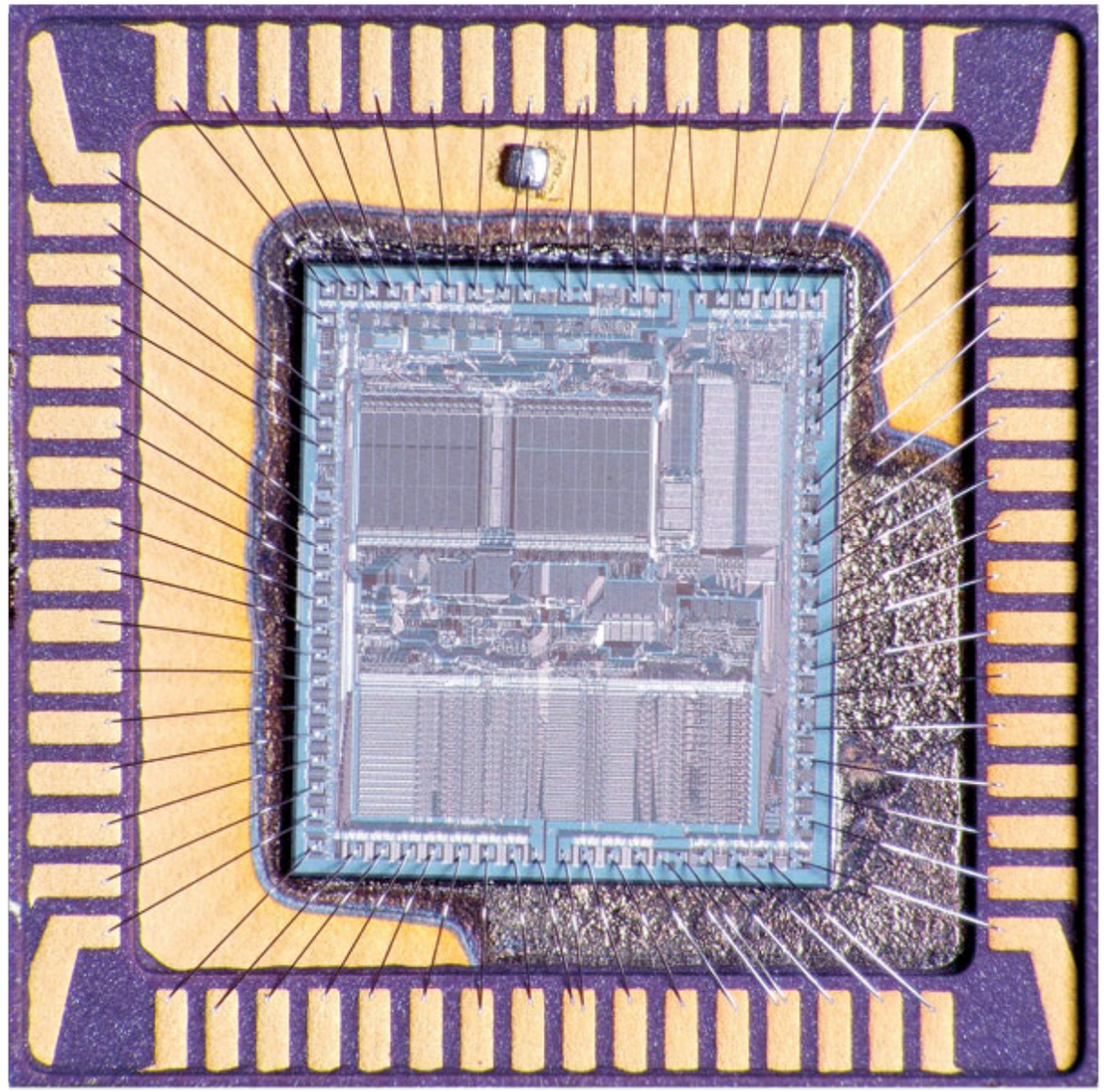
El µP 68000

El µP 68000

1. Características generales
2. Modos de direccionamiento
3. Repertorio de instrucciones
4. Técnicas de programación
5. Lenguaje ensamblador



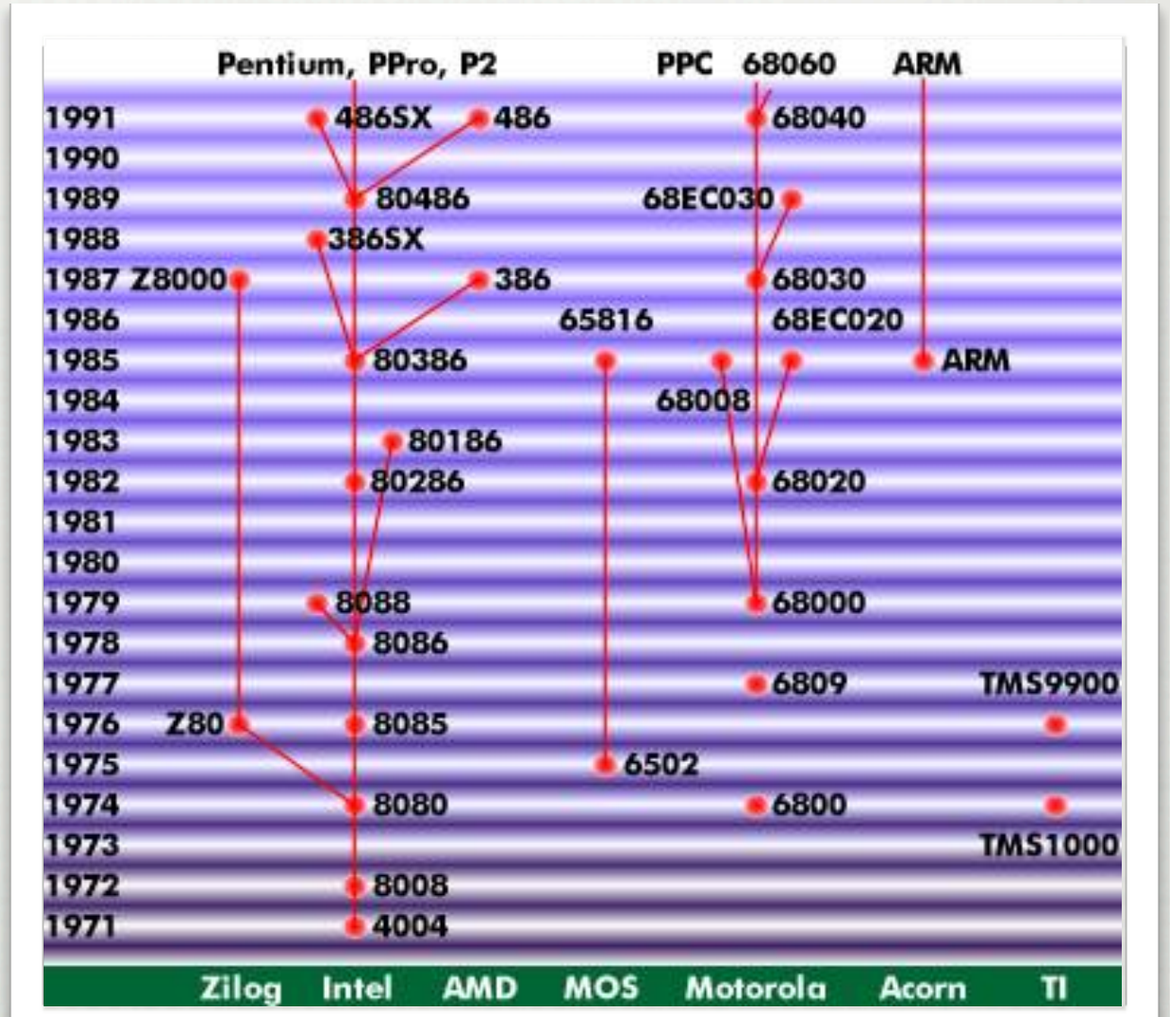
Características generales

Características generales

- Diseño VLSI, tecnología HMOS.
- Arquitectura interna de 32 bits.
- Bus de datos de 16 bits.
- Bus de direcciones de 23 bits (8 Mwords / 16 Mbytes).
- 16 registros de 32 bits (8 de datos y 8 de direcciones).
- 56 tipos de instrucciones.
- 14 modos de direccionamiento.
- Cinco tipos de datos.
- Dos modos de privilegio.
- Transferencia de datos de forma asíncrona.
- Interrupciones vectorizadas con siete niveles de prioridad.
- E/S mapeada en memoria.

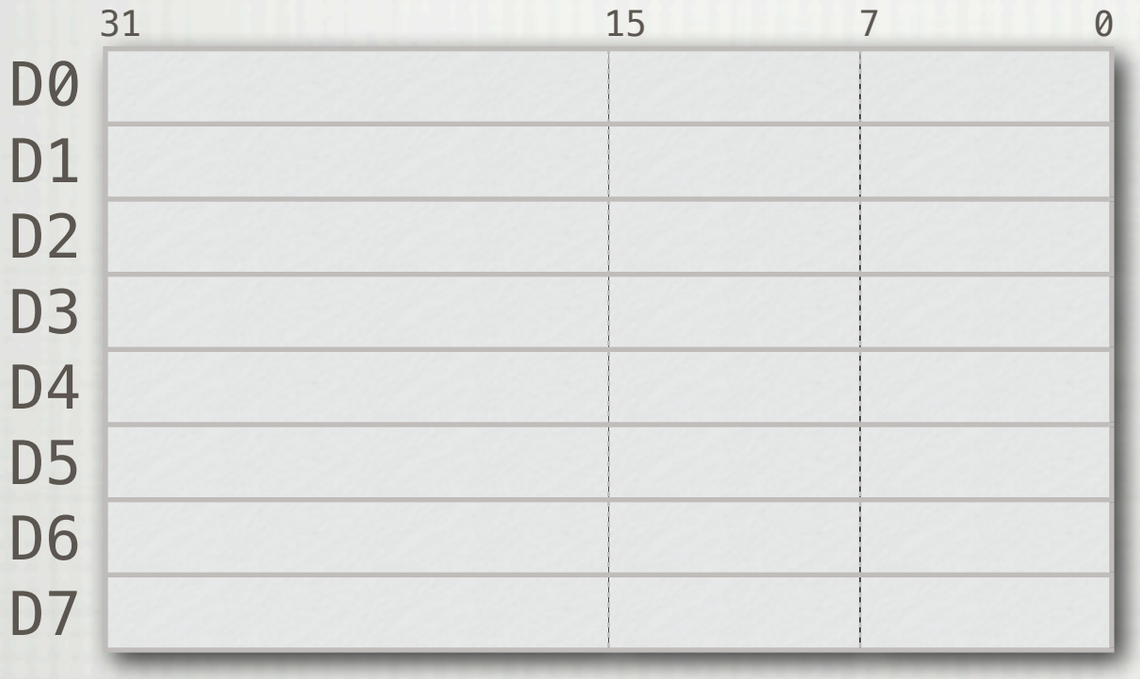
Historia y evolución

- Primer miembro de la familia 680X0, fruto del proyecto MACSS.
- Diseñado pensando en el futuro: compatibilidad hacia adelante.
- Utilizado en estaciones de trabajo, ordenadores personales, impresoras láser...



Modelo del programador

Registros de datos



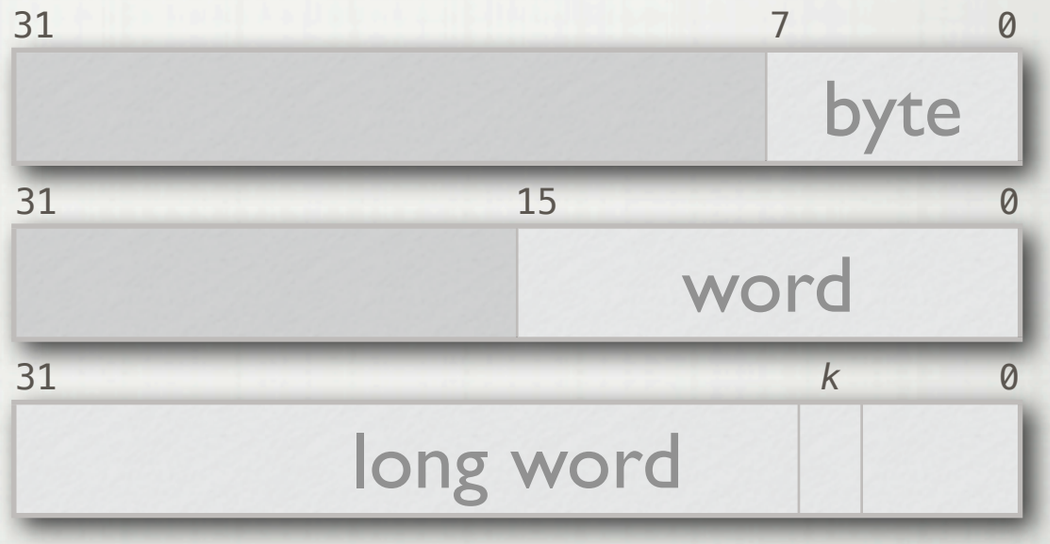
Registros de direcciones

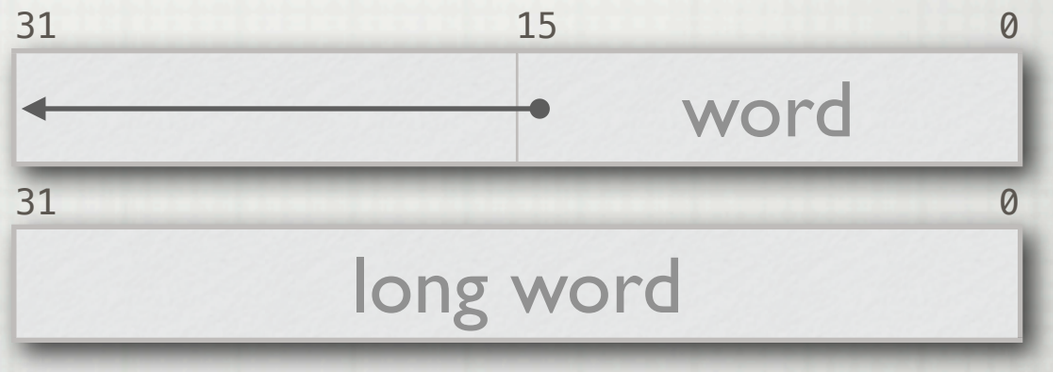
Tipos de datos

- Bits: bits individuales tanto en memoria como en registros de datos.
- Bytes: datos de 8 bits, en memoria y en registros de datos.
- Words: datos de 16 bits, en memoria, registros de datos y registros de direcciones.
- Long words: datos de 32 bits, en memoria, registros de datos y registros de direcciones.
- BCD: grupos de dos dígitos decimales de 4 bits cada uno.

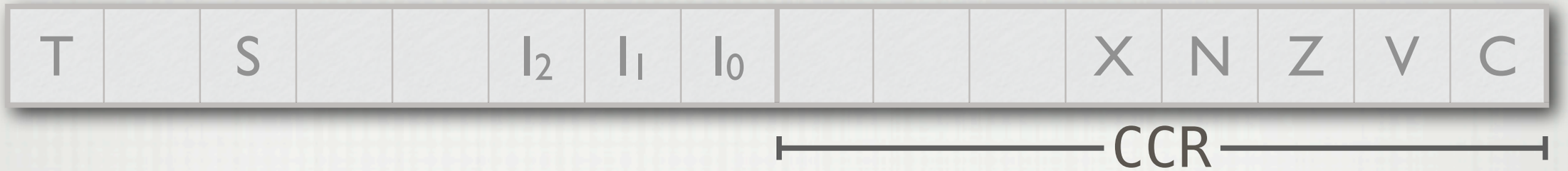
Registros de datos y direcciones

- Registros de datos (D0-D7):
  - Bits (0-31)
  - Bytes (8 bits más bajos)
  - Words (16 bits más bajos)
  - Long words (registro completo)
  - BCD (dos dígitos en los 8 bits más bajos)
- Registros de direcciones (A0-A7):
  - Words (16 bits más bajos) y long words (registro completo)
  - Al almacenar una word se le extiende su signo a 32 bits, afectando a todo el registro.





Registro de estado (SR)

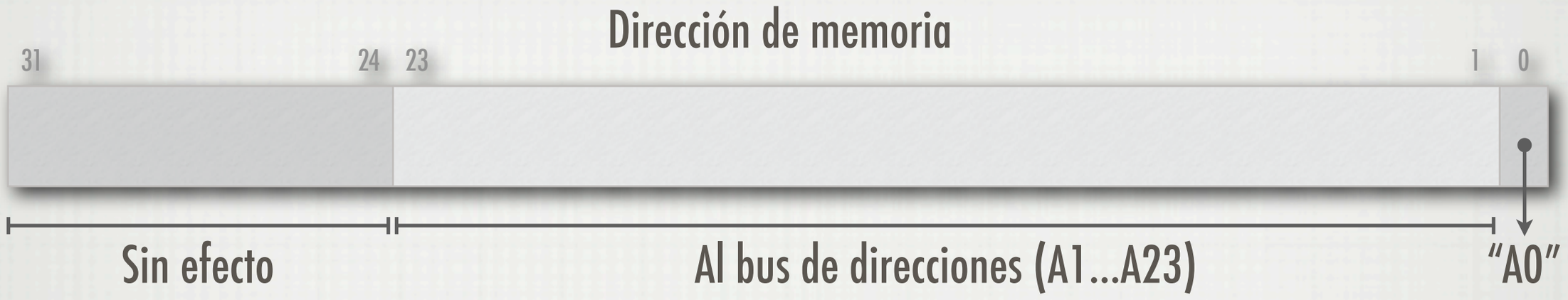


- C: Carry (llevada).
- V: Overflow (desbordamiento).
- Z: Zero (resultado cero).
- N: Negative (signo negativo).
- X: Extend (extensión).
- T: Trace (traza).
- S: Supervisor.
- I_2, I_1, I_0: prioridad frente a interrupciones (0 a 7).
- Byte alto sólo modificable en modo supervisor.

Espacio de direcciones

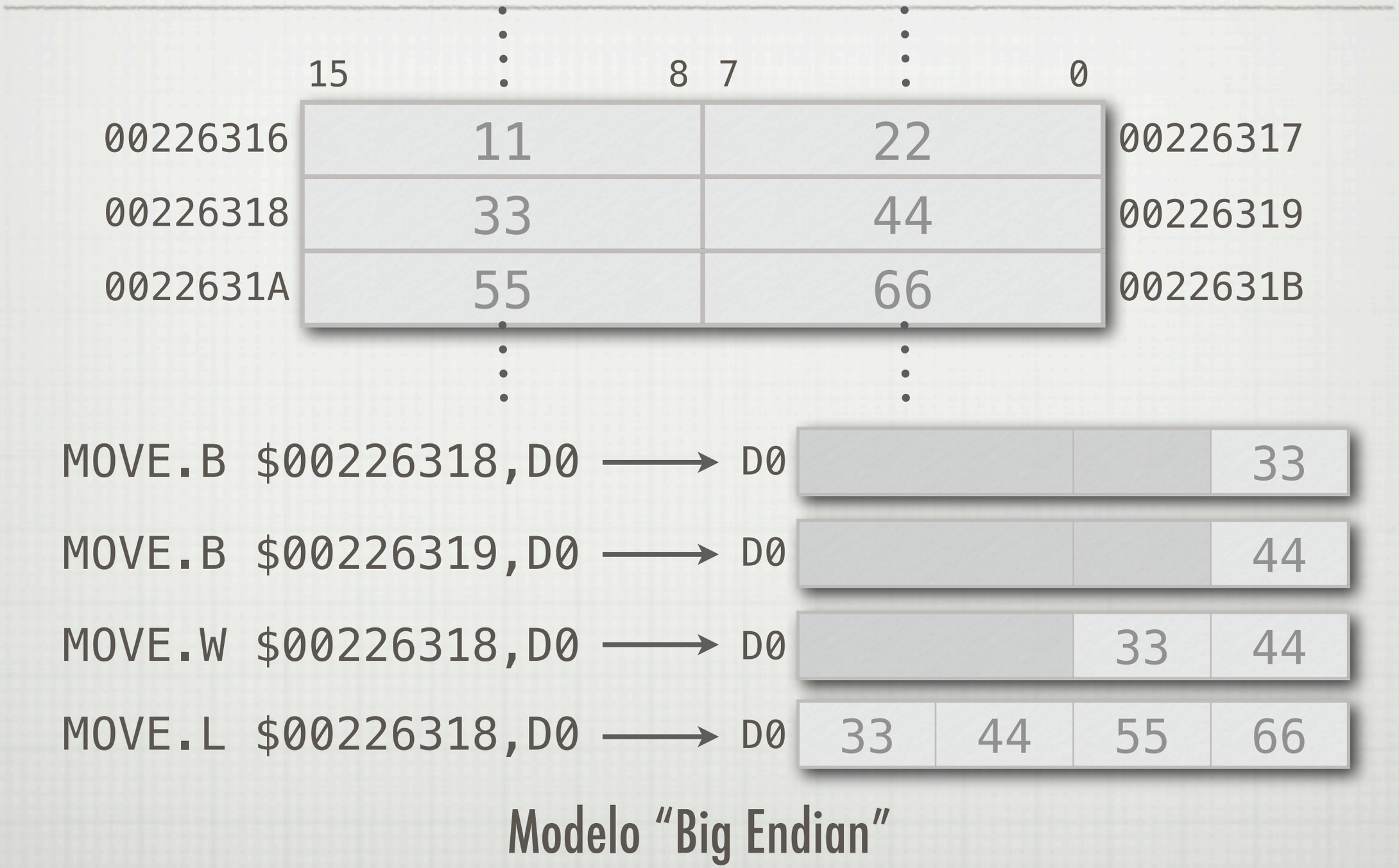
- Consta de 8 Mwords (2^{23}) o 16 Mbytes.
- Los dos bytes de cada word se pueden direccionar independientemente
  $\Rightarrow$ las direcciones que maneja el 68000 son de bytes.
- El 68000 puede leer o escribir:
  - Bytes (utilizando sólo la mitad superior o inferior del bus de datos).
  - Words (utilizando todo el bus de datos).
  - Long words (realizando dos ciclos de word consecutivos).

Espacio de direcciones



- Si se transfiere una word (o long word), "A0" debe ser cero.
- Si es un byte y "A0" es 0 (dirección par), se activa la línea $\overline{\text{UDS}}$ y la transferencia se realiza por las líneas D8...D15.
- Si es un byte y "A0" es 1 (dirección impar), se activa la línea $\overline{\text{LDS}}$ y la transferencia se realiza por las líneas D0...D7.

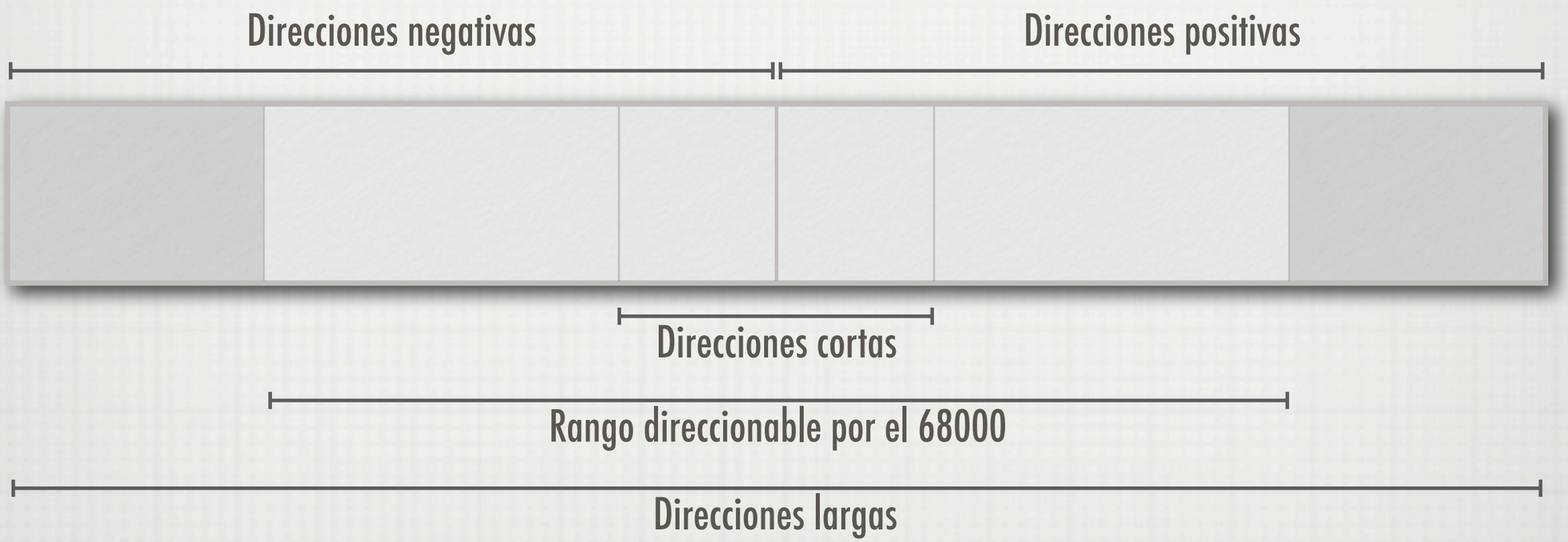
Espacio de direcciones



Modelo “Big Endian”

Espacio de direcciones

El 68000 emplea aritmética de 32 bits con signo para las direcciones:



Direcciones cortas: de `FFFF8000` a `FFFFFFFF`, y de `00000000` a `00007FFF` (16 bits)
Direcciones largas: de `80000000` a `FFFFFFFF`, y de `00000000` a `7FFFFFFF` (32 bits)
Direccionable por el 68000: de `FF800000` a `FFFFFFFF`, y de `00000000` a `007FFFFF` (24 bits)

Instrucciones y operandos

- Hay instrucciones sin operandos (pocas), con un solo operando (el destino), o con dos operandos (el origen y el destino).
- Ejemplo: instrucción CLR (“clear”):

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	1	0	tamaño		espec. op. destino					

CLR.W (A3) :

								word		modo=(An)			registro n=3		
0	1	0	0	0	0	1	0	0	1	0	1	0	0	1	1

CLR.B ($031A5C).L :

								word				modo=(XXX).L			
0	1	0	0	0	0	1	0	0	0	1	1	1	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
0	0	0	1	1	0	1	0	0	1	0	1	1	1	0	0

Modos de direccionamiento

Modos de direccionamiento

- Registro:
  - Datos
  - Direcciones
- Directo a memoria:
  - Absoluto corto
  - Absoluto largo.
- Indirecto a memoria:
  - Por registro de direcciones.
  - Con postincremento.
  - Con predecremento.
  - Con desplazamiento.
  - Con desplazamiento e índice.
- Relativo al contador de programa:
  - Con desplazamiento
  - Con desplazamiento e índice.
- Inmediato:
  - Inmediato normal.
  - Inmediato rápido.
- Implícito.

14 modos en total.

Modos de direccionamiento

- Clasificación de los modos de direccionamiento:
  - Datos: si se emplea para acceder a operandos con datos.
  - Memoria: si se emplea para acceder a operandos en memoria.
  - Alterable: si permite modificar el operando.
  - Control: si puede emplearse para acceder a datos sin que importe su tamaño.

Modos de direccionamiento

Modo de direccionamiento	Datos	Memoria	Alterable	Control
Registro de datos	X		X	
Registro de direcciones			X	
Indirecto por registro	X	X	X	X
Indirecto por registro con postincremento	X	X	X	
Indirecto por registro con predecremento	X	X	X	
Indirecto por registro con desplazamiento	X	X	X	X
Indirecto por registro con desplazamiento e índice	X	X	X	X
Absoluto corto	X	X	X	X
Absoluto largo	X	X	X	X
Relativo al PC con desplazamiento	X	X		X
Relativo al PC con desplazamiento e índice	X	X		X
Inmediato	X	X		

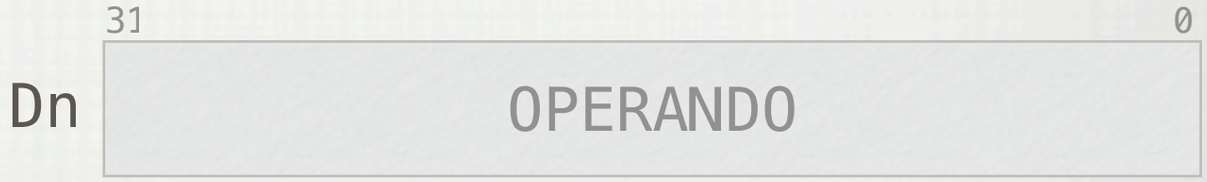
Modos de direccionamiento

- Concepto de dirección efectiva (effective address):
  - Consiste en el lugar donde se encuentra el operando.
  - Si está en memoria, es la dirección de memoria donde se halla.
    Si está en un registro, es ese registro.
  - Cada modo de direccionamiento define la manera en que se calcula la dirección efectiva del operando correspondiente.
  - Especificación con seis bits: tres para modo y tres para registro.

Directos a registro

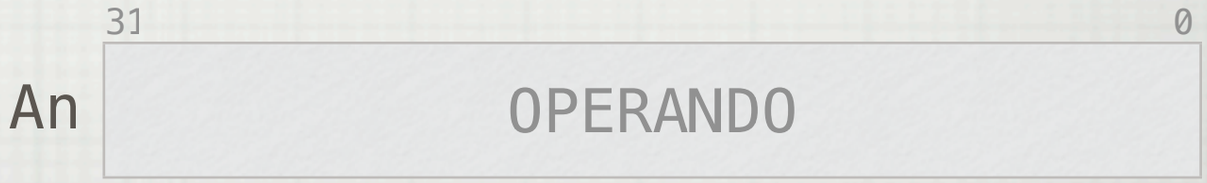
Registro de datos:

- Cálculo: EA = Dn
- Sintaxis: **`Dn`**
- Modo: 000
- Registro: n



Registro de direcciones:

- Cálculo: EA = An
- Sintaxis: **`An`**
- Modo: 001
- Registro: n



Indirectos a memoria por registro

Por registro de direcciones:

- Cálculo: EA = (An)
- Sintaxis: `(An)`
- Modo: 010
- Registro: n

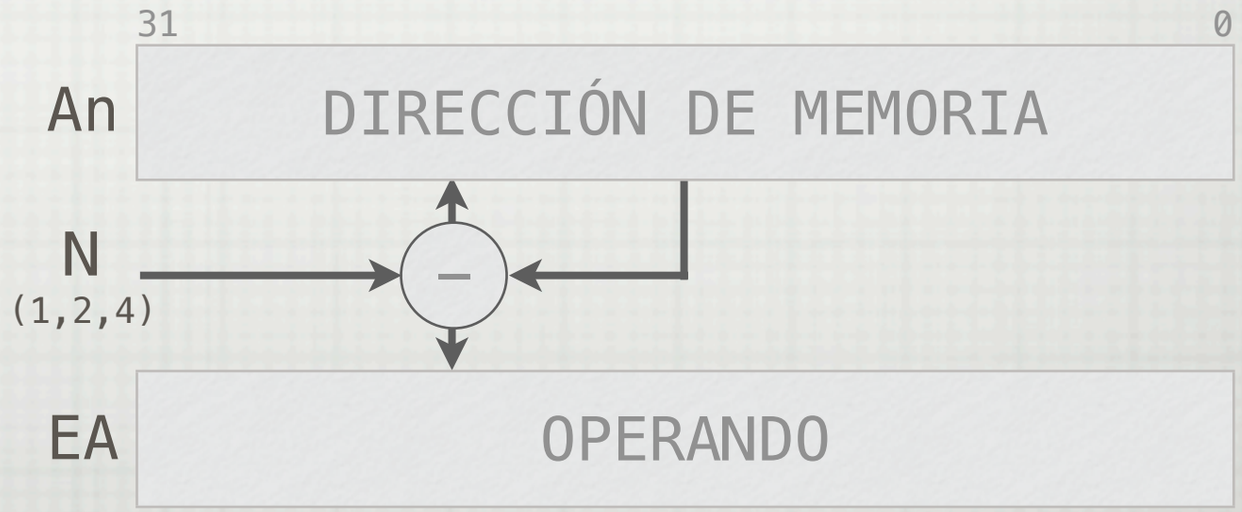
Indirectos a memoria por registro

Por registro de direcciones con postincremento:

- Cálculo: EA = (An); An←An+N
- Sintaxis: `(An)+`
- Modo: 011
- Registro: n

Por registro de direcciones con predecremento:

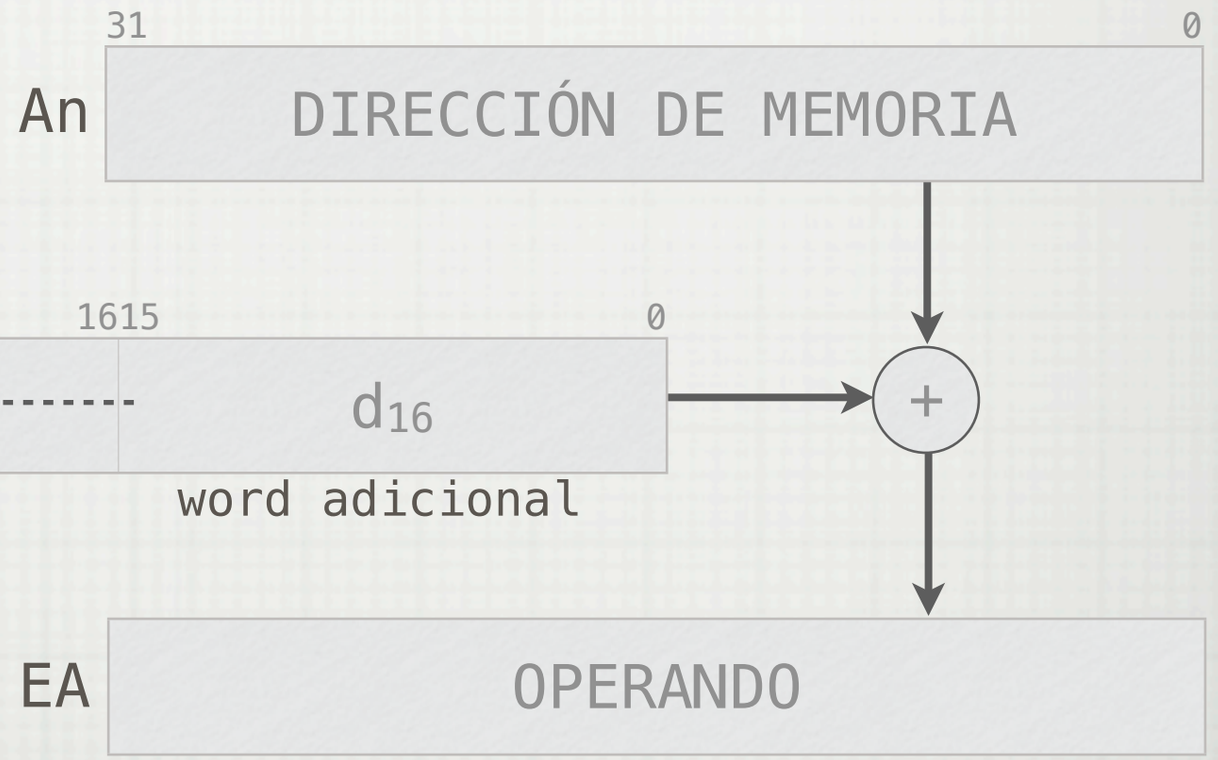
- Cálculo: An←An-N; EA = (An)
- Sintaxis: `-(An)`
- Modo: 100
- Registro: n



Indirectos a memoria por registro

Por registro de direcciones con desplazamiento:

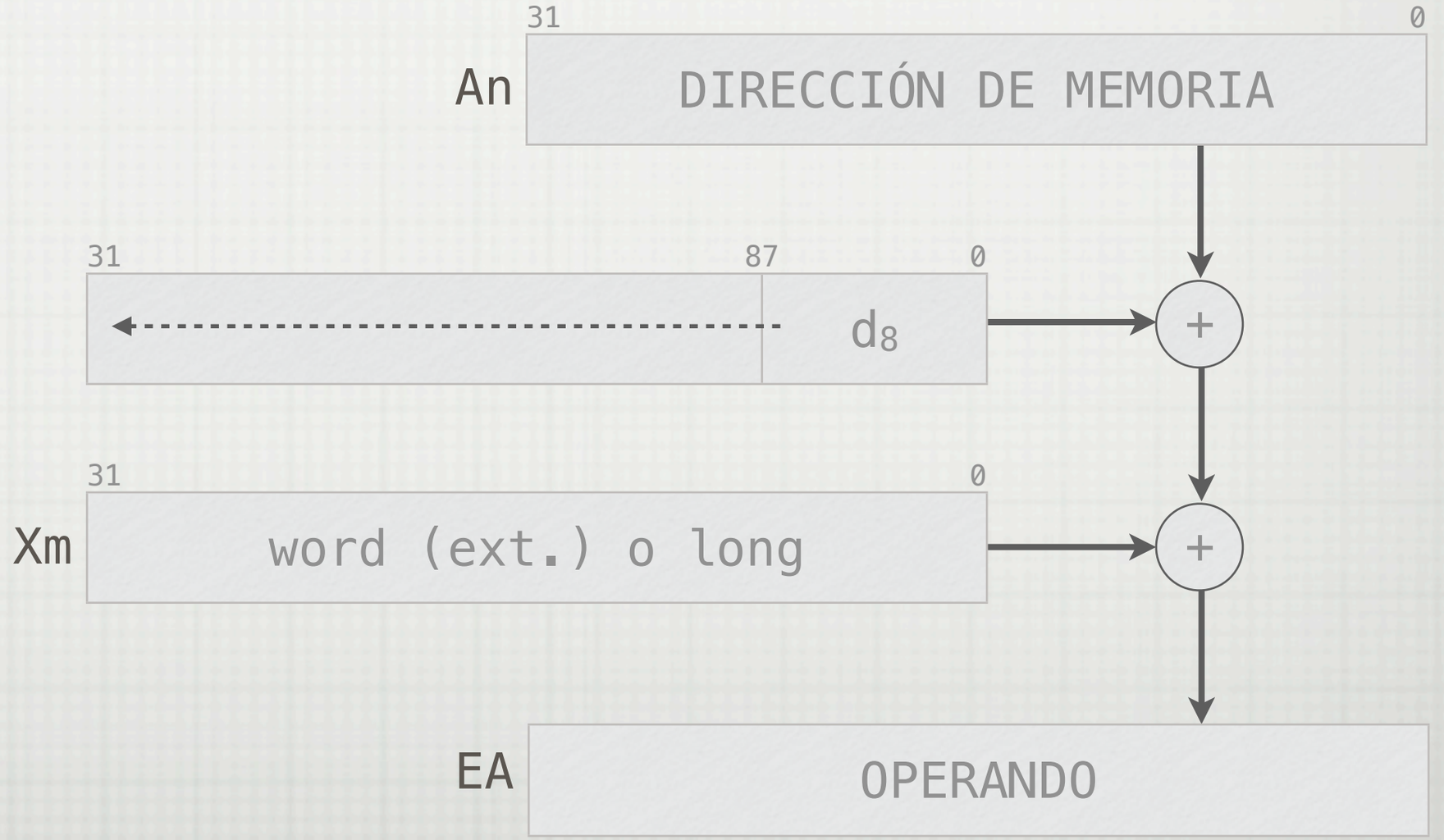
- Cálculo: EA = (An) + d_{16}
- Sintaxis: **d_{16}(An) o (d_{16},An)**
- Modo: 101
- Registro: n



Indirectos a memoria por registro

Por registro de direcciones con desplazamiento e índice:

- Cálculo: EA = (An) + (Xm) + d_8
- Sintaxis: **d_8(An,Xm.*S*) 0 (d_8,An,Xm.*S*)**
- Modo: 110
- Registro: n



Indirectos a memoria por registro

Formato de la word adicional de los modos de direccionamiento con desplazamiento e índice:

15	14 13 12	11	10	9	8	7 6 5 4 3 2 1 0
D/A	registro	W/L	0	0	0	desplazamiento

Bit 15: indicador del tipo de registro índice (0→datos, 1→direcciones)
Bits 14 a 12: número del registro índice (0 a 7)
Bit 11: tamaño del índice (0→word, 1→long)
Bits 10 a 8: siempre a 0
Bits 7 a 0: desplazamiento (número de 8 bits con signo)

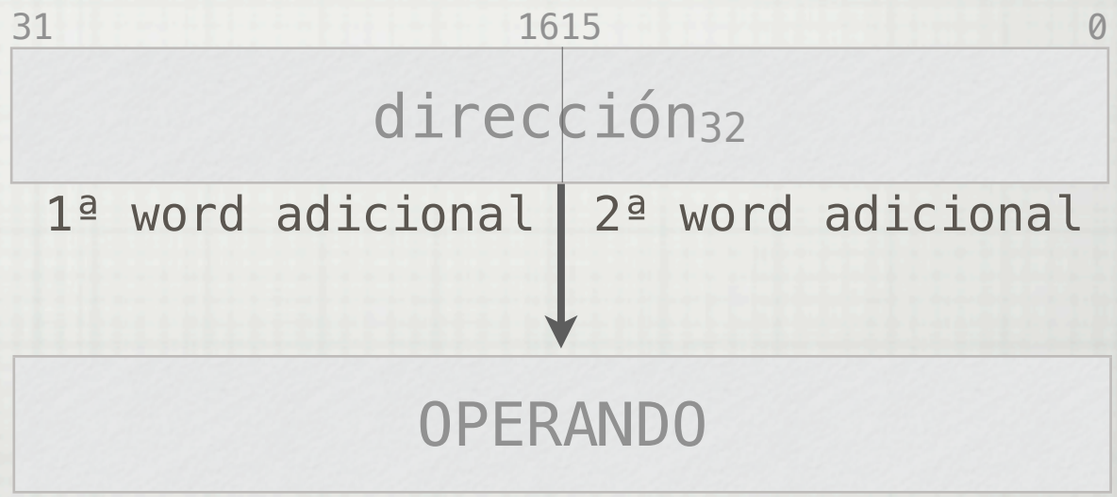
Directos a memoria

Absoluto corto

- Cálculo: se suministra la EA
- Sintaxis: `xxx.W o (xxx.W)`
- Modo: 111
- Registro: 000

Absoluto largo

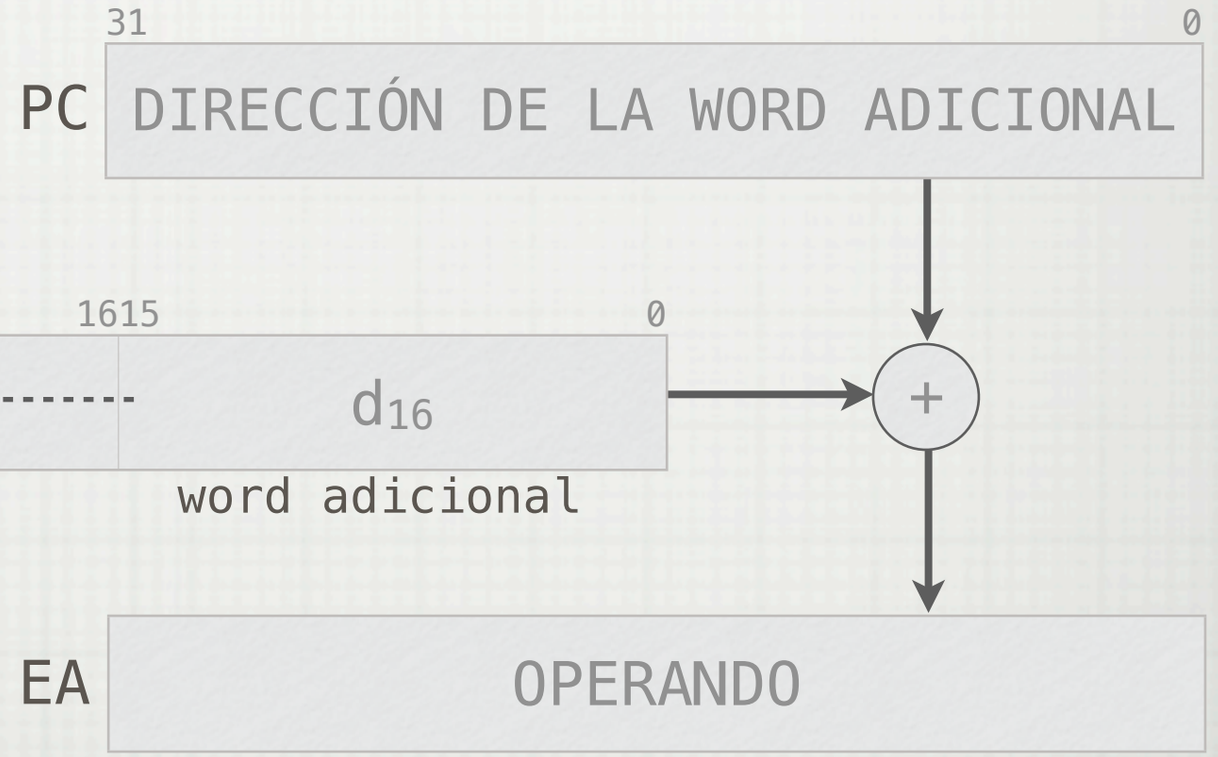
- Cálculo: se suministra la EA
- Sintaxis: `xxx.L o (xxx.L)`
- Modo: 111
- Registro: 001



Relativos al PC

Relativo al PC con desplazamiento

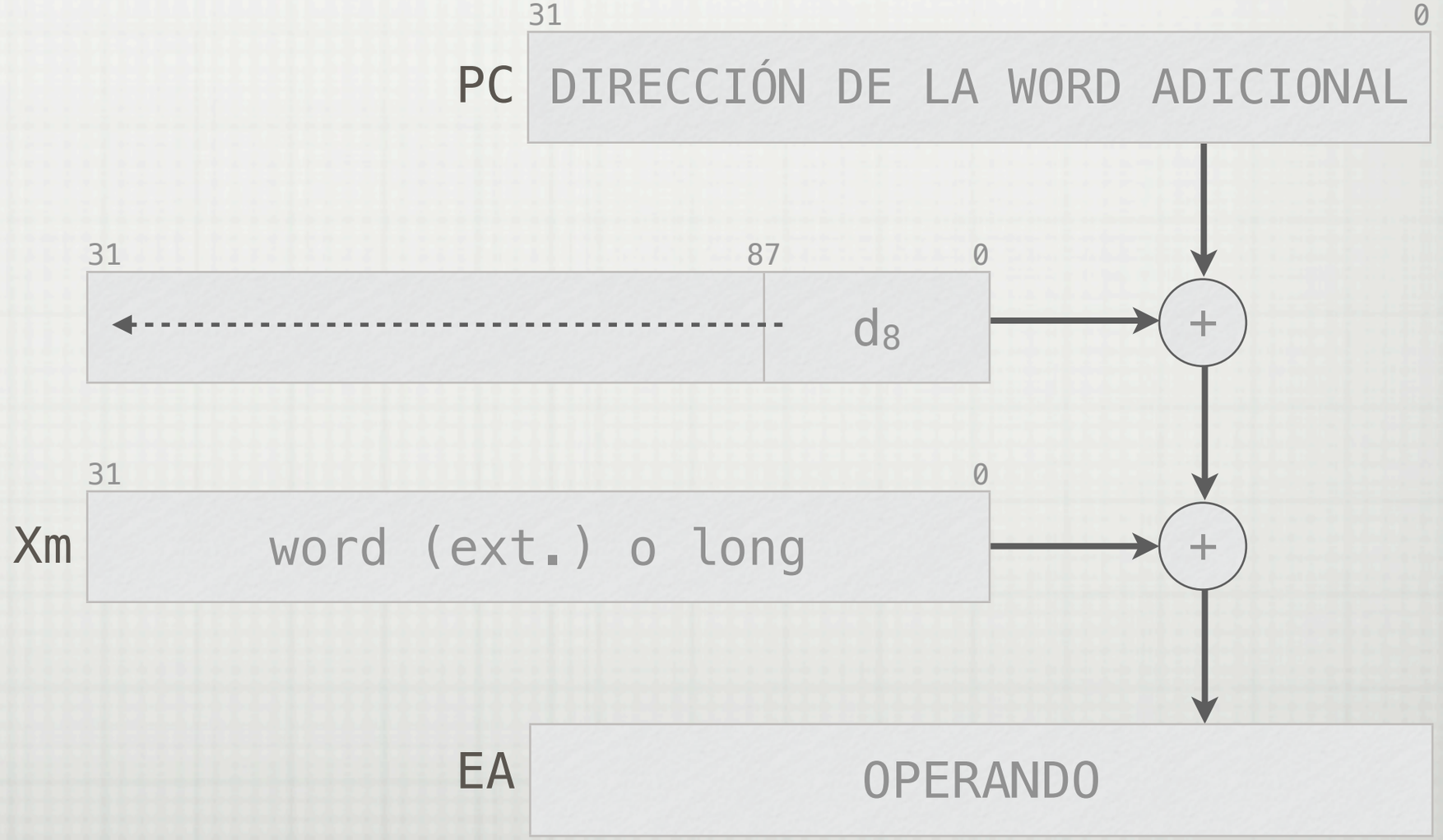
- Cálculo: EA = (PC) + d_{16}
- Sintaxis: **d_{16}(PC) 0 (d_{16},PC)**
- Modo: 111
- Registro: 010



Relativos al PC

Relativo al PC con desplazamiento e índice:

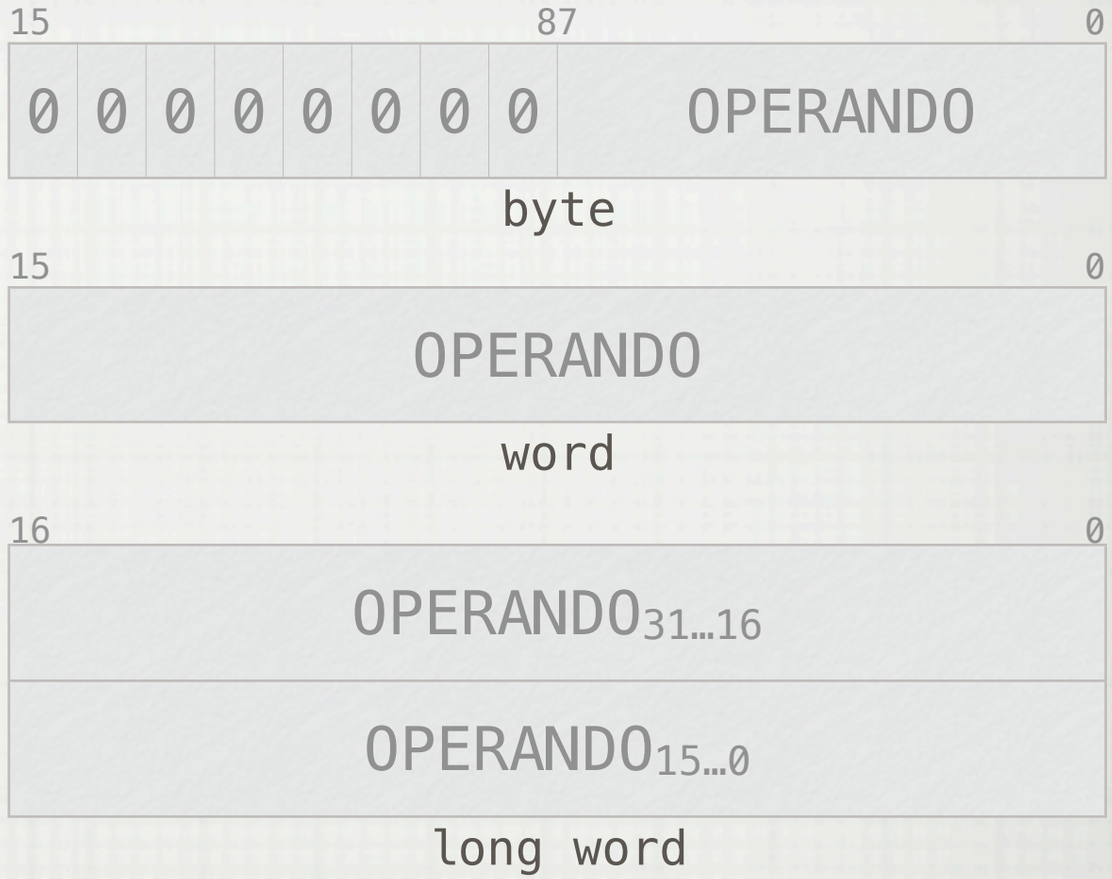
- Cálculo: EA = (PC) + (Xm) + d_8
- Sintaxis: **d_8(PC,Xm.*S*) 0 (d_8,PC,Xm.*S*)**
- Modo: 111
- Registro: 011



Inmediatos

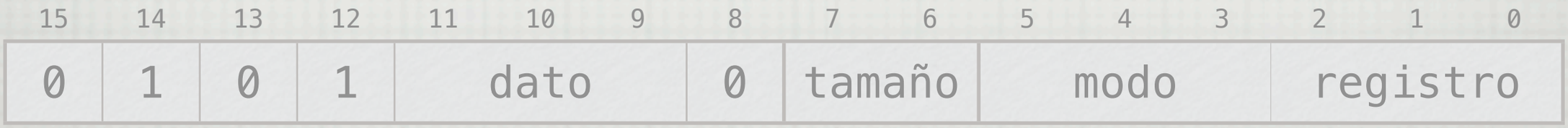
Inmediato:

- Cálculo: se suministra el operando
- Sintaxis: `#xxx`
- Modo: 111
- Registro: 100



Inmediato rápido:

- Cálculo: se suministra el operando
- Sintaxis: `#xx`
- Codificación dependiente de la instrucción (ejemplo: **`ADDQ #dato,<ea>`**)



Repertorio de instrucciones

Repertorio de instrucciones

- 56 tipos de instrucciones.
- Fruto de un análisis exhaustivo para determinar los tipos de instrucciones más utilizados, y las formas más frecuentes de hacerlo.

Movimiento de datos	Aritmética de enteros	Aritmética BCD
Operaciones lógicas	Desplazamiento y rotación	Manipulación de bits
Control de programa	Control del sistema	Comunicaciones multiprocesador

Movimiento de datos

Inst.	Sintaxis	Tam.	Restricciones	Operación	X	N	Z	V	C
MOVE	*ea*$_{org}$,*ea*$_{dst}$	BWL	*ea*$_{org}$=*,*ea*$_{dst}$=DA	origen→destino	-	*	*	0	0
MOVEA	*ea*,An	WL	—	origen→An	-	-	-	-	-
MOVEQ	#*d*$_8$,Dn	B→L	—	*d*$_8$→Dn (con ext. de signo)	-	*	*	0	0
MOVEM	*regs*,*ea* *ea*,*regs*	WL	*ea*=CA,-(An) *ea*=C,(An)+	*regs*→destino origen→*regs*	-	-	-	-	-
MOVEP	Dn,*d*$_{16}$*(An)* d$_{16}$(An),Dn	WL	—	Dn→mem. byte a byte mem.→Dn byte a byte	-	-	-	-	-
LEA	*ea*,An	L	*ea*=C	*ea*→An	-	-	-	-	-
PEA	*ea*	L	*ea*=C	A7-4→A7;*ea*→(A7)	-	-	-	-	-
EXG	*R*$_i$,*R*$_j$	L	R$_{i,j}$=Dn,An	*R*$_i$↔*R*$_j$	-	-	-	-	-
LINK	An,#*d*$_{16}$	L	—	A7-4→A7;An→(A7);A7→An;A7+*d*$_{16}$→A7	-	-	-	-	-
UNLK	An	L	—	An→A7;(A7)→An;A7+4→A7	-	-	-	-	-

Aritmética de enteros

Inst.	Sintaxis	Tam.	Restricciones	Operación	X	N	Z	V	C
ADD	*ea*,Dn Dn,*ea*	BWL	*ea*=* *ea*=MA	origen+destino→destino	*	*	*	*	*
ADDA	*ea*,An	WL	—	origen+destino→destino	-	-	-	-	-
ADDI	#*d*,*ea*	BWL	*ea*=DA	*d*+destino→destino	*	*	*	*	*
ADDQ	#d_3,*ea*	BWL	*ea*=A,d_3=1…8	d_3+destino→destino	*	*	*	*	*
ADDX	Dn,Dn -(An),-(An)	BWL	—	origen+destino+X→destino	*	*	*	*	*
SUB	*ea*,Dn Dn,*ea*	BWL	*ea*=* *ea*=MA	destino-origen→destino	*	*	*	*	*
SUBA	*ea*,An	WL	—	destino-origen→destino	-	-	-	-	-
SUBI	#*d*,*ea*	BWL	*ea*=DA	destino-*d*→destino	*	*	*	*	*
SUBQ	#d_3,*ea*	BWL	*ea*=A,d_3=1…8	destino-d_3→destino	*	*	*	*	*
SUBX	Dn,Dn -(An),-(An)	BWL	—	destino-origen-X→destino	*	*	*	*	*

Aritmética de enteros

Inst.	Sintaxis	Tam.	Restricciones	Operación	X	N	Z	V	C
NEG	*ea*	BWL	*ea*=DA	0-destino→destino	*	*	*	*	*
NEGX	*ea*	BWL	*ea*=DA	0-destino-X→destino	*	*	*	*	*
CMP	*ea*,Dn	BWL	—	destino-origen	-	*	*	*	*
CMPA	*ea*,An	WL	—	destino-origen	-	*	*	*	*
CMPI	*#d*,*ea*	BWL	*ea*=DA	destino-*d*	-	*	*	*	*
CMPM	(An)+,(An)+	BWL	—	destino-origen	-	*	*	*	*
MULU MULS	*ea*,Dn	WW→L	*ea*=D	origen*destino→destino	-	*	*	*	0
DIVU DIVS	*ea*,Dn	LW→W	*ea*=D	destino/origen→destino	-	*	*	*	0
EXT	Dn	WL	—	extensión de signo 8→16 o 16→32	-	*	*	0	0
CLR	*ea*	BWL	ea=DA	0→destino	-	0	1	0	0

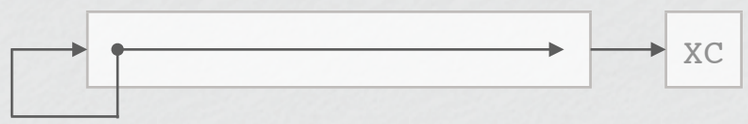
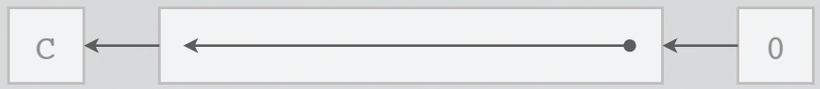
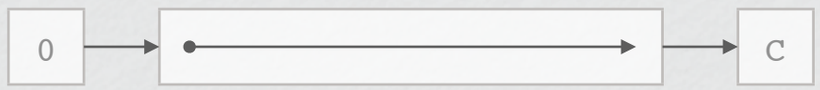
Aritmética BCD

Inst.	Sintaxis	Tam.	Restricciones	Operación	X N Z V C
ABCD	Dn,Dn -(An),-(An)	B	—	origen$_{10}$+destino$_{10}$+X→destino$_{10}$	* U * U *
SBCD	Dn,Dn -(An),-(An)	B	—	destino$_{10}$-origen$_{10}$-X→destino$_{10}$	* U * U *
NBCD	*ea*	B	*ea*=DA	0-destino$_{10}$-X→destino	* U * U *

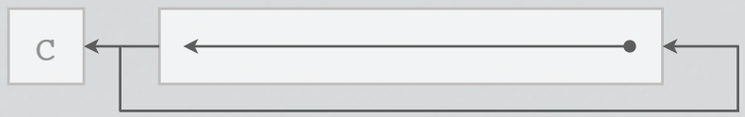
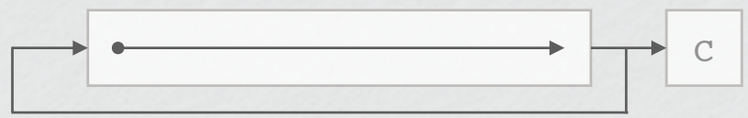
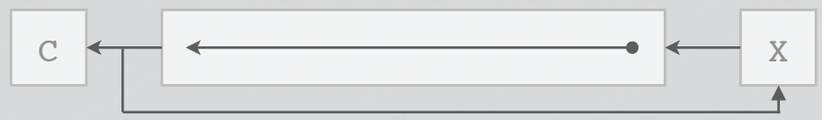
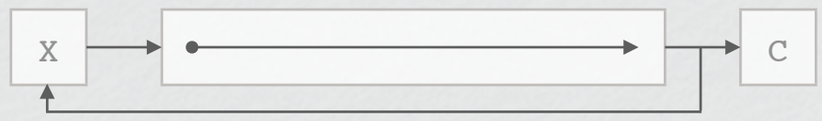
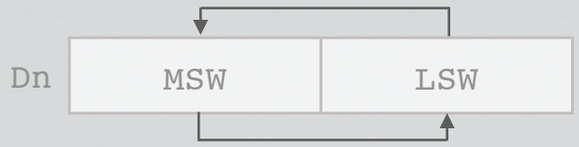
Operaciones lógicas

Inst.	Sintaxis	Tam.	Restricciones	Operación	X	N	Z	V	C
AND	*ea*,Dn Dn,*ea*	BWL	*ea*=D *ea*=MA	origen∧destino→destino	-	*	*	0	0
ANDI	*#d*,*ea*	BWL	*ea*=DA	*d*∧destino→destino	-	*	*	0	0
OR	*ea*,Dn Dn,*ea*	BWL	*ea*=D *ea*=MA	origen∨destino→destino	-	*	*	0	0
ORI	*#d*,*ea*	BWL	*ea*=DA	*d*∨destino→destino	-	*	*	0	0
EOR	Dn,*ea*	BWL	*ea*=DA	origen⊕destino→destino	-	*	*	0	0
EORI	*#d*,*ea*	BWL	*ea*=DA	*d*⊕destino→destino	-	*	*	0	0
NOT	*ea*	BWL	*ea*=DA	$\overline{\text{destino}}$→destino	-	*	*	0	0
TST	*ea*	BWL	*ea*=DA	Destino-0→destino	-	*	*	0	0

Desplazamiento y rotación

Inst.	Sintaxis	Tam.	Restricciones	Operación	X N Z V C
ASL	Dn,Dn #d_3,Dn *ea*	BWL BWL W	- d_3=1…8 *ea*=MA		* * * * *
ASR	Dn,Dn #d_3,Dn *ea*	BWL BWL W	- d_3=1…8 *ea*=MA		* * * * *
LSL	Dn,Dn #d_3,Dn *ea*	BWL BWL W	- d_3=1…8 *ea*=MA		- * * 0 *
LSR	Dn,Dn #d_3,Dn *ea*	BWL BWL W	- d_3=1…8 *ea*=MA		- * * 0 *

Desplazamiento y rotación

Inst.	Sintaxis	Tam.	Restricciones	Operación	X N Z V C
ROL	Dn,Dn #d_3,Dn *ea*	BWL BWL W	- d_3=1…8 *ea*=MA		- * * 0 *
ROR	Dn,Dn #d_3,Dn *ea*	BWL BWL W	- d_3=1…8 *ea*=MA		- * * 0 *
ROXL	Dn,Dn #d_3,Dn *ea*	BWL BWL W	- d_3=1…8 *ea*=MA		* * * 0 *
ROXR	Dn,Dn #d_3,Dn *ea*	BWL BWL W	- d_3=1…8 *ea*=MA		* * * 0 *
SWAP	Dn	W	-		- * * 0 0

Manipulación de bits

Inst.	Sintaxis	Tam.	Restricciones	Operación	X N Z V C
BTST	Dn,*ea* #d,*ea*	BL	*ea*=DA	$\overline{bit}$→Z	- - * - -
BSET	Dn,*ea* #d,*ea*	BL	*ea*=DA	$\overline{bit}$→Z;1→bit	- - * - -
BCLR	Dn,*ea* #d,*ea*	BL	*ea*=DA	$\overline{bit}$→Z;0→bit	- - * - -
BCHG	Dn,*ea* #d,*ea*	BL	*ea*=DA	$\overline{bit}$→Z→bit	- - * - -

Tamaño BYTE si el operando destino está en memoria.
Tamaño LONG si el operando destino está en un registro de datos.

Control de programa

Inst.	Sintaxis	Tam.	Restric.	Operación
		Condicionales		
B*cc*	*desp*	BW		Si *cc* ⇒ PC+*desp*→PC
DB*cc*	Dn,*desp*	W		Si $\overline{cc}$ ⇒ Dn-1→Dn; si Dn≠-1 ⇒ PC+desp→PC
S*cc*	*ea*	B	*ea*=DA	Si *cc* ⇒ 11111111_2→destino. Si $\overline{cc}$ ⇒ 0→destino
		Incondicionales		
BRA	*desp*	BW		PC+*desp*→PC
BSR	*desp*	BW		A7-4→A7;PC→(A7);PC+*desp*→PC
JMP	*ea*	-	*ea*=C	*ea*→PC
JSR	*ea*	-	*ea*=C	A7-4→A7;PC→(A7);*ea*→PC
NOP	-	-		-
		Retornos		
RTS	-	-		(A7)→PC;A7+4→A7
RTR	-	-		(A7)→CCR;A7+2→A7;(A7)→PC;A7+4→A7

Control de programa

cc	Significado	Comprobación
T*	True	1
F*	False	0
NE	Not Equal	Z=0
EQ	Equal	Z=1
CC/HS	Carry Clear/Higher or Same	C=0
CS/LO	Carry Set/Lower	C=1
HI	Higher	C=0 ∧ Z=0
LS	Lower or Same	C=1 ∨ Z=1
VC	Overflow Clear	V=0
VS	Overflow Set	V=1
PL	Plus	N=0
MI	Minus	N=1
GE	Greater or Equal	N=V
LT	Less Than	N≠V
GT	Greater Than	(N=V)∧(Z=0)
LE	Less or Equal	(N≠V)∨(Z=1)

* No disponible para la instrucción `Bcc`

Control del sistema

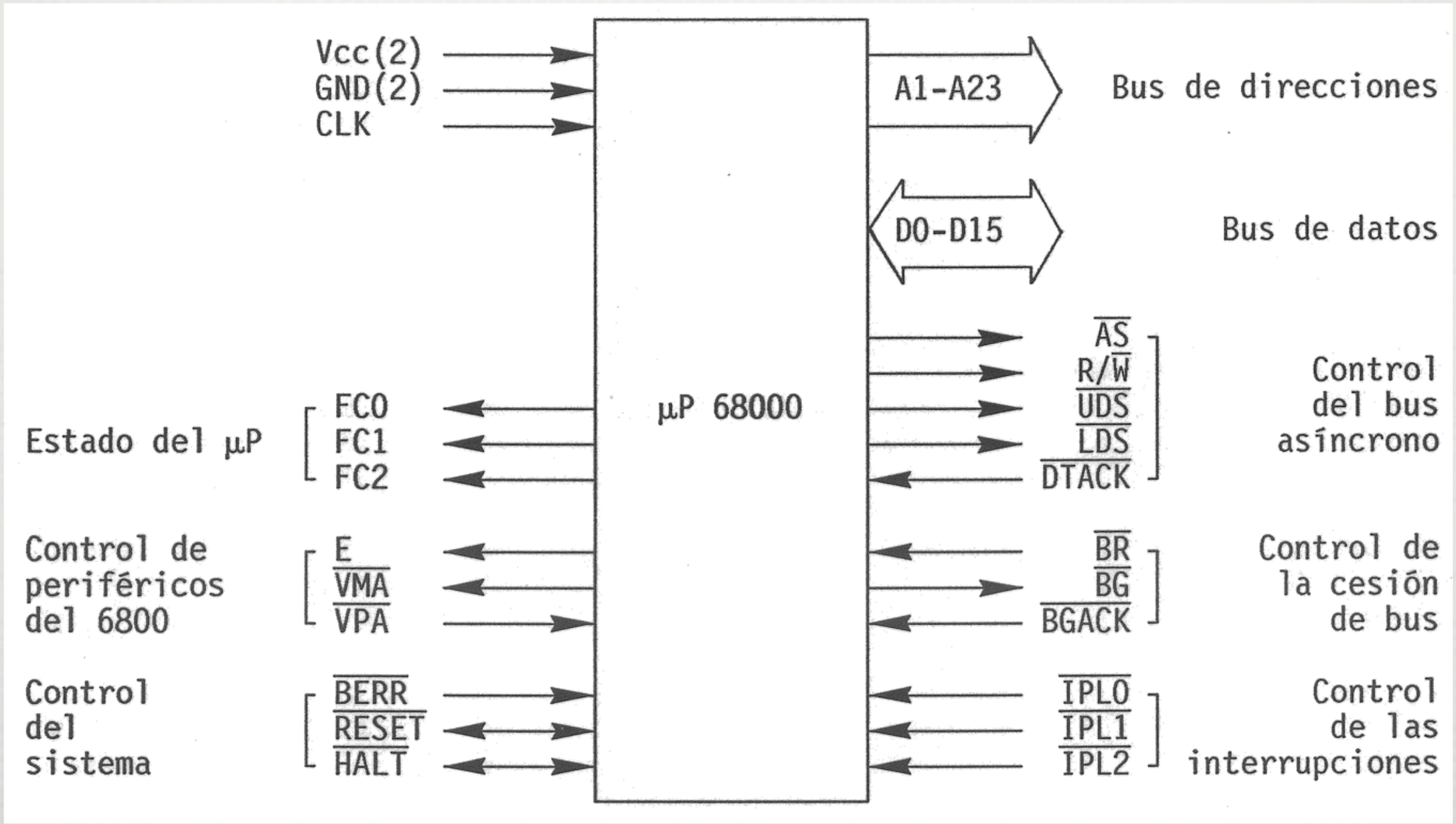
Inst.	Sintaxis	Tam.	Restricciones	Operación	X N Z V C
			Privilegiadas		
MOVE	<*ea*>,SR	W	*ea*=D	origen→SR	* * * * *
ANDI	#*dato*,SR	W	—	*dato*∧SR→SR	* * * * *
ORI	#*dato*,SR	W	—	*dato*∨SR→SR	* * * * *
EORI	#*dato*,SR	W	—	*dato*⊕SR→SR	* * * * *
MOVE	USP,An An,USP	L	—	USP→An An→USP	- - - - -
RESET	—	—	—	Activa la señal RESET	- - - - -
STOP	#*dato*$_{16}$	—	—	*dato*$_{16}$→SR; parar	* * * * *
RTE	—	—	—	(SSP)→SR;SSP+2→SSP; (SSP)→PC;SSP+4→SSP	* * * * *

Control del sistema

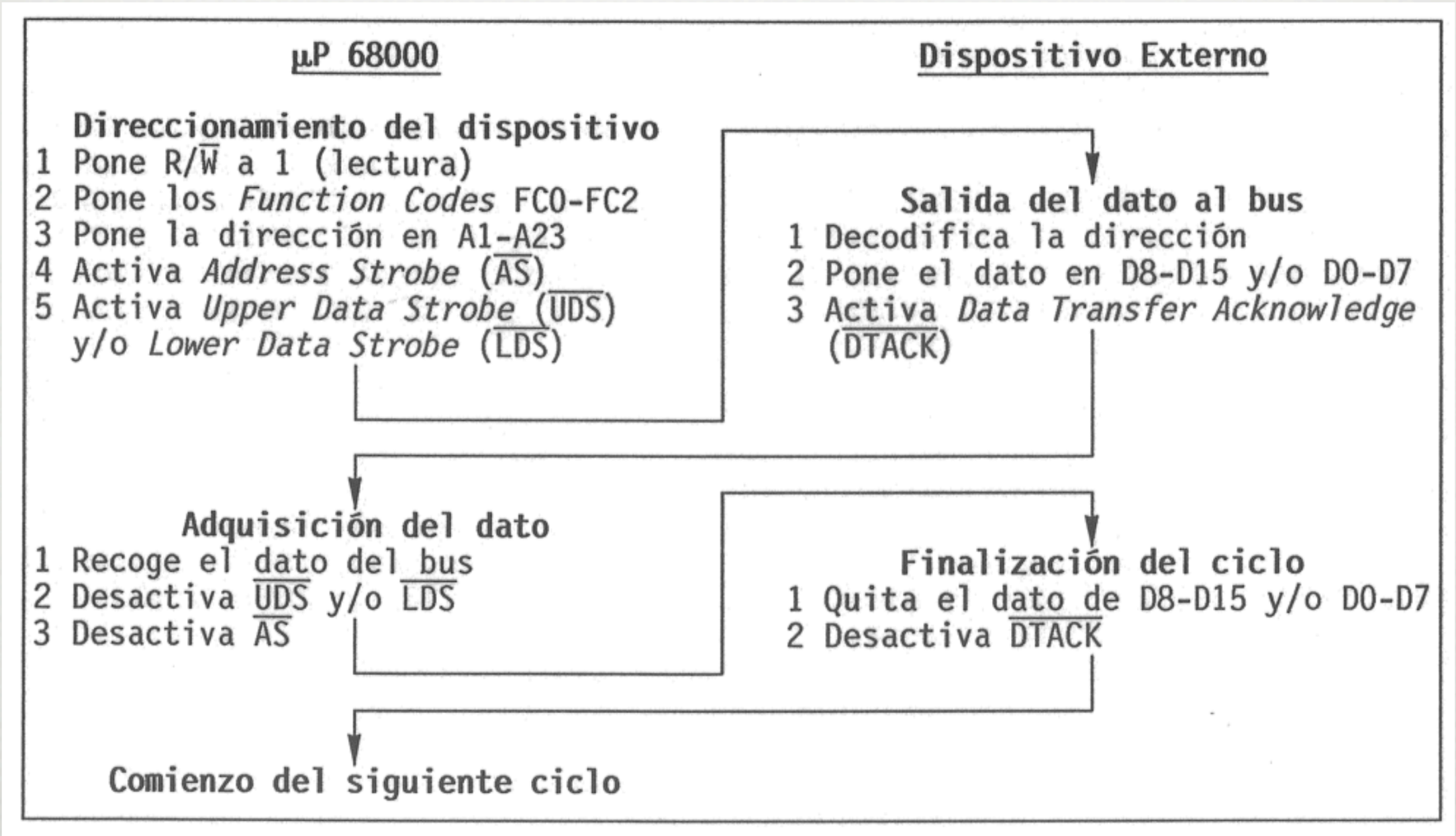
Inst.	Sintaxis	Tam.	Restricciones	Operación	X N Z V C
			Registro de condiciones		
MOVE	<*ea*>,CCR	W	*ea*=D	origen(LSB)→CCR	* * * * *
MOVE	SR,<*ea*>	W	*ea*=DA	SR→destino	- - - - -
ANDI	#*dato*,CCR	B	—	*dato*∧CCR→CCR	* * * * *
ORI	#*dato*,CCR	B	—	*dato*∨CCR→CCR	* * * * *
EORI	#*dato*,CCR	B	—	*dato*⊕CCR→CCR	* * * * *
			Generación de "traps"		
TRAP	#$dato_4$	—	—	Excepción TRAP 0…15	- - - - -
TRAPV	—	—	—	V=1 ⇒ excepción	- - - - -
CHK	<*ea*>,Dn	W	*ea*=D	Dn<0 o Dn>origen ⇒ excepción	- * U U U
			Comunicaciones multiprocesador		
TAS	<*ea*>	B	*ea*=DA	destino-0;1→bit 7 del destino	- * * 0 0

Buses y señales

Señales del 68000



Ciclo de lectura



Ciclo de escritura