Course project

Modified DP

Instruction set

- rrr – register
- Aaaaaa –MA
- iiiiiiii – 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	0011 0000 00aaaaaa	A ← M[aaaaaa]	Load accumulator from memory
STM aaaaaa,A	0100 0000 00 aaaaaa	M[aaaaaa] ← A	Load memory from accumulator
LDI A,iiiiiiii	0101 0000 iiiiiiii	A ← iiiiiiii	Load accumulator with immediate value (iiiiiiii is a signed number)
Arithmetic and logical instructions			
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

Instruction set

- Smmm –Sign and magn.

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	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

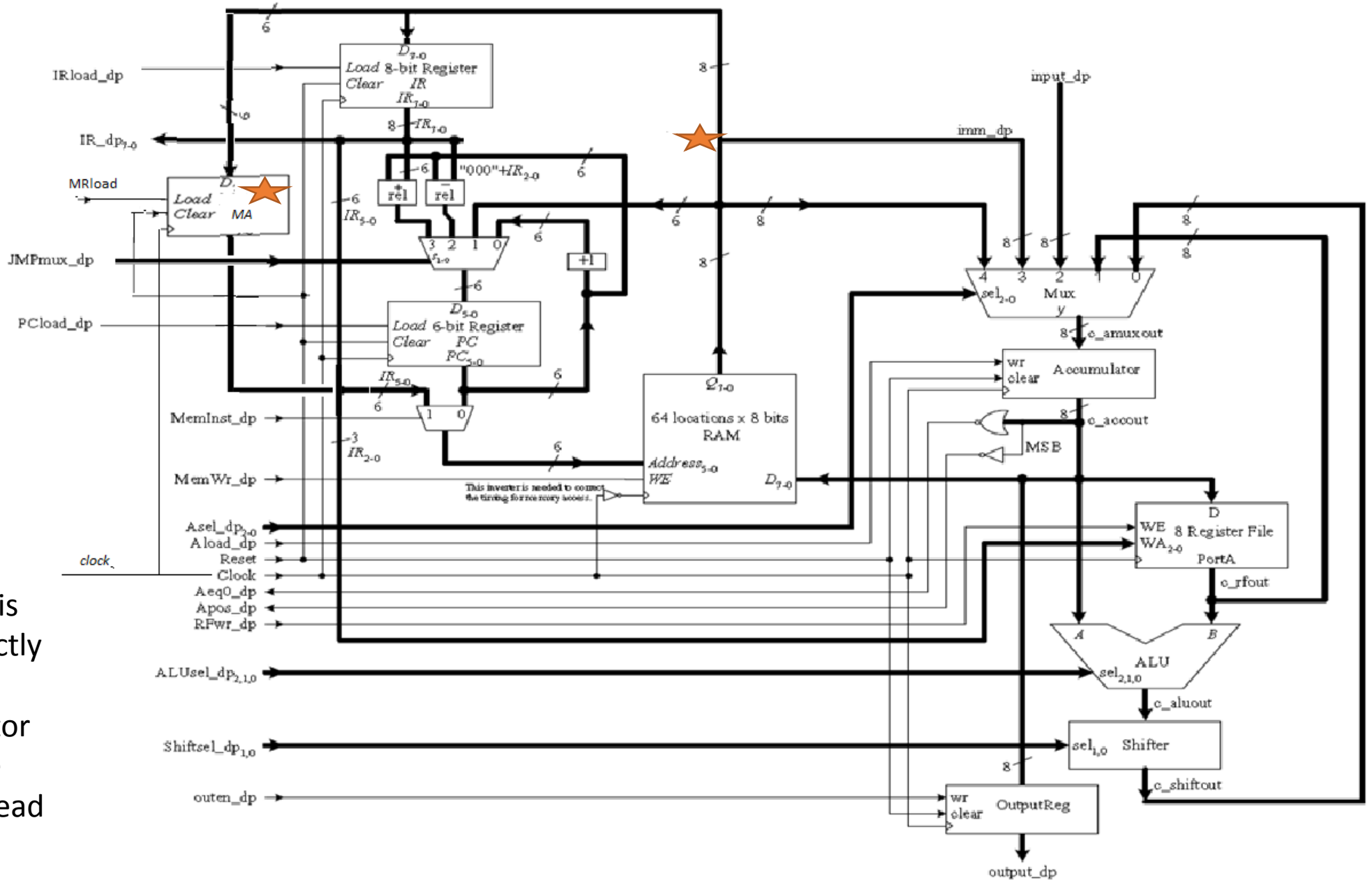
JPR relative	1001 smmm	if(A == positive and smmm != 0) then
		if (s == 0) then PC = PC + mmm
		else PC = PC – mmm
			Relative jump if A is positive (smmm is in sign and magnitude format)

Input / Output and Miscellaneous

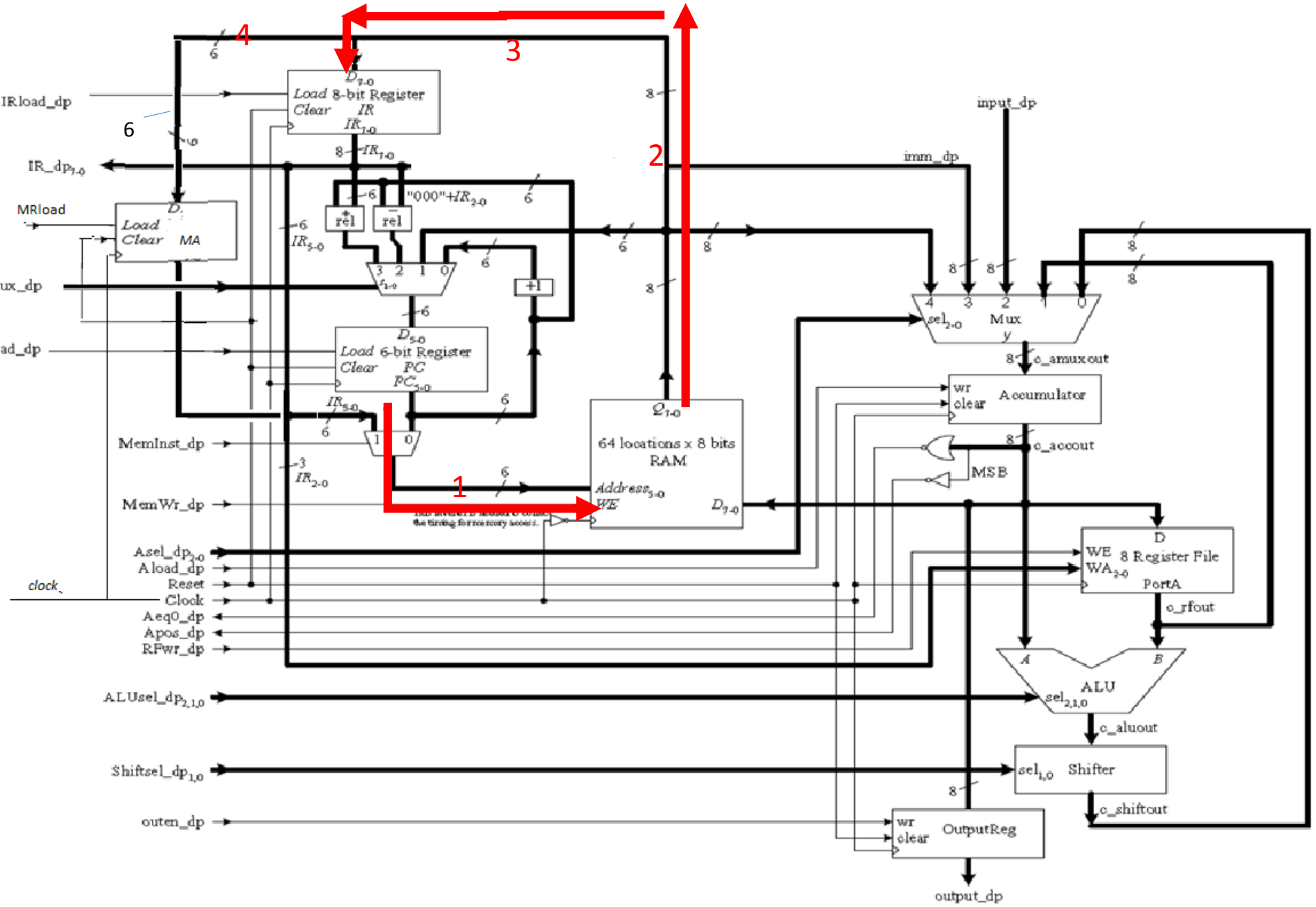
In A	1111 0000	A ← input	Input to accumulator
Out A	1111 0001	output ← A	Output from accumulator
HALT	1111 0010	Halt	Halt execution
NOP	0000 0000	no operation	No operation

Modified DP

- Added memory Address register
- Imm field is given directly to the accumulator during the memory read cycle

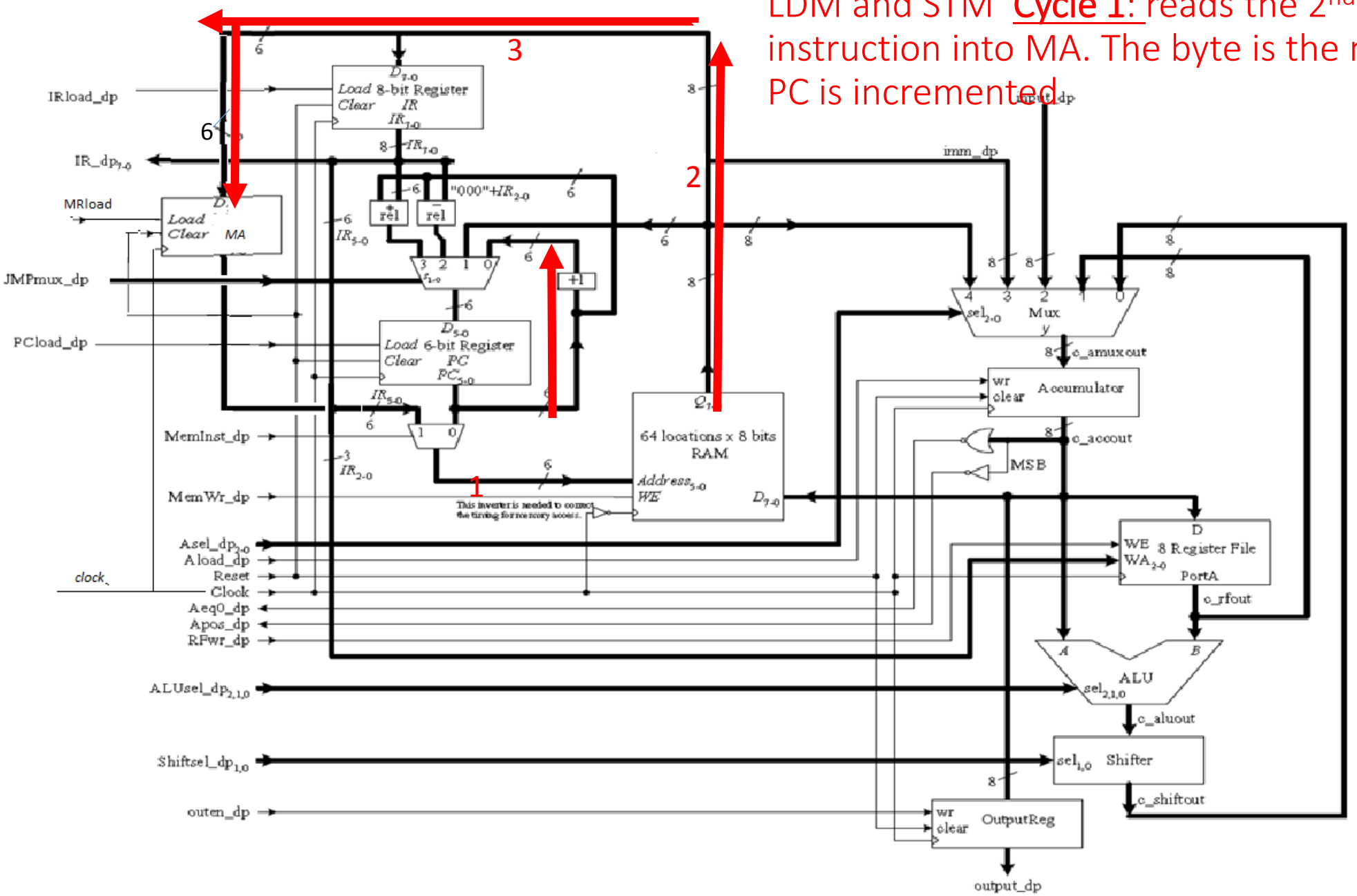


Fetch in Cycle 0



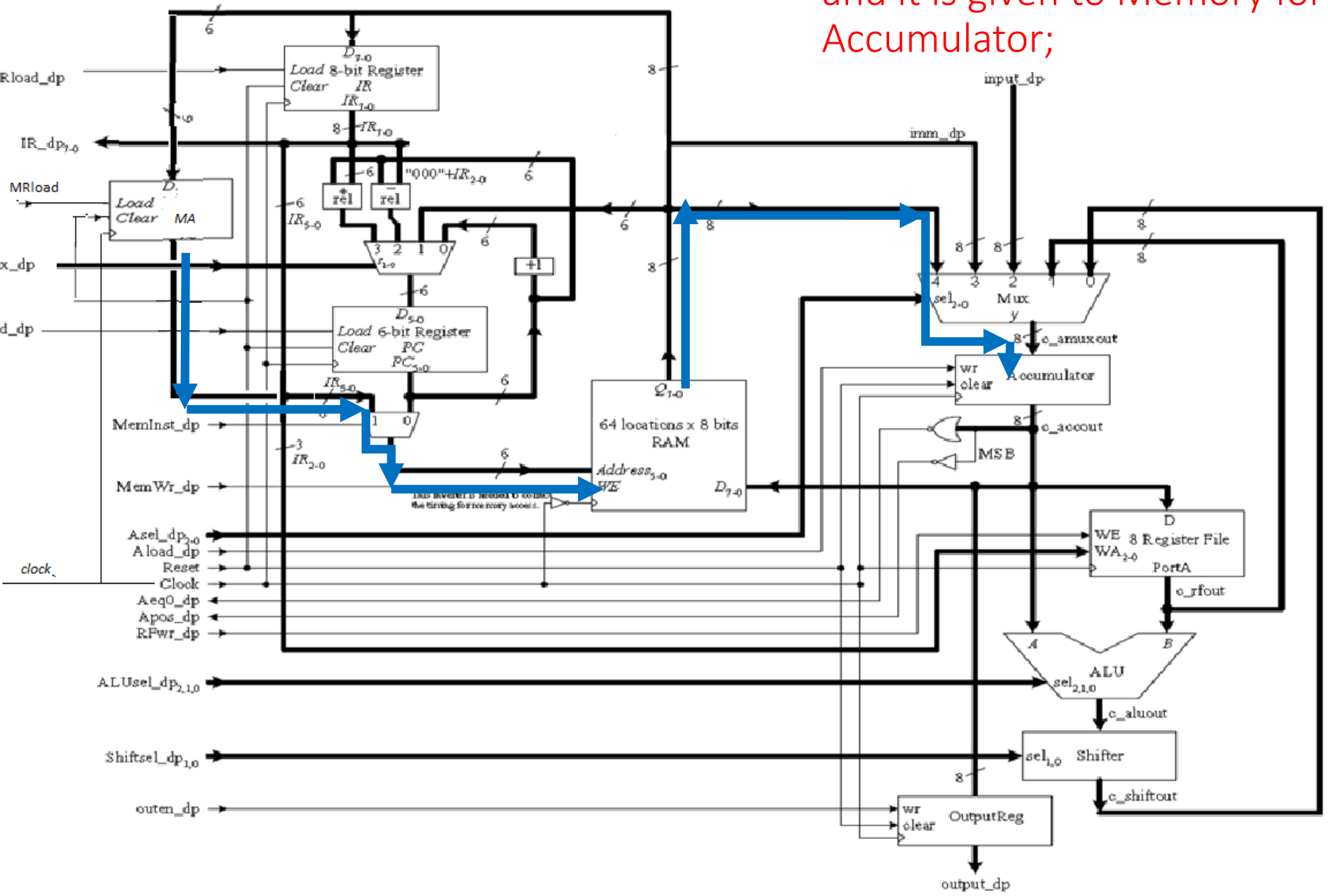
- IR loaded in Cycle 0 with the first byte (could be the only byte) of Instruction.
- It is provided to CU in the beginning of the Cycle 1, the decode delay is included into the cycle (as in multi-cycle MIPS) and the decision is made on the control signals to assert in the second cycle: Cycle 1
- PC is incremented and loaded.

LDM and STM Cycle 1: reads the 2nd byte of the instruction into MA. The byte is the memory address. PC is incremented.

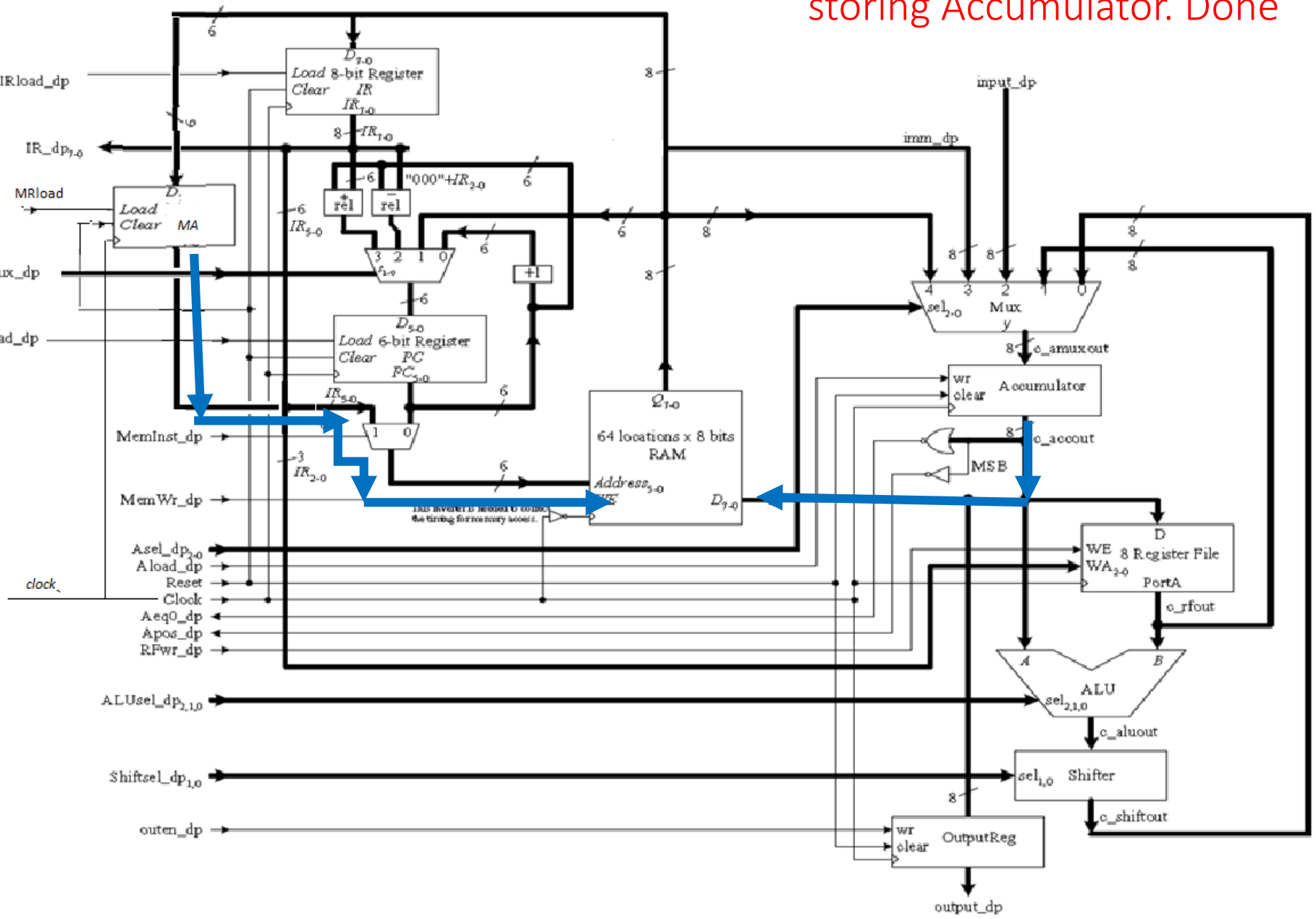


LDM A,aaaaaa	0011 0000 00aaaaaa	A ← M[aaaaaa]

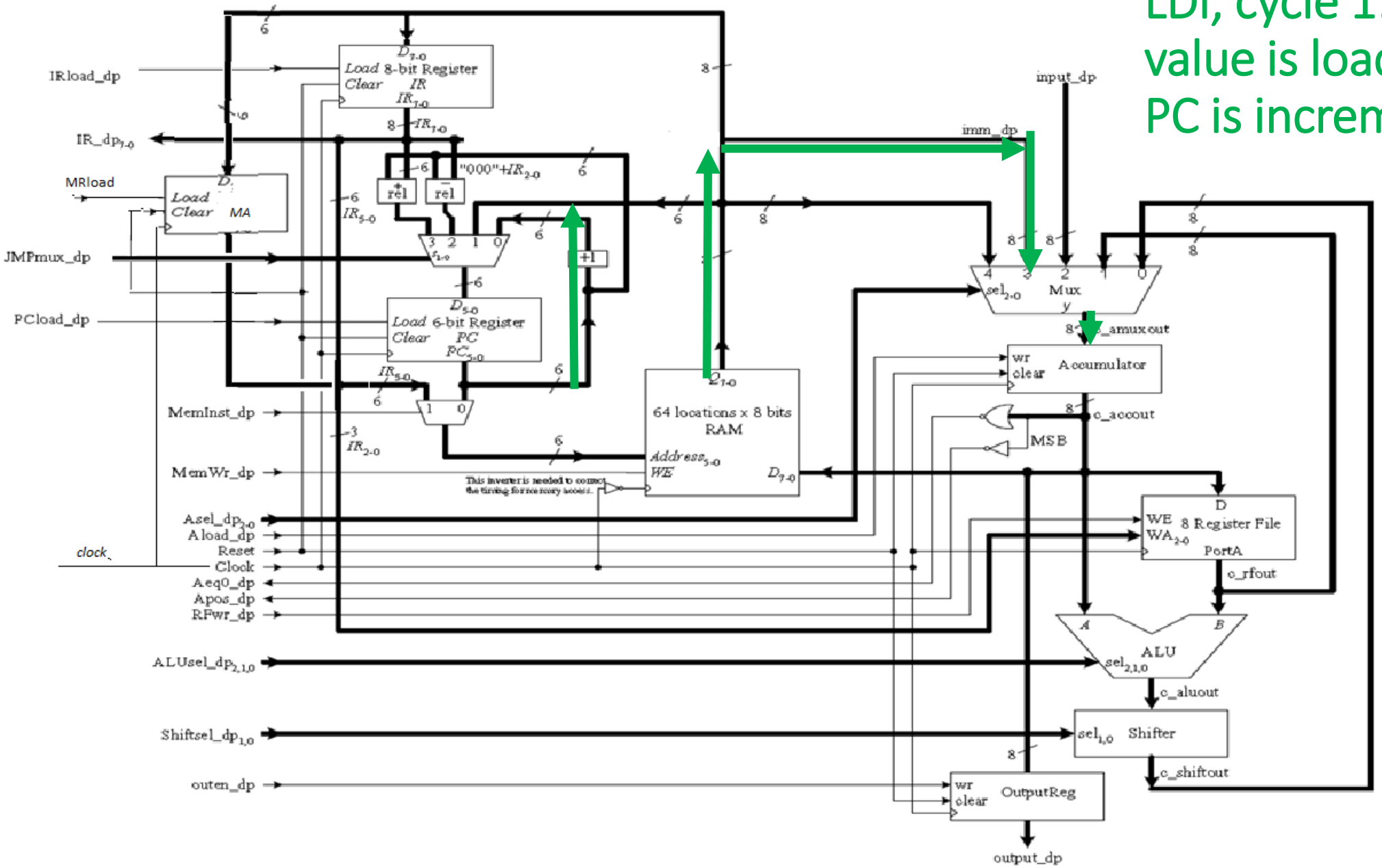
LDM Cycle 2: reads the IR. The byte is the memory addre and it is given to Memory for reading data into Accumulator;



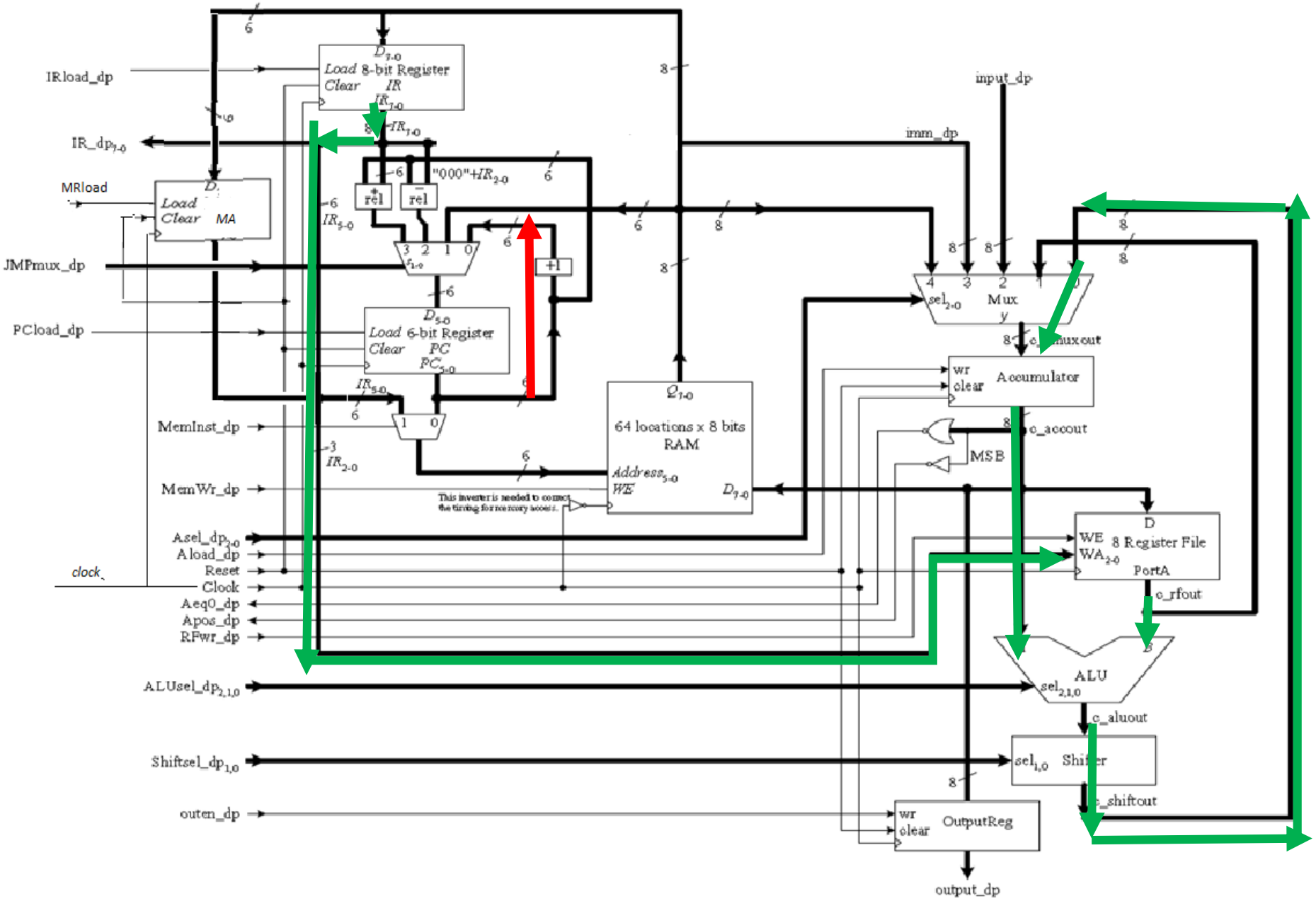
STM Cycle 2: reads Memory address is given to Memory f
storing Accumulator. Done



LDI, cycle 1: immediate value is loaded: PC is incremented



Register ALU op
Cycle 1: perform the operation; increment PC

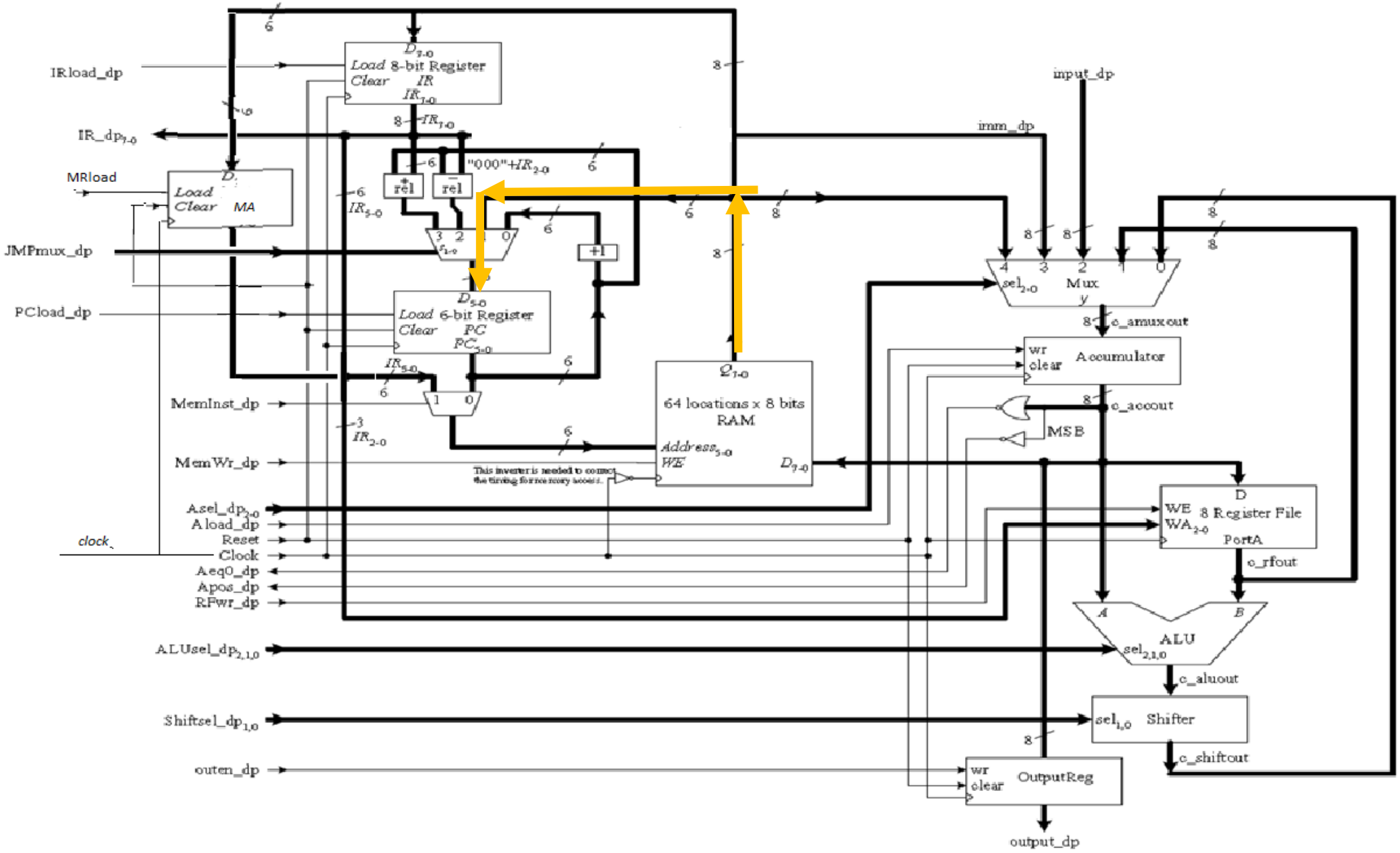


JZ
Cycle 1

- Read 6 bits of

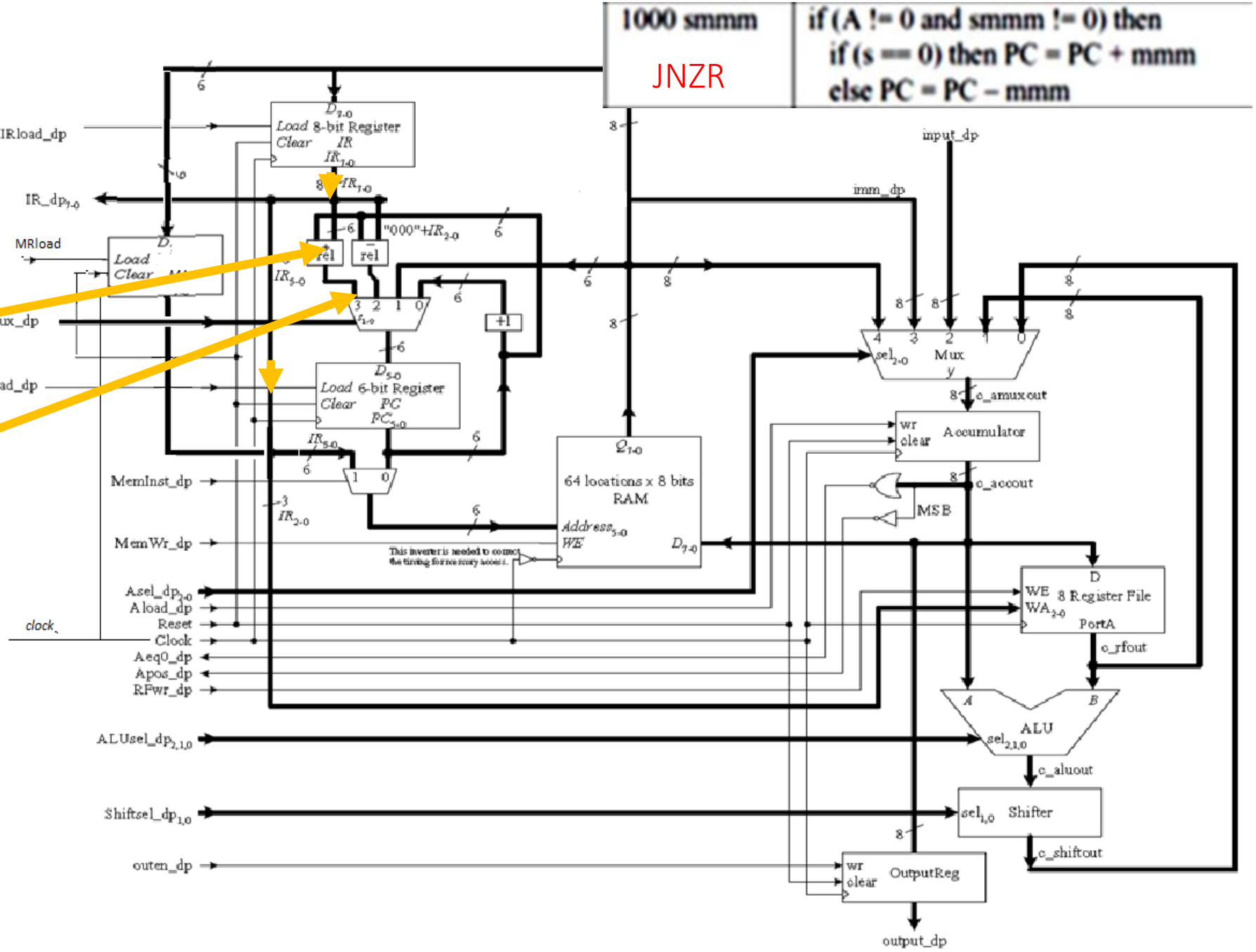
the 2 nd byte to PC

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



JNZR Cycle 1

- If IR_dp<3..0> != 0 then If IR<3>=0 then JMPmux=11, else JMPmux=10



Cycles per instruction

- Some instructions complete in 2 cycles , such as NOP and Halt (Fetch and Decode), jumps,arithmetic, in, out
- LDM and STM complete in 3 cycles
- Unfinished FSM is shown