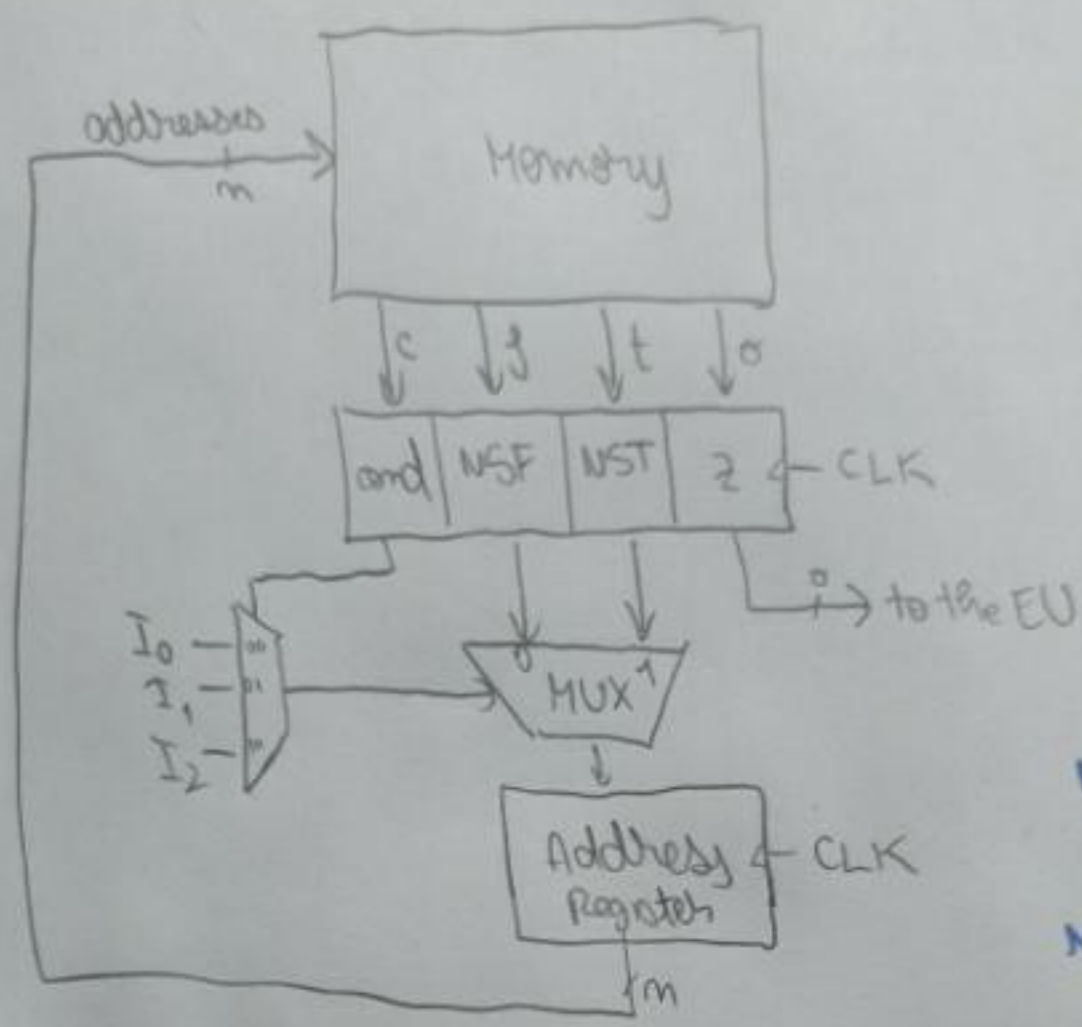
5. ~~bla bla~~

a) The implementation is done with state addressing with pair addresses, because the outputs only depend on the current state, so this is a Moore machine.



All inputs have 0 on the left and 1 on the right

NSF = next state if false

NST = next state if true

We have: 3 inputs so condition is on 2 bits, C=2

5 states so we encode them on 3 bits

$\Rightarrow n = j = t = 3$

3 outputs ~~state encode~~ $\Rightarrow$ 3 bits, $\theta = 3$

b) The necessary memory capacity is

$2^n (c + j + t + o) = 2^3 (3+3+2+3) = 2^3 \cdot 11$ ~~$2^3(3+3+2+3) = 2^4 \cdot 13$~~

a) We continue the implementation

States encoding: Standby: ~~0000~~ 000

Run: 001

Suspend: 010

Resume: 011

Wait: 100

Cond	
00	I_0
01	I_1
10	I_2
11	X

Memory Map:

Address	cond	NSFalse	NSTrue	outputs $z_1 z_2 z_3$
Standby 000	00	000	001	1 1 0
Run 001	01	010	001	0 1 1
Suspend 010	xx	011	011	1 0 0
Resume 011	01	001	100	0 0 0
Wait 100	10	100	000	1 1 0
101	xx	xxx	xxx	xxx
110	xx	xxx	xxx	xxx
111	xx	xxx	xxx	xxx

c) Standby: A, Run: B, Susp.: C, Res.: D, Wait: E

States \ I	0	1
A	A/110	B/110
B	C/011	B/011
C	D/100	D/100
D	B/000	E/000
E	E/110	A/110

Each state only has 1 input, Do we consider that one of the columns for simpler representation

d) The only possible ~~compatible~~ equivalent states are A and E because they have the same output, that would mean A has to be equivalent with E and A with B, which isn't possible

So the automaton's nr of states can't be reduced;