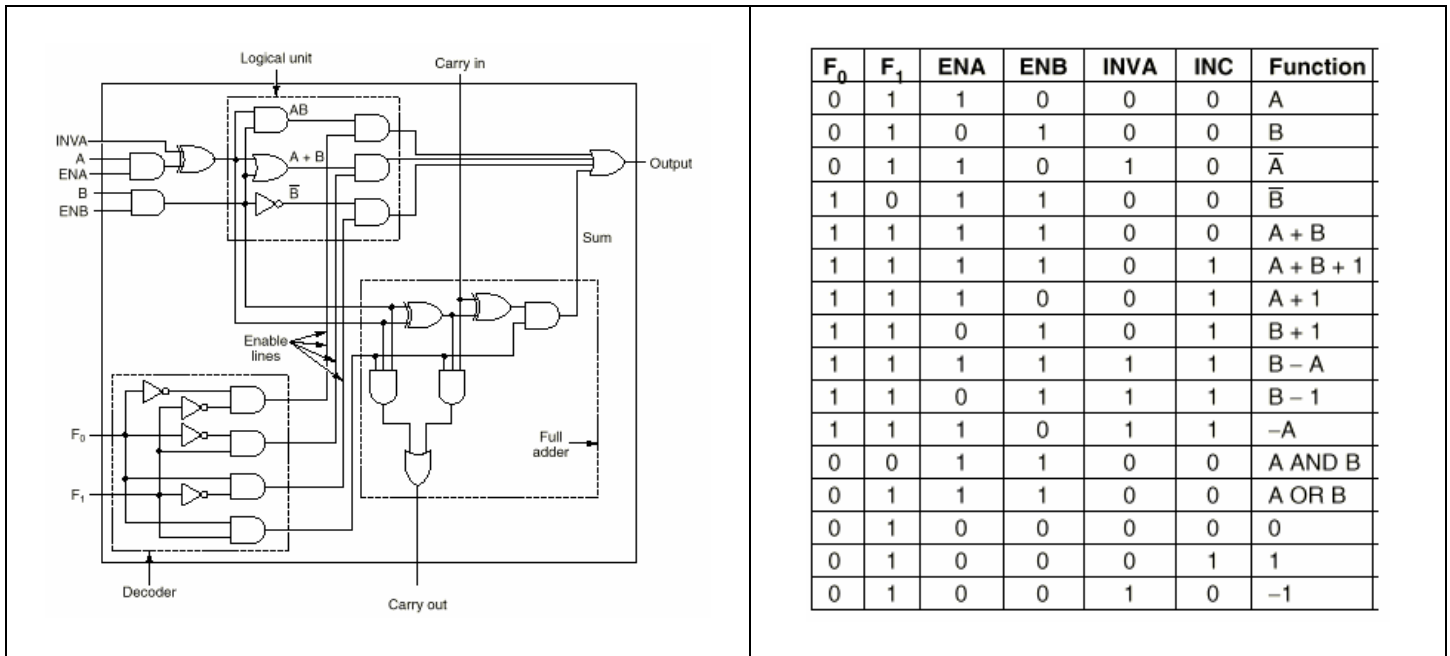


CSCE 491 – Capstone Computer Engineering Project
Homework Assignments #1 and #2
Arithmetic Logic Unit (ALU) Design

The design for this ALU comes from the Tanenbaum, 4th ed., *Computer Architecture*, text. The gate-level schematic of the circuit is shown in Figure 1, and the truth table representation of the functionality to be supported in your design is shown in Figure 2.



Figures 1 & 2. The 1-bit ALU design specification.

We will be developing various design models of this combinational logic function block using the algorithmic state machine (ASM) method for modeling and designing finite state machines and their arithmetic data paths.

HW #1: Editing, Compiling, Simulating and Printing a Design Model

For this assignment, you will take the provided ASM design file printout, **HW-01.pdf**, and use it as your guide for entering your own version of the design. You will each be responsible for using *flowHDL* to enter this design as shown in the attached diagrams in the following figures. The design entry and verification consists of the following parts:

- (1) Declare the signals and buses in the Bus Table, as shown in Figure 3, with one minor difference. You'll need to declare all buses to have Element type = "wire" instead of "context". (Note we are starting with a single-bit ALU unit in this assignment). The dialogue box for creating and modifying signals and buses in your design is shown in Figure 4. You access this through the Tables -> Bus Table pull down menu item.

Code	Bus Name	Type	Mode	Element	Default Value	Value	Width
	F0	Binary	Input	Wire	0	0	1
	F1	Binary	Input	Wire	0	0	1
	F0F1	Binary	Internal	Wire	0	0	2
	A	Binary	Input	Wire	0	0	1
	B	Binary	Input	Wire	0	0	1
	Carry_in	Binary	Input	Wire	0	0	1
	Carry_out	Binary	Output	Wire	0	0	1
	ENA	Binary	Input	Wire	0	0	1
	ENB	Binary	Input	Wire	0	0	1
	INVA	Binary	Input	Wire	0	0	1
	Out_1	Binary	Output	Wire	0	0	1
	Sum	Binary	Internal	Wire	0	0	1
	log_out1	Binary	Internal	Wire	0	0	1
	log_out2	Binary	Internal	Wire	0	0	1
	log_out0	Binary	Internal	Wire	0	0	1

Figure 3. Bus Table signal list.

Bus Definition

Bus Name:

Bus Mode Option:

- Input
- Output
- Internal
- Internal Output
- Bidirectional

Bus Type Option:

- Binary
- MVL3
- MVL4
- MVL7
- MVL9
- Enumerated

Bus Width Option:

- 1 bit
- 2 bits
- 4 bits
- 8 bits
- 16 bits
- Other Width

Bus Width:

Bus Element Option:

- Register
- Latch
- Wire
- Context
- Asynchronous Register

**** Hexadecimal Format ****

Default Value:

Enumerated List:

Default Value List:

Figure 4. Bus Table signal entry dialog box.

Note: you will **not** enter declarations for CLK and RES signals, since they are provided by default in *flowHDL*. Although this is a combinational logic design unit, with no state machine to be clocked, we'll still need the clock and reset signals for our output register from the ALU.

- (2) Set the specific parameters for the clock signal using the Clocking Scheme. Set the cycle time to 10 ns. This is available through the 'Tables' pull-down menu in *flowHDL*. Select the 'Clocking Scheme' button to display the dialog box, as shown in the next figure.

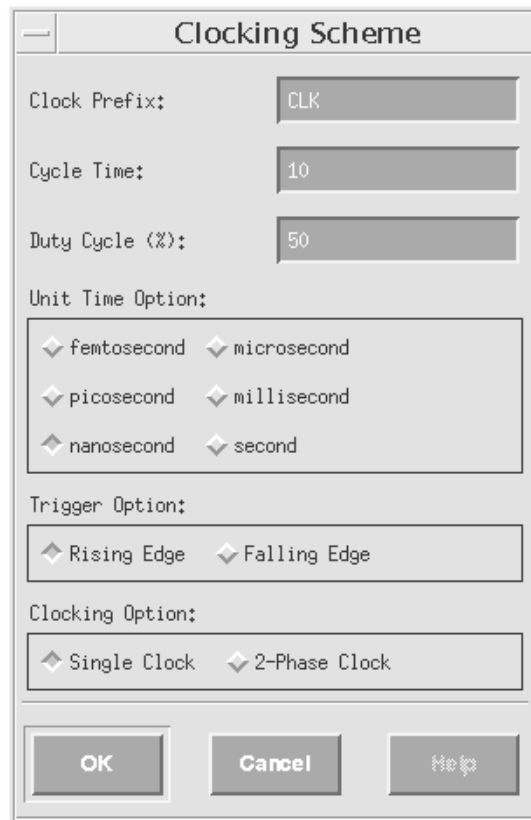


Figure 5. Clocking specification entry dialog box.

- (3) Draw the diagram for the ALU model, as shown in Figure 6 (also see HW01-491.pdf). You'll note that this model has two states, connected as shown in the figure. The operations defined for each state are indicated as assignment statements, including "nested" macro-functions, which are scheduled for each of the connected states. The set of statements below and to the left of the second state are the assignment expressions associated with the 2nd state.

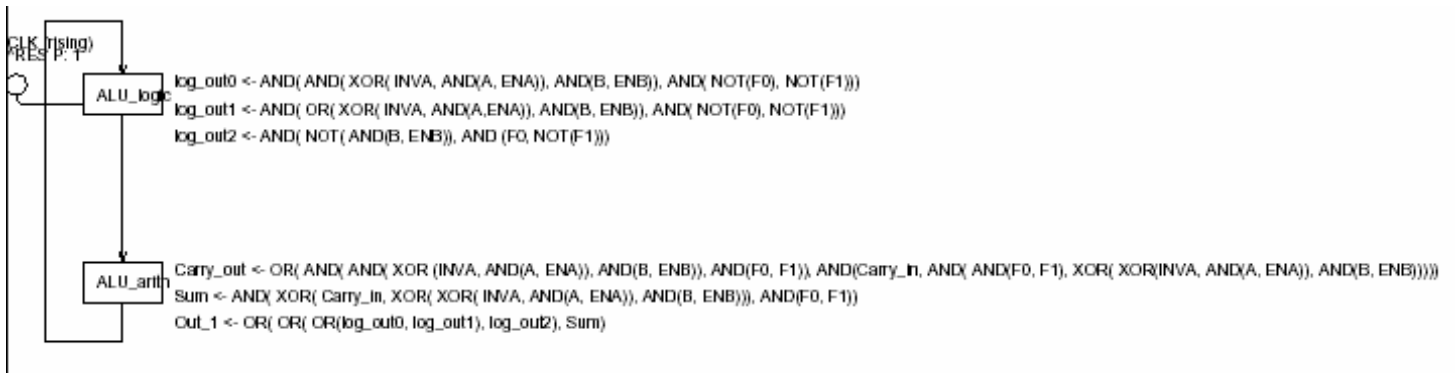


Figure 6. Two-state ASM thread for ALU.

- (4) Attempt to compile the model, by selecting the “Compile” menu option under the “Tools” menu. Once you compile the model, you should get a dialog box indicating that you have no errors.
- (5) If you get errors during the compilation, you should go to the state where errors are flagged (with a little “bug” symbol, and click the object while holding down the ‘CTRL’ key (so, CTRL + Left Mouse). This will bring up an explanation of the error. NOTE: make sure the “CAPS LOCK” key is not set.

On encountering an error, the messages will take some time to get used to, so if you see them, try to decipher what is being indicated as the problem. Usually, the cause of errors is that you are using signals in an assignment expression that have different bus widths, different signal types, or signal modes (*input*, *internal*, *internal_out*, versus *output*). Or, you may have a mismatch in the width of buses used as arguments to a macro and the bus that gets assigned the result. Or, the problem could be with parenthesis matching up. So, you may have to debug the design a bit. (However, the model does compile properly.)

- (6) Once you have compiled the model, you will want to verify its correctness using the *flowHDL* model simulator. By setting stimuli for the design in the Bus Table, you can check the model’s behavior by looking at the updated bus values in the Bus Table, or observe the signal values in the Waveform Viewer.

You need to select the “Tools -> Wave Viewer” to open the window for viewing waveforms. You need to select the “Tools -> Simulate” to open the simulator panel. Move the simulator panel to the upper left or right side of the screen.



Figure 7. Simulator cockpit.

To start simulation, you need to set a breakpoint on the reset state of the design. This is done by clicking on this state with the left mouse button. The state will change color (white) indicating that a breakpoint has been set. Then, you will click the “reset” button on the simulator control panel, as shown in Figure 8. This will start the simulation clock, and pull the reset line, simulating the system reset. Then, you will press the “step” button, advancing the simulation by one clock cycle.

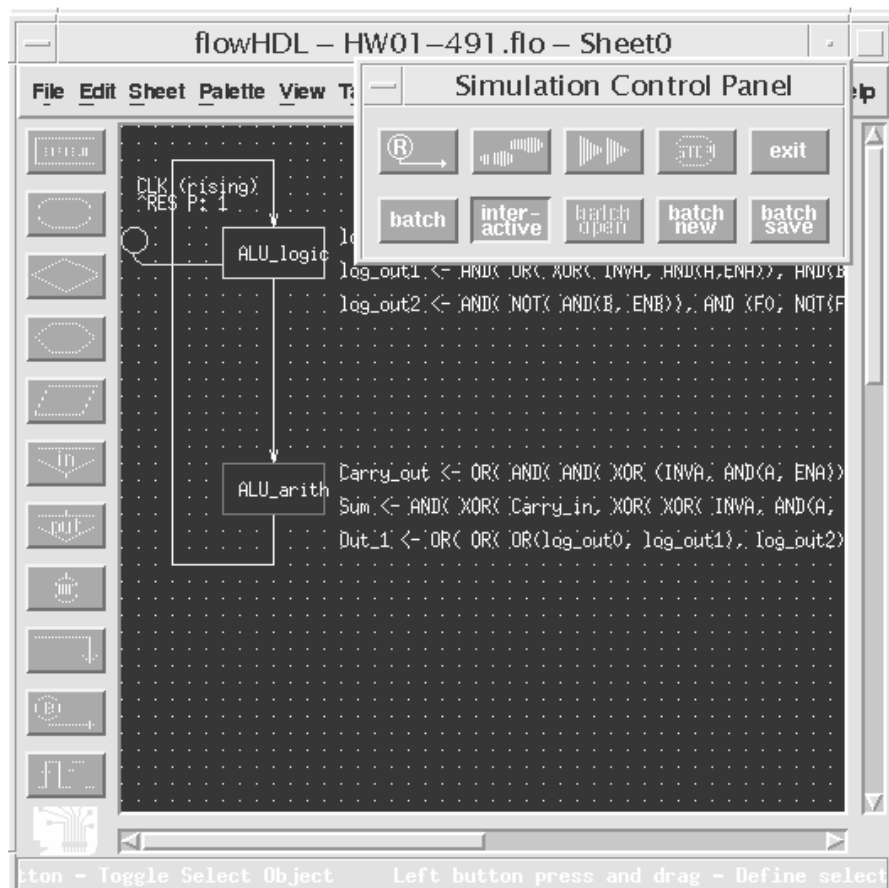


Figure 8. Canvas and Cockpit in Simulator mode.

Now, you can enter input values for (a) A and B operands, (b) ENA and ENB data input enable lines, (c) F0 and F1 function select lines, and (d) INV line, for a given test run. You will want to plan 4 separate test runs with these inputs, and write them into the Test Planning Worksheet template (see attached to this document or on the web page). After setting these values in the Bus Table, you will “step” through the design until the operation is completed. Figure 9 shows how these values are set in the Bus Table when flowHDL is in *simulation mode*.

After setting input values in the Bus Table, and as you “step” the simulation, you will see the waveform display advance along with the clock, showing the updated values of your signals. You will be looking for the specific output from the ALU,

and you will be interested in checking whether it is giving the correct result (given the operation and operands you specify in your test cases), and also whether it is giving the result in the time frame we are expecting it. (Remember, the ALU is supposed to be a combinational logic circuit, so we should see the results on the output in one clock cycle. The Wave Viewer display should look something like Figure 10. NOTE: you will need to come up with 4 separate test scenarios.

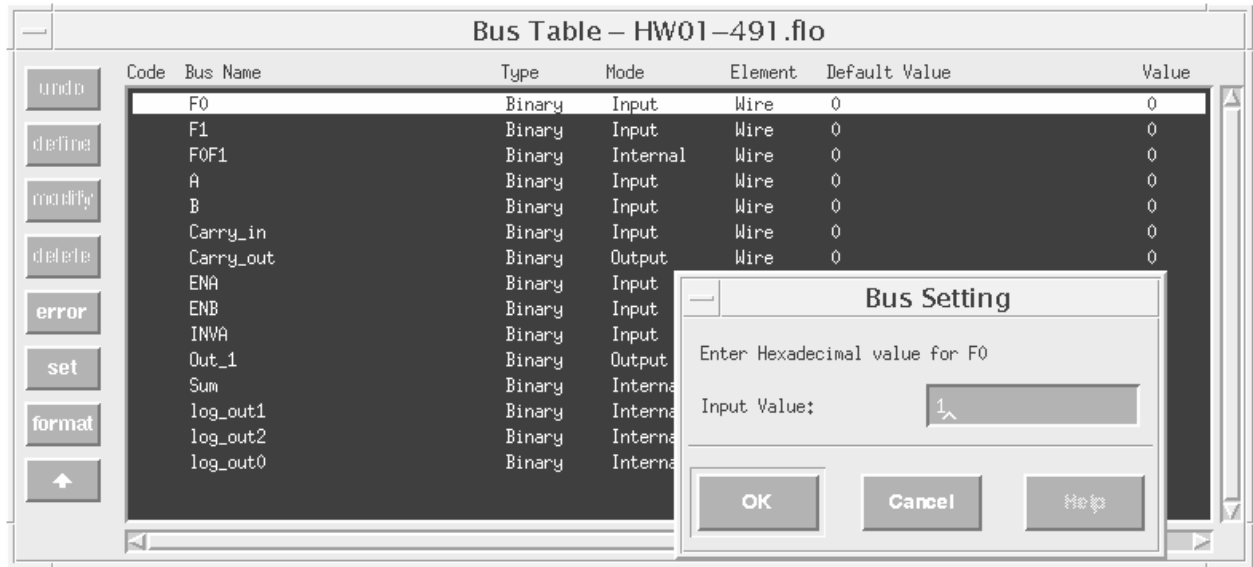


Figure 9. Entering simulation input values in the Bus Table in Simulation Mode.

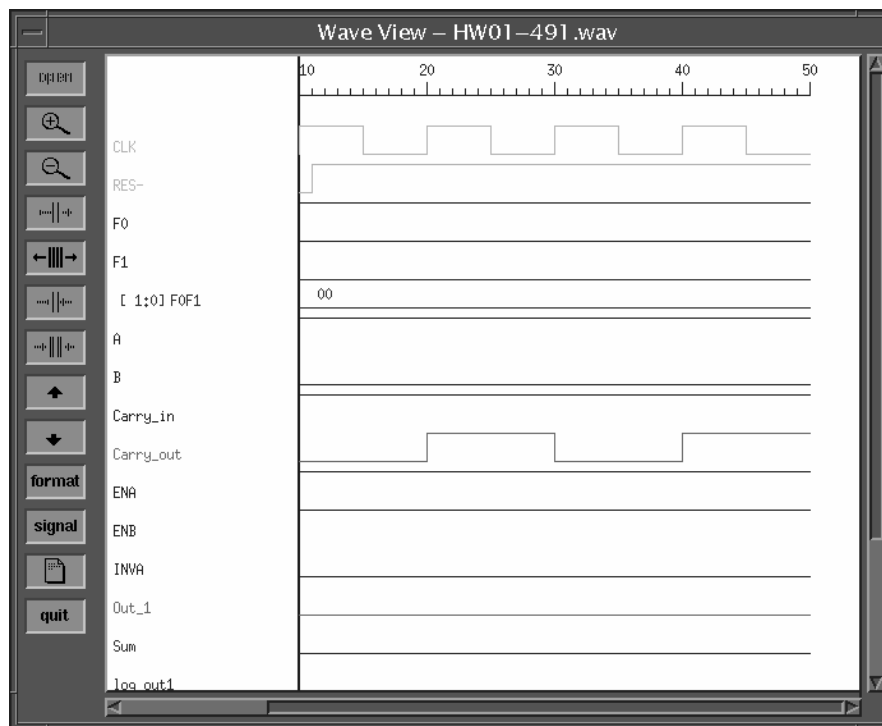


Figure 10. Simulation results viewed in the Wave Viewer window.

- (7) Now, once you get the model done, you'll want to do the following:
- (7a) enter your design information, through the “Options -> Design -> Information” menu dialog. Enter a new design name (so spaces or non-alphanumeric, except “_” underline character, are not permitted), and specify your name and date and design version (which is v1.0). Use the information as shown in Figure 11.
 - (7b) print your design, using the “File -> Print-> Output” menu dialog. The default printer is the one in 1D39. You should select to print the entire design workspace (which will print the Canvas sheets and the Bus Table). You'll probably want to set the format to “landscape” instead of portrait for this design, and also reduce the size of the image to be printed. This can be done using the “File -> Print -> Setup” dialog. To print the design, you select the “File -> Print -> Output”, which lets you print to the printer (“l39” in 1D39) or to a file (which can be printed from the *Ghostscript* application).
 - (7c) print your waveform, using the dialog box accessed from the Wave Viewer (the “piece of paper” icon on the tool panel of the Wave Viewer). This dialog is shown in Figure 12. Each time you simulate, you'll want to rename the “HW01.wav” file created after each simulation run and written into your working directory. Each waveform for each test will need to be printed out.

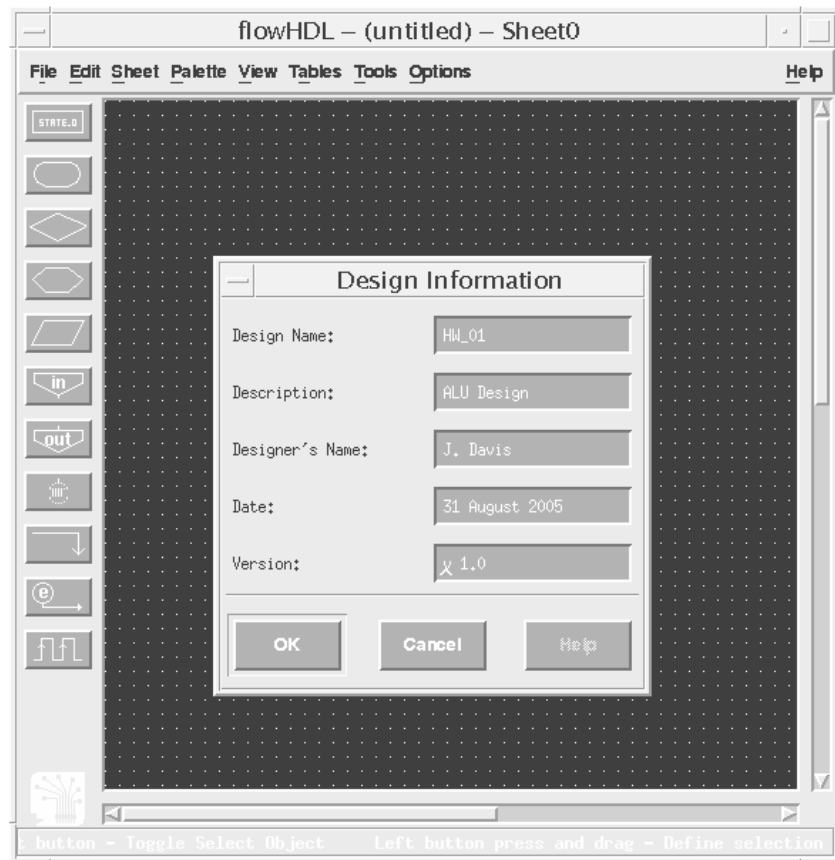


Figure 11. Design Information entry dialog box.

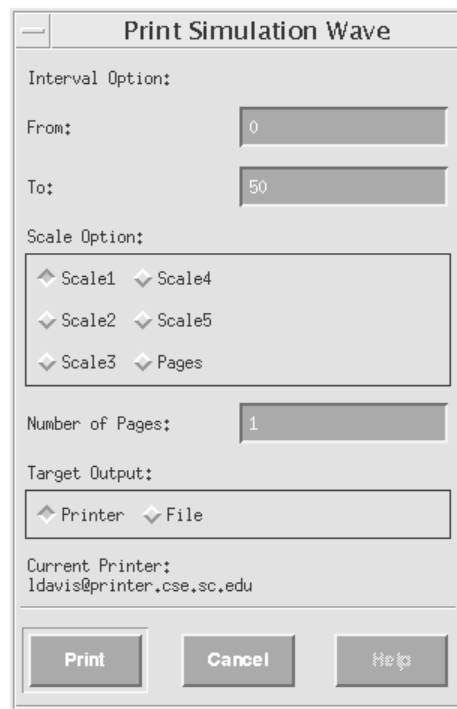


Figure 12. Waveform print dialog box.

- (8) Finally, you will want to save your model, using the “File -> Save” dialog box. Save the file as **HW01.flc**, and make sure you are writing it into your “csce491” directory (which you should create). In fact, you will want to periodically save your model after you start creating it in *flowHDL*. Just like most software, it has been known to crash. So, save yourself some aggravation, save your model often during the editing process.
- (9) **Remember:** You also need to preserve the waveform file after each simulation run and print each for inclusion in your assignment submission package. After you simulate each of your 4 test cases, you’ll want to rename the existing file, **HW01.wav**, to **HW01-1.wav** for simulation results from test run #1, rename the second one as **HW01-02.wav** for simulation results from test run #2, and so on.

Before making your assignment submission, you’ll want to check the Assignment Submission Guidelines, located on the course web page. Specifically, you will need to complete a cover page that has the Effort Distribution worksheet. I will want you to keep track of the time spent on each activity: (1) analysis & thinking about the problem, (2) editing, and (3) simulation & test planning. The other information we won’t use on these assignments.

Don’t forget to answer the questions for Assignment #1, attached on separate pages.

Also, you will need to submit your assignments electronically.

HW #2: Creating and Verifying an alternate Design Model

For this assignment, you will take the provided ASM design file printout, **HW-02.pdf**, and use it as your guide for entering your own version of the design, to be named as **HW-02.f1o**. Note that we will be using a different style of representing the combinational logic, using more of the control constructs of the ASM notation, rather than heavy use of nested macro functions.

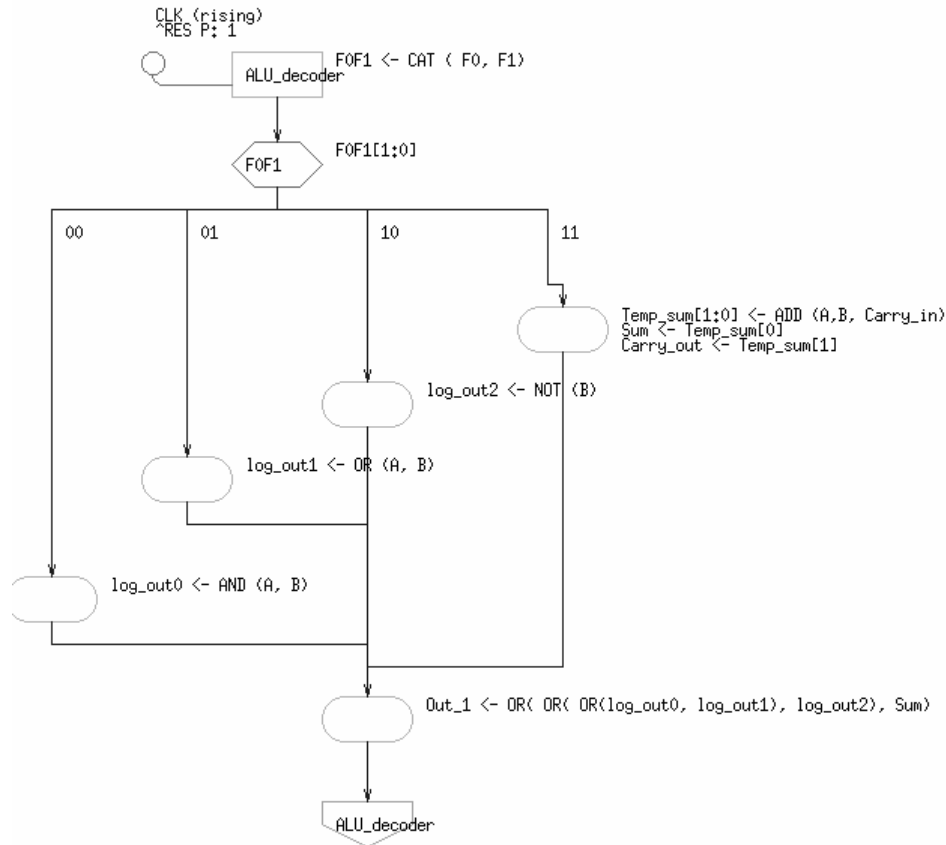


Figure 13. Alternate design model for ALU.

In this model variation, we represent the functionality of the ALU with a single state with a Case selecting among 4 branches, given the 2-bit concatenated signal “F0F1” that is initialized using the CAT() Macro-function. Each of the Case selections has a Mealy style conditional output, in which macro assignment expressions carry out the ALU’s computations. After each of the alternative function assignments, we have another conditional output that groups the 4 assignments, and chooses among them for assigning the final output.

Note that the bus declarations are the same for this model as for the one you created for HW #1, except that (1) we use bus “F0F1”, and (2) we define a new 2-bit signal, “Temp_out” that is used to assign the result of using the ADD macro (which we use

instead of the more primitive gate-level formulation for addition). See the Lecture Notes to see how the ADD Macro works.

You'll use the *flowHDL* editor to create the alternate ALU model, you'll check the design and compile it, and then you'll simulate it for 4 separate test cases to check for correct functionality and timing. You should use the same test data set that you created for HW #1.

This design doesn't meet the specification, as given in the schematic and truth table representations shown in Figures 1 and 2. You'll need to make some changes to this model and submit to modified version, **not** the original version you have created. You will submit the simulation waveforms for simulating the initial version of **HW02.f1o**, and the design model for the revised version, as **HW02-v2.f1o**, changing the *Design Information* in Options -> Design -> Information dialog, as shown for Assignment #1. This modified version will be submitted, along with simulations for the original and for the modified versions.

You will submit the same set of deliverables as for Homework Assignment #1. So, you'll need to keep the waveform files, naming them as **HW02-1.wav**, **HW02-2.wav**, etc., for simulation of the first version, and **HW02-v2-1.wav**, **HW02-v2-2.wav**, etc., for simulation of the modified version of the design.

As with Assignment #1, don't forget to answer the questions about your modeling activity in Assignment #2, given in the following pages, just as you have done in Assignment #1.

STATEMENT OF PERSONAL RESPONSIBILITY: You can discuss the use of the tool in carrying out this and other assignments, but please do not discuss the solution of the design problems—unless you are working with your team members—as this will be considered cheating). I need to give everyone opportunity to get comfortable with the tool and the design methods, and this is how we will do it.

CSCE 491 – Capstone Computer Engineering Project
Homework Assignments #1 & #2
Arithmetic Logic Unit (ALU) Design

Name / Date

You will turn this portion of the assignment in along with your design models, simulation output, Test Planning Worksheet, and Effort Distribution worksheet.

HW #1 Lab questions:

This design has two states, and operations scheduled on each state. Is there any problem with this design? Does it meet the function and timing requirements? Please explain.

If you see a problem with how the design has been carried out, how would you fix the problem? How might you model the design differently, or change the model as entered, so that the problem goes away?

Can you edit the existing model we have created, **HW-01.flo**, and modify the model so that it functions correctly? If you can, re-create the deliverables, naming the new file as

HW-01-v2.fl0, the waveform files as **HW-01-v2-1.wav**, **HW-01-v2-2.wav**, etc. Remember, you are going to use the same test data scenarios for this second model. This portion of the assignment, if you can complete it, is worth 5 additional bonus homework points.

HW #2 Lab questions:

This design has a single state, and uses a Case for the conditional logic to select the appropriate function to be carried out during each clock cycle. Is there any problem with this design model? Does it meet the function and timing requirements? If not, what is it missing? Please explain.

If you see a problem with how the design has been carried out, how would you fix the problems with the model? How might you model the design differently, or change the model as entered, so that the problems go away?

NOTE: As for HW #1, you need to make modifications to the model to overcome its problems and have it meet the specification as provided in Figs 1 & 2.

STATEMENT OF PERSONAL RESPONSIBILITY: You can discuss the use of the tool in carrying out this and other assignments, but please do not discuss the solution of the design problems—unless you are working with your team members—as this will be considered *cheating*. I need to give everyone opportunity to get comfortable with the tool and the design methods, and this is how we will do it.