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

EE 360N, Fall 2005
Yale Patt, Instructor
Aater Suleman, Linda Bigelow, Jose Joao, Veynu Narasiman, TAs
Exam 1, October 19, 2005

Name:_____

Problem 1 (20 points):_____

Problem 2 (20 points):_____

Problem 3 (20 points):_____

Problem 4 (20 points):_____

Problem 5 (20 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.

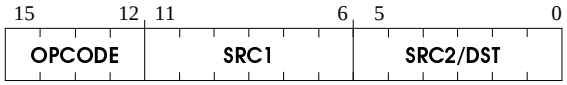
GOOD LUCK!

Name:_____

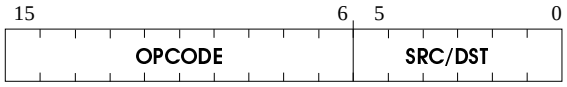
Problem 1 (20 points)

Part a (4 points): The VAX has an instruction INSQUE ptr1, ptr2 which causes the node pointed to by ptr1 to be inserted into a doubly linked list immediately after the node pointed to by ptr2. We say the doubly linked list is a [ ] supported in the VAX ISA.

Part b (5 points): The PDP-11 had several two-operand instructions with the format



one-operand instructions with the format



and zero-operand instructions with the format

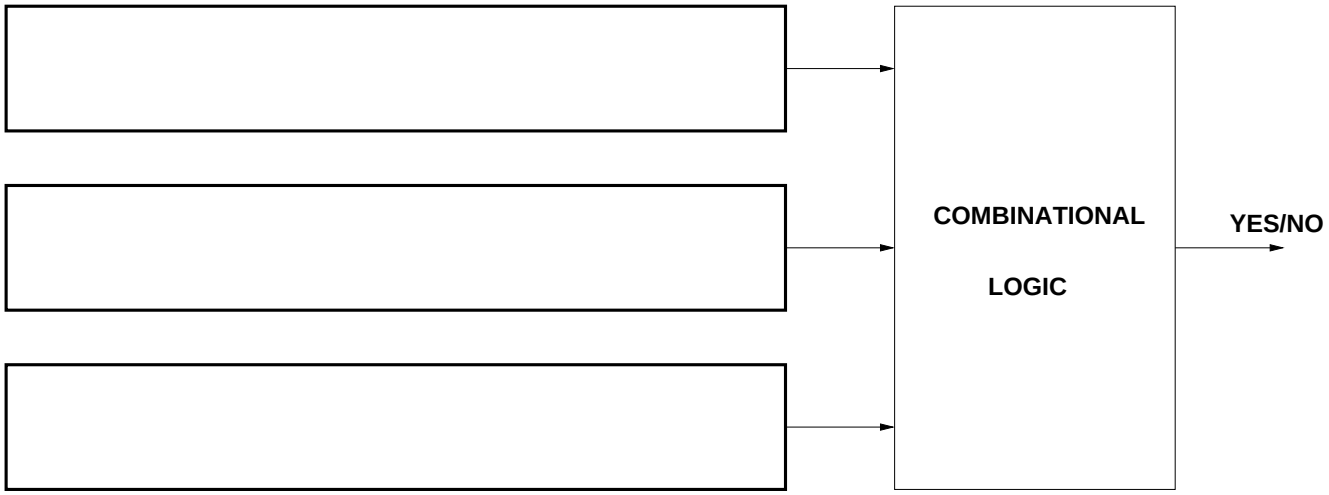
The ADD instruction had opcode 0110. The **prefix property** guaranteed

[ ].

Name:_____

Problem 1 continued

Part c (6 points): The virtual memory management system is called into play on every memory access instruction. In addition to translation, the right to access the information in the memory location is determined. The logic is as follows:



Identify the three elements that are input to the combinational logic required to make the yes/no access decision.

Part d (5 points): A page table has a base register and a length (aka limit) register. The purpose of the length register is

_____.

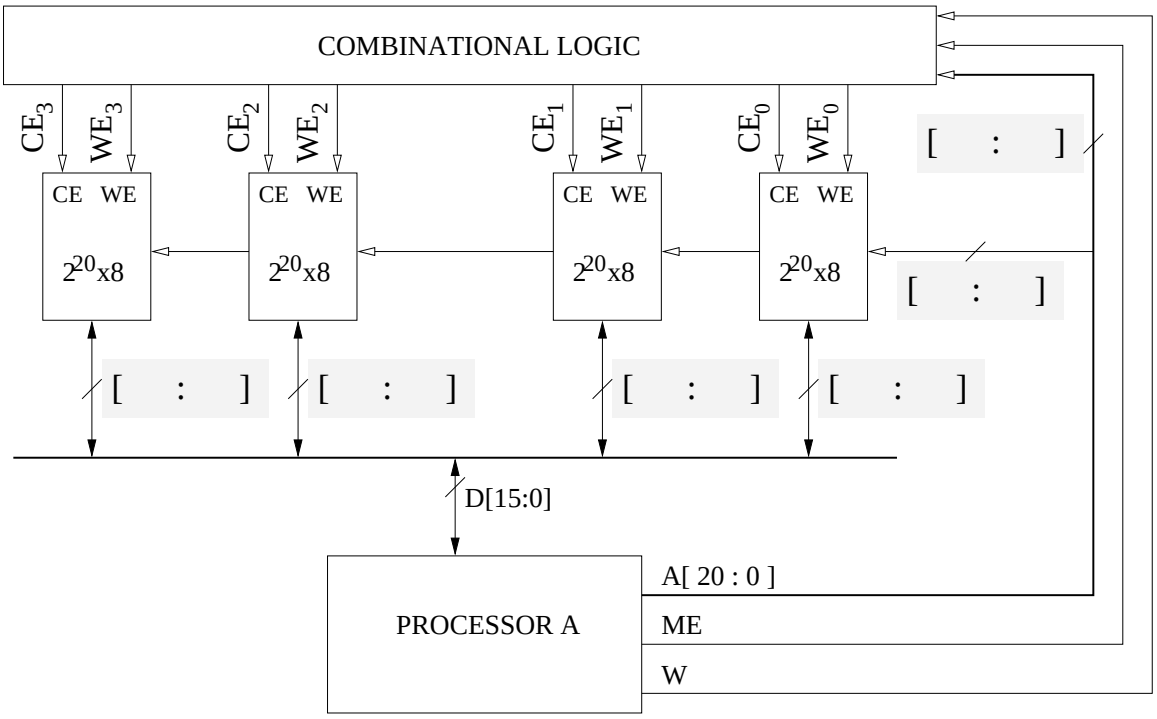
What would happen if we did not have one?

_____.

Name:_____

Problem 2 (20 points)

Part a (10 points): You are given four 1 MB, 8-bit wide, DRAM chips to build a 4 MB, 2-way interleaved, **word-addressable** main memory of a little-endian computer. Specify the bit fields indicated in the shaded regions for each unspecified signal.



Write the logic equations CE_0, WE_0, ..., CE_3, WE_3 in terms of the inputs shown. Do not use more inputs than required to specify each output.

CE_0 = _____

CE_1 = _____

CE_2 = _____

CE_3 = _____

WE_0 = _____

WE_1 = _____

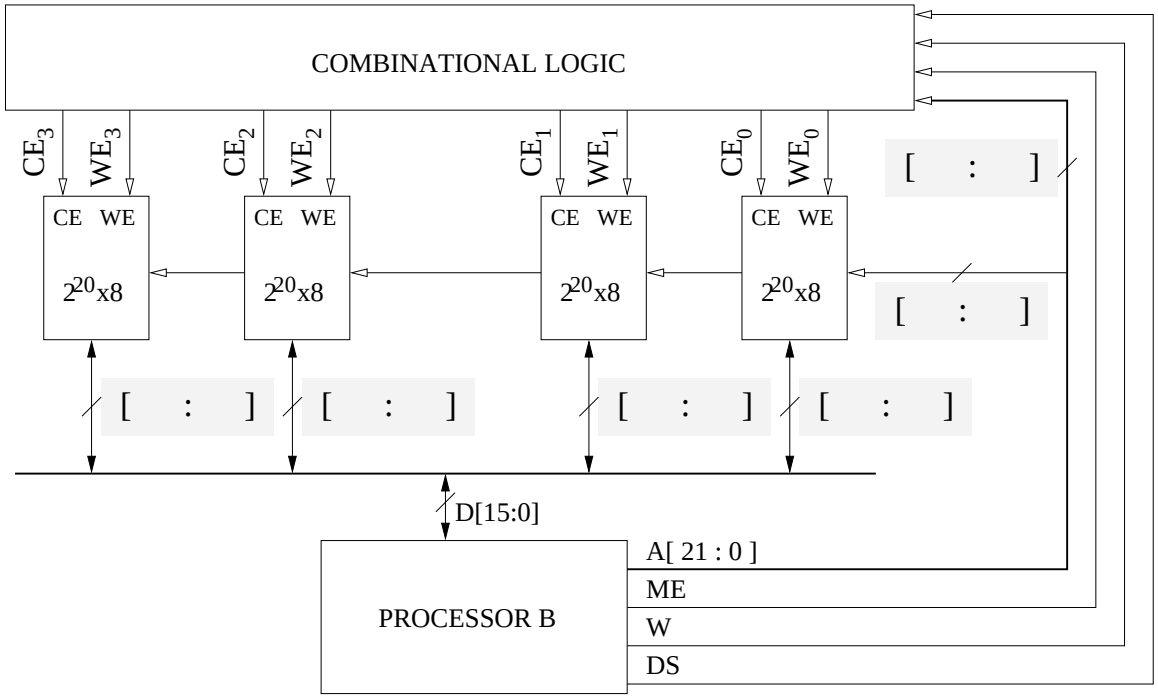
WE_2 = _____

WE_3 = _____

Name:_____

Problem 2 continued:

Part b (10 points): Suppose the memory system of Part a was changed to **byte-addressable**. That is, the four 1 MB, 8-bit wide DRAM chips form a 4 MB, 2-way interleaved, byte-addressable main memory. Specify the bit fields indicated in the shaded regions for each unspecified signal.



The ISA for Processor B supports LDB, LDW, STB, and STW instructions, but does not support unaligned accesses. Note the additional control signal DS (data size), where 1 designates word access, and 0 designates byte access. Write the logic equations for CE_0, WE_0, ..., CE_3, WE_3 in terms of the inputs shown. Do not use more inputs than required to specify each output.

CE_0 = _____

CE_1 = _____

CE_2 = _____

CE_3 = _____

WE_0 = _____

WE_1 = _____

WE_2 = _____

WE_3 = _____

Name:_____

Problem 3 (20 points):

A physically addressed cache is one in which the physical memory locations of each cache line can be inferred from its tag store information.

We want to design a physically addressed, 4-way set-associative, write back cache with a random replacement policy. Line size is 8 bytes. Virtual memory is byte-addressable, with a 32-bit address space, and the page size is 4kB. Physical memory has a 26-bit address space. The memory management system restricts even numbered virtual pages to even numbered physical frames and odd numbered virtual pages to odd numbered physical frames.

We want to be able to access the TLB, tagstore, and data store of this cache concurrently.

So we do not have to worry about back-translation tables, restrictions on shared frames, or invalidating the cache on a context switch, we insist that a line of physical memory cannot be present in more than one slot in the cache at the same time.

Part a (10 points): What is the largest size (in bytes) of the data store of the cache that will satisfy all of the above?

Bytes

Part b (10 points): What is the minimum size (in bits) of the corresponding tag store?

bits

Name:_____

Problem 4 (20 points)

A machine has a two-level virtual address translation mechanism similar to the VAX.

The machine's memory system is defined as follows:

Virtual Memory Size: 4KB
Physical Memory Size: 256 Bytes
Page Size: 32 Bytes
Memory is byte-addressable

Virtual address space is partitioned into P0 space, P1 space, system space and reserved space. The space a virtual address belongs to is specified by the most significant two bits of the virtual address, with 00 indicating P0 space, 01 indicating P1 space, and 10 indicating system space.

A PTE consists of a Valid bit, 3 bits for protection, a Reference bit, a Modified bit, and the Page Frame Number (PFN).

Part a (2 points): What is the minimum size of a PTE in **Bytes**?

[ ] Bytes

Part b (2 points): Using the PTE size from part a, what is the maximum size of a process' P0 page table in **Bytes**?

[ ] Bytes

Name:_____

Problem 4 continued

Part c (8 points): If P0BR is x8C0 and the SBR is xC0, what is the physical address corresponding to virtual address x243? Assume that the valid bit of a PTE is its most significant bit and the PFN is stored in its least significant bits.

The following blocks are the contents of a selected areas of physical memory that you may need to solve this problem:

xCE	x8000	xE0	x8004	x00	x0002	x80	x0438
xD0	x0007	xE2	x00A0	x02	x8002	x82	x0000
xD2	x8003	xE4	x0003	x04	x8001	x84	x8005
xD4	x0000	xE6	x0007	x06	x0134	x86	x0002

Note: There may be more information contained here than needed to solve the problem.

Physical Address: [ ]

Name:_____

Problem 4 continued

Part d (3 points): Let's say that during the translation of virtual address x243, the valid bit of the 2nd PTE encountered was a 0, resulting in a page fault. What range of virtual addresses in P0 space would also result in this page fault?

From [ ] to [ ]

Part e (5 points): Let's say that during the translation of virtual address x243, the valid bit of the 1st PTE encountered was a 0, resulting in a page fault. What range of virtual addresses in P0 space would also result in this page fault?

From [ ] to [ ]

Name:_____

Problem 5 (20 points):

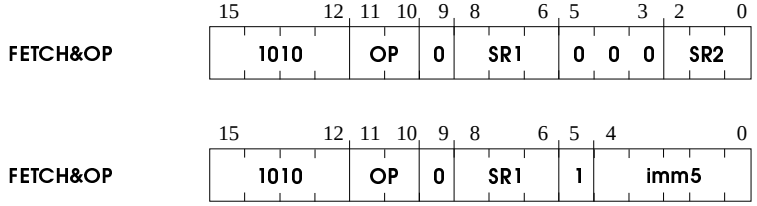
We wish to use the unused opcode 1010 to implement a new instruction FETCH&OP, which loads a word from memory, performs an operation (OP) on it, and stores the result back to the same memory location. The original value which was loaded from memory is saved in R7, and the condition codes are set based on that value. OP can be ADD, AND, or XOR. The second operand can be either the contents of a register or an immediate value. The specification of this instruction is as follows:

Assembler Formats
FETCH&ADD SR1, SR2
FETCH&ADD SR1, imm5

FETCH&AND SR1, SR2
FETCH&AND SR1, imm5

FETCH&XOR SR1, SR2
FETCH&XOR SR1, imm5

Encodings



Encoding of the 2-bit value OP is as follows:
ADD/00
AND/01
XOR/10
The value 11 is reserved

Operation

```
TEMP =  MEM[SR1];
if ( bit[5] == 0 )
     MEM[SR1] = TEMP OP SR2;
else
     MEM[SR1] = TEMP OP SEXT(imm5);
R7 = TEMP;
setcc();
```

Name:_____

Problem 5 continued:

To implement FETCH&OP, we have added the following structures to the LC-3b data path:

1. A 16-bit register called TEMP
   This requires a new control field, LD.TEMP, which is the load enable signal for the TEMP register.
2. A two-input mux, AMUX, to the A input of the ALU.
   This requires a new control field, AMUX, specified as follows:
   SR1/0, the SR1 source from the register file
   TEMP/1, the value of the TEMP register
3. A two-input mux, ALUMUX, to the control input of the ALU.
   This requires a new control field, ALUMUX, specified as follows:
   ALUK/0, the ALUK bits from the control store
   X/1, the 2-bit signal X

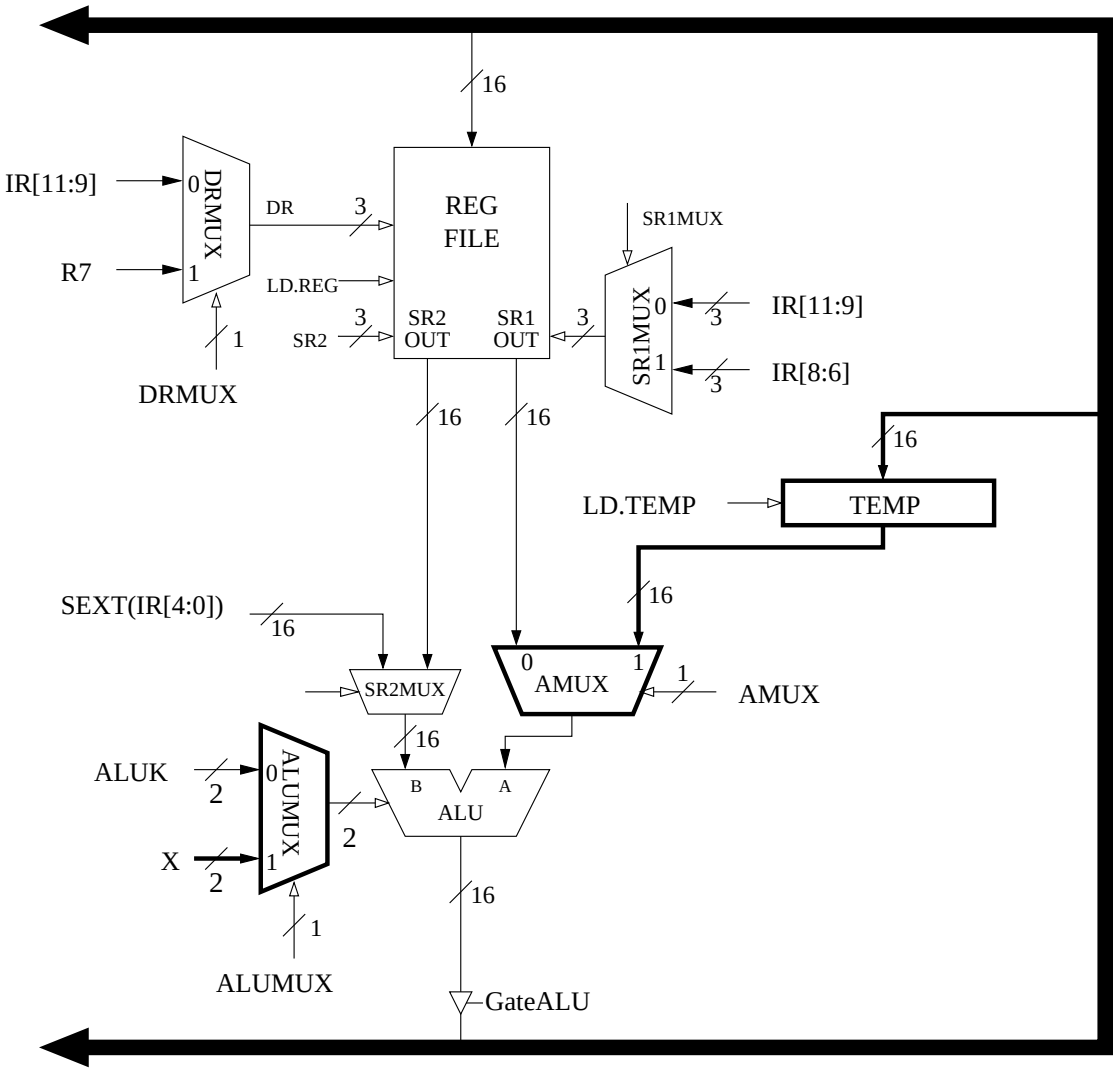


Figure 1: Modified datapath to support FETCH&OP instruction

Name:_____

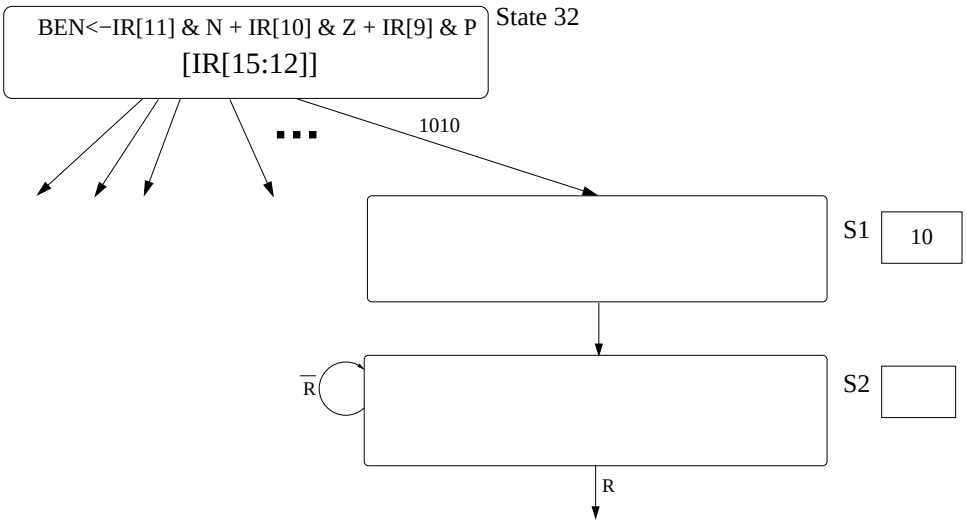
Problem 5 continued:

Part a (2 points): Identify the 2-bit signal X shown on the data path diagram:

X: [ ]

Part b (10 points): We show the beginning of the state diagram necessary to implement FETCH&OP. Using the notation of the LC-3b State Diagram, add the bubbles you need to implement the FETCH&OP instruction. Your job is to **describe** inside each bubble what happens in each state and **assign** each state an appropriate state number. You should be able to implement this state diagram using 5 or 6 new states.

NOTE: Your solution must work for instructions where SR2 is R7 e.g. FETCH&ADD R1, R7



Name:_____

Problem 5 continued:

Part c (8 points): The processing in each state you just added is controlled by asserting or negating each control signal. Enter a 1 or a 0 as appropriate for the microinstructions corresponding to the states you have added.

NOTE: Please use the encodings specified on the next page for all the signals.

	LD.REG	*LD.CC*	*LD.TEMP*	*LD.MAR*	*LD.MDR*	*AMUX*	*SR1MUX*	*DRMUX*	*ALUK[1:0]*		*ALUMUX*	*GateALU*	*GateMDR*	*MIO.EN*	*R.W.*	*DATA.SIZE*
S1																
S2																
S3																
S4																
S5																
S6																

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	

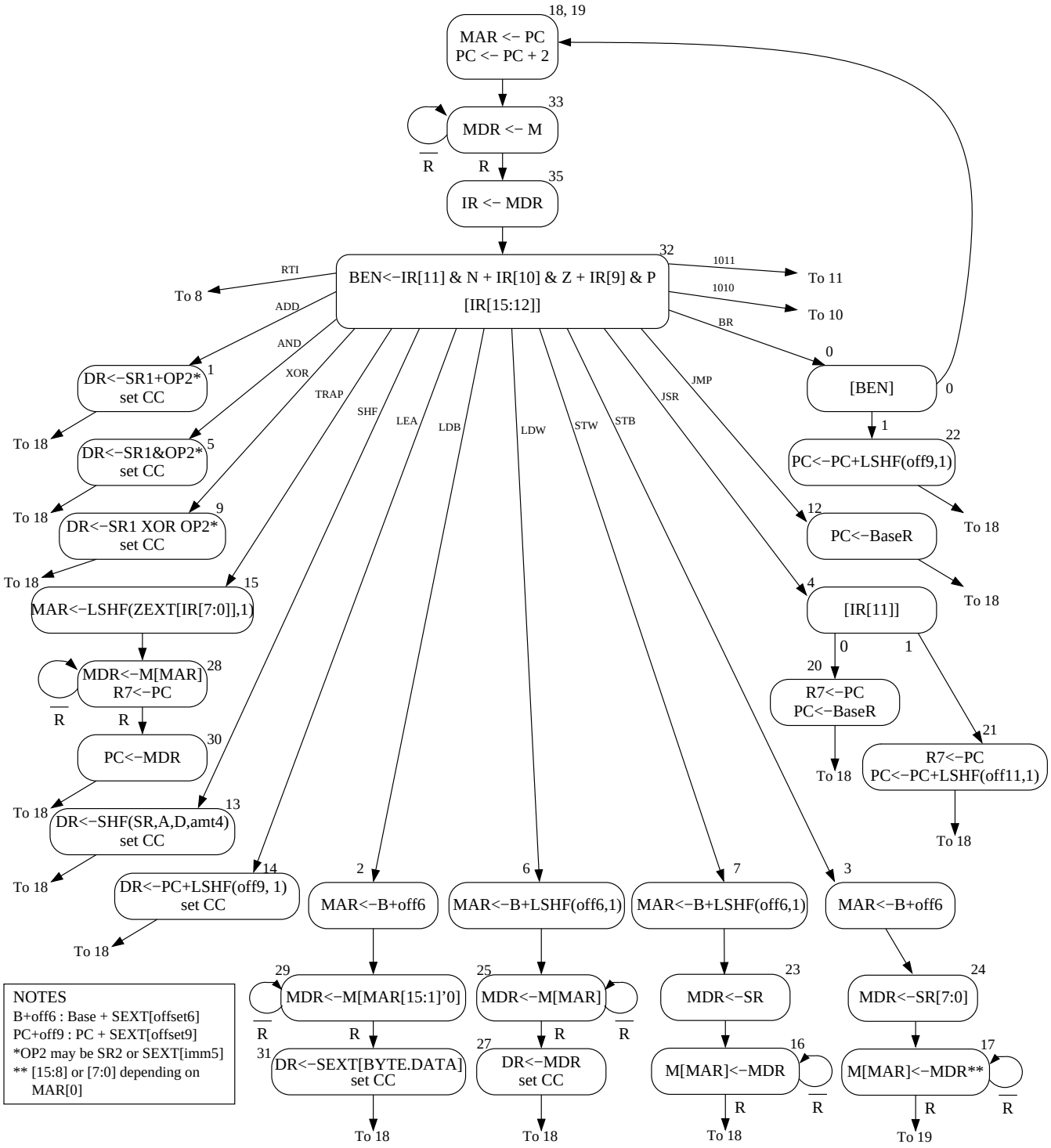


Figure 2: A state machine for the LC-3b

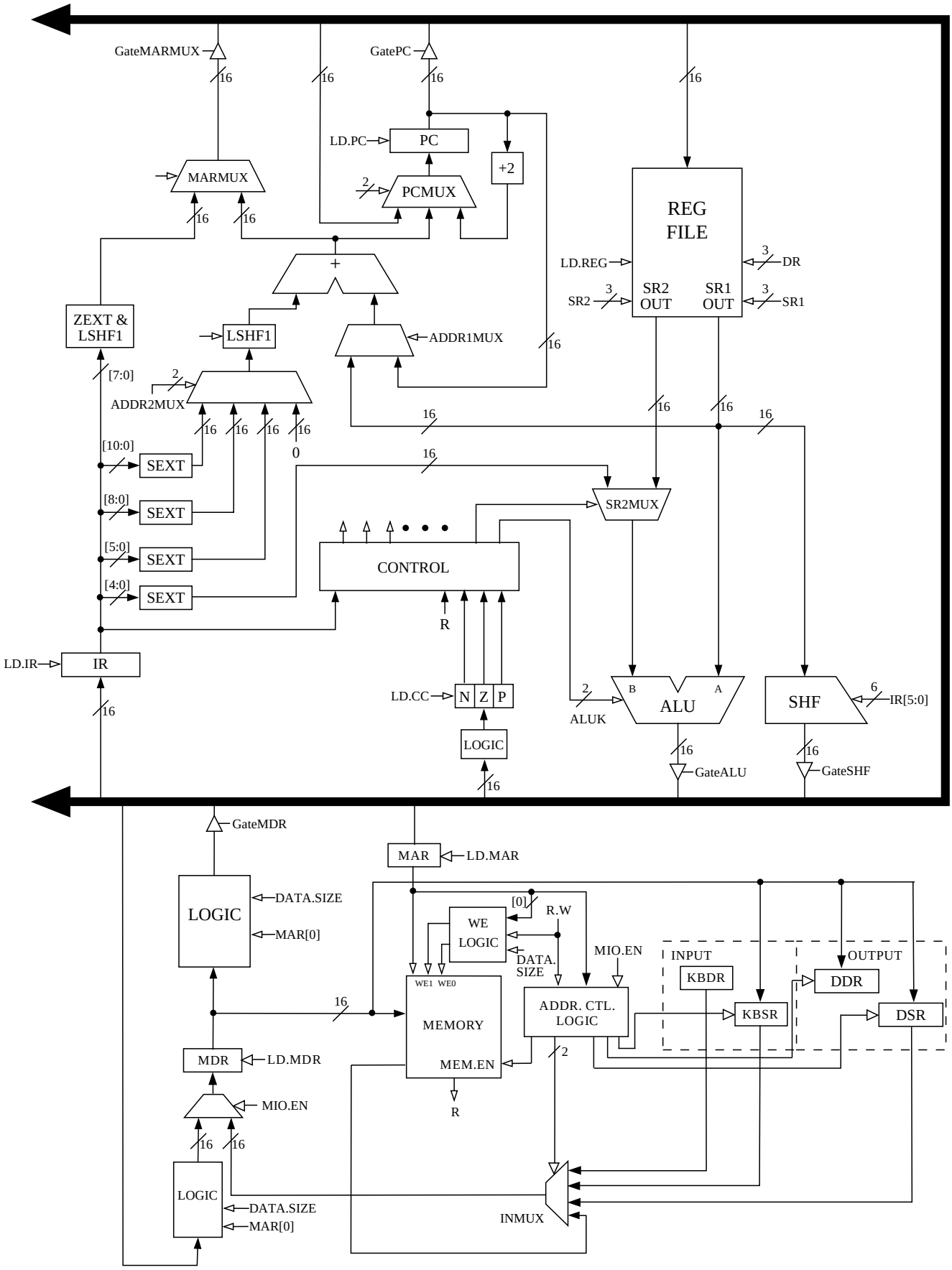


Figure 3: The LC-3b data path

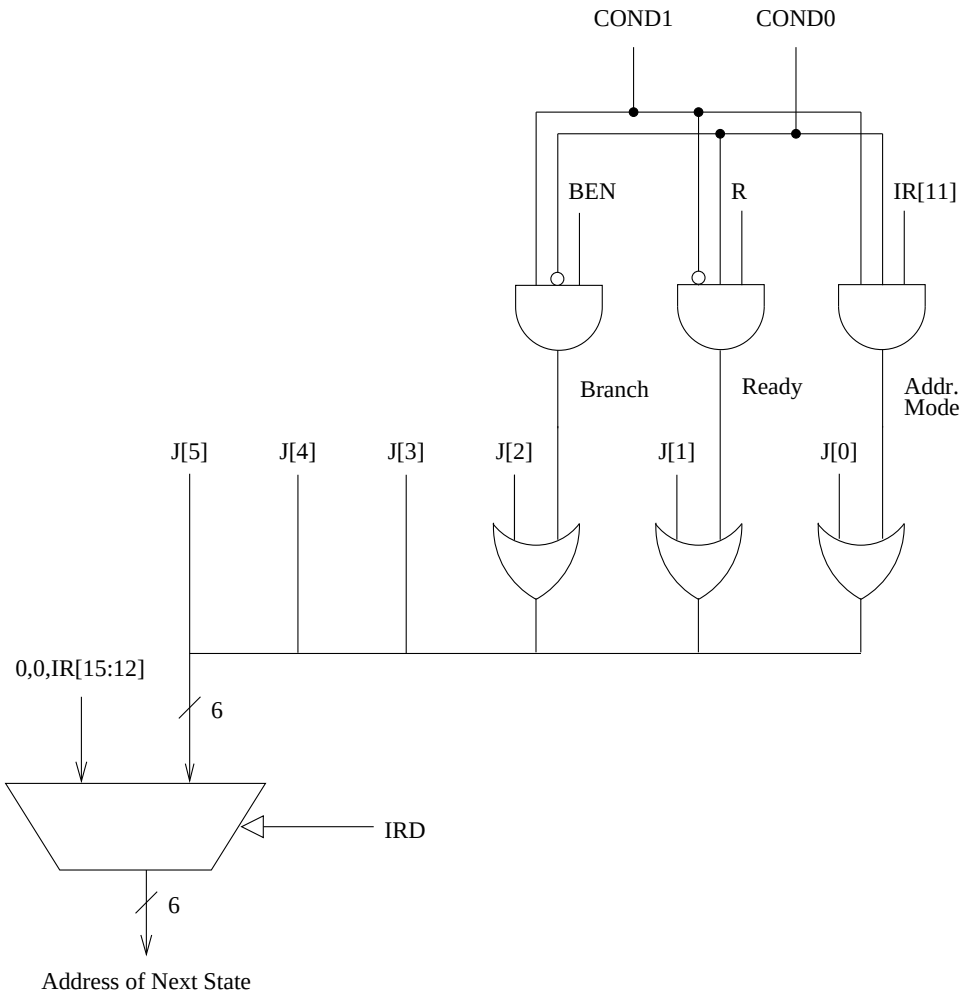


Figure 4: The microsequencer of the LC-3b base machine