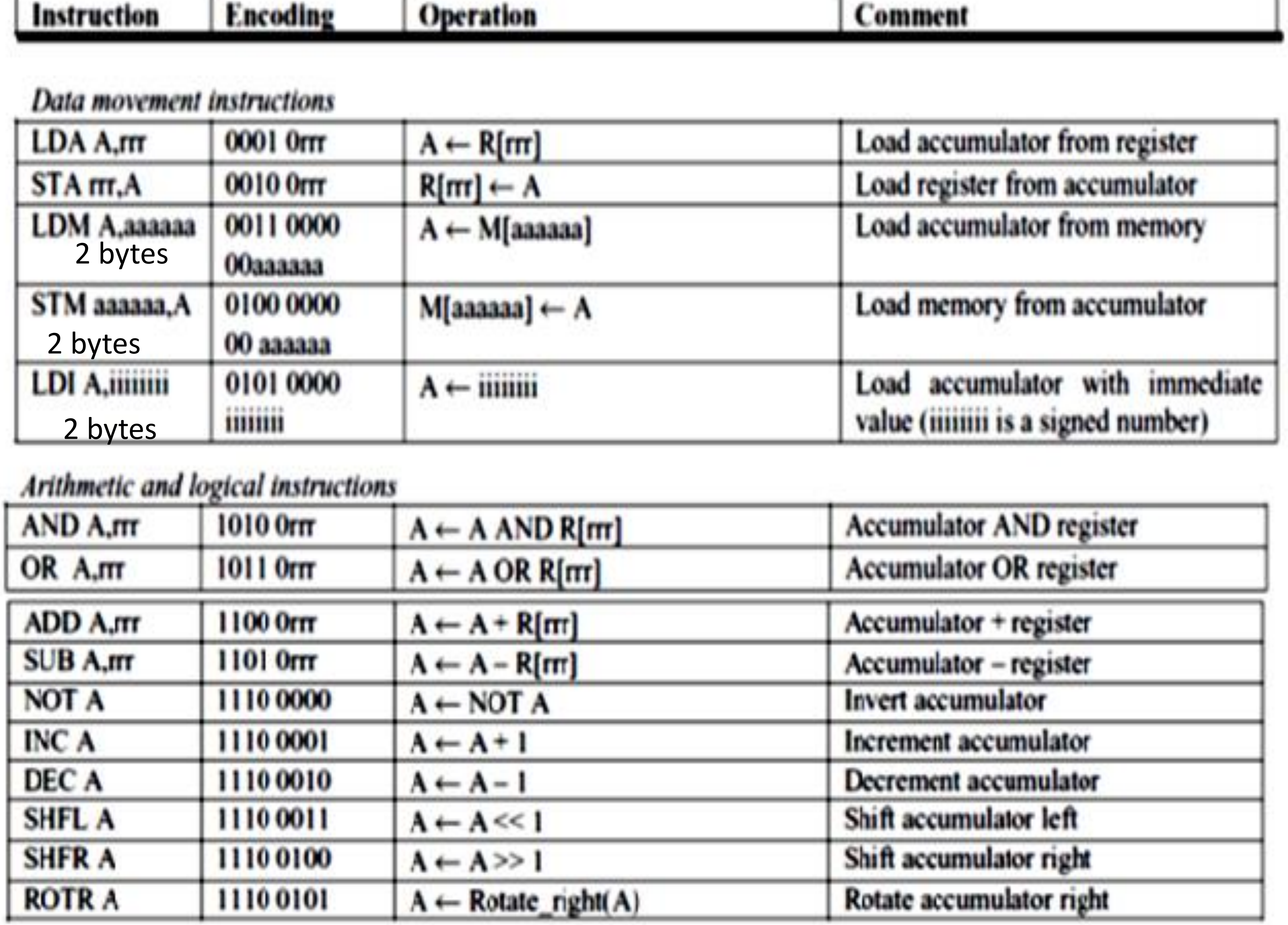
Course project

Details

ISA

- rrr – register
- Aaaaaa –MA
- iiiiiii – imm.



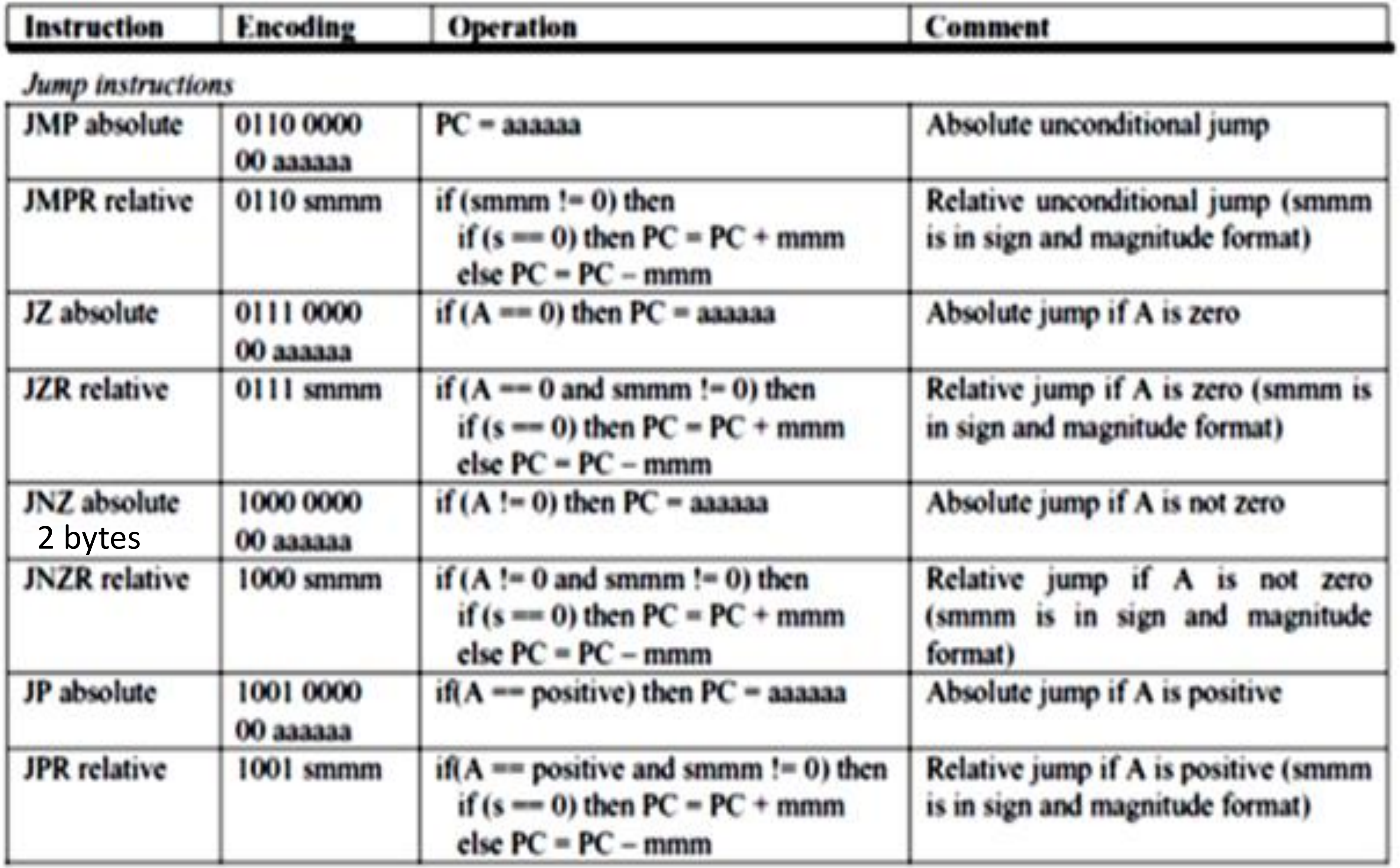
Instruction	Encoding	Operation	Comment
Data movement instructions			
LDA A,rrr	0001 0rrr	A ← R[rrr]	Load accumulator from register
STA rrr,A	0010 0rrr	R[rrr] ← A	Load register from accumulator
LDM A,aaaaaa 2 bytes	0011 0000 00aaaaaa	A ← M[aaaaaa]	Load accumulator from memory
STM aaaaaa,A 2 bytes	0100 0000 00 aaaaaa	M[aaaaaa] ← A	Load memory from accumulator
LDI A,iiiiiiii 2 bytes	0101 0000 iiiiiiii	A ← iiiiiiii	Load accumulator with immediate value (iiiiiiii is a signed number)

Arithmetic and logical instructions

Instruction	Encoding	Operation	Comment
AND A,rrr	1010 0rrr	A ← A AND R[rrr]	Accumulator AND register
OR A,rrr	1011 0rrr	A ← A OR R[rrr]	Accumulator OR register
ADD A,rrr	1100 0rrr	A ← A + R[rrr]	Accumulator + register
SUB A,rrr	1101 0rrr	A ← A – R[rrr]	Accumulator – register
NOT A	1110 0000	A ← NOT A	Invert accumulator
INC A	1110 0001	A ← A + 1	Increment accumulator
DEC A	1110 0010	A ← A – 1	Decrement accumulator
SHFL A	1110 0011	A ← A << 1	Shift accumulator left
SHFR A	1110 0100	A ← A >> 1	Shift accumulator right
ROTR A	1110 0101	A ← Rotate_right(A)	Rotate accumulator right

ISA

- Smmm –Sign and magn.



Instruction	Encoding	Operation	Comment
Jump instructions			
JMP absolute	0110 0000 00 aaaaaa	PC = aaaaaa	Absolute unconditional jump

JMPR relative	0110 smmm	if (smmm != 0) then
		if (s == 0) then PC = PC + mmm
		else PC = PC – mmm
			Relative unconditional jump (smmm is in sign and magnitude format)

JZ absolute	0111 0000 00 aaaaaa	if (A == 0) then PC = aaaaaa	Absolute jump if A is zero

JZR relative	0111 smmm	if (A == 0 and smmm != 0) then
		if (s == 0) then PC = PC + mmm
		else PC = PC – mmm
			Relative jump if A is zero (smmm is in sign and magnitude format)

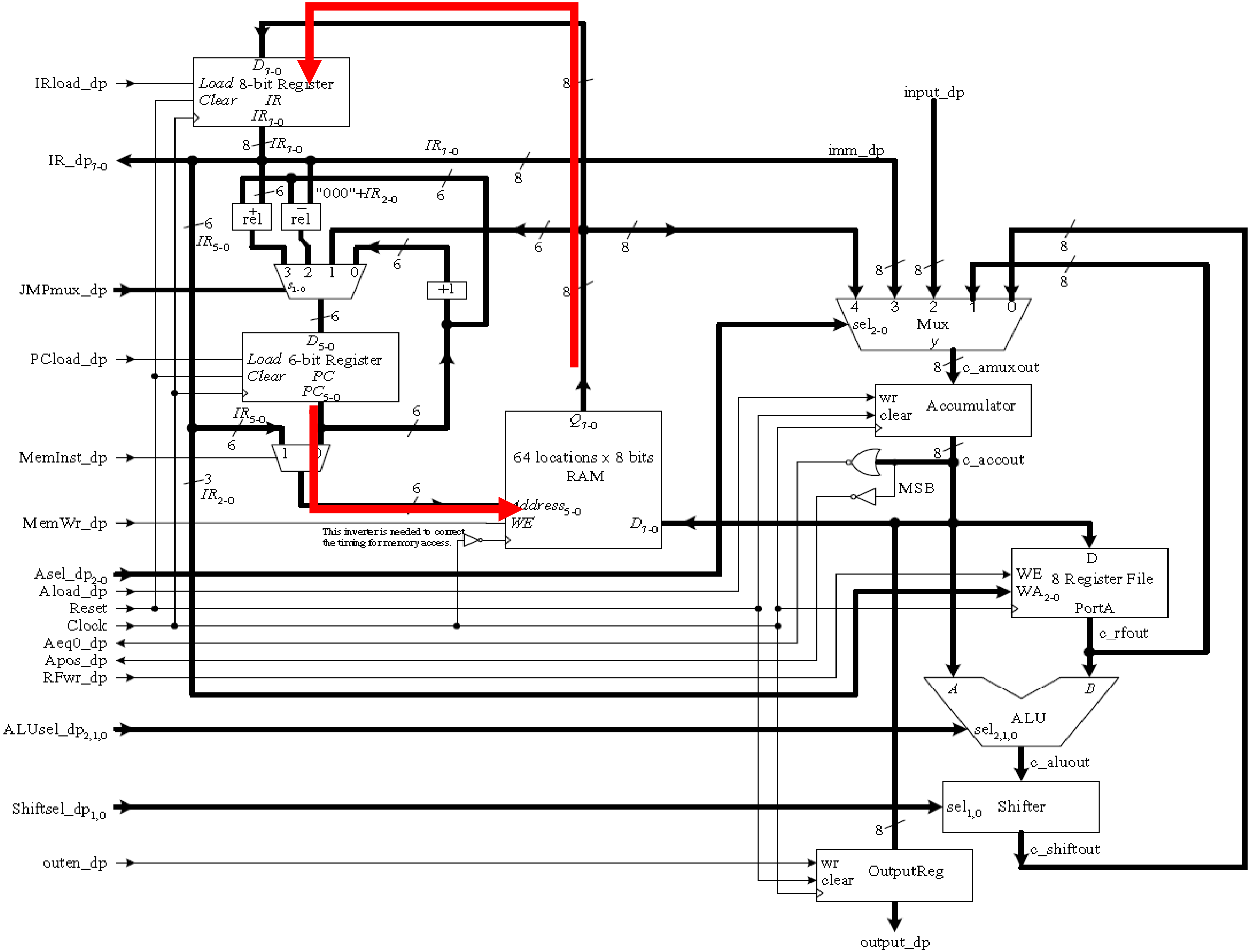
JNZ absolute 2 bytes	1000 0000 00 aaaaaa	if (A != 0) then PC = aaaaaa	Absolute jump if A is not zero

JNZR relative	1000 smmm	if (A != 0 and smmm != 0) then
		if (s == 0) then PC = PC + mmm
		else PC = PC – mmm
			Relative jump if A is not zero (smmm is in sign and magnitude format)

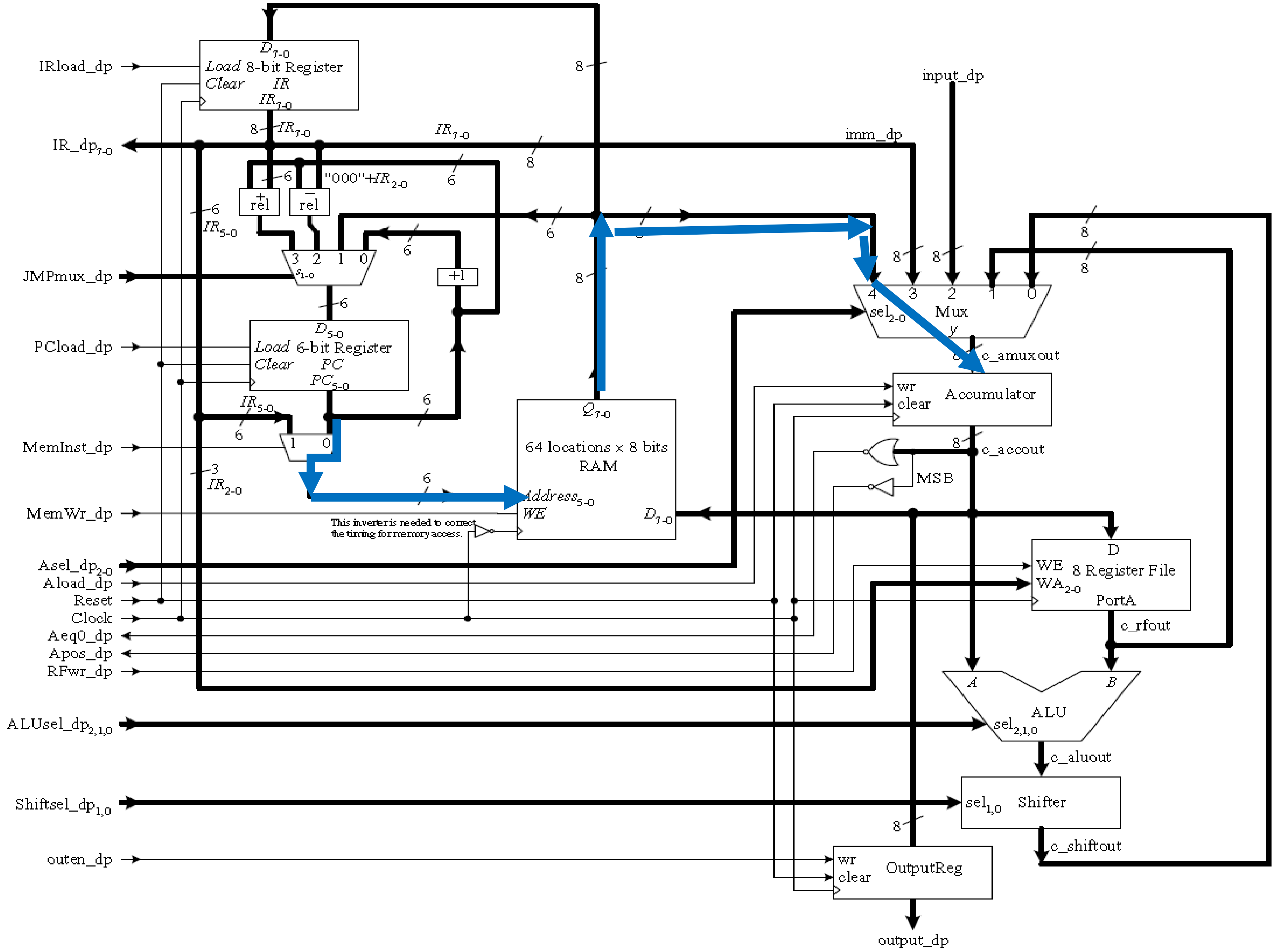
JP absolute	1001 0000 00 aaaaaa	if(A == positive) then PC = aaaaaa	Absolute jump if A is positive

JPR relative	1001 smmm	if(A == positive and smmm != 0) then
		if (s == 0) then PC = PC + mmm
		else PC = PC – mmm
			Relative jump if A is positive (smmm is in sign and magnitude format)

Fetch



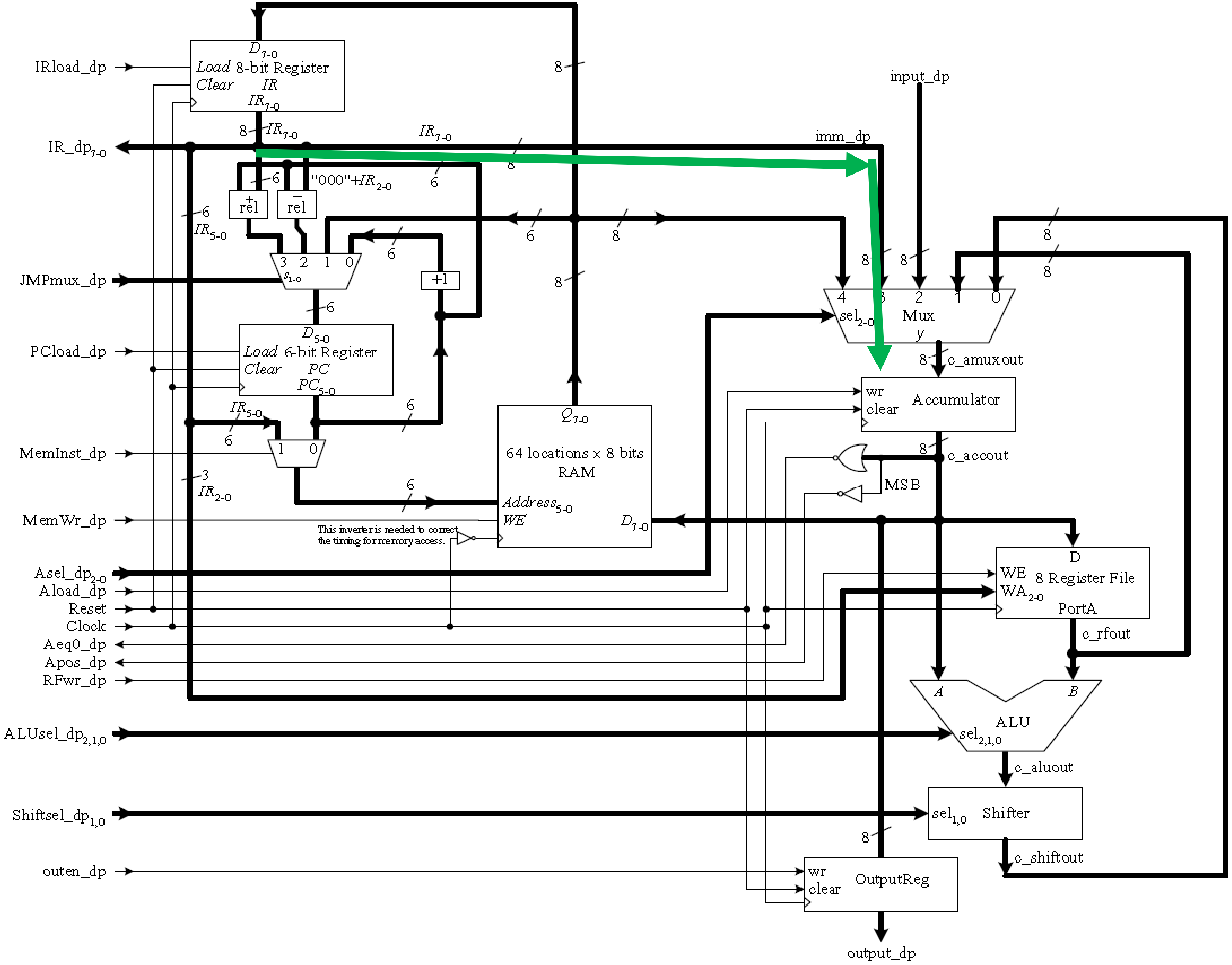
- Fetch:every cycle, PC=PC+1



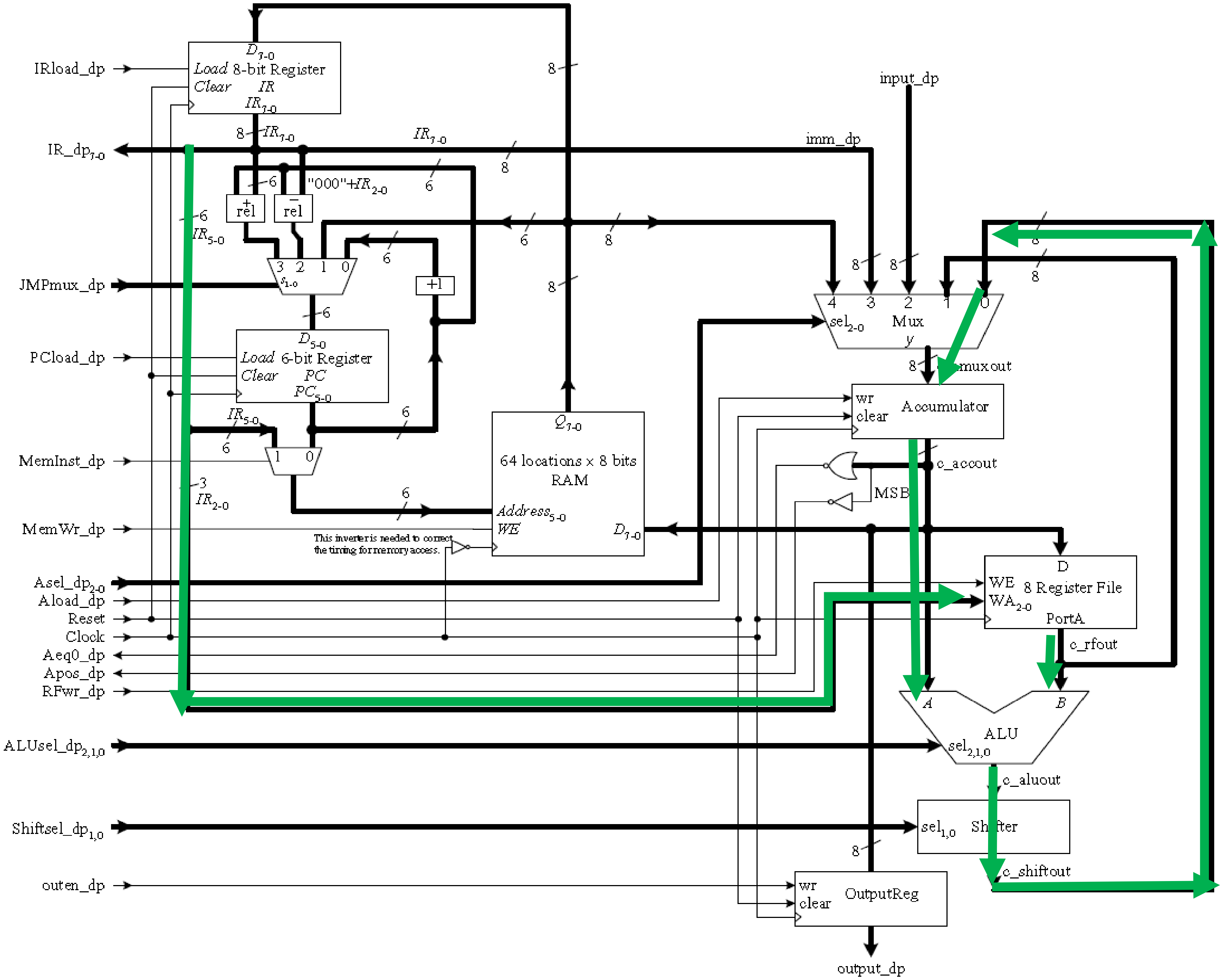
LDM, 2^{nd} cycle

- For 2-byte instruction, read memory for data (or instruction) by the address provided directly from PC occurs in the second cycle;

LDI, 2nd cycle



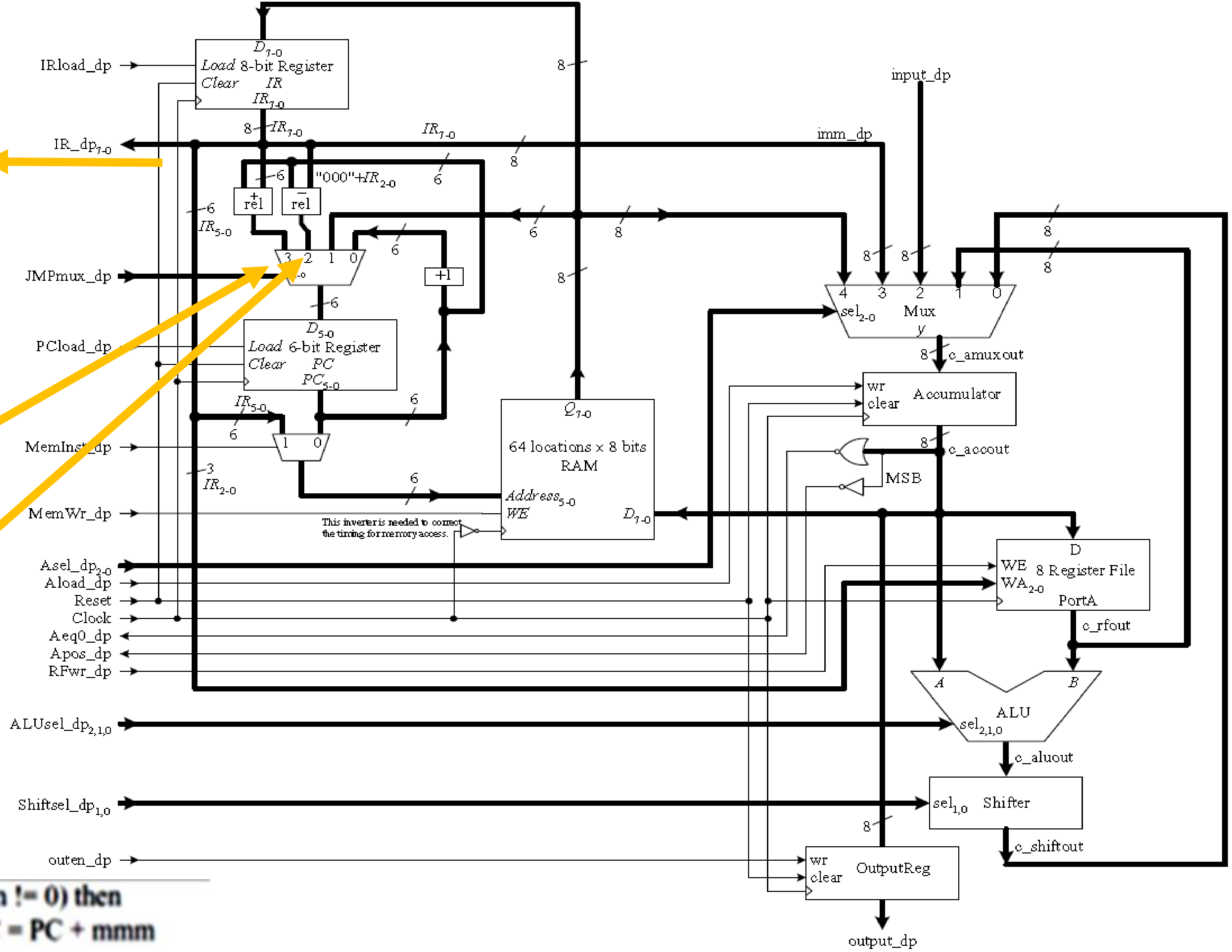
Register ALU op 2nd cycle

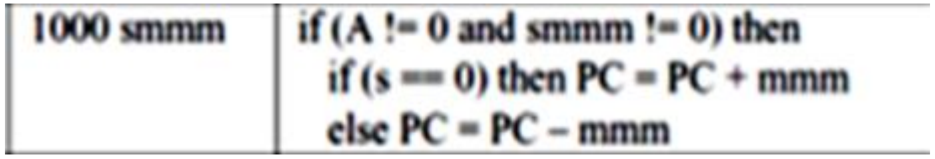


JNZR

- After fetch the Control unit has IR <7..0>
- If IR<3..0> != 0

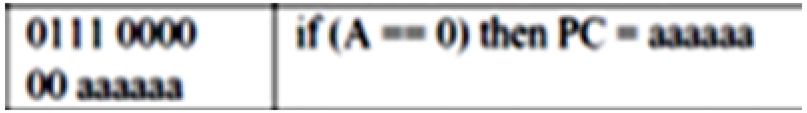
If IR<3>=0 then JMPmux=11, otherwise JMPmux=10,



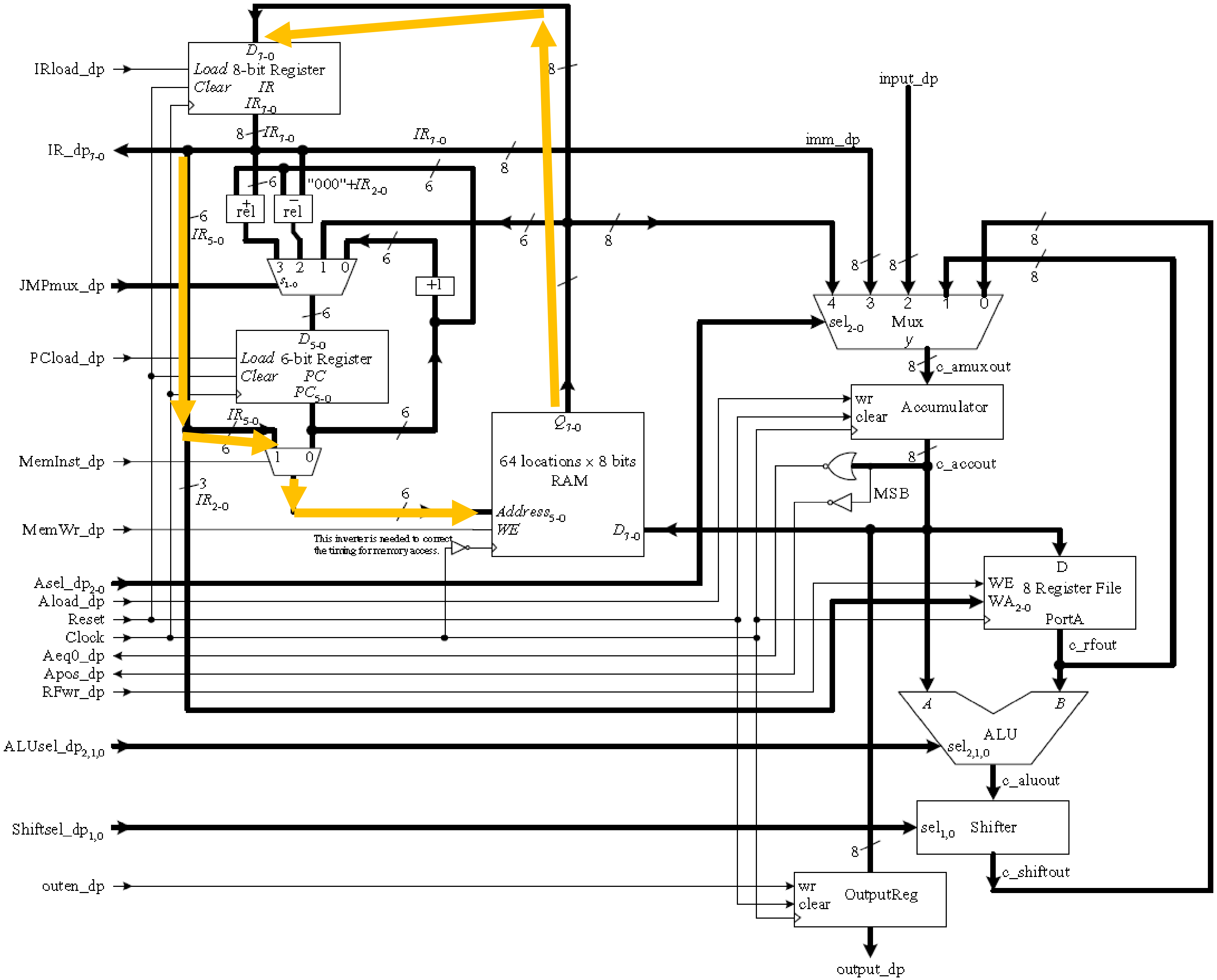


1000 smmm	if (A != 0 and smmm != 0) then
	if (s == 0) then PC = PC + mmm
	else PC = PC – mmm

JZ

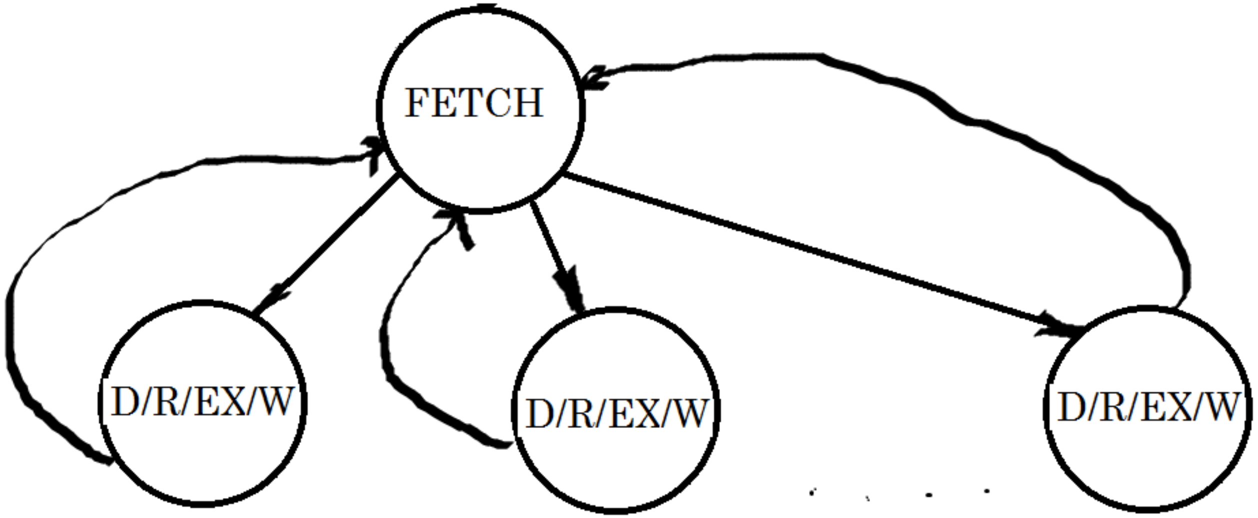


0111 0000 00 aaaaaa	if (A == 0) then PC = aaaaaa



2nd cycle

- After Fetch is complete, and the instruction in the IR, CU knows the type of operation
- All instructions will complete thereafter in the second cycle



Finish for all instructions/group of instructions

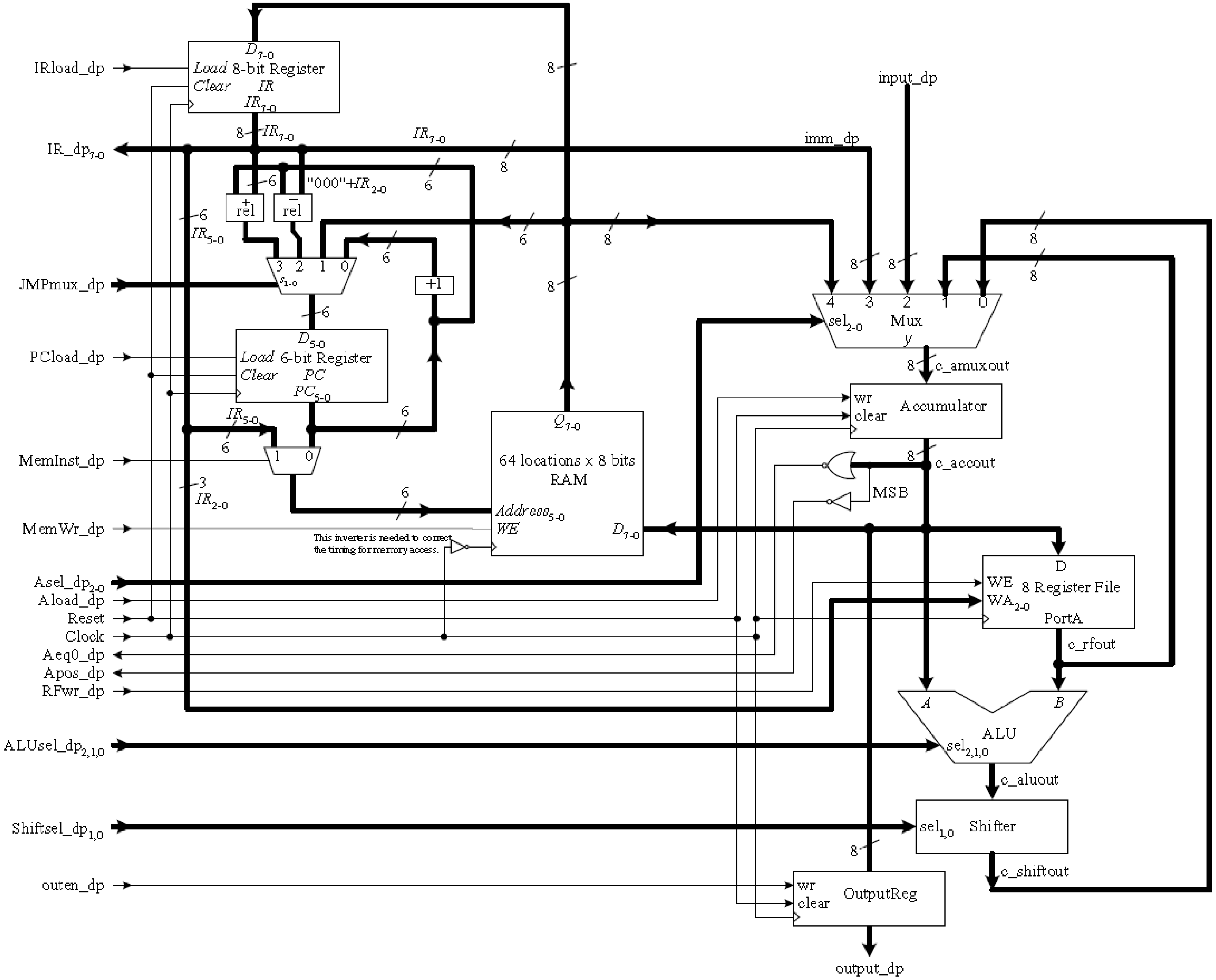
Next State logic:

- If State 0 nextstate=State 1
- If State 1: nextstate=State 0

Control Signals:

Fetch

- IRLoad =1
- MemWr=0
- MemInst=0
- JMPMux=00



2nd cycle

- Have to do