University of Puerto Rico – Mayagüez Campus
School of Engineering

INEL 4206 – Microprocessors

Problem Set 1 – Combinational Computation

(Due: Tuesday February 11 by Midnight via electronic submit)

Part 1.a: One-Bit Arithmetic Logic Unit Building Block

Using combinational logic, design a one-bit ALU (Arithmetic Logic Unit) building block. The device will receive one bit for each of the two operands, perform the operation indicated by the "Operation Code" and put the result in the output pin and any resulting carry in the carry-out output signal. The ALU will only work if the enable input is set to '1', otherwise all outputs should be '0'.

Use text labels to identify the different parts of the design (where inputs and outputs go). After you finish the design, place binary switches at the inputs and binary probes at the outputs, so your design is ready for testing as soon as it is opened.

Here are the general specifications for the ALU you have to design:

Input:

- 1 bit enable
- 2 bits for instruction selection
- 1 bit for operand A
- 1 bit for operand B
- 1 bit for carry-input

Output:

- 1 bit for operation result
- 1 bit for carry-out

Operations

Name	Operation Code S_0S_1	Result
Add	00	$A_i + B_i = C_i$
XOR	01	A_i ? $B_i = C_i$
Right Rotate†	10	A_i ✍ C_{i-1} ; A_0 ✍ C_n
Left Rotate†	11	A_i ✍ C_{i+1} ; A_n ✍ C_0

†Left (Right) Rotate is an operation in which you will take all bits of the input and shift them to the left (Right) (✍) by one bit position, the leftmost (rightmost) bit will be placed on the rightmost (leftmost) location.

Ex.

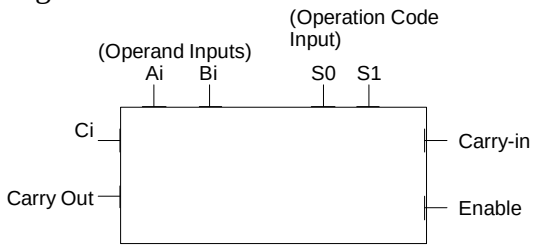
A= | 1 | 0 | 1 | 1 | 0 | 0 | 1 |

C= | 0 | 1 | 1 | 0 | 0 | 1 | 1 |

In this case, since it is a one-bit ALU-unit, just put whatever is in the carry signal (which should be 0).

Part 1.b:

Make a *package* of the circuit designed for part 1.a and name the device "1bitALU", save in a new library named "1bit-lib". A proper layout for the packaged device should be something like this:



Part 2:

Using the *device* created from part 1, create a 4 bit ALU, which receives two 4-bit numbers and returns one 4-bit number and a carry signal. In case of a unary operation (only one operand) inputs to the B operand will be ignored (don't cares).

After you finish the design instead of placing binary switches/probes for each input/output, place a Hex-Keyboard for the entry of the operands (one for A, another for B), 1 binary switch for enable, and two for selecting the operation code. Place a Hex display at the output and a binary probe at each carry-out. Indicate on the design which is the most significant bit and which is the least significant bit (of the 4 1-bit ALU's which is C_3 and which is C_0).

Implementation Details:

The work will be done using LogicWorks 4, which is available for use at CRAI Lab (S-105D).

Truth tables and K-maps must be handed in using the provided template paper. All spaces must be filled-in with their corresponding symbol 1, 0, X (don't cares), no empty spaces.

Logic Work Files must have the following filenames:

Part1.a:	1bit
Part1.b:	1bit-lib
Part2:	4ALU

On the top left corner of every design, add a text box with your name, student #, and account # for the class.

Correction criteria:

Criteria	Weight(%)
Correctness	60%
Design	20%
Efficiency	10%
Style	10%

Name: _____ Student #: _____
Assigned account #: _____

On Enable = 0, all outputs should be 0, thus it is not considered for the Truth Table nor the K-maps.

Input					Ouput	
S_0	S_1	A_i	B_i	Carry-In	C_i	Carry-Out
0	0	0	0	0		
0	0	0	0	1		
0	0	0	1	0		
0	0	0	1	1		
0	0	1	0	0		
0	0	1	0	1		
0	0	1	1	0		
0	0	1	1	1		
0	1	0	0	0		
0	1	0	0	1		
0	1	0	1	0		
0	1	0	1	1		
0	1	1	0	0		

1	0	0	0	0		
1	0	0	0	1		
1	0	0	1	0		
1	0	0	1	1		
1	0	1	0	0		
1	0	1	0	1		
1	0	1	1	0		
1	0	1	1	1		
1	1	0	0	0		
1	1	0	0	1		
1	1	0	1	0		
1	1	0	1	1		
1	1	1	0	0		
1	1	1	0	1		
1	1	1	1	0		
1	1	1	1	1		

K-Map for Ci
Carry-in =0

A_iB_i \ S_0S_1	00	01	11	10
00				
01				
11				
10				

K-Map for Ci
Carry-in =1

A_iB_i \ S_0S_1	00	01	11	10
00				
01				
11				
10				

K-Map for Carry-out
Carry-in =0

A_iB_i \ S_0S_1	00	01	11	10
00				
01				
11				
10				

K-Map for Carry-out
Carry-in =1

A_iB_i \ S_0S_1	00	01	11	10
00				
01				
11				
10				