VLSI DESIGN

Project Report

Name: Meghana Tedla

Roll No: 2021102002

PRE-LAYOUT SIMULATIONS

GATES

NOT GATE

NETLIST

```
**NOT GATE

.subckt INVERTER nodeIN Vdd Gnd nodeOUT

Mp1 nodeOUT nodeIN Vdd Vdd CMOSP W={2*Width_p} L={Length_p}
Mn1 nodeOUT nodeIN Gnd Gnd CMOSN W={2*Width_n} L={Length_n}


.ends INVERTER
```

2-INPUT NAND GATE

NETLIST

```
**2 Input NAND Gate

.subckt NAND nodeA nodeB Vdd Gnd nodeC

Mp1   nodeC   nodeA    Vdd   Vdd   CMOSP W={2*Width_p} L={Length_p}
Mp2   nodeC   nodeB    Vdd   Vdd   CMOSP W={2*Width_p} L={Length_p}
Mn1   node1   nodeA    Gnd   Gnd   CMOSN W={2*Width_n} L={Length_n}
Mn2   nodeC   nodeB   node1  Gnd   CMOSN W={2*Width_n} L={Length_n}

.ends NAND
```

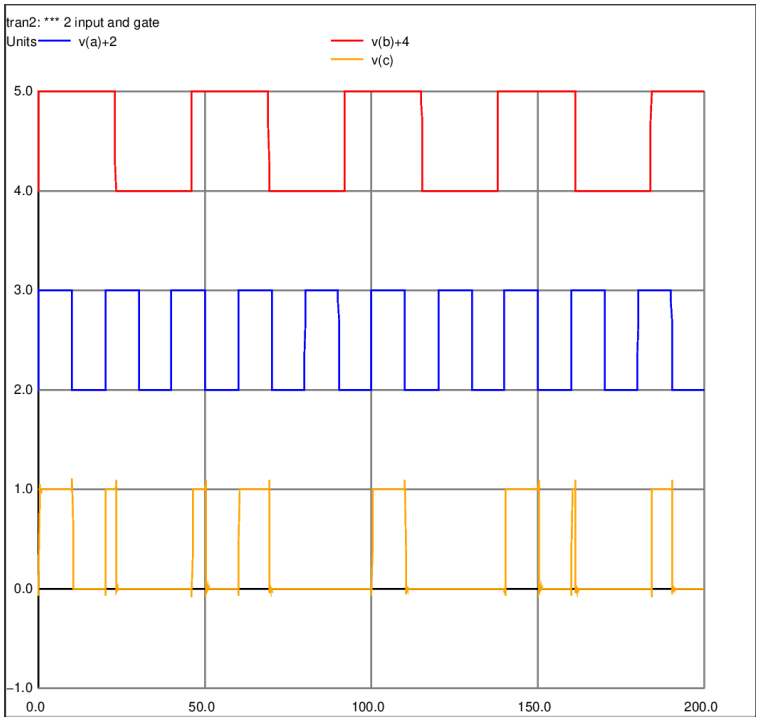
2-INPUT AND GATE

NETLIST

```
*** 2 INPUT AND GATE

.INCLUDE /home/maggy/VLSI/Final_Project/TSMC_180nm.txt

*PARAMETERS
.PARAM X=90nm

.PARAM Width_p=4*X
.PARAM Length_p=2*X

.PARAM Width_n=2*X
.PARAM Length_n=2*X

.PARAM supply=1
.global gnd vdd

.temp 25

*NET-LIST

* AND GATE
*       ----------------------------------------VDD
*         |          |                          |
*   A -- |Mp1|   B --|Mp2|                       |
*         |          |                          |
*          ------------                  ______|Mp3|
*               |                       |       |
*               ---------node1----------|       |--------C
*               |                       |       |
*         A -- |Mn1|                    |______|Mn3|
*               |________node2                   |
*               |                                |
*         B -- |Mn2|                             |
*               |                                |
*        ---------------------------------------GND

*A and B are INPUTS
*C os OUTPUT
*node1 is output of NAND gate
*node2 is the common terminal for the 2 NMOS transistors

Mp1 node1   A     vdd   vdd   CMOSP W={2*Width_p} L={Length_p}
Mp2 node1   B     vdd   vdd   CMOSP W={2*Width_p} L={Length_p}
Mn1 node1   A    node2  gnd   CMOSN W={2*Width_n} L={Length_n}
Mn2 node2   B     gnd   gnd   CMOSN W={2*Width_n} L={Length_n}
Mp3   C   node1   vdd   vdd   CMOSP W={2*Width_p} L={Length_p}
Mn3   C   node1   gnd   gnd   CMOSN W={2*Width_n} L={Length_n}
```

```
*SOURCE
VDD vdd gnd 'supply'

*Capacitive LOAD
CL C gnd 2f

*INPUT WAVEFORM
VinA A gnd pulse(0 1 0 100p 100p 10n 20n 0)
VinB B gnd pulse(0 1 0 100p 100p 23n 46n 0)

*ANALYSIS
.tran 0.1n 0.2u

*CONTROL COMMANDS
.CONTROL
set hcopypscolor = 1
set color0=white
set color1=black

run
hardcopy Pre_Layout_pd_AND.eps v(B)+4 v(A)+2 v(C)

.endc
```

SIMULATION RESULTS



HALF ADDER

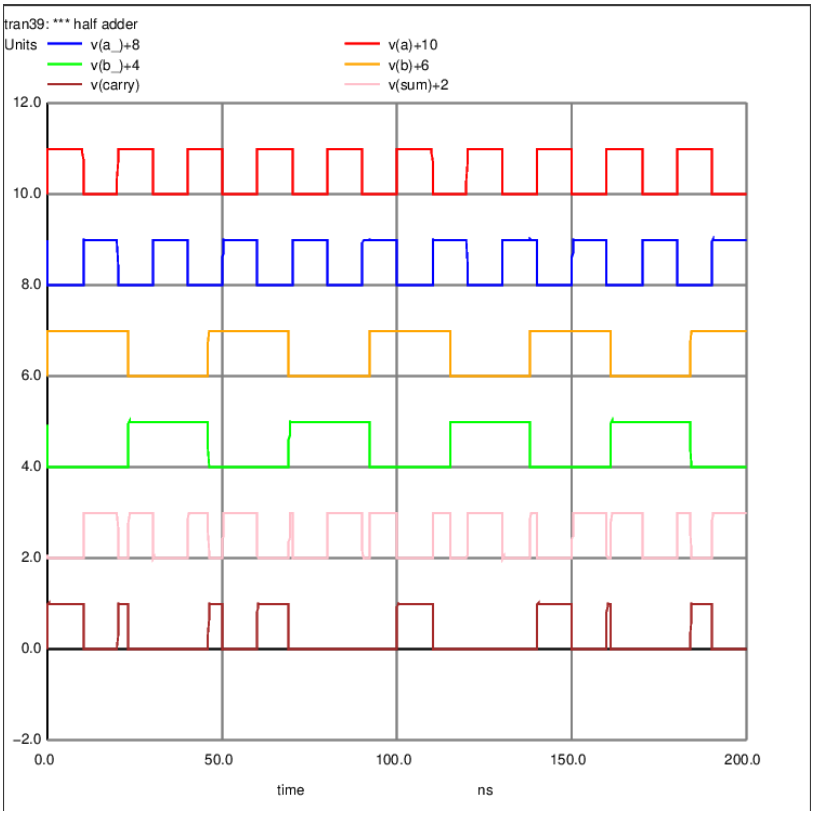
NETLIST

```
*** HALF ADDER
.INCLUDE /home/maggy/VLSI/Final_Project/TSMC_180nm.txt
.INCLUDE /home/maggy/VLSI/Final_Project/INVERTER.sub

*PARAMETERS
.PARAM X=90nm

.PARAM Width_p=4*X
.PARAM Length_p=2*X

.PARAM Width_n=2*X
.PARAM Length_n=2*X

.PARAM supply=1
.global gnd vdd

.temp 25

*NET-LIST
*HALF ADDER

*SUM
*        -------------------VDD
*           |              |
*    A' -- |Mp1|   A  -- |Mp3|
*           |              |
*   node1-----             -----node2
*           |              |
*    B  -- |Mp2|   B' -- |Mp4|
*           |______________|
*                  |
*                  ---------SUM
*                  |
*              ____|_____
*           |              |
*    A' -- |Mn1|    A -- |Mn3|
*           |              |
*   node3-----             -----node4
*           |              |
*    B' -- |Mn2|    B -- |Mn4|
*           |______________|
*                  |
*         -----------------GND
```

```
*CARRY
*       ---------------------------------------VDD
*          |           |                       |
*   A -- |Mp5|   B --|Mp6|                      |
*          |           |                       |
*          ------------                  ______|Mp7|
*                |                   |         |
*                ---------node5------|         |--------CARRY
*                |                   |         |
*         A -- |Mn5|                 |_____|Mn7|
*                |________node6                |
*                |                             |
*         B -- |Mn6|                            |
*                |                             |
*        ---------------------------------------GND

*A and B are INPUTS
*A and A_
xinv1 A vdd gnd A_ INVERTER

*B and B_
xinvv2 B vdd gnd B_ INVERTER

Mp1 node1   A_     vdd   vdd   CMOSP W={2*Width_p} L={Length_p}
Mp2 SUM     B     node1  vdd   CMOSP W={2*Width_p} L={Length_p}
Mp3 node2   A      vdd   vdd   CMOSP W={2*Width_p} L={Length_p}
Mp4 SUM     B_    node2  vdd   CMOSP W={2*Width_p} L={Length_p}
Mn1 SUM     A_    node3  gnd   CMOSN W={2*Width_n} L={Length_n}
Mn2 node3   B_     gnd   gnd   CMOSN W={2*Width_n} L={Length_n}
Mn3 SUM     A     node4  gnd   CMOSN W={2*Width_n} L={Length_n}
Mn4 node4   B      gnd   gnd   CMOSN W={2*Width_n} L={Length_n}

Mp5 node5    A     vdd   vdd   CMOSP W={2*Width_p} L={Length_p}
Mp6 node5    B     vdd   vdd   CMOSP W={2*Width_p} L={Length_p}
Mn5 node5    A    node6  gnd   CMOSN W={2*Width_n} L={Length_n}
Mn6 node6    B     gnd   gnd   CMOSN W={2*Width_n} L={Length_n}
Mp7 CARRY  node5   vdd   vdd   CMOSP W={2*Width_p} L={Length_p}
Mn7 CARRY  node5   gnd   gnd   CMOSN W={2*Width_n} L={Length_n}

*SOURCE
VDD vdd gnd 'supply'

*Capacitive LOAD
CL1 SUM gnd 2f
CL2 CARRY gnd 2f
```

```
*INPUT WAVEFORM
VinA A gnd pulse(0 1 0 100p 100p 10n 20n 0)
VinB B gnd pulse(0 1 0 100p 100p 23n 46n 0)

*ANALYSIS
.tran 0.1n 0.2u

*CONTROL COMMANDS
.CONTROL
set hcopypscolor = 1
set color0=white
set color1=black

run
hardcopy Pre_Layout_pd_HA.eps v(A)+10 v(A_)+8 v(B)+6 v(B_)+4 v(SUM)+2 v(CARRY)

.endc
```

SIMULATION RESULTS



FULL ADDER

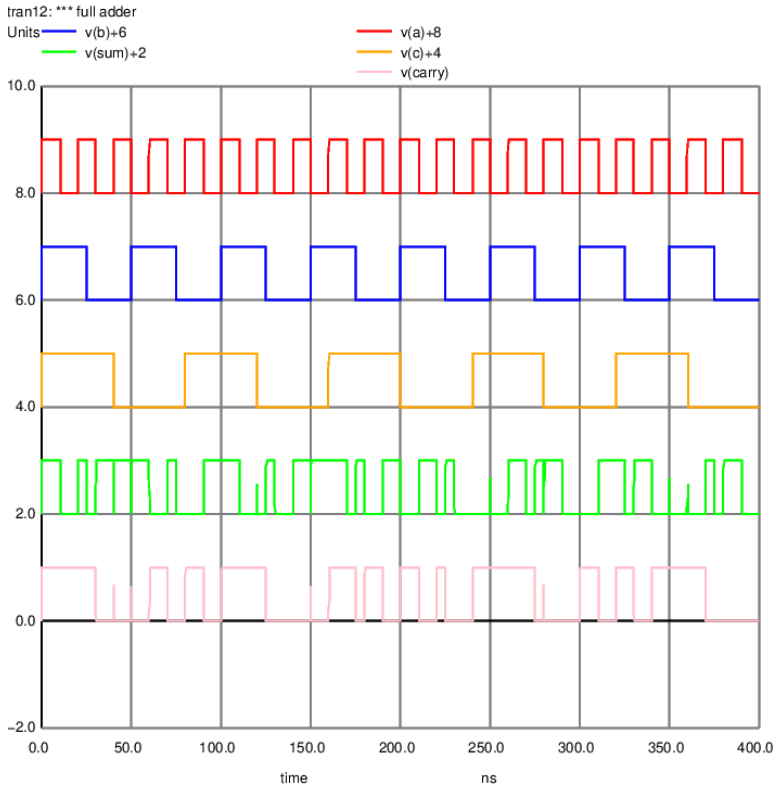
NETLIST

```
*** FULL ADDER

.INCLUDE /home/maggy/VLSI/Final_Project/TSMC_180nm.txt
.INCLUDE /home/maggy/VLSI/Final_Project/INVERTER.sub


*PARAMETERS
.PARAM X=90nm

.PARAM Width_p=4*X
.PARAM Length_p=2*X

.PARAM Width_n=2*X
.PARAM Length_n=2*X

.PARAM supply=1
.global gnd vdd

.temp 25

*NET-LIST

*FULL ADDER

*CARRY
*  ------------------------------------------------------VDD
*          |           |                 |
*   A -- |Mp1|   B --|Mp2|          B -- |Mp4|
*          |           |                 |
*          -------------                 |-------node3
*                |                       |
*                ---------node1     A -- |Mp5|
*                |                       |
*         ---- |Mp3|                      |          _____
*    C____|      |______________________|_________CARRY____|inverter|___CARRY
*         |      |                       |
*         ---- |Mn3|                     |
*                |                       |
*                ---------node2     A -- |Mn5|
*                |                       |
*          -------------                 |-------node4
*          |           |                 |
*   A -- |Mn1|   B --|Mn2|          B -- |Mn4|
*          |           |                 |
* ------------------------------------------------------GND
```

```
*SUM
*-------------------------------------------------------VDD
*         |            |            |              |
*  A -- |Mp6|   B -- |Mp7|   C -- |Mp8|     A -- |Mp10|
*         |____________|____________|              |______node7
*                      |                           |
*                      ---------node5       B -- |Mp11|
*                      |                           |______node8
*              -----|Mp9|                           |
*             |       |                     C -- |Mp12|
*      _____  |       |                           |          ___
*     CARRY ---        ------------------------------------SUM---|inverter|----SUM
*             |       |                           |
*             |       |                     C -- |Mn12|
*              -----|Mn9|                          |______node9
*                      |                           |
*                      ---------node6        B -- |Mn11|
*         _____________|____________               |______node10
*         |            |            |              |
*  A -- |Mn6|   B -- |Mn7|   C -- |Mn8|     A -- |Mn10|
*         |            |            |              |
*-------------------------------------------------------GND
*

* A B and C are inputs
* CARRY and SUM are the outputs

Mp1 node1  A  vdd  vdd CMOSP W={2*Width_p} L={Length_p}
Mp2 node1  B  vdd  vdd CMOSP W={2*Width_p} L={Length_p}
Mp3 CARRY_ C node1 vdd CMOSP W={2*Width_p} L={Length_p}

Mn1 node2  A  gnd  gnd CMOSN W={2*Width_n} L={Length_n}
Mn2 node2  B  gnd  gnd CMOSN W={2*Width_n} L={Length_n}
Mn3 CARRY_ C node2 gnd CMOSN W={2*Width_n} L={Length_n}

Mp4 node3  B  vdd  vdd CMOSP W={2*Width_p} L={Length_p}
Mp5 CARRY_ A node3 vdd CMOSP W={2*Width_p} L={Length_p}

Mn4 node4  B  gnd  gnd CMOSN W={2*Width_n} L={Length_n}
Mn5 CARRY_ A node4 gnd CMOSN W={2*Width_n} L={Length_n}

xinv1 CARRY_ vdd gnd CARRY INVERTER

Mp6 node5   A     vdd   vdd CMOSP W={2*Width_p} L={Length_p}
Mp7 node5   B     vdd   vdd CMOSP W={2*Width_p} L={Length_p}
Mp8 node5   C     vdd   vdd CMOSP W={2*Width_p} L={Length_p}
Mp9 SUM_  CARRY_ node5  vdd CMOSP W={2*Width_p} L={Length_p}
```

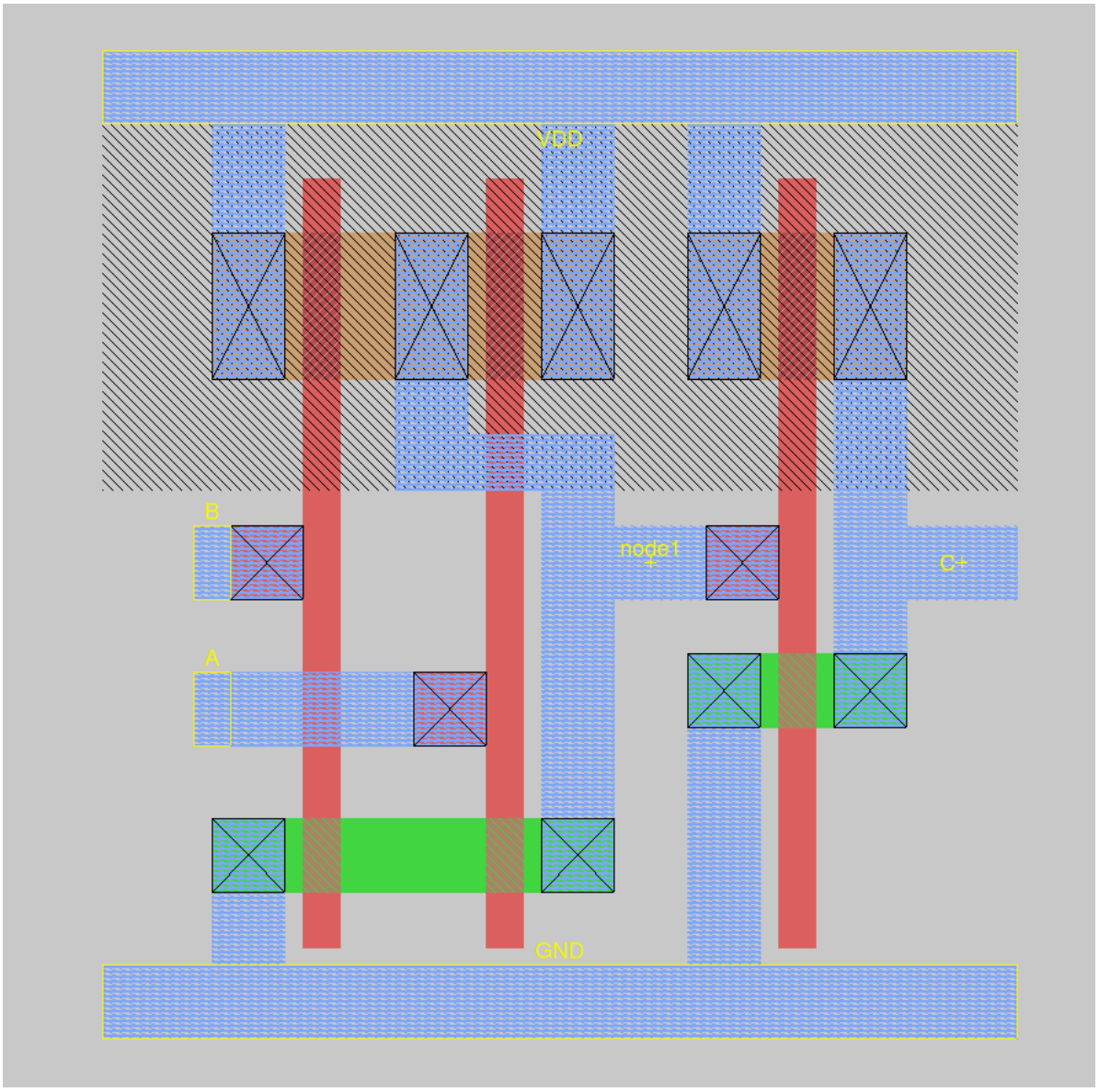
```
Mn6 node6   A     gnd   gnd CMOSN W={2*Width_n} L={Length_n}
Mn7 node6   B     gnd   gnd CMOSN W={2*Width_n} L={Length_n}
Mn8 node6   C     gnd   gnd CMOSN W={2*Width_n} L={Length_n}
Mn9 SUM_  CARRY_ node6  gnd CMOSN W={2*Width_n} L={Length_n}

Mp10 node7  A   vdd   vdd CMOSP W={2*Width_p} L={Length_p}
Mp11 node8  B  node7  vdd CMOSP W={2*Width_p} L={Length_p}
Mp12 SUM_   C  node8  vdd CMOSP W={2*Width_p} L={Length_p}

Mn10 node10 A   gnd   gnd CMOSN W={2*Width_n} L={Length_n}
Mn11 node9  B  node10 gnd CMOSN W={2*Width_n} L={Length_n}
Mn12 SUM_   C  node9  gnd CMOSN W={2*Width_n} L={Length_n}

xinv2 SUM_ vdd gnd SUM INVERTER

*SOURCE
VDD vdd gnd 'supply'

*Capacitive LOAD
CL1 SUM gnd 2f
CL2 CARRY gnd 2f

*INPUT WAVEFORM
VinA A gnd pulse(0 1 0 100p 100p 10n 20n 0)
VinB B gnd pulse(0 1 0 100p 100p 25n 50n 0)
VinC C gnd pulse(0 1 0 100p 100p 40n 80n 0)

*ANALYSIS
.tran 0.1n 0.4u

*CONTROL COMMANDS
.CONTROL
set hcopypscolor = 1
set color0=white
set color1=black

run
hardcopy Pre_Layout_pd_FA.eps v(A)+8 v(B)+6 v(C)+4 v(SUM)+2 v(CARRY)

.endc
```

SIMULATION RESULTS

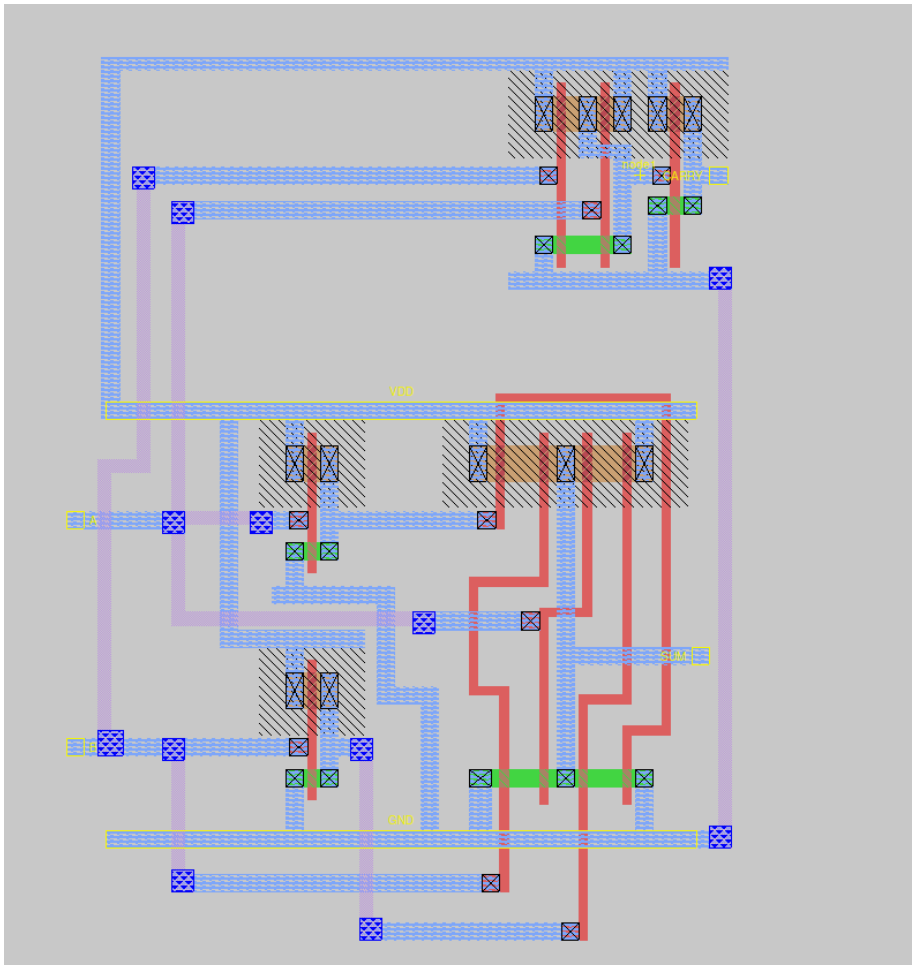


MAGIC LAYOUTS

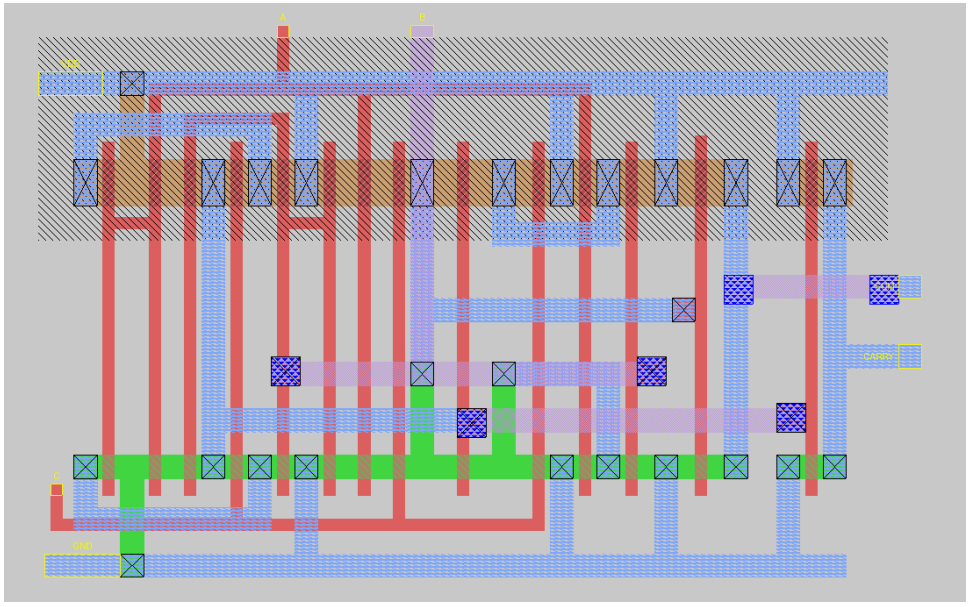
AND GATE



HALF ADDER



FULL ADDER



POST-LAYOUT SIMULATIONS

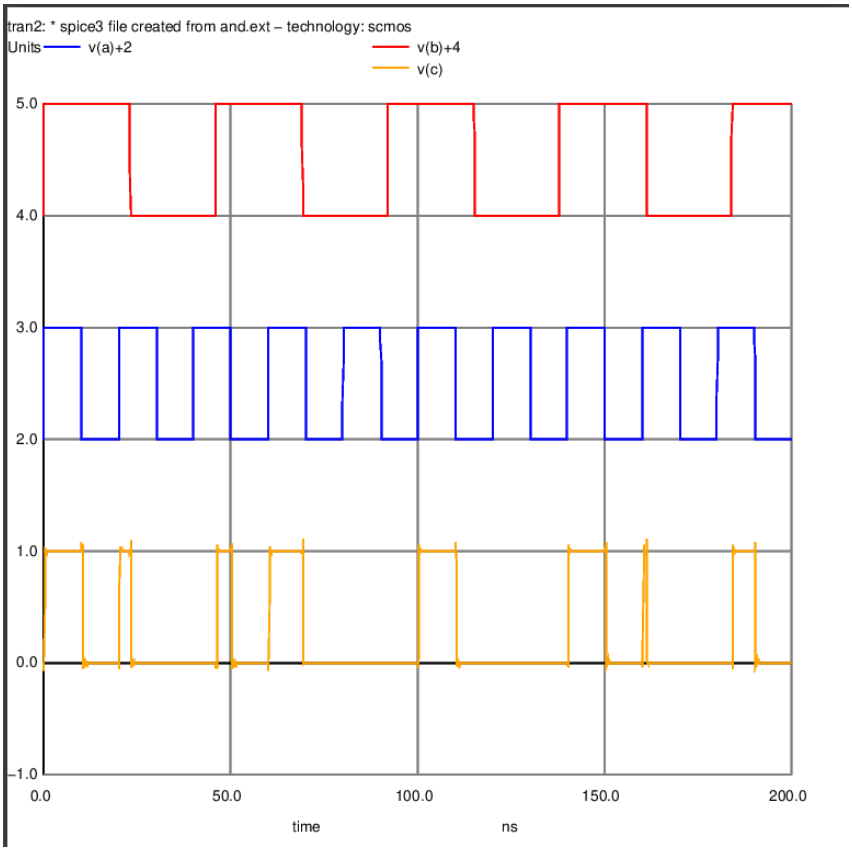
AND GATE

NETLIST

```
* SPICE3 file created from AND.ext - technology: scmos

.include /home/maggy/VLSI/Final_Project/TSMC_180nm.txt

.option scale=0.09u

*PARAMETERS
.param supply=1

.global gnd vdd

*SOURCE
VDD vdd gnd 'supply'

*INPUT WAVEFORM
VinA A gnd pulse(0 1 0 100p 100p 10n 20n 0)
VinB B gnd pulse(0 1 0 100p 100p 23n 46n 0)

M1000 C node1 VDD w_n24_n5# CMOSP w=8 l=2
+  ad=40 pd=26 as=120 ps=78
M1001 VDD B node1 w_n24_n5# CMOSP w=8 l=2
+  ad=0 pd=0 as=64 ps=32
M1002 node1 B a_n11_n27# Gnd CMOSN w=4 l=2
+  ad=20 pd=18 as=32 ps=24
M1003 node1 A VDD w_n24_n5# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1004 C node1 GND Gnd CMOSN w=4 l=2
+  ad=20 pd=18 as=40 ps=36
M1005 a_n11_n27# A GND Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
C0 B node1 0.12fF
C1 A B 0.16fF
C2 VDD node1 0.22fF
C3 GND node1 0.23fF
C4 w_n24_n5# VDD 0.20fF
C5 w_n24_n5# node1 0.10fF
C6 VDD C 0.11fF
C7 GND Gnd 0.19fF
C8 node1 Gnd 0.26fF
C9 B Gnd 0.21fF
C10 A Gnd 0.18fF
C11 w_n24_n5# Gnd 1.21fF


*ANALYSIS
.tran 0.1n 0.2u
```

```
*CONTROL COMMANDS
.CONTROL
set hcopypscolor = 1
set color0=white
set color1=black

run
hardcopy Post_Layout_pd_AND.eps v(B)+4 v(A)+2 v(C)

.endc
```

SIMULATION RESULTS



HALF ADDER

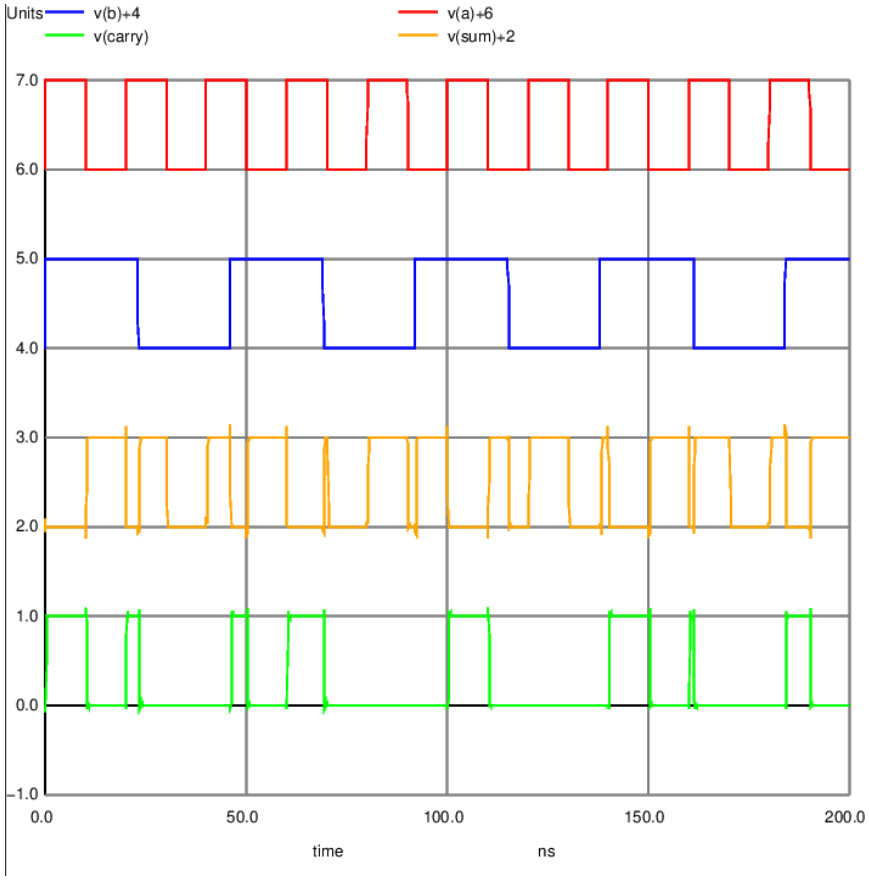
NETLIST

```
* SPICE3 file created from HA.ext - technology: scmos

.include /home/maggy/VLSI/Final_Project/TSMC_180nm.txt

.option scale=0.09u

*PARAMETERS
.param supply=1

.global gnd vdd

*SOURCE
VDD vdd gnd 'supply'

*INPUT WAVEFORM
VinA A gnd pulse(0 1 0 100p 100p 10n 20n 0)
VinB B gnd pulse(0 1 0 100p 100p 23n 46n 0)
```

```
M1000 a_n38_n315# A GND Gnd CMOSN w=4 l=2
+  ad=20 pd=18 as=128 ps=112
M1001 a_6_n367# B GND Gnd CMOSN w=4 l=2
+  ad=28 pd=22 as=0 ps=0
M1002 node1 B VDD w_6_n223# CMOSP w=8 l=2
+  ad=64 pd=32 as=288 ps=184
M1003 CARRY node1 GND Gnd CMOSN w=4 l=2
+  ad=20 pd=18 as=0 ps=0
M1004 a_5_n297# a_n38_n315# VDD w_n9_n303# CMOSP w=8 l=2
+  ad=64 pd=32 as=0 ps=0
M1005 GND a_n38_n315# a_24_n367# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=32 ps=24
M1006 VDD a_n38_n367# a_25_n297# w_n9_n303# CMOSP w=8 l=2
+  ad=0 pd=0 as=56 ps=30
M1007 a_n38_n367# B GND Gnd CMOSN w=4 l=2
+  ad=20 pd=18 as=0 ps=0
M1008 a_n38_n315# A VDD w_n51_n303# CMOSP w=8 l=2
+  ad=40 pd=26 as=0 ps=0
M1009 node1 A a_19_n245# Gnd CMOSN w=4 l=2
+  ad=20 pd=18 as=32 ps=24
M1010 a_25_n297# A SUM w_n9_n303# CMOSP w=8 l=2
+  ad=0 pd=0 as=64 ps=32
M1011 a_24_n367# a_n38_n367# SUM Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=28 ps=22
M1012 CARRY node1 VDD w_6_n223# CMOSP w=8 l=2
+  ad=40 pd=26 as=0 ps=0
M1013 VDD A node1 w_6_n223# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1014 SUM A a_6_n367# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1015 a_19_n245# B GND Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1016 SUM B a_5_n297# w_n9_n303# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1017 a_n38_n367# B VDD w_n51_n355# CMOSP w=8 l=2
+  ad=40 pd=26 as=0 ps=0
C0 VDD w_n51_n355# 0.05fF
C1 a_n38_n315# GND 0.07fF
C2 VDD w_n9_n303# 0.11fF
C3 A B 1.39fF
C4 a_n38_n367# VDD 0.11fF
C5 SUM A 0.12fF
C6 VDD w_n51_n303# 0.05fF
C7 CARRY w_6_n223# 0.03fF
C8 a_n38_n315# VDD 0.72fF
C9 VDD w_6_n223# 0.13fF
C10 A w_n9_n303# 0.06fF
C11 B w_n51_n355# 0.06fF
C12 GND CARRY 0.04fF
C13 a_n38_n367# A 0.18fF
C14 A w_n51_n303# 0.06fF
C15 node1 w_6_n223# 0.10fF
C16 B w_n9_n303# 0.06fF
C17 a_n38_n367# B 0.32fF
C18 a_n38_n315# A 0.06fF
C19 GND VDD 0.20fF
C20 SUM w_n9_n303# 0.02fF
C21 SUM a_n38_n367# 0.08fF
C22 A w_6_n223# 0.06fF
C23 GND node1 0.23fF
C24 CARRY VDD 0.11fF
C25 a_n38_n367# w_n51_n355# 0.03fF
C26 SUM a_n38_n315# 0.08fF
C27 B w_6_n223# 0.06fF
C28 GND A 0.22fF
C29 a_n38_n367# w_n9_n303# 0.06fF
C30 CARRY node1 0.05fF
C31 GND B 0.73fF
C32 VDD node1 0.22fF
C33 a_n38_n315# w_n9_n303# 0.19fF
C34 a_n38_n367# a_n38_n315# 0.19fF
C35 a_n38_n315# w_n51_n303# 0.03fF
C36 VDD A 0.31fF
C37 node1 A 0.12fF
C38 VDD B 0.17fF
C39 SUM VDD 0.03fF
C40 a_n38_n367# GND 0.19fF
C41 SUM Gnd 0.19fF
```

```
C42 a_n38_n367# Gnd 0.32fF
C43 a_n38_n315# Gnd 0.20fF
C44 GND Gnd 0.69fF
C45 CARRY Gnd 0.05fF
C46 VDD Gnd 0.18fF
C47 node1 Gnd 0.26fF
C48 A Gnd 1.07fF
C49 B Gnd 1.12fF
C50 w_n51_n355# Gnd 0.48fF
C51 w_n9_n303# Gnd 1.12fF
C52 w_n51_n303# Gnd 0.48fF
C53 w_6_n223# Gnd 1.00fF


*SOURCE
VDD vdd gnd 'supply'

*INPUT WAVEFORM
VinA A gnd pulse(0 1 0 100p 100p 10n 20n 0)
VinB B gnd pulse(0 1 0 100p 100p 23n 46n 0)

*ANALYSIS
.tran 0.1n 0.2u


*CONTROL COMMANDS
.CONTROL
set hcopypscolor = 1
set color0=white
set color1=black

run
hardcopy Post_Layout_pd_HA.eps v(A)+6 v(B)+4 v(SUM)+2 v(CARRY)

.endc
```

SIMULATION RESULTS



FULL ADDER

NETLIST

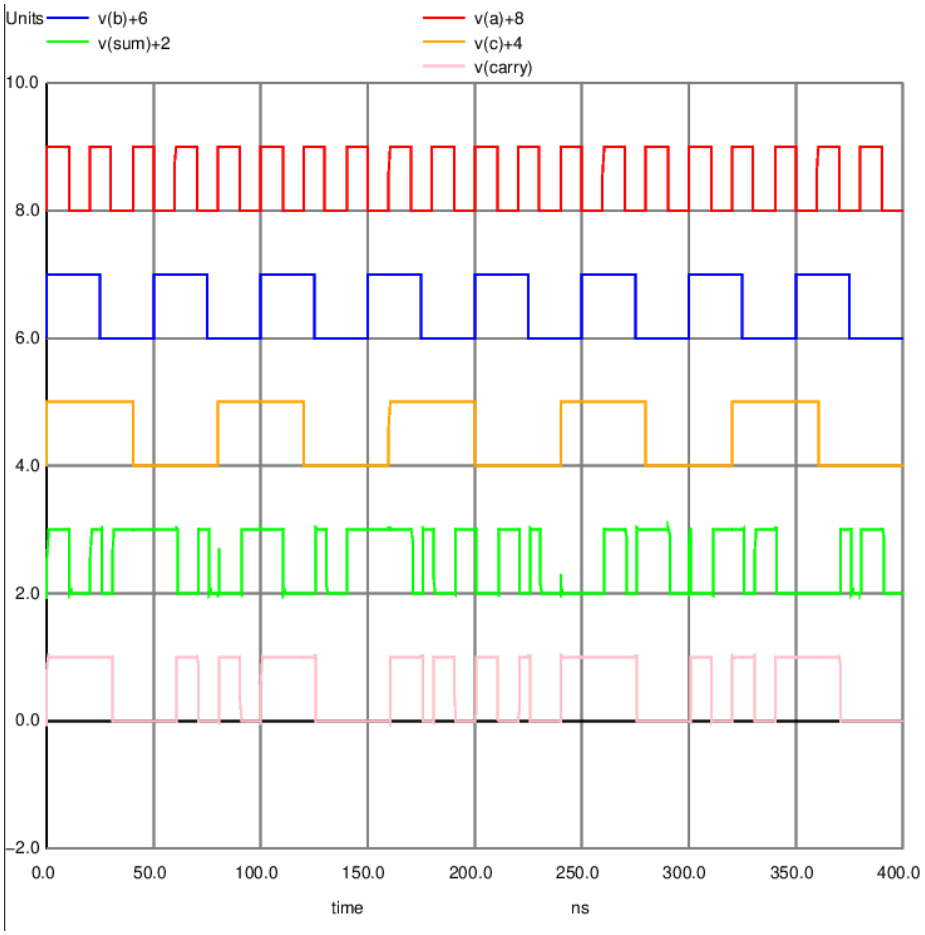
```
* SPICE3 file created from FA.ext - technology: scmos

.include /home/maggy/VLSI/Final_Project/TSMC_180nm.txt

.option scale=0.09u

*PARAMETERS
.param supply=1

.global gnd vdd

*SOURCE
VDD vdd gnd 'supply'

M1000 SUM a_n25_n513# VDD w_n88_n472# CMOSP w=8 l=2
+  ad=56 pd=30 as=324 ps=176
M1001 a_n14_n513# A GND Gnd CMOSN w=4 l=2
+  ad=132 pd=82 as=140 ps=110
M1002 a_n61_n513# B a_n67_n466# w_n88_n472# CMOSP w=8 l=2
+  ad=48 pd=28 as=32 ps=24
M1003 a_n31_n466# A a_n37_n466# w_n88_n472# CMOSP w=8 l=2
+  ad=32 pd=24 as=32 ps=24
M1004 a_n82_n466# C a_n61_n513# w_n88_n472# CMOSP w=8 l=2
+  ad=88 pd=54 as=0 ps=0
M1005 a_n75_n513# C GND Gnd CMOSN w=4 l=2
+  ad=52 pd=34 as=0 ps=0
M1006 VDD A a_n82_n466# w_n88_n472# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1007 a_n25_n513# C a_n31_n513# Gnd CMOSN w=4 l=2
+  ad=100 pd=58 as=16 ps=16
M1008 VDD B a_n82_n466# w_n88_n472# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1009 GND C a_n14_n513# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1010 SUM a_n25_n513# GND Gnd CMOSN w=4 l=2
+  ad=28 pd=22 as=0 ps=0
M1011 a_n67_n466# A VDD w_n88_n472# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1012 a_n37_n466# B VDD w_n88_n472# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1013 a_n14_n466# a_n61_n513# a_n25_n513# w_n88_n472# CMOSP w=8 l=2
+  ad=136 pd=66 as=72 ps=34
M1014 a_n61_n513# B a_n67_n513# Gnd CMOSN w=4 l=2
+  ad=24 pd=20 as=16 ps=16
M1015 CARRY a_n61_n513# VDD w_n88_n472# CMOSP w=8 l=2
+  ad=48 pd=28 as=0 ps=0
M1016 a_n31_n513# A a_n37_n513# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=16 ps=16
```

```
M1017 a_n82_n513# C a_n61_n513# Gnd CMOSN w=4 l=2
+  ad=44 pd=38 as=0 ps=0
M1018 a_n75_n513# A a_n82_n513# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1019 GND B a_n82_n513# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1020 VDD B a_n14_n466# w_n88_n472# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1021 a_n67_n513# A a_n75_n513# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1022 a_n37_n513# B GND Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1023 a_n14_n466# A VDD w_n88_n472# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1024 a_n14_n513# a_n61_n513# a_n25_n513# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1025 CARRY a_n61_n513# GND Gnd CMOSN w=4 l=2
+  ad=20 pd=18 as=0 ps=0
M1026 GND B a_n14_n513# Gnd CMOSN w=4 l=2
+  ad=0 pd=0 as=0 ps=0
M1027 a_n25_n513# C a_n31_n466# w_n88_n472# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
M1028 VDD C a_n14_n466# w_n88_n472# CMOSP w=8 l=2
+  ad=0 pd=0 as=0 ps=0
C0 VDD CARRY 0.08fF
C1 a_n25_n513# SUM 0.03fF
C2 C a_n14_n466# 0.08fF
C3 SUM GND 0.03fF
C4 a_n14_n466# A 0.08fF
C5 C A 0.50fF
C6 w_n88_n472# a_n61_n513# 0.15fF
C7 a_n82_n466# a_n61_n513# 0.12fF
C8 B a_n61_n513# 0.31fF
C9 a_n14_n513# a_n61_n513# 0.09fF
C10 a_n25_n513# w_n88_n472# 0.09fF
C11 CARRY a_n61_n513# 0.07fF
C12 B a_n25_n513# 0.28fF
C13 a_n82_n513# a_n61_n513# 0.12fF
C14 a_n25_n513# a_n14_n513# 0.13fF
C15 a_n14_n466# w_n88_n472# 0.06fF
C16 a_n14_n513# GND 0.07fF
C17 C w_n88_n472# 0.19fF
C18 CARRY GND 0.04fF
C19 A w_n88_n472# 0.89fF
C20 B C 0.43fF
C21 a_n82_n466# A 0.08fF
C22 a_n82_n513# GND 0.49fF
C23 a_n14_n466# VDD 0.25fF
```

```
C24 C a_n14_n513# 0.08fF
C25 B A 0.91fF
C26 w_n88_n472# SUM 0.03fF
C27 A a_n14_n513# 0.08fF
C28 VDD A 0.67fF
C29 C a_n82_n513# 0.29fF
C30 a_n25_n513# a_n61_n513# 0.14fF
C31 VDD SUM 0.07fF
C32 GND a_n61_n513# 0.08fF
C33 SUM CARRY 0.17fF
C34 a_n82_n466# w_n88_n472# 0.12fF
C35 C a_n61_n513# 0.17fF
C36 B w_n88_n472# 0.67fF
C37 B a_n82_n466# 0.16fF
C38 A a_n61_n513# 0.08fF
C39 VDD w_n88_n472# 0.56fF
C40 a_n82_n466# VDD 0.65fF
C41 B a_n14_n513# 0.14fF
C42 w_n88_n472# CARRY 0.02fF
C43 SUM a_n61_n513# 0.13fF
C44 a_n25_n513# C 0.08fF
C45 B VDD 0.09fF
C46 a_n25_n513# A 0.08fF
C47 C GND 0.08fF
C48 GND Gnd 0.25fF
C49 a_n82_n513# Gnd 0.12fF
C50 a_n14_n513# Gnd 0.10fF
C51 CARRY Gnd 0.17fF
C52 SUM Gnd 0.43fF
C53 a_n14_n466# Gnd 0.03fF
C54 a_n25_n513# Gnd 0.09fF
C55 a_n61_n513# Gnd 0.20fF
C56 C Gnd 0.93fF
C57 B Gnd 1.45fF
C58 A Gnd 0.75fF
C59 w_n88_n472# Gnd 2.39fF

*SOURCE
VDD vdd gnd 'supply'

*INPUT WAVEFORM
VinA A gnd pulse(0 1 0 100p 100p 10n 20n 0)
VinB B gnd pulse(0 1 0 100p 100p 25n 50n 0)
VinC C gnd pulse(0 1 0 100p 100p 40n 80n 0)

*ANALYSIS
.tran 0.1n 0.4u
```

```
*CONTROL COMMANDS
.CONTROL
set hcopypscolor = 1
set color0=white
set color1=black

run
hardcopy Post_Layout_pd_FA.eps v(A)+8 v(B)+6 v(C)+4 v(SUM)+2 v(CARRY)

.endc
```

SIMULATION RESULTS



PRE-LAYOUT PROPAGATION DELAYS

AND GATE

```
  Measurements for Transient Analysis

delay_lh_a          =  2.029255e-10 targ=  2.529255e-10 trig=  5.000000e-11
delay_hl_a          =  1.609871e-10 targ=  1.031099e-08 trig=  1.015000e-08
pd_a                =  1.81956e-10
delay_lh_b          =  2.029255e-10 targ=  2.529255e-10 trig=  5.000000e-11
delay_hl_b          =  1.653085e-10 targ=  2.331531e-08 trig=  2.315000e-08
pd_b                =  1.84117e-10
```

HALF ADDER

```
  Measurements for Transient Analysis

delay_lh_a_inv      =  7.270791e-11 targ=  1.022271e-08 trig=  1.015000e-08
delay_hl_a_inv      =  5.651948e-11 targ=  1.065195e-10 trig=  5.000000e-11
pd_a_inv            =  6.46137e-11
delay_lh_b_inv      =  6.383620e-11 targ=  2.321384e-08 trig=  2.315000e-08
delay_hl_b_inv      =  5.582063e-11 targ=  1.058206e-10 trig=  5.000000e-11
pd_b_inv            =  5.98284e-11
delay_lh_a_carry    =  1.933771e-10 targ=  2.433771e-10 trig=  5.000000e-11
delay_hl_a_carry    =  1.522618e-10 targ=  1.030226e-08 trig=  1.015000e-08
pd_a_carry          =  1.72819e-10
delay_lh_b_carry    =  1.933771e-10 targ=  2.433771e-10 trig=  5.000000e-11
delay_hl_b_carry    =  1.718717e-10 targ=  2.332187e-08 trig=  2.315000e-08
pd_b_carry          =  1.82624e-10
delay_lh_a_sum      =  1.550552e-10 targ=  1.030506e-08 trig=  1.015000e-08
delay_hl_a_sum      =  9.601530e-11 targ=  2.014602e-08 trig=  2.005000e-08
pd_a_sum            =  1.25535e-10
delay_lh_b_sum      =  1.477459e-10 targ=  2.329775e-08 trig=  2.315000e-08
delay_hl_b_sum      =  8.663975e-11 targ=  4.613664e-08 trig=  4.605000e-08
pd_b_sum            =  1.17193e-10
delay_lh_ainv_sum   =  1.648550e-10 targ=  4.026964e-08 trig=  4.010478e-08
delay_hl_ainv_sum   =  1.180707e-10 targ=  3.034486e-08 trig=  3.022679e-08
pd_ainv_sum         =  1.41463e-10
delay_lh_binv_sum   =  1.647268e-10 targ=  9.226988e-08 trig=  9.210515e-08
delay_hl_binv_sum   =  1.130998e-09 targ=  7.034483e-08 trig=  6.921383e-08
pd_binv_sum         =  6.47863e-10
```

FULL ADDER

```
  Measurements for Transient Analysis

delay_lh_carry_inv  =  8.815379e-11 targ=  2.190399e-10 trig=  1.308861e-10
delay_hl_carry_inv  =  1.350504e-10 targ=  3.054494e-08 trig=  3.040989e-08
pd_carry_inv        =  1.11602e-10
delay_lh_sum_inv    =  1.250701e-10 targ=  2.616475e-10 trig=  1.365773e-10
delay_hl_sum_inv    =  9.795082e-11 targ=  1.046978e-08 trig=  1.037183e-08
pd_sum_inv          =  1.11510e-10
delay_lh_a_sum      =  5.713277e-10 targ=  3.072133e-08 trig=  3.015000e-08
delay_hl_a_sum      =  5.041978e-08 targ=  1.104698e-07 trig=  6.005000e-08
pd_a_sum            =  2.54956e-08
delay_lh_b_sum      =  6.557176e-08 targ=  1.907218e-07 trig=  1.251500e-07
delay_hl_b_sum      =  2.553792e-08 targ=  7.558792e-08 trig=  5.005000e-08
pd_b_sum            =  4.55548e-08
delay_lh_c_sum      =  3.058839e-08 targ=  7.073839e-08 trig=  4.015000e-08
delay_hl_c_sum      =  9.042744e-08 targ=  2.905774e-07 trig=  2.001500e-07
pd_c_sum            =  6.05079e-08
delay_lh_a_carry    =  8.024709e-08 targ=  1.802971e-07 trig=  1.000500e-07
delay_hl_a_carry    =  3.047712e-08 targ=  7.052712e-08 trig=  4.005000e-08
pd_a_carry          =  5.53621e-08
delay_lh_b_carry    =  8.024709e-08 targ=  1.802971e-07 trig=  1.000500e-07
delay_hl_b_carry    =  4.049494e-08 targ=  9.054494e-08 trig=  5.005000e-08
pd_b_carry          =  6.03710e-08
delay_lh_c_carry    =  6.028394e-08 targ=  2.203339e-07 trig=  1.600500e-07
delay_hl_c_carry    =  3.037712e-08 targ=  7.052712e-08 trig=  4.015000e-08
pd_c_carry          =  4.53305e-08
```

POST-LAYOUT PROPAGATION DELAYS

AND GATE

```
  Measurements for Transient Analysis

delay_lh_a          =  2.420373e-10 targ=  2.920373e-10 trig=  5.000000e-11
delay_hl_a          =  2.075342e-10 targ=  1.035753e-08 trig=  1.015000e-08
pd_a                =  2.24786e-10
delay_lh_b          =  2.420373e-10 targ=  2.920373e-10 trig=  5.000000e-11
delay_hl_b          =  1.994297e-10 targ=  2.334943e-08 trig=  2.315000e-08
pd_b                =  2.20734e-10
```

HALF ADDER

```
  Measurements for Transient Analysis

delay_lh_a_carry    =  2.557521e-10 targ=  3.057521e-10 trig=  5.000000e-11
delay_hl_a_carry    =  2.068126e-10 targ=  1.035681e-08 trig=  1.015000e-08
pd_a_carry          =  2.31282e-10
delay_lh_b_carry    =  2.557521e-10 targ=  3.057521e-10 trig=  5.000000e-11
delay_hl_b_carry    =  2.185298e-10 targ=  2.336853e-08 trig=  2.315000e-08
pd_b_carry          =  2.37141e-10
delay_lh_a_sum      =  1.748002e-10 targ=  1.032480e-08 trig=  1.015000e-08
delay_hl_a_sum      =  9.137763e-11 targ=  2.014138e-08 trig=  2.005000e-08
pd_a_sum            =  1.33089e-10
delay_lh_b_sum      =  2.227987e-10 targ=  2.337280e-08 trig=  2.315000e-08
delay_hl_b_sum      =  9.595899e-11 targ=  4.614596e-08 trig=  4.605000e-08
pd_b_sum            =  1.59379e-10
```

FULL ADDER

```
  Measurements for Transient Analysis

delay_lh_a_sum      =  8.505337e-10 targ=  3.100053e-08 trig=  3.015000e-08
delay_hl_a_sum      =  2.063249e-08 targ=  8.068249e-08 trig=  6.005000e-08
pd_a_sum            =  1.07415e-08
delay_lh_b_sum      =  5.086421e-08 targ=  1.760142e-07 trig=  1.251500e-07
delay_hl_b_sum      =  2.588697e-08 targ=  7.593697e-08 trig=  5.005000e-08
pd_b_sum            =  3.83756e-08
delay_lh_c_sum      =  3.089442e-08 targ=  7.104442e-08 trig=  4.015000e-08
delay_hl_c_sum      =  7.050749e-08 targ=  2.706575e-07 trig=  2.001500e-07
pd_c_sum            =  5.07010e-08
delay_lh_a_carry    =  8.047030e-08 targ=  1.805203e-07 trig=  1.000500e-07
delay_hl_a_carry    =  3.062896e-08 targ=  7.067896e-08 trig=  4.005000e-08
pd_a_carry          =  5.55496e-08
delay_lh_b_carry    =  8.047030e-08 targ=  1.805203e-07 trig=  1.000500e-07
delay_hl_b_carry    =  4.063632e-08 targ=  9.068632e-08 trig=  5.005000e-08
pd_b_carry          =  6.05533e-08
delay_lh_c_carry    =  6.059997e-08 targ=  2.206500e-07 trig=  1.600500e-07
delay_hl_c_carry    =  3.052896e-08 targ=  7.067896e-08 trig=  4.015000e-08
pd_c_carry          =  4.55645e-08
```

PRE-LAYOUT LEAKAGE POWERS

AND GATE

```
VA = 0 VB = 0
Leakage Power = 4.098E-12

VA = 0 VB = 1
Leakage Power = 7.11889E-12

VA = 1 VB = 0
Leakage Power = 6.41764E-12

VA = 1 VB = 1
Leakage Power = 9.72432E-12
```

HALF ADDER

```
VA = 0 VB = 0
Leakage Power = 2.53541E-11

VA = 0 VB = 1
Leakage Power = 3.00668E-11

VA = 1 VB = 0
Leakage Power = 3.0868E-11

VA = 1 VB = 1
Leakage Power = 2.81227E-11
```

FULL ADDER

```
VA = 0 VB = 0 VC = 0
Leakage Power = 2.91769E-11

VA = 0 VB = 0 VC = 1
Leakage Power = 7.19375E-11

VA = 0 VB = 1 VC = 0
Leakage Power = 2.64297E-10

VA = 0 VB = 1 VC = 1
Leakage Power = 2.28614E-11

VA = 1 VB = 0 VC = 0
Leakage Power = 3.04967E-11

VA = 1 VB = 0 VC = 1
Leakage Power = 2.36783E-11

VA = 1 VB = 1 VC = 0
Leakage Power = 2.49239E-11

VA = 1 VB = 1 VC = 1
Leakage Power = 1.61042E-09
```

POST-LAYOUT LEAKAGE POWERS

AND GATE

```
VA = 0 VB = 0
Leakage Power = 9.35119E-12

VA = 0 VB = 1
Leakage Power = 1.33264E-11

VA = 1 VB = 0
Leakage Power = 1.25915E-11

VA = 1 VB = 1
Leakage Power = 2.31311E-11
```

HALF ADDER

```
VA = 0 VB = 0
Leakage Power = 4.12758E-11

VA = 0 VB = 1
Leakage Power = 3.91364E-11

VA = 1 VB = 0
Leakage Power = 4.1341E-11

VA = 1 VB = 1
Leakage Power = 5.57533E-11
```

FULL ADDER

```
VA = 0 VB = 0 VC = 0
Leakage Power = 3.3387E-11

VA = 0 VB = 0 VC = 1
Leakage Power = 3.73508E-11

VA = 0 VB = 1 VC = 0
Leakage Power = 3.91323E-11

VA = 0 VB = 1 VC = 1
Leakage Power = 3.59227E-11

VA = 1 VB = 0 VC = 0
Leakage Power = 3.11265E-11

VA = 1 VB = 0 VC = 1
Leakage Power = 3.96673E-11

VA = 1 VB = 1 VC = 0
Leakage Power = 3.97958E-11

VA = 1 VB = 1 VC = 1
Leakage Power = 4.11446E-11
```

4 BY 4 BIT MULTIPLIER PRE-LAYOUT

NETLIST

```
*** 4 BIT MULTIPLIER

.INCLUDE /home/maggy/VLSI/Final_Project/TSMC_180nm.txt

.INCLUDE /home/maggy/VLSI/Final_Project/AND.sub
.INCLUDE /home/maggy/VLSI/Final_Project/HA.sub
.INCLUDE /home/maggy/VLSI/Final_Project/FA.sub

*PARAMETERS
.PARAM X=0.09u

.PARAM Width_p=10*X
.PARAM Length_p=2*X

.PARAM Width_n=5*X
.PARAM Length_n=2*X

.PARAM supply=1
.PARAM tr=10p

.global gnd vdd

.temp 25

*NET-LIST

* Inputs: A3 A2 A1 A0   B3 B2 B1 B0
* Outputs: C S6 S5 S4 S3 S2 S1 S0
* 16 AND gates
* 4 Half adders
* 8 Full adders
* Inputs nodes to HA  is given as HA_ax HA_bx (x = 1, 2, 3, 4)
* Inputs nodes to FA  is given as FA_ax FA_bx FA_cx (x = 1, 2, 3, 4, 5, 6, 7, 8)

xAnd1   A0 B0 vdd gnd  S0   AND
xAnd2   A0 B1 vdd gnd HA_a1 AND
xAnd3   A1 B0 vdd gnd HA_b1 AND
xAnd4   A0 B2 vdd gnd FA_a1 AND
xAnd5   A2 B0 vdd gnd HA_a2 AND
xAnd6   A1 B1 vdd gnd HA_b2 AND
xAnd7   A1 B2 vdd gnd FA_a3 AND
xAnd8   A0 B3 vdd gnd FA_b2 AND
xAnd9   A3 B0 vdd gnd HA_a4 AND
xAnd10  A2 B1 vdd gnd HA_b4 AND
xAnd11  A2 B2 vdd gnd FA_a7 AND
xAnd12  A3 B1 vdd gnd FA_b7 AND
xAnd13  A1 B3 vdd gnd FA_b4 AND
xAnd14  A2 B3 vdd gnd FA_a8 AND
xAnd15  A3 B2 vdd gnd FA_b8 AND
xAnd16  A3 B3 vdd gnd FA_b6 AND

xHA1 HA_a1 HA_b1 vdd gnd  S1   FA_c1 HA
xHA2 HA_a2 HA_b2 vdd gnd FA_b1 FA_c3 HA
xHA3 HA_a3 HA_b3 vdd gnd  S4   FA_c5 HA
xHA4 HA_a4 HA_b4 vdd gnd FA_b3 FA_c7 HA

xFA1 FA_a1 FA_b1 FA_c1 vdd gnd  S2   FA_c2 FA
xFA2 FA_a2 FA_b2 FA_c2 vdd gnd  S3   HA_a3 FA
xFA3 FA_a3 FA_b3 FA_c3 vdd gnd FA_a2 FA_c4 FA
xFA4 FA_a4 FA_b4 FA_c4 vdd gnd HA_b3 FA_a5 FA
xFA5 FA_a5 FA_b5 FA_c5 vdd gnd  S5   FA_c6 FA
xFA6 FA_a6 FA_b6 FA_c6 vdd gnd  S6     C   FA
xFA7 FA_a7 FA_b7 FA_c7 vdd gnd FA_a4 FA_c8 FA
xFA8 FA_a8 FA_b8 FA_c8 vdd gnd FA_b5 FA_a6 FA

*SOURCE
VDD vdd gnd 'supply'
```

```
*Capacitive LOAD
CL1 S0 gnd 0.01f
CL2 S1 gnd 0.01f
CL3 S2 gnd 0.01f
CL4 S3 gnd 0.01f
CL5 S4 gnd 0.01f
CL6 S5 gnd 0.01f
CL7 S6 gnd 0.01f
CL8 C  gnd 0.01f

*INPUT WAVEFORM
VA0 A0 gnd PWL(0 0 {0+500m} 0 {0+500m+tr} 1 {0+1000m} 1 {0+1000m+tr} 0 {0+1500m} 0 {0+1500m+tr} 1 {0+2000m} 1 {0+2000m+tr} 0 {0+2500m} 0 {0+2500m+tr} 1 {0+3000m} 1 {0+3000m+tr} 0 {0+3500m}
VA1 A1 gnd PWL(0 0 {0+500m} 0 {0+500m+tr} 0 {0+1000m} 0 {0+1000m+tr} 1 {0+1500m} 1 {0+1500m+tr} 1 {0+2000m} 1 {0+2000m+tr} 0 {0+2500m} 0 {0+2500m+tr} 0 {0+3000m} 0 {0+3000m+tr} 1 {0+3500m}
VA2 A2 gnd PWL(0 0 {0+500m} 0 {0+500m+tr} 0 {0+1000m} 0 {0+1000m+tr} 0 {0+1500m} 0 {0+1500m+tr} 0 {0+2000m} 0 {0+2000m+tr} 1 {0+2500m} 1 {0+2500m+tr} 1 {0+3000m} 1 {0+3000m+tr} 1 {0+3500m}
VA3 A3 gnd PWL(0 0 {0+500m} 0 {0+500m+tr} 0 {0+1000m} 0 {0+1000m+tr} 0 {0+1500m} 0 {0+1500m+tr} 0 {0+2000m} 0 {0+2000m+tr} 1 {0+2500m} 1 {0+2500m+tr} 1 {0+3000m} 1 {0+3000m+tr} 1 {0+3500m}

VB0 B0 gnd PWL(0 0 {0+500m} 0 {0+500m+tr} 1 {0+1000m} 1 {0+1000m+tr} 0 {0+1500m} 0 {0+1500m+tr} 1 {0+2000m} 1 {0+2000m+tr} 0 {0+2500m} 0 {0+2500m+tr} 1 {0+3000m} 1 {0+3000m+tr} 0 {0+3500m}
VB1 B1 gnd PWL(0 0 {0+500m} 0 {0+500m+tr} 0 {0+1000m} 0 {0+1000m+tr} 1 {0+1500m} 1 {0+1500m+tr} 1 {0+2000m} 1 {0+2000m+tr} 0 {0+2500m} 0 {0+2500m+tr} 0 {0+3000m} 0 {0+3000m+tr} 1 {0+3500m}
VB2 B2 gnd PWL(0 0 {0+500m} 0 {0+500m+tr} 0 {0+1000m} 0 {0+1000m+tr} 0 {0+1500m} 0 {0+1500m+tr} 0 {0+2000m} 0 {0+2000m+tr} 1 {0+2500m} 1 {0+2500m+tr} 1 {0+3000m} 1 {0+3000m+tr} 1 {0+3500m}
VB3 B3 gnd PWL(0 0 {0+500m} 0 {0+500m+tr} 0 {0+1000m} 0 {0+1000m+tr} 0 {0+1500m} 0 {0+1500m+tr} 0 {0+2000m} 0 {0+2000m+tr} 1 {0+2500m} 1 {0+2500m+tr} 1 {0+3000m} 1 {0+3000m+tr} 1 {0+3500m}


*ANALYSIS
.TRAN 0.1m {10000m}
```

```
*Smallest Path
* A0
.measure tran delay_LH_A0_S0
+ TRIG v(A0) val = 0.5 rise = 1
+ TARG v(S0) val = 0.5 rise = 1
.measure tran delay_HL_A0_S0
+ TRIG v(A0) val = 0.5 fall = 1
+ TARG v(S0) val = 0.5 fall = 1
.measure tran pd_A0_S0
+param='(delay_LH_A0_S0+delay_HL_A0_S0)/2' goal=0

*B0
.measure tran delay_LH_B0_S0
+ TRIG v(B0) val = 0.5 rise = 1
+ TARG v(S0) val = 0.5 rise = 1
.measure tran delay_HL_B0_S0
+ TRIG v(B0) val = 0.5 fall = 1
+ TARG v(S0) val = 0.5 fall = 1
.measure tran pd_B0_S0
+param='(delay_LH_B0_S0+delay_HL_B0_S0)/2' goal=0

************************************************************

*Longest path
* A1
.measure tran delay_LH_A1_S6
+ TRIG v(A1) val = 0.5 fall = 1
+ TARG v(S6) val = 0.5 rise = 1
.measure tran delay_HL_A1_S6
+ TRIG v(A1) val = 0.5 fall = 2
+ TARG v(S6) val = 0.5 fall = 2
.measure tran pd_A1_S6
+param='(delay_LH_A1_S6+delay_HL_A1_S6)/2' goal=0

.measure tran delay_LH_A1_C
+ TRIG v(A1) val = 0.5 fall = 1
+ TARG v(C) val = 0.5 rise = 1
.measure tran delay_HL_A1_C
+ TRIG v(A1) val = 0.5 rise = 5
+ TARG v(C) val = 0.5 fall = 3
.measure tran pd_A1_C
+param='(delay_LH_A1_C+delay_HL_A1_C)/2' goal=0
```

```
*B0
.measure tran delay_LH_B0_S6
+ TRIG v(B0) val = 0.5 fall = 3
+ TARG v(S6) val = 0.5 rise = 2
.measure tran delay_HL_B0_S6
+ TRIG v(B0) val = 0.5 rise = 6
+ TARG v(S6) val = 0.5 fall = 3
.measure tran pd_B0_S6
+param='(delay_LH_B0_S6+delay_HL_B0_S6)/2' goal=0

.measure tran delay_LH_B0_C
+ TRIG v(B0) val = 0.5 fall = 2
+ TARG v(C) val = 0.5 rise = 1
.measure tran delay_HL_B0_C
+ TRIG v(B0) val = 0.5 rise = 8
+ TARG v(C) val = 0.5 fall = 3
.measure tran pd_B0_C
+param='(delay_LH_B0_C+delay_HL_B0_C)/2' goal=0

*B1
.measure tran delay_LH_B1_S6
+ TRIG v(B1) val = 0.5 fall = 1
+ TARG v(S6) val = 0.5 rise = 1
.measure tran delay_HL_B1_S6
+ TRIG v(B1) val = 0.5 rise = 4
+ TARG v(S6) val = 0.5 fall = 6
.measure tran pd_B1_S6
+param='(delay_LH_B1_S6+delay_HL_B1_S6)/2' goal=0

.measure tran delay_LH_B1_C
+ TRIG v(B1) val = 0.5 fall = 1
+ TARG v(C) val = 0.5 rise = 1
.measure tran delay_HL_B1_C
+ TRIG v(B1) val = 0.5 fall = 4
+ TARG v(C) val = 0.5 fall = 3
.measure tran pd_B1_C
+param='(delay_LH_B1_C+delay_HL_B1_C)/2' goal=0
```

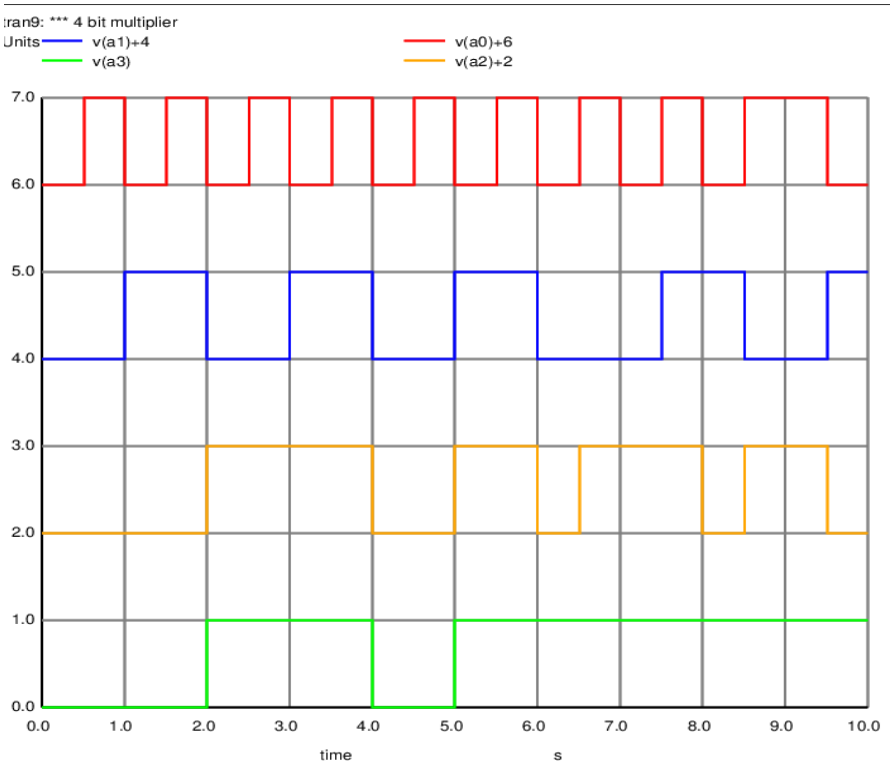
```
.CONTROL
set hcopypscolor = 1
set color0=white
set color1=black

run
hardcopy 4_Bit_Multiplier_A.eps V(A0)+6 V(A1)+4 V(A2)+2 V(A3)
hardcopy 4_Bit_Multiplier_B.eps V(B0)+6 V(B1)+4 V(B2)+2 V(B3)
hardcopy 4_Bit_Multiplier_P.eps V(C)+16 V(S6)+14 V(S5)+12 V(S4)+10 V(S3)+8 V(S2)+6 V(S1)+4 V(S0)+2

.endc
```

- delay calculations:
  - delay_LH_AX_SY: Delay between input signal given to node AX (X = 0,1, 2, 3) and output signal estimated at node SY (Y = 0, 1, 2, 3, 4, 5, 6, 7) when the output signal transitions low to high.
  - delay_HL_AX_SY: Delay between input signal given to node AX (X = 0,1, 2, 3) and output signal estimated at node SY (Y = 0, 1, 2, 3, 4, 5, 6, 7) when the output signal transitions from high to low.
  - delay_AX_SY: The propagation delay of VinAX given by (delay_LH_AX_SY + delay_HL_AX_SY)/2

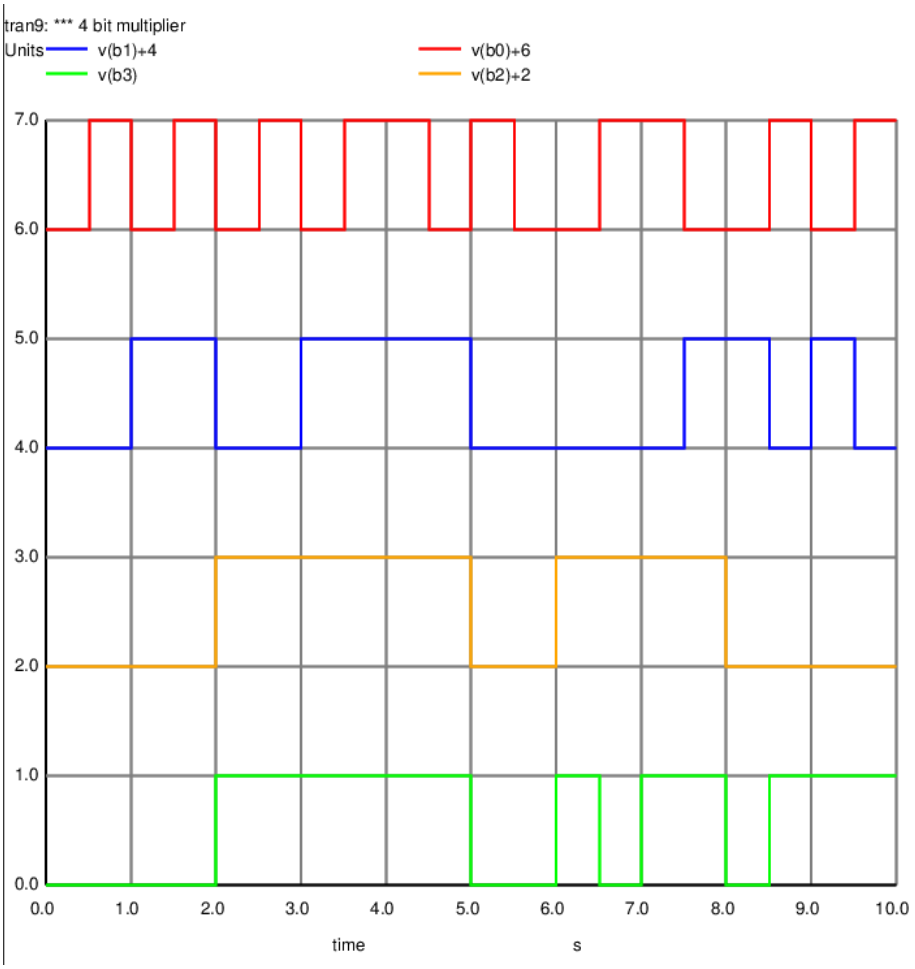
  - delay_LH_BX_SY: Delay between input signal given to node BX (X = 0,1, 2, 3) and output signal estimated at node SY (Y = 0, 1, 2, 3, 4, 5, 6, 7) when the output signal transitions from low to high.
  - delay_HL_BX_Y: Delay between input signal given to node BX (X = 0,1, 2, 3) and output signal estimated at node SY (Y = 0, 1, 2, 3, 4, 5, 6, 7) when the output signal transitions from high to low.
  - delay_BX_SY: The propagation delay of VinBX given by (delay_LH_BX_SY + delay_HL_BX_SY)/2

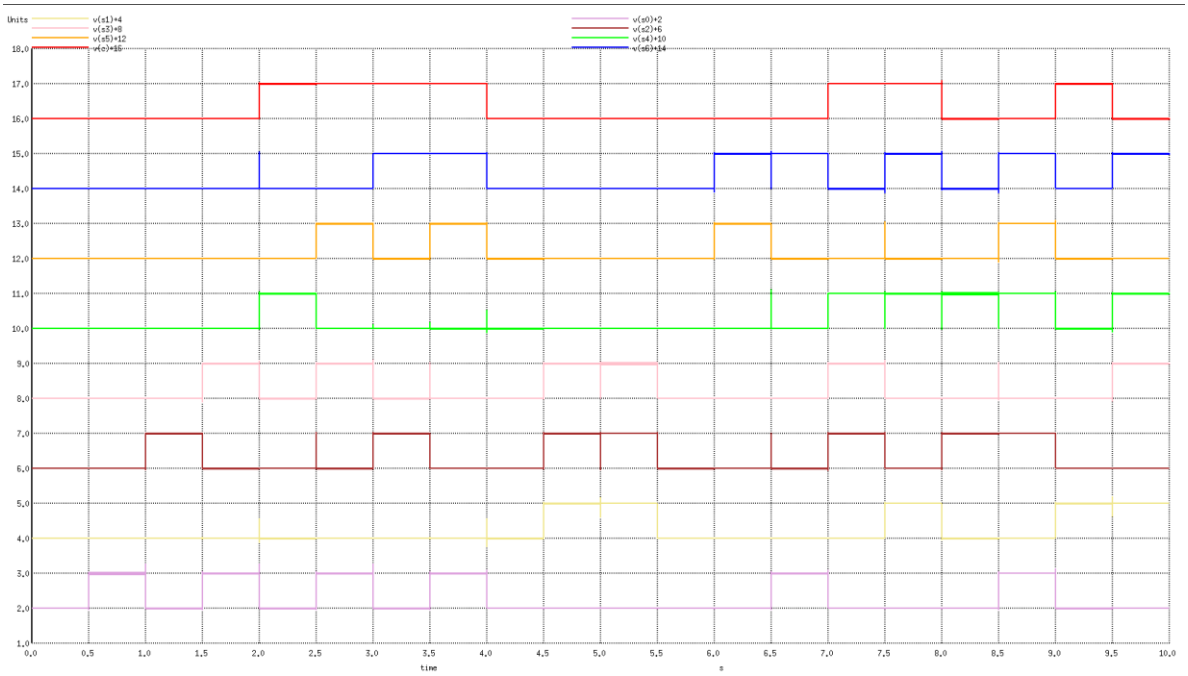
SIMULATION

Input A0, A1, A2, A3

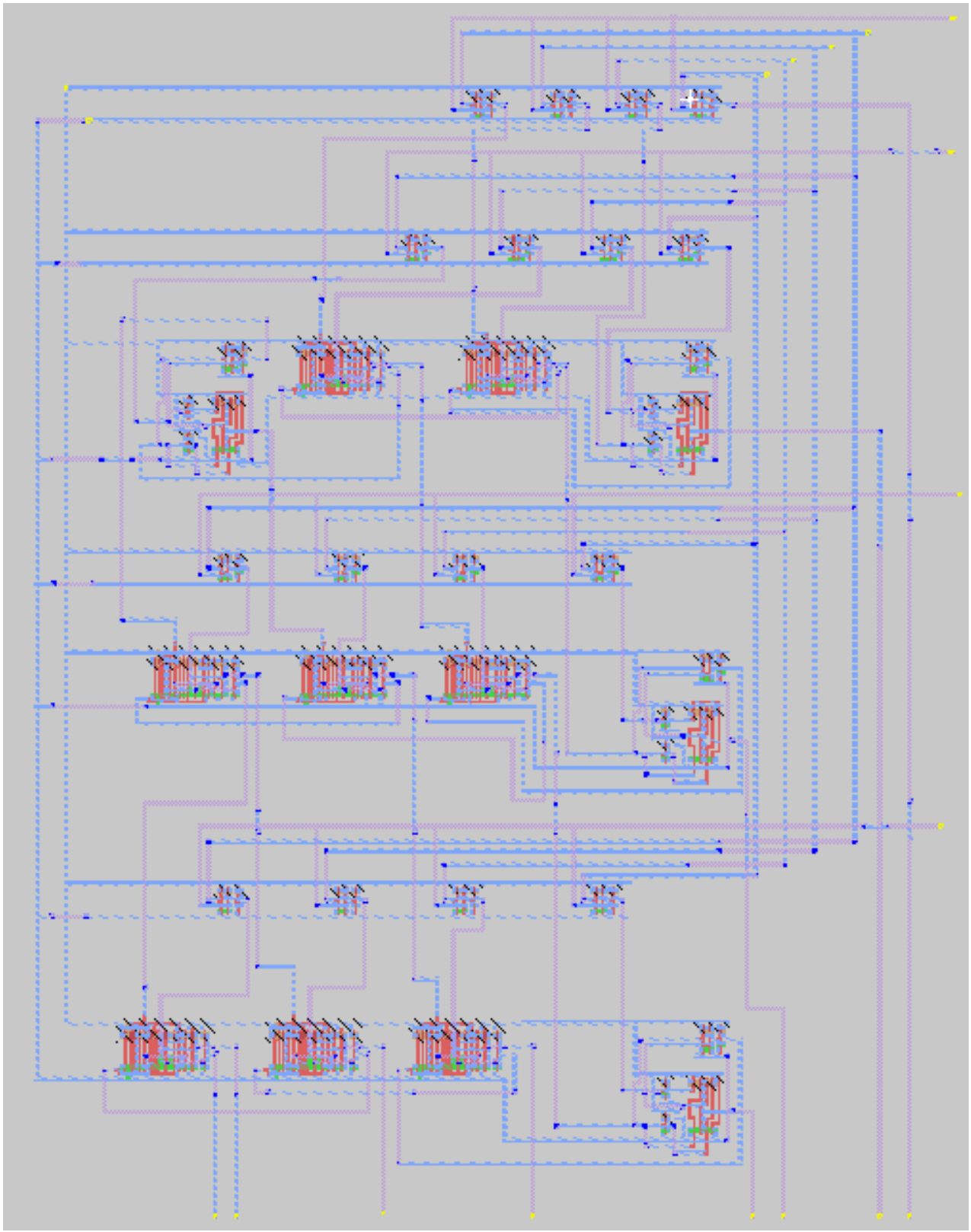


Input B0, B1, B2, B3





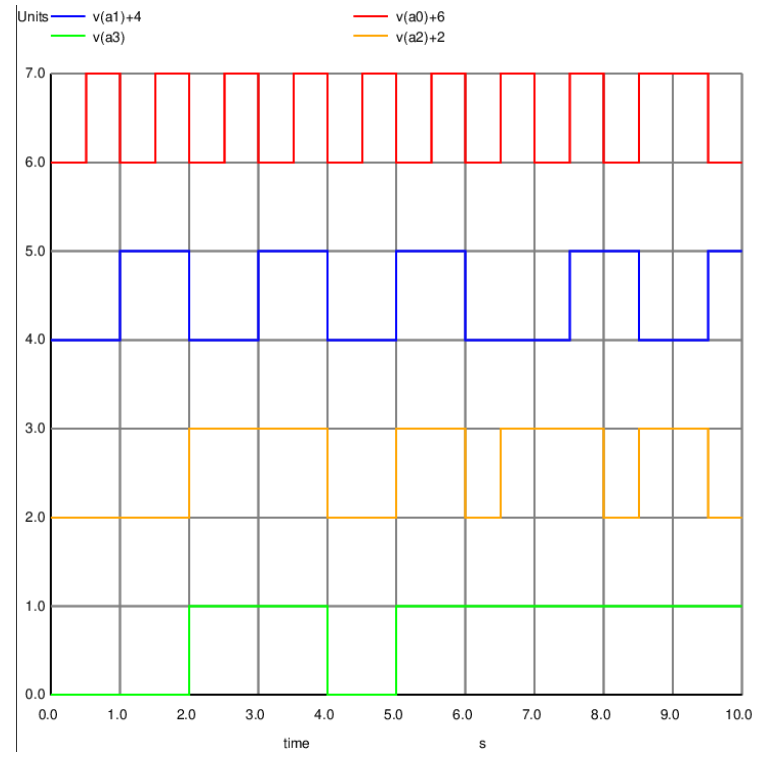
4 BY 4 BIT MULTIPLIER MAGIC LAYOUT



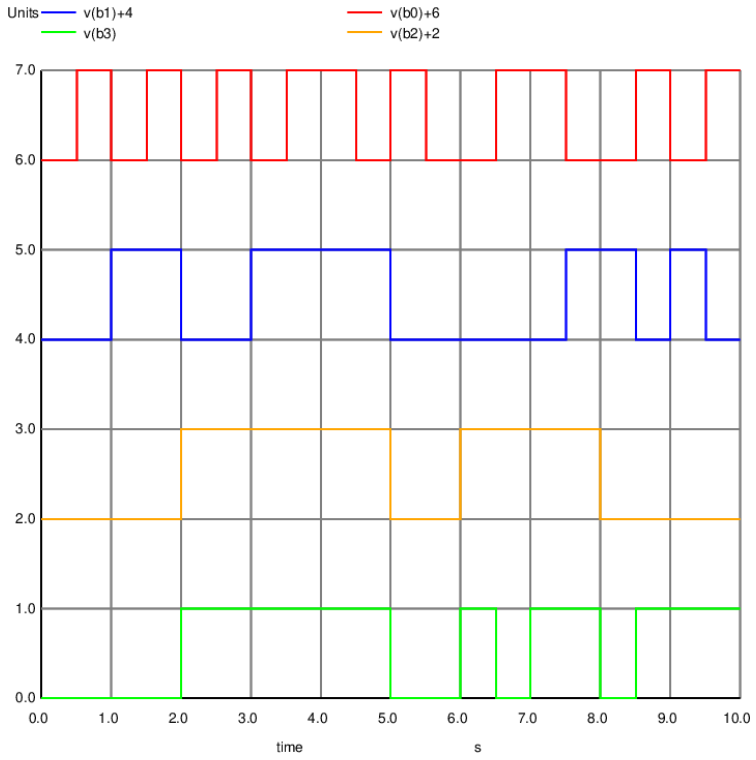
4 BY 4 BIT MULTIPLIER POST-LAYOUT

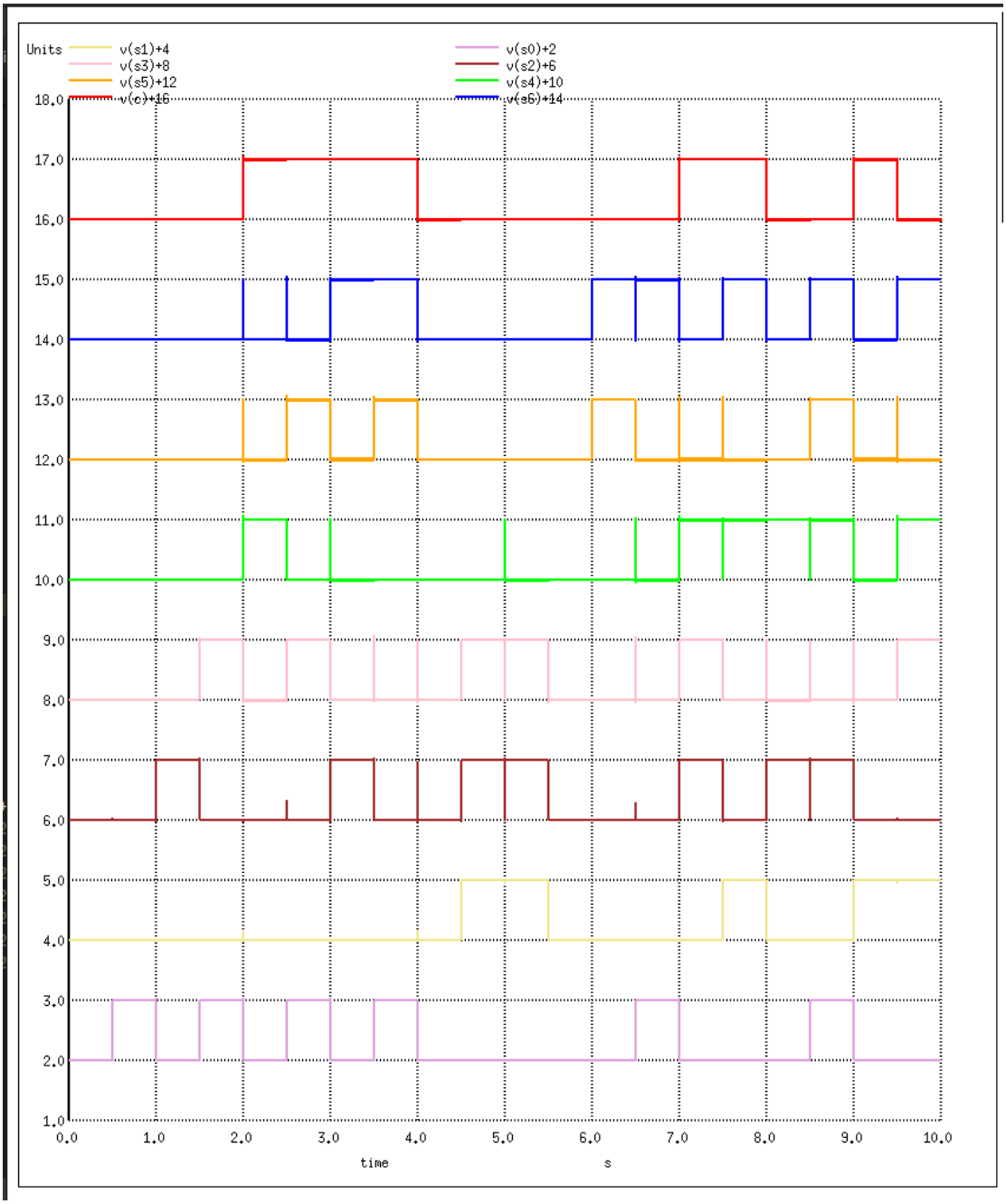
SIMULATIONS

Input A0, A1, A2, A3



Input B0, B1, B2, B3




Units
v(s1)+4
v(s3)+8
v(s5)+12
v(c)+16
v(s0)+2
v(s2)+6
v(s4)+10
v(s6)+14
18.0
17.0
16.0
15.0
14.0
13.0
12.0
11.0
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
time
s

4 BY 4 BIT MULTIPLIER PRE-LAYOUT PROPAGATION DELAYS

The shortest path in the circuit is from input A0 and B0 to output S0. Therefore, the propagation delay between output S0, input A0, and output S0, input B0 will be the least. While the longest path in the circuit is from inputs A1 and B1 to output S6 and C. Therefore, the propagation delay between output S6, C and input A1, and output S6, C and input B1 will be maximum (worst case delay).

```
Delay_LH_A0_S0 = 1.85934E-10
Delay_HL_A0_S0 = 8.61529E-11
Delay_A0_S0 = 1.36043E-10
```

```
Delay_LH_B0_S0 = 1.85934E-10
Delay_HL_B0_S0 = 8.61529E-11
Delay_B0_S0 = 1.36043E-10
```

Worst Case:

```
Delay_LH_A1_S6 = 5.96775E-10
Delay_HL_A1_S6 = 5.68112E-10
Delay_A1_S6 = 5.82444E-10

Delay_LH_A1_C = 1.13651E-09
Delay_HL_A1_C = 1.83821E-09
Delay_A1_C = 1.48736E-09
```

```
Delay_LH_B1_S6 = 5.96775E-10
Delay_HL_B1_S6 = 2.71486E-09
Delay_B1_S6 = 1.65582E-09

Delay_LH_B1_C = 1.13651E-09
Delay_HL_B1_C = 2.39309E-09
Delay_B1_C = 1.7648E-09
```

4 BY 4 BIT MULTIPLIER POST-LAYOUT PROPAGATION DELAYS

```
Delay_LH_A0_S0 = 4.56975E-10
Delay_HL_A0_S0 = 2.82176E-10
Delay_A0_S0 = 3.69575E-10
```

```
Delay_LH_B0_S0 = 4.56975E-10
Delay_HL_B0_S0 = 2.82176E-10
Delay_B0_S0 = 3.69575E-10
```

Worst Case:

```
Delay_LH_A1_S6 = 9.65683E-10
Delay_HL_A1_S6 = 9.97544E-10
Delay_A1_S6 = 9.81614E-10

Delay_LH_A1_C = 2.74904E-09
Delay_HL_A1_C = 2.45715E-09
Delay_A1_C = 2.6031E-09
```

```
Delay_LH_B1_S6 = 9.65683E-10
Delay_HL_B1_S6 = 5.34932E-09
Delay_B1_S6 = 3.1575E-09

Delay_LH_B1_C = 2.74904E-09
Delay_HL_B1_C = 4.92007E-09
Delay_B1_C = 3.83456E-09
```

4 BY 4 BIT MULTIPLIER PRE-LAYOUT LEAKAGE POWERS

Low supply voltage requires the device threshold to be reduced in order to maintain performance. As the device threshold voltage is reduced, it results in an exponential increase of leakage current in the subthreshold region. The leakage power is no longer negligible in such low voltage circuits. Estimates of maximum leakage power can be used in the design of the circuit to minimize the leakage power. The leakage power is dependent on the input vector. For each input assignment the leakage power dissipated by the logic block can be different.

So, the Leakage power is maximum when all inputs are 0 and its minimuum when all inputs are 1.

Leakage Powers for some Input Combinations:

```
VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 0 VB2 = 0 VB3 = 0
Leakage Power = 8.91862E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 0 VB2 = 0 VB3 = 1
Leakage Power = 8.91862E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 0 VB2 = 1 VB3 = 0
Leakage Power = 8.91862E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 0 VB2 = 1 VB3 = 1
Leakage Power = 8.80132E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 1 VB2 = 0 VB3 = 0
Leakage Power = 8.91863E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 1 VB2 = 0 VB3 = 1
Leakage Power = 8.80132E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 1 VB2 = 1 VB3 = 0
Leakage Power = 8.80132E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 1 VB2 = 1 VB3 = 1
Leakage Power = 8.68402E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 1 VB1 = 0 VB2 = 0 VB3 = 0
Leakage Power = 9.03592E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 1 VB1 = 0 VB2 = 0 VB3 = 1
Leakage Power = 8.91862E-06

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 1 VB1 = 0 VB2 = 1 VB3 = 0
Leakage Power = 8.91862E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 0 VB2 = 0 VB3 = 0
Leakage Power = 9.37081E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 0 VB2 = 0 VB3 = 1
Leakage Power = 9.43999E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 0 VB2 = 1 VB3 = 0
Leakage Power = 9.38584E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 0 VB2 = 1 VB3 = 1
Leakage Power = 9.46912E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 1 VB2 = 0 VB3 = 0
Leakage Power = 9.34974E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 1 VB2 = 0 VB3 = 1
Leakage Power = 9.41894E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 1 VB2 = 1 VB3 = 0
Leakage Power = 9.41727E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 1 VB2 = 1 VB3 = 1
Leakage Power = 9.62678E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 1 VB1 = 0 VB2 = 0 VB3 = 0
Leakage Power = 9.48056E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 1 VB1 = 0 VB2 = 0 VB3 = 1
Leakage Power = 9.54974E-06

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 1 VB1 = 0 VB2 = 1 VB3 = 0
Leakage Power = 9.49561E-06
```

The Leakage Power for 256 combinations of input values has been calculated using a python script

```
import os
import sys
import fileinput


def convert_to_binary(N):
    binary = bin(N).replace('0b','')

    A = binary[::-1]

    while len(A) < 8:
        A += '0'

    binary = A[::-1]

    return list(binary)


old_A0 = "VA0 A0 gnd 0"
old_A1 = "VA1 A1 gnd 0"
old_A2 = "VA2 A2 gnd 0"
old_A3 = "VA3 A3 gnd 0"

old_B0 = "VB0 B0 gnd 0"
old_B1 = "VB1 B1 gnd 0"
old_B2 = "VB2 B2 gnd 0"
old_B3 = "VB3 B3 gnd 0"

for num in range(0, 256):
    A = convert_to_binary(num)

    f = open("4_bit_Multiplier_Pre_Layout_leakage.txt", "a")

    file_name = "4_bit_Multiplier_Pre_Layout_leakage.cir"
    File = open(file_name, 'r+')

    new_A0 = "VA0 A0 gnd " + A[0]
    new_A1 = "VA1 A1 gnd " + A[1]
    new_A2 = "VA2 A2 gnd " + A[2]
    new_A3 = "VA3 A3 gnd " + A[3]

    new_B0 = "VB0 B0 gnd " + A[4]
    new_B1 = "VB1 B1 gnd " + A[5]
    new_B2 = "VB2 B2 gnd " + A[6]
    new_B3 = "VB3 B3 gnd " + A[7]
```

```
s = "VA0 = "+ str(A[0]) + " VA1 = "+ str(A[1]) + " VA2 = "+ str(A[2]) + " VA3 = "+ str(A[3]) +"\nVB0 = "+ str(A[4]) + " VB1 = "+ str(A[5]) + " VB2 = "+ str(A[6]) + " VB3 = "+ str(A[7])
f.write(s)
f.close()


for line in fileinput.input(file_name):
    File.write(line.replace(old_A0, new_A0))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_A1, new_A1))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_A2, new_A2))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_A3, new_A3))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_B0, new_B0))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_B1, new_B1))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_B2, new_B2))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_B3, new_B3))
File.close()
```

```
old_A0 = new_A0
old_A1 = new_A1
old_A2 = new_A2
old_A3 = new_A3

old_B0 = new_B0
old_B1 = new_B1
old_B2 = new_B2
old_B3 = new_B3

os.system("ngspice 4_bit_Multiplier_Pre_Layout_leakage.cir")

f = open("4_bit_Multiplier_Pre_Layout_leakage.txt", "a")
f.write("\n")
f.close()
```

Here the python file is used to replace the input voltages 256 times and to write the leakage powers into a text file.

4 BY 4 BIT MULTIPLIER POST-LAYOUT LEAKAGE POWERS

Low supply voltage requires the device threshold to be reduced in order to maintain performance. As the device threshold voltage is reduced, it results in an exponential increase of leakage current in the subthreshold region. The leakage power is no longer negligible in such low voltage circuits. Estimates of maximum leakage power can be used in the design of the circuit to minimize the leakage power. The leakage power is dependent on the input vector. For each input assignment the leakage power dissipated by the logic block can be different.

So, the Leakage power is maximum when all inputs are 0 and its minimuum when all inputs are 1.

Leakage Powers for some Input Combinations:

```
LEAKAGE POWERS FOR ALL INPUT COMBINATIONS

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 0 VB2 = 0 VB3 = 0
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 0 VB2 = 0 VB3 = 1
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 0 VB2 = 1 VB3 = 0
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 0 VB2 = 1 VB3 = 1
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 1 VB2 = 0 VB3 = 0
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 1 VB2 = 0 VB3 = 1
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 1 VB2 = 1 VB3 = 0
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 0 VB1 = 1 VB2 = 1 VB3 = 1
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 1 VB1 = 0 VB2 = 0 VB3 = 0
Leakage Power = 4.69514E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 1 VB1 = 0 VB2 = 0 VB3 = 1
Leakage Power = 4.69515E-05

VA0 = 0 VA1 = 0 VA2 = 0 VA3 = 0
VB0 = 1 VB1 = 0 VB2 = 1 VB3 = 0
Leakage Power = 4.69515E-05
```

```
VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 0 VB2 = 0 VB3 = 0
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 0 VB2 = 0 VB3 = 1
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 0 VB2 = 1 VB3 = 0
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 0 VB2 = 1 VB3 = 1
Leakage Power = 9.34195E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 1 VB2 = 0 VB3 = 0
Leakage Power = 9.34194E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 1 VB2 = 0 VB3 = 1
Leakage Power = 9.34195E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 1 VB2 = 1 VB3 = 0
Leakage Power = 9.34195E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 0 VB1 = 1 VB2 = 1 VB3 = 1
Leakage Power = 9.34195E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 1 VB1 = 0 VB2 = 0 VB3 = 0
Leakage Power = 4.69515E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 1 VB1 = 0 VB2 = 0 VB3 = 1
Leakage Power = 4.69515E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 1 VB1 = 0 VB2 = 1 VB3 = 0
Leakage Power = 4.69515E-05

VA0 = 0 VA1 = 0 VA2 = 1 VA3 = 1
VB0 = 1 VB1 = 0 VB2 = 1 VB3 = 1
Leakage Power = 4.69515E-05
```

The Leakage Power for 256 combinations of input values has been calculated using a python script

```python
import os
import sys
import fileinput


def convert_to_binary(N):
    binary = bin(N).replace('0b','')

    A = binary[::-1]

    while len(A) < 8:
        A += '0'

    binary = A[::-1]

    return list(binary)


old_A0 = "VA0 A0 gnd 0"
old_A1 = "VA1 A1 gnd 0"
old_A2 = "VA2 A2 gnd 0"
old_A3 = "VA3 A3 gnd 0"

old_B0 = "VB0 B0 gnd 0"
old_B1 = "VB1 B1 gnd 0"
old_B2 = "VB2 B2 gnd 0"
old_B3 = "VB3 B3 gnd 0"

for num in range(0, 256):
    A = convert_to_binary(num)

    f = open("4_bit_Multiplier_Post_Layout_leakage.txt", "a")

    file_name = "4_bit_Multiplier_Post_Layout_leakage.spice"
    File = open(file_name, 'r+')

    new_A0 = "VA0 A0 gnd " + A[0]
    new_A1 = "VA1 A1 gnd " + A[1]
    new_A2 = "VA2 A2 gnd " + A[2]
    new_A3 = "VA3 A3 gnd " + A[3]

    new_B0 = "VB0 B0 gnd " + A[4]
    new_B1 = "VB1 B1 gnd " + A[5]
    new_B2 = "VB2 B2 gnd " + A[6]
    new_B3 = "VB3 B3 gnd " + A[7]
```

```
s = "VA0 = "+ str(A[0]) + " VA1 = "+ str(A[1]) + " VA2 = "+ str(A[2]) + " VA3 = "+ str(A[3]) +"\nVB0 = "+ str(A[4]) + " VB1 = "+ str(A[5]) + " VB2 = "+ str(A[6]) + " VB3 = "+ str(A[7])
f.write(s)
f.close()


for line in fileinput.input(file_name):
    File.write(line.replace(old_A0, new_A0))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_A1, new_A1))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_A2, new_A2))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_A3, new_A3))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_B0, new_B0))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_B1, new_B1))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_B2, new_B2))
File.close()

File = open(file_name, 'r+')
for line in fileinput.input(file_name):
    File.write(line.replace(old_B3, new_B3))
File.close()
```

```
old_A0 = new_A0
old_A1 = new_A1
old_A2 = new_A2
old_A3 = new_A3

old_B0 = new_B0
old_B1 = new_B1
old_B2 = new_B2
old_B3 = new_B3

os.system("ngspice 4_bit_Multiplier_Post_Layout_leakage.spice")

f = open("4_bit_Multiplier_Post_Layout_leakage.txt", "a")
f.write("\n")
f.close()
```

Here the python file is used to replace the input voltages 256 times and to write the leakage powers into a text file.

OBSERVATIONS AND CONCLUSIONS

PROPAGATION DELAYS

- The propagation delay of a logic gate is the difference in time (calculated at 50% of input-output transition), when output switches, after application of input.
- Rise time (t_r) is the time, during transition, when output switches from 10% to 90% of the maximum value.
- Fall time (t_f) is the time, during transition, when output switches from 90% to 10% of the maximum value.
- The propagation delay from high to low (t_{pHL}) is the delay when output switches from high-to-low, after input switches from low to high. The delay is usually calculated at the 50% point of input-output switching
- Delay in a gate can be simplified as the amount of time it takes to discharge the load capacitance that the gate or fet is driving.

$$I = q/t = C*V/t$$

$$t = C*V/I$$

1) to the first order, delay (time) is inversely proportional to drive current. So, increasing the drive current will reduce the delay.

2) Increasing the MOS width will increase its drive current.

Therefore, increasing MOS width will increase its drive current which will reduce the discharge time of the load (reduce delay).

LEAKAGE POWERS

The power consumed by the sub-threshold currents and by reverse-biased diodes in a CMOS transistor is considered leakage power. In CMOS circuits, very small current flows even with zero gate to source voltage (Vgs) and is termed as leakage current. The leakage power of a CMOS logic gate does not depend on input transition or load capacitance and hence it remains constant for a logic cell. Leakage power is primarily the result of unwanted subthreshold current in the transistor channel when the transistor is turned off.

DIFFERENCE BETWEEN PRE-LAYOUT AND POST-LAYOUT SIMULATIONS

The main purpose of pre-layout simulation is to develop design constraints, while post-layout simulation's main goal is to verify compliance with those constraints.

The pre-layout simulation serves to prove board-level design concepts. A pre-layout simulation shows if a configuration allows for a signal at the receiver to meet the design specifications. Also, with pre-layout simulation, the engineer can understand the limitations of the design and create a plan to successfully implement the design.

Post-layout simulation, on the other hand, is mainly used to verify the completed design. It verifies all design constraints after their creation. Post-layout simulation also comes in handy when comparing simulation versus measurements. This is important to ensure that the constraints created by pre-layout simulation are based on sound modeling of the PCB.

PRE-LAYOUT AND POST-LAYOUT DELAYS

We observe that the post-Layout delays are more than the Pre-Layout delays. This is due to the different capacitances and the sizing of the MOSFETs in magic.

The Post layout simulations take into consideration the parasitic capacitances of the MOSFETs, while the Pre layout simulations are done just to verify whether the outputs meet the design specifications. Therefore, the effective capacitance increases. As output capacitance increases, the total time required to charge this capacitance increases, and so the delay increases.

The sizing of the MOSFET in post-layout simulation is based on proven design parameters which is why the sizing in the pre-layout and post-layout simulations could be different, leading to different delay values.

PRE-LAYOUT AND POST-LAYOUT LEAKAGE POWERS

We observe that the post-Layout leakage powers are more than the Pre-Layout powers.

Leakage power is primarily the result of unwanted subthreshold current in the transistor channel when the transistor is turned off. In general, subthreshold leakage currents are exponentially dependent on temperature, process variations and threshold voltage (Vt). Substrate source voltage (Vbs) will vary the threshold voltage of the device.

VERILOG TO CHECK FUNCTIONALITY OF A 4 BY 4 BIT MULTIPLIER

Verilog is a Hardware Description Language (HDL) which is used to model electronic systems. It is commonly used for design and verification of digital circuits.

The Verilog code was written to verify the functionality of a 4-bit multiplier.

We observe from the plots that the module has a functionality of a 4 by 4 multiplier.

CODE

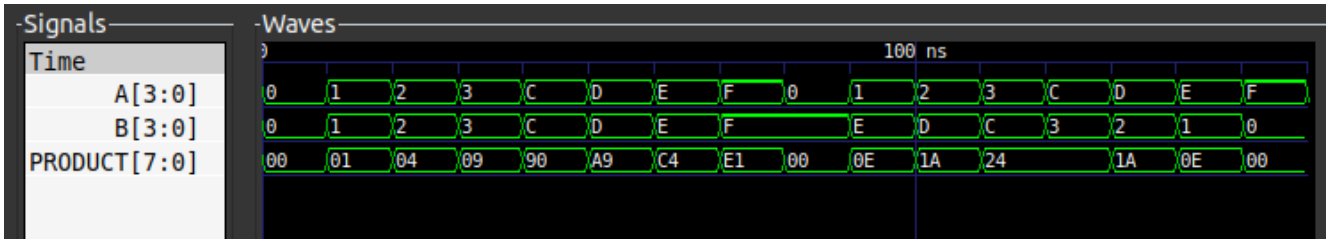
```
// verilog code for a 4 BIT MULTIPLIER using HA and FA

module HA(A, B, SUM, CARRY);

input A, B;
output SUM, CARRY;

assign SUM=A^B;
assign CARRY=(A&B);

endmodule


module FA(A, B, Cin, SUM, CARRY);

input A, B, Cin;
output SUM, CARRY;

assign SUM=(A^B^Cin);
assign CARRY=((A&B)|(A&Cin)|(B&Cin));

endmodule


module Four_Bit_Multiplier(A0, A1, A2, A3, B0, B1, B2, B3, PRODUCT0, PRODUCT1, PRODUCT2, PRODUCT3, PRODUCT4, PRODUCT5, PRODUCT6, PRODUCT7);


input A0, A1, A2, A3;
input B0, B1, B2, B3;
output PRODUCT0, PRODUCT1, PRODUCT2, PRODUCT3, PRODUCT4, PRODUCT5, PRODUCT6, PRODUCT7;

wire HA_a1, HA_b1, HA_a2, HA_b2, HA_a3, HA_b3, HA_a4, HA_b4;
wire FA_a1, FA_b1, FA_c1, FA_a2, FA_b2, FA_c2, FA_a3, FA_b3, FA_c3, FA_a4, FA_b4, FA_c4, FA_a5, FA_b5, FA_c5, FA_a6, FA_b6, FA_c6, FA_a7, FA_b7, FA_c7, FA_a8, FA_b8, FA_c8;

and(PRODUCT0, A0, B0);
and(HA_a1, A0, B1);
and(HA_b1, A1, B0);
and(FA_a1, A0, B2);
and(HA_a2, A2, B0);
and(HA_b2, A1, B1);
and(FA_a3, A1, B2);
and(FA_b2, A0, B3);
and(HA_a4, A3, B0);
and(HA_b4, A2, B1);
and(FA_a7, A2, B2);
and(FA_b7, A3, B1);
and(FA_b4, A1, B3);
and(FA_a8, A2, B3);
and(FA_b8, A3, B2);
and(FA_b6, A3, B3);

HA HA1(HA_a1, HA_b1, PRODUCT1, FA_c1);
HA HA2(HA_a2, HA_b2, FA_b1, FA_c3);
HA HA3(HA_a3, HA_b3, PRODUCT4, FA_c5);
HA HA4(HA_a4, HA_b4, FA_b3, FA_c7);

FA FA1(FA_a1, FA_b1, FA_c1, PRODUCT2, FA_c2);
FA FA2(FA_a2, FA_b2, FA_c2, PRODUCT3, HA_a3);
FA FA3(FA_a3, FA_b3, FA_c3, FA_a2, FA_c4);
FA FA4(FA_a4, FA_b4, FA_c4, HA_b3, FA_a5);
FA FA5(FA_a5, FA_b5, FA_c5, PRODUCT5, FA_c6);
FA FA6(FA_a6, FA_b6, FA_c6, PRODUCT6, PRODUCT7);
FA FA7(FA_a7, FA_b7, FA_c7, FA_a4, FA_c8);
FA FA8(FA_a8, FA_b8, FA_c8, FA_b5, FA_a6);


endmodule
```

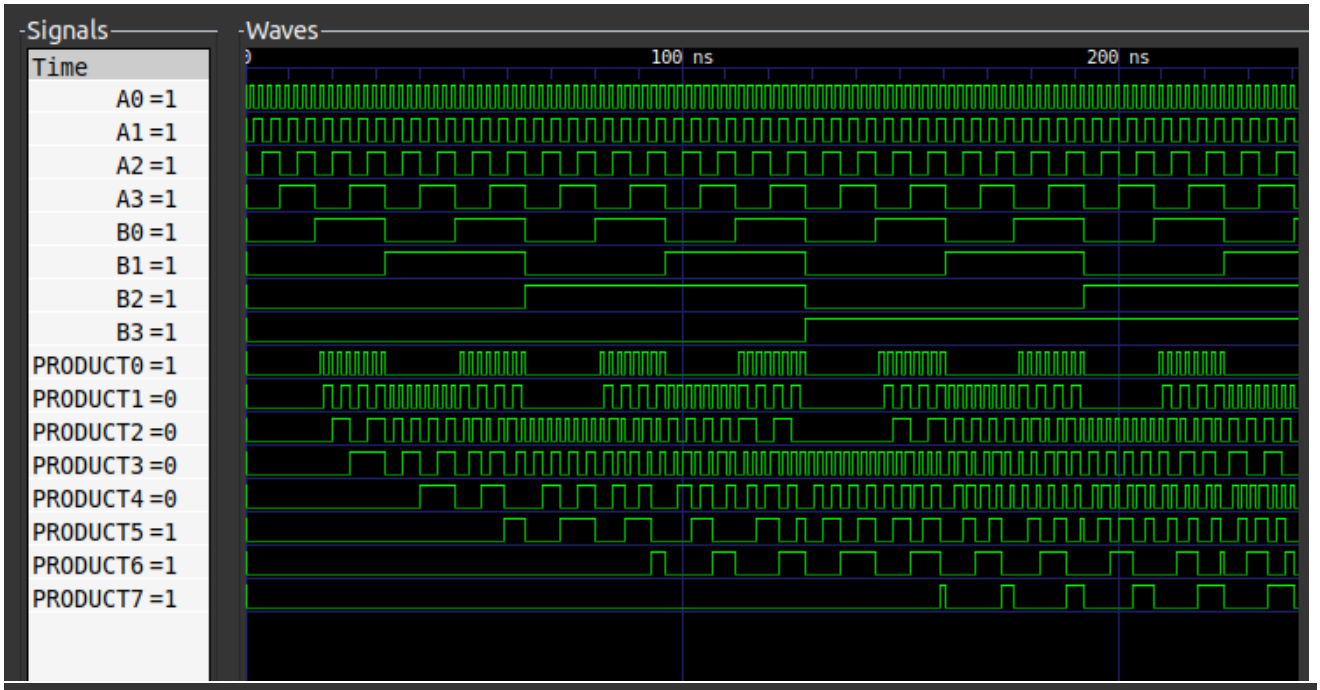
TEST BENCH

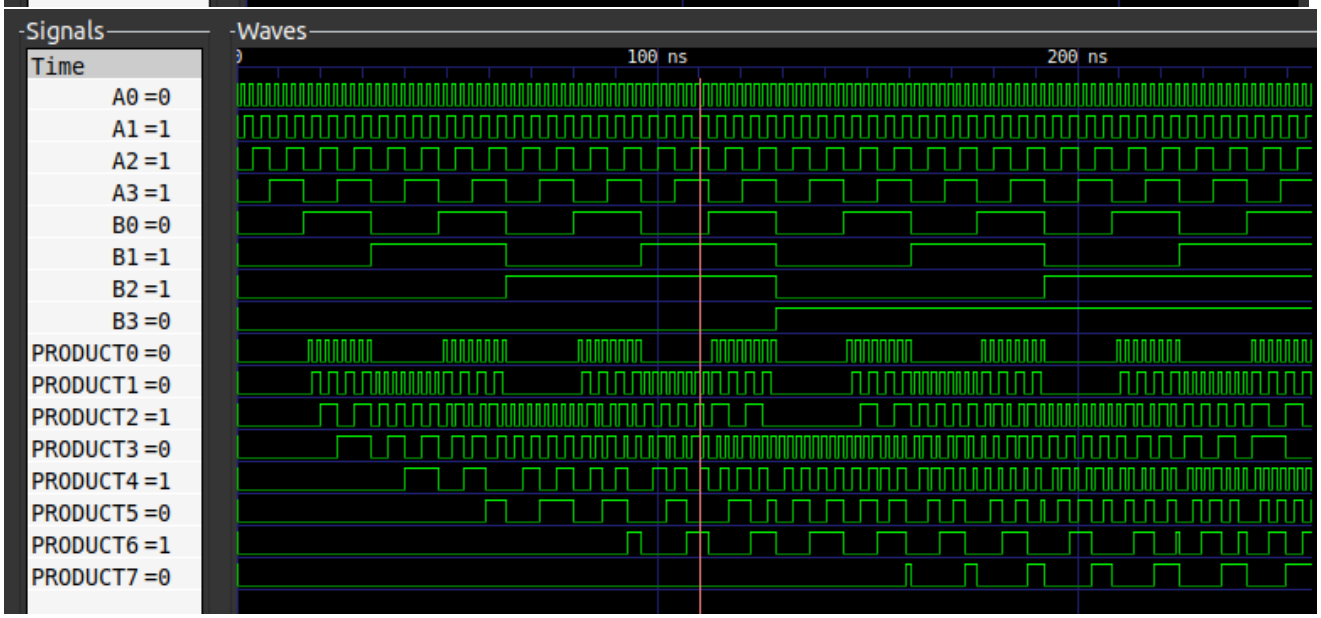
```
//Test Bench for checking the functionality of 4_bit_Multiplier.v

`timescale 1ns/10ps

module test_bench();

reg A0;
reg A1;
reg A2;
reg A3;
reg B0;
reg B1;
reg B2;
reg B3;
wire PRODUCT0;
wire PRODUCT1;
wire PRODUCT2;
wire PRODUCT3;
wire PRODUCT4;
wire PRODUCT5;
wire PRODUCT6;
wire PRODUCT7;


Four_Bit_Multiplier UUT (A0, A1, A2, A3, B0, B1, B2, B3, PRODUCT0, PRODUCT1, PRODUCT2, PRODUCT3, PRODUCT4, PRODUCT5, PRODUCT6, PRODUCT7);

integer i;
initial
    begin

        $dumpfile("test_bench.vcd");
        $dumpvars(0, test_bench);

        A0=0;
        A1=0;
        A2=0;
        A3=0;

        B0=0;
        B1=0;
        B2=0;
        B3=0;

    end
```

```
initial
    begin

        for(i = 1; i < 256; i = i+1)
        begin

            #1
            {B3, B2, B1, B0, A3, A2, A1, A0} = i;

        end


    end

always @(*)
$display("Time = %0t \n A0 = %b \t A1 = %b \t A2 = %b \t A3 = %b \n B0 = %b \t B1 = %b \t B2 = %b \t B3 = %b", $time, A0,A1,A2,A3,B0,B1,B2,B3);


endmodule
```

OUTPUT




Signals
Waves
Time
A0 =1
A1 =1
A2 =1
A3 =1
B0 =1
B1 =1
B2 =1
B3 =1
PRODUCT0 =1
PRODUCT1 =0
PRODUCT2 =0
PRODUCT3 =0
PRODUCT4 =0
PRODUCT5 =1
PRODUCT6 =1
PRODUCT7 =1
100 ns
200 ns


Signals
Waves
Time
A0 =0
A1 =1
A2 =1
A3 =1
B0 =0
B1 =1
B2 =1
B3 =0
PRODUCT0 =0
PRODUCT1 =0
PRODUCT2 =1
PRODUCT3 =0
PRODUCT4 =1
PRODUCT5 =0
PRODUCT6 =1
PRODUCT7 =0
100 ns
200 ns