EECE.2650 Logic Design Assignment 5 Due Friday 12/18/20

Objective:

The objective of this assignment is to design an ALU (Arithmetic Logic Unit).

PART I. Design

Design Specifications:

Design an Arithmetic Logic Unit that performs a set of operations on up to two 4-bit binary numbers based on a 3-bit operation code (OpCode).

Inputs:

clk: clock input, 10ns period 50% duty-cycle

DataA[3..0]: 4-bits of Data

DataB[3..0]: 4-bits of Data

RESET: Active low reset that sets the ALU to an initial state, with all data set to zero.

OpCode[2..0]: 3-bit Control input that represents a code for each operation.

START: 1-bit Control input that starts the operation after the OpCode has been set.

Outputs:

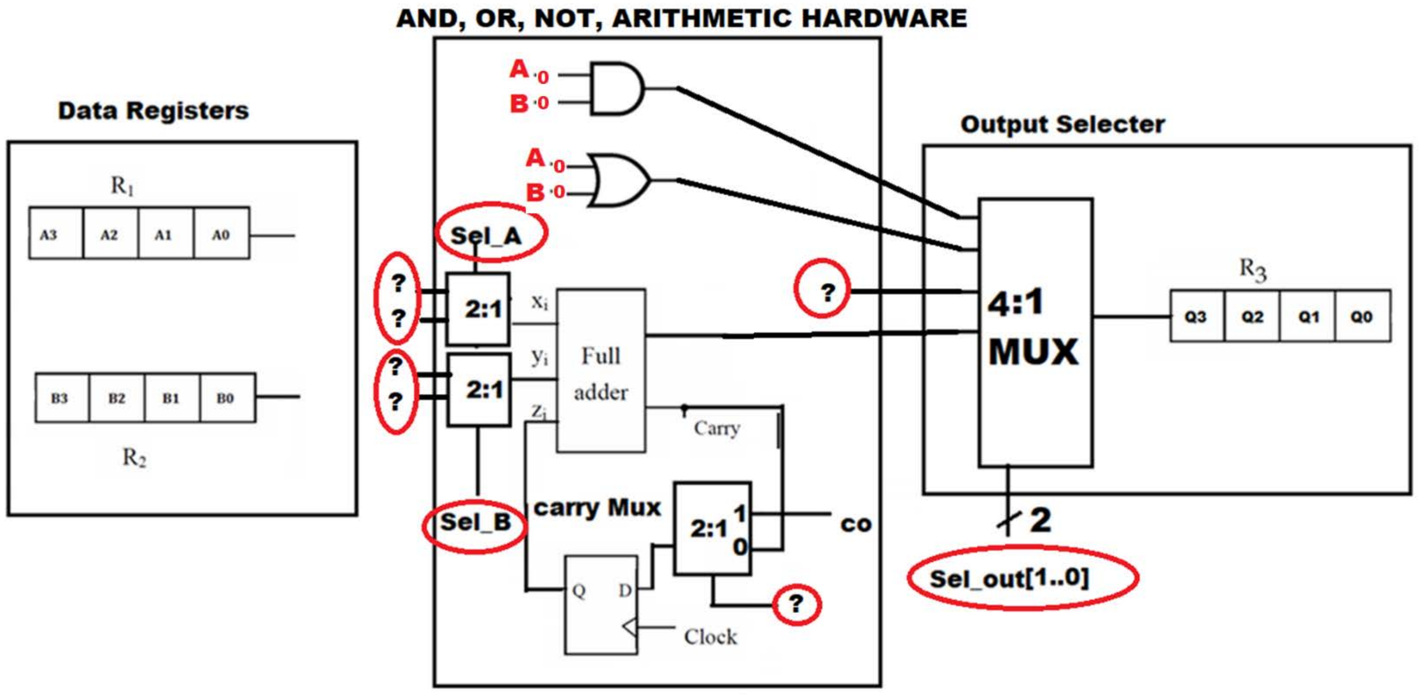
Q[3..0]: 4-bit result (Q3 is the MSB)

ALU Function Set

OpCode	**Function**
000	A + B (addition)
001	A + 1 (increment by 1)
010	~A (Compliment A)
011	B – A (subtraction)
100	A - 1 (decrement by 1)
101	A | B (bitwise OR)
110	A & B (bitwise AND)
111	-A (Mathematical Negative A)

Design: The design will consist of 3 modules: A Data Path, a State Generator, and a Control Circuit. The final circuit will connect these three modules to the inputs and outputs listed above. Each of the following sections should be its own bdf file.

Data Path:



Data Path Hardware:

THREE universal shift/parallel load registers (74194) : R1,R2,R3

ONE 1-bit Full adder (from Assignment 2)

ONE D-Type FF (for the carry)

Unlimited AND, OR, NOT, XOR, XNOR Gates

ONE 4:1 Multiplexer (74153M in Quartus – see details in the *Specialized Chips* section)

THREE 2:1 Multiplexers (21mux in Quartus)

Data Path Inputs:

S1, S0, RESET, clk,

DataA[3..0], DataB[3..0]

Sel_A, Sel_B, Sel_out[1..0]   (select bits for each MUX)

c0 (Initial carry for the serial adder)

const – a variable that can be set to a 1 or 0

Data Path Outputs:

Q[3..0]

State Generator: The State Generator should be designed to cycle through each state of the operation. Since these are 4-bit numbers, the operation will need at least 5 clock cycles, Load + 4 shift operations.

State Generator Behavior:

Hold in the initial state (T0) until START is set to 1

Perform the operation by shifting the data through the data path.

Wait until START is set back to 0 before returning to the initial state.

Asynchronously return to the initial state (T0) when the system RESET is pushed

State Generator Hardware:

For the state generator you should use 6 States with 1-hot encoding.

You can utilize the following chips:

- Unlimited AND, OR, NOT, XOR, XNOR Gates
- 6 DFF

State Gen Inputs:	**State Gen Outputs:**
RESET	T[**0**..5] – present state
START	
clk	

Control Circuit: The Control Circuit is a purely combinational circuit that will take the user inputs and translate them into signals for the Data Path and the State Generator.

Control Circuit Inputs:

T[0..5]

START

OpCode[2..0]

Control Circuit Outputs:

S1, S0

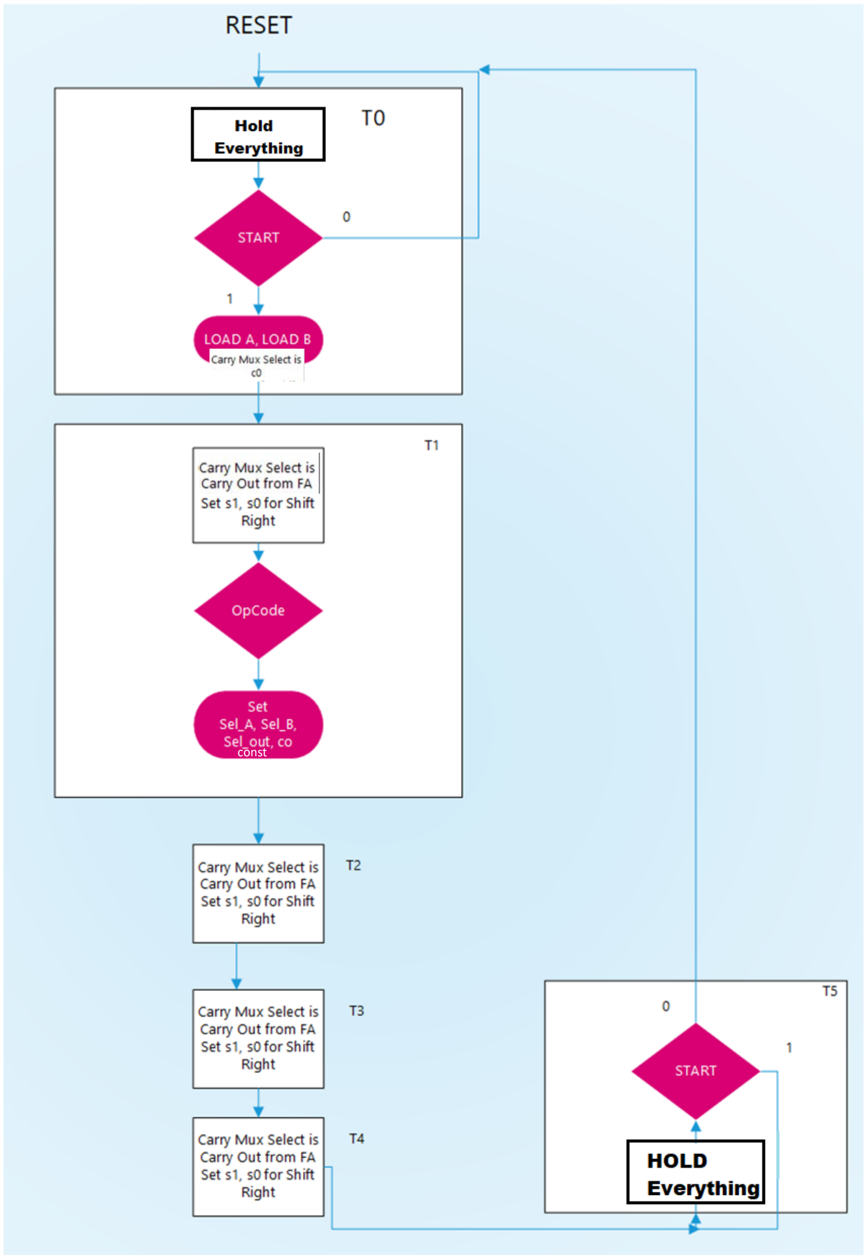
Sel_A, Sel_B, Sel_out[1..0]

c0, const

Control Circuit Hardware:

Unlimited AND, OR, NOT, XOR, XNOR Gates

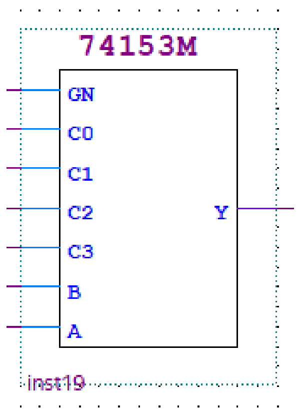
ASM CHART:


RESET
Hold Everything
T0
START
0
1
LOAD A, LOAD B
Carry Mux Select is c0
T1
Carry Mux Select is Carry Out from FA Set s1, s0 for Shift Right
OpCode
Set Sel_A, Sel_B, Sel_out, co const
T2
Carry Mux Select is Carry Out from FA Set s1, s0 for Shift Right
T3
Carry Mux Select is Carry Out from FA Set s1, s0 for Shift Right
T4
Carry Mux Select is Carry Out from FA Set s1, s0 for Shift Right
T5
0
START
1
HOLD Everything

Specialized Chips used in this design:

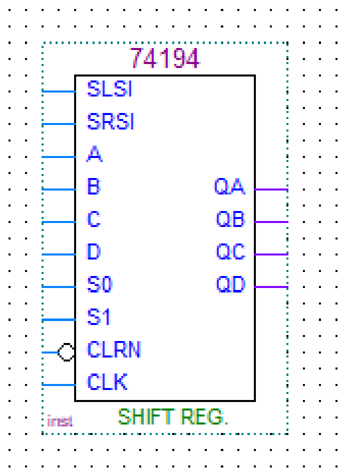
74153M 4:1 Mux with ACTIVE_LOW ENABLE (GN)

B A	Y
0 0	C0
0 1	C1
1 0	C2
1 1	C3



74194 Shift Register:

MODE	CLRN	S1	S0	QA	QB	QC	QD
RESET	0	X	X	0	0	0	0
HOLD	1	0	0	QA	QB	QC	QD
SHIFT LEFT	1	1	0	QB	QC	QD	SLSI
SHIFT RIGHT	1	0	1	SRSI	QA	QB	QC
PARALLEL LOAD	1	1	1	A	B	C	D



X = Don't care

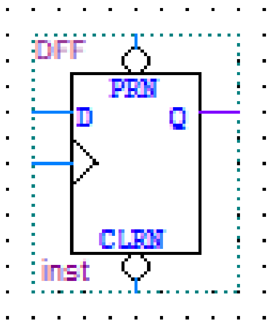
All transistions except for the asynchronous reset occur on RISING CLOCK EDGE

DFF is a D-type Flip flop that updates on the Rising Clock Edge:

PRN = asynchronous ACTIVE-LOW PRESET

CLRN = asynchronous ACTIVE-LOW CLEAR

D	Q
0	0
1	1

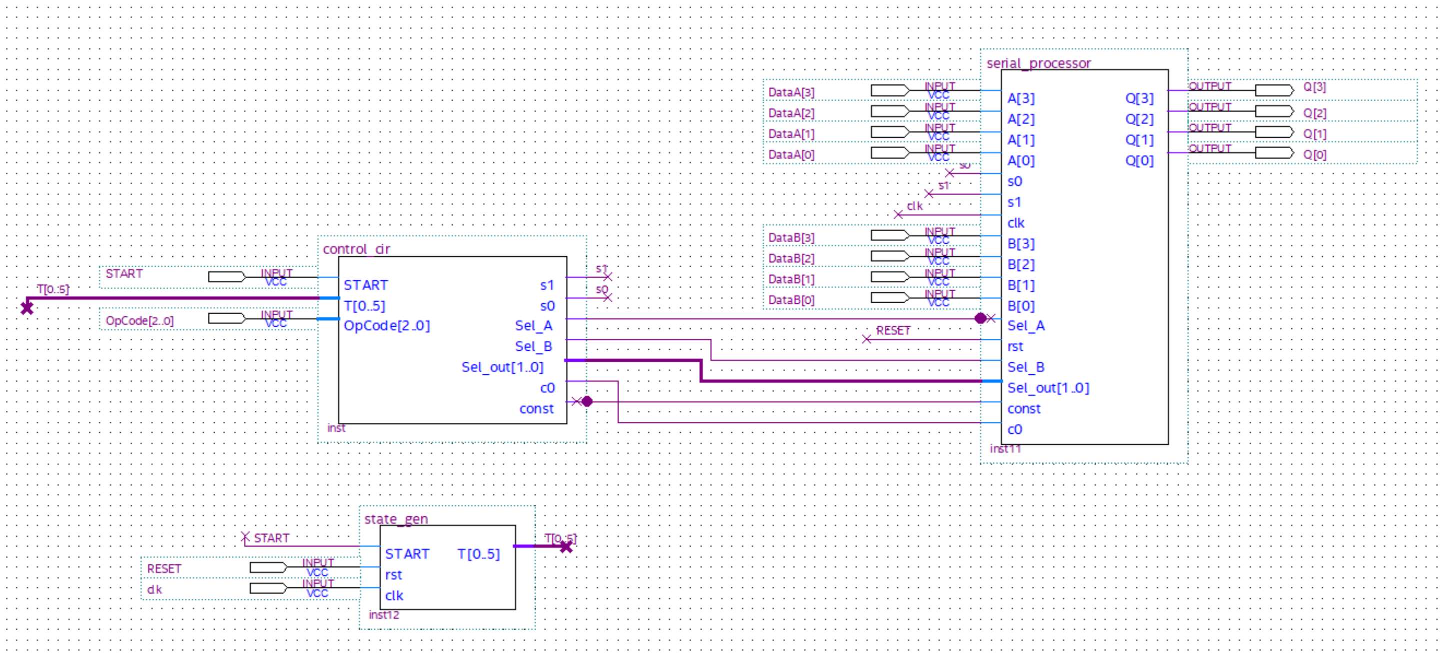


Final Design:

Take each Block Diagram File for the three circuit modules above and turn them into individual .bsf files

Connect the bsf files in a single bdf as below:

There should be no additional gates in the block diagram.



NOTE: The INPUT AND OUTPUT NAMES on this bdf MUST MATCH EXACTLY:

Inputs:

clk

DataA[3..0]

DataB[3..0]

RESET

OpCode[2..0]

START

Outputs:

Q[3..0]

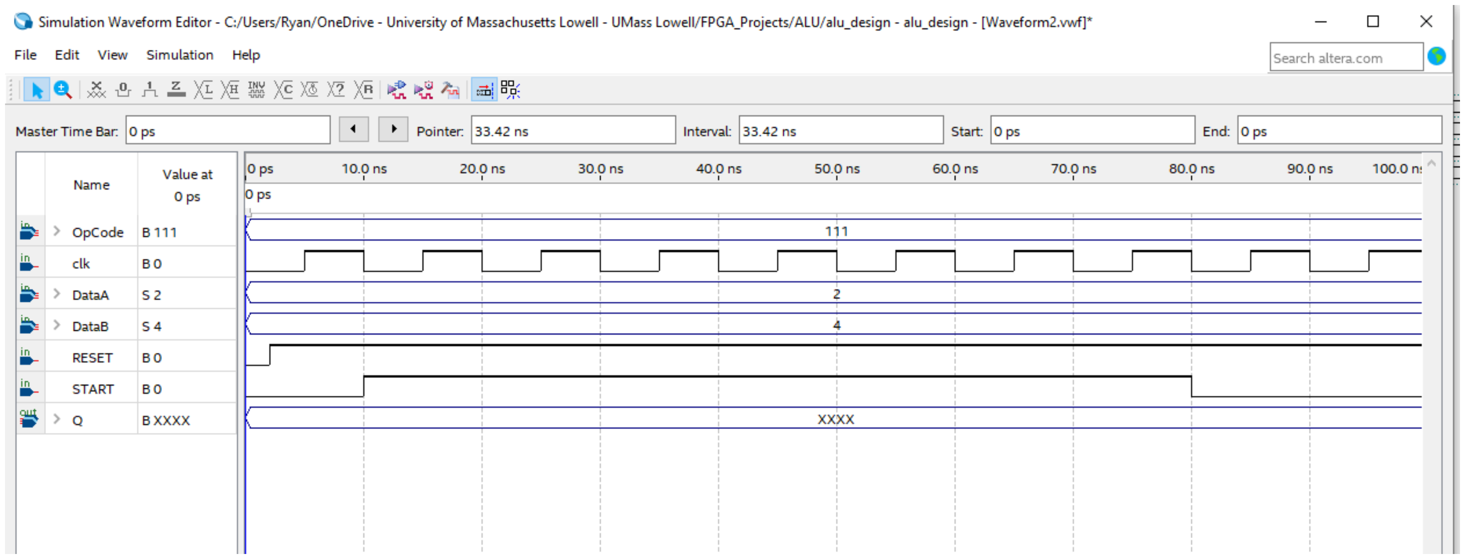
Other internal signals, such as Sel_A, do not need to match exactly.

Part II: Test

Test each OpCode by doing 8 simulations. Each simulation should last for 100ns.

Set START = 1 from 10ns to 80ns. This way you can make sure that the system is holding in T5 as specified.

Use the template below for your simulations:



You can choose to use whatever numbers you want for A, B. Note that we did not add overflow detection to this circuit.

Part III: Write-up

Use the separate document, *Assignment 5 Template* for your submission.

Part IV: What to Submit

You will submit 2 files:

1. The write up (Part III)
2. Your entire project in archived form (.qar file).

To archive your project, simply go to Project -> Archive Project
The .qar file will be crated in your working directory.