COE 360 : Principles of VLSI Design

Semester 032

16-BIT PIPELINED ADDER

PHASE II

Prepared By

Mohammad Banaemah	995976
Mohammad Isa Sunaryo	987861

Section # 03

Supervised By

Dr.Muhammad El-Rabaa

Table of Contents

Table of Figures

Introduction

In the 2nd phase of designing a *16-bit VLSI Pipelined adder*, transistor level design using WINSPICE is completed considering the following:

- The technology used for the design.
- The transistor level schematic of the design.
- Measuring the overall delay from the input to the output.
- Identifying the different pull-up and pull down paths.
- Determining the correct sizing (widths) of each transistor.
- Design Testing and Verification

This report discusses all of the issues mentioned above and provides the design with some schematics. First, it starts with the transistor level schematic followed by measuring the overall delay in the longest (critical) path. Then, the sizing of the width of each transistor is determined using Simulations Program with Integrated Circuits Emphases (WINSPICE). Finally, some of the tests for the design are discussed. The concept behind this 16-bit pipelined adder is to divide the operations into stages of 4-bit Carry Ripple Adder.

Design Requirements

The design of the 16-bit VLSI Pipelined Adder follows the following specifications:

1. Use of AMI's 0.5 µm technology.
2. All gates should have symmetrical noise margins.
3. Minimum area to operate at 800 MHz frequency with a capacitance load of pF per output.

Design Alternatives

There are several alternatives in designing the 16-bit VLSI Pipelined Adder:
First design is to divide the 16-bit adder into blocks of 1-bit full adders. This will consequently increase the delay in the critical path for the carry-bit which has the following logic level gates:

i) 16 blocks of 1-bit Full adder
ii) 17 blocks of 1-bit Flip-Flop

The second design alternative is to divide the 16-bit adder into 4 blocks of 4-bit full adder. This will reduce the delay. This delay in the critical path for the carry-bit which has the following blocks:

i) 4 blocks of 4-bit Full adder
ii) 5 blocks of 4-bit Flip-Flop

Hence, the second design is selected to meet the required delay of 1250 ps period. Calculations are performed based on the following equations:

1) $T_d = \frac{1}{2} C_{load} \, [R_{eq_{pd}} + R_{eq_{pu}}]$

Since equal noise margins must be satisfied for each gate, this equation is used:

$T_d = C_{load} \times R_{eq_{pd}}$

Because $R_{eq_{pd}} = R_{eq_{up}}$

2) $R_{eq_{pd}} = \dfrac{1}{\mu_n C_{ox}\left(\dfrac{W_n}{L}\right)(V_{DD} - V_{tn})}$

3) $Cin = Cox * L * (W_p + W_N)$

Design Description

In designing the 16-bit VLSI Pipelined Adder, it is strongly recommended to divide the full design into designing small blocks for simplicity as follows:

1) Inverter
2) NOR Gate
3) Transmission Gate
4) Master Latch
5) Slave Latch
6) Sum-circuit of 1-bit Full Adder
7) Carry-circuit of of 1-bit Full Adder
8) 1-bit Full Adder
9) 1-bit Flip-Flop
10) 4-bit Full Adder
11) 4-bit Flip-Flop
12) Inverter Buffer Chain

Transistor-level Implementation

For each block, we will show the logical-level block diagrams followed by logic equations. Then, we will show the timing diagrams obtained by WinSpice to determine the time delay for each block. After that, the implementations of the circuit in transistor-level schematic diagrams are shown with the sizing of each transistor written within the block.

1) Inverter

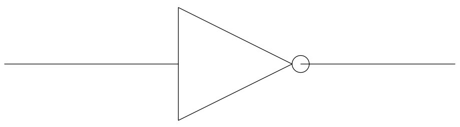


Figure 1 - Logical Diagram of Inverter

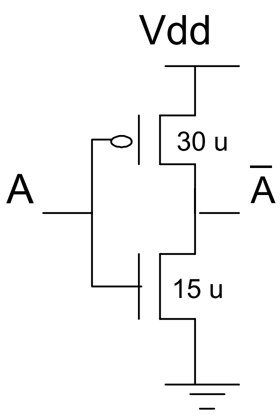


Figure 2 - Transistor-level Diagram of Inverter

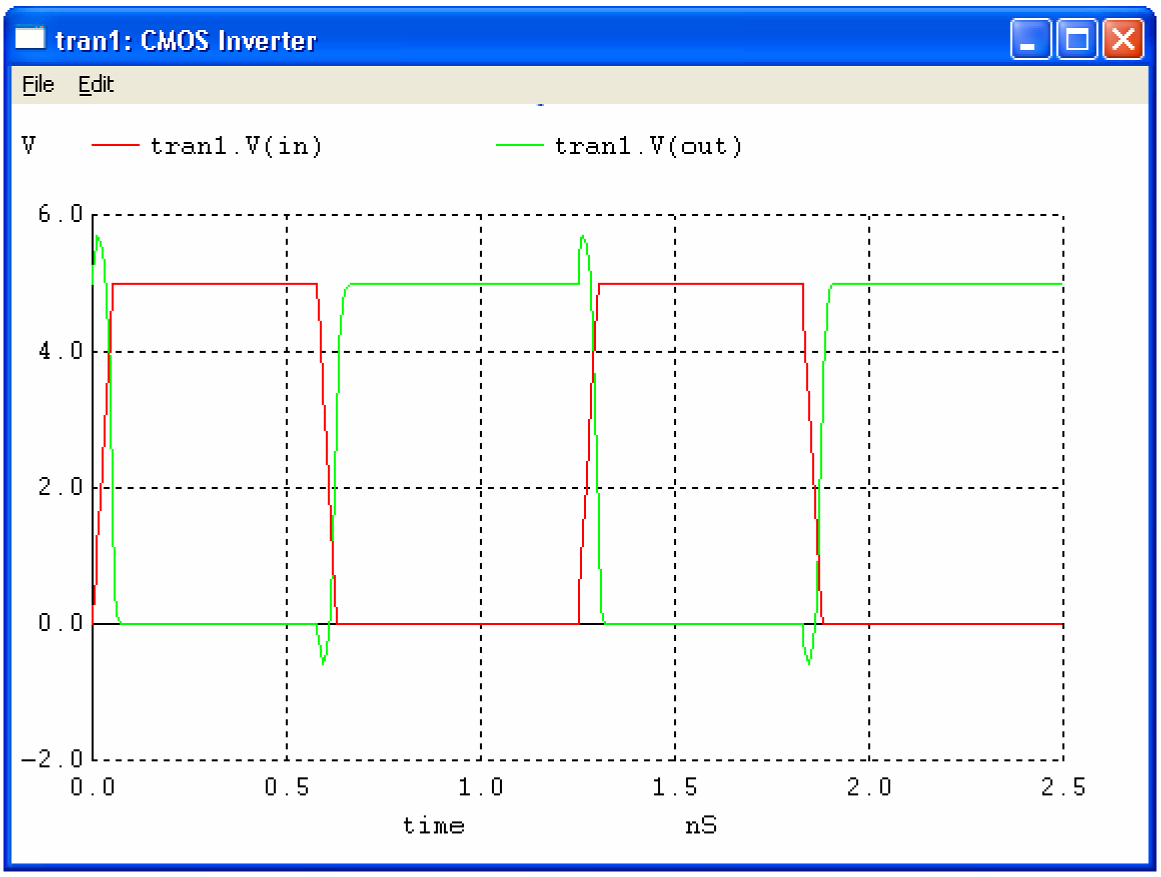


Figure 3 - Simulation of CMOS Inverter

The maximum delay required for this inverter equals to 63ps taken at 2.5 V.

2) NOR Gate

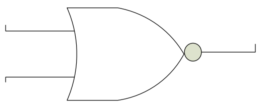


Figure 4 - Logical Diagram of NOR Gate

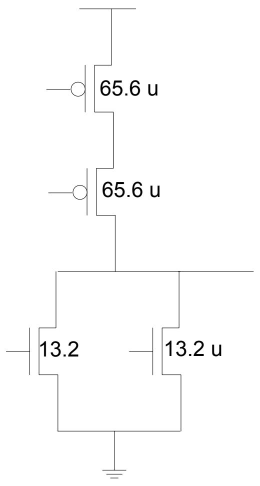


Figure 5 - Transistor-Level Diagram of NOR Gate

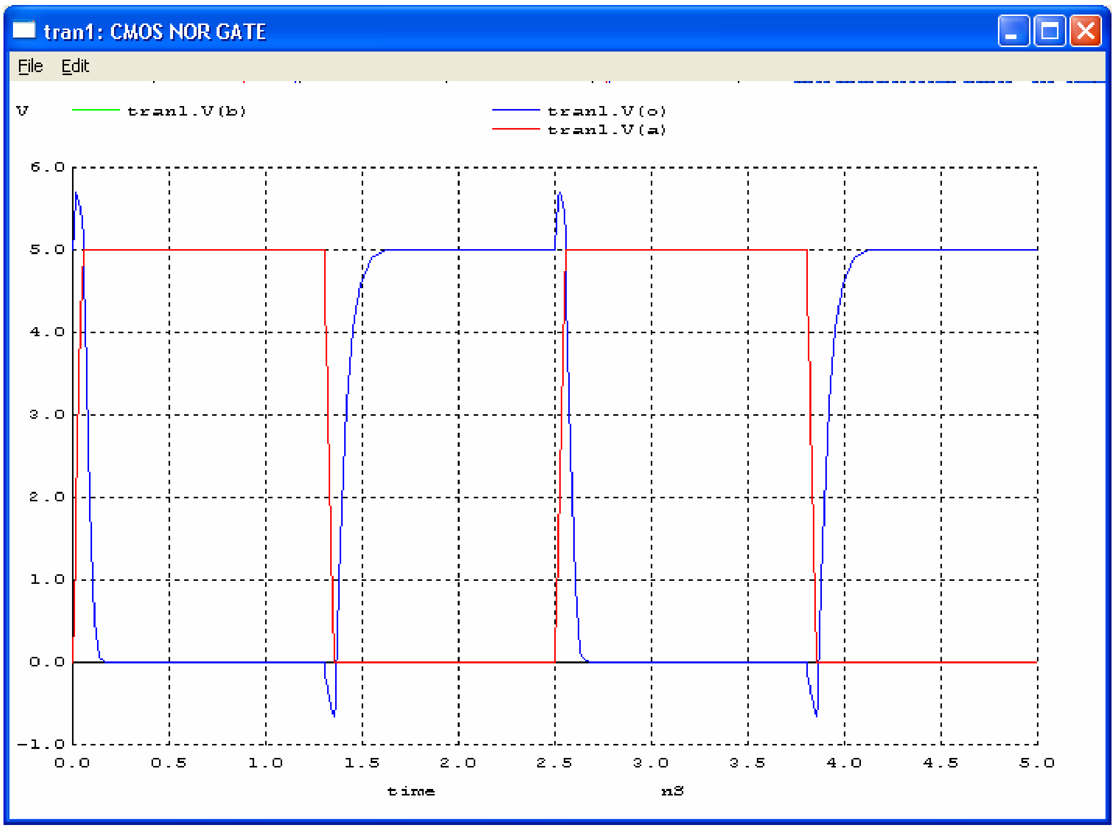


Figure 6 - Simulation of CMOS NOR Gate

The maximum delay required for this NOR Gate equals to 142ps taken at 2.5 V.

3) Transmission Gate

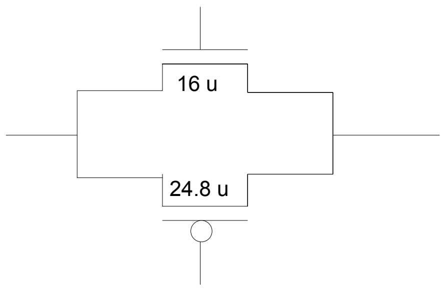


Figure 7 - Transistor-Level of Transmission Gate

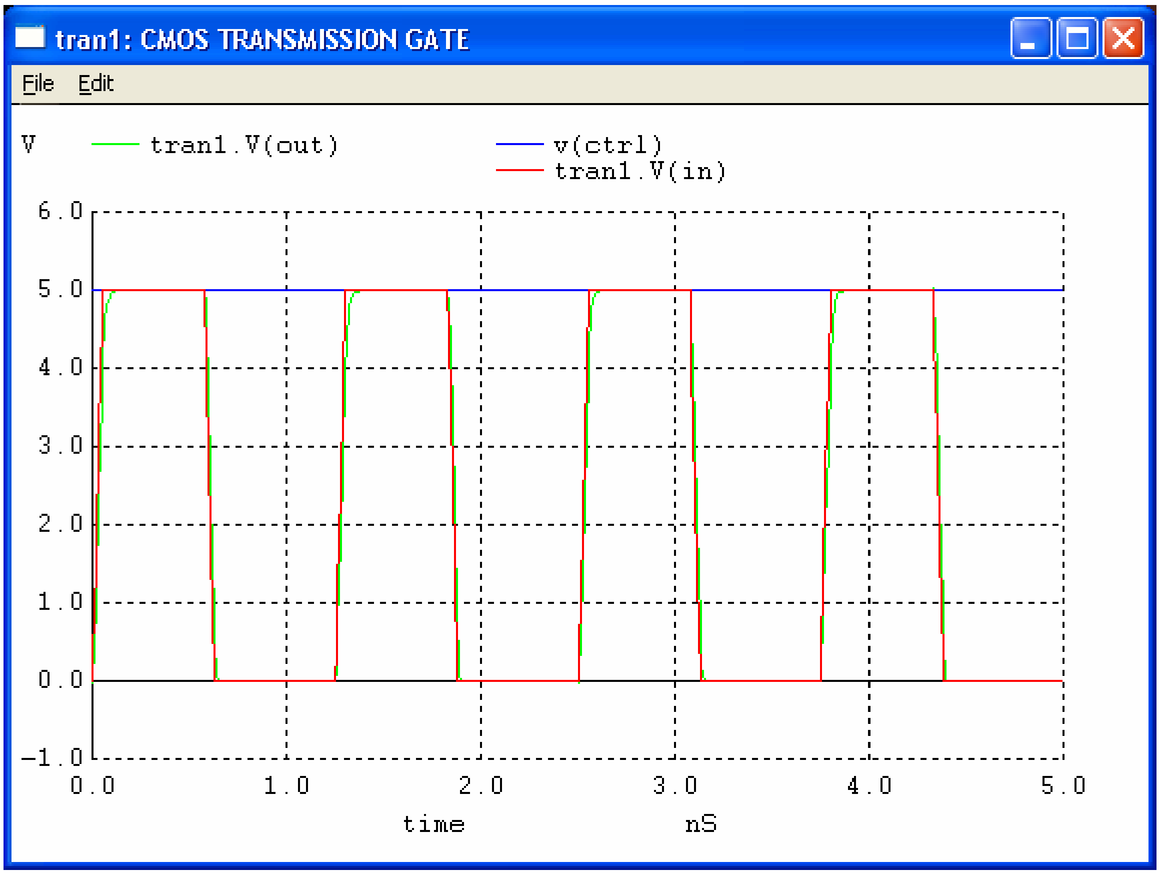


Figure 8 - Simulation of Transmission Gate

The maximum delay required for this transmission gate equals to 34ps taken at 2.5 V

4) Master Latch

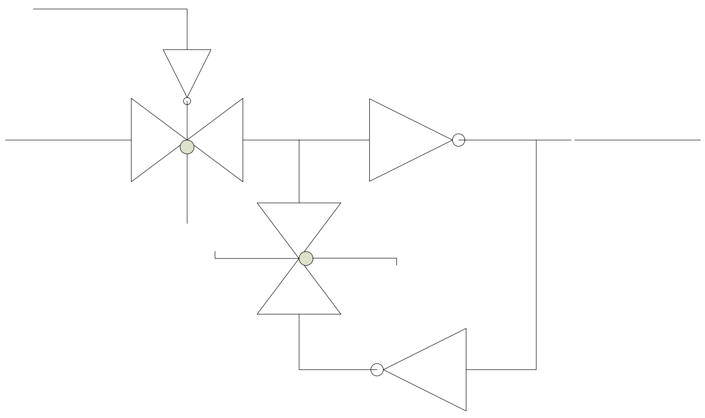
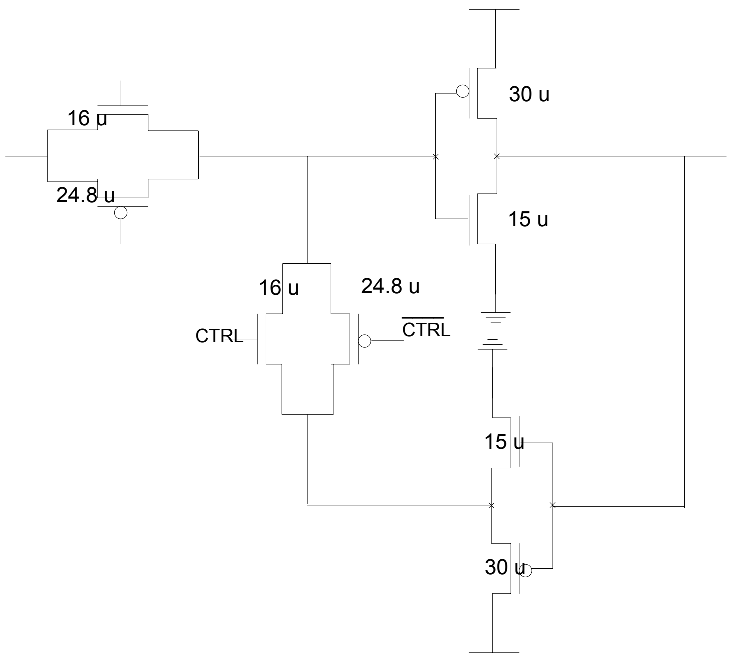


Figure 9 - Logical Diagram of Master Latch

CLK



in

Figure 10 - Transistor-level Diagram of Master Latch

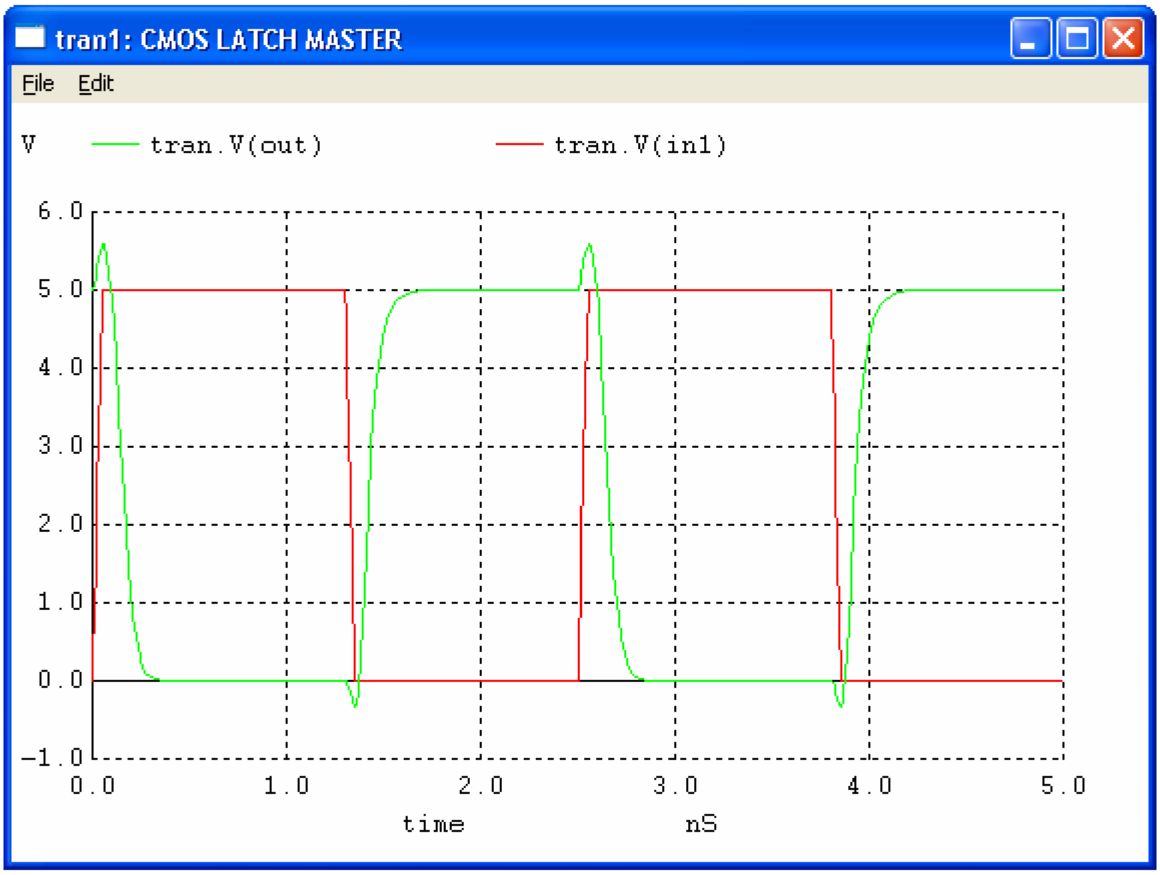


Figure 11 - Simulation of CMOS Master Latch

The maximum delay required for this master latch gate equals to 128ps taken at 2.5 V.

5) Slave Latch

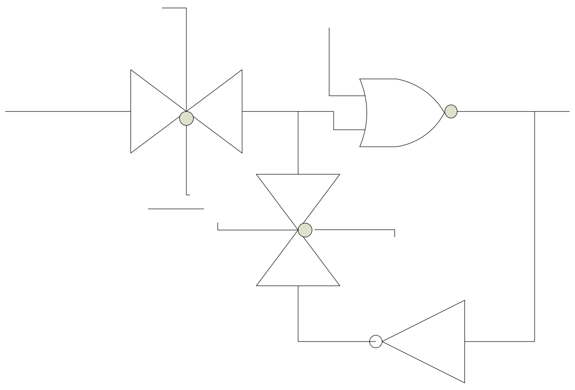


Figure 12 – Logical Diagram of Slave Latch

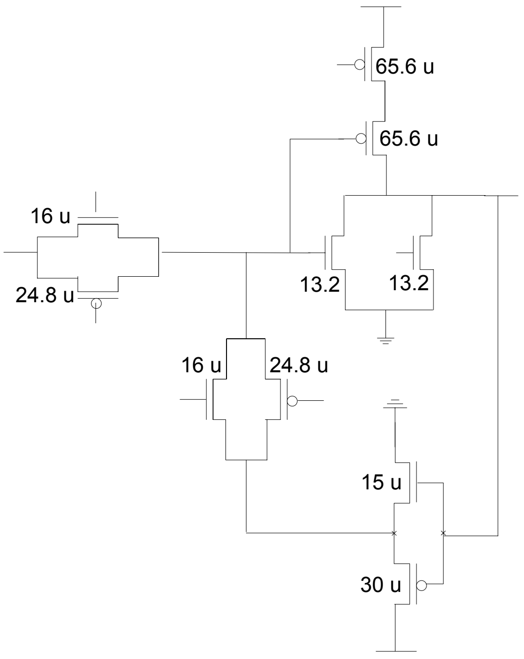


Figure 13 - Transistor-level Diagram of Slave Latch

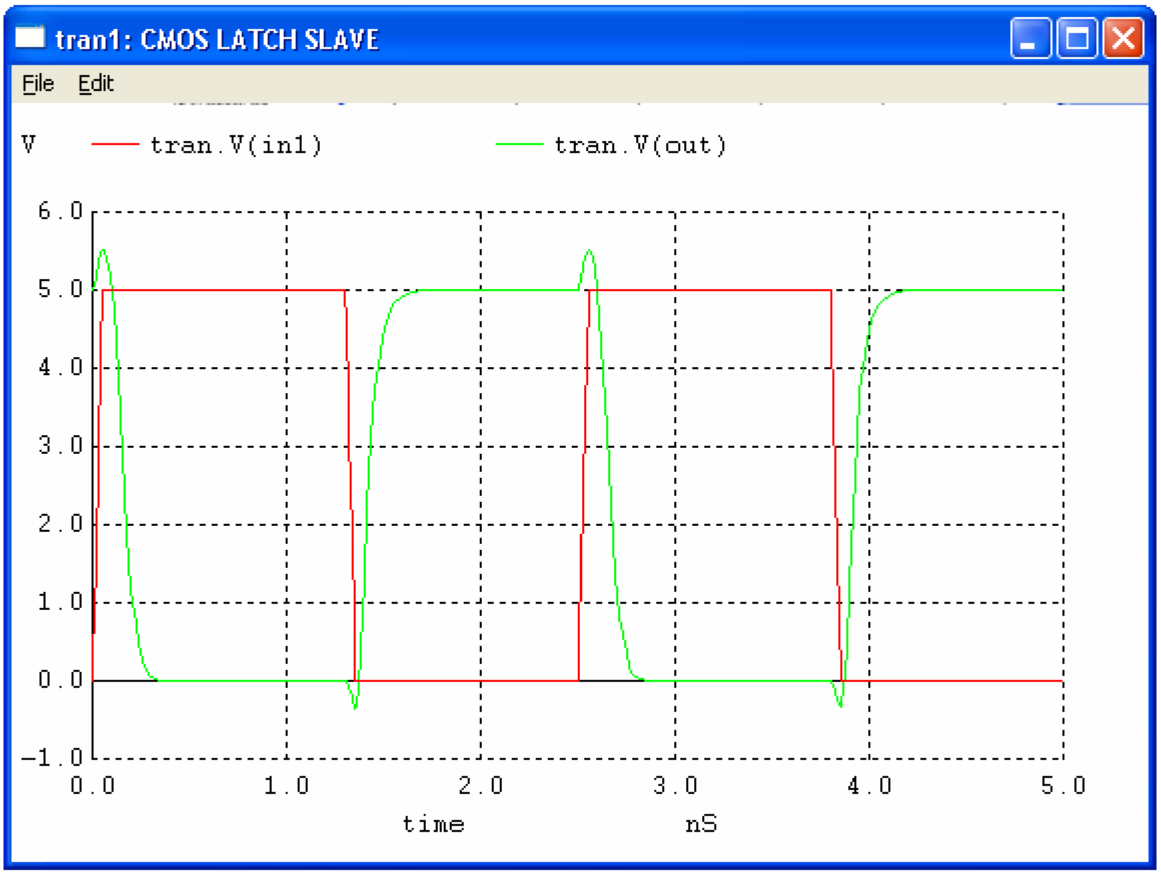


Figure 14 - Simulation of CMOS Slave Latch

The maximum delay required for this slave latch gate equals to 135ps taken at 2.5 V.

6) 1-bit Full Adder

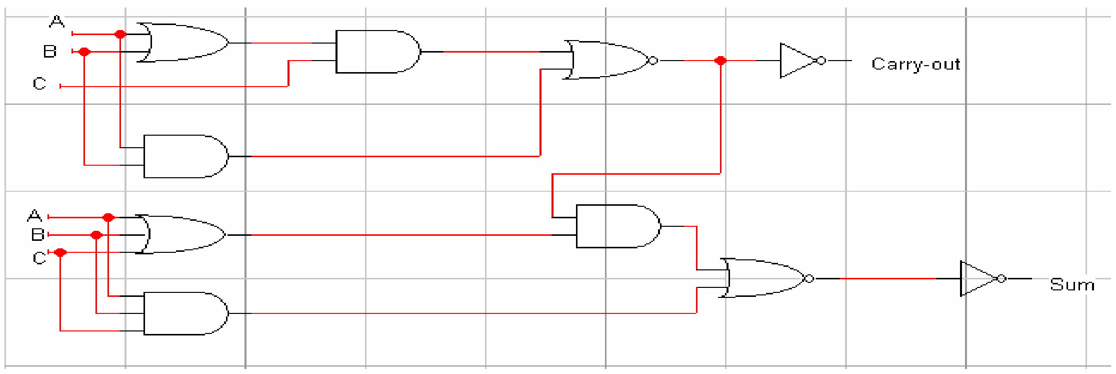


Figure 15 - Logical Diagram of 1-bit Full Adder

The functions of the Carry-out and the Sum correspond to the following logic equations:

$$Carry-out = \overline{AB + C(A+B)}$$

$$Sum = \overline{(A+B+C)\overline{Carry-out} + ABC}$$

This implementation of the block in the transistor-level with the sizing is as follows:

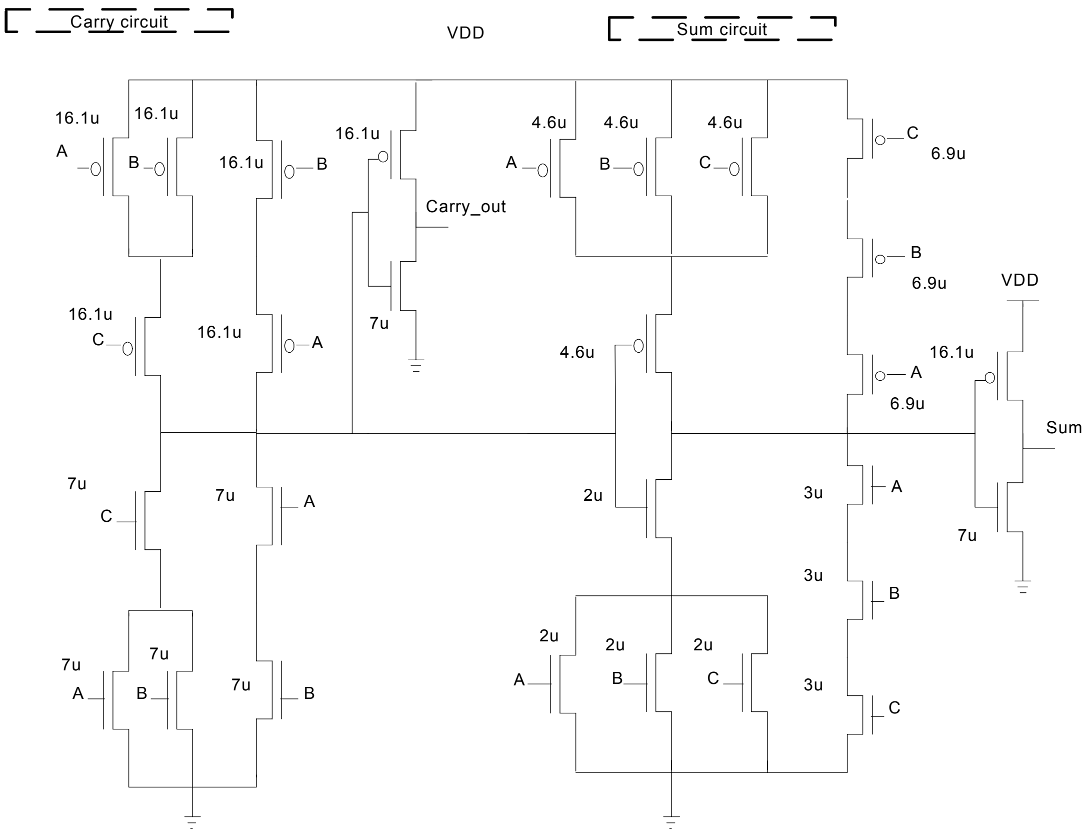


Figure 16 - Transistor-level Diagram of 1-bit Full Adder

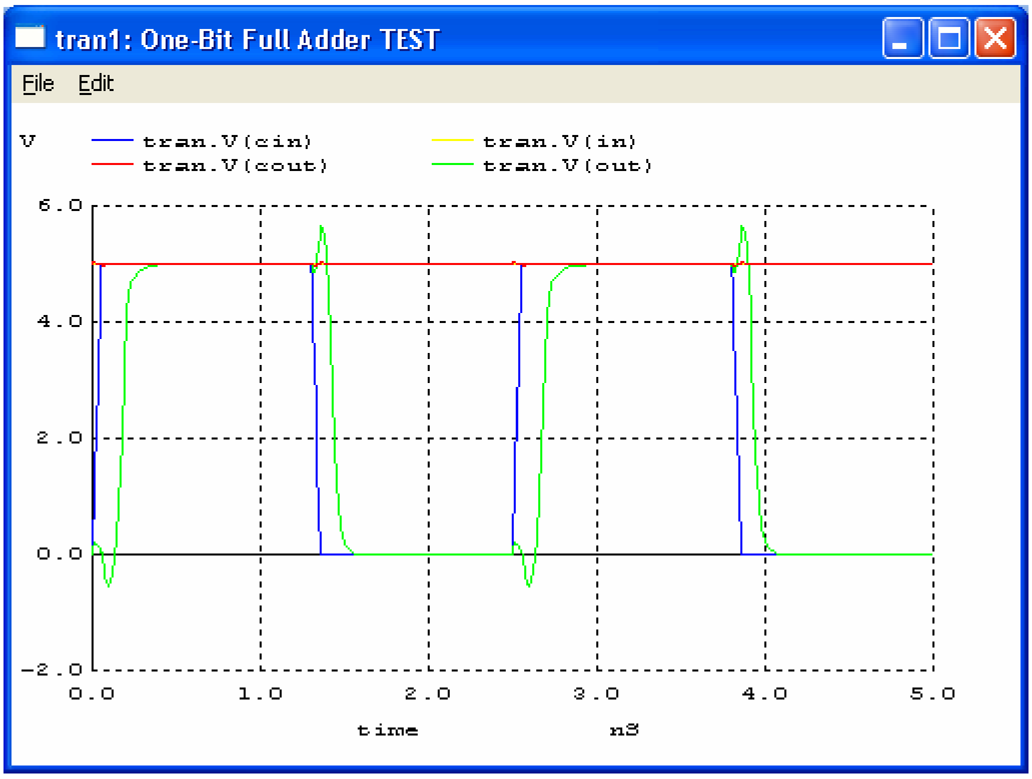


Figure 17 - Simulation of 1-bit Full Adder

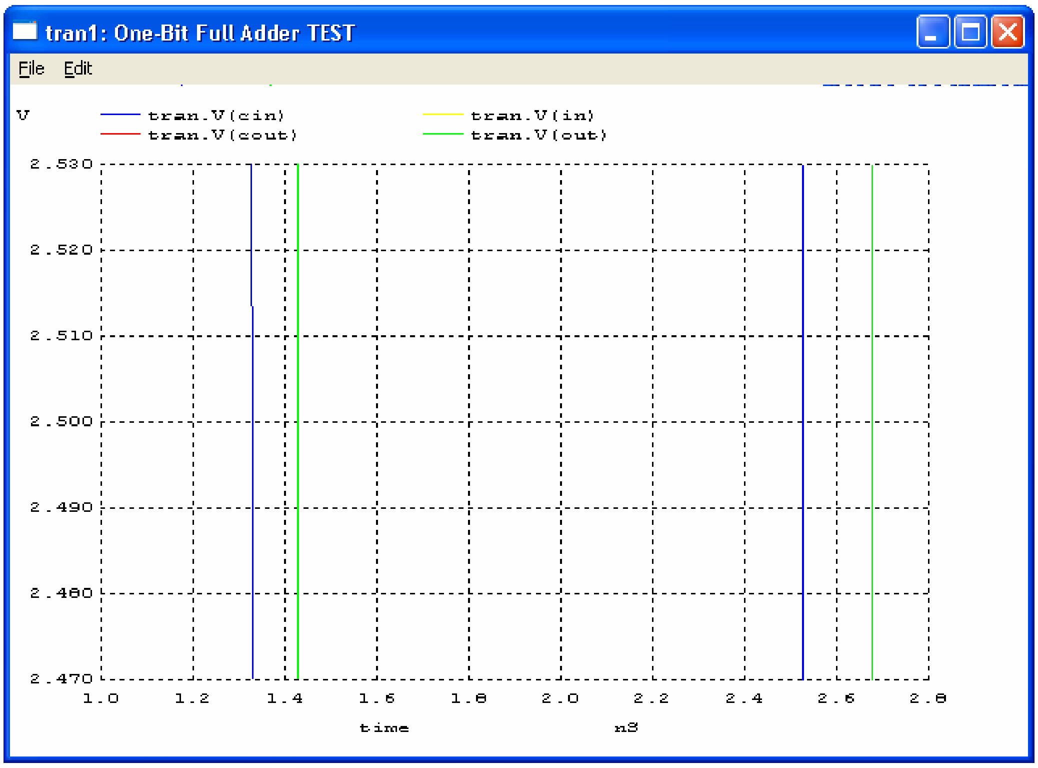


Figure 18 - Simulation of 1-bit Full Adder at 2.5V in Zoom

Note that the maximum delay required taken at 2.5 V varies between 100ps (at the Falling-Edge) and 157ps (at Rising-Edge).

7) 1-bit Flip-Flop

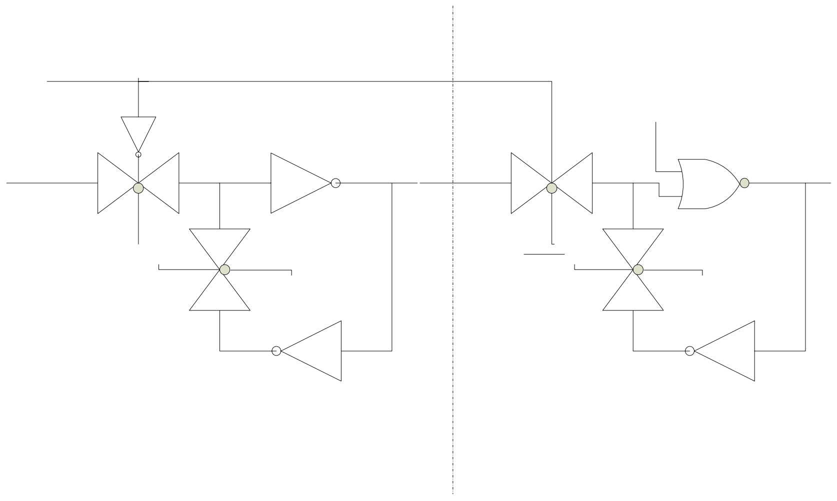


Figure 19 - Logical Diagram of 1-bit Flip-flop

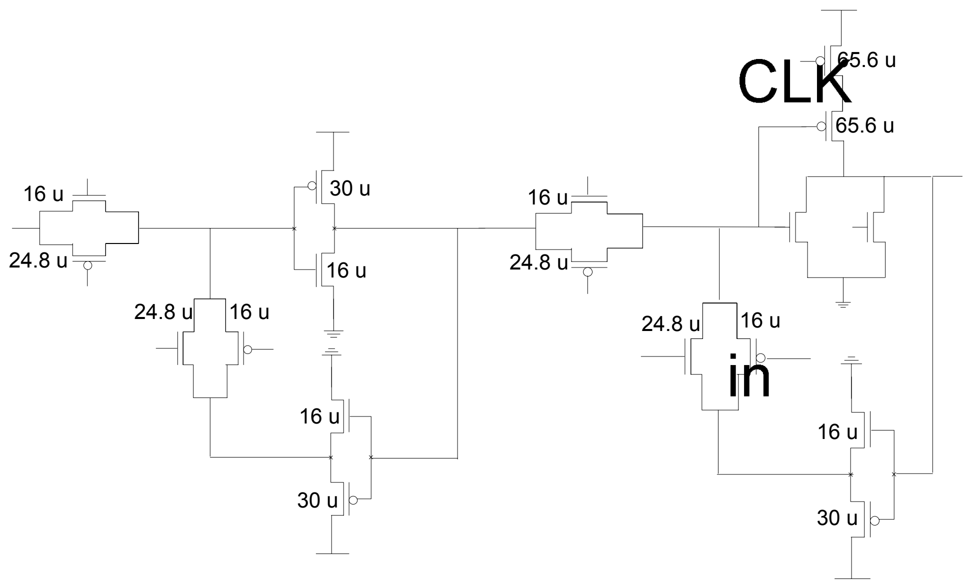


Figure 20 - Transistor-level Diagram of 1-bit Flip-Flop

CLK

8) 4-bit Full Adder

This is the same as the 1-bit Full Adder as in Figure 15 and Figure 16 for the logical and transistor-level diagrams except that the carry-out bit is propagated to the next 1-bit Full Adder and so on resulting in a 4-bit Carry-Ripple Full Adder.

9) Inverter Buffer Chain

This buffer chain is placed at each output of the 16-bit adder to drive a load capacitance of 2 pF. The sizing of the inverters in the buffer chain was chosen to meet an input capacitance of 500 fF. Thus, this buffer chain contains four inverters and this number is calculated using this relationship:

$$n = \left\lceil \ln \frac{C_{load}}{C_{in}} \right\rceil = \left\lceil \ln \frac{2pF}{500fF} \right\rceil = 4 \text{ and } \alpha = e = 2.7$$

$$l = 0.6u, Cox = 2.5fF/um^2$$

According to the hand calculations, the inverter buffer chain that used to drive a load capacitance of 2pF is calculated as follows:

1st stage inverter:

$C_{in0} = 500fF = (W_P + W_N) * l * Cox$

$500 = (W_P + W_N) * 1.5$

$W_P + W_N = 333$

$W_P = 222$

$W_N = 111$

2nd stage inverter:

$W_P = 599.4u$

$W_N = 299.7u$

3rd stage inverter:

$W_P = 1618u$

$W_N = 809.7u$

4th stage inverter:

$W_P = 4369.6u$

$W_N = 2184.3u$

In the diagram shown below, a simulation is conducted with a capacitance load of 2pF and to obtain an input capacitance of 500fF

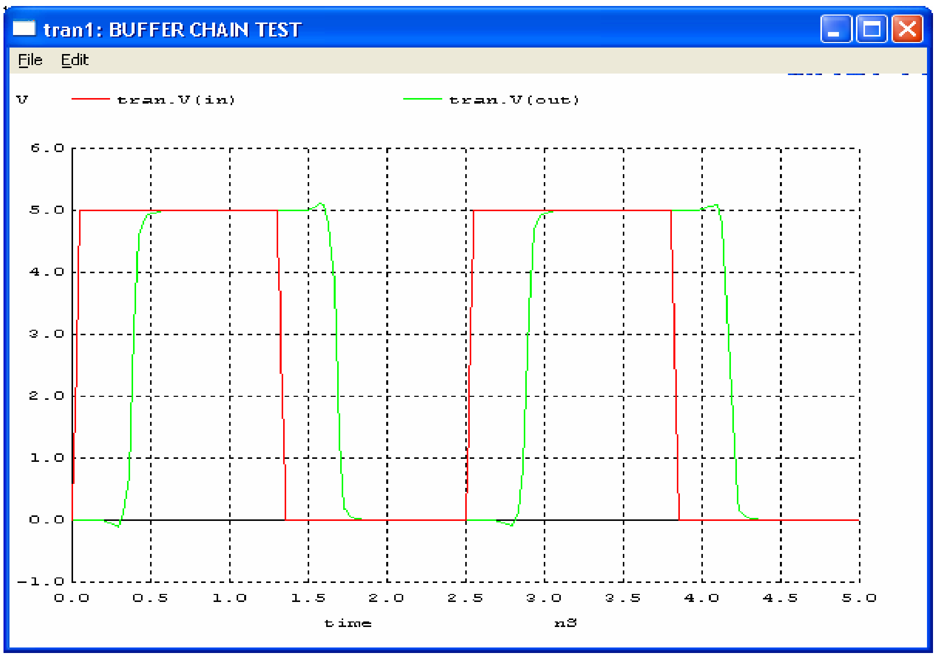


Figure 21 - Simulation of Buffer Chain Inverter

Design Calculations

In this part, we show how to calculate the sizing and delay of the transistors and the parameters used using the AMI 0.5 Micron Technology. There are certain parameters to be used in the calculations for this technology explained as follows:

1- The Channel length L = 0.6 u

2- The ratio for the PMOS and NMOS width is: $W_p / W_n = \mu_n / \mu_p = 452/218 = 2$

3- The oxide capacitance is:

$$C_{ox} = \frac{\varepsilon_{ox}}{t_{ox}} = \frac{4 \times 8.85 \times 10^{-14}}{1.4 \times 10^{-8}} = 2.5 \times 10^{-15}\ F/\mu m^2$$

Using hand calculations, only approximate results can be obtained for the sizing of the widths of the transistor. Consequently, by using transient analysis and varying the sizing of widths with the assumed ratio in WinSpice, real results for the sizing of widths can be realized and the input capacitance of the gates at each stage can be calculated using the following equation:

$$C_{in} = C_{ox} \times l \times \left(W_p + W_n\right)$$

Where C_{ox}, the length and the width ratio are set as mentioned above.

When calculating the input capacitance of a gate, it is assumed to be taken from the propagation of the longest path to the output of the current stage so this input capacitance will be the load capacitance of the next stage until it reaches the next input capacitance.

At first, the design is started from the output capacitance load which is 2pF and moved backward to the inputs of the 16-bit Adder. This has been done for several times as part of tweaking to minimize the delay as possible using WinSpice. The goal of the design is to have one stage operation delay within 1.25 ns (800 MHz). However, after several trials of designs, a delay of 2 ns is obtained for one stage operation as shown below in Figure 23. Therefore, this design can be considered as the closest to the required specifications. Moreover, if the frequency is reduced to 400 MHz where the time period equals to 2.5ns, we will be in the safe side such that the outputs for the next stage will be ready before the beginning of next period.

Critical Path Calculation

Determining the correct critical path gives the right estimation for the delay to be within the required time period. It is found that the longest path of one stage in the 16-bit VLSI Pipelined Adder centers at the propagation of the carry-bit path. Using the second design alternative, the delay in this critical path for the carry-bit propagation within one period (i.e. from Flip-flop to the next Flip-flop passing through one block of 4-bit Full Adder) will be the summation of the delays in the following blocks:

i) 2 blocks of 4-bit Flip-Flop
ii) 1 block of 4-bit Full Adder

The delay in one block of 4-bit Flip-Flop equals to half of the total delays either of the following gates in the case of Master Latch:

- 4 inverters
- 4 Transmission Gates

Or the following gates in the case of Slave Latches

- 4 NOR Gates
- 4 Transmission Gates

Moreover, the delay in one block of 4-bit Full Adder equals to the total delays of the propagation of the carry-bit path in a single 1-bit Full Adder as shown in the figure below.

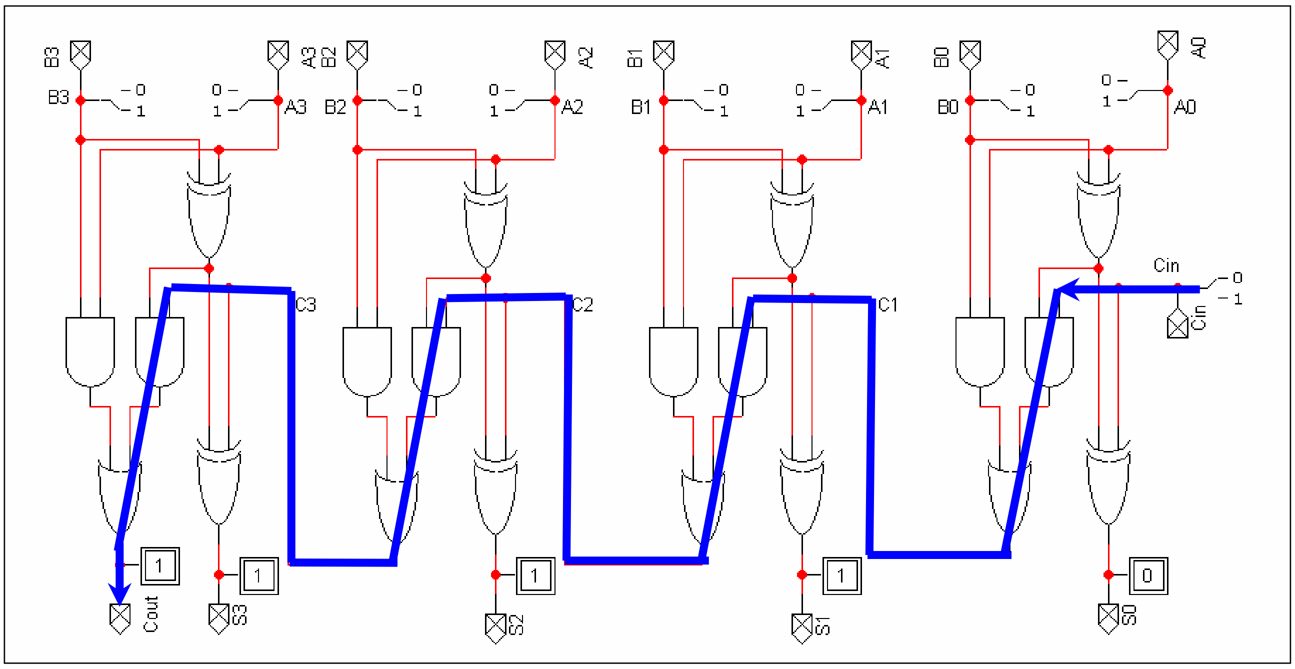


Figure 22 - Critical Path Diagram

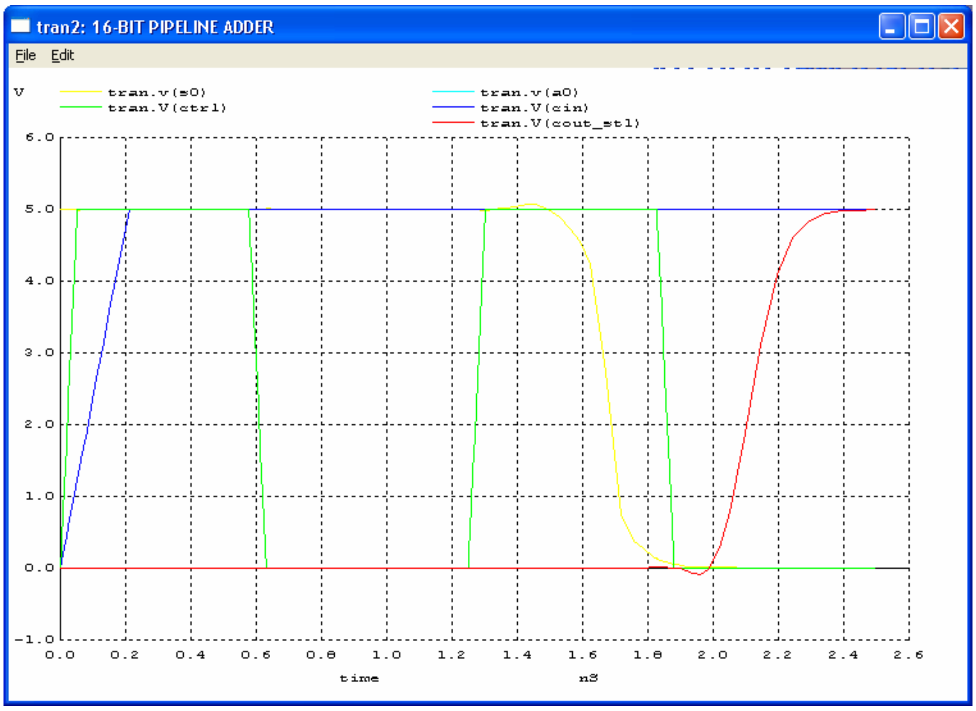

tran2: 16-BIT PIPELINE ADDER

Figure 23 - 1st Stage of 16-bit Pipelined Adder with T_D = 2 ns at 800MHz

Output Verification & Testing

In each block design, the function of the circuit is verified correctly at each stage first. This is important to ease the debugging and tweaking of the full design of the 16-bit Pipelined Adder if there were any errors throughout the stages. The design of this adder is divided into blocks (sub-circuits) and then connecting them to build the complete 16-bit pipelined adder.

To ensure the correct functionality of the 16-bit pipelined adder, several schemes of testing are conducted and the outcomes obtained lead to acceptable results.

Note that we need 5 clock cycles to obtain the first output, and then this output will appear at the 6th cycle. After that, at each cycle we obtain a new output.

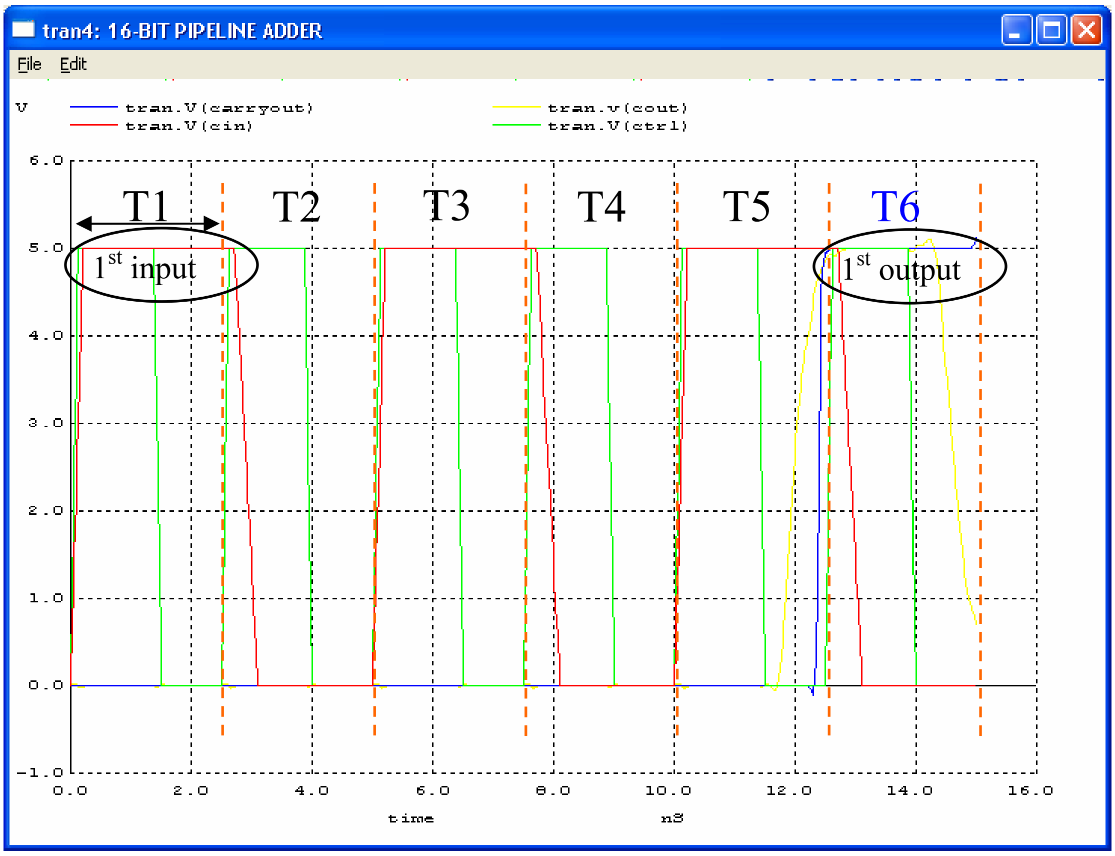


Figure 24 – Detailed Diagram of the 16-bit Pipelined Adder Outputs at 400MHz

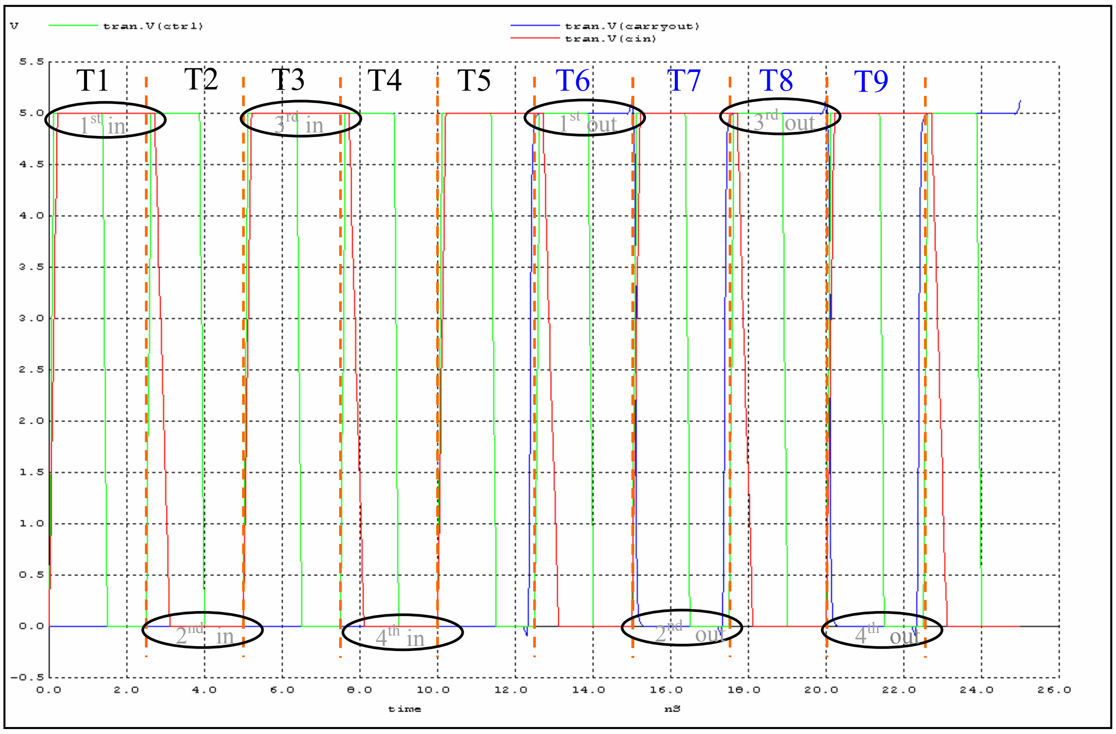


Figure 25 - Detailed Diagram of the 16-bit Pipelined Adder Outputs at 400 MHz

The following diagram shows a simulation of the 16-bit Pipelined Adder at 500MHz.

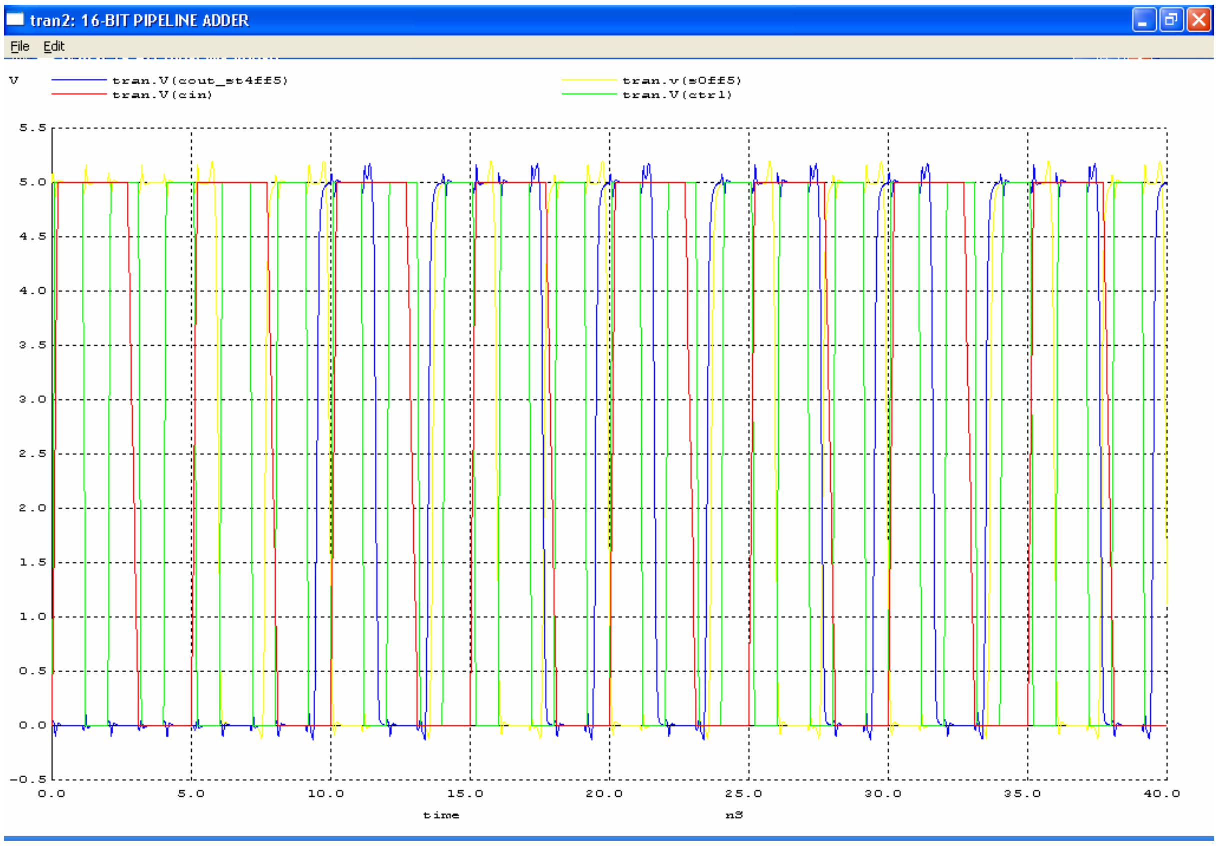


Figure 26 – Another Simulation of the 16-bit Pipelined Adder Outputs at 500 MHz

In the following simulation, random inputs are given at 800 MHz frequency:

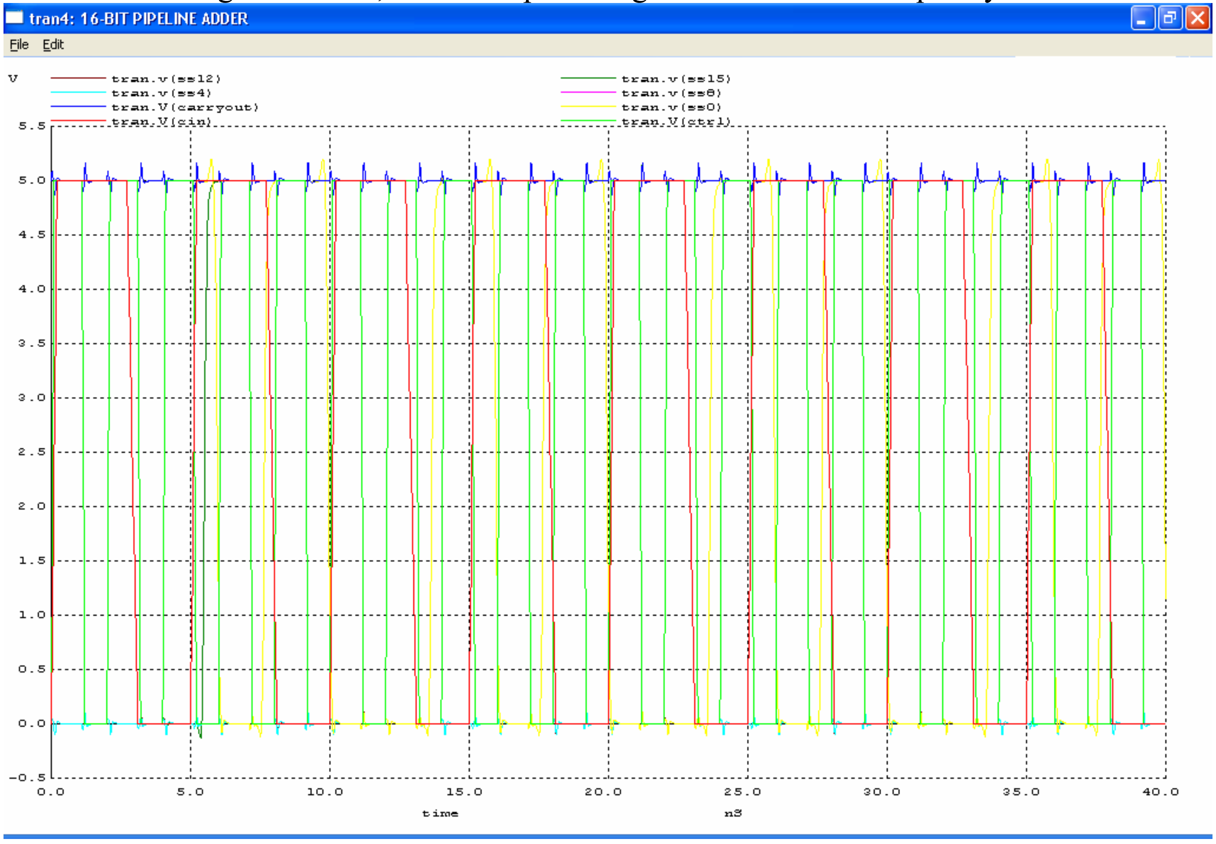


Figure 27 – Another Simulation of the 16-bit Pipelined Adder Outputs at 800 MHz

Appendix

```
**** The Inverter subcircuit ****
.SUBCKT INV in out

VDD vdd 0 5

*    D  G   S   B
MP1 out in vdd  vdd CMOSP L=0.6U W=30U
MN1 out in 0    0   CMOSN L=0.6U W=15U
.ENDS INV
*--------------------------------------------*

**** THE TRANSMISSION GATE ***
.SUBCKT TG in ctrl ctrlbar out

VDD vdd 0 5

*    D  G   S   B
MP1 in ctrlbar out  vdd CMOSP L=0.6U W=24.8U
MN1 in ctrl    out  0   CMOSN L=0.6U W=16U
.ENDS TG
*--------------------------------------------*

******** THE NOR GATE **********
.SUBCKT NOR a b o

VDD v 0 5

*   D G S B
MP1 v b x v CMOSP L=0.6U W=65.6U
MP2 x a o v CMOSP L=0.6U W=65.6U
MN1 o b 0 0 CMOSN L=0.6U W=13.2U
MN2 o a 0 0 CMOSN L=0.6U W=13.2U
.ENDS NOR
*--------------------------------------------*

******** THE NAND GATE ********
.SUBCKT NAND a b o

VDD v 0 5

*   D G S B
MP1 v b o v CMOSP L=0.6U W=65.6U
MP2 v a o v CMOSP L=0.6U W=65.6U
MN1 o a x 0 CMOSN L=0.6U W=13.2U
MN2 x b 0 0 CMOSN L=0.6U W=13.2U
.ENDS NAND
*---------------------------------------------*


***** MASTER LATCH **********
*             IND INC INCB OUT
.SUBCKT MLATCH  in  ctrl ctrlbar out

XIN1 out1 out INV
XIN2 out in2 INV

XN1 in ctrlbar ctrl out1 TG
XN3 in2 ctrl ctrlbar out1 TG
.ENDS MLATCH
*******************************************
```

```
******** SLAVE LATCH ********
*             IND INC INCB INR OUT
.SUBCKT SLATCH  in ctrl ctrlbar clr out

XIN1 out1 clr out NOR
XIN2 out in2 INV

XN1 in ctrl ctrlbar out1 TG
XN3 in2 ctrlbar ctrl out1 TG
.ENDS SLATCH
*-------------------------------------------*

******** ONE-BIT FLIP FLOP ********
.SUBCKT FF in ctrl clr out

XINV ctrl ctrlbar INV
XN1 in ctrl ctrlbar mout MLATCH

XN3 mout ctrlbar  ctrl clr out SLATCH
.ENDS FF
*-------------------------------------------*

********** FOUR-BIT FLIP FLOP *********
.SUBCKT FF4 in0 in1 in2 in3 ctrl clr out0 out1 out2 out3

XFF0 in0 ctrl clr out0 FF
XFF1 in1 ctrl clr out1 FF
XFF2 in2 ctrl clr out2 FF
XFF3 in3 ctrl clr out3 FF
.ENDS FF4
*--------------------------------------------*

************** ONE BIT ADDER ********************
.SUBCKT ADDER1BIT a b c sum cout
* nodes: 1st input= a, 2nd input = b, carry in= c
*       carry out= cout, sum = sum

VDD vdd 0 5
   ************************
   ***** Carry out Circuit:
  * D    G    S   B
MP1 px1  a    vdd vdd      CMOSP L=0.6U W=16.1U
MP2 px1  b    vdd vdd      CMOSP L=0.6U W=16.1U
MP3 out1 c    px1 vdd      CMOSP L=0.6U W=16.1U
MP4 px2  b    vdd vdd      CMOSP L=0.6U W=16.1U
MP5 out1 a    px2 vdd      CMOSP L=0.6U W=16.1U
MP6 cout out1 vdd vdd      CMOSP L=0.6U W=16.1U

MN1 out1 c    nx1 0 CMOSN L=0.6U W=7U
MN2 nx1  a    0   0 CMOSN L=0.6U W=7U
MN3 nx1  b    0   0 CMOSN L=0.6U W=7U
MN4 out1 a    nx2 0 CMOSN L=0.6U W=7U
MN5 nx2  b    0   0 CMOSN L=0.6U W=7U
MN6 cout out1 0   0 CMOSN L=0.6U W=7U
   *****************
   **** Sum Circuit:
   * D    G    S   B
MP11 px3  a    vdd vdd     CMOSP L=0.6U W=4.8U
MP22 px3  b    vdd vdd     CMOSP L=0.6U W=4.8U
MP33 px3  c    vdd vdd     CMOSP L=0.6U W=4.8U
MP44 out2 out1 px3 vdd     CMOSP L=0.6U W=4.8U
MP55 px4  a    vdd vdd     CMOSP L=0.6U W=6.9U
MP66 px5  b    px4 vdd     CMOSP L=0.6U W=6.9U
MP77 out2 c    px5 vdd     CMOSP L=0.6U W=6.9U
MP88 sum  out2 vdd vdd     CMOSP L=0.6U W=16.1U

MN11 out2 out1 nx3 0       CMOSN L=0.6U W=2U
MN22 nx3  a    0   0       CMOSN L=0.6U W=2U
MN33 nx3  b    0   0       CMOSN L=0.6U W=2U
MN44 nx3  c    0   0       CMOSN L=0.6U W=2U
MN55 out2 a    nx4 0       CMOSN L=0.6U W=3U
```

```
MN66 nx4  b    nx5 0       CMOSN L=0.6U W=3U
MN77 nx5  c    0   0       CMOSN L=0.6U W=3U
MN88 sum  out2 0   0       CMOSN L=0.6U W=7U
.ENDS ADDER1BIT
*********** END OF ONE BIT ADDER ****************
**************************************************

*********** FOUR BIT ADDER *********************
.SUBCKT ADDER4BIT a0 b0 a1 b1 a2 b2 a3 b3 cin
+              s0 s1 s2 s3 cout

XAdder0   a0  b0  cin    s0 cout0 ADDER1BIT
XAdder1   a1  b1  cout0  s1 cout1 ADDER1BIT
XAdder2   a2  b2  cout1  s2 cout2 ADDER1BIT
XAdder3   a3  b3  cout2  s3 cout  ADDER1BIT
.ENDS ADDER4BIT
********** END OF FOUR BIT ADDER ***************
*************************************************

********  SINGLE STAGE 4BIT FULL ADDER **********
.SUBCKT SS4FA       ina0 inb0 ina1 inb1 ina2 inb2 ina3 inb3 cin
+            s0 s1 s2 s3 cout_st1 ctrl clr

XFF1bit_CIN cin ctrl clr cin_adder FF
XFF4bit_A ina0 ina1 ina2 ina3 ctrl clr a0 a1 a2 a3 FF4
XFF4bit_B inb0 inb1 inb2 inb3 ctrl clr b0 b1 b2 b3 FF4

XADD1 a0 b0 a1 b1 a2 b2 a3 b3 cin_adder s0 s1 s2 s3 cout_st1 ADDER4BIT
.ENDS SS4FA
****** END OF SINGLE STAGE 4BIT FULL ADDER ******

*************************************************
   *******************************************
     *******  THE 16-BIT PIPELINE ADDER ****
   *******************************************
*************************************************

.SUBCKT PIPELINEADDER16 a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15
+                   b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 b10 b11 b12 b13 b14 b15
+                   cin ctrl clr
+                   ss0 ss1 ss2 ss3 ss4 ss5 ss6 ss7 ss8 ss9 ss10 ss11 ss12 ss13
ss14 ss15
+                   carryout

* The Adder is divided into five stages to
* achive the PIPELINING approach.
* Each stage is consists of flipflops and one SS4FA subckt,
* except stage 5, it consists of flipflops only.

   ************* STAGE # 1:  ***************
XFFCIN cin ctrl clr coutff1 FF

XST1_SS4FA    a0 b0 a1 b1 a2 b2 a3 b3 coutff1
+            s0 s1 s2 s3 cout_st1 ctrl clr SS4FA

XST1FFA4_A7   a4  a5  a6  a7  ctrl clr a4ff1  a5ff1  a6ff1  a7ff1  FF4
XST1FFA8_A11  a8  a9  a10 a11 ctrl clr a8ff1  a9ff1  a10ff1 a11ff1 FF4
XST1FFA12_A15 a12 a13 a14 a15 ctrl clr a12ff1 a13ff1 a14ff1 a15ff1 FF4

XST1FFB4_B7   b4  b5  b6  b7  ctrl clr b4ff1  b5ff1  b6ff1  b7ff1  FF4
XST1FFB8_B11  b8  b9  b10 b11 ctrl clr b8ff1  b9ff1  b10ff1 b11ff1 FF4
XST1FFB12_B15 b12 b13 b14 b15 ctrl clr b12ff1 b13ff1 b14ff1 b15ff1 FF4
   ****************************************

   ************* STAGE # 2:  ***************
XST2FFS0_S3   s0 s1 s2 s3  ctrl clr s0ff2  s1ff2  s2ff2  s3ff2 FF4

XST2FFCOUT_ST1 cout_st1 ctrl clr cout_st1ff2 FF

XST2_SS4FA a4ff1  b4ff1 a5ff1 b5ff1 a6ff1 b6ff1 a7ff1 b7ff1 cout_st1ff2
+         s4 s5 s6 s7 cout_st2 ctrl clr SS4FA
```

```
XST2FFA8_A11  a8ff1  a9ff1  a10ff1 a11ff1 ctrl clr a8ff2  a9ff2  a10ff2 a11ff2 FF4
XST2FFA12_A15 a12ff1 a13ff1 a14ff1 a15ff1 ctrl clr a12ff2 a13ff2 a14ff2 a15ff2 FF4
XST2FFB8_B11  b8ff1  b9ff1  b10ff1 b11ff1 ctrl clr b8ff2  b9ff2  b10ff2 b11ff2 FF4
XST2FFB12_B15 b12ff1 b13ff1 b14ff1 b15ff1 ctrl clr b12ff2 b13ff2 b14ff2 b15ff2 FF4
   *****************************************

   ************* STAGE # 3:  ***************
XST3FFS0_S3   s0ff2 s1ff2 s2ff2 s3ff2 ctrl clr s0ff3 s1ff3 s2ff3 s3ff3 FF4
XST3FFS4_S7   s4    s5    s6    s7    ctrl clr s4ff3 s5ff3 s6ff3 s7ff3 FF4

XST3FFA8_A11  a8ff2  a9ff2  a10ff2 a11ff2 ctrl clr a8ff3  a9ff3  a10ff3 a11ff3 FF4
XST3FFB8_B11  b8ff2  b9ff2  b10ff2 b11ff2 ctrl clr b8ff3  b9ff3  b10ff3 b11ff3 FF4

XST3FFCOUT_ST2 cout_st2 ctrl clr cout_st2ff3 FF

XST3_SS4FA a8ff3  b8ff3 a9ff3 b9ff3 a10ff3 b10ff3 a11ff3 b11ff3 cout_st2ff3
+         s8 s9 s10 s11 cout_st3 ctrl clr SS4FA

XST3FFA12_A15 a12ff2 a13ff2 a14ff2 a15ff2 ctrl clr a12ff3 a13ff3 a14ff3 a15ff3 FF4
XST3FFB12_B15 b12ff2 b13ff2 b14ff2 b15ff2 ctrl clr b12ff3 b13ff3 b14ff3 b15ff3 FF4
   *****************************************

   ************* STAGE # 4:  ***************
XST4FFS0_S3   s0ff3 s1ff3 s2ff3 s3ff3 ctrl clr s0ff4 s1ff4 s2ff4 s3ff4 FF4
XST4FFS4_S7   s4ff3 s5ff3 s6ff3 s7ff3 ctrl clr s4ff4 s5ff4 s6ff4 s7ff4 FF4
XST4FFS8_S11  s8    s9    s10   s11   ctrl clr s8ff4 s9ff4 s10ff4 s11ff4 FF4

XST4FFA12_A15 a12ff3 a13ff3 a14ff3 a15ff3 ctrl clr a12ff4 a13ff4 a14ff4 a15ff4 FF4
XST4FFB12_B15 b12ff3 b13ff3 b14ff3 b15ff3 ctrl clr b12ff4 b13ff4 b14ff4 b15ff4 FF4

XST4FFCOUT_ST3 cout_st3 ctrl clr cout_st3ff4 FF

XST4_SS4FA a12ff4  b12ff4 a13ff4 b13ff4 a14ff4 b14ff4 a15ff4 b15ff4 cout_st3ff4
+         s12 s13 s14 s15 cout_st4 ctrl clr SS4FA
   *****************************************

   ************* STAGE # 5:  ***************
XST5FFS0_S3   s0ff4 s1ff4 s2ff4  s3ff4  ctrl clr ss0  ss1  ss2  ss3  FF4
XST5FFS4_S7   s4ff4 s5ff4 s6ff4  s7ff4  ctrl clr ss4  ss5  ss6  ss7  FF4
XST5FFS8_S11  s8ff4 s9ff4 s10ff4 s11ff4 ctrl clr ss8  ss9  ss10 ss11 FF4
XST4FFS12_S15 s12   s13   s14    s15    ctrl clr ss12 ss13 ss14 ss15 FF4

XST5FFCOUT_ST4 cout_st4 ctrl clr carryout FF
* cout_st4ff5 = The Final Carry out

.ENDS PIPELINEADDER16
*******************************************************
********* END OF THE 16-BIT PIPELINE ADDER ***********
*******************************************************

***********  BUFFER CHAIN *******
*********************************
*** THE INVERTERS USED IN THE CHAIN:
.SUBCKT INV1 in out
VDD vdd 0 5
MP1 out in vdd  vdd CMOSP L=0.6U W=222U
MN1 out in 0    0   CMOSN L=0.6U W=111U
.ENDS INV1
******************************************
.SUBCKT INV2 in out
VDD vdd 0 5
MP1 out in vdd  vdd CMOSP L=0.6U W=599.4U
MN1 out in 0    0   CMOSN L=0.6U W=299.7U
.ENDS INV2
******************************************
.SUBCKT INV3 in out
VDD vdd 0 5
MP1 out in vdd  vdd CMOSP L=0.6U W=1618U
MN1 out in 0    0   CMOSN L=0.6U W=809U
.ENDS INV3
```

```
******************************************
.SUBCKT INV4 in out
VDD vdd 0 5
MP1 out in vdd  vdd CMOSP L=0.6U W=4369.6U
MN1 out in 0    0   CMOSN L=0.6U W=2184.3U
.ENDS INV4
******************************************

*** THE CHAIN:
.SUBCKT BUFFER in out

XINV1 in   out1 INV1
XINV2 out1 out2 INV2
XINV3 out2 out3 INV3
XINV4 out3 out  INV4

.ENDS BUFFER
*********** END OF THE BUFFER CHAIN *******
*******************************************
```