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)

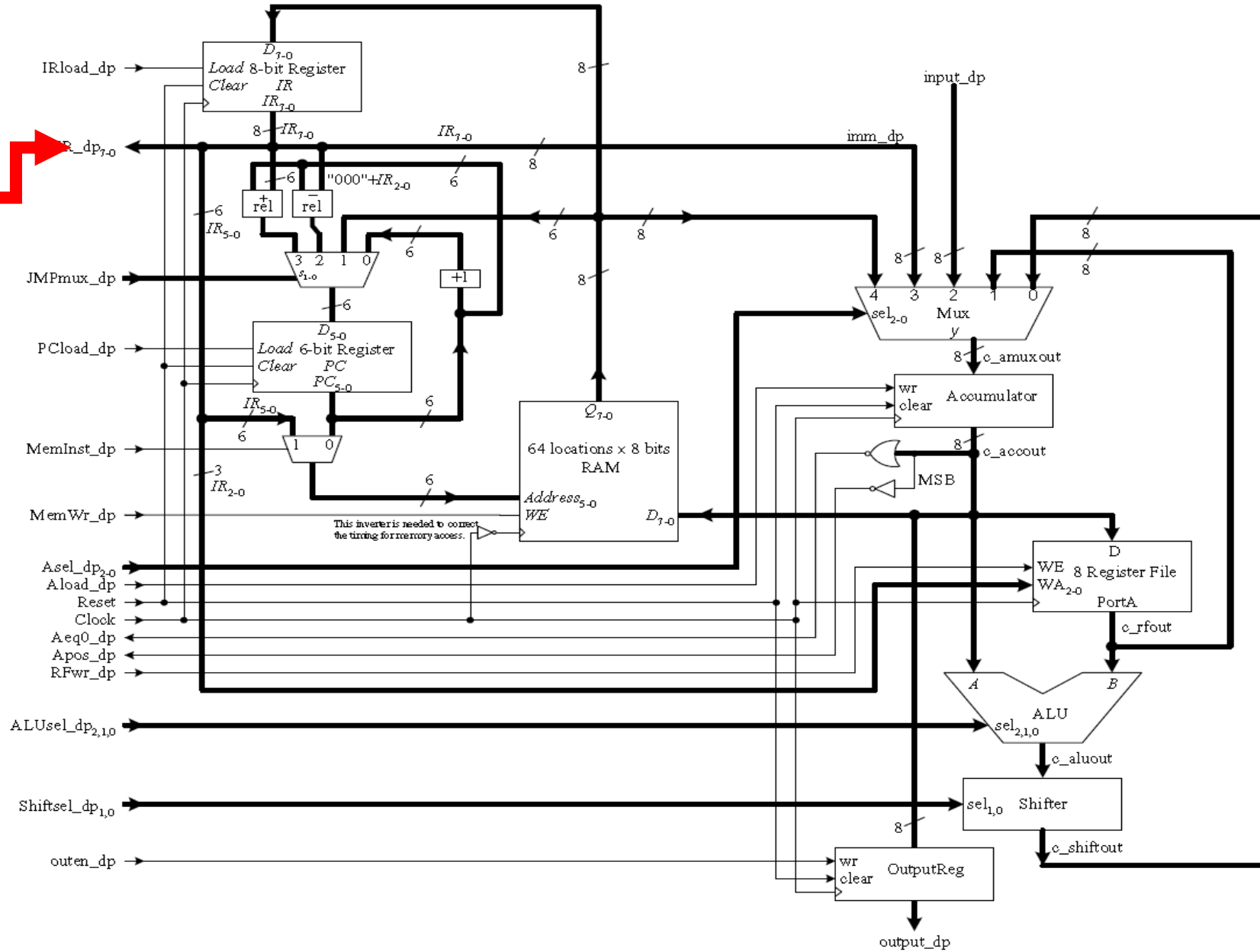
Input / Output and Miscellaneous

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

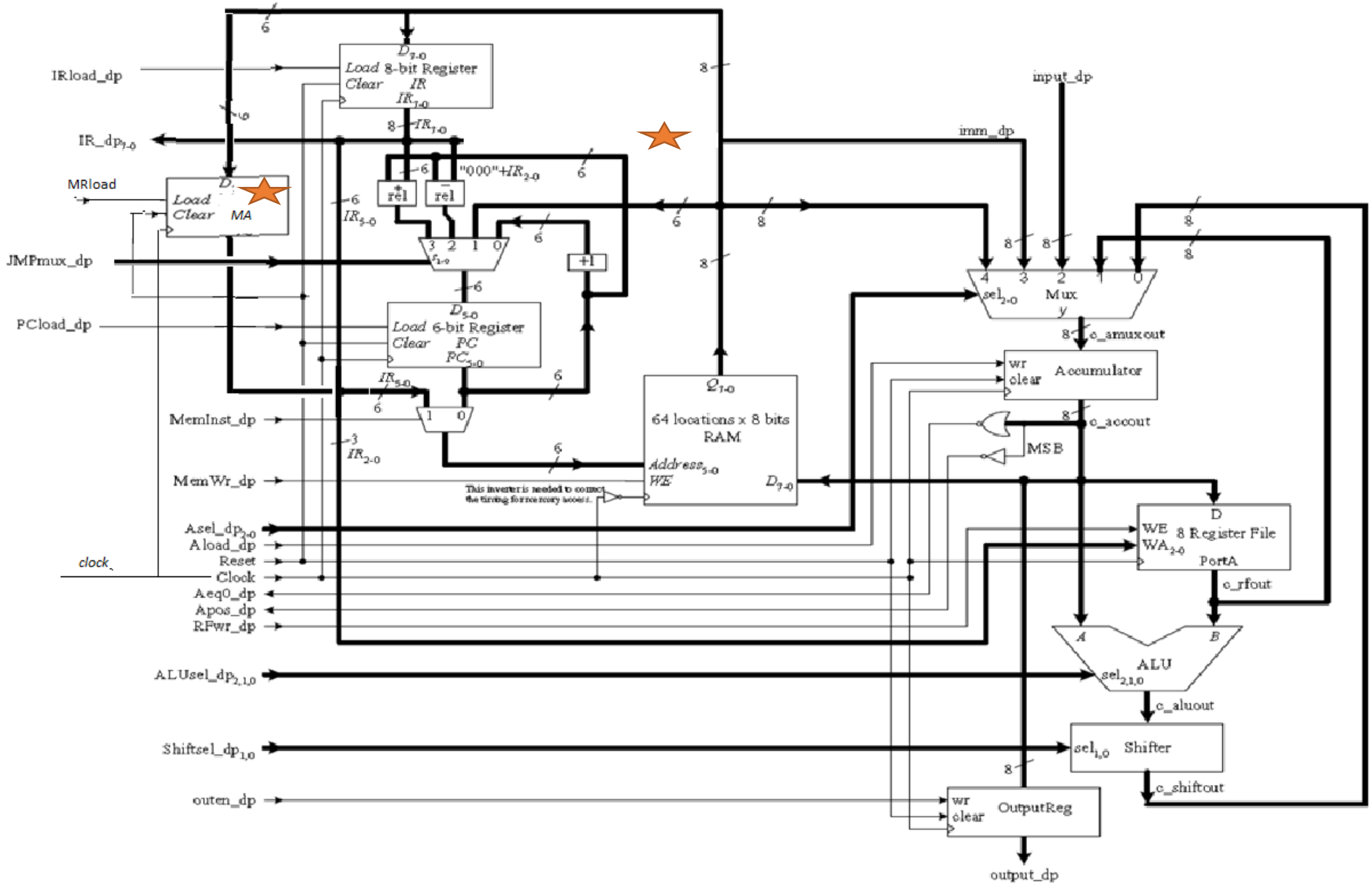
Suggested as a solution in the textbook (Dig. Logic and MP design, E. Hwang)

➢Problems related to efficiency:

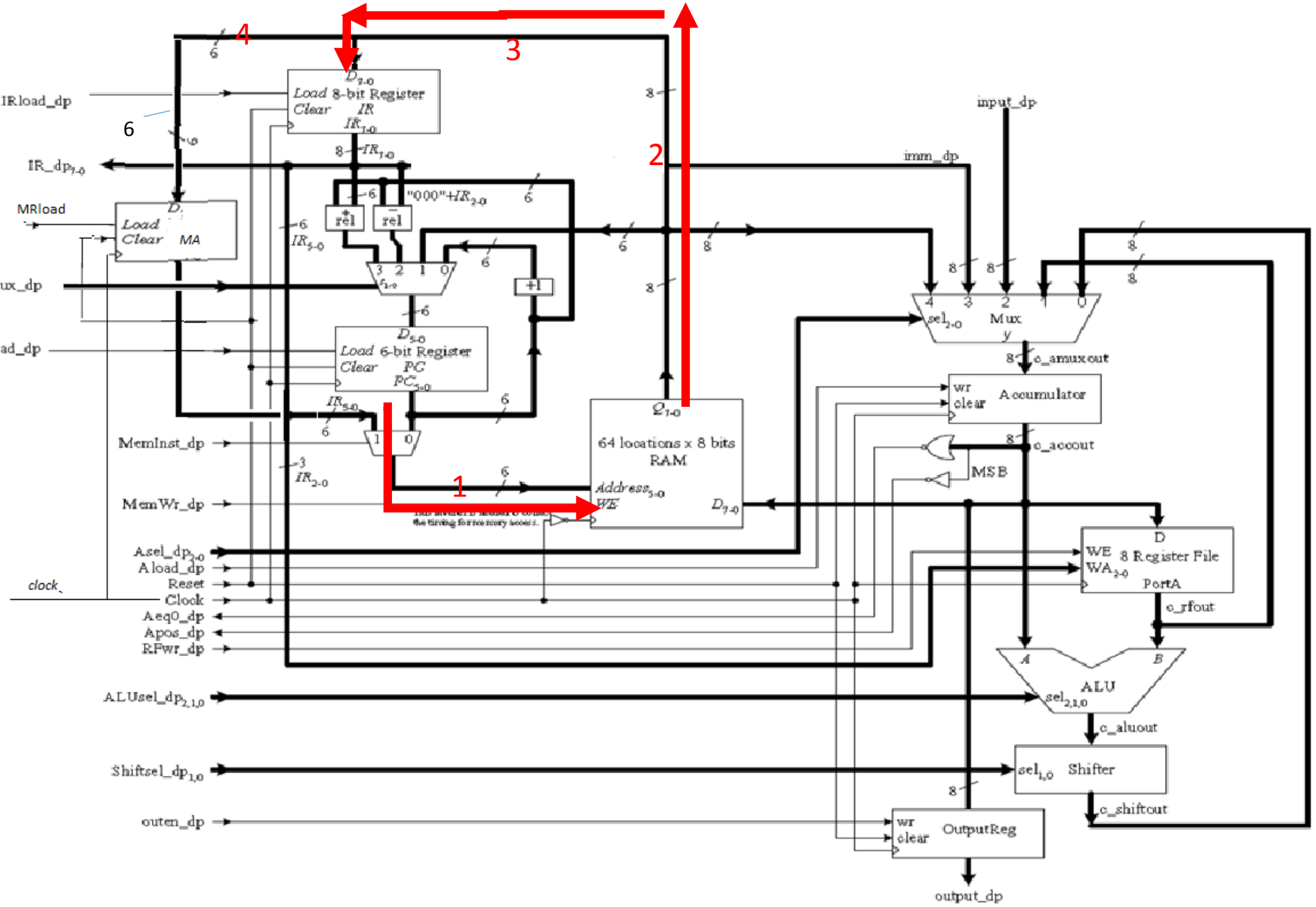
- Opcode is available to CU at the beginning of the second cycle (after fetch), and thus only at the third cycle CU can assert signals to the DP. A void cycle is included.
- The immediate operand is also not available until 3rd cycle.
- Memory address will overwrite the instruction and the opcode will not be available



Modified DP


IRload_dp
IR_dp7-0
MRload
JMPmux_dp
PCload_dp
MemInst_dp
MemWr_dp
Asel_dp2-0
Aload_dp
Reset
clock
Clock
Aeq0_dp
Apos_dp
RFwr_dp
ALUsel_dp2,1,0
Shiftsel_dp1,0
outen_dp
Load 8-bit Register
Clear IR
Load 6-bit Register
Clear PC
Load
Clear MA
rel
rel
"000"+IR2-0
64 locations x 8 bits RAM
Address5-0
WE
This inverter is needed to correct the timing for memory access.
input_dp
imm_dp
Mux
sel2-0
c_amuxout
wr
clear
Accumulator
c_accout
MSB
D
WE 8 Register File
WA2-0
PortA
c_rfout
A
B
ALU
sel2,1,0
c_aluout
sel1,0 Shifter
c_shiftout
wr
clear
OutputReg
output_dp

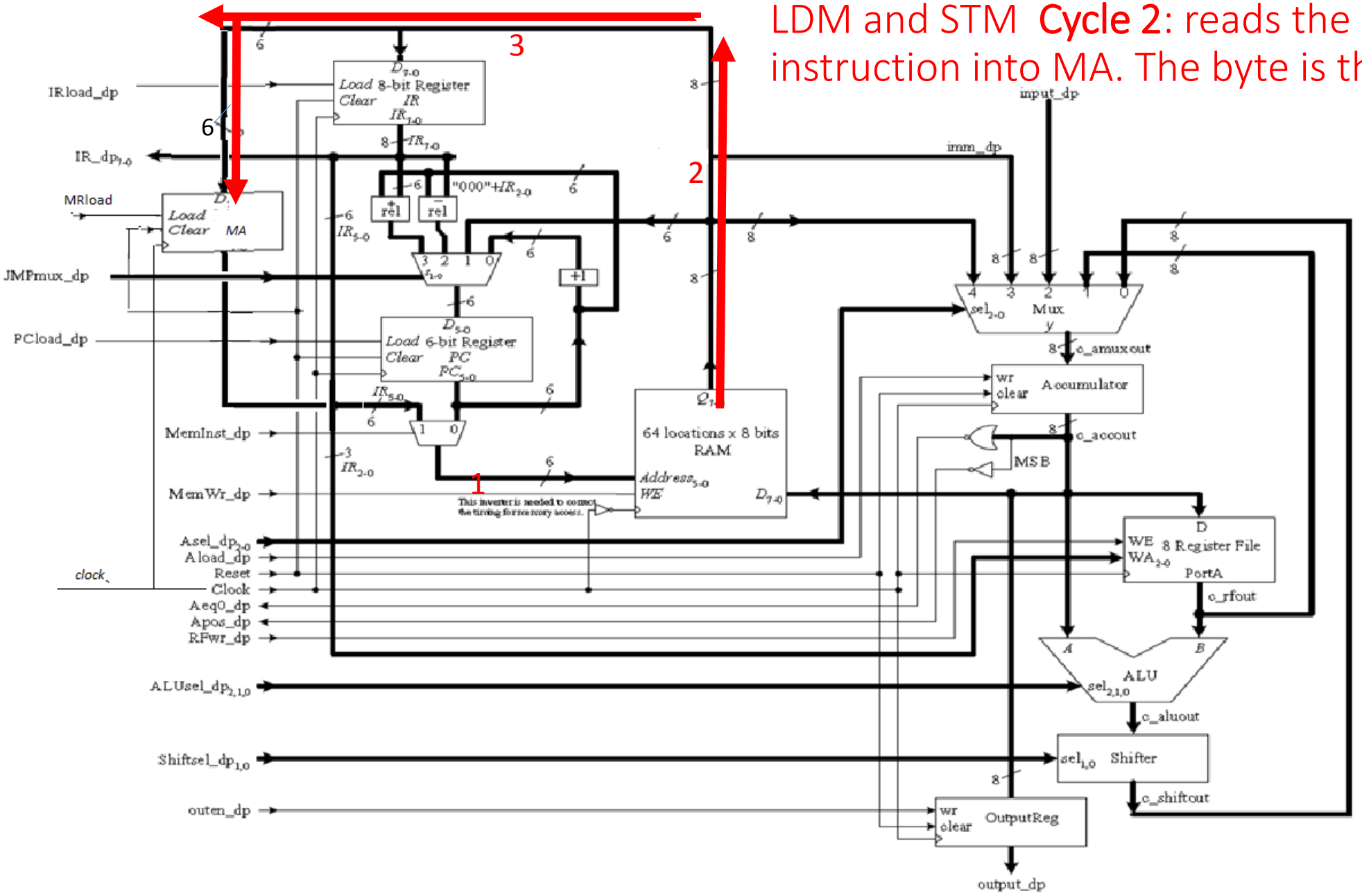
Fetch in Cycle 0



Fetch in Cycle 0 and Decode in Cycle 1

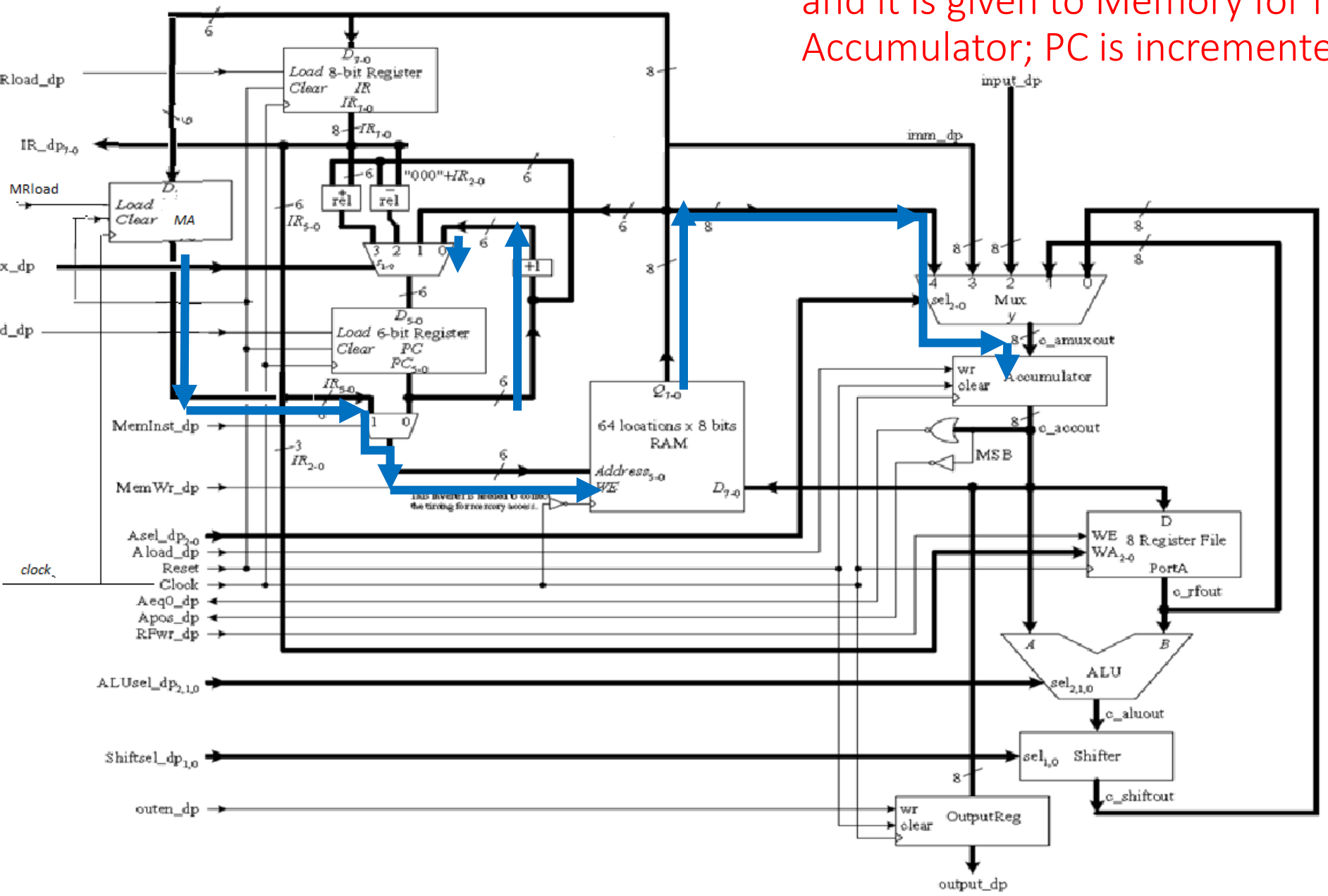
- IR loaded in Cycle 0 with the first byte (could be the only byte) of Instruction.
- It is provided to CU in the beginning of the Cycle 1
- PC is incremented and loaded
- Cycle 1 is empty ; allowing CU to decode

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

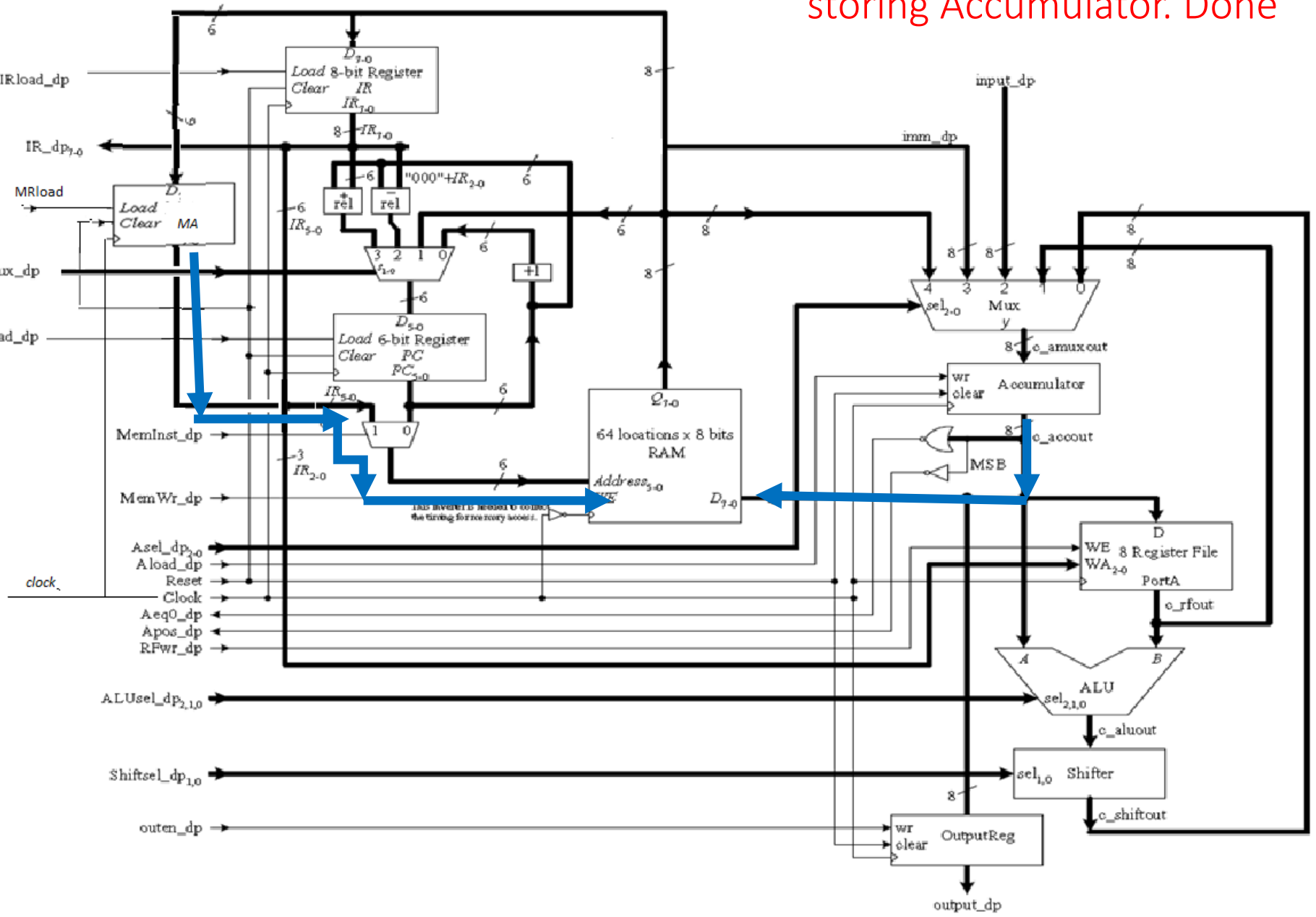


LDM A,aaaaaa	0011 0000 00aaaaaa	A ← M[aaaaaa]

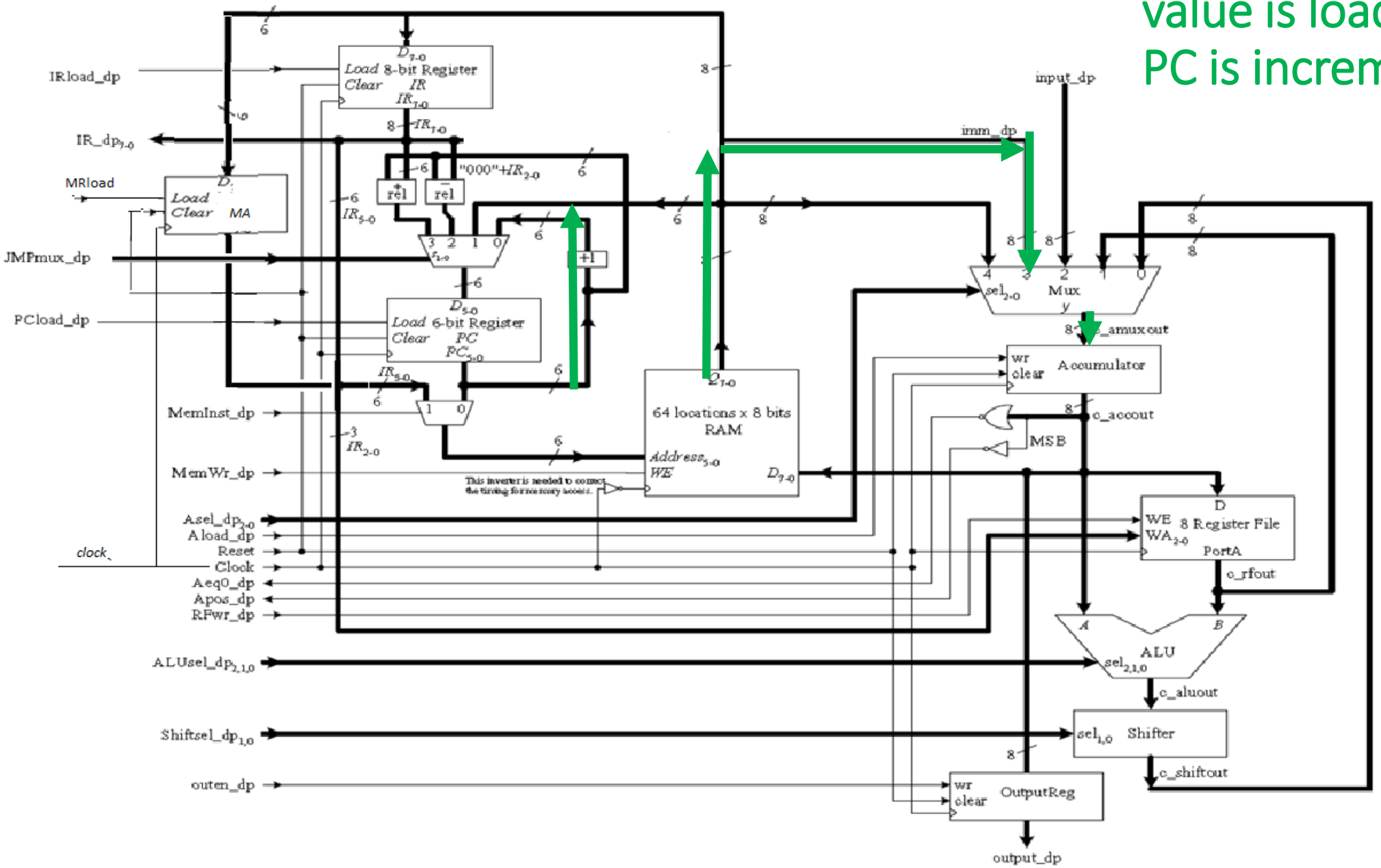
LDM Cycle 3: reads the IR. The byte is the memory addres
and it is given to Memory for reading data into
Accumulator; PC is incremented.



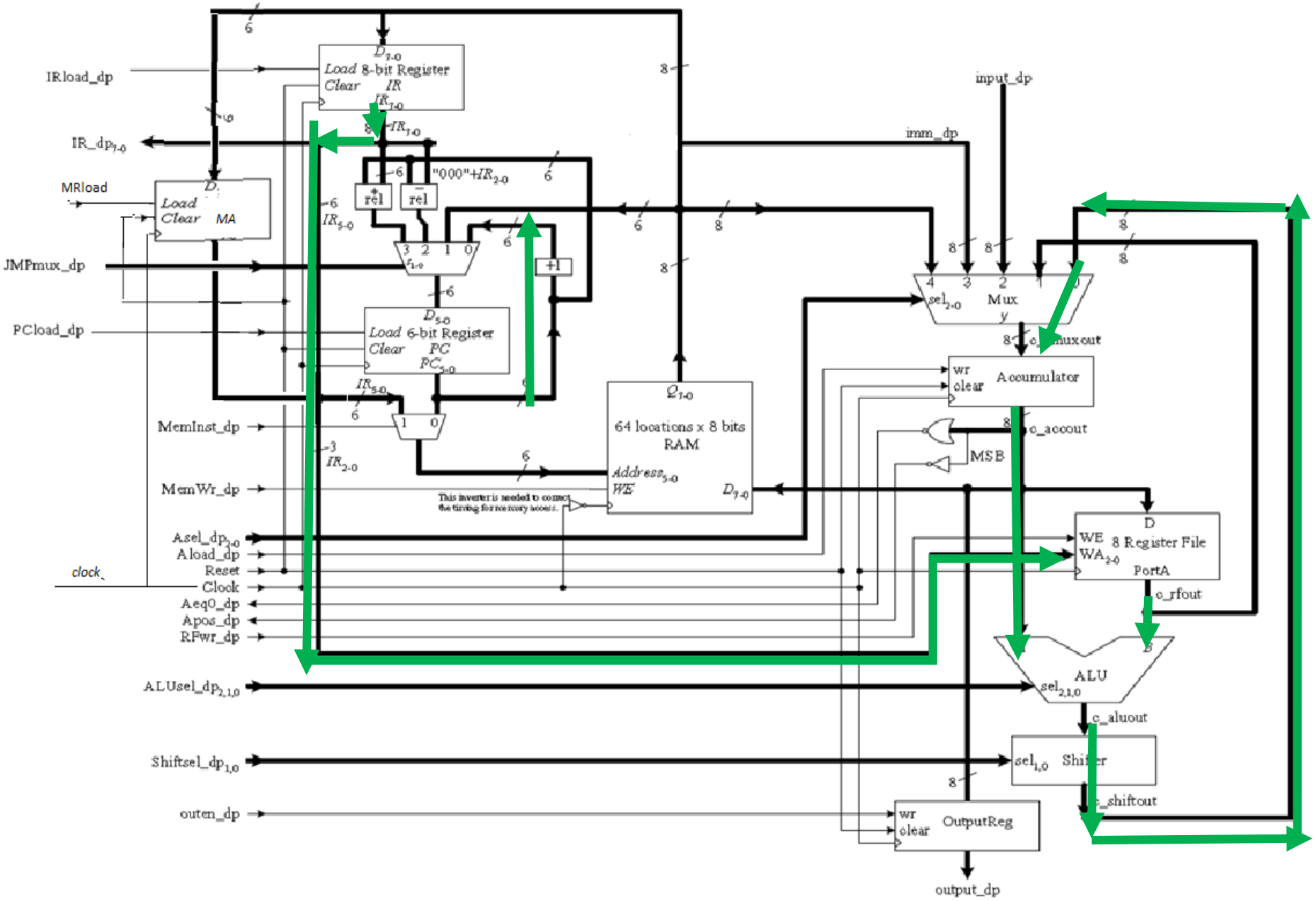
STM Cycle 3: reads Memory address is given to Memory f
storing Accumulator. Done



LD, cycle 2: Immediate value is loaded:
PC is incremented



Register ALU op
Cycle 2: perform the operation; increment PC

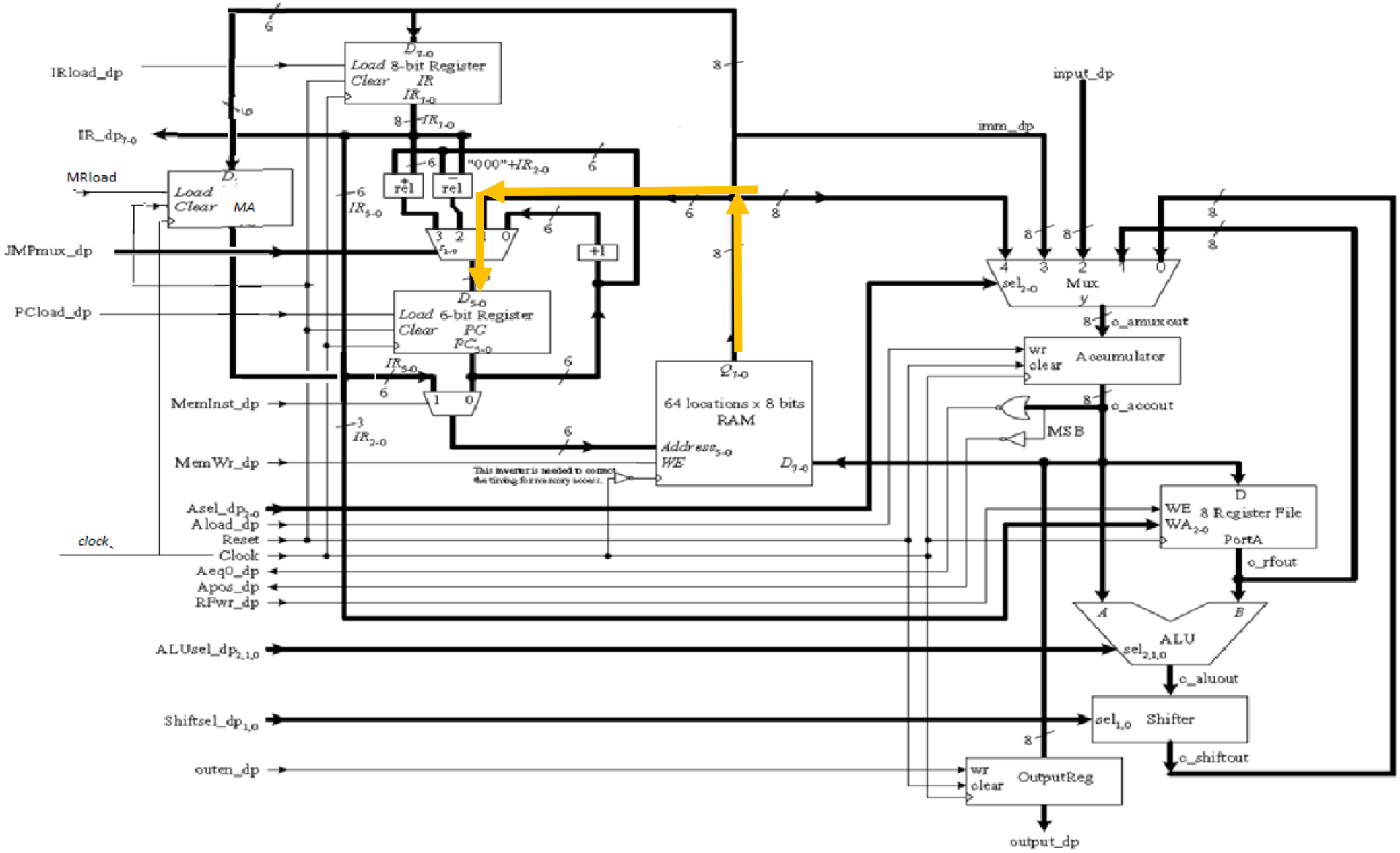


JZ
Cycle 2

- Read 6 bits of

the 2 nd byte to PC

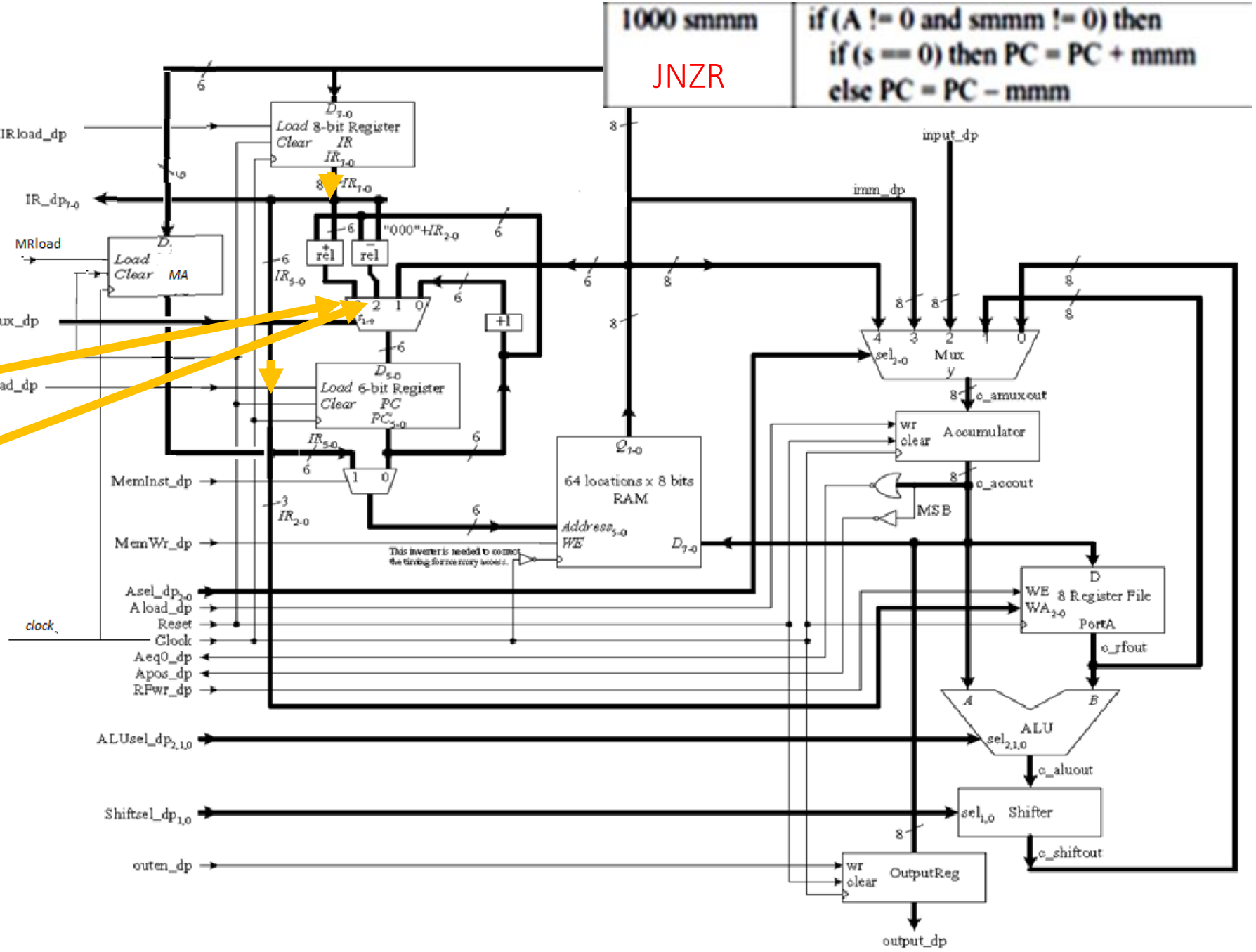
0111 0000 00 aaaaaa	if (A == 0) then PC = aaaaaa



JNZR Cycle 2

- If IR_dp<3..0> != 0

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



FSM

- Some instructions complete in 2 cycles , such as NOP and Halt
- Some in 3 cycles: jumps and arithmetic, in, out
- LDM and STM complete in 4 cycles