IECE 520

VLSI – Final Exam

Design of a 4bit Ripple carry adder with Flipflops

Submitted by:
Name: Skanda Krishnan Balasubramanian
Email: sbalasubramanian@albany.edu

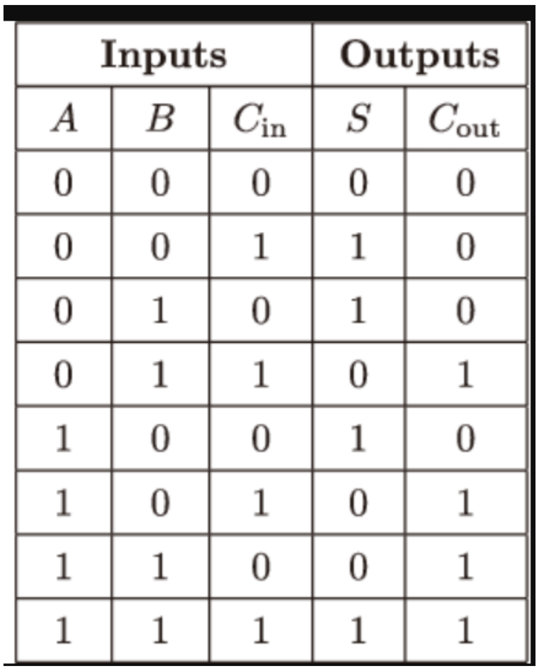
The Sub circuits used for the below designs have been documented in the circuit_documentation pdf uploaded along with this.

Sub circuits used include: invertor, transmission gate, tristate invertor(from invertor and transmission gate), nand gate, xor gate(from invertor and transmission gate), and gate(from nand gate), or gate(from nand gate) and flipflops(from invertor tristate and transmission gate).

All delays and frequencies are calculated at extreme conditions. Questions 1,2,3 and part of 5th have been answered.

1. Draw a 1-bit full adder (FA) circuit using MAGIC, make sure it passes all DRCs, extract the circuit level netlist, and simulate its behaviour using SPICE

The Full adder Truth Table is as Follows:

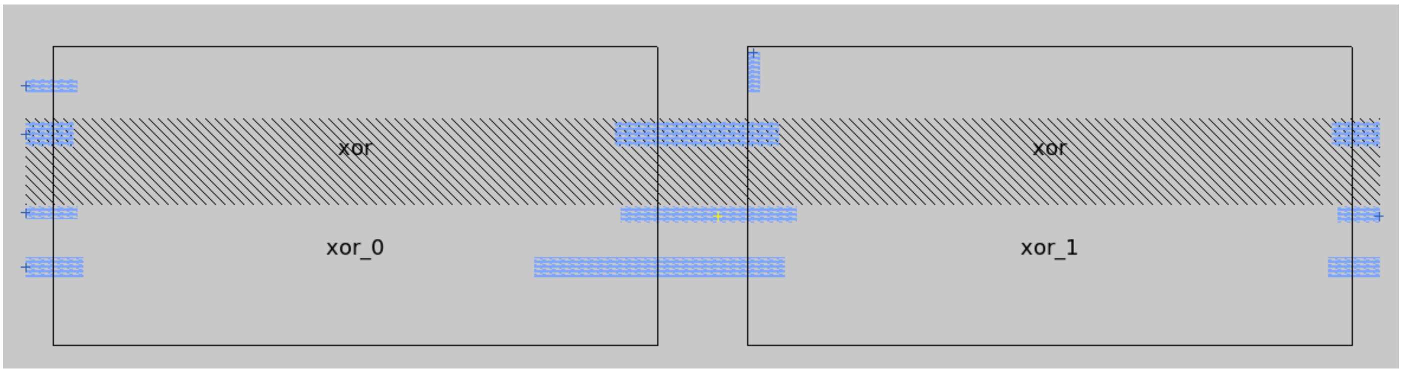


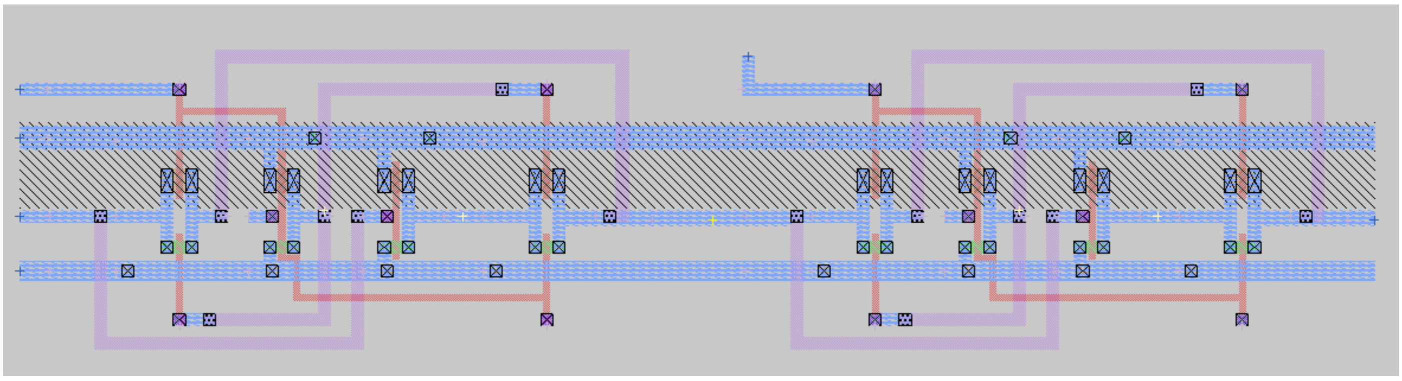
Inputs			Outputs	
A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

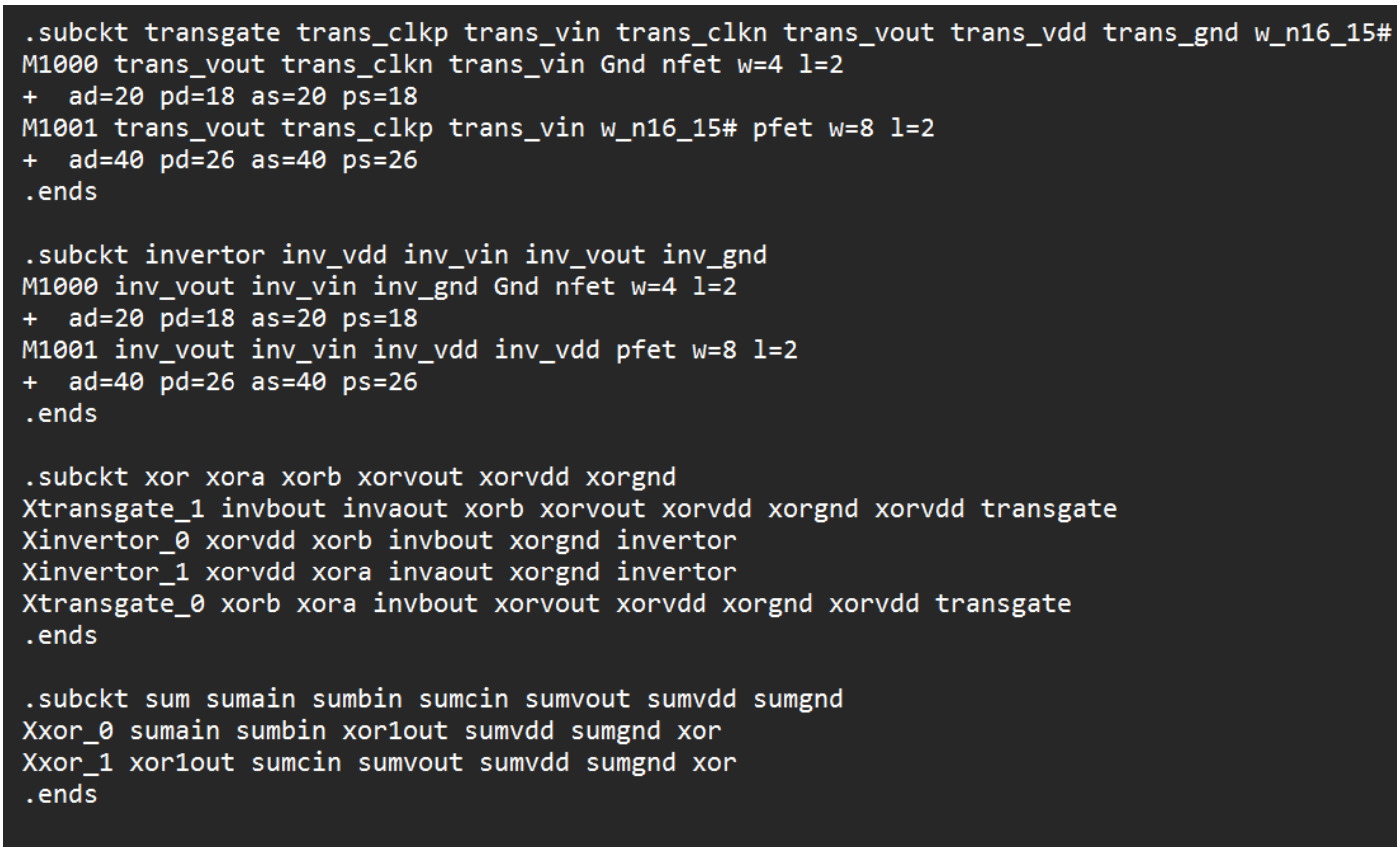
Boolean Expression of S = A (xor) B (xor) Cin

Boolean Expression of Cout = (A (and) B) or (B (and) Cin) or (A (and) Cin)

Sum : A (xor) B (xor) Cin

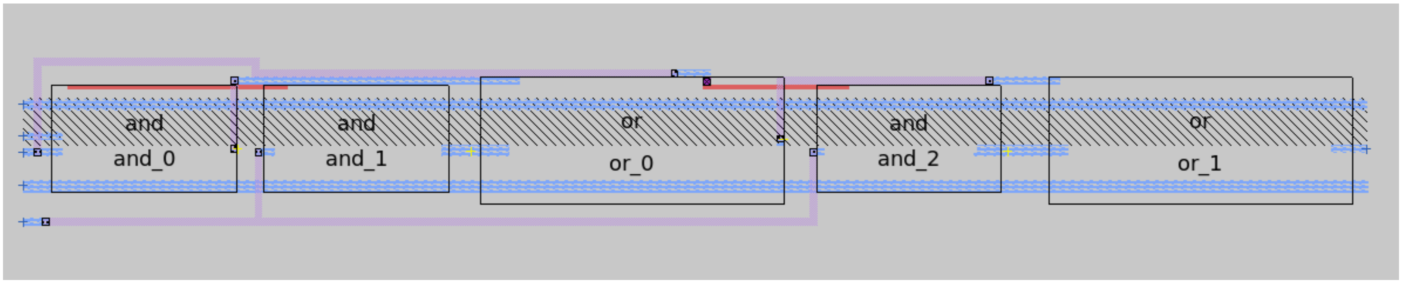


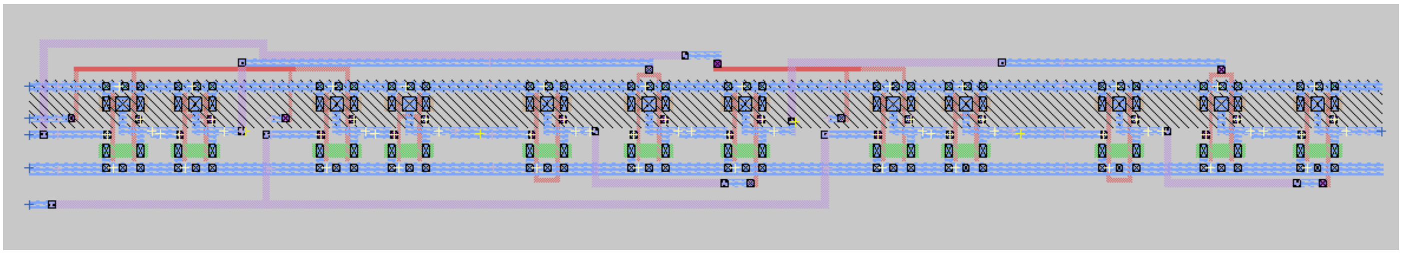




```
.subckt transgate trans_clkp trans_vin trans_clkn trans_vout trans_vdd trans_gnd w_n16_15#
M1000 trans_vout trans_clkn trans_vin Gnd nfet w=4 l=2
+  ad=20 pd=18 as=20 ps=18
M1001 trans_vout trans_clkp trans_vin w_n16_15# pfet w=8 l=2
+  ad=40 pd=26 as=40 ps=26
.ends

.subckt invertor inv_vdd inv_vin inv_vout inv_gnd
M1000 inv_vout inv_vin inv_gnd Gnd nfet w=4 l=2
+  ad=20 pd=18 as=20 ps=18
M1001 inv_vout inv_vin inv_vdd inv_vdd pfet w=8 l=2
+  ad=40 pd=26 as=40 ps=26
.ends

.subckt xor xora xorb xorvout xorvdd xorgnd
Xtransgate_1 invbout invaout xorb xorvout xorvdd xorgnd xorvdd transgate
Xinvertor_0 xorvdd xorb invbout xorgnd invertor
Xinvertor_1 xorvdd xora invaout xorgnd invertor
Xtransgate_0 xorb xora invbout xorvout xorvdd xorgnd xorvdd transgate
.ends

.subckt sum sumain sumbin sumcin sumvout sumvdd sumgnd
Xxor_0 sumain sumbin xor1out sumvdd sumgnd xor
Xxor_1 xor1out sumcin sumvout sumvdd sumgnd xor
.ends
```

Carry: (A and B) or (A and Cin) or (Cin and B) :

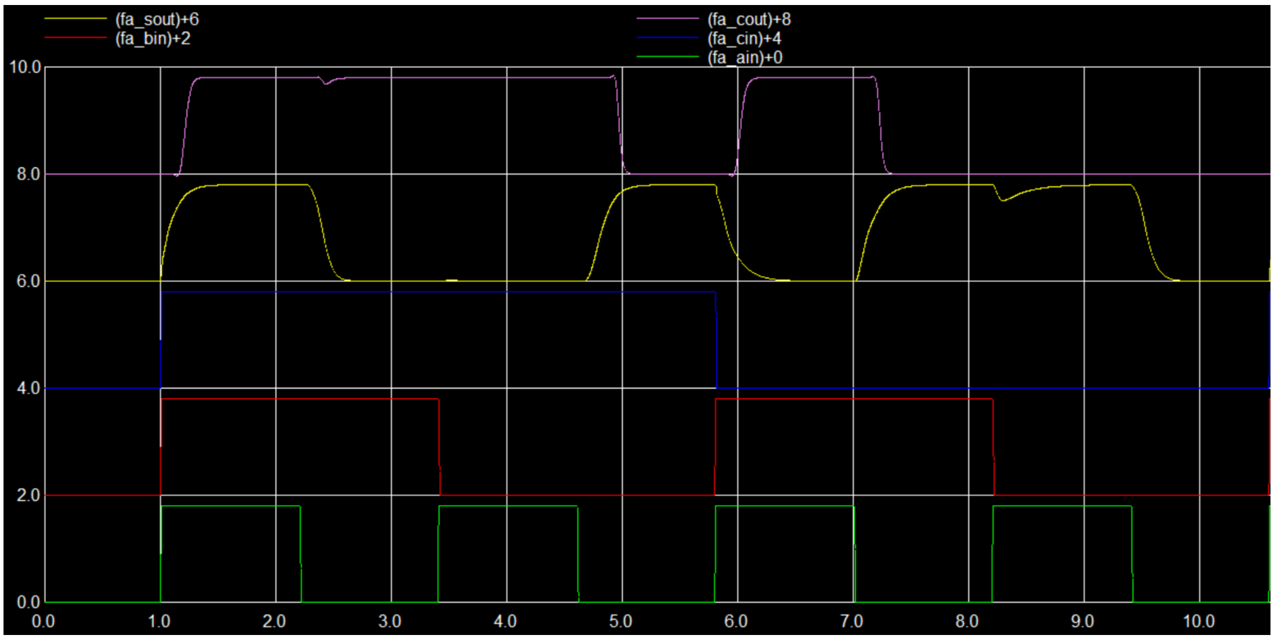




```
.subckt nand_two out inA inB gnd vdd
M1000 vdd inB out vdd pfet w=8 l=2
+  ad=112 pd=60 as=80 ps=36
M1001 out inB a_n10_n7# Gnd nfet w=8 l=2
+  ad=56 pd=30 as=80 ps=36
M1002 out inA vdd vdd pfet w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1003 a_n10_n7# inA gnd Gnd nfet w=8 l=2
+  ad=0 pd=0 as=56 ps=30

.ends

.subckt or orain orbin orout orvdd orgnd
Xnand_two_0 nand0out orain orain orgnd orvdd nand_two
Xnand_two_2 orout nand1out nand0out orgnd orvdd nand_two
Xnand_two_1 nand1out orbin orbin orgnd orvdd nand_two

.ends

.subckt and anda andb andvout andvdd andgnd
Xnand_two_0 nandout anda andb andgnd andvdd nand_two
Xnand_two_1 andvout nandout nandout andgnd andvdd nand_two

.ends

.subckt carry carryain carrybin carrycin carryout carryvdd carrygnd
Xor_0 and1out and0out or0out carryvdd carrygnd or
Xor_1 and2out or0out carryout carryvdd carrygnd or
Xand_0 carryain carrycin and0out carryvdd carrygnd and
Xand_1 carrybin carrycin and1out carryvdd carrygnd and
Xand_2 carrybin carryain and2out carryvdd carrygnd and

.ends
```

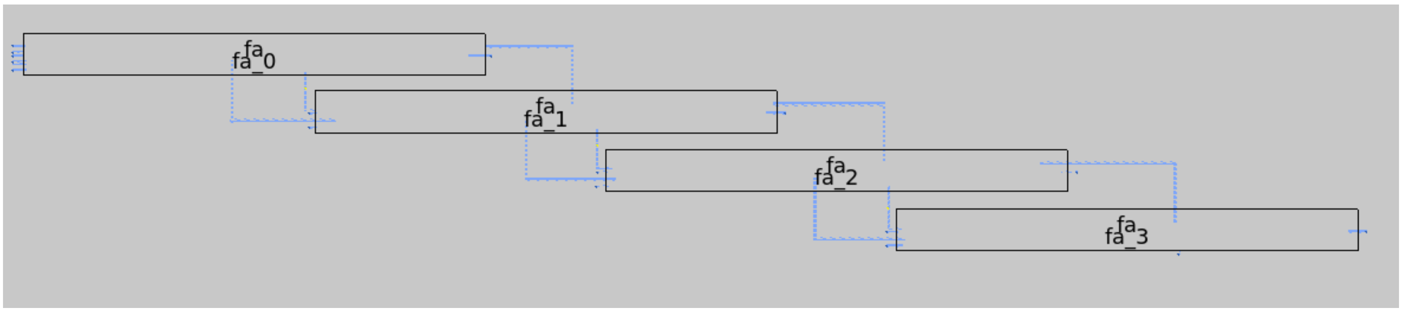
The below Graph goes through the whole truth table values for Carryout and Sum

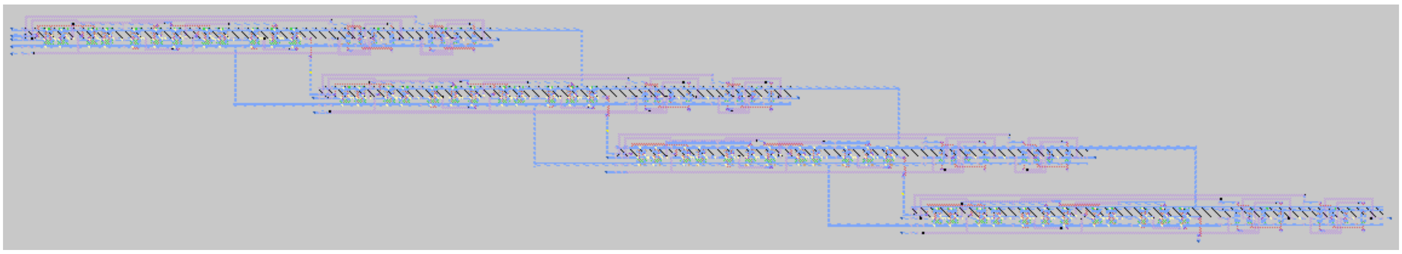
```
Vdd fa_vdd 0 dc 1.8
VG1 fa_ain 0 pulse(0 1.8 1000p 10p 10p 1200p 2400p)
VG2 fa_bin 0 pulse(0 1.8 1000p 10p 10p 2400p 4800p)
VG3 fa_cin 0 pulse(0 1.8 1000p 10p 10p 4800p 9600p)
```

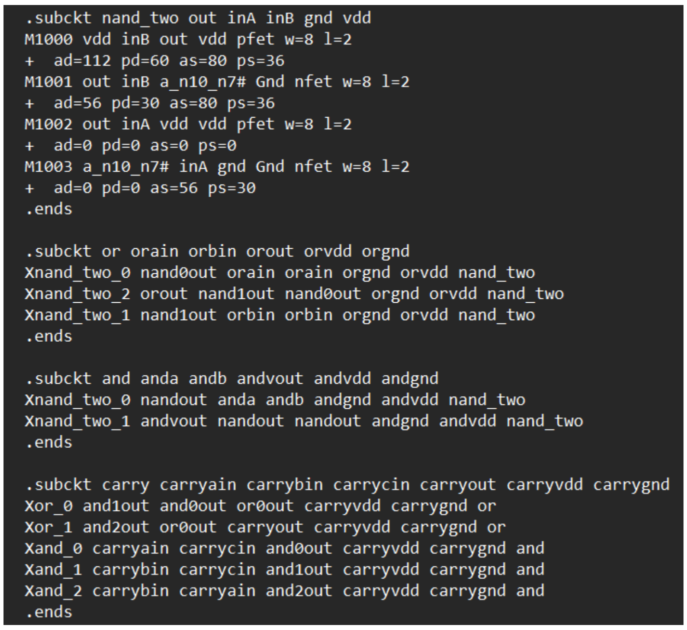

(fa_sout)+6
(fa_bin)+2
(fa_cout)+8
(fa_cin)+4
(fa_ain)+0
10.0
8.0
6.0
4.0
2.0
0.0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0

2. Use hierarchical design flow to instantiate four units of FA to create a 4-bit ripple carry adder (RCA), make sure it passes all DRCs, extract the circuit level netlist, and simulate its behaviour using SPICE with (a) 4+5 and (b) 4-5 examples using two's complement representation. (20 points)

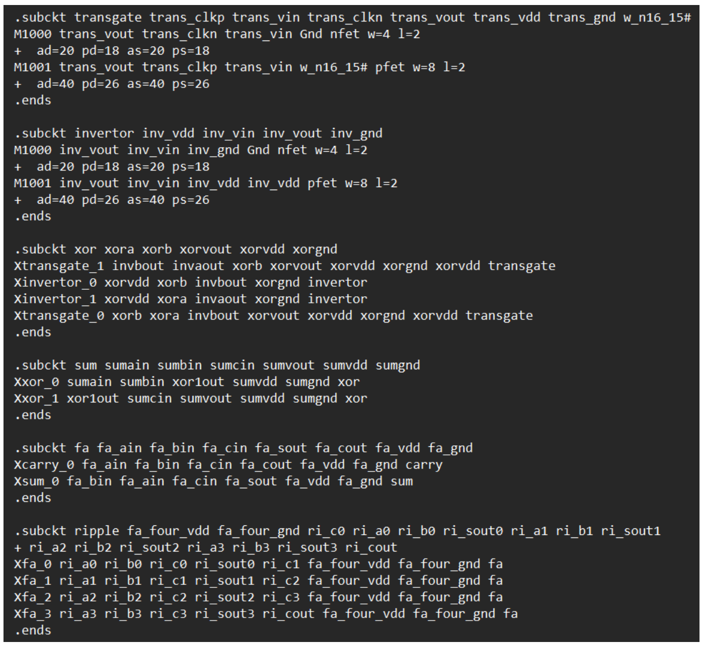
Using the Above Full adder, we can create a Ripple carry adder like below:







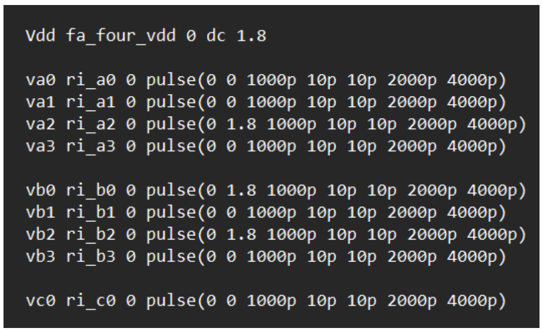
```
.subckt nand_two out inA inB gnd vdd
M1000 vdd inB out vdd pfet w=8 l=2
+  ad=112 pd=60 as=80 ps=36
M1001 out inB a_n10_n7# Gnd nfet w=8 l=2
+  ad=56 pd=30 as=80 ps=36
M1002 out inA vdd vdd pfet w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1003 a_n10_n7# inA gnd Gnd nfet w=8 l=2
+  ad=0 pd=0 as=56 ps=30
.ends

.subckt or orain orbin orout orvdd orgnd
Xnand_two_0 nand0out orain orain orgnd orvdd nand_two
Xnand_two_2 orout nand1out nand0out orgnd orvdd nand_two
Xnand_two_1 nand1out orbin orbin orgnd orvdd nand_two
.ends

.subckt and anda andb andvout andvdd andgnd
Xnand_two_0 nandout anda andb andgnd andvdd nand_two
Xnand_two_1 andvout nandout nandout andgnd andvdd nand_two
.ends

.subckt carry carryain carrybin carrycin carryout carryvdd carrygnd
Xor_0 and1out and0out or0out carryvdd carrygnd or
Xor_1 and2out or0out carryout carryvdd carrygnd or
Xand_0 carryain carrycin and0out carryvdd carrygnd and
Xand_1 carrybin carrycin and1out carryvdd carrygnd and
Xand_2 carrybin carryain and2out carryvdd carrygnd and
.ends
```



```
.subckt transgate trans_clkp trans_vin trans_clkn trans_vout trans_vdd trans_gnd w_n16_15#
M1000 trans_vout trans_clkn trans_vin Gnd nfet w=4 l=2
+  ad=20 pd=18 as=20 ps=18
M1001 trans_vout trans_clkp trans_vin w_n16_15# pfet w=8 l=2
+  ad=40 pd=26 as=40 ps=26
.ends

.subckt invertor inv_vdd inv_vin inv_vout inv_gnd
M1000 inv_vout inv_vin inv_gnd Gnd nfet w=4 l=2
+  ad=20 pd=18 as=20 ps=18
M1001 inv_vout inv_vin inv_vdd inv_vdd pfet w=8 l=2
+  ad=40 pd=26 as=40 ps=26
.ends

.subckt xor xora xorb xorvout xorvdd xorgnd
Xtransgate_1 invbout invaout xorb xorvout xorvdd xorgnd xorvdd transgate
Xinvertor_0 xorvdd xorb invbout xorgnd invertor
Xinvertor_1 xorvdd xora invaout xorgnd invertor
Xtransgate_0 xorb xora invbout xorvout xorvdd xorgnd xorvdd transgate
.ends

.subckt sum sumain sumbin sumcin sumvout sumvdd sumgnd
Xxor_0 sumain sumbin xor1out sumvdd sumgnd xor
Xxor_1 xor1out sumcin sumvout sumvdd sumgnd xor
.ends

.subckt fa fa_ain fa_bin fa_cin fa_sout fa_cout fa_vdd fa_gnd
Xcarry_0 fa_ain fa_bin fa_cin fa_cout fa_vdd fa_gnd carry
Xsum_0 fa_bin fa_ain fa_cin fa_sout fa_vdd fa_gnd sum
.ends

.subckt ripple fa_four_vdd fa_four_gnd ri_c0 ri_a0 ri_b0 ri_sout0 ri_a1 ri_b1 ri_sout1
+ ri_a2 ri_b2 ri_sout2 ri_a3 ri_b3 ri_sout3 ri_cout
Xfa_0 ri_a0 ri_b0 ri_c0 ri_sout0 ri_c1 fa_four_vdd fa_four_gnd fa
Xfa_1 ri_a1 ri_b1 ri_c1 ri_sout1 ri_c2 fa_four_vdd fa_four_gnd fa
Xfa_2 ri_a2 ri_b2 ri_c2 ri_sout2 ri_c3 fa_four_vdd fa_four_gnd fa
Xfa_3 ri_a3 ri_b3 ri_c3 ri_sout3 ri_cout fa_four_vdd fa_four_gnd fa
.ends
```

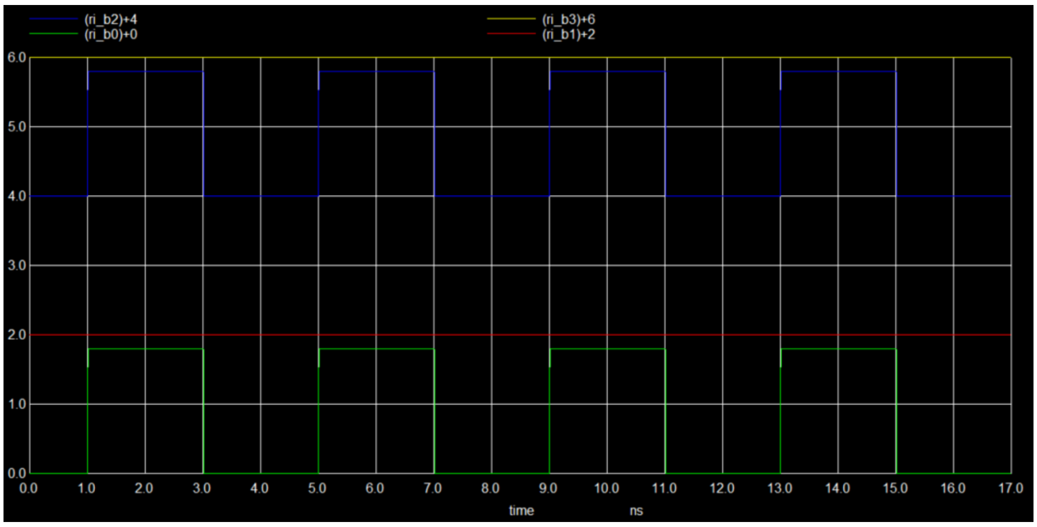
a) 4+5 addition through the 4 bit ripple carry adder

Voltage pulses configured as below. A represented as ri_A3, ri_A2, ri_A1, ri_A0 , B represented as ri_B3, ri_B2, ri_B1, ri_B0 and Cin as ri_C0 with ri_A0 and ri_B0 as the least significant bits

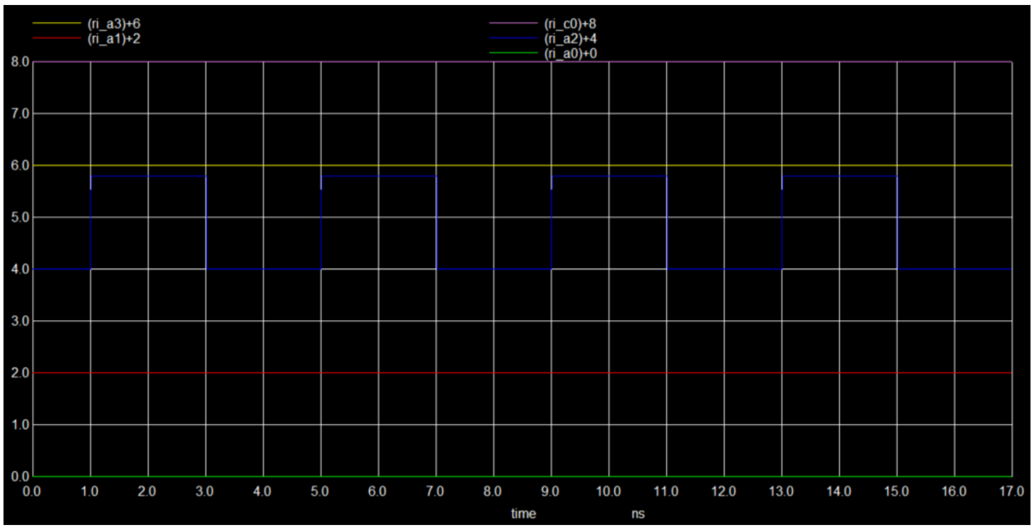


```
Vdd fa_four_vdd 0 dc 1.8

va0 ri_a0 0 pulse(0 0 1000p 10p 10p 2000p 4000p)
va1 ri_a1 0 pulse(0 0 1000p 10p 10p 2000p 4000p)
va2 ri_a2 0 pulse(0 1.8 1000p 10p 10p 2000p 4000p)
va3 ri_a3 0 pulse(0 0 1000p 10p 10p 2000p 4000p)

vb0 ri_b0 0 pulse(0 1.8 1000p 10p 10p 2000p 4000p)
vb1 ri_b1 0 pulse(0 0 1000p 10p 10p 2000p 4000p)
vb2 ri_b2 0 pulse(0 1.8 1000p 10p 10p 2000p 4000p)
vb3 ri_b3 0 pulse(0 0 1000p 10p 10p 2000p 4000p)

vc0 ri_c0 0 pulse(0 0 1000p 10p 10p 2000p 4000p)
```

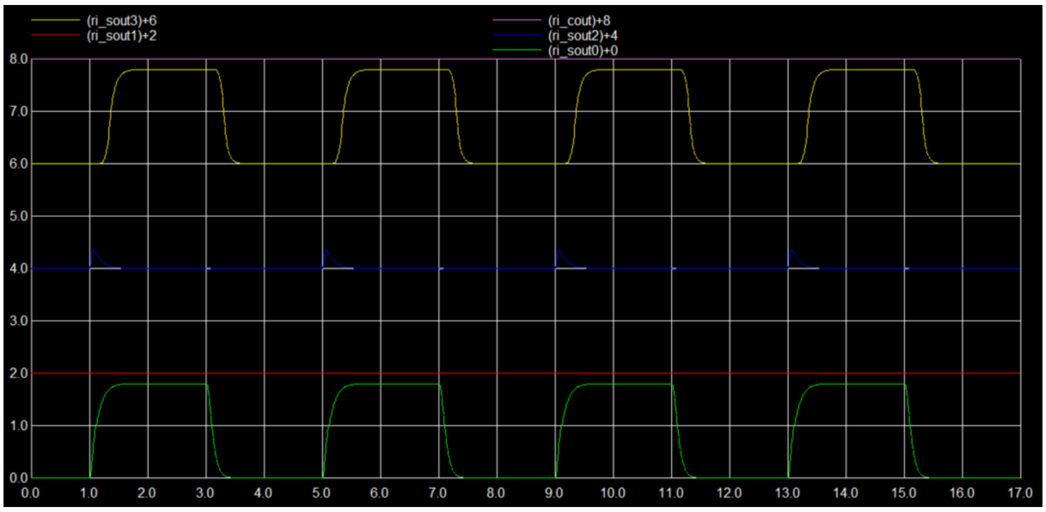
B = 0101



A = 0100, Cin = 0



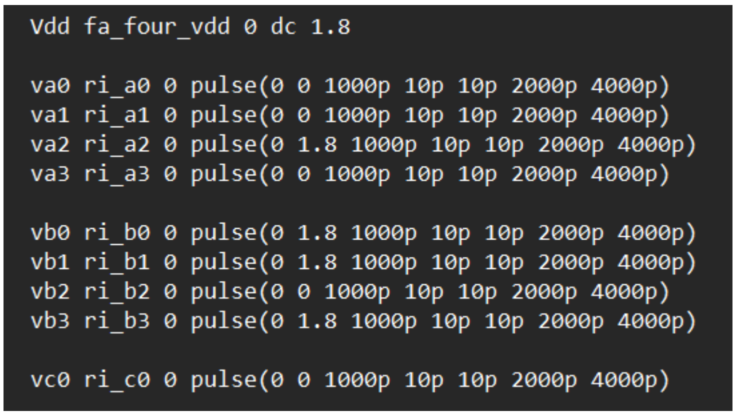
S = 1001, Cout = 0



The ripple carry adder yields the answer 9 represented in binary.

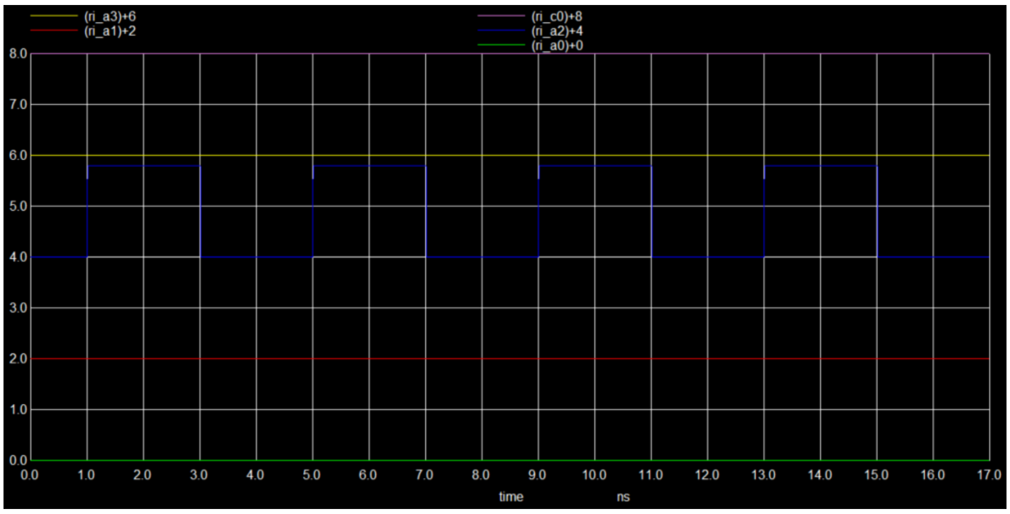
b) 4-5 Subtraction requires 2’s compliment conversion of 5 to represent it as a negative number.

A = 4 = 0100
B = -5 = 2’s compliment (0101) = 1010+1 = 1011
Cin = 0
Voltage pulses configured as below. A represented as ri_A3, ri_A2, ri_A1, ri_A0 , B represented as ri_B3, ri_B2, ri_B1, ri_B0 and Cin as ri_C0 with ri_A0 and ri_B0 as the least significant bits

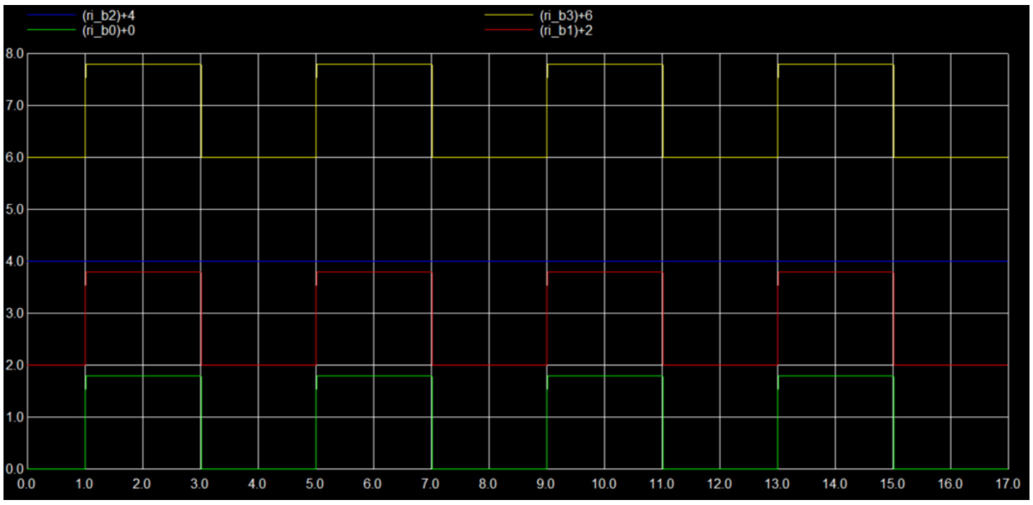


```
Vdd fa_four_vdd 0 dc 1.8

va0 ri_a0 0 pulse(0 0 1000p 10p 10p 2000p 4000p)
va1 ri_a1 0 pulse(0 0 1000p 10p 10p 2000p 4000p)
va2 ri_a2 0 pulse(0 1.8 1000p 10p 10p 2000p 4000p)
va3 ri_a3 0 pulse(0 0 1000p 10p 10p 2000p 4000p)

vb0 ri_b0 0 pulse(0 1.8 1000p 10p 10p 2000p 4000p)
vb1 ri_b1 0 pulse(0 1.8 1000p 10p 10p 2000p 4000p)
vb2 ri_b2 0 pulse(0 0 1000p 10p 10p 2000p 4000p)
vb3 ri_b3 0 pulse(0 1.8 1000p 10p 10p 2000p 4000p)

vc0 ri_c0 0 pulse(0 0 1000p 10p 10p 2000p 4000p)
```

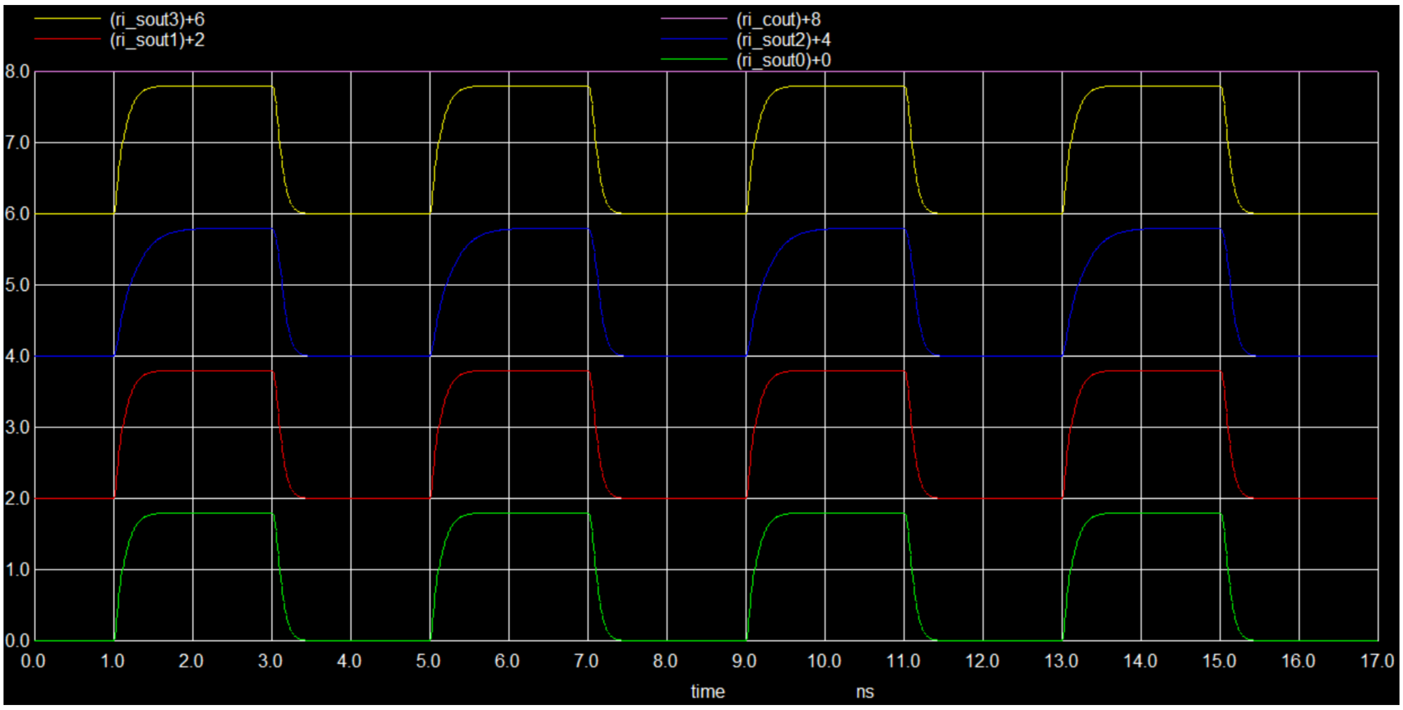
A = 0100, Cin = 0



B = 1011



S = 1111, Cout =0

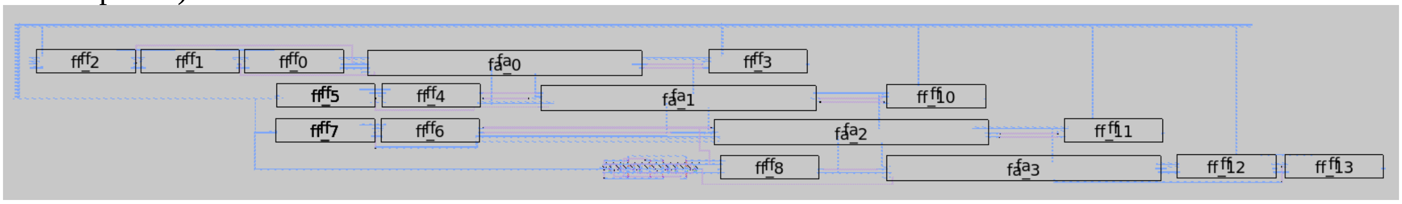


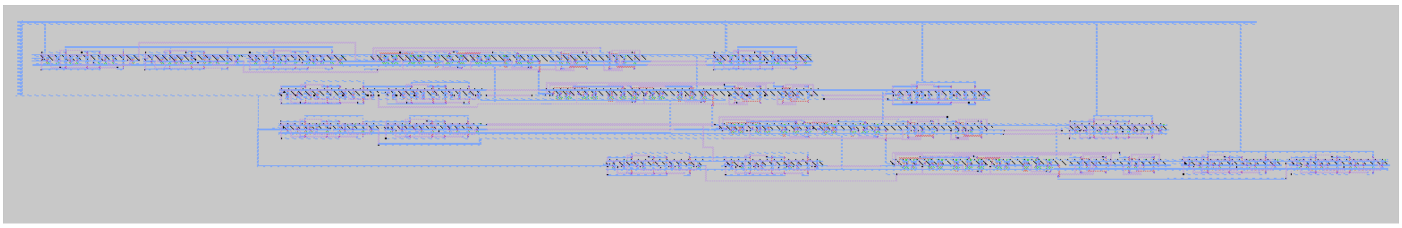
Cout is zero, indicating that the S is a 2’s compliment representation of a negative number.

S =1111 = 2’s compliment of(0000+1) = -1

Answer is -1

3. Use hierarchical design flow to add flip-flops to create a clocked version of this 4-bit RCA with inputs A[4:1] and B[4:1] and carry-input C_{in} as shown on the next slide. Make sure it passes all DRCs, extract the circuit level netlist, and find the maximum clock frequency using SPICE. (30 points).





Clk-Q /Setup/ Hold time of the flipflop

ClkQtime: Time required for the output of flipflop to reach 50% of vdd from the time at which clk edge reached 50% vdd.

Setup time: Time required by the input to be stable before the clk edge to ensure that the input is latched properly.

Hold Time : Time required by the input to be stable after the clk edge to ensure that the input is latched properly.

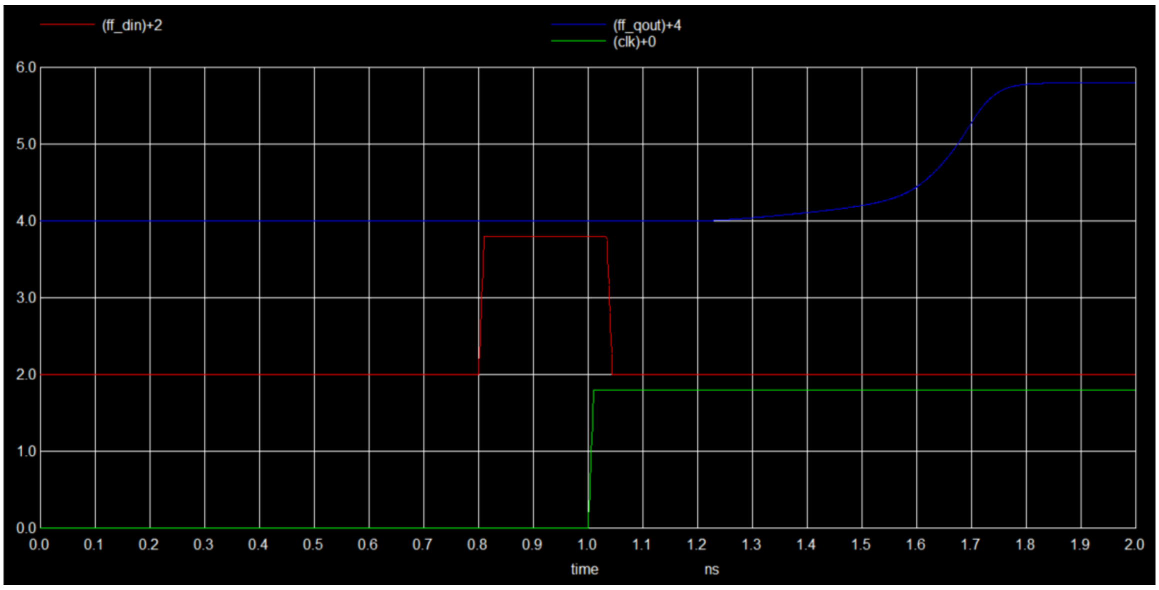
These times are calculated at the worst of timing conditions. The flipflop has been pushed to the max to get these times. This is done so that the specification of the flipflop can be safely assumed and giving any freq/times below the specified values would ensure perfect working of the flipflops.

```
VG1 clk 0 pulse(0 1.8 1000p 10p 10p 2400p 4800p)

VG2 ff_din 0 pulse(0 1.8 800p 10p 10p 224p 12000p)
```

These are the max times achieved for starting and ending the ff_din pulse. If the pulse starts later than this there would be a setup violation. If the pulse is ended before this value then there would be a hold time violation. Since this is at the edge of the setup and hold time, the CQ time would be maximum at this point.

Therefore the CQ time is also calculated at this specific condition.

```
.tran 1p 2000p
.measure tran setup_time trig v(ff_din) val=0.9 rise=1 targ v(clk) val=0.9 rise=1
.measure tran hold_time trig v(clk) val=0.9 rise=1 targ v(ff_din) val=0.9 fall=1
.measure tran cq_time trig v(clk) val=0.9 rise=1 targ v(ff_qout) val=0.9 rise=1
```



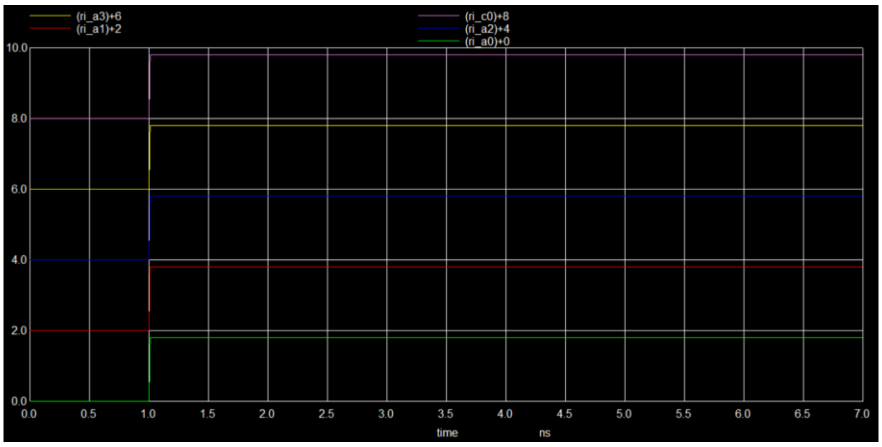
```
setup_time          =  2.000000e-10 targ=  1.005000e-09 trig=  8.050000e-10
hold_time           =  3.400000e-11 targ=  1.039000e-09 trig=  1.005000e-09
cq_time             =  6.589232e-10 targ=  1.663923e-09 trig=  1.005000e-09
```

Setup time of the Flipflop = 200psec
Hold   time of the Flipflop = 34psec
Clk-Q time of the Flipflop = 658psec

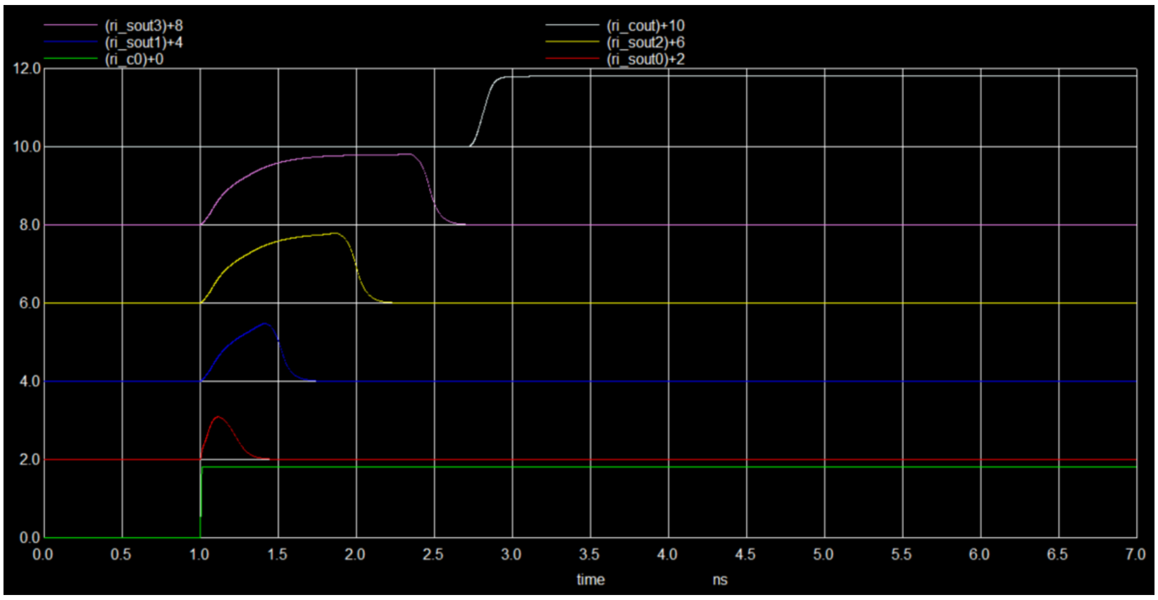
<u>Combinatorial Delay of the 4bit Ripple Carry Adder:</u>

The critical path for this delay is when all the carrys are propagated instead of generated by the FA inputs. Therefore this A needs to be 1111, B needs to be 0000, and Cin = 1. This ensures that the carry is propagated throughout the circuit and produces the maximum delay.

A = 1111,Cin =1

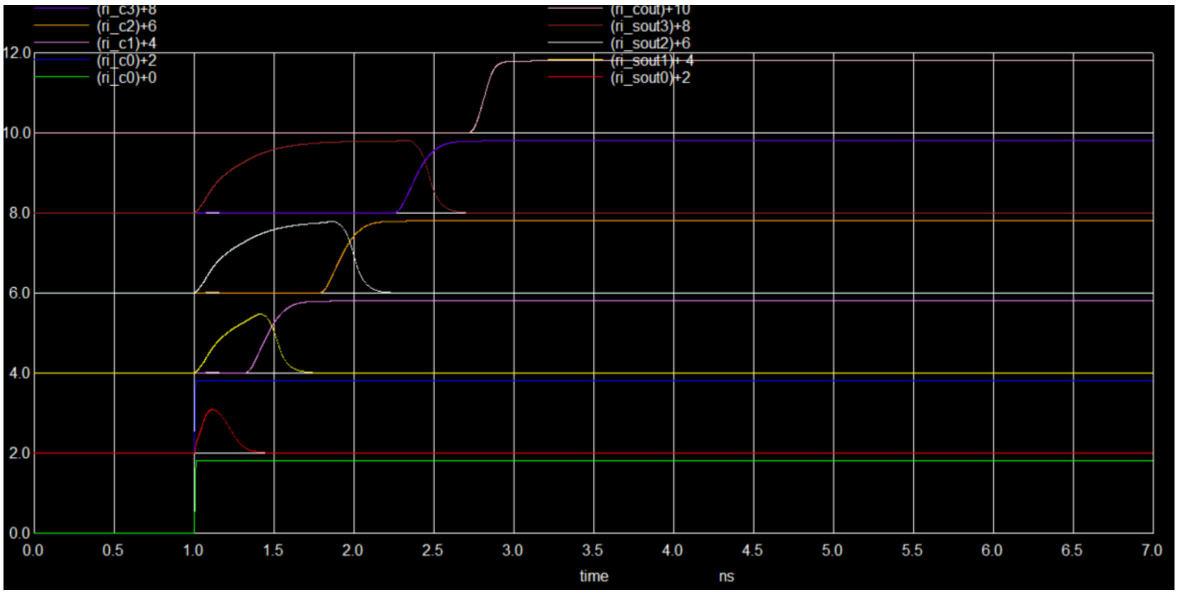


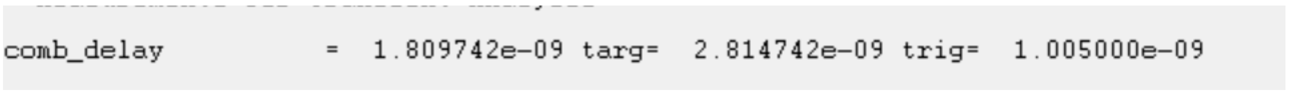
S = 0000, Cout = 1



The SUM bits goes high and comes back down as the carry is propagated through. The propagation of carry through the circuit affecting the sum is shown below:

Sum bits overlayed with its corresponding Carry in bits





```
comb_delay          =  1.809742e-09 targ=  2.814742e-09 trig=  1.005000e-09
```

combinational delay = 1809psec

The max clk frequency that can be allowed when the 4bit RCA is combined wit input and output flipflop is calculated below:

Tclk > Tcomb + Tc-q + Tsetup
Tclk > 1809psec + 658psec + 200psec

Tclk > 2667psec

Theoretical Max Freq of clk = 1/Tclk(min) = 1/2667psec = 374MHz (For working at every condition including extreme cases)

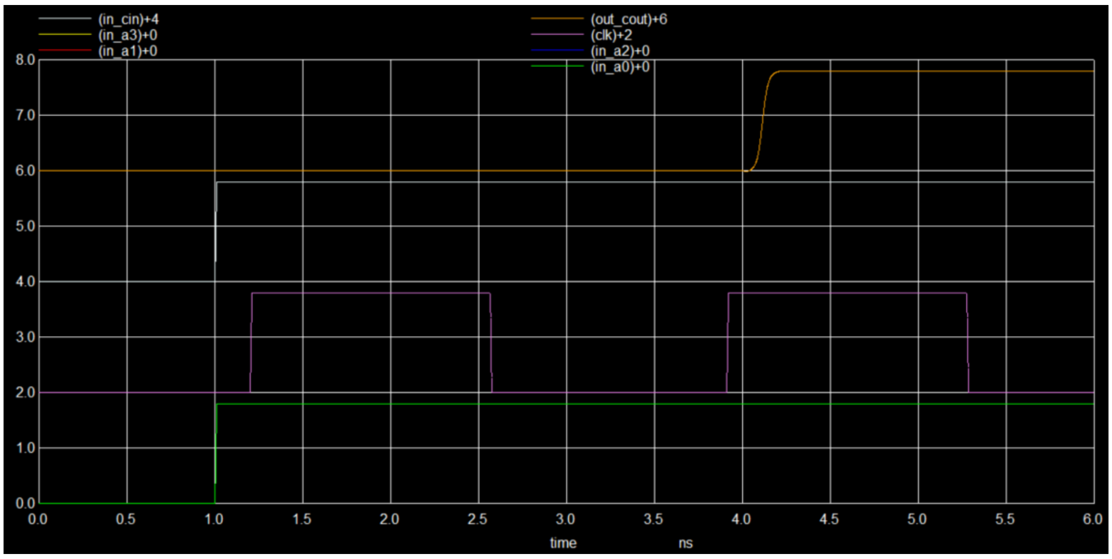
Experimentally trying out different frequencies with more than enough hold time. The setup time is kept at the extreme at 200psec.
The Max freq that was able to run was:

```
vclk clk 0 pulse(0 1.8 1200p 10p 10p 1355p 2710p)
```

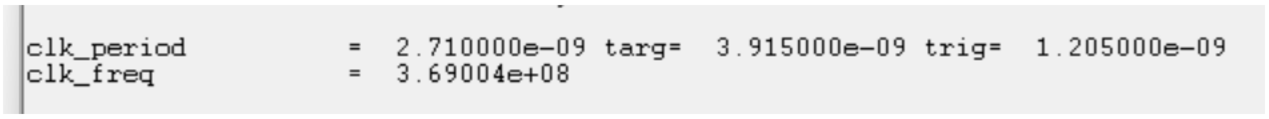
Input given to excite the same critical path of the 4bit RCA:

```
Vdd vdd 0 dc 1.8

va0 in_a0 0 pulse(0 1.8 1000p 10p 10p 10000p 20000p)
va1 in_a1 0 pulse(0 1.8 1000p 10p 10p 10000p 20000p)
va2 in_a2 0 pulse(0 1.8 1000p 10p 10p 10000p 20000p)
va3 in_a3 0 pulse(0 1.8 1000p 10p 10p 10000p 20000p)

vb0 in_b0 0 pulse(0 0 1000p 10p 10p 10000p 20000p)
vb1 in_b1 0 pulse(0 0 1000p 10p 10p 10000p 20000p)
vb2 in_b2 0 pulse(0 0 1000p 10p 10p 10000p 20000p)
vb3 in_b3 0 pulse(0 0 1000p 10p 10p 10000p 20000p)

vc0 in_cin 0 pulse(0 1.8 1000p 10p 10p 10000p 20000p)
```

Output Plot: Cout = 1 at the second clk cycle



Cout coming at the second posedge of the clk

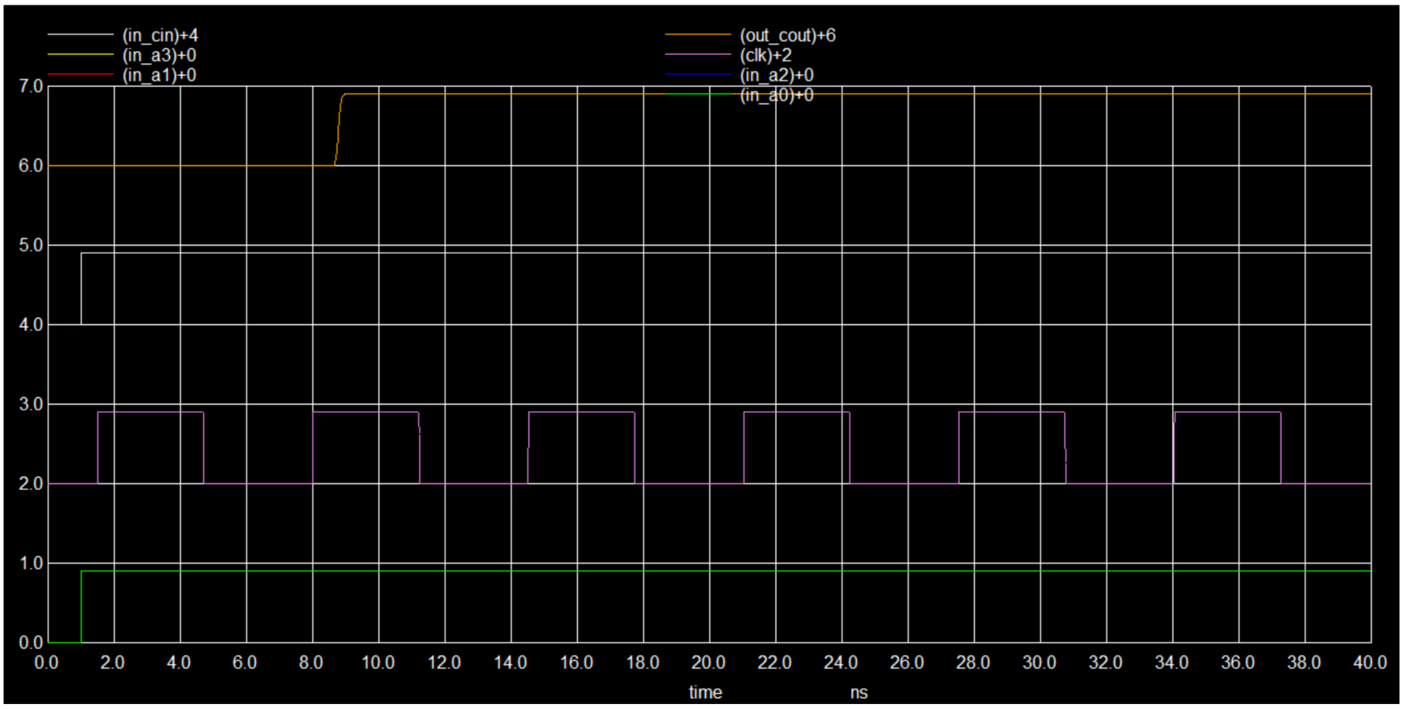


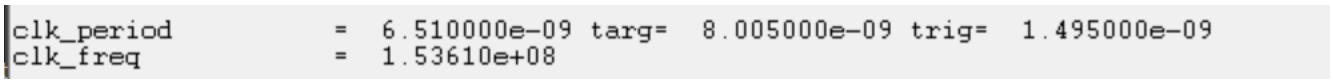
```
clk_period          =  2.710000e-09 targ=  3.915000e-09 trig=  1.205000e-09
clk_freq            =  3.69004e+08
```

Exp clk freq = 369MHz

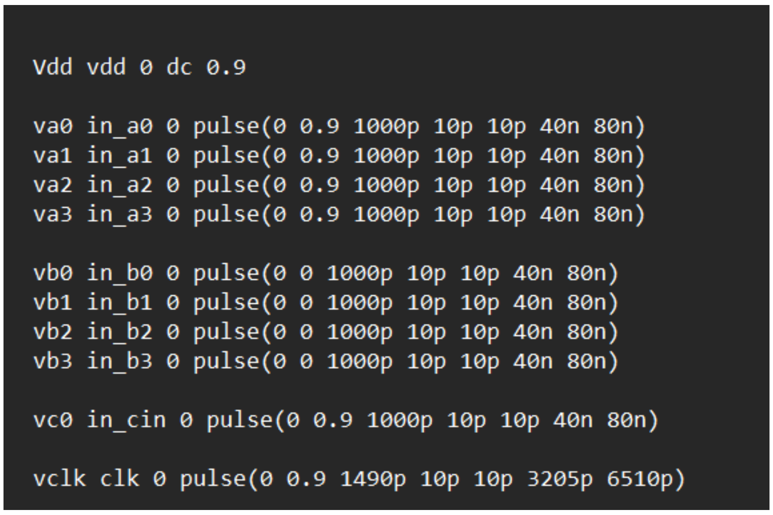
The experimental clk frequency is within 2% of the theoretical frequency.

5.





```
clk_period           =  6.510000e-09 targ=  8.005000e-09 trig=  1.495000e-09
clk_freq             =  1.53610e+08
```



```
Vdd vdd 0 dc 0.9

va0 in_a0 0 pulse(0 0.9 1000p 10p 10p 40n 80n)
va1 in_a1 0 pulse(0 0.9 1000p 10p 10p 40n 80n)
va2 in_a2 0 pulse(0 0.9 1000p 10p 10p 40n 80n)
va3 in_a3 0 pulse(0 0.9 1000p 10p 10p 40n 80n)

vb0 in_b0 0 pulse(0 0 1000p 10p 10p 40n 80n)
vb1 in_b1 0 pulse(0 0 1000p 10p 10p 40n 80n)
vb2 in_b2 0 pulse(0 0 1000p 10p 10p 40n 80n)
vb3 in_b3 0 pulse(0 0 1000p 10p 10p 40n 80n)

vc0 in_cin 0 pulse(0 0.9 1000p 10p 10p 40n 80n)

vclk clk 0 pulse(0 0.9 1490p 10p 10p 3205p 6510p)
```

Min setup = 490psec

Max clk Freq = 153MHz.