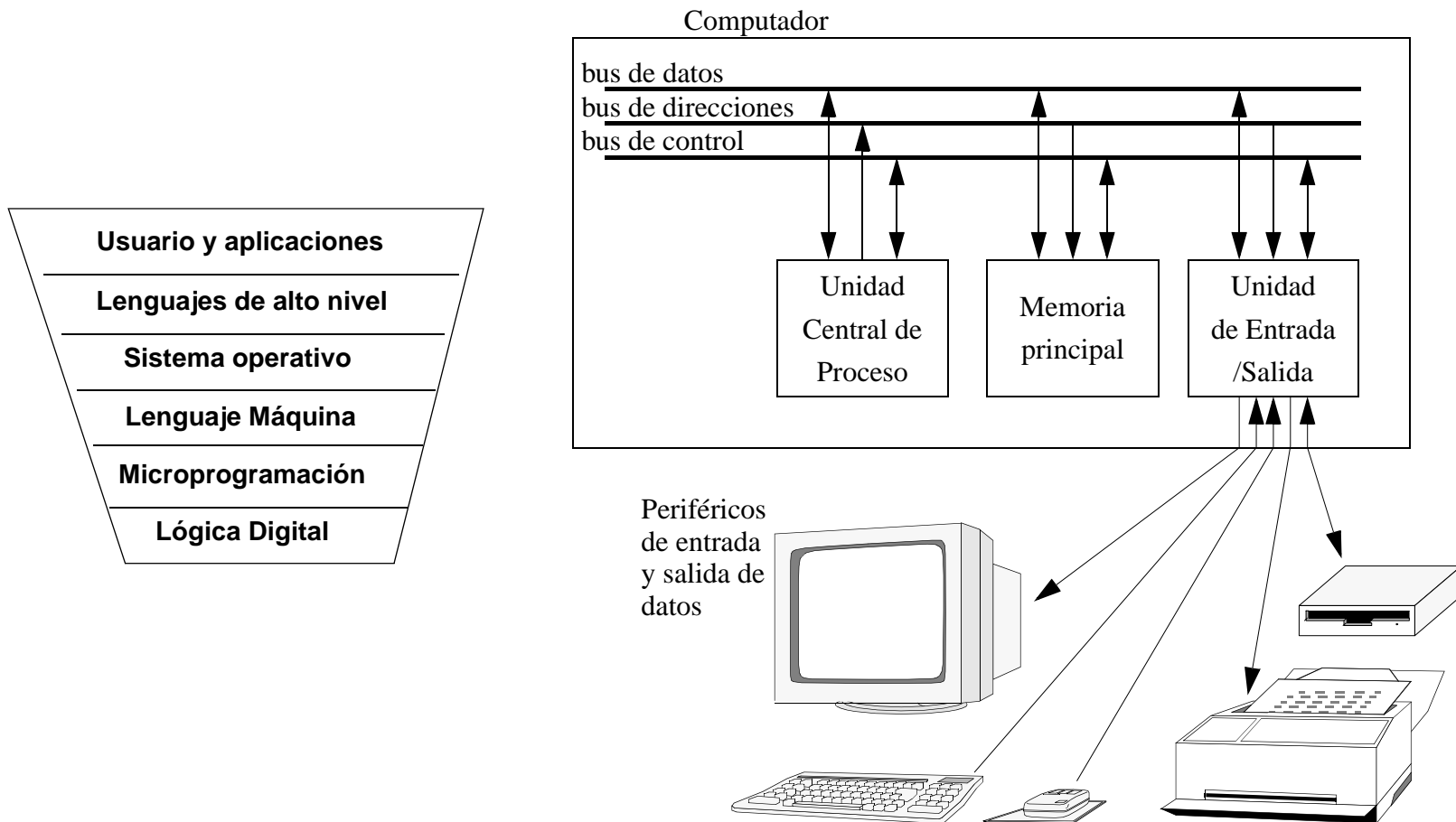
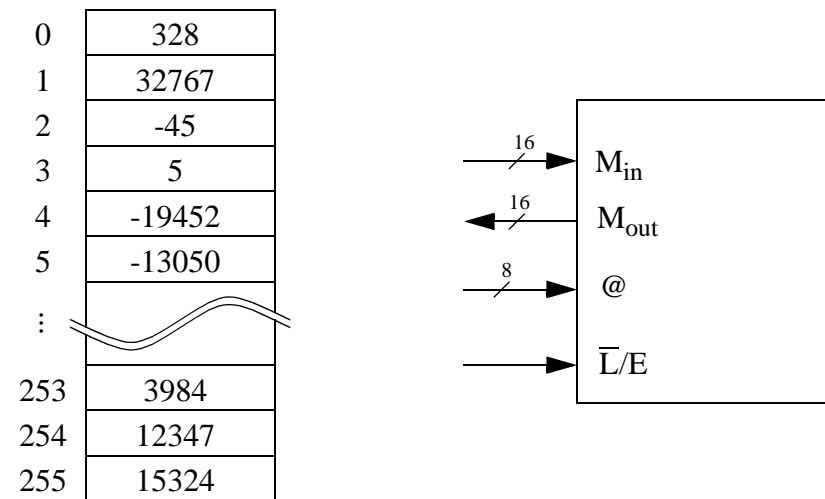


ARQUITECTURA VON NEUMANN

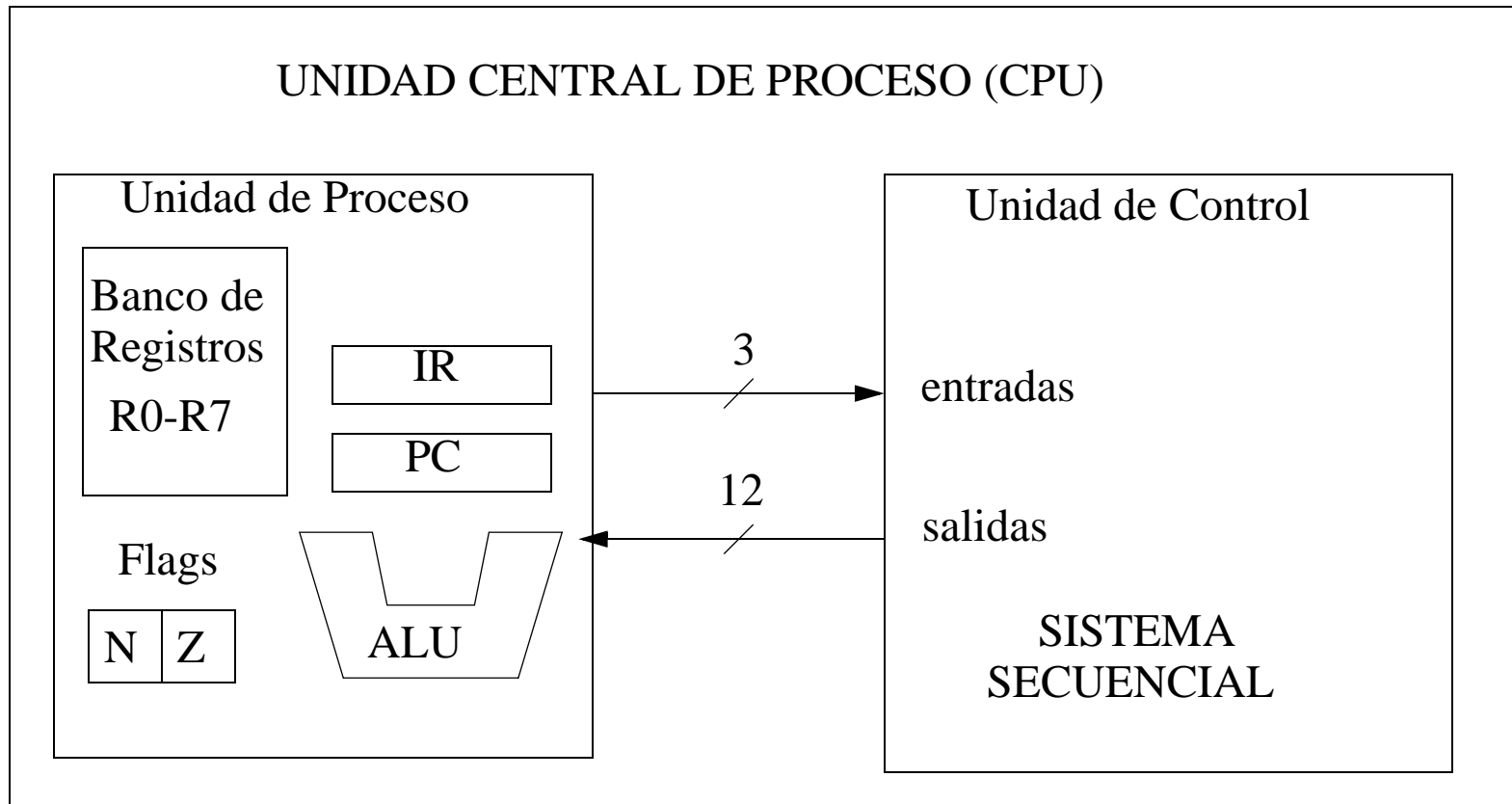


Descripción de la Arquitectura de la MR

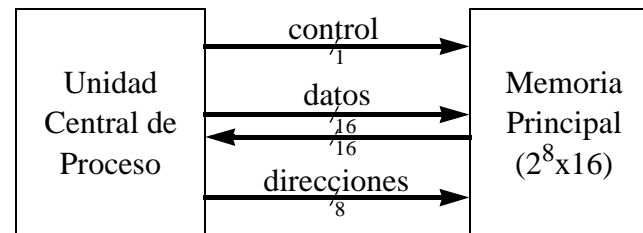
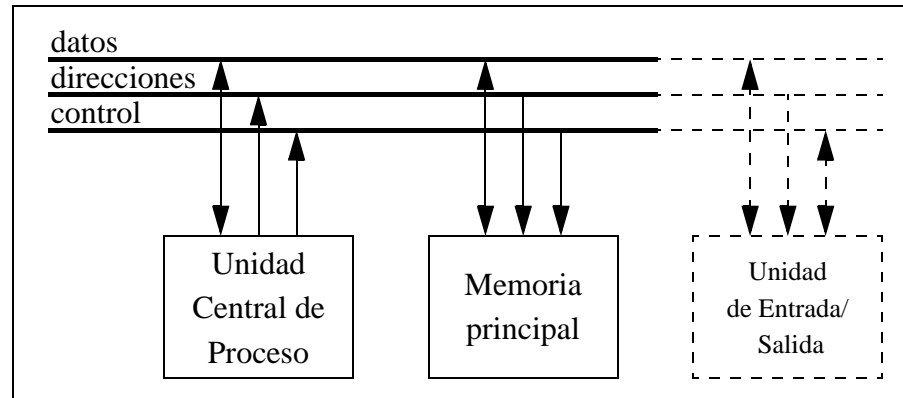
LA MEMORIA



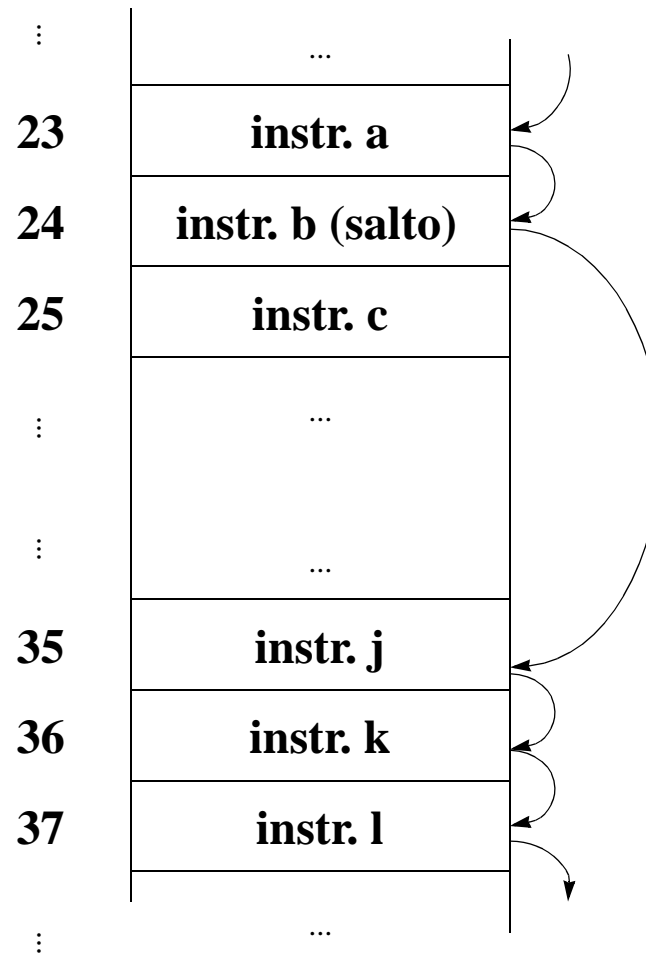
LA CPU



ARQUITECTURA DE LA MR



SECUENCIAMIENTO



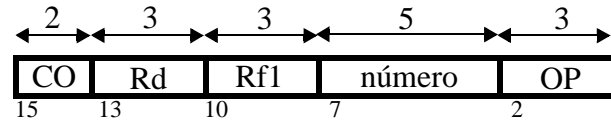
Nivel de Lenguaje Máquina de la MR

INSTRUCCIONES : ARITMÉTICO-LÓGICAS DE ACCESO A MEMORIA DE SALTO

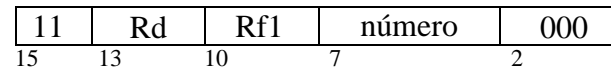
INSTRUCCIONES ARITMÉTICO-LÓGICAS

- ☐ **Suma - ADD**
- ☐ **Resta - SUB**
- ☐ **Desplazamiento aritmético a la derecha - ASR**
- ☐ **And-lógica - AND**
- ☐ **Suma con inmediato - ADDI**
- ☐ **Resta con inmediato - SUBI**

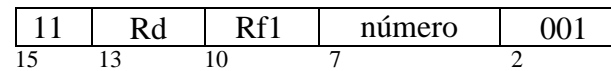
INSTRUCCIONES ARITMÉTICO-LÓGICAS



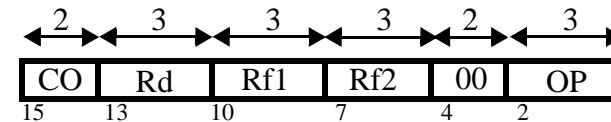
Formato de las instrucciones de suma y resta de inmediato.



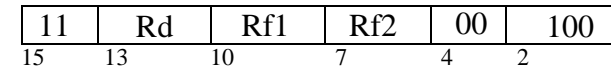
Suma con inmediato



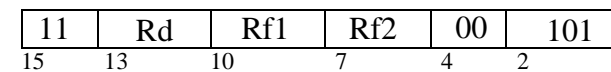
Resta con inmediato



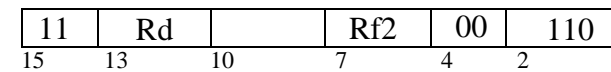
Formato de las instrucciones de suma, resta, desplazamiento a la derecha, y and lógica.



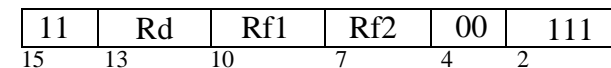
Suma



Resta



Desplazamiento a la derecha

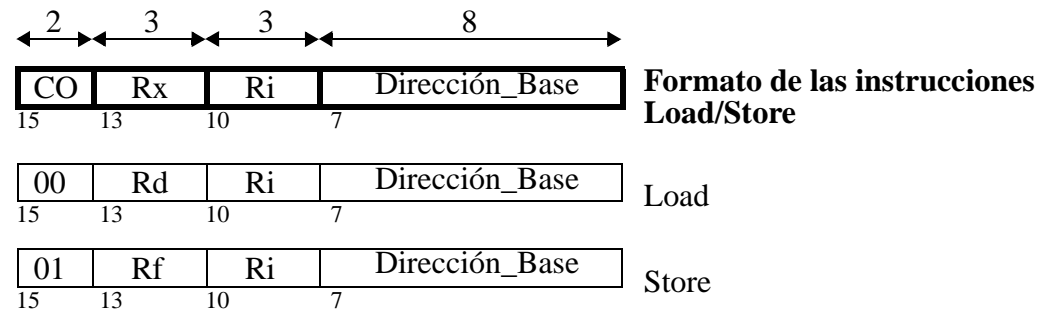


And lógica

INSTRUCCIONES DE ACCESO A MEMORIA

❑ LOAD

❑ STORE

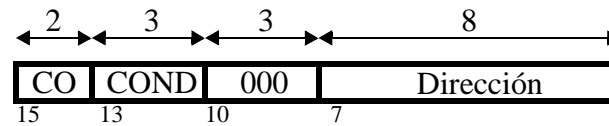


Dirección operando: Dirección_base + [Ri]

INSTRUCCIONES DE SALTO

- ☐ **Saltar si más pequeño - BL**
- ☐ **Saltar si más grande - BG**
- ☐ **Saltar si igual - BEQ**
- ☐ **Saltar si no igual - BNE**
- ☐ **Saltar si más pequeño o igual - BLE**
- ☐ **Saltar si más grande o igual - BGE**
- ☐ **Salto incondicional - BR**

INSTRUCCIONES DE SALTO



Formato de las instrucciones de salto

10	000	000	Dirección
15	13	10	7

Salto incondicional

10	001	000	Dirección
15	13	10	7

Saltar si igual

10	010	000	Dirección
15	13	10	7

Saltar si más pequeño

10	011	000	Dirección
15	13	10	7

Saltar si más pequeño o igual

10	101	000	Dirección
15	13	10	7

Saltar si no igual

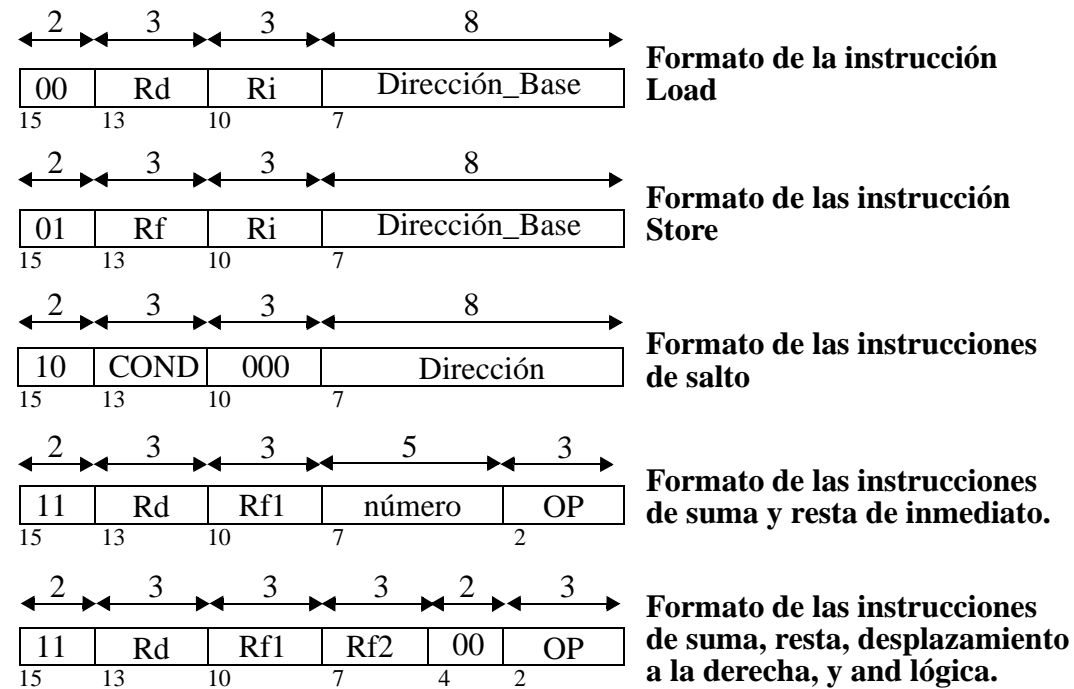
10	110	000	Dirección
15	13	10	7

Saltar si más grande o igual

10	111	000	Dirección
15	13	10	7

Saltar si más grande

RESUMEN DE INSTRUCCIONES



LENGUAJE ENSAMBLADOR

Notación en LE	CO	OP ₀₋₂	Operación	Flags de Condición
Addi Rf1, #núm, Rd	11	000	Rd := Rf1 + núm	Z := (Rf1 + núm = 0), N := (Rf1 + núm < 0).
Subi Rf1, #núm, Rd		001	Rd := Rf1 - núm	Z := (Rf1 - núm = 0), N := (Rf1 - núm < 0).
Add Rf1, Rf2, Rd		100	Rd := Rf1 + Rf2	Z := (Rf1 + Rf2 = 0), N := (Rf1 + Rf2 < 0).
Sub Rf1, Rf2, Rd		101	Rd := Rf1 - Rf2	Z := (Rf1 - Rf2 = 0), N := (Rf1 - Rf2 < 0).
Asr Rf2, Rd		110	Rd := Rf2 >> 1	Z := (Rf2 >> 1 = 0), N := Rf2 ₁₅ .
And Rf1, Rf2, Rd		111	Rd := Rf1 ∧ Rf2	Z := (Rf1 ∧ Rf2 = 0), N := (Rf1 ∧ Rf2 < 0).

Tabla 1. Instrucciones Aritmético-lógicas.

Notación en LE	CO	Operación	Flags de Condición
LOAD dir_base(Ri), Rd	00	$Rd := M[\text{dir_base} + Ri]$	$Z := (M[\text{dir_base} + Ri] = 0)$, $N := (M[\text{dir_base} + Ri] < 0)$,
STORE Rf, dir_base(Ri)	01	$M[\text{dir_base} + Ri] := Rf$	Z y N no cambian.

Tabla 2. Instrucciones de Acceso a Memoria.

Notación en LE	CO	COND	Condición	Comentarios
BR dir_absoluta	10	000	1	Salto incondicional
BEQ dir_absoluta		001	Z	Salta si igual
BL dir_absoluta		010	N	Salta si más pequeño
BLE dir_absoluta		011	$N \vee Z$	Salta si más pequeño o igual
		100	-	Codificación no usada.
BNE dir_absoluta		101	\overline{Z}	Salta si no igual
BGE dir_absoluta		110	\overline{N}	Salta si más grande o igual
BG dir_absoluta		111	$\overline{N} \vee \overline{Z}$	Salta si más grande

Tabla 3. Instrucciones de Salto.

PROGRAMA EJEMPLO

Lenguaje Alto Nivel

Programa a_por_b

var a, b, sum: entero;

a:=10;

b:=5;

sum:=0;

mientras b>0 hacer

 sum:=sum+a;

 b:=b-1;

fmientras

fprograma

Lenguaje Ensamblador

```
0  [@0]=10
1  [@1]=5
2  [@2]=0
3  LOAD 0(R0), R1
4  LOAD 1(R0), R2
5  ADDI R0, #0, R3
6  SUBI R2, #0, R0
7  BLE 11
8  ADD R3, R1, R3
9  SUBI R2, #1, R2
10 BR 6
11 STORE R3, 2(R0)
```

Lenguaje Máquina

```
0  00000000000001010
1  0000000000000101
2  0000000000000000
3  00 001 000 00000000
4  00 010 000 00000001
5  11 011 000 00000 000
6  11 000 010 00000 001
7  10 011 000 00001011
8  11 011 011 001 00 100
9  11 010 010 00001 001
10 10 000 000 00000110
11 01 011 000 00000010
```

UP: UNIDAD ARITMÉTICO-LÓGICA

OPERACIONES A REALIZAR:

☐ **SUMA: add, addi**

☐ **RESTA: sub, subi**

☐ **DESPLAZAMIENTO ARITMÉTICO 1 BIT**

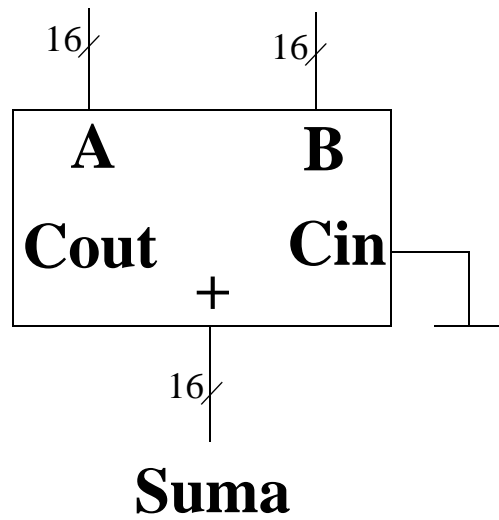
A LA DERECHA: ASR

☐ **AND LÓGICA: and**

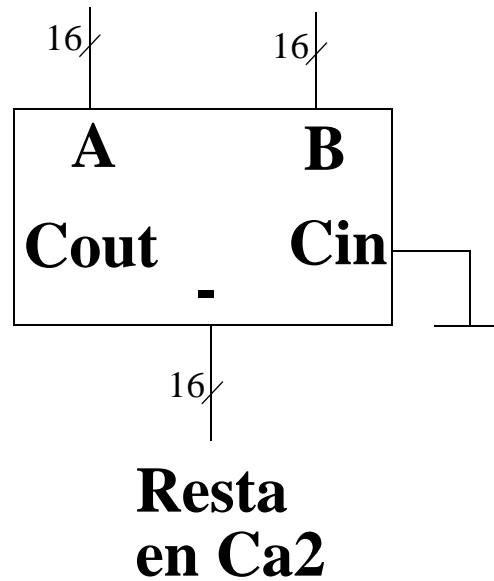
☐ **DEJAR PASAR OPERANDO: load**

UP: UNIDAD ARITMÉTICO-LÓGICA

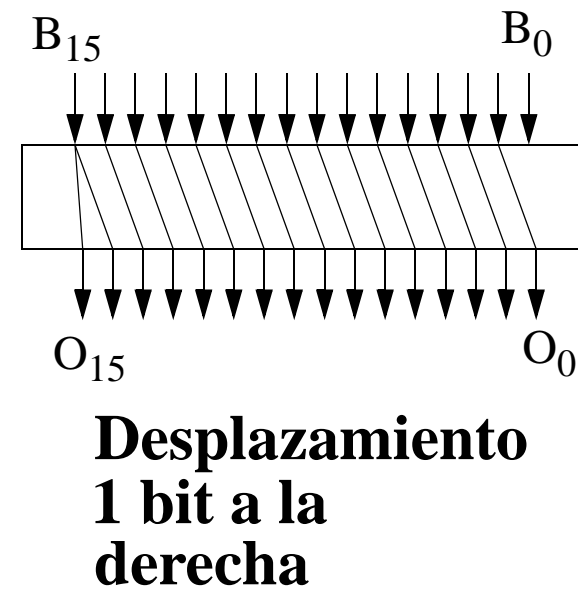
Sumador



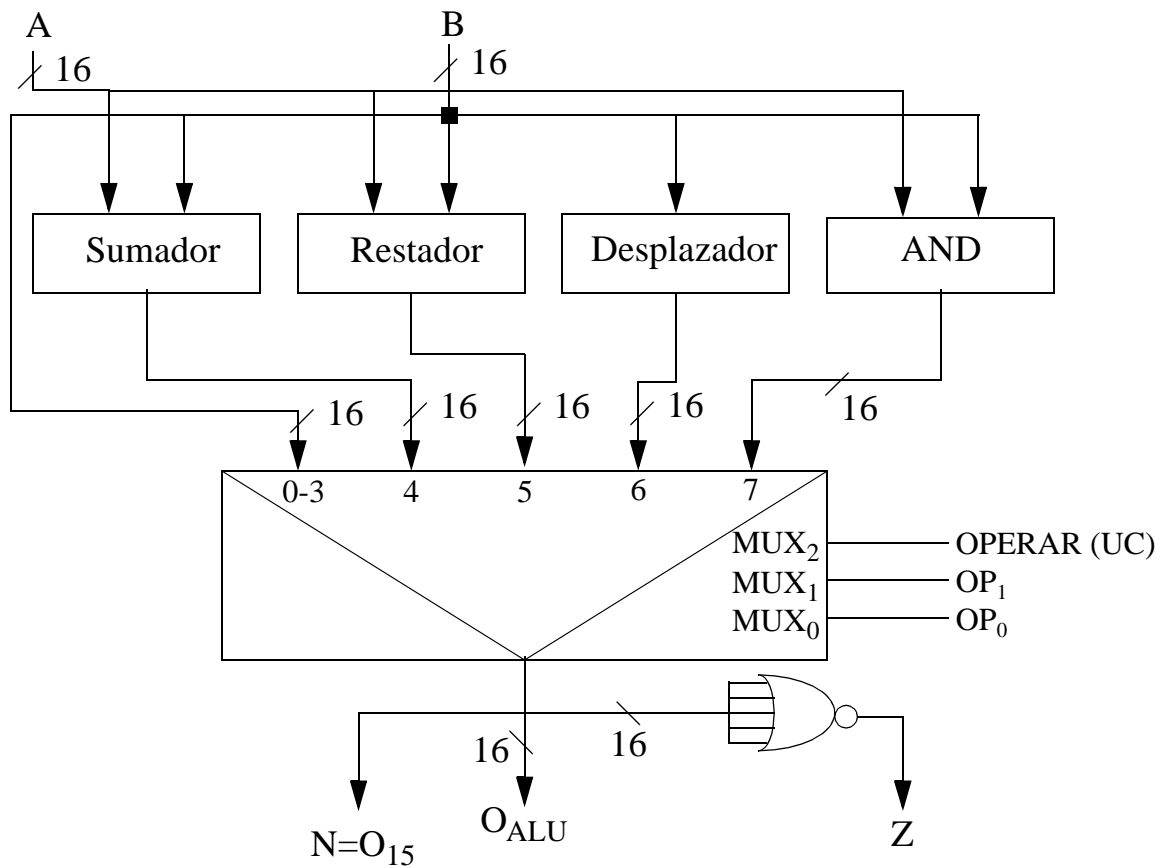
Restador



Desplazador

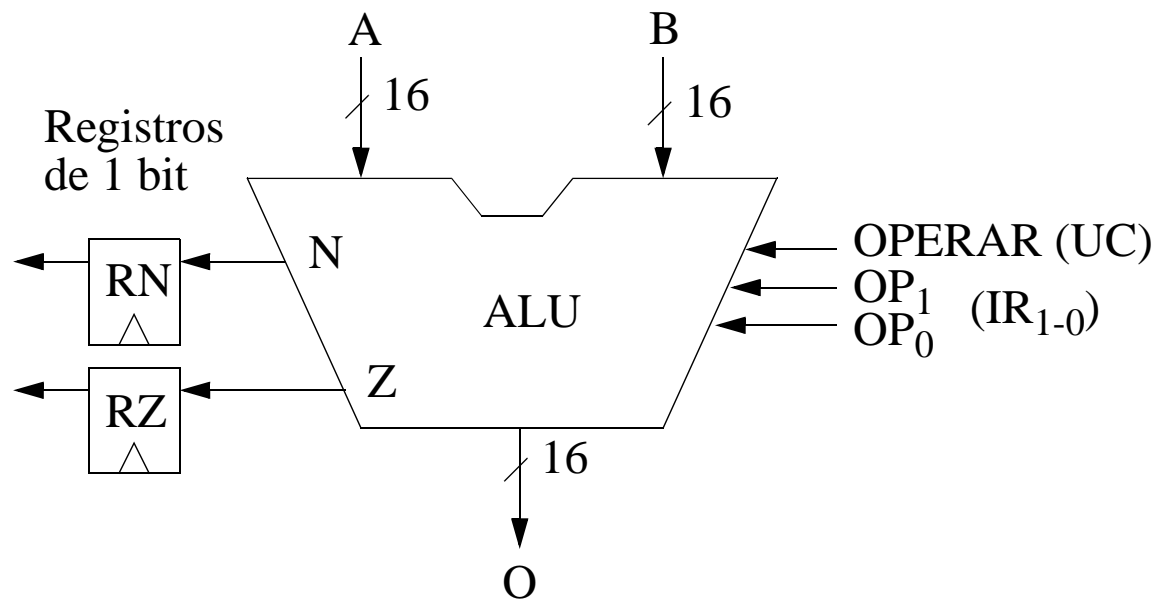


UP: UNIDAD ARITMÉTICO-LÓGICA



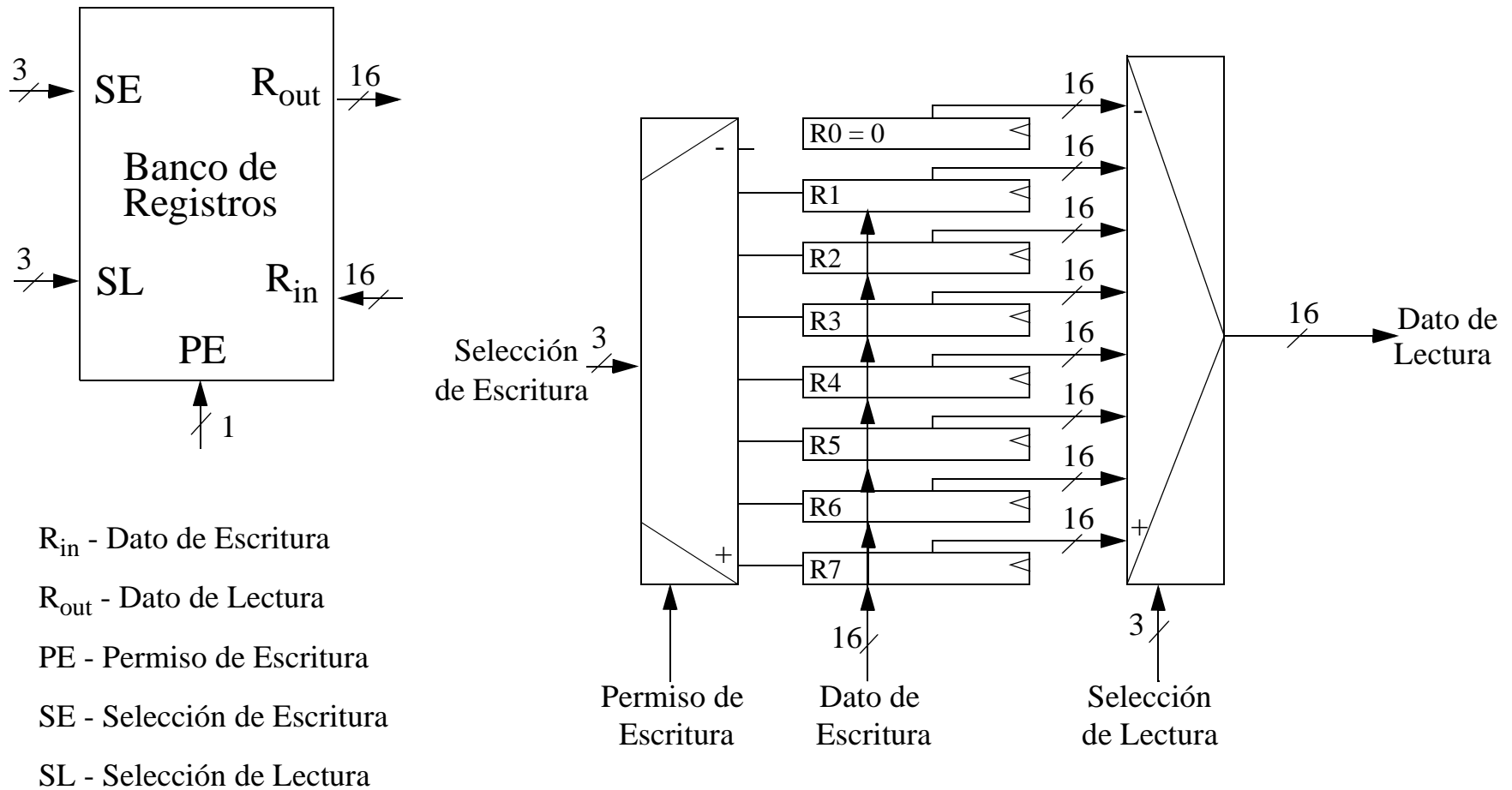
OPERAR OP ₁ OP ₀	Operación
0xx	Dejar pasar B
100	A + B
101	A - B
110	B >> 1
111	A and B

UP: UNIDAD ARITMÉTICO-LÓGICA

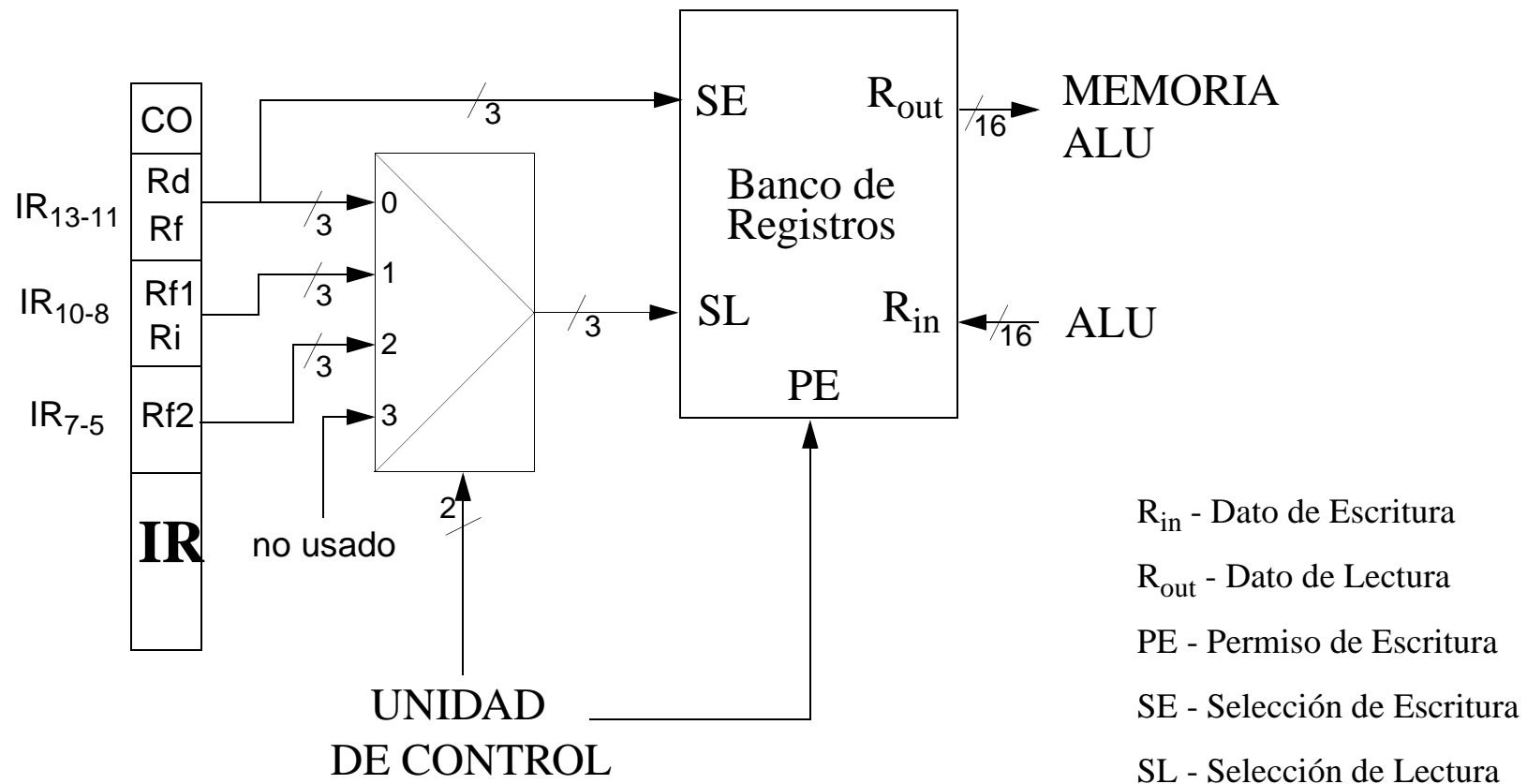


OPERAR OP ₁ OP ₀	Operación
0xx	Dejar pasar B
100	A + B
101	A - B
110	B >> 1
111	A and B

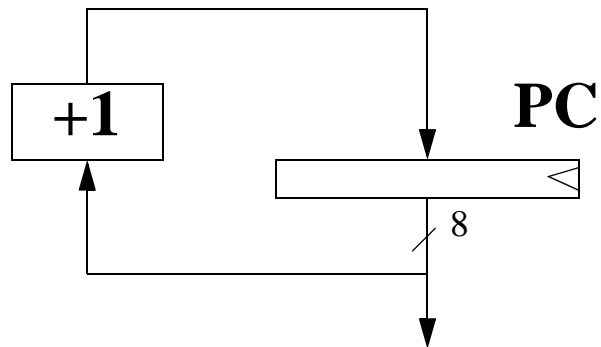
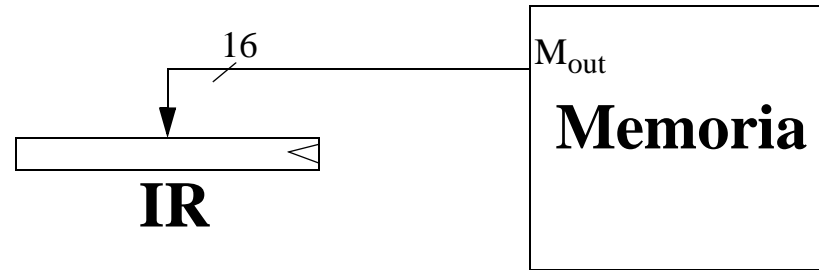
UP: BANCO DE REGISTROS



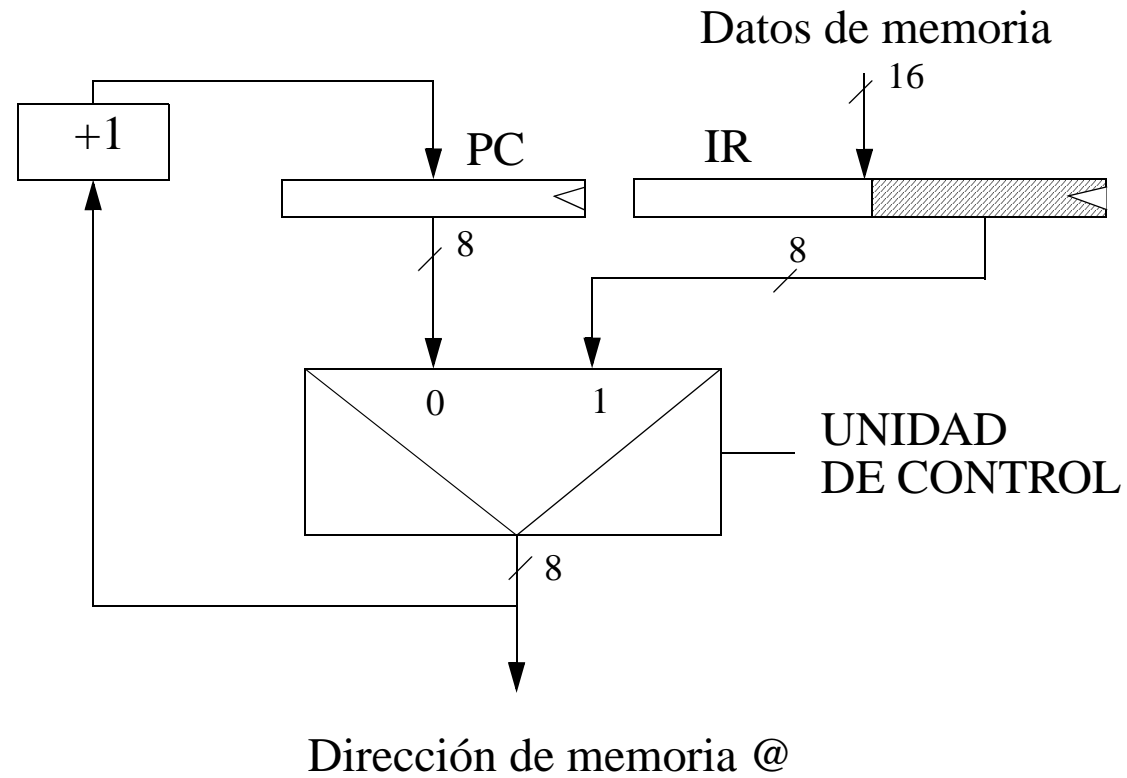
UP: COMUNICACIÓN B. DE REGISTROS-ALU-MEMORIA



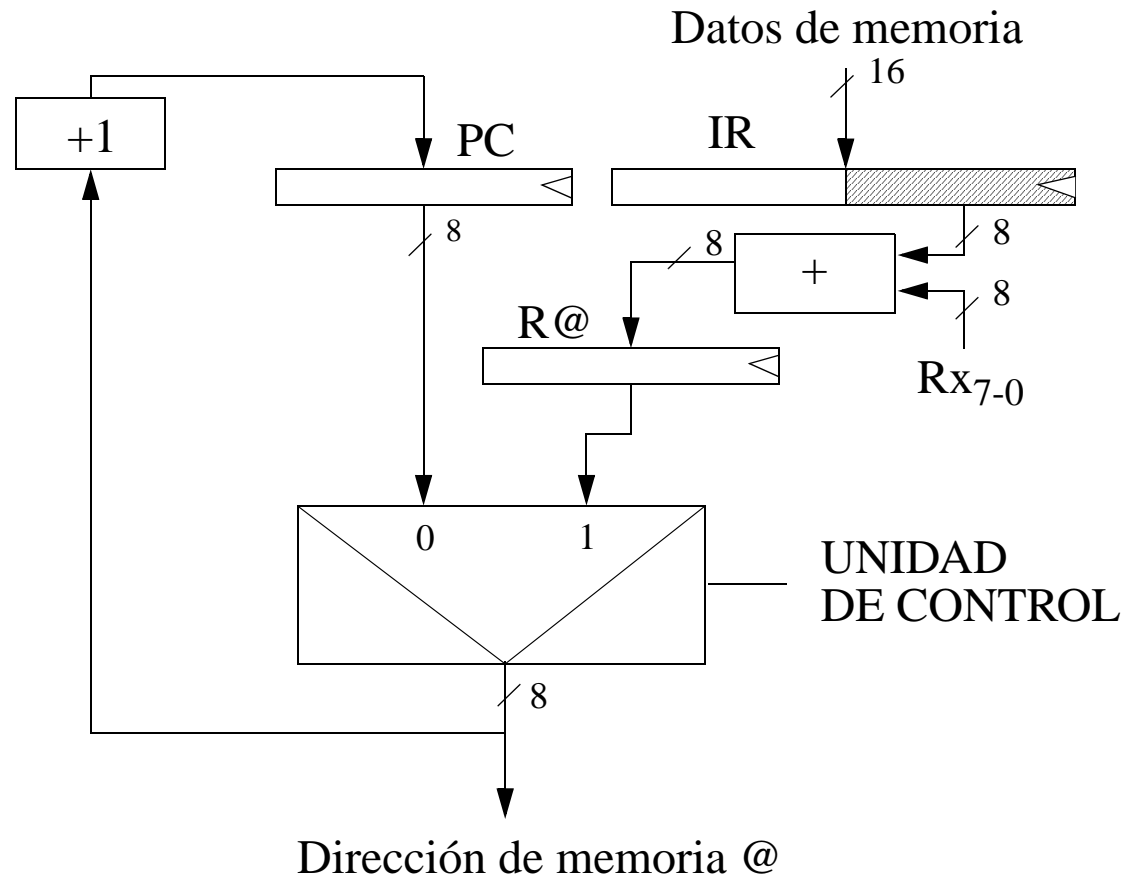
UP: REGISTROS ESPECIALES: PC, IR



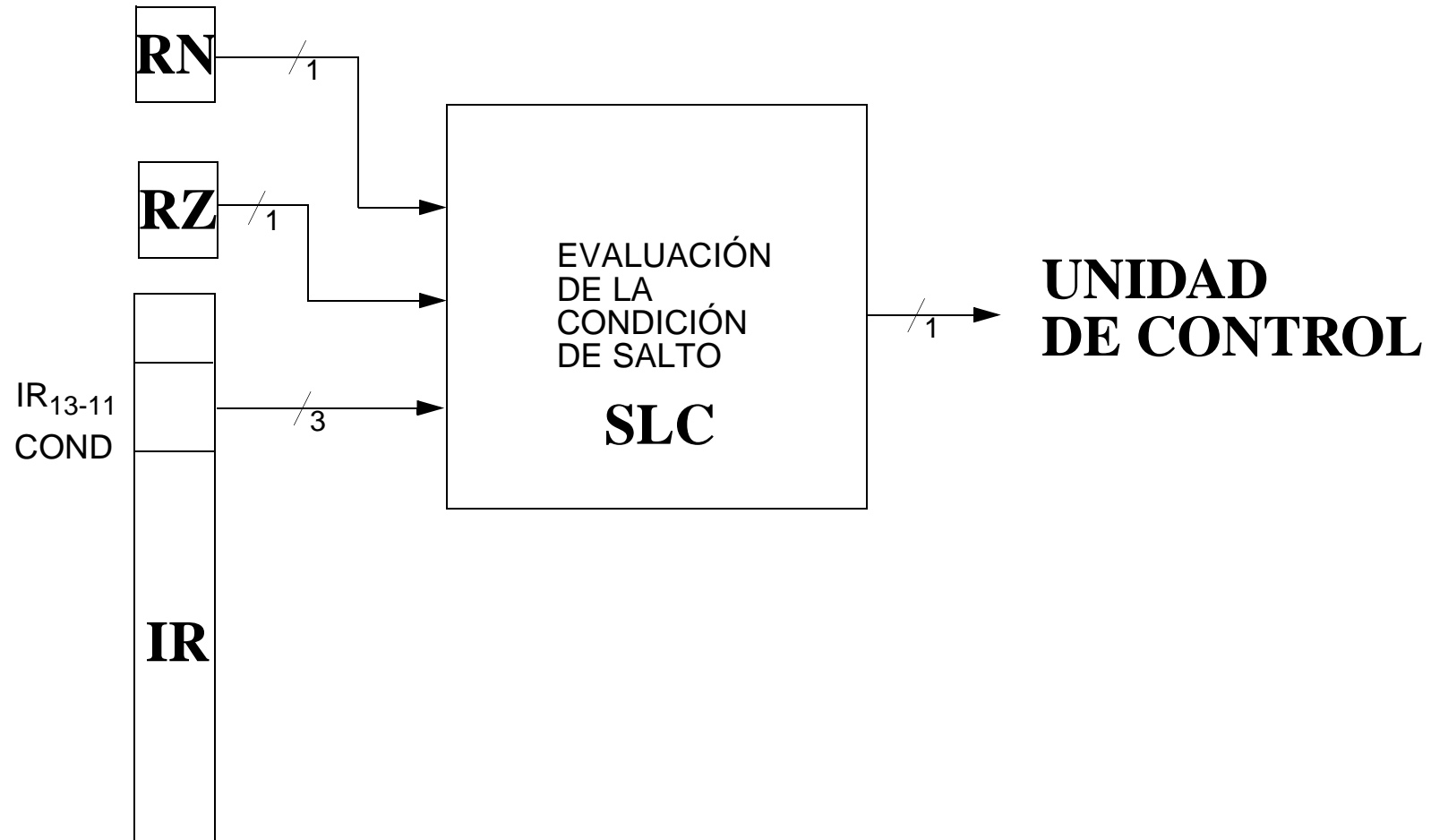
UP: SELECCIÓN DE LA DIRECCIÓN DE LA SIGUIENTE INSTRUCCIÓN A EJECUTAR



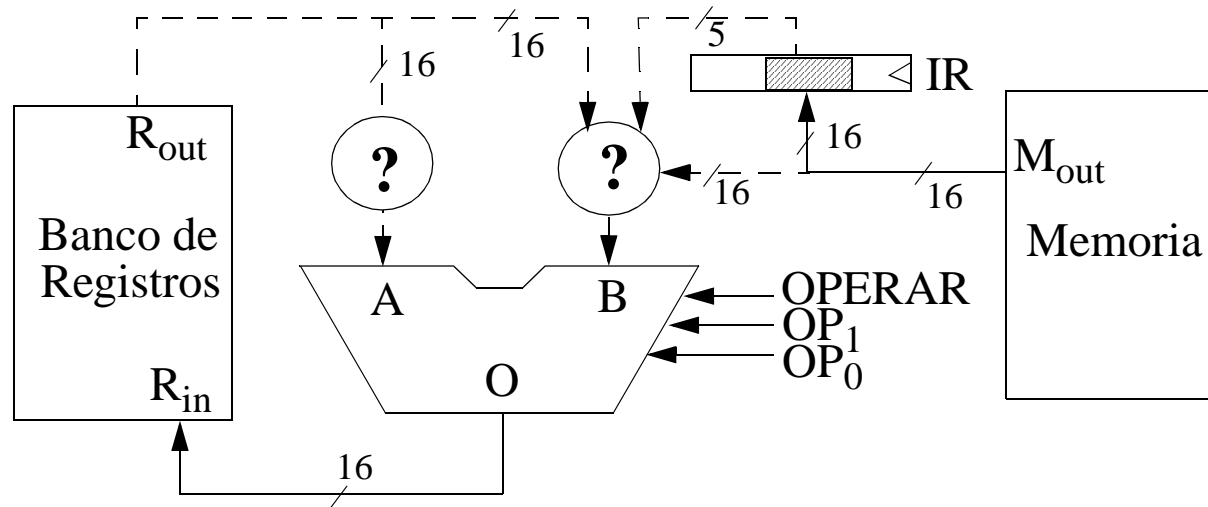
UP: LÓGICA DE GENERACIÓN DE DIRECCIONES



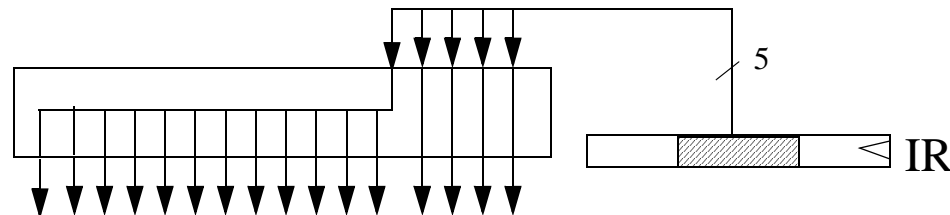
UP: EVALUACIÓN DE LA CONDICIÓN DE SALTO



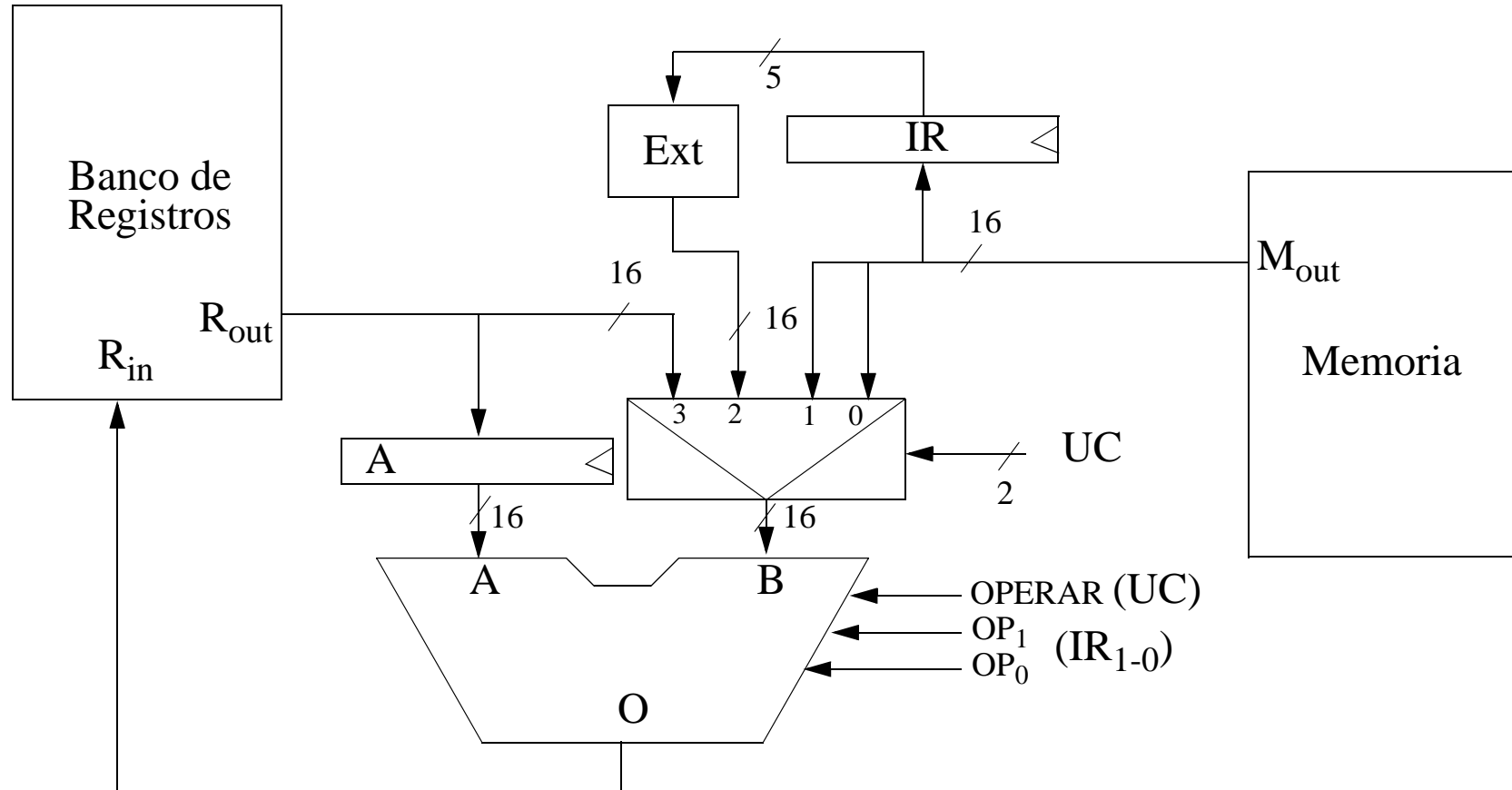
UP: COMUNICACIÓN MEMORIA-ALU-IR-B. DE REGISTROS



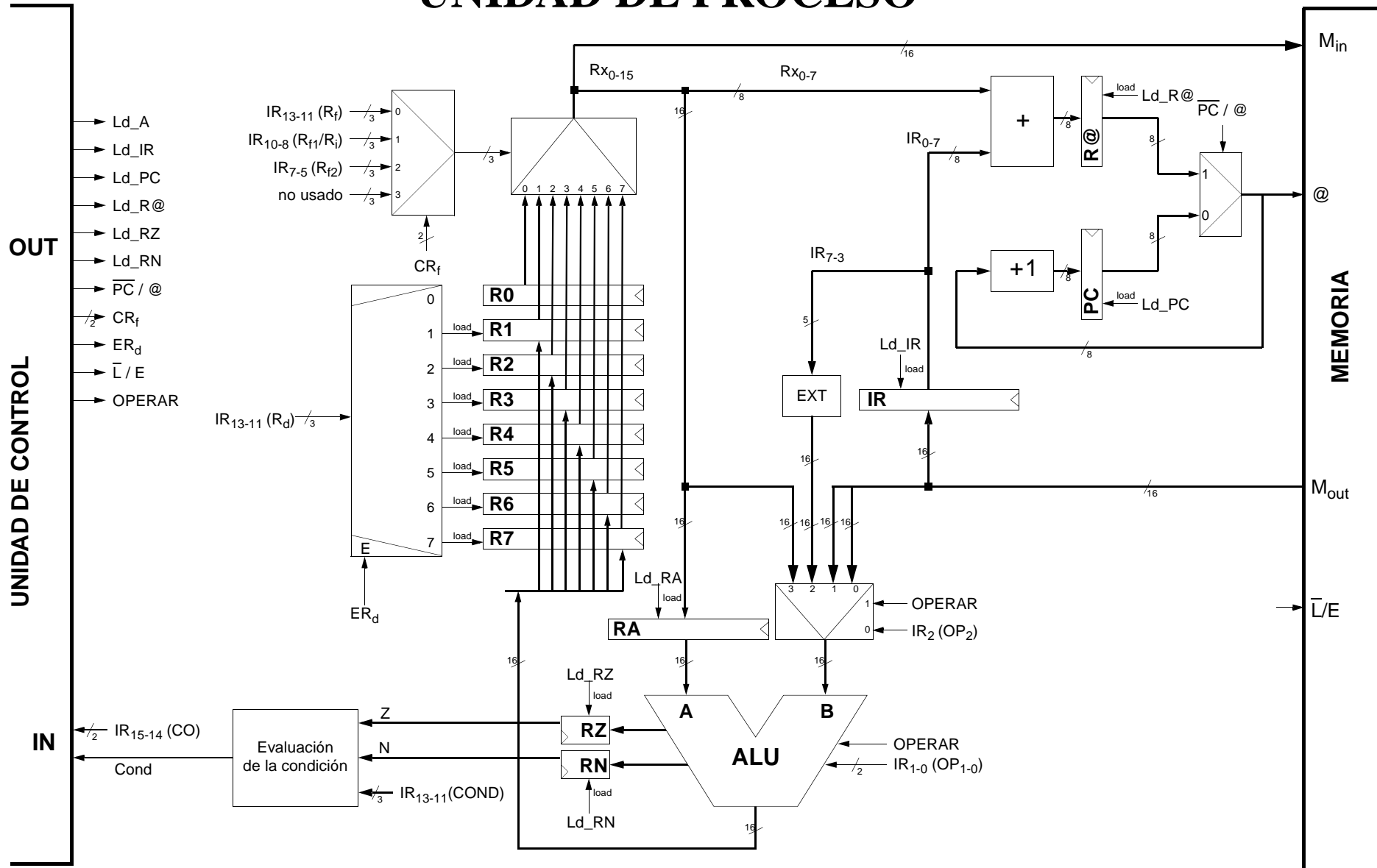
EXTENSIÓN DE SIGNO DEL OPERANDO INMEDIATO



UP: COMUNICACIÓN MEMORIA-B. DE REGISTROS-IR-ALU



UNIDAD DE PROCESO



RESUMEN DE INSTRUCCIONES

15	14	13	11	10	8	7	0
CO	Rd / Rf		Ri	dir_base			

LOAD / STORE

Instrucción	CO
LOAD dir_base(Ri), Rd	00
STORE Rf, dir_base(Ri)	01

15	14	13	11	10	8	7	0
CO	COND	000	dir_absoluta				SALTO

Instrucción	CO	COND	Condición
BR dir_absoluta	10	000	1
BEQ dir_absoluta		001	Z
BL dir_absoluta		010	N
BLE dir_absoluta		011	Z + N
BNE dir_absoluta		101	\bar{Z}
BGE dir_absoluta		110	\bar{N}
BG dir_absoluta		111	$\bar{N} + \bar{Z}$

15	14	13	11	10	8	7	5	4	3	2	0
CO		Rd		Rf1		Rf2		00		OP	
ARIT / LOGIC											

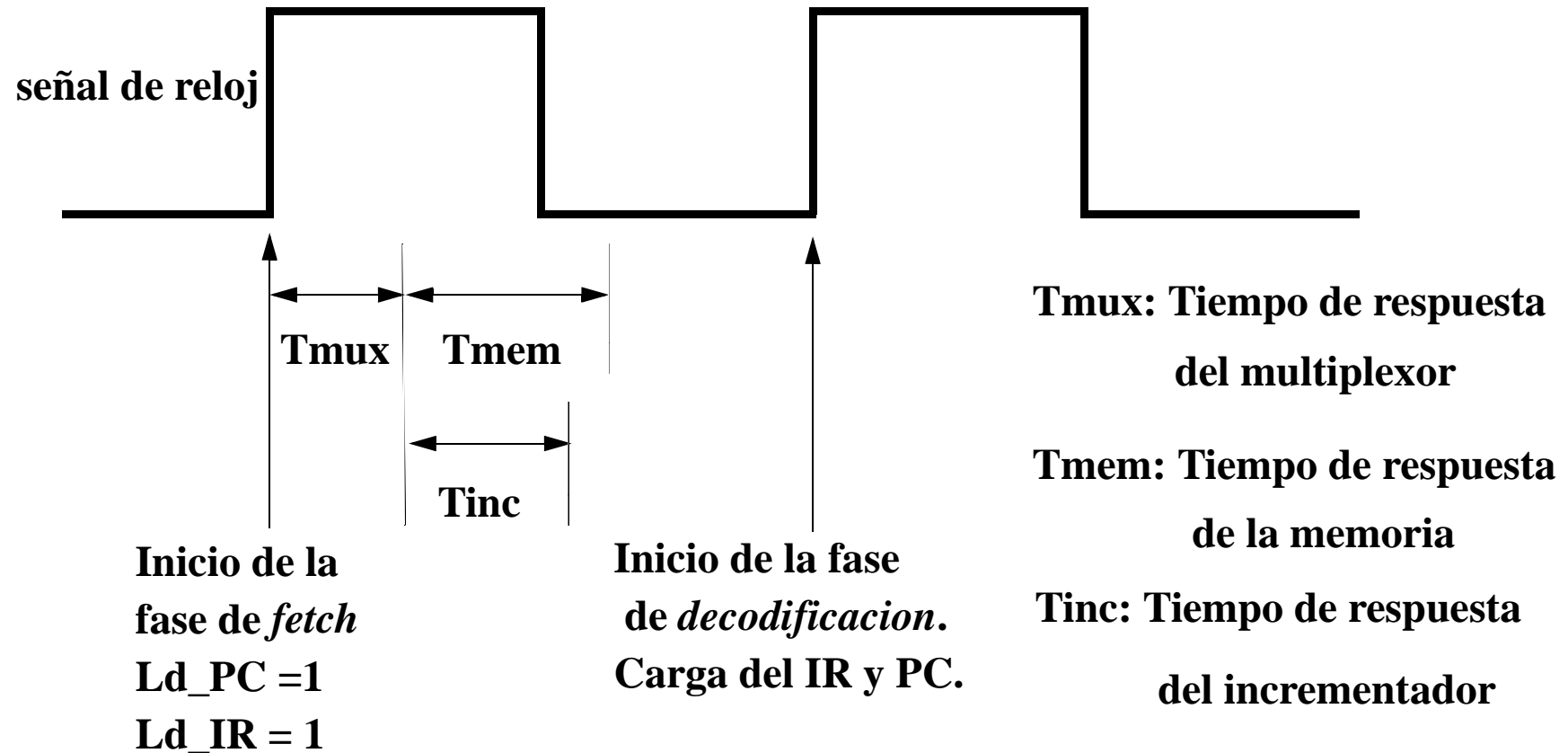
15	14	13	11	10	8	7			3	2	0
CO		Rd		Rf1		número			OP		
ARIT- INMEDIATO											

Instrucción	CO	OP
ADD Rf1, Rf2, Rd	11	100
SUB Rf1, Rf2, Rd		101
ASR Rf2, Rd		110
AND Rf1, Rf2, Rd		111
ADDI Rf1, #num, Rd		000
SUBI Rf1, #num, Rd		001

FASES DE EJECUCIÓN DE UNA INSTRUCCIÓN

- ☐ **Fetch**
- ☐ **Decodificación**
- ☐ **Búsqueda de operandos + Evaluacion Cond. de Salto**
- ☐ **Ejecución y escritura del resultado**

FASES DE FETCH Y DECODIFICACIÓN



GRAFO DE ESTADOS DE LAS FASES DE FETCH Y DECODIFICACIÓN

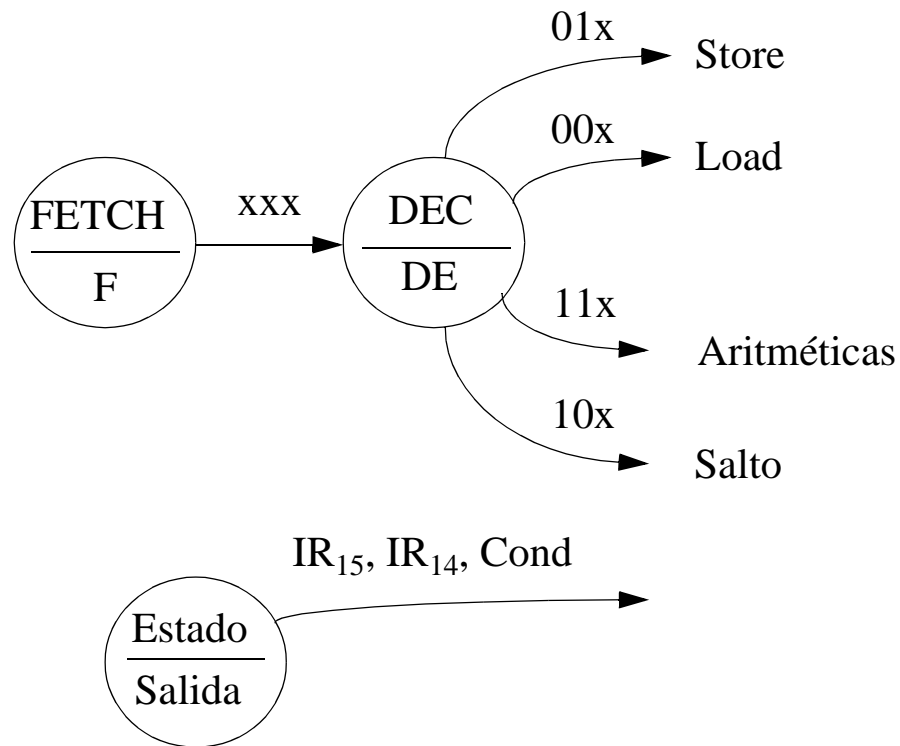
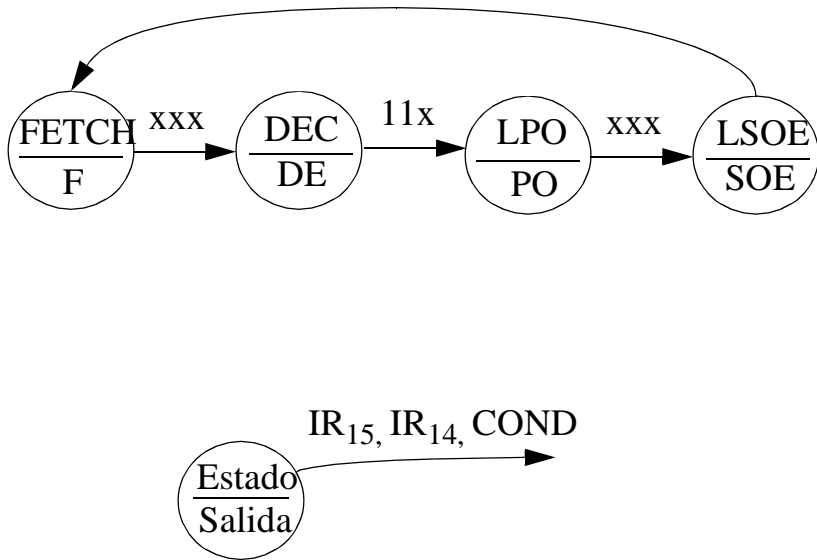


TABLA DE SALIDAS

Salidas UC	F	DE
Ld_IR	1	0
Ld_PC	1	0
Ld_R@	0	0
Ld_RA	0	0
Ld_RZ	0	0
Ld_RN	0	0
ER _d	0	0
\bar{L}/E	0	0
$\overline{PC}/@$	0	x
CR _f	xx	xx
OPERAR	x	x

GRAFO DE ESTADOS DE LAS INSTRUCCIONES ARITMÉTICAS

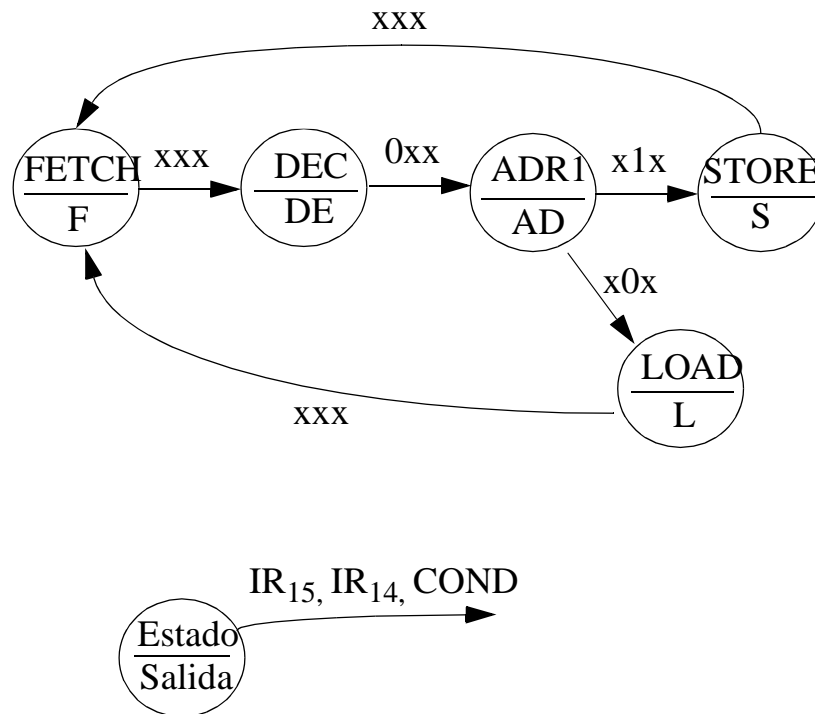
TABLA DE SALIDAS



Salidas UC	PO	SOE
Ld_IR	0	0
Ld_PC	0	0
Ld_R@	0	0
Ld_RA	1	0
Ld_RZ	0	1
Ld_RN	0	1
ER _d	0	1
\overline{L}/E	0	0
$\overline{PC}/@$	x	x
CR _f	01	10
OPERAR	x	1

GRAFO DE ESTADOS DE LAS INSTRUCCIONES DE ACCESO A MEMORIA

TABLA DE SALIDAS



Salidas UC	AD	L	S
Ld_IR	0	0	0
Ld_PC	0	0	0
Ld_R@	1	0	0
Ld_RA	0	0	0
Ld_RZ	0	1	0
Ld_RN	0	1	0
ER _d	0	1	0
\overline{L}/E	0	0	1
$\overline{PC}/@$	x	1	1
CR _f	01	xx	00
OPERAR	x	0	x

GRAFO DE ESTADOS DE LAS INSTRUCCIONES DE SALTO

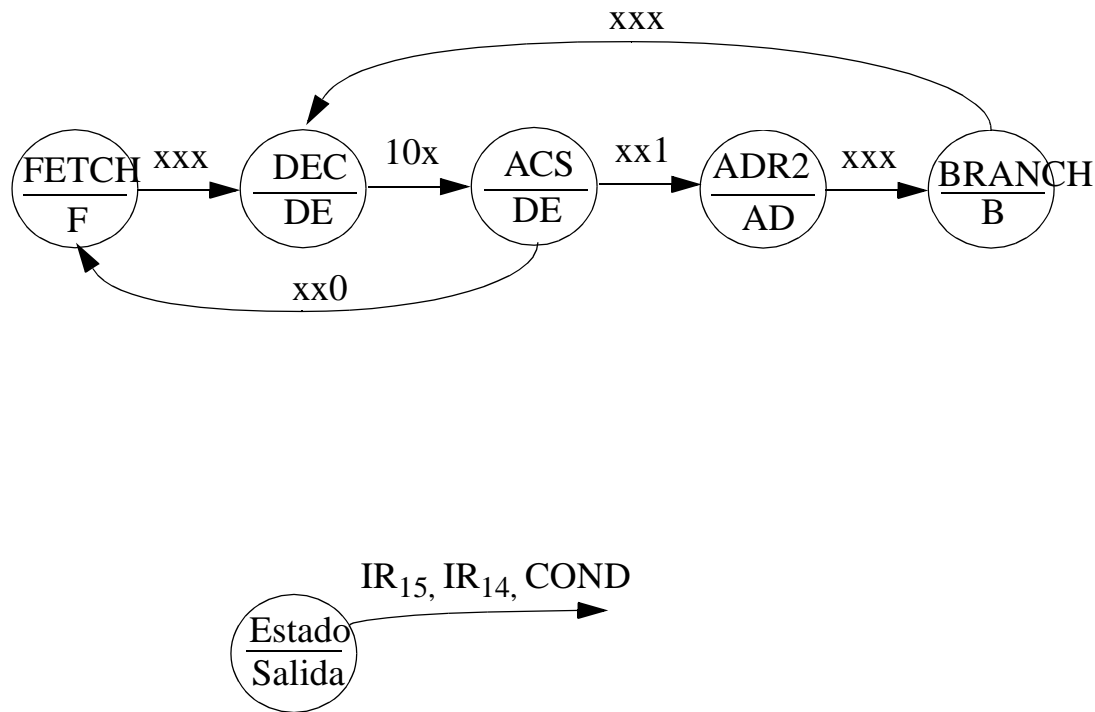


TABLA DE SALIDAS

Salidas UC	DE	AD	B
Ld_IR	0	0	1
Ld_PC	0	0	1
Ld_R@	0	1	0
Ld_RA	0	0	0
Ld_RZ	0	0	0
Ld_RN	0	0	0
ER _d	0	0	0
\overline{L}/E	0	0	0
$\overline{PC}/@$	x	x	1
CR _f	xx	01	xx
OPERAR	x	x	x

GRAFO DE ESTADOS DE LA UC DE LA MR

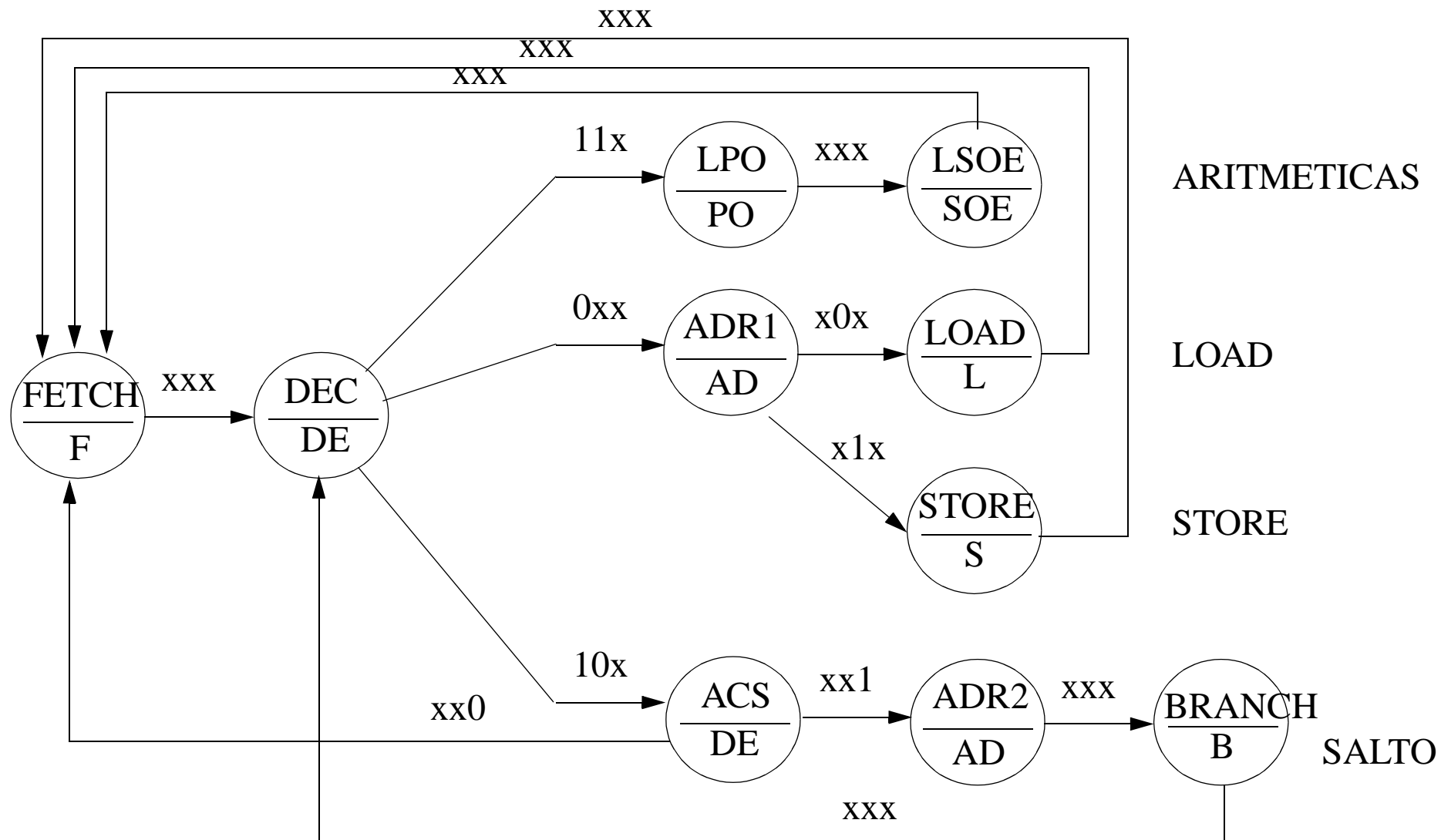
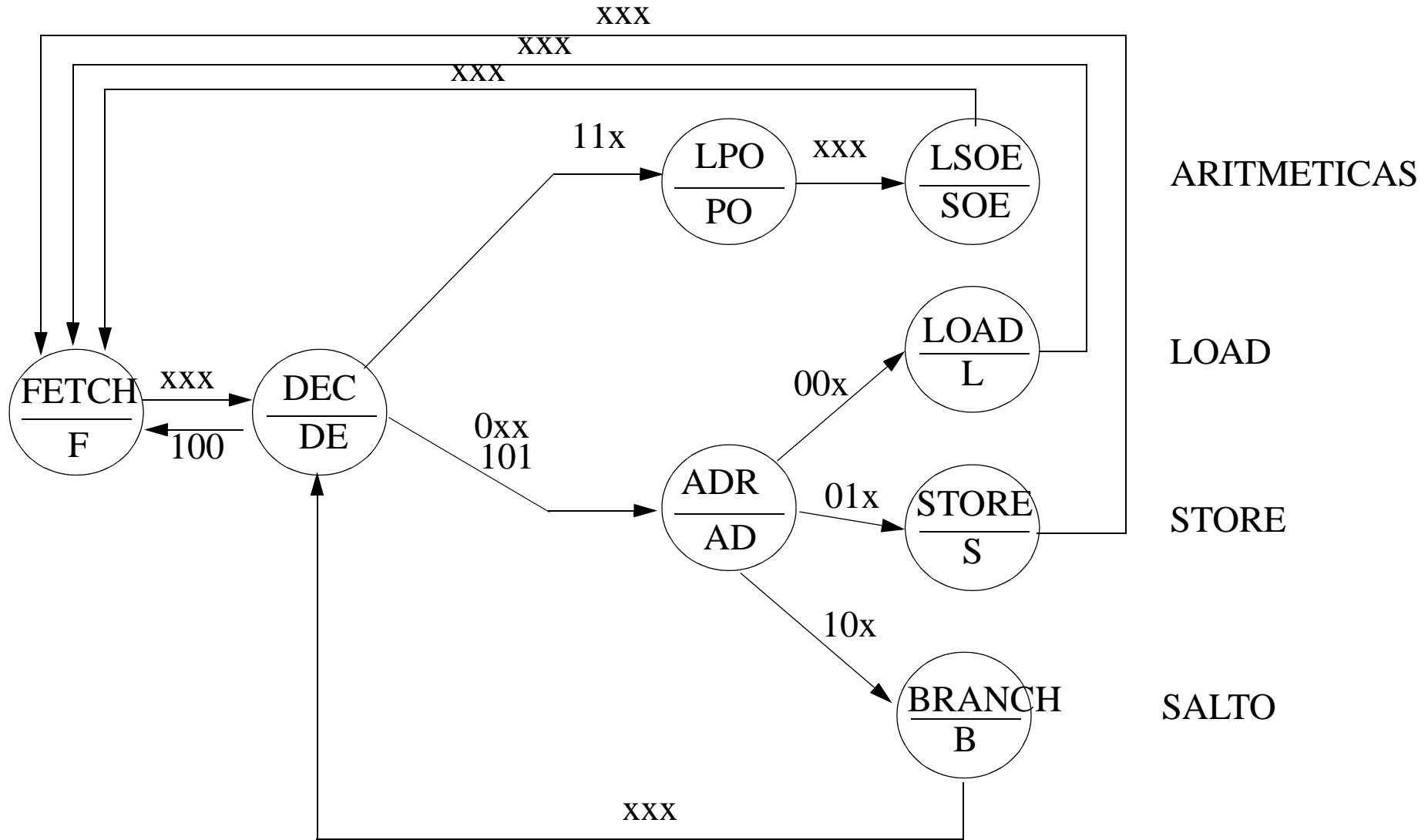


TABLA DE SALIDAS DE LA UC DE LA MR

Salidas UC	F	DE	AD	L	S	PO	SOE	B
Ld_IR	1	0	0	0	0	0	0	1
Ld_PC	1	0	0	0	0	0	0	1
Ld_R@	0	0	1	0	0	0	0	0
Ld_RA	0	0	0	0	0	1	0	0
Ld_RZ	0	0	0	1	0	0	1	0
Ld_RN	0	0	0	1	0	0	1	0
ER _d	0	0	0	1	0	0	1	0
\overline{L}/E	0	0	0	0	1	0	0	0
$\overline{PC}/@$	0	x	x	1	1	x	x	1
CR _f	xx	xx	01	xx	00	01	10	xx
OPERAR	x	x	x	0	x	x	1	x

OPTIMIZACIÓN DE LA UNIDAD DE CONTROL



GRAFO DE ESTADOS SIMPLIFICADO DE LA UC

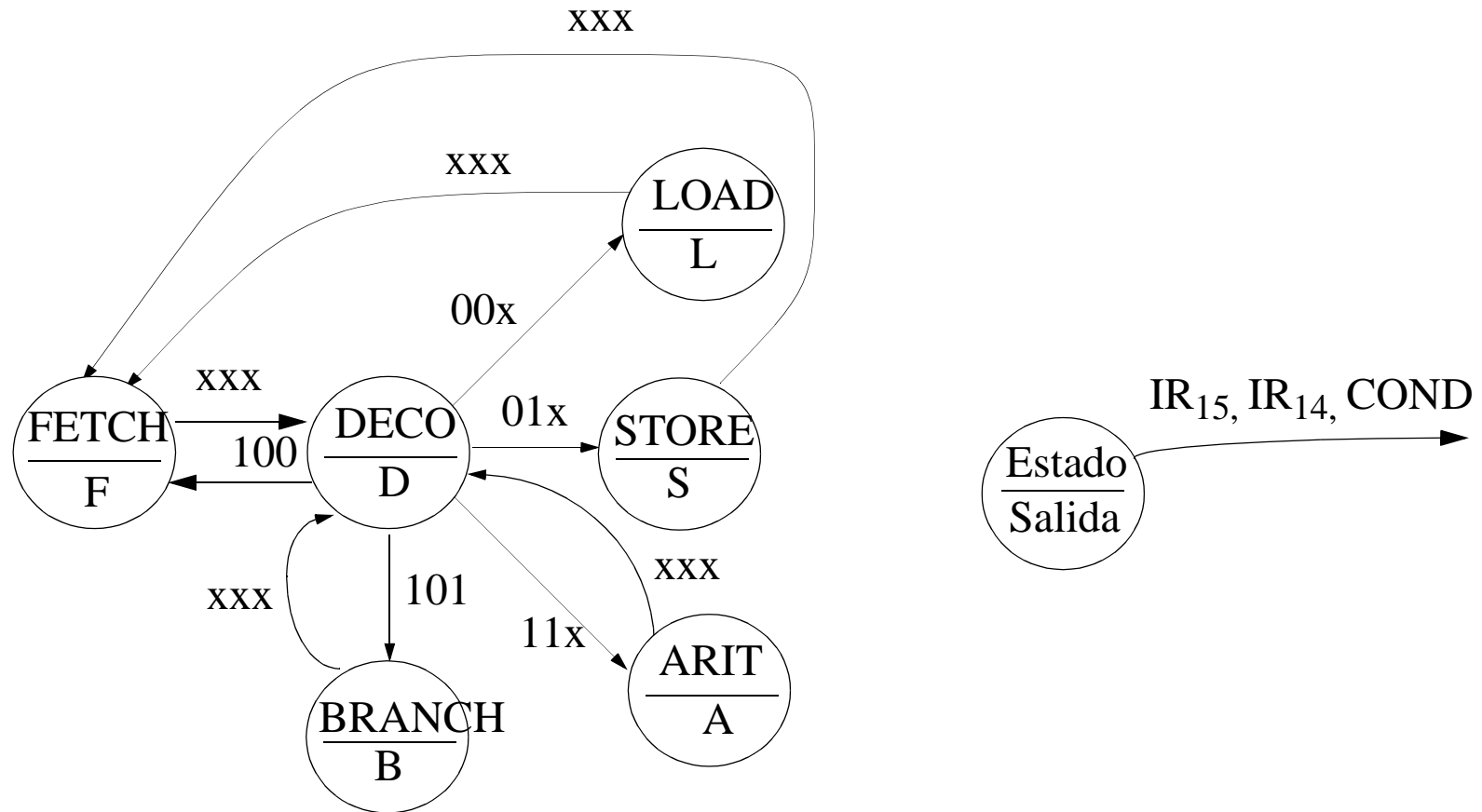


TABLA DE SALIDAS SIMPLIFICADA DE LA UC

Salidas UC	F	D	A	L	S	B
Ld_IR	1	0	1	0	0	1
Ld_PC	1	0	1	0	0	1
Ld_R@	0	1	0	0	0	0
Ld_RA	0	1	0	0	0	0
Ld_RZ	0	0	1	1	0	0
Ld_RN	0	0	1	1	0	0
ER _d	0	0	1	1	0	0
\overline{L}/E	0	0	0	0	1	0
$\overline{PC}/@$	0	X	0	1	1	1
CR _f	xx	01	10	xx	00	xx
OPERAR	x	x	1	0	x	x

IMPLEMENTACIÓN CON ROM DE LA UC

