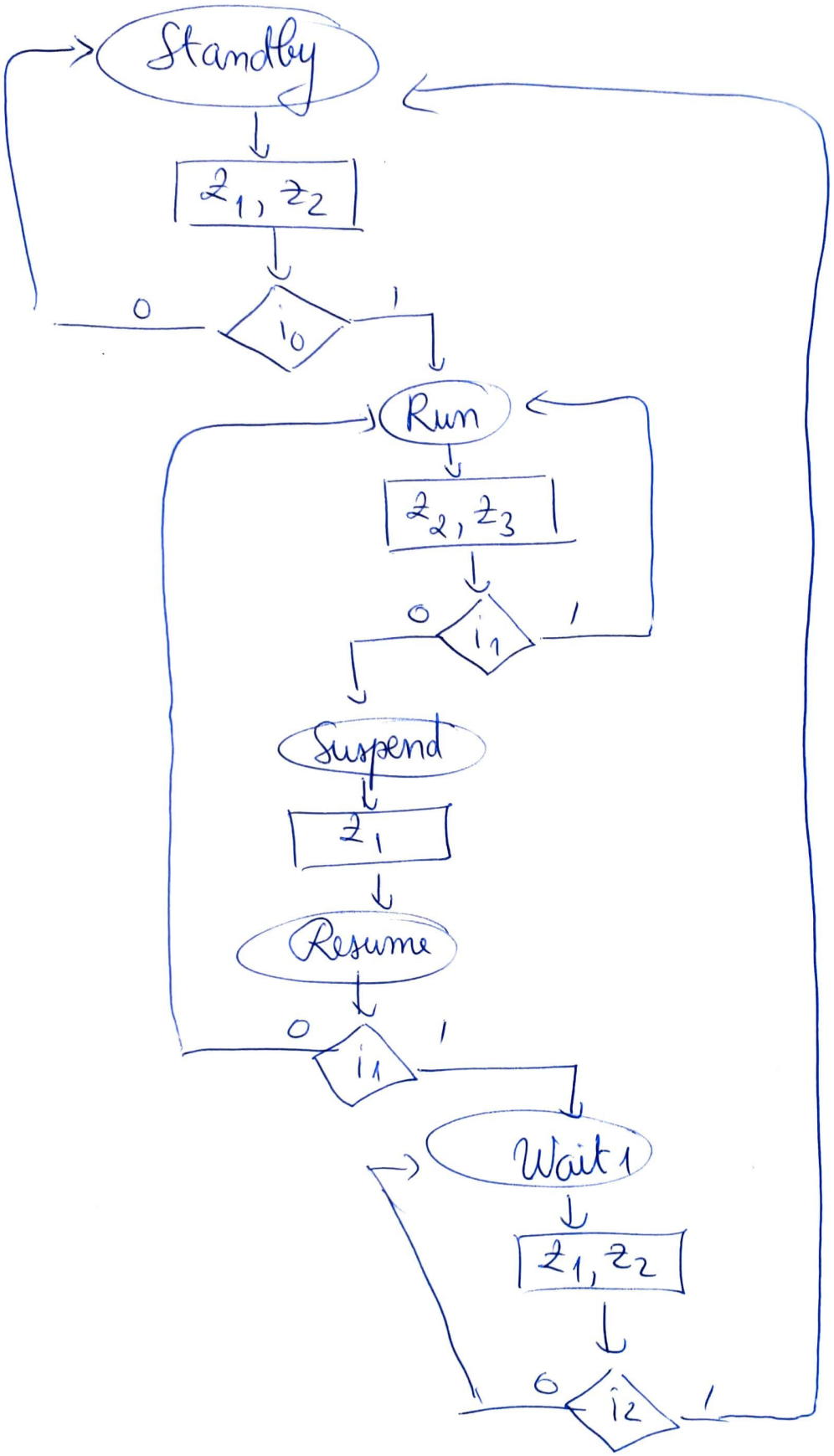
# Problem 5


Standby
$z_1, z_2$
$i_0$
0
1
Run
$z_2, z_3$
$i_1$
0
1
Suspend
$z_1$
Resume
$i_1$
0
1
Wait 1
$z_1, z_2$
$i_2$
0
1

## Problem 5

a) Memories:

I will implement the automaton, using the state adressing method, more exactly with pair adresses. We have 5 states, therefore I will encode them on 3 bits:

Standby: 000
Run: 001
Suspend: 010
Resume: 011
Wait: 100

b) Capacity
c) Trans table
d) reduction

Therefore the memory content is:

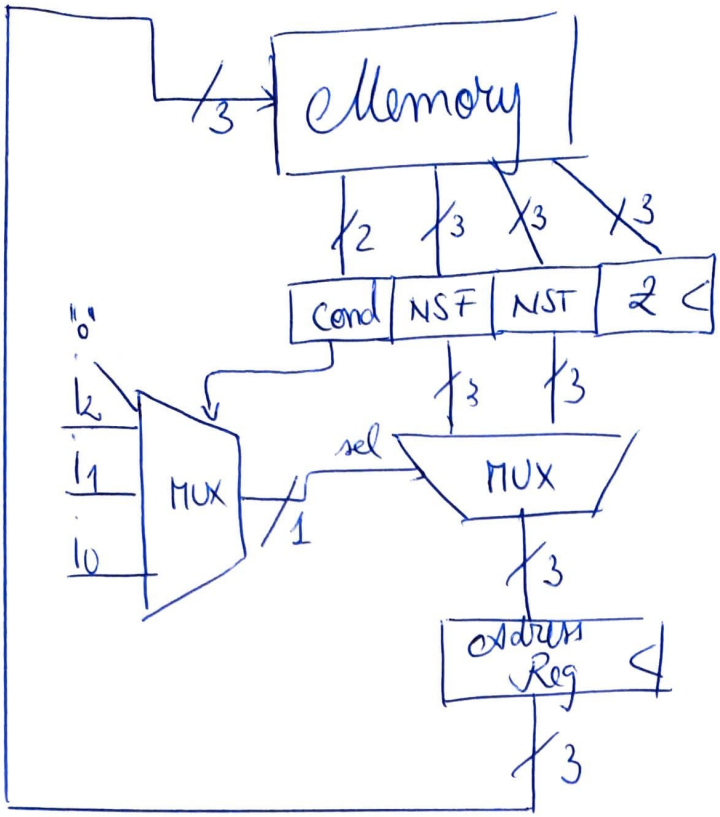
| Current State (adress) | Condition ($i_2/i_1/i_0$) | Next State False | Next State True | Output ($z_1, z_2, z_3$) (content) |
|---|---|---|---|---|
| 0h | 00 | 0h | 1h | 110 |
| 1h | 01 | 2h | 1h | 011 |
| 2h | xx | 3h | 3h | 100 |
| 3h | 01 | 1h | 4h | 000 |
| 4h | 10 | 4h | 0h | 110 |
| 5h | xx | x | x | 000 |
| 6h | xx | x | x | 000 |
| 7h | xx | x | x | 000 |

I choose this method because our automaton is a Moore Machine (outputs only depend on current state) and that meant that I didn't need to change anything. Also, by choosing the pair adress method, I can encode the States arbitrarily.

Encoding of conditions is:

$i_0 - 00$
$i_1 - 01$
$i_2 = 10$

The schematic is:




b) The capacity of our memory ~~is~~ follows the formula:

$$C = 2^u \cdot (f + t + c + m)$$

where in our case:

$u$ = bits for states = 3

$f = t$ = bits for states = 3

$c$ = cond bits (for $i_2, i_1, i_0$) = 2

$m$ = output bits = 3

$\Rightarrow$ the capacity is:

$C = 2^3 (3+3+3+2) =$

$= 8 \cdot (9+2) =$

$= 8 \cdot 11 = 88$ ~~locati~~ data locations'

c) Transitions table:

| State \ Inp | ~~$i_2$~~ ~~$i_1$~~ ~~$i_0$~~ | $z$ |
|---|---|---|
| | 000 001 010 011 110 111 101 100 | $\rightarrow i_2, i_1, i_0$ |
| 0h | | 110 |
| 1h | | 011 |
| 2h | | 100 |
| 3h | | 000 |
| 4h | | 110 |