN*2+14u'

MN2 5 2 0 0 NTRAN w=WvalN l=Lval
+ AD='WvalN*7u' AS='WvalN*7u'
+ PD='WvalN*2+14u' PS='WvalN*2+14u'

MN3 4 8 7 0 NTRAN w=WvalN l=Lval
+ AD='WvalN*7u' AS='WvalN*7u'
+ PD='WvalN*2+14u' PS='WvalN*2+14u'

MN4 7 1 0 0 NTRAN w=WvalN l=Lval
+ AD='WvalN*7u' AS='WvalN*7u'
+ PD='WvalN*2+14u' PS='WvalN*2+14u'

MN5 7 2 0 0 NTRAN w=WvalN l=Lval
+ AD='WvalN*7u' AS='WvalN*7u'
+ PD='WvalN*2+14u' PS='WvalN*2+14u'

MN6 8 4 0 0 NTRAN w=WvalN l=Lval
+ AD='WvalN*7u' AS='WvalN*7u'
+ PD='WvalN*2+14u' PS='WvalN*2+14u'

.plot i(5)
.op
.tran 0.1ns 13ns
.ac dec 10 100 100Meg
*
.end
Appendix B Hspice Output file for Ivan Sutherland’s Muller-C-element

Using: /tools/meta/h93a.02/pa/hspice

******* H S P I C E -- H93A.02 10:47:26 96/04/19 pa
Copyright 1993 (C) by Meta-Software, Inc.

This computer program is protected by copyright law and international treaties. Any dissemination or use of this program, other than that permitted by Meta-Software, Inc., is unlawful and may result in prosecution under the law.

Input File: /tmp_mnt/home/chewo/misc/research/ce_met.sp

lic: PowerView license request for hspice
lic: PowerView license DENIED for hspice
lic: USER: chewo HOSTNAME: rover HOSTID: 2015658579
lic: contacting server: rolls
lic: Server permit path: /tools/meta/h93a.02/permit.hsp
lic: Site: rockwell_newport Created: 960109 Order#: 59130
lic: token number 116560 allocated_SJF

Init: read install configuration file: /tools/meta/h93a.02/meta.cfg

*******
* c-element cmos circuit (metastability)

1

1 ****** H S P I C E -- H93A.02 10:47:26 96/04/19 pa
******

* c-element cmos circuit (metastability)

******* mos model parameters  tnom= 25.000 temp= 25.000

*******

***************************************************************************
*** common model parameters model name: 0:ntran model type:nmos ***
***************************************************************************

names values units names values units names values units
----- ------ ----- ------ ------ ----- ------ ----- ------ ------

1*** geometry parameters ***

1d= 458.39n meters lmlt= 1.00 wd= 0. meters
wmlt= 1.00 xl= 0. meters xw= 0. meters
lref= 0. meters wref= 0. meters lref= 0. meters
wref= 0. meters xref= 0. meters xwref= 0. meters
lmin= 0. meters wmin= 0. meters lmax= 0. meters
wmax= 0. meters

2*** threshold voltage parameters ***

vto= 930.26m volts nss= 1.0e+12 1/cm**2 tpg= 1.00
phi= 600.00m volts gamma= 875.90m v**0.5 bulk= gnd
ngate= 0. cm**3 nsub= 3.5e+16 1/cm**3 delvto= 0. volts
3*** gate overlap capacitance parameters ***
cgbo= 429.16p f/meter  cgdo= 565.29p f/meter  cgso= 565.29p f/meter
meto= 0. meters

4*** gate capacitance parameters ***
capop= 2.00  cf1= 0. volts  cf2= 100.00m volts
cf3= 1.00 volts  cf4= 50.00  cf5= 666.67m
cf6= 500.00  xqc= 500.00m  tox= 28.00n meters
cox= 1.23m f/m**2

5*** diffusion parasitic parameters ***
acm= 0.  is= 10.00f amps  js= 0. a/m**2
jsw= 0. amp/m  nds= 1.00  cbd= 0. farad
cbs= 0. farad  cj= 401.50u f/m**2  cjsw= 502.30p f/m

cjgate= 502.30p f/m  mj= 446.50m  mjsw= 270.50m
pb= 750.00m volts  php= 750.00m volts  tt= 0. secs
hsd= 0. meters  lsd= 0. meters  rd= 0. ohms
rs= 0. ohms  rsh= 20.58 ohms/sq  fc= 0.
alpha= 0.  vcr= 0. volts  iirat= 0.
rdc= 0. ohms  rsc= 0. ohms  n= 1.00
vnds= -1.00 volts

6*** temperature effect parameters ***
tlev= 0.  tlevc= 0.  eg= 1.11 ev
gapl= 702.00u ev/deg  gap2= 1.11k deg  xti= 0.
bex= -1.50  tcv= 0. v/deg k  trd= 0. /deg
trs= 0. /deg  cta= 0. /deg  ctp= 0. /deg

7*** noise parameters ***
kf= 0.  af= 1.00  nlev= 2.00
gdsnoi= 1.00

*** level 2 model parameters ***
delta= 1.00u  ecrit= 0. v/m  lambda= 14.47m /v
nfs= 1.2e+12 1/cm**2  ucrit= 130.37k v/cm  uexp= 94.99m
utra= 0.  uo= 362.25 cm**2/vs  xj= 250.00n meters
neff= 1.00  vmax= 100.00k m/sec  kp= 44.64u a/v**2
mob=1.2e-29  deriv= 0.

***************************************************************************
*** common model parameters model name: 0:ptran  model type:pmos ***
***************************************************************************

1*** geometry parameters ***
ld= 355.08n meters  lmlt= 1.00  wd= 0. meters
wmlt= 1.00  xl= 0. meters  xw= 0. meters
lref= 0. meters  wref= 0. meters  lref= 0. meters
wref= 0. meters  xref= 0. meters  xwref= 0. meters
lmin= 0. meters  wmin= 0. meters  lmax= 0. meters
wmax= 0. meters

2*** threshold voltage parameters ***

to= -712.80m volts  nss= 1.0e+12 1/cm**2  tpg= -1.00
phi= 600.00m volts  gamma= 563.23m v**0.5  bulk= gnd
ngate= 0. cm**3  nsub= 1.4e+16 1/cm**3  delvto= 0. volts

3*** gate overlap capacitance parameters ***
cgbo= 468.74p f/meter  cgdo= 437.89p f/meter  cgso= 437.89p f/meter
meto= 0. meters

4*** gate capacitance parameters ***
capop= 2.00  cf1= 0. volts  cf2= 100.00m volts
cf3= 1.00 volts  cf4= 50.00  cf5= 666.67m
cf6= 500.00  xc= 500.00m  tox= 28.00n meters
cox= 1.23m f/m**2

5*** diffusion parasitic parameters ***
acm= 0.  is= 10.00f amps  js= 0. a/m**2
jsw= 0. amper/m  nds= 1.00  cbd= 0. farad
cks= 0. farad  cj= 215.60u f/m**2  cjsw= 266.30p f/m
ckgate= 266.30p f/m  mj= 396.40m  mjsw= 83.90m
pb= 530.00m volts  php= 530.00m volts  tt= 0. secs
hdif= 0. meters  ldif= 0. meters  rd= 0. ohms
rs= 0. ohms  rsh= 77.34 ohms/sq  fc= 0.
alpha= 0.  vcr= 0. volts  ii= 0.
rdr= 0. ohms  rsc= 0. ohms  n= 1.00
vnds= -1.00 volts

6*** temperature effect parameters ***
tlev= 0.  tlevc= 0.  eg= 1.11 ev
gap1= 702.00u ev/deg  gap2= 1.11k deg  xti= 0.
bex= -1.50  tcv= 0. v/deg k  trd= 0. /deg
trs= 0. /deg  cta= 0. /deg  ctp= 0. /deg

7*** noise parameters ***
kf= 0.  af= 1.00  nlev= 2.00
gd= 1.00

*** level 2 model parameters ***
delta= 1.00u  ecrit= 0. v/m  lambda= 65.68m /v
nfs= 841.94g 1/cm**2  ucrit= 60.45k v/cm  uexp= 357.05m
utra= 0.  uo= 205.06 cm**2/vs  xj= 250.00n meters
neff= 1.00  vmax= 23.20k m/sec  kp= 25.29u a/v**2
mob=-1.2e-29 deriv= 0.

**warning*: element identification problem
with output 0:5 ignored

1 ***** H S P I C E -- H93A.02 10:47:26 96/04/19 pa

******
* c-element cmos circuit (metastability)
****** operating point information tnom= 25.000 temp= 25.000
******
****** operating point status is all simulation time is 0.
node =voltage node =voltage node =voltage

+0:1 = 0. 0:2 = 0. 0:3 = 5.0000
+0:4 = 5.0000 0:5 =530.8948m 0:6 = 5.0000
+0:7 = 99.5297m 0:8 = 24.2343n 0:9 = 5.0000

**** voltage sources

subckt
element 0:vdd 0:vx 0:vy
volts 5.0000 0. 0.
current -34.7823p 0. 0.
power 173.9116p 0. 0.

total voltage source power dissipation= 173.9116p watts

**** mosfets

subckt
element 0:mp1 0:mp2 0:mp3 0:mp4 0:mp5 0:mp6
model 0:ptran 0:ptran 0:ptran 0:ptran 0:ptran 0:ptran
ibs 0. 2.715e-22 0. 0. 1.810e-22 0.
ibd 2.715e-22 5.430e-22 1.810e-22 1.810e-22 5.430e-22 50.0000f
vgs -5.0000 -5.0000 -5.0000 -5.0000 -5.0000 -54.3029n
vbs 0. 27.1515n 0. 0. 18.1010n 0.
vth  -654.4781m  -654.4781m  -654.4781m  -654.4781m  -654.4781m  -616.7462m
vdsat  -1.9674  -1.9674  -1.9674  -1.9674  -1.9674  -46.5456m
beta  73.2427u  73.2427u  73.2427u  73.2427u  73.2427u  233.5456u
gam eff  487.9390m  487.9390m  487.9390m  487.9390m  487.9390m  439.2274m
gm  1.2685p  1.2685p  1.2685p  1.2685p  1.2685p  77.9203p
gds  318.2780u  318.2780u  318.2780u  318.2780u  318.2780u  1.4662p
gmb  344.0125f  344.0125f  344.0125f  344.0125f  344.0125f  13.3053p
cdtot  29.9286f  29.9286f  29.9286f  29.9286f  29.9286f  1.48336f
cgtot  20.3522f  20.3522f  20.3522f  20.3522f  20.3522f  13.5268f
cstot  29.9286f  29.9286f  29.9286f  29.9286f  29.9286f  23.8322f
cbtot  40.7453f  40.7453f  40.7453f  40.7453f  40.7453f  37.6418f

subckt
element 0:mn1  0:mn2  0:mn3  0:mn4  0:mn5  0:mn6
model 0:ntran  0:ntran  0:ntran  0:ntran  0:ntran  0:ntran
id  2.001e-16  2.8660p  694.2275f  2.5470p  2.5470p  14.6183p
ibs  -5.3089f  0.  -9.953e-16  0.  0.  0.
ibd  -50.0000f  -5.3089f  -9.953e-16  0.  0.  0.
vgs  4.4691  530.8948m  4.9005  99.5297m  99.5297m  24.2343n
vbs  -530.8948m  0.  -99.5297m  0.  0.  0.
vth  992.1501m  843.8176m  843.4924m  852.5397m  852.5397m  855.0265m
vdsat  56.0332m  52.6829m  54.1399m  52.4473m  52.4473m  1.9122
beta  176.2345u  166.1170u  177.4180u  165.0789u  165.0789u  145.5278u
gam eff  696.1949m  764.3020m  707.4551m  775.5622m  775.5622m  778.7726m
gm  2.7233f  36.9975p  9.1411p  32.8677p  32.8677p  3.1853p
gds  7.372e-17  2.1194p  197.8106f  2.5584p  2.5584p  603.2087u
cdtot  13.1586f  20.6746f  13.1589f  23.6580f  23.6580f  27.2256f
cgtot  8.2184f  8.2544f  8.2465f  8.3305f  8.3305f  10.3401f
cstot  20.7207f  24.6362f  23.6531f  24.6335f  24.6335f  27.2256f
cbtot  32.9599f  44.3554f  35.8644f  47.2600f  47.2600f  45.0593f
cgs  2.3073f  2.3436f  2.3351f  2.3409f  2.3409f  4.9330f
cgd  2.2616f  2.2612f  2.2619f  2.3400f  2.3400f  4.9330f

Opening plot unit= 15
file=../ce_met.ac0

Opening plot unit= 15
file=../ce_met.tr0
*****
* c-element cmos circuit (metastability)
***** transient analysis tnom= 25.000 temp= 25.000
*****

***** job concluded
1 ***** H S P I C E -- H93A.02 10:47:26 96/04/19 pa
*****
* c-element cmos circuit (metastability)
***** job statistics summary tnom= 25.000 temp= 25.000
*****

total memory used 183 kbytes

# nodes = 10 # elements= 16
# diodes= 0 # bjt s = 0 # jfets = 0 # mosfets = 12

analysis time # points tot. iter conv.iter
op point .11 1 42
ac analysis .05 61 61
transient .49 131 184 70 rev= 0
readin .14
errchk .10
setup .00
output .00
total cpu time .96 seconds
 job started at 10:47:26 96/04/19
 job ended at 10:47:30 96/04/19

lic: Releasing license for hspice
HSPICE job /tmp_mnt/home/chewo/misc/research/ce_met.sp completed.
Fri Apr 19 10:47:32 PDT 1996
Appendix C Hspice netlist for Output-controlled Muller-C-element

* C-element CMOS circuit
*
 options post
 .MODEL NMOS NMOS LEVEL=2 LD=0.458388U TOX=280.000000E-10
  + NSUB=3.496105E+16 VTO=0.930261 KP=4.464000E-05 GAMMA=0.8759
  + PHI=0.6 UO=362.246 UEXP=9.499054E-02 UCRIT=130365
  + DELTA=1.000000E-06 VMAX=100000 XJ=0.250000U LAMBDA=1.446553E-02
  + NFS=1.245818E+12 NEFF=1 NSS=1.000000E+12 TPG=1.000000
  + RSH=20.580001 CGDO=5.652906E-10 CGSO=5.652906E-10 CGBO=4.291585E-10
  + CJ=4.015000E-04 MJ=0.446500 CJSW=5.023000E-10 MJSW=0.270500 PB=0.750000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is 0.35 um
 .MODEL PMOS PMOS LEVEL=2 LD=0.355084U TOX=280.000000E-10
  + NSUB=1.443000E+16 VTO=-0.712799 KP=2.528866E-05 GAMMA=0.563231
  + PHI=0.6 UO=205.063 UEXP=0.357053 UCRIT=60449.2
  + DELTA=1.000000E-06 VMAX=23204.3 XJ=0.250000U LAMBDA=6.567991E-02
  + NFS=8.419363E+11 NEFF=1.001 NSS=1.000000E+12 TPG=-1.000000
  + RSH=77.339999 CGDO=4.378947E-10 CGSO=4.378947E-10 CGBO=4.687447E-10
  + CJ=2.156000E-04 MJ=0.396400 CJSW=2.663000E-10 MJSW=0.083900 PB=0.530000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is 0.38 um
 Vdd 1 0 5v
*Inputs

*Vx 8 0 5
*Vy 9 0 5
 Vx 8 0 pwl(0n5 5n5 2n5 10n 5 11n 0 29n 0 30n 5 45n 5 46n 0)
 Vy 9 0 pwl(0n5 5n5 2n5 10n 5 19n 5 19.1n 0 19.2n 0 19.3n 5)
.tran 0.1ns 60ns

M1 1 8 2 1 PMOS W=8u L=2u
M2 2 9 3 1 PMOS W=8u L=2u
M3 1 7 5 1 PMOS W=4u L=2u
M4 5 8 3 1 PMOS W=8u L=2u
M5 5 9 3 1 PMOS W=8u L=2u
M6 1 3 7 1 PMOS W=12u L=2u
M7 0 9 4 0 NMOS W=4u L=2u
M8 4 8 3 0 NMOS W=4u L=2u
M9 0 7 6 0 NMOS W=8u L=2u
M10 6 8 3 0 NMOS W=4u L=2u
M11 6 9 3 0 NMOS W=4u L=2u
M12 0 3 7 0 NMOS W=12u L=2u
.end