Problemas Tema 3: Unidad de Control

①

C_3C_2	F
0 0	R+S
0 1	R-S
1 0	R AND S
1 1	R XOR S

C_1C_0	Entrada R	Entrada S
0 0	A	A
0 1	A	B
1 0	B	A
1 1	B	B

C_4	Acción
0	B ← F
1	A ← F

a)

① Hacemos A xor A que devuelve 0 y lo mete en el registro A

C_4: 1 C_3: 1 C_2: 1 C_1: 0 C_0: 0

② igual pero en B

C_4: 1 C_3: 1 C_2: 1 C_1: 1 C_0: 1

b) ① A = A + B C_4: 1 C_3: 0 C_2: 0 C_1: 0 C_0: 1

② B = A - B C_4: 0 C_3: 0 C_2: 1 C_1: 0 C_0: 1

③ A = A - B C_4: 1 C_3: 0 C_2: 1 C_1: 0 C_0: 1

⑬ b) Codificación de las instrucciones

IR [CODOP | d]

d ≡ Modo directo (1) / indirecto (0)

f ≡ Función ALU = 0 dejar pasar entrada reg

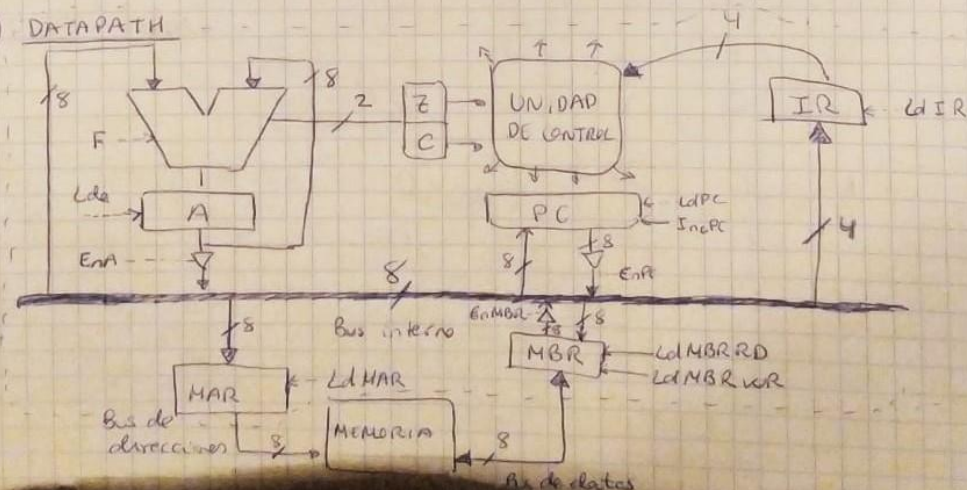
= 1 restar A - entrada

c ≡ Condición salto = 0 saltar según Z (Indirecto)

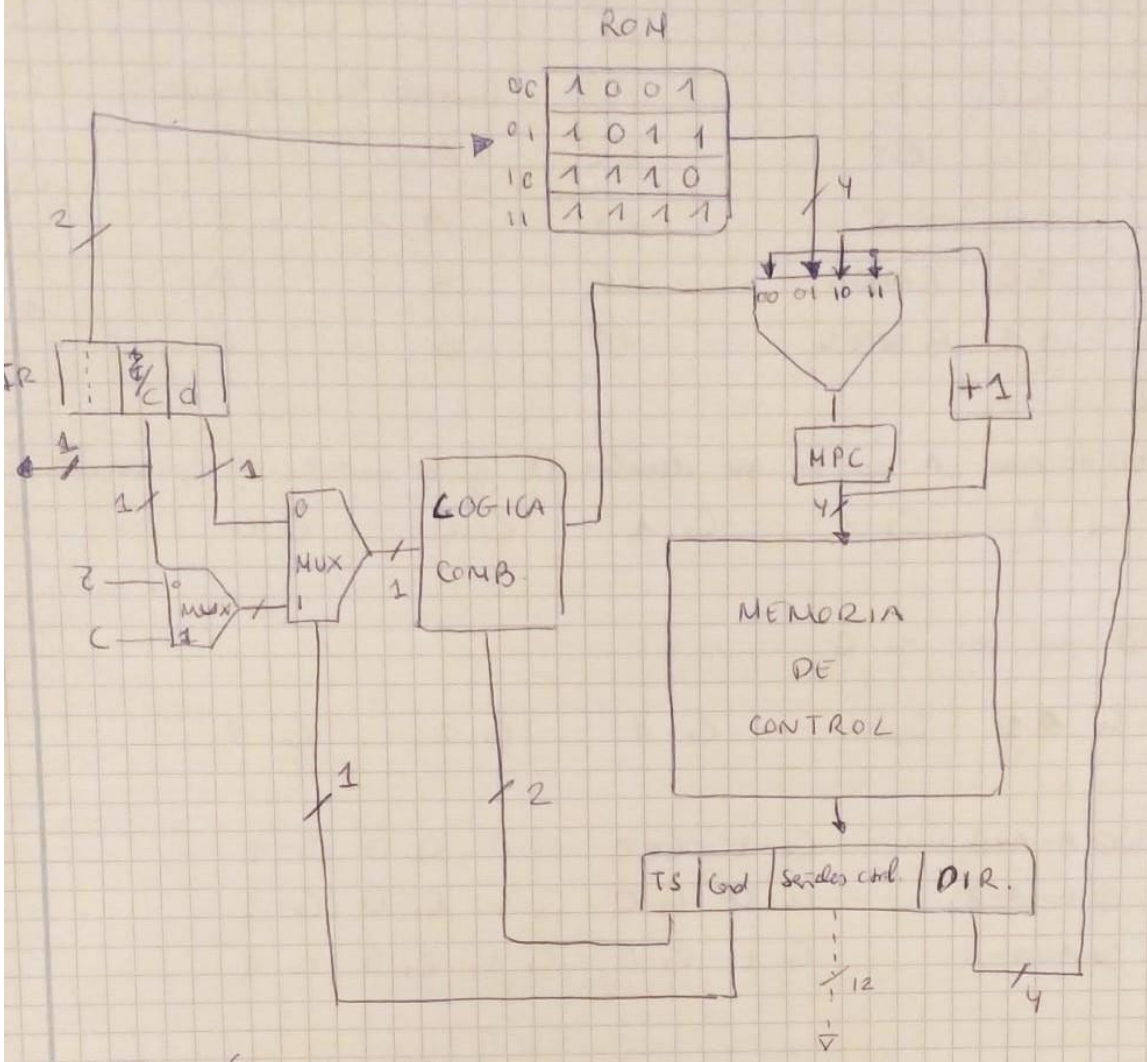
= 1 saltar según C

LOAD	0 0 F d
SUB	0 0 F d
STORE	0 1 0 d
JZ	1 0 c d
JC	1 0 c d
JMP	1 1 0 d

c) DATAPATH



d) UNIDAD DE CONTROL



TS (Tipo de Salto)

00: $\mu PC \leftarrow \mu PC + 1$

01: $\mu PC \leftarrow f(IR[3:2])$

10: $\mu PC \leftarrow$ dirección

11. Si COND = 0 then $\mu PC \leftarrow DIR$

else $\mu PC \leftarrow \mu PC + 1$

MUX

00 $\mu PC \leftarrow \mu PC + 1$

01 $\mu PC \leftarrow f(IR[3:2])$

10 $\mu PC \leftarrow$ dir

11 $\mu PC \leftarrow \mu PC + 1$

Logica combinacional

TS1	TS0	COND	MUX1	MUX0
0	0	X	0	0
0	1	X	0	1
1	0	X	1	0
1	1	0	1	0
1	1	1	1	1

$MUX1 = TS1$

$MUX0 = \overline{TS1}\, TS0 + TS0\, COND$

e) Contenido (en alto nivel) de la memoria de control

Captación / Decodificación

Fetch 0 MAR ← PC,
1 MBR ← M[MAR]; PC ← PC + 1,
2 IR[3..0] ← MBR[7..4]
3 MAR ← PC,
4 MBR ← M[MAR], PC ← PC + 1;
5 MAR ← MBR;
6 if (IR[0] = 0) then goto Indirecto,
7 goto f(IR[3..2]),
Indirecto: 8 MBR ← M[MAR]
9 MAR ← MBR, goto f(IR[3..2])

← En MAR / MBR está la dirección

EJECUCION

LOAD / SUB
10 MBR ← M[MAR],
11 A ← MBR o bien A ← A − MBR según IR[1], goto Fe

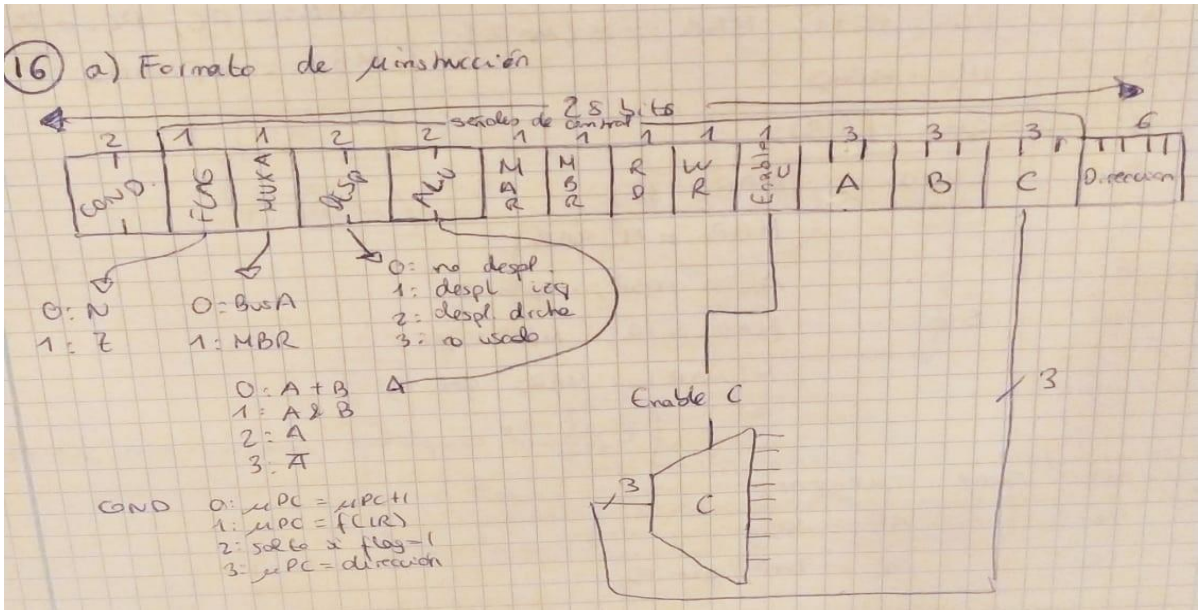
STORE
12 MBR ← A;
13 M[MAR] ← MBR, goto Fetch;

JZ / JC
14 IF (Indicador = 0) then goto Fetch
↑ dado por IR[1]

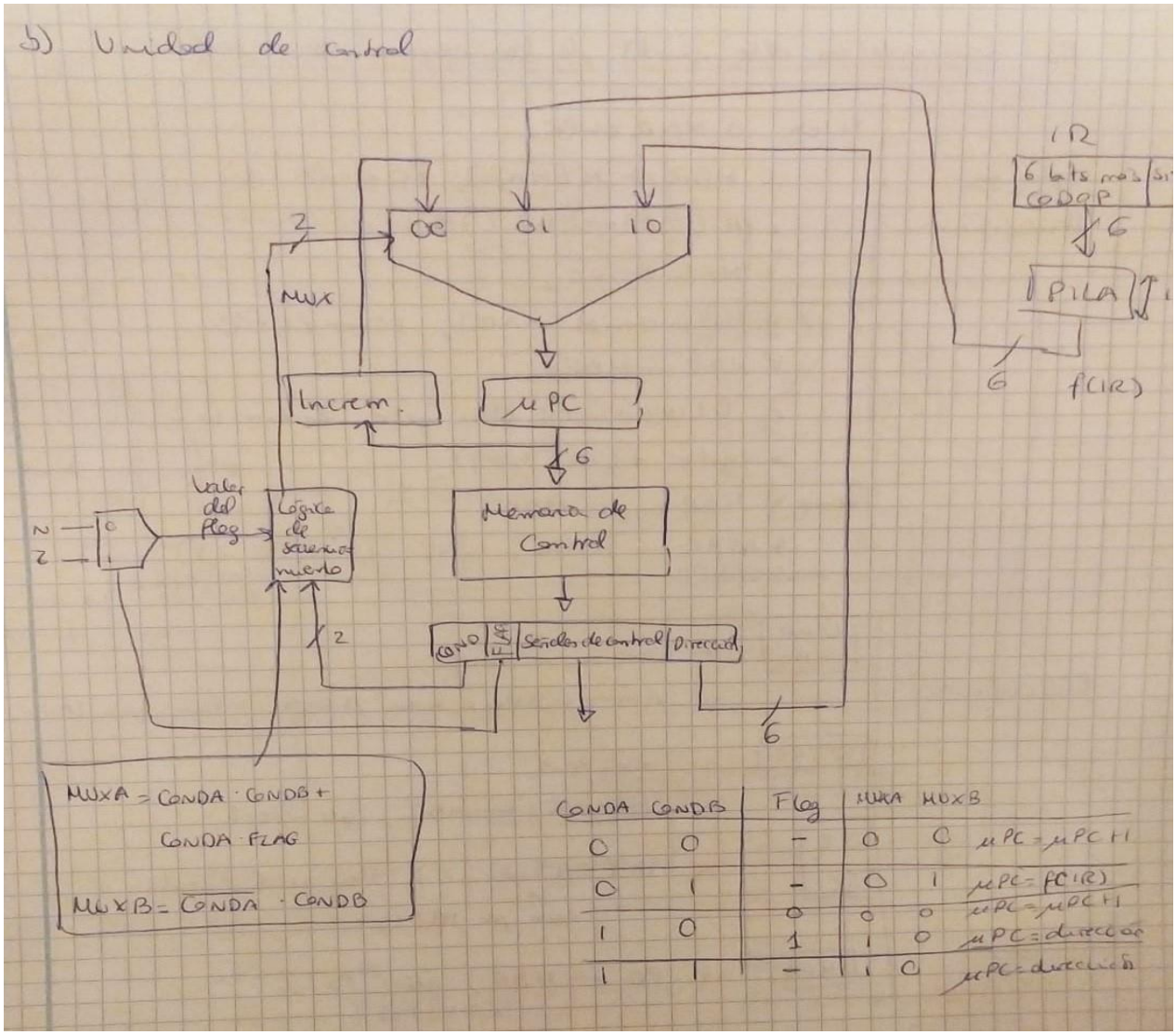
JMP
15 PC ← MBR, goto Fetch

$\lceil \log_2 16 \rceil = 4$

16) a) Formato de μinstrucción



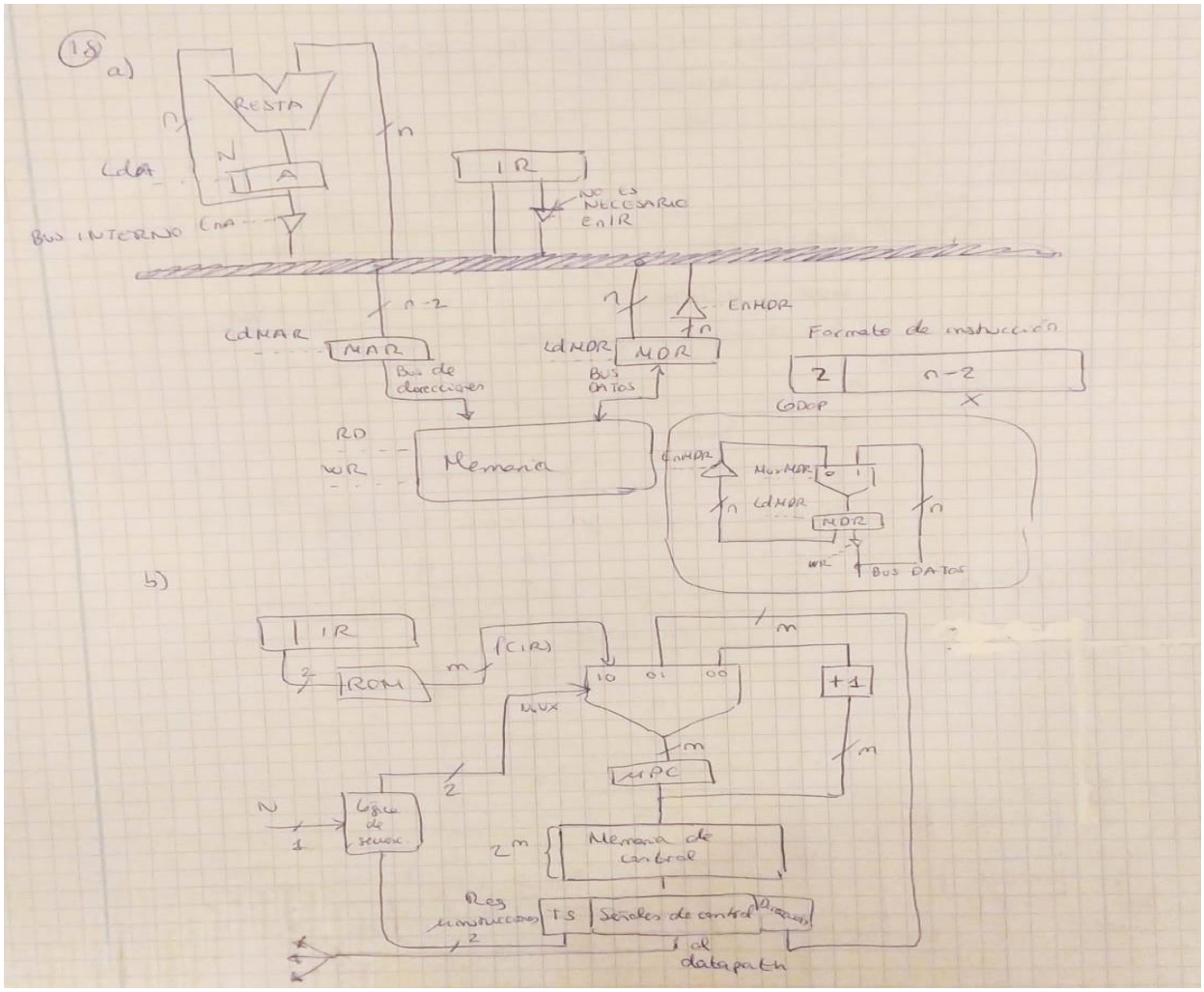
b) Unidad de control



CONDA	CONDB	Flag	MUXA	MUXB	
0	0	–	0	0	μPC = μPC+1
0	1	–	0	1	μPC = f(IR)
1	0	0	0	0	μPC = μPC+1
		1	1	0	μPC = dirección
1	1	–	1	0	μPC = dirección

) Microprograma

```
0:    MAR := PC;                                  | MAR := PC; PC := PC+1;
1:    PC := PC+1; MBR := M[MAR];                  | MBR := M[MAR];
2:    IR := MBR;
3:    goto f(IR);
4:    LODD.   MAR := IR;
5:            MBR := M[MAR];
6:            AC := MBR; goto 0;
7:    STOD.   MAR := IR; MBR := AC;
              M[MAR] := MBR; goto 0;
              ⋮ } Ejercicio: completar instrucciones que falt
47:   SNAP.   TMP := AC;
48:           AC := SP;
49:           SP := TMP; goto 0;
```



TS1	TS0	N	Significado	MUX1	MUX 0
0	0	–	μPC ← μPC + 1	0	0
0	1	–	μPC ← Dirección	0	1
1	0	–	μPC ← f(IR)	1	0
1	1	0	if N=0 goto Dirección	0	1
1	1	1	else μPC ← μPC+1	0	0

MICROPROGRAMA

Captación de instrucción

Fetch: MAR := PC

MDR := M[MAR], PC := PC + 1

IR := MDR

goto f(IR)

SUB X

MAR := {IR / MDR}

MDR := M[MAR]

A := A − MDR ; goto Fetch

JMPNEG X

if N=0 goto Fetch

PC := {IR / MDR} ; goto Fetch

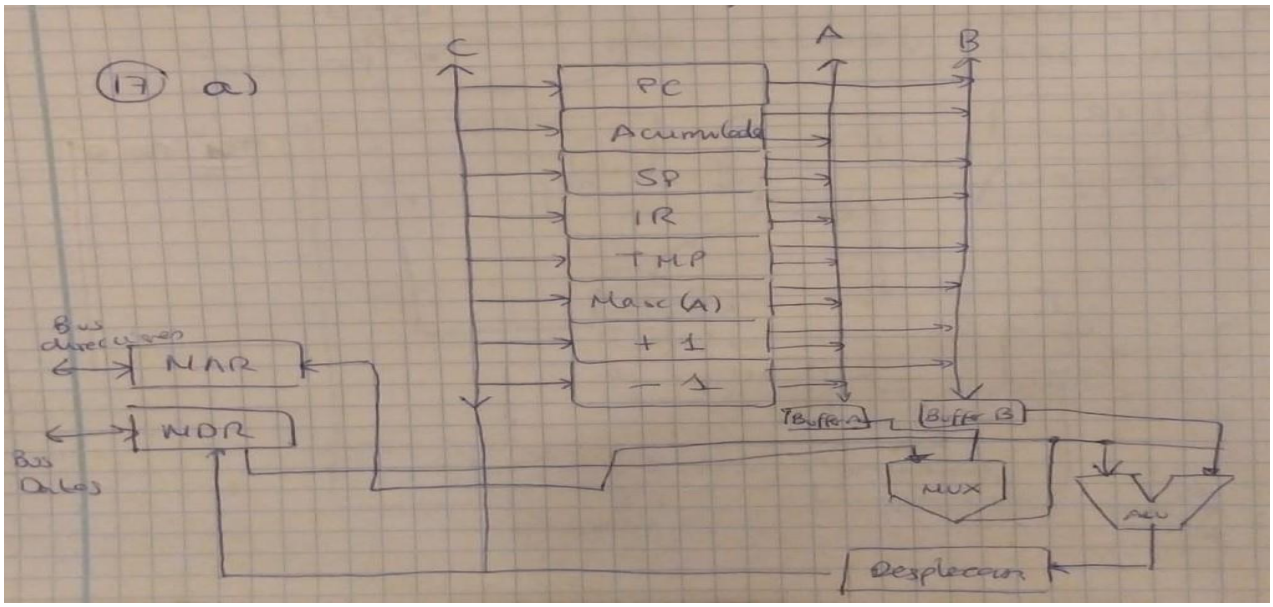
STORE X

MAR := {IR / MDR}

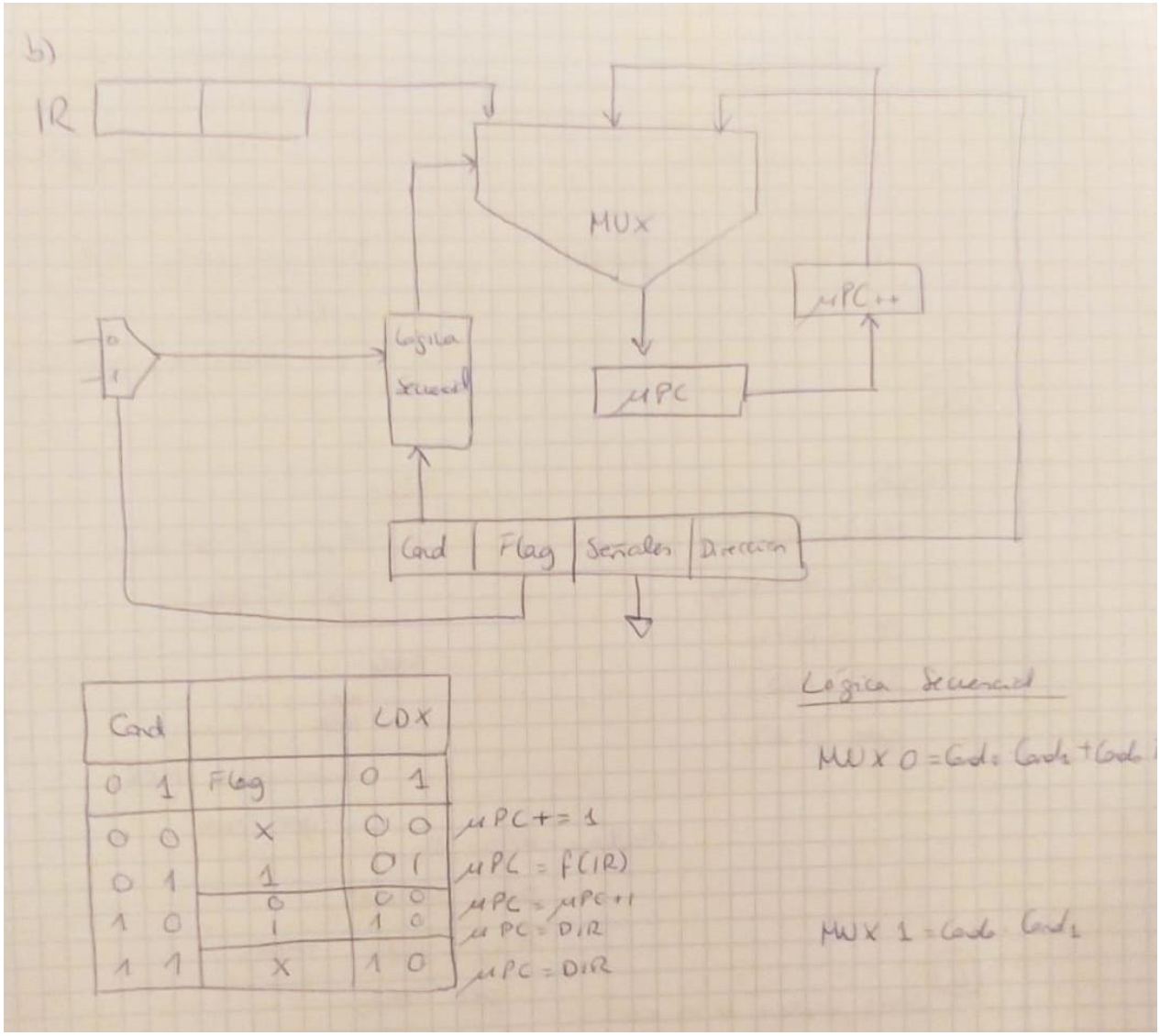
MDR := A

M[MAR] := MDR ; goto Fetch

17) a)



b)



Cod		LDX
0 1	Flag	0 1
0 0	X	0 0
0 1	1	0 1
1 0	0	0 0
	1	1 0
1 1	X	1 0

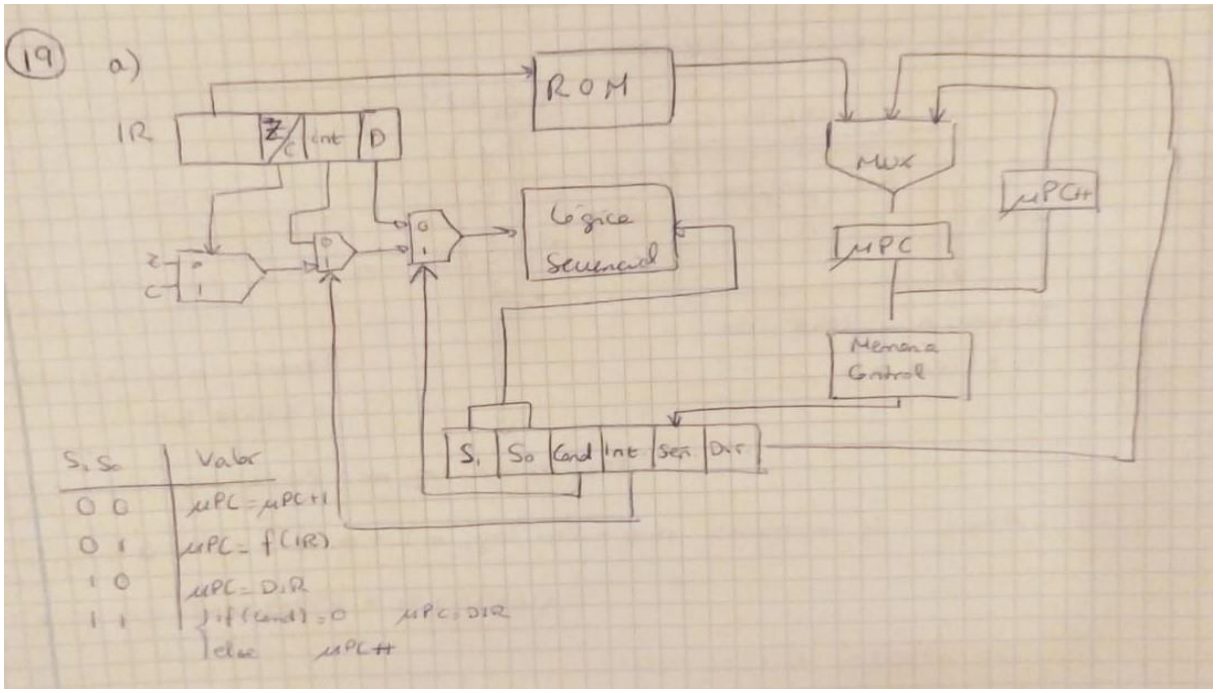
μPC+=1
μPC = F(IR)
μPC = μPC+1
μPC = DIR
μPC = DIR

Lógica Secuencial

MUX 0 = Cod$_0$ Cod$_1$ + Cod

MUX 1 = Cod$_0$ Cod$_1$

(19) a)



S_1 S_0	Valor
0 0	μPC = μPC+1
0 1	μPC = f(IR)
1 0	μPC = DIR
1 1	if (cond) = 0 μPC = DIR else μPC++

b)

Captación

MAR = PC;
MBR = M[MAR];
PC = PC+1;
IR = MBR; goto f(IR)

Interrupción

if (INT = 0)
MBR = PC, goto Interrupción

Direccionamiento Ind

if (I = 1)
MAR = IR, goto Captación;

(20)

Fetch

MAR := PC;
MBR := M[MAR];
PC := PC+1;
IR := MBR;
goto f(IR);

Addr

$R_{dst} := R_{dst} + R_{src}$;
goto Fetch;

Addx

MAR := IR;
MBR := M[MAR];
$R_{dst} := R_{dst} + MBR$;
goto Fetch;

Addx MAR = IR;
MBR = M[MAR];
$R_{dst} := R_{dst} + MBR$;
goto Fetch;

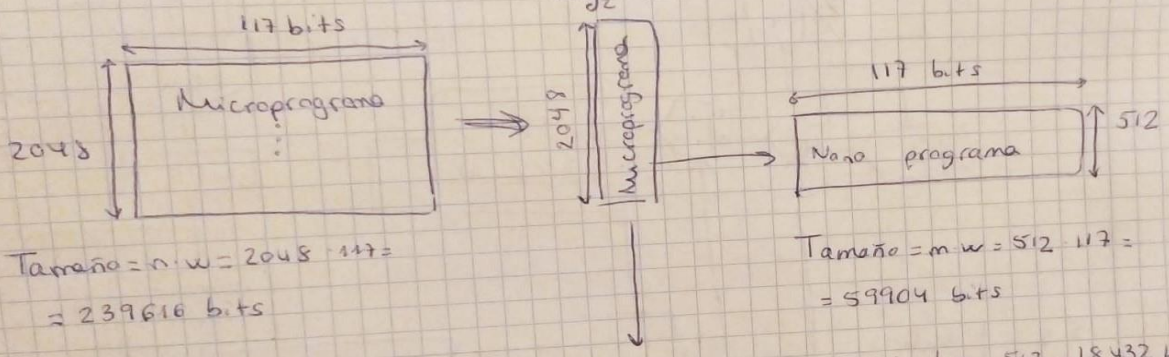
(6)

(6)

goto Fetch;

2048 palabras de 117 bits

512 instrucciones diferentes



Tamaño = n · w = 2048 · 117 =

= 239616 bits

Tamaño = m · w = 512 · 117 =

= 59904 bits

Tamaño = n $\log_2$ m = 2048 $\log_2$ 512 = 18432 bits

Ahorro de memoria = n·w − (n · $\log_2$ m) + m·w = 239616 bits −

− (18432 bits + 59904 bits) = 161280 bits

Se ahorran 161280 bits

(15) Lenguaje máquina

SUB X ; A ← A − M[x]

STORE X ; M[x] ← A

JMPNEG x ; Saltar a X si ACO

1) A valor arbitrario
2) M >= 0
3) N >= 0
4) UNO = 1
5) Al finalizar R = resultado
6) M y N deben quedar inalterados

R = M * N

R = 0 Repetir M veces la instrucción

R = R + N