VLSI and SPICE Modeling of ALU

Saeid Moslehpour
University of Hartford
moslehpou@hartford.edu

Srikrishna Karatalapu
University of Hartford
srikrishna.karatalapu@gmail.com

Abstract

The Arithmetic and Logic Unit (ALU) is a combination circuit that performs a number of arithmetic and logical operations within a microprocessor. The demand for faster and compact ALUs makes it desirable to test the ALU in conjunction with pre-design parts prior to manufacture. This may be accomplished in a process using CAD and SPICE simulation software. Our purpose is to realize a method for importing a layout drawn in Tanner L-edit and simulated in T-Spice into PSpice which is referred to as software talking. To do so we use an eight-function instruction set called Complimentary Metal Oxide Semiconductor Arithmetic and Logic Unit (CMOS ALU) which is laid out in Tanner L-edit and produces an extracted net-list which is simulated in T-Spice. An ALU equivalent design is then modeled in PSpice for further testing with pre-manufactured parts of the PSpice library.

ALU Design

The top-level module consists of a four bit ALU. An eight function instruction set CMOS ALU performs the following: addition, subtraction, AND, NAND, OR, NOR, XOR and XNOR. Each of these functions is performed on two four-bit input bitwise operations. The bitwise output results in the 8X1 multiplexer. Each of the single bit building blocks are cascaded together to form a four bit ALU [1].

The eight-function instruction set of the CMOS ALU:

ADDITION: The output performed by a Ripple-Adder consists of four sum-bits and a single carry-out bit.
SUBTRACTION: The input is complimented using twos compliment. The output of the Ripple-Adder consists of four sum bits and a single carry-out bit.
NAND: The NAND gate is a universal gate used to construct various logic operations.
AND: The AND gate is a basic gate which can be constructed in CMOS by inverting the NAND gate.
NOR: The NOR gate is a universal gate and can be used to construct various logic operations.

OR: The OR gate is a universal gate and can be constructed in CMOS by inverting the NOR gate.
XOR: The XOR gate can be constructed by a combination of NAND and NOR gates.
XNOR: This is constructed by inverting the XOR gate.

Each of the CMOS ALU functions are performed on a single bit input in Tanner L-edit using 0.6 micron technology. The layout is the extract using the M12_5 model file. The extracted net-list is then simulated using T-Spice. Figure 1. shows the four bit ALU block diagram.

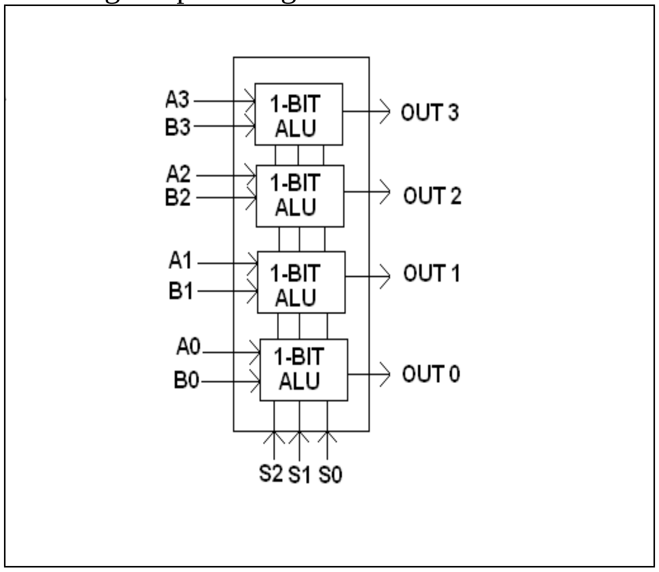


Figure 1: Four bit ALU Block Diagram

The device is then modeled in OrCAD-PSpice A/D, a simulation program that models the behavior of a circuit containing any mix of analog and digital devices. Used with OrCAD Capture for design entry, PSpice A/D is a software-based breadboard circuit that can used to test and refine a design before manufacture. [2] PSpice modeling involves writing code the PSpice editor, Verilog, to model electronic devices. The result is the ALU model.

CMOS Design Methodology

CMOS Design methodology consists of eleven components.
INVERTER: The inverter consists of an NMOS and a PMOS connected in series. The PSWITCH is connected from a ‘1’ source i.e. the VDD to the output and input The NSWITCH is connected from a ‘0’ source i.e. the GND to the output and the input.
NAND GATE: The CMOS NAND gate is derived examining the K-MAP. The ‘0’ dictates the AND structure which is constructed using two NMOS in PARALELL. The ‘1’ dictates the OR structure which is constructed using two P-MOS connected in SERIES
AND GATE: The CMOS AND gate has been designed by inverting the CMOS NAND gate. The output of AND is high only when both the inputs are high i.e. the output is low when any one of the inputs is low.
NOR GATE: The CMOS NOR gate is derived examining the K-MAP. The ‘0’ dictates the AND structure which is constructed using two NMOS in parallel. The ‘1’ dictates the OR structure which is constructed using two P-MOS connected in series.

OR GATE: The CMOS OR gate has been designed by inverting the CMOS NOR gate. The output of OR is high only when both or any one of the inputs is high i.e. the output is low only when both the inputs are low.
XOR GATE: The Exclusive-OR, or XOR function can be described verbally as, "Either A or B, but not both." The output of an XOR gate is high only when any one of the inputs is low i.e. for the output of the XOR gate to be high both the inputs should be either high or low. A XOR gate can be designed by a combination of NAND and OR gates.
XNOR GATE: The XNOR gate is designed by inverting the XOR gate .The output of the XNOR gate is high when both the inputs are low and both the inputs are high. A CMOS representation of the XNOR
FULL-ADDER: The full-adder circuit adds three one-bit binary numbers (C A B) and outputs two one-bit binary numbers, a sum (S) and a carry (C). The full-adder is usually a component in a cascade of adders, which add 8, 16, 32, etc. binary numbers. The output of XOR gate is called SUM, while the output of the AND gate is the CARRY. The AND gate produces a high output only when both inputs are high. The XOR gate produces a high output if either input, but not both, is high. The 'C' for an ADDER is always made low
SUBTRACTOR: Binary subtraction is performed by adding the two's compliment of the number to be subtracted. 2's compliment of a number can be achieved by inverting the number and adding one to it. This is achieved by inverting each bit of the number to be subtracted and adding '1' by means of the carry-in. The carry-in of a Subtractor must be '1'**.**
MULTIPLXER: A multiplexer is a combinatorial circuit that is given a certain number (usually a power of two) data inputs, let us say 2n, and n address inputs used as a binary number to select one of the data inputs. The multiplexer has a single output, which has the same value as the selected data input. In the present design an 8x1 Multiplexer has been used for each single bit out depending on input needed to send out. This also ensures a faster ALU as the functions are performed as soon as the input is fed to the ALU. [3]

The 3 control signals (S [2-0]) show the desired output in the order as shown below:

000→	ADDITION
001→	AND
010→	NAND
011→	OR
100→	NOR
101→	XOR
110→	XNOR
111→	SUBTRACTION

FOUR BIT ALU: Each of the single bit ALU are cascaded to form a four bit ALU. Figure 2. shows detailed four bit ALU block diagram

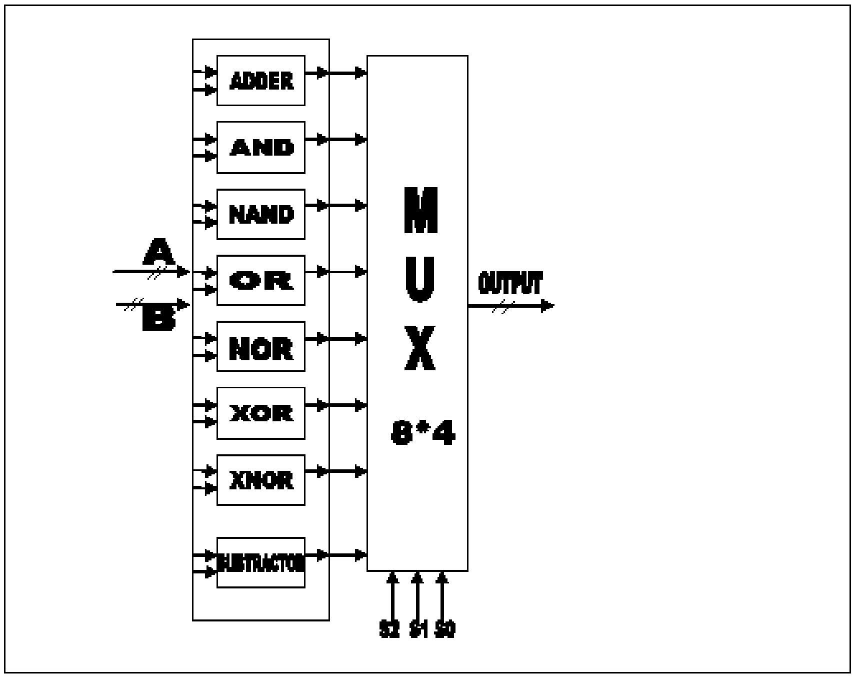


Figure 2: Detailed four bit ALU block diagram

PSpice Design Methodology

OrCAD productivity tools are designed to boost productivity for smaller design teams and individual printed circuit board (PCB) designers. The powerful, tightly integrated PCB design suites include design capture, librarian tools, a PCB editor, an auto/interactive router, and optional analog and mixed-signal simulator. The affordable, high-performance OrCAD product line is easily scalable with the full complement of Cadence Allegro PCB solutions. The OrCAD product line is owned by Cadence Design Systems, Inc. and supported by a worldwide network.

Digital Primitives

Digital primitives are primarily used in sub-circuits to model complete devices. Stimulus devices are used in the circuit to provide input for other digital devices during the simulation. Digital primitives are low-level devices whose main use is modeling off-the-shelf parts, often in combination with each other. Digital primitives should not be confused with the sub-circuits in the libraries that use them in the present design stimulus behavior has been used in creating an equivalent model of the four bit CMOS ALU[4].

Behavioral Primitives

The simulator offers three primitives to aid in the modeling of complex digital devices: the Logic Expression, Pin-to-Pin Delay, and Constraint Checker primitives. These devices are distinct from other primitives in that they allow data-sheet descriptions to be specified more directly, allowing a one-to-one correspondence using the function diagrams and timing specifications. The Logic Expression primitive, LOGICEXP, uses free-format logic expressions to describe the functional behavior device. The Pin-To-Pin Delay primitive, PINDLY, describes propagation delays using sets of rules based on the activity on the device inputs. Each of the stimulus behavior parts are described in detail below:

- Device format

```
U<name> LOGICEXP (<no. of inputs>, <no. of outputs>)
+ <digital power node> <digital ground node>
+ <input node 1> ... <input node n>
+ <output node 1> ... <output node n>
+ <timing model name>
+ <I/O model name>
+ [IO_LEVEL = <value>]
+ [MNTYMXDLY = <value>]
+ LOGIC:
+    <logic assignment>*
```

- Timing Device Format

```
MODEL <timing model name> UGATE [model parameters
```

- Arguments and options

1. LOGIC Marks the beginning of a sequence of one or more <logic assignments>. A <logic assignment> can have one of the two following forms:
<Output node> = {<logic expression >}
<temporary value> = {<logic expression>}One of the output node names as it appears in the interface list.
Assignments to an <*output node*> causes the result of the <*logic expression*> to be scheduled on that output pin. Each <*output node*> must have exactly one assignment

Any target of an assignment which is not specified as one of the nodes attached to the device defines a temporary variable. Once assigned, <*temporary values*> can be used inside subsequent <*logic expressions*>. They are provided to reduce the complexity and improve the readability of the model. The rules for node names apply to <t*emporary value*> names[5].

- Logic Expression Operators

A C-like, infix-notation expression that returns one of the five digital logic levels. Like all other expressions, <*logic expressions*> must be surrounded by curly braces { }. They can span one or more lines using the + continuation character in the first column position. The logic operators are listed below from highest-to-lowest precedence

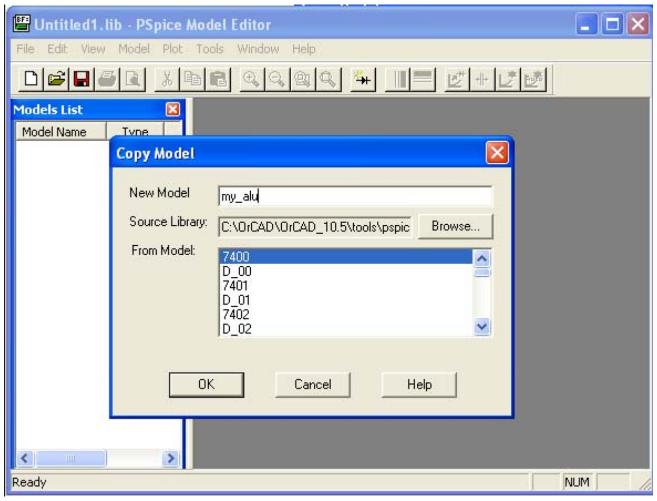
~	Unary not
&	and
^	Exclusive or
|	Or

PSPICE MODEL EDITO

The Model Editor is used either to generate a new model or edit an existing model to create a new model [7]. To generate a new model the method below has to be followed. Figure 3. shows PSpice model editor and Figure 4. shows model editor with new model creation.

a. From the File menu in Model Editor, choose mew.

- Using model menu from model menu chose copy from


Untitled1.lib - PSpice Model Editor
File Edit View Model Plot Tools Window Help
Models List
Model Name
Type
Copy Model
OK
Cancel
Help
Browse...
Ready
NUM

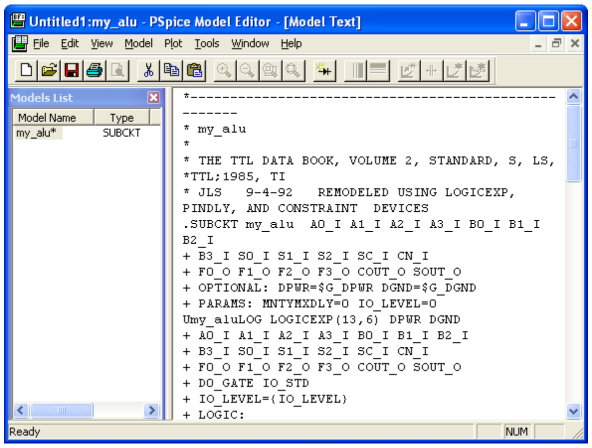
New Model: my_alu

Source Library: C:\OrCAD\OrCAD_10.5\tools\pspic

From Model: 7400, D_00, 7401, D_01, 7402, D_02

Figure 3: PSpice model editor

b. Select the any *Model* from the source library.
c. Click OK
d. Manually type in the behavioral or digital primitives of the device to be modeled


Untitled1:my_alu - PSpice Model Editor - [Model Text]
File Edit View Model Plot Tools Window Help
Models List
Model Name
Type
my_alu*
SUBCKT
Ready
NUM

```
*-----------------------------------------------
-------
* my_alu
*
* THE TTL DATA BOOK, VOLUME 2, STANDARD, S, LS,
*TTL;1985, TI
* JLS   9-4-92   REMODELED USING LOGICEXP,
PINDLY, AND CONSTRAINT  DEVICES
.SUBCKT my_alu  A0_I A1_I A2_I A3_I B0_I B1_I
B2_I
+ B3_I S0_I S1_I S2_I SC_I CN_I
+ F0_O F1_O F2_O F3_O COUT_O SOUT_O
+ OPTIONAL: DPWR=$G_DPWR DGND=$G_DGND
+ PARAMS: MNTYMXDLY=0 IO_LEVEL=0
Umy_aluLOG LOGICEXP(13,6) DPWR DGND
+ A0_I A1_I A2_I A3_I B0_I B1_I B2_I
+ B3_I S0_I S1_I S2_I SC_I CN_I
+ F0_O F1_O F2_O F3_O COUT_O SOUT_O
+ D0_GATE IO_STD
+ IO_LEVEL={IO_LEVEL}
+ LOGIC:
```

Figure 4: model editor with new model creation

e. Save the file as ".lib".

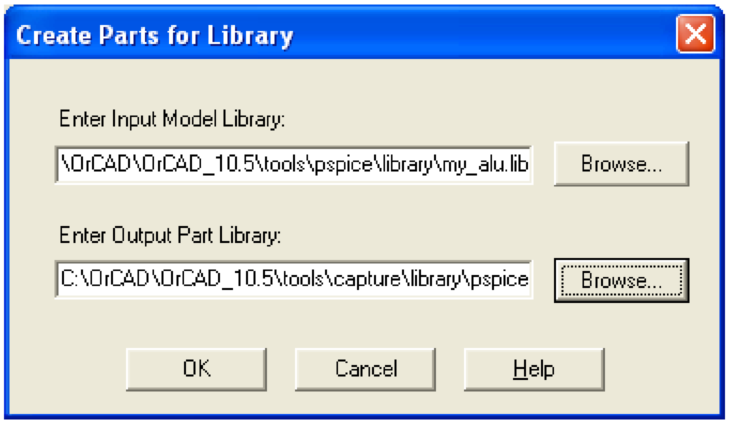
Exporting the Model to Capture Library

The Model Editor is used import the model into capture. To generate a new model in capture the method below has to be followed.

- Using File menu

a. From the file menu in Model Editor, choose export to capture part library.

Figure 5. shows PSpice export to capture part library.


Create Parts for Library

Enter Input Model Library: \OrCAD\OrCAD_10.5\tools\pspice\library\my_alu.lib

Enter Output Part Library: C:\OrCAD\OrCAD_10.5\tools\capture\library\pspice

Figure 5: PSpice export to capture part library

B. Figure 6. show the path for importing from PSpice to capture library.

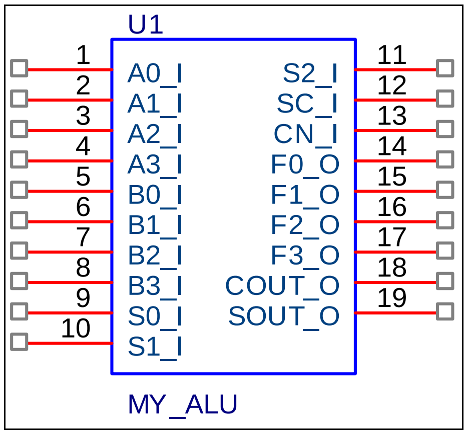


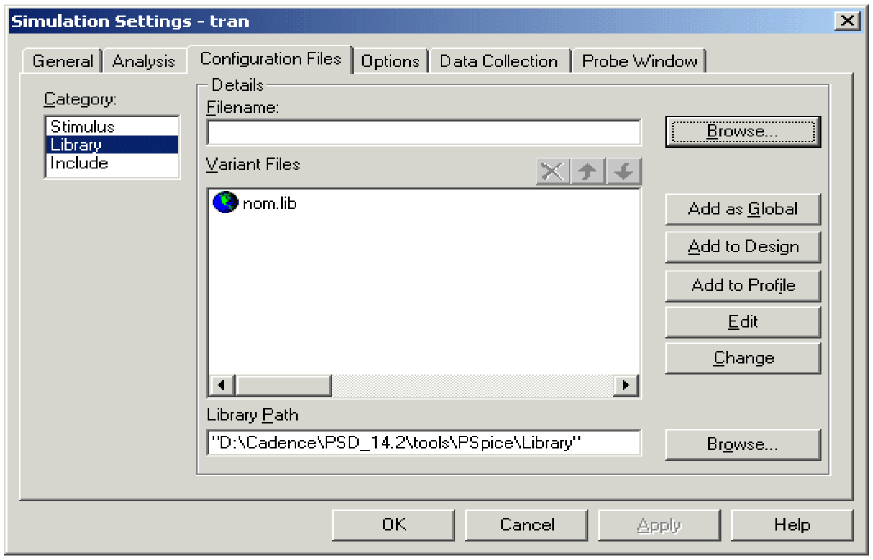
Figure 6 ALU part created in capture

Note: Capture automatically creates the part in capture

Configuring New Model Library

After generated the part library has been generated for a new/customized model library, the model library must be made available to the design. To ensure this the model library containing custom simulation models is added to the project simulation profile.

1. In Capture, open Analog or Mixed-Circuit project.
2. From the *PSpice* menu choose *Edit Simulation Profile*.
3. Select the *Configuration Files* tab.
4. In the *Category* list box, select *Library*.
5. In the Filename text box, specify the location of the model library.
6. To make the library available to all designs, click *Add as Global*. If you want the library to be used only in the current design, select *Add to Design* and close the *Simulation Settings* dialog box.
7.



Library Path: "D:\Cadence\PSD_14.2\tools\PSpice\Library"

Figure 7: Defining a global parameter

Note: Instead of editing a simulation profile, you can also create a new simulation profile. To do this, choose *New Simulation Profile* from the *PSpice* menu in Capture. Figure 7 show how to define a global parameter.

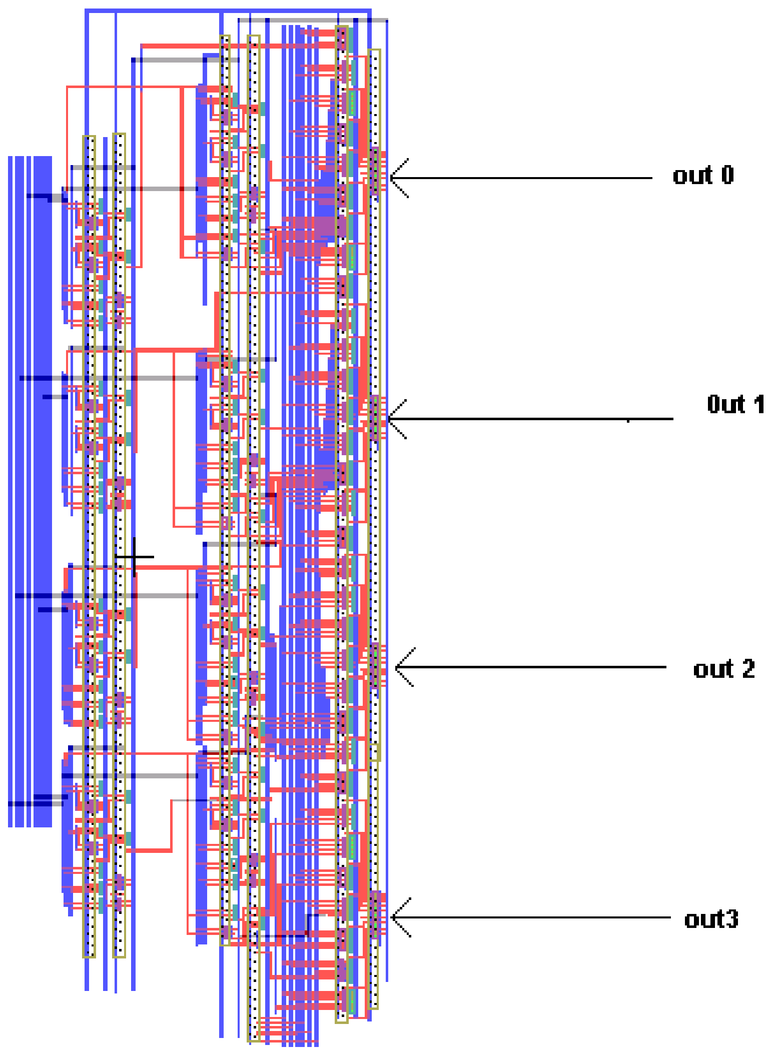


Figure 8: CMOS Four Bit ALU Layout

Figure 8. shows CMOS Four Bit ALU Layout.

ALU Measurements

1. TOTAL NO. OF DEVICES : 982 MOS
2. APPROXIMATE AREA OF THE ALU1431385.2 sqr microns
3. RISE TIME :10.13 nanoseconds
4. FALLTIME : 20.08 nanoseconds
5. TPLH : 11.53 nanoseconds
6. TPHL : 13.37 nanoseconds
7. PROPAGATION DELAY: 12.45 nanoseconds
8. SKEW: 48.

Table 1: Illustration of ALU functions

A INPUT	B INPUT	FUNCTION (CONTROL SIGNAL)	OUTPUT
1010	0100	ADD(000)	1110
1010	0100	AND(001)	0000
1010	0100	NAND(010)	1111
1010	0100	OR(011)	1110
1010	0100	NOR(100)	0111
1010	0100	XOR(101)	1110
1010	0100	XNOR(110)	0001
1010	0100	SUB(111)	0110

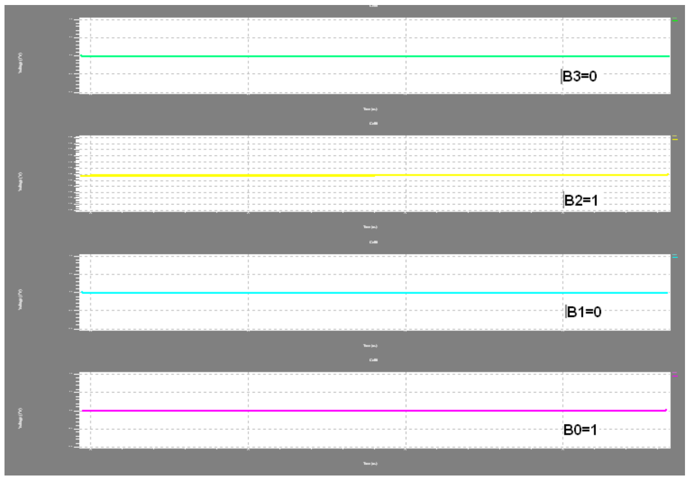


Figure 9: T-SPICE B inputs of a four bit ALU

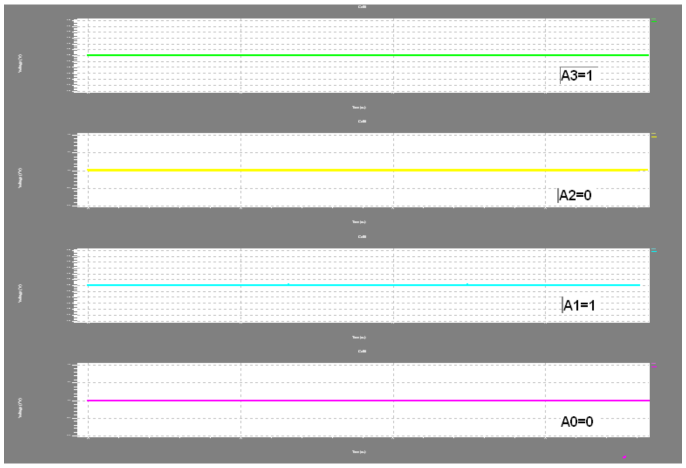


Figure 10: T-SPICE inputs of a four bit ALU

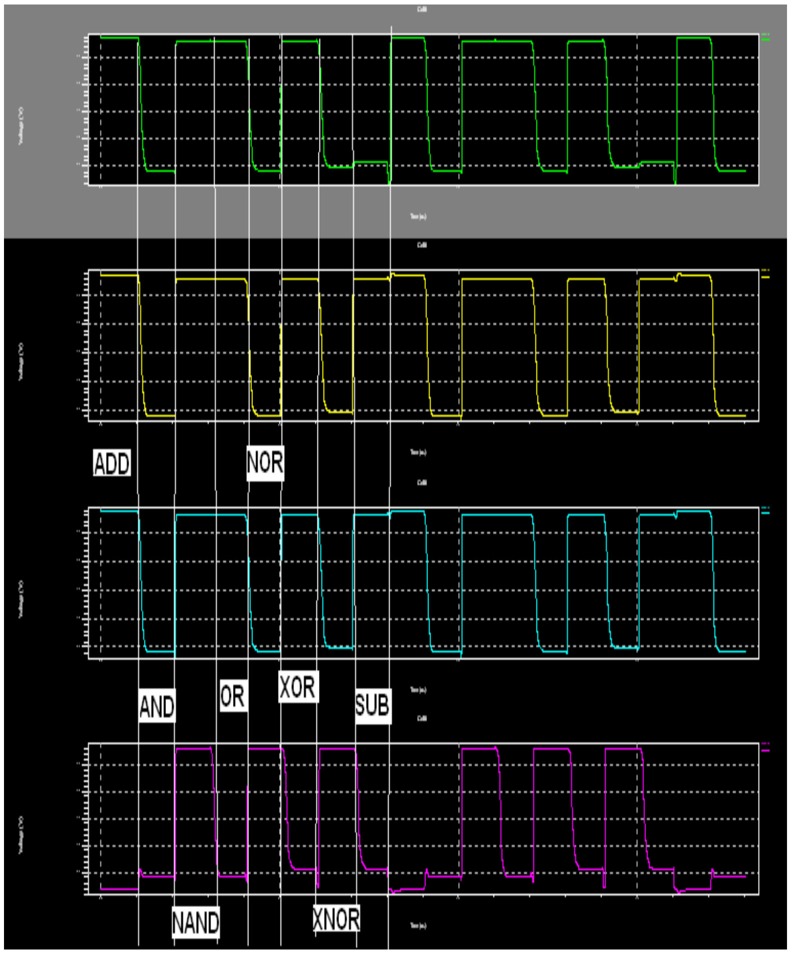


Figure 11: Outputs of four bit ALU for all functions

Table 1., Figure 9., 10 and 11 demonstrate functionality and input/output timing diagrams.

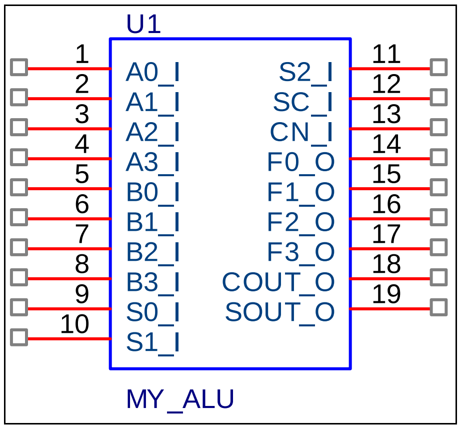


Figure 12: ALU in Capture with pin layout

Figure 12. shows ALU symbol in OrCAD capture.

PSpice Model Editor Codes

PSpice codes are demonstrated below.

```
.SUBCKT my_alu A0_I A1_I A2_I A3_I B0_I B1_I B2_I
+ B3_I S0_I S1_I S2_I SC_I CN_I
+ F0_O F1_O F2_O F3_O COUT_O SOUT_O
+ OPTIONAL: DPWR=$G_DPWR DGND=$G_DGND
+ PARAMS: MNTYMXDLY=0 IO_LEVEL=0
Umy_aluLOG LOGICEXP (13, 6) DPWR DGND
+ A0_I A1_I A2_I A3_I B0_I B1_I B2_I
+ B3_I S0_I S1_I S2_I SC_I CN_I
+ F0_O F1_O F2_O F3_O COUT_O SOUT_O
+ D0_GATE IO_STD
+ IO_LEVEL= {IO_LEVEL}
+ LOGIC:
+   A0   = {A0_I}
+   A1   = {A1_I}
+   A2   = {A2_I}
+   A3   = {A3_I}
+   B0   = {B0_I}
+   B1   = {B1_I}
+   B2   = {B2_I}
+   B3   = {B3_I}
+   S0   = {S0_I}
+   S1   = {S1_I}
+   S2   = {S2_I}
+   CN   = {CN_I}
+   SC   = {SC_I}
*
* Intermediate terms:
```

```
*
*LOGIC OUTPUT OF F0
+
I01={((~S2&~S1&~S0)&((A0^B0)^CN))|((~S2&~S1&S0)&(A0&B0))|((~S2&S1&~S0)
&~(A0&B0))|((~S2&S1&S0)&(A0|B0))}
+
I02={((S2&~S1&~S0)&~(A0|B0))|((S2&~S1&S0)&(A0^B0))|((S2&S1&~S0)&~(A0^B
0))|((S2&S1&S0)&(A0^~B0^SC))}
+ F0_O= {(I02|I01)}
*LOGIC OUTPUT OF F1
+ I11 = {(A0&B0)| (B0&CN)| (CN&A0)}
+ I12 = {(A0&~B0)| (~B0&SC)| (SC&A0)}
+I13={((~S2&~S1&~S0)&((A1^B1)^I11))|((~S2&~S1&S0)&(A1&B1))|((~S2&S1&~S
0)&~(A1&B1))|((~S2&S1&S0)&(A1|B1))}
+I14={(S2&~S1&~S0&~(A1|B1))|((S2&~S1&S0)&(A1^B1))|((S2&S1&~S0)&~(A1^B
1))|((S2&S1&S0)&(A1^~B1^I12))}
+ F1_O = {I14|I13}
*LOGIC OUTPUT OF F2
+ I21 = {(A1&B1)| (B1&I11)| (I11&A1)}
+ I22 = {(A1&~B1)| (~B1&I12)| (I12&A1)}
+I23={((~S2&~S1&~S0)&(A2^B2^I21))|((~S2&~S1&S0)&(A2&B2))|((~S2&S1&~S0)
&~(A2&B2))|((~S2&S1&S0)&(A2|B2))}
+I24={((S2&~S1&~S0)&~(A2|B2))|((S2&~S1&S0)&(A2^B2))|((S2&S1&~S0)&~(A2^
B2))|((S2&S1&S0)&(A2^~B2^I22))}
+ F2_O = {I24|I23}
*LOGIC OUTPUT OF F3
+ I31 = {(A2&B2)| (B2&I21)| (I21&A2)}
+ I32 = {(A2&~B2)| (~B2&I22)| (I22&A2)}
+I33=((~S2&~S1&~S0)&(A3^B3^I31))|((~S2&~S1&S0)&(A3&B3))|((~S2&S1&~S0)&
~(A3&B3))|((~S2&S1&S0)&(A3|B3))}
+
I34={((S2&~S1&~S0)&~(A3|B3))|((S2&~S1&S0)&(A3^B3))|((S2&S1&~S0)&~(A3^B
3))|((S2&S1&S0)&(A3^~B3^I32))}
```

PSPICE ALU WIRING

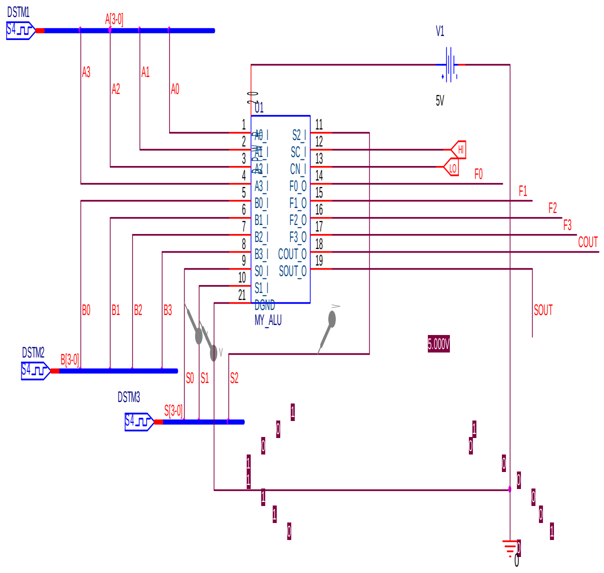


Figure 13: ALU simulation in PSpice capture

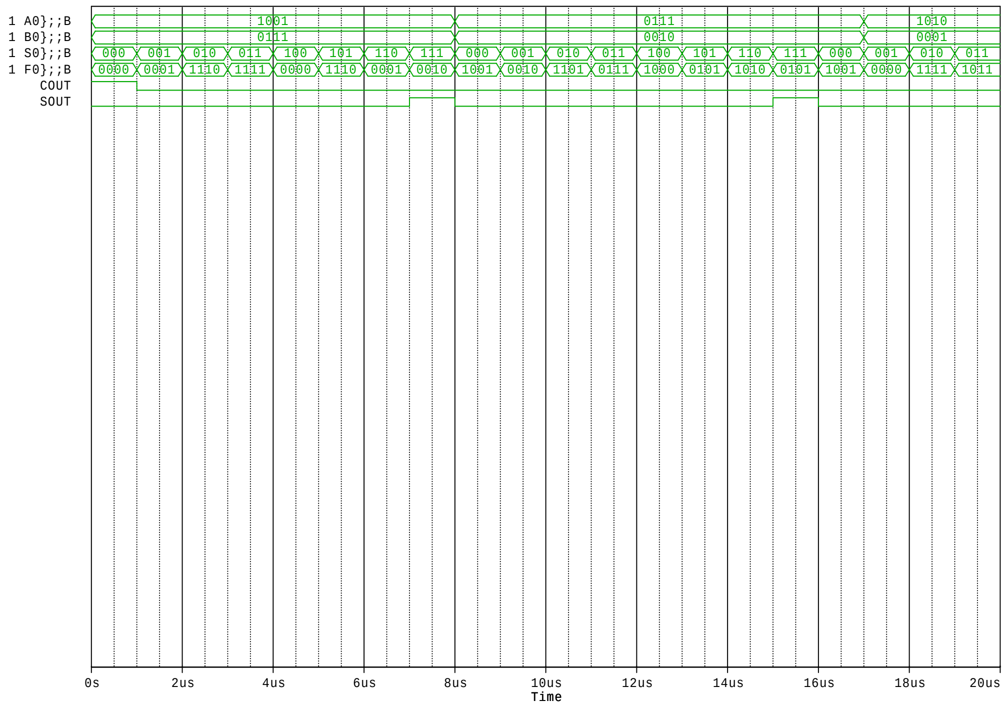


Figure 14 ALU simulation outputs in PSpice

Figure 13. shows the schematic and Figure 14 shows input/output timing diagram.

Conclusion

To stay competitive in today's market, engineers must take a design from engineering through manufacturing with shorter design cycles and faster time to market. To be successful, you need a set of powerful, intuitive, and integrated tools that work seamlessly across the entire design flow [6].

OrCAD personal productivity tools have a long history of addressing these demands-and more. And with the technique described above it makes Digital Design a mere child's play. It helps designer create and test different design of his choice with even touching a piece of hardware. The present paper outlines a simple but an important method of designing digital devices in OrCAD PSpice. The method is better explained with the help of an eight instruction set four inputs ALU.

References

[1] Behrooz Vahidi, *Senior Member, IEEE,* and Jamal Beiza "Using PSpice in Teaching Impulse Voltage Testing of Power Transformers to Senior Undergraduate Students" in IEEE TRANSACTIONS ON EDUCATION, VOL. 48, NO. 2, MAY 2005

[2] L. Puglisi, P. Ferrari, P. Tenca, and A. Rebora, "An advanced application of PSpice modeling and simulation for design optimization of push-pull dc/dc converter," in *Proc. 7th Inst. Elect. Eng. Int. Conf. Power Electronics Variable Speed Drives,* Genoa, Italy, Sep. 1998, pp. 117–120.

[3] William Gerard Hurley, *Senior Member, IEEE,* and Chi Kwan Lee "Development, Implementation, and Assessment of a Web-Based Power Electronics Laboratory in IEEE TRANSACTIONS ON EDUCATION, VOL. 48, NO. 4, NOVEMBER 2005.

[4] Stephen Prigozgy, senior IEE E member "Novel application of Spice in engineering education" in IEEE vol for education ,vol 32,No 1 Feb. 1989

[5] J. M. Deskur, "PSpice simulation of power electronic and motion control systems," *Proc. IEEE Int. Symp. Industrial Electronics*, pp. 195–200,Jul. 1997.

[6] K. T. Chau and C. C. Chan, "A Spice compatible model of permanent magnet dc motor drives," in *Proc. 1995 IEEE Int. Conf. Power Electronics and Drive Systems,* vol. 1, Kowloon, Singapore, Feb. 1995, pp.477–482.

[7] OrCAD PSpice help manual.

Biography

Saeid Moslehpour is an Assistant Professor in the Electrical and Computer Engineering Department in the College of Engineering, Technology, and Architecture at the University of Hartford. He holds PhD (1993) from Iowa State University and Bachelor of Science, Master of Science (1990), and Education Specialist (1992) degrees from University of Central Missouri. His research interests include logic design, CPLDs, FPGAs, electronic system testing and distance learning.

Srikrishna Karatalapu received his Bachelor of Science degree in electrical engineering in Hyderabad, India and Master of Engineering degree from the University of Hartford in 2006.