Your Name: ____________________________________	netid:_________________	Group #:
Name: ____	netid:_________________	
Name: ____	netid:_________________	
Name: ____	netid:_________________	

ECE 120 Worksheet 15: LC-3 Control Unit

In this discussion, you will work with the LC-3 control unit defined in the appendix of Patt & Patel. In particular, you will extend the LC-3 ISA by introducing a new instruction, and will modify the microsequencer in order to extend the ISA.

The problems presented in this discussion are taken from final exams from prior semesters.

You may detach and keep the last two pages of this discussion booklet.

1. Extending the LC-3 ISA

You are charged with adding a new instruction, called **ADS** (add and store), to the LC-3 instruction set. This instruction adds two values, just like the ADD instruction, and stores the result to **memory** instead of the register file. The destination memory address is provided in the BaseR register specified by IR[11:9] bits (so M[BaseR] ← sum). The binary encoding of this instruction is:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADS register mode	1	1	0	1	BaseR			SR 1			0	0	0	SR 2		

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADS immediate mode	1	1	0	1	BaseR			SR 1			1	Imm5				

a) In RTL form, give a sequence of (at most four) *microinstructions* that implement the execute phase of the **ADS** instruction. Make sure that your implementation **does not modify any values in the general-purpose register file** and **does not set condition codes**.

MDR ← SR1 + OP2 **(notation used in P&P for ADD / AND)**
MAR ← BaseR **(first two ops can be swapped)**
M[MAR] ← MDR

Note that the first state—regardless of the order chosen—has to be #13 (matches the opcode—the first execute state for any instruction). The last state should be #16. The middle state can be any of those given as available in part (b).

b) Determine the control ROM microinstructions that implement the RTL statements from part (a). Complete the table below by filling in 0, 1, or X as appropriate. Use don't cares wherever possible. Specify ROM addresses in **decimal**. Note that your first state number is implied by the decode strategy used in the P&P microarchitecture. Be sure to reuse the memory write state used by all other store instructions in the design (see the state diagram attached to the back of this document). If you need additional states, state numbers **55**, **56**, **57**, and **58** are available for your use.

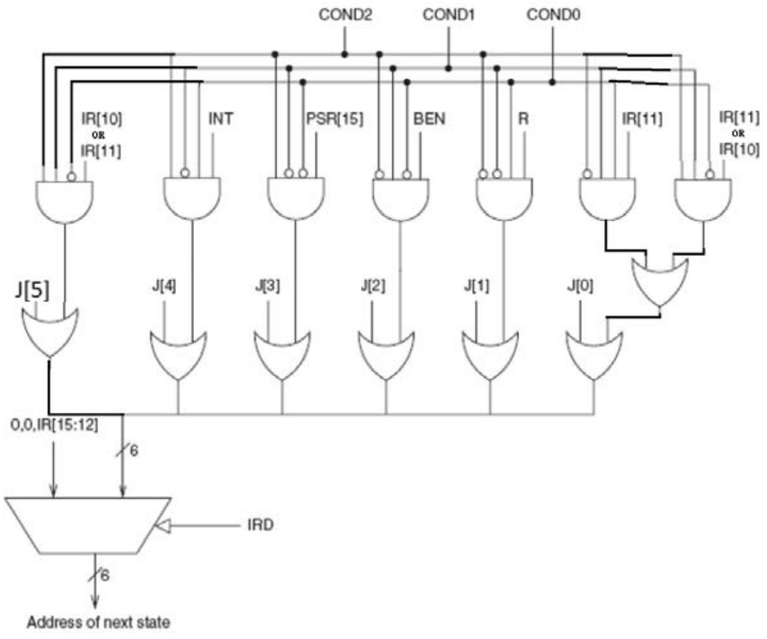
ROM address	IRD	COND(3)	J(6)	LD.BEN	LD.MAR	LD.MDR	LD.IR	LD.PC	LD.REG	LD.CC	GateMARMUX	GateMDR	GateALU	GatePC	MARMUX	PCMUX(2)	ADDR1MUX	ADDR2MUX(2)	DRMUX(2)	SR1MUX(2)	ALUK(2)	MIO.EN	R.W
13	0	000	***	0	0	1	0	0	0	0	0	0	1	0	x	xx	x	xx	xx	01	00	0	x
***	0	000	16	0	1	0	0	0	0	0	1	0	0	0	1	xx	1	00	xx	00	xx	0	x
16	0	001	16	0	0	0	0	0	0	0	0	0	0	0	x	xx	x	xx	xx	xx	xx	1	1
***	0	000	16	0	1	0	0	0	0	0	0	0	1	0	x	xx	x	xx	xx	00	11	0	x

The last line is an alternate implementation for the second state.

2. Modifying the LC-3 Microsequencer

Add a new instruction, **LOGIC**, with opcode 1101, to the LC-3. The **LOGIC** instruction performs one of four logic operations (NAND, NOR, XOR, and OR) based on the IR[11:10] bits. To implement this new instruction, we need five new states in the LC-3 FSM and need to modify the microsequencer circuit. The first state performs a secondary decode phase before executing the four logic operations.

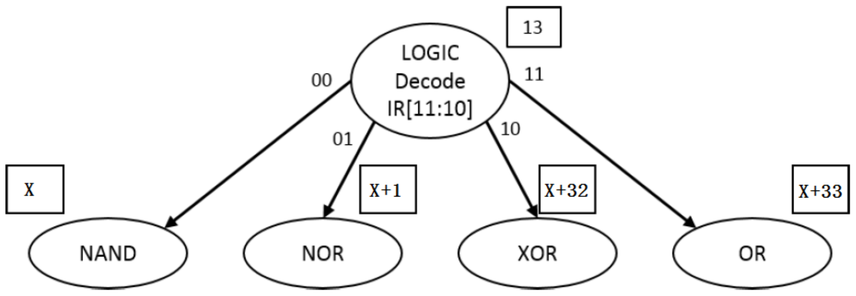
a) Use COND = 110 to determine the next state during the secondary decode phase of the logic operation. Adding at most four gates, modify the microsequencer circuit so that it can correctly decode the **LOGIC** instruction. A few modifications have already been made to give you a hint. Note that the INT and PSR[15] signals in the diagram are beyond the scope of our course.



Any bits can be modified, simplest might be just bits 5 and 0. Need to capture COND and IR[11] and IR[10], then modify one bit each for those two. The numbering below must then be based on which two bits of J were modified with IR[11:10] when COND = 110.

b) Based on your microsequencer circuit above, choose a set of *viable* next states for the execute phase states. Don't worry if the states are already in use by the LC-3 FSM.

Below is the solution when IR[11] is used in the far left NAND gate and IR[10] in the far right.


13
LOGIC Decode IR[11:10]
00
01
10
11
X
X+1
X+32
X+33
NAND
NOR
XOR
OR