AutoRanger Circuit Design

Jesse Cousins
Brian French
Ulas Karaoz

December 18, 2000

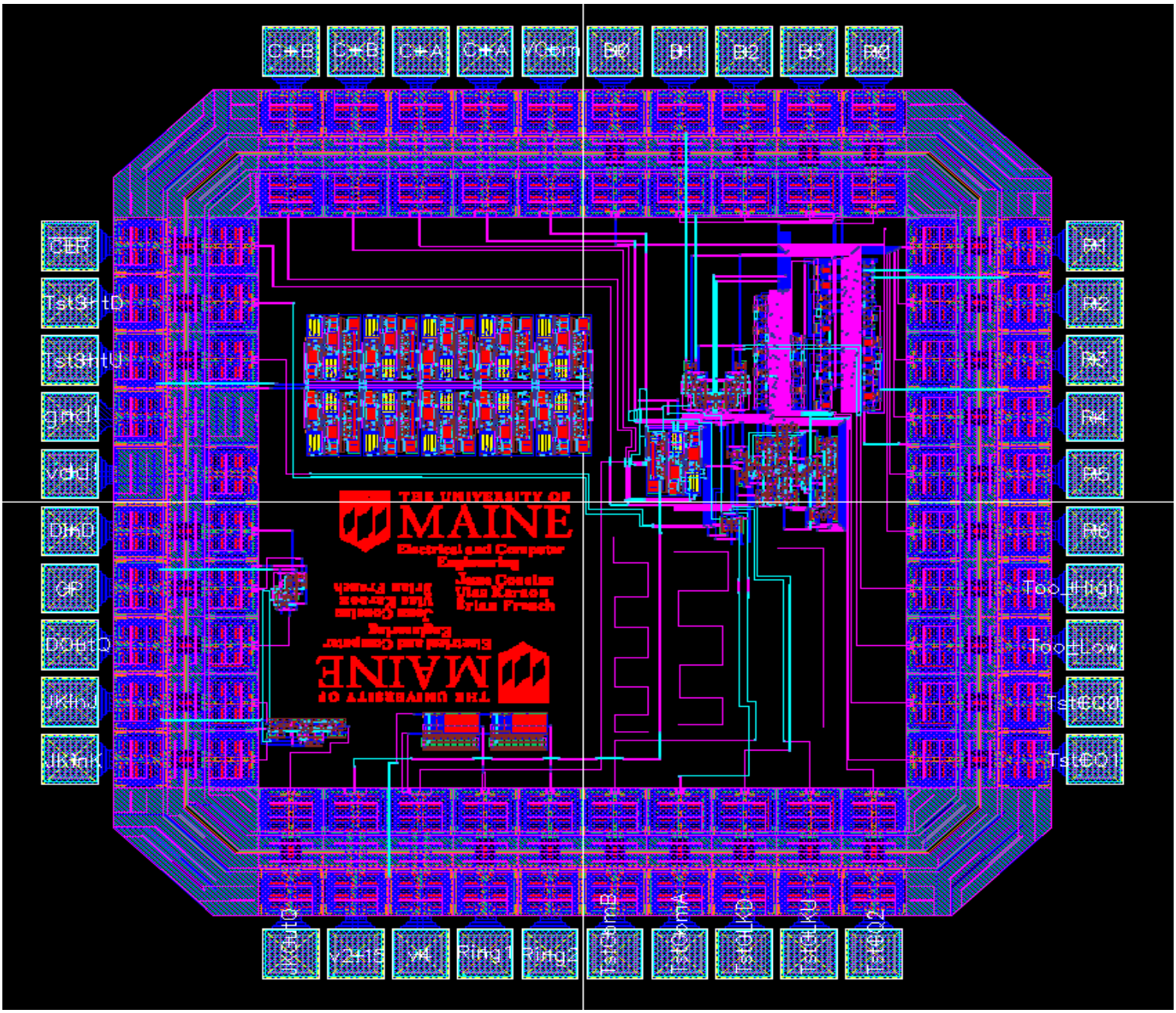


Table Of Contents

1. Introduction

1.1 Abstract

This paper presents the design and layout of an automatic order-of-magnitude resistance measurement circuit for use in sensor applications. A metal-oxide sensor's resistance can change by several orders of magnitude depending on the gas present and temperature of operation. In order to verify that a sensor is working, SRD Corp. has created a test board that compares a sensor's resistance to a series of reference resistances and indicates the order of magnitude in which the resistance falls. This paper outlines an effort to reduce this board to a single chip.

1.2 Project Objectives & Overview

AutoRanger Circuit Board Schematic

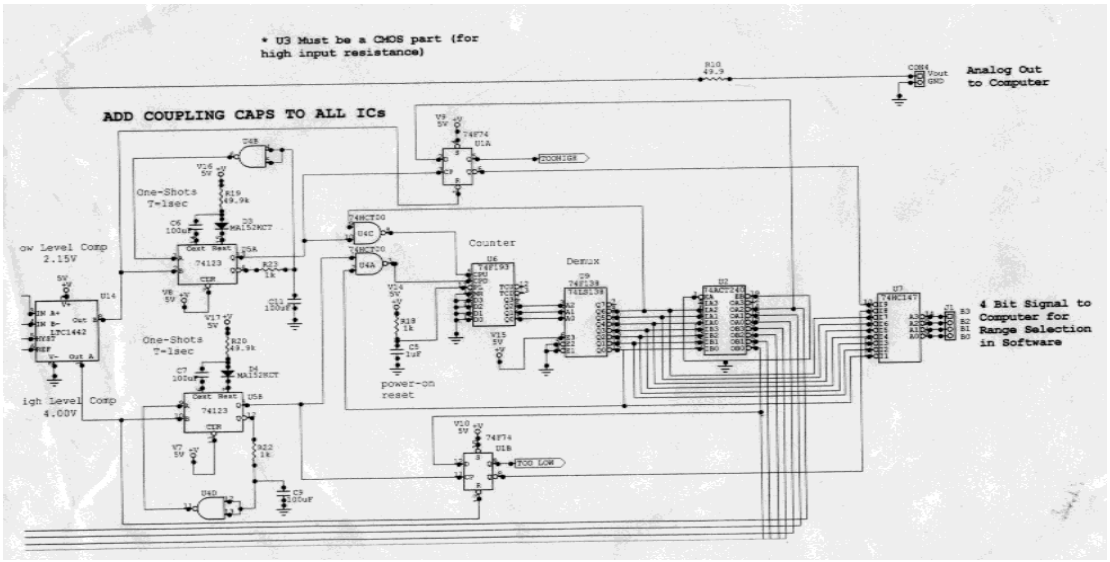


Figure 1.2

The AutoRanger circuit schematic for the board currently in use at SRD can be seen in Figure 1.2 above. An external network produces a voltage proportional to the relative value of the reference resistance and sensor. This voltage is then fed into the two comparators at the right of Figure 1.2. If the resistance is greater than 4 volts, or less than 2.15 volts, one of the two outputs will go high. The output's corresponding one-shot emits a clock pulse every second as long as the output remains high. These clock pulses are fed into a 3-bit counter that shifts up or down depending on whether the voltage was too high or too low, respectively. The output of the three-bit counter is sent to a 3 to 8 decoder. This output controls the mechanical relays that determine the reference resistance. The output of the 3 to 8 decoder is also sent to a decimal to binary-coded decimal decoder in order to generate a 4-bit resistance range output value.

2. Designs & Verifications

2.1 Latches /Flip-Flops/Counters

2.1.1 Basic Component & Design Considerations

This section describes the basic gates that were used for designing the latches, flip-flops and subsequently the counter used in the AutoRanger circuit. Each subsection will review the specific component design issues and simulation results.

The first design decision was choosing complementary CMOS for all our basic gates. The reasons for this decision are that complementary CMOS possess characteristics such as high noise margins and virtually zero static power dissipation. It also allows for creating symmetric noise margins, since we have ratioed logic with the pull-up & pull-down networks. Complementary CMOS also has VOH & VOL at VDD and GND respectively, so there are no weak 'ones' or 'zeros'. With this design we can obtain a high degree of modularity which aids in a speedy implementation. Another equally persuasive reason is that it is a static circuit. This relieves many of the headaches dynamic circuits create such as clock distribution issues, refreshing, clock feed-through problems, voltage level shifting and restoring, and many others.

The next thing to consider was if the design was to be constructed from NAND or NOR logic. In complementary CMOS NAND logic the pull-down network results in series connections with the NMOS devices, which is desired over having series connections in the pull-up network of PMOS devices. A series connection effectively slows down the gate because the aggregate resistance, through which discharging or charging occurs is effectively increased. Because PMOS has a lower mobility constant, PMOS devices in series effectively have a higher resistance than the comparable NMOS devices in series. Even though one can lower the resistance by increasing the widths of these PMOS devices, widening also increases the diffusion capacitance.

Finally, since we are using static gates all subsequent circuits built from those gates will consume very little power. This is because we have almost zero static power and low dynamic power because we are running the circuit around 1Hz.

2.1.1.1 Inverter gates

The inverter gate is used ubiquitously throughout digital design and is regarded as the basic building block of modern digital design. The static CMOS schematic design for the inverter that was used is below in Figure 2.1.1.1A.

Static CMOS Inverter Schematic

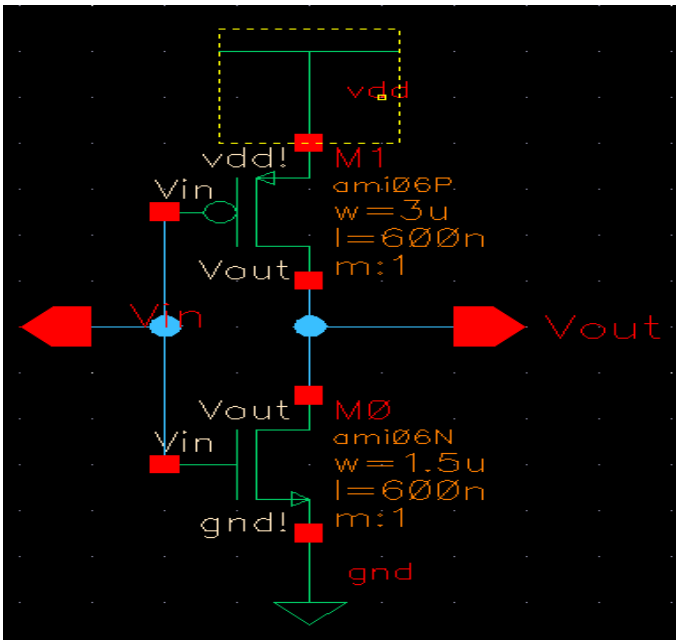


Figure 2.1.1.1A

Note in the simulations of T_{pHL} in Figure 2.1.1.1B and T_{pLH} in Figure 2.1.1.1C the input curves are on the bottom and the output curves are on the top. Also note that the load used for this inverter was another identical inverter gate. Looking at the simulations of the inverter we can see that not only does it function correctly, but that we have comparable T_{pLH} and T_{pHL} times. One can use a first order approximation of T_{pHL} to be $.69R_PC_L$ of the pull-up network and use T_{pHL} to be $.69R_nC_L$ of the Pull-Down network. Note that R_P and R_N are not constants and are in fact proportional to the drain current & voltage across the drain and source. Since the mobility constant of the PMOS is approximately two-times less than the NMOS (for this process), which effectively doubles R_P, we double width of the PMOS. This makes R_n and R_P approximately equal resulting in comparable T_{pLH} and T_{pHL} times. This results in a PMOS transistor sized at 3µm /.6µm (W/L) and an NMOS transistor sized at 1.5µm /.6µm (W/L). The simulations show T_{pLH} to be about *140ps* and T_{pHL} *190ps*.

Static CMOS Inverter Simulations of T_{pHL} and T_{pLH} Respectively

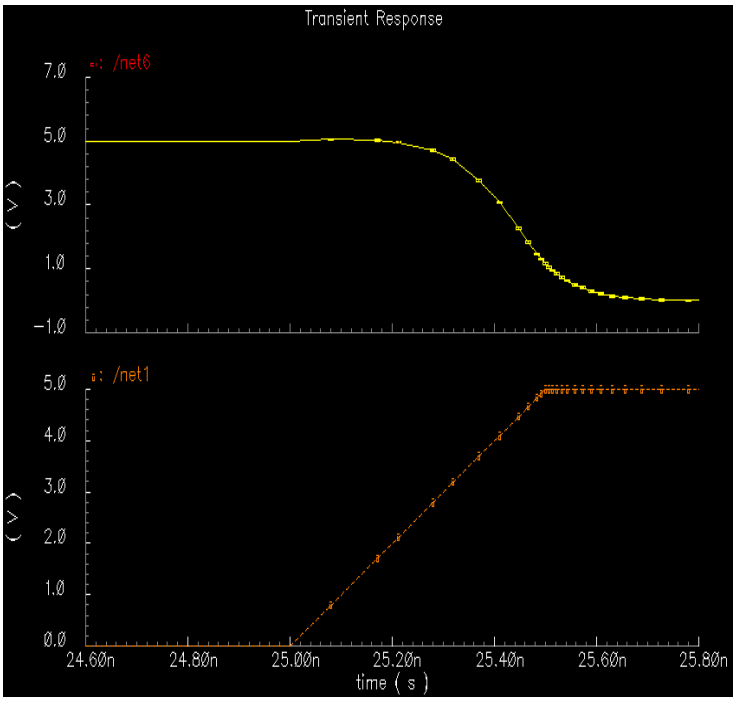


Figure 2.1.1.1B

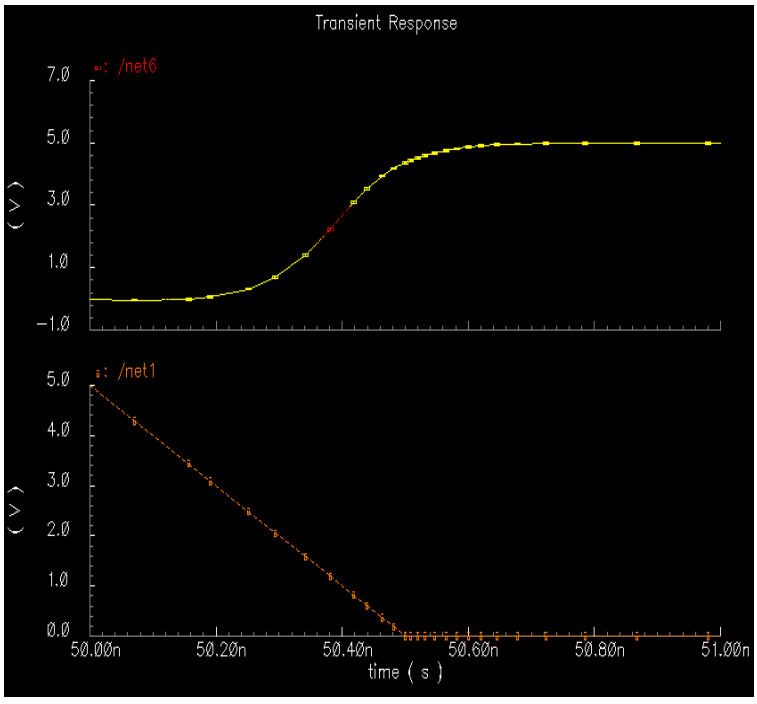


Figure 2.1.1.1C

The layout of the inverter, which was not complex, is shown in Figure 2.1.1.1D. The size is 15.15µm x 16.8µm. The height was kept at 16.8µm to keep the inverter modular with some of the other basic gates.

Static CMOS Inverter Layout

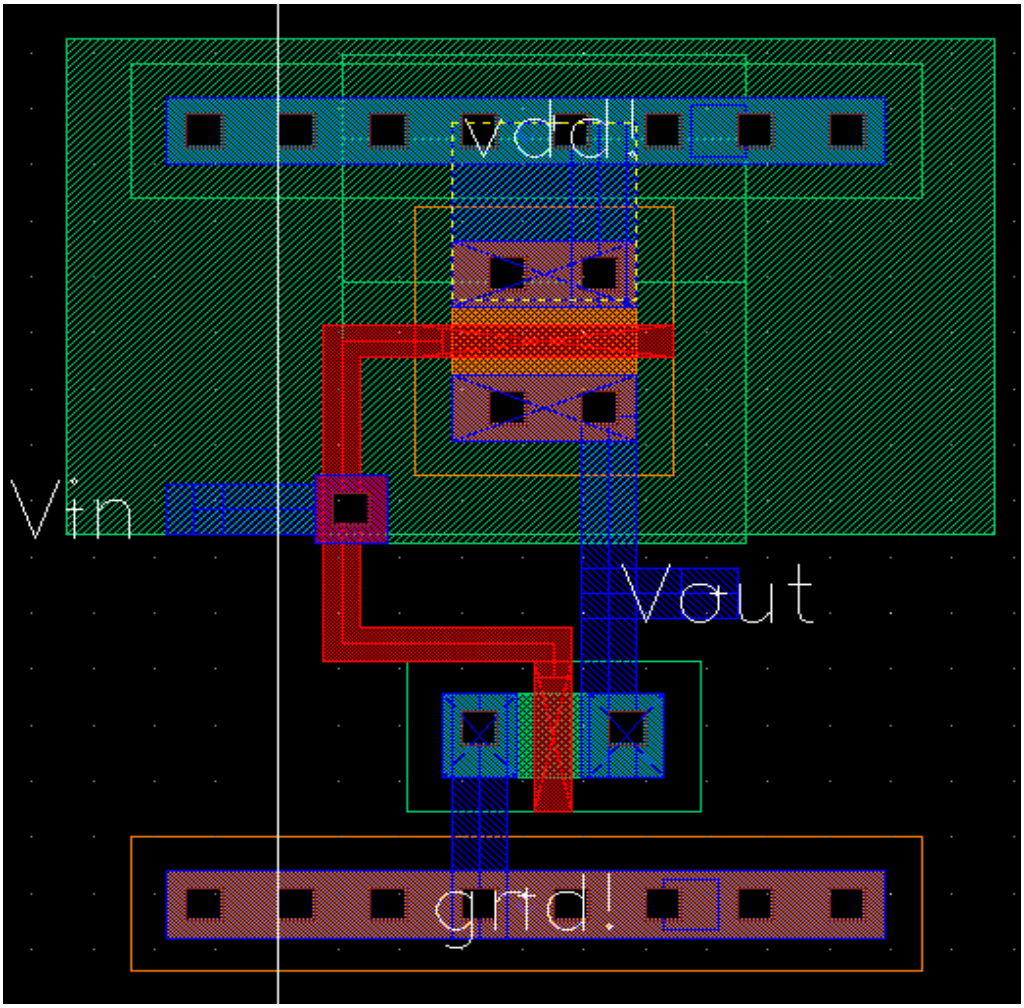


Figure 2.1.1.1D

2.1.1.2 NAND gates

The NAND gate is a complement to the inverter. Together they can be used to implement any Boolean logic functions. With the sequential circuits we are designing here we needed both two-input and three-input NAND gates. The static CMOS schematic designs for the two-input and three-input NAND gates are shown below in figures 2.1.1.2A & 2.1.1.2B respectively.

Static CMOS NAND gates Schematics

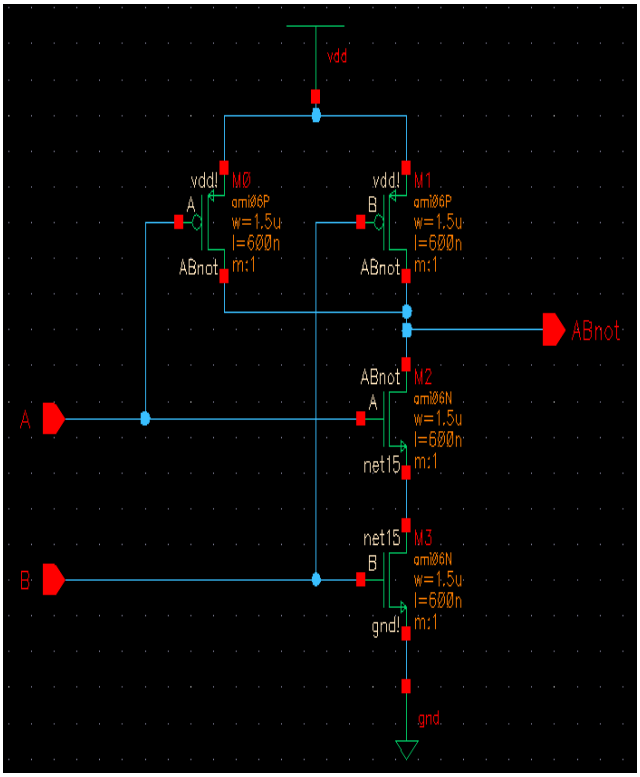


Figure 2.1.1.2A

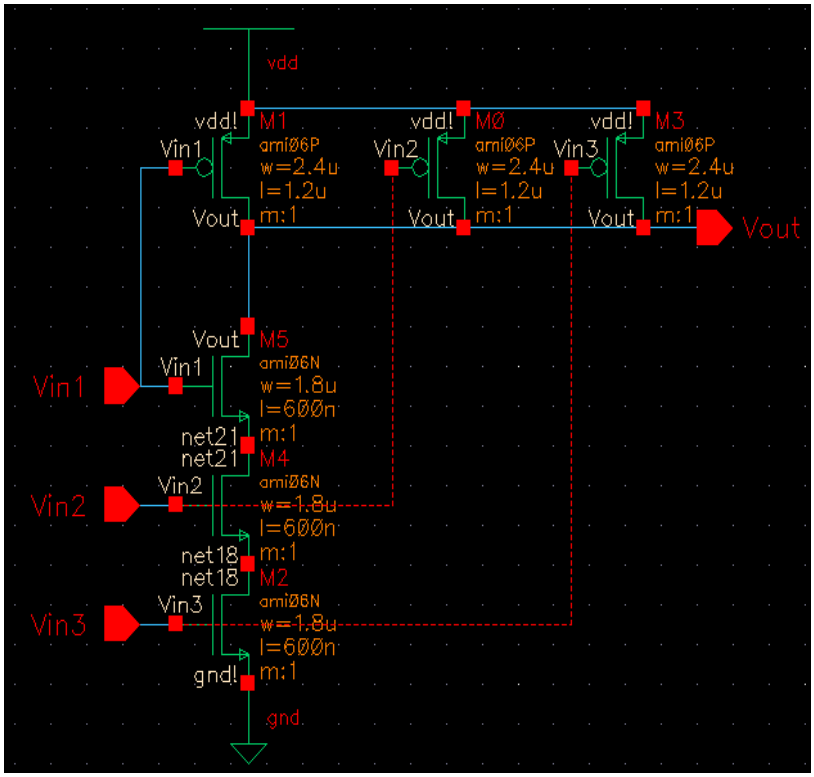


Figure 2.1.1.2B

The simulations for both two & three input NAND gates proved that the device was functioning correctly. This is compared with the truth tables in figures 2.1.1.2C & 2.1.1.2D. Also note that the load used was the inverter gate. For each NAND gate T_{pHL} would than approximated to be $.69R_PC_L$, which is the worst case of only one PMOS being on. The T_{pLH}, according to a first order approximation, would be $(.69)2R_NC_L$ and $(.69)3R_NC_L$ which is the worst case for the two-input and three-input NAND gates respectively. With mobility in mind we would like results in worst-case propagation delays to be as symmetrical as they were with the inverter. This results in worst-case effective widths and lengths. For the two-input NAND this results in a PMOS transistor sized at 1.5µm /.6µm (W/L) and an NMOS transistor sized at 1.5µm /1.2µm (W/L). For the three-input NAND this results in a PMOS transistor sized at 2.4µm /1.2µm (W/L) and an NMOS transistor sized at 1.8µm /1.8µm (W/L). For the two-input NAND simulations show worst-case T_{pLH} to be about *210ps* and worst-case T_{pHL} *230ps*. For the three-input NAND simulations show worst-case T_{pLH} to be about *250ps* and worst-case T_{pHL} *280ps*.

Two-Input NAND Truth Table

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

Figure 2.1.1.2C

Three-Input NAND Truth Table

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

Figure 2.1.1.2D

Static CMOS two-input NAND gates *Simulation*

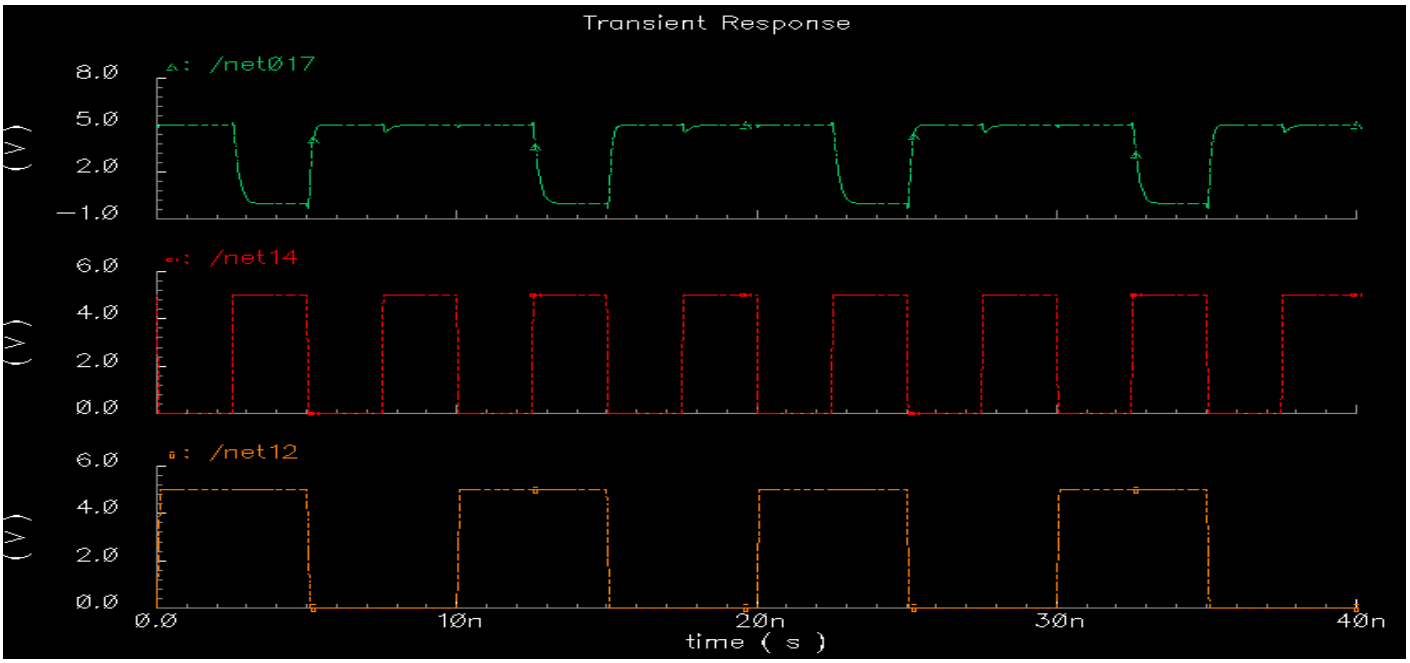


Figure 2.1.1.2E (net12, net14 inputs; net17 output)

Static CMOS three-input NAND gates *Simulation*

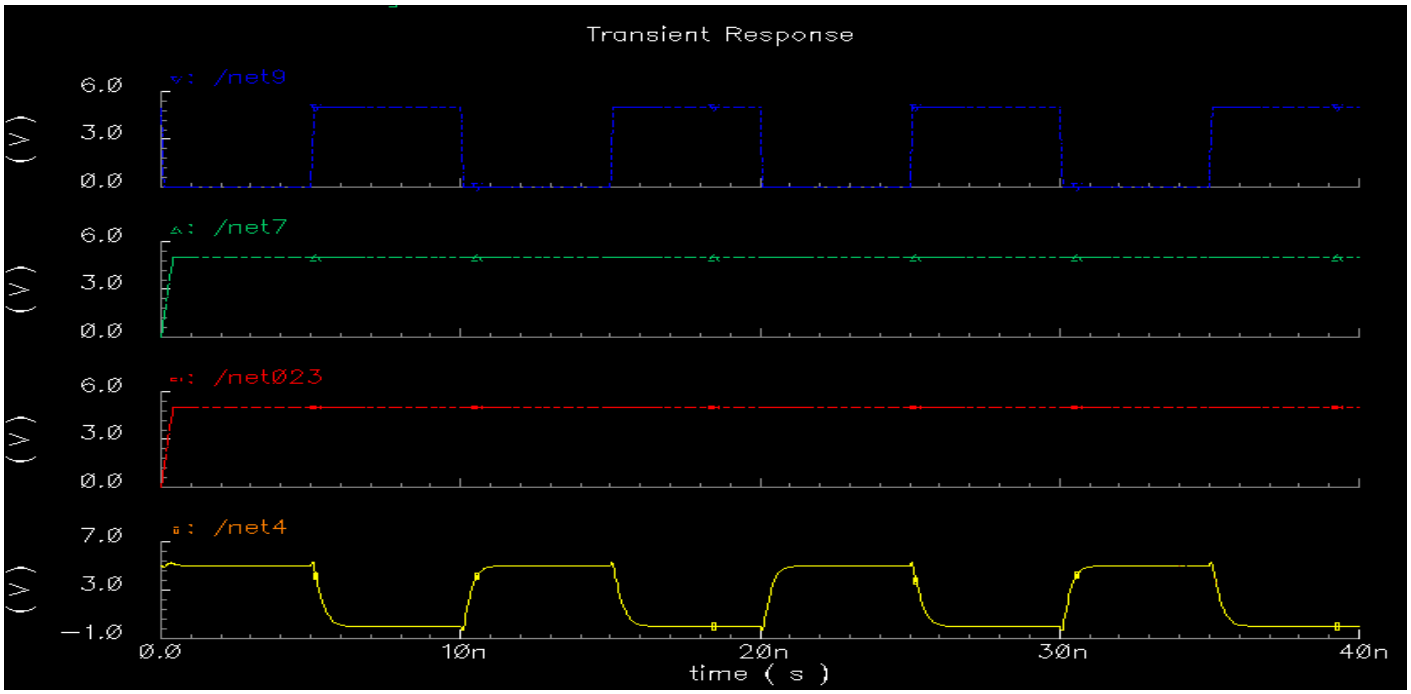


Figure 2.1.1.2F (net9, net7, net23 inputs; net4 output)

The layout of both NAND gates is shown in Figure 2.1.1.2G and Figure 2.1.1.2H. The two-input NAND gate is 15.15µm x 16.8µm. The three-input NAND gate is 19.05µm x 21.15µm.

Static CMOS NAND gates Layouts

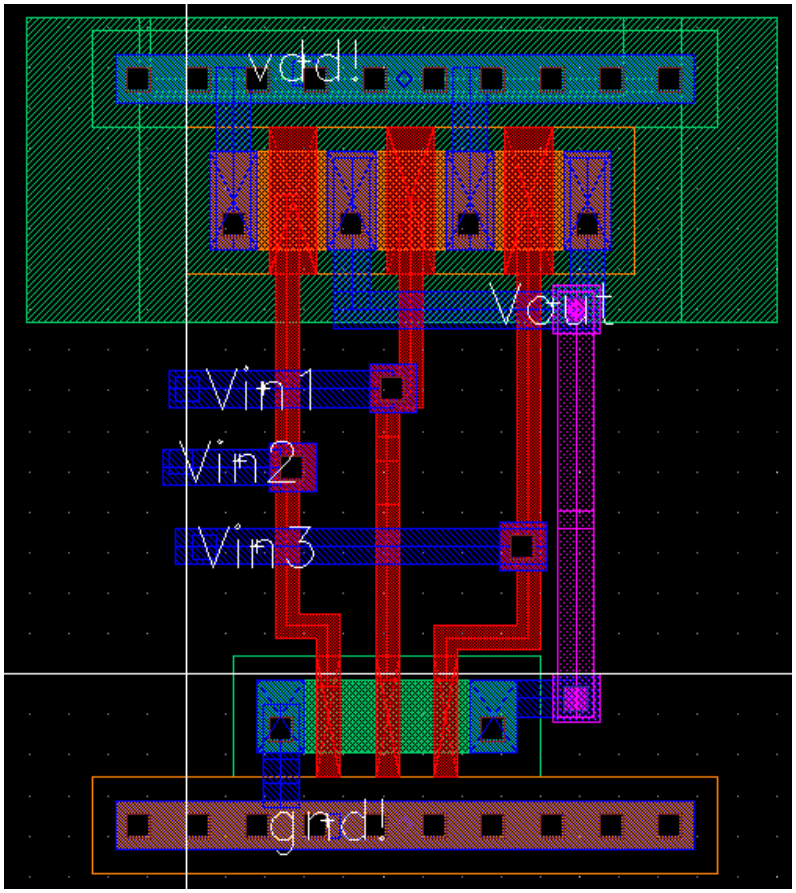


Figure 2.1.1.2G

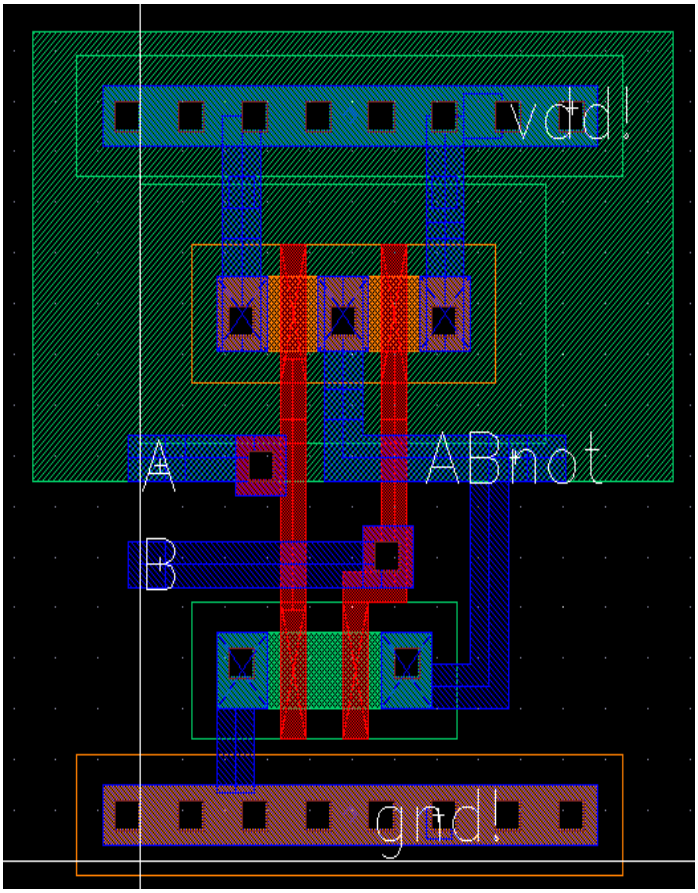


Figure 2.1.1.2H

2.1.1.3 Transmission gates

The transmission gate is a unique gate because the supply rails do not drive it. The output is either at a high impedance state or it is a value equal to the input signal. The complementary gate voltages on the PMOS and NMOS transistors control the two states. In Figure 2.1.1.3A we see that if *C* is high then *A* equals *B*. If *C* is Low then *B* is at a high impedance state. The only drawback to using this gate is that you always need complementary control signals, but this does not take from the fact that it is simple and fast. Transmission gates are commonly used in multiplexers, pass-transistor logic circuits, sequential logic devices, and many other types of circuits.

CMOS Transmission Gate Schematic

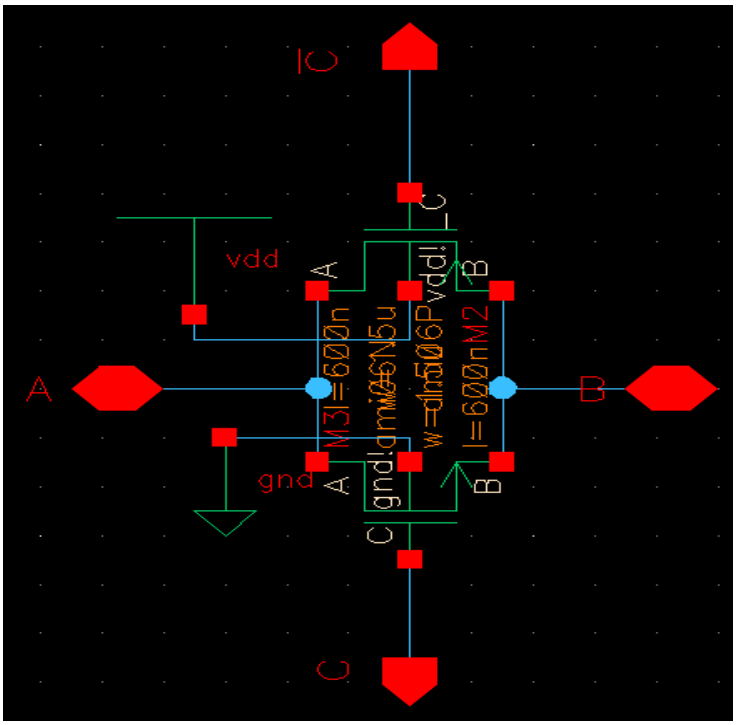


*Figure 2.1.1.*3A

In the simulations of T_{pHL} in Figure 2.1.1.3B and T_{pLH} in Figure 2.1.1.3C the input curves are on the bottom and the output curves are on the top. The load as with the other simulations is a single inverter. We verified from the simulation that the gate does indeed function correctly. One can use a first order approximation of T_{pHL} to be $.69R_{com}C_L$ which is identical to T_{pLH}, since this is non-ratioed logic. Note that R_{com} is just the composite of the NMOS and PMOS effective resistances. Thus sizing doesn't have much effect since making the devices wider reduces resistance while increasing diffusion capacitance. One would only make it larger than a minimum-sized device if you need to drive a large load. This results in both the PMOS and NMOS transistors being sized at 1.2µm /.6µm (W/L). The simulations show T_{pLH} to be about *74ps* and T_{pHL} *70ps*. Note in below simulations the transmission gate was left on.

CMOS Transmission gate *Simulations*

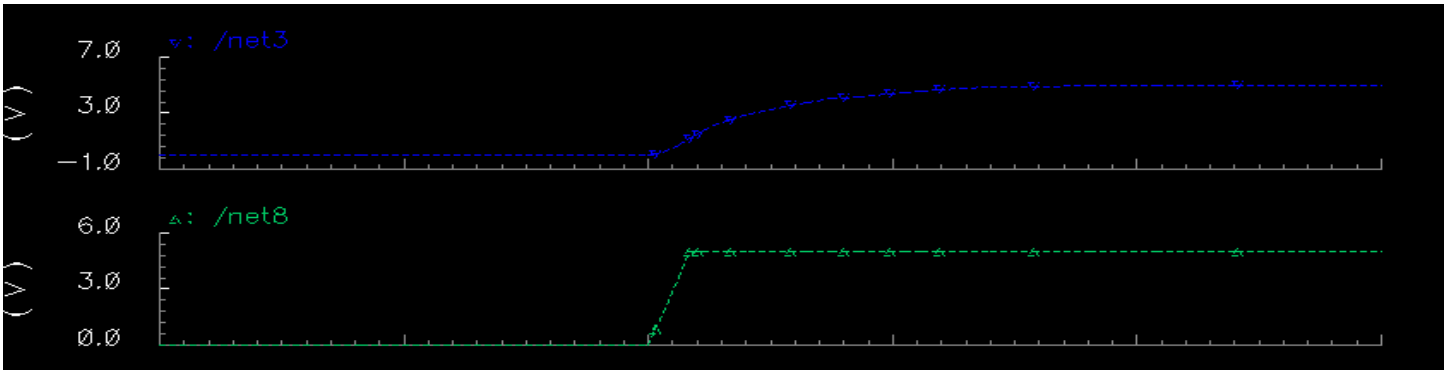


Figure 2.1.1.3B (net8 input; net3 output)

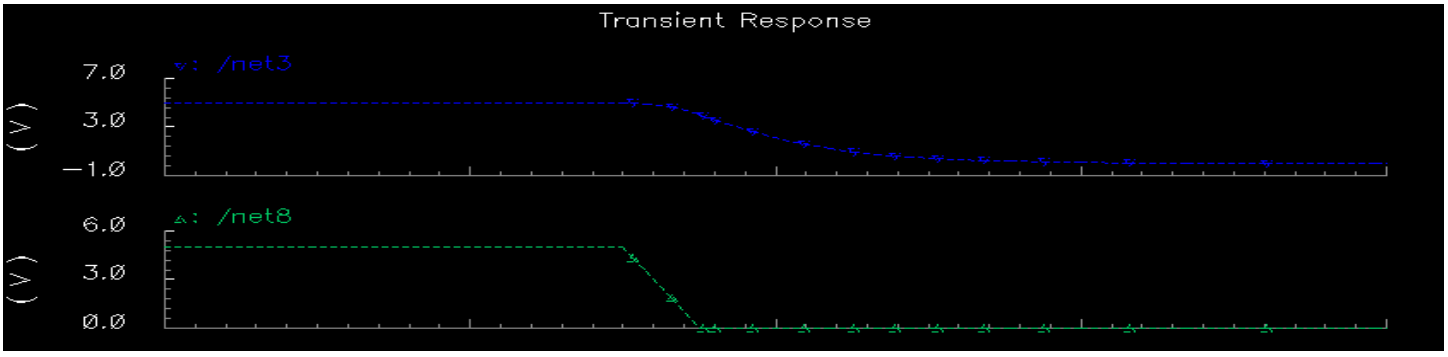


Figure 2.1.1.3C (net8 input; net3 output)

The layout of the transmission gate is shown in Figure 2.1.1.3D. The size is 15.15µm x 16.8µm. The height was kept at 16.8µm to keep the gate modular with some of the other basic gates.

Transmission gate Layout

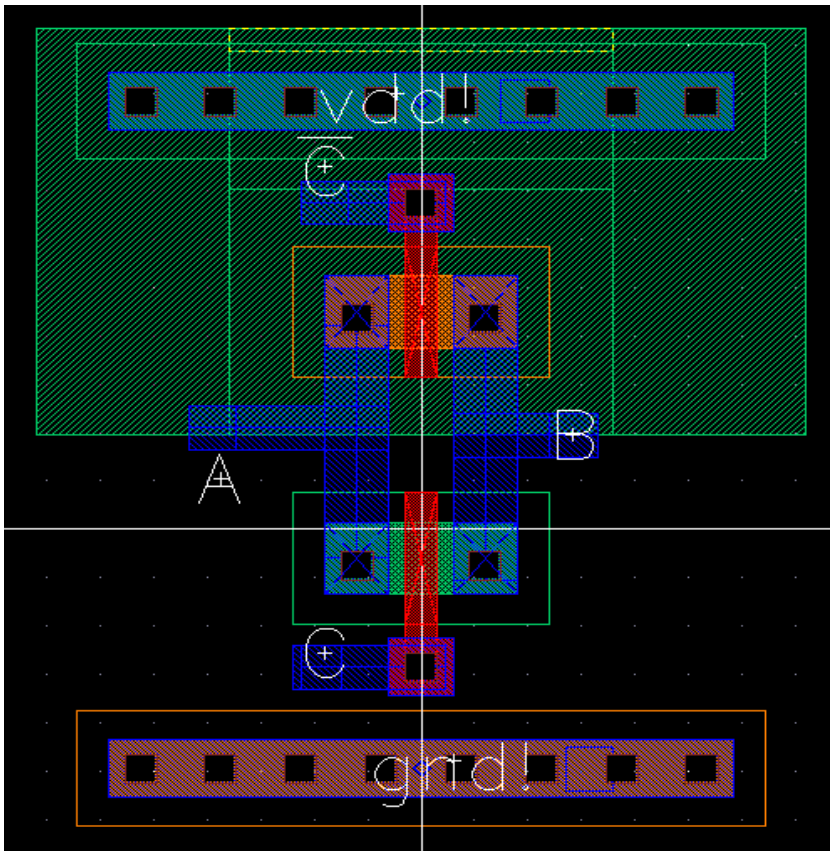


Figure 2.1.1.3D

2.1.2 SR-Latch

2.1.2.1 Description

For the designed function of the AutoRanger circuit we are required to keep track of the current operation mode of the Up/Down Counter. The two modes simply being counting up or down. For this function we required a circuit with two stable states; simply, we needed a binary memory characteristic. The natural solution was to look at bistable multivibrators, a class of static sequential circuits capable of a holding two stable states and also able to toggle between those states. The SR-Latch appeared to be the simple solution. The SR-Latch is recognized as a basic building block that can be utilized in static memories, counters, timing devices, and data sequencing devices. The essential function of an SR-Latch is that it performs as a simple 1-bit register.

2.1.2.2 Design considerations

Subsequent design issues were speed and whether or not it would be a synchronous device. As for speed, the circuit parameters call for it to run on the order of hundreds of milliseconds to seconds. Intuition reveals that for a pair of minimally sized cross-coupled NAND gates the propagation delay would be several orders of magnitude smaller then the requisite speed. The trigger pulse only needs to be slightly larger than the loop delay around the cross-coupled NAND gates (about twice the average propagation delay of the NAND gates alone). With this is mind, speed optimization techniques such as transistor ordering or progressive sizing, which are both used to drain (or load) internal and loading capacitances quickly, are not needed. In our case capacitances would have been drained through the pull-down networks of the NAND

gates, since series connections define the critical input paths. As for the question of synchronization, our register needs only to be set/reset asynchronously.

With all of this in mind we design a simple asynchronous SR-Latch in circuit Figure 2.1.2.2A. The latch simple consists of two cross-coupled minimally sized NAND gates. We can see from the logic table in Figure 2.3.2B that we have the three basic memory functions we need; *hold*, *reset*, and *set*. We can see that the SR-Latch is only triggered during negative-going pulses and holds it state when both inputs are one. So it operates using negative logic. The design calls for 2 NAND gates; resulting in a total of 8 transistors.

Static SR-Latch Schematic

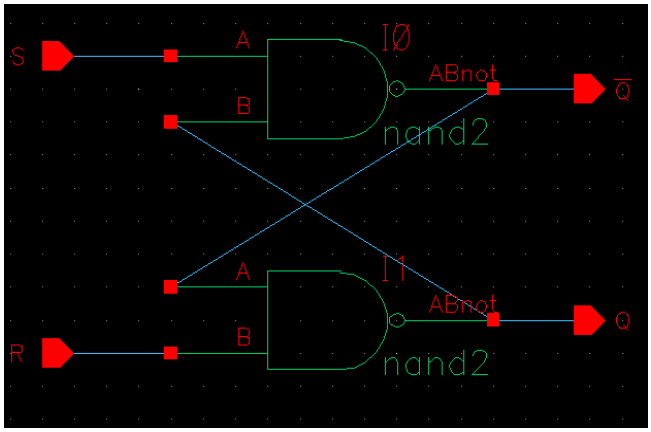


Figure 2.1.2.2A

SR-Latch Truth Table

S	*R*	*Q*	$\overline{Q}$	*Function*
0	0	1	1	*Forbidden*
0	1	0	1	Re *set*
1	0	1	0	*Set*
1	1	*Q*	$\overline{Q}$	*Hold*

Figure 2.1.2.2B

2.1.2.3 Simulation results

The simulation proved to be to have the function of the truth table in figure 2.1.2.2B. Also note that the load used was the inverter gate. We predicted the propagation to be about double that of the NAND gate. From simulation it turned out to be *580ps* for both reset and set functions, which makes sense that they are identical. The NAND gates had an average propagation delay of *220ps*, which are a little less than half. So the average propagation delay of the SR-Latch is more than double (but close) then that of the NAND gates, which also makes sense because now the load has an extra NAND gate on it.

Static SR-Latch Simulation

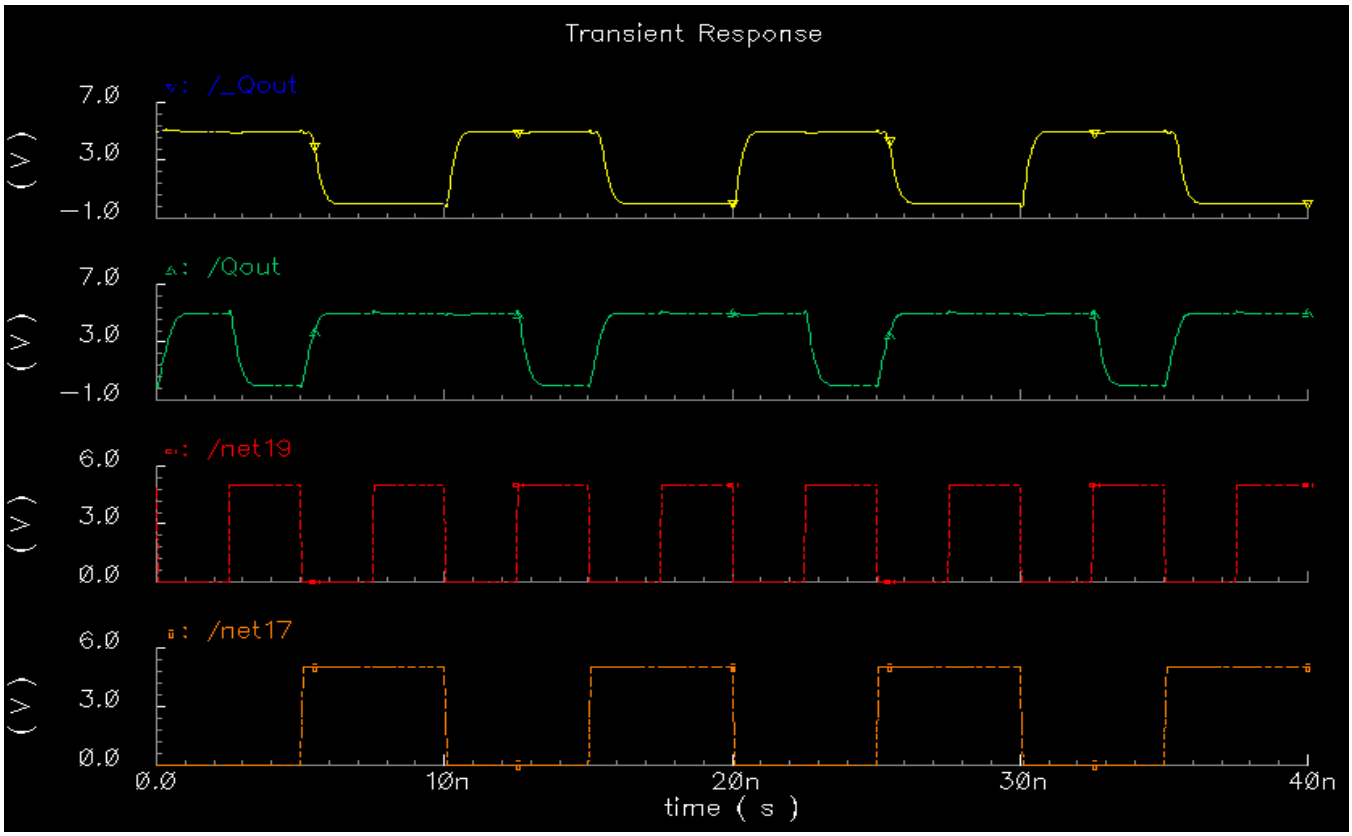


Figure 2.1.2.3(net17 Set, net19 Reset)

2.1.2.4 Layout

The layout of the SR-Latch is shown in Figure 2.1.2.4. It was simply the minimally sized NAND gates sharing their *VDD* lines. The size is 22.8µm x 30µm.

SR-Latch Layout

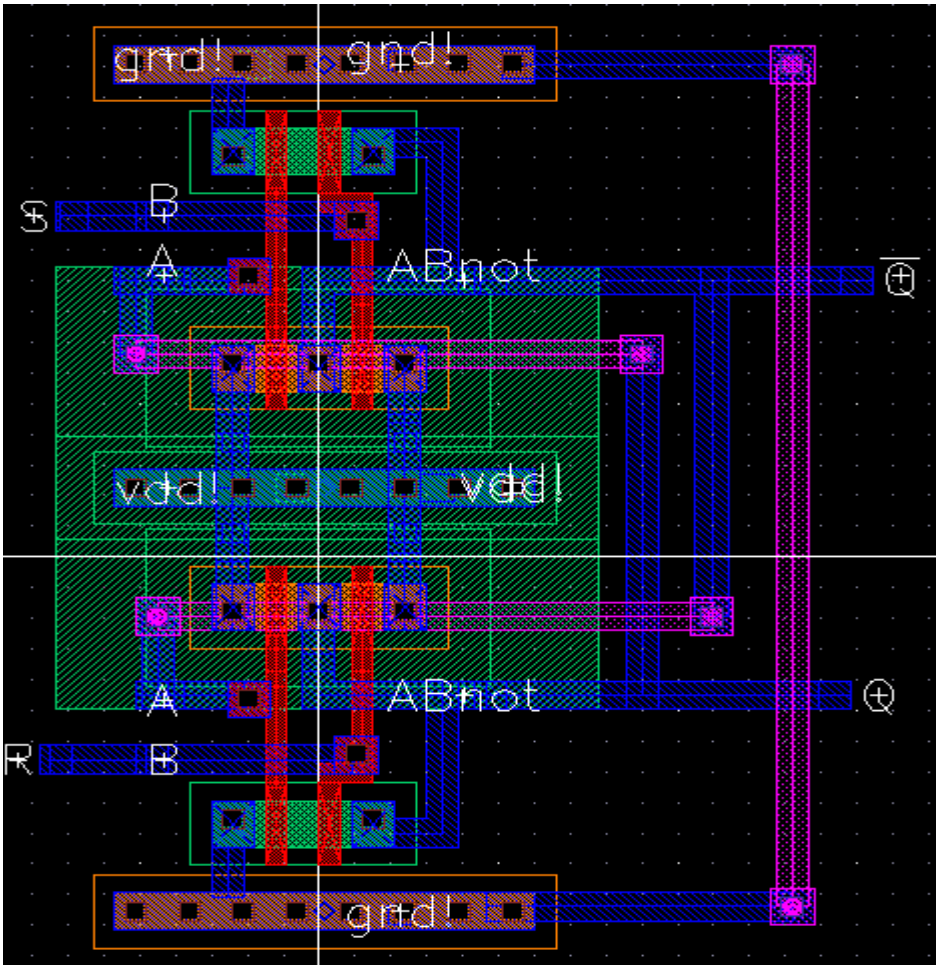


Figure 2.1.2.4

2.1.3 D Flip-Flop

2.1.3.1 Description

For the AutoRanger circuit we require two D Flip-Flops which will be used for generating a *Too_High* or *Too_Low* output signal. These signals originate from the counter when the count goes out of the specified range.

The D (Delay) Flip-Flop is a bistable multivibrator used ubiquitously throughout digital electronics to simply delay the value of its input. Effectively it could be used as a latch to hold one bit of data. As with the SR-Latch, the D Flip-Flop is used in static memories, counters, timing devices and many other devices.

2.1.3.2 Design considerations

The first thing we needed to examine was the exact function needed. This Flip-Flop needed to be a positive edge triggered device with an asynchronous clear. As for speed, the circuit design calls for robust operation only within the order of hundreds of milliseconds. Again intuition prevails and tells us that a static design with minimally sized components should run several orders faster than the necessary speed. So optimization techniques were not used such as transistor ordering or progressive sizing. If we were to use transistor ordering techniques the clock line would have been considered the critical path, since it controls the speed of the circuit. As for static versus dynamic design, since all of our basic components are static, the flip-flop would also be static.

The next step was to use a configuration that would allow the device to be edge triggered. The natural choice here was the master-slave configuration, where the master is enabled (via a clock line) half of the clock cycle, while the other half of the cycle the

slave is enabled. To make it positive edge triggered you want the device to transfer data from the master to the slave on the rising edge of the clock. So the master in enabled only while the clock is low and the slave is enabled only while the clock is high. One thing to keep in mind here is the slope of your clock when it transitions. If it is too slow than both the master and slave could be active, which is dubbed a race condition. As for how the D Flip-Flop is being used, race conditions in this particular case will not effect the functioning of the overall circuit.

The final thing to consider was what configuration would be used to hold states in the master and slave part of the flip-flop. Most designs utilize the SR-Latch, which would have served just fine for our purposes. But, with the forethought of desiring to expand this design into creating a JK Flip-Flop I opted for a transmission gate/inverter design. The reasons for this can be found in the design consideration section for the JK Flip-Flop. The design calls for 2 NAND gates, 3 inverters and 4 transmission gates; resulting in a total of 22 transistors.

In Figure 2.1.3.2 we see the schematic of the D Flip-Flop. The top input line is the *CLK* line. The next two lines on the right are the *D* line and the *CLR* line respectively. One can see that the third transmission gate from the left serves as the bridge between the master and slave portions of the circuit.

D Flip-Flop Schematic

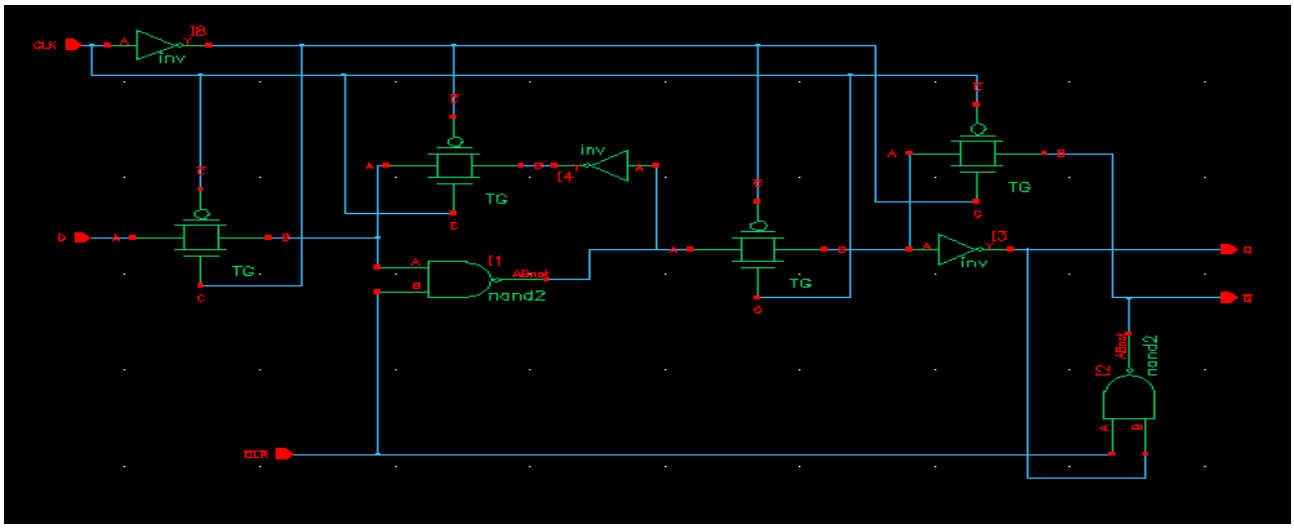


Figure 2.1.3.2

2.1.3.3 Simulation results

The simulation shows that the D Flip-Flop design functions correctly. One can see the output only changes on the positive edge of the clock. One can see that the asynchronous *CLR* also works correctly. It was found that propagation delays for data to set in the master was around *700ps* and for the slave it was around *750ps*. Since we are driving the clock so slow, these small set-up and hold times are more than sufficient to insure proper operation. The output load for the simulation was an inverter.

D Flip-Flop Simulation

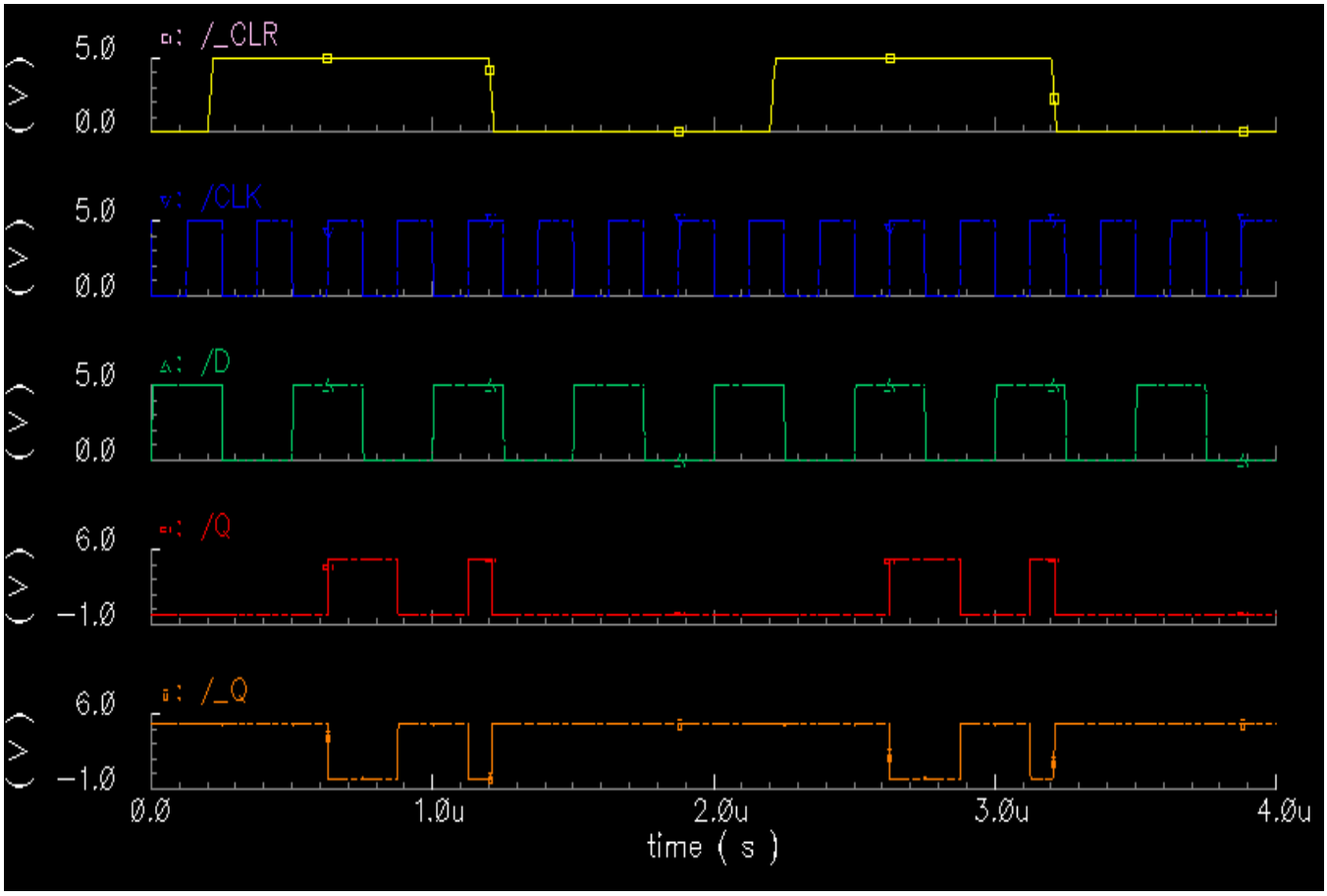


Figure 2.1.3.3

2.1.3.4 Layout

This layout was centered on using *VDD* and *GND* bus lines. Otherwise it was just connecting the inputs and outputs correctly. The Layout of the D Flip-Flop is shown in Figure 2.1.3.4. The size is 17.250µm x 49.350µm.

D Flip-Flop Layout

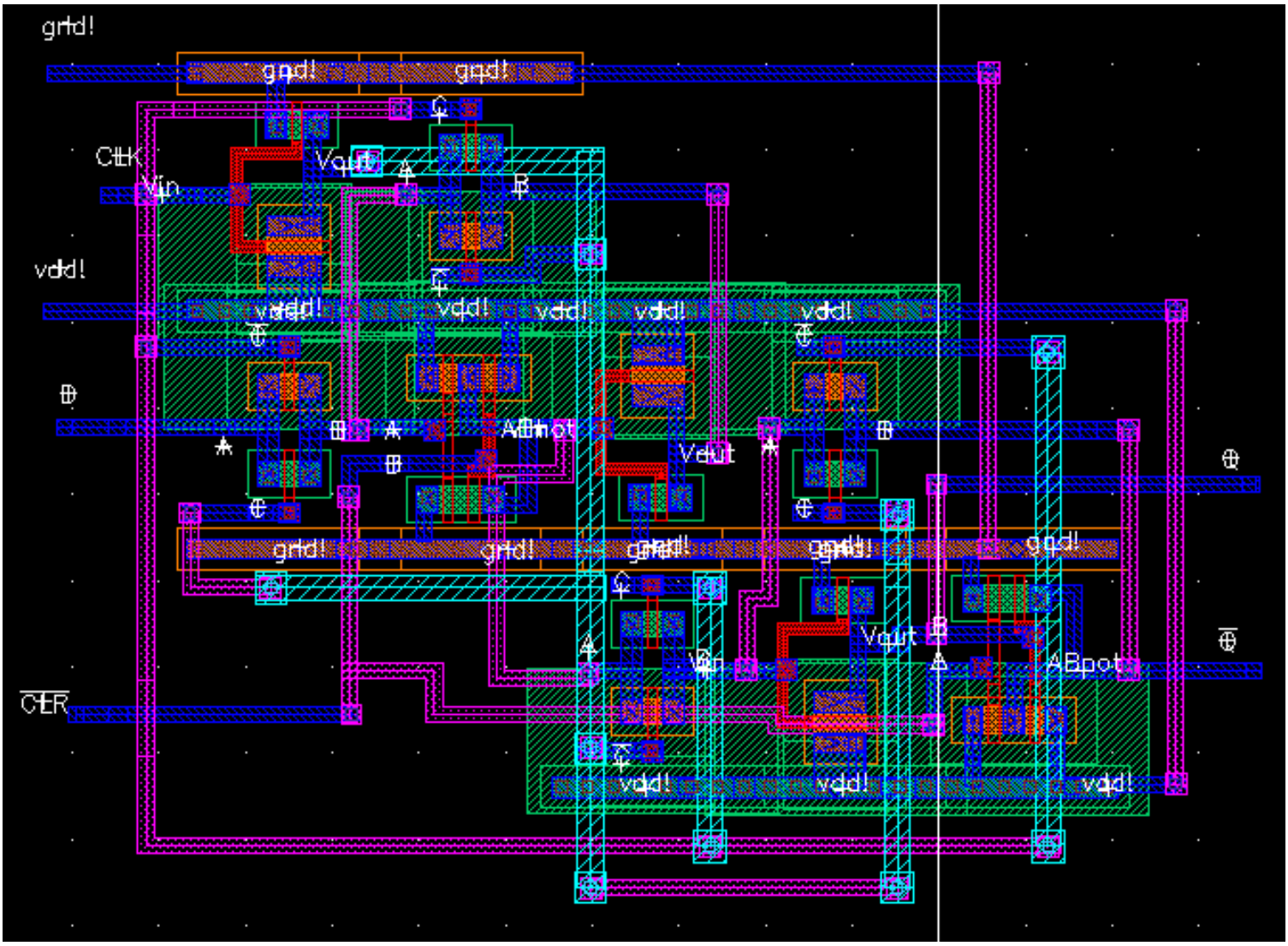


Figure 2.1.3.4

2.1.4 JK Flip-Flop

2.1.4.1 Description

For the Up/Down counter circuit we require three JK Flip-Flops which will be used for generating the three bits off the clock. The JK Flip-Flop is a bistable multivibrator in much the same way as the SR-Latch. The main difference is that it resolves the forbidden state (both inputs being high) by utilizing feedback and creating a toggle function. This toggle function comes in very handy when you want to create a counter. As with the SR-Latch and D Flip-Flop, the JK Flip-Flop is used in static memories, counters, timing devices and many other devices.

2.1.4.2 Design considerations

This Flip-Flop needs to be a positive edge triggered device with an asynchronous clear so we can reset the counter. Again, intuition prevails and tells us that a static design with minimally sized components should run several orders faster than the necessary speed. If we were to use optimization techniques such as transistor ordering techniques the clock line would be the critical path as it was in the D Flip-Flop.

From here all I needed to do was to expand the D Flip-Flop to create the JK Flip-Flop. It has the edge triggering and master/slave approach I desired so there was no need to create a whole new circuit. The critical thing to keep in mind here is the slope of your clock when it transitions. If it is too slow than both the master and slave could be active, which could create an irretrievable race condition. For example if I were in the *toggle* mode the feedback would loop around an undetermined amount of times leaving you in an undetermined state. This would certainly be a malfunction.

The final thing, which I touched upon in the D Flip-Flop section, was I used transmission gates, while most designs utilize the SR-Latch. There is an inherent ones-catching problem with the SR-Latch in a master/slave configuration, even though several books consider this design completely edge triggered. For example if the counter is in *count-up* mode and the logic feeding the Flip-Flop says to toggle then I know on the next edge I will toggle. But problem comes when the next edge is a *count-down* pulse. The logic will then say we don't want to toggle this bit, but the state is already caught in the master part of the flip-flop. Using transmission gates eliminates this problem. The design calls for 5 NAND gates, 9 inverters and 4 transmission gates; resulting in a total of 46 transistors.

In Figure 2.1.4.2 we see the schematic of the JK Flip-Flop. The top two input lines are the *J* and *K* lines respectively. The next two lines on the right are the *CLR* line and the *CLK* line respectively. One can see that the third transmission gate from the left serves as the bridge between the master and slave portions of the circuit. Also notice the feedback loop from the output portion to the input logic. This helps create the toggle function.

JK Flip-Flop Schematic

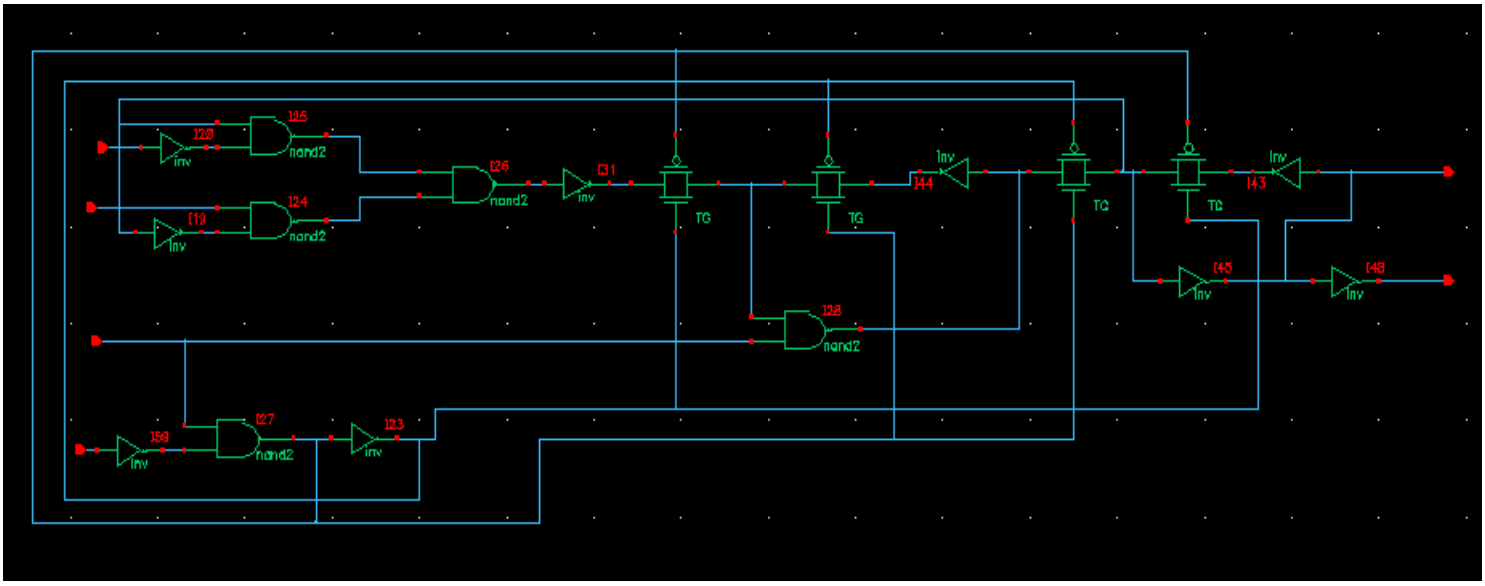


Figure 2.1.4.2

2.1.4.3 Simulation results

The simulation shows that the JK Flip-Flop design functions correctly. One can see the output only changes on the positive edge of the clock. One can see that the asynchronous *CLR* also works correctly. It was found that propagation delays for data to set in the master was around *1.5ns* and to transfer out of the slave was around *1ns*. The output load for both outputs was an inverter.

JK Flip-Flop Simulation

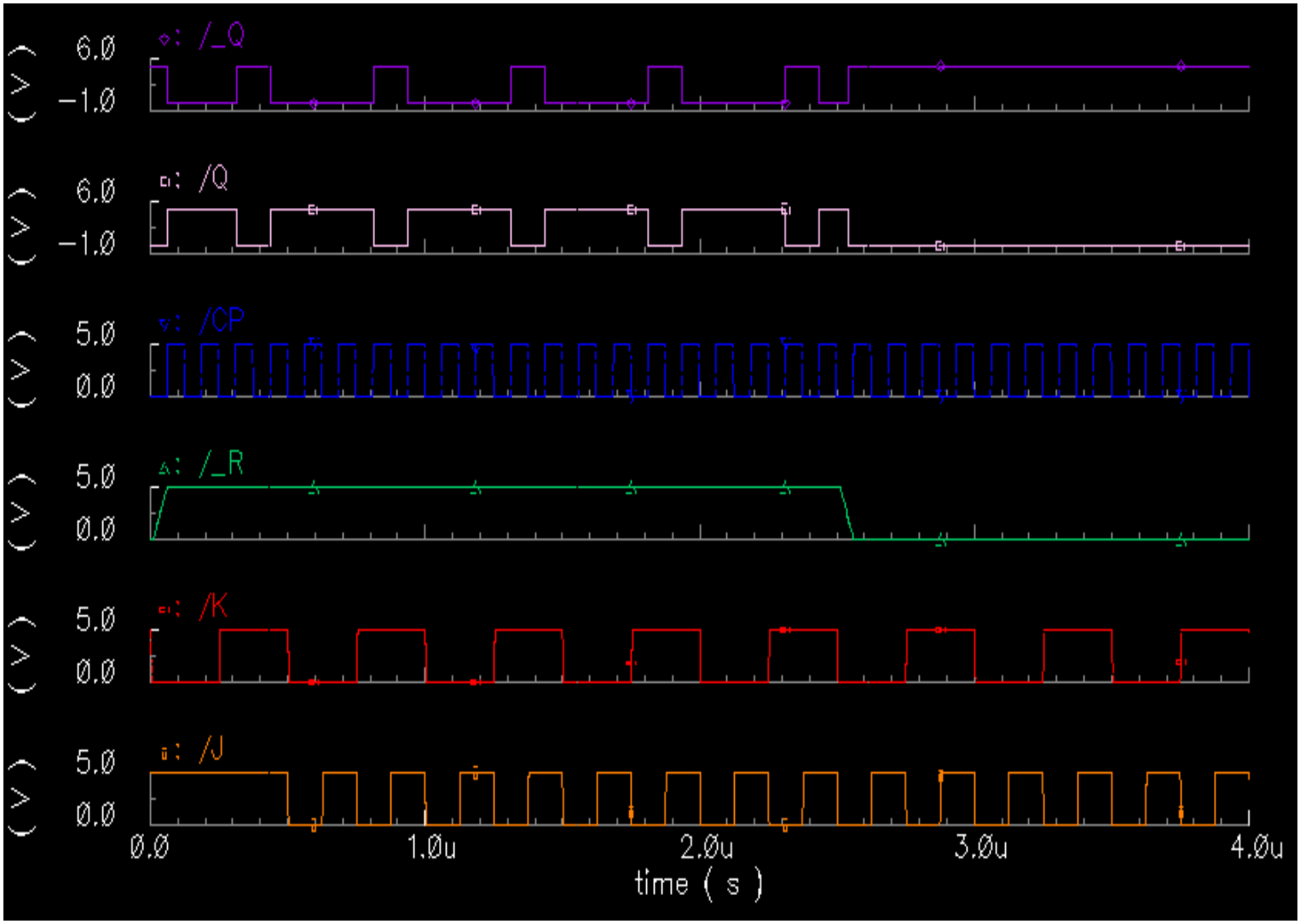


Figure 2.1.4.3

2.1.4.4 Layout

Again, as with the D Flip-Flop, this layout was centered on using *VDD* and *GND* bus lines. Otherwise it was just connecting the inputs and outputs correctly. The layout of the JK Flip-Flop is shown in Figure 2.1.4.4. The size is 109.8µm x 44.250µm. The top two input lines are the *J* and *K* lines respectively. The next two lines on the right are the *CLK* line and the *CLR* line respectively

JK Flip-Flop Layout

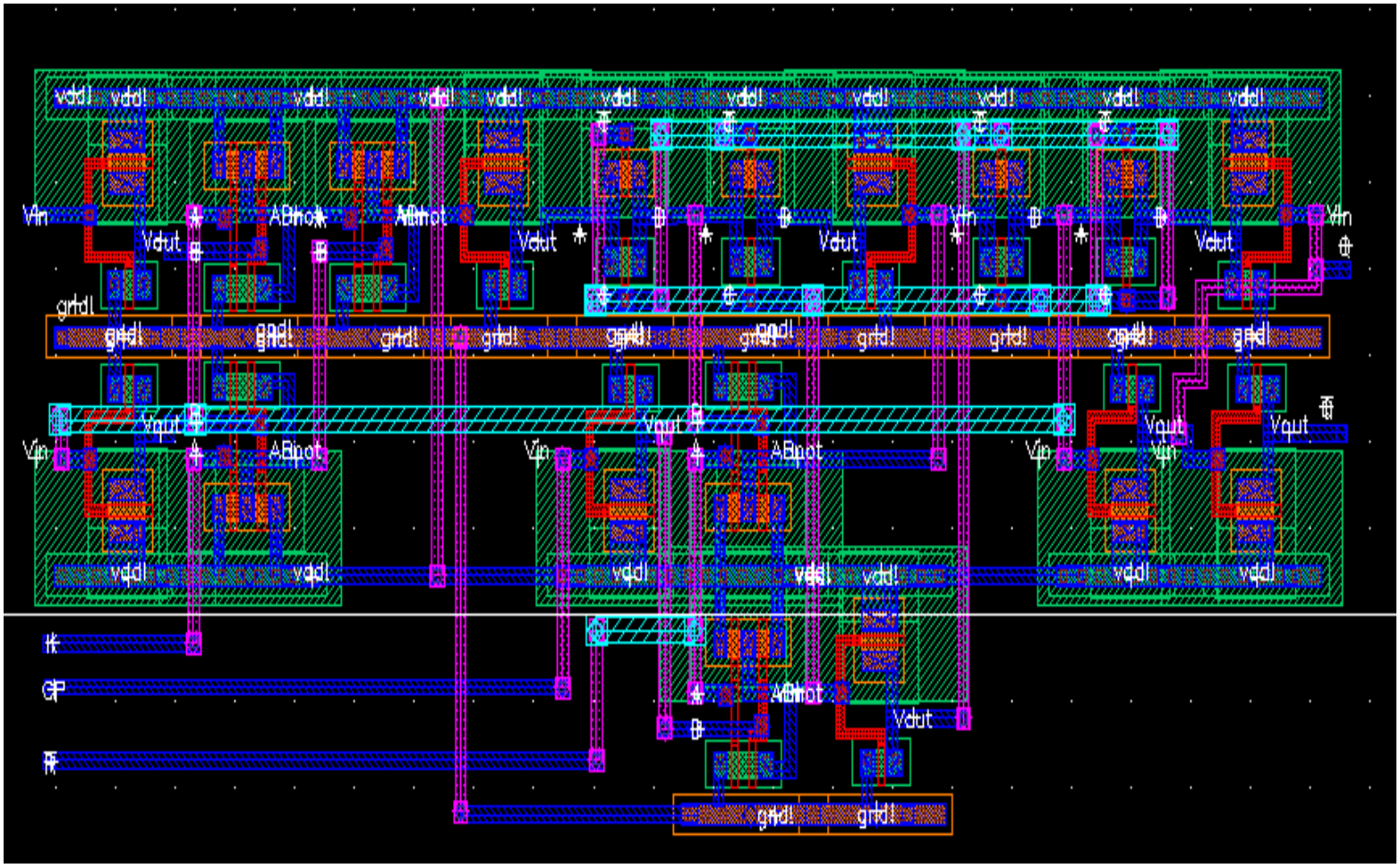


Figure 2.1.4.4

2.1.5 UP/Down Counter

2.1.5.1 Description

The Up/Down counter circuit is simply a three-bit synchronous counter that will either count up or down. This circuit gives the current output range we are in for the AutoRanger circuit. While both the *CU* (Counter UP) & *CD* (Count Down) lines are held high then the counter holds its state. When one of the lines is pulsed low then we either count up or down. Counters are used for time-out circuits, frequency dividers, counters (pretty obvious), and many other devices.

2.1.5.2 Design considerations

The Up/Down counter needs to be a positive edge triggered device with an asynchronous clear so we can reset the counter. The count only changes when one of the inputs is pulsed low. When it returns high again the count changes. This also runs slow so no need for speed optimization. The three bits will be counted using three JK Flip-Flops. The design calls for 1 SR-Latch, 5 two-input NAND gates, 2 three-input NAND gates, 1 inverter and 3 JK Flip-Flops; resulting in a total of 180 transistors.

In Figure 2.1.5.2 we see the schematic of the counter. The top two input lines are the *CU* and *CD* lines respectively. The next input line on is the *CLR* line. The three outputs are the bit count. We have an SR-Latch in the upper left corner to keep track of counting up or down. You will notice the first bit always toggles whether we count up or down. The next two bits depend on the lower order bits and the counting mode we are in. The logic basically forces the bit to either *toggle* or *hold* on a clock pulse. Also notice all the bits change at the same time.

Up/Down Counter Schematic

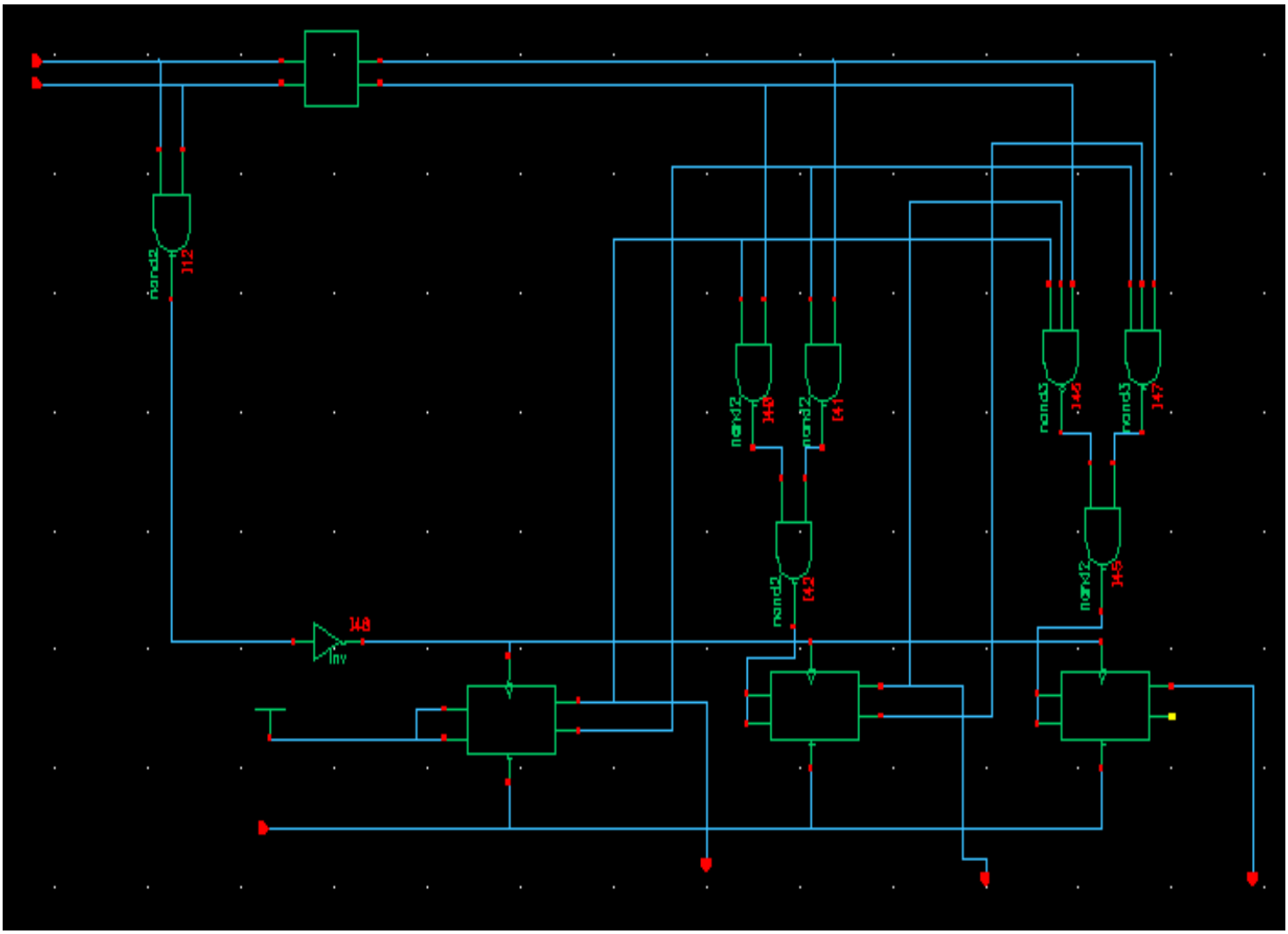


Figure 2.1.5.2

2.1.5.3 Simulation results

The simulation shows that the counter design functions correctly. One can see the output only changes on the positive edge of the *CU* or *CD* pulse. One can also see that the asynchronous *CLR* also works correctly. It was found that propagation delays for logic to set up a *toggle* or *hold* was around *1.1ns*. The time from the positive edge of the *CD* or *CU* pulse till the counter changed count was around *2ns*. The output load for all three outputs was an inverter. Notice all the output bits change at the same time.

Up/Down Counter Simulation

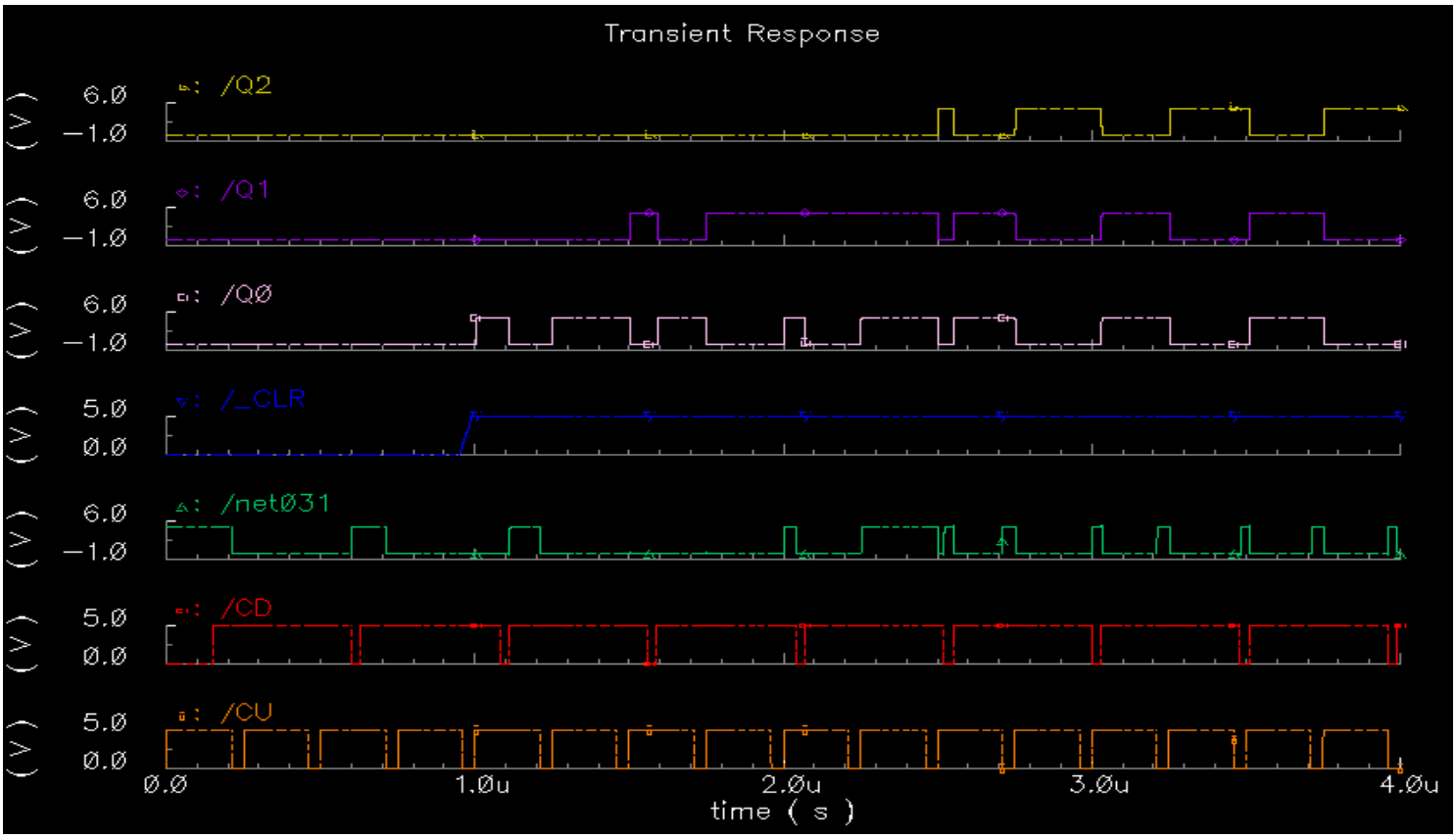


Figure 2.1.5.3

2.1.5.4 Layout

The layout of the Up/Down counter is shown in Figure 2.1.5.4. The size is 188.25µm x 164.25µm. The top two input lines are the *CU* and *CD* lines respectively. The fifth line down on the right is the *CLR* line. The last line down on the right is *bit1*. The other two lines on the right are *bit0* and *bit2*. The layout has the three JK Flip-Flops in the middle. On the top right you can see the SR-Latch. On the sides of the JK Flip-Flops are *the toggle/hold* logic controllers.

Up/Down Counter Layout

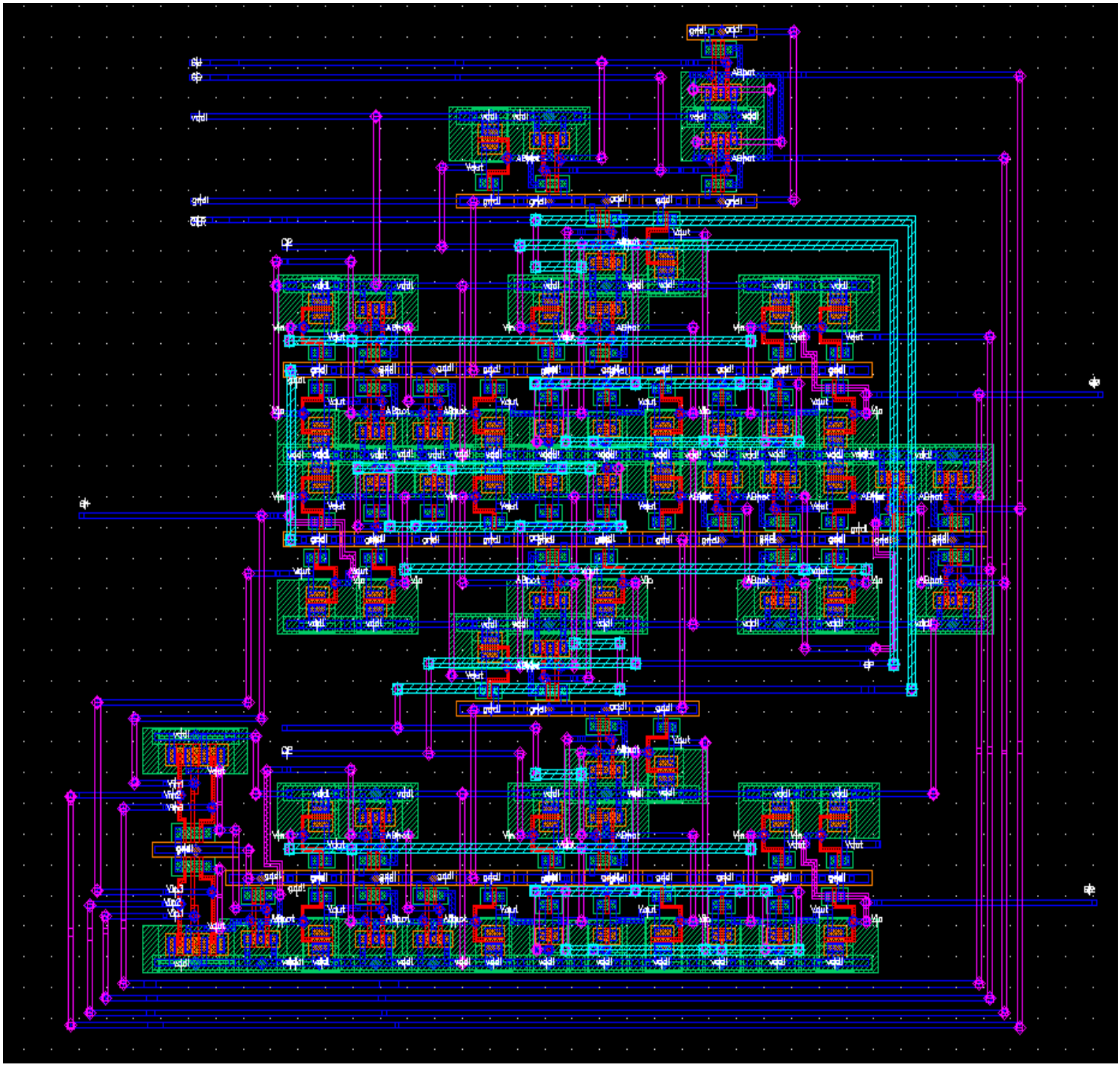


Figure 2.1.5.4

2.2 Encoder/Decoder

2.2.1 Basic Component & Design Considerations

The encoder and decoder consist of inverters and NAND gates. The different types of NAND gates required for the designs are 2,3,4, and 5 input NAND gates. The transistors in the gates are minimally sized considering a mobility ratio of 2. So the mobility of the electrons is considered to be 2 times bigger than the one of the holes (suggested by MOSIS), which then gave rise to (W/L)p:(W/L)n ratio of 2. The width to length ratio of the PMOS devices, (W/L)p, is chosen to be 5=3um/600nm where the width is 3um and the length is 600nm. On the other hand, the width to length ratio of NMOS devices, (W/L)n, is 2.5=1.5um/600nm where the width is 1.5um and the length is 600nm.

2.2.1.1 Inverter

The inverter transistors are sized as follows:

(W/L)p = 3um/600nm

(W/L)n = 1.5um/600nm

2.2.1.1 NAND Gates

The following table shows the transistors sizing for the transistors in different types of NAND gates. The values are chosen to give minimum delays under the worst switching case.

Sizing Table

	2 input NAND	3 input NAND	4 input NAND	5 input NAND
(W/L)p	3um/600nm	3um/600nm	3um/600nm	3um/600nm
(W/L)n	3um/600nm	4.5um/600nm	5um/600nm	7.5um/600nm

Figure 2.2.1.1A

In this section, the layouts of only the 4 and 5 input inverters are given since they are unique to the encoder.

4 input NAND gate layout

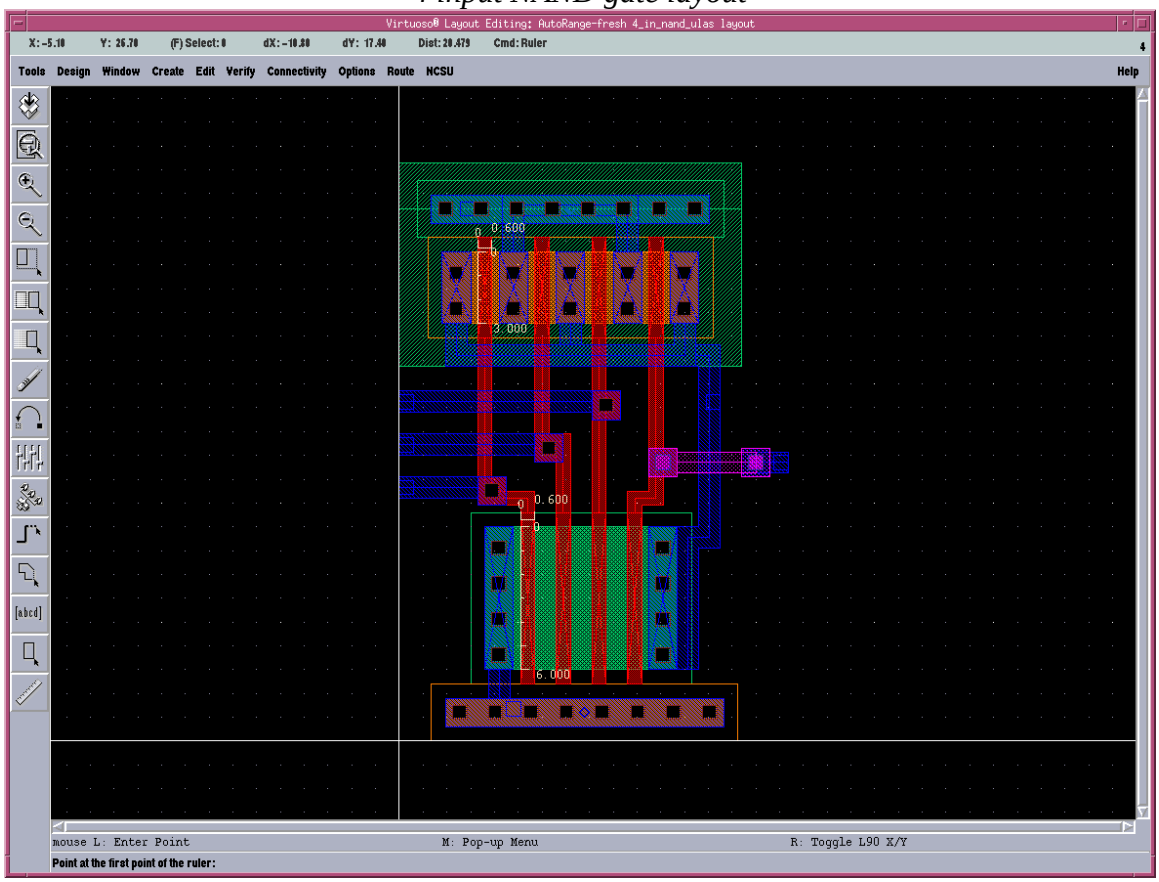


Figure 2.2.1.1B

5 input NAND gate layout

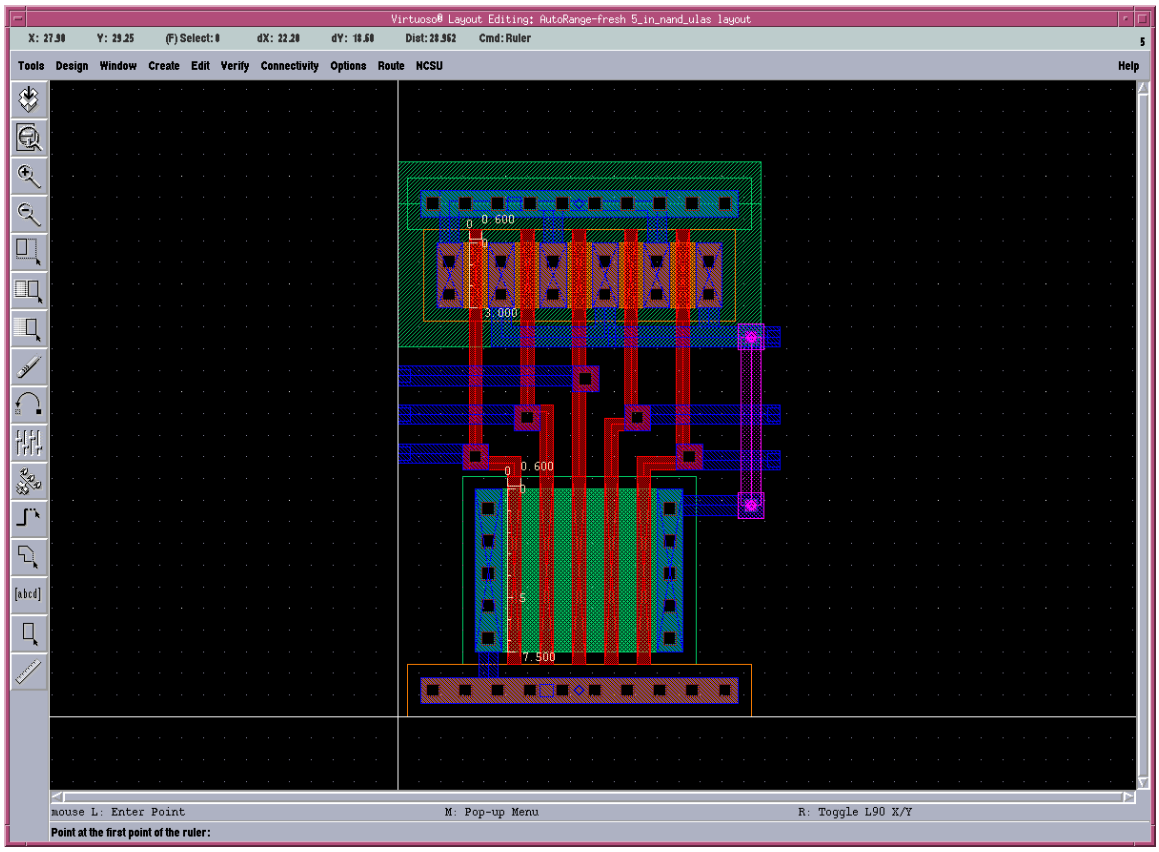


Figure 2.2.1.1C

2.2.2 Encoder

2.2.2.1 Description and Design Considerations

An encoder has 2^n (or less) input lines and n output lines. The output lines generate the binary code for the 2^n input variables. The encoder designed for the

AutoRanger Circuit is a decimal to binary encoder. This means that the encoder has 10 input lines, one for each of the 10 digits, and 4 output lines that generate the corresponding binary number. The encoder is constructed with NAND gates and inverters, and the truth is given below:

Encoder Truth Table

Inputs										**Outputs**			
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**	**D9**	**D10**	**Vout0**	**Vout1**	**Vout2**	**Vout3**
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	0	1	X	0	0	1	0
0	0	0	0	0	0	0	1	X	X	0	0	1	1
0	0	0	0	0	0	1	X	X	X	0	1	0	0
0	0	0	0	0	1	X	X	X	X	0	1	0	1
0	0	0	0	1	X	X	X	X	X	0	1	1	0
0	0	0	1	X	X	X	X	X	X	0	1	1	1
0	1	X	X	X	X	X	X	X	X	1	0	0	0
1	X	X	X	X	X	X	X	X	X	1	0	0	1

Figure 2.2.2.1

The X’s in the table show don’t care conditions. The design is a high logic design. The high logic voltage level is 5V and the low logic voltage level is 0V. The encoder assumes that only one input can be equal to 1 at any given time; otherwise the circuit has no meaning. The circuit has 10 inputs and can have 2^10 input combinations. Only 10 of these inputs have any meaning. The other inputs are don’t care conditions.

2.2.2.2 Simulation Results

Simulation results and the test circuit schematic are given below. The encoder is tested by connecting 10 square wave sources of different periods to each input. Each square wave source has half the period of the previous one (the first has 2 sec., the second 1 sec., etc.) this assures that many different input combinations can be tested. Keeping in mind that we have many don’t care conditions the results are checked and found out to be

true. It may be seen that there are some small spikes at some of the outputs, meaning that the output goes high for a very short time when it should remain low. Since the speed is not so critical for the whole design, these spikes do not cause any problem in the proper working of the circuit.

Encoder Schematics

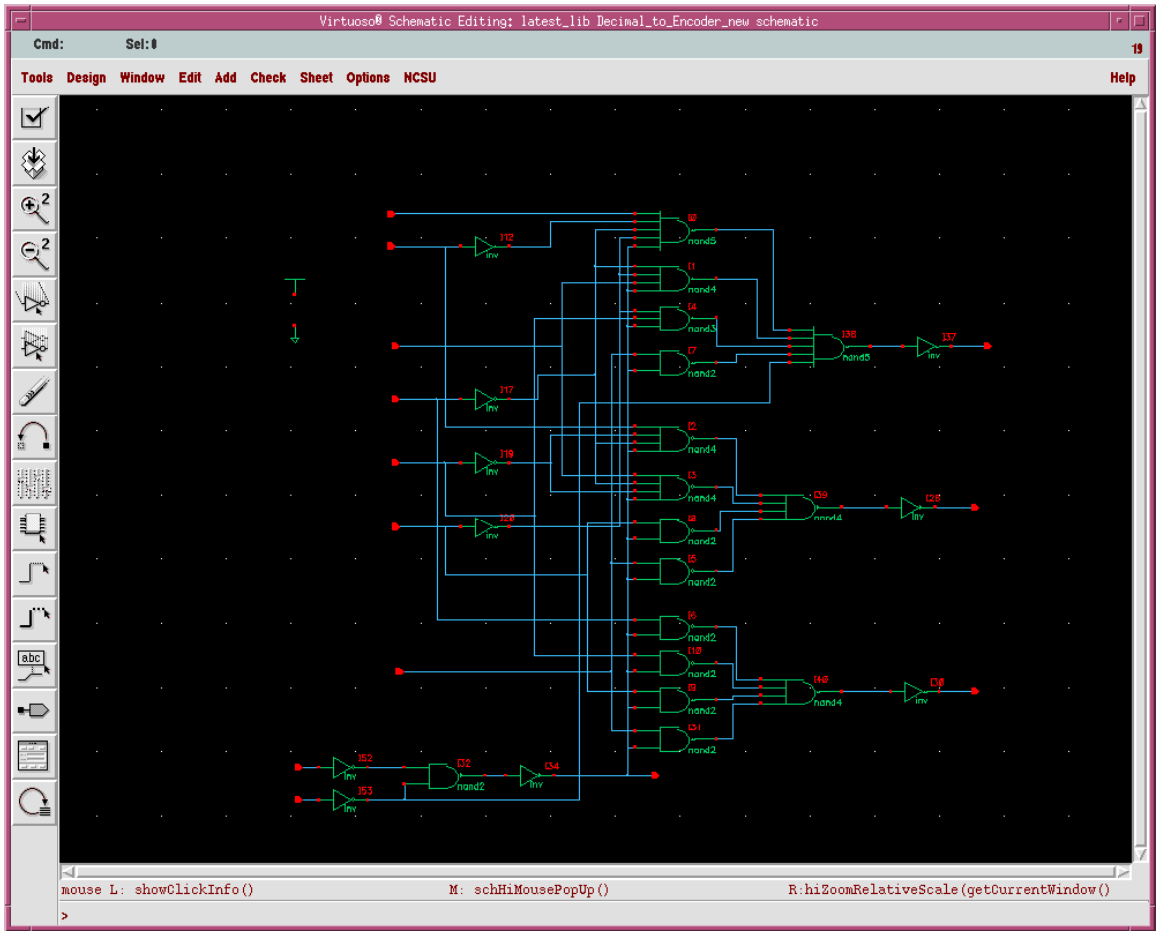


Figure 2.2.2.2A

Encoder Test Circuit

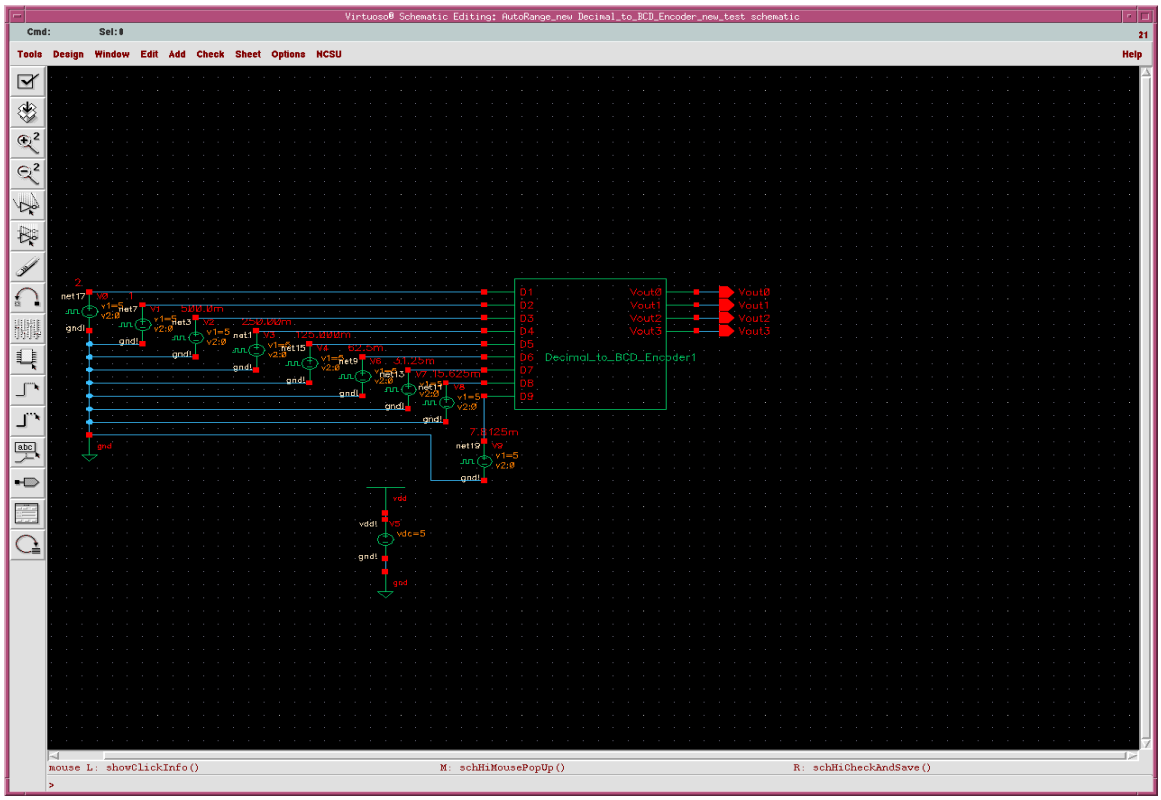


Figure 2.2.2.2B
Encoder Simulation Results

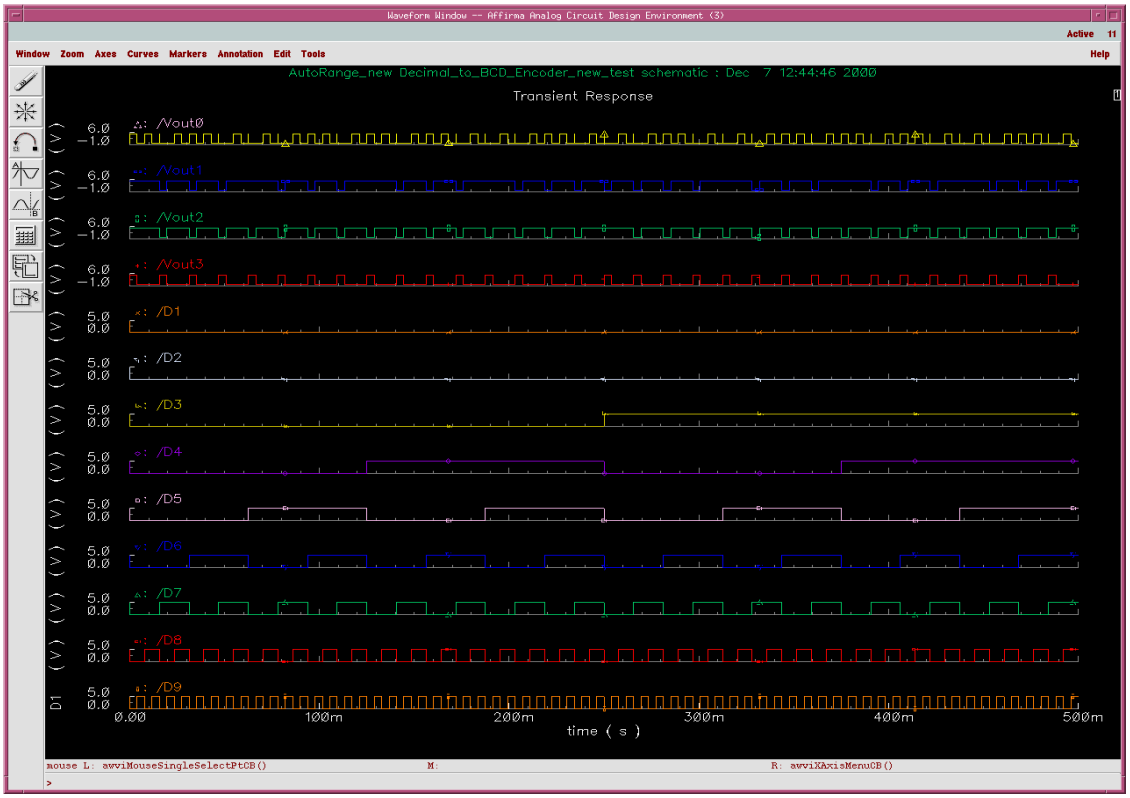


Figure 2.2.2.2C

2.2.2.3 Layout

The layout of the encoder is given below. The main design concern in the layout was an easy distribution of the inputs. Since there are many inputs, inverted inputs, and some logical functions of the inputs used many times a bus structure is used. Any inputs or logical functions of the inputs used many times are assigned a bus. The bus structure increases the design area but gives an easy to follow and ordered layout. The long bus lines do not create antenna effect since they do not go over the gates.

Encoder Layout

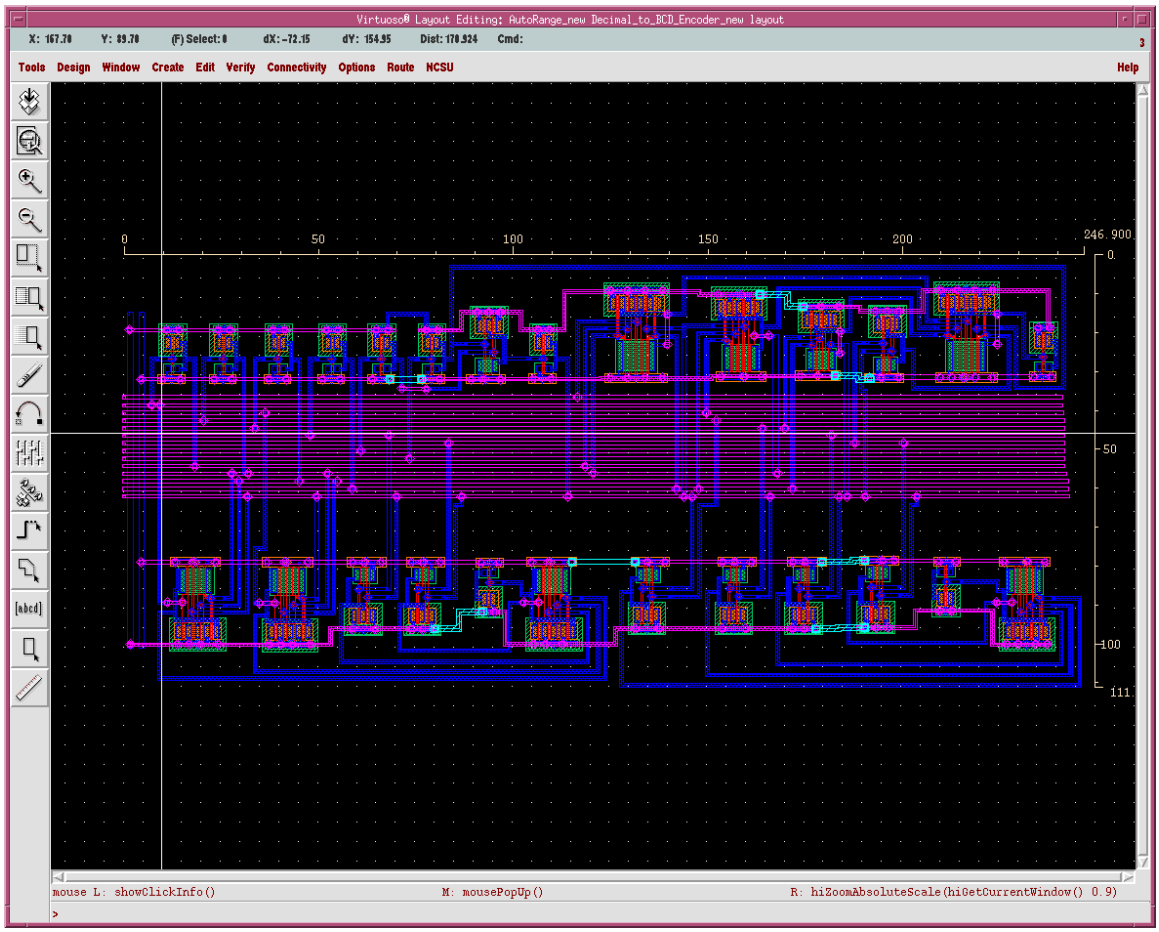


Figure 2.2.2.3

2.2.3 Decoder

2.2.3.1 Description and Design Considerations

A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique lines. If the n-bit decoded information has unused or don't care conditions, the decoder output will have less than 2^n outputs.

The decoder needed for the Autoranger Circuit is a 3-to-8 decoder. The three inputs are decoded into eight outputs, each output representing one of the minterms of the 3 input variables. The inverters at the inputs provide the complement of the inputs and each one of the NAND gates generates one of the minterms. Since this is a high logic design the output if the NAND gates are inverted. The truth table for the 3-to-8 decoder is given below:

Decoder Truth Table

Inputs			**Outputs**							
D1	**D2**	**D3**	**Vout0**	**Vout1**	**Vout2**	**Vout3**	**Vout4**	**Vout5**	**Vout6**	**Vout7**
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	1	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Figure 2.2.3.1

We can observe that the output variables are mutually exclusive because only one input can be equal to 1 at any given time. The output line whose value is equal to 1 represents the minterm equivalent of the binary number presently available in the input lines. The high and low logic voltage levels are the same as that of the encoder.

2.2.3.2 Simulations

Decoder Schematic

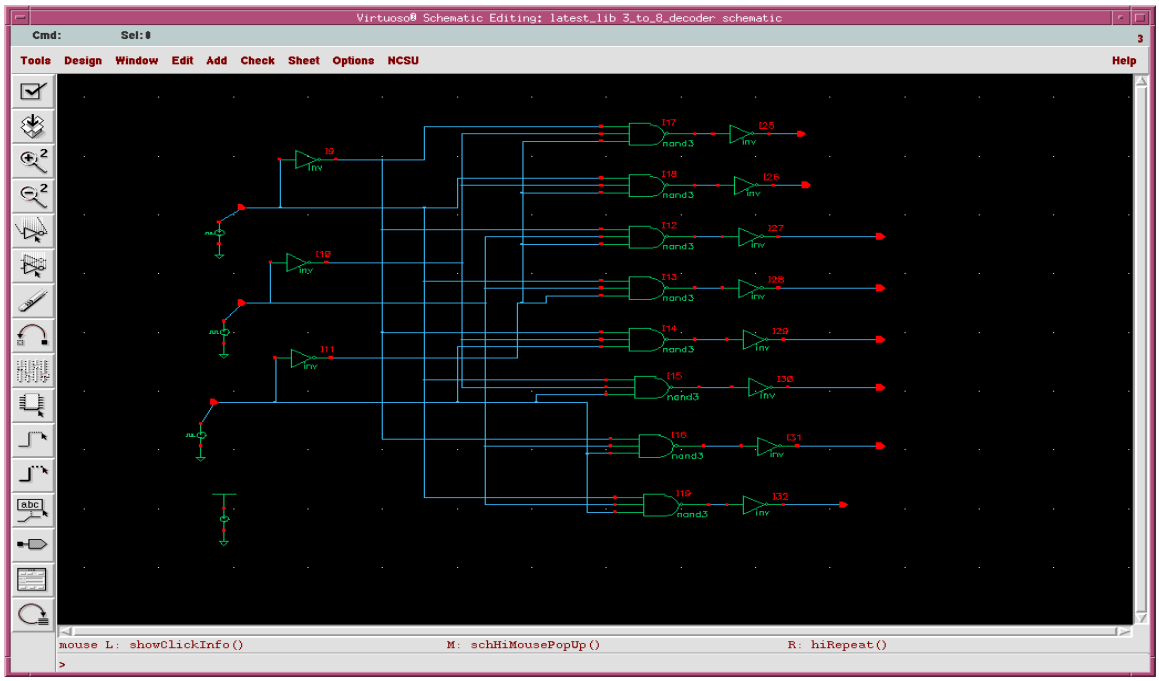


Figure 2.2.3.2A

Decoder Test Circuit

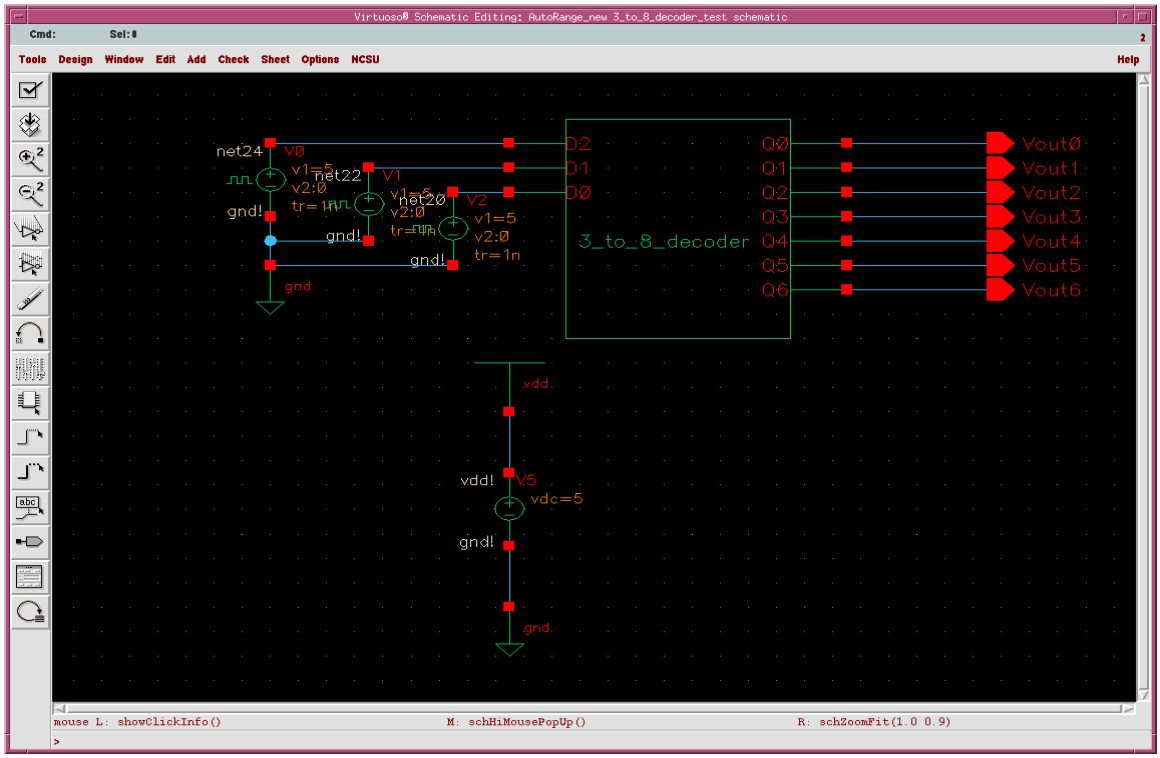


Figure 2.2.3.2B

Decoder Simulation Results

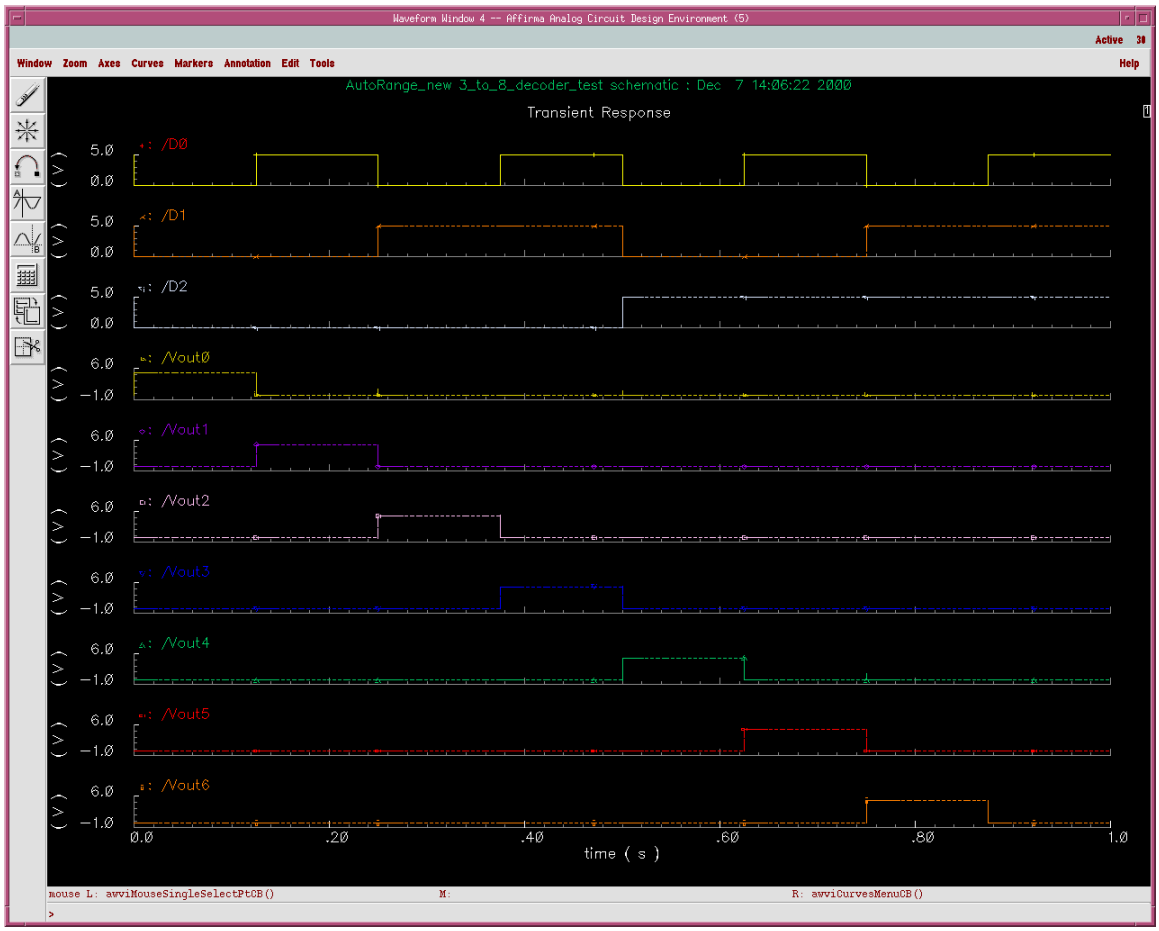


Figure 2.2.3.2C

2.2.3.3 Layout

The layout of the decoder is given below. The design considerations are exactly the same as that of the encoder.

Decoder Layout

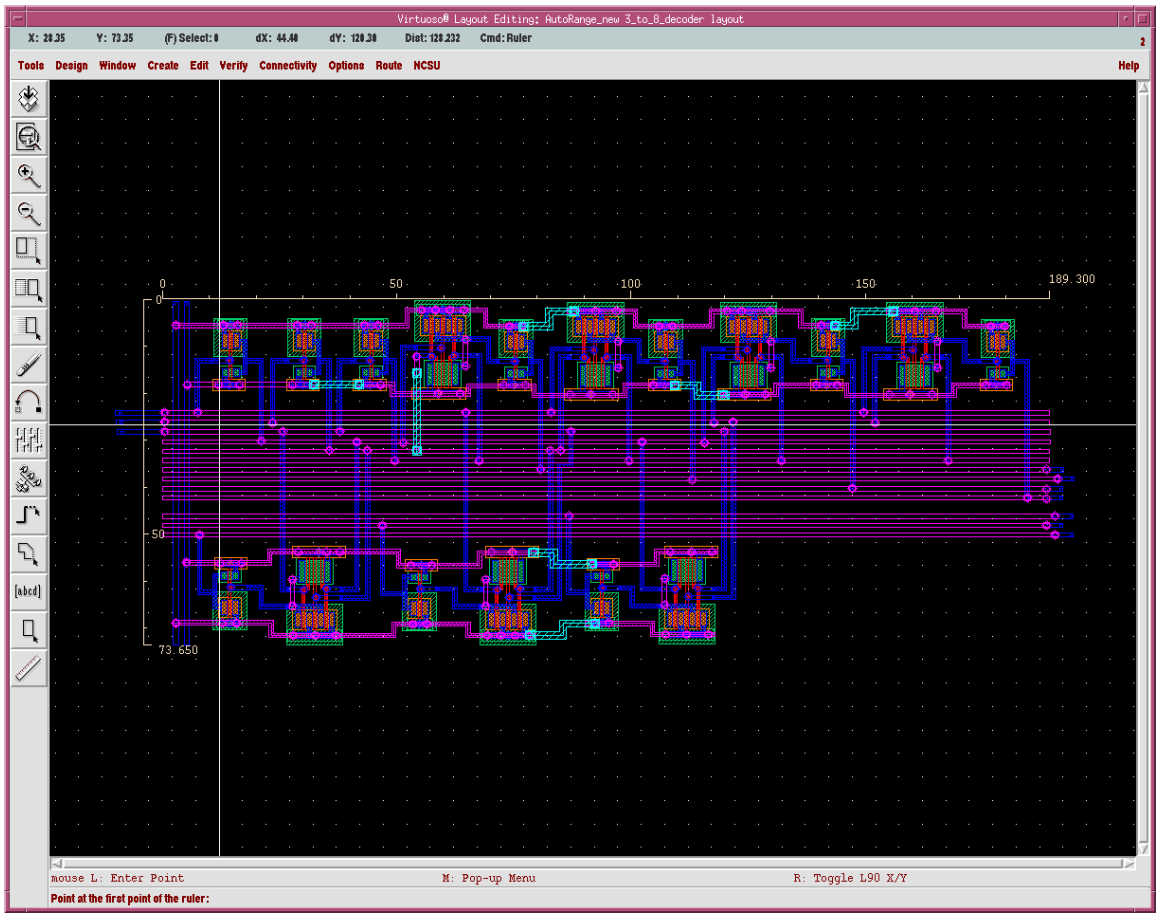


Figure 2.2.3.3

2.3 Comparator/Astable

2.3.1 Basic Component & Design Considerations

This part of the circuit is responsible for testing an input reference voltage and sending a clock pulse to one of two D flip-flops if the voltage is too high or too low. This includes all of the AutoRanger's analog design. Since the outputs will be driving logical gates, the output doesn't need to be driven very strongly. Another consideration is that the astable clock generator has to have a very slow frequency due to the ~1 second switching times of the mechanical relays which control the reference resistances. One other consideration, due to the external capacitor on the astable, is that the voltage from the gate to body of the MOSFETs cannot exceed 5 V.

2.3.2 Comparator

2.3.2.1 Description & Design Considerations

Comparator Schematic

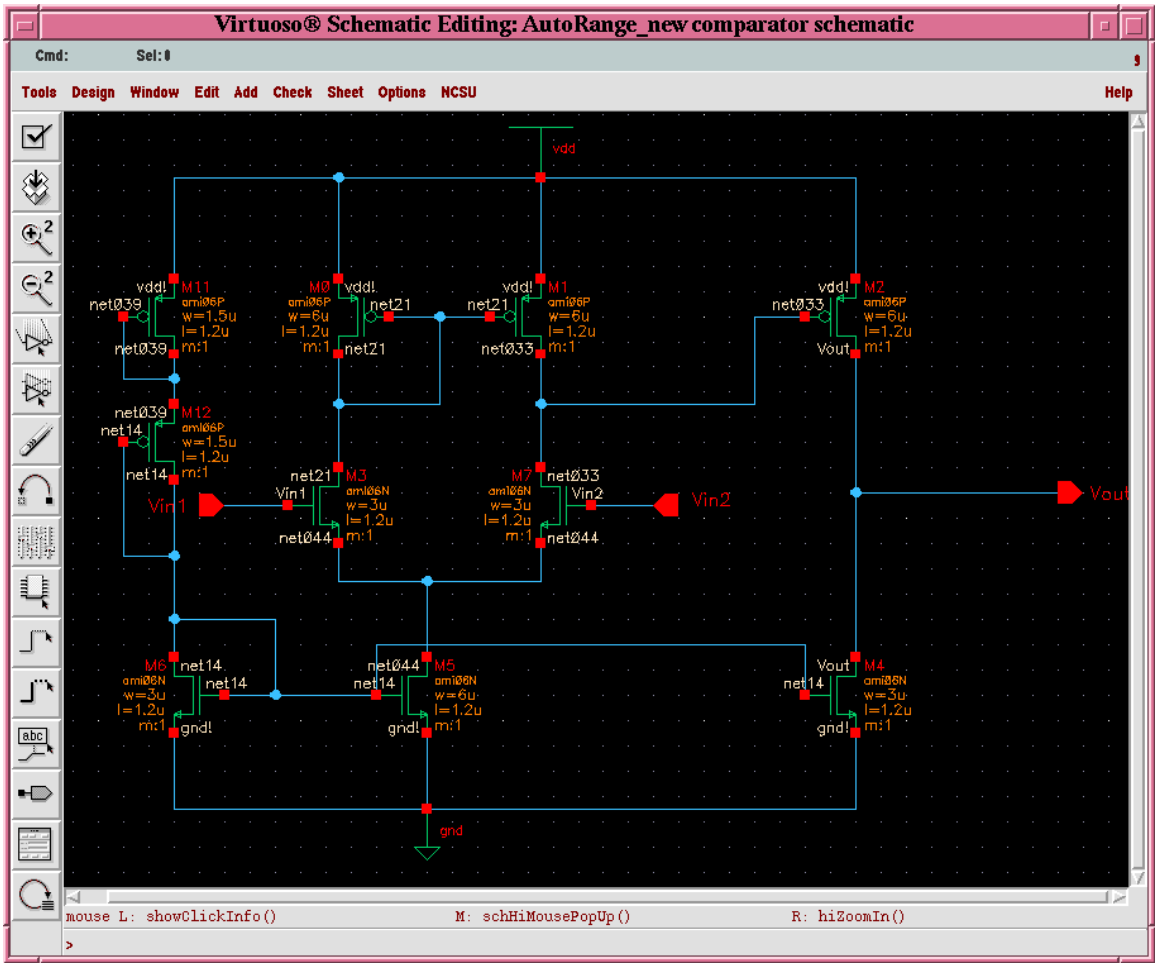


Figure 2.3.2.1

The comparator implemented here is a basic design that uses a simplified op amp configuration. It removes the op amp's standard second stage and compensation network. These are used to improve gain and push the first pole out to gain bandwidth; since it's being used as a comparator, and one in which slew rate isn't critical, neither of these features is necessary. Note that the two transistors in the upper left serve as a resistor, and provide approximately 30 μA of current through M5, which is serving as a current source for the differential pair. This saves some space on the layout. The trickiest part of the comparator design is sizing the transistors. M5 is twice the size of the two NMOSs in the differential pair, since at worst case it has to handle both M3 and M7 at maximum output. The PMOSs in the current mirror and in the output stage at the right are two times the size of the NMOSs to provide equal current output: μ_n equals approximately twice μ_p for 0.5μ technology. The MOSFETs of the output stage were also increased to double minimum size since each comparator output has to drive two other gates.

2.3.2.2 Simulation Results

Simulation of Comparator

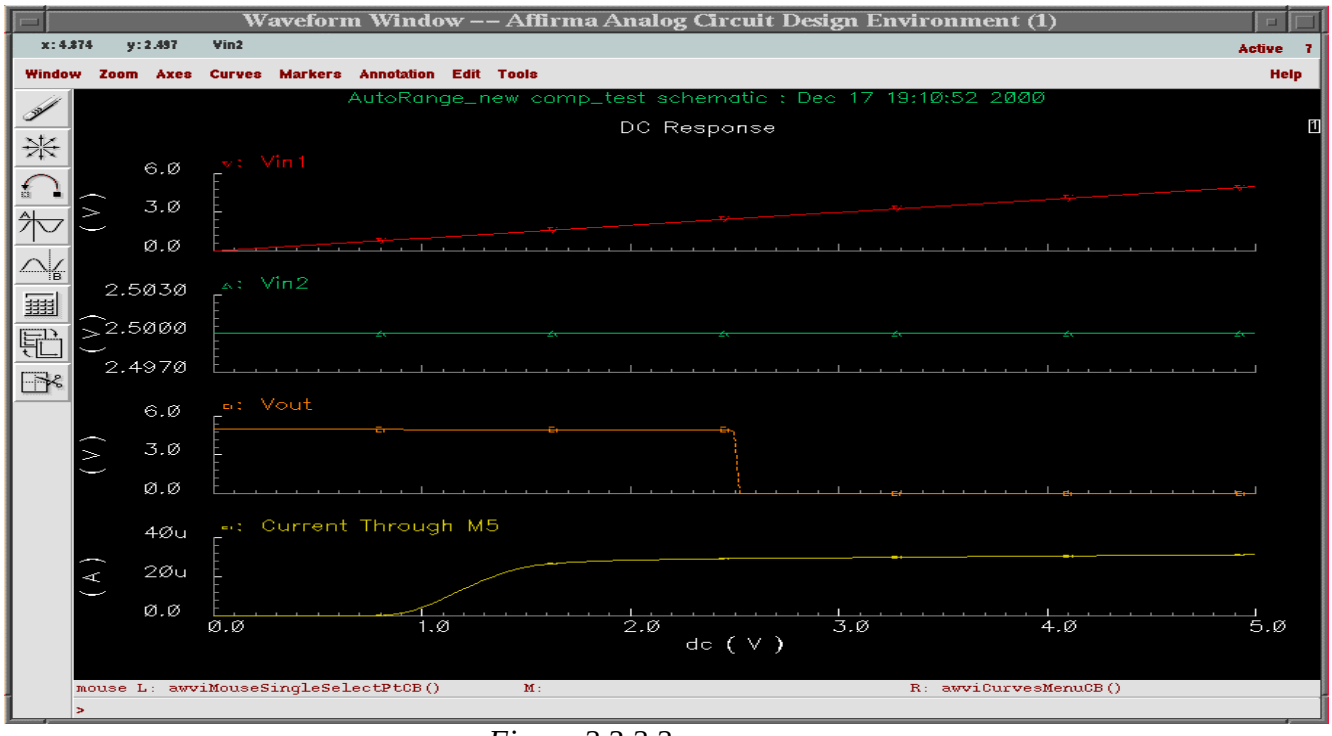


Figure 2.3.2.2

This simulation holds Vin2 at 2.5 V and swings Vin1 between 0 and 5 V. The output is exactly as anticipated-Vdd when Vin1 is below Vin2, ground when Vin1 is above. Note that once Vin1 goes above the threshold voltage, the current through M5 quickly goes to 30 µA.

2.3.2.3 Layout

Comparator Layout

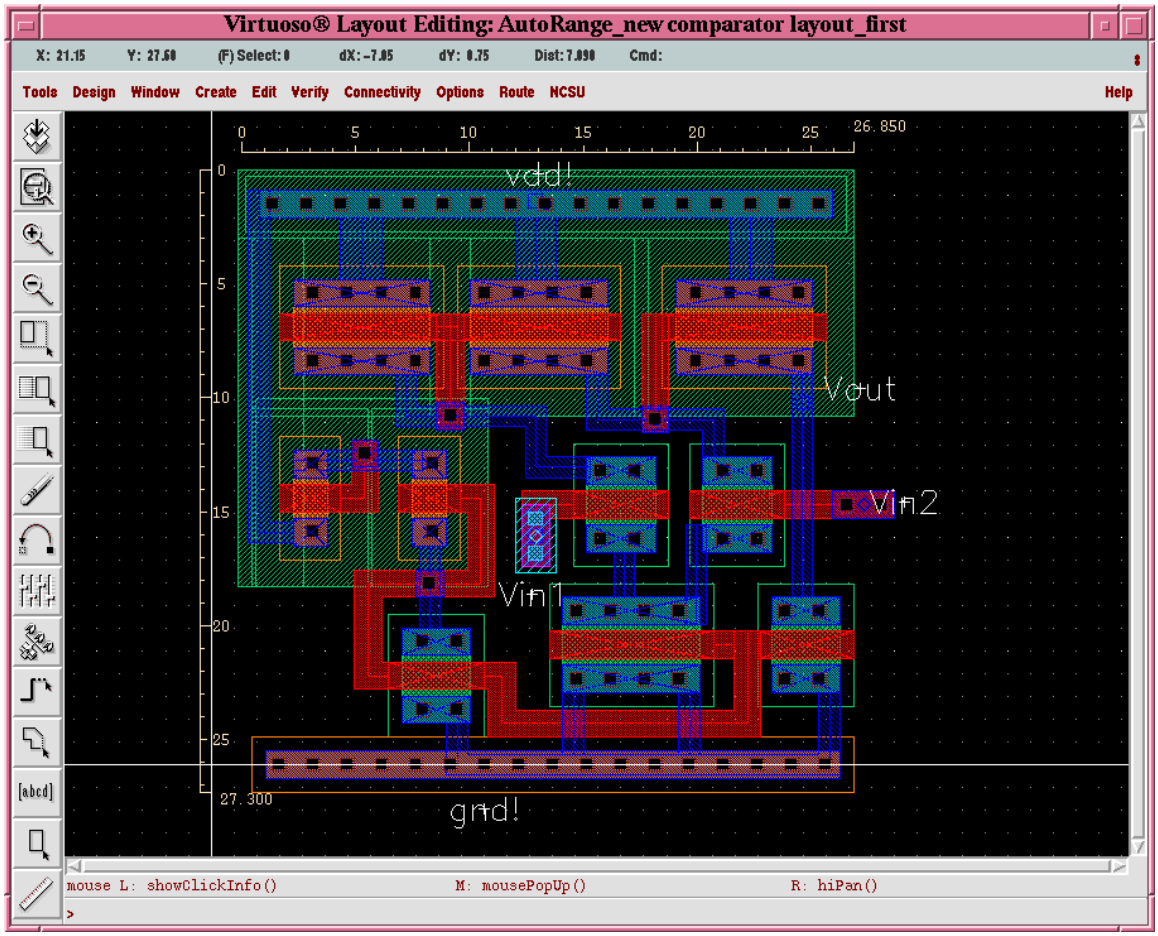


Figure 2.3.2.3

The layout is reasonably square, and has power and ground lines at the top and bottom so as to be easily integrated with other components. Some time was spent insuring that the layout was as compact as possible. Note the small size of the two PMOSs in the middle left of Figure 2.3.2.3-the equivalent resistance of approximately 380k Ω would have taken up much more room.

2.3.3 Astable Multivibrator

2.3.3.1 Description & Design Considerations

The outputs of the comparators in the board version of the AutoRanger are sent to one-shot timers, which are configured with an external RC network in order to keep putting out constant length pulses as long as the comparator outputs are high. This is functionally equivalent to an astable multivibrator. An astable requires only 1 RC network, as opposed to 2 for the one-shots, so it was decided to use an astable instead of a one-shot. The output of the astable is then ANDed with the output of the comparator to give an output that will be the output of the astable when the comparator output is high, and off when the comparator output is low.

The capacitor causes some difficulties in this circuit, shown below in Figure 2.3.3.1. Immediately after switching, the capacitor has Vdd + Vswitching over its terminals. It will be risky to have more than Vdd between the gate and body of the transistor, due to the possibility of breaking down the gate. Because of this, it was decided to connect the sources and bodies of the PMOSs in the astable inverters to 3.3 V instead of 5. These changes necessitated the creation of separate inverter gates that had the source of the PMOS connected to a pin instead of Vdd.

Astable Schematic

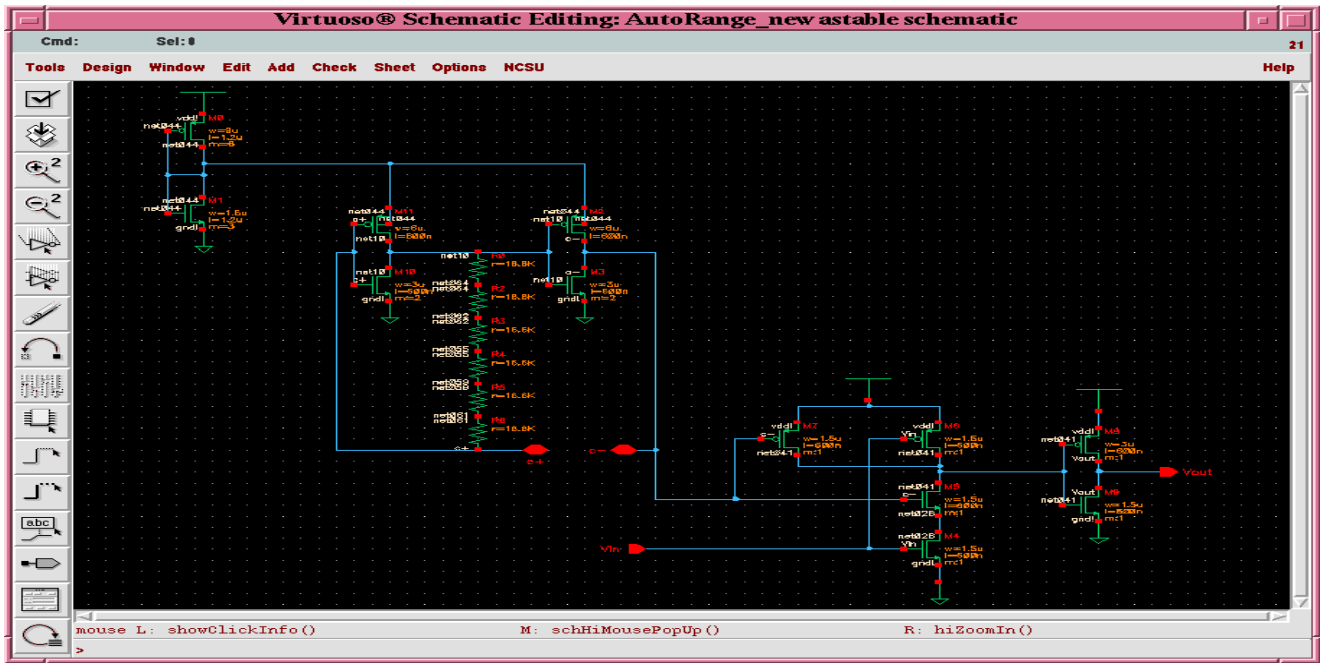


Figure 2.3.3.1

Figure 2.3.3.1 shows the astable. In the upper left is a MOSFET current divider that produces 3.3 V under a constant load. The resistor is divided up into 6 parts in order to satisfy LVS-total resistance is 99.6 kΩ. In the lower right the output of the astable is NANDed with Vin, which comes from the comparator output. This is then inverted to form an AND function.

2.3.3.2 Simulation Results

Astable Simulation

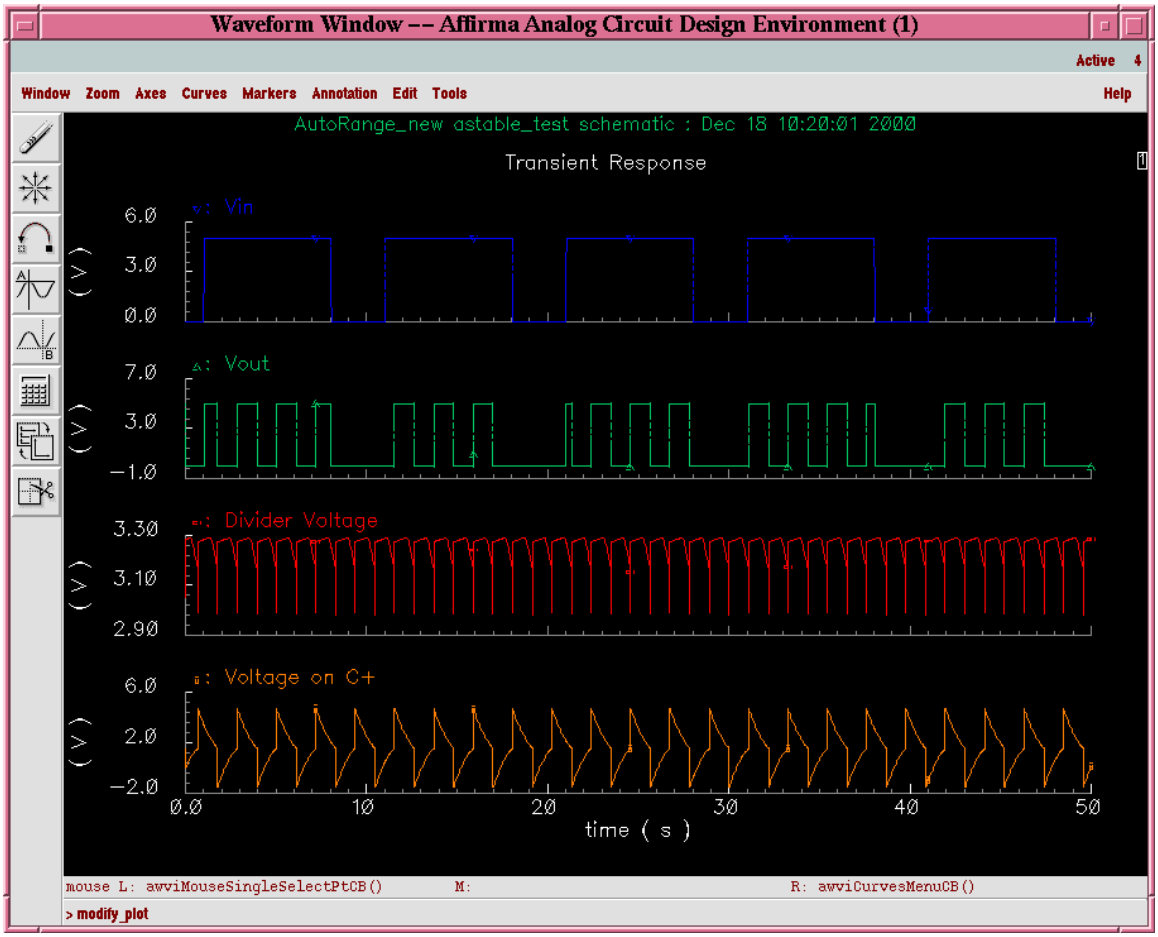


Figure 2.3.3.2

Figure 2.3.3.2 shows the simulation of the astable in operation, using a 10 µF capacitor. The first curve is the test input. The second is the output. The period is about 2.2 seconds. The third line shows the voltage divider that provides approximately 3.3 V. This drops to approximately 3 volts during switching. The last curve is the voltage at pin C+. It can be seen how the voltage ranges from –1.5V to +4.8V, which means the maximum gate voltage is 4.8V. This voltage range is much higher if the PMOS sources are set to Vdd, putting as much as 7.5 volts across the gates of the MOSFETs.

2.3.3.3 Layout

Astable Layout

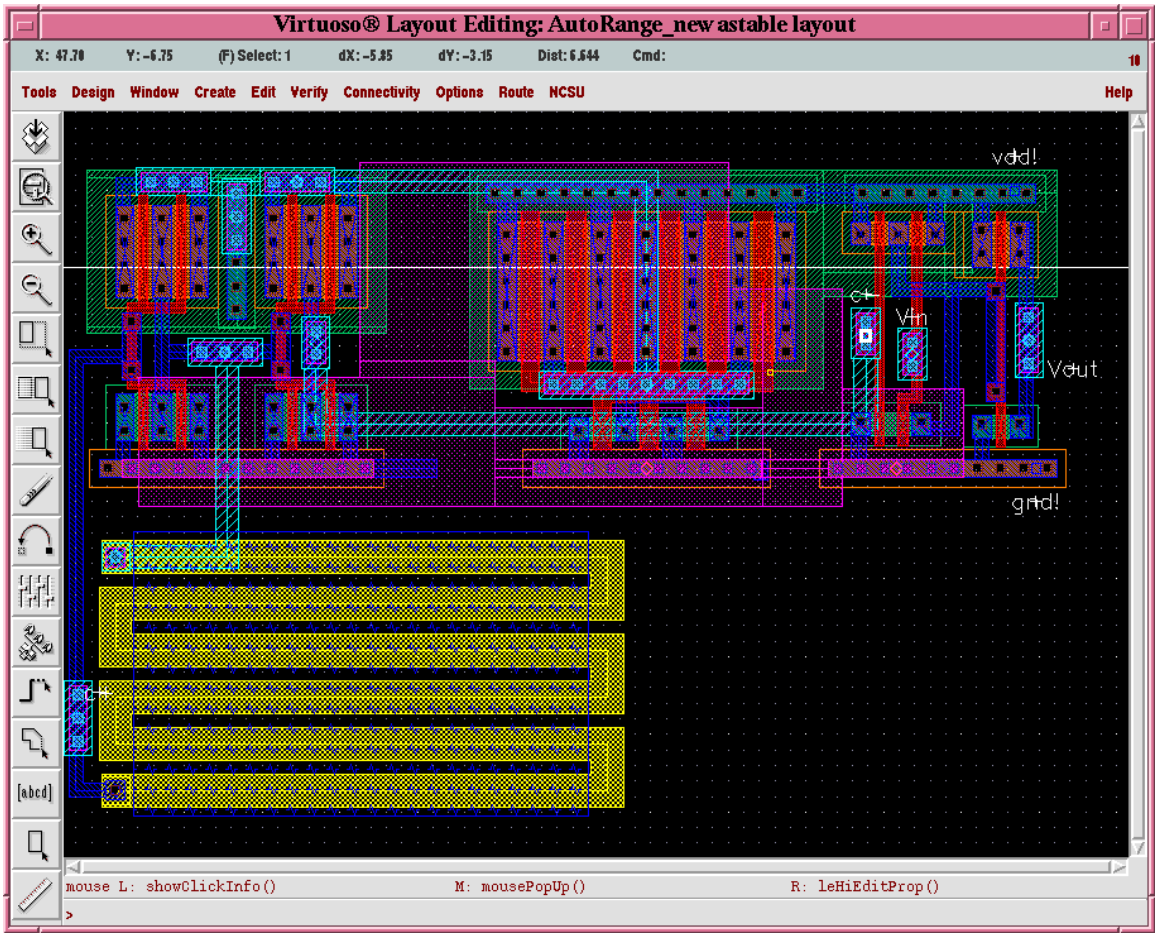


Figure 2.3.3.3

The astable layout can be seen in Figure 2.3.3.3 above. The two inverters and resistor are at the left. Note that the sources and nwell of the PMOSs are connected to the voltage divider in the center. The highres layer, combined with the serpentine pattern of the resistor, cause LVS to regard this as 6 resistors in series instead of one resistance. The NAND and inverter take up relatively little space at the far right. Metal2 covers much of the circuit, connected to ground so as to shield the devices from the effects of spikes in metal3 signal lines.

2.3.4 Input Package

2.3.4.1 Description & Design Considerations

Input Package Schematic

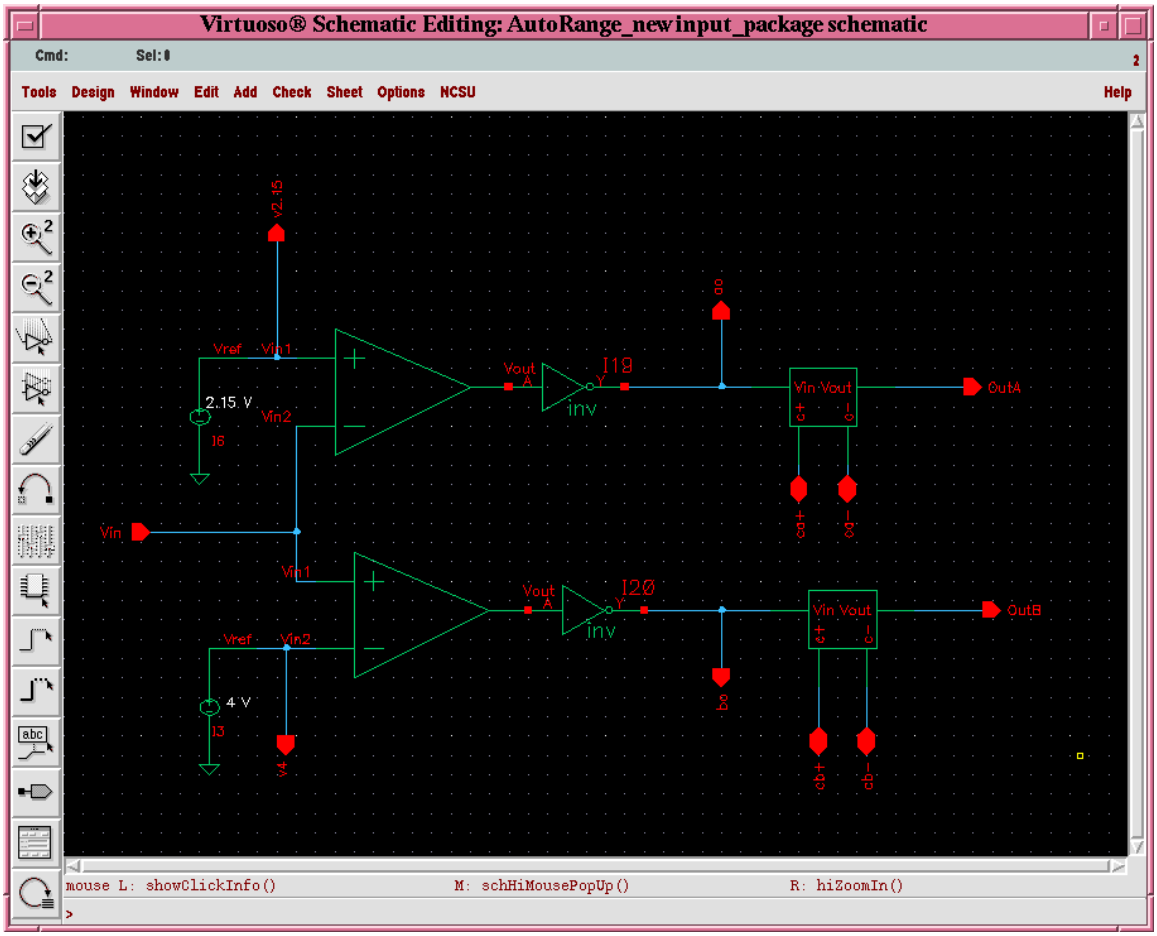


Figure 2.3.4.1

Since all analog components were adjacent, it was decided to make a package schematic and layout of the input components. This was done primarily to keep the analog input stages away from the digital lines and components in the rest of the circuit, but also to ease layout. The input package can be seen in Figure 2.3.4.1 above.

2.3.4.2 Simulation Results

Input Package Simulation

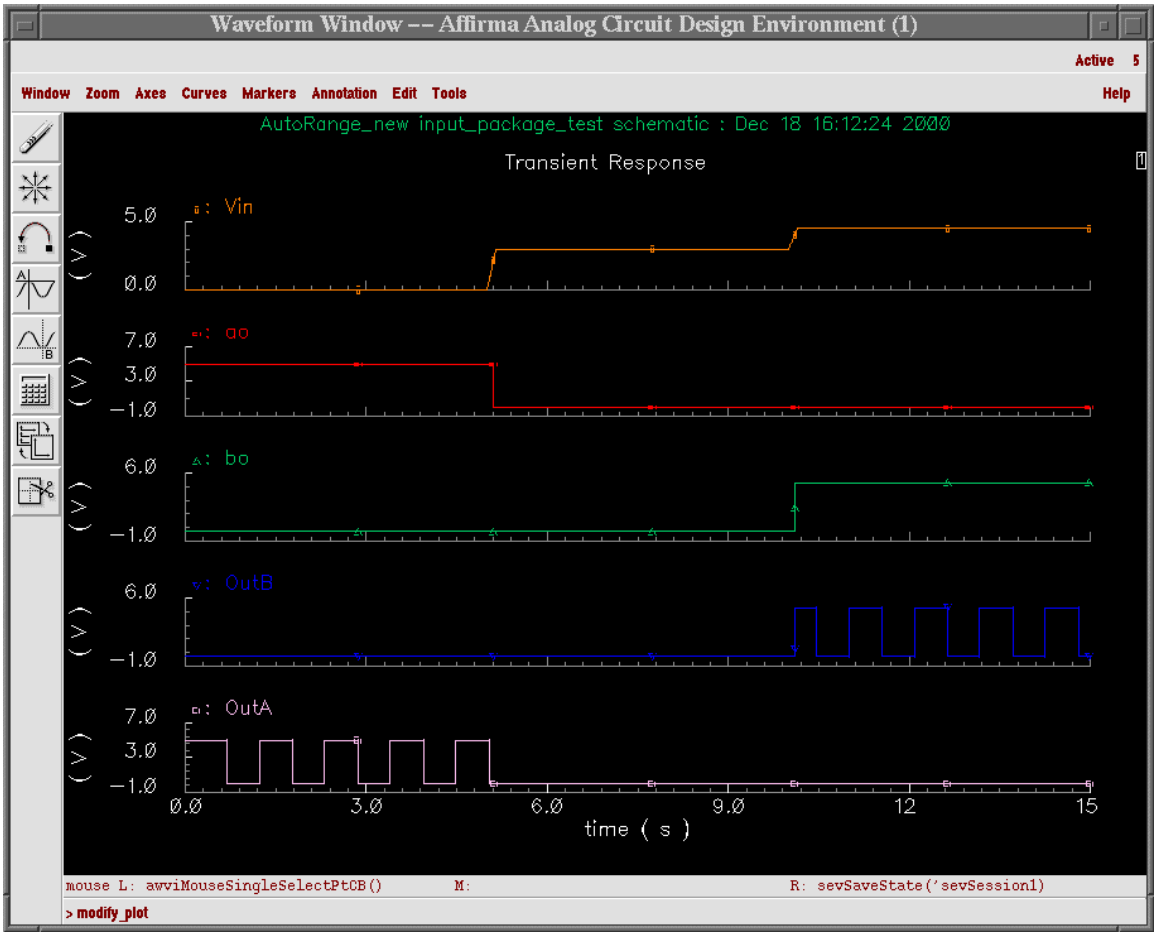


Figure 2.3.4.2

In the input simulation shown in Figure 2.3.4.2, this behavior is verified. The reference voltage Vin is compared to 2.15V and 4V. If Vin is higher than 4 V, OutB has the output of the astable on it. If Vin is lower than 2.15V, OutA has the output of the astable on it. If Vin lies between 2.15 V and 4 V, both outputs are low.

2.3.4.3 Layout

Input Package Symbolic Layout

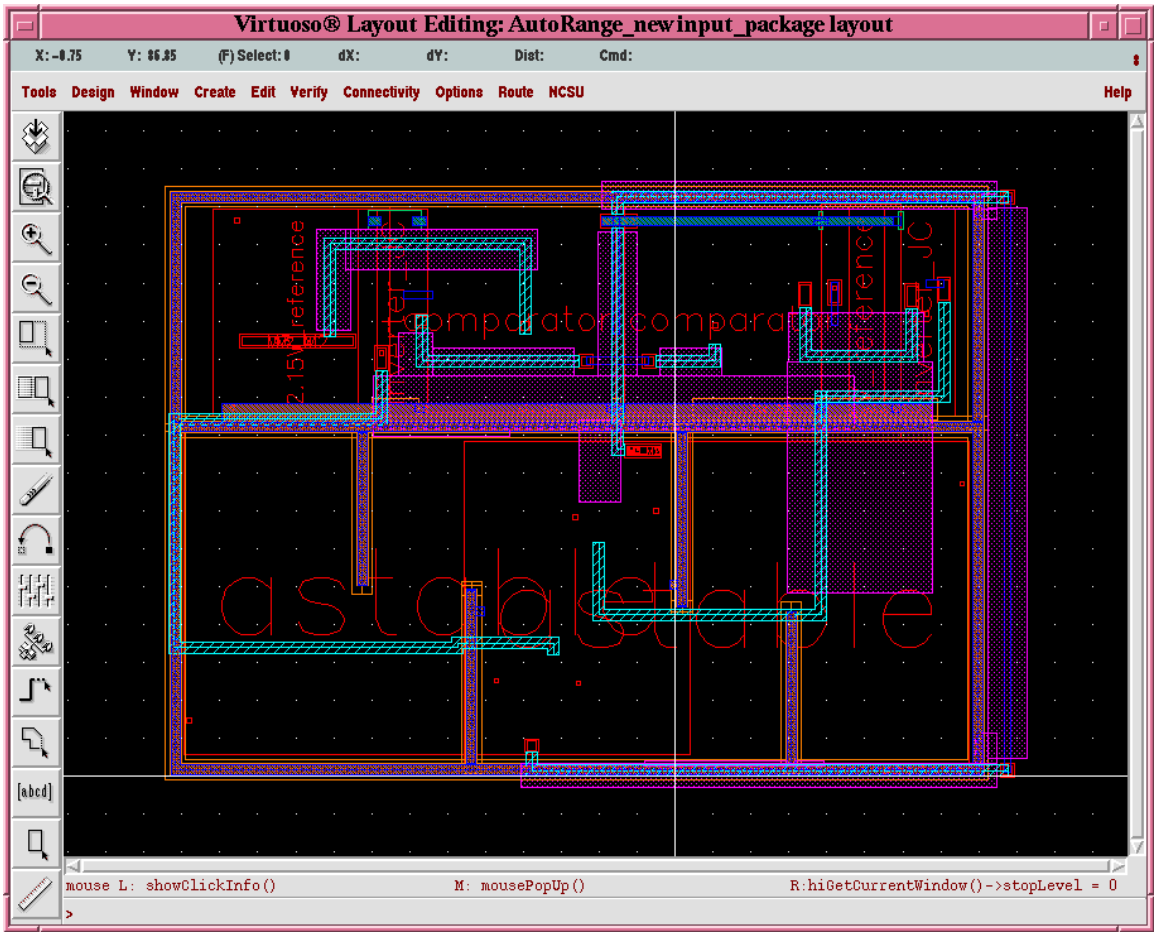


Figure 2.3.4.3A

Figure 2.3.4.3A is included to give the reader an idea of the different components involved in its layout. Interesting features in this view include the guard ring tied to ground surrounding the circuit, in order to shield it from the rest of the circuit, and the extensive metal2 planes tied to ground, designed to shield the rest of the circuit from metal3 signal lines

Input Package Layout (comprehensive view)

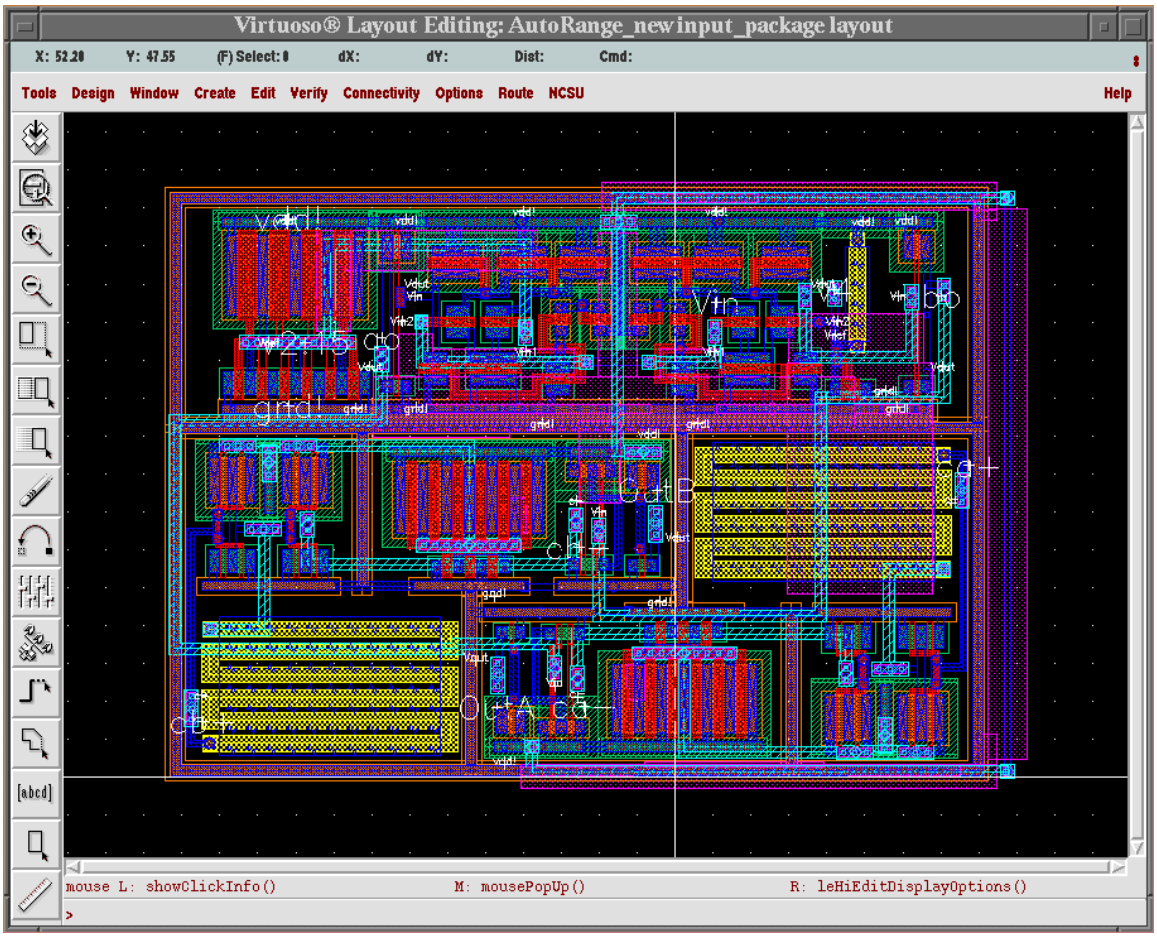


Figure 2.3.4.3B

Figure 2.3.4.3B shows the complete view of the circuit. The two astables are on bottom, with one turned upside down. The comparators, inverters, and voltage references can be seen along the top. This layout is believed to be reasonably compact-it measures 115 µm x 80 µm.

2.4 Ring Oscillator

2.4.1 Description and Design Considerations

Ring oscillators are often used to generate clock signals or to test the functionality of wafers. It was decided to implement a 31-stage ring oscillator on chip in order to get more data during testing and to use space and pins that otherwise would have been left unused.

31-Stage Ring Oscillator Output Buffer

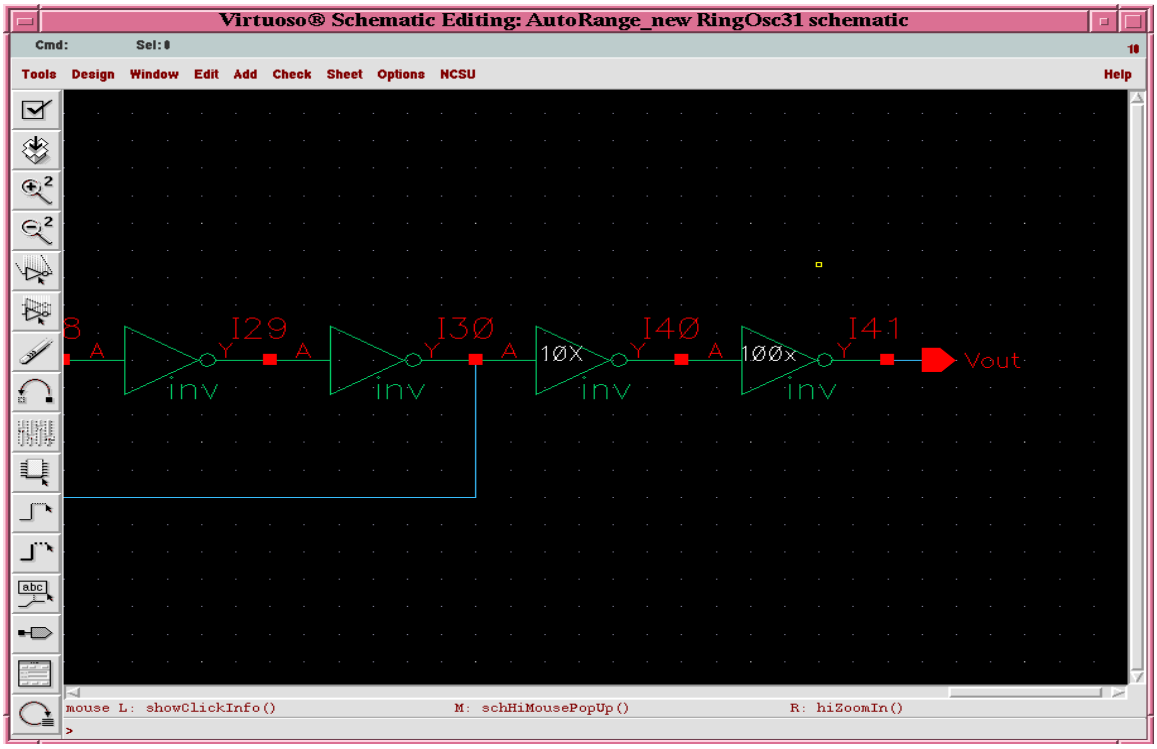


Figure 2.4.1

The multiplication factor in the schematic refers to the number of times wider the transistors in the inverters are than in the minimally sized inverter.

2.4.2 Simulation

31-stage Ring Oscillator Schematic with 20 pF Load

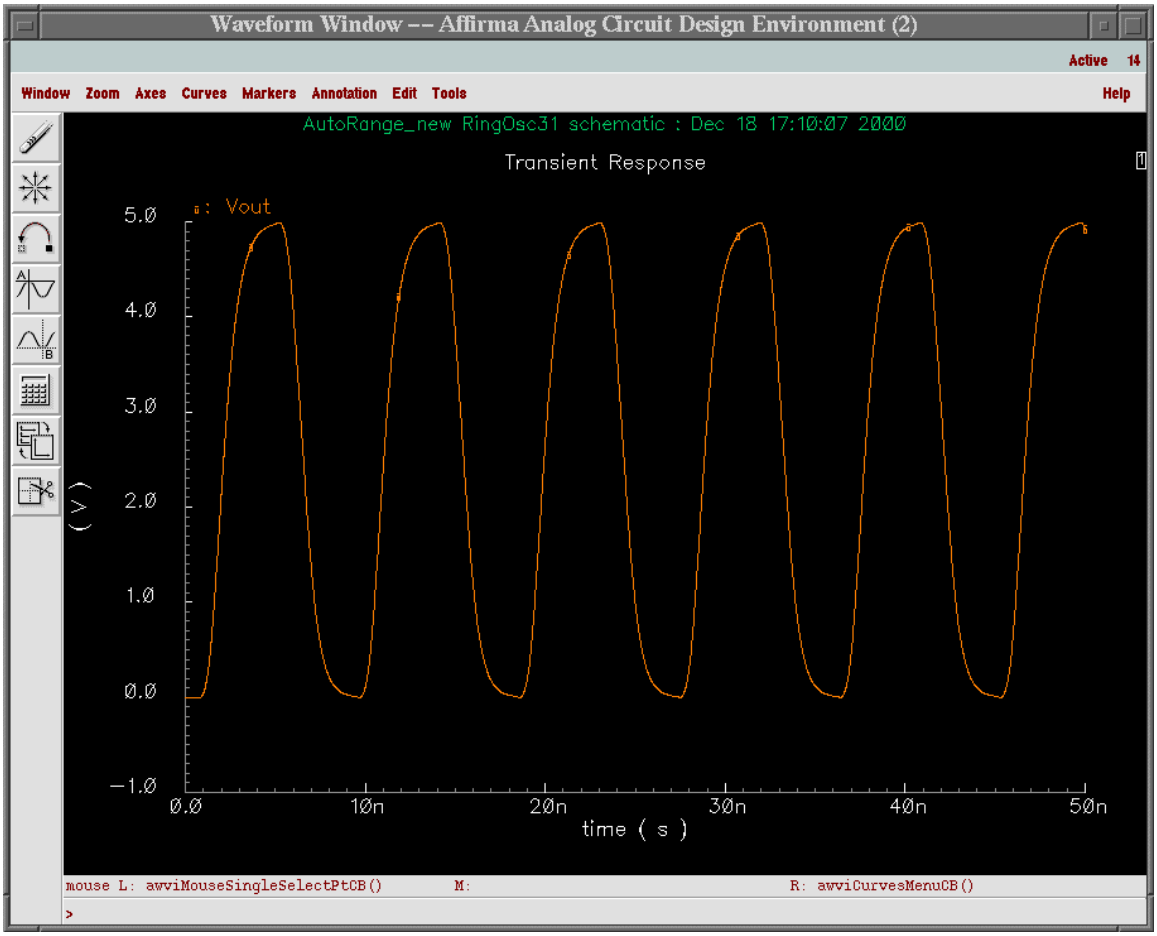


Figure 2.4.2

The simulation results in Figure 2.4.2 show that the output, which is ideally a square wave, is smoothed almost to a sine wave by the load capacitance. This output will suffice for testing. Note that the period is 8.909 ns, so the expected value for the frequency is 112.25 MHz .

2.4.3 Layout

31-Stage Ring Oscillator Layout

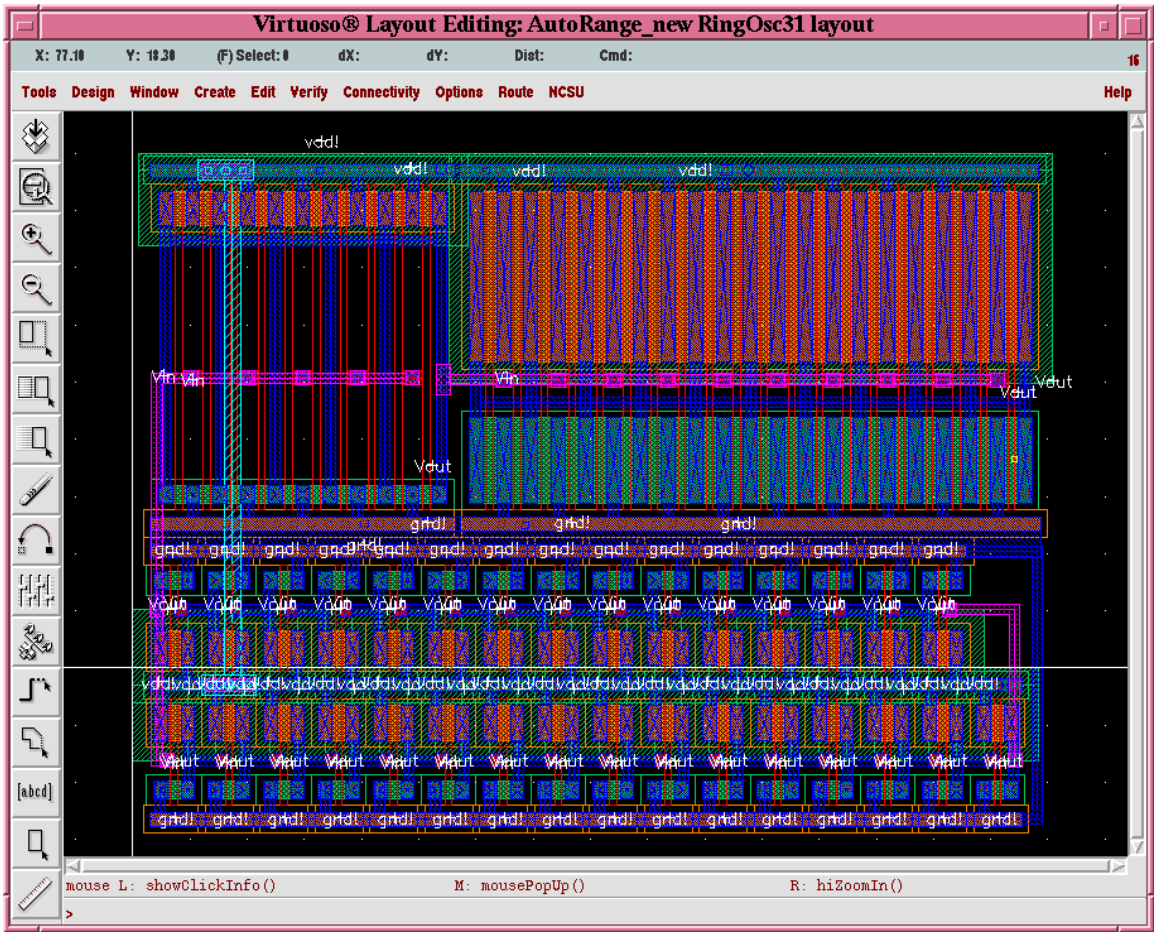


Figure 2.4.3A

The ring oscillator and output buffer shown in figure 2.4.3A are fairly easy to see. The 31 inverters are on bottom, the 10x inverter is at the upper left, and the 100x inverter is at the upper right. This circuit, however, produces an odd error with LVS, as shown below in Figure 2.4.3B.

LVS Errors in the Ring Oscillator Circuit

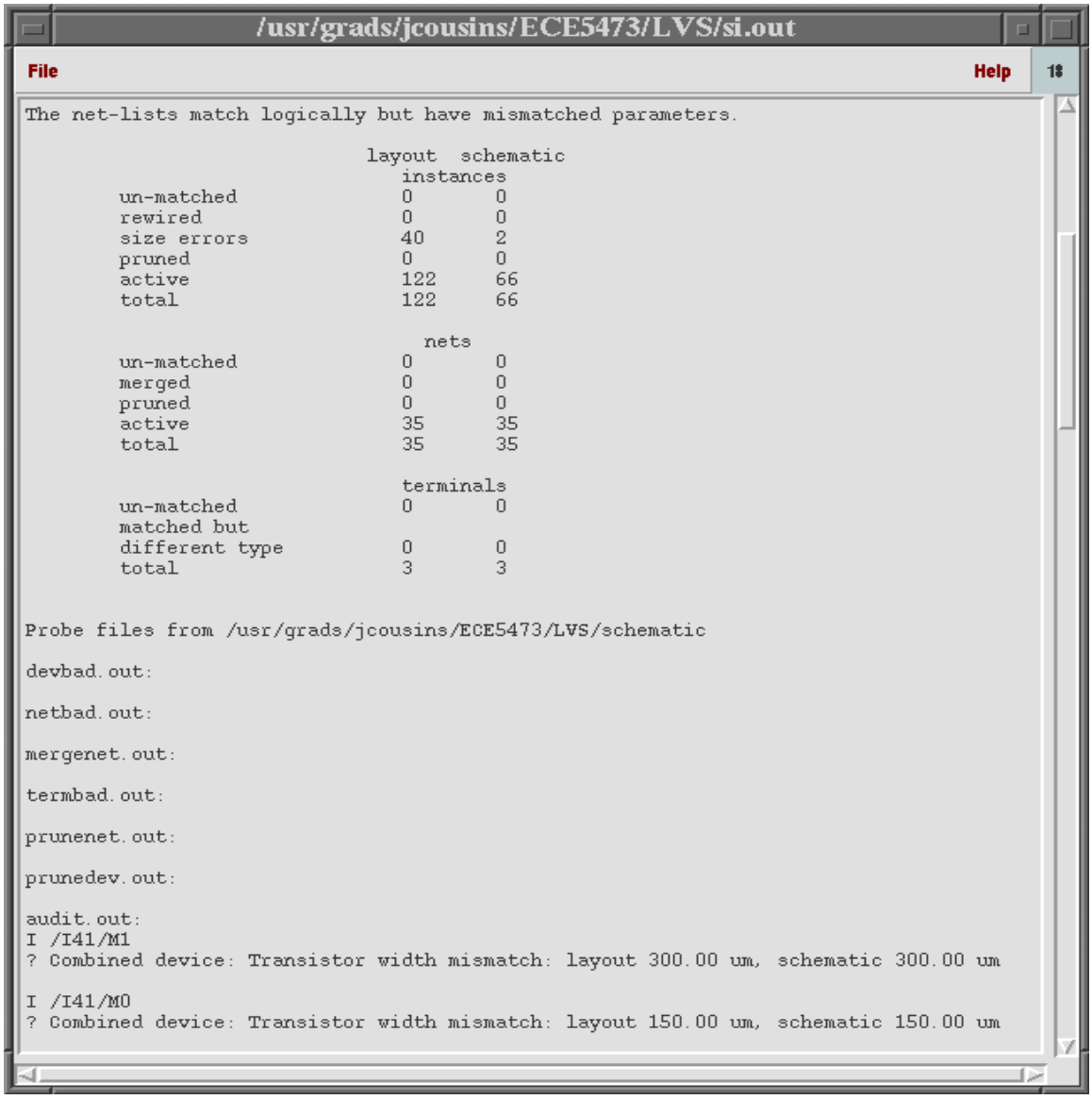


Figure 2.4.3B

LVS is giving errors due to identical width sizes. After some experimentation, it was theorized that the large size of the transistors was causing a floating-point error at some value lower than the 2 decimal places displayed in the LVS window. A careful analysis of the 100x inverter showed no reason for these odd errors, so they were ignored. They also reappear twice in the top level LVS, since 2 ring oscillators are implemented in the top level.

2.4 AutoRanger Circuit

2.4.1 Description

This circuit simply counts up when the low voltage is below 2.15v and counts up states while the high voltage is above 4v. It is also required to send back feedback so the relays can pick the next appropriate range. The rest of the output consists of a bit patterns representing the current range. If the signal is out of range the circuit will generate an appropriate *Too_High* or *Too_Low* output signal.

2.4.2 Design considerations

For the Auto-Ranger circuit we require all the basic circuits developed in early sections. We need two D Flip-Flops, which will be used for generating a *Too_High* or *Too_Low* output signal. We need a counter to keep track of range. We need the input package to generate appropriate *CU* or *CD* pulses of the clock depending on the input voltage. We need a decoder to provide output for the relays. Finally we need an encoder that takes in both the *Too_High* and *Too_Low* signals along with the current count, which together creates the current range.

In Figure 2.4.2 we see the schematic of the AutoRange circuit. The left, middle pin is the input reference voltage for the circuit to compare. The output of the Input Package either connects to the count-up or count-down section for the counter, which is in the center. You can then see the encoder and decoder strip of the outputs. Also notice the two D Flip-Flops at the top and bottom of the circuit. All the other outputs are just test points. The design calls for 1 Input Package, 2 two-input NAND gates, 4 inverters, 2 D Flip-Flops, 1 Counter, 1 Encoder and 1 Decoder; resulting in a total of 471 transistors.

AutoRanger Schematic

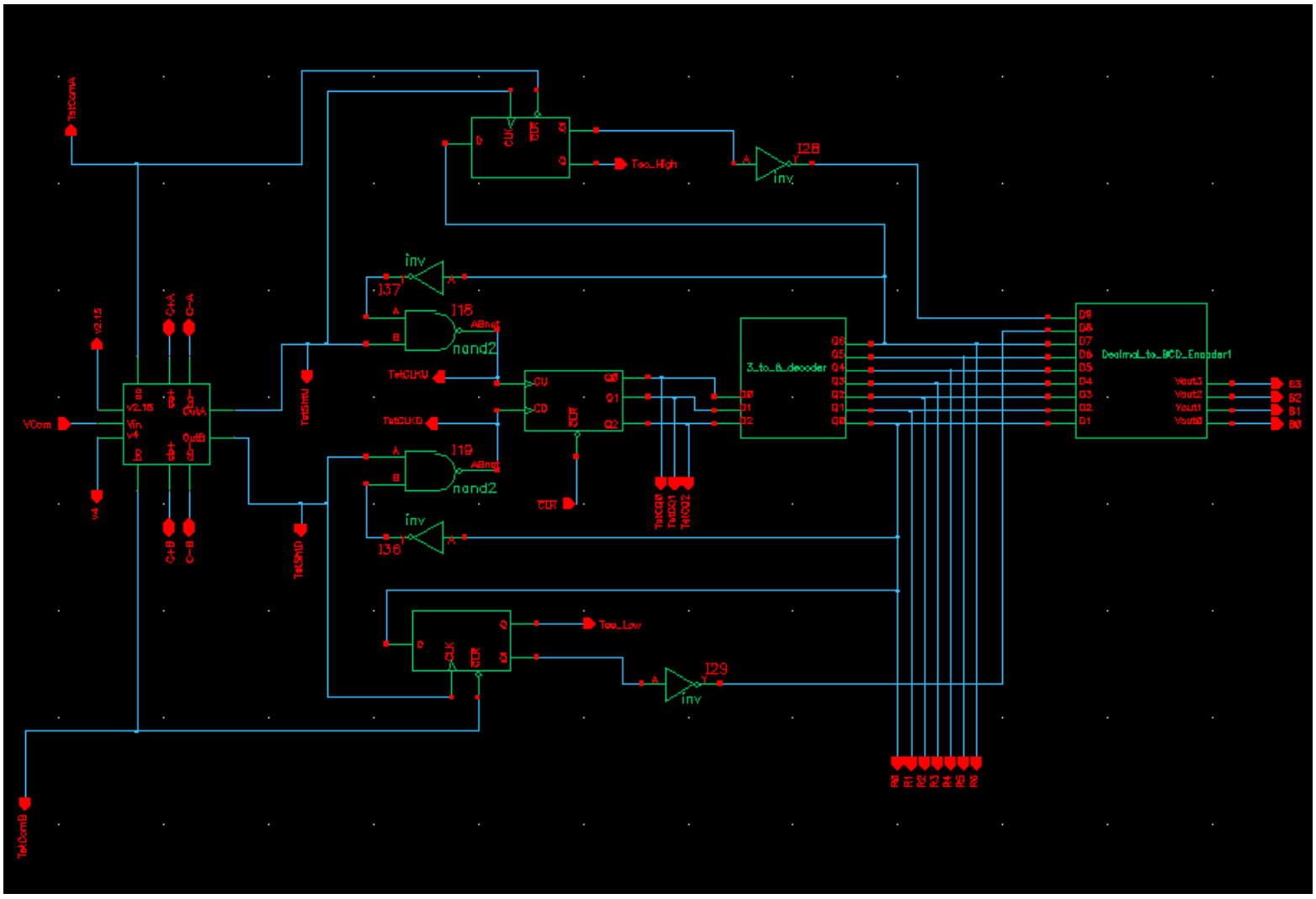


Figure 2.4.2

2.4.3 Simulation results

The simulation in figure 2.4.3A shows that the AutoRanger circuit design functions correctly. When the input voltage (top curve) is below 2.15v we count up till we get too high or the input signal changes to a value between 4v-2.15v. When it is above 4v we count down till we get too low or the input voltage signal changes to a value between 4v-2.15v. For this simulation we used two 5µF capacitors which gave us a period around 1 second. In Figure 2.4.3B notice in the input voltage (the last curve) has a high value of 3v. Notice the circuit does hold its value when the input voltage is between 2.15v-4v.

AutoRanger Simulation

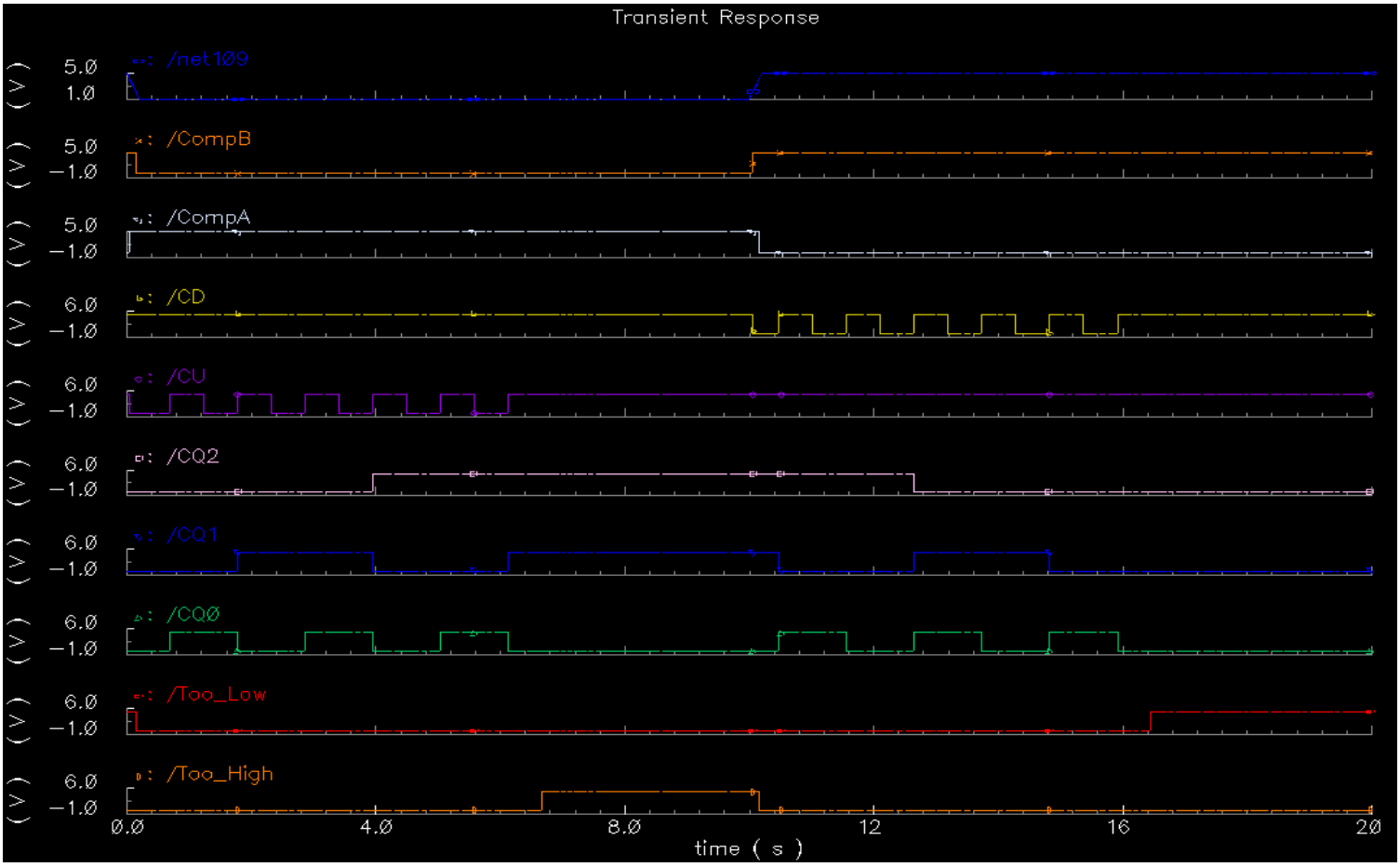


Figure 2.4.3A

AutoRanger Simulation

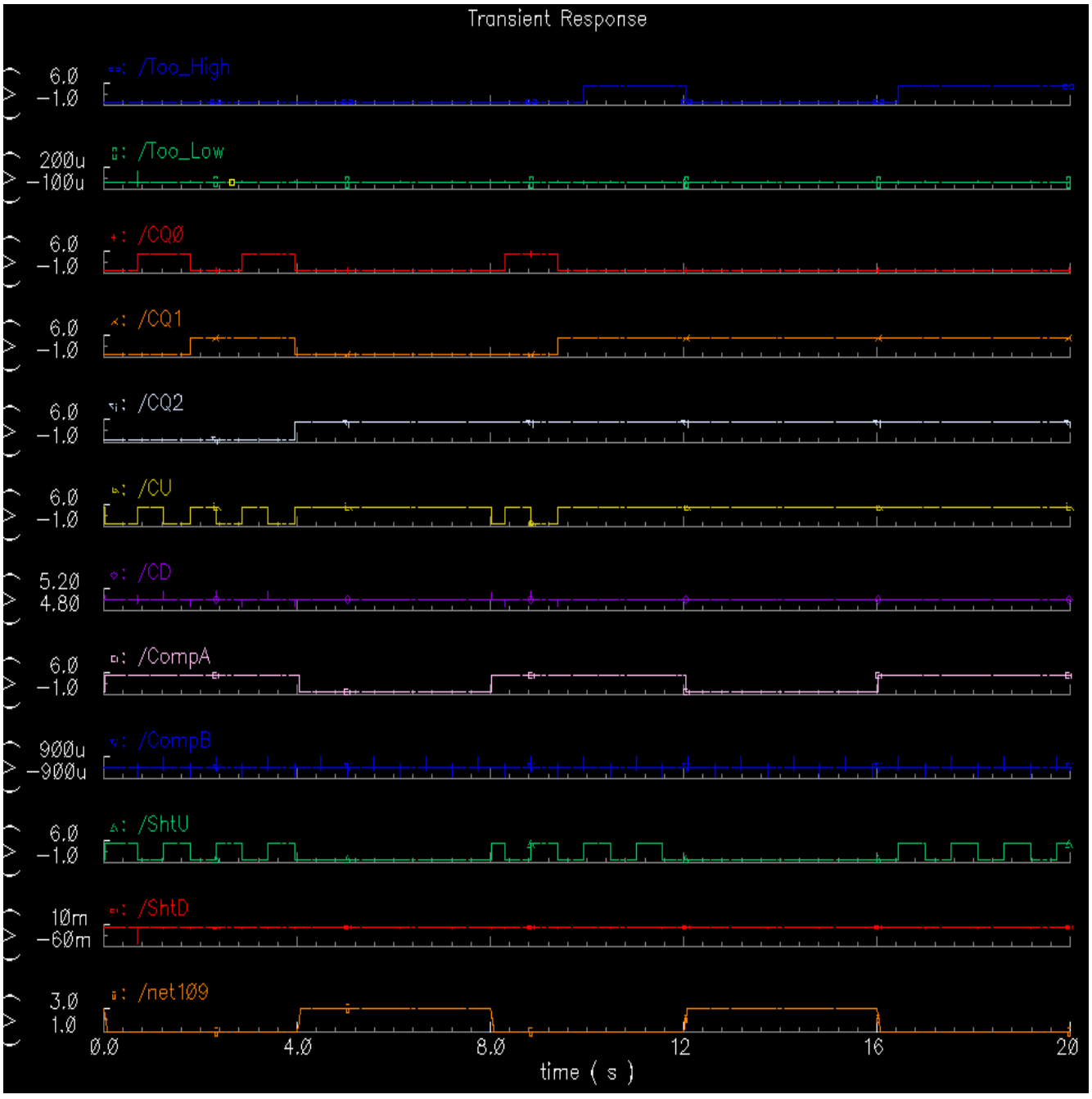


Figure 2.4.3B

2.4.4 Layout

The Layout of the AutoRanger circuit is shown in Figure 2.4.4. The size is 480.8µm x 348.9µm. On the top line near the middle of the input package is V_{com}, which is the input voltage. You can see the counter below the input package. The two circuits on the right with the long busses are the encoder and decoder.

AutoRanger Layout

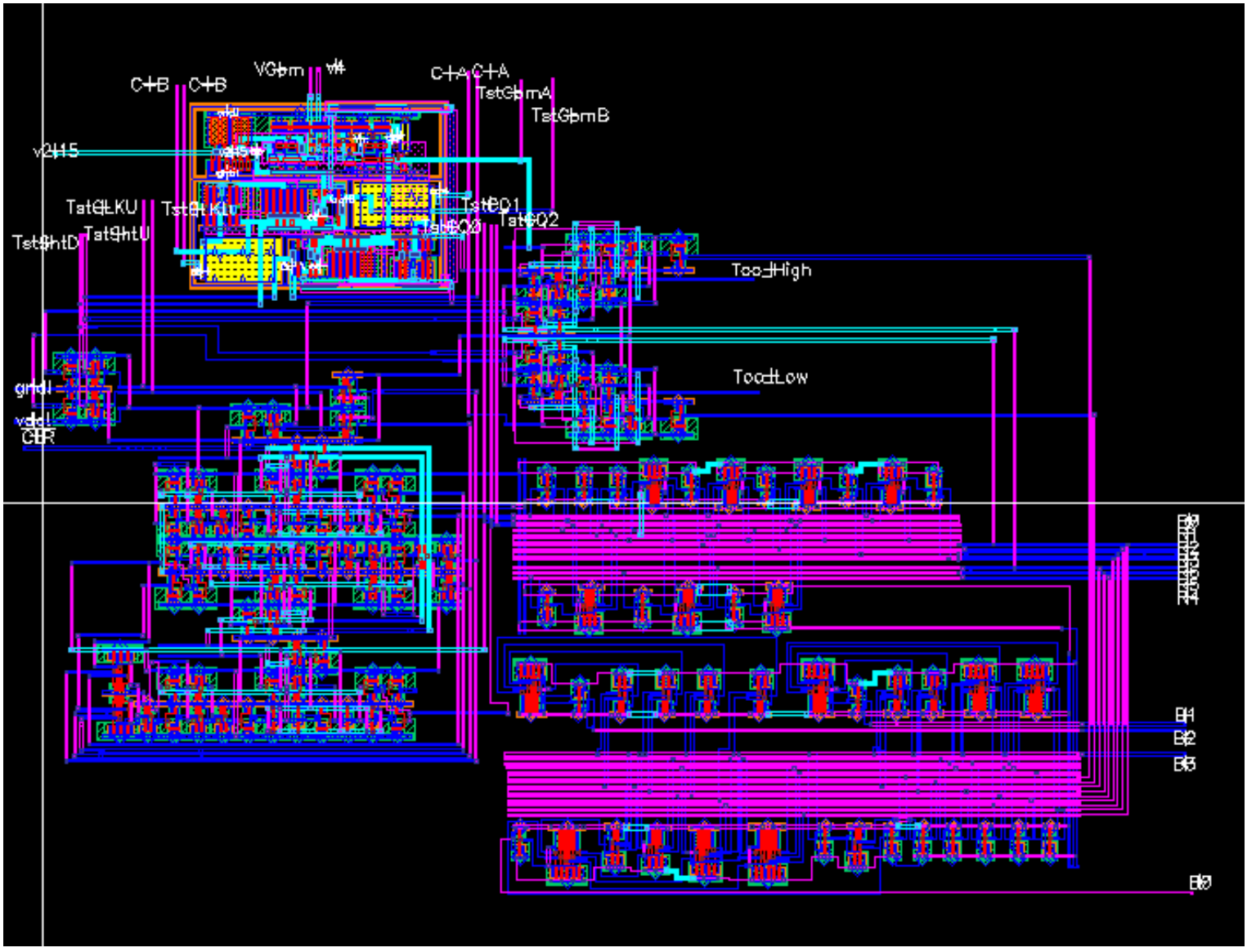


Figure 2.4.4

2.4.5 Entire Pin-Out Schematic

The pin-out of the AutoRanger circuit is shown in Figure 2.4.5. For the circuit pin-out we used all 40 pins. We pinned out the AutoRanger circuit, D Flip-Flop, JK Flip-Flop and 2 ring oscillators. The pads we used were all I/O pads except of the voltage reference test points and the capacitor connections, which were Ref pads.

Entire Pin-out of the AutoRanger Schematic

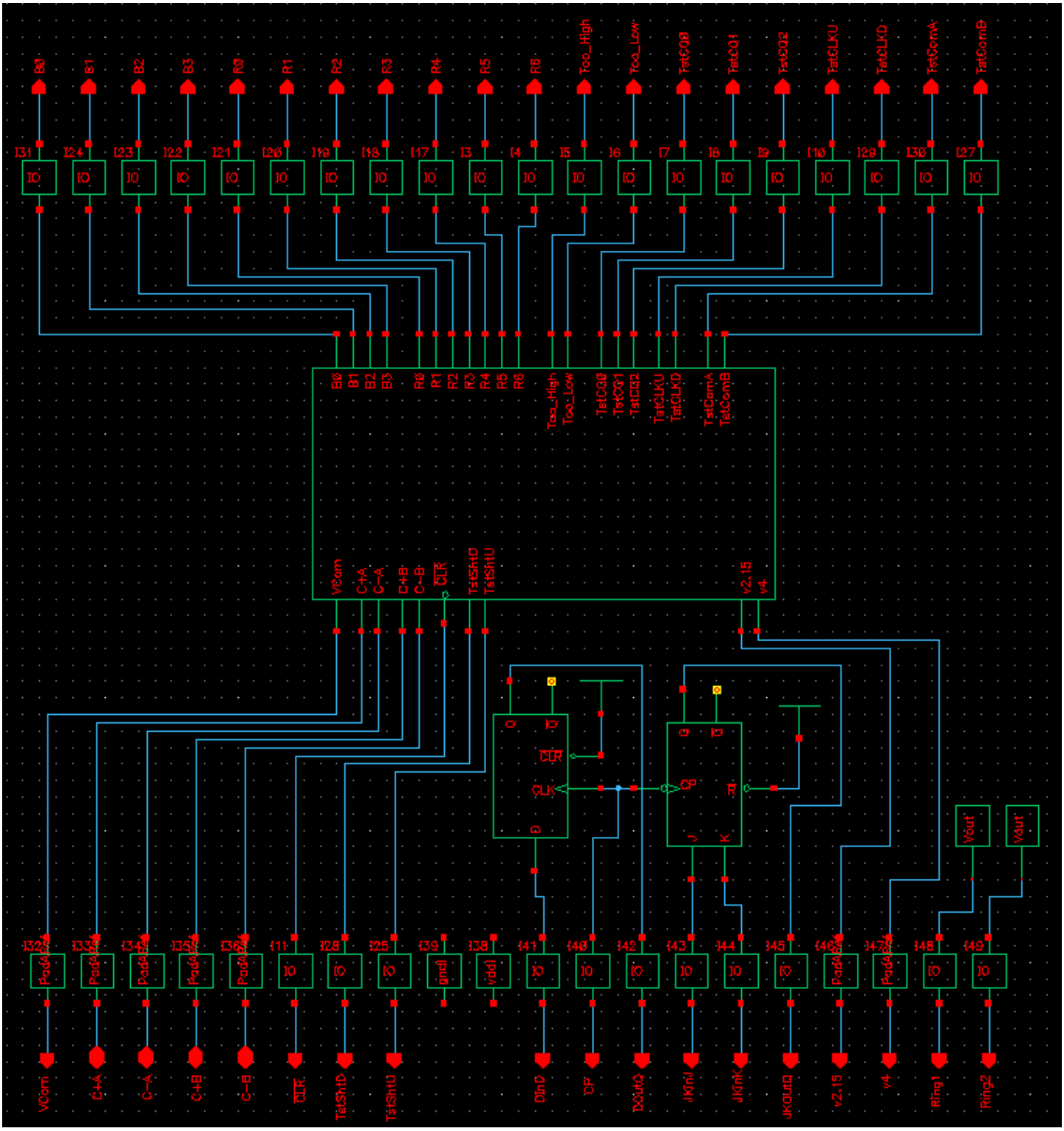


Figure 2.4.5

2.4.6 Entire Circuit Layout

A pin-out sheet can be found in Appendix D. The AutoRanger circuit is pinned out in the upper–right corner of the chip. The D Flip-Flop lower-left side, while the JK Flip-Flop and 2 ring oscillators are along the bottom. The rest is filler. The entire chip size is 1500µm x 1500µm

Entire AutoRanger Circuit Layout

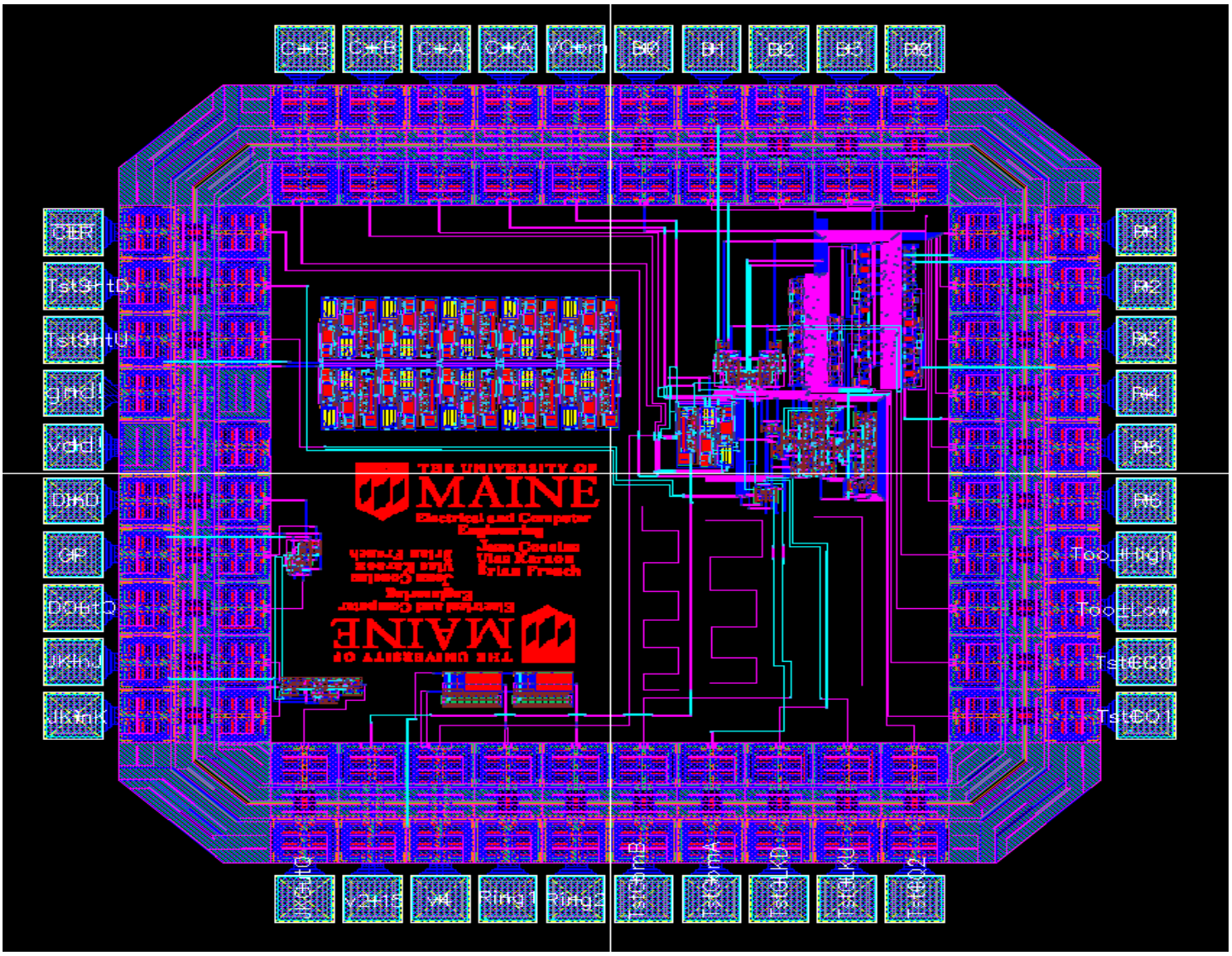


Figure 2.4.6

3. Experimental Results

3.1 Test Results

The test results section will be filled out next semester when we can test the chips.

3.2 Suggestion for Future Studies and Work

This section will be filled out next semester when we can test the chips.

3. Conclusions

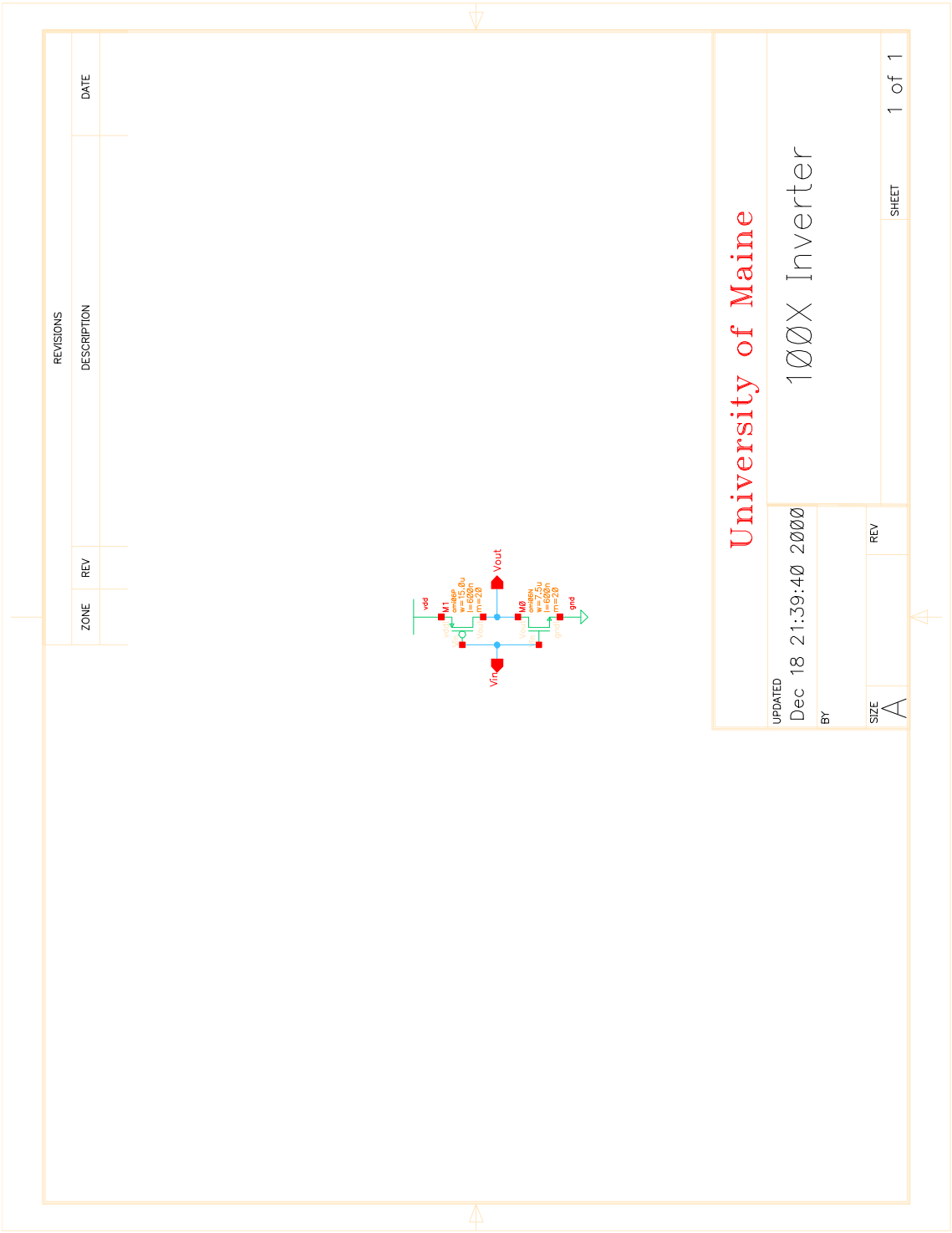
By using rigorous simulation and verifying that the layout matches the schematic, we are confident the chip will work. Some of the conclusions that can be drawn are that this chip has low power consumption, the astable circuit maintains a steep clocking edge for the counter even at low clock speeds and that the circuit would function correctly at much higher speeds.

Since all of the logic is static you will get near zero static power dissipation for these. As for the input package, there is small static power dissipation through the voltage references and current mirrors. Dynamic power also will contribute little because we will be running this circuit at the nominal speed of around 1Hz.

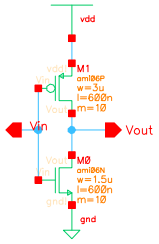
From simulation we found the critical path, which is through the counter, to be around *3.1ns*. This is an aggregate of the setup and hold times that would be needed. If we totally neglect the painfully slow mechanical relays connected in the feedback loop and if the Input Package could generate a clock of rate of around 300MHz the rest of the circuit should be able to keep up.

Finally, we did learn much from designing, simulation, and layout. But, perhaps the most important thing we learned was to always test your hierarchical designs against what the overall design is suppose to do.

Appendix A: Schematics



REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 21:08:41 2000

BY

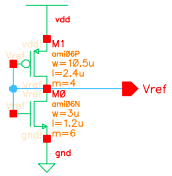
SIZE
A

REV

10X inverter

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 6 14:36:46 2000

BY

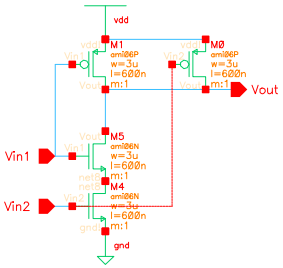
SIZE
A

REV

2.15V voltage divider

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 22:07:35 2000

BY

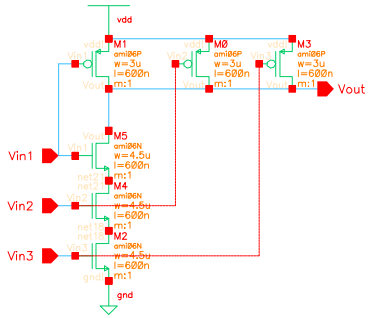
SIZE
A

REV

2 input NAND gate

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 22:19:43 2000

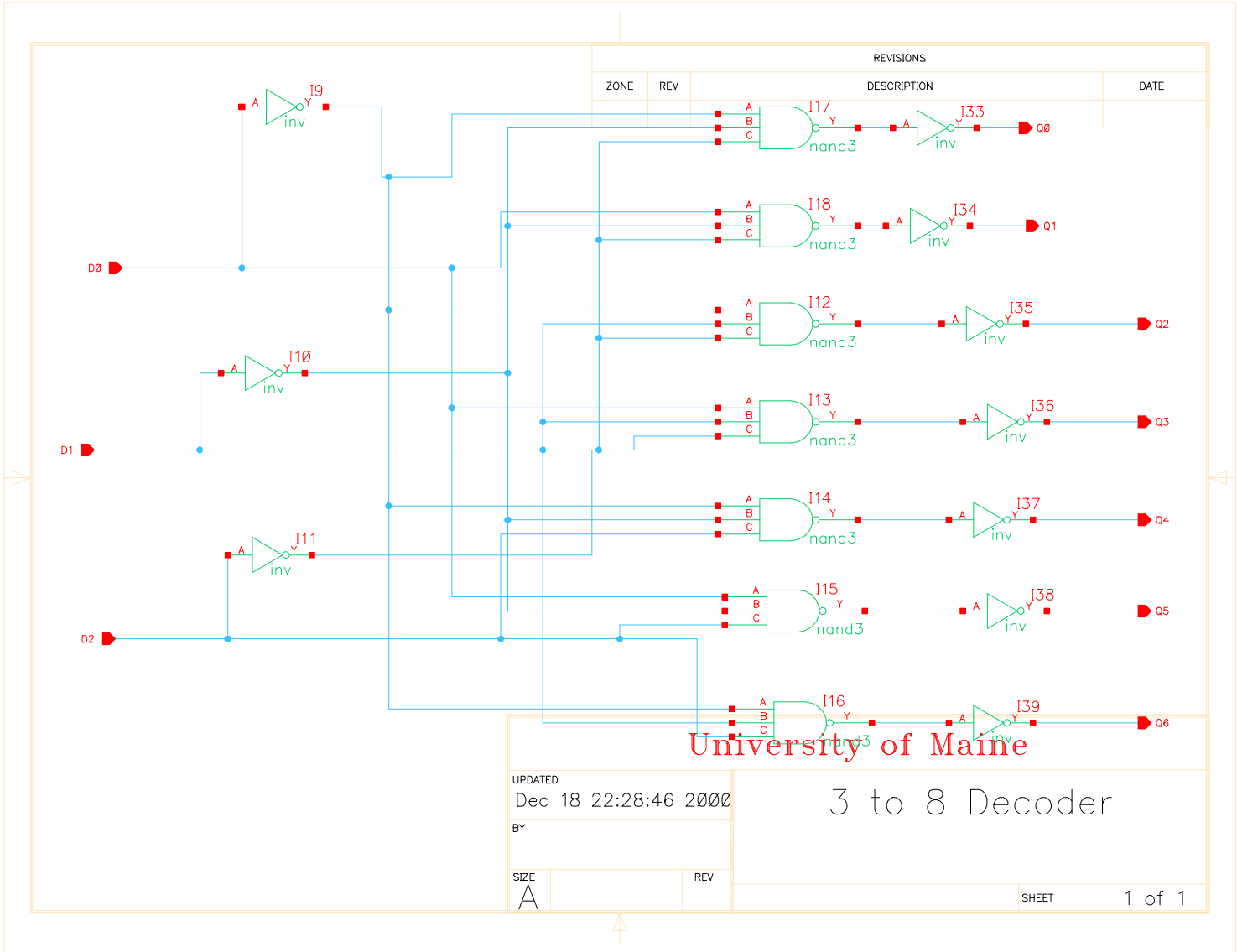
BY

SIZE
A

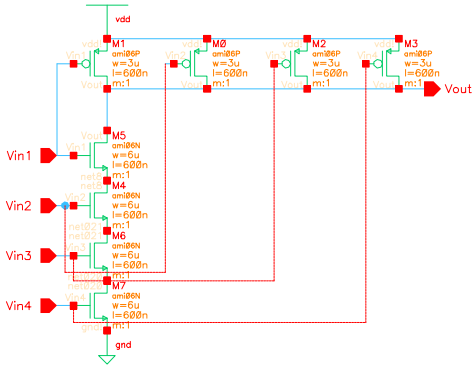
REV

3 in NAND

SHEET 1 of 1


REVISIONS
ZONE
REV
DESCRIPTION
DATE
I9
I10
I11
I12
I13
I14
I15
I16
I17
I18
I33
I34
I35
I36
I37
I38
I39
inv
nand3
D0
D1
D2
Q0
Q1
Q2
Q3
Q4
Q5
Q6
University of Maine
UPDATED
Dec 18 22:28:46 2000
BY
SIZE
A
REV
3 to 8 Decoder
SHEET
1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 22:31:36 2000

BY

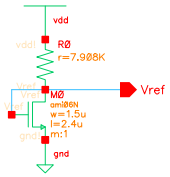
SIZE
A

REV

4 input NAND Gate

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 22:30:21 2000

BY

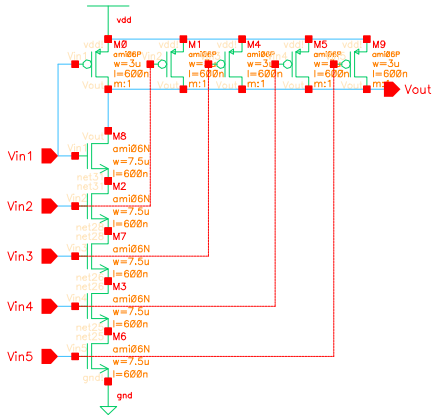
SIZE
A

REV

4 V Reference

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 22:32:40 2000

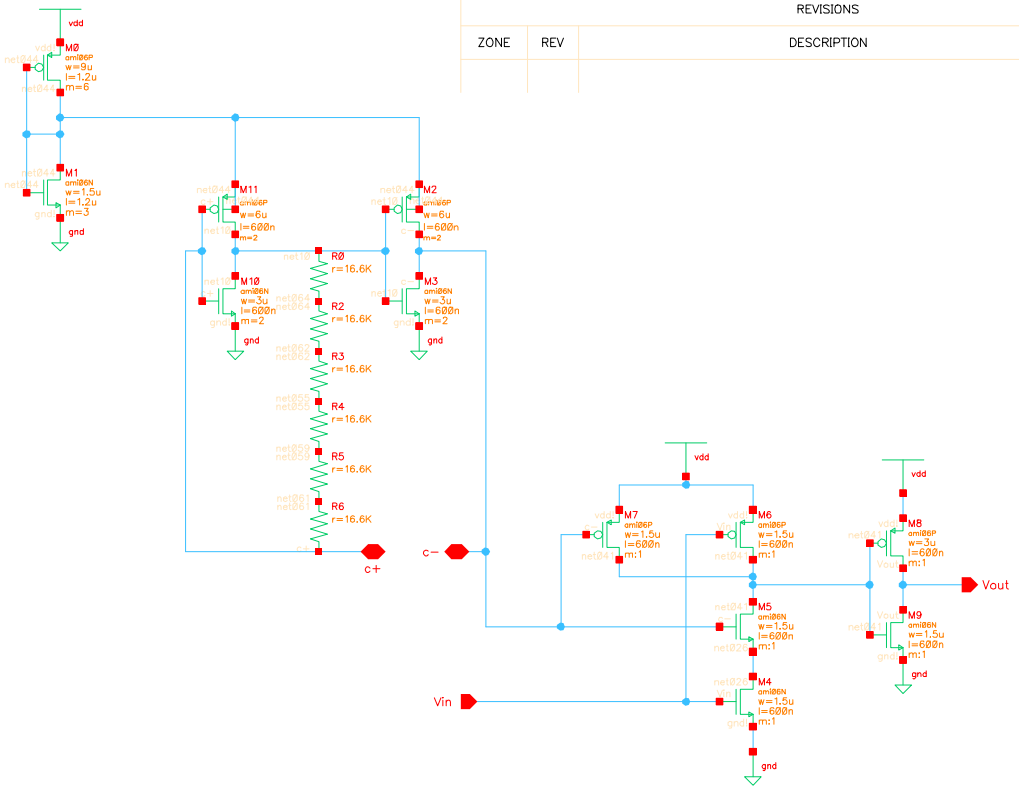
BY

SIZE
A

REV

5 input NAND gate

SHEET 1 of 1



REVISIONS			
ZONE	REV	DESCRIPTION	DATE

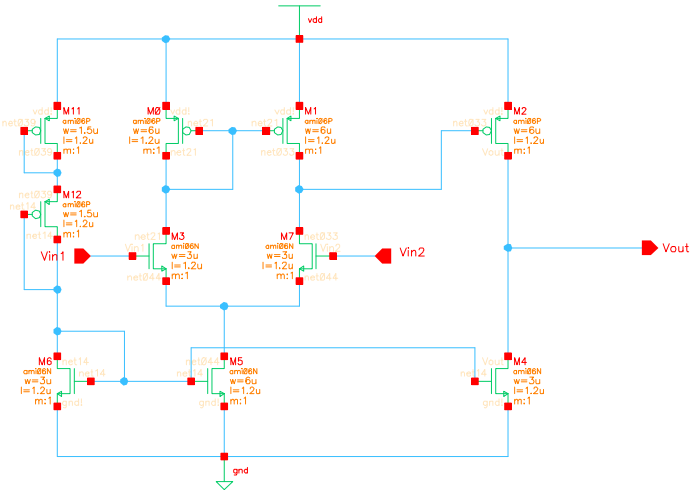
University of Maine

UPDATED	Dec 18 21:58:20 2000
BY	
SIZE	A
REV	

Astable

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 21:59:09 2000

BY

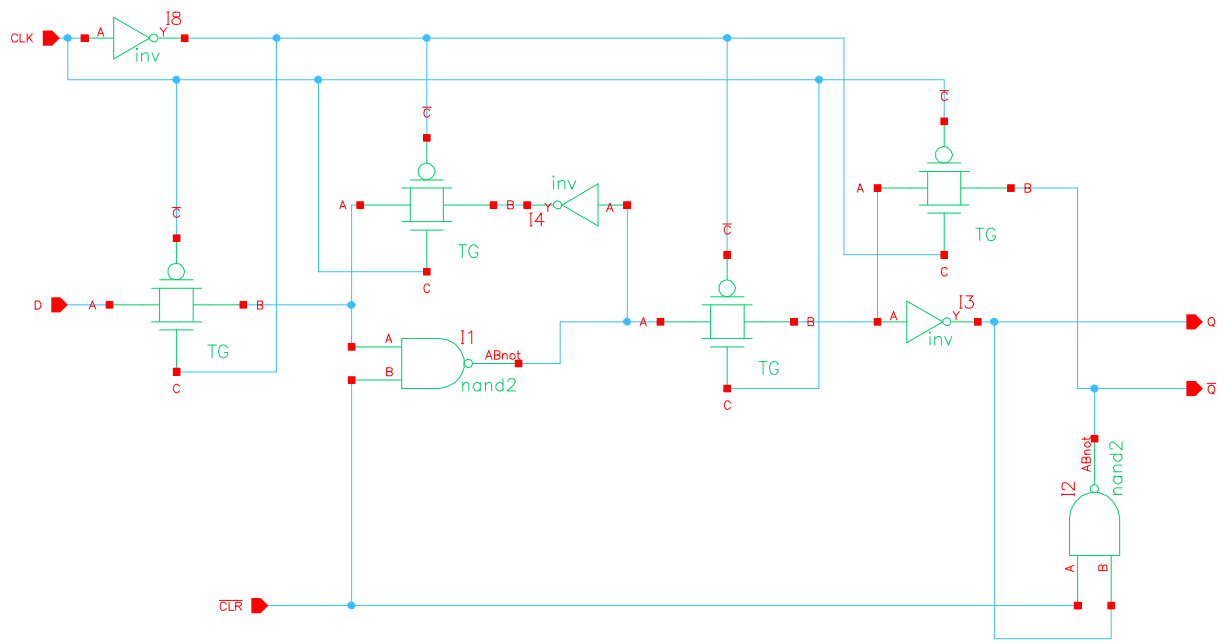
SIZE
A

REV

Comparator

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 23:09:15 2000

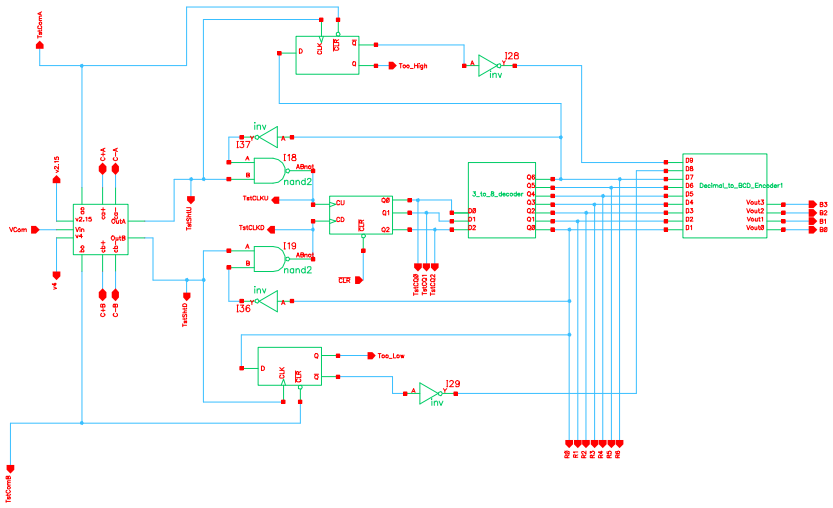
BY

SIZE
A

REV

D Flip Flop

SHEET 1 of 1



REVISIONS			
ZONE	REV	DESCRIPTION	DATE

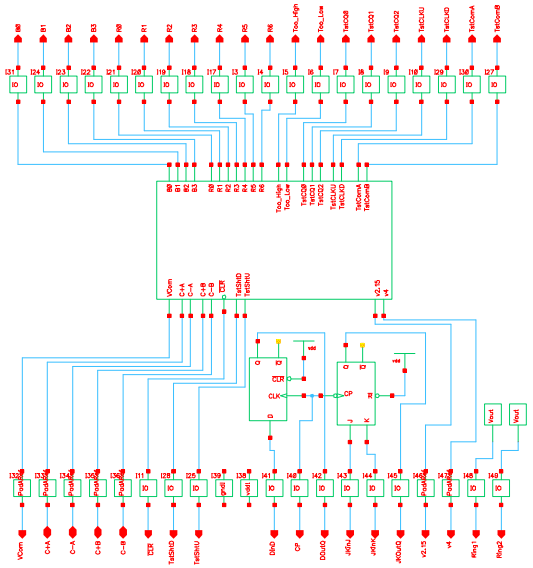
University of Maine

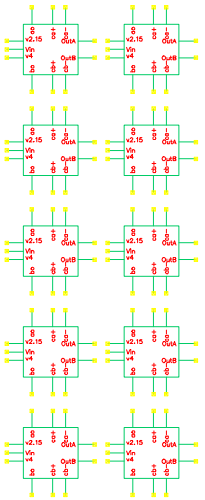
UPDATED	Dec 18 21:49:49 2000
BY	bfrench
SIZE	C
REV	

Final AutoRanger circuit

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



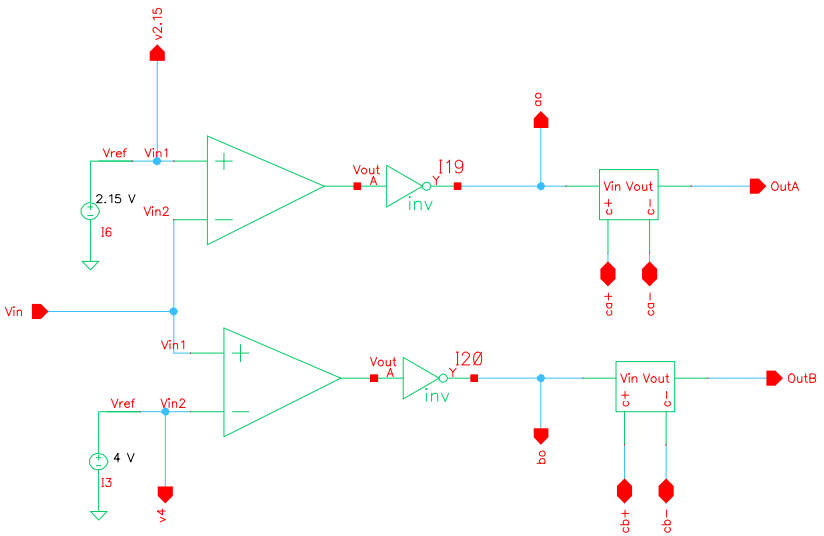


University of Maine

UPDATED	Dec 18 21:53:33 2020
BY	bfrench
SIZE	C
REV	
SHEET	1 of 1

nal Auto Ranger with Pacs

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 22:00:43 2000

BY

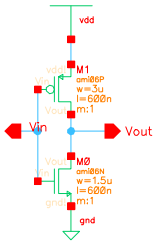
SIZE
A

REV

Input Package

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 22:01:45 2000

BY

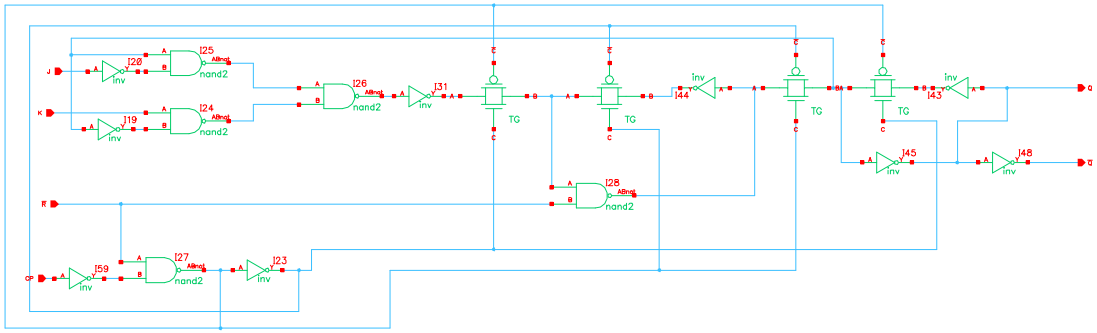
SIZE
A

REV

Basic Inverter

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED	Dec 18 23:10:29 2000
BY	bfrench
SIZE	C
REV	

JK Flip Flop

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE

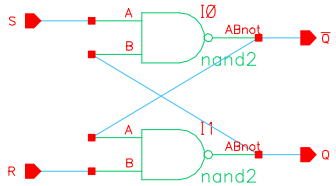
University of Maine

UPDATED	Dec 18 21:56:34 2020
BY	Jacuona
SIZE	E
REV	

31 Stage Ring Oscillator

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 23:11:50 2000

BY

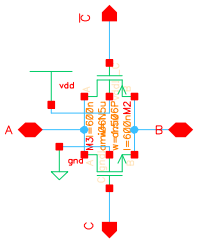
SIZE
A

REV

S–R Latch

SHEET 1 of 1

REVISIONS			
ZONE	REV	DESCRIPTION	DATE



University of Maine

UPDATED
Dec 18 23:07:31 2000

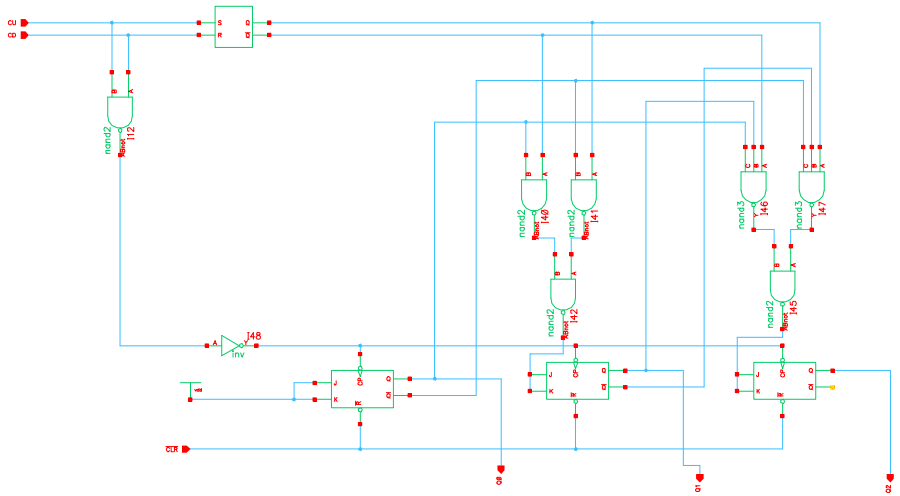
BY

SIZE
A

REV

Transmission Gate

SHEET 1 of 1



REVISIONS			
ZONE	REV	DESCRIPTION	DATE

University of Maine

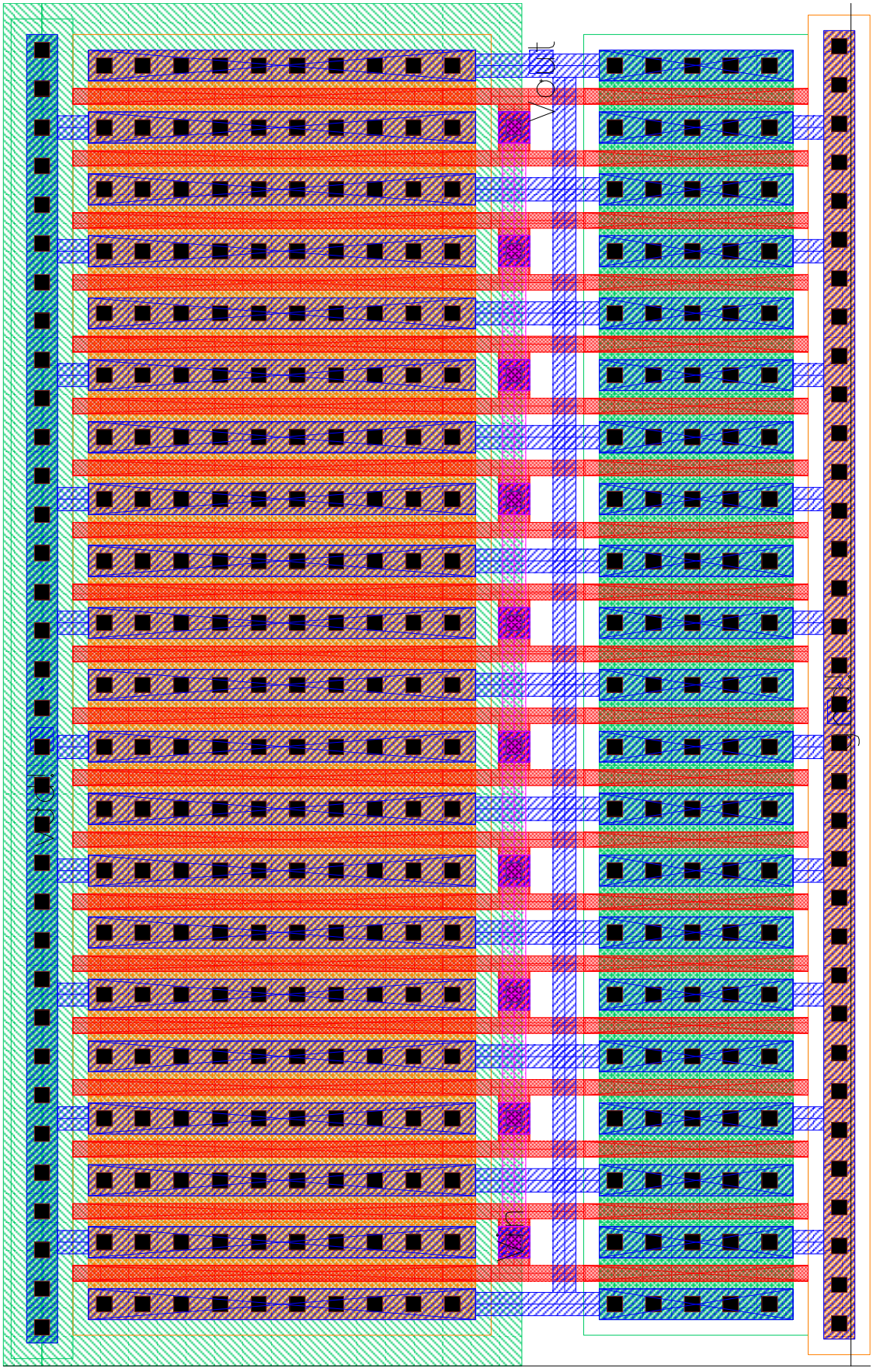
UPDATED	Dec 18 23:13:42 2000
BY	bfrench
SIZE	C
REV	

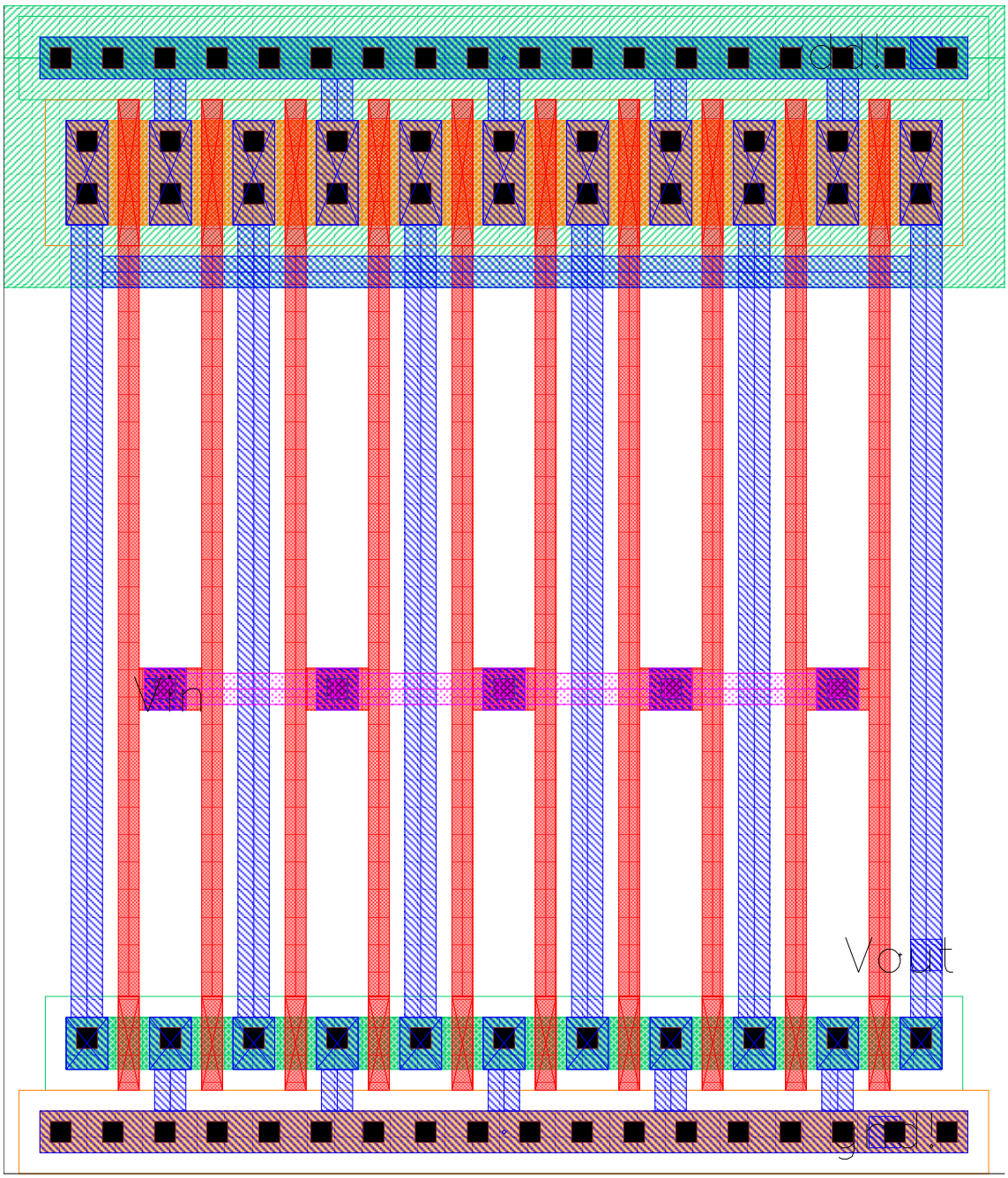
3-bit Binary Up–Down Counter

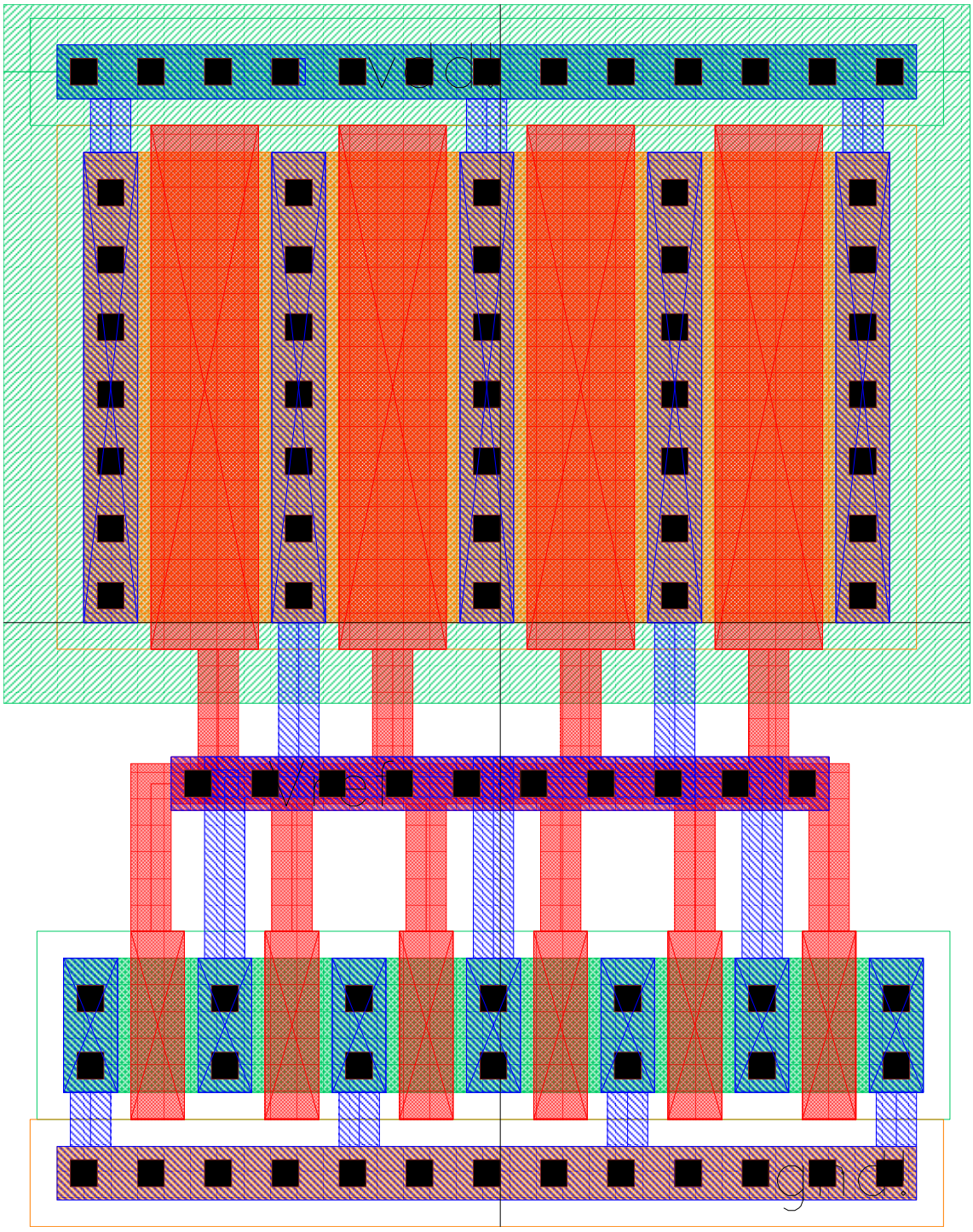
SHEET 1 of 1

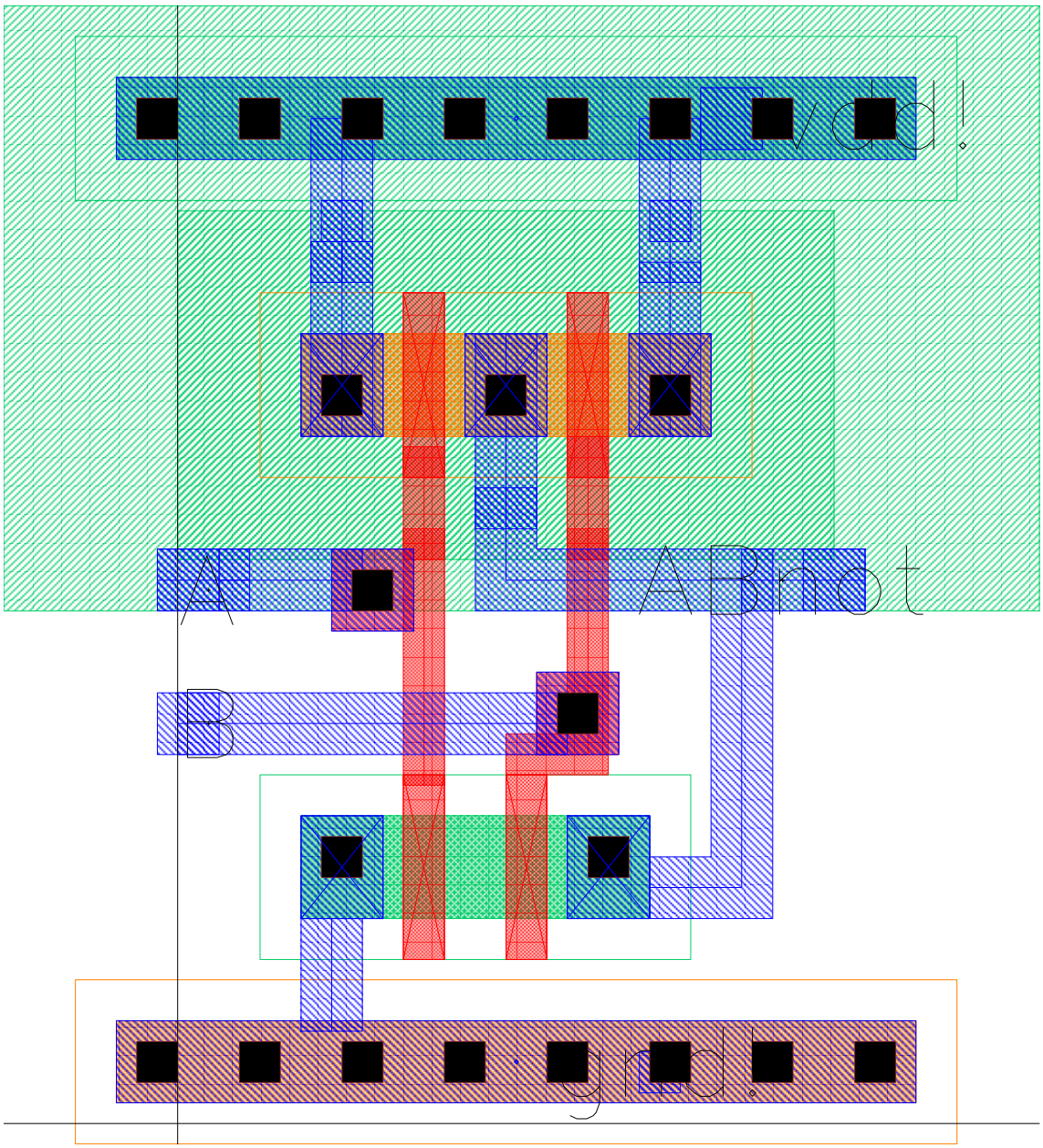
Appendix B: Layouts

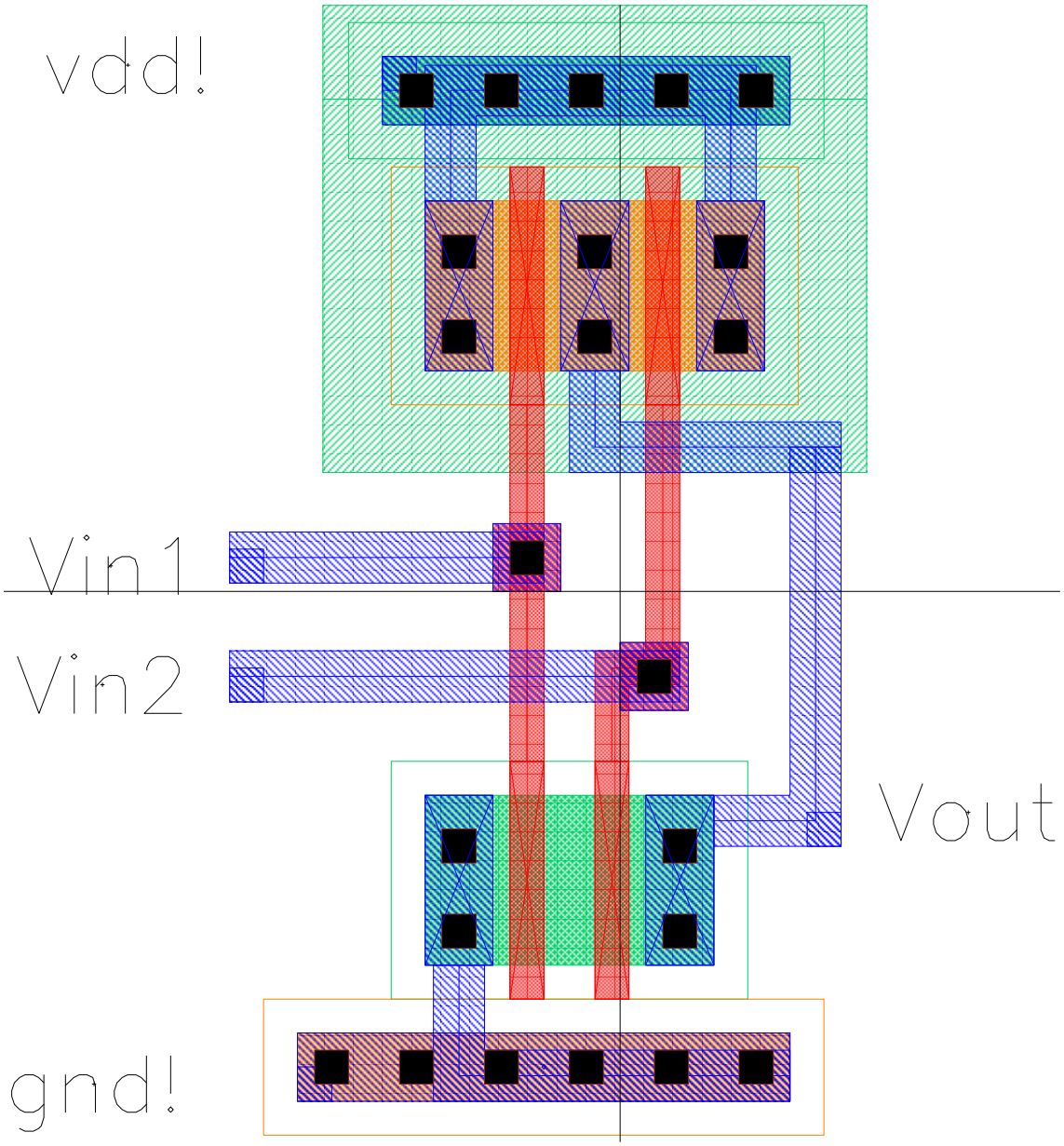
- 100x minimum size inverter
- 10x minimum size inverter
- 2.15V voltage divider
- 2-input NAND gate
- Alternate 2-input NAND gate
- 3-to-8 bit decoder
- 3-input NAND gate
- Alternate 3-input NAND gate
- 4-input NAND gate
- 4V voltage divider
- 5-input NAND gate
- AutoRanger Team Logo
- Astable
- Comparator
- Decimal to BCD encoder
- D Flip-Flop
- Full AutoRanger Circuit
- Full Chip layout
- Input Package
- Inverter
- Alternate Inverter
- Inverter for the Ring Oscillator
- Alternate Inverter 2
- J-K Flip-Flop
- 31-Stage Ring Oscillator
- S-R Latch
- Transmission Gate
- 3-bit Up-Down Counter

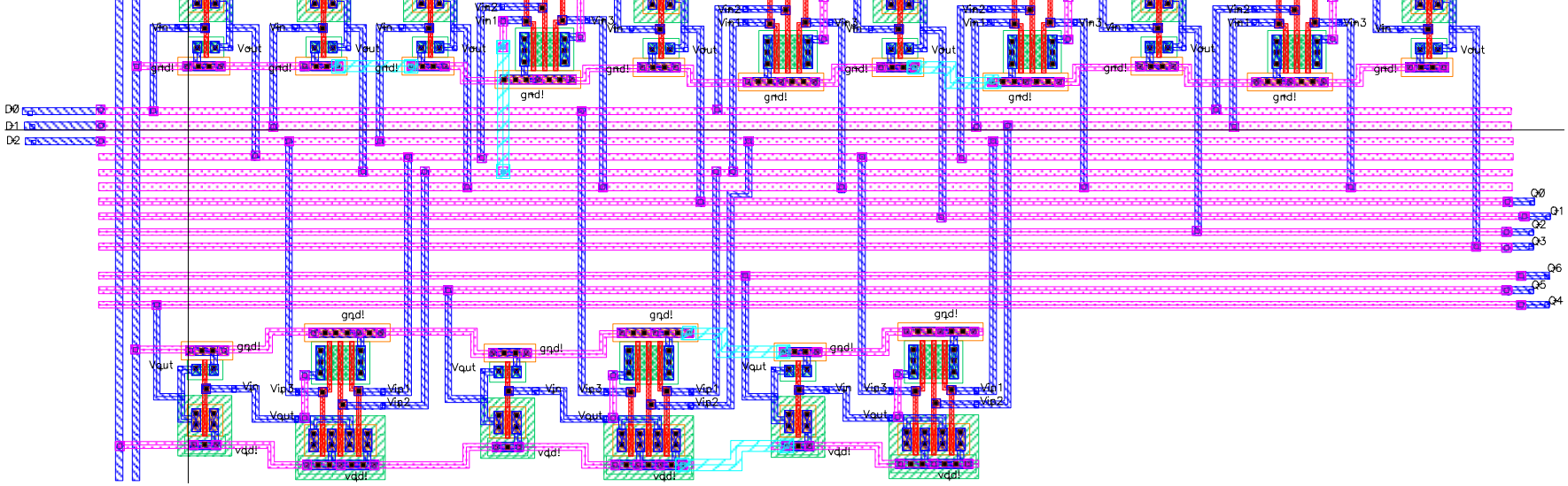

VDD
VSS

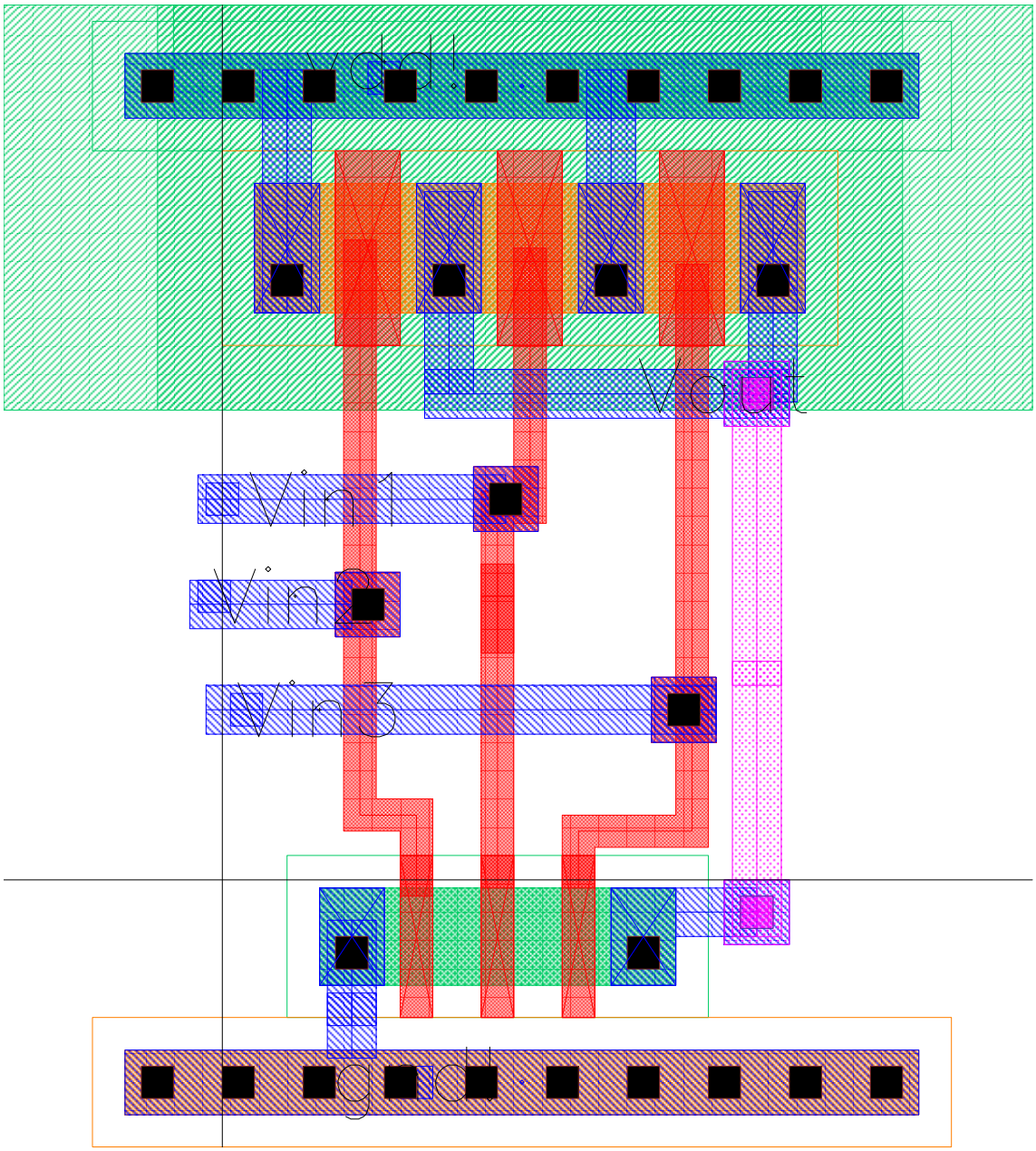

vdd
Vin
Vout
gnd

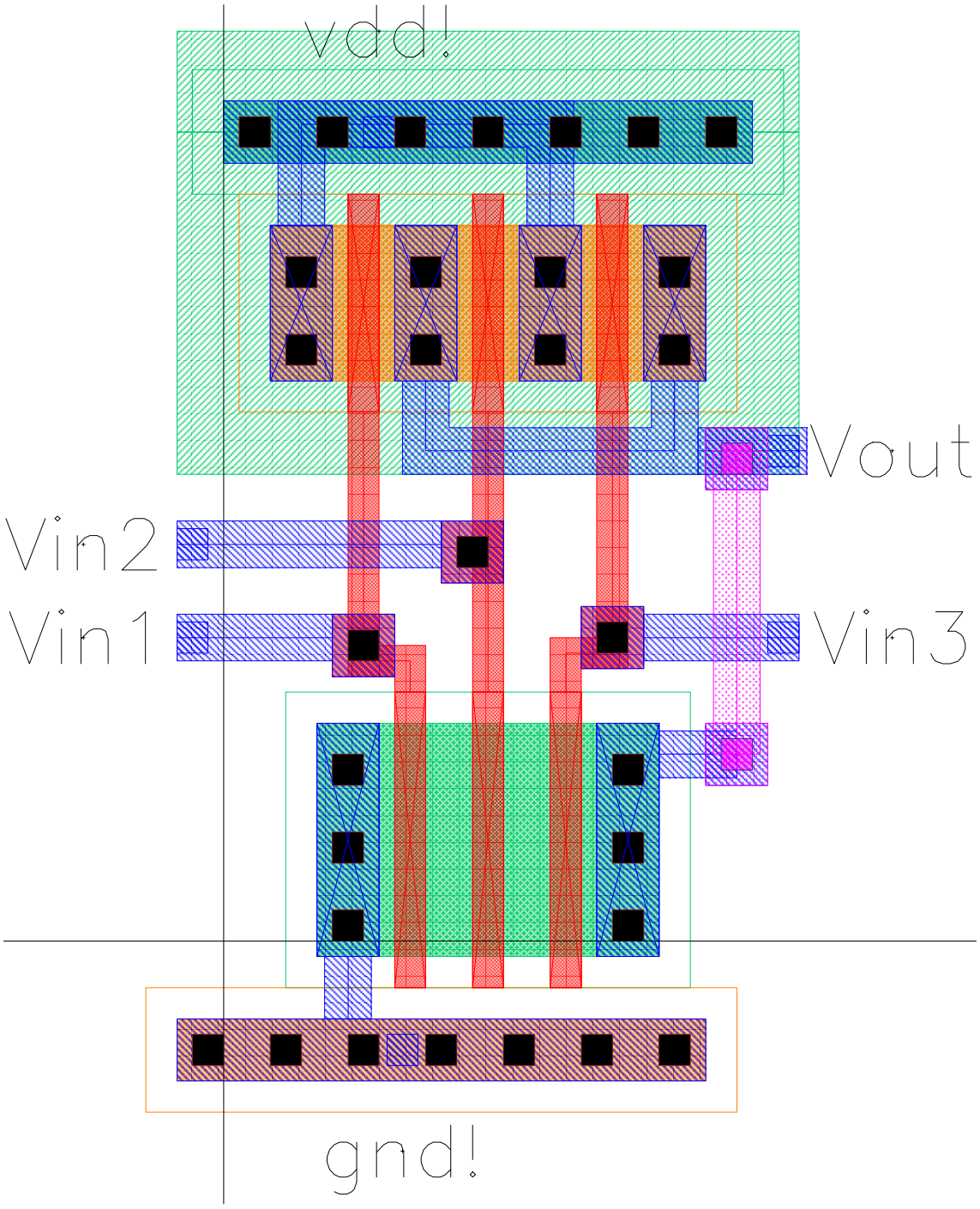

VDD
VSS
GND

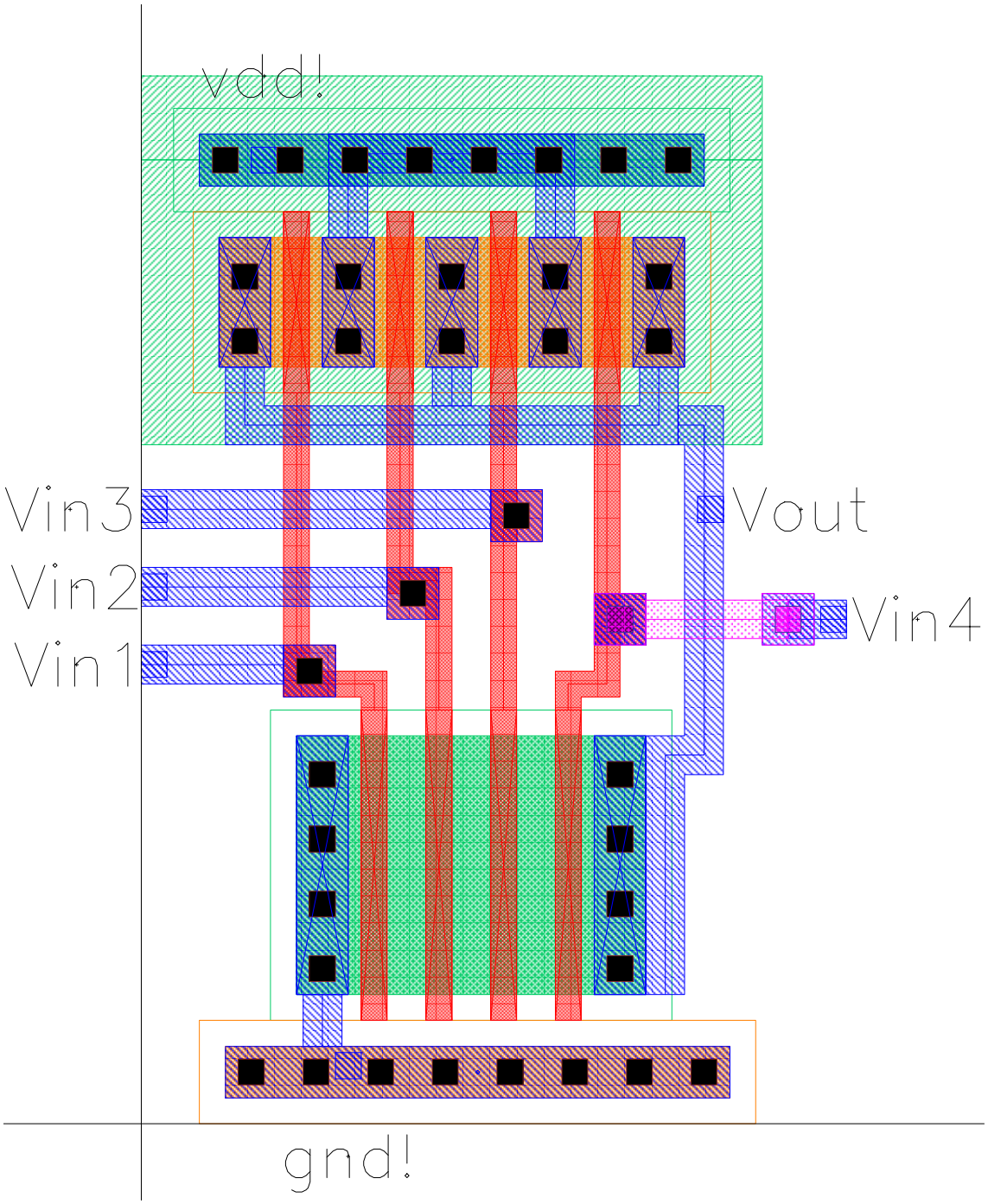

Vdd!
A
A_bar
B
Gnd!

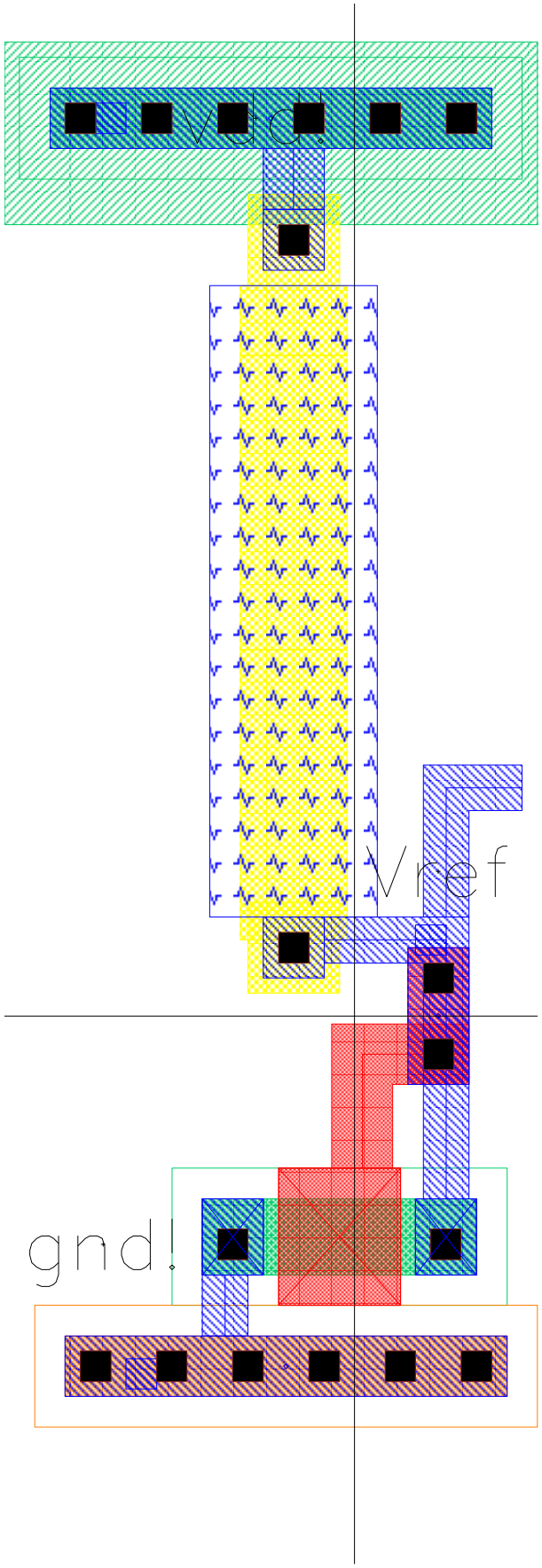

vdd!
Vin1
Vin2
Vout
gnd!

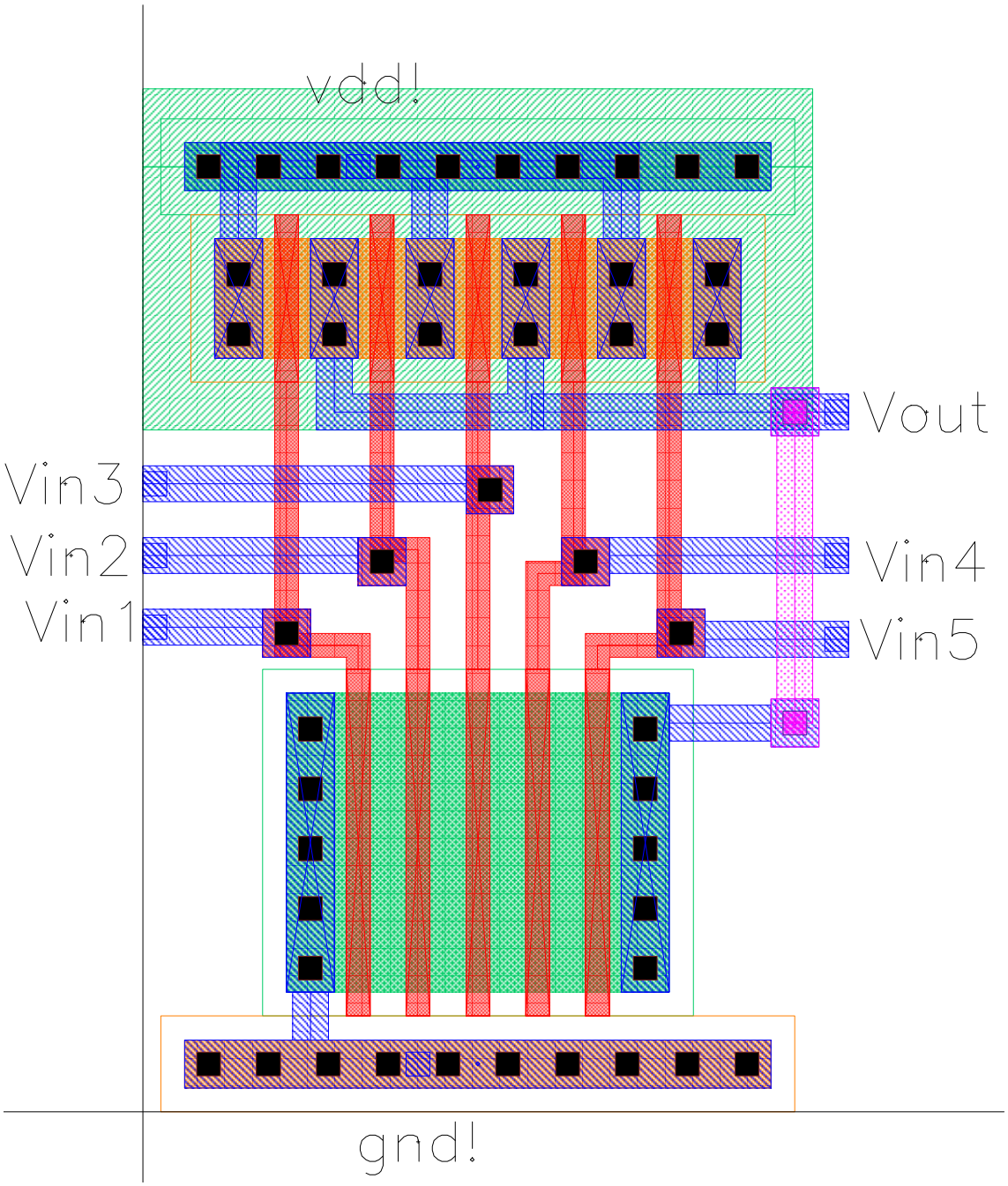

D0
D1
D2
Q0
Q1
Q2
Q3
Q4
Q5
Q6
gnd!
vdd!
Vout
Vin
Vin1
Vin2
Vin3

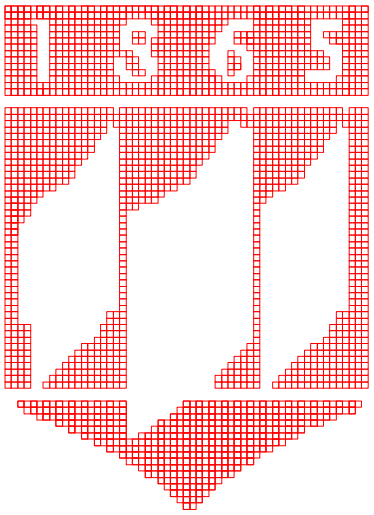

vdd!
vout
vin1
vin2
vin3
gnd!


vdd!
Vout
Vin2
Vin1
Vin3
gnd!


vdd!
Vin3
Vin2
Vin1
Vout
Vin4
gnd!


vdd!
Vref
gnd!


vdd!
Vout
Vin3
Vin2
Vin4
Vin1
Vin5
gnd!



THE UNIVERSITY OF

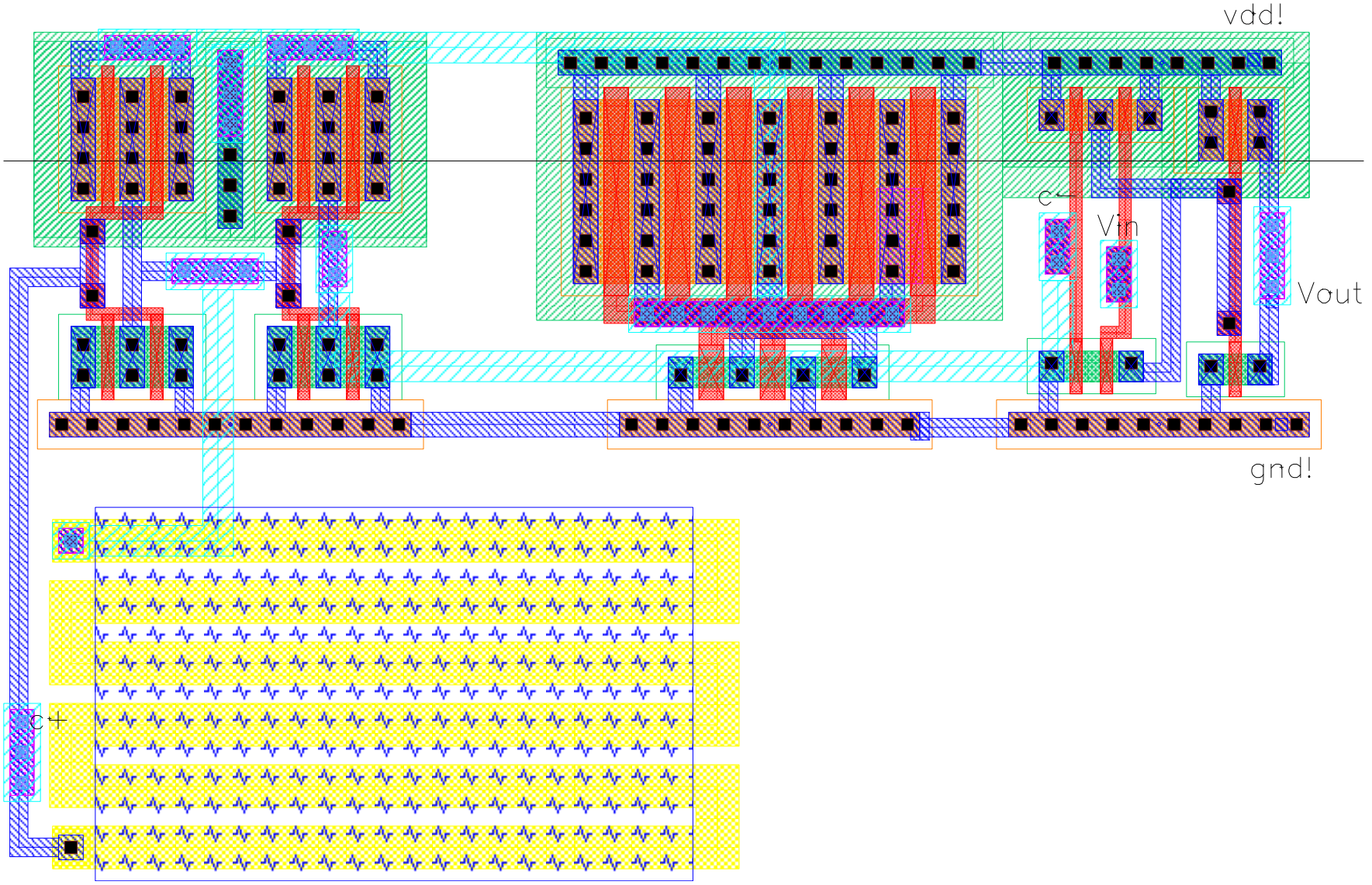
MAINE

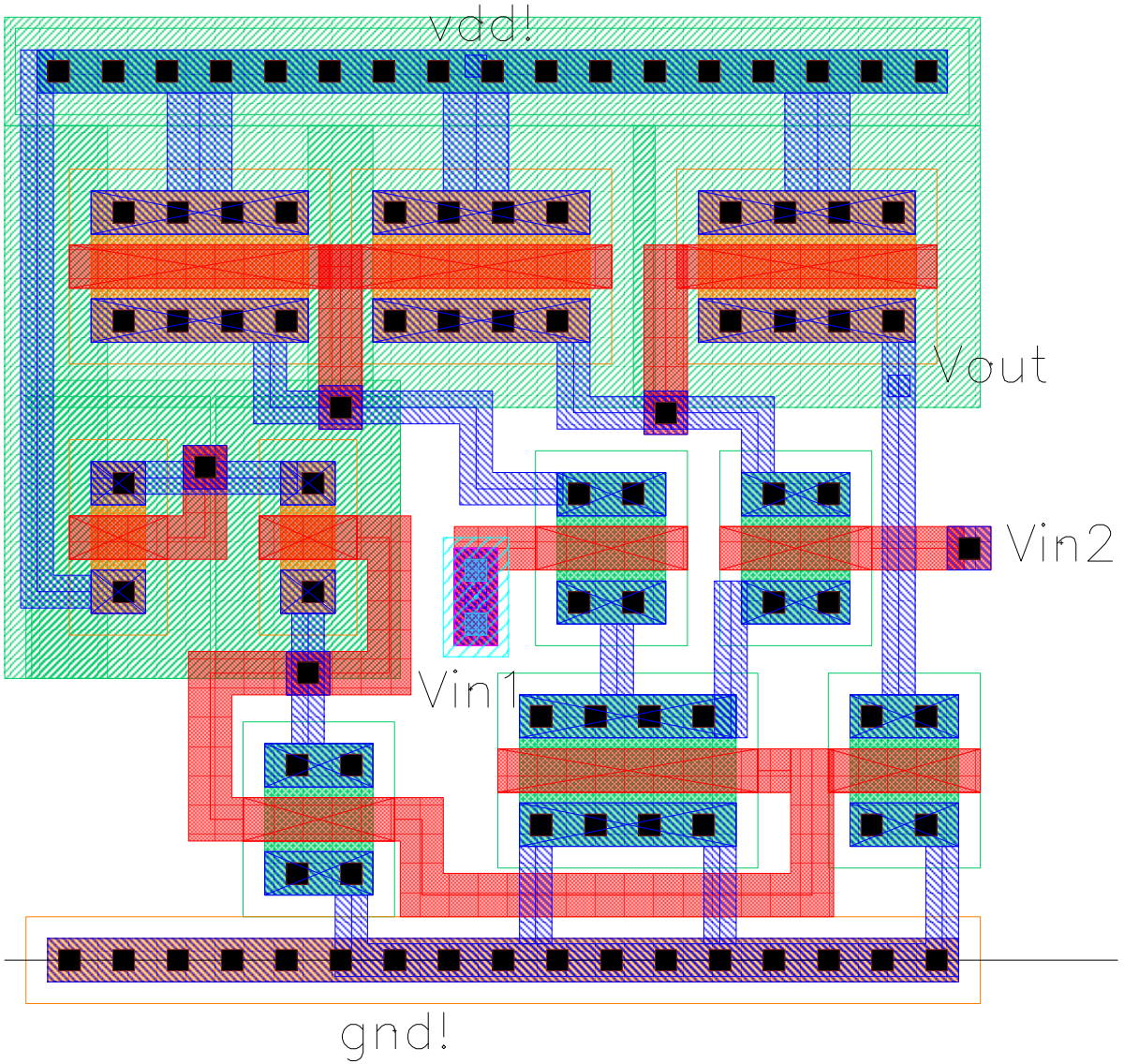
Electrical and Computer Engineering

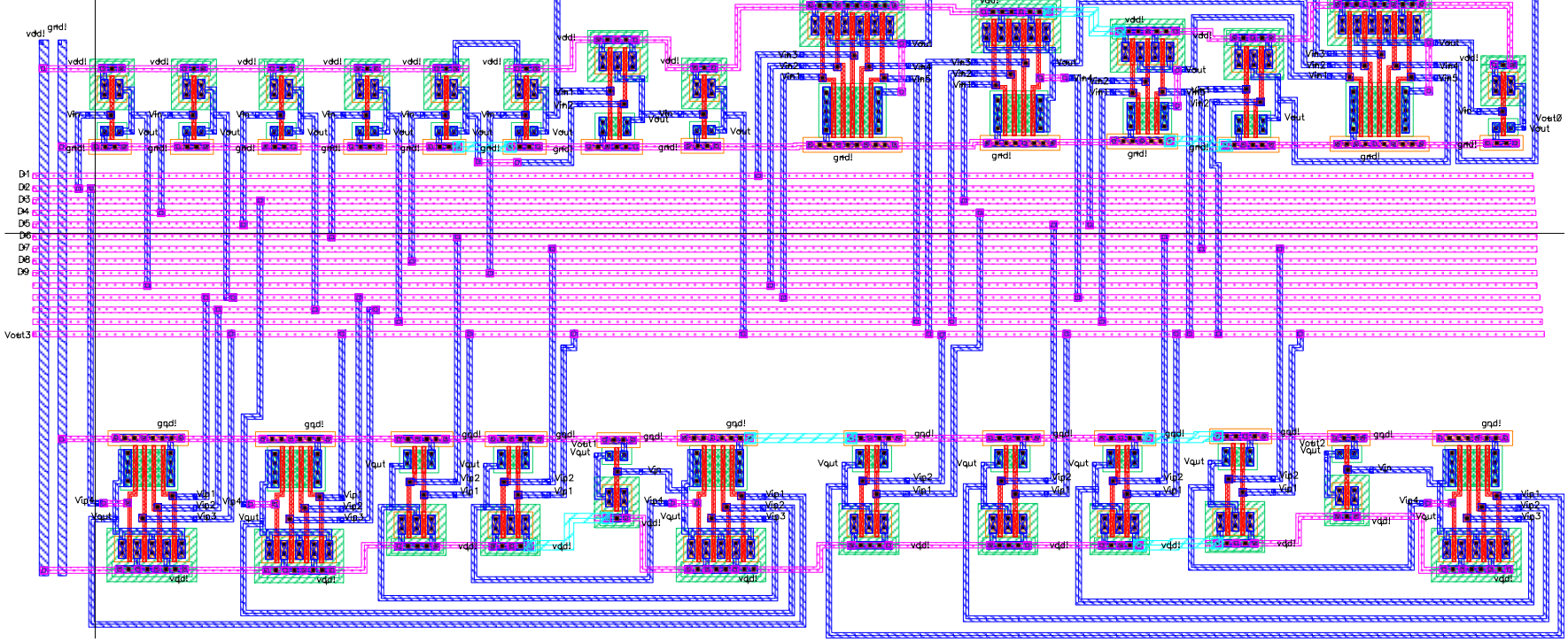
Jesse Cousins

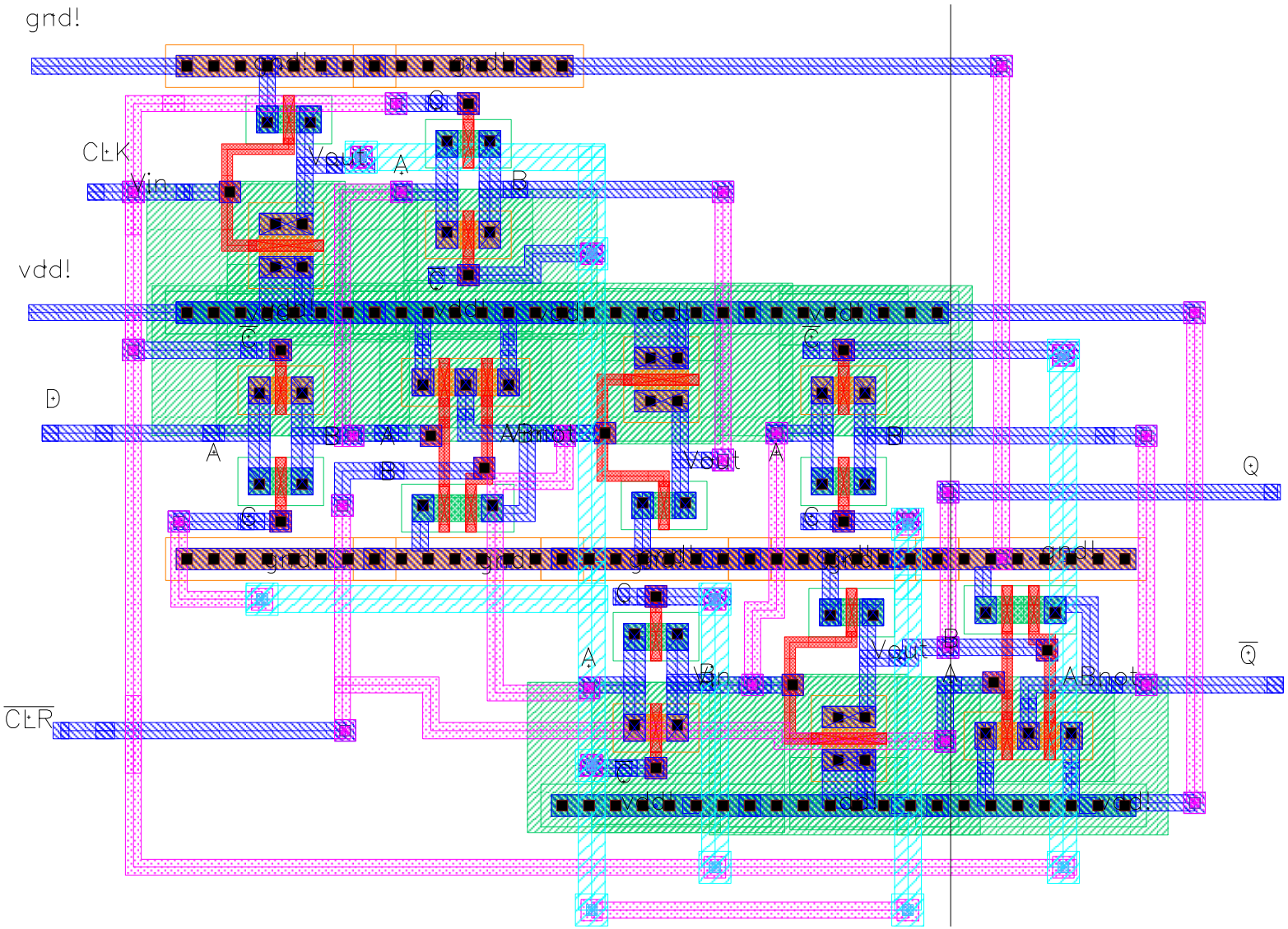
Vlas Karasz

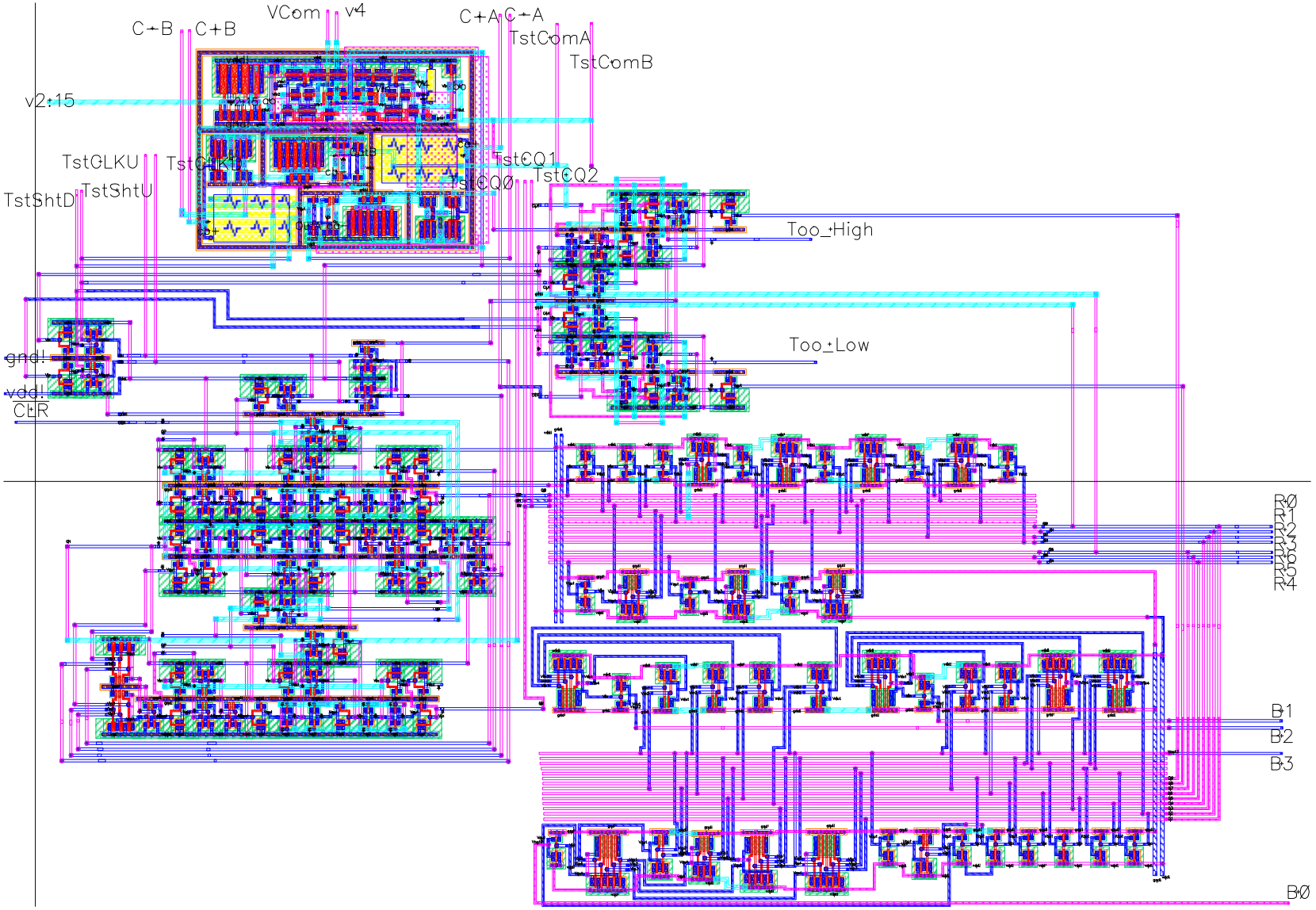
Brian French

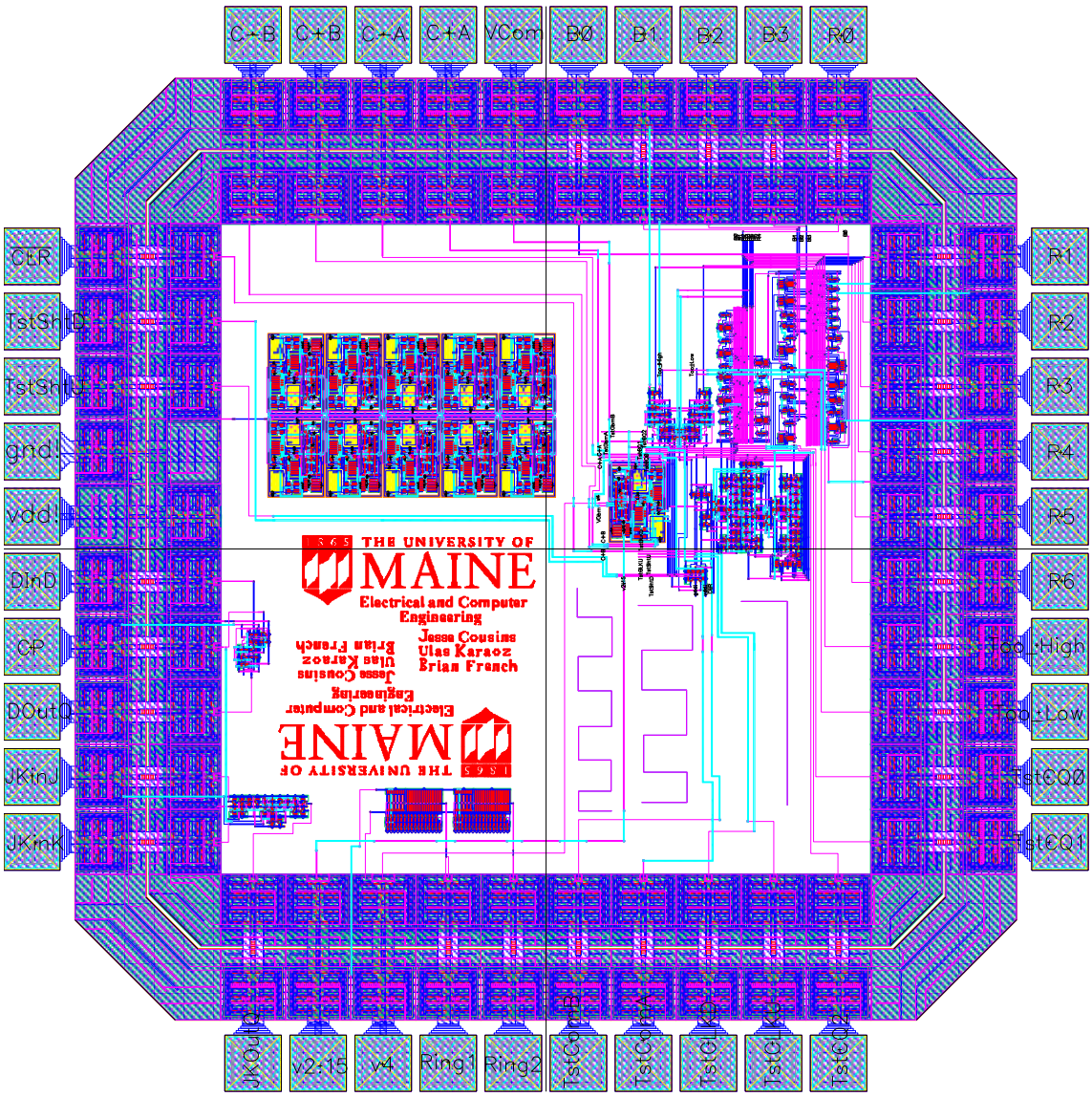

vdd!
c-
Vin
Vout
gnd!
c+

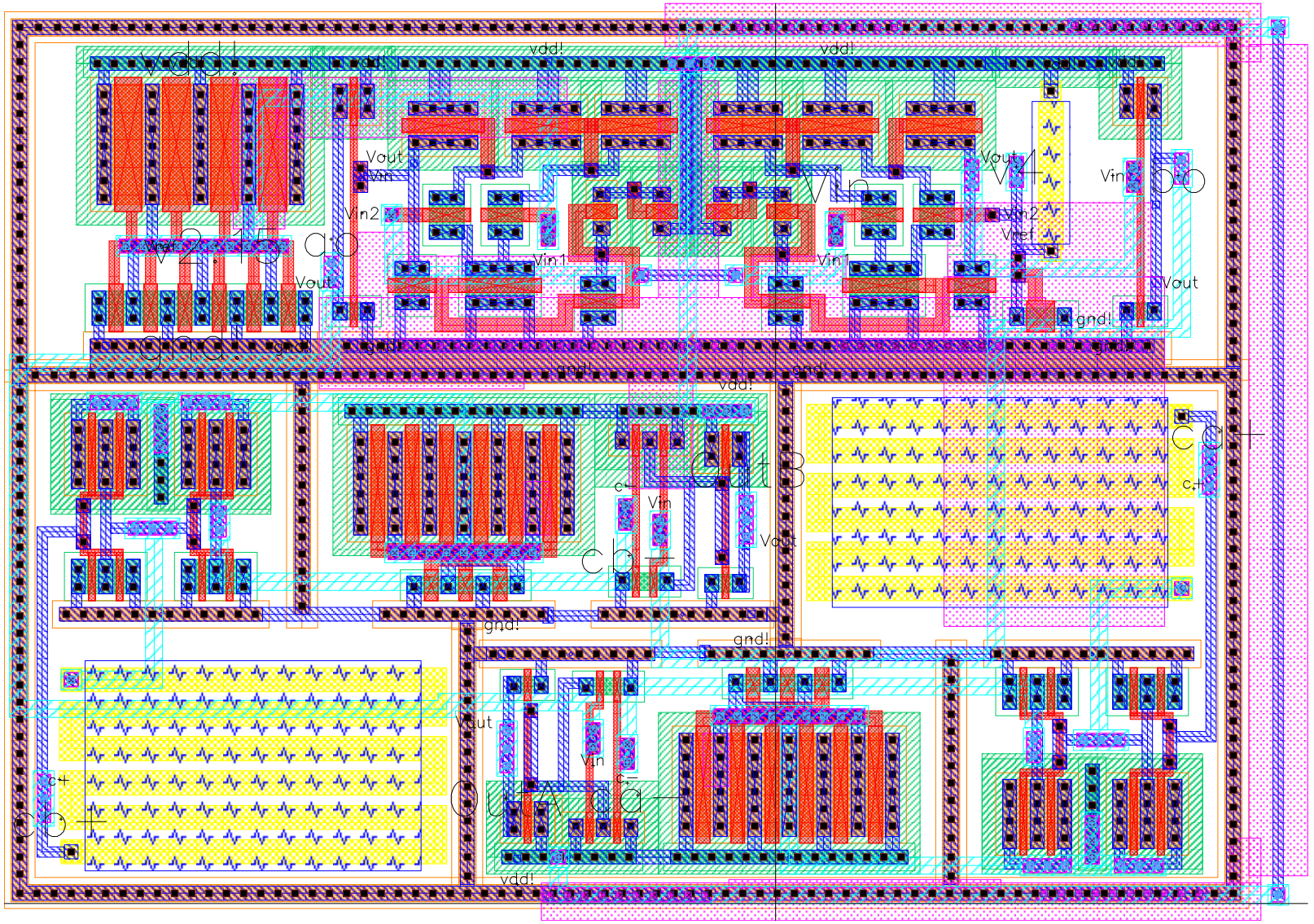

vdd!
Vout
Vin2
Vin1
gnd!

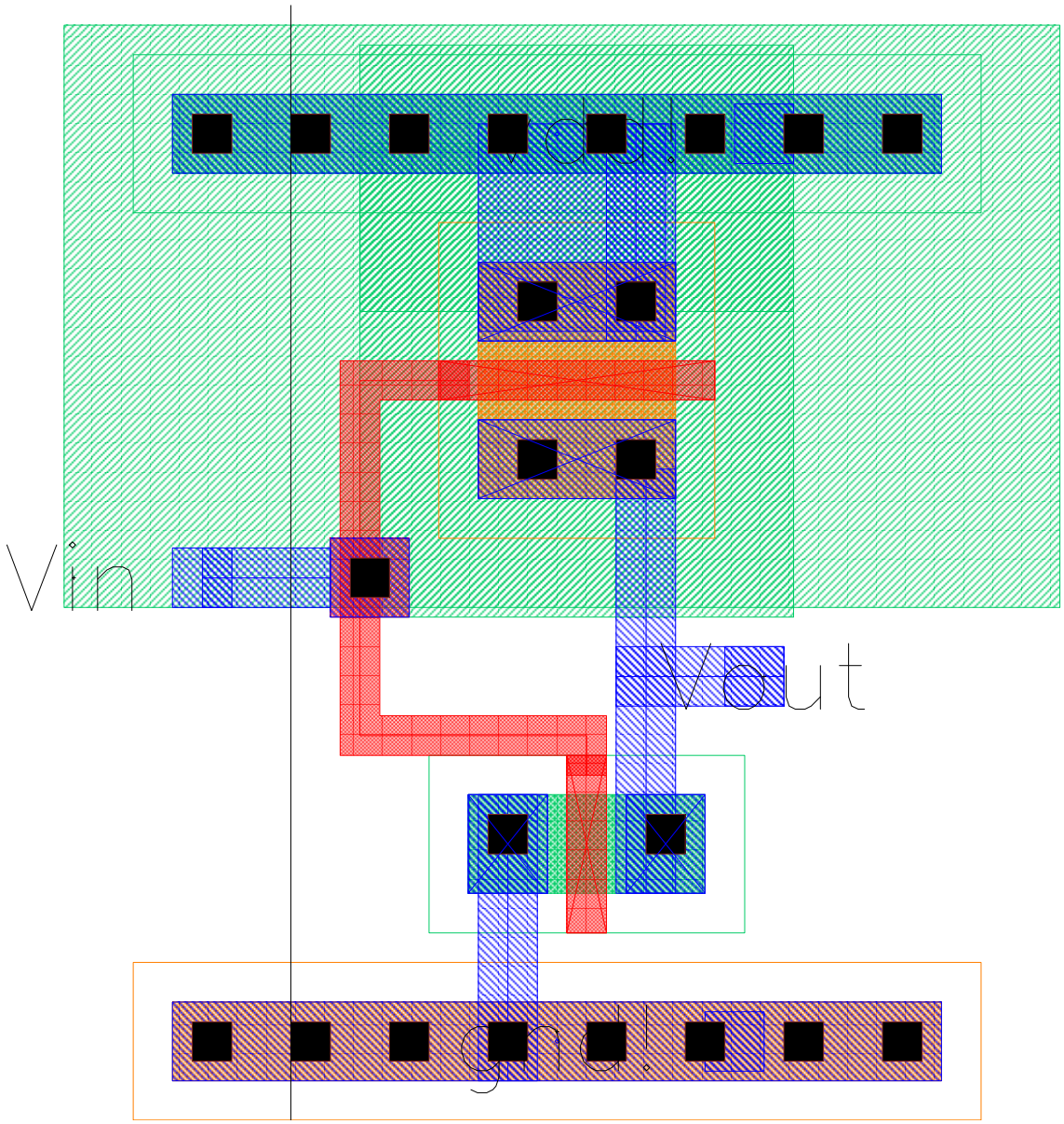

vdd!
gnd!
D1
D2
D3
D4
D5
D6
D7
D8
D9
Vout3
Vout0
Vout
Vin
Vip1
Vip2
Vip3
Vip4
Vin1
Vin2
Vin3
Vin4
Vin5
Vout1
Vout2

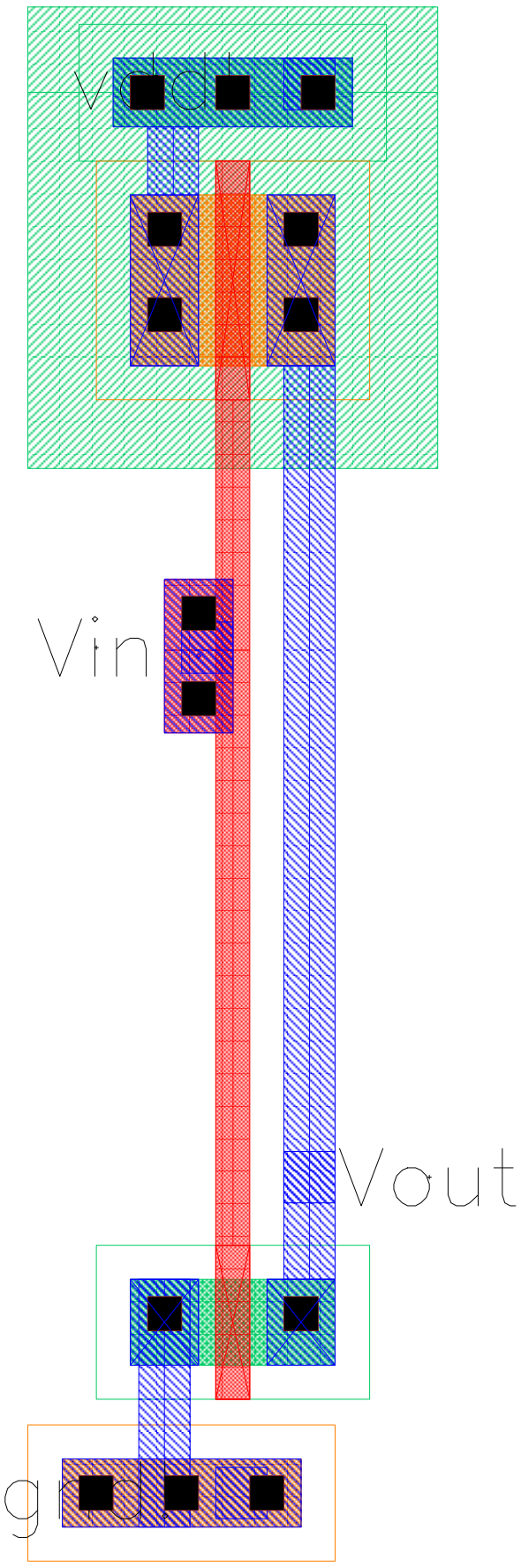

gnd!
CLK
Vin
vdd!
D
CLR
Q
$\overline{Q}$

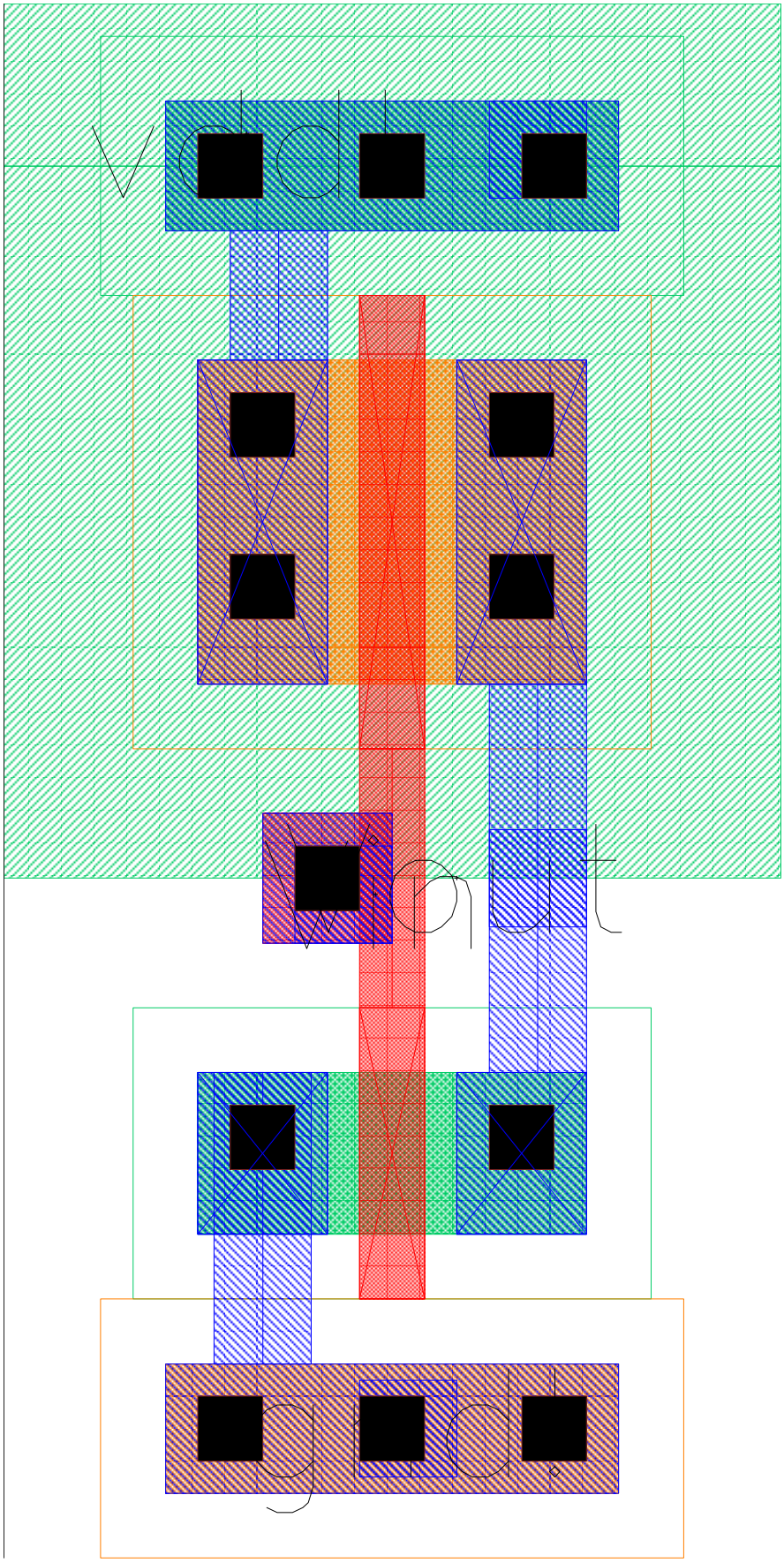

VCom
v4
C+B
C+B
C+A
C+A
TstComA
TstComB
v2:15
TstGLKU
TstGLKU
TstShtD
TstShtU
TstCQ1
TstCQ2
TstCQØ
Too_High
Too_Low
gnd!
vdd!
CLR
RØ
R1
R2
R3
R4
R5
R6
R7
B1
B2
B3
BØ

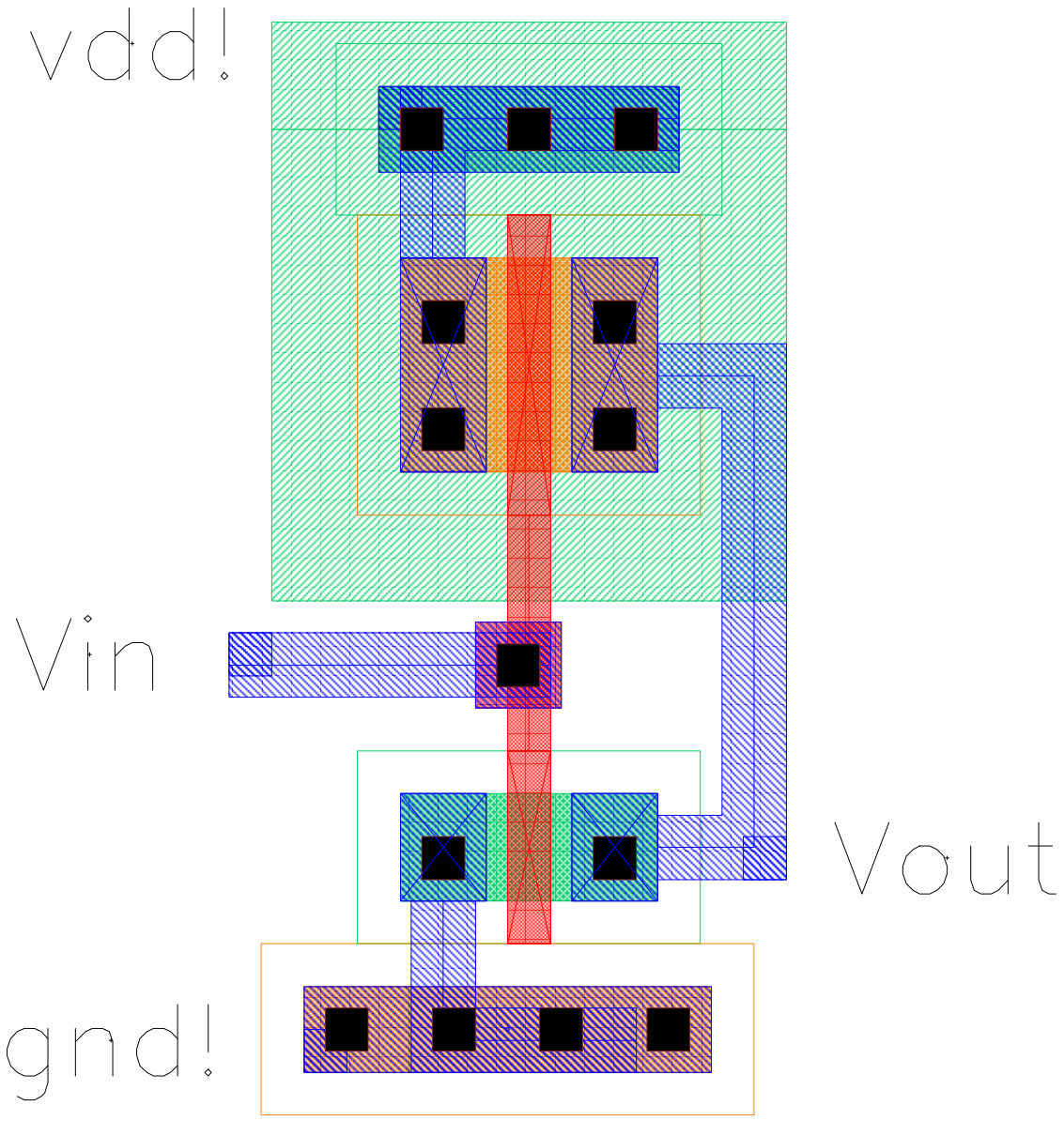

C-B
C+B
C-A
C+A
VCom
BØ
B1
B2
B3
RØ
CtR
TstShftD
TstShftQ
gnd
vdd
DInD
CP
DOutQ
JKinJ
JKinK
R1
R2
R3
R4
R5
R6
Test High
Test Low
TstCQØ
TstCQ1
JKOut
V2.15
V4
Ring1
Ring2
TstCpeB
TstCpeA
TstCJK
TstCK
TstCQb
THE UNIVERSITY OF
MAINE
Electrical and Computer
Engineering
Jesse Cousins
Ulas Karaoz
Brian French

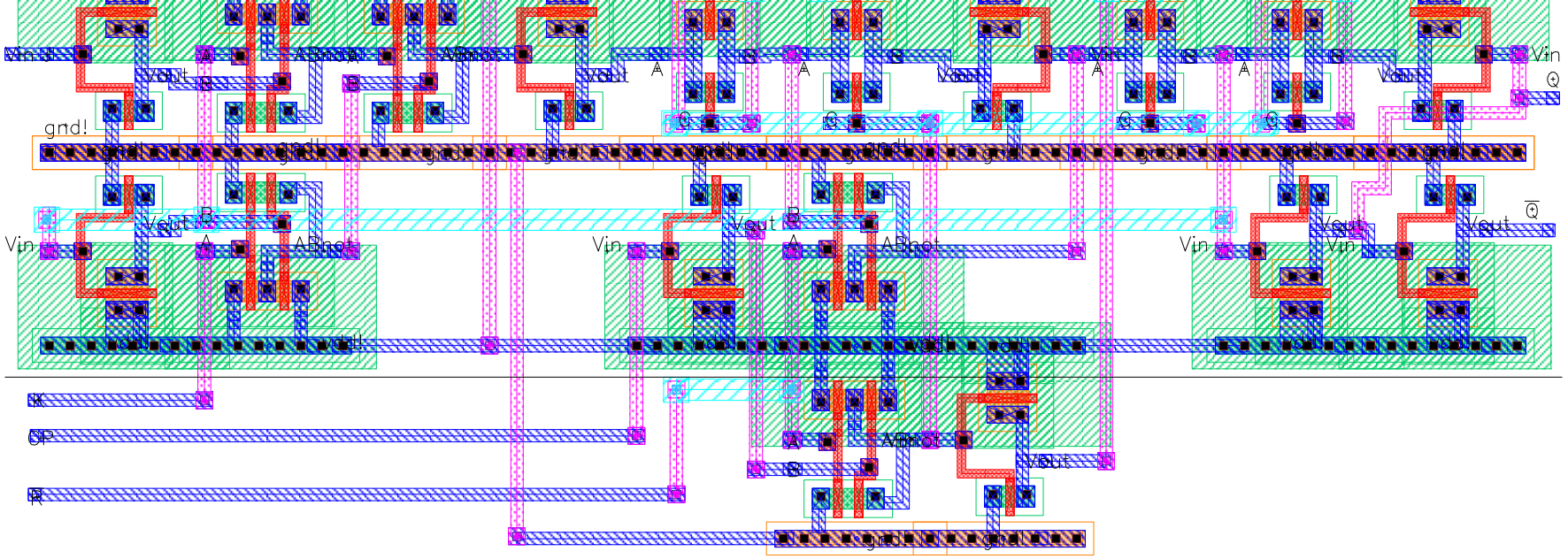

vdd!
gnd!
Vout
Vin
Vin2
Vref
Vin1
c+
c-

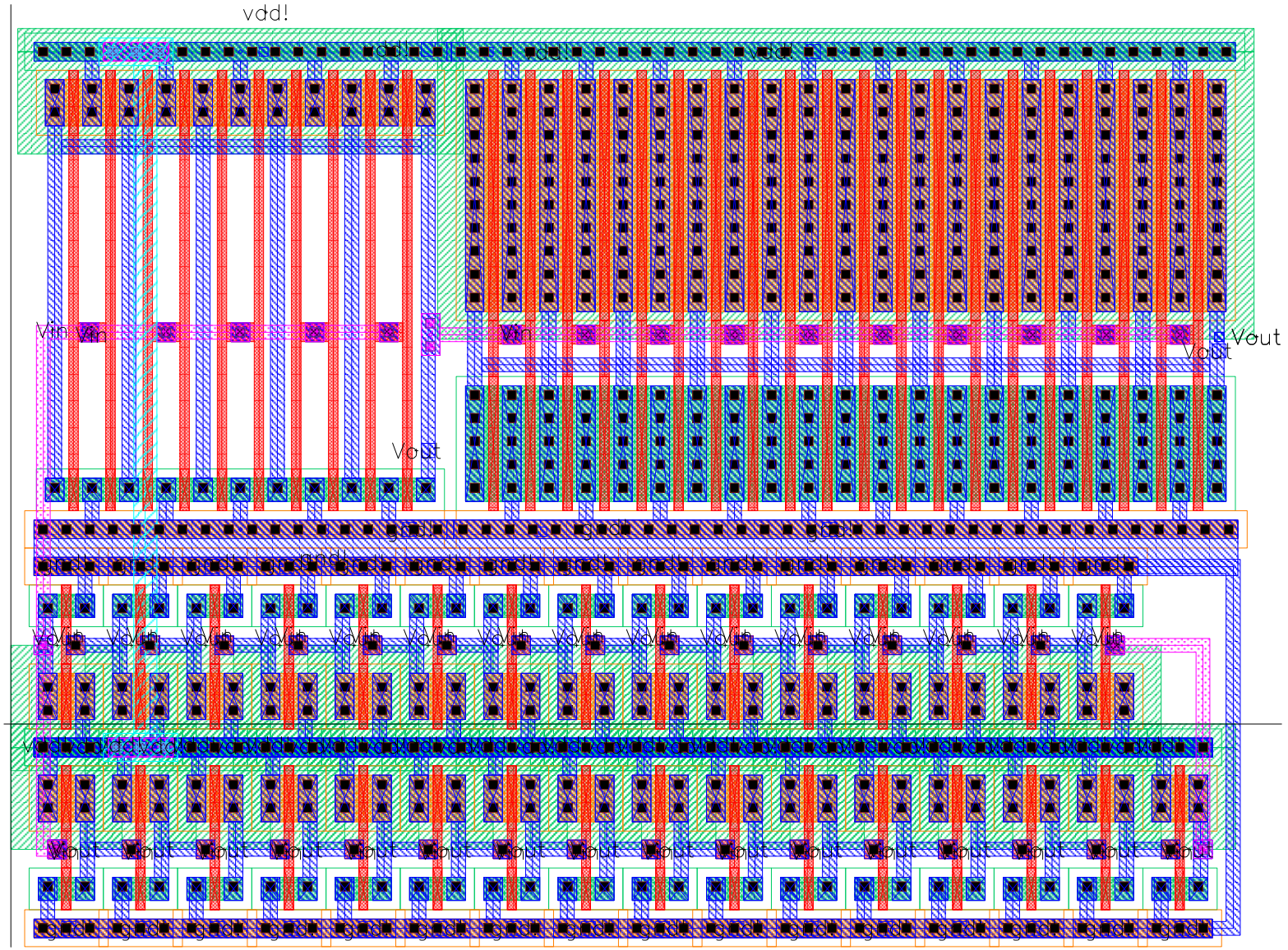

Vdd
Vin
Vout
Gnd

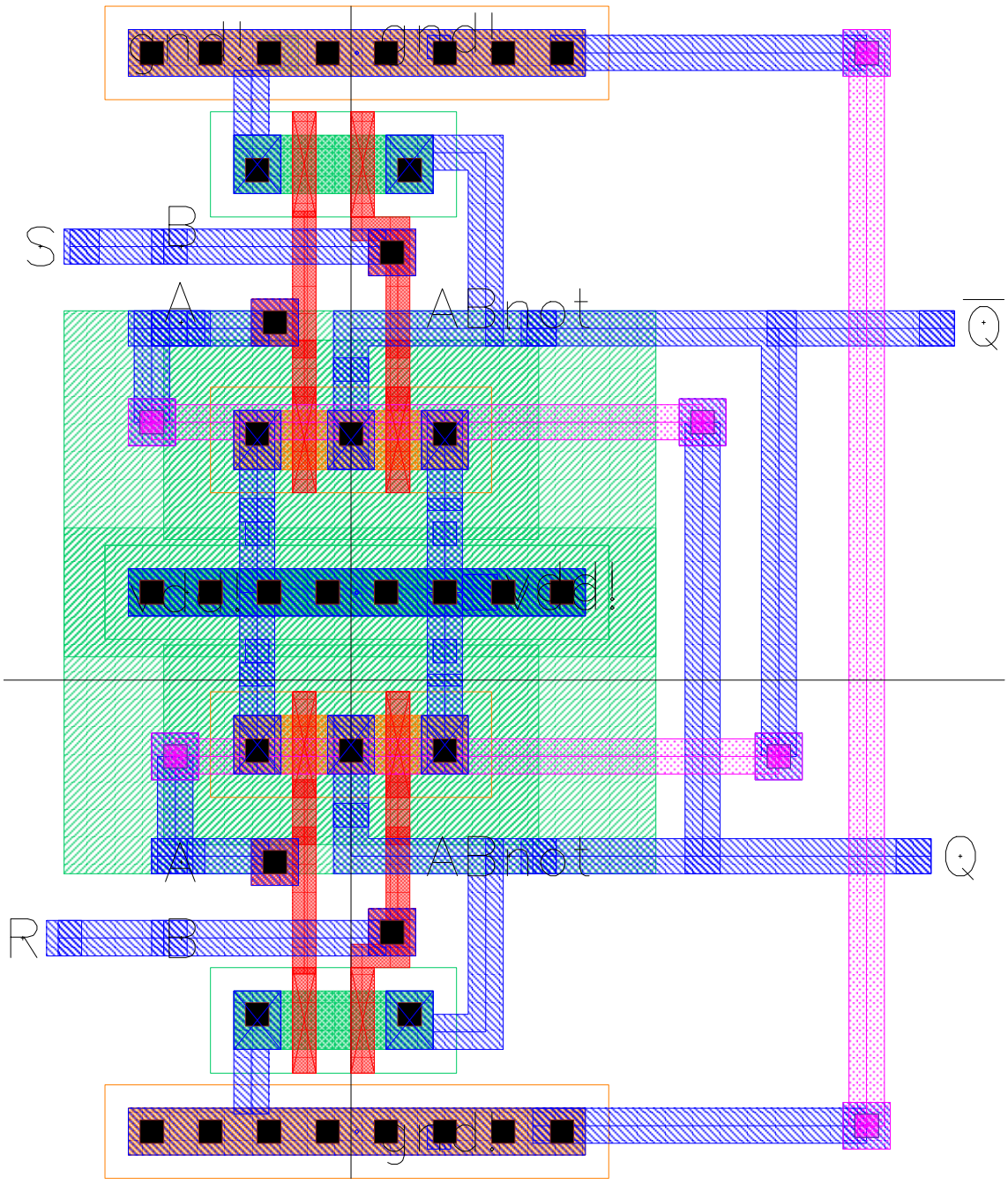

Vdd
Vin
Vout
gnd

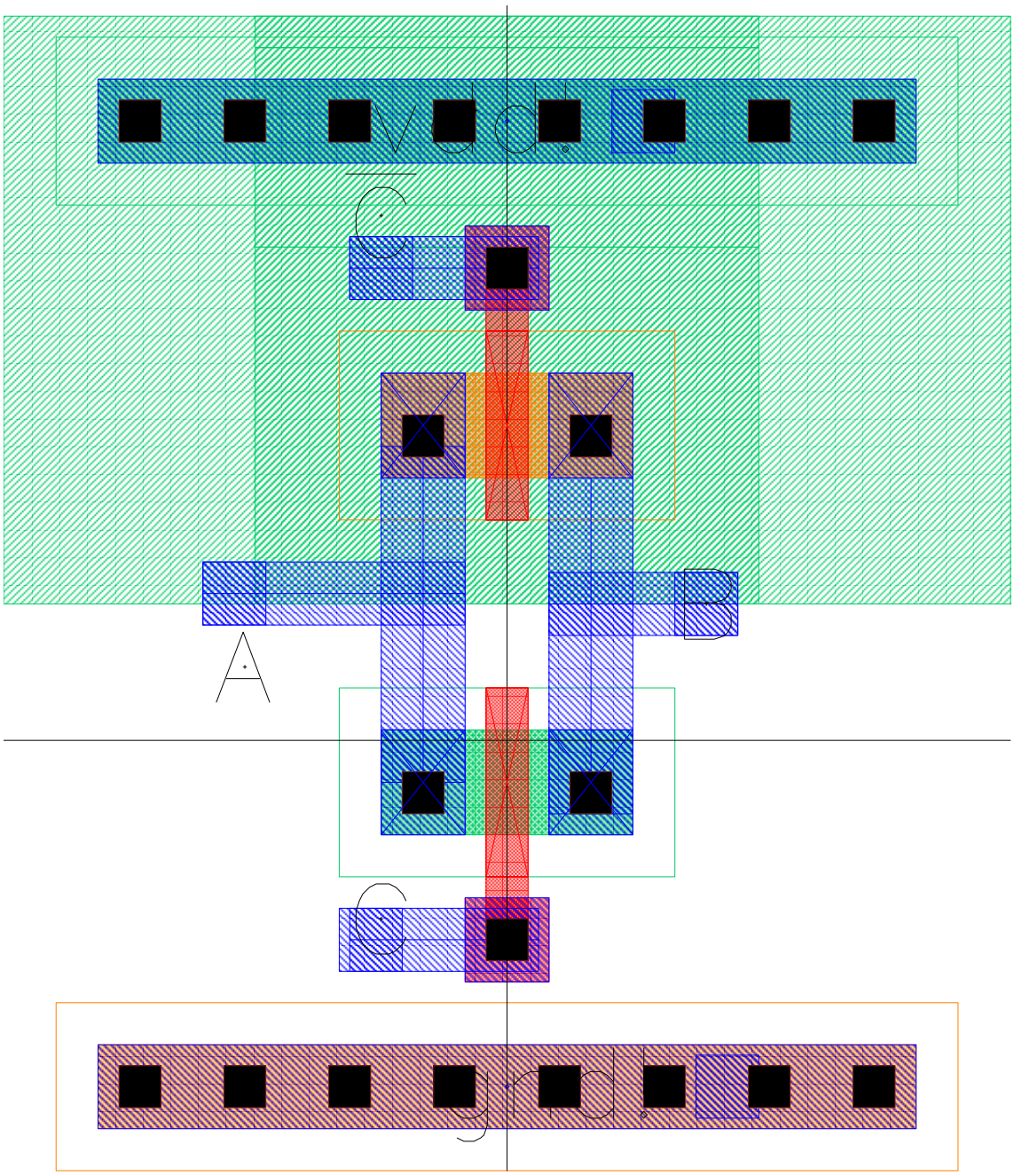


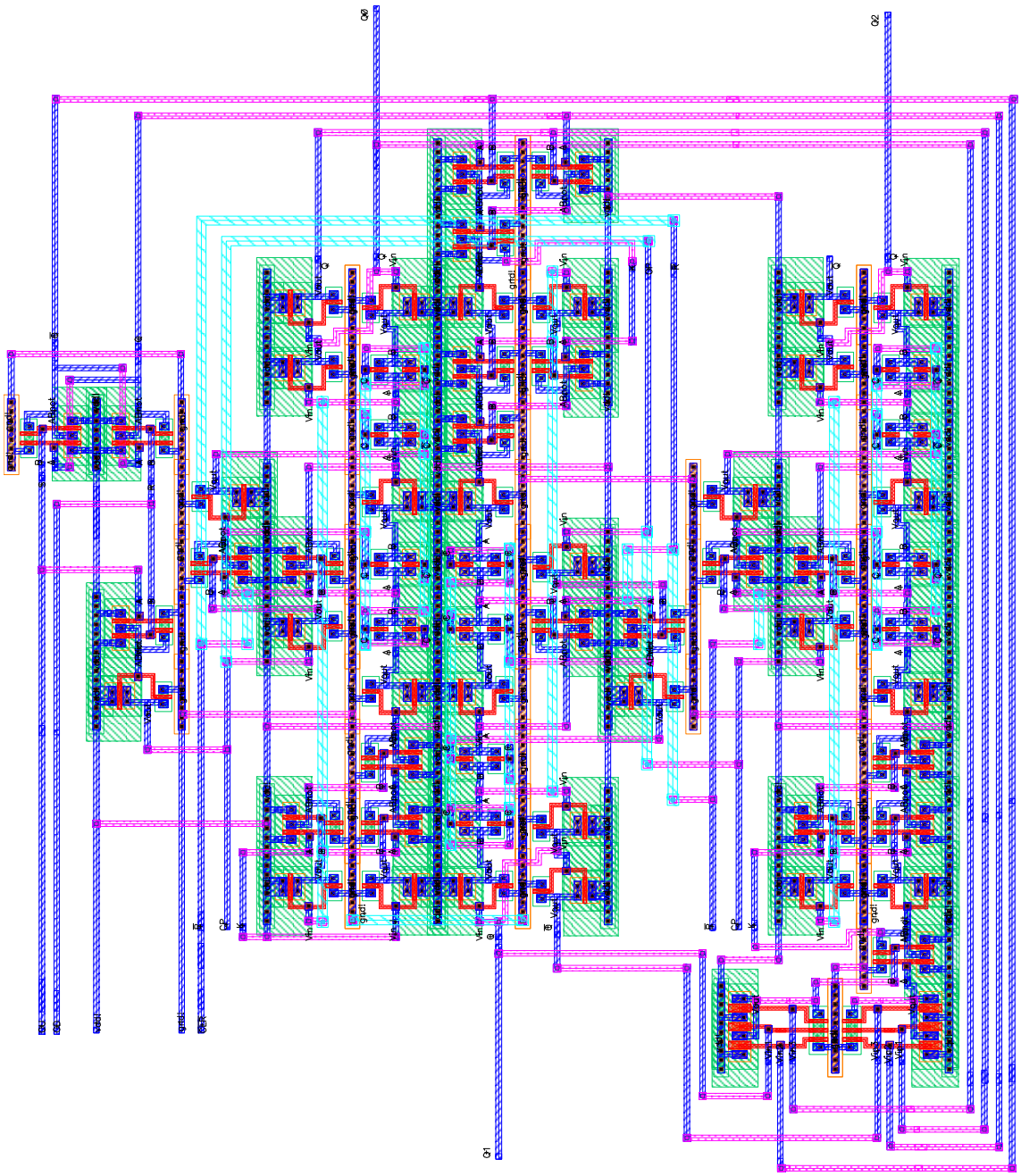

vdd!
Vin
Vout
gnd!


gnd!
Vout
Vin
A
B
ABnot
Q
$\overline{Q}$
CK
CP
R


vdd!
Vin
Vout
VoSt
gnd!


gnd!
S
B
A
ABnot
Q
vdd!
A
ABnot
Q
R
B
gnd!





Appendix C: Pin-Out

Pin Number	Function	Pin Number	Function
1	Vcom	21	TstComB
2	C+A	22	TstComA
3	C-A	23	TstCLKD
4	C+B	24	TstCLKU
5	C-B	25	TstCQ2
6	CLR	26	TstCQ1
7	TstShtD	27	TstCQ0
8	TstShtU	28	Too_Low
9	gnd	29	Too_High
10	vdd	30	R6
11	DinD	31	R5
12	CP	32	R4
13	DOutQ	33	R3
14	JKInJ	34	R2
15	JKInK	35	R1
16	JKOutQ	36	R0
17	V2.15	37	B3
18	V4	38	B2
19	Ring1	39	B1
20	Ring2	40	B0

Appendix D: Top Level Verification

@(#)$CDS: LVS version 4.4.5 10/28/1999 15:28 (cds11182) $

Like matching is enabled.
Net swapping is enabled.
Creating /usr/grads/jcousins/ECE5473/LVS/xref.out file.
Using terminal names as correspondence points.

Net-list summary for /usr/grads/jcousins/ECE5473/LVS/layout/netlist

count	
754	nets
40	terminals
143	res
1296	pmos
1263	nmos

Net-list summary for /usr/grads/jcousins/ECE5473/LVS/schematic/netlist

count	
754	nets
40	terminals
143	res
624	pmos
635	nmos

Terminal correspondence points

1	B0
2	B1
3	B2
4	B3
5	C+A
6	C+B
7	C-A
8	C-B
9	CP
10	DInD
11	DOutQ
12	JKOutQ
13	JKinJ
14	JKinK
15	R0
16	R1
17	R2
18	R3
19	R4
20	R5
21	R6
22	Ring1
23	Ring2
24	Too_High
25	Too_Low
26	TstCLKD
27	TstCLKU

28	TstCQ0
29	TstCQ1
30	TstCQ2
31	TstComA
32	TstComB
33	TstShtD
34	TstShtU
35	VCom
36	_CLR
37	gnd!
38	v2.15
39	v4
40	vdd!

The net-lists match logically but have mismatched parameters.

	layout	schematic
	instances	
un-matched	0	0
rewired	0	0
size errors	80	4
pruned	0	0
active	2702	1402
total	2702	1402
	nets	
un-matched	0	0
merged	0	0
pruned	0	0
active	754	754
total	754	754
	terminals	
un-matched	0	0
matched but different type	0	0
total	40	40

Probe files from /usr/grads/jcousins/ECE5473/LVS/schematic

devbad.out:

netbad.out:

mergenet.out:

termbad.out:

prunenet.out:

prunedev.out:

audit.out:
I /I50/I41/M1
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /I55/I41/M1
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /I50/I41/M0
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /I55/I41/M0
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

Probe files from /usr/grads/jcousins/ECE5473/LVS/layout

devbad.out:

netbad.out:

mergenet.out:

termbad.out:

prunenet.out:

prunedev.out:

audit.out:
I /+2031
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2034
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2035
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2038
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2039
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2042
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2051
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2054
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2055
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2070
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2071

? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2086
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2087
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2090
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2091
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2094
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2095
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2102
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2115
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+2105
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1836
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1839
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1840
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1843
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1870
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1873
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1874
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1877
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1878
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1881

? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1882
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1885
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1894
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1897
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1898
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1913
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1926
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1929
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1932
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+1930
? Combined device: Transistor width mismatch: layout 300.00 um, schematic 300.00 um

I /+738
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+741
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+744
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+775
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+776
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+779
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+788
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+791
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+792

? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+795
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+796
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+799
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+800
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+803
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+804
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+807
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+808
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+811
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+814
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+812
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+575
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+578
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+579
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+582
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+583
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+586
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+587
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+590

? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+593
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+624
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+625
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+628
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+637
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+640
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+641
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+644
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+645
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+648
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+651
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

I /+649
? Combined device: Transistor width mismatch: layout 150.00 um, schematic 150.00 um

Appendix E: Biography of Authors

Brian French is a native of the South Portland Area. He graduated with a BS in Computer Engineering May 98. After graduating Brian worked in the Boston Area at SAIC as a software engineer. His main focus was on deleoping AI based simulations for the Deparment of Defence. He is currently pursuing his MS in computer engineering and should graduate Dec 01. He can be reached at bfrench@eece.maine.edu

Jesse Cousins was born in Bar Harbor, Maine, in 1978. He graduated with a BS in Electrical Engineering from the University of Maine in May 2000. He is currently pursuing a Master's in Electrical Engineering at UMaine, concentrating in microelectronic design and process. He can be reached at jcousins@eece.maine.edu

Ulas Karaoz is a Senior Electrical and Computer Engineering student in Bilkent University in Ankara-TURKEY; expected graduation in May 2001.