Department of Electrical and Computer Engineering
The University of Texas at Austin

EE 460N Fall 2014
Y. N. Patt, Instructor
Stephen Pruett, Emily Bragg, Siavash Zangeneh TAs
Exam 2
November 5, 2014

Name:_____

Problem 1 (20 points):_____

Problem 2 (10 points):_____

Problem 3 (20 points):_____

Problem 4 (25 points):_____

Problem 5 (25 points):_____

Total (100 points):_____

Note: Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space provided.

Note: Please be sure your name is recorded on each sheet of the exam.

Please sign the following. I have not given nor received any unauthorized help on this exam.

Signature:_____

GOOD LUCK!

Name:_____

Problem 1 (20 points)

Part a (5 points): A parity bit can be used to detect any single bit error. But if two bits are transmitted in error, the parity bit scheme does not work. What characteristic of the bit errors makes this a non-problem for those situations that use parity for detecting bit errors.

Part b (5 points): Vector instruction A requires the result of vector instruction B as a source. Vector chaining allows vector instruction A to start processing before vector instruction B finishes. When can vector instruction A start its execution phase?

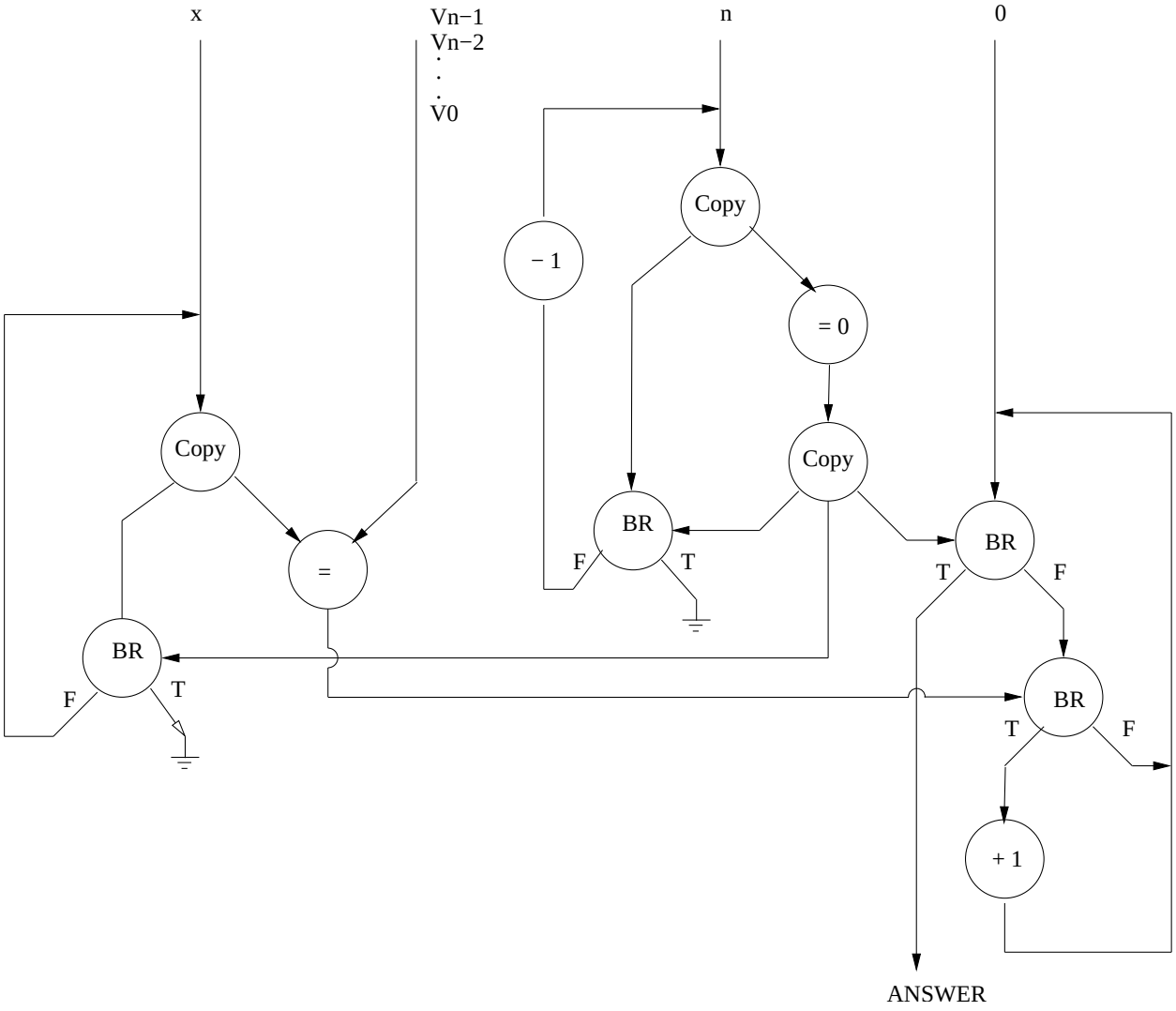
Part c (5 points): An SRAM cell consists of two cross-coupled inverters (therefore, at least 4 transistors usually, and often more). It stores a 1 or a 0 depending on which inverter is outputting a 1 and which is outputting a 0. On the other hand, a DRAM consists of only one transistor. How, then, does it store a 1 or 0?

Part d (5 points): On a page fault, the operating systems often loads a page from the disk into a frame that was previously occupied by a different page. How does the operating system know whether it is necessary to write the previously occupied page back to the disk? Please be brief and explicit. Answer in fifteen words or fewer.

Name:_____

Problem 2 (10 points)

The following data flow graph receives as inputs a value x, an n element vector $V_0, V_1, ..., V_{n-1}$, the value n, and a value 0 on its four input ports.

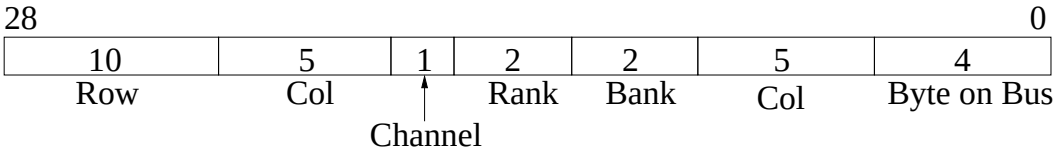


What "answer" is produced by the execution of this data flow graph?

Name:_____

Problem 3 (20 points)

Part a (10 points) Assume we have a byte addressable memory that has the following address format:



i (2 points): What is the maximum size of physical memory?

ii (2 points): Circle the correct answer: Multiple (Channels / Banks / Ranks) enable simultaneous transfer of data from different parts of memory to the processor in the same cycle.

iii (3 points): How many bits of data can be transferred between the processor and memory simultaneously?

iv (3 points): How many bytes of storage are there on a single DRAM chip?

Part b (10 points) Suppose we have a byte addressable memory system made up of 1 rank and some number of banks.

For the following program fragment, assume the elements of arrays a, b, and c occupy a single byte each. Assume that the variable "i" is kept in a register for the duration of the fragment and does not cause any memory accesses. Initially none of the elements of a, b, and c have been loaded into a row buffer. What is the minimum number of banks and the minimum size of the row buffer such that exactly 4 row buffer misses occur?

```
for ( i = 0; i < 63; i++ )
     c[i] = a[i] + b[i];
```

Banks

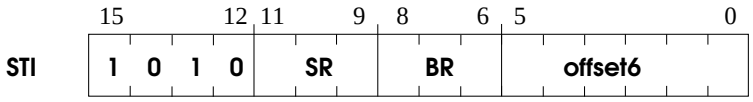
Bytes in row buffer

Name:_____

Problem 4 (25 points)

We wish to enhance the LC-3b by adding Virtual memory and a new instruction. Virtual memory will be implemented by a VAX-like scheme as we studied in class.

The new instruction will be STI SR,BR,offset and will use opcode 1010. The format is:



STI stands for Store Indirect.

STI operates as follows: We compute a virtual address (call it A) by adding the sign-extended offset to the contents of BR. The memory location specified by A contains the virtual address B. We wish to store the contents of SR into the address specified by B.

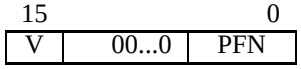
(Note: For those of you who studied the STI instruction in EE306, note that the address A is calculated differently from the way it was in the LC-3.)

Part a (5 points): To process the STI instruction, one must go through the Fetch, Decode, etc. instruction cycle. What is the maximum number of physical addresses that can be accessed in processing an STI instruction.

Part b (20 points): You are given the following information:

Virtual Address Space:	64 KB	Physical Memory Size:	4 KB
User Space Range:	x0000 to x7FFF	PTE Size:	2 Bytes
System Space Range:	x8000 to xFFFF		

PTE Format:



R0: x8000
R1: x401E
PC: x3048

A 4 entry TLB.

V	Page #	PTE	
		V	PFN
1	x0C1	1	x1A
1	x182	1	x24
0	-	-	-
0	-	-	-

Name:_____

Problem 4 continued

To process STI R0,R1,#0, seven physcial memory accesses are needed. The table below shows the VA, PA, Data, and whether or not there was a TLB hit for each of these seven physical memory accesses in the order they occurred.

Your job: Complete the table and fill in the four boxes. You can assume no page faults occur.

Virtual Address	Physical Address	Data	TLB Hit
			Yes
	x360		No
		x8040	No
		x40FE	No
		x8004	No
xB00E			No
	x1DE		No

Frame size: ☐

Number of frames: ☐

UBR: ☐

SBR: ☐

Name:_____

Problem 5 (25 points):
In this problem, we wish to add the capability to handle two new exceptions to the LC-3b ISA: (1) a **write-only** exception if the program tries to read the Display Data Register (address xFE06) and (2) a **read-only** exception if the program tries to write to the Keyboard Data Register (address xFE02).

Assume the same baseline machine you started with in Lab 4. i.e., none of the exception handling you added in lab 4 is present.

If one of these two exceptions is detected, a flag (E) will be set and a corresponding exception vector EXCV will be loaded. The exception vector for the write-only exception is x05. The exception vector for read-only is x06.

You can assume that the states (starting at state X) needed for pushing the PSR and PC on the stack, setting the privilege mode, and loading the PC with the address corresponding to the correct exception vector are done for you.

Your job: Modify the state machine, design the logic to generate E and EXCV, and modify the microsequencer.

PROBLEM IS CONTINUED ON THE NEXT PAGE!

Name:_____

Problem 5 continued:

Part a : Modify the state machine.

There are several places in the state machine where one of these exceptions could occur. We have provided enough two-state sequences for you to check and initate each exception. Complete each of the two-state sequences you need for each of the places where they are needed. Note the ”from” and ”to” boxes associated with each two-state sequence. These are used to identify where the two-state sequence is to be inserted. You may not need all the two-state sequences that we have provided.

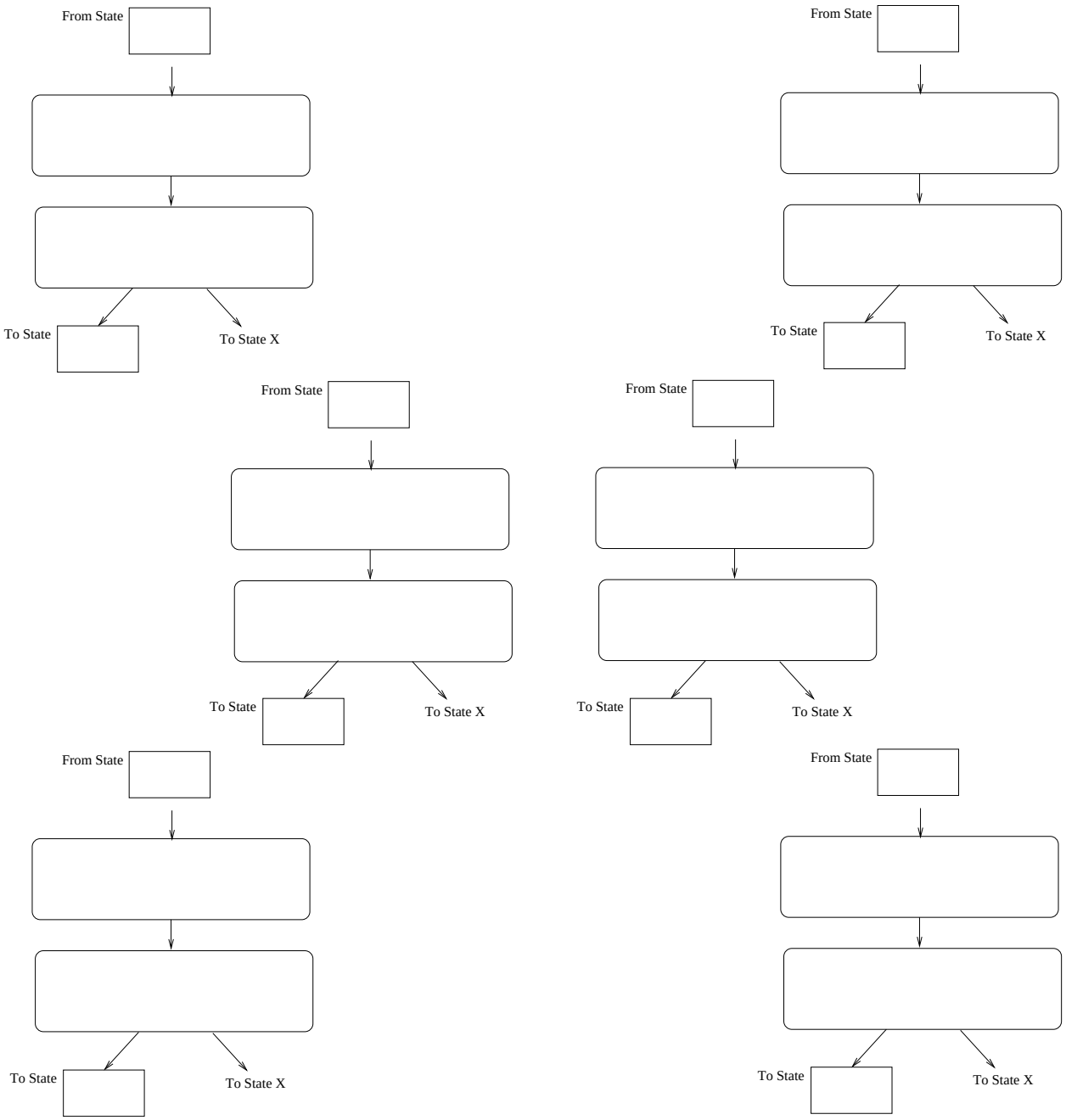


Figure 1: Two-state sequences

Name:_____

Problem 5 continued:
Part b : Design the logic to generate E and EXCV. You are not allowed to add any additional control signals to the control store.

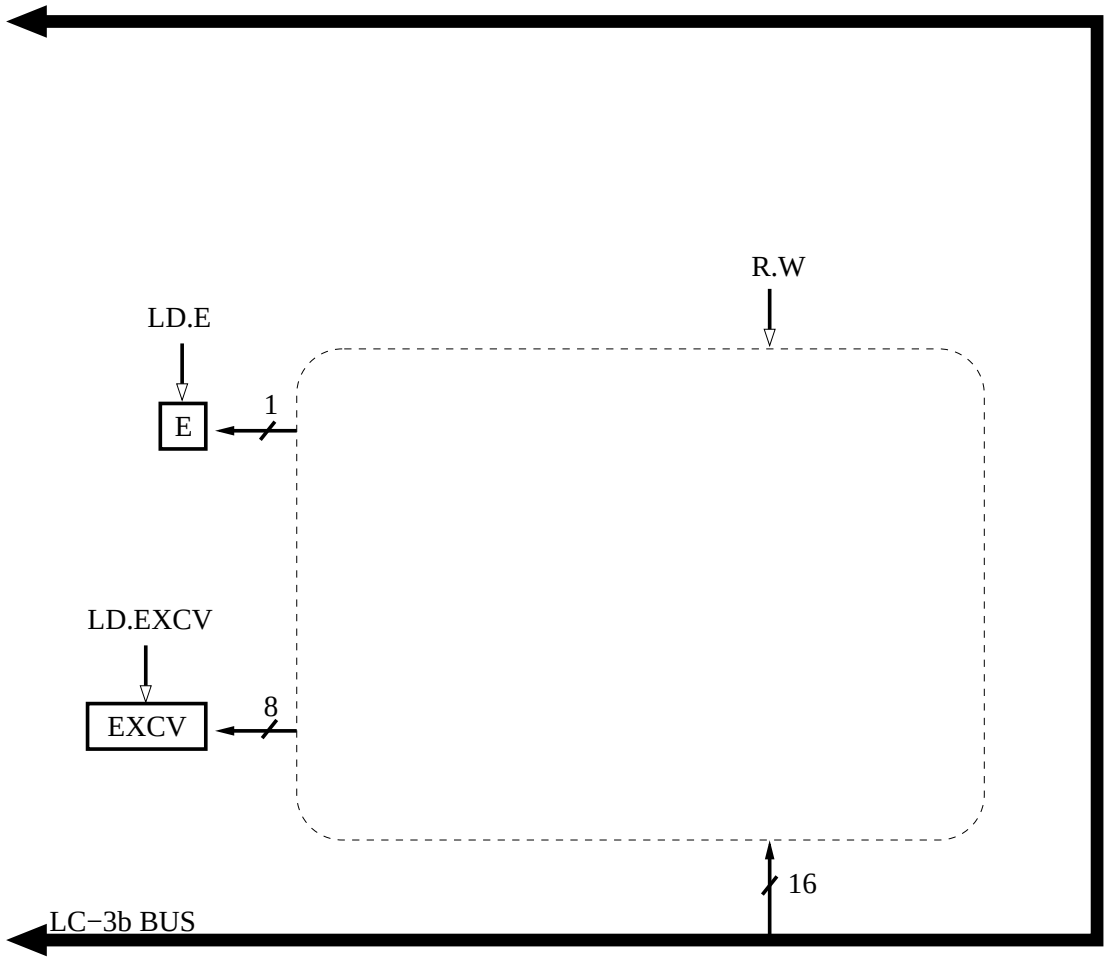


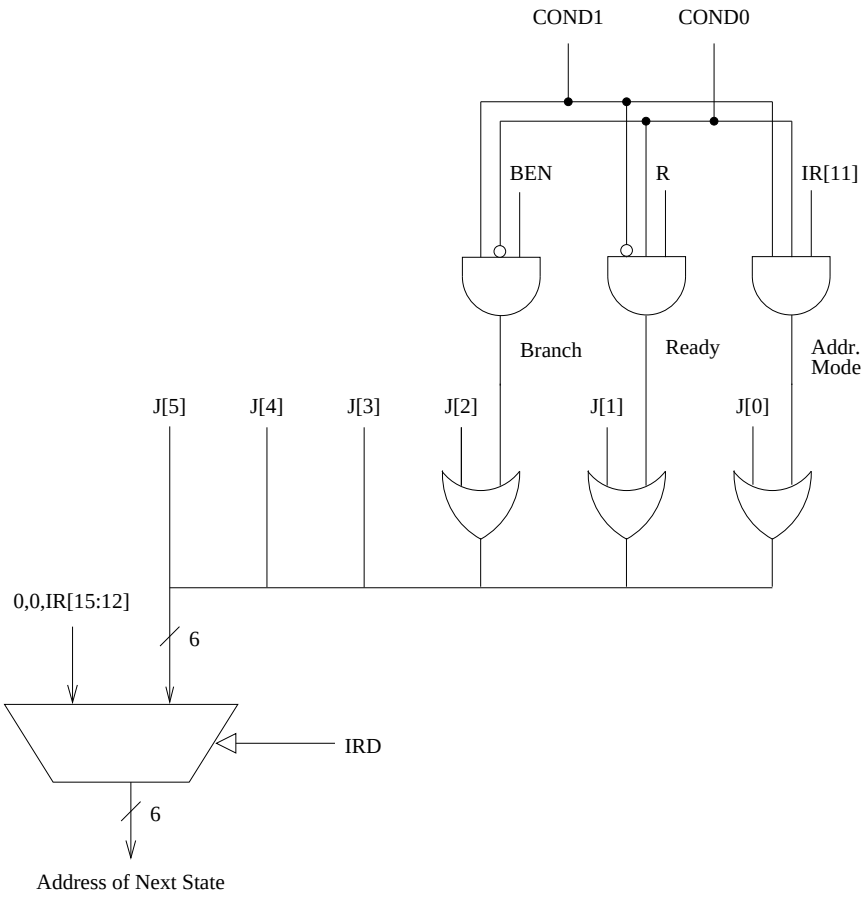
Figure 2: Modified Datapath for new exceptions

PROBLEM IS CONTINUED ON THE NEXT PAGE!

Name:_____

Problem 5 continued:

Part c : Modify the microsequencer. You are not allowed to add any additional control signals to the control store.



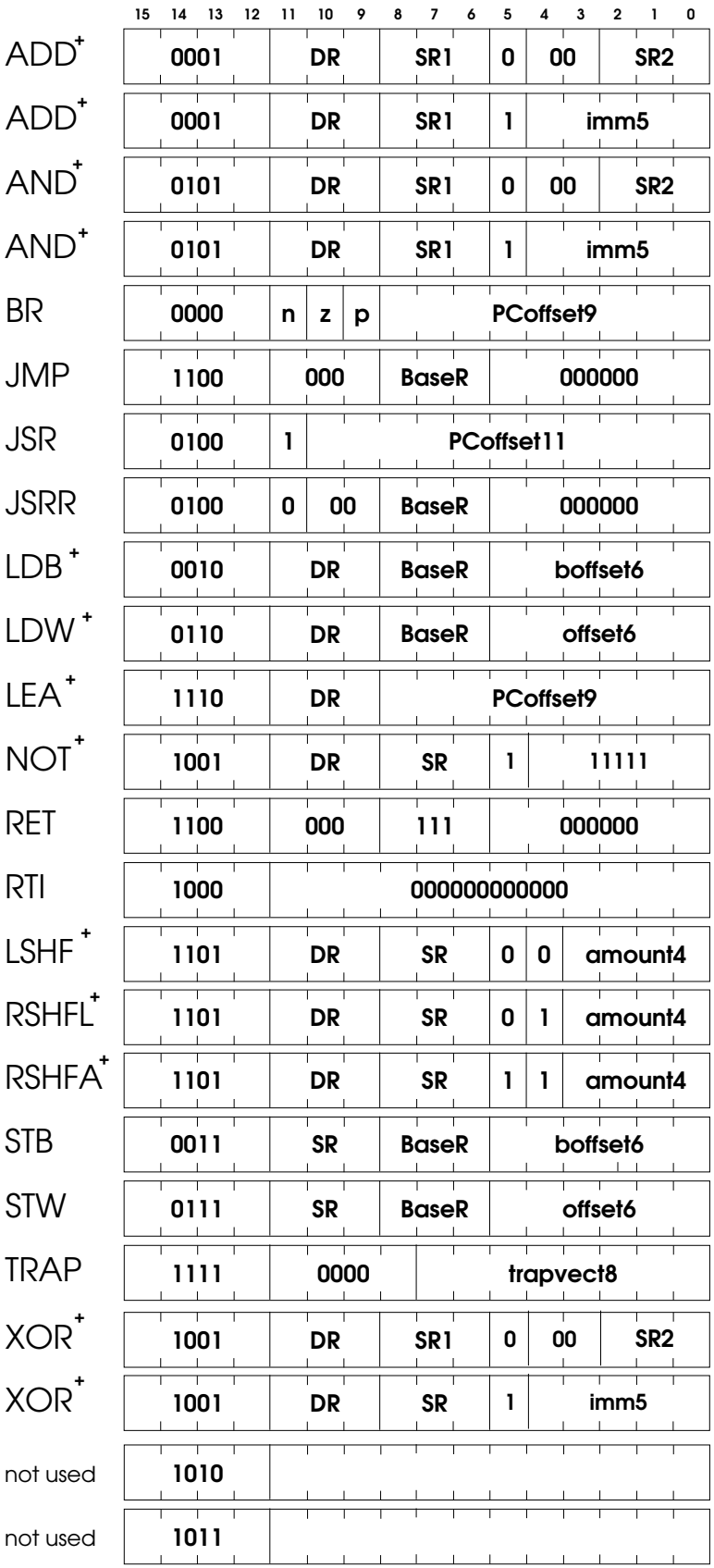


Figure 3: LC-3b Instruction Encodings

Table 1: Data path control signals

Signal Name	Signal Values	
LD.MAR/1:	NO(0), LOAD(1)	
LD.MDR/1:	NO(0), LOAD(1)	
LD.IR/1:	NO(0), LOAD(1)	
LD.BEN/1:	NO(0), LOAD(1)	
LD.REG/1:	NO(0), LOAD(1)	
LD.CC/1:	NO(0), LOAD(1)	
LD.PC/1:	NO(0), LOAD(1)	
GatePC/1:	NO(0), YES(1)	
GateMDR/1:	NO(0), YES(1)	
GateALU/1:	NO(0), YES(1)	
GateMARMUX/1:	NO(0), YES(1)	
GateSHF/1:	NO(0), YES(1)	
PCMUX/2:	PC+2(0)	;select pc+2
	BUS(1)	;select value from bus
	ADDER(2)	;select output of address adder
DRMUX/1:	11.9(0)	;destination IR[11:9]
	R7(1)	;destination R7
SR1MUX/1:	11.9(0)	;source IR[11:9]
	8.6(1)	;source IR[8:6]
ADDR1MUX/1:	PC(0), BaseR(1)	
ADDR2MUX/2:	ZERO(0)	;select the value zero
	offset6(1)	;select SEXT[IR[5:0]]
	PCoffset9(2)	;select SEXT[IR[8:0]]
	PCoffset11(3)	;select SEXT[IR[10:0]]
MARMUX/1:	7.0(0)	;select LSHF(ZEXT[IR[7:0]],1)
	ADDER(1)	;select output of address adder
ALUK/2:	ADD(0), AND(1), XOR(2), PASSA(3)	
MIO.EN/1:	NO(0), YES(1)	
R.W/1:	RD(0), WR(1)	
DATA.SIZE/1:	BYTE(0), WORD(1)	
LSHF1/1:	NO(0), YES(1)	

Table 2: Microsequencer control signals

Signal Name	Signal Values	
J/6:		
COND/2:	$COND_0$	;Unconditional
	$COND_1$	;Memory Ready
	$COND_2$	;Branch
	$COND_3$	;Addressing Mode
IRD/1:	NO, YES	

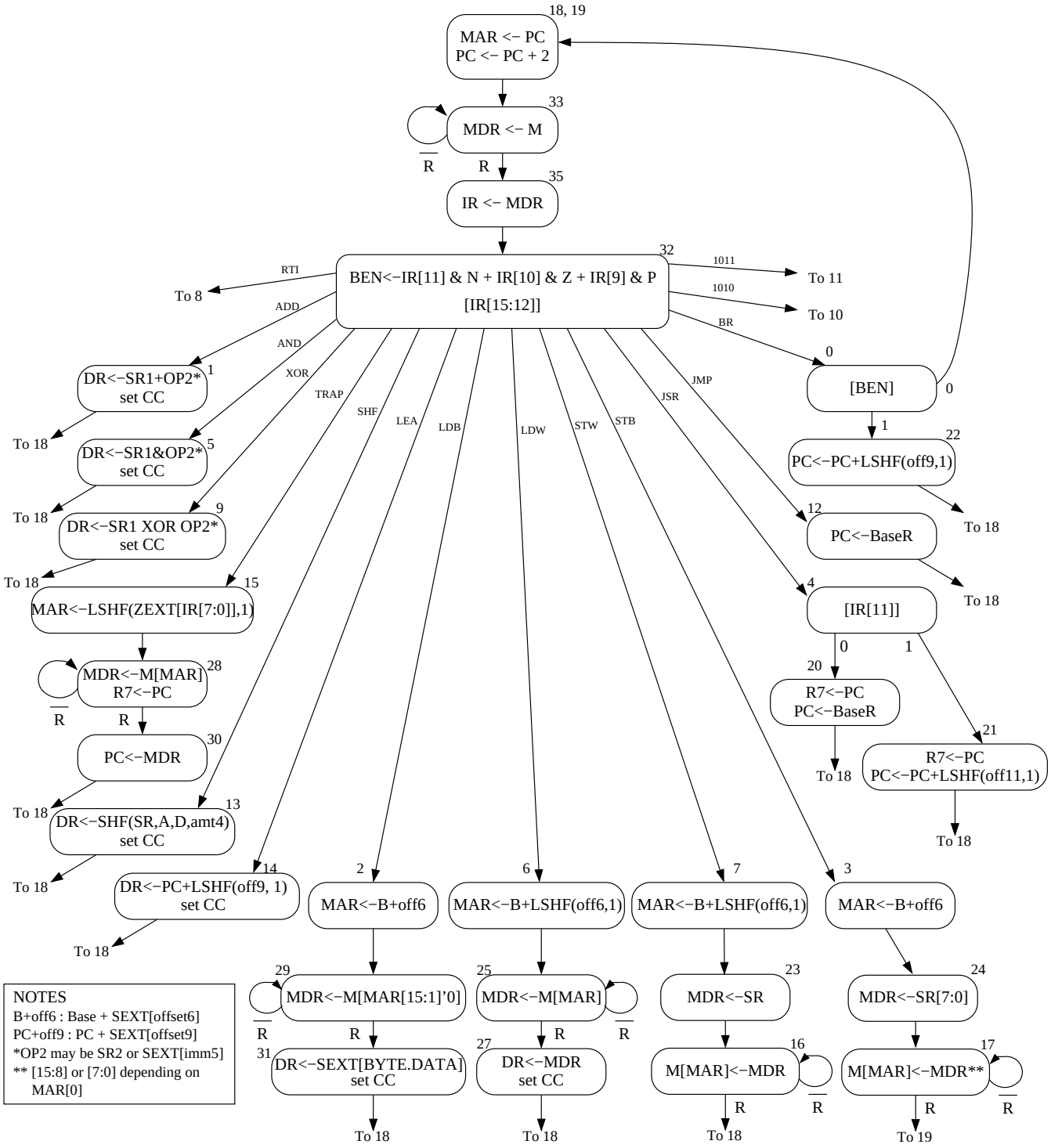


Figure 4: A state machine for the LC-3b

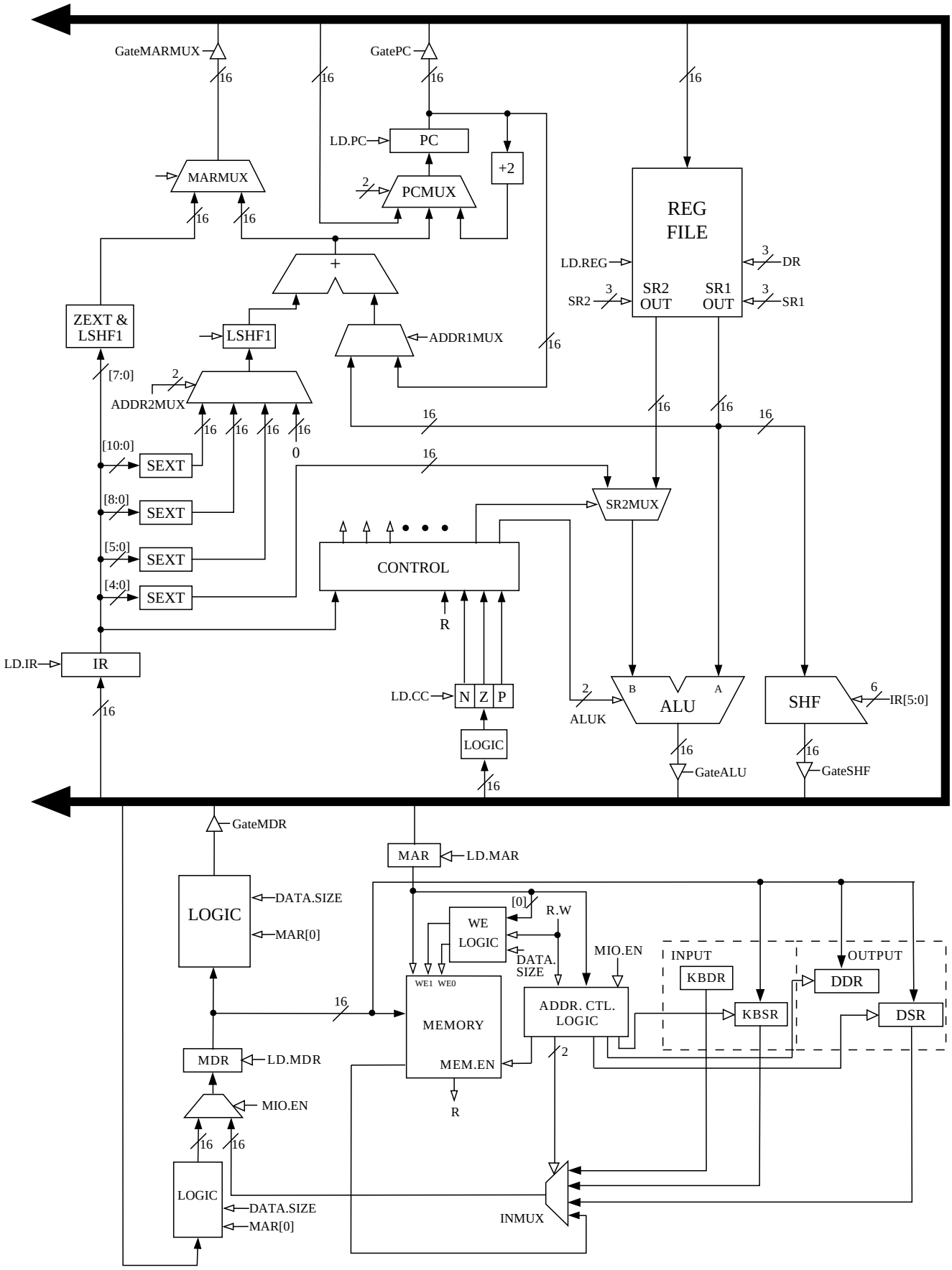


Figure 5: The LC-3b data path

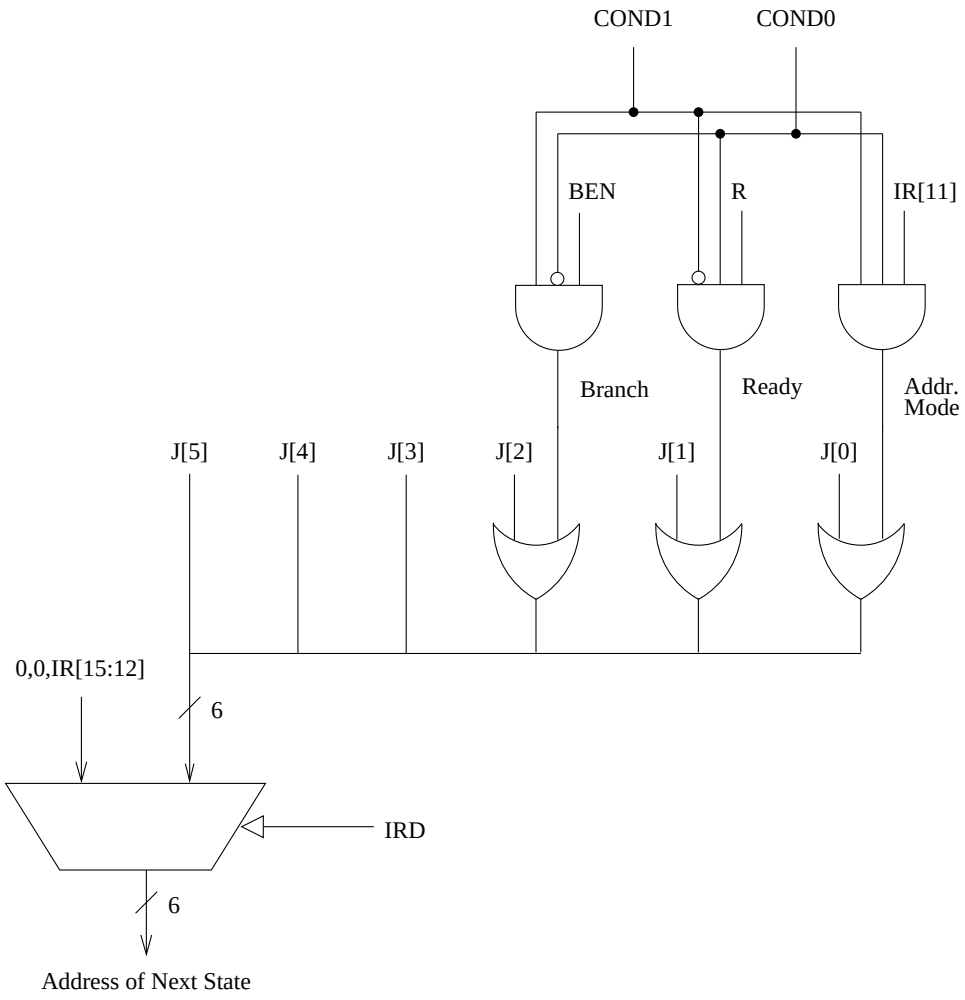


Figure 6: The microsequencer of the LC-3b base machine