

CSCE 491

Computer Engr. Design Project

2004/8/24

Lecture 3

Design Example – ALU Block

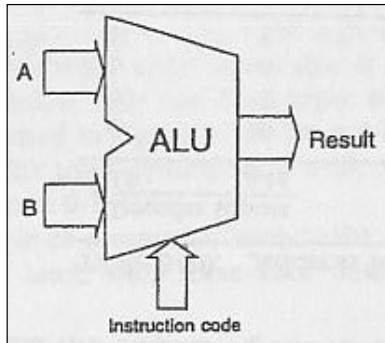
© 2003 Dr. James P. Davis

Lecture 3 - Outline

- Objective: To start some examples to give you more insight on:
 - How to start a design model from an analysis model
 - How to think about digital systems modeling using ASM charts.
- Means:
 - ✓ The ALU example (taken from Tanenbaum, 4th edition).
- Result:
 - You should be able to create this design in Nimbus for **HWs #1 and #2.**
 - You will generate a number of simulation test cases, indicating the “expected” behavior of the design. (Then, in Lab, we’ll learn how to run the simulator in Nimbus to verify the correctness of the design functionality).

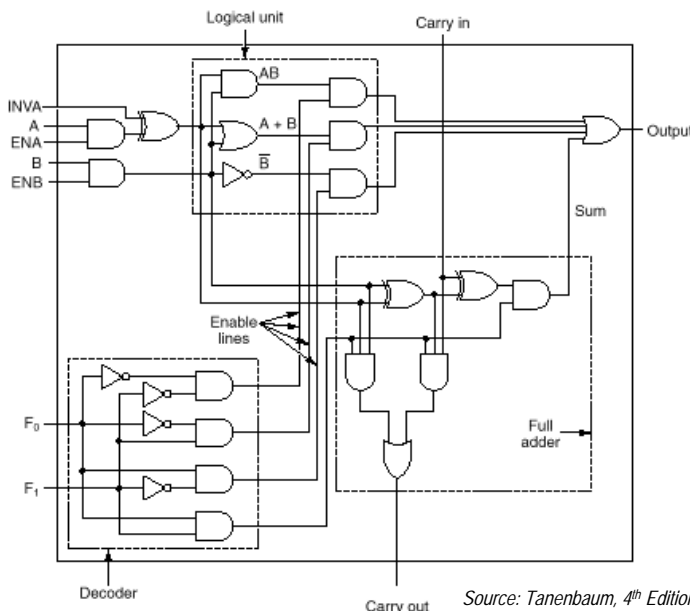
The Arithmetic Logic Unit

- The Arithmetic Logic Unit (ALU)



- ✓ The ALU provides many functions that are selectable using control signals.
- ✓ The ALU has an arithmetic circuit, a logic circuit, and decoding logic to determine which function to select.
- ✓ Only one ALU function can be selected at a time.
- ✓ ALUs are integral component of a CPU.

The Arithmetic Logic Unit



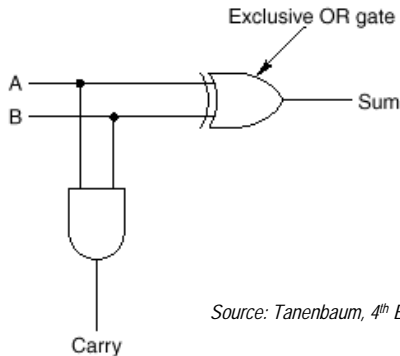
- 1-bit Arithmetic Logic Unit (ALU)

- ✓ The ALU function is provided as a single-bit operation, built up of gate-level combinational logic.
- ✓ The multi-bit ALU is created as a result of combining many single-bit ALUs in a cascaded configuration.
- ✓ This ALU can be analyzed by first constructing a truth table to obtain the Boolean equations.

Source: Tanenbaum, 4th Edition, 1999.

ALU Components – The Adder (from 212)

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



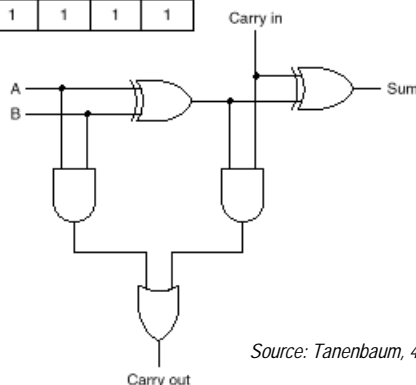
Source: Tanenbaum, 4th Edition, 1999.



- Half Adder circuit:
 - ✓ This circuit takes two single-bit inputs and adds them together to produce a Sum as output.
 - ✓ The adder also generates a Carry signal, which can be used to connect to another adder element.
 - ✓ Multiple adders are connected together to implement a multi-bit adder circuit, and more complex arithmetic functions.

ALU Components – The Adder (from 212)

A	B	Carry in	Sum	Carry out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

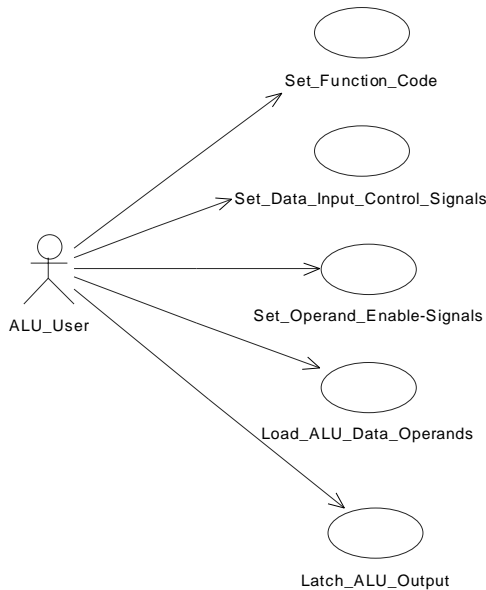


Source: Tanenbaum, 4th Edition, 1999.



- Full Adder circuit:
 - ✓ This circuit takes two single-bit inputs and adds them together to produce a Sum as output.
 - ✓ The Full Adder also has a Carry (called a Carry Out) like the Half Adder.
 - ✓ The Full Adder also has a Carry In signal, allowing Carry Out from earlier adder stage to be connected.
 - ✓ This allows a multi-bit, multi-stage adder circuit to be constructed.

The ALU Analysis Model – Use Cases



• Abstract ALU:

- ✓ This circuit takes two inputs and operates on them to produce an output.
- ✓ The output is determined by the selection of function codes, data input control, and operand enable signals.
- ✓ There is an implied ordering of tasks for setting up the ALU (not shown in the Use Case diagram): all the control signals must be stable before applying the data inputs; then the device is enabled.
- ✓ The ALU operates as a combinational logic device; so its output is latched into a register when the operation is complete.



The ALU Function Model – Truth Table

F ₀	F ₁	ENA	ENB	INVA	INC	Function
0	1	1	0	0	0	A
0	1	0	1	0	0	B
0	1	1	0	1	0	\bar{A}
1	0	1	1	0	0	\bar{B}
1	1	1	1	0	0	A + B
1	1	1	1	0	1	A + B + 1
1	1	1	0	0	1	A + 1
1	1	0	1	0	1	B + 1
1	1	1	1	1	1	B - A
1	1	0	1	1	1	B - 1
1	1	1	0	1	1	-A
0	0	1	1	0	0	A AND B
0	1	1	1	0	0	A OR B
0	1	0	0	0	0	0
0	1	0	0	0	1	1
0	1	0	0	1	0	-1

Inputs

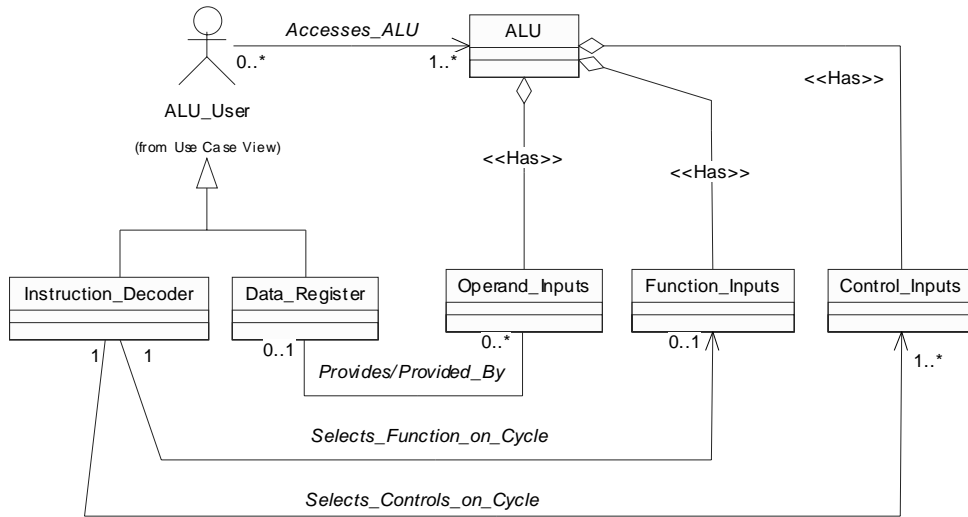
Output
Function

Functional specification:

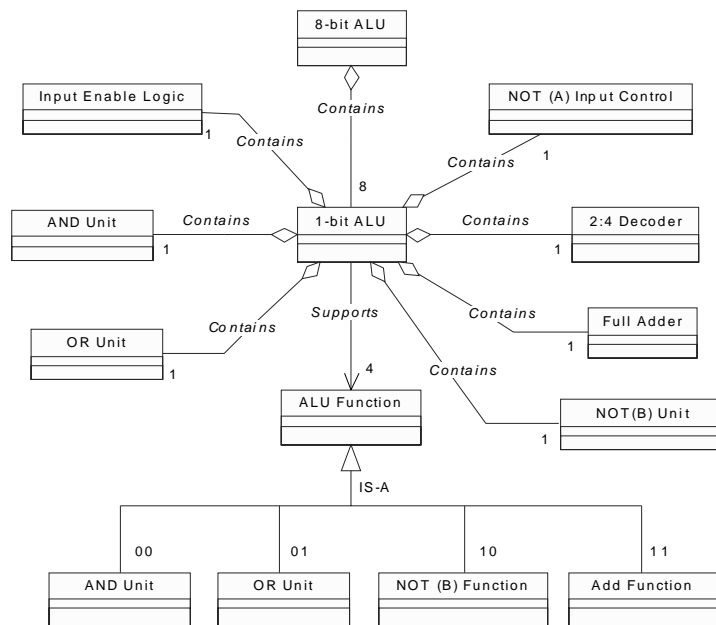
- ✓ This is a technique to represent a purely functional result.
- ✓ You list a column in the table for each input, and a column for each possible output combination.
- ✓ The table is used as a “look-up”, given some input combination, to determine the output.
- ✓ The outputs are a function of the inputs.
- ✓ Truth table results are derived based on rules of Boolean Logic, and complex functions can be built up from understanding more primitive ones.



The ALU Analysis Model – Class Diagram-1



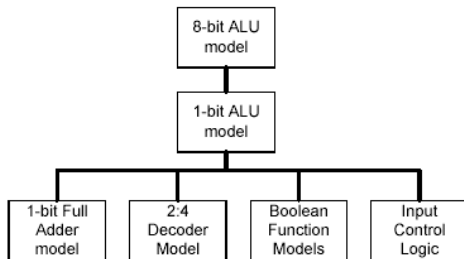
The ALU Analysis Model – Class Diagram-2



The ALU Design Model – ASM Diagram

Source: Tanenbaum, 4th ed. © 1999, Prentice-Hall

F ₀	F ₁	ENA	ENB	INVA	INC	Function
0	1	1	0	0	0	A
0	1	0	1	0	0	B
0	1	1	0	1	0	\bar{A}
1	0	1	1	0	0	\bar{B}
1	1	1	1	0	0	A+B
1	1	1	1	0	1	A+B+1
1	1	1	0	0	1	A+1
1	1	0	1	0	1	B+1
1	1	1	1	1	1	B-A
1	1	0	1	1	1	B-1
1	1	1	0	1	1	-A
0	0	1	1	0	0	A AND B
0	1	1	1	0	0	A OR B
0	1	0	0	0	0	0
0	1	0	0	0	1	1
0	1	0	0	1	0	-1



behavioral specification:

- ✓ There are several choices for the level at which we'll mode the control flow of the ALU.
- ✓ We could model it as one monolithic unit, where we simply implement all of the output combinations indicated in the truth table.
- ✓ Alternately, we could model the ALU hierarchically, as a collection of concurrent threads. Each thread implements a piece of the ALU (function decoding logic, adder, etc.).
- ✓ Which is easier to model?
- ✓ Which would be a more efficient design? (We'll have to generate a circuit to determine the answer.)
- ✓ We'll work both models and compare.

