# ASYNCHRONOUS RESEARCH CENTER

## Portland State University

**Subject:** **Fourth Class Handout – Proper Stopper**
**Date:** **October 15, 2010**
**From:** **Ivan Sutherland**
**ARC#:** **2010-is49**

**References:**

T.J.Chaney & C.E. Molnar, Anomalous Behavior of Synchronizer and Arbiter Circuits, IEEE, April 1973.
C.Seitz: System Timing. p261 of Chapter 7 in *Introduction to VLSI Systems*, Mead & Conway, Addison-Wesley, 1980: ISBN 0-201-04358-0.

## PURPOSE

This memo shows how to stop the asynchronous flow in a series of GasP elements. A special asynchronous element called a “mutual exclusion element” or “mutex” can interrupt an asynchronous flow without risk of producing invalid pulses and thus damaging data. The simulations exhibit improper and proper stopping devices.

## INTRODUCTION

When another pedestrian approaches you on the sidewalk, have you ever had difficulty in deciding on which side to pass? Have you ever started to speak at the same time as someone else and had to restart, perhaps repeatedly, to choose who will speak first? Have you ever approached an ATM machine at the same time as another customer and been confused about who should use it first? Have you ever seen a “dead heat” in a race where two runners cross the finish line at the same time?

These are examples of a fundamental problem of continuous time. Because time is continuous the interval between two events may be arbitrarily short. The theory of relativity tells us that two observers with different points of view may not even agree on which of two events happened first. Signals that arrive at very nearly the same time can cause problems. We can debate the meaning of “simultaneous.”

Today’s lesson concerns three things. First, we will explore the fundamental nature of time uncertainty. Second, we will study equipment that can assign sequence to events regardless of how close together in time they occur. Third, we will emphasize that assigning sequence may take arbitrarily long. The simulation assignment will

---

suggest how improper circuit design can result in unreliable operation. It will also show how tiny differences in time can make major differences to behavior.

## SIMULTANEOUS EVENTS?

I once attended an outdoor concert of a band consisting of many hundreds of very capable high school musicians. The musicians stood all over a football field, from goalpost to goalpost, with the conductor on a raised platform near the center of the field.

The music sounded terrible. Although the musicians were young, each was well skilled with his instrument. Moreover, each followed the conductor with meticulous precision. The problem with the music was physics, not musicianship.

A football field is about 100 yards long, slightly longer from goalpost to goalpost. Let's call that 100 meters. Because sound travels in air at about 343 meters per second, the ends of a football field are separated by about 1/3 of a second in sound time. Brass bands play march music at about 120 beats per minute, and one third of a second is nearly a full beat of such music. The music sounded bad because sound travels so slowly.

Consider the plight of two bass drummers, one at each end of the field. Each hits his drum precisely when he sees the conductor's downbeat. Their drumbeats are, in some sense, simultaneous. But because it takes 1/3 of a second for sound to get to the other drummer, each drummer thinks that the other is 1/3 second behind the beat.

I was sitting near the conductor, so I thought both drummers were only 1/6 second behind the conductor's beat. That's bad enough, but people at the ends of the field had a worse experience. The speed of sound limits the physical size of bands.

The sequence of events observed from different places may differ. The experience of the two bass drummers can occur at any scale. Relativity teaches us that it takes time for information to travel over distance, and it inevitably follows that the observed sequence may depend on the point of view. Indeed, it is meaningless to speak of "simultaneous" events or to assign an "absolute sequence" to events that are separated in space. One can be safe in describing only the sequence of events seen by a single observer.

## METASTABILITY

Flip flops have been known for nearly 100 years. Their most important property is that they are bi-stable: they assume one of two distinct states. Digital engineers acknowledge only the two stable states and avoid all other states.

Light switches are also either on or off. As a child I took delight in positioning light switches in my home half way between on and off. My childhood home had toggle switches with little control levers that stuck out from the wall. Although the lever was usually either “up” or “down”, I liked to position the lever so it stuck straight out from the wall, neither up nor down. If I were very careful, the lever would stay in that middle position, sometimes causing a wonderful hissing sound from the arc inside the switch.

The middle state of the switch was meta-stable. Given the slightest push in either direction, the spring inside the switch would flip the lever to one of its two stable states.

In August of last year I used the Rabbit Ears Pass to cross the Rocky Mountains in Colorado. At the top of the pass I photographed a large sign identifying the continental divide. See photographs. Rain falling East of the continental divide flows ultimately into the Atlantic Ocean via the Gulf of Mexico. Rain falling West of the continental divide flows ultimately into the Pacific Ocean. The continental divide is the high ground that divides the two watersheds.

Figure 1: Rabbit Ears Pass Sign

We have only to know that there is a Pacific Ocean and a Gulf of Mexico and mountains in between to know that there is a continental divide. One can move the continental divide with a shovel, but it is impossible to get rid of the divide. The divide is inherent in the mountain range between the two Oceans.

Figure 2. Views looking West and East from the top of the pass.

Likewise, every flip-flop has at least one meta-stable state between its two stable states. Like the continental divide, it is impossible to build a flip-flop without such a meta-stable state. But flip-flops get into their meta-stable state only rarely, and so not much was known about meta-stability when Chaney and Molnar published their classic 1973 paper. See reference.

Tom Chaney and Charlie Molnar worked on asynchronous systems at Washington University in Saint Louis. Charlie has now been dead for over a decade but Tom still lives and works in Saint Louis. Charlie once told me about one reviewer's comments in response to their paper. The reviewer said, in effect, "This is a really important phenomenon. If it were true I'd know about it." Indeed, meta-stability is an important part of flip-flop behavior, but until Chaney and Molnar published their paper, it was poorly understood. Indeed, for some digital designers states between the two stable states of a flip-flop were unthinkable.

The problem, of course, is that meta-stability concerns the analog behavior of the flip-flop, usually thought of as a strictly digital circuit. Thought of as an analog circuit, the flip-flop is a pair of cross-connected inverting amplifiers. Any such inverting pair of amplifiers obviously has a meta-stable operating point, namely the operating point at which each amplifier delivers exactly the same output as its input.

Of course, any deviation away from the meta-stable point inevitably leads, by positive feedback, to one of the two stable operating points. How long that takes depends on the gain and time response of the amplifiers and how near or far the amplifiers start out from their meta-stable point. The rate of departure is approximately proportional to how far the circuit is from the meta-stable point. Thus departure may be very slow at first, but will increase in speed as the distance from the meta-stable point increases. The further away from meta-stability, the faster the departure.

But how can we be sure of departure? How do we know that the meta-stable operating point is unstable? What about noise?

If we believe that amplifiers are linear near their meta-stable point we can be sure that any departure from the meta-stable point will accelerate. Moreover, we can be sure that local thermal noise will cause some initial departure from meta-stability, but that same local noise might cause a return to meta-stability. Thus the best we can say is that the exit from meta-stability is a statistical phenomenon whose properties we can measure. Moreover, once a circuit is "far enough" from meta-stability we can be sure that small local noise cannot drive it into meta-stablity again. Measurements confirm that exit from meta-stability can best be described as a random phenomenon.

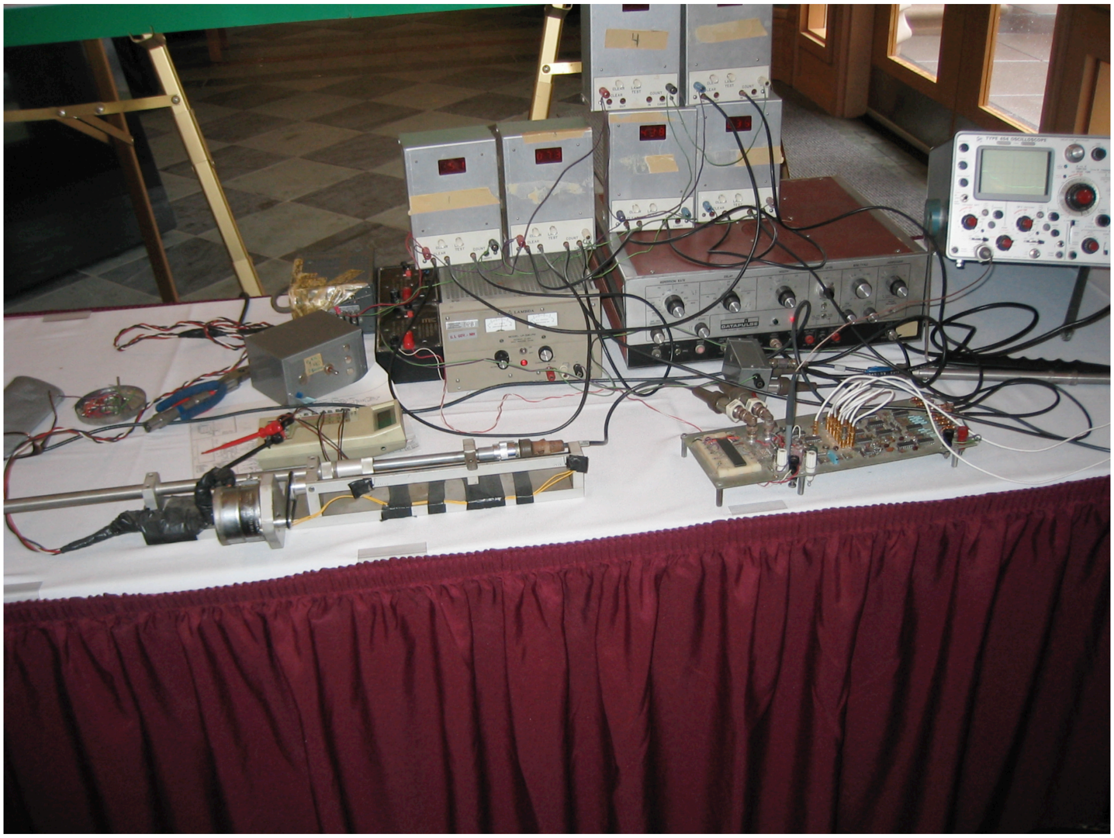


Figure 3. Chaney's meta-stability demonstration version of 2004.
The silver tube in foreground is the "trombone" with drive motor and belt.

Chaney did a convincing experiment to exhibit meta-stability, see photos. In the equipment set-up picture you can see a long metal tube called the "trombone." It is just a piece of coaxial cable with adjustable length. Chaney applied the same signal to both sides of a flip-flop, delaying one signal through the trombone and the other through a similar but fixed length of coaxial cable. Careful adjustment of the length of the trombone can drive the flip-flop into meta-stability. A change in length of a few millimeters departs from the meta-stable region. This shows how very close together in time the inputs must be to drive a flip-flop into meta-stability. Meta-stability is rare.

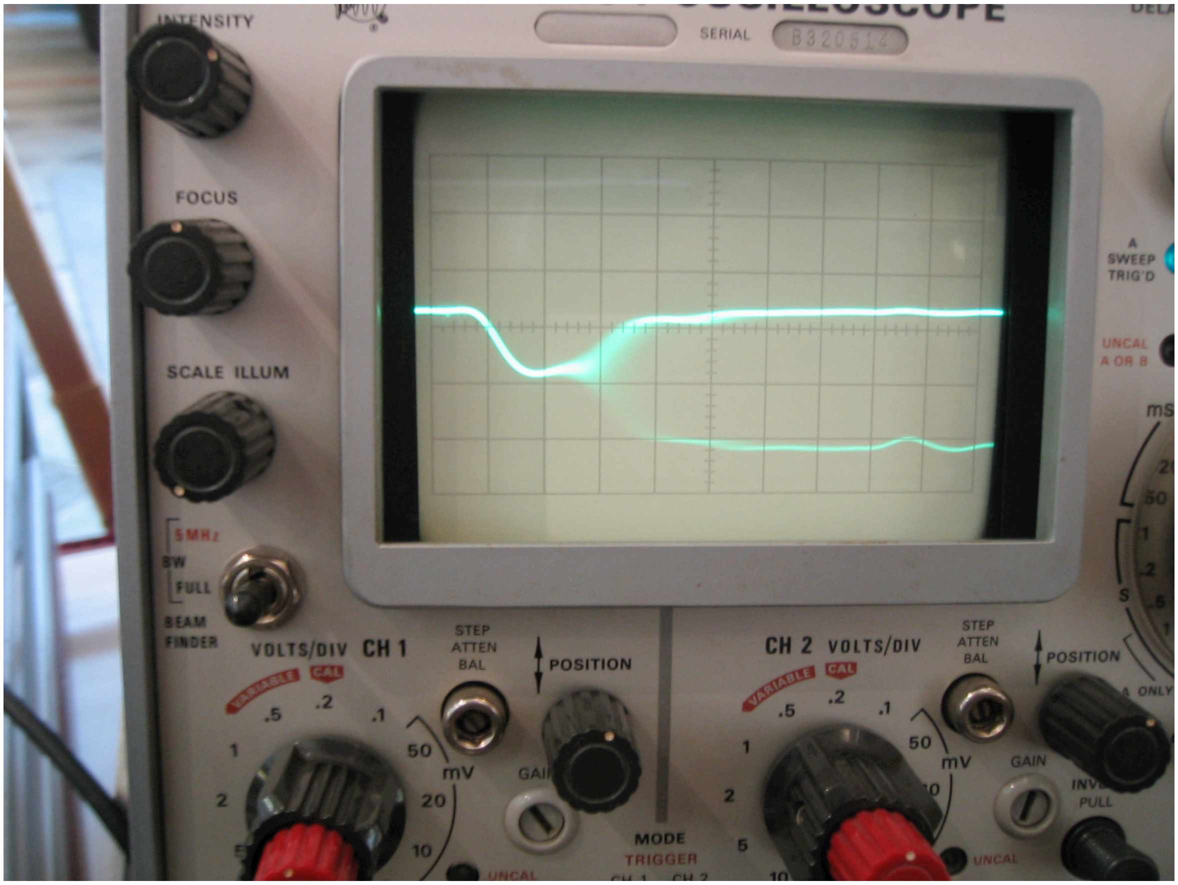


Figure 4. Chaney's meta-stability demonstration: version of 2004.
Sometimes the flip-flop settles HI sometimes LO. Some delays are quite long.

In the oscilloscope photo of Figure 4 you can see that sometimes the flip-flop exits to the upper stable state and sometimes to the lower one. The flat place before the decision shows how long the flip-flop takes to decide. Occasional single traces that are too faint to appear in the photograph report even longer periods of meta-stability. In this week's simulation exercises you will see both how carefully one must adjust the input time to cause meta-stability and that meta-stability can cause delay.

## MUTUAL EXCLUSION – see arbiter2exp

A flip-flop forms the basis for a simple mutual exclusion element. The bottom six transistors of the figure called arbiter2exp form a flip-flop of two cross-coupled NAND gates. As long as both its inputs `req[A,B]` are LO, both its outputs `AA` and `BB` remain HI. However, if either input goes HI, the output of the corresponding NAND gate will go LO **provided** that the other output doesn't also go LO. The NAND gates prevent both outputs from being LO together. The two signals `AA` and `BB` are "mutually exclusive."

However, if both inputs `req[A,B]` become HI at “nearly the same time” both `AA` and `BB` may become partially LO together. Moreover, in rare cases the flip-flop may stay in that “meta-stable” state for a surprisingly long time. How long? The duration of meta-stability depends on the fine details of the two conflicting input signals `req[A,B]`.

We can use such a “mutual exclusion element” or MUTEX to choose a sequence for events. The two request signals, `req[A,B]`, tell when two events arrive. The two grant signals `grant[A,B]` put them in sequence. Notice I didn’t say that the MUTEX will pick the one that arrived “first” because the word “first” implies an absolute ordering in time.

What the MUTEX can do is to preserve the sequence of the events if they are separated widely enough in time to have an obvious sequence. If they arrive closely in time their actual sequence probably doesn’t matter. Indeed, if they arrive closely together in time two observers might argue of which arrived first, and so the ultimate result must not depend on their sequence. Both will eventually get served.

What does matter a lot is that the MUTEX make a clean decision. It must avoid producing runt pulses that might corrupt data.

## HOW LONG TO WAIT

Because the time to exit meta-stability may vary, we must decide how long to wait for the MUTEX to announce its sequence. There are two choices both of which have inevitable consequences. On the one hand we can choose to wait “as long as it takes” for the MUTEX to reach a decision. This choice leaves uncertainty about how long to wait, and very long delays are possible albeit improbable. On the other hand we may demand a decision in some finite time. This choice leaves uncertainty about the consequence of demanding a decision before the MUTEX has left meta-stability. The consequences can involve corruption of data.

This choice is a bit like Heisenberg’s Uncertainty Principle which says we can either know the momentum of a particle or where it is but not both. We can either wait an indeterminate, but usually short, time for an answer or suffer some probability, usually small, of getting a wrong answer. Our choice may be driven by statistics that describe the behavior of the MUTEX or by the general form of our design.

Clocked designs use variants of the MUTEX as synchronizers. The flip-flop chooses whether to admit an unclocked signal to this clock domain on this clock pulse or a later clock pulse. Designers have found that in most applications a wait of two clock periods reduces the error probability to acceptable levels. However, with modern circuit families and many thousands of synchronizers in a system, it is wise to make an error probability calculation before assuming that two clock periods are enough.

Asynchronous designs simply wait as long as it takes. The appropriate probability calculation concerns expected delay rather than error rates. Except in rare applications asynchronous systems can tolerate the variable delay required to exit meta-stability.

But how do we know when meta-stability is over? We can safely declare meta-stability to be over when the two output voltages `AA` and `BB` of the cross-coupled amplifiers are sufficiently different. In chapter 7 of the Meed-Conway book, Seitz proposed a very simple circuit for detecting the end of meta-stability. The upper part of arbiter2exp is Seitz's circuit adapted for CMOS. Notice the two N-transistors directly above `AA` and `BB`. These two transistors are cross-connected source-to-drain. It follows that only one of them can conduct, and it will do so only when the voltage difference between `AA` and `BB` exceeds its threshold voltage. Thus the final outputs called `grant[A,B]` will go LO only after meta-stability is over.

## THE LIBRARY CIRCUIT - arbiter2

In the library called arbiterL you will find a mutual exclusion element called arbiter2. Arbiter2 has a pretty two-color icon that suggests the cross coupling of NAND gates. Arbiter2 is identical in content to arbiter2exp but constructed from two identical halves. I chose to do the layout as two identical halves to reduce the layout effort. Unfortunately this obscures the topology of the circuit. I included arbit2exp in this lesson to show the circuit more clearly.

You can demonstrate that arbiter2 and arbiter2exp have the same topology by using Electric's NCC tool. Make sure you have only two windows visible, one with each of the two circuits to compare. Now use the **Tools -> NCC -> Cells From Two Windows** command to compare their topology and transistor sizes. You must check a box in NCC preferences if you wish to compare transistor strengths. NCC stands for network comparison code. It is like the "layout vs schematic" (LVS) code in other CAD systems, but NCC works equally well on schematics or layout.

NAND gates have two inputs, one connecting to each of the two series transistors. The two inputs differ in speed. The NAND input closest to the output is faster than the one closest to ground. Which should connect to `req[A,B]` and which should be used for the cross-connection?

If one wishes to minimize the time for exit from meta-stability, one should cross connect the faster input. Examination of arbiter2exp will reveal that it cross connects the slower input. Why? I made that choice because meta-stability is a rare event. I wished to optimize the circuit for the common case of an uncontested request on one input. Because meta-stability is a rare event I chose the slower input for the cross connection.

## ERRORS IN THINKING ABOUT META-STABILITY

When first learning about mutual exclusion many people suggest just picking one side in the case meta-stability. That doesn't solve the meta-stability problem.

Picking one side is equivalent to saying that if I can't decide between choices A and B, I'll decide in favor of choice A. My long comments about the continental divide shed light on the fallacy of this reasoning. The "choose one" procedure is like using a shovel at the continental divide. It simply moves the problem without solving it. If indecision always leads to answer A, one must then face the chance of indecision in deciding between decision and indecision. The problem has moved, but it has not gone away. There is no known way to avoid meta-stability; it is inherent in decision. Sadly, that hasn't stopped many people from trying to find a "solution"!

## THE CROWBAR STOPPER

A decade ago we built rings of GasP circuits to test. We realized that it would be hard to stop such a ring because any stop signal inserted from outside the test chip might cause partially formed pulses, and thereby corrupt the data carried around the ring. Partially formed pulses might even drive some flip-flop into meta-stability. Meta-stability might corrupt the data carried around the ring.

Nevertheless, lacking a better plan we built a "crowbar" stop mechanism. We called it a crowbar stop because, like a crowbar, it's a crude tool not suited to delicacy. Think of stopping your bicycle by inserting a crowbar into the spokes of the front wheel. Quick stop, but possibly with damage to both bike and rider.

Molnar predicted that the "crowbar" stop mechanism used in our earliest test chips would result in about a 2% probability of data corruption. And so it turned out.

One crowbar stop mechanism appears in the upper FIFO in your homework assignment. One stage in this FIFO is called gaspCrowbar. In place of the inverter connected to `pred`, gaspCrowbar has a NAND gate. The NAND gate allows a pulse at `fire` only when the input called `crow` is LO. As soon as `crow` becomes HI, gaspCrowbar stops firing.

Suppose `crow` goes HI just as gaspCrowbar is about to act. The result might be a partial action rather than a full one. You can observe such partial failure. In the circuit called fifos, this week's simulation experiment, I have set the upper delay time to 5825.3 psec. In my simulation that value produced a runt pulse on `fire[3]`. It's called a runt pulse because "runt" is the word for the smallest puppy or kitten or pig in a litter. If your simulation is like mine, you will see a runt pulse as the last pulse on `fire[3]`.

Your task with the upper FIFO is to adjust the stop time and watch what happens to the runt pulse. Delay times smaller than my value of 5825.3 ps will stop the FIFO sooner, avoiding the partial stop. Using less than my delay, how **late** can you stop the FIFO and get exactly five good pulses on `fire[3]` and no runt? How late must you stop the FIFO to get a sixth good full size pulse on `fire[3]`? What is the difference in those delay times?

## THE PROPER STOPPER

Gaining sophistication, we realized that a mutual exclusion element could provide a clean stop to our GasP ring experiments. The lower FIFO in your simulation exercise contains a stage called gaspStop that replaces the inverter on `pred` with the mutual exclusion element called arbiter2. This mutual exclusion element uses wide transistors and thus puts more load on `pred` than the crowbar stop circuit. That is why the lower FIFO is a bit slower than the upper one.

I have set the stop time for the lower FIFO to 5653.35 psec. This particular value is of no great interest except to remind you that you can set the input time values for a simulation in SPICE very precisely. In this case I've specified it to within 10 fs or 1/100 of a ps. You will have to be even more precise for this assignment.

Your task is to discover the precise value of stop time that separates two distinct output forms. I see five output pulses on `fire[9]` with my value of stop time. What is the earliest stop time you can find that produces exactly five pulses on `fire[9]`? How much earlier than that must the stop signal be to get only four pulses on `fire[9]`? Please adjust your stop time to within 1/100 ps or less.

I recommend that you think carefully about your search strategy for finding the best value of stop delay. There are too many steps of 10 fs for a linear search. Take big steps first and home in on the proper time value. A binary search will prove reliable, but with practice you may be able to do slightly better than a binary search.

If you get your stop time adjusted carefully enough, you can observe the mutual exclusion element in meta-stability. You should observe that your fifth output pulse will occur later than you might otherwise expect. The extra delay of the mutual exclusion element when it exits from meta-stability will delay the last pulse. You should also examine the cross connections in arbiter2. The two cross connections should show identical voltages for an extended period.

Because this kind of simulation forces SPICE to respond to small differences in big numbers, SPICE sometime produces anomalous results. For example, SPICE might force meta-stability so accurately that there's no difference at all between the cross connection voltages. That might make simulated meta-stability last for ever! Moreover, if you change anything at all, even the order in which SPICE does its simulation, your results may differ. I advise you to change only the delay time between SPICE runs. Avoid other circuit changes.

We built several chips to test various aspects of the "proper stopper." Never, in all our tests, did we observe data corruption from stopping an asynchronous system with a proper stopper. The mutual exclusion element really works, even over the billions of trials we ran on real chips.

ANSWER SHEET – due by beginning of class on 26 October 2010 **4**

Name ___________________________

Turned in on Date ___________________________

**Answer about your simulation:**

My simulation uses ___________________________ technology – e.g. 180 MOSIS

I simulated ____________ nsec of time on each SPICE run.
My spice runs took ___________ seconds each to execute on server ______________.

**Answer from simulation:**

My upper FIFO operates every ______ psec for a throughput of ______ GDI/sec.
(GDI/sec is GIGA data items per second)

My lower FIFO operates every _______ psec for a throughput of _______GDI/sec.

**Get rid of the runt on `fire[3]` (but only barely):**

To produce six whole pulses on `fire[3]` my stop time must exceed __________ psec.
An earlier stop time: ___________ psec produces an incomplete sixth pulse.

To produce five whole pulses on `fire[3]` my stop time must be at least _______ psec.
A later stop time: ____________ psec produces a tiny runt sixth pulse.

A stop time range _____________ psec long produces runt or short `fire[3]` signals.
This is approximately __________ % of the full cycle of __________ psec.

**Find meta-stability in driving `fire[9]`:** (hint: use logarithmic search)

The closest two values of stop time I can find are:

______________ psec produces five whole pulses on `fire[9]`.

______________ psec produces only four whole pulses on `fire[9]`.

These stop times differ by ___________ fsec (femto-seconds = $10^{-15}$ sec)

Equal values for the cross-coupled signals in arbiter2 lasted for _________ psec.

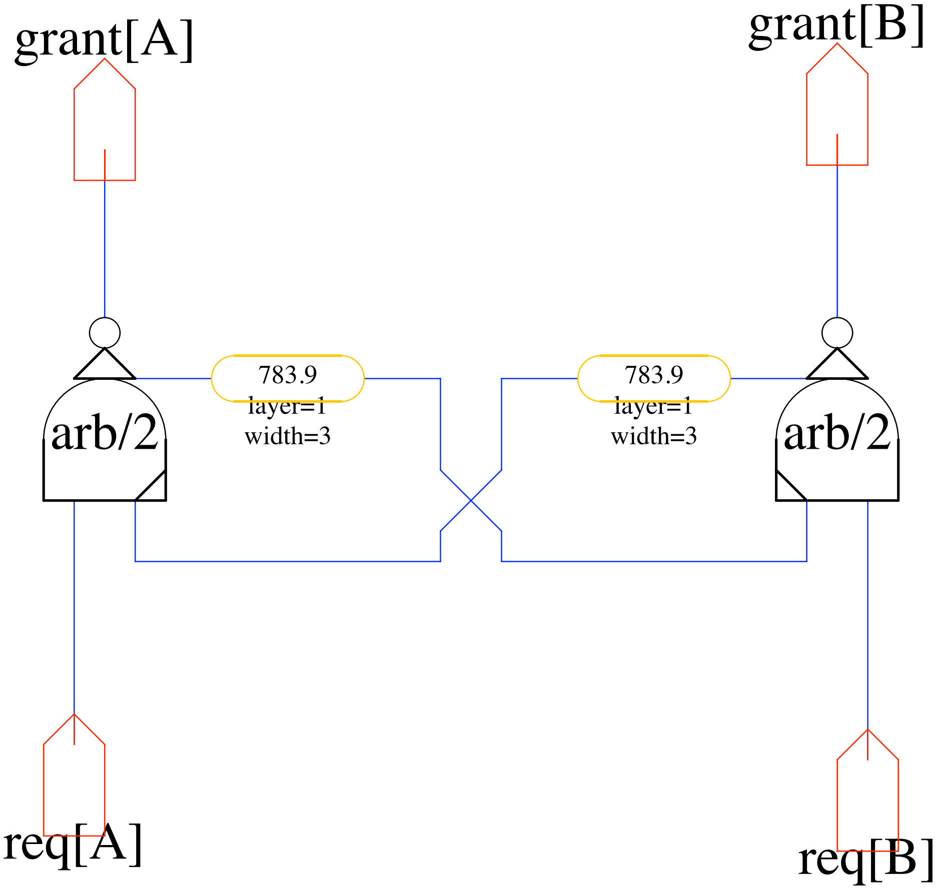
Their voltage at meta-stability was _____________ volts.

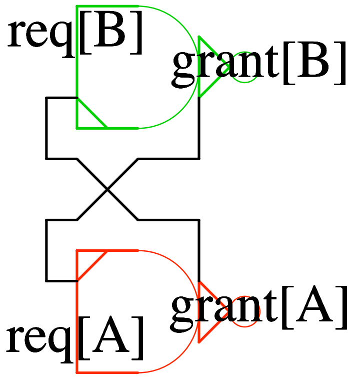
Describe any anomalous behavior of SPICE on the back of this sheet:

# arbiter2

## this is the arbiter
## with anti-metastability

ies 10 September 2007
renamed 26 October 2008

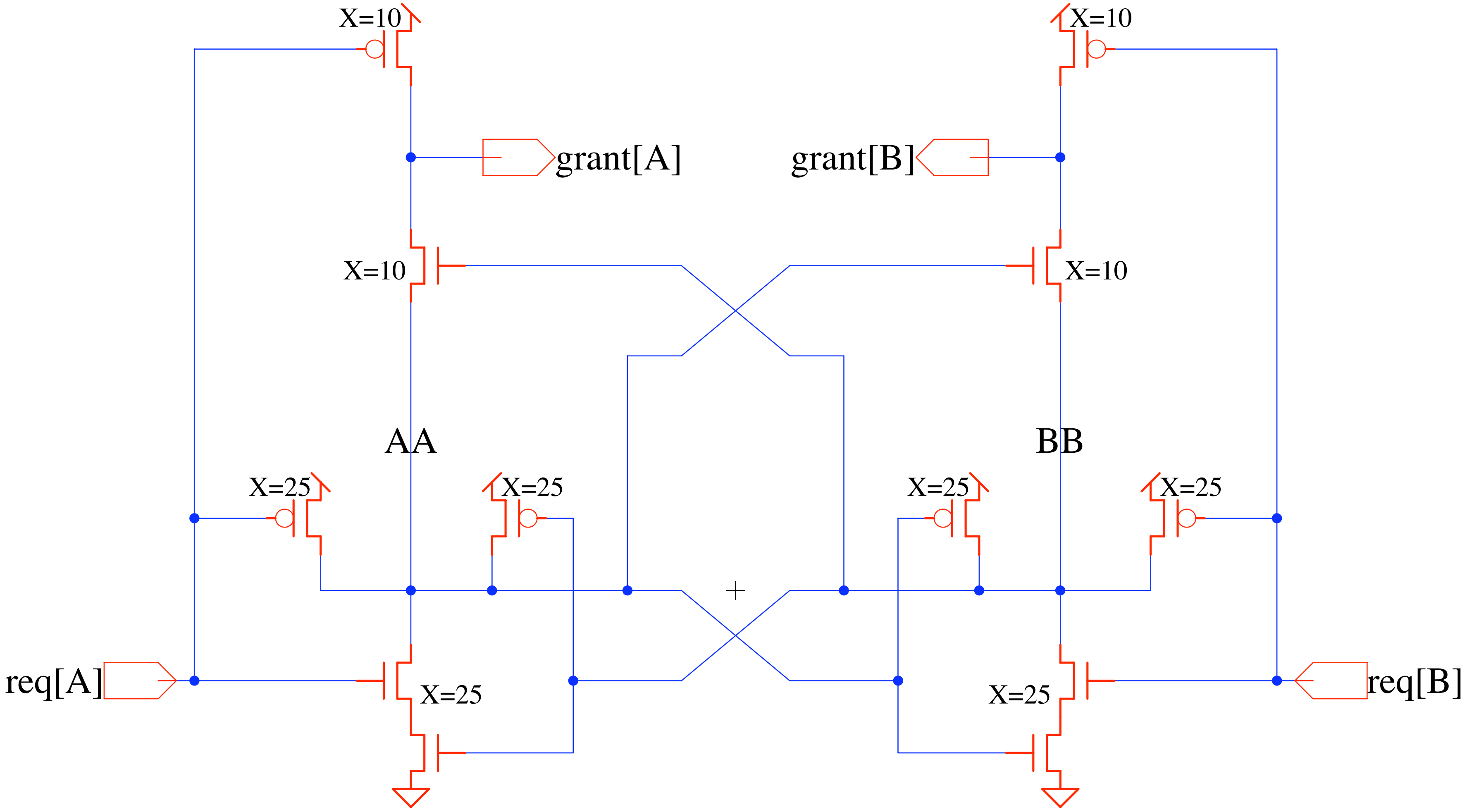




# arbiter2exp

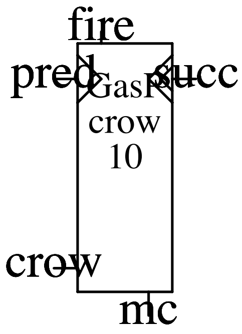
ies 11 October 2010

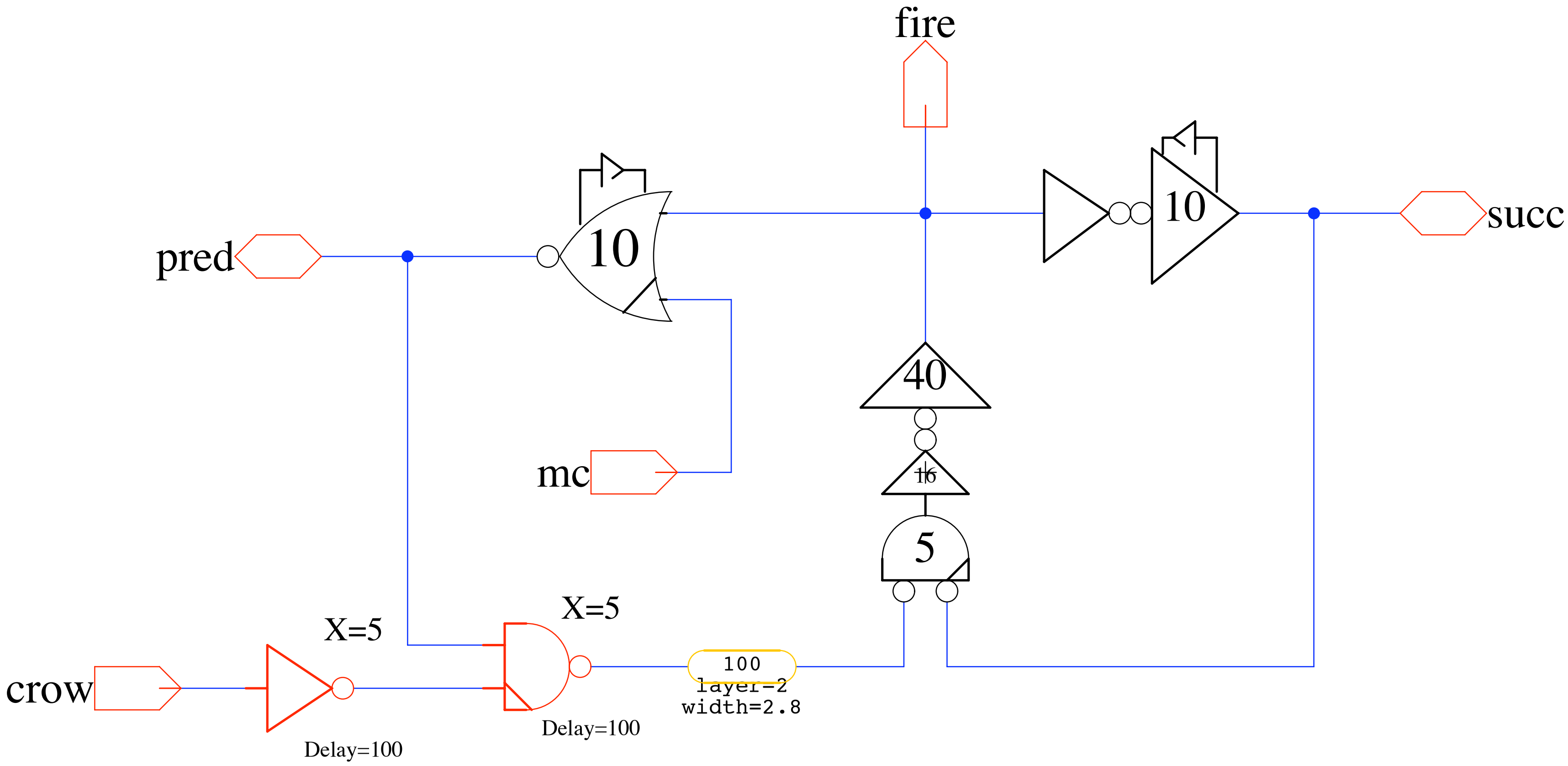
this is the arbiter2 circuit expressed as transistors



# gaspCrowbar

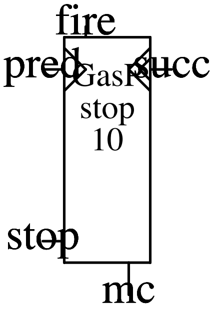
ies 11 October 2010

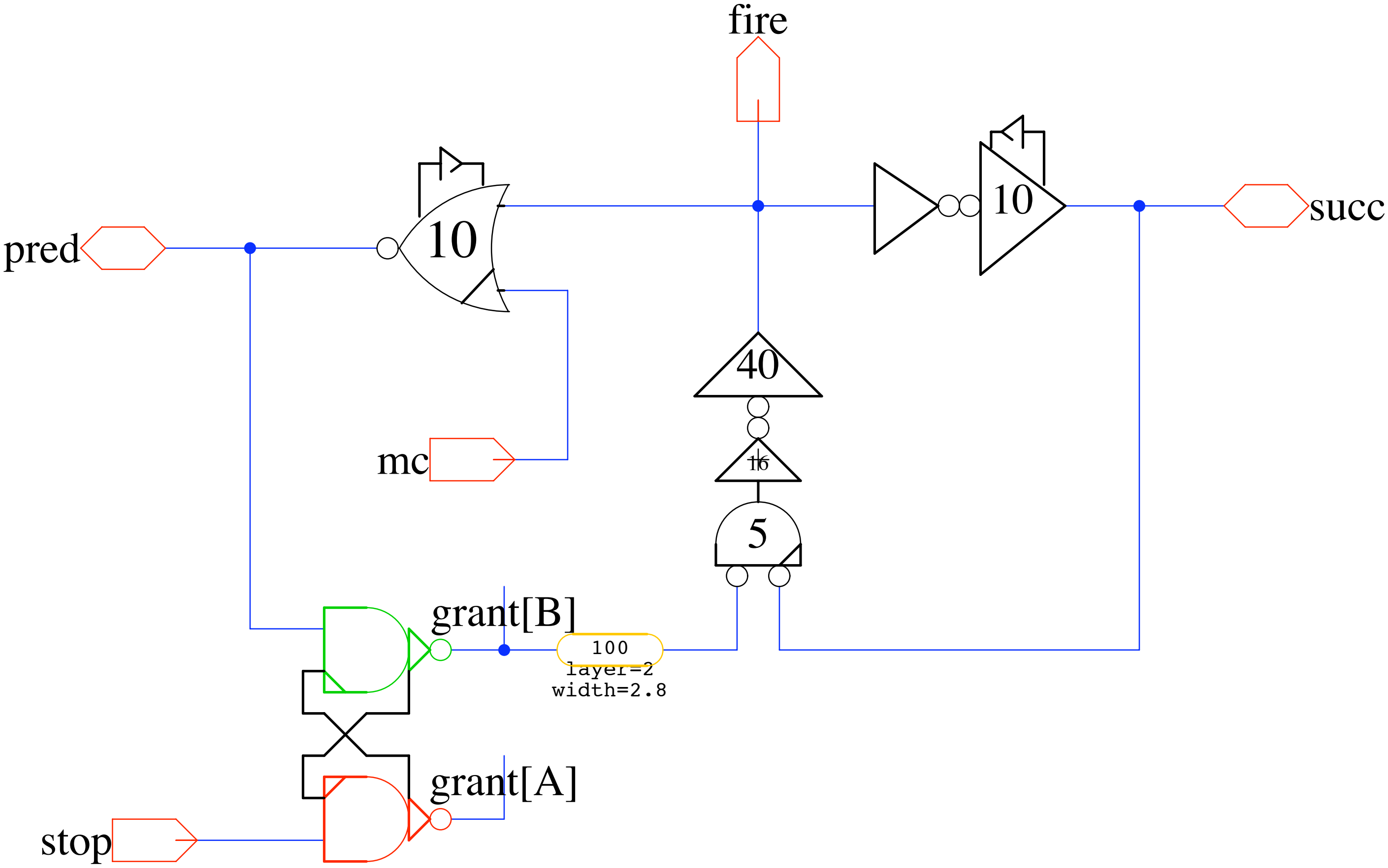




# gaspStop

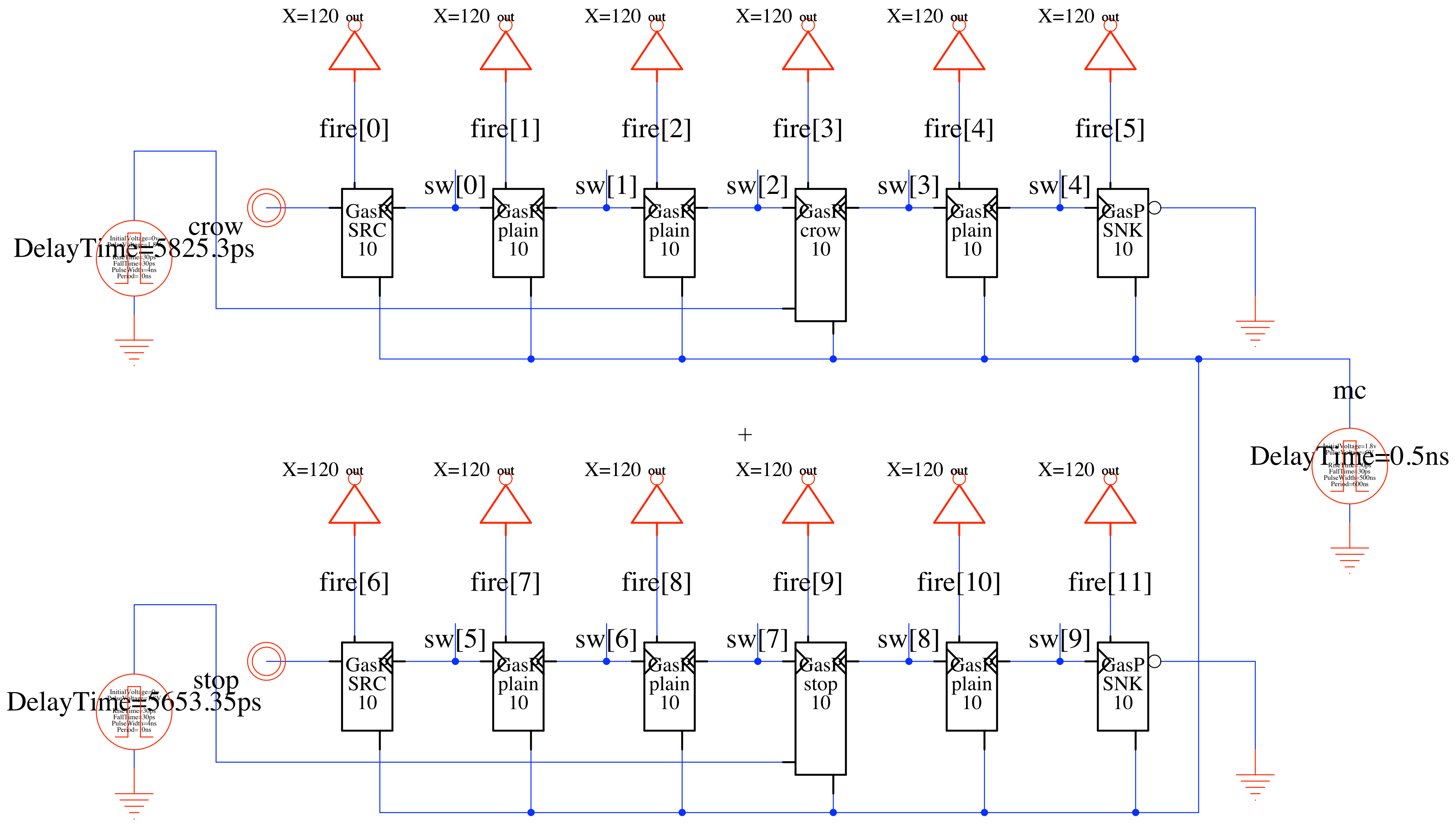
ies 17 June 2010

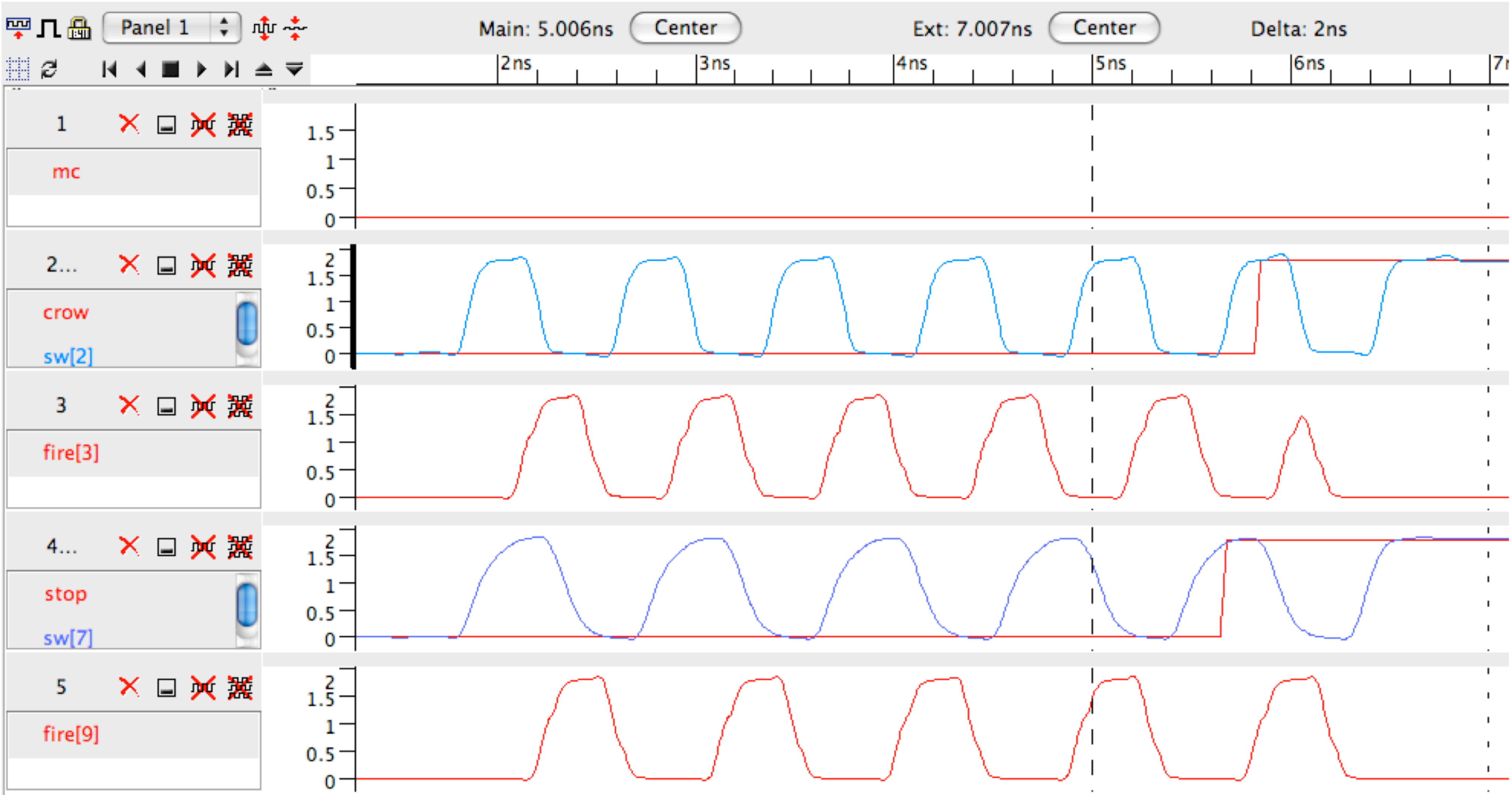




# fifos

ies 11 October 2010




Panel 1
Main: 5.006ns
Center
Ext: 7.007ns
Center
Delta: 2ns
2ns
3ns
4ns
5ns
6ns
7n
1
mc
2...
crow
sw[2]
3
fire[3]
4...
stop
sw[7]
5
fire[9]
1.5
1
0.5
0
2

ANSWER SHEET – due by beginning of class on 26 October 2010 **4**

Name ___________________________

Turned in on Date ___________________________

**Answer about your simulation:**

My simulation uses ___________________________ technology – e.g. 180 MOSIS

I simulated ____________ nsec of time on each SPICE run.
My spice runs took ___________ seconds each to execute on server ______________.

**Answer from simulation:**

My upper FIFO operates every ______ psec for a throughput of ______ GDI/sec.
(GDI/sec is GIGA data items per second)

My lower FIFO operates every _______ psec for a throughput of _______GDI/sec.

**Get rid of the runt on `fire[3]` (but only barely):**

To produce six whole pulses on `fire[3]` my stop time must exceed __________ psec.
An earlier stop time: ___________ psec produces an incomplete sixth pulse.

To produce five whole pulses on `fire[3]` my stop time must be at least _______ psec.
A later stop time: ____________ psec produces a tiny runt sixth pulse.

A stop time range _____________ psec long produces runt or short `fire[3]` signals.
This is approximately __________ % of a full cycle.

**Find meta-stability in driving `fire[9]`**: (hint: use logarithmic search)

The closest two values of stop time I can find are:

______________ psec produces five whole pulses on `fire[9]`.

______________ psec produces only four whole pulses on `fire[9]`.

These stop times differ by ___________ fsec (femto-seconds = $10^{-15}$ sec)

Equal values for the cross-coupled signals in arbiter2 lasted for _________ psec.

Their voltage at meta-stability was _____________ volts.

Describe any anomalous behavior of SPICE on the back of this sheet: