EECE6080 - HW 4

Max Thrun

December 4, 2013

Background & Design

The purpose of this lab was to implement a carry chain usuable for for an n-bit adder. A diagram illustating the relationship between the carry chain and the adders is shown below.

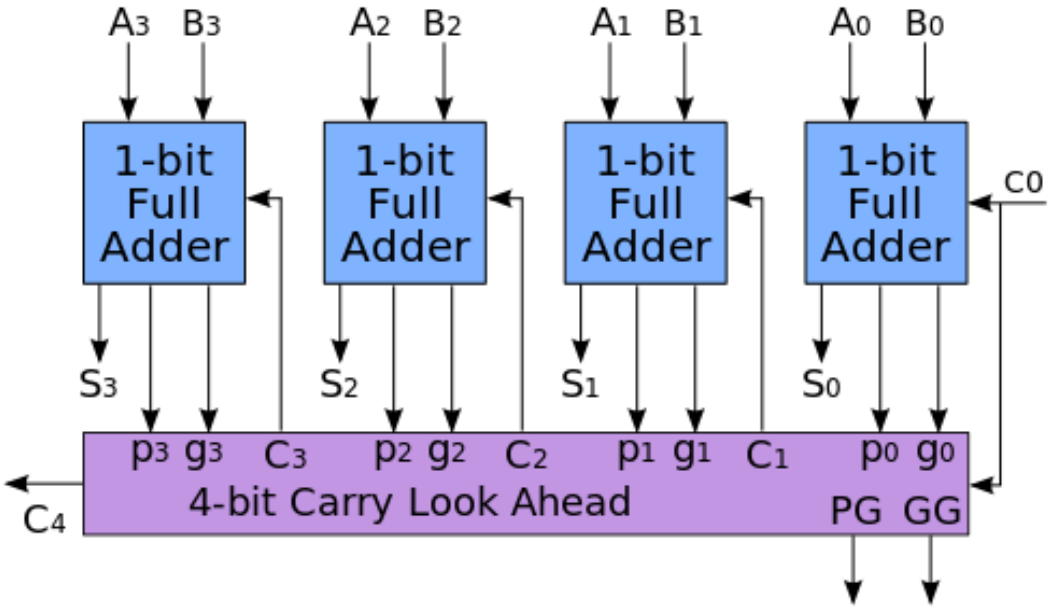

Figure 1: Carry Chain Design

An example of a 4-bit configuration where the carry chain is broken up into slices is shown below.

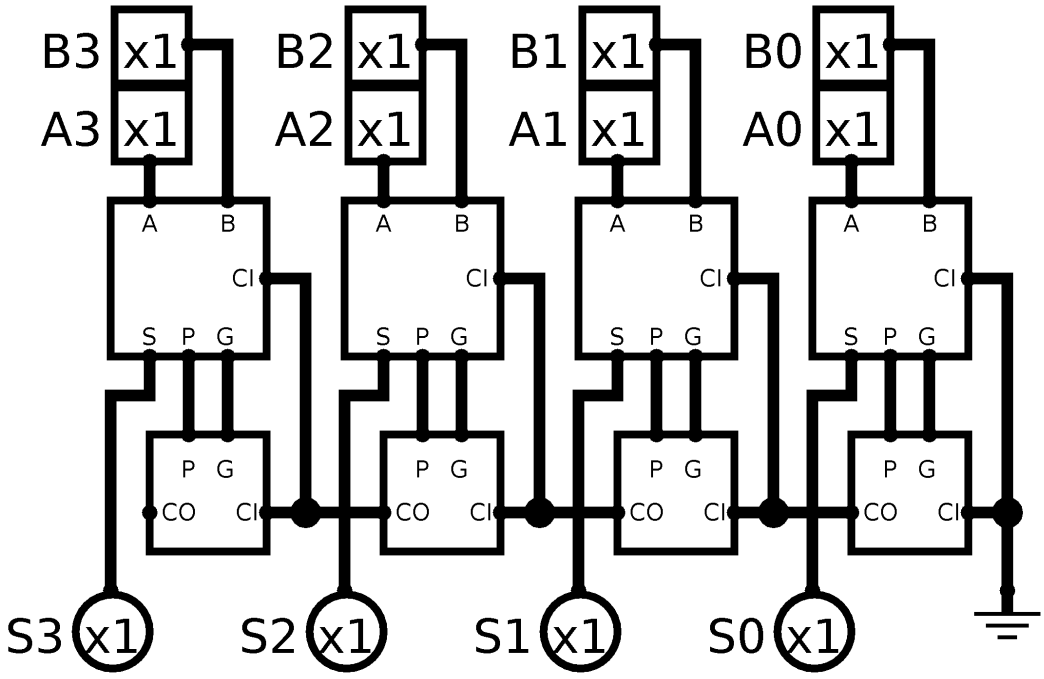

Figure 2: Top Level Design

The logic diagram for a single carry slice using only NAND gates and static inverters is shown below.

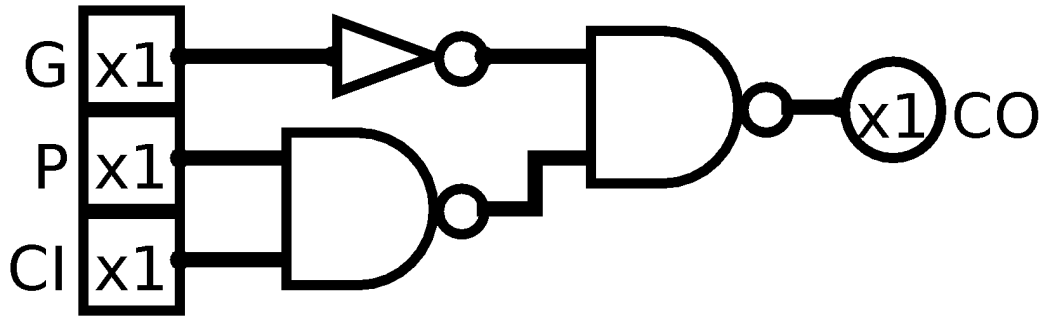

Figure 3: 1-Bit Carry Design

Translating the design into a dynamic NP zipper we achieve the schematic shown below

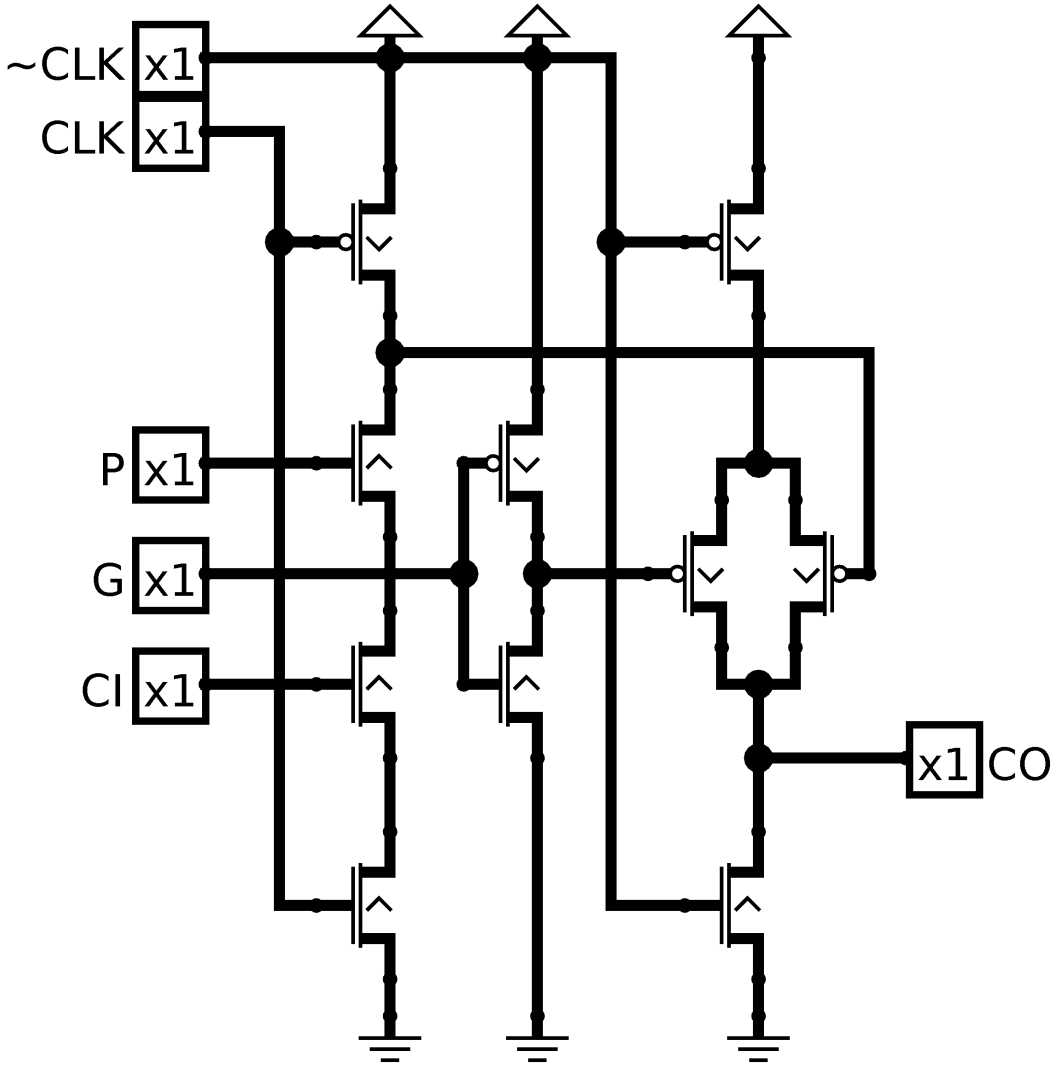

Figure 4: 1-Bit Carry Design Dynamic CMOS

Part 1

A library was created which housed subcircuits for each of the various components used to construct the carry chain. The relevant lines for Part 1 are shown below below.

```
.include ../models/model_t36s.sp

* FET Parameters
.param l  = 0.6u
.param pw = 0.9u
.param nw = 0.9u
.param ld = 2.0u

* P-NAND
.subckt pnand a b f clk vdd gnd
    mp1     vdd     clk     t12     vdd pfet w=pw l=l ad=(ld*pw) as=(ld*pw) pd=(2*ld+2*pw) ps=
    mp2     t12     a       f       vdd pfet w=pw l=l ad=(ld*pw) as=(ld*pw) pd=(2*ld+2*pw) ps=
    mp3     t12     b       f       vdd pfet w=pw l=l ad=(ld*pw) as=(ld*pw) pd=(2*ld+2*pw) ps=
    mn4     f       clk     gnd     gnd nfet w=nw l=l ad=(ld*nw) as=(ld*nw) pd=(2*ld+2*nw) ps=
.ends

* N-NAND
.subckt nnand a b f clk vdd gnd
    mp1     vdd     clk     f       vdd pfet w=pw l=l ad=(ld*pw) as=(ld*pw) pd=(2*ld+2*pw) ps=
    mn2     f       a       t23     gnd nfet w=nw l=l ad=(ld*nw) as=(ld*nw) pd=(2*ld+2*nw) ps=
    mn3     t23     b       t34     gnd nfet w=nw l=l ad=(ld*nw) as=(ld*nw) pd=(2*ld+2*nw) ps=
    mn4     t34     clk     gnd     gnd nfet w=nw l=l ad=(ld*nw) as=(ld*nw) pd=(2*ld+2*nw) ps=
.ends

* Inverter
.subckt inv a f vdd gnd
    mp  vdd     a   f       vdd pfet w=pw l=l ad=(ld*pw) as=(ld*pw) pd=(2*ld+2*pw) ps=(2*ld+2*p
    mn  f       a   gnd     gnd nfet w=nw l=l ad=(ld*nw) as=(ld*nw) pd=(2*ld+2*nw) ps=(2*ld+2*n
.ends
```

Listing 1: Library

Using the library it was trivial to instantiate and test each component of my design.

```
* Max Thrun HW4 Part 1 P-Nand

.include ../models/library.sp

vdd vdd gnd 5V

xpnand a b f clk vdd gnd pnand

.param f = 200meg
*.param f = 530meg

vclk    clk     gnd PULSE(0V 5V '(1/f)/2' 0 0 '(1/f)/2' '1/f')
va      a       gnd PULSE(5V 0V 10n 0 0 10n 20n)
vb      b       gnd PULSE(5V 0V 10n 0 0 10n 20n)

.option post
.tran 0.01n 30n

.end
```

Listing 2: P NAND

```
* Max Thrun HW4 Part 1 N-Nand

.include ../models/library.sp

vdd vdd gnd 5V

xnnand a b f clk vdd gnd nnand

*.param f = 200meg
.param f = 940meg

vclk    clk     gnd PULSE(0V 5V '(1/f)/2' 0 0 '(1/f)/2' '1/f')
va      a       gnd PULSE(0V 5V 10n 0 0 10n 20n)
vb      b       gnd PULSE(0V 5V 10n 0 0 10n 20n)

.option post
.tran 0.01n 30n

.end
```

Listing 3: N NAND

```
* Max Thrun HW4 Part 1 Inverter

.include ../models/library.sp

vdd vdd gnd 5V

xinv a f vdd gnd inv

va    a     gnd PULSE(0V 5V 10n 0 0 10n 20n)

.option post
.tran 0.01n 30n

.end
```

Listing 4: Inverter

The result of the P NAND simulation is shown below. It is clocked at a low clock speed (200MHz) to ensure a full 0-5V swing on the output. From this simulation we can measure the 10-90% rise and fall times and use this to estimate the max clock speed.

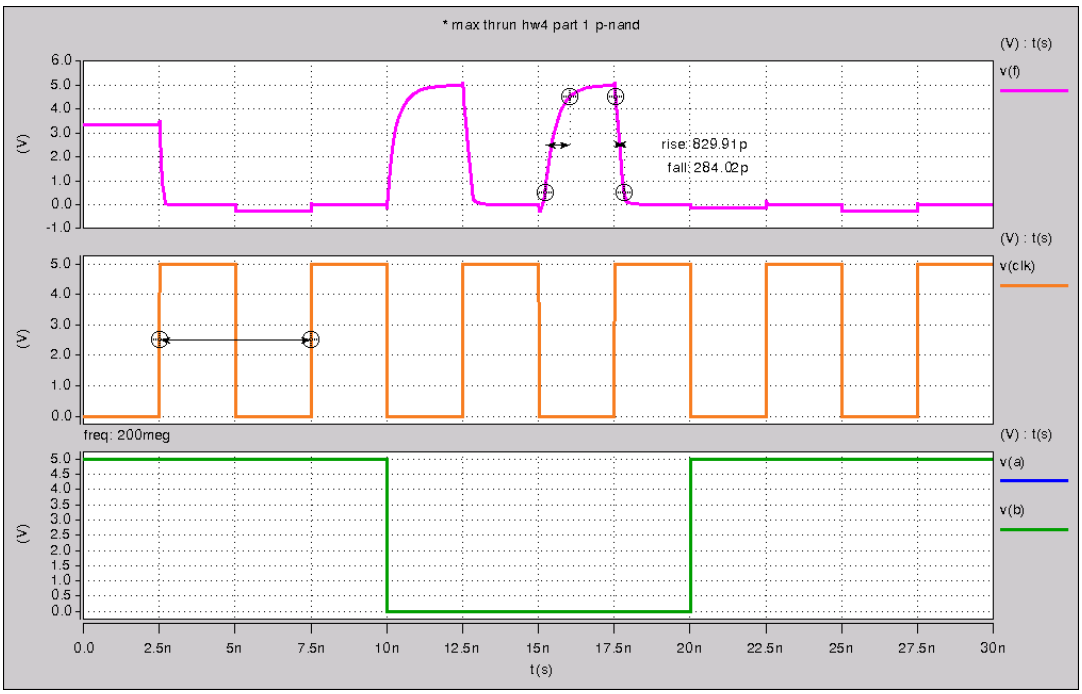

Figure 5: P NAND Simulation Result

The clock speed was then increased until the output dipped below 4.5V (90% of 5V). The max speed of the P NAND was found to be about **530MHz**.

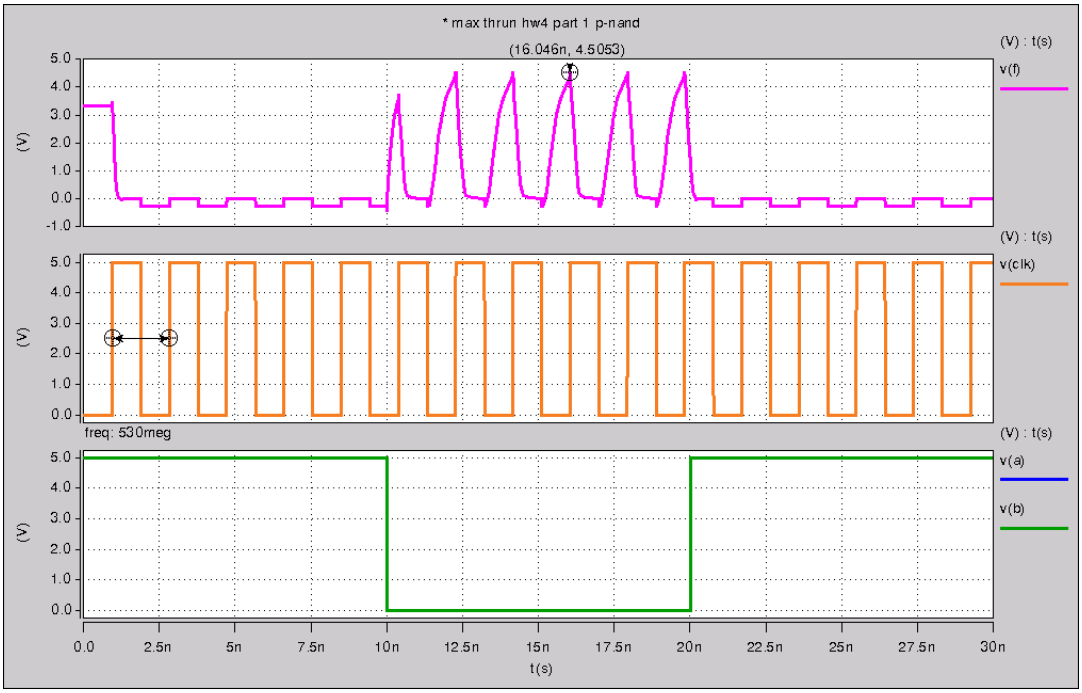

Figure 6: P NAND Simulation Result (Max Clock)

The result of the N NAND simulation is shown below. Again, it was initially clocked at a lower clock speed (200MHz) in order to mesaure the rise and fall of the full 0-5V swing.

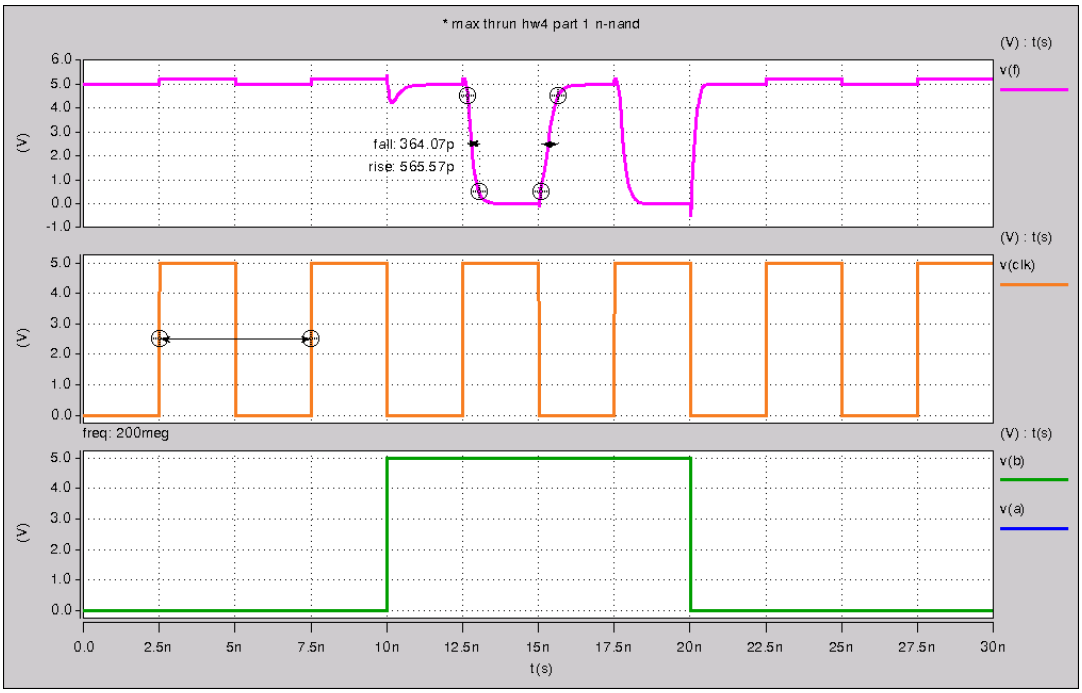

Figure 7: N NAND Simulation Result

The clock speed was then increased until the output dipped below 4.5V (90% of 5V). The max speed of the N NAND was found to be **940MHz**.

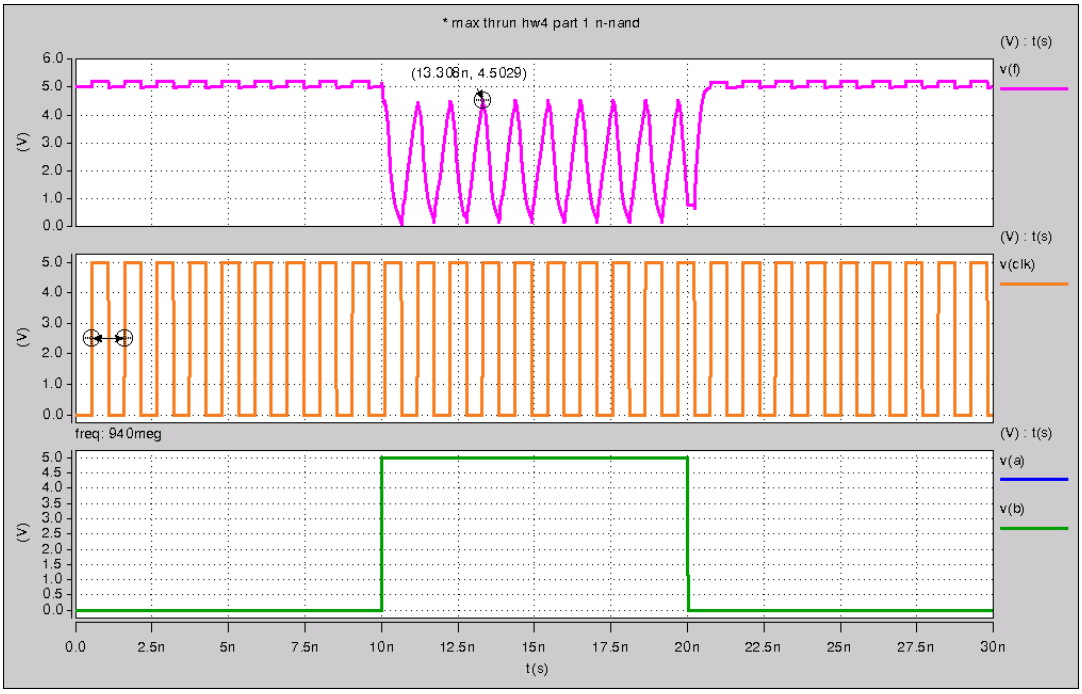

Figure 8: N NAND Simulation Result (Max Clock)

A simulation of the static inverter was also done the results of which are shown below.

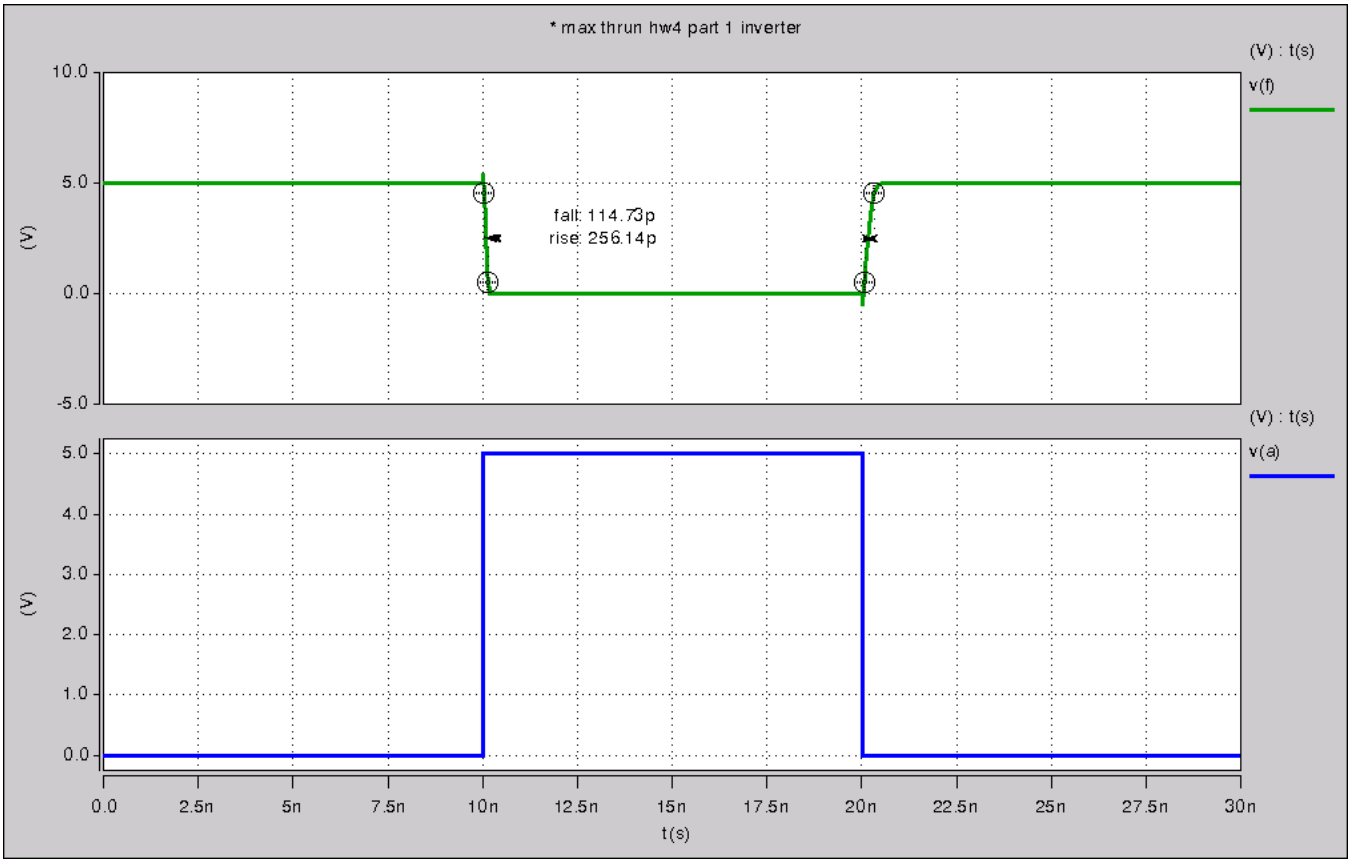

Figure 9: Inverter Simulation Result

Timing Summary

Component	Rise Time	Fall Time	Max Clock Speed
P NAND	829.91p	284.02p	530MHz
N NAND	565.57p	364.07p	940MHz
Inverter	256.14p	114.73p	2.7GHz (calculated)

Table 1: Gate Timing Summary

Part 2

A subcircuit was added to the library which instantiates and connects the components needed to construct a carry slice.

```
1
2  * Carry Slice
3  .subckt carry p g ci co nclk pclk vdd gnd
4      Xnnand p  ci f1 nclk vdd gnd nnand
5      Xinv   g  f2         vdd gnd inv
6      Xpnand f1 f2 co pclk vdd gnd pnand
7  .ends
8
9  .end
```

Listing 5: Library

The carry slice was then instantiated and run through a simulation which exercises the worst case inputs. For our carry slice the worst case condition is when P is held at 5V, G is held at 0V. A change to CI in the condition directly changes CO.

```
1  * Max Thrun HW4 Part 2
2
3  .include ../models/library.sp
4
5  VDD vdd gnd 5V
6
7  Xcarry p g ci co nclk pclk vdd gnd carry
8
9  *.param f = 125meg
10 .param f = 380meg
11
12 vpclk   pclk    gnd PULSE(5V 0V '(1/f)/2' 0 0 '(1/f)/2' '1/f')
13 vnclk   nclk    gnd PULSE(0V 5V '(1/f)/2' 0 0 '(1/f)/2' '1/f')
14 vci     ci      gnd PULSE(0V 5V 10n 0 0 10n 20n)
15 vp      p       gnd PWL(0n 5V)
16 vg      g       gnd PWL(0n 0V)
17
18 .option post
19 .tran 0.01n 30n
20
21 .end
```

Listing 6: Part 2 Spice File

The simulation was first run with a slow clock of 125MHz in order to measure the full rise and fall time.

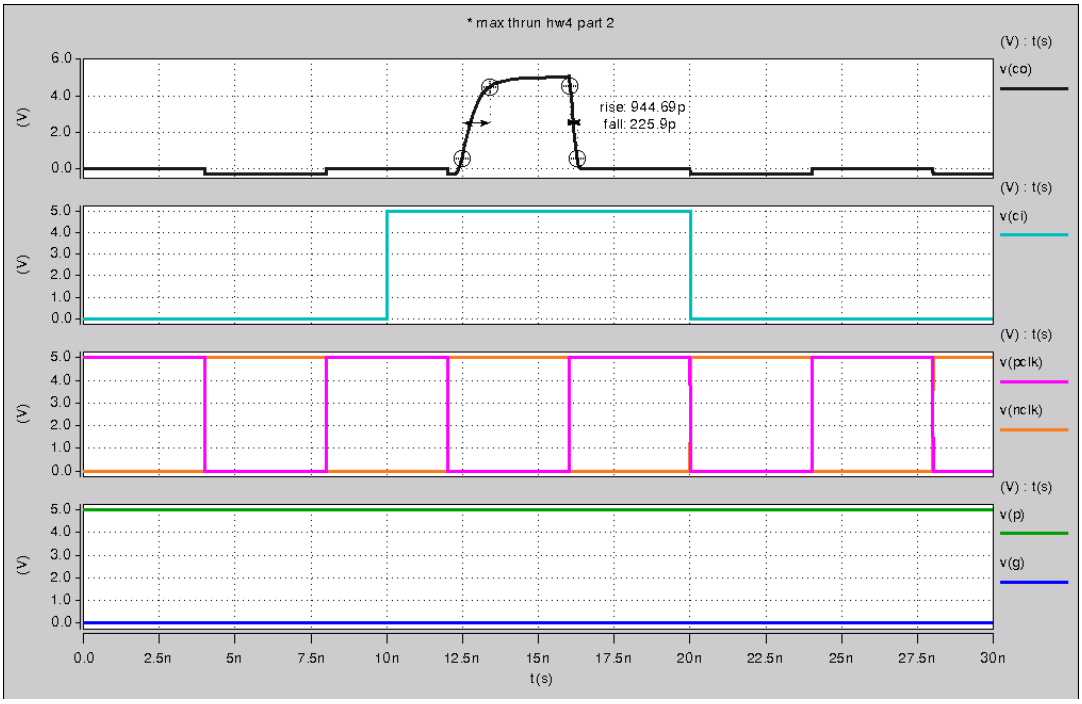

Figure 10: Part 2 Simulation Result

The clock was then increased until the output only reached 90% of 5V (4.5V). The max clock speed was found to be **380MHz**.

Figure 11: Part 2 Simulation Result (Max Clock Speed)

Part 3

Constructing a 4-bit carry chain was achieved by simply instantiating 4 carry slices and chaining the carries together. The same worst case conditions were setup for each slice which allows for a pulse at slice 0 to propagate all the way through the last slice, slice 3.

```
* Max Thrun HW4 Part 3

.include ../models/library.sp

VDD vdd gnd 5V

Xcarry0 p_0 g_0 ci   co_0 nclk pclk vdd gnd carry
Xcarry1 p_1 g_1 co_0 co_1 nclk pclk vdd gnd carry
Xcarry2 p_2 g_2 co_1 co_2 nclk pclk vdd gnd carry
Xcarry3 p_3 g_3 co_2 co_3 nclk pclk vdd gnd carry

.param f = 117meg

vpclk   pclk    gnd PULSE(5V 0V '(1/f)/2' 0 0 '(1/f)/2' '1/f')
vnclk   nclk    gnd PULSE(0V 5V '(1/f)/2' 0 0 '(1/f)/2' '1/f')
vci     ci      gnd PULSE(0V 5V 10n 0 0 10n 20n)

vp_0    p_0     gnd PWL(0n 5V)
vp_1    p_1     gnd PWL(0n 5V)
vp_2    p_2     gnd PWL(0n 5V)
vp_3    p_3     gnd PWL(0n 5V)

vg_0    g_0     gnd PWL(0n 0V)
vg_1    g_1     gnd PWL(0n 0V)
vg_2    g_2     gnd PWL(0n 0V)
vg_3    g_3     gnd PWL(0n 0V)

.option post
.tran 0.01n 30n

.end
```

Listing 7: Part 3 Spice File

The simulation result for the 4-bit configuration is shown below. We can clearly see the input pulse propagate between each slice with each slice adding some delay. The total propagation delay was found to be **3.688ns**. For this simulation we skipped to directly changing the clock speed until the amplitude of the last carry out was 90% of 5V (4.5V). We found the max clock speed to be **117MHz**.

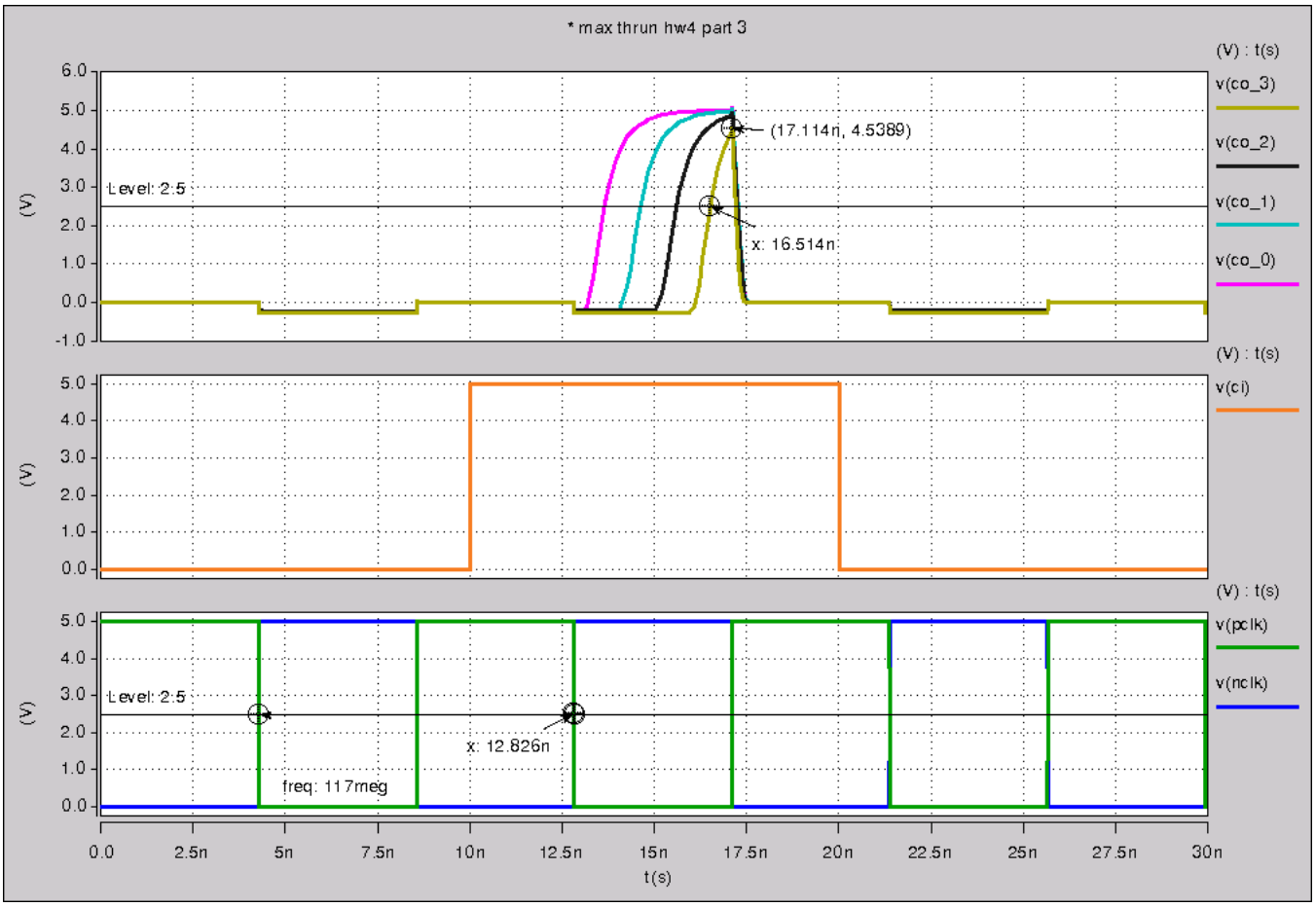

Figure 12: Part 3 Simulation Result

Part 4

In order to find the minimum clock speed I first set CI of the first slice to be constant value of 0V. This should result in the output being flat lined at 0V. I then decreased the clock speed until I found an interesting glitch started to become prominent around **1.75KHz**

```
* Max Thrun HW4 Part 4

.include ../models/library.sp

VDD vdd gnd 5V

Xcarry0 p_0 g_0 ci   co_0 nclk pclk vdd gnd carry
Xcarry1 p_1 g_1 co_0 co_1 nclk pclk vdd gnd carry
Xcarry2 p_2 g_2 co_1 co_2 nclk pclk vdd gnd carry
Xcarry3 p_3 g_3 co_2 co_3 nclk pclk vdd gnd carry

.param f = 1.75K
*.param f = 500

vpclk   pclk    gnd PULSE(5V 0V '(1/f)/2' 0 0 '(1/f)/2' '1/f')
vnclk   nclk    gnd PULSE(0V 5V '(1/f)/2' 0 0 '(1/f)/2' '1/f')
vci     ci      gnd PWL(0n 0V)

vp_0    p_0     gnd PWL(0n 5V)
vp_1    p_1     gnd PWL(0n 5V)
vp_2    p_2     gnd PWL(0n 5V)
vp_3    p_3     gnd PWL(0n 5V)

vg_0    g_0     gnd PWL(0n 0V)
vg_1    g_1     gnd PWL(0n 0V)
vg_2    g_2     gnd PWL(0n 0V)
vg_3    g_3     gnd PWL(0n 0V)

.option post
.tran 0.01n '1.5*(1/f)'

.end
```

Listing 8: Part 4 Spice File

The figure below shows the glitch at 1.7KHz starting to cross 0.5V.

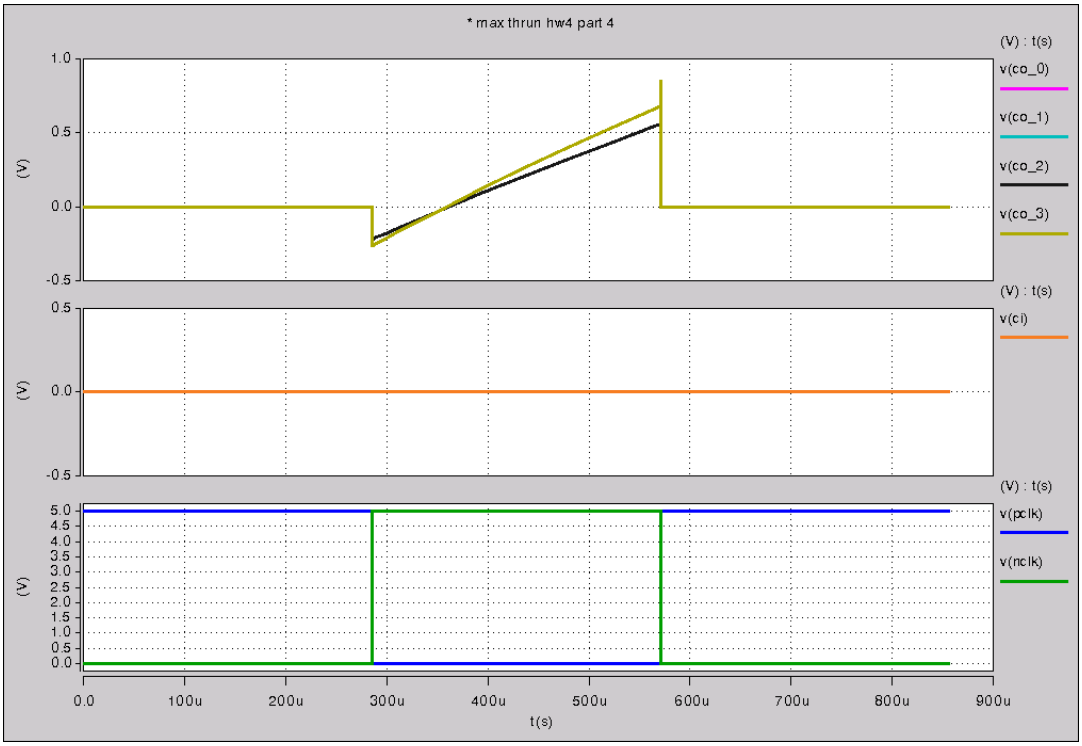

Figure 13: Part 4 Simulation Result

Decreasing the frequency even further we find that the glitch seems to have two linear regions and eventually climbs all the way to 5V.

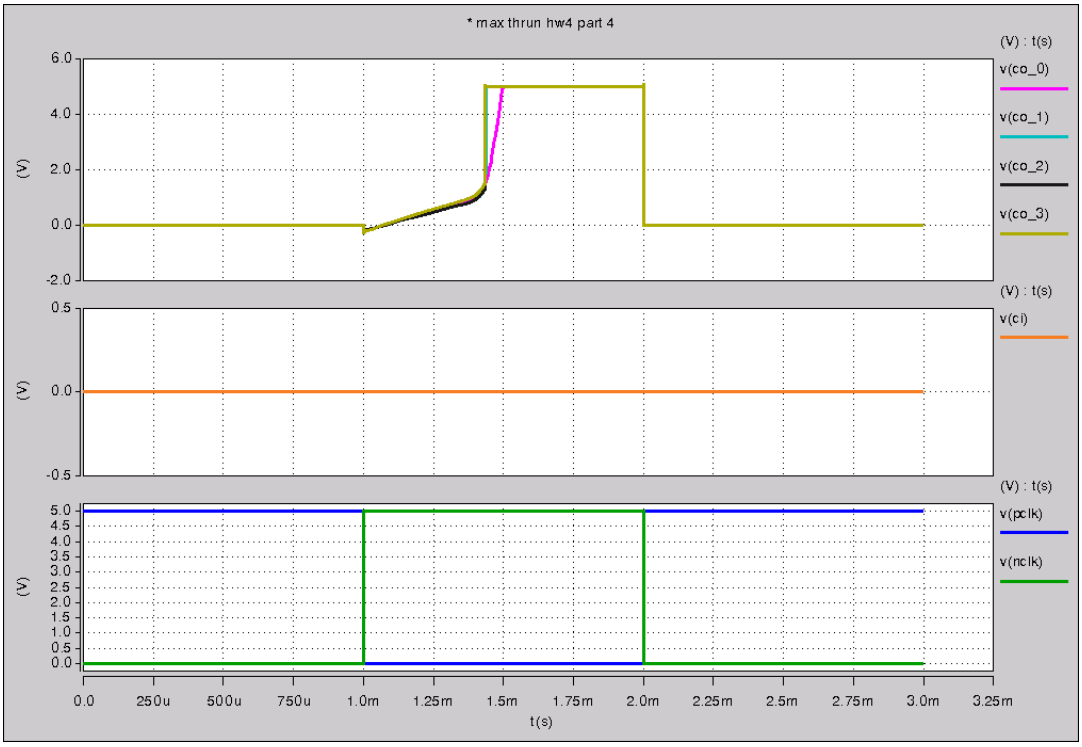

Figure 14: Part 4 Simulation Result

Final Timing Summary

Component	Max Clock Speed
P NAND	530MHz
N NAND	940MHz
1-Bit Carry	380MHz
4-Bit Carry	117MHz

Table 2: Final Timing Summary

What was learned

This homework provided a great example of designing with dynamic logic. I found it to be a lot trickier than designing with static logic as you have to ensure that your clock is running within the small 'working' window. I also found the results of part 4 really interesting. I did not not initially expect anything like what I observed. I am still not exactly sure how the carry outputs are climbing like they are so more time is needed to investigate the root cause. Overall this homework allowed me to explore a new area of digital design that I had not previously experienced at all.