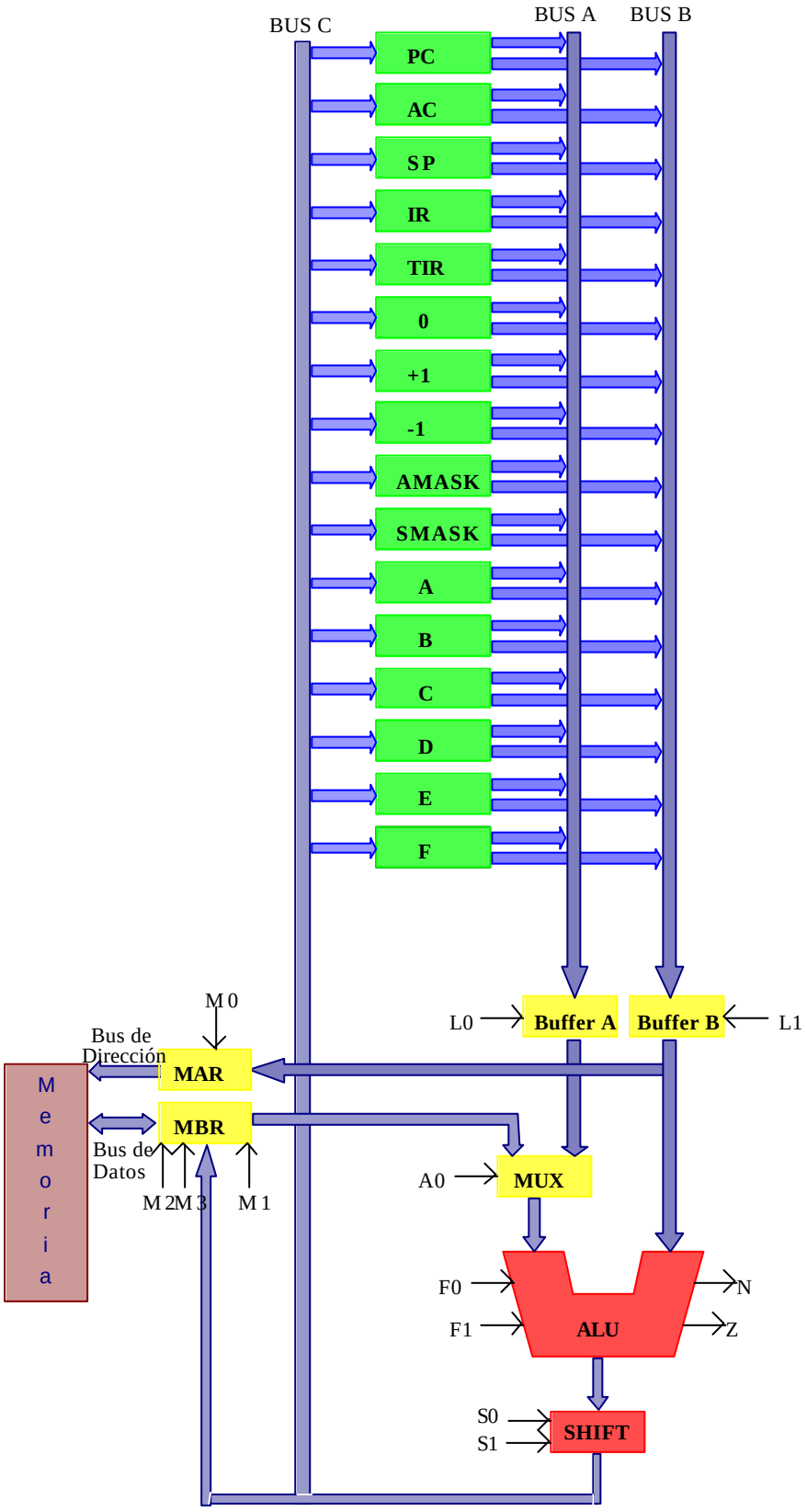
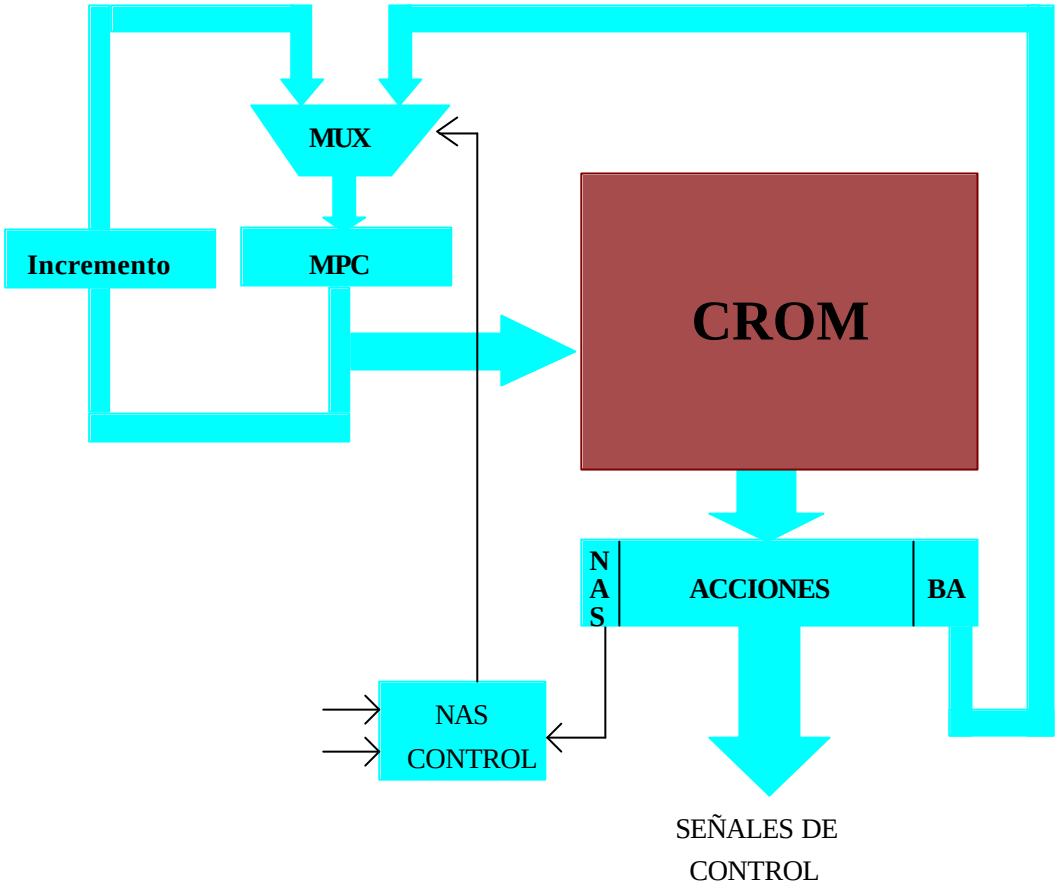
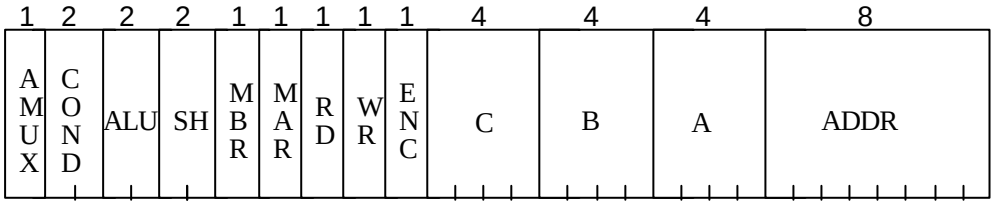
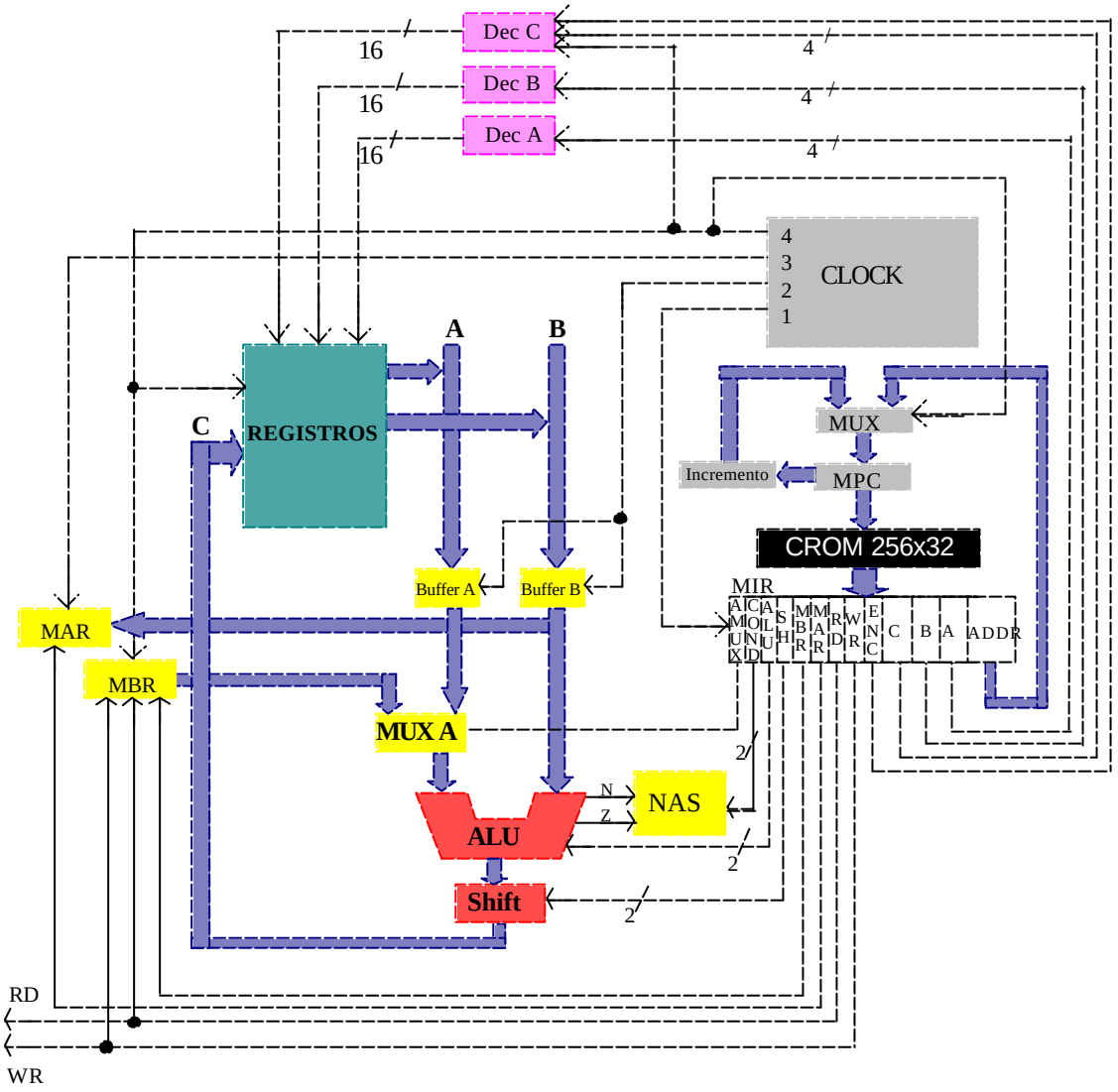

BUS C
BUS A
BUS B
PC
AC
SP
IR
TIR
0
+1
-1
AMASK
SMASK
A
B
C
D
E
F
M 0
Bus de Dirección
MAR
MBR
Bus de Datos
M 2
M 3
M 1
Memoria
L0
Buffer A
Buffer B
L1
A0
MUX
F0
F1
ALU
N
Z
S0
S1
SHIFT





AMUX	COND	ALU	SH
0=Buffer de A	0:No salta	0:A+B	0:No desplaza
1=MBR	1:Salta si N=1	1:A and B	1:Desplaza derecha
	2:Salta si Z=1	2:A	2:Desplaza izquierda
	3:Salta siempre	3: $\overline{A}$	3:No usado

1: Carga en el **MIR** la siguiente microinstrucción a ejecutar
2: Salida de los registros **A** y **B** y captura en los Buffers
3: Tiempo para la operación de la **ALU** y carga del **MAR**
4: Almacena **C** y **MBR**


Dec C
Dec B
Dec A
16
16
16
4
4
4
4
3
2
1
CLOCK
A
B
C
REGISTROS
MUX
Incremento
MPC
CROM 256x32
MIR
Buffer A
Buffer B
MAR
MBR
MUX A
ALU
NAS
N
Z
Shift
2
2
2
RD
WR

Conjunto de Instrucciones

Binario	**Nemo**	**Instrucción**	**Significado**
0000xxxxxxxxxxx	LODD	Carga directa	ac:=m[x]
0001xxxxxxxxxxx	STOD	Almacena directo	m[x]:=ac
0010xxxxxxxxxxx	ADDD	Suma directo	ac:=ac+m[x]
0011xxxxxxxxxxx	SUBD	Resta directo	ac:=ac-m[x]
0100xxxxxxxxxxx	JPOS	Salta si positivo	if ac≥0 then pc=x
0101xxxxxxxxxxx	JZER	Salta si cero	if ac=0 then pc=x
0110xxxxxxxxxxx	JUMP	Salta siempre	pc:=x
0111xxxxxxxxxxx	LOCO	Carga inmediata	ac:=x
1000xxxxxxxxxxx	LODL	Carga local	ac:=m[sp+x]
1001xxxxxxxxxxx	STOL	Almacena local	m[sp+x]:=x
1010xxxxxxxxxxx	ADDL	Suma local	ac:=ac+m[sp+x]
1011xxxxxxxxxxx	SUBL	Resta local	ac:=ac-m[sp+x]
1100xxxxxxxxxxx	JNEG	Salta si negativo	if ac<0 then pc:=x
1101xxxxxxxxxxx	JNZE	Salta si no cero	if ac≠0 then pc:=x
1110xxxxxxxxxxx	CALL	Llamado	sp:=sp-1; m[sp]:=pc; pc:=x
1111000000000000	PSHI	Apila indirecto	sp:=sp-1; m[sp]:=m[ac]
1111001000000000	POPI	Desapila indirec.	m[ac]:=m[sp]; sp:=sp+1
1111010000000000	PUSH	Apila	sp:=sp-1;m[sp]:=ac
1111011000000000	POP	Desapila	ac:=m[sp];sp:=sp+1
1111100000000000	RETN	Retorno	pc:=m[sp]; sp:=sp+1
1111101000000000	SWAP	Intercambiar	tmp:=ac;ac:=sp;sp:=tmp
11111100yyyyyyyy	INSP	Incrementa SP	sp:=sp+y
11111110yyyyyyyy	DESP	Decrementa SP	sp:=sp-y

MICROPROGRAMA

0: mar:=pc; rd;	ciclo principal , lectura de instrucción
1: pc:=pc+1; rd;	incrementa pc
2: ir:=mbr; if n then goto 28;	salva y decodifica mbr 0xxx ó 1xxx
3: tir:=lshift(ir+ir); if n then goto 19;	00xx ó 01xx
4: tir:=lshift(tir); if n then go to 11;	000x ó 001x
5: alu:=tir; if n then goto 9;	0000 ó 0001
6: mar:=ir; rd;	0000=LODD
7: rd;	
8: ac:=mbr; goto 0;	
9: mar:=ir; mbr:=ac; wr;	0001=STOD
10: wr; goto 0;	
11: alu:=tir; if n then goto 15;	0010 ó 0011
12: mar:=ir; rd;	0010=ADDD
13: rd;	
14: ac:=mbr+ac; goto 0;	
15: mar:=ir; rd;	0011=SUBD
16: ac:=ac+1; rd;	
17: a:= inv(mbr);	
18: ac:=ac+a; goto 0;	
19: tir:=lshift(tir); if n then goto 25;	010x ó 011x
20: alu:=tir; if n then goto 23;	0100 ó 0101
21: alu:=ac; if n then goto 0;	0100=JPOS
22: pc:=band(ir,amask); goto 0;	salta
23: alu:=ac; if z then goto 22;	0101=JZER
24: goto 0;	no salta
25: alu:=tir; if n then goto 27;	0110 ó 0111
26: pc:=band(ir,amask); goto 0;	0110=JUMP
27: ac:=band(ir,amask); goto 0;	0111=LOCO
28: tir:=lshift(ir+ir); if n then goto 40;	10xx ó 11xx
29: tir:=lshift(tir); if n then goto 35;	100x ó 101x
30: alu:=tir; if n then goto 33	1000 o 1001
31: a:=ir+sp;	1000=LODL
32: mar:=a; rd; goto 7;	
33: a:=ir+sp;	1001=STOL
34: mar:=a; mbr:=ac; wr; goto 10;	
35: alu:=tir; if n then goto 38	1010 ó 1011
36: a:=ir+sp;	1010=ADDL
37: mar:=a; rd; goto 13;	
38: a:= ir+sp;	1011=SUBL
39: mar:=a; rd; goto 16;	

```
40: tir:=lshift(tir); if n then goto 46;      110x ó 111x
41: alu:=tir; if n then goto 44;              1100 ó 1101
42: alu:=ac; if n then goto 22;               1100=JNEG
43: goto 0;
44: alu:=ac; if z then goto 0;                1101=JNZE
45: pc:=band(ir,amask); goto 0;
46: tir:=lshift(tir); if n then goto 50;
47: sp:=sp+(-1);                              1110=CALL
48: mar:=sp; mbr:=pc; wr;
49: pc:=band(ir,amask); wr; goto 0;           1111xxx
50: tir:=lshift(tir); if n then goto 65;
51: tir:=lshift(tir); if n then goto 59;
52: alu:=tir; if n then goto 56;
53: mar:=ac; rd;                              1111000=PSHI
54: sp:=sp+(-1); rd;
55: mar:=sp; wr; goto 10;
56: mar:=sp; sp:=sp+1; rd;                    1111001=POPI
57: rd;
58: mar:=ac; wr; goto10;
59: alu:=tir; if n then goto 62;
60: sp:=sp+(-1);                              1111010=PUSH
61: mar:=sp; mbr:=ac; wr; goto 10;
62: mar:=sp; sp:=sp+1; rd;                    1111011=POP
63: rd;
64: ac:=mbr; goto 0;
65: tir:=lshift(tir); if n then goto 73;
66: alu:=tir; if n then goto 70;
67: mar:=sp; sp:=sp+1; rd                     1111100=RETN
68: rd;
69: pc:=mbr; goto 0;
70: a:=ac;                                    1111101=SWAP
71: ac:=sp;
72: sp:=a; goto 0;
73: alu:=tir; if n then goto 76;
74: a:=band(ir,smask);                        1111110=INSP
75: sp:=sp+a; goto 0;
76: a:=band(ir,smask);                        1111111=DESP
77: a:=inv(a);
78: a:= a+1; goto 75;
```