

PEDRO DE MIGUEL AMASCOASTI

Catedrático de Arquitectura
y Tecnología de Computadores de la
Facultad de Informática de la
Universidad Politécnica de Madrid

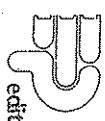
JOSÉ M. ANGULO USATEGUI

Catedrático de Arquitectura
y Tecnología de Computadores de la
Facultad de Informática de la
Universidad de Deusto

ARQUITECTURA

DE COMPUTADORES

Fundamentos e introducción al paralelismo



CUARTA EDICIÓN

1995

Índice de materias

Presentación

17

Introducción a la arquitectura de computadores

19

1.1. Definiciones	19
1.2. Estructura clásica del computador	21
1.3. Influencia de la tecnología en la evolución de la estructura básica de los computadores	23
1.3.1. Primera etapa	23
1.3.2. Segunda etapa	24
1.3.3. Tercera etapa	26
1.3.4. Cuarta etapa	28

1.4. Problemática en el desarrollo de la arquitectura de computadores

30

1.5. La historia de los computadores a través de sus generaciones	31
1.6. Primera generación (1938-1952)	32
1.7. Segunda generación (1953-1962)	35
1.8. Tercera generación (1963-1971)	36
1.9. Cuarta generación (1972-1987)	38
1.10. Quinta generación (1987)	40
1.10.1. Supercomputadores de alta velocidad	40
1.10.2. Computadores de funciones inteligentes	42

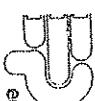
Ejercicios

43

2. Sistemas de representación de la información

45

2.1. Introducción	45
2.2. Capacidad de representación del computador	47
2.3. Tipo de representación	49
2.4. Datos alfanuméricos	50
2.5. Generalidades sobre los datos numéricos	54
2.6. Métodos posicionales	54



editorial Paraninfo S.A.

Magallanes, 25 - 28015 MADRID

(02)2515:

Gráficas ROGAR, Polígono Industrial Cubo Calleja - Fuentelatrada (Madrid)

2.7. Coma fija sin signo. Binario puro sin signo	56	3.6.2. Extensión de signo	94																																																																																						
2.7.1. Binario puro con signo.	58	3.6.3. Adición y sustracción	95																																																																																						
2.8. Coma fija con complemento restringido a la base. Complemento a uno	59	3.7. Operación de multiplicación	115																																																																																						
2.9. Coma fija con complemento a la base. Complemento a dos	61	3.7.1. Algoritmo de suma desplazamiento	116																																																																																						
2.10. Representación en exceso Z	63	3.7.2. Algoritmos de sumas Y restas	118																																																																																						
2.11. Formatos de representación decimal	64	3.7.3. Algoritmo de Booth	120																																																																																						
2.12. Coma flotante	66	3.7.4. Multiplicador rápido	121																																																																																						
2.13. Coma flotante con mantisa entera	69	3.8. Operación de división	122																																																																																						
2.14. Coma flotante con mantisa fracción. Normalización	69	3.8.1. División con restauración sin signo	122																																																																																						
2.15. Estándar IEEE P754	71	3.8.2. División sin restauración sin signo	124																																																																																						
2.16. Sistema de residuos	72	3.8.3. División con signo	125																																																																																						
2.17. Códigos de detección de errores	74	3.9. Unidades Lógico Aritméticas integradas	126																																																																																						
2.18. Códigos de paridad	74	3.9.1. Unidad Lógico Aritmética 74181	126																																																																																						
2.18.1. Paridad vertical simple o a carácter	74	3.9.2. Unidad Aritmética Am 29203	128																																																																																						
2.18.2. Paridad horizontal a nivel de bloque	74	Ejercicios	131																																																																																						
2.18.3. Códigos correctores de paridad entretejida	75																																																																																								
2.19. Claves de relación constante	76	Memorias																																																																																							
2.20. Códigos polinómicos	76	2.21. Detección y corrección automática de errores	78	4.1. Introducción	135	2.22. Representación de estructuras de datos	79	4.2. Evolución histórica	136	Ejercicios	81	4.3. Niveles de jerarquía	137	La unidad aritmética y lógica		4.4. Utilización práctica de las memorias	140	3.1. Introducción	85	4.5. Fundamentos básicos de las memorias	141	3.2. Tipos de operadores	86	4.5.1. Medio o soporte	142	3.2.1. Operadores generales y especializados	87	4.5.2. Transductor	143	3.2.2. Operadores combinacionales y secuenciales	87	4.5.3. Mecanismo de direccionamiento	143	3.2.3. Operadores paralelo y serie	88	4.6. Características de las memorias	147	3.2.4. Operadores MOS Y bipolar	88	3.3. Las operaciones de la Unidad Aritmética y Lógica	88	4.6.1. Duración de la información	147	3.4. Operaciones de desplazamiento	89	4.6.2. Modo de acceso	148	3.4.1. Desplazamientos lógicos	90	4.6.3. Velocidad	149	3.4.2. Desplazamientos circulares	91	4.6.4. Capacidad o tamaño	149	3.4.3. Desplazamientos aritméticos	91	4.7. Tipos de dispositivos de almacenamiento para memoria principal	150	3.4.4. Desplazamientos concatenados	92	3.5. Operaciones lógicas	92	4.8. Memorias de ferrita	151	3.6. Operaciones aritméticas	93	4.9. Memorias de película delgada y de h.o plateado	152	3.6.1. Cambio de signo	93	4.10. Memorias de semiconductores	153	4.11. Líneas de retardo	161	4.10.1. RAM estáticas	154			4.10.2. RAM dinámicas	156			4.10.3. Memorias ROM Y PROM	157			4.10.4. EPROM Y EEPROM	159
2.21. Detección y corrección automática de errores	78	4.1. Introducción	135																																																																																						
2.22. Representación de estructuras de datos	79	4.2. Evolución histórica	136																																																																																						
Ejercicios	81	4.3. Niveles de jerarquía	137																																																																																						
La unidad aritmética y lógica		4.4. Utilización práctica de las memorias	140																																																																																						
3.1. Introducción	85	4.5. Fundamentos básicos de las memorias	141																																																																																						
3.2. Tipos de operadores	86	4.5.1. Medio o soporte	142																																																																																						
3.2.1. Operadores generales y especializados	87	4.5.2. Transductor	143																																																																																						
3.2.2. Operadores combinacionales y secuenciales	87	4.5.3. Mecanismo de direccionamiento	143																																																																																						
3.2.3. Operadores paralelo y serie	88	4.6. Características de las memorias	147																																																																																						
3.2.4. Operadores MOS Y bipolar	88	3.3. Las operaciones de la Unidad Aritmética y Lógica	88	4.6.1. Duración de la información	147	3.4. Operaciones de desplazamiento	89	4.6.2. Modo de acceso	148	3.4.1. Desplazamientos lógicos	90	4.6.3. Velocidad	149	3.4.2. Desplazamientos circulares	91	4.6.4. Capacidad o tamaño	149	3.4.3. Desplazamientos aritméticos	91	4.7. Tipos de dispositivos de almacenamiento para memoria principal	150	3.4.4. Desplazamientos concatenados	92	3.5. Operaciones lógicas	92	4.8. Memorias de ferrita	151	3.6. Operaciones aritméticas	93	4.9. Memorias de película delgada y de h.o plateado	152	3.6.1. Cambio de signo	93	4.10. Memorias de semiconductores	153	4.11. Líneas de retardo	161	4.10.1. RAM estáticas	154			4.10.2. RAM dinámicas	156			4.10.3. Memorias ROM Y PROM	157			4.10.4. EPROM Y EEPROM	159																																						
3.3. Las operaciones de la Unidad Aritmética y Lógica	88	4.6.1. Duración de la información	147																																																																																						
3.4. Operaciones de desplazamiento	89	4.6.2. Modo de acceso	148																																																																																						
3.4.1. Desplazamientos lógicos	90	4.6.3. Velocidad	149																																																																																						
3.4.2. Desplazamientos circulares	91	4.6.4. Capacidad o tamaño	149																																																																																						
3.4.3. Desplazamientos aritméticos	91	4.7. Tipos de dispositivos de almacenamiento para memoria principal	150																																																																																						
3.4.4. Desplazamientos concatenados	92	3.5. Operaciones lógicas	92	4.8. Memorias de ferrita	151	3.6. Operaciones aritméticas	93	4.9. Memorias de película delgada y de h.o plateado	152	3.6.1. Cambio de signo	93	4.10. Memorias de semiconductores	153	4.11. Líneas de retardo	161	4.10.1. RAM estáticas	154			4.10.2. RAM dinámicas	156			4.10.3. Memorias ROM Y PROM	157			4.10.4. EPROM Y EEPROM	159																																																												
3.5. Operaciones lógicas	92	4.8. Memorias de ferrita	151																																																																																						
3.6. Operaciones aritméticas	93	4.9. Memorias de película delgada y de h.o plateado	152																																																																																						
3.6.1. Cambio de signo	93	4.10. Memorias de semiconductores	153																																																																																						
4.11. Líneas de retardo	161	4.10.1. RAM estáticas	154																																																																																						
		4.10.2. RAM dinámicas	156																																																																																						
		4.10.3. Memorias ROM Y PROM	157																																																																																						
		4.10.4. EPROM Y EEPROM	159																																																																																						

4.12. Tambor de silicio	162
4.13. Principales recursos para mejorar las prestaciones de la memoria principal	162
4.14. Memoria virtual	163
4.15. Memoria paginada	166
4.15.1. Método de correspondencia directa	167
4.15.2. Método de correspondencia asociativa	168
4.16. Memoria segmentada	169
4.17. Memoria de segmentos paginados	171
4.18. Memoria cache o inmediata	175
4.19. Tipos de memoria cache	176
4.19.1. Memorias cache de correspondencia directa	177
4.19.2. Memoria asociativa completa	177
4.19.3. Memoria cache de asociación de conjuntos	177
4.19.4. Memoria cache de correspondencia vectorizada	179
4.20. Protección de la memoria principal	179
4.20.1. Registros de borde	179
4.20.2. Círculo Y clave	181
4.20.3. Instrucción "TEST AND SET"	181
Ejercicios	182
Modos de direccionamiento y repertorio de instrucciones	
5.1. Las máquinas de programa almacenado	186
5.2. Características del lenguaje máquina	188
5.3. Modos de direccionamiento	189
5.4. Direccionamiento inmediato	190
5.5. Direccionamiento directo absoluto	191
5.6. Direccionamiento directo relativo	193
5.6.1. Direccionamiento relativo al Contador de Programa	194
5.6.2. Direccionamiento relativo a registro base	194
5.6.3. Direccionamiento relativo a registro índice	195
5.6.4. Direccionamiento a pila	196
5.7. Direccionamiento indirecto	197
5.8. Direccionamiento implícito	198
5.9. Ejemplos reales de modos de direccionamiento	198
5.9.1. IEEE P 694	198
5.9.2. Microprocesador 68000	199
5.9.3. VAX-11	199
5.10. Características y tipos de instrucciones	201
5.11. Instrucciones de movimiento o transferencia de datos	202

5.12. Instrucciones de ruptura de secuencia	203
5.12.1. Bifurcaciones condicionales	203
5.12.2. Bifurcaciones con retorno	204
5.13. Instrucciones aritméticas	206
5.14. Instrucciones de comparación	208
5.15. Instrucciones lógicas	208
5.16. Instrucciones de desplazamiento	208
5.17. Instrucciones de bit	208
5.18. Instrucciones de entrada y salida diversas	209
5.19. Formato de instrucciones	210
5.20. Características del formato de las instrucciones	211
5.21. Frecuencia de utilización de las instrucciones	216
5.22. El Ensamblador	218
Ejercicios	221
La unidad de control	
6.1. Misión de la Unidad de Control en el computador	226
6.1.1. La memoria principal	227
6.1.2. La Unidad Operativa	228
6.1.3. La Unidad de Control	229
6.2. Operaciones elementales y microinstrucciones	231
6.3. Señales de control en un computador elemental	233
6.3.1. Memoria principal	235
6.3.2. Unidad Aritmética	236
6.3.3. Banco de registros	236
6.3.4. Unidad de Control	237
6.4. Temporización de las señales de control. Periodos y fases	237
6.5. Cronogramas en la ejecución de instrucciones	238
6.5.1. Instrucción ADD 4,7	238
6.6. Diseño de la Unidad de Control	241
6.6.1. Unidad de Control en lógica cableada	242
6.6.2. Unidad de Control microprogramada	242
6.7. Unidad de Control microprogramada	242
6.8. Unidad de Control microprogramada	242
6.8.1. "Concepto de microprograma"	243
6.9. Características de la Unidad de Control microprogramada	245
6.9.1. Secuenciamiento explícito	245
6.9.2. Secuenciamiento implícito	246
6.10. Formato de las microinstrucciones	247
6.11. Estructura completa de la Unidad de Control microprogramada	250
6.11.1. Microbifurcaciones condicionales	250

6.11.2. Microbucleos y microsubrutinas	252
6.12. Empleo de varios relojes	252
6.13. Rupturas de secuencia no programada, interrupciones y excepciones o cebos	253

Apéndice: Diseño de una UCP con circuitos integrados **SSI** y **MSI**

6A.1. La orientación didáctica del proyecto	254
6A.2. Funcionamiento básico	255
6A.3. Bloque Aritmético-Lógico	257
6A.4. El Contador de Programa	261
6A.5. El Secuenciador	262
6A.6. Cronogramas de funcionamiento	265
6A.7. Modos de direccionamiento y repertorio de instrucciones	267
6A.7.1. CPL	267
6A.7.2. SEC	268
6A.7.3. CLC	268
6A.7.4. NOP	268
6A.7.5. JC	268
6A.7.6. JZ	269
6A.7.7. SBC	269
6A.7.8. JMP	270
6A.7.9. LDA	270
6A.7.10. ADC	270
6A.7.11. STA	271
6A.7.12. AND	271
6A.7.13. STB	271
6A.7.14. LDA #	271
6A.7.15. OR	272
6A.7.16. MOV A,B	272
6A.8. Diagramas de conexión de los circuitos integrados empleados en la construcción de la UCP	275
Ejercicios	278

Unidad de entrada y salida

7.1. Generalidades sobre el intercambio de información con el exterior	281
7.2. Comunicación física entre la UCP y los periféricos	282
7.3. Entradas y salidas programadas	283
7.3.1. Señales de dirección	283
7.4. Mapa de memoria común	287
7.5. Señales de datos	287

Buses

8.1. Introducción	332
8.2. Niveles de especificación de un bus	334
8.2.1. Nivel mecánico	334
8.2.2. Nivel eléctrico	334
8.2.3. Nivel lógico	334
8.2.4. Nivel de temporización básica	335
8.2.5. Nivel de transferencia elemental	335
8.2.6. Nivel de transferencia de bloque	335

7.6. Señales de control	289
7.6.1. Transferencias síncrona	289
7.6.2. Transferencias asíncrona o con interbloqueo (diálogo)	290
7.7. Acceso directo a memoria	292
7.7.1. Memoria multipuerta	292
7.7.2. Robo de ciclo	293
7.8. Control de los periféricos	297
7.9. Prioridades	298
7.9.1. Gestión distribuida de prioridades	299
7.9.2. Gestión de prioridad centralizada de Entrada/Salida	302
7.10. Descripción detallada del funcionamiento de una operación	303
7.11. Interrupciones	308
7.12. Línea de interrupción única	309
7.13. Líneas de interrupción y aceptación	310
7.14. Interrupciones vectorizadas	311
7.15. Prioridades y niveles de interrupción	312
7.16. Organización de las operaciones de E/S	315
7.17. Entrada/fálida programada	316
7.18. Mediante interrupciones	317
7.19. Interrupciones con controlador inteligente	318
7.20. Robo de ciclo	318
7.21. Canales de E/S	320
7.22. El sistema operativo y las Entradas y Salidas	322
7.23. Circuitos integrados para el diseño de E/S	323
7.24. Ejemplos de organización de E/S en máquinas comerciales	323
7.24.1. PDP-11 y VAX-11	323
7.24.2. IBM 370	327
Ejercicios	330

8.3.	Buses normalizados	335
8.4.	Paralelismo del bus	336
8.5.	Bus de ciclo partido	338
8.6.	Transferencias síncronas y asíncronas	339
8.6.1.	Transferencia en ciclo completo	339
8.6.2.	Transferencia en ciclo partido	340
8.7.	Control del bus	341
8.8.	Largo del bus	342
8.9.	Jerarquía de buses	343
8.9.1.	Buses tipo 0	343
8.9.2.	Buses tipo 1	343
8.9.3.	Buses tipo 2	344
8.9.4.	Buses tipo 3	344
8.9.5.	Buses tipo 4	345
8.9.6.	Buses tipo 5	346
8.10.	Detección y tratamiento de errores	348
Apéndice: La norma de conexión RS-232-C		
8A.1.	Introducción	349
8A.2.	Especificaciones generales	349
8A.2.1.	Especificación eléctrica	350
8A.2.2.	Especificación lógica	350
8A.2.3.	Especificación mecánica	351
8A.3.	Método de transmisión	351
8A.4.	Protocolo de comunicación	352
8A.5.	Diseño de un interfaz RS-232-C	353
Ejercicios		
54		

9.3.1.	Impresoras	370
9.3.2.	Impresoras gráficas y plotters	374
9.3.3.	Pantallas de tubo de rayos catódicos	375
9.3.4.	Salidas para microfilm	376
9.3.5.	Sintetizadores de voz	378
9.3.6.	Salida en tres dimensiones	379
9.4.	Periféricos de almacenamiento	379
9.5.	Memorias magnéticas Fundamentos de la grabación magnética	380
9.5.1.	Medio de grabación magnético	380
9.5.2.	Grabación y lectura. Códigos	381
9.6.	Banda magnética	389
9.7.	Tambores y discos	393
9.7.1.	Formatos	395
9.7.2.	Tiempo de acceso	396
9.7.3.	Velocidad de transferencia	398
9.8.	Memorias de bujibulas magnéticas	398
9.8.1.	Principios y tecnología	400
9.9.	Memorias ópticas	403
Ejercicios		
404		

Descripción de la arquitectura de los computadores de la familia "VAX"		
10.1.	Introducción	403
10.2.	Arquitectura básica del VAX-11/750	409
10.3.	El procesador central	410
10.3.1.	Unidad Operativa	412
10.3.2.	Memoria cache	412
10.3.3.	Buffer de conversión de direcciones	413
10.3.4.	Relojes	413
10.3.5.	Buffer de probabilidad de instrucciones	413
10.3.6.	Unidad de control de diagnóstico (WDCS)	413
10.3.7.	Registros generales	413
10.3.8.	Doble Palabra de Estado del Procesador (PSL)	414
10.3.9.	Memoria de Control del Usuario	415
10.4.	Estructura de los procesos	416
10.4.1.	Excepciones asíncronas del sistema (AST)	418
10.5.	La memoria principal	418
10.5.1.	Memorias ROM para el control de los periféricos	419
10.6.	Operaciones básicas de la memoria principal	420
10.6		

10.6.1. Lectura	420
10.6.2. Escritura de doble palabra	421
10.6.3. Escritura de byte y palabra	421
10.7. Registros de control y estado	421
10.8. Espacio de la memoria	422
10.8.1. Espacio de direccionamiento virtual	424
10.8.2. Código de protección	425
10.9. La traducción de las direcciones	426
10.9.1. Traducción de direcciones del espacio del sistema . .	427
10.9.2. Traducción de direcciones del espacio del proceso . .	427
10.10. Subsistema de Entradas Y Salidas. El bus "UNIBUS"	428
10.10.1. Adaptador de UNIBUS	430
10.11. El "MASSBUS"	433
10.12. Tipos de datos admitidos por las instrucciones del VAX-11 . .	437
10.13. El formato del código máquina de las instrucciones	437
10.14. Los operadores	439
10.15. Modos de direccionamiento	440
10.15.1. Modo inmediato o literal	441
10.15.2. Modo absoluto	441
10.15.3. Modo bifurcación	441
10.15.4. Modo registro	441
10.15.5. Modo indexado	444
10.15.6. Modo indirecto	444
10.16. Un breve recorrido por el repertorio de instrucciones	445
10.16.1. Instrucciones con números enteros y en coma flotante	445
10.16.2. Instrucciones aritméticas	445
10.16.3. Instrucciones lógicas y de movimiento	446
10.16.4. Instrucciones de cadenas de caracteres	449
10.16.5. Instrucciones para el tratamiento de campos de bits de longitud variable	450
10.16.6. Instrucciones de manejo de registros	451
10.16.7. Instrucciones para las bifurcaciones y los bucles	451
10.16.8. Instrucciones de manejo de subrutinas	456
10.16.9. Instrucciones de llamada y retorno a procedimientos	456
10.17. Ampliación de la gama de computadores VAX	460
10.17.1. Implementaciones de la arquitectura VAX en tecnología VLSI	463
Ejercicios	465

11

12

13

14

15

Computadores de altas prestaciones

11.1. El reforzamiento del paralelismo y el aumento de la velocidad de procesamiento	467
11.2. Clasificación de las arquitecturas de computador propuesta por Flynn	469
11.3. Clasificación comercial de los computadores	472
11.4. Introducción y concepto de la técnica de segmentación	473
11.4.1. Estructura y tipos de cadenas	477
11.4.2. Parones y choques en la cadena	478
11.4.3. Memorias entrallazadas	478
11.4.4. Los parones en la secuencia de instrucciones	481
11.4.5. La segmentación en la Unidad de Control microprogramada	483
11.5. Computadores vectoriales	486
11.5.1. Arquitectura del "Cray-1"	488
11.6. Computadores array	491
11.6.1. Arquitectura del computador AP-120B	492
11.7. Arquitectura SIMD: Procesadores matriciales y asociativos . .	496
11.7.1. Procesadores matriciales	497
11.7.2. Procesadores asociativos	500
11.8. Multiprocesadores o computadores MIMD	502
11.8.1. Multiprocesador "S-1"	508
11.8.2. Procesadores sistólicos	510
11.9. Computadores inteligentes de la quinta generación	512
11.9.1. Requerimientos de los computadores inteligentes	513
11.9.2. Estructura de los computadores de la quinta generación . .	514
11.10. Máquinas de flujo de datos	515
Ejercicios	518

Tendencias actuales en la arquitectura de microprocesadores

12.1. Evolución de los microprocesadores	520
12.2. Orientación de los microprocesadores de 32 bits	523
12.2.1. Capacidad para soportar lenguajes de alto nivel	524
12.2.2. Espacio de memoria de grandes dimensiones	524
12.2.3. Posibilidad de implantación de sistemas multitarea y multisuario	528

12.3.	Un digno precursor: el microprocesador 80286	528
12.3.1.	Diagrama de conexiones	530
12.3.2.	Registros	532

12.4. Modos de trabajo

12.4.1.	Modo real, compatible con el 8086	535
12.4.2.	Modo protegido	542

12.5. Multitarea

12.6.	Protección y niveles de privilegio	546
-------	------------------------------------	-----

12.6.1.	Puertas de llamada	546
12.7.	Interrupciones y excepciones	547

12.8. Análisis resumido del sistema lógico del 80286

12.8.1.	Modos de direccionamiento	549
---------	---------------------------	-----

12.8.2.	Juego de instrucciones	551
---------	------------------------	-----

Ejercicios

554		554
-----	--	-----

Teleinformática

557		557
-----	--	-----

Introducción

557		557
-----	--	-----

13.1.1. Redes de Área Local (LAN)

557		557
-----	--	-----

13.1.2. Redes Públicas o de Largo Alcance

558		558
-----	--	-----

13.1.3. Red Digital de Servicios Integrados (RDSI)

558		558
-----	--	-----

13.2. El modelo OSI

558		558
-----	--	-----

13.3. Medios para la transmisión de datos

561		561
-----	--	-----

13.4. La Transmisión

562		562
-----	--	-----

13.4.1. Banda Base, Banda Ancha y Banda Portadora

563		563
-----	--	-----

13.4.2. Modulación

563		563
-----	--	-----

13.5. Codificación de datos y sincronización

565		565
-----	--	-----

13.5.1. Redes

567		567
-----	--	-----

13.6.1. Red Punto a Punto y Red Multipunto

568		568
-----	--	-----

13.6.2. Escrutinio o "polling"

568		568
-----	--	-----

13.7. Multiplexido y concentración

569		569
-----	--	-----

13.8. Técnicas de acceso al medio

570		570
-----	--	-----

13.9. Nivel de datos, las tramas

571		571
-----	--	-----

13.9.1. Detección y control de errores

572		572
-----	--	-----

13.9.2. Corrección de errores

573		573
-----	--	-----

13.10. Nivel de red: Topologías

573		573
-----	--	-----

13.11. La norma X.25

575		575
-----	--	-----

Bibliografía

577		577
-----	--	-----

Índice alfabético

581		581
-----	--	-----

Presentación

Los países más avanzados en el desarrollo tecnológico dedican una atención prioritaria a la investigación de nuevas arquitecturas de los computadores, que permitan satisfacer las excepcionales necesidades que exigen las ambiciosas aplicaciones de un futuro cercano en campos como la exploración espacial, el reconocimiento de imágenes, el procesamiento de imágenes, el tratamiento de ingentes bases de datos, etc. Cualquier mejora en el rendimiento de la máquina incide, muy rotablemente, en la progresiva implantación del procesamiento automático en nuevas áreas del conocimiento.

La simplificación del diseño y la construcción de arquitecturas de computadores, como consecuencia del uso de circuitos integrados VLSI y, en especial, de los microprocesadores, aconsejan prestar un esfuerzo preferente al tema que se expone en esta obra.

Los autores han escrito este libro con la finalidad de desarrollar un primer curso sobre "Arquitectura de Computadores", en el que se describan, con toda claridad y rigor, los principios fundamentales en los que se basa la organización y la estructura de los modernos computadores, así como las líneas maestras que siguen los computadores de "altas prestaciones" y las máquinas inteligentes de la 5.ª generación. Puede ser de indudable interés en las Facultades y Escuelas Universitarias de Informática y en las Escuelas Superiores y Universitarios de Ingeniería. También es muy recomendable para los especialistas en diseño y mantenimiento de sistemas informáticos y técnicos cualificados de Microelectrónica.

Su lectura exige un conocimiento previo de Electrónica Digital. A continuación, se realiza una somera descripción del contenido de sus 12 capítulos.

El primer capítulo, *Introducción a la Arquitectura de Computadores*, describe la arquitectura propuesta por von-Neumann y su desarrollo a través del tiempo, que se ha plasmado en 5 generaciones, en las que se han ido potenciando los recursos físicos y lógicos de la máquina hasta llegar a los "supercomputadores" actuales. También se analiza la problemática que ha rodado el desacopló entre el equipo físico y el sistema lógico.

Sistemas de Representación de la Información es el título del segundo capítulo y en él se hace un detallado estudio, con numerosos ejemplos, de las diferentes formas que utilizan los computadores para representar la información que procesan.

Como una de las operaciones más frecuentes del computador es la del procesamiento aritmético y lógico, el tercer capítulo, *La Unidad Aritmética y Lógica*, describe los diversos circuitos y algoritmos que se emplean para llevar a cabo estas funciones.

El capítulo cuarto, *Memorias*, tras exponer las ideas generales sobre las memorias usadas en los computadores, se centra en el estudio de la memoria principal. Se describen los diversos procedimientos que mejoran las prestaciones de la misma, como la memoria virtual, la memoria cache y los sistemas de protección.

Modos de direccionamiento y repertorio de instrucciones es el título del quinto capítulo, que commenta, con numerosos casos prácticos, los formatos de las instrucciones, sus modos de direccionamiento, los tipos que existen en los repertorios y los principios del lenguaje Ensamblador.

El libro contiene numerosos ejercicios y casos prácticos, que facilitarán al lector la comprensión de los temas teóricos y estimularán su interés ofreciendo su aplicación en diversos modelos de máquinas comerciales.

La Unidad de Control es presentada, con todo detalle y rigor, en el capítulo sexto. Al final del mismo se incluye un Apéndice que analiza el diseño completo de una UCP didáctica, con circuitos integrados SSI y LSI.

Todas las alternativas de la *Unidad de Entrada y Salida*, sus características y los modelos de máquinas del mercado que las utilizan, se describen en el capítulo séptimo.

El octavo se destina a los *Buses* y contiene un Apéndice sobre la popular norma RS 232-C. El capítulo noveno hace una descripción exhaustiva de los principales periféricos que se usan en los computadores, dedicando una especial atención a los periféricos de almacenamiento de tipo magnético.

En el capítulo décimo se expone con detalle la arquitectura del procesador VAX-11/750, sus modos de dirección y el repertorio de instrucciones. Se ha considerado a este sistema lo bastante representativo y el capítulo se orienta como una aplicación práctica de los anteriores. El título del mismo es *Descripción de la arquitectura de los computadores VAX-11* y para su elaboración hemos contado con la colaboración de Digital Equipment.

El 11º capítulo, *Computadores de altas prestaciones*, describe, con abundantes ejemplos, los recursos que se emplean en la actualidad para mejorar la potencia de las máquinas y obtener "supercomputadores". Entre dichas técnicas se destacan la segmentación, el procesamiento paralelo y el conjunto de variantes de multiprocesadores que ya existen en el mercado. También se ofrece una panorámica de las máquinas inteligentes de la 5.ª generación.

El capítulo 12º ofrece una panorámica de las tendencias modernas aplicadas a la arquitectura de los microprocesadores de 16 y 32 bits, y en concreto al 80286 y 80386. El último capítulo se destina a presentar una introducción a la Teleinformática. También se ha incluido una colección de ejercicios, al final de cada capítulo.

Introducción a la arquitectura de computadores

1.1. DEFINICIONES

El computador es una máquina capaz de interpretar y ejecutar una serie de instrucciones. Está constituido por un conjunto de bloques lógicos electrónicos, comunicados entre sí mediante colectores de líneas o buses e interruptores, gobernados por unos controladores.

Al estudiar la arquitectura de los computadores hay que considerar los condicionantes físicos, así como también deben tenerse en cuenta el sistema lógico y las aplicaciones que han de soportar.

Los modernos computadores, construidos por los componentes derivados de la tecnología microelectrónica VLSI (Muy Alta Escala de Integración), se agrupan formando bloques lógicos digitales, capaces de interpretar y ejecutar un conjunto de instrucciones elementales (instrucciones máquina). Los usuarios emplean instrucciones de alto nivel, que son traducidas a instrucciones máquina mediante los compiladores; con la ayuda del sistema operativo, se ejecutan los programas para producir los resultados deseados.

Como muestra la figura 1-1, el computador puede estudiarse a muchos niveles.

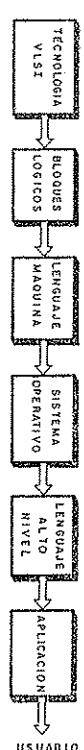


Fig. 1-1. Niveles de estudio del computador.

Los arquitectos de computadores han procedido, generalmente, del campo de la Ingeniería Electrónica y se han preocupado, en especial, de potenciar el *equipo físico*, aplicando los constantes avances de la Microelectrónica. En verdad, los costes de la circuitería o hardware se han reducido continuamente, pero ha sido a costa del encarecimiento del sistema lógico o software.

La misión fundamental de un arquitecto de computadores es definir el computador al nivel de lenguaje de máquina, esto es, especificar la máquina que "hace" el programador y establecer los objetivos de velocidad y capacidad que debe tener la máquina para proporcionar los recursos precisos para el desarrollo y explotación de los programas en los lenguajes de alto nivel, que constituyen las aplicaciones del usuario.

En la figura 1-2 se ofrece la distribución de las funciones que corresponden al sistema lógico y las que corresponden al equipo físico.

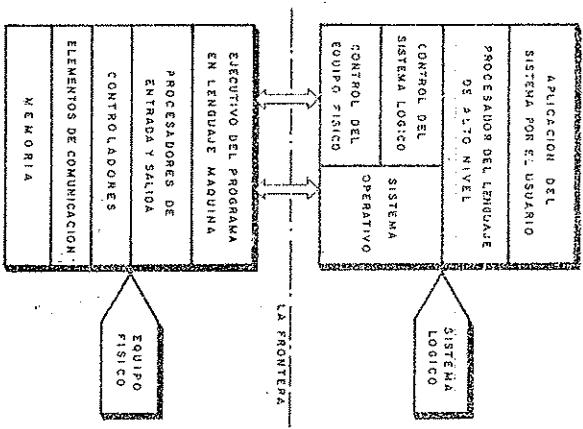


Fig. 1-2. Establecimiento de la frontera entre el sistema lógico y el equipo físico y asignación de las funciones correspondientes.

do por tal, el que se refiere al comportamiento conjunto de los elementos principales que intervienen en el sistema:

1. Lenguaje de programación.
2. Compilador.
3. Sistema operativo.
4. Máquina.

La falta de previsión de recursos estructurales que faciliten la programación, la compilación y la explotación de las aplicaciones, han supuesto un fuerte impacto para el progreso del procesamiento de la información por parte de los computadores.

Los futuros arquitectos de computadores han de saber integrar en la máquina todos los requisitos que luego se precian en su empleo, lo cual supone un elevado grado de conocimientos en las áreas del hardware y del software.

1.2. ESTRUCTURA CLÁSICA DEL COMPUTADOR

El advenimiento de las máquinas durante la primera revolución industrial trajo consigo un considerable aumento de la producción y del consumo, que motivaron constantes mejoras en los dispositivos de fabricación, en los diseños y en los cálculos industriales, así como el empleo de técnicas depuradas para el control de mercados y explotación.

Con el descubrimiento de la válvula de vacío en 1904, Fleming abrió las puertas a la historia de la Electrónica, cuyos formidables avances actúan como motor de impresión para las restantes ciencias y técnicas.

Al aplicar la tecnología electrónica en las máquinas, éstas se fueron haciendo más rápidas y potentes. Hasta mediados del siglo XIX cada máquina estaba construida y cableada para soportar una misión concreta. Fue en esa época cuando John von Neumann continuando la idea de su predecesor Babbage y otros investigadores, presentó su *máquina de programa almacenado*, a la que se denominó genéricamente computador.

La máquina de von Neumann no estaba diseñada para resolver una aplicación concreta, sino que podía emplearse en diferentes trabajos, previa la conveniente preparación del funcionamiento secuencial que debía llevar a cabo. Era una *máquina de propósito general*. De esta manera surgió el concepto de memoria, como un dispositivo capaz de almacenar las órdenes o instrucciones que debía realizar la máquina ordenadamente, así como los datos iniciales y los resultados finales que se obtenían del procesamiento de la información. La memoria sustituyó al conexionado fijo entre componentes de las máquinas cableadas, de forma que se podía alterar el com-

putador a crear una estructura que proporcione un buen rendimiento, entendiendo

portamiento de las máquinas programadas variando, simplemente, el programa de instrucciones.

Para simplificar el funcionamiento de los computadores y obtener importantes velocidades de trabajo, von Neumann utilizó elementos electrónicos *digitales*, que sólo eran estables en dos estados opuestos (saturación y bloqueo). La característica biestable de los componentes propició el uso del sistema de *numeración binario*, que, mediante sólo dos dígitos, el 1 y el 0, expresa cualquier cantidad. Consecuentemente, el computador se basa en una serie de módulos digitales electrónicos, que guardan, transmiten, reciben y procesan conjuntos de "unos" y "ceros", o sea, de *bits*.

E) diagrama general por bloques en el que se basa la máquina de von Neumann se muestra en la figura 1-3.

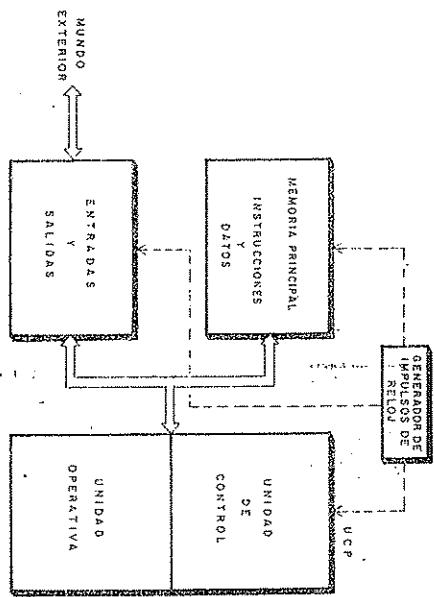


Fig. 1-3. Estructura básica de la máquina de programa almacenado de von Neumann.

Haciendo referencia a la figura 1-3, la Unidad de Control, junto con la Unidad Operativa, forman la *Unidad Central de Proceso (UCP)*, que es la sección encargada de ir tomando, ordenadamente, las instrucciones del programa almacenado en la memoria principal e ir interpretando su misión para poder ejecutarlas finalmente.

La sección de Entradas y Salidas transforma los datos provenientes del mundo exterior al código binario que es capaz de aceptar y entender la máquina. También saca al exterior, en forma útil, las informaciones y resultados obtenidos de la realización del programa de instrucciones.

2.2 / Introducción a la arquitectura de computadores

El generador de impulsos de reloj sincroniza las operaciones que se efectúan en los diversos órganos que intervienen en el tratamiento de la información.

En la memoria se guardan de forma ordenada los datos junto a las instrucciones que han de ser ejecutadas por los restantes componentes del sistema para obtener los resultados apetecidos. Sin embargo, el computador ha de poseer la capacidad de romper la secuencia de las instrucciones y producir saltos en el programa de acuerdo con los resultados que se van obteniendo en la ejecución del programa.

La notable aportación de von Neumann con el concepto de "máquina programada" ha constituido la base de los planteamientos, que aún se siguen aplicando a la mayoría de los computadores. Dicho concepto se fundamentaba en los tres principios siguientes:

1. Máquina electrónica digital, que trabaja con información codificada en binario.
2. Programa almacenado en memoria.
3. Posibilidad de provocar una ruptura de la secuencia ordenada de instrucciones en un programa.

1.3. INFLUENCIA DE LA TECNOLOGÍA EN LA EVOLUCIÓN DE LA ESTRUCTURA BÁSICA DE LOS COMPUTADORES

Hasta llegar a los supercomputadores actuales, en los que se enfatizan todas las características físicas y lógicas para conseguir una elevadísima velocidad, la estructura básica del procesador central ha sufrido una evolución que puede contemplarse en 4 etapas sucesivas muy relacionadas con la tecnología de fabricación.

1.3.1. Primera etapa

Los primeros computadores se construyeron siguiendo dos principios:

- a) El modelo propuesto por von Neumann, que incluía el concepto de *programa almacenado* y el de la *ruptura de la secuencia* en el programa.
 - b) La tecnología electrónica de la época, que hacía uso de las *válvulas de vacío*.
- En la figura 1-4 se muestra el diagrama por bloques del procesador central correspondiente a la 1.ª etapa.
- La Unidad de Control extrae la instrucción de la memoria principal, la interpreta y efectúa las siguientes operaciones:
- a) Establece el conexionado eléctrico de la ALU.

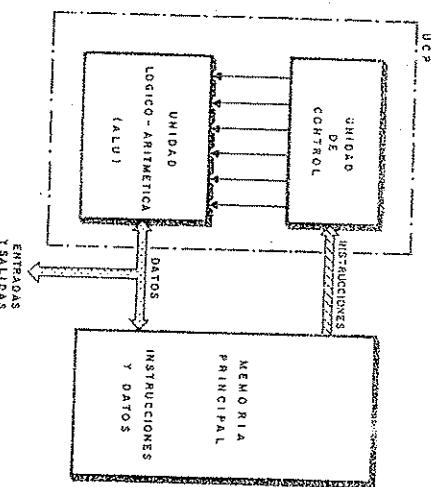


Fig. 1-4. Diagrama general por bloques de la estructura de los procesadores de la 1.ª etapa.

- Extrae los datos de la memoria.
 - Ordena la ejecución a la ALU.
 - Almacena el resultado en la memoria.
 - La Unidad Aritmético-Lógica o ALU es el bloque operativo que desarrolla todas las posibles funciones aritméticas y lógicas.
- En tanto la lógica de control como la memoria, estaban construidas con válvulas de vacío con lo que la velocidad de funcionamiento de ambas secciones era similar. La sencillez de la UCP y la escasez de registros internos de trabajo imponían una constante transferencia con la memoria empleando un conjunto reducido y básico de instrucciones máquina.

1.3.2. Segunda etapa

En esta etapa se produce un distanciamiento entre la tecnología usada en la construcción de la Unidad de Control y en la memoria principal. Los circuitos integrados de pequeña y media escala de integración (SSI y MSI) se usan para la construcción de la UCP, pero todavía no se han alcanzado cotas interesantes en la densidad de integración, por lo que en la sección de la memoria principal se aplican otras tecnologías, como los núcleos de ferrita, cuyos tiempos de acceso eran elevados.

24/ Introducción a la arquitectura de computadores

La velocidad de la memoria principal es mucho menor que la de la UCP (unas 10 veces), lo que provoca largos períodos de inactividad en la UCP, mientras se accede a memoria, en los computadores de esta etapa.

En esta época las máquinas comienzan a soportar lenguajes de alto nivel, como el FORTRAN y el COBOL, que tenían que ser traducidos a lenguaje máquina por compiladores, antes de ser ejecutados. En un intento de simplificar la compilación o traducción, se propició una potenciación de las instrucciones máquina para asentirlas a las de alto nivel.

Surgieron juegos de instrucciones complejos, en los que cada instrucción equivalía a varias operaciones simples, llamadas *operaciones elementales*. Así se evitaban muchos accesos a la memoria principal, al mismo tiempo que se sacaba el máximo rendimiento de la rápida UCP. A este tipo de computadores se les denomina CISC (Computadores de Juego de Instrucciones Complejo).

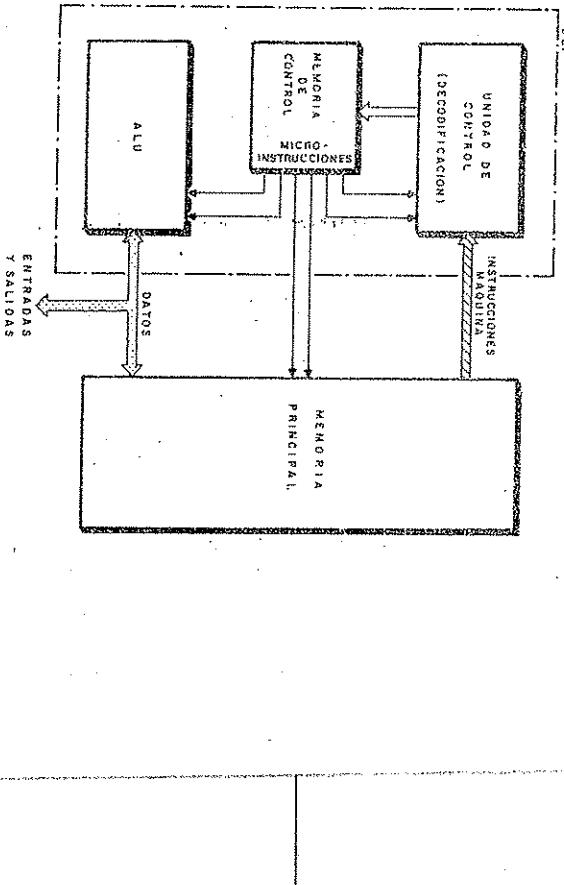


Fig. 1-5. Al aumentar la complejidad de las instrucciones máquina surgió la Memoria de Control, que se encargaba de almacenar los microcódigos o microinstrucciones correspondientes a cada macroinstrucción.

En muchos computadores la UCP pasó a contener a la *Memoria de Control*, que se trataba de una *rápidísima memoria* en la que se almacenaban las operaciones elementales correspondientes a cada instrucción compleja o macrocódigo, llamándose *microinstrucción* a cada una de las posiciones de esta Memoria de Control.

Se incrementa el proceso de decodificación de la *macroinstrucción*, pero se reduce el número de accesos a la memoria principal. Con este sistema se intentaba paliar el desfase de velocidades entre la UCP y la memoria principal. Véase la figura 1-5. En esta situación el computador quedaba limitado a la velocidad de la Memoria de Control, que, si bien se construía con semiconductores, con los que era mucho más rápida que la memoria principal, no permitía alcanzar las mismas velocidades que la lógica cableada.

En los computadores CISC microprogramados hubo un intercambio, mediante la Memoria de Control, entre la *lógica cableada* de las instrucciones simples y la *lógica programada* de las complejas. Así, se resolvía el problema derivado de las distintas velocidades entre las dos secciones principales del procesador. Además, se paliaba la divergencia semántica con las instrucciones de los lenguajes de alto nivel.

El número de instrucciones complejas de los CISC sólo está limitado por la capacidad de la Memoria de Control.

1.3.3. Tercera etapa

El vertiginoso desarrollo tecnológico de los circuitos integrados consiguió alcanzar la "Alta Escala de Integración" (LSI), que permitía la fabricación de memorias electrónicas rápidas y de cierta capacidad.

Aparece la memoria cache ultrarrápida, del tipo tampon, que se encarga de guardar la información de uso más frecuente de la memoria principal, para disminuir el número de accesos a esta última. Estas memorias equilibran la diferencia de velocidades entre la UCP y la memoria principal. Como se observa en la figura 1-6, la Unidad de Control se alimenta desde la memoria cache, o bien, cuando la información que guarda la cache está optimizada para que sea el de más uso, consiguiendo velocidades de 5 a 10 veces mayores que la memoria principal.

Los computadores con memoria cache intermedia poseen una UCP que, al tener que decodificar instrucciones complejas, tardan más en esta función que en el acceso a la memoria, por lo que este tipo de instrucción deja de parecer tan interesante. Para mejorar el diseño de los computadores se pasó a realizar un análisis de lo que sucedía con los programas en general y se comprobó:

1. Un mismo programa escrito en lenguaje de alto nivel, tenía muchas posibles traducciones a lenguaje máquina.

2. Los diseñadores de los compiladores empleaban conjuntos "reducidos" de instrucciones máquina.
3. Ciertas macroinstrucciones específicas apenas se empleaban.

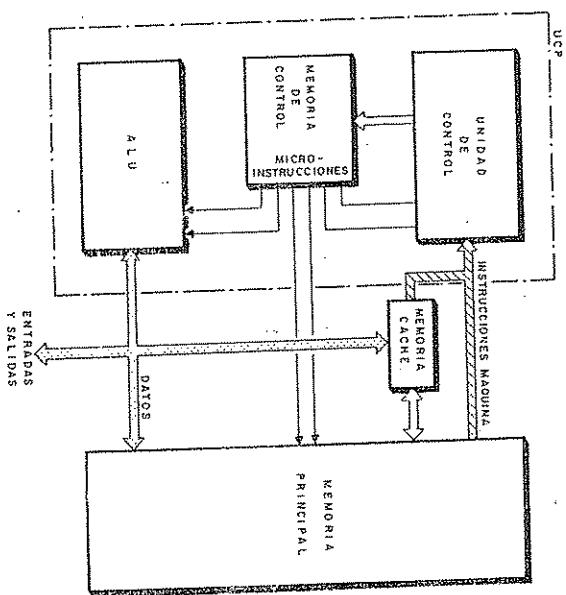


Fig. 1-6. La memoria cache almacena la información más utilizada por la memoria principal.

4. Aproximadamente el 50% del tiempo lo invierte la UCP en ejecutar instrucciones simples del tipo: CARGA, ALMACENAMIENTO Y BIFURCACIÓN. La frecuencia de utilización de las instrucciones máquina se puede analizar desde estos aspectos:

- a) *Análisis estático*. Cuando se estudian los listados de los programas y las instrucciones que aparecen en ellos.
- b) *Ánáisis dinámico*. Se tienen en cuenta las instrucciones que se van realizando en la ejecución de los programas, no las que aparecen en los listados. Tenga en cuenta que una instrucción de bifurcación condicional de un bucle, aunque sólo aparezca una vez en el listado del programa, se ejecuta muchas veces.

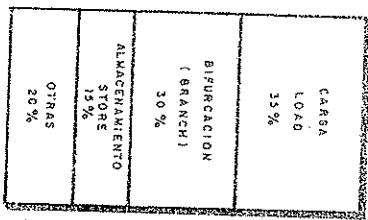


Fig. 1-7. Porcentaje de empleo de las instrucciones máquinas. Las instrucciones de carga, almacenamiento y bifurcación ocupan más del 80% del total.

Han existido numerosos investigadores que, sobre diferentes modelos de máquinas y de programas, han estudiado la frecuencia de utilización de las instrucciones máquinas. Destacan Gibson, Astley, Flynn y Fairclough. Una conclusión generalizada es que la mitad de las instrucciones máquinas se usan menos del 2%, por lo que se podrían eliminar sin afectar, prácticamente, al rendimiento. Dichas instrucciones sólo son eficaces en aplicaciones específicas.

1.3.4. Cuarta etapa

Con el fin de mejorar la velocidad de la UCP y equilibrarla con la de la memoria cache, se han adoptado los siguientes criterios en el diseño de la arquitectura de los computadores:

1. Eliminación de la microcodificación. Todas las instrucciones serían del tipo elemental, no existiendo instrucciones complejas.
 2. Reducción del tiempo del ciclo máquina, como consecuencia de la simplificación de las instrucciones.
 3. Interpretación directa de las instrucciones por el hardware y ejecución de cada una de ellas en un solo ciclo máquina.
 4. Selección del mínimo número de instrucciones simples.
- Con este nuevo enfoque surgieron los computadores RISC, Computadores de conjuntos de instrucciones Redicidas cuya arquitectura básica respondía al diagrama de la figura 1-4, aunque con la inclusión de una memoria cache, tal como aparece en la figura 1-8.

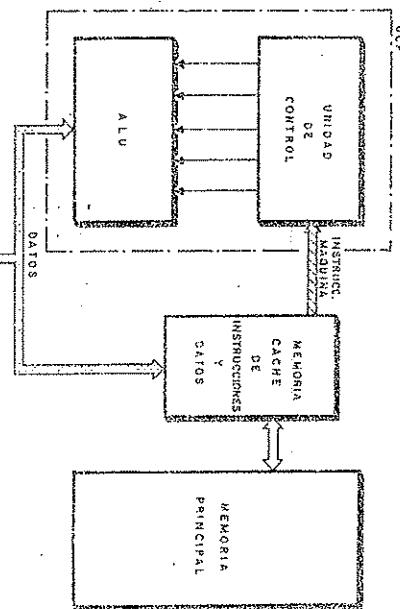


Fig. 1-8. Diagrama general de bloques de los computadores RISC en los que desaparece la Memoria de Control.

Los computadores RISC sólo disponen de las instrucciones máquina más importantes y usadas, que, generalmente, suelen ser menos de 30. Como, por otra parte, son sencillas y se llevan a cabo mediante lógica cableada, y no mediante microinstrucciones almacenadas, se pueden ejecutar en un ciclo máquina. En este tipo de computadores se ha incrementado el número de registros de propósito general para evitar accesos a memoria y ayudar a los compiladores a analizar los datos y controlar los flujos de forma óptima al conocerse las variables más usadas que se han asignado a los registros.

Las instrucciones RISC tienen normalizados tanto su formato como su longitud, lo que favorece el funcionamiento segmentado o pipe-line, al estar ubicados todos sus campos en sitios determinados. Sin embargo, dada la limitación del juego de instrucciones RISC únicamente a las funciones elementales, requieren empleo de procesadores especializados.

Conjuntos de Instrucciones Redicidas curva arquitectura básica respondía al diagrama de la figura 1-4, aunque con la inclusión de una memoria cache, tal como aparece en la figura 1-8.

Universidades como las de Berkeley y Stanford y empresas como IBM, Nixdorf, Hewlett Packard, Bull, Acorn, Inmos y Metافorth, disponen de máquinas RISC con amplias perspectivas de futuro.

1.4. PROBLEMATICA EN EL DESARROLLO DE LA ARQUITECTURA DE COMPUTADORES

Hasta hace pocos años, ha sido tradicional que los arquitectos de computadores establecieran su meta en diseñar máquinas con buenas características, pero teniendo en muy poca consideración las prestaciones del sistema lógico.

Las dos causas principales del escaso progreso en la arquitectura de computadoras en las tres primeras décadas de su evolución, han sido:

- a) La falta de una adecuada definición de las funciones que debía de soportar el hardware y las que corrían a cargo del software.
- b) El seguimiento a ultranza del modelo propuesto por von Neumann, que, aunque revolucionario en su época, no estaba diseñado para manejar los nuevos sistemas operativos, lenguajes y aplicaciones. En este sentido se puede afirmar que, desde la construcción de los computadores tipo EDVAC hasta nuestros días, sólo se han introducido unas escasas novedades entre las que destacaan:

- Empleo de *registros índice* para proporcionar el *direcciónamiento indexado*.
- Introducción de conjuntos de *registros de propósito general*.
- *Direcciónamiento indirecto*.
- *Procesadores de entradas y salidas*.
- *Memoria virtual*.

Los mayores obstáculos que presenta la arquitectura propuesta por von Neumann se centran en dos aspectos. En primer lugar, el empleo de una memoria unidimensional, de tipo secuencial, en la que las informaciones se almacenan de forma consecutiva. Los modernos computadores precisan una memoria muy flexible, así como manejar datos en estructuras multidimensionales.

Otro freno en el progreso de la arquitectura de computadores, derivada de la idea de von Neumann, hace referencia a la falta de distinción entre datos e instrucciones, lo que significa que cualquier información puede ser considerada como un dato o como una instrucción. Esta característica es desaconsejable para la ejecución de programas escritos en lenguajes de alto nivel.

Otras razones, de carácter secundario, que han favorecido el desacoplamiento entre el equipo físico y el sistema lógico, son:

- Uso exclusivo de la aritmética binaria.
- Empleo de palabras de memoria de tamaño fijo.
- Limitación del número de registros.

Las arquitecturas avanzadas de computadores exigen cambios conceptuales que hacen referencia al uso de datos autodefinidos, bien sea a base de etiquetas o de descripciones, y al desarrollo de sistemas de procesamiento en paralelo.

30 / Introducción a la arquitectura de computadores

1.5. LA HISTORIA DE LOS COMPUTADORES A TRAVES DE SUS GENERACIONES

Aunque los antecedentes del computador se remontan al ábaco griego, en realidad su historia se inicia a mediados del siglo XX. No obstante, entre sus predecesores se pueden citar:

- John Napier (1550-1617), matemático escocés que construyó una máquina a base de palillos, que realizaba las operaciones de multiplicar y dividir.
- Blaise Pascal, que en 1642 diseñó una máquina de engranajes para sumar y restar.
- Leibnitz (1646-1716), que inventó una calculadora que empleaba el sistema binario.
- Joseph Jacquard (1752-1834), que empleó tarjetas perforadas para el control de los telares.

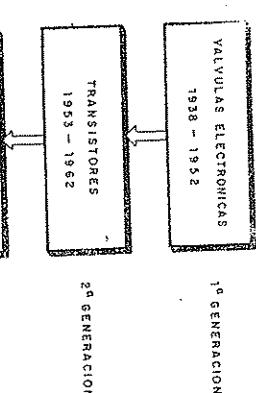


Fig. 1.9. Principales características de las cinco generaciones de computadores que se han producido hasta el presente.

1

— Charles Babbage, que sentó las bases de los actuales computadores y diseñó en 1832 una máquina diferencial para el cálculo de polinomios. En 1883 comenzó a trabajar en una máquina de propósito general capaz de resolver cualquier problema matemático y que no logró acabar.

— Hernan Hollerit, quien, a finales del siglo XIX, utilizando tarjetas perforadas codificadas, impulsó la construcción de máquinas destinadas a la elaboración de los censos. Fundó una compañía, que, al fusionarse con otras dos, dio lugar a la International Business Machines más conocida mundialmente por IBM.

Hasta la aparición y desarrollo de la Electrónica se puede afirmar que no existieron computadores tal como hoy se les considera. A partir de ese momento, la historia de los computadores quedó íntimamente ligada a la de la Electrónica, y las etapas que han ido cubriendo aquéllos, también llamadas generaciones, se han basado en los avances tecnológicos de la Electrónica. En la figura 1-9 se muestran las características más relevantes de las 5 generaciones de computadores que se han desarrollado hasta el presente.

1.3. PRIMERA GENERACIÓN (1938-1952)

Tecnología: En esta generación se usaron las válvulas de vacío para construir los computadores. Eran componentes voluminosos, caros, de elevado consumo, gran dissipación de calor y una limitada vida de funcionamiento.

Desarrollo histórico: Se considera al computador ENIAC (1946) como el primero de los fabricados con Electrónica Digital. Figura 1-10.

El proyecto del ENIAC fue dirigido por Eckert y Mauchly en la "Moore School Engineering" de la Universidad de Pensilvania. Constaba de unas 18.000 válvulas, 70.000 resistencias, 7.500 interruptores y consumía 100 kW, por lo que necesitaba ventilación forzada para disipar la gran cantidad de calor que producía.

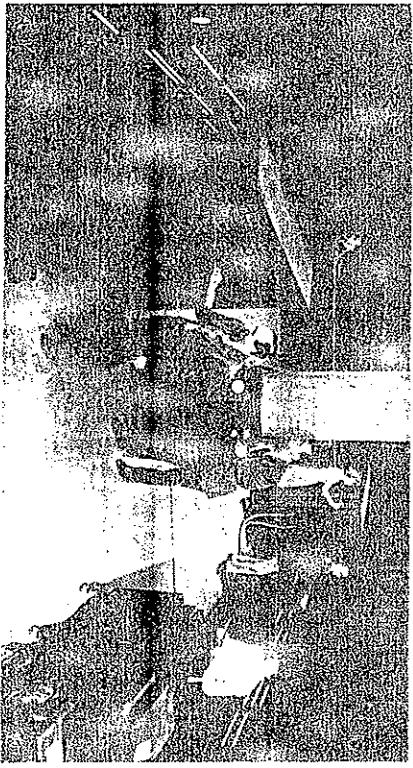
El procesador del ENIAC disponía en su estructura de 20 registros de 10 dígitos. Era capaz de sumar, restar, multiplicar y dividir en decimal; tenía tres tipos de tablas de funciones y la entrada y salida de datos y resultados se realizaban mediante tarjetas perforadas.

Fue von Neumann quien³ propuso modificar el ENIAC en dos importantes aspectos, que dieron lugar al computador EDVAC en 1952. Dichos aspectos fueron:

1. *Programa almacenando*, en sustitución del programa "cableado" usado hasta entonces. Esto suponía mantener inalterable la organización física del computador para todas las aplicaciones.
2. *Aritmética binaria codificada*, en lugar de la decimal. Permitió simplificar enormemente los circuitos electrónicos encargados de realizar los cálculos.

3.2 / Introducción a la arquitectura de computadoras

Fig. 1-10. Fotografía del computador ENIAC. Cortesía de UPI.



La arquitectura de un computador que siga el modelo de von Neumann consta de 4 bloques fundamentales, que se representan en la figura 1-11 y que son:

- *Unidad de Control:* Es la encargada de interpretar los códigos binarios con los que se expresan las instrucciones y, posteriormente, ejecutarlas.
- *Unidad Lógico-Aritmética:* También llamada Unidad Operativa y, abreviadamente, ALU, tiene la misión de efectuar las operaciones lógicas y aritméticas.
- *Memoria principal:* En este componente se almacenan los datos, las instrucciones y los resultados parciales y finales. En esta etapa la memoria principal era de tipo unidimensional y se direccionaba secuencialmente. No existía discriminación alguna entre los códigos correspondientes a los datos con los de las instrucciones.
- *Módulos de Entrada/Salida:* Introducen los datos o informaciones al sistema desde el exterior y transmiten los resultados a los periféricos.

Por otra parte, Wilkes construyó en 1949 el computador EDSAC para la Universidad de Cambridge, en el que se aportaba el concepto de memoria jerárquica en la ejecución de programas almacenados.

Junto a un sin fin de máquinas de proceso que seguirían el patrón de von Neumann y se orientaban al tratamiento de los cálculos científicos, también se comenzó en el MIT la construcción de un computador orientado a trabajar en tiempo real. Se trataba del WII, presentado en 1951.

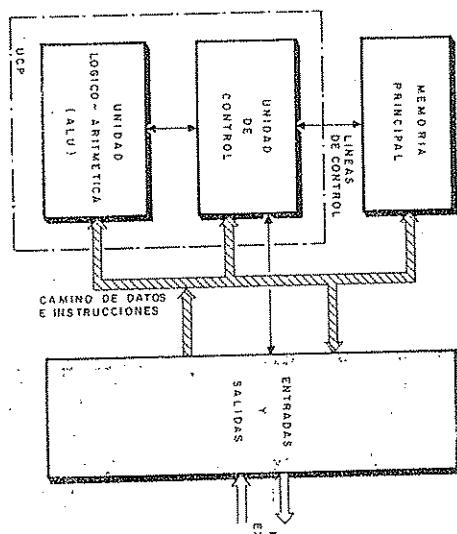


Fig. 1-11. Diagrama general por bloques correspondiente a la arquitectura propuesta por von Neumann. *Consta de 4 bloques fundamentales: Unidad de Control, Unidad Lógico-Aritmética, Memoria Principal y Módulo de ES.*

Avances en el equipo físico: La memoria principal, que inicialmente estaba formada por registros a base de válvulas de vacío, fue reemplazada por sistemas de núcleos de ferrita, cuyo ciclo medio de acceso era de unos pocos microsegundos. Previamente, también se habían utilizado los "tubos de Williams", que almacenaban 1.024 bits cada uno y tenían un tiempo de acceso de 25 μ s. Incluso, se llegaron a usar los tambores magnéticos.

Como elementos de memoria secundaria, se comenzó a sustituir las tarjetas y cintas perforadas por tambores y cintas magnéticas.

Se impulsó la arquitectura de von Neumann dirigiéndola hacia el empleo compartido de la memoria principal y los periféricos. También se introdujo la capacidad de interrumpir a la UCP.

Avances en el sistema lógico: Especialmente a partir de 1950, se mejoraron los aspectos relacionados con la lógica y la programación. Hasta entonces, sólo se había empleado el lenguaje máquina, lo que exigía excelentes programadores.

Hubo intentos de mejora de los lenguajes ensambladores, macroensambladores y de alto nivel, pero el FORTRAN no estuvo disponible hasta 1957.

Se introdujo el concepto de registros indexados, de gran valor en la simplificación de la programación.

34 / Introducción a la arquitectura de computadores

Modelos comerciales: El primer computador comercial fue el UNIVAC I, construido para la oficina del censo de EE.UU. Diseñado por Eckert y Mauchly cuando abandonaron la Universidad de Pensilvania, se caracterizó por el empleo de cintas magnéticas.

Al UNIVAC I le siguieron otros modelos, hasta llegar al UNIVAC 1103 en 1956, cuya circuitería le permitía operar en coma flotante y tenía capacidad para interrumpir los programas en curso.

Se consideran modelos representativos de esta generación, además de los de UNIVAC, los de IBM, que en 1953 lanzó al mercado el modelo 701 al que siguieron el 704 y el 709.

Otras compañías como Raytheon, Honeywell y RCA, también se interesaron por el mercado de los computadores en esta época.

1.7. SEGUNDA GENERACIÓN (1953-1962)

Tecnología: Esta generación está caracterizada, al igual que las demás, por la innovación electrónica, que, en este caso, se materializa con el descubrimiento del transistor, hecho que sucedió en 1948 en los laboratorios Bell, siendo sus protagonistas Bardeen y Brattain..

En 1954, dichos laboratorios construyeron el primer computador digital transistorizado, el TRADIC.

El transistor, al ser más pequeño, más barato y de menos consumo que la válvula, hizo a los computadores más asequibles en tamaño y precio.

Desarrollo histórico y modelos comerciales: El primer computador transistorizado de IBM data de 1960 y fue el modelo 7070. Para posibilitar el acercamiento de los pequeños clientes, IBM ofreció en 1961 el modelo 1401, con el que eran compatibles los computadores de la serie 200 de Honeywell.

Dentro de la gama de computadores científicos, los modelos 7090 y 7094 reemplazaron al 709.

UNIVAC participó en esta generación con el modelo 1107, que sustituyó al 1103.

A principios de 1960 UNIVAC e IBM disponían de dos supercomputadores en los que se habían incluido importantes mejoras arquitectónicas, aunque no tuvieron éxito comercial. Así, el LARC de UNIVAC, del que sólo se fabricaron dos unidades, disponía de un procesador de Entradas y Salidas que operaba en paralelo con la UCP. El STRECH de IBM era capaz de adelantar la ejecución de instrucciones y de corregir errores.

1

Otros eventos importantes de la segunda generación fueron:

1. El proyecto PILOT del "National Bureau of Standards" para diseño de un multiprocesador con UCP independiente y de aplicación específica.
2. El modelo D-825 de Burroughs, que se puede considerar como un verdadero multiprocesador.
3. El computador CDC 6600, construido por la empresa CDC y que constituyó uno de los logros más importantes de la época.
4. La presentación del modelo PDP-5 de Digital Equipment Corporation (DEC), que fue el precursor de los famosos minijefadores PDP de la tercera generación.

Avances en el equipo físico: Además del cambio tecnológico expuesto, en esta generación se introdujeron los canales adecuados para poder desarrollar operaciones en paralelo y de tipo asíncrono simultáneamente, bajo el control de la UCP.

Se consolidó el empleo de nuevos dispositivos de memoria, como los núcleos de ferrita, los conjuntos de discos intercambiables, etc.

A los sistemas que funcionaban en tiempo compartido se les permitió el manejo de dos bancos de memoria principal, que se seleccionaban bajo el control del programa en ejecución.

Avances en el sistema lógico: El FORTRAN, lenguaje de alto nivel de carácter científico, que acababa de ser "creado" en la generación anterior, adquirió un fuerte impulso.

La primera versión del COBOL, un lenguaje de alto nivel orientado a los negocios, surgió en 1960.

Otro lenguaje de esta generación, especializado en el campo científico, fue el ALGOL.

El PL/I aparece como un lenguaje que intenta aprovechar las ventajas de los tres citados.

Se extiende el uso del procesamiento en batch o por lotes, que consiste en la ejecución automática secuencial de los programas del usuario, uno a uno.

1.8. TERCERA GENERACION (1963-1971)

Tecnología: Los computadores de esta generación se construyen con *circuitos integrados*, que son pastillas que contienen numerosos componentes discretos (transistores, diodos, resistencias y condensadores) interconectados y formando blo-

ques funcionares. En esta época se utilizan los circuitos integrados de baja (SSI) y media (MSI) escala de integración. Los primeros contienen un máximo de 12 pines lógicas y los segundos, hasta 100. Es decir, un número de transistores inferior a 1.000, en cualquier caso.

Desarrollo histórico y modelos comerciales: En el campo de los computadores comerciales, la tercera generación está marcada con la aparición del sistema 360 de IBM de diversos modelos compatibles, que disponían el mismo juego de instrucciones máquinas.

Aparecen las *familias* de computadores, esto es, computadores de distinta potencia y precio que tienen la misma arquitectura y son, por tanto, totalmente compatibles.

Se produce la explosión de los *minicomputadores*, comunitadores de recursos limitados, pero muy assequibles. Los modelos PDP-8 y PDP-11 de DEC se hicieron populares en todo el mundo.

Dentro de los supercomputadores, destinados a ciertas aplicaciones muy restrictivas y exigentes, CDC presenta en 1969 el modelo 7600, de amplia resonancia.

Avances en el equipo físico: El conexionado de los circuitos integrados sobre placas o tarjetas de circuito impreso multicapa simplifica y potencia la configuración física del computador.

La memoria de núcleos de ferrita empieza a ser desplazada por las memorias electrónicas en circuitos integrados.

Se introducen las memorias ultrarrápidas o cache, que actúan como memorias intermedias entre la Unidad de Control y la memoria principal aumentando la velocidad de búsqueda de las instrucciones.

Se acepta para las UCP una organización basada en un conjunto simétrico de registradores de propósito general.

Se refuerzan los tipos de interrupciones añadiendo niveles de prioridad y se inicia el empleo de los sistemas de memoria virtual paginada.

Avances en el sistema lógico: Los lenguajes de alto nivel apenas se renuevan y sólo la aparición de versiones derivadas, como sucede con el lenguaje BASIC o el PASCAL, son los acontecimientos más señalados en este tema.

Por el contrario, los sistemas operativos dan un paso gigantesco estructurándose bajo el esquema de la multiprogramación, que permite la ejecución simultánea de varios segmentos de programas solapados con operaciones de E/S.

Se presta una especial atención al control automático de sistemas jerárquicos de memoria virtual y a la compartición de los recursos entre los usuarios con técnicas de protección.

A finales de la década de los 60 ya existen en el mercado sistemas que funcionan en tiempo compartido.

36 / Introducción a la arquitectura de computadores

1.9. CUARTA GENERACIÓN (1972-1987)

Tecnología: Los constantes progresos en el incremento de la densidad de integración alcanzaron, en 1971, la cota necesaria para incluir en un chip a todos los elementos que conforman la Unidad Central de Proceso. Dicho circuito integrado recibe el nombre de *microprocesador* y dio origen a un computador pequeño y barato denominado *microcomputador*.

La tecnología LSI (Alta Escala de Integración) precedió a la VLSI (Muy Alta Escala de Integración), con la que se ha conseguido introducir un millón de componentes en un circuito integrado. Con esta última tecnología se han integrado en una sola pastilla todos los elementos que componen un microcomputador. A dicha pastilla se la llama *microcomputador monopastilla*.

Desarrollo histórico y modelos comerciales de microprocesadores: Se describen, someramente, las 5 fases del desarrollo de los microprocesadores:

- 1.^a fase. *Microprocesadores de 4 bits (1971-1974):* En esta etapa de introducción de los microprocesadores, la palabra de trabajo constaba sólo de 4 bits, la estructura interna era muy sencilla y el juego de instrucciones, muy reducido.
- 2.^a fase. *Microprocesadores de 8 bits (1974-1976):* Manteniendo la filosofía de los microprocesadores de 4 bits, se aumenta el tamaño de la palabra de trabajo a 8 bits y se incrementan la velocidad, la potencia de cálculo y el juego de instrucciones.

Se adopta este componente como estándar en la industria y da origen a los microcomputadores personales.

- 3.^a fase. *Microprocesadores de 8 bits mejorados (1976-1978):* Se añaden a los microprocesadores de 8 bits nuevos recursos físicos, direccionamientos y tipos de instrucciones. También se aumenta la velocidad de funcionamiento.

Se produce la masificación en la aplicación de los microprocesadores en la industria y en los computadores personales.

- 4.^a fase. *Microprocesadores de 16 bits (1978-1980):* Se modifica progresivamente la arquitectura de von Neumann, para disponer de recursos físicos y repertorios de instrucciones que se adapten mejor a los lenguajes de alto nivel y a los sistemas operativos avanzados. Figura 1.12.

En esta época surgen los microcomputadores profesionales y nuevas aplicaciones en campos como la Robótica y la Visión Artificial.

- 5.^a fase. *Microprocesadores de 32 bits (1981-1987):* Los microprocesadores que operan con palabras de 32 bits están orientados a los tamaños de alto nivel y a los sistemas operativos que admiten multiprogramación y multisistema. Su velocidad es muy elevada, disponiendo de circuitos auxiliares para gestión de la memoria y tratamiento en coma flotante. Aparecen en el mercado

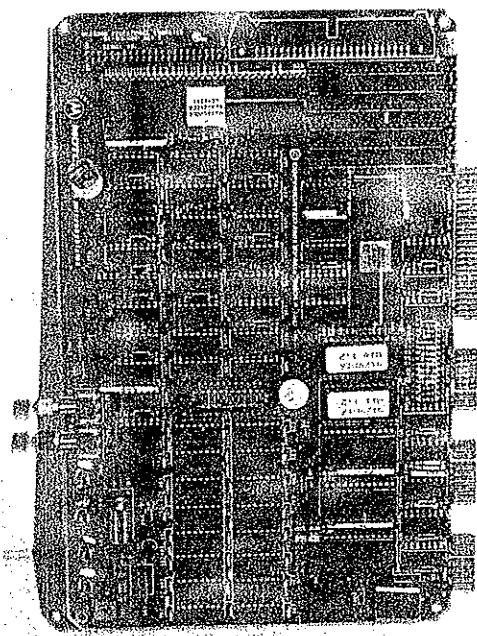


Fig. 1.12. Fotografía de una tarjeta de microcomputadora basada en un microprocesador de 16 bits.

los *micro-minis*, que, siendo microcomputadores basados en microprocesadores de 32 bits, asemejan sus prestaciones a los minicomputadores convencionales.

Los campos de aplicación de estas máquinas alcanzan a la Inteligencia Artificial, el CAD/CAM y otros.

Además del vertiginoso desarrollo de los microprocesadores, la cuarta generación de computadores está caracterizada por la construcción de supercomputadores de altas prestaciones, como el Fujitsu M382 (1981) y el Cray X-MP (1983).

Avances en el equipo físico: Los logros tecnológicos dirigidos hacia el incremento de la densidad de integración, no sólo repercutieron en el desarrollo de los microprocesadores, sino también en el de las memorias integradas, que pasaron a ser el elemento estándar de la memoria principal.

Por otro lado, se alcanzaron nuevas e importantes densidades de grabación en los medios magnéticos mediante las técnicas de magnetización vertical.

Avances en el sistema lógico: Se extienden los lenguajes de alto nivel capaces de manejar datos escalares y vectoriales.

Se normaliza el uso de la memoria virtual.

La mayoría de los sistemas operativos, además de la multiprogramación y el multiproceso, funcionan en tiempo compartido.

1.10. QUINTA GENERACIÓN (1987...)

Tecnología: En base al empleo de los modernos circuitos integrados con más de un millón de componentes, la industria y la investigación en el área de los computadores se decanta hacia la construcción de dos tipos de máquinas:

- 1) Supercomputadores de altísima velocidad.
- 2) Computadores de funciones inteligentes.

1.10.1. Supercomputadores de alta velocidad

Las recientes aplicaciones a las que se destinan los computadores requieren una elevadísima velocidad de procesamiento. Tal es el caso del proceso de imágenes en tiempo real, el control inteligente de trayectorias de robots, simulación de vehículos espaciales y otras, de laboriosos y complejos cálculos, como la prospección geológica, la meteorología y la medicina.

Para aumentar la velocidad en el procesamiento existen diversas alternativas, que pasamos a comentar:

7.º) Nuevas tecnologías: Utilizando nuevos componentes y procesos basados en el silicio, en el arseniuro de galio, en los dispositivos Josephson y en los de tipo óptico, se diseñan memorias con tiempos de acceso muy pequeños y puertas lógicas de tiempo de respuesta de pocos nanosegundos.

Sin embargo, las limitaciones técnicas y el tiempo requerido para la transmisión de las señales a lo largo de los cables y pistas de comunicación, que, a una velocidad de 300.000 km/s, necesitan más de una décima de nanosegundo para recorrer 3 cm, impiden que sólo con tecnología se pueda incrementar indefinidamente la velocidad.

2.º) Incremento del hardware: Con esta solución se pretende reducir el número de niveles lógicos por los que hay que pasar para alcanzar una solución. Este es el caso de los sumadores con acarreo anticipado.

3.º) Aumento de la complejidad de los circuitos combinacionales, en sustitución de los secuenciales: Un ejemplo de aplicación de esta técnica es la del uso de los PLA, que eleva la velocidad en la decodificación de las instrucciones.

4.º) Sustitución del software por circuitería: Esta tendencia se materializa en el intento de incluir en el hardware las funciones más frecuentes que se llevan a cabo mediante sistemas lógicos.

5.º) Nuevas estructuras de la memoria: Con este método se intenta reducir los tiempos de acceso a la memoria, que suelen ser del orden de 5 a 10 veces mayores que los de la UCP. Hay diversos procedimientos:

- a) Inserción de una memoria cache ultrarrápida entre la memoria principal y la UCP.
- b) Memoria virtual.
- c) Potenciación de los registros internos de la UCP.
- d) Memoria entrelazada, en la que la memoria principal se divide en varios módulos con un determinado número de palabras cada uno.

6.º) Multiprocesadores: Sistemas compuestos por varios procesadores.

7.º) Reforzamiento del nivel de concurrencia en las instrucciones: Mediante un proceso de eliminación de dependencias se intenta ejecutar simultáneamente varias instrucciones.

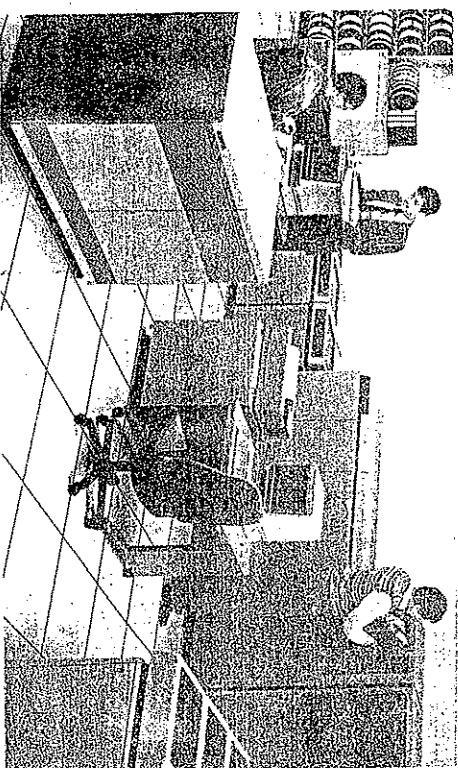


Fig. 1-13. Modernos sistemas de computadores de gestión HP 3000, de altas prestaciones y nueva arquitectura de precisión con juego reducido de instrucciones (RISC). Cortesía de Hewlett Packard.

8.º) Aumento del paralelismo en todos los niveles:

- A nivel de tareas o programas, a base de la multiprogramación, el tiempo compartido y el multiproceso.

- A nivel de segmentos o partes de un programa, lo que implica su descomposición en trozos.

- A nivel de instrucciones, que requiere un análisis de la posible dependencia de los datos.

- A nivel de las partes en que se descompone la instrucción y, por tanto, en estrecha relación con el hardware.
- 9.) *Reforzamiento de la técnica de segmentación:* Se potenció el solapamiento de las partes de una instrucción para aprovechar al máximo los recursos del sistema y aumentar su velocidad.

10.) *Procesadores de flujo de datos y sistólicos.*

1.10.2. Computadores de funciones inteligentes

Inicialmente, el procesamiento que realizaban los computadores operaba sólo sobre los *datos*, los cuales consistían, fundamentalmente, en valores numéricos, caracteres y símbolos. Luego se orientaron hacia la manipulación de *informaciones*, que no eran otra cosa que "conjuntos de datos relacionados".

Cuando se añadieron ciertos significados semánticos a la información, ésta se transformó en *conocimiento* y pasó a ser el nuevo elemento de procesamiento de los computadores.

Finalmente, los computadores de la 5.ª generación están diseñados para ser capaces de procesar *funciones inteligentes*, configuradas por un conjunto de conocimientos.

Aunque aún parecen distantes los *computadores inteligentes*, se están desarrollando importantes proyectos de investigación cuyos objetivos prioritarios se dirigen a:

- Almacenamiento de conocimientos. Introducción de hechos y reglas para configurar "bases de conocimientos".
- Realización lógica de deducciones a partir de las bases de conocimientos. En este apartado se engloban los temas dedicados a las inferencias, la solución de problemas y el aprendizaje artificial.
- Simplificación del interfaz hombre-máquina mediante el desarrollo del lenguaje natural y la visión artificial, especialmente.

EJERCICIOS

Ejercicio 1.1

Determinar si son verdaderas o falsas las siguientes afirmaciones, razonando brevemente la respuesta:

- a) Von Neumann diferenciaba entre datos e instrucciones.
- b) Von Neumann propuso modificar el ENIAC aportando el concepto de memoria jerárquica dando lugar al computador EDSAC.
- c) El uso de la memoria virtual se generaliza en la cuarta generación.
- d) Una característica fundamental de los avances lógicos de la 3.ª generación es la aparición de la multiprogramación y el tiempo compartido.
- e) Los computadores de flujo de datos son la última expresión de la arquitectura de Von Neumann.
- f) Con la memoria CACHE se incrementa el proceso de decodificación de la microinstrucción, pero se reduce al n.º de accesos a la memoria principal.
- g) El aumento de la velocidad en los computadores mediante el aumento del paralelismo en todos los niveles no tiene ningún tipo de limitación.
- h) En la actualidad, se tiende en lo posible a sustituir software por hardware para aumentar la velocidad de proceso.

Ejercicio 1.2

Establece la diferencia entre compilador e intérprete.

Ejercicio 1.3

¿Cuáles son los frenos que pone la estructura de VON NEUMANN al progreso en la arquitectura de computadores?

Ejercicio 1.4

Desarrollar una clasificación de los computadores en fases históricas, teniendo en cuenta los avances en software y hardware.

1

Ejercicio 1.5

¿Cuándo aparece la memoria CACHE? ¿Por qué? ¿Cuál es su funcionamiento?

Ejercicio 1.6

— ¿Cuál es la misión fundamental de la memoria de control? ¿Cuál es su evolución a lo largo de las 4 etapas?

Ejercicio 1.7

Indicar y explicar los bloques de que consta la CPU.

Ejercicio 1.8

— ¿Cuáles son las razones de la tendencia de computadores CISC? ¿Por qué se pasó al RISC? ¿Cuál se considera más conveniente?

Ejercicio 1.9

Explicar brevemente las nuevas tendencias de los computadores.

Sistemas de representación de la información

2

2.1. INTRODUCCION

Un computador trabaja, fundamentalmente, con dos tipos de información: las instrucciones y los datos. Consecuentemente, su estructura vendrá definida por las formas de representación de ambas informaciones.

Las instrucciones especifican las posibles operaciones que se pueden efectuar sobre los datos. De esta forma, los datos de entrada se procesan, de acuerdo con las instrucciones del programa, para obtener los datos de salida o resultados. Figura 2-1.

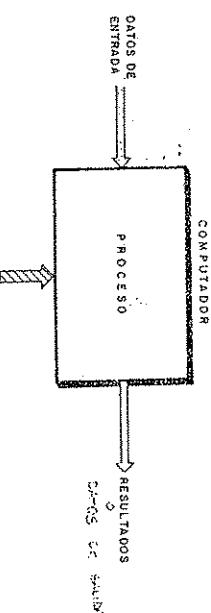


Fig. 2-1. El procesamiento de los datos de entrada al computador mediante las instrucciones del programa, genera los resultados.

En un computador existen tres clases de datos básicos:

- Datos de entrada para ser procesados.
- Datos de salida, como resultado del procesamiento.
- Datos intermedios.

DATOS DE
ENTRADA

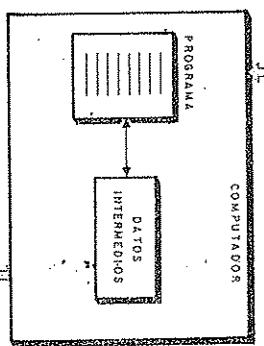


Fig. 2-2. Los tres tipos de datos básicos que existen en el computador.

El computador ha de ser capaz de representar los datos sobre los elementos materiales de que dispone al efecto y que, en general, son registros o posiciones de memoria de un número determinado de bits. De esta forma, la idea que expresa el dato, por ejemplo una cantidad, la expone y comunica el ser humano mediante la forma hablada o escrita y el computador habrá de representarla a base de varios bits, como se indica en la figura 2-3.

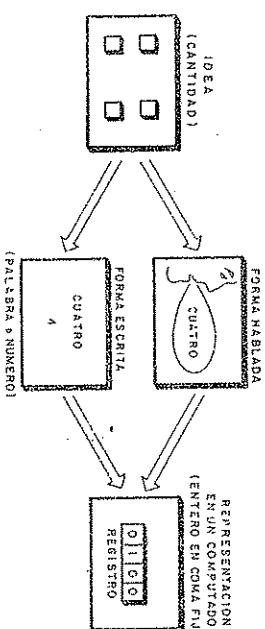


Fig. 2-3. Expresión gráfica de las tres etapas de la representación de una idea.

46 / Sistemas de representación de la información

Los datos de entrada pueden venir codificados de manera directa, a partir de nuestro lenguaje escrito; es decir, cuando se introducen datos (letras o números) por un terminal, cada uno de los caracteres del teclado se codifica dentro de un byte según un código alfanumérico que se expandirá más adelante. Estos datos, cuando son numéricos, se suelen traducir a un código más efectivo.

El proceso inverso se realiza cuando los datos de salida del computador han de ser interpretados directamente por una persona. Los formatos de datos que maneja el computador no son fácilmente reconocidos por un ser humano, por lo que deben ser traducidos apropiadamente.

También hay datos que el conjunto de instrucciones o programa necesita para efectuar sus trabajos internos. Estos datos se guardan directamente en un formato cómodo para su manipulación, no siendo necesaria traducción alguna porque se crean internamente y no salen al exterior. Figura 2-4.

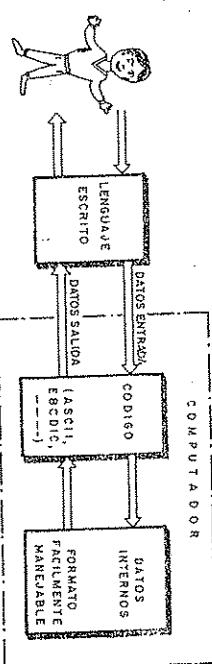


Fig. 2-4. Movimiento y transformación de los distintos tipos de datos dentro del computador.

2.2. CAPACIDAD DE REPRESENTACIÓN DEL COMPUTADOR

Los datos del computador se guardan en sus registros internos y en posiciones de memoria, constituidos por elementos materiales que determinan el espacio material. Así los datos se verán condicionados por las siguientes características:

1. Cada elemento material (registros o memoria) sólo permite diferenciar dos estados, 0 y 1, o sea, es de carácter binario.
2. El número de bits reservado para cada dato debe ser finito, por lo que existirá un número finito de representaciones.
3. Los datos se transmiten por el computador a través de unas vías de comunicación (Bus de Datos) que están configuradas por un número determinado de líneas. El número de dichas líneas, que está relacionado con el tamaño de los

elementos materiales, definen el tamaño privilegiado, que es el tamaño en bits de la información para la que se ha diseñado el funcionamiento de la máquina.

Los tamaños privilegiados constituyen una de las características más importantes del espacio material de representación en un computador y su selección es una de las acciones prioritarias en la definición de su arquitectura.

Un computador puede tener diferentes tamaños privilegiados, entre los que sobresalen:

- **Byte o carácter:** Es el espacio utilizado para representar un carácter alfanumérico. Suele ser de 8 bits, aunque hay computadores que tienen el byte de 6 bits.
- **Palabra:** Es el tamaño de la información, medido en bits, que trata el computador en paralelo. Consta, generalmente, de 4 o 2 octetos, según el sistema sea más o menos potente. Hay microprocesadores comerciales cuyo ancho de palabra es de 4, 8 y 12 bits.
- **Media palabra:** Existen modelos de máquina que pueden trabajar también con media palabra.

COMPUTADOR	PALABRA	BYTE	TAMAÑOS PRIVILEGIADOS
CDC 6600	60	6	60, 120
3 6700	48	6, 8	4, 6, 8, 48, 96
UNIVAC 1100	36	6	6, 9, 12, 18, 36, 72
IBM 370	32	8	4, 8, 16, 32, 64
VAX	32	8	8, 16, 32, 64
PDP - 11	16	8	8, 16
8080	8	8	8
M 68000	16	8	8, 16, 32

Fig. 2-5. En todo computador existe una relación natural entre los distintos tamaños privilegiados que soporta, siendo la palabra el tamaño principal.

- **Doble palabra:** Ciertos computadores, para mejorar la precisión de cálculo, admiten operar con dobles palabras, aunque suelen tratar a estos datos en dos pasos.

La palabra es el tamaño privilegiado principal de un computador, mientras que el byte y otros tamaños menores deben dividirlo de forma exacta. Los tamaños mayores han de ser múltiplos de la palabra, existiendo una "relación natural" entre los diversos tamaños privilegiados, como puede apreciarse en el cuadro de la figura 2-5.

Cuanto menor sea el tamaño de la información que pueda tratar directamente el computador, mayor será su resolución. Así, un computador que pueda manejar los datos a nivel de bit tendrá más resolución que otro que pueda procesar datos, como mínimo, de un octeto.

Disponen de resolución de 1 bit computadores como el IBM Stretch, Burroughs B 1700 y CDC Star; de 6 bits, el Univac 1600 y el PDP-10; de 8 bits, el IBM 370, el PDP-11 y el VAX; de 16 bits, el PDP-10 y de 36 bits, el IBM 7094.

2.3. TIPOS DE REPRESENTACIÓN

Los diversos tipos de representación existentes quedan clasificados, teniendo en cuenta los siguientes factores:

1. Elemento a representar (número, carácter alfanumérico, ...).
2. Forma de representación (binario puro, coma flotante, ...).
3. Características de representación (tolerante a fallos, con paridad, ...).

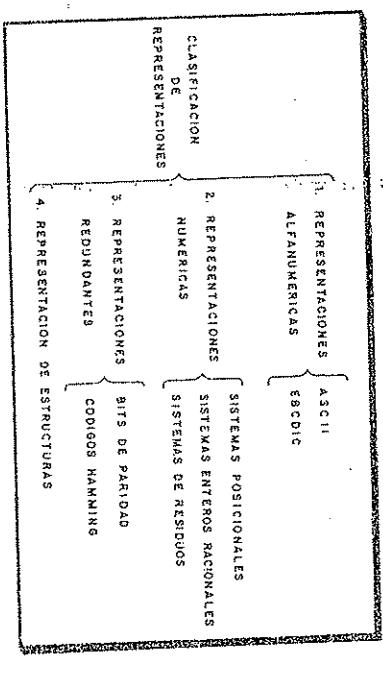


Fig. 2-6. Clasificación de las representaciones que soportan los computadores.

En el cuadro de la figura 2-6 se ofrece una posible clasificación de las representaciones. Debe tenerse en cuenta que las representaciones numéricas que soporta una máquina, de las que existen muchas alternativas, constituyen una de las características más influyentes sobre la arquitectura de la misma. El diseñador ha de elegir las representaciones numéricas más apropiadas para el uso al que se destine al computador.

2.4. DATOS ALFANÚMERICOS

Cada carácter de una cadena de datos alfanumérica se codifica en un byte. Según el número de bits que componga cada byte, el número de caracteres representables será mayor o menor. Actualmente se utilizan códigos de 7 y 8 bits. Como muestra la figura 2-4, este tipo de codificación se emplea en la entrada y salida de datos del computador, estando preparados todos sus periféricos para trabajar con ella. También se emplea internamente en el computador cuando se trata de una información en forma de texto.

En la primera y segunda generaciones de computadores, se emplearon sistemas de 6 bits por carácter, con lo que se podían representar 64 caracteres distintos (26 letras, 10 números, 28 caracteres especiales y los símbolos de puntuación). Ejemplos de estos sistemas son el BCDIC, el Fieldata y el X-3, que se muestra en la figura 2-7.

FORMATO DE BITS	00	01	10	11
ESPACIO	ε	:	*	#
0000	3	:	*	%
0010	-	+	\$,
0011	0	?	!	+/-
0100	1	A	J	/
0101	2	B	K	S
0110	3	C	L	T
0111	4	D	M	U
1000	5	E	N	V
1001	6	F	O	W
1010	7	G	P	X
1011	8	H	Q	Y
1100	9	I	R	Z
1101	<	J	H	
1110	=	K	M	
1111	=	L	N	

Fig. 2-7. Sistema de representación alfanumérica X-3 de 6 bits.

50 / Sistemas de representación de la información

COLUMNA	0	1	2	3	4	5	6	7
FILA								
SBITS	4321	7654	21000	00000	00000	01010	01110	11010
	↓	↓	↓	↓	↓	↓	↓	↓
0								
1	0001		SOH	DC1	!>	1	A	Q
2	0010		STX	DC2	">	2	B	R
3	0011		ETX	DC3	#>	3	C	S
4	0100		EOT	DC4	\$>	4	D	T
5	0101		ENQ	NAK	%>	5	E	U
6	0110		ACK	SYN	&>	6	F	V
7	0111		BEL	ETB	'>	7	G	W
8	1000		BS	CAN	(>	8	H	X
9	1001		HT	EM1)>	9	I	Y
10	1010		LF	SUB	*>	10	J	Z
11	1011		VT	ESC	+>	11	K	l
12	1100		FF	FS	,>	12	L	\
13	1101		CR	GS	=>	13	M	m
14	1110		SO	RS	>	14	N	n
15	1111		SI	US	/>	15	O	o

Ejemplo: Código para A = 7 → 00110001

Significado en inglés de las leyendas de las columnas 0 y 1

NULL	Null	DLE	Data Link Escape
SOH	Start of Heading	DC1	Device Control 1
STX	Start of Text	DC2	Device Control 2
ETX	End of Text	DC3	Device Control 3
EOT	End of Transmission	DC4	Device Control 4
ENQ	Enquiry	NAK	Negative Acknowledge
ACK	Acknowledge	SYN	Synchronous Idle
BEL	Bell (auditive signal)	ETB	End of Transmission Block
BS	Backspace	CAN	Cancel
HT	Horizontal Tabulation (punched card skip)	EM	End of Medium
LF	Line Feed	SUB	Substitute
VT	Vertical Tabulation	ESC	Escape
FF	Form Feed	FS	File Separator
CR	Carriage Return	GS	Group Separator
SO	Shift Out	RS	Record Separator
SI	Shift In	US	Unit Separator

Fig. 2-8. Listado completo del código ASCII, de 7 bits. Las columnas 0 y 1 de la figura representan los caracteres de control de periféricos.

Sistemas de representación de la información / 51

COLUMNA HEX	0	1	2	3	4	5	6	7	8	A	B	C	D	E	F
FLG (HEN)	457	0123	4567	0001	0001	0001	0001	0001	0001	0001	0001	0001	0001	0001	0001
0	0000	NUL	DEL	DLE	DS	SP	&	-							
1	0001	SOH	DC1	SOS					a	j	-	A	J	1	
2	0010	SIX	DC2	FS	SYN				b	k	s	B	K	S	2
3	0011	ETX	DC3					c	l	i	C	L	T	3	
4	0100	PF	RUE	BYP	PN			d	m	u	D	M	U	4	
5	0101	HBT	NL	LFC	RS			e	n	v	E	N	V	5	
6	0110	ESC	BS	ETB	SCI			f	o	w	F	O	W	6	
7	0111	DEL	IL	PRE	EOT			g	p	x	G	P	X	7	
8	1000	VT						h	q	y	H	Q	Y	8	
9	1001	RLF	EM					\	i	r	z	I	R	Z	9
A	1010	SM1	CC	SM	\$!	:								
B	1011	HT													
C	1100	FF	IPS	DC4	<	*	%	@							
D	1101	CR	IGS	EXQ	NAK	()								
E	1110	SO	IR5	ACK	*	:	>	*							
F	1111	SI	DC5	GET	SUB	I	-	?							

Ejemplo. Código para letra D = 0...11101101

Significado en inglés de las columnas 0 y 4

NUL	Null	CC	Cursor Control
SOH	Start of Heading	IFS	Interchange File Separator
STX	Start of Text	IGS	Interchange Group Separator
EOT	End of Text	IRS	Interchange Record Separator
PF	Punch Off	IUS	Interchanging Unit Separator
HT	Horizontal Tab	DS	Device Select
LC	Lower Case	SOS	Start of Significance
DEL	Delete	FS	Field Separator
RLF	Reverse Line Feed	BYP	ByPass
SMH	Start of Message	LF	Line Feed
VT	Vertical Tabulation	EOB/EOT	End of Block/End of Transmission Block
FF	Form Feed	PRE/ESC	Prefix/Escape
CR	Carriage Return	SM	Set Mode
SO	Shift Out	ENQ	Inquiry
SI	Shift In	ACK	Acknowledge
DLE	Data Link Escape	BEL	Bell
DC1	Device Control 1	SYN	Synchronous Idle
DC2	Device Control 2	PW	Batch On
DC3	Device Control 3	RS	Render Stop
RES	Restore	UC	Upper Case
NL	New Line	EOT	End of Transmission
BS	Backspace	DC4	Device Control 4
IL	Julie	NAK	Negative Acknowledge
CAN	Cancel	SUB	Substitute
EM	End of Medium	SP	Space

Fig. 2-9. Listado completo del código EBCDIC, de 8 bits.

52 / Sistemas de representación de la información

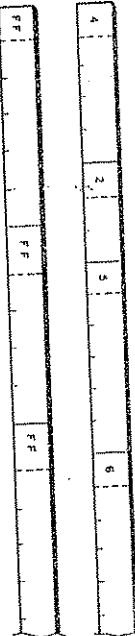


Fig. 2-10. Las cadenas de longitud variable pueden definir su longitud, bien indicando la misma al principio de cada dato (figura superior), o bien, separando los datos con un símbolo específico (figura inferior).

Sistemas de representación de la información / 53

A partir de la tercera generación de computadores y como consecuencia de la necesidad de representar letras mayúsculas y minúsculas y de disponer de caracteres especiales para el control de periféricos, surgieron los códigos de 7 bits, como el ASCII, muy usado en minicomputadores y microcomputadores, y de 8 bits, como el EBCDIC, introducido en 1964 con el IBM 360. Figuras 2-8 y 2-9.

En todos los sistemas de codificación se ha procurado cumplir tres reglas muy útiles:

1. Intentar que el reconocimiento de los caracteres que representan números sea muy sencillo, dada la repetición de operaciones de traducción desde otros métodos de representación numéricos.
2. Que los códigos para las letras mayúsculas difieran sólo en un bit con el de las letras minúsculas, para que ciertos periféricos (impresoras y terminales) que no tengan minúsculas interpreten los caracteres con facilidad.
3. Distinguir claramente los tipos de caracteres numéricos, alfabéticos y de control para simplificar los programas de interpretación de dichos elementos.

Las cadenas de caracteres alfanuméricas, manejadas y almacenadas por el computador, pueden ser:

- De longitud fija: Cada dato ocupa un determinado número de bytes fijo. Por lo tanto, puede conocerse rápidamente el comienzo y el final de cada dato.
- De longitud variable: Para determinar el final de cada dato existen dos métodos:
 - Cada dato tiene en su inicio un campo en el que se indica la longitud.
 - Cada dato se separa de los colindantes mediante un símbolo específico. Figura 2-10.

2.5. GENERALIDADES SOBRE LOS DATOS NUMÉRICOS

Los números reales se clasifican en tres tipos:

1. **ENTEROS.** Positivos y negativos.
2. **RACIONALES.** Se pueden expresar como cociente de dos enteros.

3. **IRRACIONALES.** Resto de reales que no pueden expresarse como cociente de dos enteros; es el caso del número π .

Cualquier conjunto de números reales es *infinito*, por lo que, dadas las limitaciones del espacio material de representación de los computadores, no es posible representar a todos. Por el mismo motivo, serán irrepresentables los números irracionales, que precisan de un número infinito de bits.

Generalmente, en un computador se asigna un número fijo de n bits para representar un número, siendo n el tamaño de la *palabra* o otro tamaño privilegiado. Esta limitación define el *rango* de representación, o sea, los límites. Consecuentemente, los números racionales se expresarán con un número finito de cifras fraccionarias, lo que dará lugar a una cierta precisión en la representación.

Dada esta limitación de bits, existen diversos métodos de representar los datos numéricos que optimizan la utilización de estos bits para determinadas aplicaciones.

A continuación se estudian los métodos más corrientes de representación de enteros y racionales en el sistema *posicional*, que pueden dividirse en los siguientes grupos:

- a) *Como fija sin signo.* Positivos enteros.
- b) *Como fija con signo.* Enteros.
- c) *Como fija con complemento a la base.* Enteros.
- d) *Como fija con complemento restringido a la base.* Enteros.
- e) *Como fija, BCD.* Enteros.
- f) *Como flotante normalizada.* Racionales.
- g) *Como flotante no normalizada.* Racionales.

2.6. MÉTODOS POSICIONALES

Los números se representan en dígitos sucesivos, teniendo cada dígito un peso o valor que depende del lugar donde se encuentra. Los pesos son potencias sucesivas de una base.

- 54 / Sistemas de representación de la información

El valor de un número formado por los dígitos:

$$x_n \dots x_3 x_2 x_1 x_0, x_{-1} x_{-2} \dots x_{-m}$$

tomando como pesos las potencias sucesivas de una base r :

$$r^n \dots r^3 r^2 r^1 r^0, r^{-1} r^{-2} \dots r^{-m}$$

será:

$$\begin{array}{l} \text{VÍDEO} \\ \text{MONITOR} \\ \rightarrow X = x_n r^n + \dots + x_3 r^3 + x_2 r^2 + x_1 r^1 + x_0 r^0 + x_{-1} r^{-1} + x_{-2} r^{-2} + \dots \\ + x_{-m} r^{-m} \end{array}$$

de donde se deduce que el valor del número se puede expresar simplificadamente de la siguiente forma:

$$\sum_{i=-m}^n r^i \cdot x_i$$

Por ejemplo, el número 913,4 en base 10 ($r = 10$), se puede descomponer:

$$913,4 = 9 \times 10^2 + 1 \cdot 10^1 + 3 + 4 \cdot 10^{-1}$$

El número binario 1011, se descompondrá:

$$1011 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 1 = 11_{10}$$

Los dígitos x_i son positivos y menores al valor de la base ($0 \leq x_i \leq r-1$). En el sistema decimal con $r = 10$, $0 \leq x_i \leq 9$ y en el sistema binario con $r = 2$; $0 \leq x_i \leq 1$.

Si dos bases r_1 y r_2 cumplen la condición $r_1 = r_2^p$, entonces:

$$(x_n \dots x_1 x_0, x_{-1} \dots x_{-m})_{r_2} = (A_m \dots A_1 A_0, A_{-1} \dots A_{-m})_{r_1}$$

siendo:

$$A_j = (x_{pj+p-1} \dots x_{pj+1} x_{pj})$$

De esta propiedad se deduce que los dígitos de la base r_1 se obtienen agrupando los dígitos de la base r_2 en grupos de longitud p , empezando a partir de la coma. Cada una de estas cadenas de la base menor es un número de la base mayor. Esta propiedad se puede aplicar para pasar un número binario a octal, agrupando los bits de 3 en 3, o a hexadecimal, agrupando los bits de 4 en 4.

Ejemplo: Pasar a hexadecimal el número binario siguiente:

$$\overbrace{(101101110,101)}_p = (16E,B)_{16}$$

$p = 4$, puesto que $2^4 = 16$

A veces, un número racional tiene una representación exacta en una base, mientras que en otra es de tipo periódica; por ejemplo:

$$(1/5)_{10} = (0.2)_{10} = (0.001100110011)_2$$

Al tener que encajar los números en un tamaño fijo de n bits, se trunca o pierde el período y el computador produce resultados con un cierto error.

2.7. COMA FIJA SIN SIGNO. BINARIO PURO SIN SIGNO

Es el formato más simple para representar números enteros positivos y responde a la siguiente forma:

$$x_{n-1} x_{n-2} \dots x_2 x_1 x_0$$

sabiendo X el valor que expresa:

$$X = 2^{n-1} x_{n-1} + 2^{n-2} x_{n-2} + \dots + 2^2 x_2 + 2^1 x_1 + 2^0 x_0 = \sum_{i=0}^{n-1} 2^i x_i$$

El rango o los valores que admite un número X expresado en este formato, son los enteros positivos:

$$0 \leq X \leq 2^n - 1$$

Modificando el bit menos significativo, el número variará en una unidad, y, por lo tanto, la precisión es de una unidad.

Ejemplo:

$$10_{10} = 1010_2$$

Este formato tiene el inconveniente de que, al realizar sumas, se puede perder el bit más significativo del resultado cuando no cabe en el tamaño privilegiado de trabajo (desbordamiento). El computador puede disponer de un mecanismo para avisar cuando se produce esta eventualidad.

Ejemplo: Siendo $n = 4$.

$$\begin{array}{r} 1101 \\ + 1111 \\ \hline 101010 \end{array}$$

También se pueden producir desbordamientos de capacidad cuando se realizan productos. Los computadores tienen *señalizadores*, a veces llamados "restauradores de estado", que indican estas circunstancias.

Si se emplean 4 dígitos, los números que se pueden representar en este sistema y sus correspondientes valores en decimal son los siguientes:

BINARIO PURO SIN SIGNO	DECIMAL
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

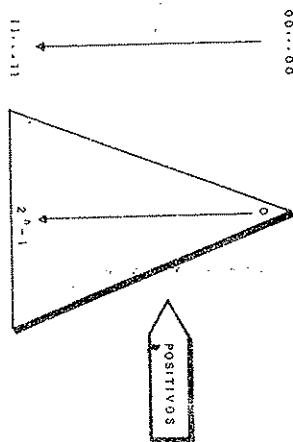


Fig. 2-11. Representación del rango de valores que alcanza el sistema de coma fija sin signo, con n bits.

En la figura 2-11 se muestra de forma gráfica el rango de valores que se pueden alcanzar con el sistema binario puro sin signo.

2.7.1. Binario puro con signo

Es igual al formato anterior, pero reservando un bit para indicar el signo. Este bit, si es un 0, define al número como positivo y si es un 1, como negativo. Figura 2-12.

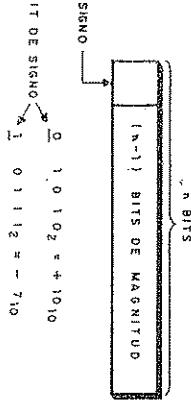


Fig. 2-12. Representación gráfica de un número en coma fija con signo y ejemplos.

El mayor número, en valor absoluto, que se puede formar teniendo en cuenta que uno de los n bits es el de signo, es $2^{n-1} - 1$. Por tanto, el rango es:

$$[-(2^{n-1} - 1) \leq X \leq 2^{n-1} - 1] \equiv [-2^{n-1} + 1 \leq X \leq 2^{n-1} - 1] \quad x \leq 1$$

La precisión sigue siendo la unidad.

El valor del número X representado por la cadena:

$$\{x_{n-1}, x_{n-2}, \dots, x_2, x_1, x_0\}$$

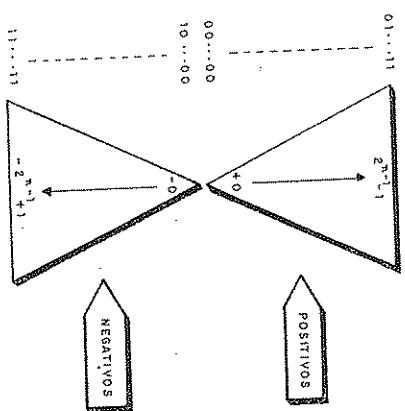
será:

$$\left. \begin{array}{l} \text{si } x_{n-1} = 0, \quad X = \sum_{i=0}^{n-2} 2^i x_i \\ \text{si } x_{n-1} = 1, \quad X = -\sum_{i=0}^{n-2} 2^i x_i \end{array} \right\} X = (1 - 2 \cdot x_{n-1}) \sum_{i=0}^{n-2} 2^i \cdot x_i$$

Además de los problemas comentados para el sistema de coma fija sin signo, en éste se presentan otros dos:

1. Existen dos "ceros": el 00 ... 00 (0) y el 10 ... 00 (-0).
2. Cuando uno de los operandos es negativo, una suma se convierte en una resta.

Fig. 2-13. Representación gráfica del rango de los números con coma fija con signo.



2.8. COMA FIJA CON COMPLEMENTO RESTRINGIDO A LA BASE. COMPLEMENTO A UNO

En este sistema los números positivos coinciden con los correspondientes a los sistemas antes descritos. Para formar los números negativos se aplica el siguiente algoritmo:

$$N_0 \text{ que representa a } (-A) \rightarrow b^n - 1 - A,$$

siendo b la base y n el número de dígitos utilizado.

Como los computadores funcionan con números binarios, el algoritmo quedará convertido en:

$$C1 = 2^n - 1 - A$$

Siendo $C1$ el número expresado en complemento a 1.

Ejemplo:

$$\begin{aligned} 10_{10} &\rightarrow 01010_2 \\ -10_{10} &\rightarrow 100000 - 1 = 01010 = 10101 \end{aligned}$$

El complemento a 1 coincide con el complemento "lógico" o negación, es decir, para formarse se cambian los 1 por 0 y viceversa.

Los valores que pueden formarse en este sistema son:

<i>COMPLEMENTO A 1</i>	<i>DECIMAL</i>
00 ... 00	+ 0
...
01 ... 11	$2^{n-1} - 1$
10 ... 00	$-(2^{n-1} - 1)$
...
11 ... 11	-0

Ejemplo:

$$\begin{array}{r}
 11100 \\
 + 10111 \\
 \hline
 10101 \rightarrow 1
 \end{array}$$

En la figura 2-14 se muestra gráficamente el rango de valores que pueden alcanzarse con el sistema de complemento a 1, deduciéndose que el mismo viene dado por:

$$[-(2^{n-1} - 1) \leq X \leq 2^{n-1} - 1] \equiv [-2^{n-1} + 1 \leq X \leq 2^{n-1} - 1]$$

dentro del cual existen dos "ceros".

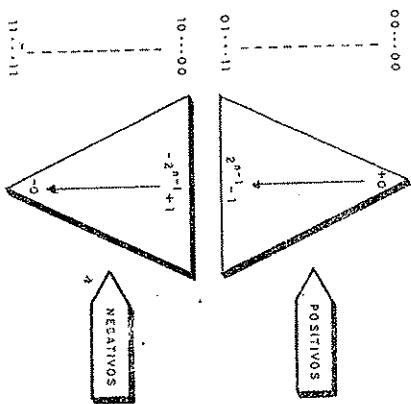


Fig. 2-14. Representación del rango de los números expresados en complemento a uno.

- 2) Lo mismo ocurre al sumar un número positivo con otro negativo, siendo el valor absoluto del positivo igual o mayor que el del negativo.

$$\begin{array}{r}
 -9 \\
 -4 \\
 \hline
 \div 5 \\
 4 \\
 \hline
 1 \\
 + 1 \\
 \hline
 00101
 \end{array}$$

Como regla general, siempre que se produzca acarreo superior ha de incrementar-se el resultado.

2.9 COMA FIJA CON COMPLEMENTO A LA BASE. COMPLEMENTO A DOS

Este formato es parecido al de complemento a 1 con la diferencia de que los números negativos se representan restando de 2^n el valor absoluto del número, siendo n el número de bits empleados para expresar el número.

- 1) Al sumar dos números negativos el resultado es una unidad inferior al correcto y, además, se produce un bit de acarreo superior C_{n-1} . La corrección se efectúa sumando el bit de acarreo C_{n-1} al resultado. Sin embargo, estas correcciones son fáciles de realizar por la ALU, por lo que se emplean en algunos computadores.

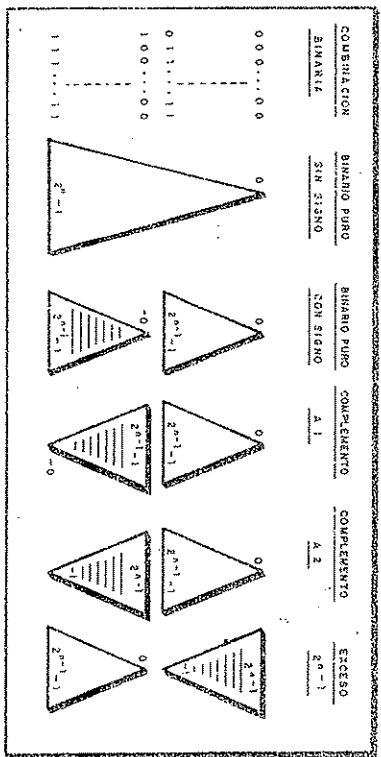


Fig. 2-16. Representación gráfica de los rangos de los diferentes sistemas estudiados.

2.11. FORMATOS DE REPRESENTACIÓN DECIMAL

Los datos de entrada y salida del computador suelen expresarse en base decimal, excepto cuando se dirigen o se reciben de periféricos de almacenamiento magnético, en cuyo caso se utiliza el sistema binario. Cuando se precisan realizar cálculos complejos, hay que transformar los datos de entrada a binario, mientras que los de salida hay que convertirlos a decimal desde el binario.

De todas formas, hay situaciones en las que conviene trabajar con formatos decimales. Es el caso de una aplicación de contabilidad en la que hay que aplicar ciertas normas de redondeo. Al operar en binario, los errores de redondeo son diferentes a los obtenidos usando el sistema decimal.

Como el sistema decimal utiliza los números comprendidos entre el 0 y el 9, para representarlos en binario se necesitan 4 bits. Por lo tanto, se desaprovechan 6 de las 16 posibles combinaciones que se pueden hacer con 4 bits.

Cada número decimal se representa por una de las 16 combinaciones, la cual dependerá del código elegido. El más usual es el *BCD*, que asigna a cada dígito decimal su valor correspondiente en binario. Figura 2-17.

Otro código es el *2-4-2-1*. Su mismo nombre indica el peso de cada bit.

En telefonía se usa el código llamado *2 entre 5*, en el que dos bits siempre están a 1 y tres a 0, sea cual sea el dígito representado. Con este procedimiento se detectan fácilmente errores en la transmisión.

Un problema que surge cuando se opera con dígitos *BCD*, es que los acarreos no se generan siempre que se deban originar. Por ejemplo, $7 + 4 = 11$ que, al ser un número menor que 16, se puede representar con 4 bits y no produce acarreo. Este motivo hace recomendable utilizar, en ocasiones, el código de exceso a 3, en el cual, si el resultado real es 10 en una suma, se representará como $16_{10} = 10_{16}$, lo que supone la generación del acarreo.

DIGITO DECIMAL	B.C.D.	EXCESO 3	2-4-2-1
0	0 0 0 0	0 0 1 1	1 1 0 0 0
1	0 0 0 1	0 1 0 0	0 0 0 1 1
2	0 0 1 0	0 1 0 1	0 0 1 0 1
3	0 0 1 1	0 1 1 0	0 0 1 1 0
4	0 1 0 0	0 1 1 1	0 1 0 0 1
5	0 1 0 1	1 0 0 0	0 1 0 1 0
6	0 1 1 0	1 0 0 1	0 1 1 0 0
7	0 1 1 1	1 0 1 0	1 0 0 1 1
8	1 0 0 0	1 0 1 1	1 0 0 1 0
9	1 0 0 1	1 1 0 0	1 0 1 0 0

Fig. 2-17. Representación de los dígitos decimales en los códigos más usuales.

El código exceso a 3 exige correcciones al sumar dos números $x + y$:

- 1.º Cuando $x + y \leq 9$, el resultado debe quedar representado como $x + y + 3$ pero se obtiene $x + 3 + y + 3 = x + y + 6$. Por lo tanto, basta con restar 3 al resultado de la suma para hallar el valor correcto.

Ejemplo:

$$\begin{array}{r} 6 \dots 101 \\ + 3 \dots 0110 \\ \hline 9 \quad 1111 \end{array}$$

$$\begin{array}{r} -0011 \\ \hline 1100 \end{array}$$

2.ª Cuando $x + y > 9$, el resultado, con acarreo, es $x + y + 6 = 16$ (+6 por los excesos en 3 de x e y , $y - 16$ por producirse acarreo). El resultado deseado es el acarreo y ($x + y + 3$), por lo que se deben sumar 3 unidades al resultado.

Ejemplo:

$$\begin{array}{r} 8 \dots 1011 \\ + 8 \dots 1011 \\ \hline 16 \quad 10110 \\ + 11 \\ \hline 11001, \text{ con acarreo} \end{array}$$

Los datos del exterior suelen introducirse al computador en código alfanumérico con formato de carácter. Para aprovechar memoria, estos caracteres, cuando son números, pueden *empaquetarse*, o sea, ser representados empleando sólo 4 bits por cada dígito. En general, se emplean números de longitud variable, separados entre sí mediante los caracteres de signo o por otro carácter especial.

Ejemplo: Representación de números con bytes de 8 bits y codificación EBCDIC.

$$\left\{ \begin{array}{l} \text{Desempaquetado: F3 F4 F5 C6} \\ \text{Empaquetado: } 03 \ 45 \ 6C \ \boxed{} \text{ signo +} \end{array} \right.$$

$$\left\{ \begin{array}{l} \text{Desempaquetado: F3 F4 F5 D6} \\ \text{Empaquetado: } 03 \ 45 \ 6D \ \boxed{} \text{ signo -} \end{array} \right.$$

2.12. COMA FLOTANTE.

Los sistemas de representación numérica con coma fija sólo pueden expresar números enteros.

Escalando adecuadamente las cantidades, se puede representar la coma y así expresar números racionales. La labor de escalado la realiza el computador, automáticamente, de forma dinámica y óptima. Así surgen los sistemas de representación en coma flotante, cuya propiedad primordial es concatenar con cada número un factor de escala.

El empleo de la coma flotante se remonta a los primeros computadores, pero cayó en desuso por la idea propuesta por J. von Neumann, que consideró trivial el problema del escalado. Hoy en día, la mayor parte de los computadores permiten trabajar de este modo.

La representación en coma flotante divide los n bits en dos partes:

a) *Mantisa*: Contiene los dígitos significativos del número.

b) *Exponente*: Indica el factor de escala, como una potencia de la base r .

La mantisa y el exponente tienen unas longitudes respectivas de p y q bits, de forma que $p + q = n$.

En coma flotante un número A viene expresado:

$$A = \text{mantisa} \cdot \text{base}^{\text{exponente}}$$

En la práctica, la base que se eleva al exponente es la base en la que se representa la mantisa. De esta manera, el exponente indica directamente el número de dígitos o posiciones que debe correrse la coma de la mantisa.

Ejemplo:

$$\begin{aligned} 0,932 \times 10^2 &= 093,2 \text{ (2 posiciones a la derecha)} \\ 0,932 \times 10^{-2} &= 0,00932 \text{ (2 posiciones a la izquierda)} \\ 0,11101 \times 2^4 &= 0111,01 \end{aligned}$$

$$0,11101 \times 2^{-4} = 0,00011101$$

La mantisa (M) y el exponente (E) se representan en alguno de los sistemas de coma fija ya comentados, de forma que la base de representación de M coincida con la base de la escala r , esto es, $b_M = r$, por lo que $r = 2^k$ pues es el único modo de ajustar fácilmente el exponente. En efecto, en este caso un desplazamiento a la izquierda o derecha de la coma, corresponde a un decreto o incremento de E , respectivamente. Los valores de r más usados en la práctica son 2 y 16.

La mantisa se representa como entero, con la coma a la derecha del dígito m_0 (mantisa entera), o como fracción, con la coma a la izquierda (mantisa fracción).

Ejemplo:

$$\begin{aligned} r &= 10 & 879 \cdot 10^{-4} & (\text{mantisa entera}) \\ & & 0,879 \cdot 10^{-1} & (\text{mantisa fracción}) \\ r &= 2 & 0,11011 \cdot 2^5 & (\text{mantisa fracción}) \\ & & 11011 \cdot 2^{-1} & (\text{mantisa entera}) \end{aligned}$$

El exponente E se representa en base 2 ($b_E = 2$) y en exceso 2^{q-1} , siendo q la longitud de E .

Ejemplo: Representación en coma flotante.

Características:

- Mantisa entera representada en la forma de signo y magnitud
- $r = 2$
- Exponente representado en exceso 32
- $n = 16$
- $q = 6$
- $p = 10$

EXPONENTE	MANTISA
100000	000000101 = $5 \cdot 2^0 = 5$
100000	100000101 = $-5 \cdot 2^0 = -5$
100010	0000000101 = $5 \cdot 2^2 = 20$
011100	1000000101 = $-5 \cdot 2^{-4} = -5/16$

En el caso de los computadores PDP-11 y VAX, las características de representación en coma flotante son:

- $n = 32$
- $r = 2$
- $q = 8$

$p = 25$ (tiene un bit implícito por lo que $p + q = n + 1$)

Exponente codificado en exceso a 128

Mantisa fracción codificada en signo magnitud

Las características de este tipo de representación en el caso del computador IBM 370 son:

- $n = 32$
- $r = 16$
- $q = 7$
- $p = 25$

Exponente codificado en exceso a 64

Mantisa fracción codificada en signo magnitud.

2.13. COMA FLOTANTE CON MANTISA ENTERA

El exponente se representa en exceso 2^{n-1} y la mantisa en binario puro con signo.

Ejemplo:

$$\begin{aligned} M &= 001000 & E &= 101 \dots x = 8 \cdot 2^1 = 16 \\ M &= 101000 & E &= 001 \dots x = -8 \cdot 2^3 = -8/8 = -1 \end{aligned}$$

El rango de expresión que se alcanza con este sistema es:

- Mantisa positiva menor = 00 ... 01 = 1₁₀
- Exponente menor = 00 ... 00 = (-2^{q-1})₁₀ ($q = n$.º bits del exponente)
- Número positivo menor = 1₁₀ · 2^{q-1}

— Mantisa positiva mayor = 011 ... 11 = (2^{p-1} - 1)₁₀ ($p = n$.º de bits de la mantisa)

— Exponente mayor = 11 ... 11 = (2^{q-1} - 1)₁₀

— Número positivo mayor = (2^{p-1} - 1) · 2^{q-1}

Si un número X es positivo, estará dentro del rango:

$$r^{2^{q-1}} \leq X \leq (2^{p-1} - 1) r^{2^{q-1}}$$

Si el número es negativo, estará comprendido en el intervalo:

$$-(2^{p-1} - 1) r^{2^{q-1}-1} \leq X \leq -r^{2^{q-1}}$$

Este sistema no es muy utilizado, siendo mucho más frecuente el de coma flotante con mantisa fracción, que se estudia a continuación.

2.14. COMA FLOTANTE CON MANTISA FRACCION.

NORMALIZACION

La mantisa suele normalizarse y para ello debe tenerse en cuenta:

1. Los dígitos nulos a la derecha inmediata de la coma se eliminan ajustando el valor al exponente.

Ejemplo:

$$0,00101 \cdot 2^3 = 0,101 \cdot 2^1$$

2. En base 2, el dígito primero es uno, por lo que puede no representarse quedando como *bit implícito*. De esta forma, la mantisa tiene un bit más de los que pueden almacenarse en memoria.

Si el número de dígitos de la mantisa es p , la precisión de ésta será:

$$2^{-(p-1)}$$

que es el peso del bit menos significativo, considerando que un bit está ocupado por el signo.

La precisión total para una base $r = 2^k$ será:

$$2^{-(p-1)} \cdot r^e = 2^{-(p-1)} \cdot 2^k \cdot e = 2^{1-p+ke}$$

De aquí se deduce que la precisión no es uniforme, por depender del exponente "e".

Si un dato se representa por el número más cercano, el dato representado x' disminuirá, como mucho, del real x , la mitad del intervalo entre dos representaciones sucesivas. Figura 2.18.

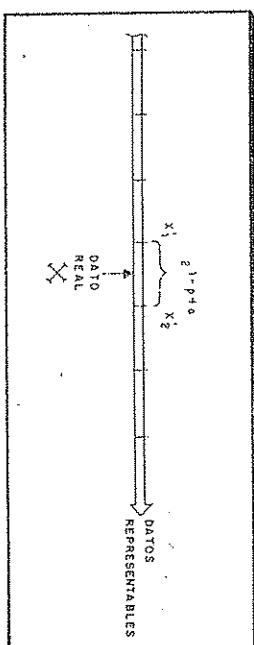


Fig. 2.18. El dato representado, como mucho, dista del real un valor equivalente a la mitad del intervalo entre representaciones.

Se deduce que el error máximo en la representación será:

$$e_a = |x'_1 - x'_2| = 2^{1-p+ek} \cdot 2^{-1} = 2^{ek-p}$$

Este será el error absoluto máximo que depende del exponente. El error relativo se puede acotar como sigue:

$$e_r \leq \frac{e_a}{x} \cong \frac{e_a}{x'} = \frac{2^{ek-p}}{M \cdot 2^a}$$

empleando el caso peor de la mantisa M , esto es, con $M = 1/r = 2^{-k}$,

70 / Sistemas de representación de la información

2.15. ESTANDAR IEEE P754

Según esta normalización, el número de dígitos reservados para representar un número admite dos variantes:

1. Formato de simple precisión.
2. Formato de doble precisión.

SIMPLE PRECISION		
SIGNO	EXPONENTE	MANTISA
0	1 69	31

DOBLE PRECISION		
SIGNO	EXPONENTE	MANTISA
0	1 11 12	63

Fig. 2.19. Formatos básicos de representación según el estandar IEEE P754.

Las propiedades fundamentales de esta norma son:

- a) Una mantisa fraccionaria normalizada, sin ceros a la derecha de la coma. En binario puro, se representa sin signo.
- b) El exponente se representa en exceso $2^{q-1} - 1$, es decir, en exceso 127, con simple precisión, y en exceso 1023, con doble precisión.
- c) Cuando $E = 255$ y $M \neq 0$, el resultado es erróneo. Este caso, se produce, por ejemplo, en una división 0/0.
- d) Cuando $E = 255$ y $M = 0$, se indica valor positivo o negativo, según s.

- e) El cero se representa con $E = 0$ y $M = 0$.

- f) Para representar números cercanos a cero se utiliza un formato desnormalizado, haciendo $E = 0$ y $M \neq 0$. En lugar del valor normalizado:

$$(-1)^{\text{signo}} \cdot 2^{\exp-127} \cdot 1, M \quad 0$$

$$(-1)^{\text{signo}} \cdot 2^{\exp-1023} \cdot 1, M$$

el valor que se representa cuando $E = 0$, es:

$$(-1)\text{signo} \cdot 2^{-126} \cdot 0, M$$

Si no fuese por esta última norma, existirían muchos valores próximos a cero que no se representarían. Figura 2-20.

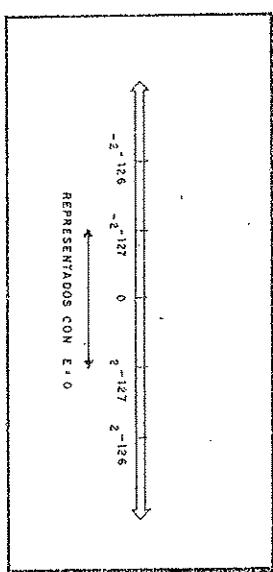


Fig. 2-20. Para representar números cercanos al cero, se usa un formato no normalizado, haciendo $E = 0$ y $M \neq 0$.

Ejemplo: Números representados con el estándar IEEE P 754

$$\begin{aligned} 78F00010_{16} &= 0111000111000000000000010000_2 = \\ &= +2^{-14} \cdot 1,111000000000000000010000 \\ C1800010_{16} &= 10110001100000000000000010000_2 = \\ &= -2^{29} \cdot 1,000000000000000000010000 \end{aligned}$$

2.16. SISTEMAS DE RESIDUOS

Este sistema se basa en el concepto de *módulo* o *residuo* de un número. Siendo una base m_1 , se define como módulo m_1 del número X , el resto o residuo obtenido al dividir X por m_1 . Se expresa como $x_1 = X \bmod m_1$.

Con este sistema se pueden representar hasta m cantidades distintas, por lo que su selección definirá el rango de valores.

Una palabra de longitud finita de n bits, representada en binario puro, equivale a una representación de residuo módulo $m = 2^n$.

El interés de estos sistemas estribaba en el empleo de varias bases simultáneamente.

En el caso de emplear dos, bases m_1 y m_2 , un número vendrá representado por la pareja de residuos con respecto a estas dos bases.

$$X \rightarrow (X \bmod m_1; X \bmod m_2) = (x_1, x_2)$$

En general, para t bases, la representación viene dada por los t residuos correspondientes ($x_1, x_2, x_3, \dots, x_t$). Si los valores de todas las bases son primos entre sí el rango de representación está dado por el producto de las bases $m_1 m_2 m_3 \dots m_t$.

La propiedad más interesante de este sistema es que las operaciones de suma, resta y multiplicación se realizan de forma independiente de los residuos.

Así, la suma de $(x_1, x_2) + (y_1, y_2)$ es: $(x_1 + y_1) \bmod m_1, (x_2 + y_2) \bmod m_2$.

Para demostrarlo hay que tener en cuenta que:

$$x_1 = X \bmod m_1 \quad y_1 = Y \bmod m_1$$

Por tanto:

$$X = Km_1 + x_1 \quad Y = hm_1 + y_1$$

Sumando:

$$X + Y = (K + h)m_1 + x_1 + y_1$$

De donde se deduce:

$$(X + Y) \bmod m_1 = (x_1 + y_1) \bmod m_1$$

Esta propiedad tiene especial trascendencia en el diseño de máquinas rápidas, así como en el cálculo con precisión múltiple. También se aplica en el diseño de unidades aritméticas tolerantes a fallos.

Existen 3 inconvenientes en el uso del sistema de representación de residuos:

1. La conversión desde/hasta base decimal es compleja.
2. Es difícil realizar la comparación de números.
3. La operación de dividir resulta muy complicada.

2.17. CODIGOS DE DETECCION DE ERRORES

El cambio de un bit en el almacenamiento o manipulación de datos origina resultados erróneos. Para detectar e incluso corregir estas modificaciones, se usan códigos con información redundante.

Los tres tipos de códigos de este tipo más conocidos son:

1. Códigos de paridad.
2. Códigos correctores de Hamming.
3. Códigos polinómicos.

2.18. CODIGOS DE PARIDAD

Se comentan algunos de los más importantes.

2.18.1. Paridad vertical simple o a nivel carácter

Al final de cada carácter se incluye un bit, de manera que la suma de "unos" del carácter completo sea par (paridad par) o impar (paridad impar).

Ejemplo

"3" ... 0 0110011 (bits de paridad par)
"1" ... 1 0110001

Si hay dos bits erróneos, este código no detecta el error, pero la mayoría de los errores se producen en un solo bit. Tampoco detecta el error con cualquier número par de bits erróneos.

2.18.2. Paridad horizontal a nivel de bloque

Por cada bloque de caracteres se crea un byte con bits de paridad. El bit 0 será el bit de paridad de los bits 0 del bloque, el bit 1, el de paridad de los bits 1 del bloque y así sucesivamente. Este código se usa conjuntamente con el vertical, constituyendo un código corrector de error. La figura 2-21 presenta un ejemplo de aplicación con paridad par.

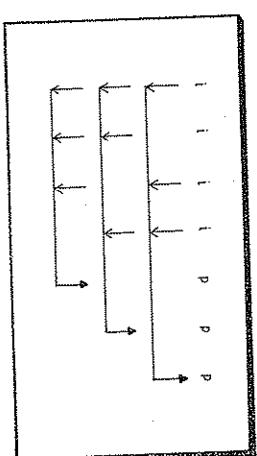
P.V.	0	1	1	0	1	P.H.	0	1	0	1	P.H.
P.V. →	0	1	1	1	1	P.V. →	0	1	1	1	P.V. →
(SIN ERROR)						(CON ERROR)					

Fig. 2-21. Ejemplo de detección de error con paridad horizontal a nivel de bloque. Al producirse el error se detecta en los dos niveles y se puede averiguar cuál es el bit erróneo y corregirlo.

2.18.3. Códigos correctores de paridad entrelazada

Si hay más de un error en la misma fila o columna, quizás se pueda detectar, pero no se podrá determinar con exactitud cuál es el erróneo.

Este sistema de corrección de error no es muy eficaz, aunque sí intuitivo. En práctica se emplean los códigos de Hamming o de paridad entrelazada.



Los códigos de Hamming permiten establecer codificaciones correctoras en base a añadir una serie de bits de paridad entrelazada, que afectan solamente a cierto bits de la palabra de información. La figura 2-22 ofrece un ejemplo de aplicación para palabras de 4 bits de información y 3 de paridad.

Fig. 2-22. Con el código de paridad entrelazado se puede determinar el error cuando es producido por 2 bits erróneos.

2.19. CLAVES DE RELACION CONSTANTE

Son los códigos "M de N", por ejemplo el "4 de 8". Este código tiene 8 bits por carácter: 4 son ceros y 4 son unos. Tienen 70 combinaciones permitiendo la representación de 70 caracteres.

$$\begin{array}{r} 10111101000 \\ \underline{1101} \\ 110001110 \\ \underline{1101} \\ 1101 \\ \underline{00001010} \\ 01110 \\ \underline{1101} \\ 00110 \end{array}$$

2.20. CODIGOS POLINOMICOS

Se utilizan, sobre todo, en las Entradas y Salidas serie. Una secuencia de K bits se representa por un polinomio $M(x)$ de grado $K - 1$.

Ejemplo

La secuencia 10111101 puede representarse por el polinomio:

$$M(x) = x^7 + x^5 + x^4 + x^3 + x^2 + 1$$

La información que se envía no es $M(x)$, sino $T(x)$, que se obtiene por medio del polinomio generador $P(x)$ de grado r.

Esta es la cadena de bits que se envía realmente.

{La suma y la resta coinciden al operar con los códigos polinómicos, pues no se consideran las llevadas o acarreos. Por eso se representan por el símbolo de la operación lógica OR EXCLUSIVA (\oplus)}.

$$T(x) = 10111101000 \oplus 00110 = 10111101110$$

Ejemplo

$$M(x) = 10111101$$

$$P(x) = 1101 \equiv x^3 + x^2 + 1 \quad (r = 3)$$

$$\begin{aligned} x^r M(x) &= x^3 (x^7 + x^5 + x^4 + x^3 + x^2 + 1) = \\ &= x^{10} + x^8 + x^7 + x^6 + x^5 + x^3 \end{aligned}$$

$$\begin{aligned} x^r M(x) &= \underline{\underline{10111101}} \quad 000 \\ M(x) &\text{ se añaden } r \text{ ceros} \end{aligned}$$

Al dividir el polinomio recibido por el polinomio generador, el resto debe ser cero, puesto que $Q(x)$ es un polinomio de resto cero.

2.21. DETECCION Y CORRECCION AUTOMATICAS DE ERRORES

Hay códigos que son lo suficientemente redundantes como para corregir los errores que detecta. Esto implica manejar un elevado número de bits adicionales por cada carácter.

Uno de estos códigos es el de Hamming, que permite corregir errores simples. Por cada 4 bits de información, envía 7.

$P_1, P_2, D_3, P_4, D_5, D_6, D_7$

siendo D_i los bits de información y P_i los de paridad.

P_1 : bit de paridad de D_3, D_5, D_7

P_2 : bit de paridad de D_3, D_6, D_7

P_4 : bit de paridad de D_5, D_6, D_7

Ejemplo

En caso de querer enviar la información: 1010, usando el código Hamming, se enviará 1011010.

Al recibir los 7 bits se comprueban los de paridad. Según los que se encuentren erróneos, se deduce cuál es el bit malo, de acuerdo con la tabla presentada en la figura 2-23.

<u>P_1</u>	<u>P_2</u>	<u>P_4</u>	<u>BIT ERROREO</u>
C	C	C	NINGUNO
E	C	C	P_1
C	E	C	P_2
E	E	C	D_3
C	C	E	P_4
E	C	E	D_5
C	E	E	D_6
E	E	E	D_7

Fig. 2-23. Tabla correspondiente al código de Hamming, que permite corregir errores simples.

78 / Sistemas de representación de la información

2.22. REPRESENTACION DE ESTRUCTURAS DE DATOS

Los datos elementales, cuyos sistemas de representación se han estudiado, se agrupan en estructuras más o menos complejas, tales como vectores, matrices, pilas, árboles, etc.

Dichas estructuras pueden ser *implícitas*, lo que significa que los datos se representan a título individual, y son los programas que los emplean los que tienen que tener en cuenta su tipo y las relaciones entre ellos.

Por ejemplo, una matriz de 16×10 elementos se puede representar como 160 datos consecutivos en coma flotante. Su representación no da información alguna

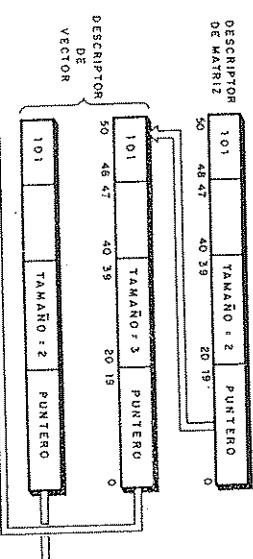


Fig. 2-24. Representación de una matriz en el computador B 6700.

Para acceder a los datos hay que recorrer el camino que indican los descriptores.

Sistemas de representación de la información / 79

sobre el tipo de datos empleado, ni sobre la relación existente entre ellos, ni si su pertenencia a una matriz. Será el programa que usé estos datos el que debe contener esta información.

Esta solución es la más empleada en la mayoría de los computadores, pero algunas máquinas, como la Burroughs B 6700, dota a cada palabra de una etiqueta ("tag") que indica las características del contenido. Dicho modelo emplea etiquetas de tres bits, que permiten diferenciar, entre otras, las siguientes palabras:

- Datos de simple precisión.
- Datos de doble precisión.
- Instrucciones.
- Descriptor de matriz, etc.

Una matriz de 2×3 elementos de simple precisión vendrá almacenada de la forma que se indica en la figura 2.24. Para acceder a los datos se recorre el camino no señalado por los descriptores.

La ventaja de este sistema es la simplificación de las instrucciones, que se hacen independientes del tipo de datos que manejan. Sin embargo, ello se hace a costa de encarecer el computador: mayor ancho de palabra y circuitos más complicados.

EJERCICIOS

Ejercicio 2.1

Se desea representar en el computador el rango de valores comprendidos entre -12.584 y 12.584. Para ello se dispone como ancho de palabra con 4 bits. Se pide:

- a) ¿Qué representación elegiría?
- b) ¿Es factible representar dicho rango en el computador?
- c) Suponiendo que se dispone de un mecanismo software, que permite el tratamiento conjunto de dos o más palabras, calcular el n.º de palabras que serán necesarias para representar el rango de valores.
- d) Determinar hasta qué número se podría representar con el n.º de palabras elegido.
- e) Determinar, asimismo, cuántos valores diferentes se pueden dar con dicho n.º de palabras.

Ejercicio 2.2

Para cada uno de los siguientes números binarios, indicar el número decimal al que corresponda según el tipo de representación que soporta:

- a)10101
- b) ...0100001
- c) ..00100011

Ejercicio 2.3

Si se quiere representar un rango de valores $-255 \leq x \leq 0$, ¿qué representación utilizará, con palabra de 8 bits?

Ejercicio 2.4

Calcular para la representación en exceso 2^{n-1} los siguientes apartados:

- a) Rango de valores de esta representación.
- b) ¿Qué característica cabe resaltar de dicha representación?
- c) Dando a n el valor 9, representar los siguientes números: 5, 135, -127 y -246.

Ejercicio 2.5

Dado el siguiente mensaje, representado en los códigos ASCII, EBCDIC y X.3, descubrir el error que encierra y corregirlo:

- Parte del mensaje en código ASCII:
0100011101001110101001001111
- Parte del mensaje en código EBCDIC:
110001011100010111000101111100
- Parte del mensaje en código X.3
01010101100011000101000

EJERCICIO 2.6

Indicar el polinomio a transmitir ($T(x)$) a partir de la siguiente secuencia de bits:

10110111

teniendo en cuenta que el grado del polinomio generador es 2. ¿Puede estar seguro el receptor de que lo recibido es correcto?

Ejercicio 2.7

Dado el siguiente mensaje:

111001001101011011101111

- Averiguar si se ha producido algún error en la transmisión teniendo en cuenta que se trabaja con palabras de 4 bits y con paridad horizontal a nivel de bloques, de forma que el bit de paridad vertical sigue a cada carácter.

Nota: Los primeros bits de información corresponden a los de más peso y el computador trabaja con paridad par.

- Generar para dicho mensaje los bits de paridad, caso de que ésta fuera impar.

Ejercicio 2.8

Se quiere representar un computador, números en coma flotante con mantisa entera; las características técnicas son las siguientes:

- Palabras de 16 bits.
 - Base = 2.
 - Exponente representado en exceso 2^{n-1} con $q = 7$.
 - Mantisa en binario puro con signo.
- Se pide:
- Máximo valor positivo representable.
 - Idem negativo.
 - ¿Cuál es la precisión de esta representación?

Ejercicio 2.9

Utilizando la representación en coma flotante con mantisa fraccionaria normalizada, donde el exponente se representa en exceso 2^{n-1} , deducir entre qué valores se puede mover un n º positivo y uno negativo.

Ejercicio 2.10

Dado el siguiente formato en coma flotante:

Signo : 1 bit, el de la izquierda.

Exponente: 6 bits en notación de exceso 32, representando un exponente que es potencia de 2, por la que se debe multiplicar la mantisa.

Mantisa: 8 bits de magnitud normalizada, con el punto decimal a la izquierda (el signo lo marca S).

- Representar en dicho formato 128 y -64.
- Multiplicarlos en binario, indicando los pasos seguidos, no olvidando normalizar el resultado si es necesario.

Ejercicio 2.11

Dados 2 números en simple precisión, averiguar qué números decimales representan:

- X_1 : 10100101011110011000110010110
- X_2 : 0110101010101000111010101101

Ejercicio 2.12

Se quiere representar en un computador números en coma flotante con mantisa fraccionaria. Las características técnicas son las siguientes:

- a) Palabra de 24 bits.
- b) Base = 2.
- c) Exponente representado en exceso 2^{9-1} .
- d) Mantisa en binario puro con signo, de 12 bits.

Se pide:

- ... Rango de valores positivos.
- ... Rango de valores negativos.
- Precisión de la representación.
- Error máximo en la representación.
- Acatación del error relativo (empleando $M = 1/2$).

La Unidad Aritmética y Lógica



3.1. INTRODUCCIÓN

La Unidad Aritmética y Lógica, llamada abreviadamente *UAL* y *ALU*, es la encargada de realizar en el computador las operaciones con los datos, de acuerdo con el programa en curso. Dado que la parte aritmética es la más importante y compleja, se la suele llamar *Unidad Aritmético*, simplemente.

Las operaciones que puede efectuar esta unidad son muy elementales, puesto que la mayoría de los computadores configuran la Unidad Aritmética con un sencillo *sumadorrestador*. La ejecución de operaciones complejas se lleva a cabo descomponiéndolas en pasos elementales, que se ejecutan a la velocidad de varios millones por segundo.

La Unidad de Control del computador se encarga de enviar la información a procesar a la Unidad Aritmética, así como el código que selecciona la operación que se debe hacer.

La estructura de la Unidad Aritmética consta de:

1. Uno o varios *operadores*, que son circuitos electrónicos que realizan una función aritmética o lógica.
2. Un *banco de registros* de tipo general, donde se almacenan los datos. Por lo general, el banco se compone de 8 o 16 registros.
3. Un registro, llamado *Acumulador*, en el que se deposita el resultado que origina el operador y que soporta la información en numerosas operaciones.
4. Un conjunto de *señalizadores de estado*, que son unos biestables que señalizan ciertas condiciones de la última operación realizada por la Unidad Aritmética.

Las más corrientes son:

Cero: Se pone a 1 si el resultado ha sido cero.

Negativo: Se pone a 1 si el resultado ha sido negativo.

Acarreo: Se pone a 1 si el resultado tiene acarreo.

Desbordamiento: Se pone a 1 si el resultado no cabe en el lugar que le corresponde (sobrepasamiento).

5. A veces, una instrucción se descompone en varias operaciones elementales, existiendo un **Secuenciador** que genera las señales apropiadas para desarrollar las diversas *microinstrucciones* que conforman la instrucción. Para cada instrucción se indica el operador que interviene, la operación a efectuar y los registradores que participan.

Un diagrama general de la estructura de bloques de la Unidad Aritmética, se ofrece en la figura 3-1.

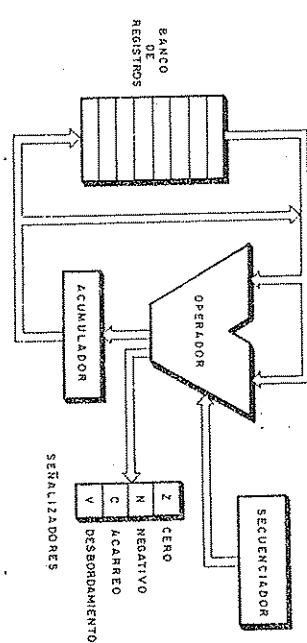


Fig. 3-1. Estructura general por bloques de la Unidad Aritmética.

3.2. TIPOS DE OPERADORES

Como ya se ha dicho, los operadores son circuitos electrónicos que realizan una o varias operaciones lógicas o aritméticas, tales como AND, OR, XOR, suma y resta. En la figura 3-2 se presenta a un operador elemental encargado de efectuar la suma binaria de dos bits junto al acarreo previo, produciendo el resultado y el acarreo final.

Los operadores tienen diferentes clasificaciones, de acuerdo con el concepto que se considere.

86 / La unidad aritmética y lógica

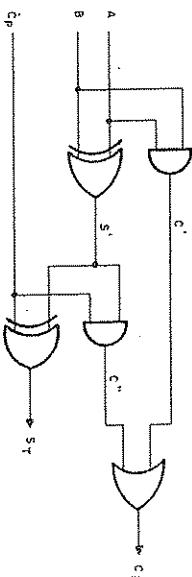


Fig. 3-2. Esquema de un operador destinado a la suma de 2 bits y el acarreo previo.

3.2.1. Operadores generales y especializados

Los operadores generales pueden realizar distintas clases de operaciones, mientras que los especializados se restringen a una muy determinada: por ejemplo, sumas y restas en coma flotante.

Los computadores convencionales poseen un solo operador, de tipo general, que hace todas las operaciones posibles. Los computadores muy sofisticados, cuya Unidad Aritmética consta de varios operadores especializados, han de soportar un elevado coste.

3.2.2. Operadores combinacionales y secuenciales

Los operadores combinacionales, como el de la figura 3-2, suministran una salida que es función del estado de las entradas. No dispone de elementos de memoria y el tiempo de respuesta está dado por la suma de los retardos de las puertas lógicas que deben atravesar las señales desde las entradas hasta la salida.

Los operadores secuenciales requieren varias fases para obtener el resultado, debiendo contar con elementos de memoria que almacenan la información que ha de transmitirse entre fases, así como un contador de las mismas. En este tipo de operadores se define un *algoritmo* con el que se halla el resultado y que contiene las etapas necesarias y la función que ha de efectuar cada una de ellas. Es habitual que la Unidad de Control genere las fases del operador, con lo que éste se simplifica notablemente.

De lo expuesto se deduce que una operación cualquiera puede realizarse de cuatro formas posibles:

- a) Mediante un circuito combinacional. La operación consta de una sola fase.
- b) Mediante un circuito secuencial que genera sus propias fases.

3

- c) Mediante un circuito secuencial cuyas fases las genera la Unidad de Control.
d) Mediante un programa.

3.2.3. Operadores paralelo y serie

Un operador paralelo realiza la operación correspondiente simultáneamente sobre todos los dígitos de los operandos. El operador serie trabaja dígito a dígito, siendo de tipo secuencial y requiere tantas fases como dígitos procesa.

3.2.4. Operadores MOS y bipolares

La tecnología que se aplica en la fabricación de los operadores determina la velocidad de funcionamiento.

Para comparar distintos diseños, se toma como base el tiempo de respuesta de una puerta NAND, al que se denomina a . El tiempo de las restantes puertas se expresa en función de a . El retardo típico de una puerta NAND TTL es de 10 ns; si es Schottky TTL, de 3 ns y si es CMOS, de 50 a 70 ns.

3.3. LAS OPERACIONES DE LA UNIDAD ARITMÉTICA Y LÓGICA

Las operaciones que puede efectuar la Unidad Aritmética, según su propósito, se clasifican en tres grandes grupos:

1. De desplazamiento.

2. Aritméticas.

3. Lógicas.

En la tabla de la figura 3-3 se muestra el conjunto de operaciones típicas que realizan las cuatro categorías de computadores: *microprocesador*, *minicomputador*, *computador medio* y *computador rápido*. También se indica la manera habitual de llevar a cabo la operación mediante unidad combinacional (C), unidad secuencial específica (S), tareas controladas por la Unidad de Control (UC), o bien, por programa (P).

OPERACIÓN	MICRO- PROCESADOR	MINI- COMPUTADOR	COMPUTADOR MEDIO	COMPUTADOR RÁPIDO
SUMA/RESTA EN BINARIO	C	C	C	C
SUMA/RESTA EN COMA FLOTANTE	P	P	UC	S
SUMA/RESTA EN BCD SERIE	P	P	UC	UC
MULTIPLICACIÓN EN BINARIO	P	P/UC	UC	C
MULTIPLICACIÓN EN COMA FLOTANTE	P	P/UC	UC	S
DIVISIÓN EN BINARIO	P	P/UC	UC	S
DIVISIÓN EN COMA FLOTANTE	P	P/UC	UC	S
OPERACIONES LÓGICAS	C	C	C	C
DESPALZAMIENTO UNIARIO	C	C	C	C
DESPALZAMIENTO MÚLTIPLE	P	P/UC	UC	C

Fig. 3-3. Operaciones típicas de la Unidad Aritmética en las cuatro categorías de computadoras y forma de llevarlas a cabo, mediante unidad combinacional (C), secuencial (S) tareas generadas por la Unidad de Control (UC) o por programas (P).

3.4. OPERACIONES DE DESPLAZAMIENTO

Los desplazamientos se realizan corriendo los bits de una palabra, dato o registro hacia la derecha o izquierda. Si el operando origen, A, está compuesto por los bits:

$$a_{n-1}, a_{n-2}, \dots, a_2, a_1, a_0$$

el operando resultado, B, después de realizar el desplazamiento, estará compuesto por una cadena de bits, que cumplen la ley:

$$b_{l+k} = a_i \quad (3.1)$$

donde k indica el número de posiciones que se han desplazado. Si $k > 0$, el desplazamiento se efectúa hacia la izquierda y viceversa.

En realidad, la expresión (3.1) no puede aplicarse a los extremos del operando, pues éstos se rellenan según el desplazamiento sea lógico o aritmético.

Aunque los computadores sencillos sólo disponen de operaciones de desplazamiento de una posición a derecha o izquierda, casi todas las máquinas modernas permiten *desplazamientos múltiples* en ambos sentidos.

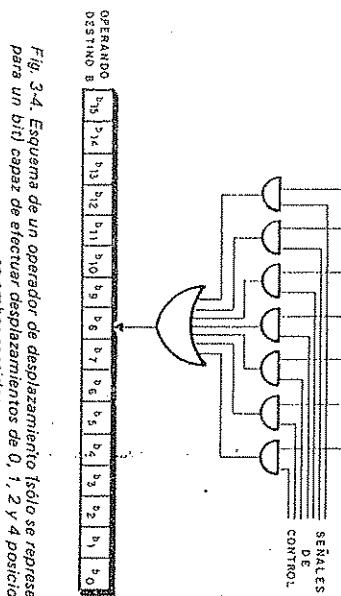
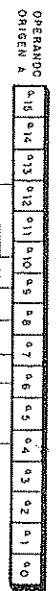


Fig. 3-4. Esquema de un operador de desplazamiento. Sólo se representa para un bit capaz de efectuar desplazamientos de 0, 1, 2 y 4 posiciones en ambos sentidos.

En la figura 3-4 se muestra un operador de desplazamiento que permite desplazamientos de 0, 1, 2 y 4 posiciones en ambos sentidos. Sólo se ha representado el esquema correspondiente a un bit. Las señales de control necesarias para definir el desplazamiento son comunes a todos los bits.

Es frecuente que coincidan el operando origen y el destino, con lo que el desplazador se reduce a un simple registro de desplazamiento.

3.4.1. Desplazamientos lógicos

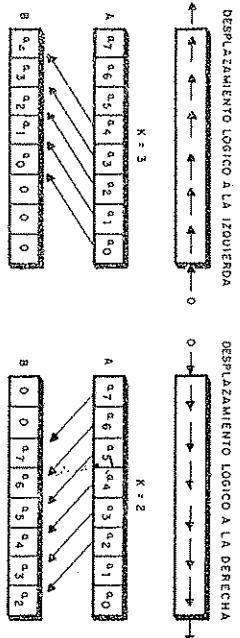


Fig. 3-5. Expresión gráfica de los desplazamientos lógicos a izquierda y derecha con ejemplos de aplicación.

Los k bits que se vacían en un extremo se rellenan con los que salen por el otro. Figura 3-6.

Observese que en los desplazamientos lógicos existe pérdida de información. También hay operadores de desplazamiento que permiten rellenar con unos las posiciones vacías.

3.4.2. Desplazamientos circulares

Los k bits que se vacían en un extremo se rellenan con los que salen por el otro. Con los desplazamientos circulares no hay pérdida de información.

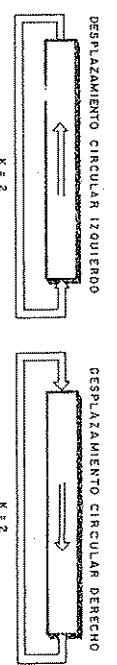


Fig. 3-6. Representación gráfica de los desplazamientos circulares en ambos sentidos, con ejemplos de aplicación.

3.4.3. Desplazamientos aritméticos

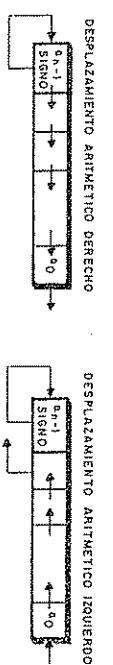


Fig. 3-7. En los desplazamientos aritméticos se mantiene invariable el bit de signo y las posiciones sobrantes se rellenan con ceros, lo que equivale a dividir o multiplicar el valor inicial por una potencia de 2. En la división se rellenan a la izquierda con el bit de signo.

3

Los desplazamientos aritméticos se realizan sobre números en representación de complemento y son parecidos a los lógicos, pero salvando o manteniendo el signo de la cantidad. Estos desplazamientos suponen una multiplicación o división por una potencia de 2. En la figura 3-7 se muestra, gráficamente, el comportamiento de este tipo de desplazamientos.

Para multiplicar por potencias de 2 un número negativo, representado en complemento a 1, se deben introducir "unos" por la derecha, en lugar de "ceros".

3.4.4. Desplazamientos concatenados

Son desplazamientos que afectan a un conjunto concatenado de dos o más elementos, que pueden ser:

1. Dos registros.
2. Un registro con el bit estable de acarreo.
3. Un registro con el bit estable de signo.

En la figura 3-8 se muestran algunos ejemplos de desplazamientos concatenados.

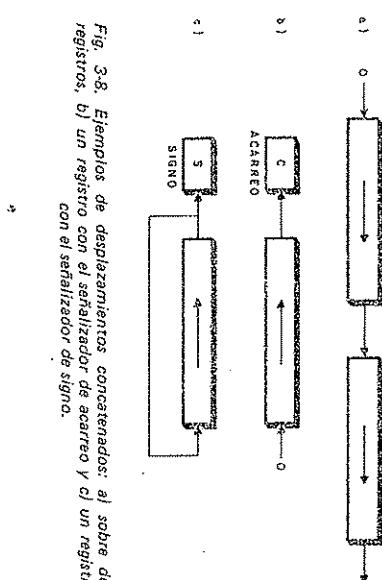


Fig. 3-8. Ejemplos de desplazamientos concatenados: a) sobre dos registros, b) un registro con el señalizador de acarreo y c) un registro con el señalizador de signo.

3.5. OPERACIONES LÓGICAS

Con este tipo de operaciones, a cada bit del operando origen se le aplica una función lógica, independientemente de los restantes bits, lo que significa que el resultado de aplicar un operador lógico a un bit no afecta al resultado de los demás bits.

92 / La unidad aritmética y lógica

3

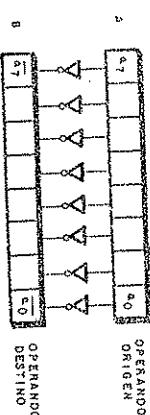


Fig. 3-9. Forma gráfica de realizar la función lógica INVERSIÓN. Se trata de una operación monádica, porque sólo necesita un operando para producir el resultado.

Las operaciones lógicas más comunes en los computadores son:

- Negación o inversión lógica (\bar{A}).
- OR o suma lógica ($A + B$).
- AND o producto lógico ($A \cdot B$).
- EOR o OR Exclusiva ($A \oplus B$).

También son frecuentes las operaciones NOR y NAND.

La operación negación se efectúa sobre un solo operando, por lo que recibe el nombre de *monádica*, mientras que las restantes operaciones lógicas actúan sobre dos operandos y se llaman *díducas*. Figura 3-9.

3.6. OPERACIONES ARITMÉTICAS

A continuación se van comentando las más importantes.

3.6.1. Cambio de signo

La operación de "cambio de signo" se efectúa de distinta manera para cada tipo de representación que se emplee.

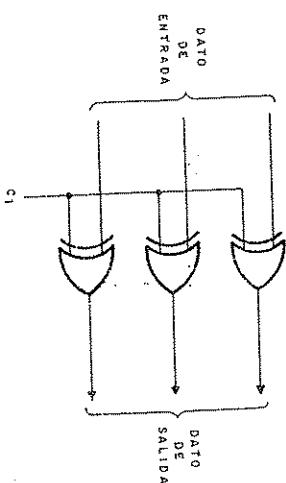


Fig. 3-10. Etapa que cambia el signo a un dato expresado en complemento a uno, cuando la señal de control $C_1 = 1$.

La unidad aritmética y lógica / §3

1. *Representación en binario puro:* Sólo se modifica el bit de signo.

2. *Representación en complemento a uno:* El cambio de signo coincide con la negación lógica. En la figura 3-10 se muestra una etapa de varias puertas XOR, que cambia el signo del operando cuando la señal de control $C_i = 1$. Cuando $C_i = 0$, se transmite el dato sin modificarlo.

3. *Representación en complemento a dos:* Equivale a realizar una negación lógica y añadir una unidad. En la figura 3-11 se muestra una etapa de varias puertas el signo a los datos expresados en complemento a dos. $c_i = a_i \oplus C_{i-1}$ y $a'_i = a_i \oplus E_i C_{i-1}$.

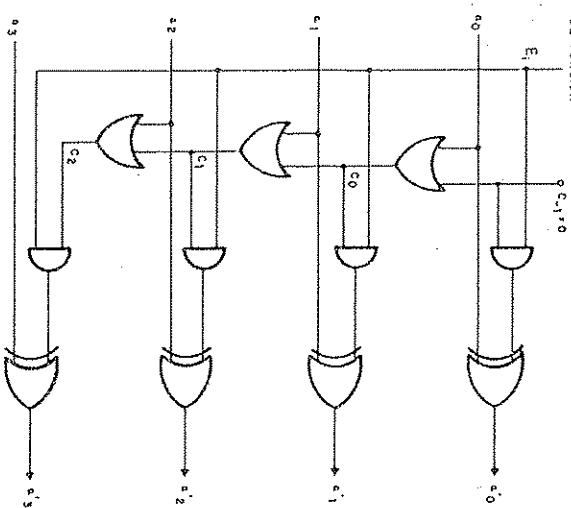
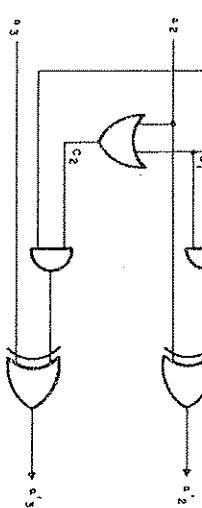


Fig. 3-11. Esquema lógico para un operador de cambio de signo a datos expresados en complemento a dos. En el momento que un bit sea 1, los restantes bits de más peso se complementan.



- b) *Representación en complemento a uno o a dos:* Los bits sobrantes de más peso se rellenan con el mismo valor que el bit de signo. Figura 3-13.

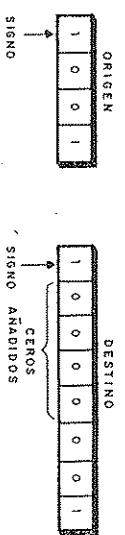


Fig. 3-13. Ejemplos de extensión de signo en números expresados en complemento a uno o a dos.

3.6.2. Extensión de signo

Cuando se traslada un valor numérico a la memoria, a un registro, a un bus, a un operador u otro elemento de mayor número de bits, los bits sobrantes deben rellenarse de forma que no cambie el significado. Según el tipo de representación elegido, el tratamiento es diferente.

3.6.3. Adición y sustracción

Estas dos operaciones se resuelven de manera similar, puesto que la resta puede considerarse como una suma si se toma el sustraendo en forma de complemento a dos.

La suma es la operación más importante sobre la que se diseña la Unidad Aritmética del computador. Los sumadores adoptan diferentes estructuras de acuerdo con el tipo de representación que manejan.

3.6.3.1. Sumador elemental

Es un operador, consistente en un circuito combinacional capaz de sumar dos dígitos binarios más el posible acarreo previo de la etapa anterior, produciendo el dígito suma y el de acarreo para la siguiente etapa.

1. *Representación en binario con signo:* El bit de signo se traslada al bit de más peso del destino, rellenando con ceros los bits sobrantes. En la figura 3-12 se presenta un ejemplo de aplicación.

- a) *Representación en binario con signo:* El bit de signo se traslada al bit de más peso del destino, rellenando con ceros los bits sobrantes. En la figura 3-12 se presenta un ejemplo de aplicación.

3

Ejemplo

$$\begin{array}{r}
 11111 \leftarrow \text{Acarreos } (c) \\
 A: \quad 010101 \\
 B: + \quad 101011 \\
 \hline
 S: \quad 100010
 \end{array}$$

Cada bit s_i de S se obtiene sumando los bits $a_i + b_i$, junto al acarreo previo c_{i-1} . Además de s_i también se genera el acarreo c_i para la siguiente etapa sumadora. La tabla de la verdad para este sumador elemental se ofrece en la figura 3-14.

ENTRADAS				SALIDAS	
1º SUMANDO	2º SUMANDO	ACARREO PREVIO	SUMA	ACARREO FINAL	c_i
a_i	b_i	c_{i-1}	s_i	c_i	
0	0	0	0	0	
1	0	0	1	0	
0	1	0	1	0	
1	1	0	0	1	
0	0	1	1	0	
1	0	1	0	1	
0	1	1	0	1	
1	1	1	1	1	

Fig. 3-14. Tabla de la verdad correspondiente a una etapa de un sumador elemental.

Según la figura 3-14:

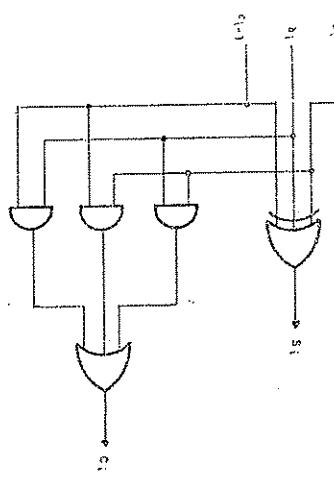
$$s_i = a_i \oplus b_i \oplus c_{i-1}$$

$$c_i = a_i b_i + a_i c_{i-1} + b_i c_{i-1}$$

y donde se puede diseñar el diagrama lógico que cumpla los algoritmos del sumador elemental y que se muestra en la figura 3-15.

3

Fig. 3-15. Diagrama lógico del sumador elemental.



En la figura 3-16 se muestra el símbolo que se utiliza para representar al sumador elemental.



Fig. 3-16. Símbolo empleado para representar al sumador elemental.

El tiempo de retardo del sumador elemental de la figura 3-15 será de 20 ns, tomando como retardo por puerta 10 ns. En realidad, el resultado s_i se obtiene en 10 ns y el acarreo c_i en 20 ns.

Por lo general, las sumas se realizan con números de varios bits, por lo que son, relativamente, complejos los sumadores, aunque todos ellos se basan en sumadores elementales. Según el tratamiento de los bits de los sumandos, existen sumadores paralelo y serie.

3.6.3.2. Sumador paralelo

Consta de tantos sumadores elementales encadenados, según se muestra en la figura 3-27, como bits tengan los operandos.

El acarreo c_{i-1} se debe poner a cero antes de producirse la suma.

El retardo total del esquema de la figura 3-17 viene dado por el máximo número de puertas que deben atravesar las señales. Como se considera que cada sumador tiene un tiempo de respuesta de 2 retardos de 10 ns cada uno para producir la suma y el acarreo, el resultado total del sumador de la figura no será correcto hasta que transcurran $n \cdot 20$ ns.

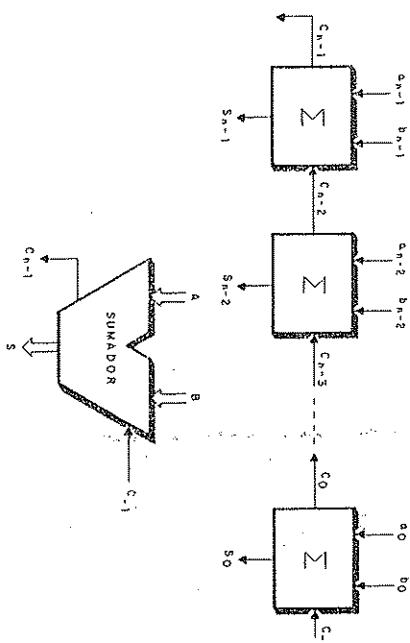


Fig. 3-17. Esquema de un sumador paralelo de n bits y símbolo representativo.

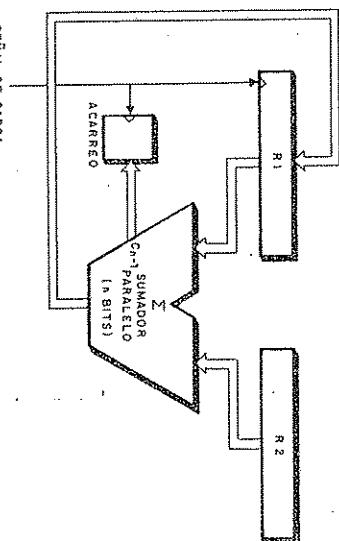


Fig. 3-18. Esquema de utilización práctica del sumador paralelo.

En la figura 3-18 se presenta el esquema de utilización del sumador paralelo, que emplea los registros R_1 y R_2 para almacenar los operandos. El resultado se almacena en R_1 , activando la señal de carga, que también sirve para determinar el acarreo c_{n-1} se deposite en el señalizador de acarreo.

3.6.3.3. Sumador serie

En este caso, un único sumador elemental va sumando cada pareja de bits de los registros que contienen los sumandos. Figura 3-19.

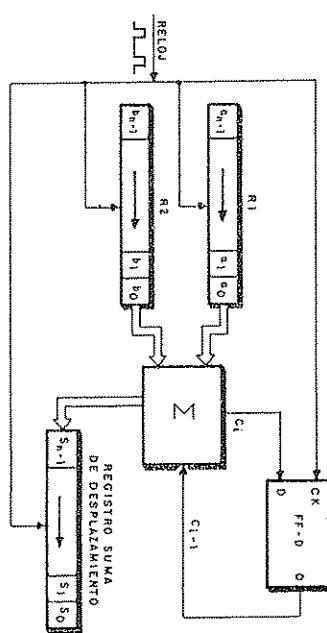


Fig. 3-19. Esquema de un sumador serie.

Un sumador elemental va recibiendo, en cada paso controlado por la señal de reloj procedente de la Unidad de Control, los dos bits que están contenidos en los registros R_1 y R_2 (figura 3-19), cuyo contenido se desplaza una posición a la derecha en cada paso. Un flip-flop, tipo D, se carga con el acarreo producido con cada suma, mientras que el previo que contenía de la fase anterior lo introduce al sumador elemental para que se sume con los bits extraídos de R_1 y R_2 . Los bits suma se guardan en otro registro de desplazamiento dedicado a este fin. Por lo tanto, con cada impulso de reloj se producen los siguientes acontecimientos:

1. Se sacan dos bits de R_1 y R_2 , que se suman con el acarreo de la suma anterior, guardado en el flip-flop D.
2. El bit suma se guarda en otro registro de desplazamiento que contendrá el resultado.
3. Además, el flip-flop D se carga con el acarreo actual, que se memoriza hasta que se efectúe la suma de la siguiente pareja de bits.

3.6.3.4. Restador paralelo

Aunque se pueden diseñar restadores elementales, como la operación más frecuente es la suma, se suele añadir al sumador un circuito que permita llevar a cabo la suma y la resta, seleccionando la operación a efectuar mediante una señal de control, llamada \bar{S}/R , que suma si $\bar{S}/R = 0$.

Sea la resta de dos operandos positivos, $A - B$. Sumando y restando 2^n se puede escribir:

$$A - B = A - B + 2^n - 2^n = A + (2^n - B) - 2^n$$

El término $(2^n - B)$ es el complemento a dos de B y se calcula haciendo la negación lógica de B y sumando una unidad al resultado. La negación se puede realizar a base de puertas XOR y el 1 que hay que sumar se consigue introduciendo 1 por el acarreo c_{n-1} . Por otro lado, restar 2^n al resultado es lo mismo que restar 1 al bit s_n inexiste, lo que en la práctica se reduce a ignorar el acarreo final c_{n-1} .

En la figura 3-20 se muestra el esquema de un sumador-restador paralelo. Si la señal $\bar{S}/R = 0$, el circuito produce la suma de los operandos, mientras que si $\bar{S}/R = 1$, genera la resta.

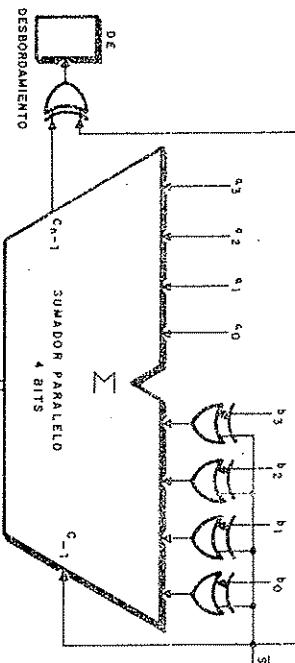


Fig. 3-20. Circuito sumador-restador paralelo. Para que efective la resta es necesario que $\bar{S}/R = 1$. En caso contrario, se suman los dos operandos.

3.6.3.5. Suma y resta en binario sin signo

Se puede emplear directamente el esquema de la figura 3-20 para este tipo de representación.

Las condiciones en las que se origina *desbordamiento* (DE) son las siguientes:

1. En el caso de la suma, el desbordamiento es directamente suministrado por el bit de acarreo c_{n-1} , puesto que se "lleva 1" que no cabe en el resultado.
2. En el caso de la resta, el desbordamiento se produce cuando $B > A$, puesto que daría resultado negativo. Dado que el resultado realmente calculado es $A + (2^n - B)$, si $B > A$, será $2^n + A - B < 2^n$ y, por tanto, $c_{n-1} = 0$, lo que significa que la condición de desbordamiento se produce cuando $c_{n-1} = 0$.

De aquí se deduce que la condición de desbordamiento sigue la expresión:

$$DE = c_{n-1} \oplus \bar{S}/R$$

3.6.3.6. Suma y resta en complemento a uno

Las operaciones de suma y resta en complemento a uno, precisan de una ligera modificación en el esquema de la figura 3-20 para funcionar correctamente.

Por un lado, una cantidad en complemento a uno sólo exige la negación lógica y, por lo tanto, no se deberá introducir la señal de control \bar{S}/R por el acarreo c_{n-1} .

Por otro lado, al sumar dos números con este tipo de representación, para que el resultado sea correcto hay que sumar el acarreo al resultado.

Ejemplo

$$\begin{array}{r} 0111 \\ + 1010 \\ \hline \end{array}$$

$$\begin{array}{r} 1\ 0001 \\ + 1 \\ \hline \end{array}$$

La suma del acarreo final c_{n-1} al resultado s_4^e puede conseguirse recirculando dicho bit hasta el acarreo c_{n-1} , como se refleja en el esquema de la figura 3-21.

La condición de desbordamiento sigue la ley:

$$DE = c_{n-1} \oplus c_{n-2}$$

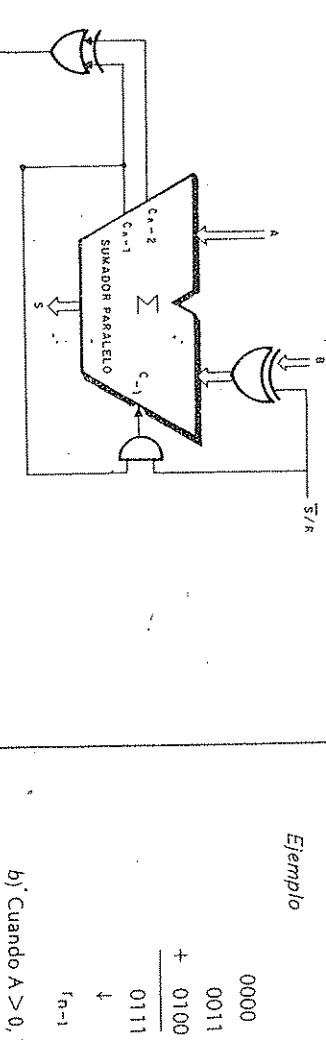


Fig. 3-21. Diagrama lógico de un sumador-restador de números expresados en complemento a uno.

3.6.3.7. Suma y resta en complemento a dos

El esquema propuesto en la figura 3-20 produce la suma realizando el complemento a dos del sustraendo (B_1), por lo que es directamente utilizable para operando en forma de complemento a dos. Sin embargo, como los bits de más peso de los operandos (a_{n-1} y b_{n-1}) representan el signo y el resto la magnitud, el desbordamiento viene dado, al igual que el complemento a uno, por:

$$DE = c_{n-1} \oplus c_{n-2}$$

Si se suman dos magnitudes positivas se produce desbordamiento cuando $c_{n-2} = 1$. Si son dos magnitudes negativas el desbordamiento se produce cuando $c_{n-2} = 0$ y como el $c_{n-1} = a_{n-1} \cdot b_{n-1} + c_{n-2} (a_{n-1} + b_{n-1})$, resulta $DE = c_{n-1} \oplus c_{n-2}$.

Al sumar dos números en complemento a dos, se pueden dar cuatro casos:

1) Los sumandos A y B son positivos

Admite dos variantes:

a) Cuando $A > 0, B > 0, A + B \leq 2^{n-1} - 1$.

Si la suma es menor que $2^n - 1$, o sea, el resultado cabe en el número de bits que tienen los operandos A y B , son cero. En este caso el desbordamiento ($c_{n-2} \oplus c_{n-1}$) será cero (como debe ser) y c_{n-1} también será cero, indicando que el resultado es un número positivo.

102 / La unidad aritmética y lógica

- b) Cuando $A > 0, B > 0, A + B > 2^{n-1} - 1$.
- Si la suma "no cabe" en el espacio reservado, c_{n-2} será uno y c_{n-1} será cero, con lo que el desbordamiento $c_{n-2} \oplus c_{n-1} = 1$. Se produce desbordamiento, puesto que el resultado sale de tipo negativo $r_{n-1} = 1$, cuando debería ser cero.

Ejemplo

0110 → Acarreos

0111 → A

$$\begin{array}{r}
 + 0110 \rightarrow B \\
 \hline
 1101 \rightarrow R
 \end{array}$$

2) Cuando $A > 0$ y $B < 0$, siendo $A \geq |B|$

c_{n-2} y c_{n-1} valdrán siempre 1, quedando la expresión del desbordamiento: $1 \oplus 1 = 0$, como debe ser, puesto que la suma de dos números de distinto signo no puede producir sobrepasamiento en ningún caso.

Ejemplo

1110 ← Acarreos

0111 ← A

$$\begin{array}{r}
 + 1010 \leftarrow B \\
 \hline
 0001 \leftarrow R
 \end{array}$$

3) Si $A > 0$ y $B < 0$, siendo $A < |B|$

6

c_{n-2} y c_{n-1} valen cero y el desbordamiento $c_{n-2} \oplus c_{n-1} = 0$. Igual que en el caso anterior, no se puede producir desbordamiento y $|v_{n-1}|$ ha de valer 1, al ser $A < |B|$.

Ejemplo

```

0000 ← A
0110 ← A
+ 1001 ← B
-----
1111 ← R

```

$$4) S \wedge A <_O y \wedge B <_O$$

Pueden producirse dos casos:

Si la suma "cabe" en el lugar reservado, siendo $c_{n-2} = c_{n-1} = 1$ y $c_{n-1} \oplus c_{n-2} = 0$. Además $r_{n-1} = 1$, indicando que el número es negativo.

Ejemplo

```

i100 ← A
1101 ← A
+ 1100 ← B
_____
1001 ← R

```

b) Si "no cabe" la suma en el lugar reservado, $c_{n-2} = 0$ y $c_{n-1} = 1$, por lo que se produce desbordamiento y $n=0$.

Ejemplo

1000	← Ac
1001	← A
+	1010 ← B
—————	
0011	← R

En la figura 3-22 se presenta un esquema de un sumador-restador de complemento a dos.

104 / La unidad aritmética y lógica

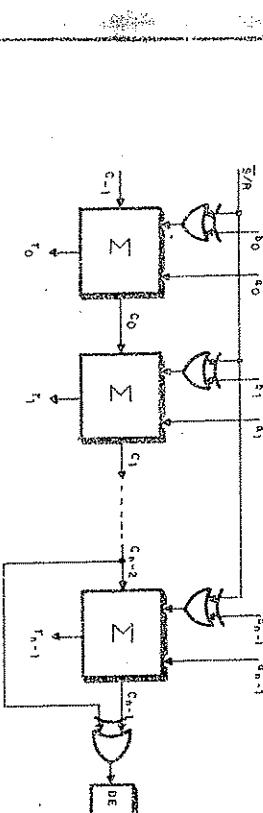


Fig. 3-22. Sumador-restador de datos expresados en complemento a dos.

3.6.3.8. Suma y resta en signo y nisgnitudo

Se utiliza la señal auxiliar de *complemento* (CPTO) para indicar si se realiza el complemento a uno del segundo operando ($CPTO = 1$) o no ($CPTO = 0$). Esta señal se obtiene mediante una combinación de a_{n-1} , b_{n-1} y \bar{S}/R , según la tabla de la figura 3-23, de acuerdo con la expresión $CPTO = \bar{a}_{n-1} \oplus b_{n-1} \oplus \bar{S}/R$, como se demostrará más adelante.

SIGNO DE A a_{n-1}	SIGNO DE B b_{n-1}	\overline{S}/R	CPIO	OPERACION	OPERACION EJECUTADA (INDEPENDIENTE DEL SIGNO)
1	1	1	1	$-A - (-B)$	$A + \overline{B}$
1	1	0	0	$-A + (-B)$	$A + B$
1	0	1	0	$-A - B$	$A + B$
1	0	0	1	$-A + B$	$A + \overline{B}$
0	1	1	0	$A + (-B)$	$A + B$
0	1	0	1	$A + (-B)$	$A + \overline{B}$
0	0	1	1	$A - B$	$A + \overline{B}$
0	0	0	0	$A + B$	$A + B$

Fig. 3-23. Tabla para la formación de valor de la señal de complemento CPTO.

Para corregir las sumas que se indican en la figura 3-23, al resultado se le invierte cuando $CPTO = 1$ y, además, la llevada del número (sin considerar el signo), c_{n-2} .

65

es nula. La señal que controla esta inversión es la salida $CPTS$, que corresponde a la función:

$$CPTS = CPTO \cdot \overline{c_{n-2}}$$

Por trabajar con complemento a uno, cuando se produce acarreo en el número (sin contar el signo) se suma el mismo mediante una realimentación a través de c_{n-1} .

Los 8 casos de la tabla de la figura 3-23 pueden comprobarse para constatar que realizan correctamente la suma con el procedimiento propuesto.

Ejemplo

Primer paso de la tabla de la figura 3-23.

$$\begin{array}{r} (-7) \rightarrow 10111 \\ (-8) \rightarrow 11000 \\ \hline \text{CPTO} = 1 \\ 1110 \\ 0001 \end{array}$$

$$CPTS = CPTO \cdot \overline{c_{n-2}} = 1 \cdot 1 = 1$$

El bit de signo (subrayado) se trata, paralelamente, con otra lógica para obtener el signo de la suma. El número será positivo ($r_{n-1} = 0$) en cualquiera de estos casos.

1. Cuando hay llevada en c_{n-2} y el signo de A es positivo.
 2. Cuando no hay llevada y la operación es resta de un número negativo.
 3. Cuando no hay llevada y la operación es suma de un número positivo.
- En los demás casos, el resultado es negativo.

Algebraicamente se representa así:

$$r_{n-1} = a_{n-1} \cdot c_{n-2} (b_{n-1} \oplus \overline{SJR}) \cdot \overline{c_{n-2}}$$

El sobrepasamiento DE se produce cuando $CPTO = 0$ y $c_{n-2} = 1$.

En la figura 3-24 se muestra el diagrama del sumador-restador de signo y magnitud.

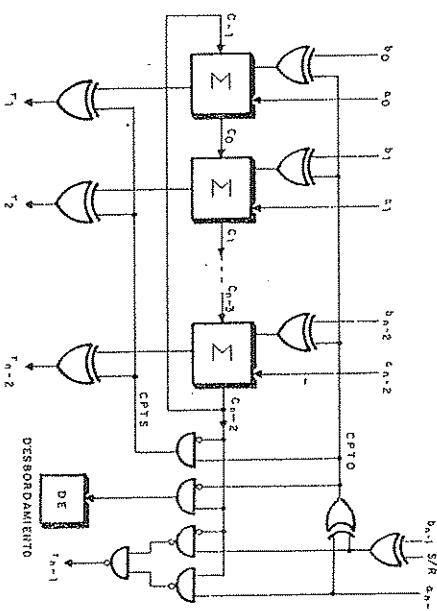


Fig. 3-24. Esquema lógico del sumador-restador de signo y magnitud.

De la tabla de la figura 3-23 se ha obtenido CTO, llamando $A = a_{n-1}$, $B = b_{n-1}$ y $R = \overline{SJR}$.

$$\begin{aligned} CPTO &= ABR + A\overline{B}R + \overline{A}B\overline{R} + \overline{A}\overline{B}\overline{R} = \\ &= (BR + \overline{B}R)A + (B\overline{R} + \overline{B}R)\overline{A} = \\ &= ((B + \overline{R}) \cdot (B + \overline{R}))A + (B \oplus R)\overline{A} = \\ &= (\overline{B}R + R\overline{B})A + (B \oplus R)\overline{A} = \\ &= (B \oplus R)A + (B \oplus R)\overline{A} = \\ &= (B \oplus R) \oplus A \end{aligned}$$

3.6.3.9. Sumador rápido de acarreo anticipado

Con objeto de aumentar la velocidad de los sumadores, se presenta un modelo que no tiene que esperar a que se originen los acarreos en los sumadores elementales para pasar de un paso sumador completo al siguiente. Utilizando una lógica auxiliar se proporcionan, simultáneamente, todas las entradas de acarreo. Los sumadores de "acarreo anticipado" son considerablemente más rápidos que los sumadores paralelos con arrastre en serie y se emplean en los computadores rápidos.

3

Si se llama C_1 al acarreo del primer sumador completo, éste valdrá 1 en los dos siguientes casos:

1. Cuando $A_1 \cdot B_1 = 1$.
2. Cuando $A_1 \circ B_1$ valen 1 y, además, el acarreo previo $C_0 = 1$.

Si se denomina G_1 a la primera condición:

$$G_1 = A_1 \cdot B_1 \quad (1)$$

Para expresar la segunda condición, se comienza definiendo P_1 :

$$P_1 = \bar{A}_1 \cdot B_1 + A_1 \cdot \bar{B}_1 = A_1 \oplus B_1 \quad (2)$$

Con estas dos definiciones, la ecuación lógica completa para la primera y segunda condición se puede escribir:

$$C_1 = G_1 + P_1 \cdot C_0 \quad (3)$$

Donde cada uno de los dos términos representa una de las dos condiciones y los subíndices 1 se refieren al primer paso sumador.

Para el segundo paso se puede escribir una ecuación similar:

$$C_2 = G_2 + P_2 \cdot C_1 \quad (4)$$

Sin embargo, en esta ecuación el término C_1 puede sustituirse por su valor de la ecuación (3), de la siguiente manera:

$$C_2 = G_2 + P_2(G_1 + P_1 \cdot C_0) = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot C_0 \quad (5)$$

Por lo tanto, la expresión de C_2 se representa en términos de G y P solamente, a excepción de C_0 . Esto es importante, ya que ahora C_2 sólo depende de las señales de entrada A y B hasta los dos primeros pasos sumadores. Esto significa que, para concretar C_2 , sólo es necesario disponer de los valores de A_1 , A_2 , B_1 y B_2 . No se precisa la salida C_1 del primer paso.

De forma semejante, la ecuación de C_3 se puede escribir, únicamente, en términos G y P :

$$C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_0 \quad (6)$$

Generalizando para un valor k de etapa sumadora:

$$C_k = G_k + P_k G_{k-1} + P_k P_{k-1} G_{k-2} + \dots + P_k P_{k-1} \dots P_1 C_0 \quad (7)$$

3

Fig. 3-25. Esta última ecuación indica que las entradas de acarreo de todas las etapas de un sumador pueden desarrollarse tan pronto como se reciben las señales A y B . Dicho esto, que los términos G y P sólo dependen de A y B . En la figura 3-25 se muestra un sumador de acarreo anticipado de 4 bits con los circuitos lógicos de generación de G y P .

Si se supone que la lógica de anticipación del acarreo consume un tiempo de 30 ns y cada paso sumador precisa de otros 30 ns, el sumador completo de la figura 3-25 tardará 60 ns en producir el resultado correcto. Para sumadores de 5 o más bits, la diferencia entre el tiempo de retraso de este tipo de sumador y los de acarreo serie, se hace cada vez mayor.

La limitación más importante del sumador con acarreo anticipado es que, a medida que aumenta el número de etapas, la ecuación (7) se hace más larga, al igual que cada uno de sus términos. Se necesitan más pueras y éstas deben poseer mayor número de entradas (mayor fan-in).

En la figura 3-26 se presenta el diagrama lógico del circuito integrado comercial 74182, que realiza la función de generador de acarreo anticipado.

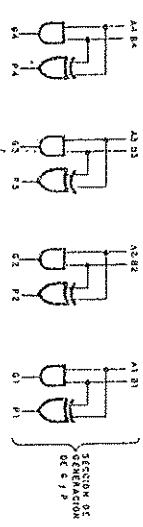


Fig. 3-25. Esquema de un sumador con acarreo anticipado y lógica auxiliar para obtener todos los términos G y P necesarios para hallar los acarreos de todas las etapas.

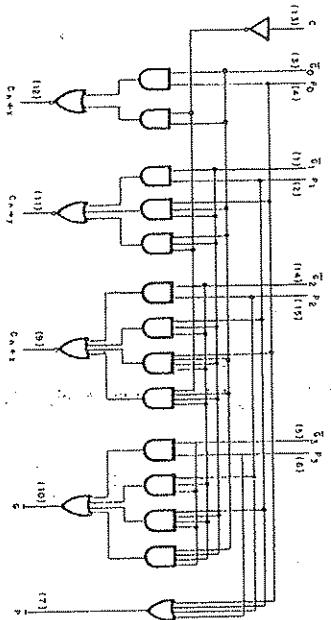


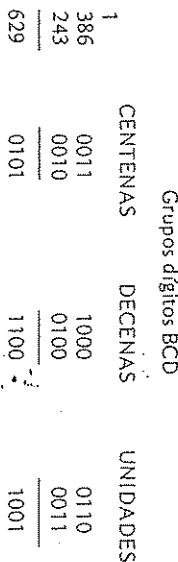
Fig. 3-26. Diagrama lógico del generador de acarreo anticipado presentado, comercialmente, en el circuito integrado 74182.

3.6.3.10. Sumador-restador BCD

En los computadores, tanto la suma como la resta se realizan casi siempre en código binario puro. En calculadoras electrónicas y otras aplicaciones se utiliza, frecuentemente, el código BCD, que es el decimal codificado en binario. Los sumadores y restadores BCD realizan las operaciones poniendo en práctica los mismos algoritmos e igualales circuitos básicos que cualquier otro sumador-restador binario; sin embargo, los datos se emplean en formato BCD, en lugar del binario.

Ejemplo

Se presenta un ejercicio de suma en BCD.



En forma decimal, la suma, incluido el acarreo de 1, nos resulta muy familiar.

Sin embargo, en BCD la suma de cada uno de los grupos de dígitos, da como resultado tres números: 5, 12 y 9. Si se leyesen directamente, la suma se escribiría: 5, (12), 9, lo cual, naturalmente, resulta incorrecto. El procedimiento adecuado determina que en las decenas se ponga sólo el número 2, arrastrando el 1 a la posición

de las centenas, para cambiar el 5 que habría en esa posición por un 6. Cuando se suma un número en formato BCD mediante circuitos lógicos, se precisa una lógica auxiliar especial para detectar el acarreo y corregir los dígitos BCD siguiendo el procedimiento expuesto. En el ejemplo anterior, además de la detección del acarreo y su propagación al grupo de las decenas, el grupo de las decenas debe corregirse para proporcionar una salida en forma binaria de 2 y no de 12, lo que ocasiona el cambio del número 1100 al 0010. La figura 3-27 muestra un circuito sumador BCD que puede generar el acarreo y corregir la salida del sumador a un dígito BCD de 4 bits.

Antes de discutir la lógica auxiliar para producir la generación de acarreo y la corrección, pero en el caso general, cualquiera de los grupos de dígitos puede exceder de sumar dos números cuya suma sea tan grande como 15 sin generar acarreo, se precisa una corrección y un acarreo decimal siempre que la suma supere a 10. En el ejemplo precedente, sólo el grupo de las decenas generó acarreo y precisaba de corrección, pero en el caso general, cualquiera de los grupos de dígitos puede exceder de 9 al hacer una suma. Por ejemplo, para hacer una suma de dos números BCD de 3 dígitos, se requieren 3 fases paralelas de sumador, cada una de ellas con capacidad para generar un acarreo y corregir la salida cuando sea necesario.

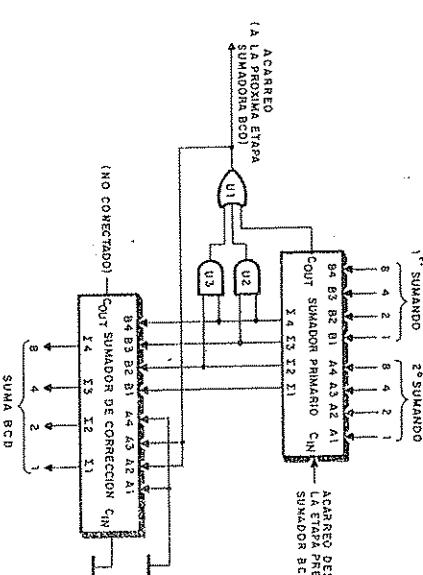


Fig. 3-27. Etapa de un típico sumador BCD.

En el esquema de la figura 3-27 existe un circuito de detección en cada etapa del sumador para proporcionar en la salida un acarreo si el número suma es mayor que 9. Este detector consiste en unas pocas puertas decodificadoras (U1 a U3), que de-

reducir los resultados de 10, 11, 12, 13, 14 ó 15, o el acarreo binario para sumas comprendidas entre los valores 16 a 19.

Para corregir la salida de la suma de cualquier fase que ha generado un acarreo se suma 6 (en binario 0110) a dicho resultado, puesto que 0110 es el complemento a dos de 1010 (10) y sumar 0110 es lo mismo que restar 1010 (en decimal 10). La corrección se efectúa en un segundo sumador, en donde se añade 0110 a la suma sin corregir. La figura 3-27 ilustra la forma en que la presencia de un acarreo a la salida de U1 (resultado superior a 9) hace que se sume 0110 en el sumador de corrección y la ausencia de acarreo, hace que se sume 0000 y no se modifique el resultado del primer sumador.

La etapa de la figura 3-27 se puede conectar en paralelo con varias etapas idénticas para formar un sumador multidígito BCD. Los sumadores de 4 bits que hay en cada etapa pueden ser de tipo paralelo simple, de tipo de acarreo circular o alguna combinación de ambos. Si la salida de acarreo está conectada a C-IN de la siguiente etapa, forma un sumador BCD con acarreo en serie.

La resta BCD puede realizarse de diversas formas. En este análisis se considerarán sólo dos: la resta por complemento a uno y la resta por complemento a nueve.

En la resta por complemento a uno se efectúa la diferencia A - B de dos cantidades BCD tratándolas como si estuviesen representadas en binario puro y haciendo el complemento a uno del sustraendo B. Posteriormente se deben hacer las correcciones pertinentes, que consisten en complementar a uno el resultado si es negativo, o sea, si no existe acarreo superior, y en sumar 1010 a aquellos dígitos que cumplían la condición XOR entre su acarreo hexadecimal y el acarreo del dígito superior. La figura 3-28 muestra unos ejemplos y la figura 3-29 un circuito que emplea esta técnica, que necesita de una etapa sumadora para la resta binaria y otra para llevar a cabo las correcciones comentadas.

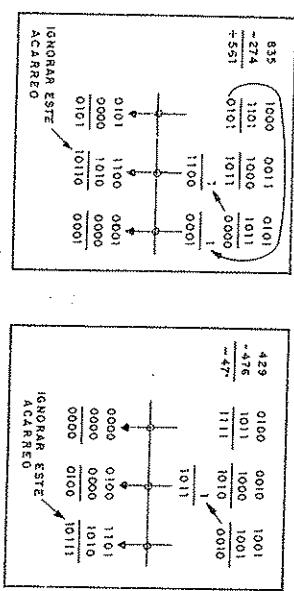


Fig. 3-28. Ejemplos de restas en BCD y algoritmos de corrección

112 / La unidad aritmética y lógica

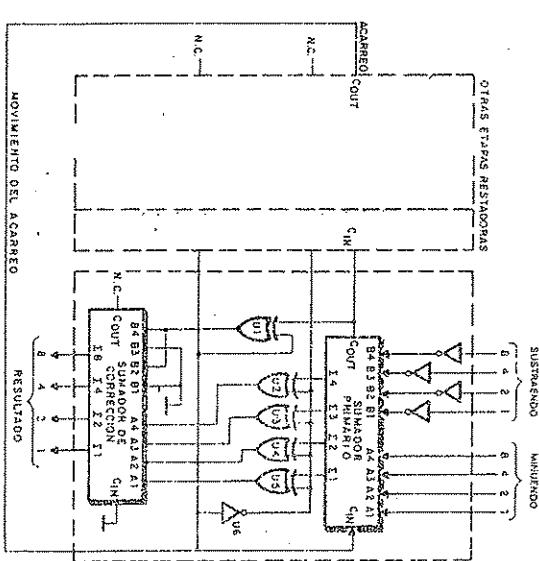
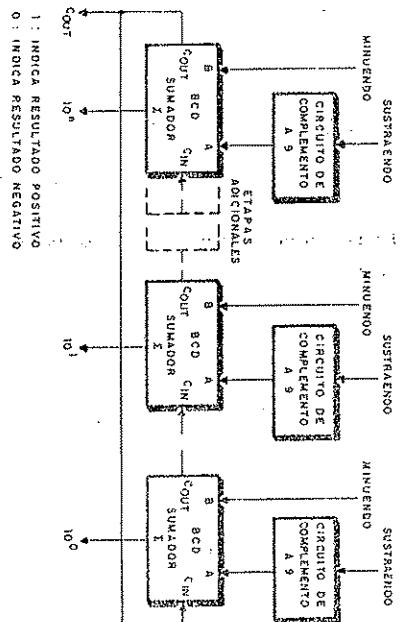


Fig. 3-29. Etapa restadora BCD del tipo de complemento a uno.



1 : INDICA RESULTADO POSITIVO
0 : INDICA RESULTADO NEGATIVO

Fig. 3-30. Típico restador BCD de complemento a nueve.

La unidad aritmética y lógica / 113

Los principios básicos de sustracción por complemento a nueve se presentan de forma gráfica en el diagrama de la figura 3-30. Obsérvese que cada uno de los sumadores BCD tiene un circuito esencialmente igual al de la figura 3-27. El algoritmo de la resta por complemento a nueve consiste en complementar el sustraendo y, después, seguir el algoritmo de suma, excepto que la salida de acarreo de la última etapa (la más significativa) se ha de conectar con la entrada de acarreo de la primera.

Debido a que un restador de complemento a nueve usa los mismos circuitos que un sumador, puede construirse un sumador-restador con menos lógica auxiliar, que la necesaria si se construyese a base de dos circuitos independientes, tales como los de las figuras 3-27 y 3-29.

3.6.3.1. Sumador-restador en coma flotante

En la representación en coma flotante, los n bits de los operandos se dividen en dos grupos, uno de p bits, que expresa la mantisa M y otro, de q bits, que representa el exponente E. Siendo r la base del exponente, el valor del operando es:

$$X = M \cdot r^E$$

Se supone que las mantisas están normalizadas, que es el caso más frecuente.

Dados los operandos A y B de mantisa MA y MB y de exponentes respectivos EA y EB, la realización de la operación de suma o de resta requiere los siguientes pasos:

1. Separación de exponentes y mantisas. Así se puede proceder a su tratamiento individual. Esta separación supone una extensión de signo para que cada elemento MA, MB, EA y EB ocupe una palabra de n bits. Puesto que la representación de los exponentes suele hacerse en exceso a 2^{q-1} , la extensión de signo no se reduce a llenar con ceros los bits adicionales. En el caso de las mantisas, se extenderá el signo de acuerdo con su representación.
2. Resta de los exponentes. El resultado de esta sustracción define el exponente previo del resultado, la mantisa que se debe desplazar para alinear los dígitos de igual peso, así como el número de desplazamientos que requiere la alineación.
3. Alineación de mantisas. Esta operación consiste en desplazar aritméticamente a la derecha el operando de menor exponente. De esta forma, se consigue que MA y MB tengan los dígitos del mismo peso, en posiciones idénticas. Es importante notar, que, si $r = 2$, cada incremento de una unidad en el exponente supone un desplazamiento de 2 lugares, lo cual es normal en bastantes máquinas que tienen $r = 16 = 2^4$. Evidentemente, en este desplazamiento se pierden dígitos significativos del operando menor.

4. Suma o resto de las mantisas. En el caso frecuente de usar la representación de signo y magnitud, habrá que determinar, analizando los signos y la operación pedida, si se hace suma o resta. La obtención de resultados que excedan de los p bits es muy frecuente en esta operación, puesto que, al estar normalizadas las mantisas, la suma de dos cantidades positivas o negativas dará siempre desbordamiento de mantisa en el resultado. Ahora bien, si se ha hecho la extensión de signo, este desbordamiento no se refleja en el operador de suma y resta, que tiene n bits. La forma de eliminar los bits sobrantes, se conoce como redondeo.

5. Normalización del resultado. Se trata de desplazar adecuadamente la mantisa resultado, de forma que su dígito más significativo ocupe el lugar izquierdo de los p bits disponibles. Simultáneamente, se corrige el exponente para reflejar los desplazamientos efectuados. Cuando existe desbordamiento de mantisa en el resultado, esta normalización produce un deslizamiento a la izquierda incrementa en uno al exponente, eliminando así este desbordamiento.

El proceso de normalización puede producir un desbordamiento del exponente, rebasando los límites de su rango ($2^q - 1; 0$). En este caso, se excede el rango total de representación y se debe activar unblestable de desbordamiento y/o acotar el resultado al mayor o menor numero capaz de ser representado.

Aunque es posible diseñar un operador combinacional que contenga los elementos necesarios para realizar las funciones descritas, su complejidad y costo hacen que sea muy excepcional el computador que lo posea.

Este operador necesitaría:

- Un restador de exponentes con un selector para el mayor.
- Un desplazador de mantisas con una serie de multiplexores para seleccionar la mantisa MA o MB a desplazar.
- Un sumador-restador de mantisas.
- Un normalizador compuesto por un incrementador-decrementador de exponente y
- Un desplazador de mantisa.

3.7. OPERACION DE MULTIPLICACION

La multiplicación se suele realizar utilizando un sumador-restador y un algoritmo adecuado. Solamente los computadores muy potentes disponen de operadores específicos de multiplicación.

Se estudian los principales algoritmos empleados para realizar productos.

2

3.7.1. Algoritmo de suma desplazamiento

Su fundamento se basa en el método *manual* de la multiplicación, que ahora se recuerda con un ejemplo.

Ejemplo

$$\begin{array}{r}
 A: \quad 101011 \\
 B: \quad \times 001011 \\
 \hline
 101011 \rightarrow A \text{ al encontrar un } 1 \text{ en } B, \text{ se suma } A \\
 101011 \rightarrow \text{Se suma } A \text{ desplazado, al encontrar un } 1 \text{ en } B \\
 000000 \rightarrow \text{Se suma } 0 \text{ desplazado, al encontrar un } 0 \text{ en } B \\
 101011 \\
 000000 \\
 \hline
 00111011001
 \end{array}$$

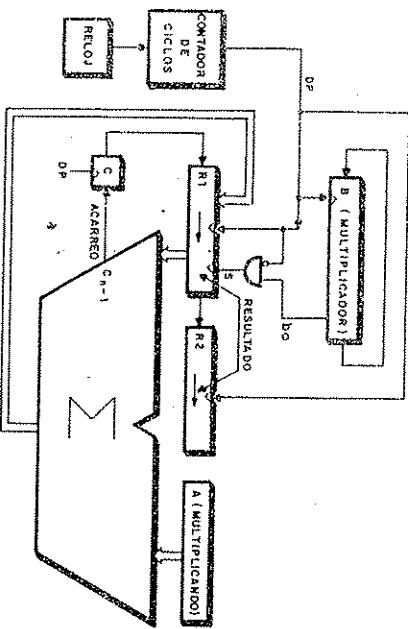


Fig. 3-31. Esquema lógico de un multiplicador que usa el algoritmo de suma desplazamiento. S es la señal que activa R_1 para que la suma se cargue en él. cuando $b_i = 1$. DP indica a B que debe rotar Y a R_1 , R_2 y C, que también deben hacerlo. La señal S introduce R_1 a la entrada del sumador, cuando $b_0 = 1$; cuando $b_0 = 0$ no se hace nada.

3

FASE	A	B	R_1	R_2
NICIAL	10 10 11	00 10 11	000 000 000 000 000	
1º PASO : COMO $b_1 = 1$ SE SUMA A A R_1 , MULTIPLICANDO POR 2 ⁵				
	10 10 11	00 10 11	10 10 11 000 000	R_1 R_2
2º PASO : SE ROTA B . SE DESPLAZAN A LA DERECHA R_1 Y R_2 (2 ⁵)	10 10 11	00 10 11	01 01 01 100 000	R_1 R_2
COMO $b_2 = 1$, SE SUMA A A R_1	10 10 11	10 01 01	00 00 00 100 000	R_1 R_2
ACARREO = 1	10 10 11	10 01 01	000 000 100 000	R_1 R_2
3º PASO : SE ROTA B Y COMO HAY ACARREO ESTE SE DESPLAZA CON R_1 Y R_2	10 10 11	11 00 10	100 000 010 000	R_1 R_2
COMO $b_3 = 0$, SE ROTA B Y SE DESPLAZA EL RESULTADO	10 10 11	01 10 01	010 000 001 000	R_1 R_2
	10 10 11	01 10 01	111 011 001 000	R_1 R_2
	10 10 11	10 11 00	011 101 100 100	R_1 R_2
	10 10 11	01 01 10	001 110 110 010	R_1 R_2
	10 10 11	00 10 11	000 111 011 001	R_1 R_2

Fig. 3-32. Ejemplo de realización de un producto en el multiplicador de la figura 3-31, siguiendo el algoritmo de suma desplazamiento.

El método empleado para realizar una multiplicación es el siguiente:

- Se van inspeccionando, sucesivamente, los bits de B (multiplicador).
 - Si $b_i = 1$, se suma al resultado el operando A (multiplicando), desplazado $(i - 1)$ lugares a la izquierda.
 - Si $b_i = 0$, nada se hace.
- Los multiplicadores reales, en lugar de desplazar A hacia la izquierda, desplazan el resultado parcial a la derecha. Es decir, se comienza desplazando el resultado parcial totalmente a la izquierda (multiplicado por 2^n). Por cada bit de B cuyo valor $b_i = 1$, se desplaza a la derecha el resultado y se suma A. Con estos desplazamientos se elimina el producto por 2^n al dividir n veces por 2 (desplazamientos a la derecha).

Ejemplo

Se hace referencia al diagrama del multiplicador, de la figura 3-31, en el que A y B contienen el multiplicando y el multiplicador y los registros R1 y R2, encadenados, acabarán conteniendo el producto.

En la figura 3-32 se muestran los distintos contenidos de los registros del multiplicador, partiendo de que se desea multiplicar 10101 por 001011.

3.7.2. Algoritmo de sumas y restas

Un número binario que contenga una sola cadena de unos puede descomponerse así:

$$\begin{array}{ccccccccc} x_k & x_{k-1} & x_{k-2} & \dots & x_1 & x_0 \\ x = 11 \dots 1100 \dots 00 = 100 \dots n = 100 \dots 0 = 2^{k+1} - 2^l \end{array}$$

Ejemplo

Descomponer el número 011100.

$$\begin{array}{r} 100000 \\ - 100 \\ \hline 011100 \end{array}$$

Por tanto, se verifica que $011100 = 100000 - 100$.

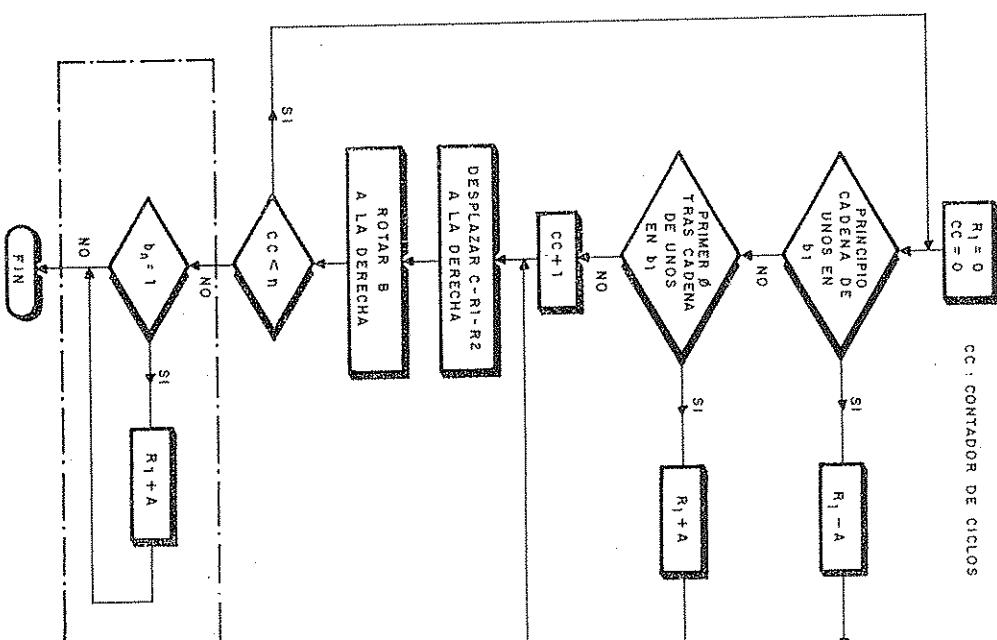


Fig. 3-33. Método que sigue un multiplicador de sumas y desplazamientos para generar el producto de dos números según el algoritmo de sumas y restas.

Un número con varias cadenas de unos puede descomponerse basándose en esta regla.

Ejemplo

Descomponer el número 0110111.

$$\begin{aligned} 0110111 &= 0110000 + 0000111 = (1000000 - 0010000) + \\ &+ (0001000 - 0000001) = (2^6 - 2^4) + (2^3 - 2^0) = -20 + 23 = 2^4 + 2^6 \end{aligned}$$

Cuando se encuentra un principio de cadena de unos, al observar el número de derecha a izquierda, se resta 2^{i-1} , siendo i la posición que ocupa el primer uno, y se suma 2^{i-1} cuando se encuentra un cero, tras una cadena de unos.

Descomponiendo de esta manera un operando del producto, si éste tiene cadenas largas de unos, se realizarán menos operaciones para realizar la multiplicación. En efecto, según el método anterior:

$$\begin{aligned} A \times B &= A \times 0110111 = A (2^0 + 2^1 + 2^2 + 2^4 + 2^5) = \\ &= A \cdot 2^0 + A \cdot 2^1 + A \cdot 2^2 + A \cdot 2^4 + A \cdot 2^5 \end{aligned}$$

hay que realizar 5 sumas.

Sin embargo, descomponiendo B:

$$A \times B = A (-2^0 + 2^3 - 2^4 + 2^6) = -A \cdot 2^0 + A \cdot 2^3 - A \cdot 2^4 + A \cdot 2^6$$

hay que realizar 4 sumas/restas.

En la figura 3-33 se muestra el organigrama de un multiplicador que siga el método expuesto, usando los mismos registros que el multiplicador de suma desplazamiento.

El principal interés de este método es que ofrece la base del *algoritmo de Booth*.

3.7.3. Algoritmo de Booth

Este método permite multiplicar números con signo, expresados en complemento a dos.

Considerese una multiplicación de dos números en complemento a dos, sin tener en cuenta el signo. Si el multiplicando A es negativo, no hay problemas puesto que

las sumas se realizan en complemento. Si el multiplicador es negativo, el resultado del producto será:

$$\begin{aligned} P &= A \cdot (2^n - |B|) = 2^{2n} - 2^{2n} + A \cdot 2^n = A \cdot |B| = \\ &= 2^{2n} - A \cdot |B| + A \cdot 2^n = 2^{2n} - 2^{2n} = 2^{2n} \end{aligned}$$

Resultado correcto

$$P = R + A \cdot 2^n - 2^{2n}$$

El resultado correcto es:

$$R = P - A \cdot 2^n + 2^{2n}$$

Por lo tanto, para corregir el producto se prescinde del bit de acarreo 2^{2n} y se suma $A \cdot 2^n$.

Los números negativos tienen un "1" en su bit de más peso. Consecuentemente, si en el algoritmo de sumas y restas se eliminan los últimos bloques, queda ya dividido $A \cdot 2^n$.

En resumen, este método es equivalente al anterior eliminando los bloques marcados en líneas de punto y raya del diagrama de la figura 3-33 y prescindiendo del bit de acarreo.

3.7.4. Multiplicador rápido

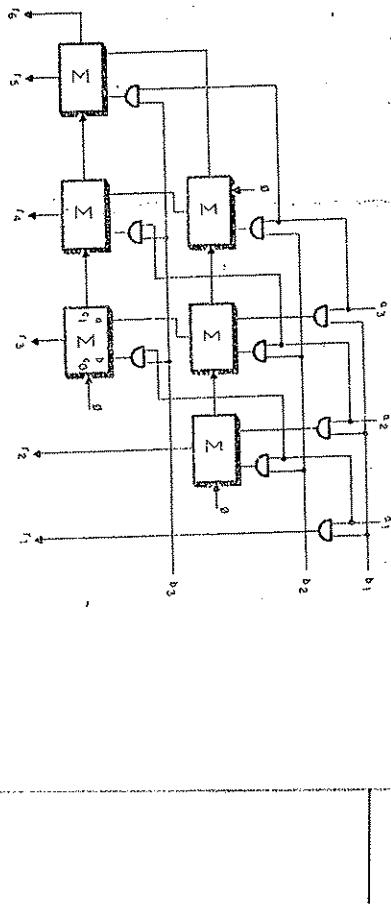


Fig. 3-34. Diagrama esquemático de un multiplicador rápido.

Este multiplicador genera a la vez todos los sumandos que se forman cuando el producto se halla manualmente y los suma directamente, añadiendo el acarreo a la columna siguiente. Figura 3-34.

Debido a la acción de las puertas AND, cuando $b_i = 0$, el sumando es 00 ... 00, y si $b_i = 1$, el sumando es an ... a₁.

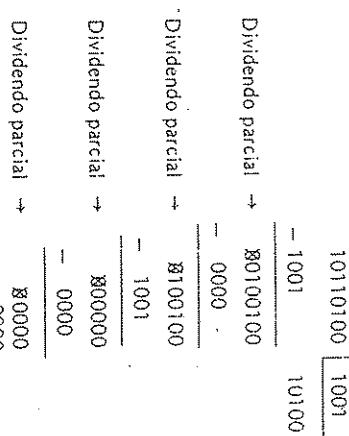
3.8. OPERACION DE DIVISION

Al igual que con la multiplicación, los computadores convencionales utilizan para realizar esta operación un sumador-restador y un algoritmo adecuado.

3.8.1. DIVISION CON RESTAURACION SIN SIGNO

Se ofrece un ejemplo para recordar la mecánica con la que se efectúa manualmente una división.

Ejemplo



Mentalmente se comprueba si el dividendo o dividendo parcial "cabe" entre el divisor. Si "cabe", se introduce un "1" al cociente y si no, un "0", desplazando una posición a la izquierda el registro que guarda el cociente. Si "cabe" (cociente = 1), al dividendo o dividendo parcial se le resta el divisor desplazado a la izquierda totalmente, obteniéndose otro dividendo parcial.

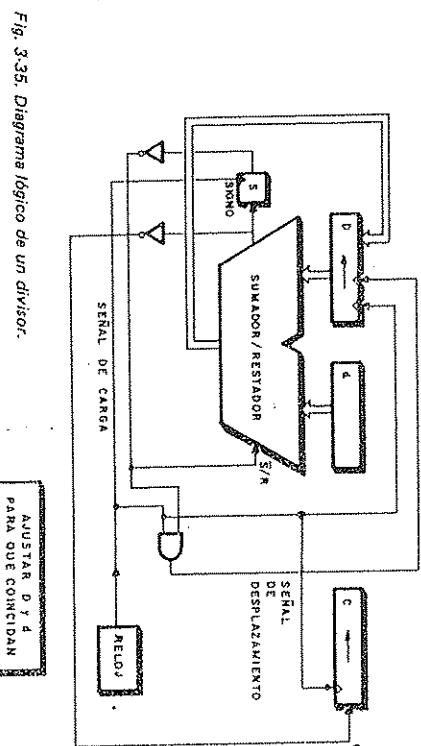


Fig. 3-35. Diagrama lógico de un divisor.

Fig. 3-36. Organigrama operativo correspondiente al divisor de la figura 3-35.

3

Para comprobar si "cable", en un principio hay que desplazar dividendo y divisor en sus registros, de forma que el bit más significativo de cada uno coincida en la misma posición.

Tras hacer la diferencia entre Dividendo (D) y divisor (d) y obtener el nuevo dividendo parcial, D , si el resultado es positivo, significa que ha "cabido" y se introduce un 1 al cociente (C), aprovechando el bit de signo S , al mismo tiempo que se desplaza a la izquierda. Si la resta es negativa, el contenido de D es incorrecto (negativo) y para restituir su valor original se suma $D + d$, originando un "falso ciclo" de suma, en el que la puerta AND impide los desplazamientos.

Cuando el bit de más peso del cociente vale 1, se acaba la división.

En la figura 3-35 se propone el diagrama lógico de un divisor y en la 3-36 el organigrama operativo del mismo.

3.8.2. División sin restauración sin signo

Según los pasos del método anterior:

Initialmente: $D = DP_i$ (dividendo parcial).
Se resta el divisor: $D = DP_i - B$.
Si $D < 0$: $D = (DP_i - B) + B$.

Se desplaza: $D = [(DP_i - B + B)/2] - B = (DP_i - B)/2 + B$.
Al mismo estado se llega con este otro proceso:

Initialmente: $D = DP_i$.
Se resta el divisor: $D = DP_i - B$.
Se desplaza: $D = (DP_i - B)/2 + B$.

Fig. 3-37. Organigrama operativo de la división sin restauración de signo.

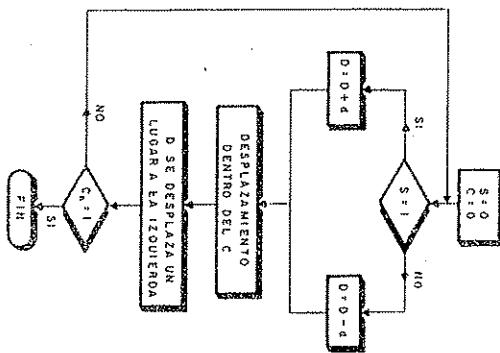


Fig. 3-37. Organigrama operativo de la división sin restauración de signo.

1.2.4 / La unidad aritmética y lógica

Se suma el divisor por haber sido negativo D en el paso anterior:

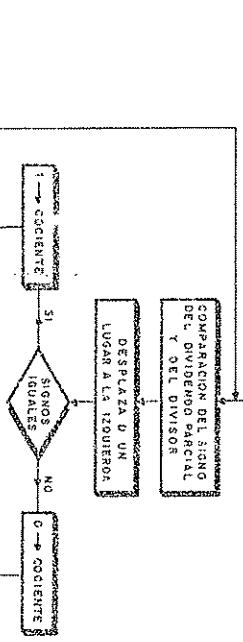
$$D = (DP_i - B)/2 + B$$

El organigrama operativo de este procedimiento se muestra en la figura 3-37.

Este algoritmo realiza la división con operandos con signo. No se demuestra su validez, pero deriva del método de división sin restauración y en la figura 3-38 se presenta el organigrama funcional.

3.8.3. División con signo

cc : contador de ciclos
COMPARACION DEL SIGNO DEL DIVIDENDO PARCIAL Y DEL DIVISOR
DESPLAZAMIENTO DENTRO DEL C
LUGAR A LA IZQUIERDA



3

La unidad aritmética y lógica / 125

3.9. UNIDADES LÓGICAS ARITMÉTICAS INTEGRADAS

Se presentan algunos ejemplos de circuitos integrados comerciales destinados a implementar la Unidad Aritmética y que son capaces de realizar varias operaciones.

3.9.1. Unidad lógico aritmética 74181

El circuito integrado 74181 es un operador combinacional de 4 bits, que permite la realización de 16 operaciones lógicas y 16 aritméticas. Su diseño está basado en un sumador de $4 + 4$ bits, dotado de generador anticipado de acarreo, al que se le han añadido una serie de puertas adicionales para realizar el resto de las funciones.

El diagrama de conexionado del 74181 se ofrece en la figura 3-39 y la nomenclatura que en ella se utiliza tiene el siguiente significado:

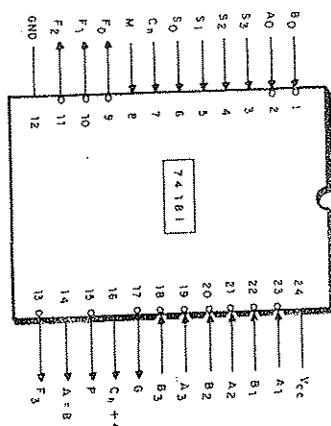


Fig. 3-39. Diagrama de conexionado de la ALU 74181.

M = 1	N = 0 - FUNCIONES ARITMÉTICAS	
S3 S2 S1 SO	FUNCIONES LÓGICAS Cn = 1 (SEN ACARREO)	Cn = 0 (CON ACARREO)
0 0 0 0	$F = \bar{A}$	$F = A$
0 0 0 1	$F = \bar{A} + \bar{B}$	$F = A + B$
0 0 1 0	$F = \bar{A} \cdot B$	$F = A + \bar{B}$
0 0 1 1	$F = 0$	$F = \text{MENOS 1 (COMPLEMENTO A 2)}$
0 1 0 0	$F = \bar{A} \bar{B}$	$F = A \text{ MAS } \bar{A} \bar{B}$
0 1 0 1	$F = \bar{B}$	$F = (A + B) \text{ MAS } \bar{A} \bar{B}$
0 1 1 0	$F = A + B$	$F = A \text{ MENOS } B \text{ MENOS 1}$
0 1 1 1	$F = \bar{A} \bar{B}$	$F = A \bar{B}$
1 0 0 0	$F = \bar{A} + B$	$F = A \text{ MAS } AB$
1 0 0 1	$F = A \oplus B$	$F = A \text{ MAS } B \text{ MENOS 1}$
1 0 1 0	$F = A \oplus B$	$F = A \bar{B} \text{ MENOS 1}$
1 0 1 1	$F = \bar{A} B$	$F = A \bar{B}$
1 1 0 0	$F = \bar{C}$	$F = A \text{ MAS } AB \text{ MAS 1}$
1 1 0 1	$F = A \oplus B$	$F = A \text{ MAS } B \text{ MAS 1}$
1 1 1 0	$F = A + \bar{B}$	$F = (A + \bar{B}) \text{ MAS } AB$
1 1 1 1	$F = A + B$	$F = (A + \bar{B}) \text{ MAS } A \text{ MAS 1}$

Fig. 3-40. Tabla que especifica las 32 operaciones que puede efectuar el 74181.

La figura 3-40 muestra una tabla con las 32 funciones que puede realizar el 74181.

Los circuitos 74181 se pueden conectar en cascada para construir un operador del ancho de palabra que se deseé.

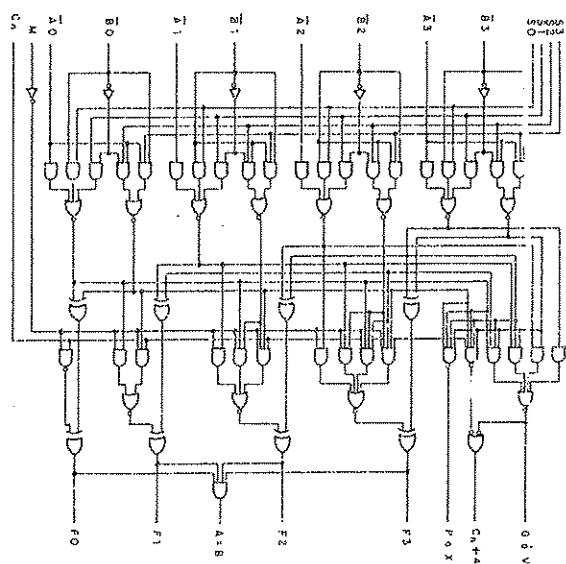


Fig. 3-47. Diagrama lógico de la Unidad Lógico Aritmética 74181.

3.9.2. Unidad aritmética Am 29203

Se trata de un circuito integrado que opera con datos de 4 bits, aunque, si se desea operar con palabras de n bits, basta colocar en cascada $n/4$ pastillas AM 29202.

Consta de los siguientes elementos:

- Un operador aritmético-lógico para datos de 4 bits.
 - 16 registros de 4 bits a los que se puede acceder por dos puertas.
 - Un desplazador, que ejecuta desplazamientos lógicos y aritméticos de una posición en un registro especial, Q.
 - Detector de resultado cero.
 - Lógica de control que decodifica las órdenes recibidas.

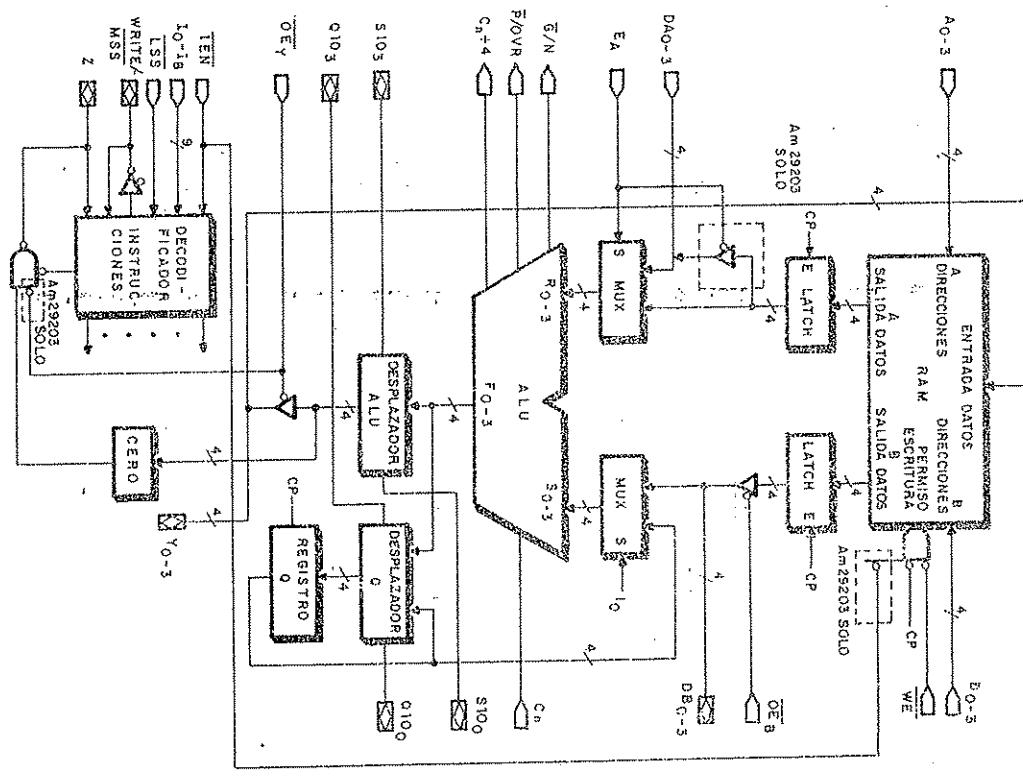


Fig. 3-42. Diagrama lógico del Ám 29203.

Además de operaciones especiales de ayuda en la implementación de algoritmos para multiplicación y división, normalización de coma flotante, dispone de 8 funciones lógicas y 5 aritméticas:

LOGICAS

ARITMETICAS

- | | |
|------------------|-----------------------|
| $F = 1$ | $F = S - R - 1 + C_n$ |
| $F = 0$ | $F = R - S - 1 + C_n$ |
| $F = R \cdot S$ | $F = R + S + C_n$ |
| $F = R \oplus S$ | $F = S + C_n$ |
| $F = R + S$ | $F = R + C_n$ |
| $F = R + S$ | $F = R + S$ |
| $F = R \cdot S$ | $F = R \cdot S$ |
| $F = R + S$ | $F = R + S$ |

La figura 3.42 muestra el diagrama lógico del Am 29203.

EJERCICIOS

Ejercicio 3.1

Responde a las siguientes cuestiones:

- Averiguar los valores de la salida y del acarreo si se supone una ALU 74181 que tiene en sus entradas las siguientes señales:
 - operando 1 = 1011
 - operando 2 = 0100
 - código de operación = 0010
 - carry de entrada = 1
 - tipo de función (lógica/aritmética) = 0
- Partiendo de sumadores BCD diseñar el circuito que realice la adición de números de 2 dígitos. Sobre dicho circuito indicar el estado de todas las líneas, supuestos los siguientes números BCD: 45 y 52.
- A partir de los siguientes números en coma flotante:

01010	100	0
MANTISA	EXPON	SIGNO

10100	010	0
MANTISA	EXPON	SIGNO

Indicar detalladamente los pasos a seguir para realizar su suma.

- Al realizar la resta de dos números de 3 dígitos expresada en complemento a 1, ¿cómo se detecta el hecho de que el resultado sea negativo?

- Indicar el tipo de operación, monádica o diádica, en los siguientes casos:

- cambio de signo
- desplazamiento concatenado de dos registros
- desplazamiento lógico de 4 bits a la izquierda
- extensión de signo.

3

Ejercicio 3.2

Diseñar un operador de cambio de signo que realice la operación en complemento a 1 ó complemento a 2, dependiendo de una señal (0 → CA2, 1 → CA1). Deducir las ecuaciones del sistema y desarrollar un ejemplo con el número 01100.

Ejercicio 3.3

Diseñar un operador de desplazamiento sobre un registro de 8 bits, que permita realizar las siguientes operaciones:

- Desplazamiento nulo y 2 bits a la izquierda y a la derecha.
- Desplazamiento de 1, 2 y 4 bits a la izquierda.

Ejercicio 3.4

Diseñar un operador lógico complejo que a partir de registros de 8 bits realice la siguiente operación:

$$bi = [(ai_{i-2} + ai_{i+3}) \cdot ai + ai_{i-3} \cdot ai_{i+2}]$$

Ejercicio 3.5

A partir del sumador-restador de la figura 3.21 del libro, diseñar un sumador-restador en complemento a 2, incluyendo la condición de desbordamiento.

Ejercicio 3.6

Sobre la figura 3.30 del libro, indicar qué sucede cuando el acarreo del último bit es 1. Si no le parece correcto ¿qué modificaciones propondría? ¿Cuál le parece la más acertada? ¿Por qué?

Ejercicio 3.7

Dados los operandos: 11010 y 01101, se pide:

3

Ejercicio 3.8

Dadas las siguientes especificaciones para un sumador serie de 3 bits:

- cada ciclo de reloj dura 10 ns
- todos los registros y flip-flops tienen una señal de clock individual.

Se pide desarrollar el cronograma o diagrama de tiempos, incluyendo en el mismo los resultados parciales Ci, Ci – 1 y Si.

Ejercicio 3.9

Diseñar un circuito que responda a la siguiente operación:

$$\begin{array}{l} A \rightarrow \text{operando de 5 bits} \\ B \rightarrow \text{operando de 5 bits} \\ \left\{ \begin{array}{l} [(A + B) \times 8] \rightarrow A \\ \dots \\ \dots \end{array} \right\} / 4 \end{array}$$

Indicar detalladamente las líneas utilizadas y suponer que no se dispone de un circuito divisor.

Si A = 10000 y B = 01110 indicar los valores parciales de la operación y el resultado final, comprobando éste con el calculado aritméticamente.

Ejercicio 3.10

Incluir y corregir las especificaciones necesarias para que el algoritmo representado en la figura 3-33 del libro sea efectivo y funcione.

Ejercicio 3.11

Efectuar la siguiente multiplicación en binario: 101001 × 011010

3

- a) Mediante el algoritmo de sumas-desplazamientos.
 b) Mediante el algoritmo de sumas-restas.
 c) Mediante el algoritmo de Booth, supuestos números en complemento a 2.

Ejercicio 3.12

Realizar la siguiente división utilizando el método de restauración sin signo,
 $11010000 : 1000$.

Memorias

4.1. INTRODUCCION

La memoria es un bloque fundamental del computador, cuya misión consiste en almacenar los datos y las instrucciones. La figura 4.1 presenta la estructura del computador, mostrando su memoria principal, órgano que almacena los datos e instrucciones de los programas en ejecución.

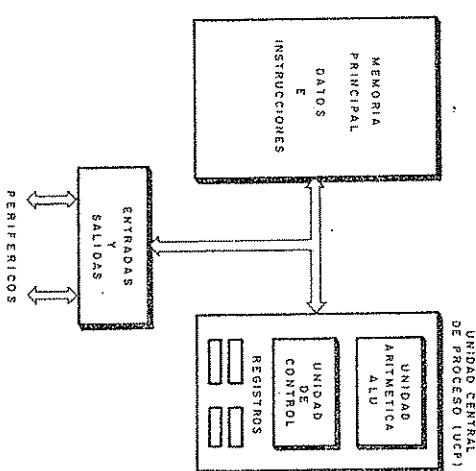


Fig. 4.1. Bloques fundamentales del computador.

En la figura 4-1, la UCP, mediante su Unidad de Control y con el apoyo de los registros de trabajo, se encarga de buscar las instrucciones y datos en la memoria principal, realizar las operaciones adecuadas con la ALU y sacar o introducir información al desde los periféricos exteriores a través de los módulos de Entrada y Salida. Las transferencias entre bloques del computador se hacen por un conjunto de líneas, que reciben el nombre de colectores o buses.

A veces, la memoria principal no tiene la suficiente capacidad para contener todos los datos e instrucciones, en cuyo caso se precisan otras memorias auxiliares o secundarias, que funcionen como periféricos del sistema y cuya información se traspasa a la memoria principal cuando se necesita.

La memoria sólo puede realizar dos operaciones básicas: *lectura* y *escritura*. En la lectura, el dispositivo de memoria debe recibir una dirección de la posición de la que se quiere extraer la información depositada previamente. En la escritura, además de la dirección, se debe suministrar la información que se desea grabar.

La mayoría de las memorias sólo son capaces de almacenar en sus celdas un bit 1 o 0. Por tanto, se denomina *punto de memoria* al elemento que almacena un bit.

La implementación práctica de un punto de memoria es muy diversa y depende de la tecnología de fabricación. Así, existen puntos de memoria realizados con flip-flops, elementos perfectamente definidos en el espacio. También existen puntos de memoria sin una posición fija sobre el soporte físico; se desplazan a lo largo del mismo, como ocurre con las memorias de burbujas magnéticas.

4.2. EVOLUCION HISTORICA

En los calculadores de la década de los 30 se emplean *tarjetas perforadas* como memorias. La dirección de las posiciones quedaba determinada por la posición de ruedas dentadas. Luego se emplearon *relés electromagnéticos*.

El computador ENIAC utilizaba, en 1946, *válvulas electrónicas de vacío* para construir sus bistables que actuaban como punto de memoria. Además, tenía una ROM de 4 Kbits construida a base de resistencias.

Al comienzo de la década de los 50, se usaron las *líneas de retardo de mercurio* con 1 Kbit por línea, como memoria. Igualmente se empleó el *tubo de Williams*, que tenía una capacidad de 1,200 bits y consistía en un tubo de rayos catódicos con memoria.

El UNIVAC I introdujo en 1951 la primera unidad comercial de *banda magnética*, que tenía una capacidad de 1,44 Mbit y una velocidad de 100 pulgadas/s.

El primer computador comercial que usó como memoria principal al *tambor magnético* fue el IBM 650 en 1954. Dicho tambor giraba a 12,500 r.p.m. y tenía una capacidad de 120 Kbits.

En 1953, el MIT dispuso de la primera memoria operativa de *ferritas*, que fue muy popular hasta mediados de los años 70.

Fue IBM, en 1968, quien diseñó la primera memoria comercial de *semiconductores*. Tenía una capacidad de 64 bits.

También, el modelo 350 de IBM en 1956 fue quien utilizó el primer *disco con brazo móvil y cabeza flotante*. Su capacidad era de 40 Mbits y su tiempo de acceso, de 500 ms.

Tecnologías nuevas, como la de *burbujas magnéticas*, efecto Josephson, *acoplamiento de carga*, de tipo óptico y otras, compiten en la actualidad por desplazar a las memorias de semiconductor basadas en silicio, que ya han alcanzado capacidades superiores a 1 Mbit en una pastilla con rápidísimo tiempo de acceso y coste razonable.

4.3. NIVELES DE JERARQUIA

La memoria principal, de acuerdo con la figura 4-1, abastece a la UCP de datos e instrucciones. Como la UCP se construye con circuitos integrados rápidísimos, obliga a que la memoria emplee un tiempo mínimo en grabar y obtener información. Sin embargo, las memorias rápidas son de pequeña capacidad, por lo que existirá un *nivel* veloz, pero de poco tamaño, en el computador, que actúe como memoria principal así como niveles sucesivos de menor rapidez y mayor capacidad.

La información se deposita en uno de los niveles de acuerdo a su prioridad de uso. Así, cuando un programa o unos datos de archivo son poco empleados, se almacenan en el *nivel inferior* más lento y de mayor capacidad; si en un momento determinado se necesitan, se trasladan al *nivel superior* más rápido.

Además de la velocidad operativa de una memoria o *tiempo de acceso* a la información y la *capacidad de almacenamiento*, existe otra propiedad fundamental que influye en la determinación de su nivel jerárquico; se trata del *coste/bit*. Estas tres características son contrapuestas, es decir, que si se desea un tiempo de acceso muy bajo, el coste/bit será alto y la capacidad o tamaño será bajo, tal como se refleja en los gráficos de las figuras 4-2 y 4-3.

La memoria ideal debe poseer una capacidad elevada junto a un tiempo de acceso muy pequeño y un precio reducido. Dichas características, por ahora, son tecnológicamente irreconciliables.

Hay que recurrir a una "especialización", usando memorias muy rápidas y de pequeña capacidad en aquellas partes del sistema en las que la velocidad es el factor predominante. Las memorias lentas y de gran capacidad se destinarán a las partes donde prime o sobresalga el factor capacidad.

COSTE
(PESOS/BYTE)

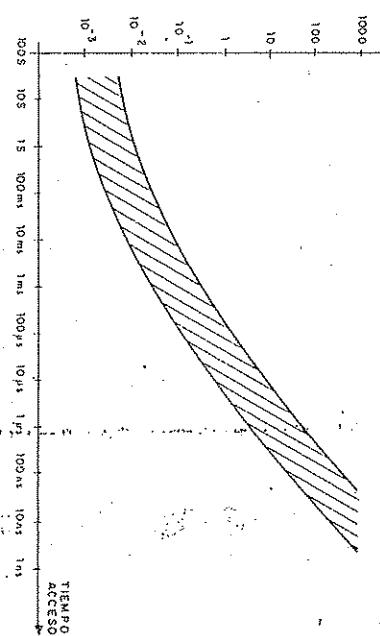


Fig. 4.2. La reducción del tiempo de acceso en las memorias incrementa su precio.

Se pueden distinguir los siguientes niveles jerárquicos:

- a) *Memorias cache o tampon*. Son de acceso aleatorio, de baja capacidad (16 Kbyte) y muy rápidas. Sus tiempos de acceso oscilan entre los 20 y los 200 ns. Tecnológicamente, son memorias a base de semiconductores.
- b) *Memoria principal*. De acceso aleatorio, capacidad no muy alta (128 K - 106 Mbyte) y tiempo de acceso comprendido entre los 200 ns y los 2 μ s. Generalmente están fabricadas con semiconductores.

- c) *Memorias intermedias*. Suelen ser de acceso secuencial, aleatorio o híbridas, con un tiempo de acceso de varios ms y con una elevada capacidad (10 G). Realmente este nivel no se ha extendido mucho y suele ser sustituido por un nivel de memoria secundaria referente al disco de cabeza fija y de uso exclusivo del Sistema Operativo y por el nivel de memoria auxiliar, que incluye a los discos móviles y las cintas.

- d) *Memorias auxiliares*. Son lentes y de gran capacidad. Como ejemplos representativos se pueden citar los discos de cabeza móvil y las cintas magnéticas.

En la figura 4.4 se muestra un gráfico que recoge la jerarquía de los diversos tipos de memoria.

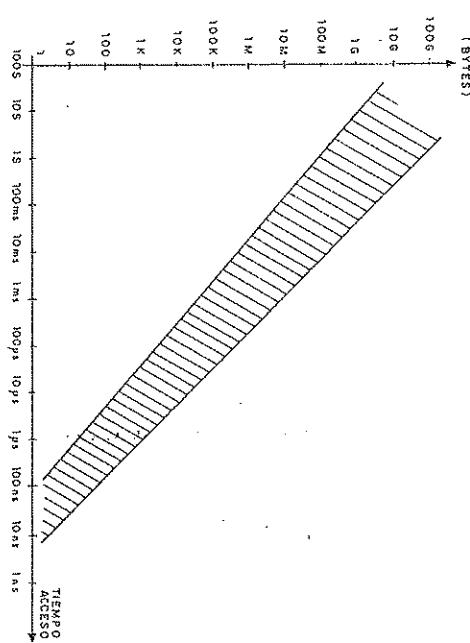


Fig. 4.3. Las altas capacidades de almacenamiento imponen tiempos de acceso elevados.

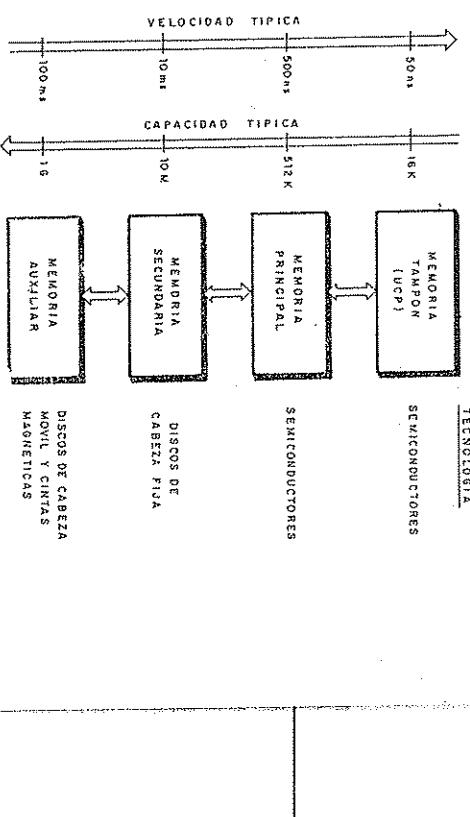


Fig. 4.4. Niveles de jerarquía en la memoria.

En la tabla de la figura 4-5 se recoge una clasificación jerárquica de la memoria, junto a sus características más relevantes.

NIVEL DE JERARQUÍA	CAPACIDAD (BYTES)	VELOCIDAD	ACCESO
REGISTROS	6 - 100	10 - 30 ns	ALEATORIO
MEMORIA CÁSCHE	1K - 20K	50 - 200 ns	ALEATORIO
MEMORIA PRINCIPAL	10K - 10M	200 - 1000 ns	ALEATORIO
MEMORIA SECUNDARIA	1M - 16	10 - 100 ms	DIRECTO
MEMORIA AUXILIAR DE DISCO	50M - 10G	100 ms	DIRECTO
MEMORIA AUXILIAR DE CINTA	ILIMITADO	MINUTOS	SECUENCIAL

Fig. 4-5. Tabla detallada de los niveles de jerarquía de memoria, con indicación de sus parámetros típicos.

4.4. UTILIZACION PRACTICA DE LAS MEMORIAS

A nivel de sistema, los dispositivos de memoria pueden contemplarse como bloques a los que hay que suministrar una dirección y una señal de control para especificar la operación que se desea realizar; además, hay que enviar o recibir el dato o bloque correspondiente. La figura 4-6 representa las líneas de comunicación precisas en los dispositivos de memoria.

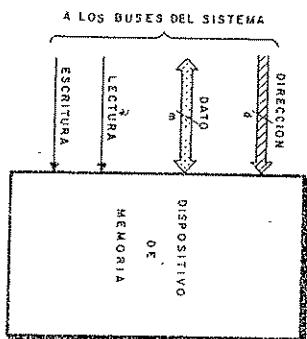


Fig. 4-6. Símbolo de un dispositivo de memoria con sus líneas de comunicación con el sistema.

Las memorias de acceso aleatorio utilizan acceso individualizado a nivel de palabra. Para hacer la conexión con el resto del sistema, se suelen emplear dos registros, uno que almacena la dirección y el otro, el dato a leer o escribir. Figura 4-7.

El registro D de direcciones (figura 4-7) tendrá una longitud d, tal que 2^d sea mayor o igual que la capacidad de la memoria. El registro M de datos tendrá una longitud m igual al tamaño de la palabra con que trabaja la memoria. Ambos registros se encuentran conectados a los buses del sistema.

Las señales de control indicarán a la memoria el tipo de operación a realizar, así como el comienzo de dicha operación. La memoria, a su vez, puede generar una señal de control de fin de la operación, muy útil en la sincronización con el sistema.

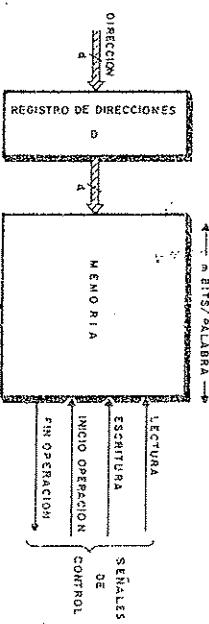


Fig. 4-7. Símbolo de una memoria de acceso aleatorio, con sus señales de control.

Las memorias de acceso aleatorio forman la memoria principal del sistema y su conexión a los buses del mismo, es directa. Sin embargo, las memorias dinámicas forman la memoria auxiliar y su conexión se efectúa mediante una unidad de intercambio.

Existe la posibilidad de constituir memorias con varias parejas de registros D y M, llamadas *memorias multipuerta*.

4.5. FUNDAMENTOS BASICOS DE LAS MEMORIAS

Para que un dispositivo de memoria pueda realizar sus funciones principales de lectura y escritura, ha de disponer de tres elementos:



1. **Medio o soporte:** Donde se almacenan estados de energía diferentes, que codifican la información guardada.
2. **Transductores:** Tanto para la lectura como para la escritura, que sean capaces de generar la energía necesaria en la grabación y detectar el estado existente en la lectura.
3. **Mecanismo de direccionamiento:** Permite leer y escribir la información en el lugar y tiempos precisos.

4.5.1. Medio o soporte

Para que un medio sirva como almacenamiento de información, ha de presentar, al menos, dos estados estables (o semiestables), que se caractericen por una magnitud física discreta, por ejemplo, el momento magnético, la carga eléctrica, la corriente, etc. Además, se ha de poder pasar de un estado a otro mediante la aplicación de una señal (energía) externa, y se ha de poder detectar el estado existente en un momento determinado. Finalmente, ha de ser posible cambiar de estado tantas veces como se quiera sin que sufra o se modifique el medio.

Los medios o soportes más empleados son los de tipo magnético. Se construyen de forma que presenten una dirección preferente de magnetización, que llamarémos A → B. Si se magnetiza mediante un campo externo el medio en el sentido A → B, tenemos un estado, mientras que, si se magnetiza en sentido contrario A → B, tenemos el otro.

Los medios pueden ser *discretos* o *continuos*. En el primer caso se emplea un dispositivo físico aislado para almacenar cada bit, mientras que en el segundo se almacenan unos a continuación de otros en el mismo medio, distinguiéndose entre sí por el método de escritura. El toro de ferrita y el blostable son medios discretos, y la cinta o el disco, un medio continuo.

En general, los medios discretos son más caros que los continuos, pero requieren unos transductores más sencillos y presentan un direccionamiento más rápido y sencillo.

Los medios pueden clasificarse atendiendo al tiempo que la información permanece grabada sobre ellos. Hay tres tipos:

- 1) *La información se mantiene de forma permanente.* Suelen ser lo normal en medios magnéticos. Se dice que la memoria es *voltátil*.
- 2) La información desaparece cuando se deja de alimentar o suministrar energía a la memoria. Por ejemplo, esto ocurre con las memorias con semiconductores, tipo R.A.M. Entonces la memoria es *voltátil*.
- 3) *La información se va degradando paulatinamente y lleva un momento en que no se puede leer.* Esto sucede con los condensadores, que, cuando están cargados con una tensión, pueden representar un estado, y descargados, el otro. Sin embargo, los condensadores, al no ser perfectos, van descargándose hasta que pierden la información. Para que estos tipos de memoria sean útiles, hay que recargar a todos sus elementos periódicamente para compensar las pérdidas. Esta acción se conoce como *refresco*.

Finalmente, y como ya se ha dicho, hay medios que guardan una equivalencia entre una posición física y un punto de memoria, es decir, la información permanece fija en el medio, como en un disco, pero hay otros medios en los que la información se traspasa, como en las memorias de burbujas magnéticas.

4.5.2. Transductor

Hay transductores de lectura y de escritura. La función de los de escritura es suministrar la energía necesaria al medio para que adopte el estado deseado. Por otro lado, la función de los transductores de lectura es detectar las magnitudes físicas para reconocer el estado en que se encuentra el medio.

Los transductores suelen ser caros, por lo que se tiende a reducir su número. Pero debe existir un compromiso entre el número de transductores y la velocidad de la memoria; si hay pocos transductores, la memoria será lenta.

Los transductores pueden estar físicamente *unidos al medio* o pueden ser *pendientes* de él. En el primer caso, hay un cableado que permite acceder al punto de memoria deseado. En el segundo, el punto de memoria al que se quiere acceder debe posicionarse frente al transductor para realizar la operación de lectura y escritura. Como ejemplos de estas dos situaciones se pueden citar las *memorias de ferrita*, en la cual los transductores están unidos de forma fija a las ferritas por conductores eléctricos, y la *cinta magnética*, en la que se debe enfrentar al cabezal de lectura/escritura la zona a la que se pretende acceder.

Reciben el nombre de memorias *estáticas* las que tienen el transductor unido al medio y *dinámicas* aquéllas en las que debe moverse el medio para posicionar frente al transductor.

Los transductores de las memorias dinámicas, por lo general, trabajan con un nivel de señal muy bajo, por lo que son más caros que los filos. De todas formas, la relación *número de bits por sensor* es mucho mayor que en las estáticas, en cuales los transductores están cableados a unos pocos puntos de memoria, porque el coste final por bit es mucho menor.

4.5.3. Mecanismo de direccionamiento

Se encarga de seleccionar el punto de memoria deseado.



En una memoria estática el mecanismo de direccionamiento es inherente a su propia construcción. El conexionado de los transductores, junto a la selección de estos últimos, especifica de forma inequívoca el punto de memoria al que se accede. Por este motivo, este tipo de memorias reciben la denominación de *memorias de direccionamiento cableado*.

Por el contrario, en las memorias dinámicas, al estar compartidos los transductores, no existe la mencionada relación. La selección se consigue mediante la Unidad de Control que interpreta una información adicional, que se llama *información de direccionamiento*, y que se almacena junto a los datos.

A continuación se presentan los diferentes tipos de direccionamientos:

4.5.3.1. Direccionamiento en memorias estáticas o "cableado"

El mecanismo de direccionamiento de una memoria estática ofrece dos formas típicas llamadas 2D y 3D. Para analizarlas se considera que la memoria tiene 2^m palabras de n bits cada una. Se trata de acceder simultáneamente a los n bits de una palabra, cuya dirección precisa de m bits para que quede definida.

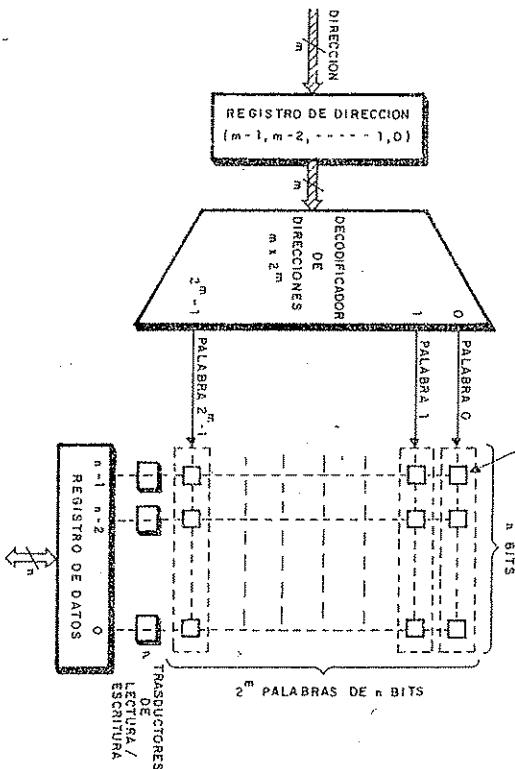


Fig. 4-8. Organización de una memoria con direccionamiento 2D.

Direccionamiento 2D: Tal como se muestra en la figura 4-8, en este direccionamiento todos los bits de la misma posición en cada palabra (bits 0, 1, ...) están conectados a la misma pareja de transductores. Habrá n parejas de transductores.

Para seleccionar la palabra deseada (figura 4-8) se decodifican los m bits de dirección en un decodificador $m \times 2^m$, que tiene una señal de salida individualizada para cada palabra de memoria.

Se usa la misma conexión para la lectura que para la escritura, bastando activar al transductor correspondiente para definir la operación.

Direccionamiento 3D: Como indica la figura 4-9, se establecen n planos de memoria (uno para cada bit de la palabra). Dentro de cada plano se selecciona el punto de memoria haciendo coincidir las líneas de selección X e Y .

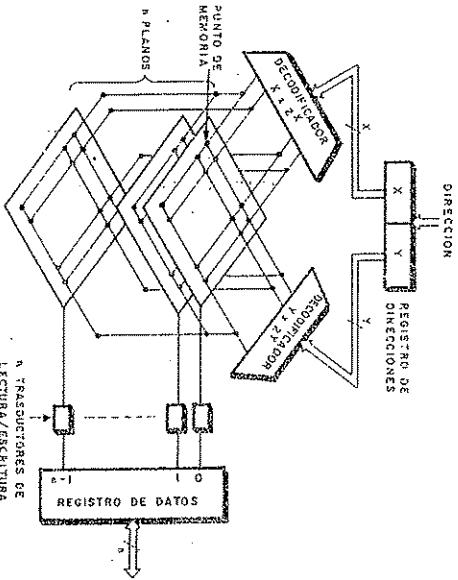


Fig. 4-9. Organización de una memoria con direccionamiento 3D.

La dirección de m bits se divide en dos partes m_x y m_y , que se decodifican en dos decodificadores de 2^{m_x} y 2^{m_y} salidas, respectivamente. Las señales de salida $2^{m_x} + 2^{m_y}$ se usan como coordenadas que seleccionan entre las $2^{m_x+m_y} = 2^m$ posiciones de cada plano de la memoria.

Cada plano tiene una pareja de transductores de lectura y escritura, que están conectados a todos los puntos del plano. La ventaja del método 3D es que dos decodificadores de m_x y m_y entradas son mucho más sencillos que uno de $m_x + m_y = m$ entradas. Como inconveniente, el punto de memoria es más complejo, puesto que debe ser capaz de activarse sólo cuando sus dos líneas x e y estén activas.

Direcciónamiento 2D/1/2:

Es un direccionamiento en el que se emplean más transductores que los n necesarios en los casos anteriores. La selección de los puntos de memoria se hace mediante la coincidencia de una línea de decodificación X y la elección del bloque de transductores adecuado.

4.5.3.2. Direccionamiento en memorias dinámicas y de propagación

En estas memorias hay que añadir una información adicional de direccionamiento, que se almacena ocupando parte de los puntos de memoria.

La técnica más empleada consiste en empaquetar la información en bloques o regístritos a los que se añade una cabecera, que, entre otras cosas, incluye una identificación del bloque o registro. Figura 4-10.

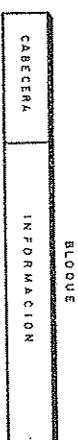


Fig. 4-10. La información se empaqueta en un bloque que contiene una cabecera.

Finalmente, se establecen las diferencias entre los dispositivos de *acceso secuencial*, frente a los de *acceso o direcciónamiento directo*. En los primeros solo existe un transductor, por lo que, para acceder a una posición, se debe recorrer todo el medio hasta alcanzar la posición deseada. Esto sucede con la cinta magnética. En los dispositivos de acceso directo hay varios transductores colocados en diversas posiciones, de forma que se puede pasar de una posición a otra seleccionando el transductor adecuado. Dentro de la zona asignada a cada transductor, el funcionamiento sigue siendo de tipo secuencial, pero, al ser menor esta zona, el tiempo perdido en esperar que la posición deseada alcance el transductor, es menor que si todo el dispositivo fuese de acceso secuencial. Ejemplos de memorias dinámicas de acceso directo son los discos y los tambores, y de memoria de propagación de acceso directo, las memorias de burbujas magnéticas.

4.6. CARACTERISTICAS DE LAS MEMORIAS

Se ofrece un estudio detallado de las 4 características que mejor definen y diferencian a los diferentes tipos de memorias:

- Duración de la información.
- Modo de acceso.
- Velocidad.
- Capacidad o tamaño.

También existen otras propiedades secundarias que, frecuentemente, se deben tener en cuenta como son la potencia o consumo, el coste, la densidad de bits y las recomendaciones de uso, entre otras.

4.6.1. Duración de la información

En relación con la permanencia de la información grabada en las memorias, hay 4 posibilidades:

4.6.1.1. Memorias permanentes

Son las que contienen siempre la misma información y no pueden borrarse. La información puede haberse grabado en el proceso de fabricación de la memoria o puede haberse efectuado posteriormente en un proceso de grabado destructivo o permanente. Como ejemplos de este tipo de memorias se pueden citar las tarjetas y cintas de papel perforado y las memorias de semiconductores tipo ROM.

4

En contraposición a este tipo de memorias de sólo lectura, están las de lectura-escritura, que pueden grabarse cuantas veces se quiera. Como alternativa intermedia están las memorias que, para borrarse, precisan de un complejo proceso especial, como los tipos EEPROM.

4.6.1.2. Memorias volátiles

Precisan estar continuamente alimentadas de energía. Si se corta dicho suministro se borra la información que poseen. En contraposición, están las memorias no volátiles, en las que permanece la información aunque se elimine la alimentación.

4.6.1.3. Memorias de lectura destructiva

Son memorias cuya lectura implica el borrado de la información. Para que la información no desaparezca, se requiere una escritura, posterior a la lectura, que vuelva a grabar lo que se ha leído. El ejemplo clásico de este tipo de memorias es el de ferritas. Entre las memorias de lectura no destructiva destacan las de semiconductores, discos y banda magnética.

4.6.1.4. Memorias con refresco

La información sólo dura un cierto tiempo. Para que la información no desaparezca, hay que regrabar la información de forma periódica (refresco).

4.6.2. Modo de acceso

El modo de acceso de las memorias puede ser por palabras o por bloques. En el primer caso, en cada operación de lectura o escritura se transmite una sola palabra. En el segundo, cada operación implica un bloque completo de información.

El acceso por palabra, llamado *acceso aleatorio*, solamente se emplea en las memorias estáticas, en las que el tiempo para direccionar cualquier punto de memoria siempre es el mismo. Por esto motivo se suelen denominar a las memorias estáticas, memorias de acceso aleatorio o de tipo RAM (Random Access Memory).

El acceso por bloques se utiliza en memorias dinámicas y de propagación. Este acceso tiene su justificación en que estas memorias tardan bastante en alcanzar el punto donde está almacenado el primer dato de un bloque, pero tardan muy poco en acceder a los siguientes. Para obtener un buen rendimiento del dispositivo, conviene acceder en bloques de un tamaño determinado, que justifiquen el tiempo perdido en el acceso a la zona del bloque.

4

Los computadores actuales disponen de una memoria principal de acceso aleatorio y una memoria secundaria de acceso por bloques.

4.6.3. Velocidad

En términos generales, la velocidad de la memoria será el "tiempo que tarda en realizar una operación de lectura o escritura". Debido a las peculiaridades de las memorias, la cuantificación de este concepto requiere analizar los tiempos de las distintas fases de las operaciones de lectura y escritura.

Se analizan en principio las memorias estáticas o de direccionamiento cableado. En ellas, dado que todos los puntos de almacenamiento están cableados de forma equivalente, los tiempos son independientes de la dirección.

El *tiempo de acceso* o *tiempo de lectura*, t_A , es el tiempo que tarda la memoria en suministrar una palabra desde el momento en que se proporciona la dirección y la señal de lectura.

El *tiempo de escritura*, t_E , es el tiempo que tarda la memoria en grabar información desde el momento en que se suministra la dirección, la información y la señal de escritura.

En las memorias dinámicas y de propagación hay que considerar el tiempo que se tarda en hacer coincidir el transductor con la información y la velocidad a que se escriben o leen palabras consecutivas.

En las memorias dinámicas se llama "tiempo de acceso" al tiempo que tarda el transductor móvil en alcanzar la posición deseada más el tiempo que tarda la zona buscada del medio en llegar a estar frente al transductor. Para las memorias de propagación, es el tiempo que tarda la información buscada en pasar frente al transductor.

En ambos casos, una vez accedida la zona deseada, pueden escribirse o leerse una serie de palabras a alta velocidad. Se hace referencia a la velocidad o cadencia de transferencia y se expresa en bytes/s, K bytes/s o Mbytes/s. Esta cadencia está determinada por la velocidad del soporte, así como por el espacio ocupado por cada punto de memoria (densidad de grabación).

4.6.4. Capacidad o tamaño

Se denomina "capacidad" o "tamaño" a la cantidad de información que puede almacenar una memoria.

La capacidad se puede expresar en unidades de bits, bytes o palabras, aunque lo más corriente es hacerlo en bytes. Se emplean los prefijos K = kilo ($1\text{K} = 2^{10} = 1.024$ unidades), M = Mega ($1\text{M} = 1.024\text{K}$) y G = Giga ($1\text{G} = 1.024\text{M}$).

En las memorias de direccionamiento cableado existe una relación directa entre los bits necesarios para llevar a cabo el direccionamiento y la capacidad. En efecto, si la memoria tiene H bytes y se accede a nivel de byte, significa que se necesitan $2^m \geq H$.

Como las memorias dinámicas y de propagación almacenan, además de los datos, información de direccionamiento y reloj, hay que distinguir entre *capacidad neta o útil*, que es la capacidad de almacenamiento de información de usuario y la *capacidad bruta*, que se halla sumando a la neta, la de direccionamiento y reloj.

La capacidad bruta de una memoria dinámica viene dada por el *tamaño del soporte* y por la *densidad de grabación*. El tamaño se puede expresar como la longitud total de pista utilizada y la densidad de grabación viene dada por la longitud de pista necesaria para grabar un punto de memoria.

Algunas memorias dinámicas están hechas de forma tal, que el soporte es intercambiable.

4.7. TIPOS DE DISPOSITIVOS DE ALMACENAMIENTO PARA MEMORIA PRINCIPAL

MEMORIAS ELECTRÓNICAS	
ACCESO ALEATORIO	
RAM	ESTÁTICAS { BIPOLES } MOS DINÁMICAS: MOS
LECTURA Y ESCRITURA	VOLATILES
ROM	NO PROGRAMABLES { BIPOLES } MOS
SÓLO LECTURA	PROGRAMABLES { BIPOLES } MOS (UNA VEZ) REFRD: MOS (VARIAS VECES)
ACCESO SECUENCIAL	REGISTROS DE DESPLAZAMIENTO { BIPOLES } MOS DISPOSITIVOS DE ACOPLO DE CARRAS, CEO : MOS
LECTURA Y ESCRITURA	BURBUJAS MAGNÉTICAS (BOBINAS MAGNÉTICAS NO-VULCANIZADAS) SERIE (BUCLE MAYOR/PEQUEÑO) NO VOLATILES CINTAS MAGNETICAS, TAMBORES MAGNETICOS (BOBINAS MAGNÉTICAS ESTÁTICAS)
ACCESO ASOCIATIVO	{ BIPOLES } VOLATILES (CAM)

Fig. 4-11. Clasificación de las memorias de tipo electrónico.

Se pasa a realizar una revisión de los distintos dispositivos de almacenamiento más usados en la memoria principal:

- Memorias de ferritas.
- Memorias de semiconductores.
- Tambor de silicio.

En la tabla de la figura 4-11 se ofrece una posible clasificación, más detallada, de las memorias de tipo "electrónico" o de semiconductores.

4.8. MEMORIA DE FERRITA

Aunque hoy en día están en desuso, la práctica totalidad de las memorias principales, desde mediados de la década de los 50, hasta los años 70, se han construido con ferritas. Una muestra de la importancia de este dispositivo histórico es que el escudo de las Facultades y Escuelas de Informática se basa en un toro de ferrita.

El punto de memoria es un toro o anillo de ferrita, que presenta dos direcciones de magnetización. Las primeras ferritas tenían un diámetro exterior de 0,3 cm y las últimas, de 0,05 cm.

La conexión a los transductores se realiza mediante hilos de cobre barnizados, que pasan por el interior de las ferritas. Esta parte del proceso de fabricación consistía en "coser" las ferritas y se hacía a mano. Era fácil que, al coserlas, hubiera roturas de ferritas, lo que obligaba a difíciles reparaciones.

La conexión se hacía con 2, 3 ó 4 hilos. En la figura 4-12 se muestra la conexión con 3 hilos. Evidentemente, el cosido con menos hilos era más sencillo, pero implicaba los transductores. El direccionamiento era, generalmente, 2D/1/2.

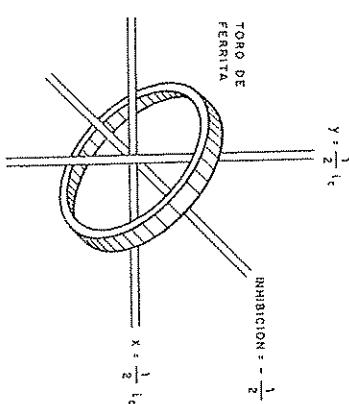


Fig. 4-12. Estructura fundamental de un toro de ferrita cosido con 3 hilos.

Las propiedades fundamentales de estas memorias son las siguientes:

- Memoria estática con direccionamiento cableado, tipo RAM.
- No volátil, pues, si se deja de alimentar, las polarizaciones de las ferritas se mantienen invariables.
- De lectura destructiva. La escritura exige un borrado previo, pues solamente se puede pasar del "0" al "1".
- Sólo se considera el tiempo de ciclo (lectura + escritura) pues los accesos siempre requieren un ciclo. La velocidad de los primeros prototipos era de 20 μ s y se ha llegado a modelos de 275 ns.
- La capacidad de estas memorias varía de unos pocos K a unos pocos Megas. Se construyen con módulos de 4 K.
- Valores típicos de anchos de palabra han sido 8, 16, 32 y 36 bits.

4.9. MEMORIAS DE PELÍCULA DELGADA Y DE HILO PLATEADO

Ambos tipos de memorias fueron un intento, de poco éxito comercial, de sustituir las ferritas de dos hilos por una estructura de fabricación más sencilla, de menor tamaño y, por tanto, de mayor velocidad.

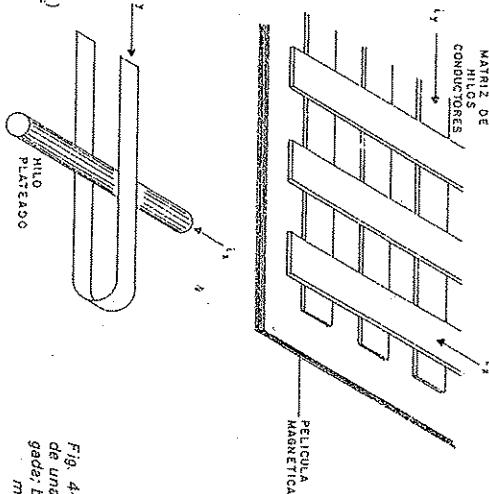


Fig. 4-13. a) Estructura general de una memoria de película delgada; b) estructura básica de una memoria de hilo plateado.

En los dispositivos de película delgada se parte de una fina capa magnetizable sobre la que se establece una matriz de hilos conectados a los transductores. La zona próxima al cruce de dos hilos realiza la misma función que un toro de ferrita de dos hilos. Dicha capa se deposita sobre un soporte y tiene un espesor de unos 1.000 Å = 10^{-4} mm. La figura 4-13 a) esquematiza este tipo de memoria.

En los dispositivos de hilo plateado el material magnético se deposita en una fina capa que recubre uno de los dos conductores. La zona de este depósito, próxima al cruce de ambos hilos, forma el equivalente a la ferrita. Véase la figura 4-13 b).

4.10. MEMORIAS DE SEMICONDUCTORES

Este tipo de memoria se emplea actualmente, con carácter universal, como memoria principal de los computadores.

Se hace referencia a los aspectos operativos de estas memorias. Todas las memorias que se van a tratar en este apartado son de direccionamiento cableado, o sea, de acceso aleatorio o RAM. Sin embargo, dentro de estas memorias se ha desarrollado otra terminología que resulta un poco confusa, pues repite términos empleados con otro sentido. Se puede establecer la siguiente clasificación:

a) De lectura y escritura (RAM)

- Estáticas.
- Dinámicas o con refresco.

b) De sólo lectura

- ROM (Read Only Memory).
- PROM (Programmable Read Only Memory).
- EPROM (Erasable Programmable Read Only Memory).
- EEPROM (Electrically Erasable Read Only Memory).

Las memorias con semiconductores se presentan en pastillas integradas que tienen una matriz de memoria, un decodificador de direcciones, los transductores correspondientes y el tratamiento lógico de algunas señales de control. Figura 4-14. Existen muchas configuraciones, pero la mayoría de estas memorias manejan los siguientes elementos y señales:

- Bus de datos: Es un colector o conjunto de líneas triestado que transportan la información almacenada en memoria. El bus de datos se puede conectar a las líneas correspondientes de varias pastillas.
- Bus de direcciones: Cuando está completo, es un conjunto de m líneas que transportan la dirección y que permite codificar 2^m posiciones de memoria.

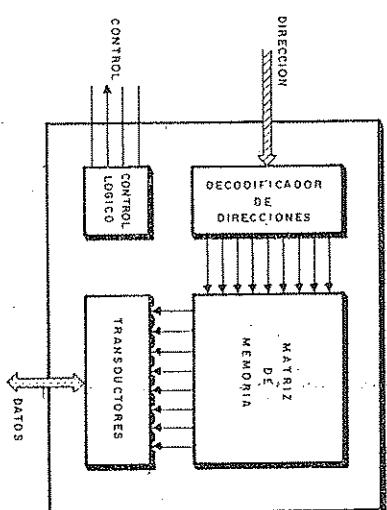


Fig. 4-14. Estructura interna y conexiónada de una memoria típica de semiconductores.

Puede estar multiplexado, de forma que primero se transmiten $m/2$ bits y luego, el resto.

- Señales de control típicas

\overline{OE} : Activa la salida triestado de la memoria.

\overline{CS} ó \overline{CE} : Activa la pastilla o chip.

\overline{WE} : Señal de escritura. Para realizar una escritura, además de activarse esta señal, también lo estarán \overline{CS} ó \overline{CE} .

\overline{RAS} ó \overline{CAS} : Las líneas \overline{RAS} (Row Address Strobe) y \overline{CAS} (Column Address Strobe) sirven para decodificar las filas y columnas de las RAM dinámicas.

- Ancho de palabra típico: 1, 4 u 8 bits.

4.10.1. RAM estáticas

Son las memorias de semiconductor basadas en galdas de tipo bivoltaje; por tanto, mantienen su estado siempre que no se interrumpe la alimentación.

Pastillas típicas de este tipo de memoria, son las siguientes:

*2114 ó *Capacidad*: 4 Kbits. Configuración 1 K X 4.

4044 N.º *pastillas*: 18. Dirección: 10 bits. Datos: 4 líneas bidireccionales. 2 líneas de control CS y WE . 2 de alimentación.

Tiempo de acceso: 200 ns.
Consumo: 550 mW. Alimentación: 5 V.

*2128 ó 4016 Capacidad: 16 K. Configuración: 2 K X 8.
N.º de pastillas: 24. Dirección: 11 bits. Datos: 8 bidireccionales.
Control: 3 (CS , OE y WE). Alimentación: 2.

Consumo: 380 mW. Alimentación: 5 V. Tiempo de acceso: 150 ns.

*51188

Capacidad: 64 Kbits. Configuración: 8 K X 8.
N.º de pastillas: 28. Dirección: 13 bits. Datos: 8 bidireccionales.
Tiempo de acceso: 120 ns.

Consumo: 200 mW. Alimentación: 5 V.

Señales de control: 4 (\overline{CS} , WE y \overline{OE}).

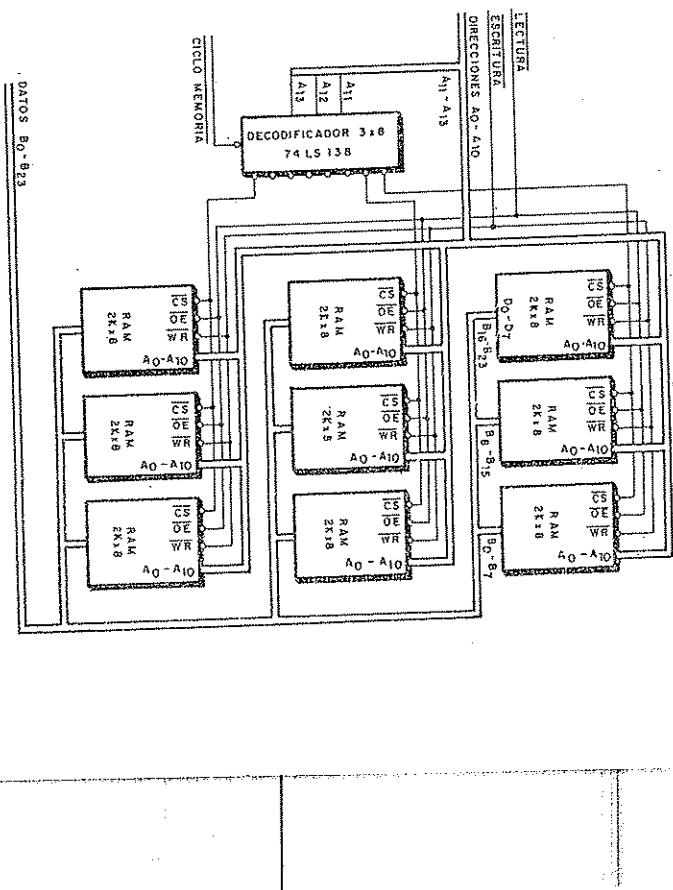


Fig. 4-15. Esquema de una configuración de 5 K palabras de 24 bits a base de memoria RAM estática.

4

Las propiedades más destacadas de las memorias RAM estáticas son:

- Permiten operaciones de lectura y escritura.
- Direcciónamiento aleatorio.
- Volatilidad: si se interrumpe la alimentación, se borran.
- Se usan en pequeñas memorias de menos de 32 K.
- Son más caras que las RAM dinámicas y su consumo es mayor, pero son muy sencillas de emplear y conectar.
- Las pastillas tienen bus de datos bidireccional de 4 u 8 bits de ancho, con salida triestado.

La figura 4-15 ofrece un ejemplo de aplicación de este tipo de memorias.

4.10.2. RAM dinámicas

Estas memorias usan celdas de un solo transistor de tipo MOS. El estado se almacena en un condensador que se forma entre los electrodos del transistor y que tiene a descargarse, por lo que hay que realizar un refresco periódico para evitar la pérdida de información.

Pastillas típicas de este tipo de memorias son las siguientes:

*3716 ó Capacidad: 16 K bits. Configuración: 16 K X 1.
4116 N.º Pastillas: 16. Dirección: 7. Datos: 2 (E y S). Control: 3 (\overline{RAS} , \overline{CAS} y WE). Alimentación: 4.

Tiempo de acceso: 150 ns. Ciclo: 375 ns.
Consumo: 500 mW. Alimentación +5 V y ± 12 V.
Período de refresco: 2 ms.

*2764 ó Capacidad: 64 Kbits. Configuración: 64 K X 1.
4164 N.º Pastillas: 16. Dirección: 8. Datos: 2 (E y S). Control: 3 (\overline{RAS} , \overline{CAS} y WE). Alimentación: 2.

Tiempo de acceso: 150 ns. Ciclo: 270 ns.
Consumo: 200 mW. Alimentación: 5 V.
Período de refresco: 4 ms.

*37256 ó Capacidad: 256 Kbits. Configuración: 256 K X 1.
4256 N.º de pastillas: 16. Dirección: 9. Datos: 2 (E y S). Control: 3 (\overline{RAS} , \overline{CAS} y WE). Alimentación: 2.

Tiempo de acceso: 120 ns. Ciclo: 240 ns.
Consumo: 200 mW. Alimentación: 5 V.
Período de refresco: 4 ms.

4

A continuación se citan las principales características de este tipo de memoria:

- Permiten operaciones de lectura y escritura.
- Direcciónamiento y acceso aleatorios.
- Volatilidad y refresco.
- Más económicas que las estáticas en tamaños medios o grandes (mayores de 64 K).
- Las pastillas tienen un ancho de palabra de 1 bit y el bus de direcciones está multiplexado en el tiempo, lo cual reduce el número de pastillas necesarias. Datos de salida en triestado.

La figura 4-16 presenta un ejemplo de direcciónamiento de estas memorias. Cada pastilla de memoria es de 64 K y se direcciona enviando primero la dirección de la fila por las líneas A0-A7, a la vez que se activa \overline{RAS} . Tras un retardo se introduce la dirección de la columna por las mismas líneas, a la vez que se activa la señal \overline{CAS} .

Las líneas A16 y A17 controlan un decodificador, cuya CS es la señal \overline{CAS} , produciendo las señales $\overline{CAS1}$, $\overline{CAS2}$ y $\overline{CAS3}$ que sirven para seleccionar uno de los 4 bloques de 64 X 8 bits. La señal M/R indica el inicio de un ciclo de memoria o de refresco. El circuito de refresco que actúa cada 2 ms está formado por un reloj de 30 ns y un contador para la dirección de refresco. La señal \overline{CAS} se deriva de la \overline{RAS} mediante el retardo adecuado.

En el mercado hay pastillas, como la DA 8409, que realizan todas estas funciones y simplifican el diseño con memorias dinámicas.

4.10.3 Memorias ROM y PROM

Estas memorias sólo permiten la operación de lectura. La información contenida se graba en el proceso de fabricación (ROM) o bien en un proceso posterior irreversible (PROM).

Pastillas típicas de estos tipos son:

*2732 ó Capacidad: 32 Kbits. Configuración: 4 K X 8.
2932 N.º de pastillas: 24. Dirección: 12. Datos de salida: 8.
Control: 2 ($\overline{CS1}$ y $\overline{CS2}$). Alimentación: 2.

Tiempo de acceso: 350 ns.
Consumo: 400 mW. Alimentación: 5 V.

*2764 ó Capacidad: 64 Kbits. Configuración: 8 K X 8.
2932 N.º de pastillas: 24. Dirección: 14. Datos de salida: 8.
Control: 1 (\overline{CS}). Alimentación: 2.

Tiempo de acceso: 300 ns.
Consumo: 400 mW. Alimentación: 5 V.

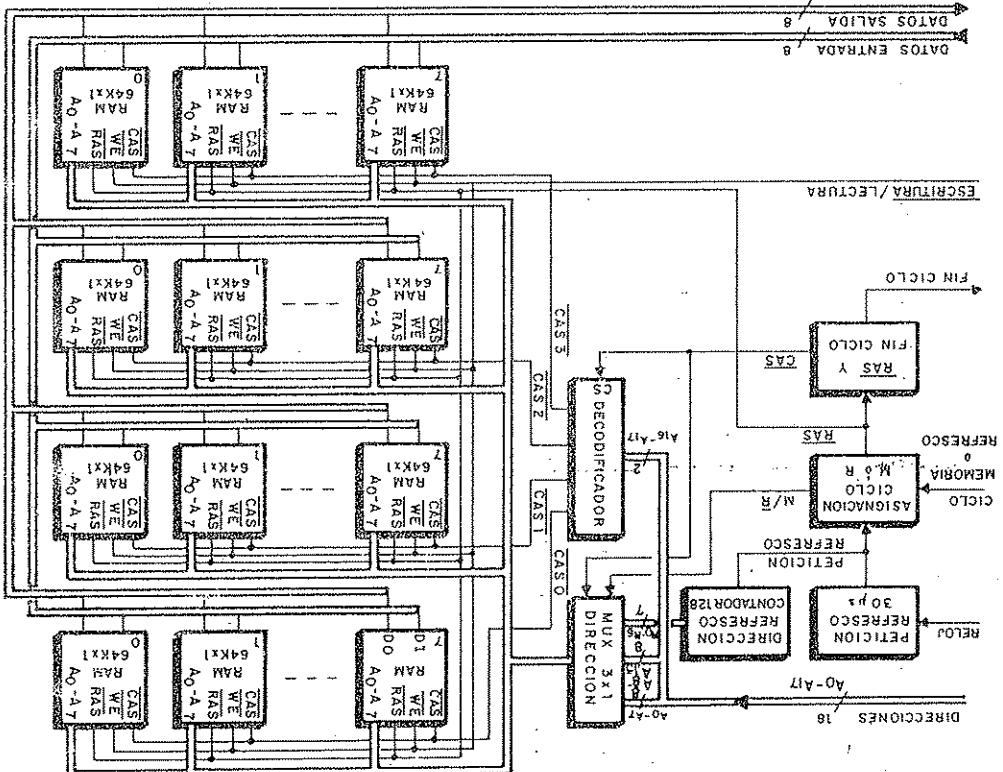


Fig. 4-16. Configuración de una memoria dinámica de 256 Kbytes.

4

4

*27128 ó Capacidad: 128 Kbits. Configuración: 16 K X 8.

Nº de pastillas: 28. Direcciones: 14. Datos de salida: 8.

Control: 2 \overline{CE} y \overline{OE} . Alimentación: 2.

Tiempo de acceso: 400 ns.

Consumo: 400 mW. Alimentación: 5 V.

*27256 ó Capacidad: 256 Kbits. Configuración: 32 K X 8.

Nº de pastillas: 28. Direcciones: 15. Datos de salida: 8.

Control: 1 (CS). Alimentación: 2.

Tiempo de acceso: 250 ns.

Consumo: 200 mW. Alimentación: 5 V.

Una aplicación muy interesante de este tipo de memorias es la destinada a guardar los programas de arranque de los computadores.

- Entre sus principales características, destacan:
 - Sólo permiten la lectura.
 - Son de acceso aleatorio.
 - Son permanentes o no volátiles: la información no puede borrarse.
 - Tienen un ancho de palabra de 8 bits, con salida triestado.

La figura 4-17 muestra un ejemplo de aplicación de este tipo de memoria, en combinación con otra configuración de tipo RAM.

4.10.4. EPROM Y EEPROM

Las memorias EPROM y EEPROM son memorias permanentes, de sólo lectura, pero que se pueden borrar mediante un proceso especial, como es la radiación ultravioleta en el caso de las EEPROM, o elevadas corrientes en el caso de las EEPROM. Son más caras que las ROM y EPROM, pero son eficaces cuando hay probabilidad de tener que hacer cambios en el contenido de estas memorias.

Son típicas las siguientes pastillas EEPROM:

*XX16 - Capacidad: 16 Kbits. Configuración: 2 K X 8.

Nº de pastillas: 24. Direcciones: 11. Datos: 8. Control: 2 (\overline{CE} y \overline{OE}).

Alimentación: 3, una para grabación.

Tiempo de acceso: 400 ns.

Consumo: 300 mW. Alimentación: 5 V (grabación a 25 V).

*XX32

Capacidad: 32 Kbits. Configuración: 4 K X 8.

Nº de pastillas: 24. Direcciones: 12. Datos: 8. Control: 1.

Alimentación: 3, una para grabación.

Tiempo de acceso: 500 ns.

Consumo: 500 mW. Alimentación: 5 V (grabación a 25 V).

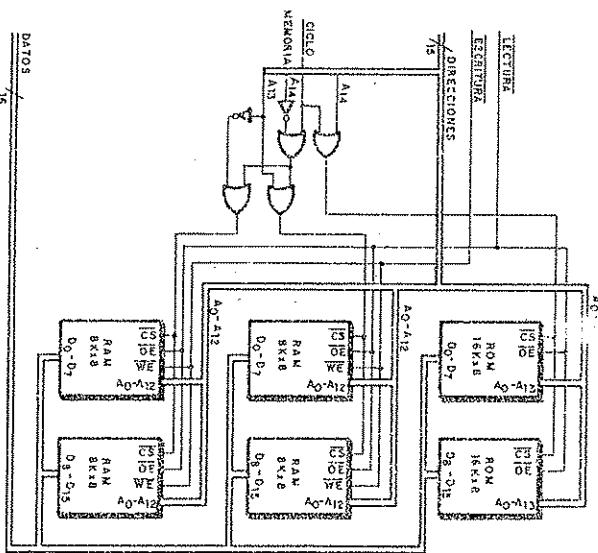


FIG. 4-17. Memoria formada por 16 K X 16 de ROM y 16 K X 16 de RAM

*
X
64

Nº de patillas: 28. *Direcciones:* 13. *Datos:* 8. *Control:* 3.
Alimentación: 3, una para grabar.
Tiempo de acceso: 450 ns.
Consumo: 400 mW. *Alimentación:* 5 V (25 V para grabar).
Capacidad: 128 Kbits. *Configuración:* 16 K X 8.

Consumo: 400 mW. **Alimentación:** 5 V (25 V grabación). **Características principales de audio:** 16 bits, 44.1 kHz.

- Tienen un ancho de palabra de 8 bits, con salida triestado.
- Son de tipo no volátil, aunque pueden borrarse.
- Son de acceso aleatorio.
- Solo permiten la lectura.

4.11. LINEAS DE RETARDO

Las líneas de retraso son memorias de propagación, poco empleadas en la actualidad, cuya estructura básica de funcionamiento se ofrece en la figura 4-18.

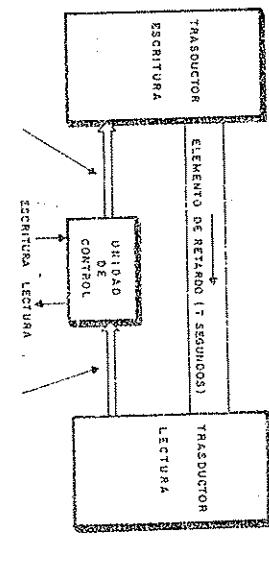


Fig. 4-18. Estructura de una línea de retraso.

Si se supone que la línea de retraso transmite una señal empleando 1 segundos en esta transmisión y se dan los valores V_4 y V_5 a la señal introducida en la línea, se genera un tren de impulsos que, mediante un código adecuado, permiten representar un tren de "ceros" y "unos". Los códigos a emplear serán del tipo de los que se utilizan en la grabación magnética y que se comentan posteriormente.

Suponiendo que la representación de cada bit emplea t segundos y, puesto que la línea contiene T segundos de señal, quiere decir que la línea en cada instante contiene $n = T/t$ bits. Ahora bien, si la señal de salida de la línea se reintroduce a la entrada, siempre habrá los mismos n bits desplazándose por la línea; ya se dispone de una memoria.

La operación de lectura necesita esperar a que la zona de datos a leer aparezca en la salida de la línea de retraso. El tiempo medio de espera o *tiempo de acceso*, será $T/2$. Una vez alcanzada esta zona, se van obteniendo los datos a una velocidad de $1/t$ bits/s. La operación de lectura es inherente a la línea de retraso, pues se deben leer y reintroducir continuamente los datos para conservarlos.

La escritura se efectúa cambiando la secuencia que se lee de la línea, por una nueva que represente el valor deseado. El tiempo medio de acceso será también de $T/2$ y la velocidad de escritura de $1/t$ bits/s.

En este tipo de memoria la lectura y la escritura se hacen bit a bit. Si se quiere hacer a nivel de byte, la unidad de control se encargará de la conversión serie-paralelo correspondiente.

Para diseñar las líneas de retardo, se han utilizado distintas propiedades físicas, destacando las siguientes:

- **Acústica:** Se empleó una columna de mercurio que se excitaba y tenía media- te transductores de cuarzo. Estos transductores generan una onda acústica que se propaga a 1,5 km/s en el mercurio. Si la columna tiene 1,5 m, el retardo total será de 1 ms y con un reloj de 1 MHz se pueden almacenar 1.000 bits. También se han empleado líneas de cuarzo de reflexión múltiple con unos retardos totales de 1 ms.
- **De compresión:** Las ondas de tensión se transmiten en aleaciones de hierro-níquel, permitiendo almacenar 1 Kbit.

Las principales características de estas memorias:

- Ser volátiles.
- Son del tipo de propagación.
- Tienen poca capacidad de almacenamiento.
- La lectura es destructiva.
- No se utilizan en la actualidad.

4.12. TAMBOR DE SILICIO

La idea del tambor de silicio fue apoyada al comienzo de la década de los 70, sin mucho éxito. Consistió en constituir una arquitectura similar a la de un disco de cebra fija o tambor, pero a base de registros de desplazamiento. Consiguiendo la correspondencia entre pistas y registros de desplazamiento, se puede fabricar un "tambor" que no tenga partes móviles.

4.13. PRINCIPALES RECURSOS PARA MEJORAR LAS PRESTACIONES DE LA MEMORIA PRINCIPAL

La memoria es uno de los componentes básicos y esenciales de todo computador, puesto que toda la información que manipula, en forma de datos o instrucciones, se almacena en la memoria.

Los progresos en el campo de las memorias también han caminado paralelamente al desarrollo de la Electrónica, que ha sido el causante de la sustitución de los antiguos dispositivos de núcleos de ferrita, que configuraban la memoria principal, por los modernos, constituidos con tecnología VLSI.

La memoria principal ha de ser capaz de cubrir dos requisitos indispensables:

1. Suficiente capacidad para contener los programas y datos en fase de ejecución.
2. Un tiempo de acceso a sus posiciones de memoria en consonancia con el tiempo de procesamiento por parte de la Unidad de Control. La realización de una instrucción se compone de una *fase de búsqueda*, en la que hay que acceder a la memoria principal para encontrar el código máquina, y de una *fase de ejecución*, en la que la Unidad de Control interpreta el código y regula la ejecución de dicha instrucción en la ALU y en los restantes elementos del sistema. Un funcionamiento equilibrado del computador exige que la fase de búsqueda, dependiente de las características de la memoria principal, dure un tiempo similar al de la fase de ejecución en la UCP.

Para intentar cubrir estas exigencias se emplean dos mecanismos, que reciben los nombres de *memoria virtual* y *memoria cache*.

La memoria virtual tiene como objetivo aumentar la capacidad real de la memoria principal, mientras que la memoria cache incrementa la velocidad de los accesos.

4.14. MEMORIA VIRTUAL

Un computador usa memoria virtual cuando las direcciones que generan sus programas hacen referencia a un espacio mayor que el espacio físico realmente disponible en la memoria principal. El programador tiene la impresión de trabajar con un tamaño físico de la memoria principal mucho mayor que el que tiene. Hay que diferenciar entre el *mapa de direcciones lógicas o virtuales* y el *mapa de direcciones físicas o reales*. El espacio lógico utiliza como soporte un dispositivo de almacenamiento externo, como puede ser un disco, mientras que el espacio físico es el que ocupa la memoria principal existente.

La ejecución de programas en un computador requiere que sus datos e instrucciones residan en la memoria principal. Por lo tanto, las máquinas con memoria virtual, que actualmente son mayoría, disponen de un módulo, transparente para el usuario, que asigna, inteligentemente, partes de la memoria principal disponible entre los programas que compiten para conseguir su ejecución de forma compartida. Figura 4-19.

El programador que dispone de memoria virtual ya no tiene que preocuparse con los tediosos problemas que surgían en la técnica de solapamiento u overlap, en la que tenía que dividir el programa, que no cabía en la memoria principal, en una serie de trozos de tamaño parecido a los que se denominaba *módulos*. Luego, el propio programa contenía las instrucciones de E/S necesarias para tratar estos módulos en una memoria externa en el momento adecuado e introducirlos, ordenadamente, en una zona de solapamiento compartida en la memoria principal, tal como se refle-

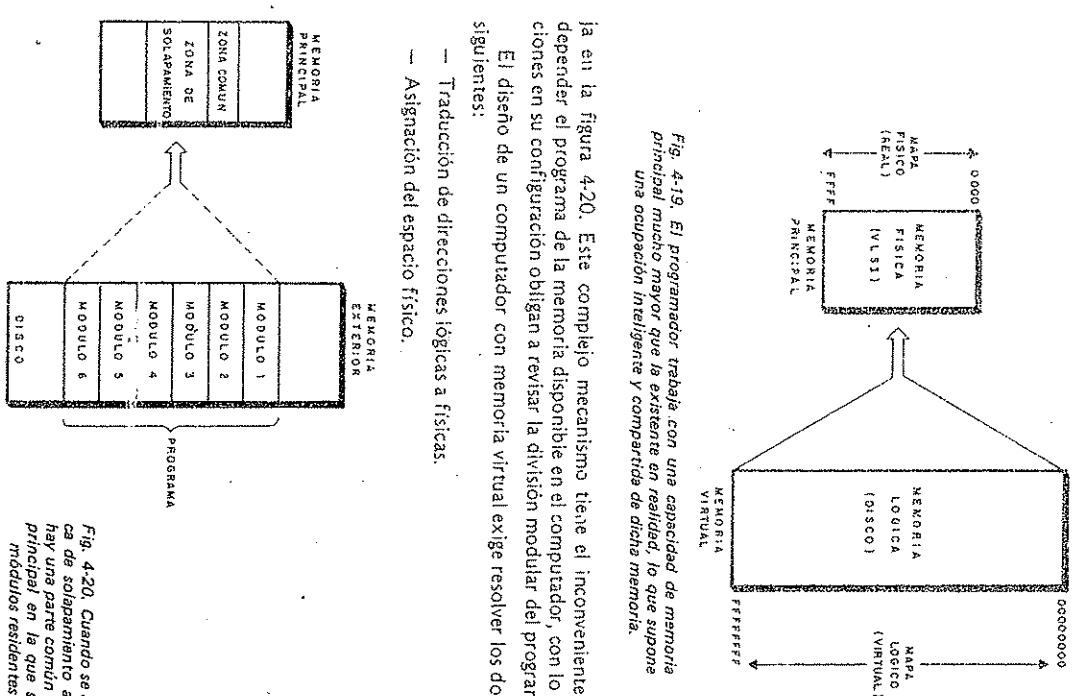


Fig. 4-19. El programador trabaja con una capacidad de memoria principal mucho mayor que la existente en realidad, lo que supone una ocupación inteligente y compartida de dicha memoria.

ja en la figura 4-20. Este complejo mecanismo tiene el inconveniente de hacer depender el programa de la memoria disponible en el computador, con lo que variaciones en su configuración obligan a revisar la división modular del programa.

El diseño de un computador con memoria virtual exige resolver los dos aspectos siguientes:

- Traducción de direcciones lógicas a físicas.
- Asignación del espacio físico.

Fig. 4-20. Cuando se aplica la técnica de solapamiento a un programa hay una parte común en la memoria principal en la que se guardan los módulos residentes en el disco.

La traducción es precisa porque programas y datos deben situarse en la memoria principal, controlada por direcciones físicas. Sin embargo, las direcciones que generan los programas para acceder a los operandos o para efectuar bifurcaciones, así igual que el contenido del Contador de Programa, hacen referencia al espacio lógico. El mecanismo de traducción se encarga de efectuar la operación básica que se muestra en la figura 4-21.

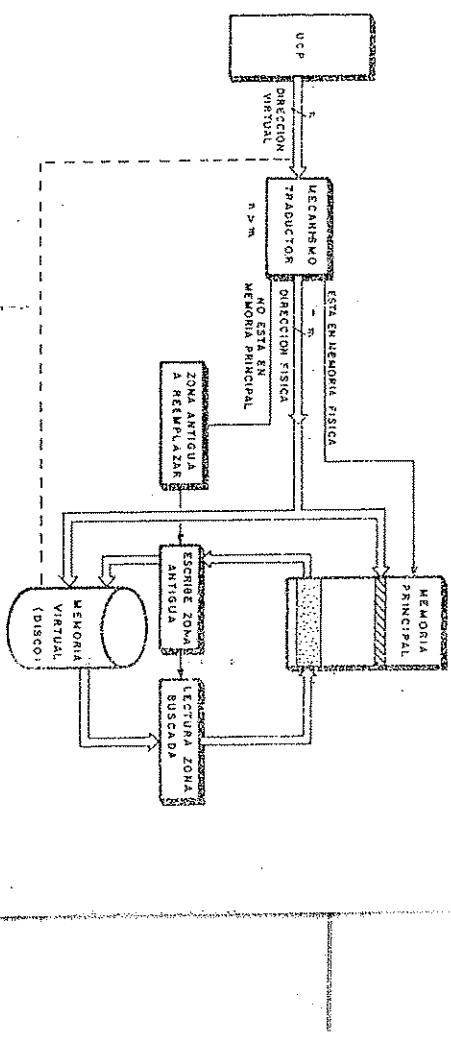


Fig. 4-21. Mecanismo básico de traducción de las direcciones ligeras a físicas.

Con respecto a la asignación de memoria física, como el espacio físico es mucho menor que el lógico, no podrán residir en la memoria principal más que una pequeña parte de los procesos que hay en ejecución en el computador en un momento dado. Esto supone que, en muchas ocasiones, se intente acceder a posiciones no residentes en memoria física. Por lo tanto, debe existir un procedimiento que, de forma dinámica, asigne el espacio físico, de manera que todos los procesos tengan en la memoria principal aquellas partes que se requieran para su ejecución. Cada vez que se accede a una información que no esté en la memoria física, hay que proceder a una transferencia de información desde el espacio lógico al espacio físico para traer al hueco que se dejó en la memoria principal la nueva información necesaria para seguir ejecutando el proceso que originó este problema.

Los distintos modelos de memoria virtual se diferencian por sus políticas de solapamiento y por los métodos que emplean en la organización de la memoria. Los más importantes son:

- Memoria paginada.
 - Memoria segmentada.
 - Memoria de segmentos paginados.
- Todos estos sistemas encuentran como problema crítico que los requerimientos de la memoria de algunos programas específicos son difíciles de predecir. Y por ello, la fracción de memoria que debe asignarse a un programa es variable en cada caso. Además, la política de solapamiento y compartición debe tener en cuenta ciertas características internas de los programas que, inevitablemente, determinan la construcción modular y estructurada de los mismos. Dichas características son:

1. *Localización temporal*. Es la tendencia de un proceso a referirse, en un futuro próximo, a elementos utilizados recientemente. Las variables y los stacks del proceso son ejemplos de elementos que ejercitan esta característica.
2. *Localización espacial*. Es la tendencia que tienen los procesos a referirse a elementos próximos al espacio virtual antes recorrido.
3. *Localización secuencial*. Tendencia de los procesos a referenciar elementos de la secuencia inmediata.

Para decidir qué fracción de memoria principal ha de ser destruida o cargada en disco si ha sido modificada cuando se necesita leer otra, las reglas o criterios más empleados son:

- a) *Regla FIFO* (first-in, first-out). Se destruye la fracción que más tiempo lleva en la memoria principal para dejar un hueco en ésta.
- b) *Regla LRU* (least recently used). La porción que lleva más tiempo sin haber sido usada o actualizada.
- c) *Regla LIFO* (last-in, first-out). El hueco aparece en la memoria principal destruyendo o devolviendo a disco (si se ha modificado) la parte que lleva en memoria el menor tiempo.
- d) *Regla LFO* (least frequently used). Deja hueco la porción que se ha pedido menos veces desde que comenzó el proceso.
- e) *Regla RAND* (random). Se elige una porción al azar.
- f) *Regla CLOCK*. Cuando se coloca un bit de uso en cada entrada de una cola FIFO y se establece un puntero que la convierte en circular. Es una aproximación al algoritmo LRU con una cola FIFO simple.

4.15. MEMORIA PAGINADA

Este modelo organiza el espacio virtual y el físico en bloques de tamaño fijo, llamados *páginas*. En un momento determinado la memoria principal contendrá algunas de los bloques lógicos. Como las distintas posiciones dán un bloque lógico y un físico están ordenadas de forma idéntica, simplemente hay que traducir el número del bloque lógico al correspondiente del bloque físico.

Los métodos de traducción son diversos, desde el más básico de *correspondencia directa* al más complejo de *correspondencia asociativa*, donde la búsqueda se realiza mediante el contenido de una memoria asociativa que mantiene las correspondencias virtual-física más recientemente utilizadas. En la práctica se utiliza una técnica mixta en la que las páginas más recientemente empleadas se encuentran en un memoria asociativa y todas ellas en una tabla de correspondencia directa.

4.15.1. Método de correspondencia directa

La dirección virtual consta de dos campos: *un número de página virtual (npv)*, *un desplazamiento (d)*, dentro de la página indicada. Con el número de la página virtual se accede a una entrada de un *Tabla de Páginas (TP)* que proporciona, además de la dirección física de la página, una serie de información complementaria. Localizada la página física, el desplazamiento (d) sirve para completar la posición concreta dentro de ella.

En el momento de arranque, cada proceso activo del sistema crea en la memoria principal una Tabla de Páginas (TP) que contiene una entrada por cada posible página virtual. La configuración de las entradas de TP se muestra en la figura 4-22 y consta de los siguientes campos:

1. *Bit de validación (V)*, que, cuando está activado, indica que la página existe. Si $V = 0$, la página no existe y se creará cuando haga falta.
2. *Bit de modificación (M)*, que indica si la página ha sido modificada en memoria. Este bit se utiliza en los algoritmos de reemplazo y actualización de memoria.
3. *Código de acceso autorizado a la página (CAA)*, que puede ser de lectura, escritura y/o ejecución; son 2 bits.
4. *Bit de memoria/disco (D)*, indica si la página reside en la memoria principal o no. En caso negativo, se activa un mecanismo de "falta de página", que provoca el traspase de dicha página desde el disco a la memoria.
5. *Dirección de la página (DP)*, que contiene la dirección de la página en memoria principal o la dirección de la misma en memoria virtual o disco, según que la página esté activa o inactiva de acuerdo con el señalizador D.

4

La dirección donde comienza la Tabla de Páginas (TP) está almacenada en el Registro Base de la Tabla de Páginas (RBTP). Para acceder sucesivamente a las entradas de la TP se incrementa el valor correspondiente al número de página virtual (npv) al que se guarda en RBTP.

Para calcular la dirección física en memoria se concatena DP (número de página) con el desplazamiento d. Esto, si la página se encuentra en la memoria principal.

El problema de este método es que el número de entradas a la Tabla de Páginas (TP) ha de coincidir con el número de páginas virtuales, que es muy grande.

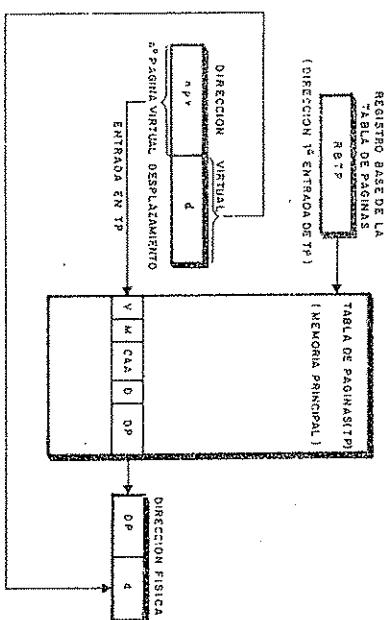


Fig. 4-22. Mecanismo de correspondencia directa para traducir la dirección virtual a física en una memoria de tipo paginado.

El empleo de una *tabla inversa*, es decir, ordenada por el número de la página física, reduce el número de entradas necesario, pero requiere una traducción con una función de búsqueda. En efecto, puesto que a esta tabla se entra por el número de la página física, hay que ir leyendo secuencialmente sus elementos hasta que se encuentre el número de página lógica de partida. Como esta búsqueda hay que realizarla en cada acceso a memoria principal, el computador que la soporta sería lento.

4.15.2. Método de correspondencia asociativa

En este caso se dispone de una tabla inversa en tecnología asociativa, esto es, con memoria tipo CAM, que se encarga ella misma de soportar el proceso de búsqueda a muy alta velocidad, suministrando el número de página física o una indicación de que la palabra lógica direccionada no se encuentra en memoria, en cuyo caso se elimina una página de la memoria principal (si no se ha modificado) y se trae la nueva al hueco que deja. Figura 4.23.

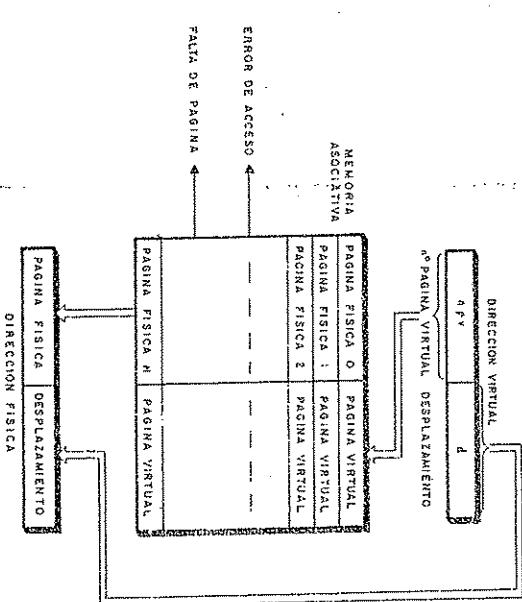


Fig. 4-23. Traducción de dirección virtual a física mediante memoria asociativa.

La memoria asociativa es aquella en la que se producen múltiples accesos de forma simultánea. En un simple acceso se pueden direccionar todas las posiciones que satisfacen un criterio de selección. Dado el elevado coste de las memorias asociativas, la tabla en CAM suele ser incompleta, albergando el conjunto de páginas "activas" en un momento determinado. Si la CAM origina falta, hay que acudir a la TP para comprobar si está en la memoria principal y, en su caso, actualizar la CAM. Si da falta la TP, hay que proceder a un cambio de página entre memoria principal y CAM.

4.16. MEMORIA SEGMENTADA

Este modelo explota el concepto de modularidad de los programas construidos estructuralmente. Los módulos son conjuntos de informaciones que pueden tratarse

4

ción de que la palabra lógica direccionada no se encuentra en memoria, en cuyo caso se elimina una página de la memoria principal (si no se ha modificado) y se trae la nueva al hueco que deja. Figura 4.23.

(compilarse, ejecutarse, etc.) independientemente y que se relacionan mediante llamadas interprocedimientos, constituyendo programas que se denominan *segmentos*.

La segmentación es una técnica que organiza el espacio virtual en bloques de tamaño variable, que reciben el nombre de *segmentos*, que se colocan en memoria mediante algoritmos de localización de espacio libre.

Los elementos de un segmento se identifican mediante la dirección del segmento al que pertenecen y un desplazamiento dentro del mismo. A semejanza con el modelo anterior, existe un *Registro Base de la Tabla de Segmentos* (*RBTS*), que dirige el comienzo de la *Tabla de Segmentos* (*TSG*), de las que existe una por cada proceso activo. Cada entrada de la *Tabla de Segmentos* se compone de los siguientes campos:

1. *Código de acceso autorizado* (*CAA*), que indica el modo de acceso permitido al segmento.
2. *Campo de longitud* (*L*), que indica la longitud del segmento.
3. *Bit de memoria/disco* (*D*), que indica si el segmento está o no en memoria.
4. *Campo de dirección de segmento* (*DS*), que contiene la dirección absoluta del segmento en memoria o la posición del segmento en disco, según el valor del señalizador *D*.

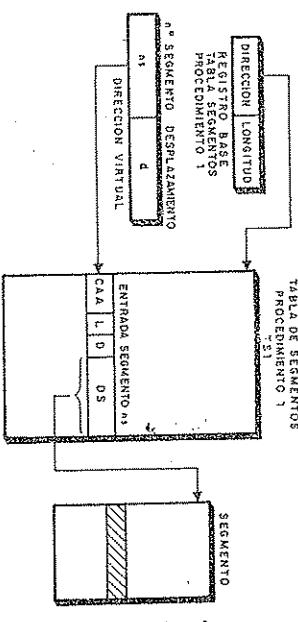


Fig. 4-24. Expresión gráfica simplificada del procedimiento de traducción de dirección virtual a física en una memoria segmentada.

Dada la naturaleza variable en cuanto a longitud de los segmentos, se precisa algún algoritmo que localice espacio libre para que resida el segmento apropiado, ya que no es corriente encontrar un bloque continuo en la memoria, para colocarlo por

completo. Estos algoritmos forman parte del mecanismo de interrupción de *falta de página* y los más relevantes son:

- a) *De mejor ajuste*: Minimiza el desperdicio, seleccionando el mejor agujero o fragmento inútil en el que se puede colocar el segmento.
- b) *De peor ajuste*: Localiza el agujero que maximiza el desperdicio al colocar el segmento.
- c) *De primer agujero*: Localiza el agujero con una dirección inicial inferior en el que se puede colocar el segmento.

- d) *Algoritmo buddy*: Utiliza técnicas de compactación de memoria, fusionando espacios inútiles, de forma que se configuran bloques continuos del tamaño adecuado.

Evidentemente, los segmentos pueden ser compartidos por muchos procesos. Algunos sistemas utilizan tablas auxiliares, que apoyan la búsqueda de segmentos compartidos, como la *Tabla de Segmentos Activos* (*TSA*), que indica cuáles son los segmentos activos en memoria en cada instante y la *Tabla de Segmentos Conocidos* (*TSC*), que contiene en cada entrada un nombre-segmento/número-segmento por cada segmento ya utilizado en el proceso.

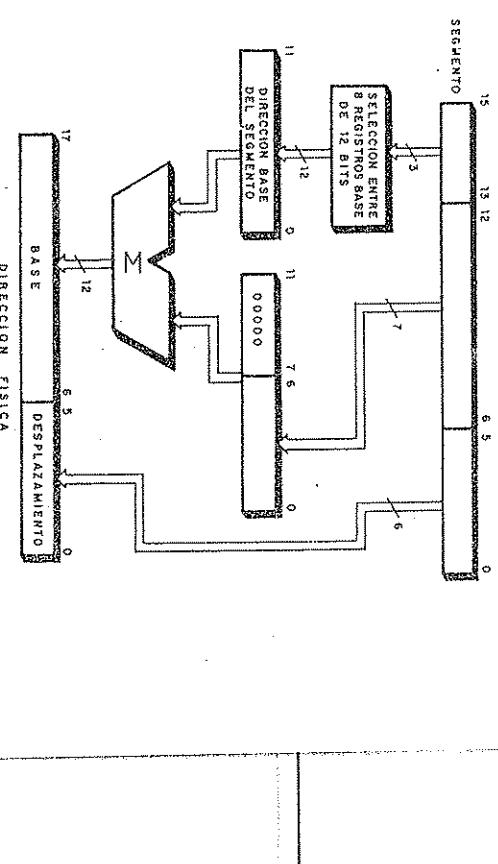


Fig. 4-24 bis. Traducción de la dirección virtual a física en los microcomputadores PDP-11.

4

Uno de los procedimientos más aceptados para la gestión de la memoria virtual es el que utilizan los minicomputadores PDP-11 de Digital Equipment Corporation. Como se muestra en la figura 4.24 bis, la dirección virtual de 16 bits, se divide en un campo de 3 bits, que selecciona uno de los 8 registros base de 12 bits existentes, y otro campo de 13 bits de desplazamiento. La dirección física de 18 bits se calcula sumando al registro base, los 7 bits de más peso del desplazamiento precedidos de cinco ceros, y concatenando al resultado los 6 bits de menor peso del desplazamiento. Se logra variar la longitud de los segmentos entre 64 bytes y 8 K bytes.

4.17. MEMORIA CON SEGMENTOS PAGINADOS

Esta memoria combina las ventajas de los dos modelos anteriores. Cada segmento se divide en páginas, de forma que, para acceder a cualquier elemento de un segmento, el sistema acude a la Tabla de Páginas (TP) de dicho segmento. Figura 4.25. Este procedimiento lo usa el microprocesador 80386 de Intel.

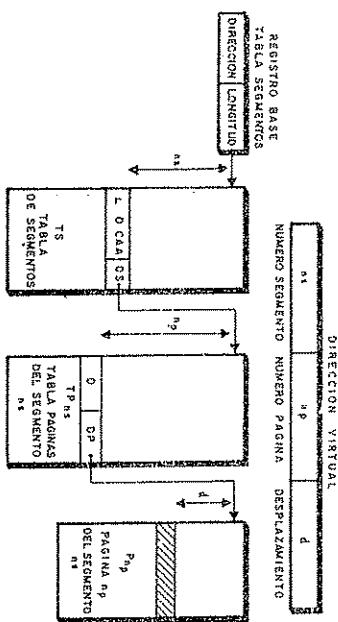


Fig. 4.25. Mecanismo de traducción de dirección virtual a real por correspondencia directa con memoria de segmentos paginados.

Si se aplica la técnica asociativa, que se muestra en la figura 4.26, para realizar la traducción, el tratamiento de las interrupciones de "fallo en el acceso" debe contemplar los siguientes aspectos:

1. Ausencia en el número de página en la memoria asociativa, en cuyo caso se obtendrá la dirección de página de la TP correspondiente.
2. Ausencia del número de segmento en la memoria asociativa. Supone una búsqueda en la Tabla de Segmentos activos (TSA) y, en el peor de los casos, en

Fig. 4.26. Funcionamiento de la traducción cuando se utiliza la técnica asociativa.

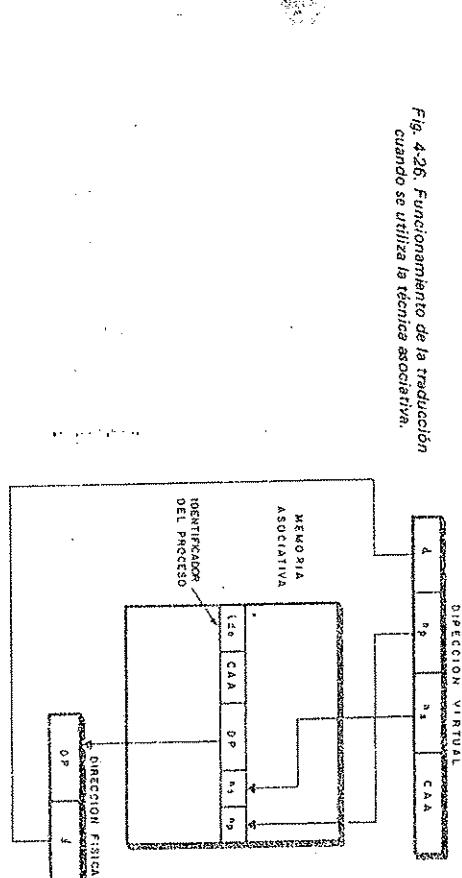


Fig. 4.26 bis. Gestión de la memoria virtual, empleada por National Semiconductor.

el directorio de ficheros del disco, para recuperar el segmento y/o los atributos.

3. En sistemas multiprogramados, la activación de un nuevo proceso que genera su propio espacio de dirección invalida las entradas anteriores de la memoria asociativa. Figura 4-26.

En la figura 4-26 bis se muestra gráficamente el procedimiento que emplea National, en un microprocesador de 32 bits, para gestionar la memoria virtual. La dirección virtual de 24 bits dispone de 3 campos. El de más peso de 8 bits relaciona el número de segmento TS, en una tabla de segmentos TS de 256 entradas de 32 bits. La entrada de TS selecciona una de las 256 tablas de página TP (128 entradas de 32 bits), en la cual se relaciona una entrada mediante el 2.º campo de 7 bits de la dirección virtual. Finalmente la entrada de la TP apunta a una de las 32.000 páginas, de 512 bytes cada una, en que se divide la memoria principal. La posición de la página seleccionada la determina el 3.º campo de 9 bits de la dirección virtual.

Finalmente, en la figura 4-27 se muestra el esquema general de un procesador dotado con memoria virtual. En él, en lugar de emplearse una memoria asociativa grande, cara y lenta, se divide la tabla de páginas físicas, con su equivalente lógica,

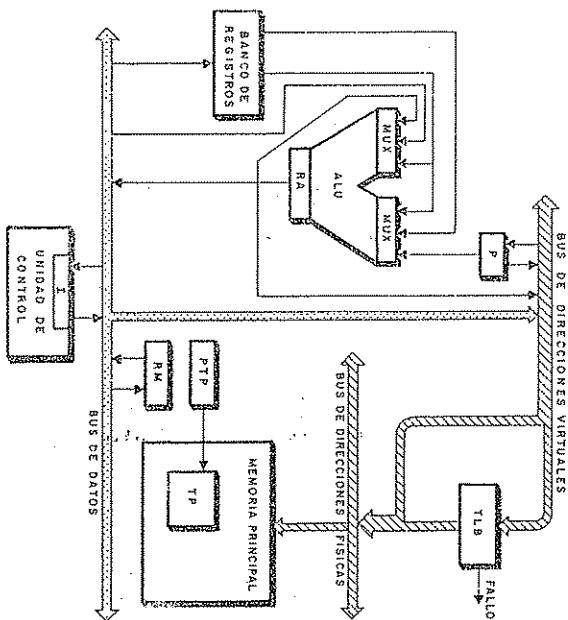


Fig. 4-27. Esquema general de un comparador con memoria virtual.

en dos partes. La primera reside en una pequeña memoria asociativa de alta velocidad que alberga aquellos elementos de la tabla que han sido empleados más recientemente. Esta memoria recibe el nombre de TLB (Translation Lookaside Buffer). La segunda consiste en una tabla directa, llamada TP, que reside en la memoria principal y que contiene todas las páginas lógicas asignadas a un proceso. El funcionamiento es el siguiente:

- Las direcciones generadas por la UCP son enviadas a la TLB, que produce la correspondiente página física o una señal de fallo.
- Si la dirección está en la TLB, se accede a la palabra de memoria principal, con lo que finaliza el proceso.
- En caso de fallo de la TLB, se pasa a leer TP para comprobar si la página está o no en memoria principal. Como ya se ha dicho, TP reside en memoria principal y hay un registro (PTP) dedicado a apuntar el principio de dicha tabla.
- Si se produce acierto en TP, se actualiza la TLB, eliminando uno de sus elementos para incluir la traducción de la nueva página. Además, se accede a la palabra deseada en la memoria principal.
- Si la TP da fallo, o sea, la referencia deseada no está en una página de la memoria principal, hay que determinar, de acuerdo con una política de reemplazo, qué página hay que eliminar para poder traer de disco la que originó el fallo.
- Puesto que el proceso de reemplazo de páginas es una operación larga (hay que acceder al disco), se congela el proceso que ha causado el fallo y se lanza un nuevo proceso.

Generalmente, es el sistema operativo quien determina la página a reemplazar, congela el proceso en curso y da control a otro proceso. Sin embargo, la función de lectura en la tabla TP para comprobar si la página está en memoria principal, hay que realizarla por hardware, por razones de velocidad. La propia Unidad de Control, o bien una unidad especial, soporta dicha función.

4.18. MEMORIA CACHE O INMEDIATA

Es una memoria auxiliar, de pequeña capacidad y alta velocidad, que se añade a la memoria principal para acelerar su funcionamiento. Suele ser del orden de 5 a 10 veces más rápida que la memoria principal y su tamaño oscila normalmente entre 8 y 64 K.

En la memoria cache se guardan las posiciones de memoria principal que más frecuentemente se prevee se van a usar. De esta forma, al realizar un acceso a la memoria principal, primero se comprueba si esa posición se halla en la memoria cache, ganando el acceso mucha velocidad en caso afirmativo. El funcionamiento

de la memoria cache con respecto a la memoria principal es muy parecido al de la memoria principal con respecto a la virtual.

Mientras que el concepto de memoria virtual se orientaba hacia el aumento de capacidad, la memoria cache se diseñó para incrementar la velocidad de acceso.

Las memorias cache y principal se dividen en bloques o líneas de unos pocos bytes, que hacen la función de páginas. En este caso, la propia cache se encarga de realizar la traducción de los números de bloque, generando la señal de fallo cuando la referencia deseada no se encuentra en ella. La secuencia de funcionamiento es la siguiente:

1. La UCP genera una dirección. En el caso de tratarse de una máquina virtual, la dirección virtual debe traducirse previamente a física. Conocida la dirección física, ésta se envía a la cache.
2. La memoria cache realiza la traducción a dirección de cache y comprueba si tiene la referencia; en caso afirmativo, realiza el acceso y finaliza el proceso.
3. En caso de producirse fallo en la cache, se accede directamente a la palabra deseada de la memoria principal y se procede a seleccionar un bloque para ser sustituido en la cache.
4. Si el bloque a sustituir no ha sufrido modificación en la cache, el proceso se limita a leer de memoria principal el nuevo bloque y almacenarlo en la cache. Si el bloque fue modificado, previamente habrá que leerlo de la cache y almacenarlo en la memoria principal, antes de rellenarlo con la nueva información.

Puesto que el tiempo empleado en leer un bloque de unas pocas palabras de memoria principal es muy pequeño, todo el proceso de selección del bloque a reemplazar se debe efectuar por hardware.

El empleo apropiado de la cache puede dar una tasa de aciertos superior al 90%, por lo que la velocidad aparente de la memoria principal se aproxima mucho a la de la cache.

4.19. TIPOS DE MEMORIA CACHE

De acuerdo con el modo de "traducción" de las direcciones de memoria principal a direcciones de memoria cache, éstas se clasifican en los siguientes tipos:

1. De correspondencia directa.
2. De asociación completa.
3. De asociación de conjuntos.
4. De correspondencia vectorizada.

4.19.1. Memorias cache de correspondencia directa

Se establece una correspondencia entre el bloque K de la memoria principal y el bloque k , módulo n , de la cache, siendo n el número de bloques de la memoria cache.

Este tipo es simple y económico, por no requerir comparaciones asociativas en las búsquedas. De todas formas, en sistemas multiprocesador pueden registrarse graves contingencias en el caso de que varios bloques de memoria correspondan concurrentemente en un mismo bloque de la cache.

Como se refleja en la figura 4-28, que muestra un esquema de la operatividad de estos dispositivos, una dirección de memoria consta de 3 campos:

- Campo de bloque.
- Campo de etiqueta.

4.19.2. Memoria asociativa completa

En este modelo se establece una correspondencia entre el bloque k de la memoria y el bloque $/$ de la cache, en la que $/$ puede tomar cualquier valor.

No se produce contención de bloques y es muy flexible, pero su implementación es cara y muy compleja, ya que el modelo se basa completamente en la comparación asociativa de etiquetas.

4.19.3. Memoria cache de asociación de conjuntos

Se divide la memoria en c conjuntos de n bloques, de forma que al bloque k de memoria corresponde uno cualquiera de los bloques de la memoria del conjunto k , módulo c . La búsqueda se realiza asociativamente por el campo de etiqueta y directamente por el número del sector. De este modo se reduce el costo frente al modelo anterior, manteniendo gran parte de su flexibilidad y velocidad. Es la estructura más utilizada.

4.19.4. Memoria cache de correspondencia vectorizada

El modelo divide a la memoria principal y a la cache en s sectores compuestos de n bloques. La relación se establece de cualquier sector a cualquier sector, siendo marcados los bloques no referenciados del sector como no válidos. Esta estructura también reduce costos, minimizando el núcleo de etiquetas para la comparación asociativa.

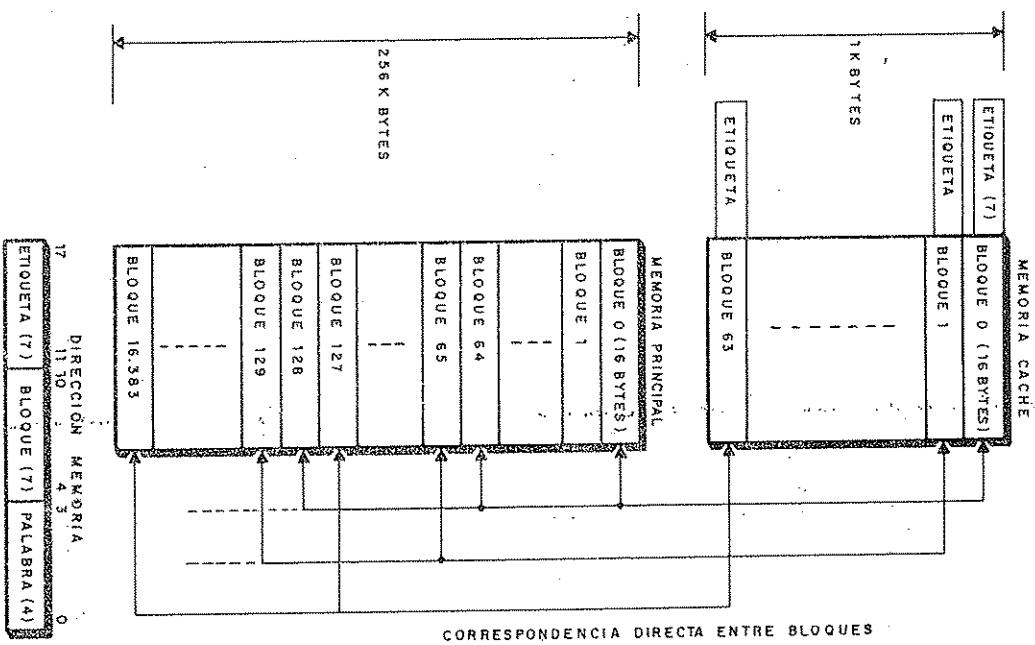


Fig. 4-28. Estructura de una memoria cache de 64 bloques de 16 bytes cada uno y de memoria principal de 16.384 bloques de 16 bytes cada uno (2.56 Kbytes) cuando utiliza un sistema de correspondencia directa.

4.20. PROTECCIÓN DE LA MEMORIA PRINCIPAL

Se trata de un mecanismo que evita que un proceso acceda a una zona de memoria que no le ha sido asignada. El tema de protección adquiere su mayor importancia en los ambientes concurrentes multiusuario, en los que previene la intrusión de los programas de usuario entre sí y de los programas usuario con el supervisor del sistema.

Los métodos que existen para proteger la memoria son variados, pero el más común se basa en dividir el espacio direccionable en *parcelas* que poseen una cláusula específica de acceso. Cuando las parcelas son compartidas por varios procesos deberá incluirse en la clave de acceso un identificador del mismo. Este método se utiliza mucho en sistemas con memoria virtual paginada, puesto que su filosofía es compatible con la división de la memoria que se requiere.

A continuación se describen los mecanismos más importantes usados en la protección de memoria.

4.20.1. Registros de borde

Este sistema se puede aplicar en el caso de asignar a los procesos zonas contiguas de memoria. En este supuesto, cada proceso tiene asignada una franja de memoria como se representa en la figura 4-29. La protección queda garantizada por dos registros de borde o *límite*, que contienen los valores superior e inferior de las direcciones de la franja del proceso activo, auxiliados por un comparador, que verifica cada referencia generada por el proceso para garantizar que se encuentra dentro de la zona válida. En el caso de fallo en la comparación, el comparador invalida el acceso a memoria y traslada el control al sistema operativo, que se encarga de cancelar el proceso causante del intento de violación, produciendo los correspondientes mensajes.

El sistema operativo carga los registros de borde con los valores del proceso antes de otorgarle el control. La carga se efectúa mediante instrucciones privilegiadas que los procesos normales no pueden ejecutar para evitar que puedan autopilarse su zona de memoria.

4.20.2. Cerrojo y clave

Los registros de borde no permiten establecer matizaciones o niveles en la protección, como pueden ser los siguientes:

- Acceso de lectura y escritura (por ejemplo, a programas y datos de otros procesos).
- Acceso de lectura (por ejemplo, a programas y datos de otros procesos).

- En los accesos a memoria, se compara el código del proceso con el cerrojo del bloque direccionado. Si coinciden, se atiende la petición de acceso. En caso contrario, se rechaza el acceso y se emplea esta señal de violación como excepción o trap, para bifurcar al sistema operativo.

4.20.3. Instrucción "TEST AND SET"

Hay un tipo de situación frecuente que exige una protección especial. Se trata de las zonas de datos accesibles, a nivel de lectura y escritura simultáneas, por varios procesos. Normalmente, es necesario que cuando un proceso está haciendo una actualización de esos datos, otros programas tengan prohibido el acceso a los mismos. Para ello se emplea un único bit de inhibición por zona de datos, bit que debe ser manejado mediante una instrucción especial "TEST AND SET", en el caso de estar el bit a cero, lo cambia a uno y el programa es autorizado a usar en exclusiva esa zona de datos. Si una interrupción u otra causa pasan el control a otro programa que desea acceder a la zona, al ejecutar la instrucción TEST AND SET sobre el bit correspondiente, y estar a uno, se le niega el acceso a esos datos. Cuando el programa finaliza su operación sobre los referidos datos, pone a cero el bit de inhibición, con lo que la zona queda disponible para otros procesos.

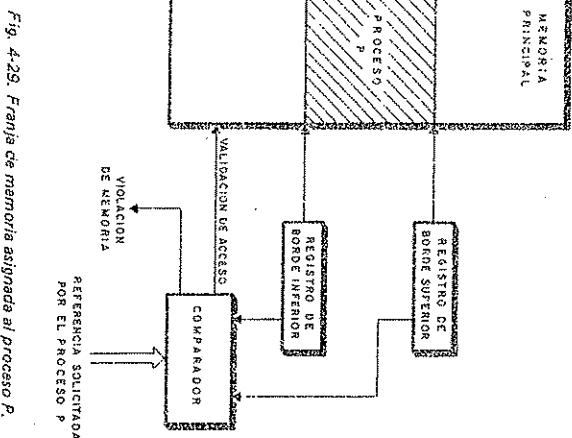


Fig. 4-29. Franja de memoria asignada al proceso P.

- Acceso de escritura (por ejemplo, de un usuario de un buffer de E/S).
- Acceso de ejecución (por ejemplo, de un programa de otro usuario).

Para diferenciar estas alternativas se establece un *cerrojo* y una *clave*. La memoria se divide en parcelas, de tamaño fijo o variable, a las que se asignan unos códigos de acceso para los diversos niveles de protección previstos. Para que un proceso pueda emplear, en un nivel de acceso determinado, una de esas parcelas debe conocer la clave correspondiente.

Esta protección se puede implantar en la memoria virtual Y en la física. En el primer caso, es el sistema operativo quien comprueba la correspondencia de las claves, antes de ordenar el acceso de un proceso a una página lógica. En el segundo caso, se implanta una circuitería específica para soportar la protección.

- El mecanismo hardware que emplean algunas máquinas para la protección funciona de la siguiente manera:
 - Se divide la memoria física en bloques o parcelas de varios Kbytes, a los que se asigna una serie de bits de cerrojo.
 - Cada proceso tiene asignado un código, que el sistema operativo almacena en un registro antes de entregarle el control.

4

EJERCICIOS

Ejercicio 4.1

¿De qué tamaño son las palabras en una memoria de 16 Mbytes, para la que se necesitan 20 bytes de direccionamiento?

Nota: Cada ciclo de lectura consta de 4 períodos de reloj.

Disponemos de una memoria cuyo tiempo de acceso es de 1 msg. Suponiendo que la CPU trabaja a 5 MHz, calcular la cantidad de ciclos de espera TW que va a haber cuando se ejecuta una instrucción de lectura.

Ejercicio 4.2

Si tenemos una memoria dinámica de 1 Mbit, ¿cuántos bits serán necesarios para el registro o contador de la dirección de refresco?

Ejercicio 4.3

Responda verdadero (V) o falso (F) a las siguientes afirmaciones, explicando su decisión.

- a) La memoria debe tener un tiempo de acceso muy pequeño.
- b) Existen computadores en los cuales la UCP es más rápida que la memoria.
- c) El computador, en lo referente a la memoria, recurre a su especialización.
- d) Una memoria veloz implica poca capacidad y bajo coste.
- e) La memoria virtual trabaja siempre con programación estructurada.
- f) El modo de direccionamiento de 3 direcciones es interesante por trabajar a nivel de bit.
- g) La memoria CACHE se utiliza para aumentar la capacidad de la memoria.
- h) La longitud de la dirección virtual ha de ser siempre igual a la de la dirección física.
- i) Los segmentos de un proceso tienen tamaño variable.
- j) La memoria CACHE solamente es más eficaz caso de que la dirección buscada resida en ella.

Ejercicio 4.7

Tenemos en memoria principal 1500 registros correspondientes a 1500 empleados de una empresa de alimentación; estos 1500 empleados serán los activos de un determinado momento, supuesta una plantilla de 5000 individuos. Los registros contienen la siguiente información:

- Código empleado 9 (5)
 - Nombre-Apellidos X (35)
 - DNI X (10)
 - Situación X
 - Categoría laboral XX
- Explique detalladamente en qué consisten los 3 procedimientos conocidos para protección de la memoria e indique cuál es, bajo su punto de vista, el más eficaz.

Ejercicio 4.4

Explique detalladamente en qué consisten los 3 procedimientos conocidos para protección de la memoria e indique cuál es, bajo su punto de vista, el más eficaz.

4

- Ejercicio 4.8
- Importe bruto 9 (5)
 - Sueldo Base 9 (5)
- Suponiendo páginas de 1 k y siendo 16 los bytes de cada entrada de la tabla de páginas, se pide:

a) Diseñar la memoria física y virtual.

b) Establecer la longitud de las direcciones tanto física como virtual.

c) Se desea incrementar el sueldo base del empleado 15584 en 10.000 ptas., se sabe que el n.º de página virtual es:

$$\text{npu} = \frac{\text{Clave} - 500}{2}$$

y que la tabla de páginas contiene las siguientes entradas:

7690	
1	230
7691	0
7692	0
1	134
7693	

Indicar paso a paso el proceso a realizar para hacer efectivo dicho aumento.

Ejercicio 4.8

Diseñar una memoria que contenga ROM y RAM en la proporción 1 : 3, sabiendo que se manejará en un sistema basado en el 8086 donde se emplearán 15 líneas del bus de direcciones.

4

Ejercicio 4.9

Usando pastillas modelo 2128 y 2764 establecer el mapa de memoria, decodificador de direcciones y diseño de una memoria con 14 K de RAM y 32 K de ROM, a la que se podrá acceder en palabras de 16 bits.

Ejercicio 4.10

Sobre un banco de memoria de 32 K X 8, especificar el circuito de refresco necesario para alimentar la memoria dinámica 4116 que lo compone. Conectar todas las señales. Si ha de responder a la dirección 0400 en adelante, y el bus de direcciones consta de 16 líneas, conectar los CS al decodificador.

Modos de direccionamiento y repertorio de instrucciones



5.1. LAS MAQUINAS DE PROGRAMA ALMACENADO

El trabajo de un computador consiste en el procesamiento de los datos de acuerdo con un programa de instrucciones almacenado en la memoria principal. Dicho procesamiento se realiza en la UCP, que comprende:

- La Unidad de Control
- La Unidad Operativa
- La memoria principal

La Unidad de Control se encarga de interpretar y controlar la ejecución de las instrucciones. La Unidad Operativa tiene la misión de llevar a cabo las operaciones lógicas o aritméticas que implican las instrucciones. La memoria principal almacena los datos y el programa en ejecución. Finalmente, la máquina dispone de una Unidad de E/S que transfiere información entre el mundo exterior (periféricos) y la UCP, en ambos sentidos. Figura 5.1.

La secuencia básica de ejecución de cualquier programa se compone de los siguientes pasos, que se repiten hasta alcanzar el final del programa.

1. La Unidad de Control, a través del *Contador de Programa*, envía la dirección de la memoria principal que contiene el código de la instrucción a ejecutar.
2. La Unidad de Control recibe el código de la instrucción y la decodifica.
3. El *Secuenciador* de la Unidad de Control genera una serie de señales de gobierno para todos los elementos del sistema, que se encargan de ir ejecutando la instrucción en pasos elementales, que reciben el nombre de *microinstrucciones*.

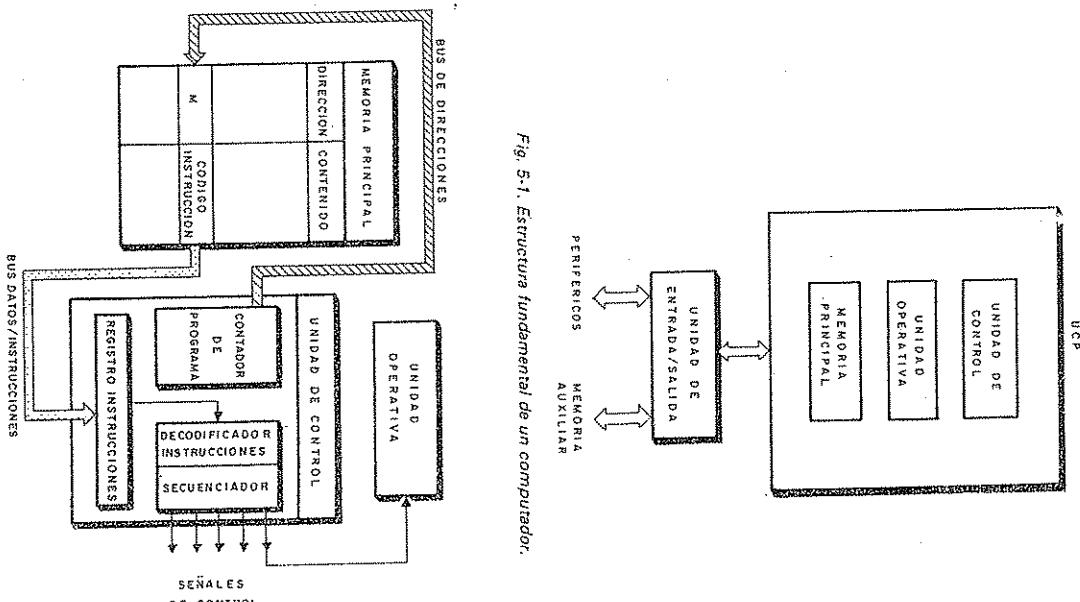


Fig. 5.1. Estructura fundamental de un computador.

Fig. 5.2. Búsqueda y decodificación de una instrucción.

Todos los elementos de la UCP tienen carácter binario y las informaciones correspondientes a los datos y las instrucciones se componen de "unos" y "ceros". En consecuencia, el único lenguaje que entiende la máquina es el binario. Pero ¿qué tipo de operaciones es capaz de interpretar y ejecutar directamente un computador? Sólo acepta un conjunto de instrucciones muy restringido y sencillo, que se denomina *lenguaje máquina* o de *bajo nivel*. Cada familia de computadores tiene su propio "lenguaje nativo" o máquina y el mismo constituye una de las características más importantes de su arquitectura. Los fabricantes diseñan la UCP y el Decodificador de instrucciones que interpreta las posibles instrucciones que es capaz de ejecutar, atendiendo a las aplicaciones a las que se destine.

Las instrucciones son almacenadas, transferidas y tratadas como cadenas de bits. Para reducir las expresiones y evitar equivocaciones en el sistema binario, frecuentemente se expresan las instrucciones en sistema hexadecimal, que es más compacto y claro.

5.2. CARACTERÍSTICAS DEL LENGUAJE MAQUINA

El lenguaje máquina, que es el que puede interpretar y ejecutar directamente el computador, está compuesto por una serie de instrucciones que se denominan *juego de instrucciones del computador*. Generalmente, todas las instrucciones máquina son muy simples.

Ejemplos

Algunas instrucciones típicas de UCP son:

1. Instrucción: A900; UCP: 6502 (microprocesador)

Operación: Carga en el registro A el dato inmediato que sigue al Código OP (00).

Expresión gráfica: $(A) \leftarrow 00$

2. Instrucción: 3A22AA; UCP: microprocesador Z-80

Operación: Carga en la posición ^a de la memoria principal AA22 el contenido del registro A.

Expresión gráfica: M(A,A22) $\leftarrow A$

3. Instrucción: 3C47; UCP: IBM 370

Operación: Multiplica los contenidos de los registros R4 y R7, dejando el resultado en R7.

Expresión gráfica: R4 $\leftarrow R4 \times R7$

Las 4 propiedades más importantes que cumplen las instrucciones máquina son las siguientes:

1. Realizan una sencilla y única función para simplificar su decodificación. Sin embargo, existen computadores que disponen de alguna instrucción compleja, como en el caso del VAX con su instrucción "FA" para llamarás a subrutinas.
2. emplean un número fijo de operandos, los cuales adoptan una representación determinada. Así, la instrucción 3C47 del IBM 370 multiplica los contenidos de los registros R4 y R7, considerándolos representados en coma flotante de 32 bits; por el contrario, la instrucción 1C47 multiplica dichos registros, pero suponiendo que se hallan representados en complemento a dos.
3. Para facilitar la decodificación, es bastante sistemática la codificación.
4. Las instrucciones son autocontenidas e independientes, lo que significa que poseen toda la información precisa para su ejecución y no dependen de la posición que ocupen en la memoria o en el programa.

El hecho de que las instrucciones máquina sean autocontenidas y que deban encadenarse para formar programas, implica que contienen las siguientes informaciones.

- Operación a realizar
- Identificación de los operandos que participan en la operación.
- Identificación del destino o resultado
- Situación de la siguiente instrucción

La definición y las características de un juego de instrucciones, vienen fijadas por los siguientes puntos:

- a) *Representaciones posibles* de los datos (ya comentadas en un capítulo previo).
- b) *Modos de direccionamiento*, que son los diversos sistemas con los que se localizan los datos y las instrucciones.
- c) *Operaciones* que se pueden efectuar
- d) *Formato* de las instrucciones, es decir, modo en el que se codifican.

5.3. MODOS DE DIRECCIONAMIENTO

Son las diversas formas que disponen las instrucciones para determinar el valor de un operando o la posición de un operando o una instrucción. La denominación de "modos de direccionamiento" proviene del hecho de que, normalmente, se especifica la dirección donde se encuentra el dato o la instrucción.

Se llama *objeto* a la instrucción, operando o resultado que se desea direccionar. El objeto puede residir en la propia instrucción, en un registro o en la memoria principal y la finalidad de los modos de direcciónamiento es especificar el lugar donde se encuentra el objeto.

En bastantes ocasiones la instrucción no contiene al objeto o a su dirección real, puesto que así se obtienen las siguientes ventajas:

- Ahorro de espacio.* Las instrucciones conforman programas cortos, que se leen y ejecutan más rápidamente.
- Código reubicable y reentrante.* Código reubicable es el que se puede ejecutar en cualquier zona de memoria y código reentrante es aquél que puede llamarse a sí mismo.
- Estructuras de datos.* Se simplifica notablemente el manejo de estructuras de datos (tablas, matrices, colas, listas, etc.) mediante ciertos modos de direcciónamiento.

En la figura 5-3 se muestran clasificados, los diversos modos fundamentales,

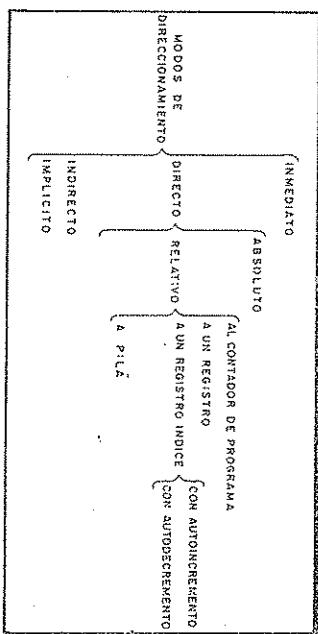


Fig. 5-3. Modos de direcciónamiento generales.

5.4. DIRECCIONAMIENTO INMEDIATO

El objeto, que en este caso es un operando, se encuentra contenido en la propia instrucción. Figura 5-4.

190 / Modos de direcciónamiento y repertorio de instrucciones

Ejemplos

1. Código OP: A9

Función: Carga inmediata del registro A.

UCP: 6502

Instrucción: A900

Expresión gráfica: (A) ← 00

2. Código OP: 16

Función: Carga inmediata del registro D.

UCP: Z-80

Instrucción: 164B

Expresión gráfica: (D) ← 4B

3. Código OP: 92

Función: Transferir un dato inmediato a una posición de memoria.

UCP: IBM 370

Instrucción: 92 7C 47B8

Expresión gráfica: M (x) ← 7C. El dato inmediato 7C se carga en la posición de memoria x, que se calcula con la información 47B8, mediante un direcciónamiento relativo con un registro base.

Muchas máquinas admiten diversos tamaños del operando inmediato. Así, VAX permite operandos inmediatos de 6, 8, 16, 32 y 64 bits, con lo que el tamaño del dato se adapta mejor al espacio de memoria.

5.5. DIRECCIONAMIENTO DIRECTO ABSOLUTO

En principio, se llama direcciónamiento *directo* al que expresa la dirección real del objeto, en contraposición con el indirecto.

Direcciónamiento *absoluto* expresa que la instrucción contiene la dirección efectiva del objeto, sin compactar.



Fig. 5-4. Formato de una instrucción con modo de direcciónamiento inmediato.

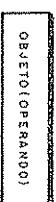


Fig. 5-5. Formato de una instrucción con modo de direcciónamiento directo absoluto.

Direccionalamiento *directo absoluto* significa que la instrucción contiene la dirección real, sin compactarla, del objeto.

Este direccionalamiento admite tres variantes:

1. La información contenida en la instrucción es una dirección completa de la memoria principal. El número de bits que se emplean depende del mapa de memoria que puede direccionar la UCP. Así, por ejemplo, los microprocesadores de 8 bits, como el Z-80, el 6502, el 8085, con un mapa de memoria de 64K necesitan 16 bits para formar una dirección. Figura 5-5.



Fig. 5-5. Direccionalamiento directo absoluto, en el que, dentro de la instrucción, se proporciona la dirección completa donde se encuentra el objeto. En un microprocesador de 8 bits, el código OP es de 8 bits y la dirección consta de 16 bits.

Ejemplo

La instrucción 3A35F7 corresponde a una instrucción del Z-80 que transfiere al registro A el contenido de la posición de memoria F735.

2. La información contenida en la instrucción es una dirección *limitada*, que hace referencia a una parte determinada del mapa de memoria.

En los microprocesadores es muy usual este tipo de direccionalamiento, que reduce la información que acompaña a la instrucción. Suele llamarse "direccionalamiento por página base" o "por página cero". En este último caso, en lugar de proporcionar los 16 bits de la dirección, sólo se suministran 8 en la instrucción, quedando fijados los otros 8 a un valor determinado, que suele ser cero.

Ejemplo

La instrucción A561 corresponde al microprocesador 6502, transfiere al registro A el contenido de la dirección de memoria 0061.

3. La información contenida en la instrucción es el *identificador de un registro* en el que se halla almacenado el objeto.

5.6. DIRECCIONAMIENTO DIRECTO RELATIVO

La instrucción no contiene la dirección del objeto, sino un *desplazamiento D*, sobre una dirección marcada por un "puntero". La dirección efectiva se calcula sumando el desplazamiento D al puntero de referencia que se encuentra en un registro.

Este tipo de direccionalamiento presenta como principal ventaja la utilización de menos bits que el direccionalamiento directo absoluto, puesto que el desplazamiento puede tener menos bits que los que definen una posición de memoria principal, ya que el registro puntero puede ser de cualquier tamaño. Sin embargo, con el direccionalamiento relativo hay que efectuar una operación de suma (registro puntero + desplazamiento).

El desplazamiento D puede ser positivo o negativo, por ejemplo, representando a D en complemento a dos. Así se accede a una zona alrededor de la posición marca da por el puntero.

En la figura 5-6 se representa la zona accesible para el caso de un desplazamiento de n bits en complemento a dos, en un mapa de memoria cuya dirección es de m bits, de forma que $n < m$, siendo n el número de bits del puntero y m el del bus de direcciones.

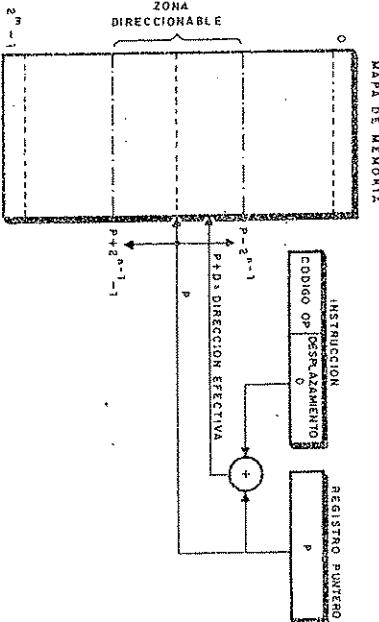


Fig. 5-6. Expresión gráfica, que representa el modo de funcionamiento del direccionalamiento relativo.

La suma que precisa este tipo de direccionalamiento retardará un poco la ejecución de la instrucción.

Las bases del código reentrante y reubicable, así como algunas técnicas de protección de memoria, residen en este direccionalamiento.

Según el registro puntero que se use y su tratamiento, se producen diversas variantes del direccionamiento relativo, que se comentan a continuación.

5.6.1. Direccionamiento relativo al Contador de Programa

Se usa al Contador de Programa como registro puntero. Es un direccionamiento muy adecuado para alcanzar instrucciones próximas a las que se está ejecutando, por ejemplo, en las bifurcaciones que dan lugar a los bucles. Figura 5-7.

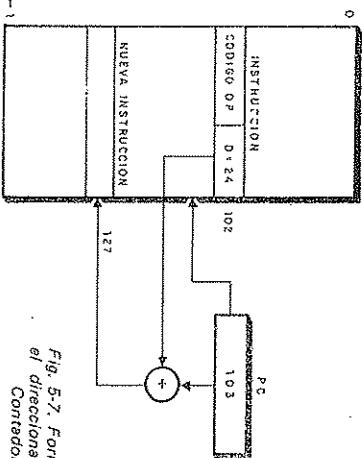


Fig. 5-7. Forma de actuación en el direccionamiento relativo al Contador de Programa.

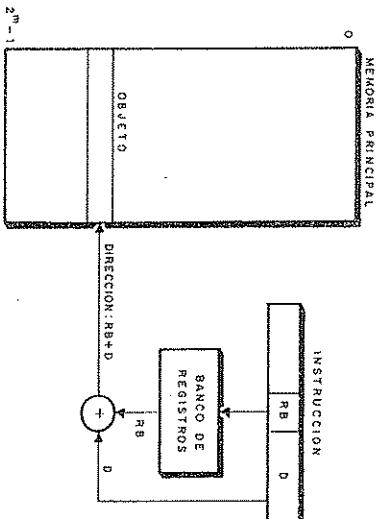


Fig. 5-8. Direccionamiento directo relativo a un registro base (RB).

Conviene resaltar que este direccionamiento se diferencia del relativo a "registro índice", en que el valor del registro base no suele modificarse y el del índice, sí.

El direccionamiento relativo a registro base es muy apropiado para manejar zonas de datos. Cargando a RB con la primera posición de dicha zona, se alcanza cualquier dato sin más que conocer su posición relativa.

Ejemplo

Instrucción: 92 7C 47B8

Expresión gráfica: $M(RB + 7B8) \leftarrow 7C$

Es una instrucción del IBM 370 que dirige la ejecución de forma relativa y viene especificada por la información 47B8. El primer carácter (4) representa al registro base R4 y los otros (7B8), el desplazamiento. La dirección del destino será R4 + 7B8 y en ella se cargará el dato inmediato 7C, que también contiene la instrucción.

5.6.2. Direccionamiento relativo a registro base

Se usa como registro puntero un "registro base" (RB). La UCP dispone, en este caso, de varios registros base que pueden ser de propósito general o de uso específico.

La instrucción, además del desplazamiento D, deberá contener el identificador del registro base. Figura 5-8.

194 / Modos de direccionamiento y repertorio de instrucciones

5.6.3. Direccionamiento relativo a registro índice

Es una variante del direccionamiento anterior, en el que el registro puntero es un registro índice (RI), cuyo valor se modifica para ir recorriendo los elementos de una tabla o vector. En efecto, si cada elemento de una tabla ocupa una palabra de me-

moria Y, cada vez que se emplea el registro índice, se incrementa una unidad, se van obteniendo las direcciones de los elementos sucesivos de la tabla.

Es muy frecuente que se realice de forma automática el "autoincremento" o el "autodecremento" del RI, dando lugar a las siguientes variantes:

- a) *Pre-autoincremento*: Primero se incrementa RI y luego se obtiene la dirección del objeto mediante la suma $(R_I + D)$.
- b) *Pre-autodecremento*: Se decrementa RI y luego se obtiene la dirección de la suma $(R_I - D)$.
- c) *Autoincremento*: Primero se calcula la dirección $(R_I + D)$ y luego se incrementa RI.
- d) *AutodeCREMENTO*: Se calcula la dirección $(R_I + D)$ y seguidamente se decremente RI.

Los incrementos y decrementos de RI se han de adaptar a la longitud de los operandos. El VAX permite operandos de 1, 2 ó 4 bytes y posee direccionamiento con autoincremento o autodecremento, que son de 1, 2 ó 4 unidades, de acuerdo con el tamaño en bytes del operando.

Ejemplo

Instrucción: A0514382

Expresión gráfica: $R_1 \leftarrow R_1 + M(R_3 + 2.R_2); R_3 \leftarrow R_3 + 2$.

Se trata de una instrucción del VAX que suma dos palabras de 2 bytes cada una. El primer operando se encuentra en el registro R1 y el segundo está en memoria en la posición $R_3 + 2.R_2$, donde R2 se multiplica por dos, al ser de 2 bytes los operandos, R3 se postautoincrementa en 2 por la misma razón. Los números 5, 4 y 8, que preceden a los valores de los registros usados, sirven para definir el modo de direccionamiento empleado.

5.6.4 Direcciónamiento a pila

Se trata de un caso particular del direccionamiento relativo y es muy usado en microprocesadores y microcomputadores.

La UCP debe disponer de uno o varios *registros punteros de pila* (SP), conteniendo la dirección de memoria principal donde se ubica la cabecera de la pila.

La pila puede crecer según direcciones de memoria ascendentes o descendentes. En el primer caso, si $SP = 1003$ y se introduce un nuevo elemento en la pila, el $SP = 1004$; si, por el contrario, se elimina un elemento de la pila, $SP = 1002$. Los direccionamientos requeridos para soportar este funcionamiento del SP son:

1. Para insertar un elemento se usa un direccionamiento relativo al registro SP con pre-autoincremento.
2. Para extraer un elemento, el direccionamiento es con autodecremento.

Si la pila crece según direcciones decrecientes, la inserción de un elemento precisa de preautoincremento.

Las instrucciones con direccionamiento a pila son *muy compactas*, puesto que, si sólo existe un registro SP la instrucción no requiere información alguna sobre dirección.

5.7 DIRECCIONAMIENTO INDIRECTO

En este caso se comienza con un direccionamiento directo (absoluto o relativo) que proporciona una dirección que contiene la dirección del objeto y no al propio objeto como ocurría con los modos anteriores. En consecuencia, para corregir el valor del objeto, hay que realizar un acceso adicional a la memoria principal. Figura. 5.9.

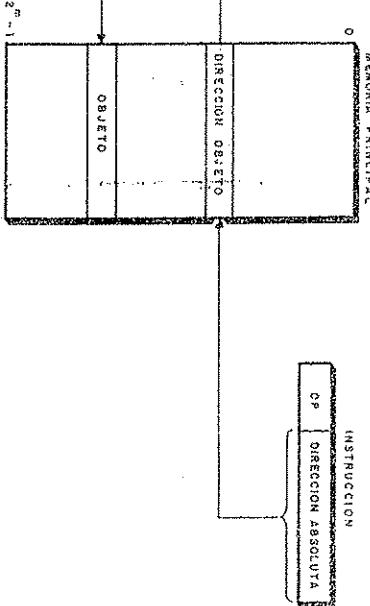


Fig. 5.9 Esquema de funcionamiento del direccionamiento indirecto.

En la figura 5.10 se presenta un ejemplo de "indirección" de un nivel, aunque no parece que ofrezca mucha utilidad, puesto que exige varios accesos a memoria y ocupa varias posiciones.

La indirección se puede añadir a todos los direccionamientos expuestos anteriormente.

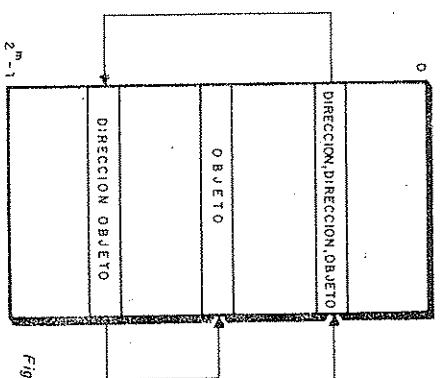


Fig. 5-10. Direccionamiento indirecto de "un nivel".

5.8. DIRECCIONAMIENTO IMPLICITO

La instrucción no contiene información sobre la situación del objeto, porque éste se encuentra en un lugar determinado (registro o posición de memoria).

Este direccionamiento presenta la ventaja de no ocupar espacio en la instrucción, aunque la hace muy restringida y poco flexible.

En muchos microprocesadores que disponen de registro Acumulador, existen varias instrucciones que emplean direccionamiento implícito del registro A.

5.9. EJEMPLOS REALES DE MODOS DE DIRECCIONAMIENTO

Citaremos los modos de tres UCP comerciales.

5.9.1. IEEE P 694

Los modos del estándar IEEE P694 se ofrecen en la tabla de la figura 5-11.

En la figura 5-11 se llama direccionado "índice" al relativo general, que se obtiene sumando un desplazamiento más un registro llamado índice.

198 / Modos de direccionamiento y repertorio de instrucciones

MODO DE DIRECCIONAMIENTO	PREFIJO	EJEMPLO
ABSOLUTO	/	/ DIR
PAGINA BASE	!	! DIR
INDIRECTO	[]	[DIR]
RELATIVO AL PC	\$	\$ DIR
INMEDIATO	#	# VALOR
INDICE	[]	[DIR [INDICE]]
REGISTRO	*	* DIR
AUTOPREINCREMENTO	++	++ DIR
AUTO POST INCREMENTO	++	DIR ++
AUTOPREDECREMENTO	--	-- DIR
AUTO POST DECREMENTO	--	DIR --
INDIRECTO PREINDEXADO	[]	[DIR [INDICE]]
INDIRECTO POST INDEXADO	[]	[DIR [INDICE]]

Fig. 5-11. Modos de direccionamiento según la norma IEEE P694.

El direccionado indirecto preindexado realiza primero la suma ($R_i + D$) y luego la indirección. El postindexado funciona de forma opuesta.

5.9.2. Microprocesador 68000

En la figura 5-12 se presenta una tabla con los modos de direccionamiento por "dirección efectiva" del microprocesador 68000 de Motorola.

5.9.3. VAX-11

En esta familia de minicomputadores hay 7 modos que utilizan registros generales.

1. Por registro: Un registro contiene el operando.
2. Por registro indirecto: Un registro contiene la dirección del operando.
3. Con autodecrecimiento: El contenido del registro, primero se decremente según el tamaño del operando (una unidad por byte) y luego se usa como dirección del operando.

TIPO	SINTAXIS	LUGAR DONDE RADICA EL OPERANDO	DIRECCION Efectiva		modo de ejecución
			MODO	REGISTRO	
por registro de datos	R_n	EN UNO DE LOS REGISTROS DE DATOS	000	REG. REG. (000)	
por más de direcciones	A_n	EN UNO DE LOS REGISTROS DE DIRECCIONES	001	REG. REG.	
por registro de dirección	LA_n	EN LA DIRECCION CONTENIDA EN UN REGISTRO DE DIRECCIONES	010	REG. REG.	
registro de indirecto	LA_n	EN LA DIRECCION CONTENIDA EN UN REGISTRO DE DIRECCIONES	010	REG. REG.	
registro indirecto	$(A_n)_+$	EN LA DIRECCION CONTENIDA EN UN REGISTRO DE DIRECCIONES	011	REG. REG.	
con post incremento	$(A_n)_+$	EN LA DIRECCION CONTENIDA EN UN REGISTRO DE DIRECCIONES	011	REG. AN. INC.	
registro indirecto	$- LA_n$	EL REG. AN. SE INCREMENTA EN 1	100	REG. REG.	
con predecremento	$- LA_n$	EL REG. AN. SE DECREMENTA EN 1	100	REG. REG.	
registro indirecto	d_{16}	LA DIRECCION DE OPERANDO SE HALLA SUMANDO DEL CONTENIDO DE UN A, CON UN DESPLAZAMIENTO DE 16 BITS (COPR.) A ZO DADO	101	Nº REG. REG.	
REG. INDIRECTO-INDIRECTO	d_{16}, X_n	LA DIRECCION SE HALLA SUMANDO AL CONTENIDO DE UN REG. IND. DE 16 BITS, LA DIRECCION DE OPERANDO SE HALLA SUMANDO AL CONTENIDO DE UN REG. IND. DE 16 BITS	110	Nº REG. REG.	
absoluto	$a_{16} + W$	LA DIRECCION DE OPERANDO SE COMO 16 BITS EN LA INSTRUCCION DE TRAS DE LA PALABRA OP	111	0 0	
absoluto	L_n	LA DIRECCION DETRAS DE LA PALABRA OP S	111	0 0	
relativo al contador	d_{16}	LA DIRECCION DEL OPERANDO SE HALLA SUMANDO A LA DIRECCION DEL OPERANDO SE HALLA SUMANDO A UN DESPLAZAM. DE 16 BITS, LA DIRECCION DE TRAS DE LA PALABRA OP, QUE NO CONTIENE 'P' (Z+2)	111	0 0	
de programa con desplazamiento	$d_{16}(PC)$	LA DIRECCION SE HALLA SUMANDO UN DESPLAZAM. DE 16 BITS, CON EL CONTENIDO DE UN REG. (INCRE. O DESP. DE DATOS O DIRECCIONES) Y CON LA DIRECCION QUE CONTIENE EL DESPLAZAMIENTO (PC+2)	111	0 1	
inmediato	I_m	INSTRUCCIONES EN LA PROPIA INSTRUCCION (8,16,32)	111	1 0 0	

Fig. 5-12. Modos de direccionamiento del microprocesador 68000 de Motorola.

5.10 CARACTERISTICAS Y TIPOS DE INSTRUCCIONES

El juego de instrucciones de un computador debe ser *completo y eficaz*.

Un juego de instrucciones completo permite calcular en un tiempo finito cualquier tarea computable. Además, el juego de instrucciones ha de ser eficaz, lo que equivale a poseer una alta velocidad de cálculo sin exigir una excesiva complejidad de la Unidad de Control.

Una función $f(x)$ es "computable" cuando puede calcularse en una máquina de Turing con un número de pasos finito.

El juego de instrucciones puede ser realmente sencillo. La máquina de Turing utiliza sólo 4 instrucciones:

- Escritura.

- Mover a la izquierda una posición y leer.

- Mover a la derecha una posición y leer.

- Parar.

El juego de instrucciones puede ser muy reducido. Así Minsky ha demostrado que con sólo 2 instrucciones: *incrementar y decrementar y bifurcar si cero*, se puede resolver cualquier problema resoluble por otros computadores. Von der Poel, ha diseñado un computador de una sola instrucción.

Definir el juego de instrucciones de un computador es uno de los puntos más críticos de su diseño.

Sé hace un estudio de las instrucciones más utilizadas en los computadores, clasificadas en 8 grupos, tal como lo establece Fairclough:

1. Movimiento o transferencia de datos
2. Rotura de la secuencia del programa
3. Aritméticas
4. Lógicas
5. De comparación
6. De bit
7. De desplazamiento
8. De entrada y salida y diversas

Las instrucciones máquinas son cadenas de "unos" y "ceros", totalmente inexpressivas para el ser humano. Para su fácil identificación, se usan nomencláculos, que son un conjunto de letras representativas de la operación que realiza la instrucción recogidas de su nombre en inglés. Este tipo de representación lo emplea el lenguaje Ensamblador del que se hablará posteriormente.

6. Por desplazamiento: El valor almacenado en el registro se usa como base de direcciones de una tabla de direcciones. También se suma un desplazamiento de 8, 16 ó 32 bits.
7. Por desplazamiento: El valor almacenado en el registro se usa como base de direcciones de una tabla de direcciones. También se suma un desplazamiento de 8, 16 ó 32 bits.
- Finalmente, además de existir direccionado inmediato, todos los modos anteriores (excepto por registro) pueden utilizar un registro índice para modificar su valor.

5.11. INSTRUCCIONES DE MOVIMIENTO O TRANSFERENCIA DE DATOS

Las instrucciones de este grupo permiten repetir en el operando destino la información almacenada en el operando origen, sin que éste se modifique. Destino y origen pueden ser registros o posiciones de memoria.

Aunque la transferencia normal se efectúa con palabras, también pueden afectar a una parte fraccionaria o a múltiples palabras.

A continuación se ofrece la denominación más frecuente de la mayoría de las instrucciones de este grupo, siguiendo las reglas recomendadas por la normativa IEEE 694.

MOVE. Transfiere el contenido de un registro a otro o de una posición de memoria a otra.

STORE. Transfiere el contenido de un registro a memoria.

LOAD. Transfiere el contenido de una posición de memoria a un registro.

MOVE BLOCK. Transfiere un bloque de datos.

TABLA RESUMEN DE LAS CARACTERÍSTICAS FUNDAMENTALES DE LAS INSTRUCCIONES DE TRANSFERENCIA					
	EJEMPLO DE SINTAXIS	EXPRESIÓN GRÁFICA	OPERANDO AFFECTADO	PLAZOS	COMENTARIOS
MOVE	MOVE D3, \$ 2000	[posic.] → (\$ 2000)	16,32	Indirecto	TRANSFIRE INFORMACIÓN ENTRE CUATRO REGISTROS EN UNA ZONA DE MEMORIA
MOVE	MOVEW A1,4,\$ FF	[A1(32,...,A6,31)] →	16,32	—	TRANSFIRE ENTRE REGISTROS LOS CONTENIDOS DE UNA ZONA DE MEMORIA
MOVEP	MOVEP L0,2(4,1)	[L0(32)] → [REGISTRO CON DIRECCIÓN DE 2]	16,32	—	TRANSFIRE LOS REGISTROS DE 8 BITS CONCERNIOS A LOS DÍGITOS PAREJAS 00-07 (IMPARES) Y VICEVERSA CON EXTSIGNO A 32 BITS EN EL REG. INDICADO
MOVE	MOVE #F2,03	# F2 → (03) EXTSIGNO	32	Indirecto	TRANSFIRE LOS CONTENIDOS DE DOS REGISTROS ENTRE SI
EXG	EXG D2,A3	(D2) ↔ (A3)	32	—	INTERCAMBIA LOS CONTENIDOS DE DOS REGISTROS ENTRE SI
SWAP	SWAP D3	D3(31-16) ↔ D3(15-0)	16	Indirecto	INTERCAMBIA ENTRE SI LAS DOS MITADES(ES) DE UN REG. DE DATOS
LEA	LEA (A1),R5	(A1) → R5	32	—	CARGA LA DIRECCIÓN EFECTIVA INDICADA EN UN REG. DIRECCIONES
PEA	PEA 1,4,0	(1,8) → SP	32	—	CARGA LA DIRECCIÓN EFECTIVA INDICADA DENTRO DEL SP

Fig. 5-13. Características más significativas de las instrucciones de transferencia del microprocesador 68000.

202./ Modos de direccionamiento y repertorio de instrucciones

MOVE MULTIPLE. Copia el contenido del origen en múltiples posiciones de memoria.

EXCHANGE. Intercambia el contenido de los operandos especificados.

CLEAR. Pone a "ceros" el destino. A veces se llama RESET.

SET. Pone a "unos" el destino.

PUSH. Transfiere el operando origen a la cabecera de la pila.

POP. Transfiere al destino la cabecera de la pila.

5.12. INSTRUCCIONES DE RUPTURA DE SECUENCIA

Este grupo de instrucciones altera la secuencia normal de ejecución del programa. En vez de pasar a la instrucción que ocupa el siguiente lugar del programa, se irá a otra posición de memoria.

En la secuencia normal de un programa el PC se incrementa para pasar a ejecutar la siguiente instrucción. Si la instrucción en curso tiene un tamaño de K palabras, PC se incrementa en K unidades para acceder a la siguiente instrucción. Si, en lugar de dicho incremento, el PC se carga con una nueva posición, será la instrucción allí existente la que se ejecuta. Se dice que el programa se ha bifurcado.

Hay dos tipos de bifurcaciones:

- Condicionales.
- Con retorno.

5.12.1. Bifurcaciones condicionales

Las instrucciones de bifurcación condicional dan lugar a dos secuencias distintas del programa:

1. Cuando no se cumple la condición de bifurcación, se sigue la secuencia normal y el PC → PC + K.
2. Cuando se cumple la condición de bifurcación, el PC se carga con la dirección de la bifurcación.

Las condiciones de bifurcación se establecen sobre los bitsables de estado, que almacenan ciertas condiciones de las operaciones realizadas automáticamente y que se agrupan en el denominado Registro de Estado. Las condiciones pueden basarse en el estado de uno o varios bitsables.

La norma IEEE P649 comprende las siguientes condiciones que se relacionan al lado de su correspondiente mnemotécnico o nómico.

ZERO (Z)	Cero
NOT ZERO (NZ)	No cero
EQUAL (GE)	Igual
NOT EQUAL (NE)	Diferente
CARRY (C)	Acarreo
NOT CARRY (NC)	Sin acarreo
POSITIVE (P)	Positivo
NEGATIVE (N)	Negativo
OVERFLOW (V)	Desbordamiento
NO OVERFLOW (NV)	Sin desbordamiento
GREATER THAN (GT)	Mayor que
GREATER THAN OR EQUAL (GE)	Mayor que o igual a
LESS THAN (LT)	Menor que
HIGHER (H)	Menor que o igual a
NOT HIGHER (NH)	Superior a
LOWER (L)	No superior
NOT LOWER (NL)	Inferior a
PARTY EVEN (NL)	No inferior a
PARTY ODD (PO)	Paridad par
	Paridad impar

Las condiciones enunciadas consideran a los operandos sin signo, por lo que son distintas de las de "mayor" y "menor".

5.12.2. Bifurcaciones con retorno

Estas instrucciones *salvan* la dirección de la instrucción que ocupa la posición siguiente, o sea, el valor (PC + K). Con ellas se puede *retornar* al punto donde se produjo la bifurcación y continuar en la siguiente dirección.

Se usan en las *"llamadas a subrutina"*, que reciben el nombre de CALL o BRANCH TO SUBROUTINE. En la figura 5-14 se muestra la función de este tipo de instrucción, en la que aparece un programa con varias llamadas a la misma subrutina S1. Al saltar a S1, se guarda la dirección de partida, y al volver a la subrutina, se carga el PC con el mismo punto. En la primera llamada se salva la dirección 121 y se carga el PC con la dirección 1200, que es la de principio de S1. Al acabar S1 se recupera 121 y el PC continúa con esta dirección. Con la segunda llamada, se salva la dirección 246 y al completarse S1, se recupera.

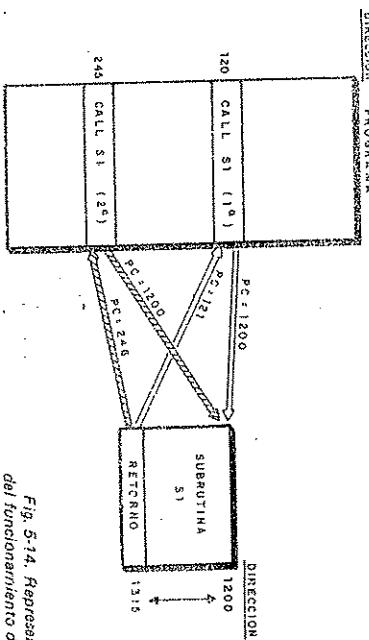


Fig. 5-14. Representación gráfica del funcionamiento de una subrutina.

Una de las características fundamentales de las bifurcaciones con retorno es la que determina el lugar donde se guarda temporalmente la dirección de retorno. De dicho lugar depende que se puedan realizar *llamadas anidadas*, o sea, que una subrutina llame a otra, y *llamadas sucesivas*, que significa que una subrutina se llama a sí misma. Se enumeran las alternativas más usuales empleadas para guardar la dirección de retorno.

a) *Registro especial*. Si se guarda la dirección de retorno en un registro determinado, no se pueden realizar llamadas anidadas.

b) *Registro general*. Tiene el mismo inconveniente que el caso anterior y, además, ocupa uno de los registros de trabajo de la UCP.

c) *Almacenamiento en la propia subrutina*. Por ejemplo, se puede reservar la primera palabra de la subrutina para almacenar la dirección de retorno. Con este procedimiento se soportan llamadas anidadas, pero no recursivas, porque devueltan las posiciones de retorno anteriores.

d) *Almacenamiento en una pila de control*. El sistema debe mantener una *pila* organizada en la memoria donde se almacenan las direcciones de retorno.

Al inicializarse una subrutina se deben realizar las dos operaciones siguientes:

PUSH PC + K (Se salva en la pila PC + K)

PC ← Dirección de subrutina

Al finalizar la subrutina basta con realizar la instrucción POP PC (Carga en el PC el último elemento de la pila) para que continúe el programa en el punto donde se saltó a la subrutina.

Para que funcione correctamente la subrutina debe dejar a la pila en el mismo nivel en que la recibió.

Este procedimiento permite llamadas anidadas y recursivas, ya que cada nueva llamada va introduciendo en la pila su dirección de retorno sin destruir las anteriores.

Como los retornos se hacen en orden inverso a las llamadas, la pila siempre tiene en la cabecera la dirección de retorno adecuada.

Tengase en cuenta que la dirección de retorno sólo es uno de los parámetros que se envían a una subrutina.

Las denominaciones más frecuentes empleadas en estas instrucciones, recogidas en su mayor parte del estándar IEEE P694, son:

BRANCH: Bifurcación condicional o incondicional. También recibe el nombre de JUMP.

CALL: Bifurcación con retorno, que se emplea para llamar a una subrutina. Puede ser condicional o incondicional. También recibe el nombre de JUMP TO SUBROUTINE o BRANCH AND LINK.

RETURN: Restituye al PC la dirección de retorno de la subrutina. Existe el caso especial de "retorno desde una interrupción" que recibe el nombre de RETURN FROM INTERRUPT.

SKIP: Es una bifurcación condicional especial, muy compacta, que sólo salta una instrucción, por lo que no necesita guardar la dirección de retorno. Se emplea en unión con un BRANCH incondicional para construir bucles.

Las denominaciones más corrientes, según la norma del IEEE P694, son:

ADD

Suma

ADD WITH CARRY

Suma añadiendo el acarreo del lado anterior.

SUBTRACT

Resta teniendo en cuenta la llevada de la resta anterior.

INCREMENT

Incremento

DECREMENT

Decremento

MULTIPLY

Multiplicar

DIVIDE

Dividir

EXTEND

Extensión de un operando a un tamaño mayor.

NEGATE

Cambio de signo

Valor absoluto

Complemento

Tabla resumida de las instrucciones aritméticas

MENÚNECO	TIPO DE SINTAXIS	EXPRESIÓN GRÁFICA	OPERANDO AFFECTADO	FLAGS	COMENTARIOS
A, DD	ADD D2, D3	[D2][D3] → [D3][D2]	6,16,32	N,Z,C,V,X	SUMA ENTRE FUENTE Y DESTINO. NO SE REGISTRA EN OPERADOR.
ADDA	ADD A3,B,-(A2)	(A2,-B)+A3 → (A2)	16,32	—	SUMA CON UNO AL OTRA DIFERENCIA. ADICIONAR DATO INMEDIATO.
ABDI	ADD #3,D5	(D5)+→(D5)	6,16,32	N,Z,C,V,X	ADICIONAR DATO INMEDIATO ES DE 12.
ADDX	ADDX D1,D3	(D1)+(D3)X → (D3)	8,16,32	N,Z,C,V,X	SUMA MULTIPLE. SUMA RESULTADO ES 6,12 NO CAMBIAR.
ABTD	ABTD D1,D2	(D1)-(D2)X → (D2)	0	C,Z,X	SUMA BCD CON EL BIT DE EXTENSIÓN.
SUB	SUB D6,F7,05	(D6)-(F7,05) → (D6)	8,16,32	N,Z,C,V,X	RESTA BINARIA.
SUBA	SUBA D2,A5	(A5)-(D2) → (A5)	16,32	—	EL OPERANDO NO ES UNA DIRECCIÓN.
SUBI	SUBI D3,F4,S,F0	(F0)-(D3) → (F0)	8,16,32	N,Z,C,V,X	RESTA DE UN DATO INMEDIATO. SUMO SIEMPRE COMO DATO INMEDIATO.
SQRT	SQRT D1,D2	(D2)-(D1)X → (D2)	8,16,32	N,Z,C,V,X	RESTA BINARIA CON BIT DE EXTENSIÓN.
NEG	NEG \$ F4,S,-	(-1,F4,B,S) → \$ F4,F0	8,16,32	N,Z,C,V,X	COMPLEMENTO A 2. RESTA OPERANDO.
NEGX	NEG X D3	(-1,D3)-X → (D3)	8,16,32	N,Z,C,V,X	EL SE REGISTRA EL OPERANDO.
NO.C	NO.C D2	(-1,D2)-X → (D2)	8	X,Z,C	SI X=1 COMPLEMENTA A 0.
MULU	MULU D4,D3	(D4)(D3) → (D3)	16	Z,N	MULTIPLICACIÓN SIN SIGNO.
MULS	MULS D4,D3	(D4)(D3) → (D3)	16	Z,N	MULTIPLICACIÓN CON SIGNO.
DIVUD	DIVUD D2,D4	(D4)(16) → (D2)	32-16	N,Z,V	DIVISION SIN (DIV) CON 9 SIGNOS.
EXT-L	EXT-L D5	EXT DE SIGNO DEL DATO 15A13	16,32	H,Z	EXTENSION DE SIGNO.
CLR	CLR L D2	0 → (D2)	16,32	—	PONE A LOS BITS INDICADOS.
CMP	CMP D3,D2	(D3)(D2) → FLAGS	6,16,32	N,Z,C,V	RESTA CON OPERANDO SOLO FLAG.
CMPA	CMPA \$F0,A2	(A2)-(F0) → FLAGS	16,32	N,Z,C,V	RESTA CON OPERANDO UN REG. DIRECCIONAL.
CMRI	CMRI D2,D3	(16,-D2) → FLAGS	8,16,32	N,Z,C,V	RESTA DATO INMEDIATO.
CMRM	CMRM A1,(A2),COPARACION	OPERA DIRECTORIOS CON POS. INCREMENTO.	8,16,32	N,Z,C,V	OPERA CON POS. INCREMENTO.
STI	STI B,D2	0 → (D2)	8,16,32	N,Z	COPAREA CON EL OPERANDO. FLAGS.
TAS	TAS D5,F5	F5 → (F5)	8	N,Y	LE PONE A SU SIT.

Fig. 5-15. Principales características de las instrucciones aritméticas del 60000.

- 5.13. INSTRUCCIONES ARITMÉTICAS**
- Son las que se refieren a las operaciones que puede llevar a cabo la Unidad Aritmética.

5

5.14. INSTRUCCIONES DE COMPARACION

La operación COMPARE consiste en restar o hacer la operación OR EXCLUSIVA, bit a bit, entre dos o más operandos. No se obtiene, ni queda almacenado, el resultado de la operación, y sólo quedan afectados los señalizadores del Registro de Estado. A este tipo de instrucciones le sigue una bifurcación condicional en base a los bitables que modifica.

La operación TEST es una comparación con cero.

5.15. INSTRUCCIONES LOGICAS

Realizan operaciones lógicas en todos los bits de los operandos, de forma independiente. Las más usuales son:

AND OR NOT XOR

5.16. INSTRUCCIONES DE DESPLAZAMIENTO

Se citan las más importantes:

SHIFT: Se trata de un desplazamiento lógico o aritmético de los bits del operando a derechas o a izquierdas.

ROTATE: Es una operación de rotación de los bits del operando a derechas o a izquierdas. En la rotación puede incluirse el acarreo.

5.17. INSTRUCCIONES DE BIT

Comprueban un bit del operando y su valor lo reflejan en el señalizador Z (cero). Además, pueden poner a 1 ó a 0 a dicho bit. Las más frecuentes son:

BIT TEST

Comprueba un bit.

BIT SET

Comprueba un bit y lo pone a 1.

BIT CLEAR

Comprueba un bit y lo pone a 0.

TABLA RESUMIDA DE LAS INSTRUCCIONES DE MANIPULACION DE BITS

MENÚNICO	EJEMPLO	EXPRESIÓN GRÁFICA	OPERANDO	FLAG	COMENTARIOS
TEST	NEUTR 01	TEST 01 (001) DE 01	0, 1, 32	Z	TESTADO DE UN BIT
BSET	BSET # 30 DS	01 → Z	8, 32	Z	TESTADO DE UN BIT Y PUESTA A 1
BCLR	BCLR # 32 FFO	01 → 01	8, 32	Z	TESTADO DE UN BIT Y PUESTA A 0
BCIO	BCIO # 3, 0 9	01 → Z	8, 32	Z	TESTADO DE UN BIT + INVERSIÓN

Fig. 5-16. Características del grupo de instrucciones encargadas de la manipulación de bits en el 68000.

5.18. INSTRUCCIONES DE ENTRADA Y SALIDA Y DIVERSAS

Las instrucciones de Entrada/Salida son, en realidad, instrucciones de transferencia, con la peculiaridad de que el destino o el origen es un registro de un periférico. Muchos computadores no tienen este tipo de instrucciones, realizándose el trámite de Entradas/Salidas como si fuesen posiciones de memoria, con las instrucciones LOAD, STORE o MOVE.

Las más corrientes son:

INPUT: Transfiere la información desde un registro de un periférico a la memoria o a un registro de la UCP. A veces, se denomina READ.

OUTPUT: Es la operación inversa a INPUT. También se llamará WRITE.

TEST I/O: Lee la información de control de un periférico.

CONTROL I/O: Envía información de control a un periférico.

Finalmente, los juegos de instrucciones disponen de algunas especiales, que no pueden incluirse en los grupos anteriores y entre las que destaca:

HALT: Sirve para detener la ejecución de un programa hasta que se produce una interrupción exterior que arranca la ejecución de otro programa.

WAIT: También detiene la ejecución del programa hasta que se cumple una condición interna o externa, que hace que prosiga el programa original.

CONVERT: Cambia los formatos de las instrucciones, generalmente para realizar operaciones de entrada y salida.

NO OPERATION: No realiza operación alguna.

5.19. FORMATO DE LAS INSTRUCCIONES

El formato de una instrucción define la longitud o número de bits que la componen y el significado o misión de cada uno de ellos.

La información que debe contener el código de una instrucción es la siguiente:

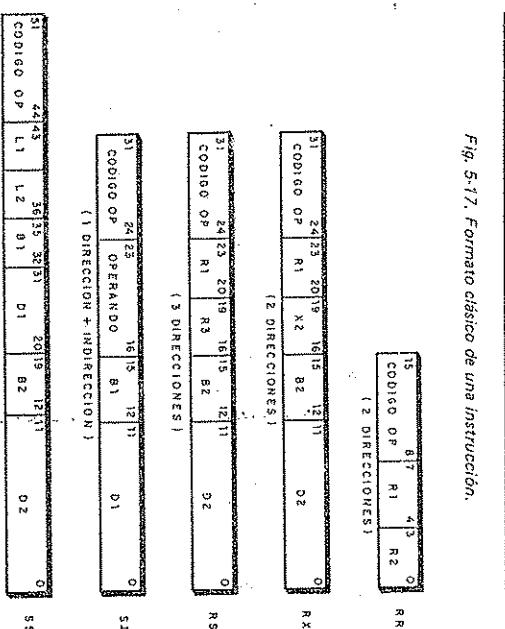
- Operación a realizar
- Dirección de los operandos
- Tipos de operandos
- Dirección del resultado

— Dirección de la siguiente instrucción.

El formato de una instrucción se divide en *campos*, que son cadenas de bits contiguos. Cada campo contiene un tipo de información específica. En la figura 5.17 se muestra un formato típico de una instrucción.

31	CÓDIGO OPERACIÓN	24-33	DIRECCIÓN OPERANDO 1	10-9	DIRECCIÓN OPERANDO 2	0
----	---------------------	-------	-------------------------	------	-------------------------	---

Fig. 5.17. Formato clásico de una instrucción.



Los dos campos básicos en el formato de una instrucción son el *código de operación*, que indica la operación a realizar y el *campo de dirección*, que determina la dirección de un dato, resultado o instrucción a la que hay que bifurcarse. Dependiendo del tipo de direccionamiento, el campo de dirección se divide en *subcampos*.

Por ejemplo, pueden existir 3 subcampos: uno que indica la existencia de indirección, otro que selecciona el registro índice y otro destinado al desplazamiento.

Además de los dos campos básicos, suelen encontrarse otros como el de extensión de código de operación, de longitud de operandos, de modo de direccionamiento, etc.

En la figura 5.18 están representados gráficamente los formatos de las instrucciones de la familia de computadores IBM 360/370.

El formato de las instrucciones de la familia VAX, que se muestra en la figura 5.19, es muy flexible. Se compone de un código de operación de 8 bits, seguido de hasta 5 especificaciones de operando. Cada especificador de operando se compone de 2 o 3 partes: 1) Código de direccionamiento; 2) Dirección de un registro de 2, 4 o 8 bits; 3) Campo de 8, 16 o 32 bits, que expresa un desplazamiento, un valor inmediato o una dirección.

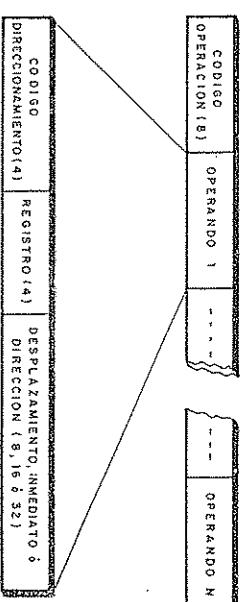


Fig. 5.19. Formato de las instrucciones de la familia de minicomputadores VAX.

5.20. CARACTERÍSTICAS DEL FORMATO DE LAS INSTRUCCIONES

El formato o formato de las instrucciones de un computador debe reunir las siguientes características:

1. Un computador debe poseer uno o unos pocos formatos de instrucción, donde deben encajar todas las instrucciones. *Cuanto menos formatos existan, más sencilla será la Unidad de Control* que las interpreta.

siguientes características:

1. Un computador debe poseer uno o unos pocos formatos de instrucción, donde deben encajar todas las instrucciones. *Cuanto menos formatos existan, más sencilla será la Unidad de Control* que las interpreta.
- Modos de direccionamiento y repertorio de instrucciones / 21 -

2. Los formatos son *requeridos* y *sistemáticos*. Para simplificar la decodificación y minimizar los caminos internos por donde se mueve la información de la instrucción, los campos del mismo tipo tienen la misma longitud y ocupan la misma posición. Así, por ejemplo, el primer campo es el código de operación.

3. Para acortar el tamaño se emplean técnicas de *direcccionamiento implícito*, como las siguientes:

a) Menos en las instrucciones de bifurcación, no hay campo de instrucción siguiente. Se sobreentiende que está situada físicamente a continuación.

b) No se asignan campos para especificar la representación de los operandos. El código de operación implica el tipo de los operandos. Así, por ejemplo, el VAX dispone de 14 códigos de operación para la suma. Si se usa el 80, se suman palabras de 16 bits y, si se usa el 40, se supone que los operandos son de 32 bits en coma flotante.

4. Los tamaños de los formatos *encojan* fácilmente en la palabra de la máquina. Lo más corriente es que el formato ocupe una palabra, pero también se usa de 1/2, 1/4 y 2 palabras.

5. En caso de existir varios formatos, el código de operación sirve para diferenciarlos.

6. Los tamaños de los campos que expresan direcciones, ya sea del banco de registros o de la memoria principal, deben corresponder a los mapas direccionalos. Por ejemplo, es bastante frecuente manejar un banco de 16 registros, lo que obliga a emplear campos de 4 bits para seleccionar uno de ellos. Las direcciones completas deben tener la longitud necesaria para abarcar todo el mapa; no así los desplazamientos que suelen ser más pequeños.

7. Se suelte diseñar un campo de código de operación de tamaño fijo en todos los formatos. Con d bits en este campo se dispone de 2^d códigos de operación diferentes.

Sin embargo, la frecuencia de utilización de las instrucciones difiere muchísimo de unas a otras, como se expondrá posteriormente. Por ello, se podría optimizar el espacio ocupado por los programas, empleando una codificación que utilice menos bits para las instrucciones más frecuentes. Huffman establece un procedimiento para asignar códigos de tamaños óptimos, partiendo de la tabla de utilización. En el caso de un computador con 8 instrucciones, cuyas utilidades, dadas en forma de probabilidad P_i , son las de la tabla de la figura 5-20, se obtienen las codificaciones que se indican según los diversos métodos.

2.1.2 / Modos de direccionamiento y repertorio de instrucciones

PROBABILIDAD DE USO P_i	CODIFICACION NORMAL	CODIFICACION HUFFMAN	CODIFICACION CON EXTENSION DE CAMPO
0,5	000	0	00
0,25	001	10	01
0,125	010	110	10
0,0625	011	11100	11000
0,03125	100	11101	11001
0,015625	101	11110	11010
0,0078125	110	111110	11011
0,00390625	111	111111	11100

Fig. 5-20. Tabla con diversas codificaciones del código de operación según la probabilidad de uso.

El numero medio de bits empleado con cada tipo de codificación se calcula con el sumatorio $\sum P_i C_i$; donde P_i es la probabilidad de uso de la instrucción i , y C_i es su número de bits. Con los códigos de la figura 5-20 se obtienen los siguientes resultados:

Codificación normal 3 bits

Codificación Huffman 2,08 bits

Codificación con extensión de campo 2,45 bits

Como resulta bastante engorrosa la decodificación de los códigos de Huffman, algunas máquinas emplean una solución intermedia, consistente en el uso de un campo de código de operación fijo, pero con un campo de extensión que se emplea en las instrucciones menos frecuentes.

8. Segundo el número de direcciones explícitas del formato, hay instrucciones de 0, 1, 2, 3, 4, ... direcciones. Para operaciones diádicas es muy frecuente el empleo de formatos de dos direcciones.

El número óptimo de direcciones del formato para minimizar la memoria ocupada por los programas depende de muchos factores: arquitectura, tipo de algoritmo a programar y forma de programar. A título de ejemplo, y sin tratar de sacar conclusiones absolutas, se plantea la programación de la siguiente expresión:

$$X = A/C + B \cdot D$$

Se supone que, tanto los datos como el resultado, residen en la memoria principal. Se emplean máquinas hipotéticas de 0, 1, 2 y 3 direcciones.

A. Máquina de 0 direcciones. El contenido de la pila se expresa <>.

PUSH A	<A>
PUSH C	<C, A>
DIVIDE	<A/C>
PUSH B	<B, A/C>
PUSH D	<D, B, A/C>
MULTIPLY	<B · D, A/C>
ADD	<B · D + A/C>
POP X	

En este programa se han empleado 8 códigos de operación y 6 direcciones de memoria (AC) y varios registros generales.

LOAD A

AC \leftarrow A

DIVIDE C

AC \leftarrow A/C

MOVE R1

R1 \leftarrow A/C

LOAD B

AC \leftarrow B

MULTIPLY D

AC \leftarrow AC · D; (B · D)

ADD R1

AC \leftarrow AC + R1; (B · D + A/C)

STORE X

X \leftarrow AC

Se emplean 7 códigos de operación, 5 direcciones de memoria y 2 de registros.

C. Máquina de 2 direcciones

MOVE R1, A	R1 \leftarrow A
DIVIDE R1, C	R1 \leftarrow A/C
MOVE R2, B	R2 \leftarrow B
MULTIPLY R2, D	R2 \leftarrow R2 · D; (B · D)
ADD R1, R2	R1 \leftarrow R1 + R2; (A/C + B · D)
MOVE X, R1	X \leftarrow R1

Se emplean 6 códigos de operación más 5 direcciones de memoria y 7 de registros.

2.14 / Modos de direccionamiento y repertorio de instrucciones

D. Máquina especial de 2 direcciones. Se supone que los operandos A y B se pueden destruir.

DIVIDE A, C	A \leftarrow A/C
MULTIPLY B, D	B \leftarrow B · D
ADD A, B	A \leftarrow A + B
MOVE X, A	X \leftarrow A

Emplea 4 códigos de operación y 8 direcciones en memoria.

E. Máquina de 3 direcciones

DIVIDE R1, A, C	R1 \leftarrow A/C
MULTIPLY R2, B, D	R2 \leftarrow B · D
ADD X, R1, R2	X \leftarrow R1 + R2

Emplea 3 códigos de operación, 5 direcciones a memoria y 4 registros.

Seguidamente se hacen dos supuestos para calcular el espacio ocupado por los programas anteriores:

- a) Códigos de operación de 8 bits. Direcciones a memoria relativas, específicas con registros (4 bits) y un desplazamiento de 12 bits. Direcciones de registro de 4 bits. Los bits requeridos en cada caso son los siguientes:

- | | |
|------------------|-----------------------------------|
| A. 0 direcciones | 8 · 8 + 6 · 16 = 160 bits |
| B. 1 dirección | 7 · 8 + 5 · 16 + 2 · 4 = 144 bits |
| C. 2 direcciones | 6 · 8 + 5 · 16 + 7 · 4 = 156 bits |
| D. 2 direcciones | 4 · 8 + 8 · 16 = 140 bits |
| E. 3 direcciones | 3 · 8 + 5 · 16 + 4 · 4 = 120 bits |

- b) Códigos de operación de 6 bits. Direcciones a memoria relativa al PC y un desplazamiento de 10 bits. Direcciones de registros de 4 bits. Ahora los bits que cada programa son:

- | | |
|------------------|-----------------------------------|
| A. 0 direcciones | 8 · 6 + 6 · 10 = 108 bits |
| B. 1 dirección | 7 · 6 + 5 · 10 + 2 · 4 = 100 bits |
| C. 2 direcciones | 6 · 6 + 5 · 10 + 7 · 4 = 114 bits |
| D. 2 direcciones | 4 · 6 + 8 · 10 = 104 bits |
| E. 3 direcciones | 3 · 6 + 5 · 10 + 4 · 4 = 84 bits |

Aunque los resultados apuntan una ventaja para la máquina de 3 direcciones, otro tipo de programas harían recomendable otra máquina. Además, también

Modos de direccionamiento y repertorio de instrucciones / 21

En sucesión en mayor complejidad de la Unidad de Control de una máquina de 3 direcciones.

9. El modelo de ejecución de un computador determina el dispositivo en que están almacenados los operandos para realizar las operaciones. Hay 4 modelos de ejecución:
- Pila
 - Registro-Registro
 - Registro-Memoria
 - Memoria-Memoria

El modelo de ejecución está estrechamente ligado al formato de las instrucciones, pues el direccionamiento implicado en cada modelo es diferente.

Hay máquinas con varios modelos de ejecución, que dan lugar a uno o varios formatos de instrucción para cada uno de ellos.

5.2.1. FRECUENCIA DE UTILIZACION DE LAS INSTRUCCIONES

Una de las principales características de la arquitectura de un computador es su juego de instrucciones. Consecuentemente, se han realizado múltiples estudios sobre la utilización de las instrucciones en los computadores comerciales, con objeto de optimizar futuros diseños.

Estos estudios se orientan al análisis de la *frecuencia de utilización* de las instrucciones, e, incluso, las secuencias de instrucciones más frecuentes, que podrían ser integradas en una sola instrucción.

El análisis de la frecuencia de utilización de las instrucciones puede ser estático o dinámico. En el primer caso, se consideran, únicamente, los listados de los programas, mientras que en el segundo se analizan las instrucciones que se van leyendo al ejecutar los programas. Obsérvese que, debido a los bucles y subrutinas, estas medidas son diferentes. Las medidas estáticas evalúan las necesidades de almacenamiento de los programas y las dinámicas la cantidad de accesos a memoria que hay que realizar para ejecutar un programa. Hay instrucciones con poca frecuencia de empleo desde el punto de vista estático, pero alta desde el dinámico. Así, la instrucción SKIP, que aparece una vez por bucle, se ejecuta en cada uno de sus pasos.

Los estudios realizados se refieren a un conjunto de programas, con los que se pretende reflejar la mayoría de las aplicaciones, escritas y ejecutadas sobre máquinas específicas. Son relevantes las investigaciones de:

216 / Modos de direcciónamiento y repertorio de instrucciones

1. Gibson.—Sobre porcentajes de utilización dinámica en un IBM 7090.

2. Ashley.—Realizados en un IBM 360 sobre uso dinámico de instrucciones en compiladores y trabajos científicos.
3. Flynn.—También en un IBM 360 sobre programas en Cobol.

4. Foster.—Implementando un computador CDC 3600 estudió la utilización estática y dinámica de las instrucciones.
5. Fairclough.—Que utilizó varios computadores y microprocesadores para estudio de la frecuencia de utilización de instrucciones.

De todos los estudios realizados pueden obtenerse las siguientes conclusiones:

- a) La mitad de las instrucciones se dedican a mover informaciones en el computador. Es el grupo más importante de instrucciones.
- b) Las bifurcaciones constituyen el segundo grupo de instrucciones más empieado. Ello se debe a que una gran cantidad de los bucles empleados son muy cortos.
- c) Aproximadamente el 50% de las instrucciones de los repertorios de los computadores se usan menos del 2%. Esto hace suponer que dichas instrucciones se podrían eliminar sin afectar a las prestaciones de la máquina. Sin embargo, algunas de esas instrucciones pueden ser muy importantes en algunas aplicaciones concretas.

Hasta la década de los 80 se ha proponido por un rico y variado juego de instrucciones como exponente de una mejor arquitectura. Esta tendencia se basaba en los siguientes hechos:

1. Un amplio juego de instrucciones facilita el diseño de compiladores.
2. Los grandes juegos de instrucciones reducen el tamaño de los programas.
3. La microprogramación y el precio decreciente de las memorias, permiten grandes juegos de instrucciones a precio reducido.
4. Como las microinstrucciones son más rápidas que las instrucciones, el poner muchas funciones en forma de microprogramas hace a la máquina más rápida.
5. El empleo del modelo de ejecución a pila es superior al de registros.

Ejemplos de computadores relativamente modernos, que siguen estos principios son el IBM 360/1168, el VAX 11/780, el Dorado de Xerox y el iAPX 432 de Intel. Véase la tabla de la figura 5.22.

Se denominan CISC (Complex Instruction Set Computers) a las máquinas que propagulan por juegos de instrucciones grandes, frente a los RISC (Reduced Instruction Set Computers) que se inclinan por un juego de instrucciones reducido.

CARACTERÍSTICAS	MODELO	IBM 370/168	VAX 11/780	DURRADO	LAPX 432
AÑO	1975	1976	1978	1982	
NÚMERO DE INSTRUCCIONES	208	305	270	222	
TAMAÑO MEMORIA CONTROL	420 K BITS	480 K BITS	136 K BITS	64 K BITS	
LONGITUD DE INSTRUCCIONES	16 - 48 BITS	16 - 456 BITS	8 - 244 BITS	6 - 321 BITS	
MODELO DE EJECUCIÓN	REG - REG REG - MEM MEM - MEM	REG - REG REG - MEM MEM - MEM	PILA PILA MEM - MEM	PILA PILA MEM - MEM	
TAMAÑO MEMORIA CACHE	128 BITS	64 BITS	64 BITS	...	

Fig. 5-21. Características de algunos computadores que propician juegos de instrucciones extensas.

Los argumentos presentados a favor de la arquitectura RISC son los siguientes:

1. El uso de memorias principales de semiconductores hace que la relación de velocidad de la memoria principal y la UCP sea parecida.
2. La inclusión de la memoria cache mejora la velocidad de la memoria principal, de forma que el tiempo de acceso a esta memoria se hace equivalente al de la Memoria de Control.
3. Convirtiendo la Memoria de Control en memoria rápida, tipo cache, o de almacenamiento temporal de datos, se puede reducir el coste del sistema y/o aumentar su velocidad.
4. El diseño de compiladores es más fácil para juegos de instrucciones reducidos. En primer lugar, porque las nuevas técnicas de optimización de compiladores hacen que, difícilmente se justifiquen instrucciones complejas y, en segundo lugar, porque un juego de instrucciones sencillo y ortogonal simplifica el diseño, al limitar los casos a considerar.
5. El coste de programar una función es mucho menor que el de microprogramarla. Además, su modificación es más sencilla.

5.22. EL ENSAMBLADOR

El lenguaje máquina puro es el binario, que, además de ser ininteligible para el ser humano tiene una estructura larga y propensa a fáciles equivocaciones de un bit por otro. Aunque la representación del lenguaje máquina en hexadecimal acorta y diferencia los dígitos más claramente, sigue manteniendo su falta de significado.

2.1.8 / Modos de direccionamiento y repertorio de instrucciones

Con objeto de representar las instrucciones máquina con cierto simbolismo gico, se ha diseñado el *lenguaje Ensamblador*, que identifica a cada instrucción un grupo de letras (3 ó 4) escogidas entre las más significativas de la palabra, en gés, que define la operación que realiza. A este conjunto de letras se llama *nemónico*, porque "recuerdan" la función de la instrucción.

Ejemplos

Del juego de instrucciones correspondiente al microprocesador 6502, se citan los siguientes nemáticos:

INSTRUCCIÓN

"Load Accumulator" (Cargar el Acumulador)	LDA
"Compare X" (Comparar a X)	CPX
"Increment X" (Incrementar X)	INX
"Branch not equal" (Bifurcación si no es igual)	BNE

El Decodificador de Instrucciones de la Unidad de Control de la UCP no es capaz de interpretar directamente los nemáticos del lenguaje Ensamblador, sino únicamente los códigos binarios correspondientes. Se llama *programa Ensamblador* al software de traducir el *programa fuente*, escrito en lenguaje Ensamblador, a código máquina, dando lugar al *programa objeto*, que ya es directamente ejecutable por la máquina. Figura 5-22.



Fig. 5-22. El programa Ensamblador reduce las instrucciones (nemáticos) del programa fuente, escrito en lenguaje Ensamblador, a código máquina dando lugar al programa objeto.

Las instrucciones del lenguaje Ensamblador están formadas por nemáticos, símbolos, que expresan datos, direcciones y comandos de ayuda. El propio compilador se encarga de traducir a código máquina dichas instrucciones, utilizando el programa Ensamblador.

Las principales características del lenguaje Ensamblador son:

1. Expresa las instrucciones mediante nemáticos.
2. Utiliza símbolos, a base de caracteres alfanuméricos, para expresar datos y direcciones.

3. Además, dispone de una serie de "directivos" o "comandos" que reservan e inicializan posiciones de memoria, controlan la entrada y salida del Ensamblador y gobernan el contador de posiciones, haciendo mucho más sencilla la programación.

4. Se pueden expresar los valores numéricos en diferentes sistemas de numeración.

La sentencia fuente en Ensamblador consta de tres campos:

- **Etiquetas:** Es un símbolo que se antepone a las instrucciones para poder referenciarlas posteriormente.

- **Código de operación y operandos:** Contiene al nemónico y a los operandos que participan en la instrucción. También puede usarse este campo para colocar directivos.

- **Comentarios:** Son anotaciones aclaratorias que no procesa la máquina y cuya misión es la de hacer más comprensible la estructura del programa.

Ejemplo

Un pequeño programa en lenguaje Ensamblador para el microprocesador 6502 es el siguiente:

ETIQUETA	CÓDIGO OPERACIÓN Y OPERANDOS	COMENTARIOS
INIC	LDA #0	A = 0
	LDX #0	Contador a cero
SALTO	STA \$41,X	
	INX	
	CPX #8	Contador es 8?
	BNE SALTO	Si no es 8, bifurca
	JMP INIC	Salto a INIC

Las ventajas del lenguaje Ensamblador son:

- a) Permite una construcción rápida y sencilla de los programas, que son fácilmente interpretados por personas diferentes a las que crearon el programa.

Un inconveniente de este lenguaje (común con todos los lenguajes de alto nivel) es la necesidad de un uso previo del computador, junto al programa traductor (Ensamblador), para transformar el programa fuente en programa objeto.

La actuación del computador cuando traduce un programa fuente, mediante el programa Ensamblador, se lleva a cabo en dos etapas o pasos:

El programa Ensamblador asigna valores a los símbolos empleados para definir datos y direcciones.

Los símbolos, junto a sus correspondientes valores, se guardan en una tabla de memoria para ser usados en el segundo paso.

Segundo paso

El programa Ensamblador genera los códigos objeto en binario, para lo que utiliza los valores de los símbolos definidos en el primer paso.

En esta etapa también se examinan la estructura de las instrucciones y se señalan los errores oportunos en aquellas que hayan sido mal construidas.

EJERCICIOS

Ejercicio 5.1

Una máquina tiene instrucciones de 12 bits y direcciones de 4 bits. Algunas instrucciones tienen una dirección y otras dos. Supongamos que hay 6 instrucciones de 2 direcciones. ¿Cuál es el número máximo de instrucciones de una dirección?

Ejercicio 5.2

Responder verdadero o falso a las siguientes afirmaciones, explicando su decisión.

- a) El análisis de frecuencias estático y dinámico sólo se diferencia al estudiar un programa con bucles y subrutinas.
- b) Una instrucción es más eficiente cuantas más funciones distintas realice.
- c) En el direccionamiento directo relativo se retrasa la ejecución de la instrucción debido a la suma que precisa.
- d) El juego de instrucciones es más potente, única y exclusivamente cuanto mayor número de modos de direccionamiento soporta.
- e) Todas las familias de computadores poseen el mismo lenguaje máquina.
- f) Las instrucciones máquina dependen siempre de la posición que ocupan en memoria.
- g) La "indirección" consiste en que la instrucción no contiene información sobre la situación del operando.

h) El uso de la pila permite realizar llamadas a subrutinas anidadas.

i) Cuantos más formatos existan, más sencilla es la UCF que los interpreta.

j) Las bifurcaciones son las instrucciones más utilizadas en el juego del computador.

Ejercicio 5.3

Sobre la figura 5-3 del libro, indicar 2 instrucciones diferentes para cada modo de direccionamiento posible, del 8086.

Ejercicio 5.4

Explique detalladamente el modo de direccionamiento directo relativo indicando:

- Su funcionamiento.
- Ventajas e inconvenientes.
- Variantes.
- Casos de aplicación de dichas variantes.

Ejercicio 5.5

Indicar las características del formato de las instrucciones, así como las razones que aduce para ello.

Ejercicio 5.6

Indicar el número total de bits que se necesitarán para resolver la siguiente función:

$$X = (A/B + C) (D + E)$$

para máquinas de 0, 1, 2 y 3 direcciones. ¿Qué conclusiones saca?

Notas:

- El código de operación de cada instrucción es de 4 bits.
- Para direccionar un registro se precisan 5 bits.

2.2 / Modos de direccionamiento y repertorio de instrucciones

— Las direcciones de memoria son relativas y se forman con un registro de 16 y un desplazamiento de 8.

Ejercicio 5.7

¿Cómo se realizarían en el microprocesador 8086 las operaciones a nivel de: (Bit clear, bit set y bit test).

Ejercicio 5.8

Indicar las instrucciones del 8086/8088 que se adaptan a la norma IEEEPC

Ejercicio 5.9

El juego de instrucciones de un computador es de 8 instrucciones máquina aplicamos Huffman a dicho juego, obtenemos, según las distintas probabilidades siguiente tabla:

Probabilidad	Huffman
0,4	1
0,15	001
0,15	010
0,1	011
0,1	0000
0,06	00010
0,02	000110
0,02	000111

Se pide:

- a) Calcular el n.º de bits de uso por instrucción.
- b) Si utilizamos un código de extensión de campo de 2 bits como código fijo, 3 para la extensión, construir dicho código suponiendo que el campo de tensión se utiliza para instrucciones de probabilidad $\leq 0,1$.
- c) Comparar el n.º de bits por instrucción en ambos códigos.

Ejercicio 5.10

Sobre el programa dado a continuación, realizar el análisis de las frecuencias de uso según los métodos que conozca. ¿Qué conclusiones deduce de ello?

```

ORA = $0C00          STA CRA
DDRA = $0C00          LDA ORA
CRA = $0C01          STA 200,X
ORB = $0C02          PHP
DDR = $0C02          PLA
CRB = $0C03          ROR
ROR
CRB = $0C03          ROR
ROR
*= $1800             BCC PT1
LDA # $00             LDA # $00
INICIO CLI           STA ORB
LDX # $FF             CL1
TXS                  CPX # $FF
LDA # $0              BEQ PT2
STA CRA
STA CRB
STA DDRA
STA # $FF             PT1
STA DDRB
LDA # %00110110       INX
STA CRA
LDA # %00000111       JMP INIC
STA CEB
LDX # $00             PT2
LDY # $00             BRN
SEI                  SEI
INIC                 JMP INIC
LDA # %00110110       (n.º total de instrucciones 49).

```

Ejercicio 5.11

A partir de un computador IBM 360/370 se desea el diseño del formato de instrucciones sabiendo que:

- Se trata de una máquina de 2 direcciones.
 - El tamaño de la palabra del computador es de 32 bits.
 - Existen 16 registros base.
 - La memoria se encuentra dividida en trozos de 8 M.
 - Se permite la indirección para uno de los operadores y el direccionamiento absoluto para el otro (éstos se encontrará en una de las 256 primeras posiciones de memoria).
- Se pide: ¿Cuántas instrucciones es capaz de soportar?

La Unidad de control



6.1. MISIÓN DE LA UNIDAD DE CONTROL EN EL COMPUTADOR

Un computador digital, tal como se refleja en la figura 6-1, consta de tres bloques fundamentales:

1. UCP o Unidad Central de Proceso, formada por la Unidad de Control y la Unidad Operativa.
2. Memoria principal, donde residen los programas y los datos.
3. Unidad de Entradas y Salidas, que comunica a la máquina con los periféricos exteriores.

Comparando al computador con el ser humano, la UCP puede asemejarse al "cerebro" de la máquina. Mantiene el control general y el envío de información a todos los elementos. La memoria principal hace las veces del "cerebro" y la Unidad de Entrada Y Salida actúa a modo de "extremidades y sentidos", por los que recibe y entrega la información.

Los tres bloques están comunicados entre sí mediante conjuntos de líneas que transportan información binaria del mismo tipo. A dichos conjuntos se les denomina colectores, aunque en el lenguaje técnico se les conoce por buses.

La Unidad de Control se encarga de determinar la dirección de la memoria o de la Unidad de Entrada Y Salida en la que se realiza el acceso o transferencia de información. Para esta selección utiliza un colector de líneas digitales, que recibe el nombre de *bus de direcciones*. Si este bus está formado por n líneas, se tendrá acceso a 2^n posiciones diferentes.

Una vez elegidos los elementos de trabajo, hace falta otro conjunto de líneas para transferir la información. A este grupo de líneas se le llama *bus de datos*. Así

como el bus de direcciones es de carácter unidireccional porque su contenido sólo parte de la Unidad de Control; el bus de datos es bidireccional y tristado, pues está compartido por todos los elementos del computador. La Unidad de Control establece cuál es el elemento emisor y cuál actúa de receptor. El número de líneas del bus de datos define la palabra de trabajo del computador.

Finalmente, hay un colector de líneas que transporta señales auxiliares de gobieno y sincronización conocido como *bus de control*. Transporta los impulsos reflejados, las señales que indican si la operación es lectura o escritura, las de habilitación de buffers y registros, las de inicialización, etc.

A continuación, se hace una somera descripción de los bloques básicos del computador.

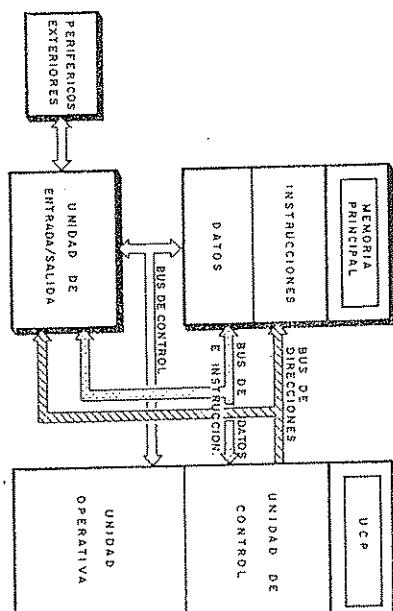


Fig. 6-1. Representación de los bloques fundamentales del computador interconectados a través de los buses.

6.1.1. La memoria principal

Los códigos binarios de las instrucciones que ha de ejecutar la UCP, así como datos o informaciones digitales a procesar o ya procesadas, residen en la memoria principal. No obstante, pueden existir memorias auxiliares de discos o cintas, por ejemplo, que contengan otros programas y datos o partes de un programa que ese momento no se están procesando.

Las memorias principales de los computadores son, generalmente, de semiconductores y están fabricadas en forma de circuitos integrados. Hay dos grandes grupos:



RAM: Memorias de lectura y escritura de carácter volátil, que suelen guardar, de forma temporal, los datos que entran o salen de la ejecución del programa de trabajo.

ROM: Sólo se pueden leer y son no volátiles. Guardan los códigos de las microinstrucciones que conforman las instrucciones.

En la figura 6-2 se muestra la constitución interna de una memoria y su conexión con los buses del sistema.

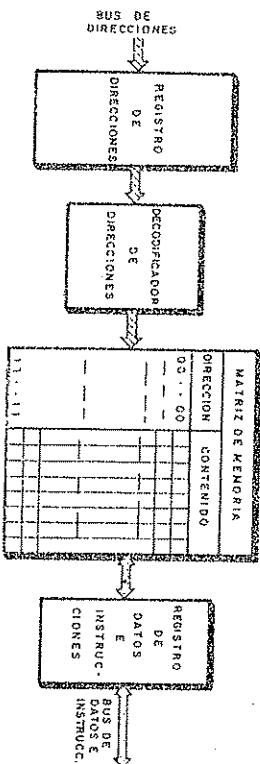


Fig. 6-2. Estructura interna de la memoria principal de un computador y su conexión con los buses del sistema.

Cada vez que la Unidad de Control envía por el bus de direcciones una que corresponda a la memoria, ésta la registra, la decodifica y, finalmente, selecciona una posición en la que se graba o se lee la información que proviene o sale por el bus de datos a través del Registro de datos e instrucciones de la figura 6-2. Las memorias ROM sólo pueden leerse.

6.1.2. La unidad operativa

Su constitución y comportamiento es similar al de una Unidad Lógico Aritmética (ALU). Es la sección encargada de efectuar una serie de operaciones, que soportan a la mayoría de las instrucciones del computador. Actúa combinadamente con una serie de registros y controla el Registro de Estado, que se compone de varios bistables, que funcionan como señalizadores, avisando de ciertas peculiaridades del resultado cuando se realiza una operación en la ALU. Figura 6-3.

A veces, el registro que contiene uno de los operandos que se introducen a la ALU, también actúa como depositario del resultado, en cuyo caso suele recibir el nombre de *Acumulador*.

228 / La unidad de control

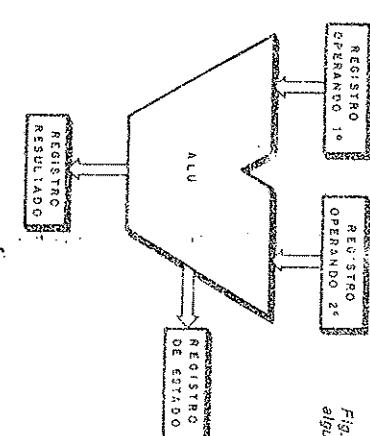


Fig. 6-3. La ALU en combinación con algunos registros es la base de la Unidad Operativa del computador.

6.1.3. La unidad de control

La misión prioritaria de esta sección es interpretar y controlar la ejecución de las instrucciones recibidas desde la memoria principal. De acuerdo con la figura 6-4, la Unidad de Control recibe, a través del bus de datos, el código binario de la instrucción en curso, que lo registra convenientemente. Después, el Decodificador de instrucciones selecciona las posiciones de una memoria ROM, llamada *Memoria de Control*, que corresponden a dicha instrucción y en las que se hallan grabados los códigos de las señales que controlan cada uno de los pasos elementales en que se descompone la instrucción y que reciben la denominación de *microinstrucciones*.

Por último, el *Secuenciador*, se encarga de sacar y distribuir a los elementos del sistema las correspondientes señales de control de cada microinstrucción y, de esta manera, ejecutar ordenadamente la instrucción en curso. Figura 6-4.

Dentro de la Unidad de Control se encuentra el *Contador de Programa* (PC), encargado de enviar por el bus de direcciones la posición de la memoria donde se encuentra la siguiente instrucción. Aunque normalmente este contador se incrementa en cuanto la memoria principal acepta la dirección anterior, existen instrucciones de bifurcación que permiten variar su contenido de forma diferente, dando lugar a rupturas de programa y toma de decisiones de acuerdo con los resultados que se van obteniendo.

Además de recoger las instrucciones de la memoria principal, interpretarlas y ejecutarlas, la Unidad de Control resuelve las situaciones anómalas o de conflicto que pueden ocurrir en el computador.

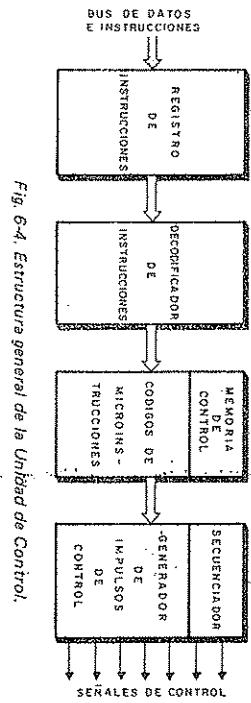
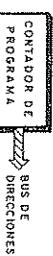


Fig. 6-4. Estructura general de la Unidad de Control.

La información que utiliza la Unidad de Control para llevar a cabo su cometido es la siguiente:

- Instrucción.
- Registro de estado con sus señalizadores.
- Contador de períodos.
- Señales de E/S.

El código de operación (OP) de la instrucción indica a la Unidad de Control la operación que se debe ejecutar y el modo de direccionamiento utilizado. Luego se localizan los operandos, se mandan a la Unidad Aritmética, en su caso, y se almacena el resultado.

El registro de estado contiene información sobre el resultado de la operación anterior y de posibles situaciones anómalas o especiales, tales como desbordamiento, interrupciones, errores de paridad, etc., que exigen una acción determinada por parte de la Unidad de Control. En general, la información sobre el estado del procesador se usa para hacer rupturas condicionales en la secuencia del programa, bien indicadas mediante instrucciones, o bien, originadas por interrupciones externas o situaciones de error.

Las señales de Entrada y Salida permiten el diálogo con los periféricos, tema que se expondrá en un capítulo posterior.

6.2. OPERACIONES ELEMENTALES Y MICROINSTRUCCIONES

La ejecución de cada instrucción se descompone en una serie ordenada de pequeños pasos, que reciben el nombre de *operaciones elementales*. Ejemplos de esas operaciones son:

- Lectura de un operando.
- Incremento del Contador de Programa.
- Ejecución de una operación aritmética.
- Suma de la base más el desplazamiento para hallar el valor efectivo de la dirección de la memoria.

La ejecución de cada operación elemental requiere la activación de un conjunto de señales de control, que genera la Unidad de Control a través de su Secuenciador.

En cada ciclo máquina, generado por un reloj, el Secuenciador envía una serie de señales de control que realizan una o varias operaciones elementales, que configuran una *microinstrucción*. Una instrucción consta de varias microinstrucciones y cuando se realiza la última de una instrucción, la Unidad de Control se prepara para recibir el código OP de la siguiente instrucción del programa y comenzar la ejecución de sus respectivas microinstrucciones.

Las operaciones elementales que puede hacer todo sistema digital, se clasifican en dos grupos:

1. Operaciones de transferencia.
2. Operaciones de proceso.

En las operaciones de transferencia se comienza seleccionando los elementos cuyas posiciones de memoria o registros) uno de los cuales actúa como *origen* y el otro como *destino*. Luego se establece un camino de comunicación entre las salidas del origen y las entradas del destino. Finalmente, se envía al destino una señal para que se "cargue" con la información que existe en su entrada.

En la figura 6-5 se presenta a un colector de líneas de comunicación o bus al que están conectados tres registros A, B y C. Una operación elemental de transferencia entre los registros A y C exige establecer un camino, "abiriendo" las salidas del registro A al bus, lo que se consigue activando la señal de control TA. Debido a la configuración del bus, las salidas de los registros serán triestado y la activación de TA origina la desaparición del estado flotante en las salidas del registro A y la transferencia de su contenido al bus. Una vez estabilizada la información en el bus Y, tanto, en las entradas de los registros conectados a él, se envía la señal de "carga" al elemento destino, que, en este ejemplo, consiste en la activación de la señal Tc del registro C.

6

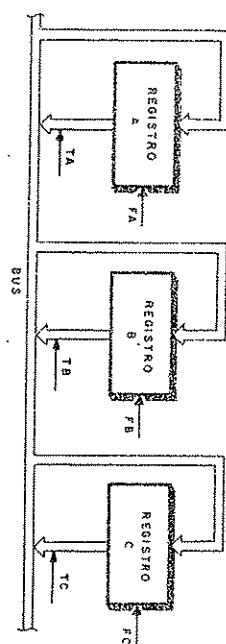


Fig. 6-5. Ejemplo de realización de una operación elemental de transferencia entre los registros interconectados por un bus.

La figura 6-5 muestra el diagrama de las señales de control utilizadas en esta operación elemental de transferencia. La flecha de la señal FC define el flanco de activación.

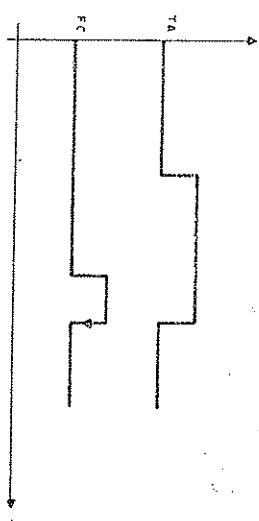


Fig. 6-6. Activación de las señales TA Y FC, que origina la transferencia del contenido del registro A al C de la figura 6-5.

Las operaciones elementales de proceso tienen un planteamiento básico idéntico a las de transferencia. La diferencia fundamental es que la información origen se hace pasar por un operador en su camino hacia el destino. En caso de que la operación sea diádica, los dos orígenes u operandos confluyen en el operador que produce el resultado.

En la figura 6-7 se muestra un operador de suma elemental de los operandos que recibe por los multiplexores MUX1 y MUX2, los cuales reciben los contenidos de los registros A, B, C y D. El resultado del operador puede cargarse en cualquiera de los registros, activando su señal de carga correspondiente.

En la figura 6-8 se presenta el cronograma de las señales que se requieren para efectuar la suma $A := A + D$.

232 / La unidad de control

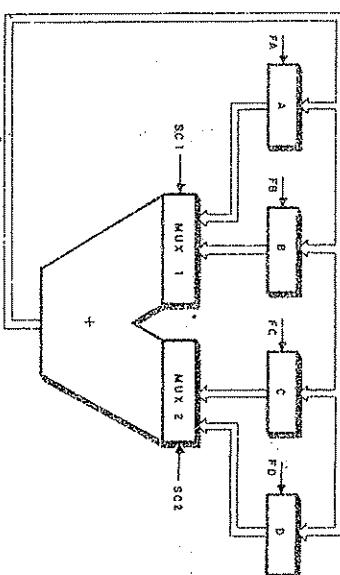


Fig. 6-7. El operador elemental de suma recibe los dos operandos desde los registros A, B, C y D y deposita el resultado en uno de ellos.

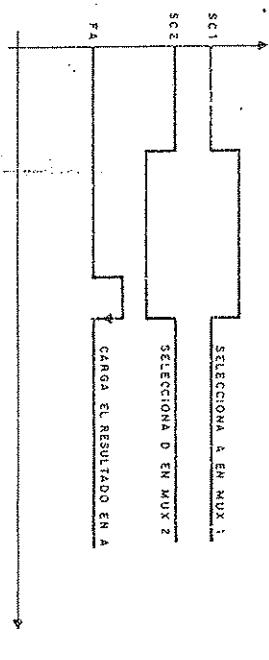


Fig. 6-8. Cronograma de las señales requeridas para realizar la suma $A := A + D$ en el circuito mostrado en la figura 6-7.

6.3. SEÑALES DE CONTROL EN UN COMPUTADOR ELEMENTAL

En la figura 6-9 se presenta una máquina elemental, que sigue la estructura propuesta por von Neumann; en la que se especifican las señales de control de cada elemento. En base a este sencillo computador se analizarán las instrucciones máquinas y se deducirán los cronogramas correspondientes.

Se describen, seguidamente, cada uno de los bloques principales del computador de la figura 6-9 y se indica la función de las señales de control que les gobiernan.

La unidad de control / 233

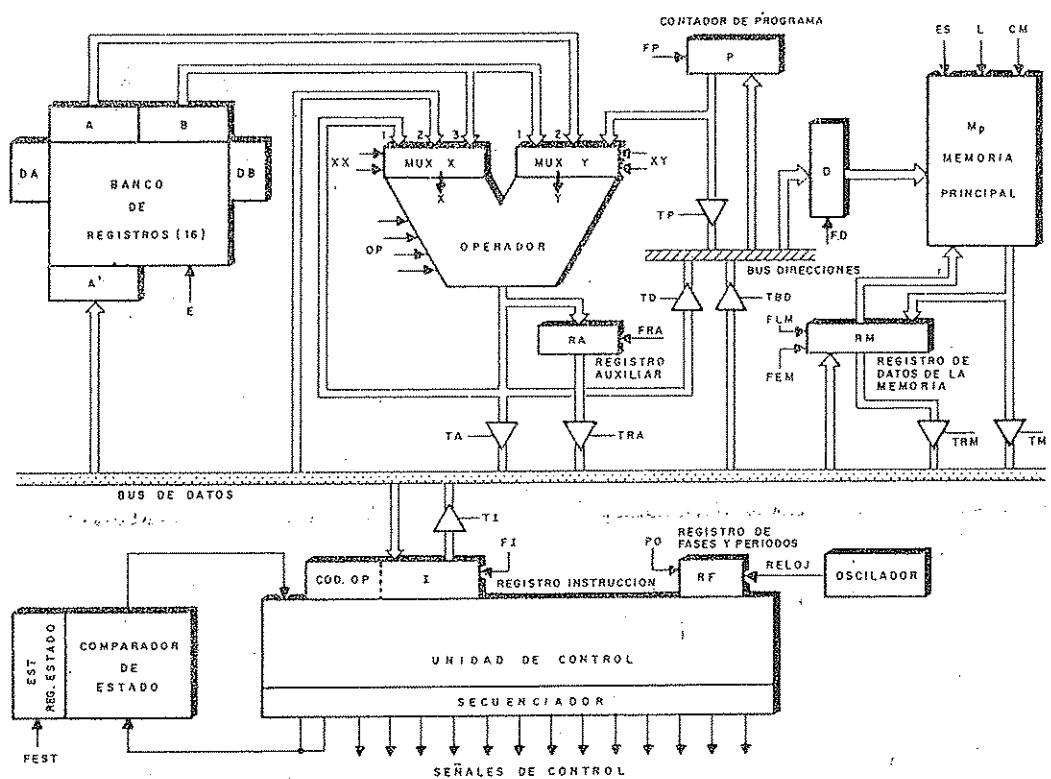


Fig. 6-9. Computador elemental, con estructura von Neumann, en el que se indican las señales de control de todos sus elementos.

6.3.1. Memoria principal

La memoria principal Mp lleva asociados dos registros auxiliares, el de direcciones D y el de datos RM. En la figura 6-10 se muestran los cronogramas de los ciclos de lectura y escritura en los que participan las siguientes señales:

CM: Señal que inicia un ciclo de memoria.

L: Señal que especifica un ciclo de lectura.

ES: Señal que sirve para especificar un ciclo de escritura.

FD: Señal de flanco, que "carga" en el registro de direcciones D, la información disponible en el bus de direcciones.

TM: Señal de activación triestado, que conecta la salida de la memoria al bus de datos.

FLM: Señal que "carga" en el registro RM la salida de la memoria. Se emplea al final del ciclo de lectura.

FEM: Señal que "carga" en el registro RM la información existente en el bus de datos. Es precisa su utilización cuando se procede a llevar a cabo una lectura.

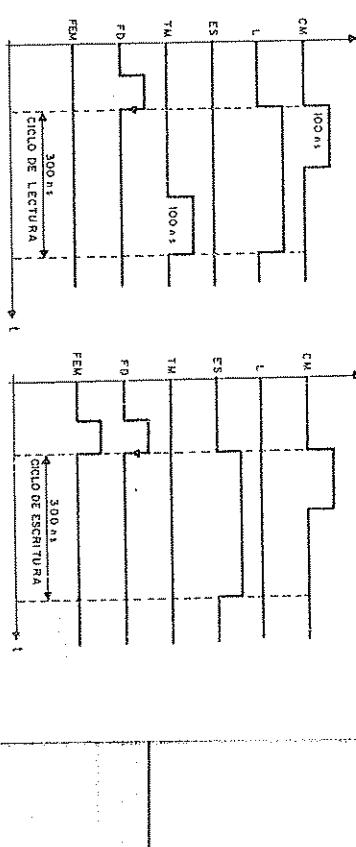


Fig. 6-10. Cronogramas de los ciclos de lectura y escritura de la memoria principal del computador elemental.

Si, tanto el ciclo de lectura como el de escritura de la figura 6-10 requieren 300 ns, las señales L y ES han de tener una duración de 300 ns. Por otro lado, las señales CM y TM serán de 100 ns, que es el período básico que se considera en el computador.

En el ciclo de lectura, la salida de la memoria se ha "abierto" al bus de datos en el último período, mediante la señal TM. De esta forma, la información leída está disponible al final del ciclo en el bus de datos, pudiéndose almacenar en cualquiera de los registros que éste tiene conectados.

En el ciclo de escritura hay que almacenar, previamente, el dato en el registro RW, lo que se consigue con la señal FEM.

En ambos ciclos, la señal FD introduce, antes del comienzo de los ciclos, la dirección de memoria a la que hay que acceder.

6.3.2. Unidad aritmética

Se alimenta a través de un par de multiplexores (MUX X y MUX Y) de tres entradas y una salida. Los multiplexores se gobiernan con las señales XX1 y XX2, que, en forma general, se denominan XX_1 y XX_2 , que se designan por XY. Las entradas 1, 2 y 3 de los multiplexores se seleccionan con las señales XX y XY al tomar los valores 00, 01 y 11, respectivamente.

Al basarse la Unidad Aritmética en la conocida ALU modelo 74181, se requieren cuatro señales de control (OP) para seleccionar una de las 16 operaciones posibles, en cada uno de los modos de trabajo, lógico y aritmético.

La ALU dispone de un registro auxiliar RA, que se carga con la señal FRA. El registro RA es transparente al usuario, lo que significa que no se puede direccionar en las instrucciones máquina, puesto que sólo sirve para almacenar resultados intermedios.

La salida de la ALU se conecta a tres elementos:

1. Entrada 1 del multiplexor X.
2. Bus de datos, con la señal TA.
3. Bus de direcciones, con la señal TD.

La ALU no almacena datos. Sólo establece un camino entre uno o dos orígenes y el destino, por cuyo motivo sus señales de control XX, XY, OP, TA y TD son activas por nivel y sirven para establecer el camino deseado entre el origen y el destino, por lo que han de permanecer estables durante todo el tiempo que dure la operación elemental de proceso.

6.3.3. Banco de registros

Consta de 16 registros, que tienen dos puertas A y B de salida y una A' de entrada. Mediante las direcciones DA y DB de 4 bits se pueden leer dos registros a la vez, cuyos contenidos se obtienen en las salidas A y B. Además, las entradas del registro

de direcciones DA quedan conectadas a la entrada A' por lo que puede cargarse con nueva información procedente del bus de datos, activando la señal de flanko E. El contenido de DA no se modifica hasta que se activa E, independientemente de la información en A.

Dado que las direcciones de los registros empleados forman parte de las instrucciones, la Unidad de Control deberá recoger los bits correspondientes del código de la instrucción, de acuerdo con su formato, y encaminarlos a DA o DB.

6.3.4. Unidad de Control

Junto a la Unidad de Control, propiamente dicha, que contiene al Secuenciador que genera las señales de control del sistema en cada microinstrucción, se necesitan 4 registros de propósito específico, que se describen someramente.

1. *Contador de Programa (P)*. Con la señal FP se "carga" nueva información en este registro. Además, la señal triestado TP de activación, pasa el contenido de P al bus de direcciones. Finalmente, P también está conectado a la entrada 3 del multiplexor Y.

2. *Registro de Instrucciones (I)*. Se carga con el flanko de la señal FI. Puesto que los operandos inmediatos y los desplazamientos, en los direccionamientos relativos, hay que enviarlos a la Unidad Aritmética, se precisa la señal triestadio de activación TI que, además, realiza la expansión de signo adecuada para ajustar estas informaciones al ancho de la palabra de la Unidad Aritmética.

3. *Registro de Estado (EST)*. Almacena, entre otras cosas, las señales de estado generadas por la Unidad Aritmética. La "carga" se efectúa mediante varias señales de flanko FEST, que originan una carga selectiva, dependiendo de la instrucción ejecutada.

4. *Registro de fases y períodos (RF)*. Se trata de un registro contador que se va incrementando con los impulsos de un oscilador o reloj maestro. Permite dirigir los correspondientes períodos y fases de las instrucciones. La señal PO pone a este registro a cero, iniciando la cuenta de los períodos en cada instrucción. Cada período corresponde a una microinstrucción.

6.4. TEMPORIZACION DE LAS SEÑALES DE CONTROL.

PERIODOS Y FASES

La mayoría de los computadores poseen un funcionamiento sincrónico, gobernado por un oscilador o reloj general. Los flancos de este reloj representan la temporización básica del sistema, puesto que determinan el menor tiempo que puede durar una operación elemental.

Se llama *período* a la duración de un tiempo elemental determinado por el reloj maestro. En el computador básico que se describe, el reloj trabaja a una frecuencia de 10 MHz, lo que equivale a un período de 100 ns.

La ejecución de una instrucción máquina, de tipo general, se divide en 4 fases:

1. Lectura del código de la instrucción. Fase de búsqueda o fetch.
2. Lectura de los operandos y decodificación de la instrucción.
3. Ejecución de la operación.
4. Almacenamiento del resultado.

Evidentemente, no todas las instrucciones tienen estas 4 fases perfectamente diferenciadas, pero todas las instrucciones se pueden formar con una o varias de ellas. Así, una suma tiene las 4 fases diferenciadas, mientras que una bifurcación cuenta solamente con la primera y la segunda, que sirven para determinar la dirección de la bifurcación.

Cada fase puede necesitar uno o varios períodos, como se muestra en la figura 6-11.

El objetivo de dividir las instrucciones en fases reside en alcanzar una sistematización de su tratamiento para simplificar el diseño de la Unidad de Control.

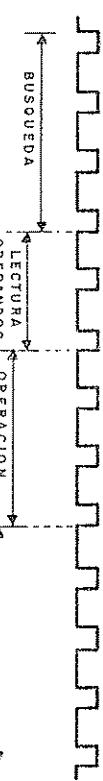


Fig. 6-11. Las fases de una instrucción pueden necesitar uno o varios períodos.

6.5. CRONOGRAMAS EN LA EJECUCION DE INSTRUCCIONES

Se pasa a desarrollar las operaciones elementales que conforman algunas instrucciones y se construyen los cronogramas correspondientes de las señales de control.

6.5.1. Instrucción ADD 4,7

Se trata de una sencilla instrucción de suma en modelo de ejecución Registro-Registro. Suma los registros R4 Y R7, depositando el resultado en R4. El formato de esta instrucción se muestra en la figura 6-12.

238 / La unidad de control

15	8-7	4-5	0
CÓDIGO OPERACIÓN	OPERANDO 1 ^a	OPERANDO 2 ^b	

Fig. 6-12. Formato de la instrucción ADD 4,7 que ocupa 16 bits.

La ejecución de la instrucción consta de las siguientes operaciones elementales:

- 1) $D := P$
Transferencia del contenido del Contador de Programa P al Registro D.
- 2) $I := M(D)$
Ejecución de un ciclo de lectura en la memoria principal, "cargando" el dato en el registro I.
- 3) Decodificación de la instrucción leída.
- 4) $R4 := R4 + R7$
Realización de la suma, para lo que se establecen los siguientes caminos:
a) Conexión de R4 a la puerta Y del operador, vía la puerta B del banco de registros.
b) Conexión de R7 a la puerta X del operador, vía la puerta A del banco de registros.
c) Activación de las señales OP con el código correspondiente a la $X + Y$.
d) Conexión de la salida del operador a la entrada A' del banco de registros.
- 5) Finalmente, se debe preparar la ejecución de la siguiente instrucción.

En la figura 6-13 se ofrece el cronograma con el que se efectúan las 5 operaciones elementales de esta instrucción, que se pasan a comentar:

1) $D := P$

Para cargar el registro de Direcciones con el contenido del Contador de anteriores (-1) se debe activar la señal de flanco FD.

2) $I := M(D)$

Para realizar un ciclo de lectura en la memoria principal se necesitan períodos (300 ns), que son los períodos 0, 1 y 2. Se emplean las señales de control CM y L. Como el resultado de la lectura hay que almacenarlo Registro de instrucciones I, se activa la señal TM en el período 2, abriendo memoria al bus de datos. También se activa la señal FI, que carga el contenido de bus de datos en I.

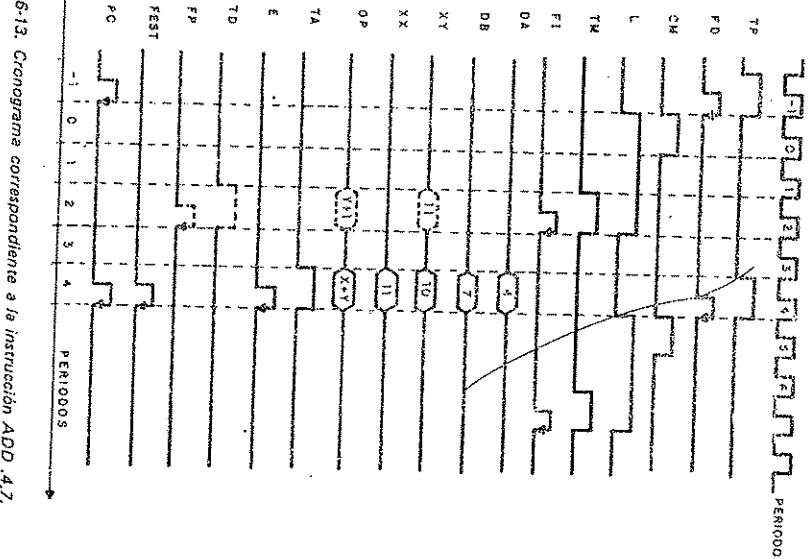


Fig. 6-13.

3) La decodificación de la instrucción supone el reconocimiento de su código de operación. Desde el punto de vista del cronograma, esto significa que no se

pueden generar señales de control, correspondientes a la instrucción $l \leftarrow d$, hasta un cierto tiempo después de estar almacenada en el registro l . Este tiempo viene determinado por los retardos internos de la Unidad de Control, que en este ejemplo se supone que es de un período (período 3).

4) $R_4 = R_4 + R_7$

Para conectar R_4 a la entrada Y del operador hay que introducir por los 4 bits del Registro de Direcciones DA del banco de registros, la dirección de R_4 que es 0100. Además, las dos señales X_Y deberán tomar el valor 10 para seleccionar la entrada 2 del multiplexor Y (figura 6-9). De forma similar, para conectar R_7 a la entrada X , hay que hacer que $DB = 0111$ y $XX = 11$, con el fin de seleccionar la entrada 3 del multiplexor X . La Unidad de Control encaminará

los 4 bits del formato de la instrucción que representa a R_4 (bits 4 al 7 en la figura 6-12) a la entrada DA y los bits 0 al 3, a la entrada DB del banco de registros. Luego se debe activar el operador con el código O_P de la suma. La salida del operador se abre al bus de datos con la señal T_A , pudiéndose acceder a la entrada A' del banco de registros, donde se encuentra el destino R_4 . Todas estas señales se activan en el período 4, en cuyo final, la señal de flanco E almacena el resultado de la suma en R_4 .

El cronograma de la figura 6-13 tiene realizado a trazo continuo el incremento del Contador de Programa, operación elemental que se realiza en el ciclo 2, restando el contenido del registro P a través de la Unidad Aritmética, que funciona como simple incrementador. Las señales que establecen este camino son: $XY = 11$, $O_P = Y + 1$ y T_D . La señal de carga F_D almacena el valor incrementado, nuevamente, en P .

Observese que el incremento del Contador de Programa puede hacerse en cualquier ciclo posterior a "-1". Lo normal es hacerlo en el período muerto de la fase de búsqueda, pero lo más tarde posible para solapar al máximo el final de la instrucción anterior con el principio de la instrucción en curso.

Al final de la instrucción, se carga el Registro de Estado con la activación de $FEST$ y se pone a cero el Registro de Fases con P_O .

6.6. DISEÑO DE LA UNIDAD DE CONTROL

De la figura 6-9 se deduce que la Unidad de Control parte del código de operación contenido en el Registro de Instrucciones I , así como del período determinado por el contador RF . Con estas informaciones, a las que ocasionalmente se debe añadir la salida de "comparador de estado", quedan definidas las señales de control que se han de activar.

El diseño de la Unidad de Control exige una especificación previa de todas las instrucciones máquina que deberá interpretar, que puede hacerse a nivel de cronograma, a nivel de operaciones elementales, o, de una forma equivalente, mediante un lenguaje simbólico, que exprese, con todo detalle, las operaciones a realizar y su secuenciamiento.

Toda instrucción se encarga de iniciar la siguiente.

La duración de las señales viene determinada por el reloj, debiéndose acomodar a un número entero de períodos.

La Unidad de Control es un mero traductor o circuito combinatorial que convierte al código O_P más el período en las señales de control especificadas en el control.

Para tener una idea de la dimensión y complejidad que reviste el diseño de la Unidad de Control, basta con tener en cuenta que un modelo sencillo con códigos

de operación de 8 bits y cuyas instrucciones pueden alcanzar hasta 32 períodos y 16 como media, precisa de un registro RF de 5 bits. Este pequeño computador tendrá del orden de 150 señales de control, por lo que se trata de diseñar un circuito con 14 señales de entrada y 150 de salida, con unas 4.000 combinaciones de excitación diferentes.

A todo ello, hay que añadir los análisis y ajustes de retardos en los distintos caminos internos de la Unidad de Control.

Hay dos métodos para diseñar y construir esta Unidad:

- Mediante *lógica cableada*.
- Mediante una memoria.

6.7. UNIDAD DE CONTROL EN LOGICA CABLEADA

La Unidad de Control cableada se construye mediante puertas lógicas y se diseña siguiendo alguno de los métodos clásicos de diseño lógico.

La ventaja principal de este tipo de diseño es que dota al circuito de una gran rapidez, consiguiendo una velocidad de funcionamiento mayor que usando memoria. Sin embargo, dada la complejidad del circuito, su diseño y puesta a punto son muy costosos y laboriosos. Además, es muy difícil realizar modificaciones posteriores.

Este tipo de Unidad de Control se usa en computadores muy potentes.

6.8. UNIDAD DE CONTROL MICROPROGRAMADA

En este caso se emplea una memoria para almacenar la información de las señales de control de los períodos de cada instrucción. Para generar un cronograma es suficiente ir leyendo las distintas palabras de esta memoria, que recibe el nombre de *Memoria de Control*.

Como a la información grabada, correspondiente a las señales de control de un período, se la llama *microinstrucción*, la Unidades de Control con este diseño reciben el apelativo de *microprogramadas*.

El ámbito de aplicación de las Unidades de Control microprogramadas es, típicamente, el de los computadores de tamaño medio. En los muy pequeños, la estructura de microprogramación es demasiado compleja y no resulta económica, mientras que, para los grandes, su funcionamiento es demasiado lento.

La ventaja principal de la Unidad de Control microprogramada es la simplicidad conceptual, junto al hecho de que la información de control reside en una memoria, lo que supone una fácil modificación y ampliación. Además, se pueden incluir instrucciones complejas con muchos períodos, poniendo una memoria de control suficientemente grande.

La micropgramación permite construir computadores capaces de soportar varios juegos de instrucciones, bastando con cambiar el contenido de la memoria control. La idea fue propuesta en 1950 por M. V. Wilkes y otros científicos, y como se refleja en la figura 6-14, la Unidad de Control constaba de dos memorias A y B, fabricadas con matrices de diodos.

En la figura 6-14, las microinstrucciones están almacenadas en la memoria de donde son leídas mediante un árbol de decodificación. La memoria B guía la dirección de la siguiente *microinstrucción*. Las bifurcaciones se producen con señal que selecciona entre dos entradas a la matriz de la memoria B.

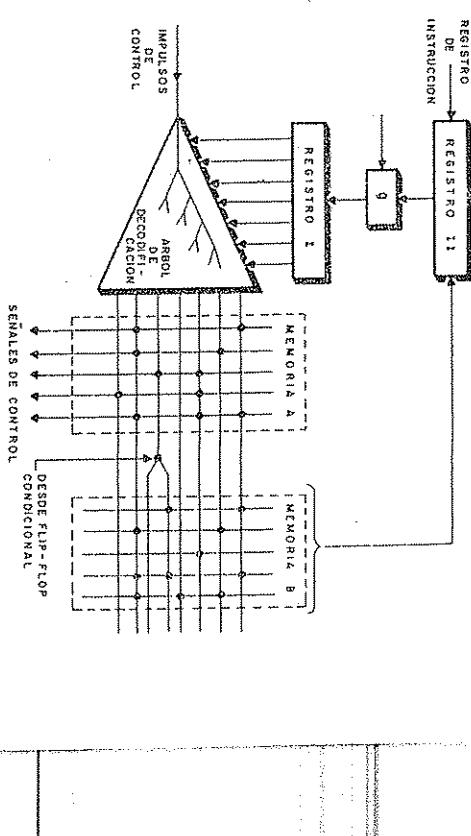


Fig. 6-14. Esquema propuesto por M. V. Wilkes para la Unidad de Control.

Historicamente, la micropgramación se aplicó por primera vez a la gama b. de la familia de computador IBM 360, a mediados de la década de los 60. Por razones de velocidad no se aplicó a la gama alta de dicha familia.

6.8.1. Concepto de micropgrama

Un micropgrama es una cadena de "unos" y "ceros" que representa los valores de las señales de control durante el conjunto de períodos que componen una instrucción.



Se llama *micropograma* al conjunto de microinstrucciones que conforman el cronograma de una instrucción. Ejecutar un micropograma consiste en ir leyendo cada una de las microinstrucciones que lo componen, enviando la información leída a los elementos del computador como señales de control. Lacadencia de lectura a la controlada por el reloj principal de la máquina.

A título de ejemplo, en la figura 6-15 se muestra un micropograma con las 5 instrucciones que constituyen la instrucción ADD .4,7, comentada.

SEÑALES DE CONTROL	MICROINSTRUCCIONES			
	i_0	i_1	i_2	i_3
T_P	0	0	0	0
F_D	0	0	0	1
C_M	1	0	0	0
I	1	1	1	0
T_M	0	1	1	0
F_L	0	0	1	0
O_A	0	0	0	0
D_A1	0	0	0	0
D_A2	0	0	0	0
D_A3	0	0	0	0
D_A4	0	0	0	0
D_B1	0	0	0	0
D_B2	0	0	0	0
D_B3	0	0	0	0
X_{T1}	0	0	0	0
X_{T2}	0	0	0	0
X_{T3}	0	0	0	0
O_P1	0	0	0	0
O_P2	0	0	1	0
O_P3	0	0	0	0
T_A	0	0	0	0
F_O	0	0	0	0
F_P	0	0	1	0
F_EST	0	0	0	0
P_O	0	0	0	1

Fig. 6-15. Micropograma correspondiente a la instrucción ADD .4,7, que está compuesta por 5 microinstrucciones.

Para establecer las microinstrucciones de la figura 6-15 se ha considerado que el código para el incremento es $OP = 0110$ y el de la suma es $OP = 1011$.

En las microinstrucciones se han tratado de forma similar las señales activas por nivel y las que son activas por flanco. Para obtener una señal de flanco basta con recordar una de nivel con la señal de reloj, lo que se puede hacer, sencillamente, con una puerta AND.

244 / La unidad de control

6.9. CARACTERÍSTICAS DE LA UNIDAD DE CONTROL MICROPROGRAMADA

Para que una Unidad de Control microprogramada pueda llevar a cabo su cometido, ha de cumplir tres condiciones básicas:

1. Disponer de una memoria de control con capacidad suficiente para almacenar los micropogramas de todas las instrucciones.
2. Tener un procedimiento para hacer corresponder a cada instrucción de máquina su micropograma. Esto significa que debe convertir cada código de operación de la instrucción en la dirección de la memoria de control donde comienza su micropograma.

3. Poser un mecanismo para ir leyendo las sucesivas microinstrucciones del micropograma en curso y bifurcar a un nuevo micropograma cuando finaliza el que se está ejecutando.

Para resolver las condiciones 2. y 3 existen dos alternativas:

- a) Con secuenciamiento explícito.
- b) Con secuenciamiento implícito.

6.9.1. Secuenciamiento explícito

Este tipo, seguido por Wilkes, consiste en dotar a cada microinstrucción de la dirección de la siguiente. Como las microinstrucciones de un micropograma pueden estar desordenadas, la 2.ª condición se cumple sin más que reservar las primeras posiciones de la memoria de control para guardar la primera microinstrucción de cada instrucción, y justamente en la dirección expresada por su código de operación. No se necesita mecanismo alguno de secuenciamiento, puesto que cada microinstrucción suministra la dirección de la siguiente en la memoria de control.

El encadenamiento con una nueva instrucción se resuelve mediante una señal de control que toma como dirección de la siguiente microinstrucción el nuevo código de operación.

Este tipo de secuenciamiento tiene el inconveniente de utilizar una gran cantidad de memoria de control. Por ejemplo, una memoria de control de 4 K palabras, requiere 12 bits para su direccionamiento, lo que supone destinar 4 K X 12 bits para direcciones.

En la figura 6-16 se ofrece el esquema de una Unidad de Control con secuenciamento explícito. El oscilador genera la señal de "carga" en el registro de microinstrucciones, estableciendo el período básico de la máquina. La parte de dirección se realimenta a la memoria de control para iniciar la lectura de la siguiente microinstrucción. El multiplexor de direcciones permite seleccionar entre el código de

operación de una nueva instrucción o la dirección de la microinstrucción. Este multiplexor, gobernado por una señal de control, se encarga de que la última microinstrucción de un microprograma enganche con la primera microinstrucción del siguiente.

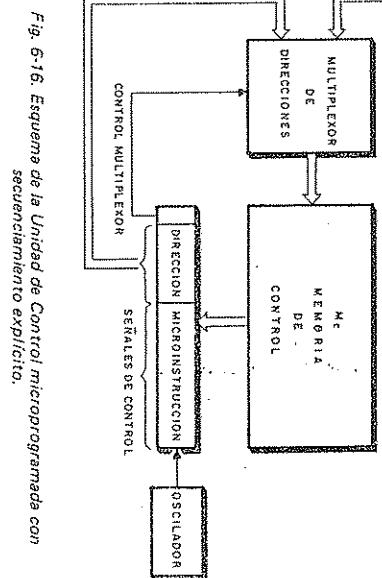
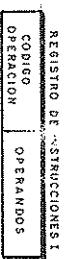


Fig. 6-16. Esquema de la Unidad de Control microprogramada con secuenciamiento explícito.

6.9.2. Secuenciamiento implícito

Con este sistema es necesario tener ordenadas las microinstrucciones de cada microprograma en posiciones consecutivas de la memoria de control. De esta forma, ya no es preciso que cada microinstrucción contenga la dirección de la siguiente, obteniendo un considerable ahorro de memoria.

Para cumplir la condición que resuelve la correspondencia a cada instrucción máquina con su microprograma, se introduce una etapa traductora o decodificadora entre el código de operación y el multiplexor de direcciones que se utilizaba en el esquema de la figura 6-16. Este traductor es una pequeña memoria ROM o PLA, que, cuando el código de operación es de 8 bits y la memoria de control tiene una capacidad de 4 K palabras, el tamaño de este elemento es de 256 palabras de 12 bits.

El mecanismo de lectura consecutiva de microinstrucciones y bifurcación a un nuevo microprograma, al terminar el que se halla en curso de ejecución, se implementa con un registro de direcciones y un incrementador.

246 / La unidad de control

La figura 6-17 presenta el esquema general de la Unidad de Control microprogramada con secuenciamiento explícito. La temporización principal la proporciona el oscilador, que genera una señal periódica empleada en cargar el registro de direcciones y el de microinstrucciones. El encadenamiento de microprogramas es similar al del caso anterior. Una señal de control activa al multiplexor, de forma que viene de la ROM o PLA, como dirección siguiente.



Fig. 6-17. Unidad de Control microprogramada con secuenciamiento explícito.

6.10. FORMATO DE LAS MICROINSTRUCCIONES

El concepto básico de microinstrucción aparece como un conjunto de bits, uno para cada señal de control. Esta solución, conceptualmente sencilla, no es, sin embargo, muy empleada por su coste. En general, cada microinstrucción realiza una operación elemental en cada período, lo que da lugar a que su formato sea lleno de "ceros".

Para reducir la longitud de las microinstrucciones se las codifica de manera que tengan menos bits que señales de control. A veces, se las asigna una duración de un período.

La unidad de control / 246



Se llaman *microinstrucciones horizontales*, cuando las microinstrucciones no están codificadas. Por el contrario, recibe el nombre de *microprogramación vertical* cuando las microinstrucciones están muy codificadas. Según el grado de codificación, existen microprogramaciones más o menos horizontales o verticales.

Las microinstrucciones horizontales tienen un formato muy largo, pero permiten utilizar los elementos del computador en paralelo. Las microinstrucciones verticales tienen formatos cortos, pero impiden el paralelismo.

Las señales de control que gobernan un mismo elemento de la máquina se agrupan formando un campo de microinstrucciones. Por ejemplo, las señales tristado que controlan el acceso a un bus se agrupan formando un campo; las señales de gobierno de la Unidad Aritmética, o el banco de registros, o la memoria, formarán sus campos respectivos.

Para acortar el tamaño de las microinstrucciones se codifican todos o algunos de los campos distintos, en el caso de no usar codificación se necesitarían 13 señales para gobernar dicho bus, pero, dado que de estas 13 señales sólo puede estar activa una en cada período, se puede emplear un campo de 4 bits, que, mediante un sencillo decodificador 4×16 , hace posible conectar al bus hasta 16 elementos distintos.

La figura 6-18 ofrece un esquema ilustrativo sobre la *codificación de campos* en una microinstrucción, cuyo principal inconveniente es que requiere una etapa decodificadora con su correspondiente coste y retardo.

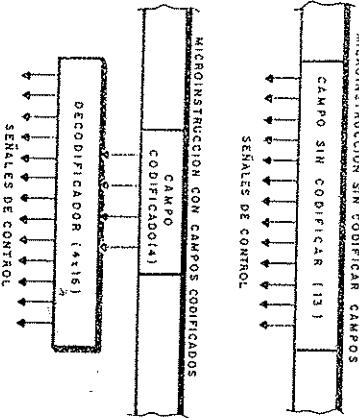


Fig. 6-18. Una técnica habitual para conseguir acortar el formato de las microinstrucciones es la codificación de campos.

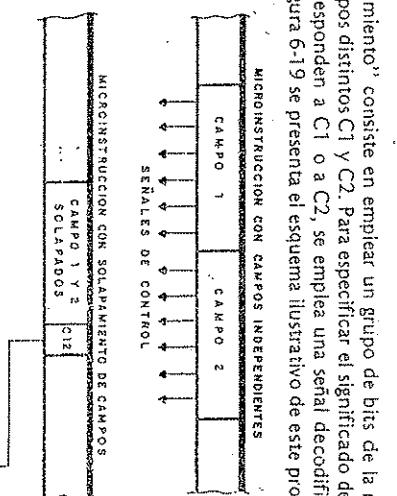


Fig. 6-19. Para el acortamiento en la longitud de las microinstrucciones se solapan dos campos C1 y C2. Un decodificador, con una señal auxiliar C12, determina en cada caso, el campo al que corresponde la información presente.

La alternativa más drástica para reducir el tamaño de las microinstrucciones es realizar una *codificación total* de la misma. Se parte de la definición de todas las microinstrucciones que componen el repertorio de instrucciones Y, seguidamente, se codifican con el menor número de bits. Este procedimiento precisa de un decodificador complejo y con un retardo muy grande, pero reporta un ahorro importante de la capacidad de la memoria de control.

En la figura 6-20 se muestra el esquema de la Unidad de Control con codificación total, en la que las informaciones contenidas en la memoria de control no son más que las direcciones para extraer de la memoria auxiliar la correspondiente microinstrucción.

Finalmente, cabe presentar dos técnicas usadas en ocasiones especiales:

- Microinstrucciones de varios períodos*, en cuyo caso la microinstrucción contiene información suficiente para realizar las órdenes elementales de varios períodos. Se precisa un contador de períodos y una información sobre la duración de cada microinstrucción, contenida en ella misma.

Para acortar la longitud de las microinstrucciones también se usa el *solapamiento de campos*, que se basa en el hecho de que, normalmente, en cada período sólo están activas unas pocas señales de control. Es más, con frecuencia existen señales exclusivas, o sea, que no se pueden activar simultáneamente.

El "solapamiento" consiste en emplear un grupo de bits de la microinstrucción para dos campos distintos C1 y C2. Para especificar el significado de los bits y determinar si corresponden a C1 o a C2, se emplea una señal decodificadora adicional C12. En la figura 6-19 se presenta el esquema ilustrativo de este procedimiento.



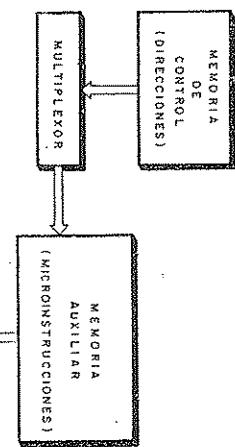


Fig. 6-20. En la "codificación total", la memoria de control contiene las direcciones de la memoria auxiliar donde residen las microinstrucciones.

b) *Nanicontrol*. Cuando se emplea micropogramación vertical y microinstrucciones de varios períodos de duración puede usarse una Unidad Nanoprogramada para la ejecución de las microinstrucciones. En este caso existirían dos niveles de micropogramación.

6.11. ESTRUCTURA COMPLETA DE LA UNIDAD DE CONTROL MICROPROGRAMADA

Las estructuras contempladas de la Unidad de Control micropogramada con secuenciamiento explícito o implícito, sólo pueden ejecutar microprogramas lineales, pero no permiten la ruptura de la secuencia del micropograma.

Las instrucciones de bifurcación condicional presentan dos cronogramas alternativos a partir del punto en el que se hace la comprobación de la condición de bifurcación. Los microprogramas han de ofrecer una microbifurcación condicional para seguir la rama o alternativa deseada. Hay que añadir a la arquitectura básica de la Unidad de Control un mecanismo de *microbifurcación condicional*.

También es recomendable añadir a la Unidad de Control mecanismos para establecer *microbucleos* y *llamadas a microsubrutinas*, teniendo en cuenta que muchas instrucciones tienen partes comunes y hay algunas con operaciones que se repiten, como las de "desplazamiento múltiple", las de "multiplicación" y las de "división".

6.11.1. Microbifurcaciones condicionales

La bifurcación condicional exige que la microinstrucción de bifurcación pueda elegir entre dos direcciones para poder seguir uno de los dos caminos posibles.

En el caso de secuenciamiento explícito, la microinstrucción de bifurcación debería contener las dos direcciones completas, pero, como esto sería muy costoso, se acostumbra a diferenciar ambas direcciones en un solo bit, que toma el valor 0, resultado del comparador de la condición de bifurcación.

En el secuenciamiento implícito se ha de poder elegir entre la microinstrucción siguiente y otra distinta. Por tanto, la microinstrucción de bifurcación habrá de contener dicha dirección. Se suele usar el solapamiento de campos. Es factible solo par el campo de dirección de la microinstrucción de bifurcación con otros de señales de Entrada/Salida, de memoria, etc.

En la figura 6-21 se ofrece una Unidad de Control en la que se han incluido elementos precisos para realizar microbifurcaciones condicionales. Obsérvese que el resultado del comparador se emplea en la selección del multiplexor de dirección, permitiendo elegir entre la dirección anterior incrementada o la contenida en el campo solapado C4 de la propia microinstrucción. El campo C2 establece la condición de bifurcación, por lo que sirve de entrada al comparador. El bit C3 se encarga de "desolapar" el campo C4, empleando un demultiplexor de dos salidas, que decide si este último se comporta como dirección de salto o como señales de control.

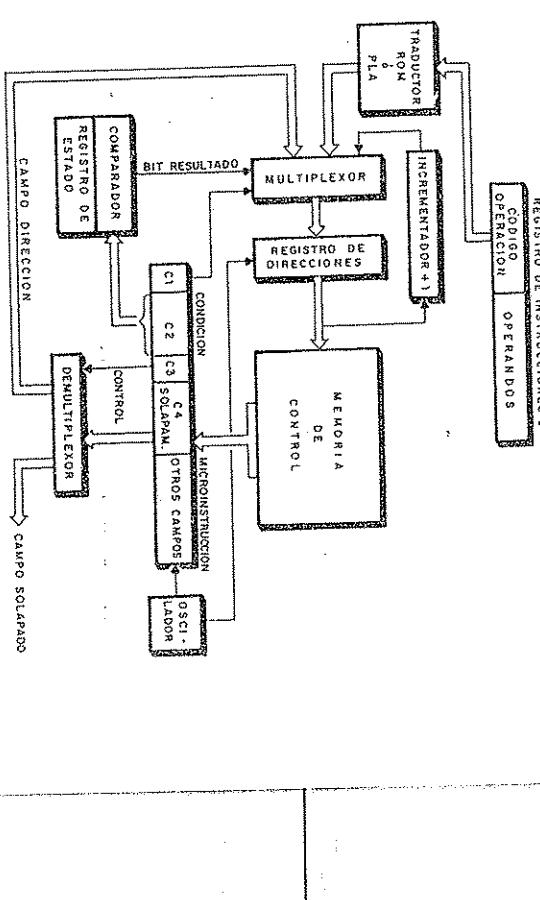


Fig. 6-21. Esquema de la Unidad de Control con capacidad para realizar microbifurcaciones.

6.11.2. Microbucleos y microsubrutinas

La realización de *bucles* exige una bifurcación condicional, más un contador con autodetención, que se emplea como condición de bifurcación. Dicho contador debe ser accesible desde la microinstrucción, para poder inicializarla con el número de veces que se desea repetir el microbucle.

La *llamada o subrutina* necesita un almacenamiento para "salvaguardar" la dirección de retorno. La solución más usual consiste en "colgar" del registro D una *pila*, en la que se guardan estas direcciones de retorno.

6.12. EMPLEO DE VARIOS RELOJES

La utilización de dos o más señales de reloj posibilita la generación de distintas operaciones sucesivas en el mismo período.

En la figura 6-22 se muestran las distintas señales que se obtienen de dos relojes T_1 y T_2 desfasados $1/4$ de período, empleando solamente puertas lógicas.

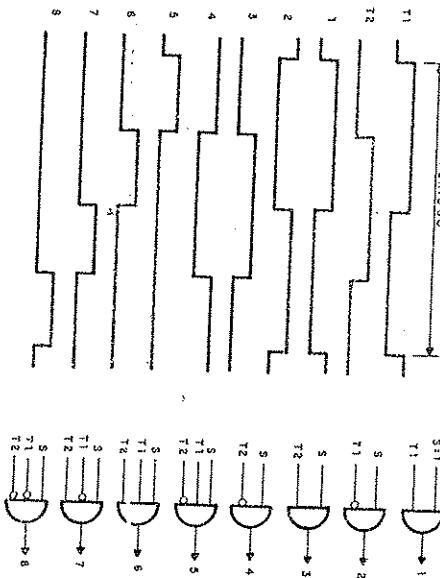


Fig. 6-22. Con dos relojes T_1 y T_2 , desfasados un cuarto de período y utilizando diversas puertas lógicas, se pueden obtener distintas formas de onda.

6.13. INTERRUPCIONES Y EXCEPCIONES O CERPOS

La Unidad de Control estudiada hasta el momento ejecuta, de manera inamovible, las instrucciones sucesivas y contiguas de un programa. Solamente cuando se encuentra una bifurcación se rompe esta secuencia y se salta a ejecutar en otra zona de memoria.

Existen situaciones en las que interesa detener un programa y saltar a otro. Esto ocurre por ejemplo:

1. Cuando aparece un error de ejecución, tal como un desbordamiento del resultado en una operación aritmética.
2. Cuando surge un error de paridad en la lectura de memoria, o un intento de ejecutar una instrucción con código de operación prohibido, o un intento de acceder a una zona de memoria protegida.
3. Cuando un periférico requiere la atención de la UCP.

En estos casos ha de producirse una ruptura de secuencia no programada, puesto que no se produce por una instrucción de bifurcación del programa en ejecución.

El mecanismo para producir estas rupturas de secuencia no programadas es similar al usado en las bifurcaciones condicionales. En efecto, todas las condiciones que pueden producir esta ruptura de secuencia automática se reflejan en sendos bistables de estado. Además, todas las instrucciones llevan al final de su cronograma una bifurcación condicional implícita sobre el valor de dichos bistables. Cuando uno de ellos se activa, se para la ejecución del programa en curso y se bifurca.

Cuando la causa de esta bifurcación no programada proviene del exterior de la UCP, se dice que se ha producido una *interrupción*. Por ejemplo, una unidad de disco interrumpe a la UCP cuando termina una operación de entrada y salida.

Cuando la causa de la bifurcación no programada es de origen interno de la UCP, se habla de una *excepción o cecho (trap)*.

La forma en que se genera la dirección de la bifurcación en estos casos se basa en el salto a una posición prefijada o cableada, lo cual es frecuente en el caso de las excepciones. Con las interrupciones, suele ser habitual que el propio periférico peticionario de la interrupción proporcione la dirección de la bifurcación. En ambos casos, dicha dirección corresponde a un programa del sistema operativo, que trata la excepción o la interrupción y sustituye el control del programa que se ha interrumpido.

APÉNDICE

Diseño de una UCP con circuitos integrados SSI y MSI

6A.1. LA ORIENTACION DIDACTICA DEL PROYECTO

El objetivo del diseño de esta *Unidad Central de Proceso* (UCP) ha sido el procurar un acercamiento "natural" al funcionamiento del computador para los conocedores de la Electrónica digital. Teniendo en cuenta esta finalidad, se han elegido para su construcción circuitos integrados clásicos y sencillos que han determinado las características fundamentales de la UCP y, por cuyo motivo, no está recomendada su aplicación a caso real alguno. Las hay mejores, integradas en un solo chip, bajo la denominación de *microprocesadores*.

Siguiendo la idea propuesta por von Neumann, la máquina universal o computadora se compone de tres bloques fundamentales que se presentan en la figura A6-1, y que se hallan interconectados entre sí, a través de unos colectores de líneas, llamados buses.

La UCP, se encarga de direccionar la memoria mediante el bus de direcciones y obtener las instrucciones y los datos por el bus de datos. El procesamiento de la información lo realiza en la Unidad de Control con el apoyo de los registros de trabajo y de acuerdo con el significado de las instrucciones que provienen del programa.

La Unidad de Controles es la encargada de decodificar las instrucciones y enviar las diversas señales de gobierno a todos los elementos que intervienen en una operación a través del bus de control. También es función de la Unidad de Control cargar el bus de direcciones con el valor de la posición de memoria o de Entrada/Salida que se quiere leer o escribir.

Por último, la Unidad de Entrada/Salida procura una adaptación de la información que entra y sale de la máquina desde/a los periféricos exteriores.

Todos las secciones que componen la UCP, desde 1971, se han podido reunir en una sola pastilla de circuito integrado a la que se denomina microprocesador.

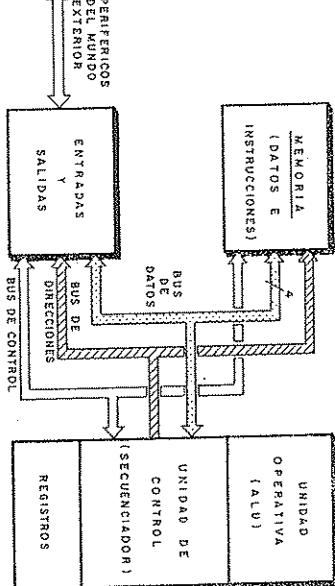


Fig. A6-1. Estructura básica de un computador.

6A.2. FUNCIONAMIENTO BASICO

Para facilitar la comprensión del funcionamiento de la UCP se ha diseñado un modelo, cuyo correcto funcionamiento ha sido ampliamente comprobado, utilizando circuitos integrados de pequeña y mediana escala de integración y que, además, son muy populares.

Las señales que controlan la operatividad de los circuitos integrados de la UCP se destinan a implementar la instrucción en curso de ejecución. Sin embargo, la complejidad de una instrucción conlleva su ejecución en "pasos elementales" definidos por un *reloj*. Las partes elementales de una instrucción reciben el nombre de *microinstrucciones*, las cuales son conjuntos de 1 y 0 que se dirigen a las patitas de entradas de control de los circuitos integrados.

Como quiera que la máquina que admite una UCP definida siempre las mismas, también lo serán las microinstrucciones; por lo tanto, los grupos de bits que conforman las microinstrucciones se graban de forma permanente en memorias no volátiles, tipo ROM, que, en este caso, se llaman *Memoria de Control*. De aquí surge el concepto de *Secuenciador*, que no es más que un mecanismo que se encarga de ir sacando ordenadamente de la Memoria de Control las microinstrucciones correspondientes a la instrucción en ejecución. Figura A6-2.

Las señales de control que genera el Secuenciador actúan, entre otros, sobre la Unidad Operativa que suele ser una Unidad Lógico-Aritmética (ALU). Las instrucciones que admite la UCP están muy ligadas a las que puede efectuar la ALU.

En este diseño se ha elegido la conocida ALU 74181, que maneja operandos de 4 bits, con lo que queda definido el tamaño de la *palabra* de trabajo de la UCP y en

6

tamaño del bus de datos. Su esquema de conexiónado se muestra en la figura A6-3 y necesita 4 líneas (S_0-S_3) para la selección de la operación que va a realizar. Además, la señal M selecciona entre las 16 operaciones lógicas o las 16 operaciones aritméticas que puede efectuar esta ALU. $M = 1$, operaciones lógicas y $M = 0$, operaciones aritméticas.

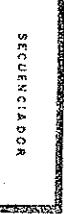


Fig. A6-2. El Secuenciador es un dispositivo encargado de sacar ordenadamente de la Memoria de Control las señales que conforman las microinstrucciones que corresponden a la instrucción en curso.

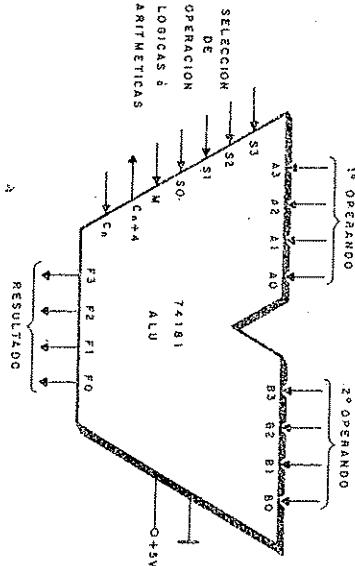


Fig. A6-3. Esquema de conexiónado de la ALU 74181.

La tabla de la figura A6-4 muestra el conjunto de las 32 operaciones de la ALU.

Los 27 circuitos integrados que forman parte de la arquitectura de esta UCP son de tipo estándar y disponibles, generalmente, en los comercios del ramo. De ellos, 6 contienen simples puertas lógicas y los restantes son de uso más específico: ALU, contadores, buffers, bascula-cerrojo, EPROM, etc.

El estudio de la UCP diseñada se desglosa en tres grandes bloques que se presentan por separado:

1. Bloque aritmético-lógico.
2. Contador de Programa.
3. Secuenciador.

Fig. A6-4. Tabla que muestra las 32 operaciones lógicas y aritméticas que puede realizar la ALU 74181.

OPERACIONES	
LÓGICAS	ARITMÉTICAS
$F = \overline{A}$	$F = A \text{ MAS } C \text{ (U: ACARREO)}$
$F = \overline{A} \cdot \overline{B}$	$F = (A + B) \text{ MAS } C$
$F = 0000$	$F = 1111 \text{ MAS } C$
$F = \overline{A} \overline{B}$	$F = A \text{ MAS } A \overline{B} \text{ MAS } C$
$F = \overline{A} + \overline{B}$	$F = (A + B) \text{ MAS } A \overline{B} \text{ MAS } C$
$F = A \oplus \overline{B}$	$F = A \text{ MAS } B \text{ MAS } C$
$F = B$	$F = (A + \overline{B}) \text{ MAS } AB \text{ MAS } C$
$F = A \cdot B$	$F = AB \text{ MENOS } 1 \text{ MAS } C$
$F = 1111$	$F = A \text{ MENOS } A \text{ MAS } C$
$F = A + \overline{B}$	$F = (A + B) \text{ MAS } A \text{ MAS } C$
$F = A + B$	$F = (A + \overline{B}) \text{ MAS } A \text{ MAS } C$
$F = A$	$F = A \text{ MENOS } 1 \text{ MAS } C$

6A.3. BLOQUE ARITMÉTICO-LOGICO

Esta parte de la UCP es la encargada de realizar las posibles operaciones de procesamiento de datos de 4 bits. Se basa en la ALU 74181, que ya se ha explicado.

En la figura A6-5 se muestra el esquema correspondiente.

La ALU dispone de dos entradas de datos de 4 bits: una procede, directamente, del bus interno de datos, mientras que la otra se recibe de una bascula-cerrojo de 4 bits, 74175, formada internamente por 4 flip-flop tipo D. A este cerrojo se le llama Registro A y la información que guarda y entrega a la ALU proviene, también, del bus de datos interno de la UCP. Los flip-flop D del 74175 se cargan mediante

6

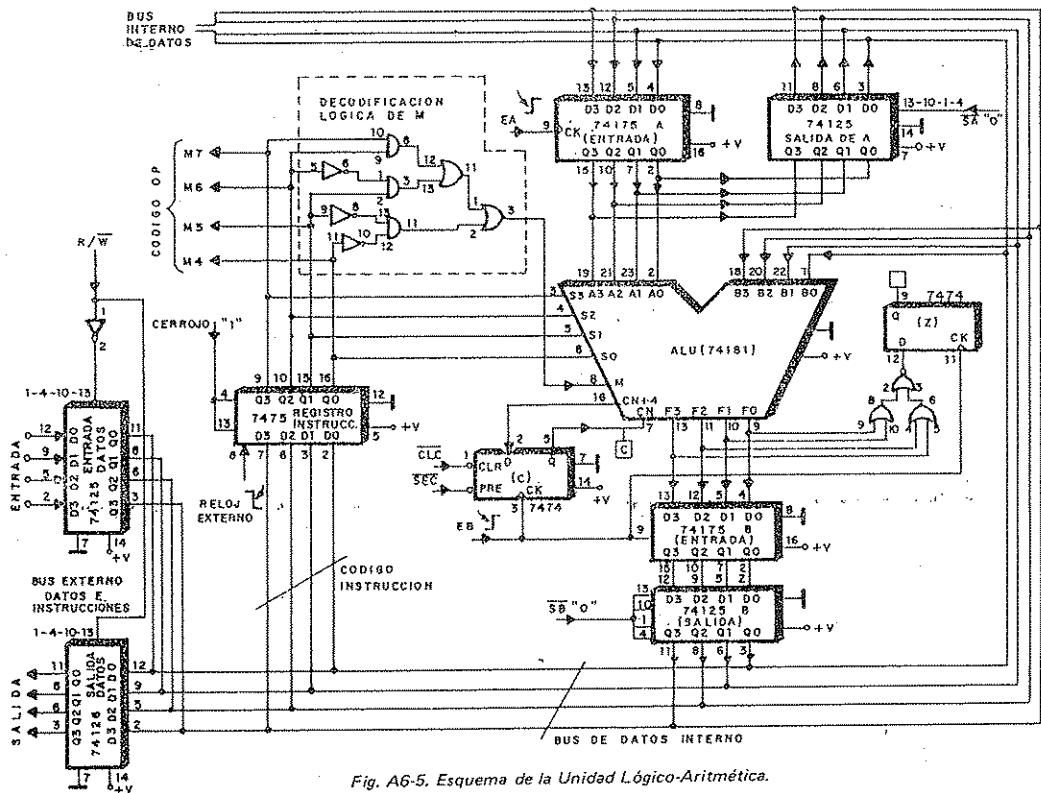


Fig. A6-5. Esquema de la Unidad Lógico-Aritmética.

dante el fianco ascendente que reciben por su patita de entrada de reloj (CK) y que procede de la línea EA controlada por el Secuenciador.

Con objeto de, caso de ser necesario, enviar al bus de datos interno la información del registrador A, la salida del 74175 (también llamado *Entrada A*), se envía a una entrada de la ALU y a un buffer triestado, modelo 74125, que traslada la información del registro A al bus interno de datos, cuando, desde el Secuenciador, se active permanentemente, el 74125 sólo transmite lo que recibe por su entrada al activarse SA.

De forma parecida, la salida de la ALU se guarda en otra bascula-cerrojo (latch) 74175, que recibe el nombre de *Registro B*. Otro buffer tristado 74125 (Salida f) se encarga de trasladar la información del registro B al bus interno de datos, cuando desde el Secuenciador, se activa la señal SB.

Cuando la ALU efectúa una operación, genera un acarreo de salida (C_{n-1}), que representa el señalizador de acarreo C que consiste en un flip-flop que también se emplea para introducir su contenido a la entrada previa de acarreo (C_n), antes de realizarse una operación. El flip-flop de acarreo C , recibe dos señales desde el secuenciador, una sirve para ponerle a 1 (SEC) y la otra, a 0 (CLC). La señal de relajación del flip-flop C se activa con la señal EB (Entrada B), que carga al registro B con el resultado de salida de la ALU.

Un conjunto de 3 puertas lógicas examinan la salida de la ALU y controlan flip-flop D, en el que se almacena el estado del serializador de cero (Z), el cual pone a 1 cuando el resultado de una operación ha sido cero. Ambos serializadores C y Z, están integrados en el mismo circuito integrado 7474.

Finalmente, la ALU necesita recibir 4 señales que seleccionan la operación, mátrix (M) que determina si es de tipo lógico o aritmético. Las señales que seleccionan la operación proceden del código OP de la instrucción que entra desde el bus externo de datos e instrucciones, el cual dispone de dos circuitos integrados 74125. 74126, que contienen 4 buffers de salida triestado y que actúan como entrada de salida de datos, respectivamente. La línea R/W del Secuenciador define si es entrada o salida de datos.

Cuando por el 74125 se recibe un código C_4' , éste se carga en el circuito integrado 7475 (cuádruple cerrojo de flip-flop D), que funciona como un *Registro de Introducción*, y luego se aplica a las líneas de selección de la ALU (S0-S1-S2 y S3). Mediante un circuito decodificador, a base de puertas lógicas, se comprueban las señales M4-M5-M6 y M7 y se obtiene el valor de M que define en la ALU si se trata de una operación lógica o aritmética.



6A.4. EL CONTADOR DE PROGRAMA

Dentro de la Unidad de Control de la UCP, uno de los componentes fundamentales es el PC, cuya misión es proporcionar la dirección de la memoria en donde se encuentra la instrucción que hay que ejecutar. El circuito del PC se ofrece en la figura A6-6.

En la UCP que se describe, el PC puede funcionar de tres maneras posibles, que se describen separadamente para facilitar su comprensión.

1. Forma habitual

En general, el PC se incrementa cada vez que se ejecuta una instrucción. En la UCP que se describe, el Secuenciador incrementa al PC a través de la señal $PC + 1$.

El PC está formado por dos contadores binarios, 74197, en serie, lo que sirve para proporcionar una dirección de 8 bits. Los contadores 74197 son de 4 bits y tienen la posibilidad de ser borrados con la señal CLEAR.

El recuento del PC se carga en un buffer de salida triestado, implementado en el circuito 74244, que se compone de dos conjuntos de 4 líneas, que en este momento actúan simultáneamente. La salida de este buffer, controlado por la señal G, carga el contenido del bus de direcciones.

2. En modo de direccionamiento absoluto

En este caso, el bus de direcciones envía la posición de memoria en la que hay que cargar o extraer un dato, para que, a continuación, contenga el PC en la siguiente posición a la que tenía previamente. Se trata de la aplicación en una instrucción del direccionamiento del operando del modo "absoluto" en el que se proporciona en la instrucción directamente la dirección en donde se encuentra el operando, que hay que leer o escribir.

En este procedimiento, se comienza cargando el Registro de Direcciones, formado por dos básculas 7475 que se encargan de guardar los 4 bits de la parte alta de la dirección (DH) y los 4 de la parte baja (DL). Las básculas 7475 son activadas por dos señales procedentes del Secuenciador y que se denominan DH y DL.

Una vez cargados los 7475, su contenido pasa al buffer de 8 líneas 74241, que es activado por la señal ABS del Secuenciador, que, al mismo tiempo, desconecta al buffer 74244, cuyas salidas quedan en estado flotante.

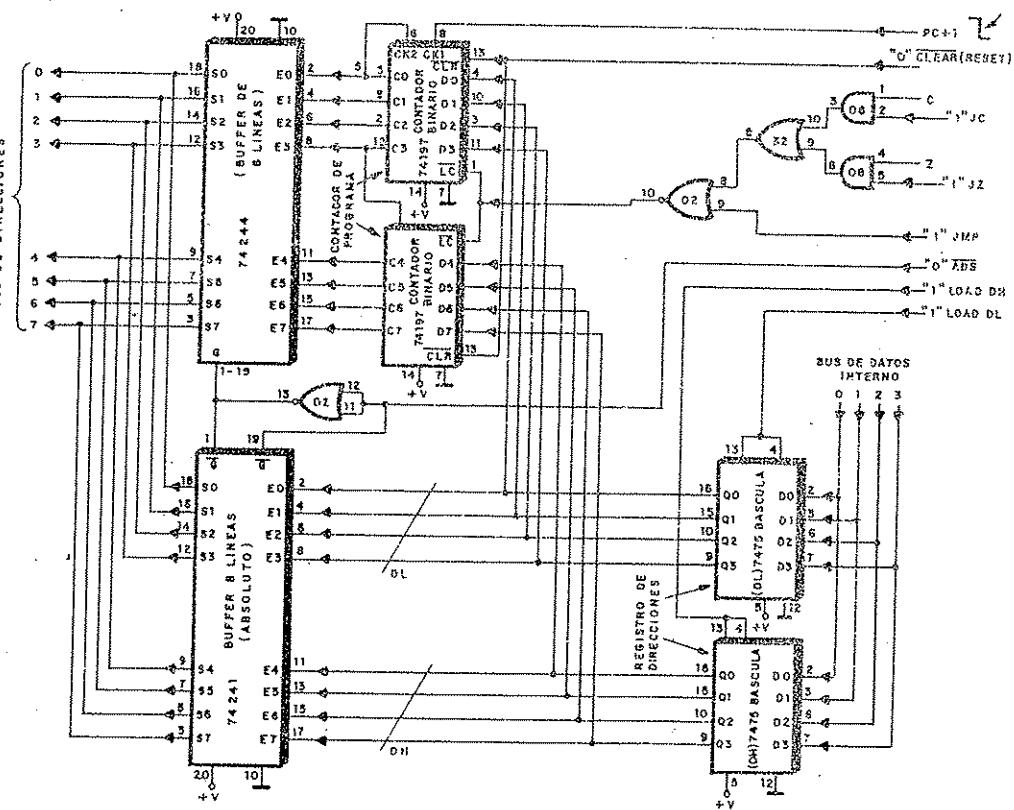


Fig. A6-6. Circuito del Contador de Programa.

3. Saltos

Cuando en un programa se realiza un salto, se pasa a la dirección correspondiente al salto Y, luego, el PC continúa incrementándose a partir de la misma. Este funcionamiento está en contraposición con el modo absoluto, en el que se volvía a la dirección previa del PC.

Dada la sencillez de la UCP, no se le ha provisto de saltos con posibilidad de retorno, en los cuales hay que guardar el contenido del PC, para recuperarlo posteriormente.

Cuando se efectúa un salto, se recibe desde el Secuenciador, la activación de una de las tres señales siguientes: JMP, JC o JZ. Entonces, por medio de un conjunto de puertas lógicas decodificadoras, se activa la señal LC de los contadores 74197 los cuales se cargan con el contenido del Registro de Direcciones (7475 X 2). Luego, el recuento continúa a partir de la dirección cargada en el PC.

6A.5. SECUENCIADOR

Este bloque de la UCP es el encargado del control de todo el sistema a través de sus líneas de salida. Su principal función consiste en recibir el código OP de la instrucción a ejecutar y generar una serie de microinstrucciones o pasos elementales que la realicen. Cada microinstrucción no es más que un conjunto de estados lógicos que salen por las líneas del Secuenciador hasta los elementos que participan en esa acción.

Cada instrucción, según su complejidad, consta de un número variable de microinstrucciones que se ejecutan una detrás de otra. Figura A6-7.

En el diseño que se commenta, todas las microinstrucciones que componen el repertorio de instrucciones de la UCP se han grabado en un par de memorias EPROM, modelo 2716, que tienen una capacidad de 2 K X 8 bits. Estas memorias disponen de 11 líneas para direccionar sus 2 K posiciones, de 8 bits cada una. Como en esta aplicación el número de posiciones a utilizar es inferior a 2 K, sólo se emplean 8 de dichas líneas en cada EPROM, llevando a tierra las 3 restantes.

Las dos memorias EPROM se direccionan en paralelo, con lo que se obtiene un total de 2 bytes de información (un byte de cada pastilla). Los 16 bits obtenidos en cada direccionamiento de la pareja de EPROM, constituyen las señales de salida del Secuenciador, que sirven para controlar todo el sistema. Las salidas de las memorias 2716 pasan por 3 básculas, tipos 74174 y 74175, antes de salir al exterior conformando el bus de control.

Los 8 bits que forman la dirección, que sirve simultáneamente para las dos memorias, están distribuidos en dos conjuntos de 4 bits cada uno. Los 4 bits de más peso de la dirección (A4-A5-A6 y A7) corresponden con los 4 bits del código OP

de la instrucción, que proviene del Registro de Instrucción (7475). Los 4 bits de menor peso de la dirección, proceden del circuito contador 74197, que recibe el nombre de *Contador de Microinstrucciones* (CMI). El CMI se incrementa una unidad con cada impulso de reloj externo. De esta manera, el código OP, fija el valor de los 4 bits de más peso de la dirección de la pareja de memorias (sólo podrá haber un máximo de $2^4 = 16$ instrucciones). Seleccionada, con los 4 bits de más peso, el comienzo de la posición de la instrucción a ejecutar, el CMI se va incrementando, con cada impulso de reloj y va recorriendo hasta 16 posiciones de las memorias, al ir variando el valor de los 4 bits de menor peso de la dirección. En cada posición de memoria hay una microinstrucción Y, por lo tanto, el mayor número de microinstrucciones que puede tener una instrucción, será de 16.

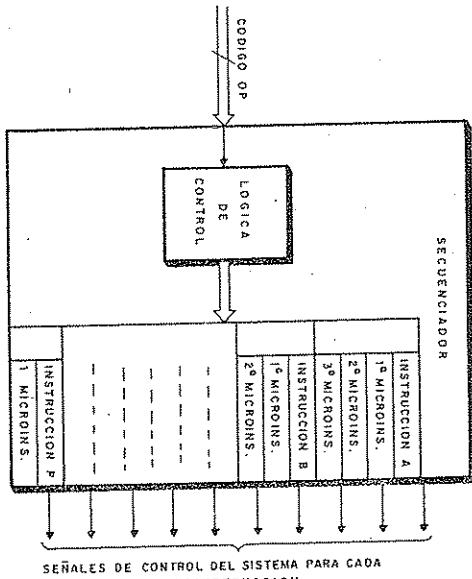


Fig. A6-7. Esquema ilustrativo del comportamiento del Secuenciador.

En resumen, el código OP más el CMI, que se incrementa al ritmo del reloj, van direccionando las dos memorias en paralelo. En cada ciclo de reloj se direcciona una posición de cada memoria Y, así se obtiene una microinstrucción de 16 bits, los cuales salen por las líneas de control del Secuenciador hasta los diferentes elementos.

El CMI se incrementa en los flancos descendentes de reloj, mientras que los básculas de salida de las señales de cada instrucción recogen la información en los flancos ascendentes para retener la información una vez pasado el tiempo de acceso a las memorias.

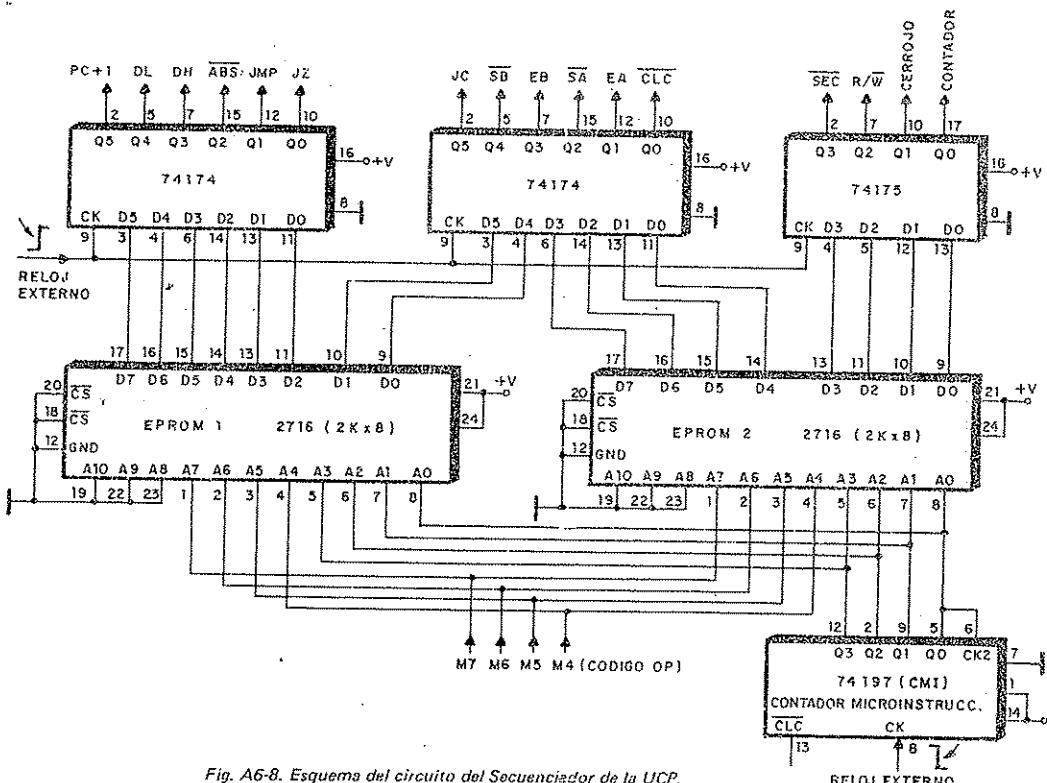


Fig. A6-8. Esquema del circuito del Secuenciador de la UCP.

6A.6. CRONOGRAMAS DE FUNCIONAMIENTO

Para iniciar el funcionamiento de la UCP se ha dispuesto un pulsador de RESET que pone a 0 el PC y así recoge el código OP que existe en una posición concreta de la memoria principal. También pone a 0 el CM1, para formar la dirección completa de la instrucción.

Transcurridos 300 ns después de haberse direccionado la primera instrucción, comienza a funcionar el reloj desde nivel bajo. En su primer flanco ascendente se cargan las basculas de salida de las señales de control y en el flanco descendente se posterior se incrementa el CM1, con lo que se posiciona la siguiente microinstrucción. Cuando se llega a la última microinstrucción, el mismo Secuenciador se encarga de poner a 0 el CM1 y cargar el Registro de Instrucción con el siguiente código OP.

En la figura A6-9 se presentan los cronogramas correspondientes a las señales de control para 3 instrucciones básicas:

SEC: Poner a 1 el señalizador de acarreo C.

LDA: Cargar el Registro A.

STA: Almacenar en memoria el Registro A.

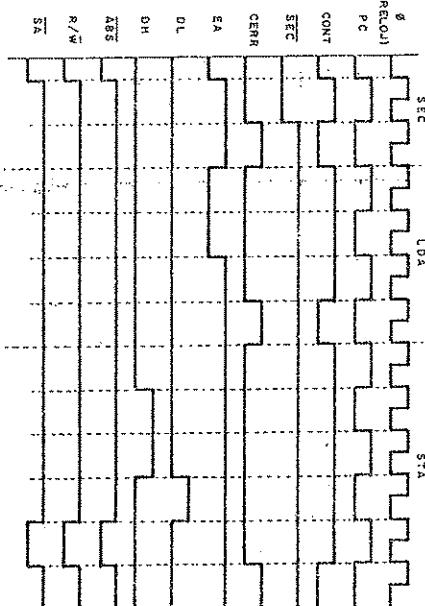


Fig. A6-9. Cronogramas correspondientes a las instrucciones SEC, LDA y STA.



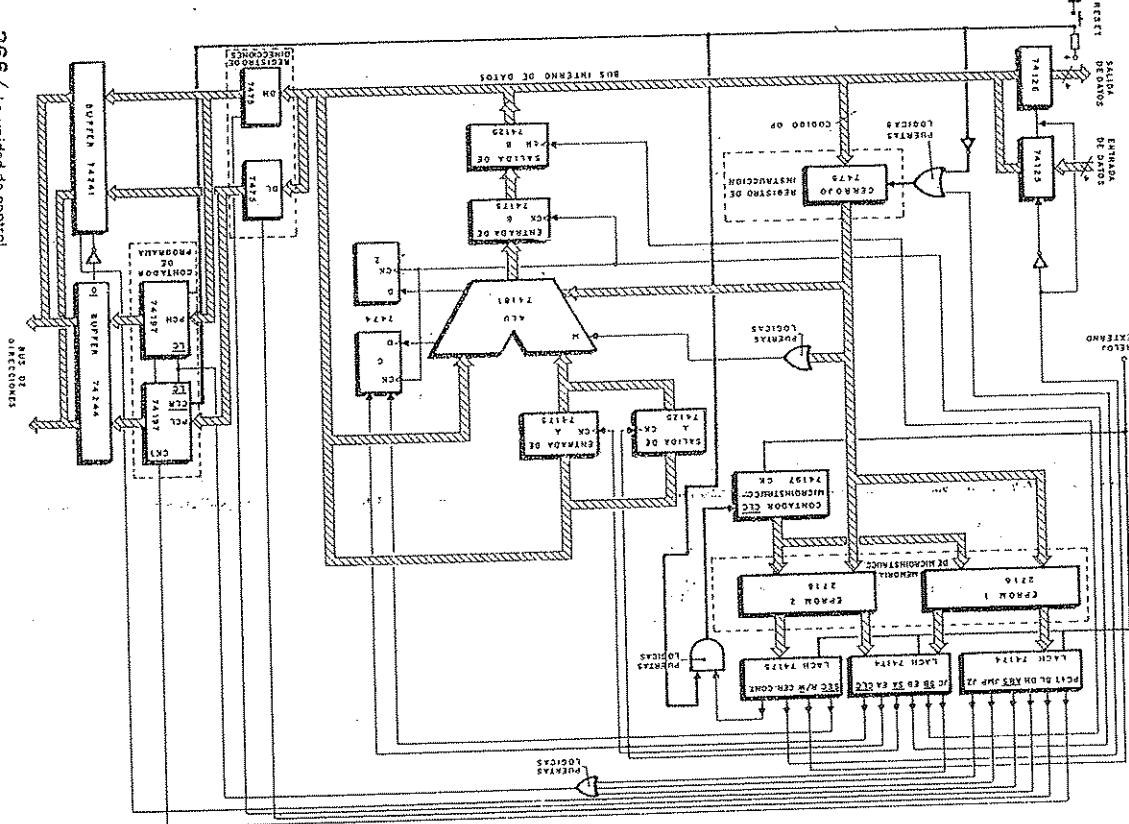


Fig. A6-10. Esquema completo de la UCP.

6A.7. MODOS DE DIRECCIONAMIENTO Y REPERTORIO DE INSTRUCCIONES

Una de las características fundamentales de una UCP es su juego de instrucciones. Muchos de los componentes de la UCP se han incluido para poder implementar instrucciones o microinstrucciones. Por eso, la arquitectura física de un procesador está muy ligada con su sistema lógico, ya desde las operaciones más elementales.

Como ya se ha explicado, los modos de direccionamiento posibles en este dispositivo son las diversas formas de trabajo del Contador de Programa, que son tres:

1. Modo de direccionamiento absoluto.

2. Modo de direccionamiento relativo al PC. Saltos sin retorno.

3. Modo de direccionamiento inmediato.

Respecto al repertorio de instrucciones, ya se ha comentado que estaba restringido a un máximo de 16, puesto que, con los 4 bits del código OP, sólo se podían alcanzar 16 combinaciones distintas. Además, cada instrucción puede constar un máximo de 16 microinstrucciones, puesto que, una vez definida la instrucción con los 4 bits del código OP, que eran los 4 bits de más peso de la pareja de EPROM con el Contador de Microinstrucciones (CMI), iba incrementando los 4 bits de menor peso de la dirección para ir sacando una microinstrucción en cada ciclo de lectura.

Se describen las instrucciones que se han implementado con microinstrucciones grabadas en la pareja de memorias tipo 2716.

6A.7.1. CPL

Esta instrucción complementa el contenido del Registro A en la ALU y el resultado lo deposita en el Registro B.

Consta de dos microinstrucciones:

1.^a **microinstrucción:** El código OP es 0000₂, y actúa sobre la selección de operación de la ALU. El secuenciador activa la señal correspondiente al próximo incremento del PC (PC + 1) y se activa el Registro A.

2.^a **microinstrucción:** Aquí se produce el incremento del PC y se activa la entrada del Registro B (EB) para recibir el resultado del complemento del Registro A que ha realizado la ALU. El cerrillo (CER), que actúa como Registro de Instrucciones, se activa para poder recoger el próximo código OP y la señal del Contador (CMI) pasa a 0, con lo que se borra el CMI para comenzar la exploración de las microinstrucciones de la siguiente instrucción desde 0.



Con esta instrucción, los biesables de estado o señalizadores pueden quedar afectados de la siguiente manera:

- Si $C = 1$, no se modifica sea cual sea el resultado de la operación. Z queda afectado por el resultado.
- Si $C = 0$, este señalizador se comporta inversamente que el Z.

6A.7.2. SEC

Esta instrucción, cuyo código OP es 1_{16} , pone a 1 el señalizador de acarreo C.

Consta de dos microinstrucciones. La primera prepara el incremento del PC y activa la línea SEC. En la segunda, se borra el CMI y se incrementa el PC.

Se recomienda consultar las tablas de las figuras A6.11 y A6.12 donde se han representado los códigos grabados en las memorias EPROM y la secuencia de las microinstrucciones que componen cada una de las instrucciones.

6A.7.3. CLC

Sirve para poner el señalizador C a 0. El código OP es 2_{16} .

En su primera microinstrucción, el Secuenciador prepara el incremento del PC y activa la línea CLC. En la segunda, borra el CMI y se produce el incremento del PC.

6A.7.4. NOP

No opera. Ocupa 4 ciclos de reloj sin afectar a elemento alguno ni realizar operación. Su código OP es 3_{16} .

En sus tres primeras microinstrucciones mantiene preparado el incremento del PC y en la cuarta se lleva a cabo dicho incremento, se borra el CMI y se prepara el cerrojo de instrucciones para poder cargar un nuevo código OP.

6A.7.5. JC

Es una instrucción de salto condicional. Cuando el señalizador de acarreo C está a 1, se produce un salto del PC a la posición de memoria que se especifica en el segundo y tercer operandos que siguen al código OP de la instrucción. El código OP es el 4_{16} .

6A.7.6. JZ

Es un salto condicional a la posición de memoria especificada por el segundo y tercer operandos que siguen al código OP, en el caso de que el señalizador Z = 1. Si Z = 0, el programa sigue la secuencia normal. El código OP es el 5_{16} .

También consta esta instrucción de 7 microinstrucciones como JC y sólo difiere de esta última en la quinta microinstrucción, que, en lugar de activar el señalizador la línea JC, activa la JZ.

6A.7.7. SBC

Con esta instrucción se resta al contenido del Registro A, el contenido de la posición direcciónada (direcciónamiento absoluto) y el acarreo. El resultado de la operación queda almacenado en el Registro B y puede afectar a C y Z. El código OP es el 6_{16} .

Consta de 6 microinstrucciones. En la primera, prepara el incremento del PC. En la segunda, se incrementa el PC y se carga la parte alta del Registro de Direcciones. En la tercera, se prepara el PC y se carga la parte baja del Registro de Direcciones. En la quinta, se prepara al PC y se activa la línea ABS, con lo que la posición direcciónada sale por el bus de direcciones. En la última microinstrucción se incrementa el PC, el contenido del bus de datos pasa a la ALU donde se efectúa la resta y se activa la línea EB, con lo que el resultado queda almacenado en el Registro B. El CMI y el cerrojo o registro de instrucciones quedan preparados para recibir la próxima instrucción.



En la primera microinstrucción se prepara el incremento del PC. En la segunda, se produce el incremento y se carga la parte alta del Registro de Direcciones. En la tercera, se prepara el incremento del PC y se carga la parte baja del Registro de Direcciones. En la quinta, se vuelve a preparar el PC, se activa la línea JC y, si C = 1, se permite la carga del PC con el contenido del Registro de Direcciones que guarda la posición a la que se produce el salto. El sexto ciclo máquina, prepara al PC y sirve para dar tiempo a que la memoria tenga acceso al dato (código OP). En la séptima microinstrucción se incrementa el PC, se borra el CMI y se prepara el cerrojo de instrucciones para recibir la siguiente instrucción.

Si C = 0, no se produce el salto y el programa sigue su secuencia normal.

6A.7.8, JMP

Es un salto incondicional a la posición de memoria indicada en el segundo y

Consta de 7 microinstrucciones o ciclos máquina, que siguen la misma secuencia que en la instrucción JC, a excepción de la quinta microinstrucción en la que el Secuenciador, en vez de activar la línea JC, activa la JMP.

6A.19. LDA

Carga el Registro A con el contenido de la posición de memoria direccionalada por el segundo y tercer operandos que siguen al código OP, es decir, de forma absoluta. Su código es 816.

Consta de 6 microinst.

menta al PC y se carga la parte alta del Registro de Direcciones. En la tercera, se prepara el PC y se sigue cargando la parte alta. En la cuarta, se carga la parte baja del Registro de Direcciones. En la quinta, se prepara el PC, se activa la línea ABSY y se prepara la línea EA (se pasa a 0). La sexta incrementa el PC y el contenido del bus de datos pasa al Registro A. Se incrementa el PC y se preparan el CMI y el cerrojo para la siguiente instrucción.

G.A., I.U. ADU

Se suma el contenido del Registro A con el contenido de la posición dirección dada en la instrucción y el acarreo. El resultado queda almacenado en el Registro B. Y puede afectar el estado de los señalizadores. El código OP es 9₁₆.

Consta de 6 microinstrucciones. En la primera, se prepara el PCC

mento. En la segunda, se incrementa el PC y se carga la parte alta del Registro de Direcciones. En la tercera, se prepara el PC y se sigue cargando la parte alta. En la cuarta, se carga la parte baja del Registro de Direcciones. En la quinta, se prepara al PC y el Secuenciador activa la línea ABS, con lo que la dirección existente en el Registro de Direcciones sale por el bus de direcciones hacia la memoria; también se prepara en este ciclo máquina la línea EB. En la última microinstrucción, el contenido del bus de datos (dato recogido de la memoria) pasa a la ALU que efectúa la operación de suma y, al activarse la línea EB, el resultado queda almacenado en el Registro B. Se incrementa P y se preparan el CM1 y el cerrojo para recibir un nuevo código OP.

270 / La unidad de control

6A.7.11. STA

Transfiere el contenido del Registro A a la posición de memoria direccionada por el segundo y tercer operandos que siguen al código OP (A_{16}).

Consta de 6 microinstrucciones. Las cuatro primeras son iguales a las que se han comentado para la instrucción LDA. La quinta, prepara el PC, activa la línea ABA.

activa la línea S_A y coloca la señal R/W en modo de escritura. En el cuadro de máquinas se incrementa el PC y se preparan el CM1 y el cerrojo para recibir una nueva instrucción.

6A.7.12. AND

Esta instrucción efectúa la operación lógica AND entre los bits del Registro A y el contenido de la posición de memoria direccionada por el segundo y tercer octeto r andos que siguen al código Op (B_{16}). Consta de 6 ciclos máquina, que son semejantes a las restantes operaciones lógicas o aritméticas que se realizan con la ALU.

卷之三

6A.7.13. STE

El contenido del Registro B se transfiere a la posición de memoria dirección por el segundo y tercer operando que siguen al código OP (C₁₆).

6A,7,14, LDA #

Es una instrucción de carga inmediata del Registro A. El segundo operando que sigue al código OP (D_{16}) es el valor que se carga, de forma inmediata, en Registro A.

Consta de 4 ciclos máquina. En el primero, se prepara al PC. En el segundo, se incrementa el PC y se prepara la línea EA. La tercera microinstrucción prepara el PC y activa EA. En la última microinstrucción, se incrementa el PC y se prepara el CMI y el cerojo.

6A.7.15. OR

Realiza la operación lógica OR entre el Registro A Y el contenido de la posición de memoria direccionada por el segundo y tercer operandos. El resultado queda almacenado en el Registro B y su código OP es E16.

Consta de 6 microinstrucciones semejantes a las anteriores instrucciones de tipo lógico o aritmético.

6A.7.18. MOV A,B

Una instrucción sirve para transferir el contenido del Registro A al Registro B, pasando por la ALU. Los señalizadores quedan afectados como en la instrucción de suma. El código OP es el F₁₆.

EB. En la ultima, se incrementa el PC, se activa EB y el corrijo (registro de instrucciones) y el CMI (Contador de Microinstrucciones) queda inicializado para recibir un nuevo código OP.

En las secciones de las figuras A6-1 y A6-2 se han representado gráficamente los microprogramas de las señales de control que salen del Secuenciador y que conforman las microinstrucciones de cada instrucción del repertorio.

Fig. A6-11. Tabla con el secuenciamiento de las señales de control en las microinstrucciones que componen las instrucciones del repertorio.

◎

6.A.8. DIAGRAMAS DE CONEXIONADO DE LOS CIRCUITOS INTEGRADOS EMPLEADOS EN LA CONSTRUCCIÓN DE LA UCP

En las figuras A6-13, A6-14 y A6-15 se muestran los diagramas de conexión de todos los circuitos integrados utilizados en la construcción de la UCP descrita.

Fig. A6-12. Continuación de la tabla de microinstrucciones de la figura A6-11.

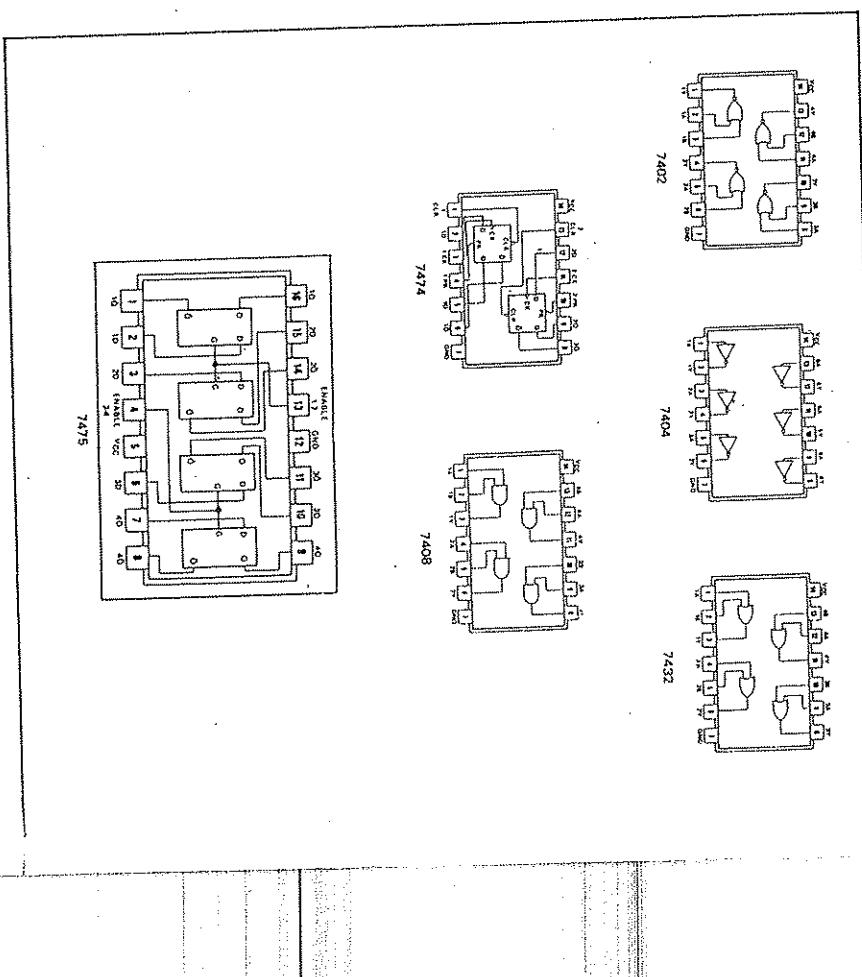


Fig. A6-13. Diagrama de conexionado de circuitos integrados.

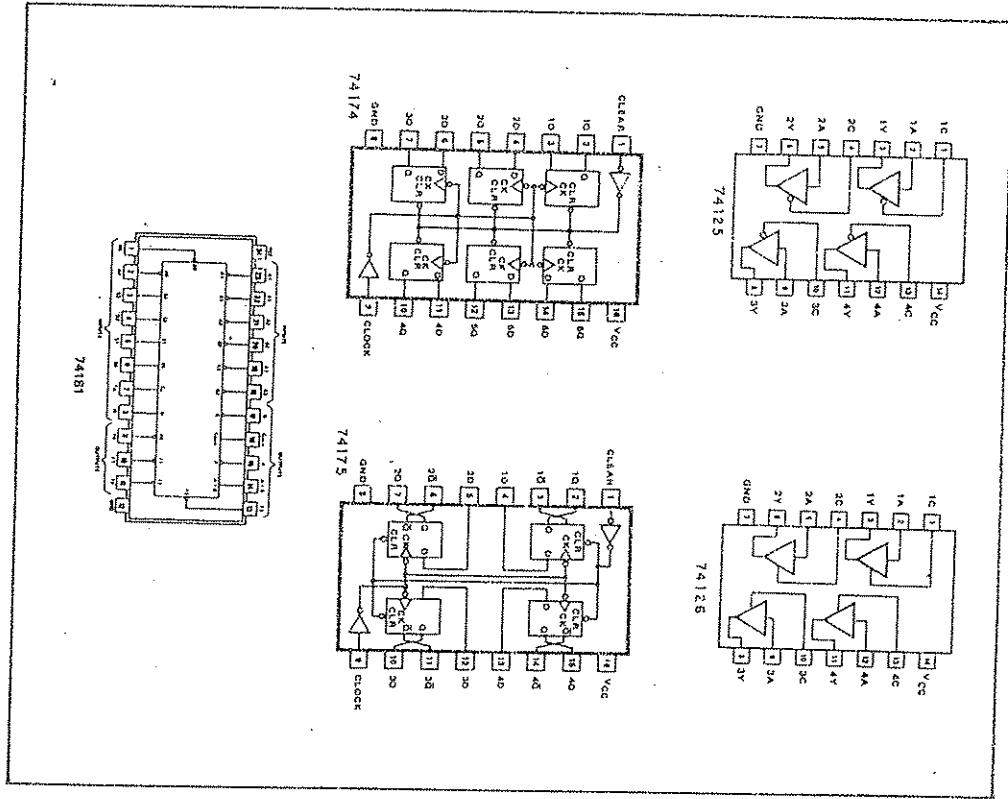


Fig. A6-14. Diagrama de conexionado de circuitos integrados.

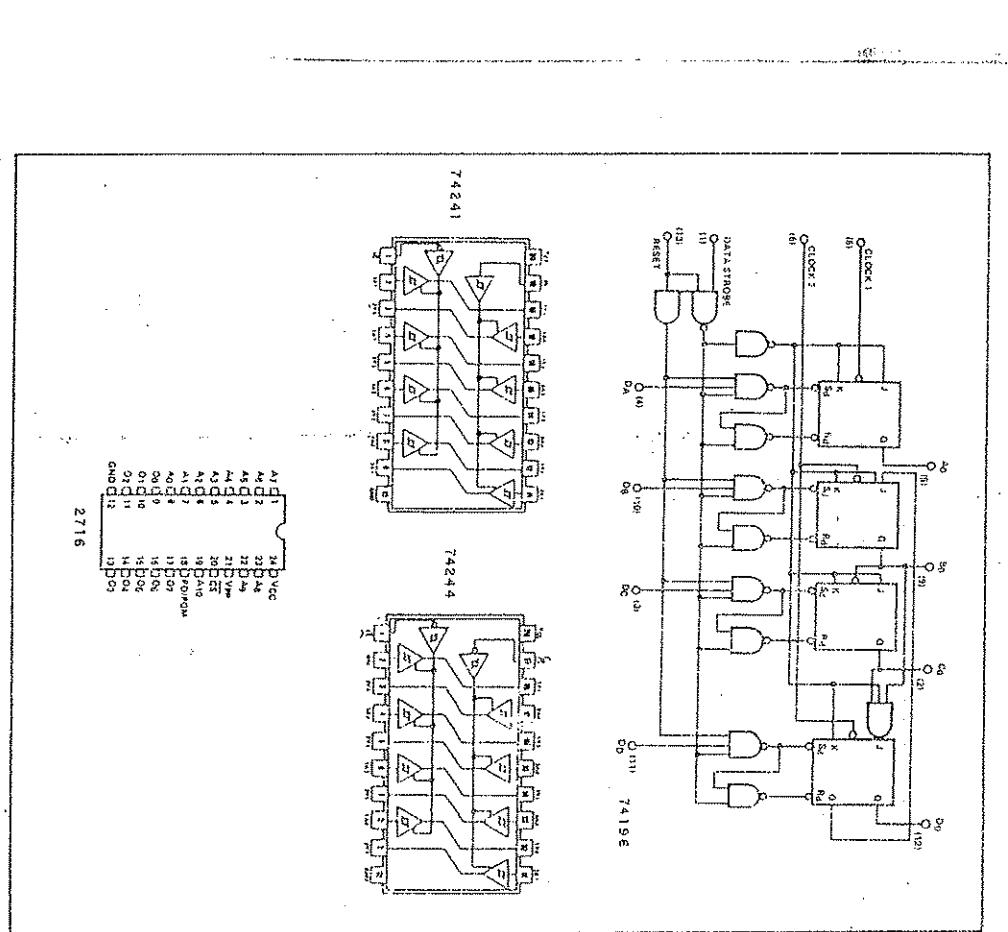


Fig. A6-15. Diagrama de conexionado de circuitos integrados.



EJERCICIOS

Ejercicio 6.1

Indicar los procedimientos existentes para reducir el tamaño de la memoria de control.

Ejercicio 6.2

En la unidad de control microprogramada con microbituración condicional, en el caso de secuenciamiento implícito, indicar la función del multiplexor y su selección de canal. ¿Cuál se usa en cada caso? ¿Para qué se usa el comparador?

Ejercicio 6.3

Supuesto un juego de instrucciones con el siguiente formato:

CÓDIGO 7 bits	OPERACION 4 bits	OPERANDO
------------------	---------------------	----------

y teniendo en cuenta que por término medio cada instrucción contiene 8 microinstrucciones, indique el tamaño de la memoria de control así como la longitud de las microinstrucciones, caso de secuenciamiento explícito e implícito:

- a) Cuando la microprogramación es horizontal.
- b) Cuando la microprogramación es vertical y campos codificados.
- c) Cuando la microprogramación es vertical y campos solapados.

Nota: Hay 16 señales de control.

Ejercicio 6.4

Sobre la figura 6.9 del libro, extraiga el juego de instrucciones completo correspondiente. De acuerdo con los direccionamientos que se pueden realizar, diseñar el formato de la instrucción, suponiendo palabras de 8 bits y una memoria de 256 posiciones, en palabras de 8 bits.

278 / La unidad de control

Ejercicio 6.5

Explicar detalladamente las diferencias entre una unidad de control por lógica cableada y una unidad de control microprogramada. Indicar las ventajas y desventajas de cada una de ellas.

Ejercicio 6.6

Dado el siguiente microprograma:

	T ₁	T ₀	T ₁	T ₂	T ₃	T ₄	T ₅	T ₆	T ₇
TP	1	0	0	0	0	0	0	0	1
FD	1	0	0	0	0	1	0	0	1
CM	0	1	0	0	0	0	1	0	0
L	0	1	1	1	0	0	1	1	1
ES	0	0	0	0	0	0	0	0	0
TM	0	0	0	1	0	0	0	0	1
F1	0	0	0	1	0	0	0	0	0
T1	0	0	0	0	0	1	0	0	0
XX1	—	—	—	—	—	0	—	—	0
XX2	—	—	—	—	—	—	1	—	1
DB0	—	—	—	—	—	—	1	—	—
DB1	—	—	—	—	—	—	1	—	—
DB2	—	—	—	—	—	—	1	—	—
DB3	—	—	—	—	—	0	—	—	—
XY1	—	—	—	1	—	0	—	—	0
XY2	—	—	—	1	—	1	—	—	1
OP3	—	—	—	0	—	0	—	—	0
OP2	—	—	—	0	—	0	—	—	0
OP1	—	—	—	0	—	0	—	—	0
OPO	—	—	—	0	—	1	—	—	1
TD	0	0	0	1	0	1	0	0	0
DAO	—	—	—	—	—	—	—	—	0
DA1	—	—	—	—	—	—	—	—	0
DA2	—	—	—	—	—	—	—	—	1





DA3	--	--	--	--	--	--	--	--	0
TA	0	0	0	0	0	0	0	1	
E	0	0	0	0	0	0	0	1	
PO	1	0	0	0	0	0	0	1	
FP	0	0	0	1	0	0	0	0	

Indicar a qué instrucción corresponde, así como el cronograma correspondiente.

Ejercicio 6.7

Modificar el esquema de la figura 6.9, de forma que los resultados de la ALU y los datos leídos de la memoria principal vayan, inevitablemente, a los registros temporales RA y RM y de ahí al bus de datos. Diseñar el cronograma para la instrucción MOV R5, R8.

Ejercicio 6.8

A partir del computador de la figura 6.9, diseñar el cronograma y microprograma para la ejecución de la instrucción MUL de dos operandos, con direccionamiento relativo absoluto: MUL DIR1, DIR2.

Ejercicio 6.9

Añadir al circuito de la figura 6.9 una pila direccionalada por un registro SP (Puntero Stack) que pueda ser cargado desde el bus de datos, decrementando e incrementando, según la operación a realizar en la pila. Ejecutar un microprograma que realice la carga del SP con una dirección inmediata y la transferencia de los datos contenidos en el registro R6 al registro R10 a través de la pila.

Ejercicio 6.10

Dibujar el esquema de una máquina de cero direcciones, especificando las líneas que salen del secuenciador.
Si se utiliza unidad de control microprogramada, dibujar una unidad de control con secuenciamiento explícito.

Unidad de entrada y salida



7.1. GENERALIDADES SOBRE EL INTERCAMBIO DE INFORMACIÓN CON EL EXTERIOR

La Unidad de Entrada y Salida (E/S) se encarga de realizar la conexión y adaptación de la UCP con una gran variedad de dispositivos periféricos.

Dada la gran variedad de periféricos existentes, su adaptación a la máquina presenta diversos problemas, entre los que destacan:

1. Los periféricos tienen *velocidades de transmisión* que se extienden desde unos pocos bytes por segundo, hasta un millón de bytes por segundo.
2. Los periféricos suelen usar un *ancho de palabra de un byte*, que no coincide con la palabra de trabajo de gran parte de los computadores.
3. Algunos periféricos son de *lectura*, otros, de *escritura* y, finalmente, otros son de *lectura y escritura simultáneas*.

El intercambio de información entre la UCP y los periféricos ha de resolver dos aspectos fundamentales:

- a) Debe existir un mecanismo que permita la transmisión de información y que se encargue también de solucionar cuestiones como el direccionamiento del periférico, el camino a través del que se efectúa la transferencia, la posible conversión serie/paralelo, la conversión de código, etc.
- b) Hay que diseñar un mecanismo de control que determine el origen y el destino de la información, cantidad a transmitir, códigos de protección, etc.

Estos dos mecanismos precisos se reparten funciones que pueden provenir del controlador del periférico, de la UCP y de los programas de entrada y salida.

Se define como *transferencia elemental de información*, a la que tiene por objeto el envío o recepción de una sola unidad de información (byte o palabra), ya sea ésta un dato o una palabra de control.

En la transferencia elemental hay que realizar las siguientes funciones:

1. Establecimiento de la *comunicación física* entre la UCP y el periférico, para la transmisión de 1 bit, 1 byte o 1 palabra. Se evaluarán los diferentes mecanismos físicos de comunicación y las señales necesarias con sus cronogramas y sistemas de sincronización.
2. *Control de los periféricos*, en el que se incluye el encendido y apagado del mismo, así como la interpretación y modificación de su estado. La UCP gober-ará las señales de control necesarias.

Una *operación de Entrada/Salida* es la que realiza la transferencia de un "conjunto" de datos, como pueden ser un sector de un disco, un registro de una cinta o una línea de un terminal de pantalla.

1. Recuento de bytes o palabras, para conocer el fin de la operación.
2. Sincronización de la velocidad de la UCP y del periférico, que es distinta a la requerida en la transferencia elemental de un byte o una palabra.

3. Detección de errores mediante códigos de paridad o polinómicos, con repetición de la transferencia en caso necesario.

4. Almacenamiento temporal de la información, conversión de códigos, conver-sión serie/paralelo, etc.

7.2. COMUNICACION FISICA ENTRE LA UCP

Y LOS PERIFERICOS

Para efectuar una transferencia elemental (bit, byte o palabra) entre el periférico y la UCP existen dos alternativas básicas:

1. Por ejecución de una instrucción, "Entrada/Salida programada".
 2. Por acceso directo a memoria.
- Para analizar los detalles de la comunicación, se representa al periférico como un conjunto de registros destinados a recibir o enviar la información, acompañados de una lógica de control que interpreta las señales de control enviadas por la UCP y que genera otras señales de control que aquélla exige del periférico.

7.3. ENTRADAS Y SALIDAS PROGRAMADAS

Con este procedimiento la transferencia de un byte o palabra se realiza mediante la ejecución de una instrucción específica de E/S, aunque, en ciertas ocasiones, se pueden emplear instrucciones de transferencia a memoria, del tipo LOAD y STORE.

Al decodificar la instrucción de E/S, la Unidad de Control del computador genera una serie de señales de control, envía al exterior la dirección del periférico, incluida en la mencionada instrucción y envía el dato o se dispone a recibirla, de-pendiendo del tipo de acceso.

Las instrucciones de E/S son, fundamentalmente, de transferencia, del tipo MOVE, donde un operando es un registro del controlador del periférico y el otro es un registro general de la UCP, o bien, una posición de memoria.

El camino físico que se establece para soportar la transferencia elemental entre el registro del periférico y el de la UCP o la memoria, se lleva a cabo utilizando tres tipos de señales:

- Señales de dirección, transmitidas por el bus de direcciones.
- Señales de datos, que se transfieren por el bus de datos.
- Señales de control, correspondientes al bus de control.

7.3.1. Señales de dirección

La instrucción de E/S viene acompañada de una dirección que se debe reproducir en las líneas del bus de direcciones. Si esta información consta de p bits, permite establecer 2^p direcciones distintas, sirviendo cada una de ellas para especificar lo que se denomina *puerta de E/S*. Cada periférico emplea una o varias puertas para comunicarse con la UCP, sirviendo cada una para enviar y/o recibir informaciones a nivel de byte o palabra.

El mapa de direcciones de las puertas es independiente, hay que establecer un mecanismo de decodificación que convierta cada dirección en una señal de selección de la puerta correspondiente. La decodificación puede tener carácter *centralizado* o ser *independiente* para cada puerta.

En la figura 7-1 se muestra el esquema de decodificación centralizada, en la que se decodifican 6 bits de direcciones, de los 8 que tiene el bus, mediante 9 pastillas decodificadoras 3×8 , configurando un decodificador de 6×64 , del que se obtiene 64 señales de selección de dirección de las puertas E/S. El decodificador 0 se activa con la señal de control $10/\bar{M}$ (Entrada-Salida/Memoria) y activa a uno de los otros 8.

Otra técnica consiste en que cada puesta de E/S reconozca su propia direc-ción, mediante un detector particular para la dirección asignada. Un ejemplo de este

procedimiento se presenta en la figura 7-2, en cuyo esquema existen una serie de inversores que forman la palabra de dirección deseada y una puerta AND que hace el producto lógico de todos los bits que componen la dirección.

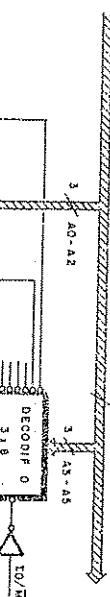


Fig. 7-1. Esquema de una codificación centralizada de direcciones. Se decodifican 6 líneas del bus de direcciones mediante 9 pastillas decodificadoras de 3×8 , que proporcionan un total de 64 señales de selección de dirección de puentes de E/S.

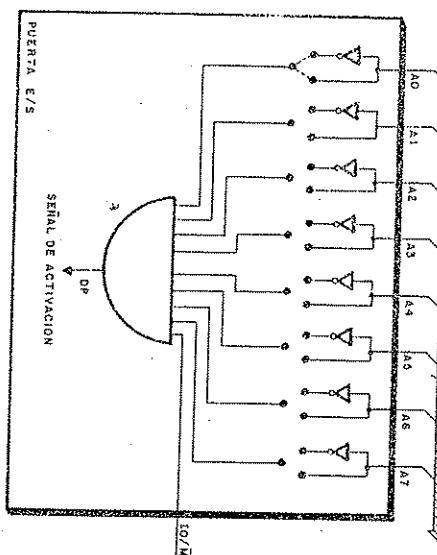


Fig. 7-2. La puerta de E/S tiene su propio detector de la dirección asignada, que está formado por varios inversores y una doble puerta AND.

Tanto en la figura 7-1 como en la 7-2, sólo cuando la señal $10/\bar{M}=1$, lo que significa acceso a periférico y no a memoria, se pueden activar las señales de selección de periférico DP.

Es muy frecuente que no exista una correspondencia biunívoca entre direcciones y puertas, es decir, que a cada dirección concreta no le corresponda una puerta determinada, con objeto de simplificar la decodificación de estas direcciones. Si, por ejemplo, se dispone de una máquina de 8 bits de dirección de E/S y sólo se quieren conectar 8 periféricos con una puerta cada uno, se observa claramente que, de las 256 combinaciones posibles con los 8 bits, la mayoría no se usan. Como se muestra en la tabla de la figura 7-3, la decodificación de direcciones admite tres soluciones en el caso propuesto.

- Solución 1. Se usan sólo 3 bits de dirección y los otros 5 quedan indefinidos (X), otros 5 bits valen cero.

- Solución 2. Se usan 3 bits de dirección y los otros 5 quedan indefinidos (X), o sea, pueden tomar cualquier valor.

- Solución 3. Se emplea un bit para cada periférico.

Obsérvese que en la solución 1, figura 7-3, existe una relación biunívoca entre periféricos y direcciones. En la solución 2, a cada periférico le corresponden varias direcciones, por ejemplo al periférico 2 le corresponden 32 direcciones, porque 5 de los bits no influyen en la selección. Finalmente, en la solución 3 corresponden 128 direcciones a cada periférico, pero en esta situación muchas direcciones activarían simultáneamente a varios periféricos, por lo que las únicas direcciones permitidas son aquellas 8 que tienen un solo bit a 1 y los demás a 0.

La figura 7-4 presenta una interpretación gráfica del uso del mapa de memoria de E/S para las soluciones comentadas.

Las soluciones 2 y 3 desaprovechan el mapa de memoria, pero, al no necesitar más de 8 periféricos, reducen el coste del sistema. Finalmente, hay que destacar que la última solución tiene la ventaja de no precisar decodificador, por lo que es la más económica, pero tiene el gran inconveniente de ofrecer un cierto riesgo; en efecto, si por error se realiza una entrada que activa a más de un periférico, se producirá un *choque en el bus*, puesto que dos o más elementos se conectan simultáneamente, lo que puede provocar la destrucción de algún componente o, al menos, un resultado erróneo.

7.4. MAPA DE MEMORIA COMUN

Algunos computadores, como los DEC, y microprocesadores, como los de Motorola, no hacen diferencia entre el mapa de E/S y el mapa de la memoria principal, tratando a las puertas de E/S como si fuesen direcciones de memoria convencionales. Esta técnica se puede emplear en cualquier computador, sin más que eliminar las señales que identifican las direcciones de E/S ($10/\bar{M}$), empleando en su lugar las de acceso a la memoria principal.

Suele ser habitual reservar a las E/S la zona superior o la inferior del mapa de memoria común, como se representa en la figura 7-5.

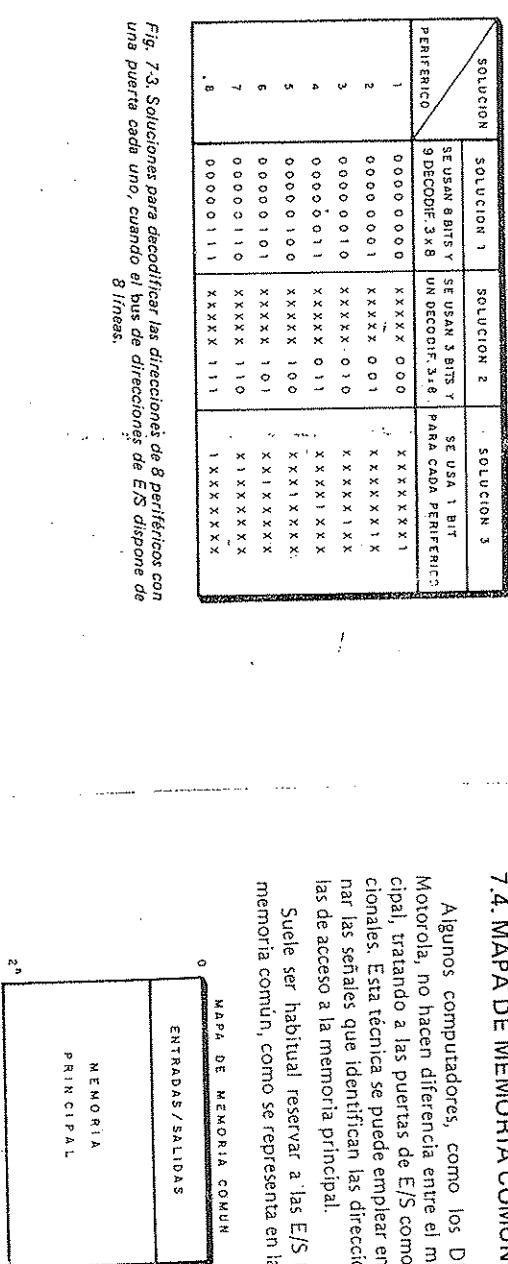


Fig. 7-3. Soluciones para decodificar las direcciones de 8 periféricos con una puerta cada uno, cuando el bus de direcciones de E/S dispone de 8 líneas.

SOLUCION	SOLUCION 1	SOLUCION 2	SOLUCION 3
PERIFERICO	SE USAN 8 BITS Y 9 DECODIF. 3x8	SE USAN 3 BITS Y UN DECODIF. 3x8	SE USA 1 BIT PARA CADA PERIFERICO
1	0000 0000	xxxx 0000	xxxxxx 0000
2	0000 0001	xxxxx 001	xxxxxx 001
3	0000 0010	xxxxx 010	xxxxxx 010
4	0000 0011	xxxxx 011	xxxxxx 011
5	0000 0100	xxxxx 100	xxxxxx 100
6	0000 0101	xxxxx 101	xxxxxx 101
7	0000 0110	xxxxx 110	xxxxxx 110
8	0000 0111	xxxxx 111	xxxxxx 111

Fig. 7-4. Distribución del mapa de memoria de E/S para las 3 soluciones propuestas en la tabla de la figura 7-3.

7.5. SEÑALES DE DATOS

Aunque en este capítulo se considera, generalmente, el caso del mapa de direcciones de E/S independiente, su extensión al caso de mapa de memoria compartida es inmediata y se produce eliminando las señales que diferencian las E/S y la memoria ($10/\bar{M}$), y empleando los bits de dirección comunes. También, habrá que sustituir las instrucciones INPUT y OUTPUT por las LOAD, STORE o MOVE.

Los datos se transmiten por un conjunto de líneas bidireccionales, que reciben el nombre de bus de datos, aunque también existe la posibilidad de utilizar dos conjuntos de líneas unidireccionales.

La figura 7-6 muestra el conexionado de las puertas de E/S a un bus bidireccional. El bus de datos de E/S coincide, generalmente, con el bus interno de datos del computador, que se prolonga al exterior a través de una etapa aisladora. La comunicación de cada puerta al bus hace uso de un buffer triestado, activado por la combinación simultánea de las señales de dirección de palabra del periférico (\bar{RD}); así, la puerta se conecta al bus y envía una palabra a la UCP. También existe un registro activado por la combinación de DP y WR (escritura) que sirve para que la puerta de E/S tome un dato del bus.



Fig. 7-6. Conexión de las puertas E/S al bus de datos bidireccional.

Otra solución para adaptar el bus de datos a los periféricos, consiste en utilizar un multiplexor que conecta individualmente a cada periférico las líneas del bus de datos. Figura 7-7.

El sistema de conexión individual con multiplexor, además de exigir una gran cantidad de líneas de conexión, tiene el inconveniente de la difícil expansión cuando se completa el multiplexor, cosa que no sucede con el procedimiento empleado en la figura 7-6.

7.6. SEÑALES DE CONTROL

Las señales de control sirven para especificar el tipo de transferencia, esto es, si es entrada o salida, y establecer su temporización, para lo cual se emplean dos métodos:

1. Síncrono.
2. Asíncrono.

7.6.1. Transferencia síncrona

En la figura 7-8 se muestra el cronograma de una transferencia síncrona de escritura, en la cual, la señal WR establece el período en que la UCP tiene datos válidos en sus líneas de salida (bus de datos). En dicha figura, este tiempo corresponde a la suma $t_1 + t_2 + t_3$, que se obtiene añadiendo el tiempo de la señal WR un margen anterior y otro posterior. Por tanto, el flanco de subida de la señal WR activa el registro de la puerta de E/S, debe estar comprendido en ese intervalo de validez. La solución clásica consiste en emplear el propio flanco de subida de la señal WR para activar la carga del registrador del periférico (figura 7-6), que se produce con un cierto retraso respecto al mencionado flanco. En la figura 7-8 se señala con una flecha la generación de un flanco por otro.

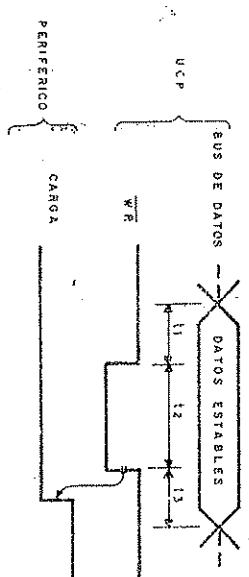


Fig. 7-7. Utilizando un multiplexor se puede conectar el bus de datos a cada periférico, de forma individual.

En la transferencia de lectura, la señal \overline{RD} marca el tiempo en el que el periférico dispone del bus para colocar su dato. Se suele emplear la propia señal \overline{RD} para abrir el buffer triestado al bus (figura 7-6), de forma que, con el inevitable retraso, éste contenga el dato estabilizado, tal como se indica en la figura 7-9, en la que se señalan las interrelaciones entre el principio y el final de \overline{RD} y la aparición del dato en el bus.

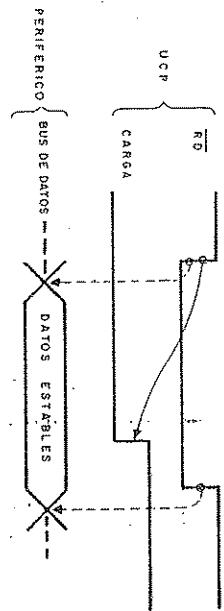


Fig. 7-9. Cronograma de una transferencia síncrona de lectura.

En la transferencia síncrona, la UCP establece un ritmo o temporización que debe seguir el periférico, fracasando la transferencia cuando el periférico no es capaz de leer o escribir en el bus en el tiempo especificado. Además, la UCP no recibe señal alguna que indique si la transferencia se ha realizado con éxito, ya que podría haberse direccionado un periférico desconectado o inexistente, a pesar de lo cual la UCP almacenaría un dato, que, evidentemente, será incorrecto.

7.6.2. Transferencia asíncrona o con interbloqueo (diálogo)

Esta transferencia requiere una señal de aceptación ACP, con la que el periférico contesta a la petición de transferencia generada por la UCP. Se establece un "diálogo" que permite adaptar el cronograma a las necesidades de tiempo del periférico, puesto que, hasta que no se activa ACP, no se completa la transferencia. En la figura 7-10 se muestran los cronogramas de transferencia asíncrona para lectura y escritura. En el caso de lectura, no se completa la transferencia hasta que se produce el ACP. En el caso de escritura, la UCP mantiene la señal \overline{WR} alta hasta que el periférico responde con ACP.

La transferencia asíncrona tiene dos ventajas:

1. Permite adaptar periféricos de distintos tiempos de funcionamiento, ya que la UCP se adapta a las necesidades de cada caso.
2. Existe una mayor garantía de la validez del dato, puesto que exige una contestación positiva del periférico para completar la transferencia. Para evitar que

el sistema quede bloqueado, ya sea porque no existe el periférico o porque éste no contesta, se establece un periodo de espera máximo, transcurrido el cual, se da la transferencia como errónea.

El diálogo que establece la UCP con el periférico, en la transferencia asíncrona recibe la denominación de *handshaking*, que se puede traducir como *interbloqueo*.

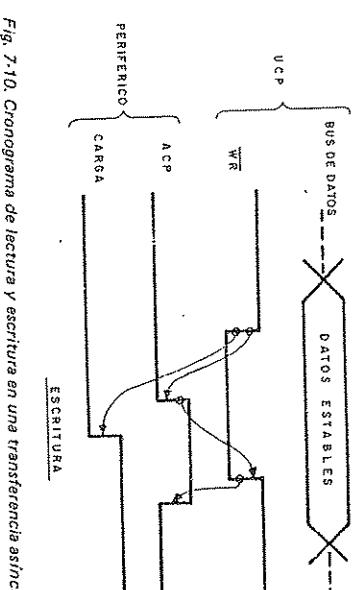
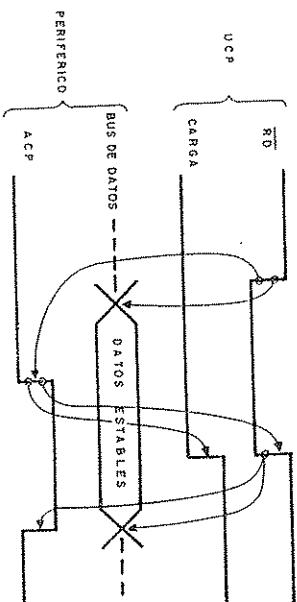


Fig. 7-10. Cronograma de lectura y escritura en una transferencia asíncrona.

7.7. ACCESO DIRECTO A MEMORIA

Se ha estudiado la forma de realizar una transferencia elemental mediante E/S programadas, haciendo uso de instrucciones que ejecuta la UCP. Se pasa a analizar el segundo procedimiento para llevar a cabo transferencias elementales, que se denuncian a continuación.

nina acceso *directo a memoria (DMA)* y en el que el controlador del periférico es el encargado de establecer la comunicación directa con la memoria, sin intervención de la UCP, o sea, sin ejecutar instrucción alguna.

Para realizar el DMA, el controlador del periférico debe conocer las siguientes informaciones:

1. Dirección de la memoria principal a la que hay que acceder, para poder generar las señales de dirección apropiadas.
 2. Dirección a la que responde el periférico.
 3. Tipo de operación: lectura o escritura.
 4. Número de bytes a transferir.
- En la fase de inicialización de la operación de E/S habrá que suministrar al controlador del periférico las informaciones expuestas.
- Se puede realizar el acceso directo a memoria de dos formas básicas:
- Por memoria multipuerta.
 - Por robo de ciclo.

7.7.1. Memoria multipuerta

Cuando la memoria principal dispone de varias puertas de acceso, una se reserva para conectar la UCP y las restantes se destinan a los controladores que disponen de DMA. Figura 7-11.

Es frecuente que una memoria multipuerta esté dividida internamente en varios bloques para permitir accesos en paralelo. De esta forma se consigue que las peticiones de acceso, que se reciben por cada puerta, se puedan tratar en paralelo siempre que no direccionen el mismo bloque de memoria.

El mayor inconveniente de la memoria multipuerta es su coste, puesto que, además de prioridades, para resolver los choques, o sea, las peticiones simultáneas de un mismo bloque, concediendo un acceso y retardando los demás.

Las señales que debe generar el controlador del periférico para comunicarse con la memoria son:

- Dirección de la memoria principal.
- Dato; si se trata de una lectura.
- Fin del ciclo de memoria.

Se precisa una señal de sincronización entre la memoria y el controlador del periférico, puesto que la duración del ciclo de memoria es variable, dependiendo de la existencia o no de choque. La señal de fin de ciclo indica al controlador que ya tiene disponible el dato o que ya ha terminado la escritura.

7.7.2. Robo de ciclo

En este método la memoria sólo tiene una puerta, que debe ser compartida por la UCP y los periféricos.

La transferencia requiere que, previamente, se pongan de acuerdo la UCP y los controladores de periféricos para gobernar los buses y las señales de control de la única puerta que tiene la memoria. En general, el control pertenece a la UCP, que lo cederá cuando lo solicite un periférico, perdiendo entonces o *dejándose robar* un ciclo o fase. Esta acción consta de los siguientes pasos:

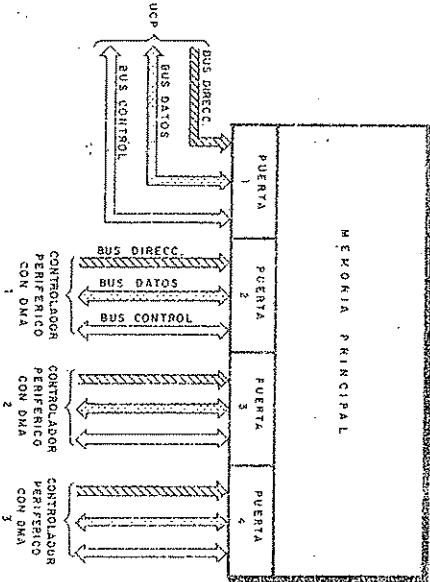


Fig. 7-11. Forma de implementar el DMA de los periféricos cuando la memoria principal es multipuerta. Los buses de cada puerta son totalmente independientes.

1. El controlador del periférico solicita a la UCP un acceso a memoria principal o sea, solicita robar un ciclo mediante la activación de la correspondiente señal de control (HOLD o BSRQ).

2. La UCP contesta, mediante una señal de control (HOLDA o BSAK), concediendo el ciclo. Al mismo tiempo, pone en estado de alta impedancia sus buses y las señales de control de acceso a memoria y a periféricos. Para conceder el robo del ciclo solicitado, la UCP tiene que esperar a que finalice la fase de la instrucción en curso.

3. El controlador del periférico realiza el acceso a la memoria principal.

4. Finalizado el acceso por parte del controlador, éste devuelve a la UCP el control de los buses y demás señales, desactivando la señal de petición de ciclo (HOLD o BSRQ).

Mientras el controlador del periférico realiza el acceso a la memoria principal, la UCP queda a la espera. La ejecución de las instrucciones sufre una pequeña parada cada vez que se solicita el robo de ciclo. Cuando la UCP recupera el control, continúa la ejecución en la fase siguiente a la que se produjo la detención.

El robo de ciclo hace que no sea fija la duración de las instrucciones. Este hecho hace que tornario en consideración cuando se programan temporizaciones con bucles de espera.

En la figura 7-12 se muestra el esquema de conexiónado de un periférico para poder realizar el acceso directo a memoria por robo de ciclo.

El controlador del periférico de la figura 7-12 posee una serie de registros para almacenar la dirección de la memoria principal, la dirección del periférico, el número de bytes que quedan por transferir y si la operación es de entrada o de salida. También debe ser capaz de ir incrementando el registro de dirección de memoria y de decrementar el registro de recuento de bytes, para actualizar estas informaciones cada vez que se realiza un acceso. Para efectuar todas las funciones encadenadas al controlador, éste dispone de una "unidad de control" propia, que genera las correspondientes señales de control. En concreto, ha de enviar a la UCP las siguientes señales:

- BSRQ, para solicitar el robo de ciclo.
- FD, señal de flanco que carga el registro de direcciones D de la UCP.
- L, señal activa por nivel, que indica ciclo de lectura.
- E, señal activa por nivel, que indica ciclo de escritura.
- CM, señal activa por flanco, que carga el bus de datos en el registro RM.
- FRM, señal activa por flanco, que carga el bus de datos en el registro RM.
- TM, señal de nivel, que abre la memoria al bus de datos.
- INT, señal de solicitud de interrupción.

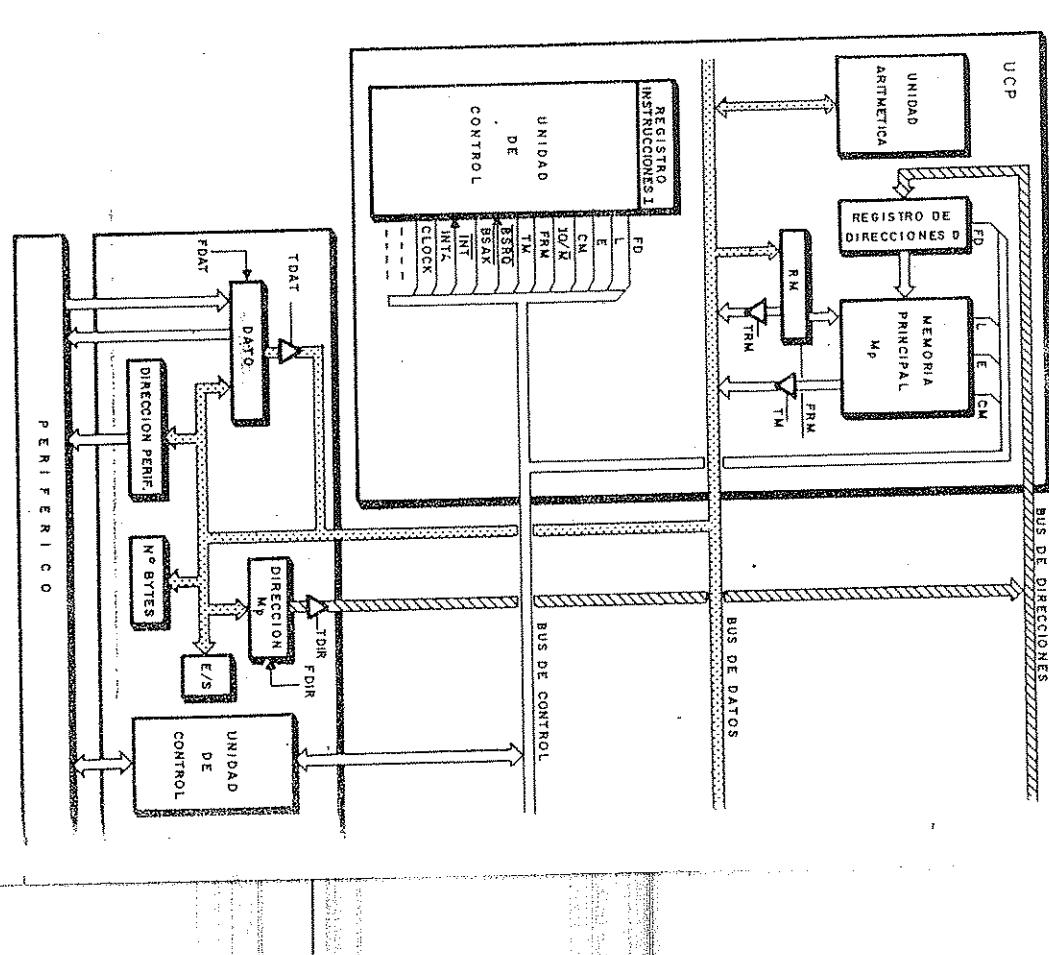


Fig. 7-12. Esquema de conexiónado del controlador de un periférico, capaz de realizar acceso directo a memoria por robo de ciclo.

Por otro lado, el controlador del periférico ha de interpretar las señales generadas por la UCP, tales como B_{SAK}, INTA, L, E, I/O/M, etc.

Finalmente, el controlador ha de generar ciertas señales internas, entre las que destacan:

- TDAT, señal de nivel que abre el registro de datos del controlador al bus de datos.
- TDIR, señal de nivel que abre el registro de direcciones del controlador al bus de direcciones.
- FDAT, señal de flanco que carga en el registro de datos del controlador el contenido del bus al que se halla conectado.
- FDIR, señal de flanco que carga en el registro de direcciones del controlador el contenido del bus al que está conectado.
- INC, señal automática que incrementa el registro de direcciones del controlador y decremente la cuenta de bytes que quedan por transferir.

En la figura 7-13 se muestra el cronograma de acceso a memoria por robo de ciclo, reflejándose la sincronización de la transferencia por las flechas que aparecen entre los flancos de las señales.

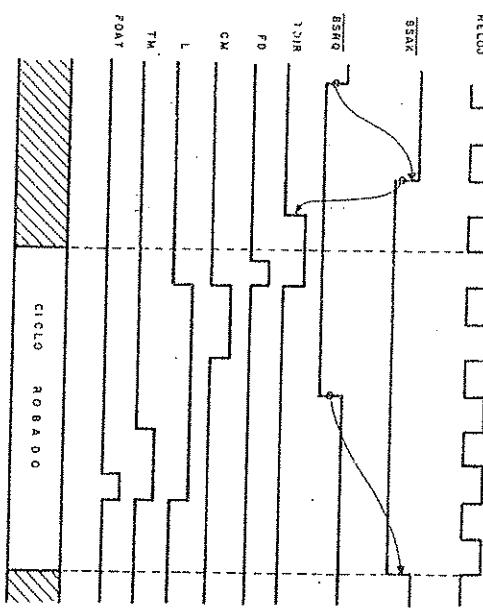


Fig. 7-13. Cronograma de acceso directo a memoria por robo de ciclo.

rata, inicializar el acceso directo a memoria, la UCP debe enviar al controlador las siguientes informaciones:

1. Direcciones del origen y del destino.
2. Sentido de la transferencia.
3. Número de bytes a transferir.

Para recibir esta información, previa al DMA, se suele dotar al controlador del periférico de una o más puetas de Entrada/Salida programadas. Mediante un programa compuesto por varias instrucciones de salida, que soportan las direcciones asignadas al controlador, la UCP envía la información necesaria y, una vez recibida, comienzan los ciclos de acceso directo a memoria.

Acabada la operación de E/S, el controlador del periférico envía una señal de interrupción a la UCP para avisar que quedan disponibles los buses.

7.8. CONTROL DE LOS PERIFERICOS

La mayor parte de los periféricos disponen de un conjunto de señales de "estado" y de "control", que permiten conocer su situación interna y simplificar su gobierno.

Las señales de estado informan sobre:

- Si el periférico está encendido o apagado.
- Si el periférico se halla actuando.
- Si el periférico es operativo o no.
- Si se produce un error de operación, como en el caso de un disco en el que no se encuentra la pista cero.
- Las señales de control sirven para ordenar ciertas acciones en el periférico, tales como:
 - Conectar o desconectar el periférico.
 - Seleccionar opciones.
 - Salto de página en una impresora.
 - Localización de marcas en cintas magnéticas.
- Para que el computador acceda a las señales de estado y envíe las de control, existen dos procedimientos:
 1. Manejar estas informaciones como si fuesen datos y transmitirlas mediante E/S programadas. En este caso, el periférico tiene dos puetas con dos direcciones distintas, una para datos y otra para control. Leyendo a la puesta de control se obtiene el estado del periférico y escribiéndola se actúa sobre sus señales de control.

2. Mediante la diferenciación en un mapa de direcciones de control, añadiendo una señal que distinga entre E/S de datos y E/S de control. Esta señal la genera la Unidad de Control al mismo tiempo que envía la dirección del periférico.
- El tratamiento de las señales de estado y control forma parte de las operaciones de E/S, pues para hacer, por ejemplo, una operación de lectura en un disco flexible, hay que comprobar que la unidad esté encendida, que esté insertado el disco y que no se halle realizando otra operación.

7.9. PRIORIDADES

El problema de las "prioridades" se plantea cuando dos o más elementos tienen que compartir un mismo recurso y se pueden producir peticiones simultáneas. Hay que decidir a qué elemento se atiende primero.

Se da el nombre de *peticionarios* a los elementos que comparten un recurso y *fase de servicio*, al tiempo empleado en atender a cada uno de ellos.

El control de prioridades se presenta en situaciones como las que se describen:

- Varios periféricos solicitan, al mismo tiempo, un robo de ciclo. El ciclo robo do será la fase de servicio.
- Varios periféricos solicitan, al mismo tiempo, una interrupción.
- Varios periféricos, conectados a la misma puerta de memoria multipuerta, solicitan un ciclo de memoria simultáneamente.
- Varias puertas de una memoria multipuerta tratan de acceder, simultáneamente, al mismo bloque de memoria.

Para resolver estas situaciones críticas existen dos procedimientos:

— Gestión distribuida de prioridades.

— Gestión centralizada de prioridades.

Hay dos aspectos fundamentales que debe resolver la política de gestión de prioridades:

1. Selección del peticionario agraciado con la nueva fase de servicio. Métodos tales como el de prioridad fija, petición más antigua, selección aleatoria, etc., pueden aplicarse, aunque algunos, como el de petición más antigua, exigen disponer de mayor información.
2. Una vez se ha concedido una fase de servicio a un peticionario, éste puede ser detenido o no al recibirse una petición de mayor prioridad. En general, no interesa detener las fases de servicio cuando son de corta duración.

Los mecanismos de "prioridad fija" sin detención de fase de servicio son los más sencillos y utilizados para el tipo de situaciones encuadradas dentro de las operaciones elementales de E/S.

El funcionamiento puede ser síncrono, cuando es fijo el tiempo de servicio. De lo contrario, hay que establecer un dispositivo interbloqueo o de diálogo, donde el peticionario que obtiene el servicio mantenga una señal que retenga el recurso todo el tiempo necesario.

7.9.1. Gestión distribuida de prioridades

Este método se recomienda cuando los distintos peticionarios están conectados al recurso común mediante un bus. Para resolver la prioridad se emplea uno de los dos procedimientos siguientes:

- Encadenamiento o daisy-chain.
- Lógica distribuida.

7.9.1.1. Encadenamiento o daisy-chain

Existe un elemento *maestro* que, de alguna manera, supervisa la concesión del recurso, aunque sean los propios peticionarios quienes determinan cuál de ellos se queda con la fase de servicio. En muchas ocasiones, el propio recurso actúa como maestro.

La concesión de la fase de servicio se basa en la actuación de dos señales comunes a todos los peticionarios y el maestro. La primera, que llamaremos \overline{PETI} , es de petición y actúa como entrada para el maestro y como salida para todos los peticionarios, que se conectan a ella en forma de "OR cableada", la cual permite que uno o varios peticionarios soliciten al maestro la concesión del servicio, poniendo a nivel bajo esta señal. La segunda señal, \overline{CONC} , corresponde a la concesión que genera el maestro en respuesta a \overline{PETI} . La señal \overline{CONC} la reciben todos los peticionarios, a modo de "testigo", para atender sólo a uno de ellos.

En algunos casos, existe una tercera señal de temporización \overline{TEMP} que establece el comienzo y la duración del servicio, aunque en otros, la misma señal \overline{CONC} realiza esta función.

La técnica de encadenamiento se basa en unos principios fundamentales, que la circuitería empleada para la implementación debe respetar:

1. Existe una línea única de petición a la que se conectan todos los dispositivos, que debe permitir se realicen peticiones simultáneas.

2. Hay una sola línea de concesión, por la que el maestro envía un impulso o flanco a los dispositivos, que están conectados en cadena, de forma que la señal de concesión recorre en serie, a manera de un testigo, a los peticionarios y para en el primero que desea ser atendido.

3. Un mecanismo de encadenamiento garantiza que solamente un dispositivo toma la señal de concesión y evita que se cancele una fase de servicio cuando ya ha sido concedida.

4. La prioridad viene fijada por el orden en que se conectan los peticionarios a la cadena tiene la mayor opción o prioridad a quedarse con el pulso de concesión.

La solución propuesta en la figura 7-14 muestra cómo puede realizarse la conexión de los peticionarios a las líneas PETI y CONC. La conexión a la línea PETI se efectúa mediante colector abierto para formar una función OR cableada, de manera que cualquier peticionario pueda poner un 0 en la línea, solicitando atención, sin interferir eléctricamente a los demás.

Continuando el análisis de la figura 7-14 se observa que, mientras la señal CONC = 1, los biestables de petición BIPET dejan pasar las peticiones, en tanto de sus respectivos periféricos (PETPER). Sin embargo, cuando llega el impulso de concesión Y CONC = 0, se congela los BIPET, de forma que no puede variar la situación de petición de los periféricos, evitándose las ambigüedades en la obtención de la fase de servicio, así como que ésta pueda ser cancelada por un dispositivo de mayor prioridad. Durante este tiempo, la cadena de señales PRI forma una serie de "ceros" que se interrumpe en el primer dispositivo que tenga BIPET = 1, es decir, por el dispositivo de mayor prioridad que esté solicitando el servicio. El flanco de subida de CONC permite a este último dispositivo cargar un "uno" en su biestable de conexión BICON Y pasar a la fase de servicio. Sólo hay un dispositivo que puede poner a 1 el biestable BICON. Una vez atendida la petición, el periférico pone a 1 la señal PETI y borra el biestable BICON.

La figura 7-15 presenta el cronograma correspondiente a dos peticionarios en el supuesto de funcionamiento sincrónico, o sea, en el caso de que el dispositivo no intervenga en la duración del servicio, la cual está caracterizada por la señal TEMP. Las flechas de la figura señalan las dependencias entre las señales PETI, CONC y TEMP.

En resumen, la técnica de encadenamiento es sencilla y económica de implantar, pero la prioridad de los periféricos es invariante, al quedar determinada por su situación respecto al maestro. No es muy rápida, puesto que hay que dejar el tiempo necesario para que el testigo (CONC) recorra el máximo número de dispositivos previstos en el diseño.

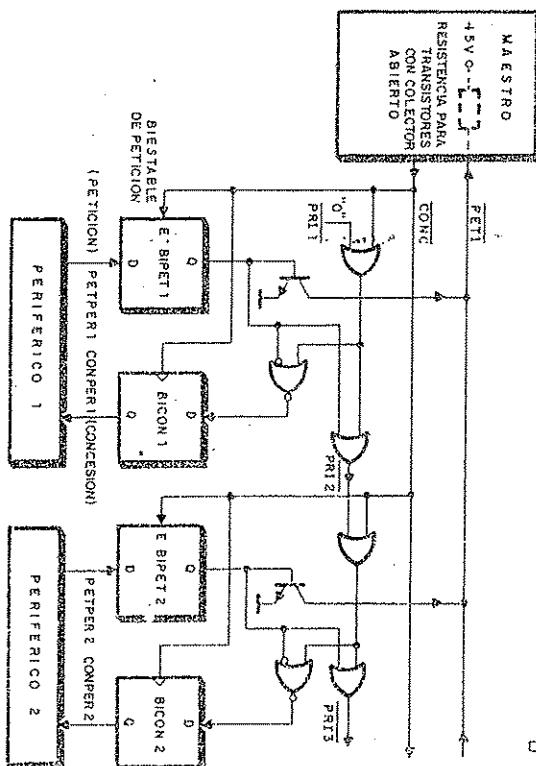


Fig. 7-14. Conexión de los periféricos a las líneas PETI Y CONC cuando se utiliza la técnica de "encadenamiento" para la gestión de prioridades. Cuanto más cercano este el periférico al maestro, mayor nivel de prioridad posee.

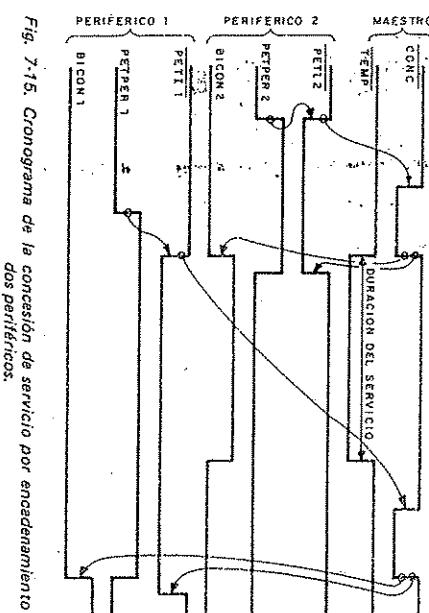


Fig. 7-15. Cronograma de la concesión de servicio por encadenamiento a dos periféricos.

7.9.1.2. Lógica distribuida

En este procedimiento no hay un maestro que participe en la asignación de prioridad, quedando a cargo de los propios peticionarios la manera de ponerse de acuerdo para decidir la concesión de la fase de servicio.

A título de ejemplo, se presenta el método de concesión de bus empleado en el bus S-100. Hay 4 líneas de prioridad DMA0-DMA3, con colector abierto. Cuando algún peticionario desea obtener la fase de servicio, coloca en dichas líneas su "código de prioridad". Puesto que con 4 líneas se pueden obtener 16 códigos, el bus S-100 admite a 16 peticionarios. Cada peticionario, al mismo tiempo que intenta coner en las líneas DMA0-DMA3 su prioridad, debe observar el código presente. Si éste es mayor que el suyo, se debe retirar. Una pareja de señales HLDA y HOLD evitan interferencias en la concesión del control del bus.

7.9.2. Gestión de prioridad centralizada

Este tipo de gestión de prioridad es recomendable cuando la conexión de los dispositivos se realiza usando un multiplexor, aunque también se emplea en conexión de bus.

Como se refleja en la figura 7-16 hay un órgano común, llamado *gestor de prioridades*, que puede residir en el dispositivo compartido, que recibe todas las señales individuales de petición PET_i. El gestor de acuerdo a la política establecida, selecciona al dispositivo que debe recibir la fase de servicio y se lo comunica con otra señal independiente CONC.

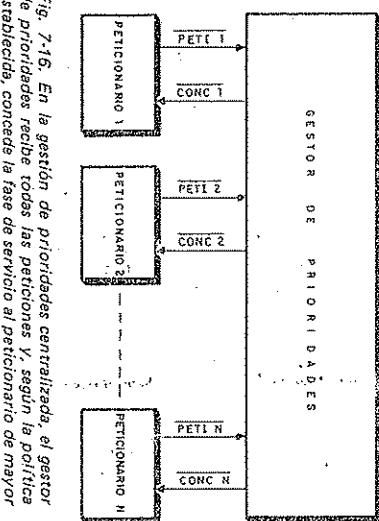


Fig. 7-16. En la gestión de prioridades centralizada, el gestor de prioridades recibe todas las peticiones y, según la política establecida, concede la fase de servicio al peticionario de mayor prioridad.

7.9.2.1. Gestión de prioridad híbrida

Emplea las características de la gestión centralizada y la distribuida. La figura 7-17 muestra una solución clásica: un mecanismo centralizado establece una serie de niveles de prioridad a cada uno de los cuales se pueden conectar una serie de peticionarios. Las prioridades, dentro de cada nivel, se asignan, por ejemplo, por el mecanismo de encadenamiento.

Finalmente, conviene destacar que los diversos periféricos que se conectan a cada puerta de una memoria multipuerta suelen actuar en la forma de bus con prioridad distribuida, que también es una situación de prioridad híbrida.

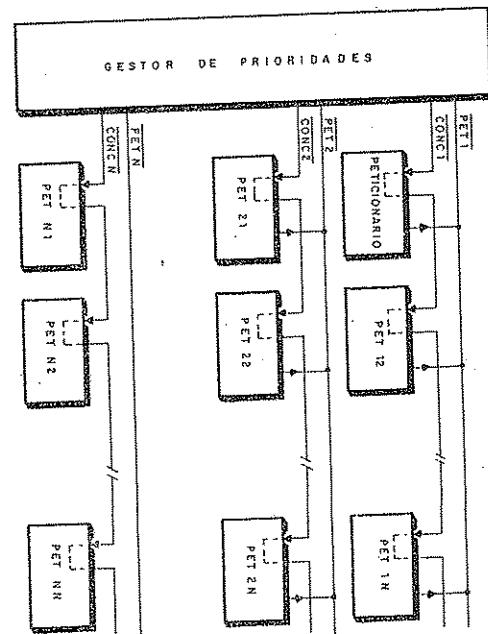


Fig. 7-17. Gestión híbrida de prioridades.

7.10. DESCRIPCION DETALLADA DEL FUNCIONAMIENTO DE UNA OPERACION DE ENTRADA/SALIDA

Para realizar un análisis completo de una operación de E/S se ha seleccionado, como ejemplo, la lectura de un registro que se halla grabado y almacenado en una cinta de papel perforado. Aunque esta tecnología ha caído en desuso en la actualidad, cubre todas las funciones de E/S y tiene la ventaja de ser clara y sencilla.

Se hace referencia a una cinta de papel con 8 perforaciones de datos más la guía, que es más pequeña, tal como se refleja en la figura 7-18.

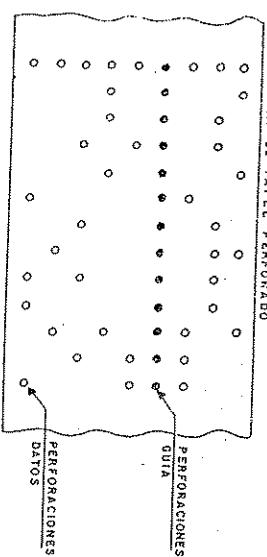


Fig. 110. La cinta de papel consta de 8 perforaciones para datos más la de guía, que es más pequeña.

en ASCII con siete bits, más uno de paridad. El registro está dividido en bloques de 19 bytes, con el formato siguiente:

Cubecano: Formada por los cara

Datos: Configurados por 16 bytes a cada

Color: Consistente en un byte obtenido mediante la suma de los 3 caracteres alfanuméricos.

puede constar de hasta 13 caracteres y se debe comprobar que coincide con el nombre del registro deseado. El último bloque también contiene el nombre del registro precedido de ^^.

La figura 7-15 muestra el esquema del mecanismo de lectura, cuyo transductor está formado por 9 fototransistores que generan un 0 cuando reciben luz desde un fotodiodo a través de una perforación. Al avanzar la cinta de papel, por encima de los fototransistores, éstos van detectando las perforaciones, produciendo unas señales como las que se representan en la figura 7-20. La presencia de un impulso de guía, más pequeño que los de datos, indica la presencia de un byte de información. El propio pulso de guía, al ser menor, sirve para cargar el byte de datos en un registrador (cerrojo 74374). Este impulso también se emplea para almacenar un 1 en el bistable de estado de "carácter disponible" CARD, que sirve para indicar a la UCP que ya puede ser leído el registro cerrojo.

304 / Unidad de entrada y salida

La velocidad de arrastre de la cinta de papel no es crítica para el proceso de tecnología, puesto que sus variaciones se reducen al modificar la escala de tiempos de las señales, pero sin alterar las formas y las relaciones entre los impulsos. Figura 7-20

Suponiendo una velocidad de arrastre de 1 m/s, y, dado que la separación de las perforaciones es de 2,55 mm., se puede calcular la cadencia de lectura, obteniendo-se aproximadamente 400 cps (caracteres por segundo), lo que corresponde a leer un carácter cada 2,5 ms. Para un computador de tipo medio, que pueda ejecutar un millón de instrucciones por segundo, este tiempo entre caracteres supone la ejecución de 2.500 instrucciones.

La operación de E/S propuesta debe ir tornando los bytes a medida que los genera el lector, lo que no supone problema, dada la lentitud del lector. Si un carácter no se recoge antes que illegue el siguiente, se destruye; puesto que se carga el nuevo encima. El mecanismo clásico para sincronizar la UCP y el periférico consiste en hacer esperar a la UCP hasta que el periférico tenga el dato disponible. Esto se logra

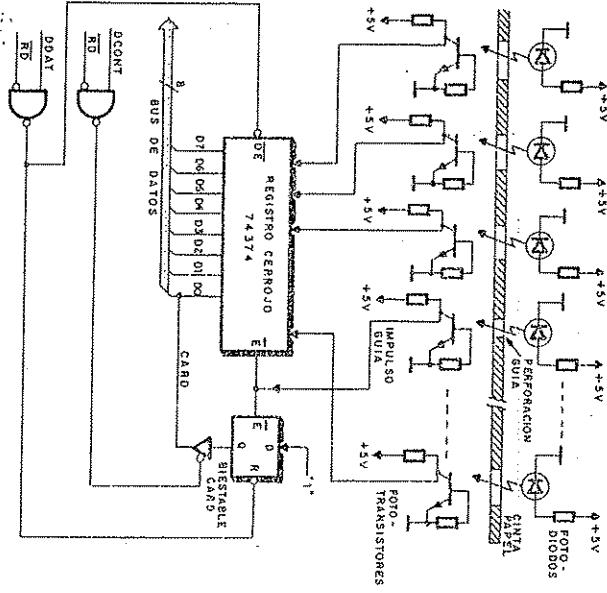


Fig. 1-19. Esquema del dispositivo de lectura de cinta de papel perforado. DCONT es el dispositivo de lectura del estado y DDAT la de lectura de datos.

Unidad de entrada y salida / 305

con un bucle de espera que comprueba, constantemente, el estado del periférico. Cuando se detecta que el bit de "Carácter disponible" CARD está a 1 al activarse RD y DCONT, la UCP sale del bucle y se recoge el dato del registro cerchio mediante una instrucción de entrada (INPUT), que activa RD y DDAT, a la vez que resetea el biesetable CARD.

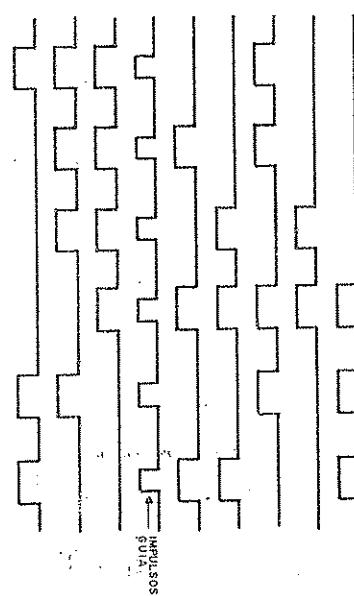


Fig. 7-20. Formas de onda del lector de cinta de papel perforado.

Se puede reestructurar un programa, a manera de subrutina, que permita obtener un dato del lector y cuyo ordinograma se ofrece en la figura 7-21.

El programa de lectura de la cinta de papel deberá disponer de los siguientes datos de partida:

- Dirección de destino en la memoria.
- Nombre del registro.
- Número de bloques del registro.

Además, dicho programa constará de las 4 partes fundamentales siguientes:

1. Inicialización.
2. Búsqueda del primer bloque del registro en base a si leyendo secuencialmente hasta que se encuentre un bloque con el nombre del registro precedido por `^N^X`.
3. Lectura de los bloques de datos.
4. Comprobación del cierre del registro, al detectarse que el último bloque contiene el nombre precedido de `^N^X`.

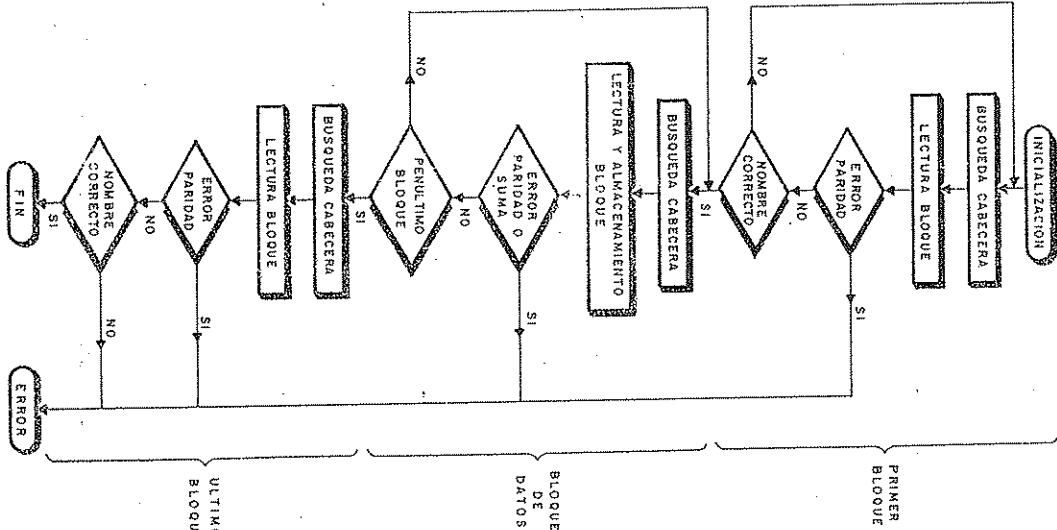


Fig. 7-21. Ordinograma funcional diseñado para proceder a la lectura de la cinta de papel perforado.

7.1.1. INTERRUPCIONES

Las interrupciones pueden considerarse como bifurcaciones externas al programa, provocadas por señales que provienen del exterior de la UCP.

La finalidad de una interrupción es reclamar la atención de la UCP sobre algún acontecimiento o hecho externo, pidiendo que se ejecute un programa específico que trate adecuadamente ese acontecimiento.

Un ejemplo típico es la interrupción que provocan los controladores de DMA cuando terminan la transmisión del bloque de datos pedido. De esta forma, ponen en conocimiento de los programas que tratan la entrada y salida del computador, que han acabado y están disponibles para realizar otra operación.

A veces, las interrupciones las origina la propia UCP, en el caso de las excepciones o errores, cuando se produce un acontecimiento anómalo en su funcionamiento. Como el tratamiento es idéntico, se hace referencia a una interrupción en su sentido más amplio.

En una interrupción se suceden dos etapas diferenciadas:

1. Diálogo de señales, necesario para que la UCP acepte la interrupción y haga la bifurcación al programa adecuado.
2. Tratamiento de la interrupción, consistente en la ejecución del programa correspondiente.

En primer lugar se analiza el tema de la aceptación de la interrupción y hay que establecer un mecanismo para determinar la dirección de bifurcación que es el inicio de la rutina de tratamiento. Existen dos alternativas para definir esta dirección.

- a) Dirección fija, especificada por los circuitos electrónicos de la UCP.

b) La dirección es suministrada por el dispositivo que provoca la interrupción.

Esta solución es mucho más flexible que la precedente, pues permite que cada periférico tenga su rutina de tratamiento propia y ubicada donde se deseé. Sin embargo, complica el diálogo de comienzo de la interrupción y el diseño de la UCP y los controladores de los periféricos periféricos.

Se suele emplear una solución intermedia, que consiste en que el dispositivo suministre, no una dirección completa, sino una dirección codificada en unos pocos bits, a los que se añaden otros fijos que completan la dirección.

La Unidad de Control acepta las interrupciones al final de cada instrucción. Esto significa que al final de todas las instrucciones existe una interrogación implícita sobre las señales de interrupción; si la interrogación es contestada afirmativamente, se produce la bifurcación y se acepta la interrupción.

Hay que hacer una consideración importante al diseñar los mecanismos de aceptación y tratamiento de las interrupciones y consiste en la posibilidad de que

aparezca un *bucle infinito* en la bifurcación que produce la interrupción. Así, por ejemplo, si un dispositivo activa su señal de petición de interrupción, al finalizar la instrucción en curso se realiza una bifurcación a la primera instrucción I_1 de tratamiento. Pero, si el dispositivo no desactiva dicha señal de petición, entonces, terminada la ejecución de I_1 , se volvería a aceptar la interrupción, con lo que, de nuevo, se bifurcaría a la instrucción I_1 .

Para evitar este tipo de problemas, la aceptación de una interrupción convierte a un mecanismo de inhibición de las interrupciones. Por ejemplo, cuando se acepta una interrupción, se activa un bistable que impide que se acepte otra. Luego se desactiva el bistable y, nuevamente, se permiten las interrupciones.

Como se prevé que se conecten a los computadores muchos dispositivos que produzcan interrupciones, se les dota de un sistema flexible de diálogo. Hay tres formas básicas de conexión:

1. Línea de interrupción única.
2. Líneas de interrupción y aceptación.
3. Interrupciones vectorizadas.

7.12. LINEA DE INTERRUPCIÓN UNICA

El computador dispone de una línea de interrupción \overline{INT} , que sirve para que cualquier dispositivo solicite una interrupción. Esta línea se configura en forma de colector abierto (OR cableada), con lo que cualquier dispositivo puede poner en ella un 0 para solicitar la interrupción sin problemas electrónicos de interferencias con los demás dispositivos.

Dado que no se dispone de otra señal, ni de aceptación por parte de la UCP, ni de dirección por parte de los dispositivos, este procedimiento obliga a ubicar en una posición fija de la memoria a la rutina de tratamiento de la interrupción y que ella se comience investigando cuál ha sido el dispositivo que interrumpió, para bifurcar a su correspondiente rutina.

Para evitar el bucle infinito, la UCP, al mismo tiempo que acepta la interrupción, inhibe a todas ellas, desactivando uno de sus bistables de estado (I_1). Luego, la rutina de tratamiento interroga a los dispositivos para averiguar cuál ha sido el periférico. El mecanismo de interrogación exige que los dispositivos tengan un bistable de estado que se ponga a 1, cuando solicite interrupción. Una vez detectado el periférico de interrupción, se borra su bistable, con lo que se evita el bucle infinito. Seguidamente, si procede, se vuelve a dar permiso a las interrupciones, activando en la UCP el bit de estado correspondiente.

La técnica de ir interrogando sucesivamente a los distintos periféricos para conocer al peticionario, recibe el nombre de Polling en inglés; que, a veces, se traduce por muestreo y también por escrutinio y exploración.

La asignación de prioridades se realiza siguiendo el orden en que la rutina de tamiento analiza los biestables de interrupción de los dispositivos peticionarios.

La figura 7-22 representa el esquema de conexión mediante el procedimiento de línea de interrupción única y en ella se han agrupado los bistables de interrupción de varios dispositivos en la misma puerta de control, representada por un buffer triestado, de manera que se logre una aceleración en el proceso de interrupción puesto que con una sola instrucción de entrada se obtiene el estado de los dispositivos. El bistable de inhibición de interrupciones de la UCP (\overline{INT}) puesto a 1 impide el paso de la señal \overline{INT} .

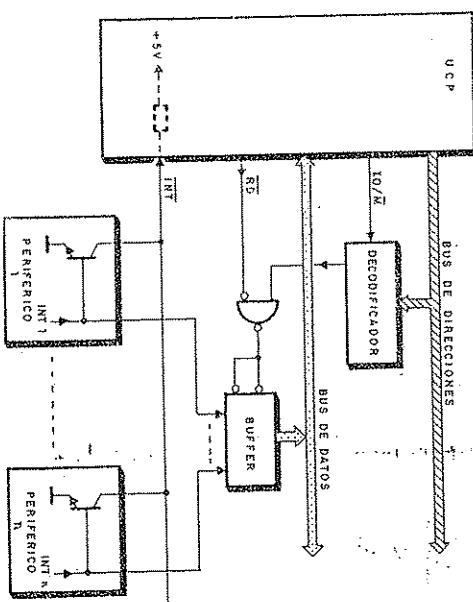


Fig. 7-22. Esquema que presenta una configuración de diferentes periféricos conectados al computador mediante una línea de interrupción única.

7.13. LINEAS DE INTERRUPCION Y ACEPTACION

310/ Unidad de entrada y salida
Esta solución puede considerarse como una extensión del procedimiento anterior, al que se ha añadido una línea de aceptación de la interrupción INTA.

que el periférico no tiene que ser capaz de introducir información en el bus de direcciones, simplificando su diseño.

Con este procedimiento, son los propios periféricos que provocan la interrupción, los que suministran la dirección de bifurcación en la que se encuentra la rutina de tratamiento.

Las alternativas clásicas para direccionar el comienzo de la rutina de tratamiento de la interrupción son:

Direccionalamiento absoluto. El periférico peticionario proporciona al bus de direcciones la dirección completa del comienzo de la rutina de tratamiento.

Direccionamiento relativo. El periférico sólo envía parte de la dirección, que es completada por la UCP, o bien añadiendo más bits, o bien sumando una

cantidad. Esta información se suele enviar a través del bus de datos, por lo que el periférico no tiene que ser capaz de introducir información en el bus de direcciones, simplificando su diseño.

Direcccionamiento relativo indirecto. En este caso, la dirección que envía el periférico es la posición relativa en una tabla de direcciones de rutinas se tra-

Unidad c.: entrada y salida / 311
tamiento de interrupciones. Mediante un direccionamiento indirecto, se accede a la entrada correspondiente de la tabla y se bifurca a la rutina deseada.

Empleando un sistema de encadenamiento, se pueden establecer las prioridades de los dispositivos externos, reconociendo ellos mismos, cuando la UCP les acepta, la petición de interrupción, por lo que automáticamente deben desactivar su peti-

La rutina de tratamiento comienza interrogando a los dispositivos para determinar si el paciente ha dejado de dormir en la noche o ha tenido que levantarse a la hora de la noche.

La secuencia de funcionamiento con este sistema se descompone en los siguientes pasos:

1. El dispositivo peticonario de la interrupción pone un 0 en \overline{INT} .

2. La UCP acepta la interrupción y envía un 0 por la línea TINTA e inhibe a las interrupciones.

3. El dispositivo de mayor prioridad que desea interrumpir, al reconocer INTA, ojita su 0 de la linea INT y bone un 1 en su bistable de interrupcion.

4. La rutina de tratamiento lee los biestables y detecta al peticionario. Acto se-

guido, le envía una instrucción de salida para borrar el susodicho biesstable.

7.14. INTERRUPCIONES VECTORIZADAS

Para realizar la interrupción vectorizada se necesitan las siguientes señales:

- $\overline{\text{INT}}$ (petición de interrupción).
- $\overline{\text{INTA}}$ (reconocimiento y aceptación de interrupción).
- Dirección (comienzo de la rutina de tratamiento).

Las señales de dirección se pueden enviar por un bus especial, por el bus de datos, o por el de direcciones. El cronograma correspondiente a las señales de interrupción vectorizada se muestra en la figura 7-23, en la que la señal $\overline{\text{TEMP}}$ sirve para indicar al periférico que se ha quedado con la interrupción, en qué momento debe poner la información de dirección en el bus empleado para este menester.

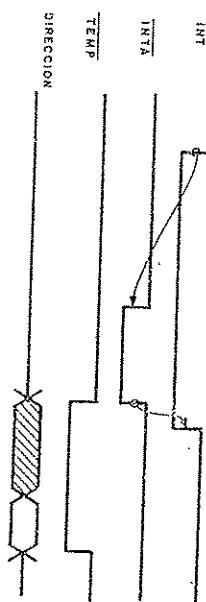


Fig. 7-23. Cronograma de aceptación de una interrupción vectorizada.

La gran ventaja de las interrupciones vectorizadas es que acceden directamente a la rutina que las debe tratar, evitando así el tiempo empleado en otros métodos en la exploración o polling, exigida en la identificación del periférico peticionario de la interrupción.

7.15. PRIORIDADES Y NIVELES DE INTERRUPCIÓN

En todo sistema con más de un dispositivo con capacidad de interrupción, hay que establecer un mecanismo de prioridad para atender las posibles interrupciones. Este mecanismo tiene que resolver dos funciones importantes:

1. Caso de existir peticiones simultáneas, determinar el dispositivo al que se concede la interrupción.
 2. En cada programa de atención a una interrupción hay que establecer qué tipos de interrupciones debe tolerar.
- Cuando se ejecuta un programa de atención a una interrupción, se pueden producir las siguientes alternativas:

312 / Unidad de entrada y salida

- a) Que se ejecute sin posibilidad de admitir interrupción alguna.
- b) Que pueda ser interrumpida por ciertos tipos de periféricos.
- c) Que pueda ser interrumpida por todos los periféricos.
- d) Que el programa tenga fases con distintas alternativas de las tres que se han comentado.

El primer y tercer caso son variantes del segundo, pudiéndose realizar de forma muy sencilla, sin más que contar con el biesstable de inhibición general de interrupciones $BGII$, que, cuando está en estado de inhibición, el programa no admite tipo de interrupción alguno, mientras que, si está desinhibido, acepta todas. Como la UCP inicia cualquier interrupción desactivando todas ellas, la rutina de tratamiento comienza según el criterio de la alternativa a), luego, la propia rutina, si así lo contempla, modificará esta situación variando el estado de $BGII$ (ejecutando la instrucción Disable Interrupt, DI).

En el segundo caso hay que añadir ciertos circuitos, que permiten inhibir y desinhibir, selectivamente, a los distintos periféricos. Se suele adoptar una estructura de gestión de prioridades híbrida, a la que se añade un conjunto de biestables de inhibición, tal y como se muestra en la figura 7-24.

Como ocurre que, en ocasiones, se conectan más periféricos al computador que niveles existen, en cada nivel se establece una prioridad interna, por ejemplo, mediante encadenamiento, lo cual permite conectar un gran número de periféricos.

Para la inhibición de las diversas interrupciones se recurre a los tres sistemas siguientes:

1. Se hace uso del biesstable $BGII$.
2. Se asocia a cada nivel de interrupción un biesstable, llamado *máscara*, que permite inhibir (enmascarar) cada nivel, de forma selectiva.
3. Se usa un registro, común a todos los niveles, que guarda el número del menor nivel que puede interrumpir. Dicho nivel y los superiores son los que pueden ocasionar interrupción.

A menudo se dispone de los dos últimos mecanismos, porque así se puede asignar a cada programa o proceso un nivel de prioridad, que no puede automodificar, pero, si en determinadas fases se necesitase eliminar algunas interrupciones, podría emplearse el mecanismo de máscara.

La figura 7-24 contempla un caso con una serie de niveles de prioridad, cada uno asociado a la pareja de señales $\overline{\text{INT}}(i)$ e $\overline{\text{INTA}}(i)$ y con los tres mecanismos de inhibición antes mencionados. Los registros de máscara (R_M) y de nivel (R_N), y el biesstable $BGII$ representan la información de prioridad de la máquina. Estos elementos

pueden leerse y escribirse con instrucciones especiales, para modificarlos de acuerdo a las necesidades. La señal $\overline{INT(i)}$ atraviesa el doble filtro impuesto por su correspondiente bit de máscara y por la condición $i \geq RN$. El decodificador convierte el código de registro RN en los correspondientes ceros, para filtrar las peticiones de interrupción. La señal que verdaderamente provoca la interrupción es \overline{INT} , que se genera si alguna señal $\overline{INT(i)}$ consigue atravesar su puerta AND y si, además, el bistable $BG11$ está a 1. El codificador genera el valor i de la señal $\overline{INT(i)}$ de mayor nivel que atraviesa el filtro. Esta información se usa en el multiplexor que encamina la señal \overline{INTA} a su nivel correspondiente y en la UCP.

Se intenta establecer cierto paralelismo entre el funcionamiento de la UCP y los periféricos, de forma que los controladores de los periféricos se encarguen de realizar las transferencias de E/S, con la menor intervención posible de la UCP. Esta actuación, que en un computador monolítico puede no estar justificada, puesto que si la UCP nada va a hacer durante las fases de E/S, resulta más económico que se encargue ella misma del control de los periféricos, esté plenamente justificada en el caso de la multiprogramación, ya que puede ocuparse de ejecutar otros procesos, mientras que los controladores de los periféricos se encargan de traer y llevar información.

Existen todo un abanico de posibilidades que van desde obligar a la UCP a encargarse de todo, mediante E/S programadas, hasta descargar toda la responsabilidad en los controladores, que, en el caso límite, funcionan como computadores de propósito específico.

Para comprender más fácilmente todas las alternativas, se comienza exponiendo un ejemplo práctico sobre el que se explican.

7.16. 1. Descripción de una operación de lectura en un disquete

Se propone la realización de una operación de lectura de un sector de 128 bytes de un disquete de 8 pulgadas que gira a 360 rpm. Se requieren las siguientes fases:

- 1) Hay que comprobar el estado de la unidad de disquete:
 - a) Comprobar si está encendida o conectada la unidad, lo que supone leer y analizar la palabra de estado que define la situación del periférico.
 - b) Comprobar si hay un disquete insertado.
 - c) Comprobar que la unidad está desocupada.

Finalmente, debe tenerse en cuenta que la asignación de prioridad a una interrupción depende de dos factores: la importancia de su misión y el tiempo de respuesta, puesto que, cuanto menor sea el tiempo de respuesta de un periférico, mayor será su prioridad.

7.16. ORGANIZACION DE LAS OPERACIONES DE ENTRADA Y SALIDA

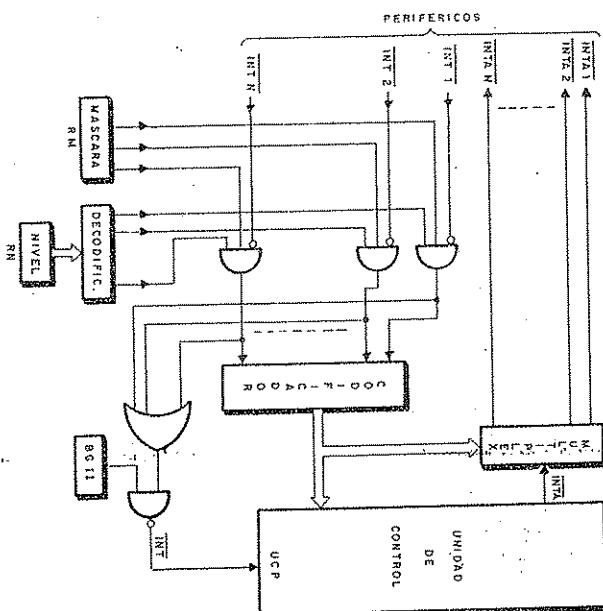


Fig. 7-24. Unidad de aceptación de interrupciones.

- Unidad de entrada y salida / 315
- 2) Hay que enviar una orden para posicionar el brazo en la pista que contenga el sector buscado. Si se conoce la posición actual, se transmiten tantos pulsos como pistas haya que avanzar o retroceder. Si no se conoce la posición del brazo, hay que llevarlo hasta la pista cero y, al detectarse esta posición, se retrocede hasta la pista deseada. La detección de la pista se efectúa con un fotodiodo que se activa al alcanzar la posición final del brazo. Se considera como tiempo medio de acceso a la pista, 100 ms.

3.) Activación de la cabeza, hay que activar el electroimán de la cabeza lectora para que se apoye sobre el diskette. Esta operación se considera que consume un tiempo medio de 50 ms.

4.) Activada la cabeza, hay que detectar el comienzo de la pista. Esta función se realiza mediante un bucle que comprueba el bit de estado de comienzo de pista, bit que se activa cuando el fotodiodo detecta el agujerito de comienzo de pista. El tiempo medio de esta fase es de 1/2 revolución, lo que supone 83,3 ms. Este tiempo no existe en las unidades de disco modernas.

5.) Hay que ir reconociendo los bytes leídos, detectando las cabeceras de los sectores, hasta encontrar el sector deseado. El tiempo medio de esta función es de 1/2 revolución, o sea, 83,3 ms. En esta operación los bits que se leen en serie hay que convertirlos a bytes.

6.) Localizado el sector, hay que ir aceptando sus bytes, introduciéndolos en la zona de memoria deseada y efectuando la comprobación del código detector de errores. La velocidad de transferencia se puede calcular sabiendo que el disco gira a 360 rpm y que una pista tiene 5.208 bytes, lo que hace corresponder a cada byte un tiempo de 32 μ s. Se dispone, por tanto, de 32 microsegundos para realizar el bucle de aceptación de los bytes, lo que da lugar a ir comprobando su paridad e ir calculando el código polinómico de detección de error en el sector.

Las operaciones comentadas para la lectura del diskette pueden llevarse a cabo siguiendo diversas posibilidades de distribución del trabajo entre el controlador del diskette y la UCP, que pasamos a describir.

7.17. ENTRADA/SALIDA PROGRAMADA

Considerando las velocidades de trabajo de la unidad de diskette, cualquier computador moderno es capaz de soportar todas las funciones mediante E/S programadas.

En el ejemplo anterior, los tiempos que dispone la UCP para efectuar cada fase son los siguientes en ms:

t ₁ .—Comprobación del estado	0,1
t ₂ .—Posicionamiento del brazo (39 interrupciones)	3,9
t ₃ .—Activación de la cabeza (1 interrupción)	0,1
t ₄ .—Detección del comienzo de pista (1 interrupción)	0,1
t ₅ .—Detección de la cabecera deseada	83,3
t ₆ .—Lectura del sector	4,2
TOTAL	91,7

La fase de comprobación del estado tomará el mismo tiempo t₁, que en el caso anterior. El posicionamiento del brazo ocupará, por término medio, 39 movimientos, puesto que hay un total de 77 pistas. Para realizar esta función y para activar la cabeza, se supone que existe un temporizador externo que interrumpe con un retardo fijo programado. Para la detección de la cabecera y lectura del sector se han considerado los tiempos totales, puesto que no es factible emplear interrupciones. En efecto, el tiempo entre éstas sería menor que el tiempo tardado en atenderlas, por lo que se perderían bytes, debiendo realizarse estas fases con E/S programadas, de tipo convencional.

El ahorro de tiempo de UCP es considerable, obteniéndose un consumo de 0,7 ms por byte.

Para determinar t₁ se ha supuesto que la comprobación del estado requiere ejecutar un pequeño programa de unas 100 instrucciones máquina. Para los tiempos t₂ a t₅ se han empleado los tiempos medios. Finalmente, para t₆ se ha tomado el tiempo de lectura de 131 bytes del sector, que, multiplicado por 32 μ s, da unos 4,2 ms.

El tiempo total de la operación es del orden de 321 ms, en el que la UCP está ocupada. El tiempo medio que la UCP emplea por cada byte será de $321/128 = 2,5$ ms.

7.18. MEDIANTE INTERRUPCIONES

Aquí se considera que el tratamiento de las E/S se realiza mediante interrupciones, pero la UCP efectúa todas las funciones de la operación. Habrá que calcular el número de interrupciones que ha de atender el procesador, así como el tiempo medio de la rutina de interrupción de 0,1 ms, obteniéndose la siguiente tabla de tiempo, en ms:

t ₁ .—Comprobación del estado	0,1
t ₂ .—Posicionamiento del brazo (39 interrupciones)	3,9
t ₃ .—Activación de la cabeza (1 interrupción)	0,1
t ₄ .—Detección del comienzo de pista (1 interrupción)	0,1
t ₅ .—Detección de la cabecera deseada	83,3
t ₆ .—Lectura del sector	4,2
TOTAL	91,7

7.19. INTERRUPCIONES CON CONTROLADOR INTELIGENTE

Cuando la unidad de diskette está dotada de un controlador inteligente, éste es capaz de realizar las funciones de posicionar el brazo, activar la cabeza y buscar el sector deseado. La UCP, solamente, deberá lanzar la operación de lectura, enviando al controlador, mediante E/S programadas, la identificación del sector, su tamaño y la orden de lectura. Todo ello puede hacerse con un programa que, primero comprueba el estado de la unidad y luego envía la información necesaria al controlador. A continuación, éste posiciona el brazo, activa la cabeza y detecta el comienzo de la pista, pasando inmediatamente a la lectura de las cabezas, hasta localizar el sector deseado. En ese momento, interrumpe a la UCP que, a partir de entonces, se encarga del resto de la operación de lectura, de forma idéntica a como lo hizo en los casos anteriores.

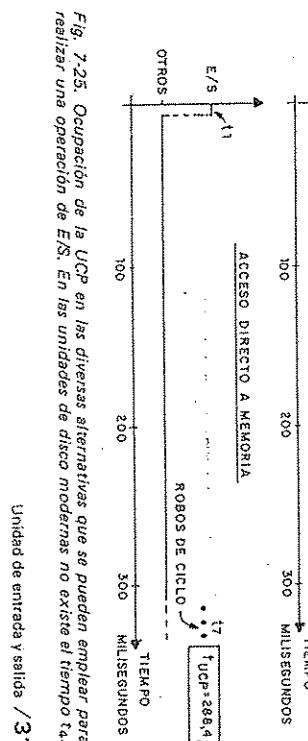
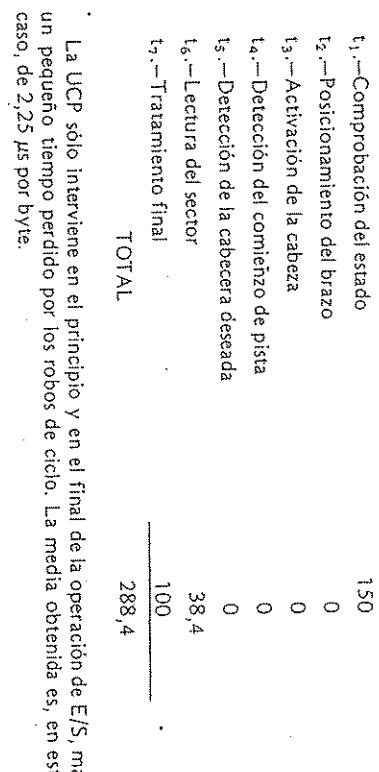
Los tiempos empleados con este procedimiento son los siguientes, en ms:

t_1 .—Comprobación de estado	0,15
t_2 .—Posicionamiento del brazo	0
t_3 .—Activación de la cabeza	0
t_4 .—Detección del comienzo de pista	0
t_5 .—Detección de la cabecera deseada	0
t_6 .—Lectura del sector	38,4
t_7 .—Tratamiento final	100
TOTAL	288,4

La reducción del tiempo total es muy importante si se usa un controlador inteligente, resultando una media de 33,6 μ s por byte.

7.20. ROBO DE CICLO

En este caso, además de realizar las funciones anteriores, el controlador del diskette envía el sector deseado empleando una serie de accesos directos a memoria, por *robo de ciclo*. En consecuencia, el tiempo t_6 queda reducido a 128 robos de ciclo. Asignando un tiempo de 300 ns por robo, queda $t_6 = 38,4 \mu$ s. Finalizada la transferencia del sector, el controlador de la unidad de diskette interrumpe a la UCP para comunicar que ha terminado la operación e informar de posibles errores encontrados. Suponiendo que el tratamiento de esta interrupción conste de unas 100 microinstrucciones, los tiempos de operación de la UCP tomarán los siguientes valores en μ s:



En la figura 7-25 se representan gráficamente las cuatro alternativas estudiadas. Aunque el dibujo no está hecho a escala, ofrece una idea clara de cómo se reparte el tiempo de ocupación de la UCP en cada procedimiento.

7.2.1. CANALES DE ENTRADA/SALIDA

Se define a un *canal* como una unidad de E/S automática, que funciona por acceso directo a memoria. En este aspecto, se considera un canal al controlador necesario para realizar la lectura de la unidad de diskette por robo de ciclo.

No obstante, se aplica el término de "canal" a algo más restrictivo, que cumplía las condiciones siguientes:

1. Acceso directo a memoria, preferentemente por puerta propia.
 2. Comunicación con la UCP a través de memoria y no por E/S programada.
- El mecanismo de comunicación suele organizarse tal como se muestra en la figura 7-26. Se dispone de un puntero PCANAL, ubicado, generalmente, en una posición fija de memoria, que apunta a una zona donde se encuentran los datos de una operación de E/S. Estos datos, llamados *DATES*, son:
- La dirección de la memoria.
 - La dirección del periférico.
 - Número de bytes.
 - Tipo de operación.

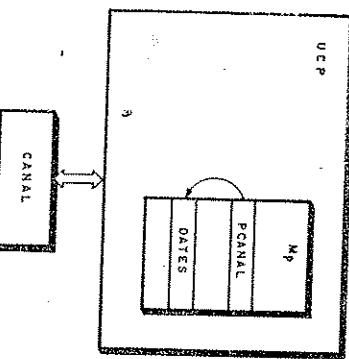


Fig. 7-26. Comunicación de la UCP con el canal de E/S.

320/ Unidad de entrada y salida

Cuando la UCP tiene preparados estos datos, envía, con una instrucción de E/S, una señal al canal, para indicarle que tiene que efectuar una operación. El canal accede directamente a memoria y, a través de PCANAL, obtiene los datos *DATES* de la operación. Finalizada ésta, el canal inserta en la correspondiente zona de *DATES*, el resumen de la operación y provoca una interrupción para comunicar que ya ha terminado.

Como el computador dispone de varios canales, hay que hablar de una *tabla de punteros PCANAL*, o de una *tabla de datos *DATES**.

El procedimiento acorta tiempos en la duración de los programas de inicialización y finalización de la operación de E/S, pero no los elimina. Para mejorar esta situación se utiliza el *encadenamiento de operaciones de E/S*.

Para soportar este encadenamiento, sólo hay que colocar al final de la zona *DATES*, o bien una nueva zona *DATES*, o bien una indicación de fin. El canal, al completar la operación correspondiente a *DATES* 1, introduce la información de estado en *DATES* 1 y lee *DATES* 2 para seguir trabajando. El proceso se repite hasta que el canal encuentra la indicación de fin.

En la figura 7-27 se muestra la tabla de punteros PCANAL con el conjunto de *DATES* de un canal. Cada *DATES* se compone de la siguiente información:

- a) Orden o tipo de operación a realizar.
- b) Dirección absoluta de la memoria principal.
- c) Número de bytes.
- d) Dirección en el periférico.
- e) Bit de interrupción, que, si está activado, interrumpe el canal.
- f) Bit de encadenamiento, que, cuando está activo, el canal toma la siguiente orden de *DATES* y prosigue.
- g) Estado. Esta información la coloca el canal al acabar la operación de E/S.

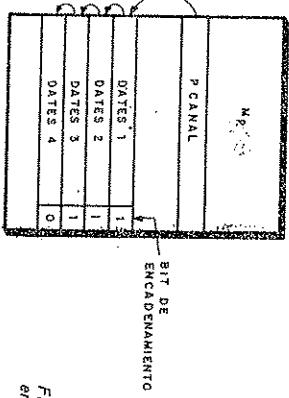


Fig. 7-27. Representación gráfica de encadenamiento de órdenes de canal.

Unidad de entrada y salida / 321

Al conjunto de DATÉS se le denomina *programa canal* que es generado, dinámicamente, por la UCP a medida que van surgiendo necesidades de E/S.

Los canales se suelen estructurar en base a un bus común, al que se conectan diversos periféricos. Puesto que la entrada de un canal es completa y cara, deberá ser compartida por varios periféricos, ya que éstos no suelen operar de manera concurrente.

Sin embargo, esta estructura presenta el inconveniente de que, en un momento dado, pueden existir canales parados y canales con una lista de espera, lo que redundaría en una pobre eficacia en el sistema de E/S. Para evitar esta situación, algunos fabricantes han introducido el concepto de *canal flotante*, que se puede asignar a cualquiera de los periféricos. Así, las peticiones de operaciones de E/S se asignan, no a un canal determinado, sino a un conjunto de canales, que se van repartiendo el trabajo.

Un *canal simple* está diseñado para periféricos de alta velocidad de transferencia, tales como discos y cintas. Sólo admite una operación en el tiempo.

Un *canal multiplexor* está diseñado para periféricos lentos, tales como impresoras, lectoras de documentos y terminales. Permite realizar, simultáneamente, varias operaciones de E/S, puesto que, en su funcionamiento, se divide en varios subcanales que trabajan de forma cíclica, atendiendo cada uno a un periférico.

7.22. EL SISTEMA OPERATIVO Y LAS ENTRADAS Y SALIDAS

Cuando se desea tratar los datos de un cierto fichero, mediante un programa de alto nivel, el programador se limita a escribir una sentencia de apertura para el mencionado fichero Y, seguidamente, programará instrucciones de lectura (READ, ACCEP, ...) o de escritura (WRITE, PRINT, ...) sobre dicho fichero.

Las sentencias de alto nivel para realizar operaciones de E/S se compilan como llamadas al sistema operativo, siendo éste el responsable de llevar a cabo estas operaciones. Para ello, el sistema operativo ha de soportar las siguientes funciones:

1. Manejar los directorios de los periféricos. En concreto, ha de poder convertir el nombre simbólico de un fichero en las direcciones físicas de los distintos sectores que lo almacenan, a través del correspondiente directorio. Ha de poder insertar en el directorio un nuevo fichero Y, también, borrarlo.
2. Generar órdenes a los periféricos, elaborando, en su caso, los programas de canal.
3. Gobernar los periféricos, enviándoles las órdenes pertinentes y atendiendo sus necesidades y problemas.

3.22 / Unidad de entrada y salida

4. Ordenar las peticiones de E/S, manteniendo las colas necesarias y generando los mensajes a los programas peticionarios.

5. Tratar las situaciones de error, repitiendo las operaciones y/o avisando de lo ocurrido.

6. Asignar las zonas de memoria buffer necesarias para estas operaciones, así como realizar una posible conversión de códigos y la agrupación de bytes en palabras y viceversa.

Por todo lo expuesto, la ejecución de una sentencia de E/S de alto nivel exige del sistema operativo una complicada trama de acciones, de las cuales una pequeña parte corresponden al control de las operaciones físicas de E/S.

7.23. CIRCUITOS INTEGRADOS PARA EL DISEÑO DE E/S

Hay una gran variedad de circuitos integrados, de alto nivel de integración, que sirven para simplificar el diseño de la entrada y salida en los sistemas basados en microprocesador. Suelen estar diseñados para microprocesadores específicos por lo que se habla de *familias* de circuitos auxiliares.

Se ofrece una muestra representativa de los distintos tipos de circuitos que se encuentran en estas familias, haciendo referencia a la del microprocesador 8085, por ser una de las más características.

- a) 8255. *Puerta de E/S*. Este circuito integrado permite manejar tres puertas de E/S de 8 bits cada una.
- b) 8257. *Controlador de acceso a memoria*. Sirve para conectar 4 dispositivos a un microprocesador de 8 bits, usando la técnica de "modo de ciclo".
- c) 8259. *Controlador de interrupciones*. Permite establecer hasta 8 niveles de prioridad en las interrupciones del sistema.
- d) 8271. *Controlador de diskette*. Controla hasta 4 diskettes de 8" o de 5 1/4", de acuerdo al formato IBM 3740, realizando todas las funciones necesarias de lectura, escritura y control de las unidades de diskette.

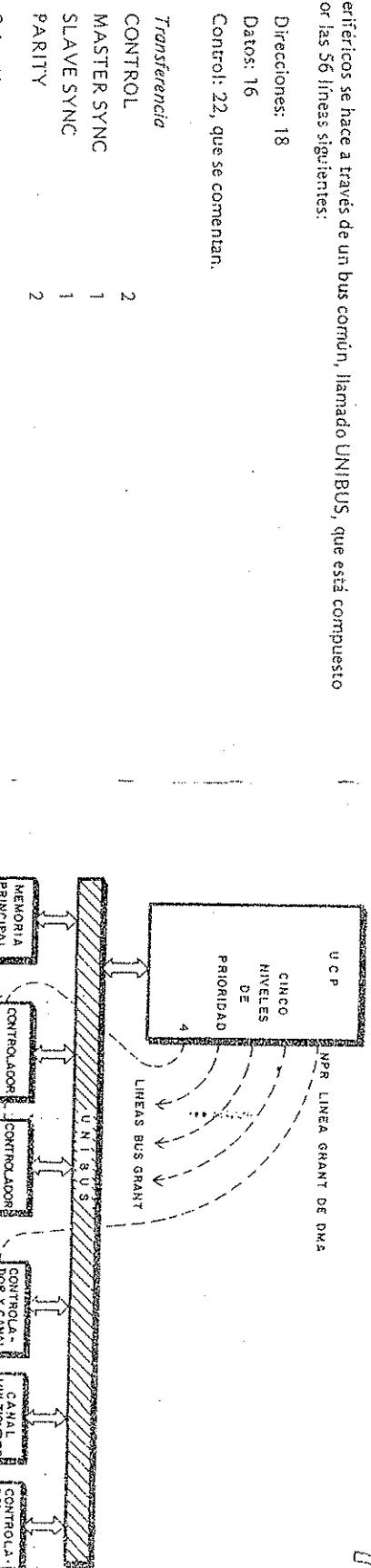
7.24. EJEMPLOS DE ORGANIZACION DE E/S EN MAQUINAS COMERCIALES

7.24.1. PDP-11 Y VAX-11

El microcomputador PDP-11 emplea un mapa común para E/S y memoria principal, asignando las 4 K primeras posiciones a E/S. La comunicación física con los

periféricos se hace a través de un bus común, llamado UNIBUS, que está compuesto por las 56 líneas siguientes:

Direcciones: 16
Control: 22, que se comentan.



Transferencia
CONTROL 2
MASTER SYNC 1
SLAVE SYNC 1
PARITY 2
Selección
BUS REQUEST 5
BUS GRANT 5
SELECT ACK 1
INTERRUPT 1
BUS BUSY 1
INITIALIZATION 1

Fallo de alimentación
AC LOW 1
DC LOW 1

El PDP-11 permite establecer hasta 5 daisy-chain en base a las 5 parejas de señales de BUS REQUEST y BUS GRANT, pero todas ellas realizadas sobre un único bus, como se refleja en la figura 7-28.

La transmisión en el UNIBUS es asíncrona, empleándose las señales MASTER SYNC y SLAVE SYNC para soportar el interbloqueo o diálogo correspondiente, mientras que las señales CONTROL sirven para especificar si se realiza una lectura o una escritura. La figura 7-29 muestra un cronograma sobre la transferencia en el UNIBUS.

Una ampliación del sistema de E/S del PDP-11 se ha usado en la familia de miniordenadores VAX-11, tal como se aprecia en la figura 7-30.

A continuación se propone una somera descripción de los elementos reflejados en la figura 7-30.

324 / Unidad de entrada y salida

Fig. 7-28. Estructura general de las E/S en el minicomputador PDP-11.

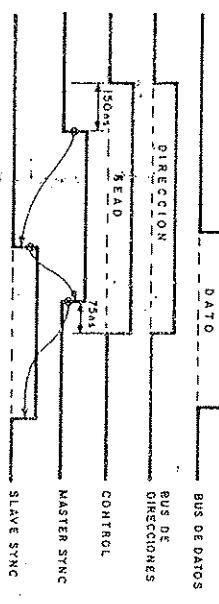


Fig. 7-29. Cronograma de una transferencia (lectura) sobre las líneas del UNIBUS en el PDP-11.

Consta de 54 líneas, con grupos sincrónico y asíncrono para la transferencia de datos. Trabaja a 2 MB/s.

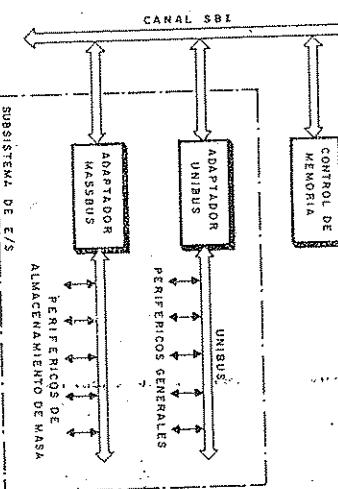
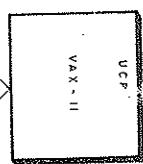


Fig. 7-30. Organización general del subsistema de E/S de la familia de minicomputadoras VAX-11.

7.24.1.1. SBI

Es un canal bidireccional que enlaza a la UCP con los controladores de memoria y los adaptadores a los buses UNIBUS y MASSBUS.

Consta de 64 líneas, 32 de las cuales se destinan a datos. Trabaja a 13 MB/s.

7.24.1.2. UNIBUS

Tiene una estructura similar que la comentada en el PDP-11 y conecta a la mayoría de los dispositivos y periféricos de baja velocidad, así como los elementos especiales del usuario. Trabaja a 1,5 MB/s.

7.24.1.3. MASSBUS

Puede soportar hasta 4 adaptadores y cada uno a 8 controladores de dispositivos de almacenamiento.

3.2.5 / Unidad de entrada y salida

7.24.2. IBM 370

Las máquinas IBM 360 y 370 organizan su E/S en base a canales, como se muestra en la figura 7-31. Cada canal comunica con los periféricos mediante un bus que consta de las siguientes 34 líneas unidireccionales:

BUS IN	8
BUS OUT	8
CONTROL IN	7
CONTROL OUT	9
SELECT OUT	1
SELECT IN	1

Las líneas BUS IN Y BUS OUT se usan para enviar datos, direcciones de periféricos y el estado de éstos. Puesto que hay 8 bits de dirección, se podrían conectar hasta 256 periféricos por canal, aunque no es frecuente pasar de 8. Las señales SELECT IN Y SELECT OUT permiten configurar un daisy-chain, de forma que las prioridades dentro del canal se resuelven por encadenamiento. Las prioridades entre canales las resuelve un gestor centralizado.

Los canales pueden ser simples, multiplexados a nivel byte y multiplexados a nivel bloque. Estos últimos optimizan las transferencias, cuando se producen varias solicitudes simultáneas.

Los canales se comunican con los periféricos a nivel byte, pero con la UCP a nivel de palabras de 64 bits, con lo que se reduce el número de accesos a memoria. Los canales simples y multiplexados por bloque alcanzan una velocidad de transferencia de 12 Mbit/s, mientras que el multiplexador de bytes sólo llega a 1,5 Mbit/s, debido al tiempo empleado en la demultiplexación.

La arquitectura del IBM 360/370 tiene un sistema vectorizado de interrupciones que soporta 32 tipos de interrupciones, que pueden enmascararse con 20 bits de máscara. De estas interrupciones y bits de máscara se emplean 7 para E/S. Las interrupciones se agrupan en 5 categorías de prioridad, que se citan en orden de nivel decreciente:

- Errores de máquina.
- Llamadas a supervisor.

- A anomalías de programa (desbordamientos, dirección ilegal, código OP prohibido...).

- Externas.
- Entrada/Salida.

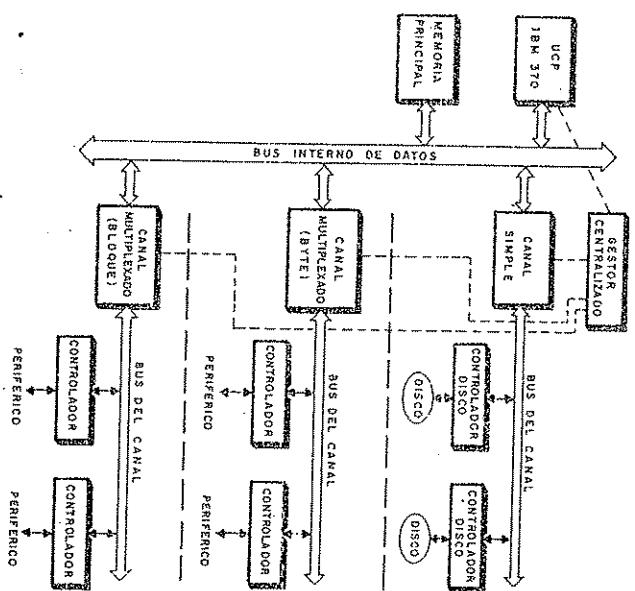


Fig. 7-31. Estructura general de las E/S del IBM 3670/360.

Cada clase de interrupción tiene asociadas dos palabras de 64 bits en posiciones fijas de la memoria principal, que reciben el nombre de OLD PSW y NEW PSW. La figura 7-32 presenta la OLD PSW y la NEW PSW de las interrupciones de E/S, mientras que el formato de la *palabra de estado PSW*, se ofrece en la figura 7-33, donde puede observarse que existe un campo de 16 bits para almacenar el código de interrupción. Este código no es más que el vector que ha de introducir el dispositivo que interrumpe y sirve para identificar su rutina de tratamiento. Cuando se genera una interrupción, la PSW de la UCP, que es la palabra de estado actual, debe ser salvaguardada en la OLD PSW y sustituida por la NEW PSW, que contiene el vector de interrupción.

328 / Unidad de entrada y salida

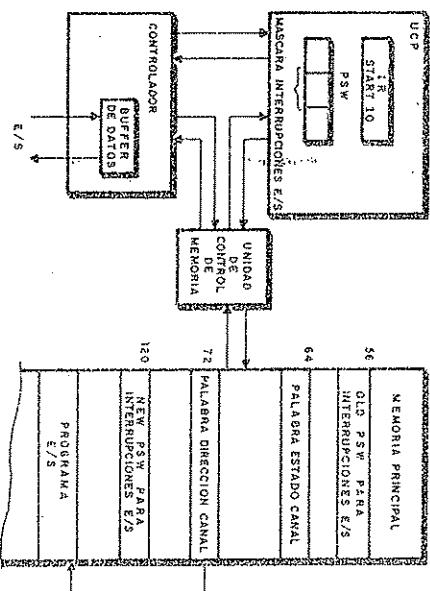


Fig. 7-32. Comunicación entre la UCP y los canales del IBM 370/360.

SISTEMA	KEY	INTERRUPCIÓN	CODIGO LONGITUD		CONDICION
			Z	Y	
0	7 6	11 12	31	31	35 36 39 40
					43

Fig. 7-33. Formato de la palabra de estado PSW del IBM 370/360.

La programación del canal se hace mediante órdenes de 64 bits de longitud, cuyo formato y tipos se exponen en la figura 7-34. Los códigos Z que allí se indican dependen de cada tipo de periférico y sirven para especificar las condiciones de la operación.

CÓDIGO	DIRECCIÓN	CONDICIÓN	ORDEN		NÚMERO BYTES
			Z	Y	
0	7 8	31 32	35 37	47 48	63

Fig. 7-34. Formato de la orden de programación de un canal del IBM 370/360.

junto a los códigos de las órdenes.

Unidad de entrada y salida / 329

EJERCICIOS

Ejercicio 7.1

Indicar, detalladamente, los pasos necesarios para efectuar una operación de lectura de un disquete con indicación de los tiempos parciales en cada paso, suponiendo E/S programadas.

Ejercicio 7.2

Comentar los problemas que se producen en la gestión de prioridades. Soluciones concretas para cada problema. Formas de atender a la interrupción que tiene la CPU.

Ejercicio 7.3

Se dispone de un microprocesador 8085 que controla 22 periféricos en las direcciones 10 a 25.

Se pide:

- Implementar el esquema correspondiente a una verificación de puertas de E/S centralizado, a base de decodificadores 3×8 .
- Idem, con decodificación independiente.

Ejercicio 7.4

Existen 34 semáforos conectados a un computador central con un bus de direcciones de 16 bits. Si cada situación del semáforo, corresponde a una posición del mapa de E/S, diseñe el esquema y codificación necesaria, explicando su funcionamiento, si se utiliza un direccionamiento centralizado con decodificadores 2×4 .

Ejercicio 7.5

¿Es posible conectar un PPI 8255 mediante direccionamiento por líneas independientes? ¿Por qué? En caso negativo, adaptarlo suponiendo un bus de direcciones de 20 bits y 18 periféricos seleccionados. Establecer las direcciones a las que responde el PPI. ¿Son éstas consecutivas?

330 / Unidad de entrada y salida

Ejercicio 7.6

Diseñar el circuito lógico necesario para la conexión de 3 periféricos con líneas de petición y cedición, suponiendo la técnica de encadenamiento para la gestión de prioridades, indicando los niveles lógicos de líneas y flip-flops para los siguientes casos:

- La petición la hacen los tres periféricos.
- La petición la hace únicamente el segundo en la cadena de prioridades.

Ejercicio 7.7

Dibujar los cronogramas correspondientes al ejercicio anterior.

Ejercicio 7.8

¿Podría trabajar el microprocesador 6502 con interrupciones vectorizadas y gestión de prioridades de periféricos a través de un PIA R6520?

En caso afirmativo diseñar la lógica necesaria para que funcione y explicar (no realizar) el programa de control y gestión de prioridades.



Buses



— **Temporización.** Desde el punto de vista de la temporización, el bus puede ser de *ciclo completo* o de *ciclo partida*. En el primero, el bus está ocupado todo el tiempo que dura la transferencia elemental entre los dos dispositivos que se comunican. En el segundo, se divide el tiempo del bus en una serie de pequeños períodos (*time slots*), cada uno de los cuales sirve para enviar un mensaje. La transferencia elemental se divide en dos períodos: el de petición y el de contestación.

8.1. INTRODUCCIÓN

Entre los elementos básicos que definen la estructura de un computador, se encuentran los elementos de comunicación, que comprenden a los *enlaces* y a los *comunicadores*.

Un enlace es un dispositivo que permite transmitir información entre dos o más elementos, mientras que un commutador se encarga de dirigir la información entre varios enlaces.

El elemento de comunicación más común en los computadores es el bus, que consta de un enlace que comunica, selectivamente, un cierto número de componentes o dispositivos, de acuerdo con ciertas normas o reglas de conexión. El bus incluye, por lo tanto, los dos conceptos de enlace y de commutador, puesto que debe permitir, en cada momento, la especificación de los dispositivos que se comunican a través de él.

Se llama ciclo de bus la operación básica del bus y en ella se realiza una transferencia elemental entre dos dispositivos conectados al bus.

Las propiedades que caracterizan a un bus son las siguientes:

- **Banda base.** La información se envía directamente por el bus, sin emplear portadora.
- **Grado de paralelismo.** Normalmente el bus es *paralelo*, transmitiendo simultáneamente todos los bits de una palabra. A veces, se utiliza el denominado bus multiplexado, que no es más que un bus paralelo, pero con un ancho menor que el de la palabra a enviar, por lo que, para hacer una transferencia elemental completa, hay que multiplexar al bus en el tiempo.

rápida y, cuando un dispositivo está apagado o desconectado, toma, de forma natural, el valor negado correcto.

- Soporte. Es el material empleado para hacer la conexión física de los distintos dispositivos. Puede ser:

- Pistas de circuito impreso
- Cables
 - Parejas de cables trenzados.
 - Cable plano
 - Cable coaxial
 - Fibra óptica

8.2. NIVELES DE ESPECIFICACIÓN DE UN BUS

La especificación de un bus se puede abordar desde los 6 niveles que se citan:

1. Mecánico
 2. Eléctrico
 3. Lógico
 4. De temporización básica
 5. De transferencia elemental
 6. De transferencia de bloque
- 8.2.1. Nivel mecánico**
- En este nivel se definen temas tales como el tipo de soporte, el número de hilos del bus, tipo de conectores, etc.
- 8.2.2. Nivel eléctrico**
- Debe indicar el circuito equivalente de los dispositivos que se conectan a las líneas del bus, así como las tensiones y/o corrientes utilizadas en el establecimiento del valor de las señales.
- 8.2.3. Nivel lógico**
- Se definen estéticamente todas las líneas o hilos del bus estableciendo la equivalencia entre los valores eléctricos de las señales y sus valores lógicos.

8.2.4. Nivel de temporización básica

En este nivel se construyen los cronogramas para la operación más elemental del bus, es decir, de un ciclo o período.

8.2.5. Nivel de transferencia elemental

Plantea el procedimiento para efectuar una transferencia de un dato por el bus. Evidentemente, en el caso de un bus de ciclo completo, este nivel coincide con el anterior, puesto que la temporización básica establece todas las condiciones necesarias para transferir un dato.

8.2.6. Nivel de transferencia de bloque

Se define el protocolo de comunicación usado en la transferencia de un bloque, el cual está compuesto por una serie de transferencias elementales y tiene por objetivo la transmisión de un bloque de información con utilidad propia.

8.3. BUSES NORMALIZADOS

Hasta hace pocos años, el bus se diseñaba de forma específica para cada modelo de computador, por lo que su estudio sólo tenía interés para los fabricantes de computadores, ya que ningún usuario tenía acceso a ese nivel de la máquina.

Sin embargo, los microprocesadores han abierto el camino del diseño de computadores a un gran número de profesionales. Como resultado, a finales de la década de los 70 han empezado a surgir una serie de normas que permiten interconectar placas y módulos de distintos suministradores para conseguir máquinas adecuadas a aplicaciones específicas.

Además, la rápida evolución de los circuitos integrados impulsa la aparición de microprocesadores y módulos VLSI cada vez más potentes. La utilización de una norma de bus apropiada, permite rediseñar un solo módulo del sistema, conservando la compatibilidad con los restantes. De esta forma, un fabricante puede ir evolucionando, incorporando las nuevas aportaciones VLSI, sin un coste excesivo de redesign.

Una norma de bus debe cubrir todos los niveles de especificación ya analizados. En concreto, debe establecer los siguientes parámetros:

- Protocolo de transmisión
- Método de control del bus. Maestros y esclavos.

- Descripción de las señales
- Diagrama de tiempos de las señales
- Especificaciones eléctricas
- Especificaciones mecánicas

Entre los buses normalizados más utilizados para los microprocesadores de 8 y 16 bits destacan el Multibus I, el STD Y el S-100. Buses para microprocesadores de 16 y 32 bits, son el VME, el Multibus II Y el NuBus Y Futurebus.

8.4. PARALELISMO DEL BUS

Generalmente, el bus paralelo tiene un ancho de palabra que coincide con el ancho de la información a transmitir. No obstante, los microprocesadores y las memorias han introducido, por la reducción de patillas que se produce en su empaquetamiento, el concepto de *bus multiplexado*, empleado comúnmente para conseguir "sacar" fuera del microprocesador las direcciones o para introducir dichas direcciones en las pastillas de memoria.

Con el multiplexado del bus se envían por los mismos hilos informaciones distintas en momentos diferentes. Existen unas señales adicionales que sirven para identificar la información que circula por el bus en cada instante. Si, como es frecuente, el resto del sistema exige que el bus no esté multiplexado, hay que añadir un registro cerrojo (latch) o un multiplexor adicional para realizar la multiplexación o la demultiplexación del bus, tal y como se muestra en la figura 8.1.

Se puede considerar el bus serie como un caso extremo de la multiplexación, puesto que está formado por una pareja de hilos por los que circula la información bit a bit. Sin embargo, aquí no existe señal para demultiplexar los bits. La demultiplexación se efectúa en base a la forma de onda enviada, de manera parecida a como se identifican y separan los bits leídos en un disco magnético.

El empleo de buses serie exige dividir la palabra en bits, para proceder a su envío, así como realizar la función inversa de agrupamiento en la recepción. Para este fin se usan registros de desplazamiento que serializan y parallelizan la información, tal y como se representa en la figura 8.2.

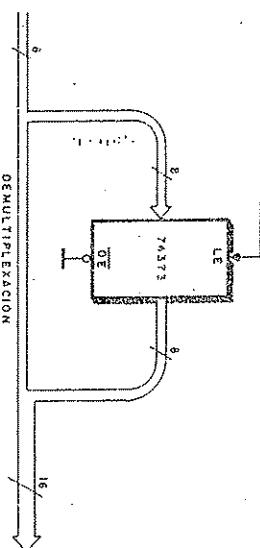


Fig. 8.1. Demultiplexación y multiplexación.

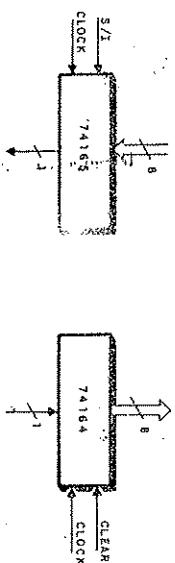


Fig. 8.2. Registros de desplazamiento que serializan y parallelizan la información.

8.5. BUS DE CICLO PARTIDO

Para describir el concepto de bus de ciclo partido se hace un análisis de una operación de transferencia típica en un bus, como es la lectura de una palabra de memoria por parte de la UCP.

En un bus de ciclo partido, el tiempo se divide en una serie de fracciones o períodos, cada uno de los cuales permite enviar un mensaje. La figura 8-3 muestra el reparto del tiempo para el caso normal de que los períodos tengan una duración fija. Y se establezcan de forma sincrona, o sea, con un reloj central que señala su principio y su final. La duración del período o ranura viene fijada por las características de transmisión del bus, pero no por el tiempo de respuesta de los dispositivos, por lo que puede ser muy pequeña.

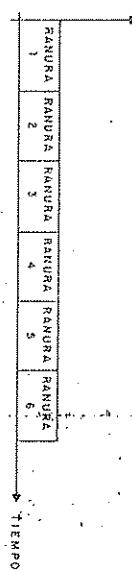


Fig. 8-3. Periodos o ranuras del bus de ciclo partido.

La operación de lectura, antes mencionada, se divide ahora en dos mensajes. El primero lo envía la UCP a la memoria y consta de la dirección deseada y una señal que determina que se trata de una operación de lectura. El segundo, lo proporciona la memoria cuando ha completado la operación y está formada por el dato leído. La figura 8-4 presenta un posible esquema para la ejecución de esta operación, comparándolo con los tiempos empleados en el caso de ciclo completo.

El mayor cuidad en el ciclo partido se consigue complicando los dispositivos que se le conectan. En el ejemplo al que se hace referencia, la memoria deberá disponer de los circuitos necesarios para llevar a cabo las siguientes funciones:

1. Almacenar la dirección deseada

2. Almacenar el tipo de operación

3. Almacenar el dato leído o enviado a grabar

4. Iniciar el envío de mensajes al peticionario

En caso de existir varios peticionarios, se debe almacenar la identificación de éstos, para poder enviarle el mensaje de contestación, y se debe disponer de un mecanismo de contestación para evitar o retener los accesos simultáneos a la memoria.

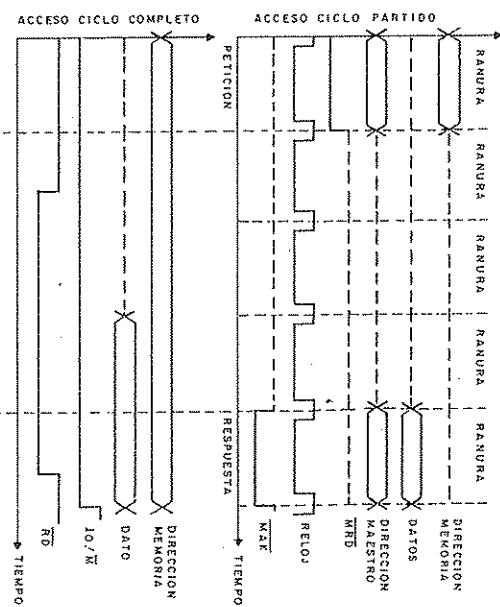


Fig. 8-4. Comparación de los tiempos para realizar una lectura en memoria en el caso de ciclo completo y en el caso de ciclo partido.

8.6. TRANSFERENCIAS SÍNCRONAS Y ASÍNCRONAS

El concepto de sincronismo se aplica a los buses en distintos niveles. Cuando las ranuras tienen una duración fija y forma predefinida, la temporización de un bus de ciclo partido es síncrona.

8.6.1. Transferencia en ciclo completo

Se aplican los mismos criterios que se estudiaron en el capítulo de Entradas/Salidas y que se pueden resumir en los puntos siguientes:

- *Lectura síncrona:* El maestro coloca en el bus la dirección deseada y supone que el esclavo, a su vez, pone el dato en el bus en el tiempo T, momento en que lo toma el maestro.
- *Escritura síncrona:* El maestro pone en el bus la dirección y el dato y supone que el esclavo recoge esta información antes de que transcurra un cierto tiempo T.



- *Lectura asíncrona:* El maestro sitúa en el bus la dirección deseada y se queda a la espera de que conteste el esclavo, confirmando que ha colocado el dato en el bus, o bien, si la espera sobrepasa cierto tiempo, genera una señal de error.
- *Escritura asíncrona:* El maestro coloca en el bus la dirección y el dato y se queda esperando la confirmación, por parte del esclavo, de que ha tomado la información. Si la espera sobrepasa un cierto tiempo, el maestro genera una señal de error.

8.6.2. Transferencia en ciclo partido

La operación del bus en ciclo partido se puede derivar de la del ciclo completo, sin más que fraccionarla en una fase de comienzo de la transferencia y en otra de finalización, ocupando cada una de estas fases una ranura. La transferencia la inicia un maestro, empleando una de las ranuras del bus y la finaliza el esclavo, empleando otra ranura, razón por la que los esclavos han de tener la capacidad de solicitar y usar ranuras para finalizar adecuadamente las transferencias. Si hay varios maestros, la solicitud enviada por el bus debe contener la identificación del maestro solicitante, de forma que el esclavo pueda contestar correctamente.

- *Lectura síncrona:* Se compone de dos ranuras, según se refleja en la figura 8-5. En la primera, el maestro envía la orden de lectura, así como la dirección deseada. Ciertio tiempo después, el esclavo, cuando dispone del dato pedido, solicita otra ranura para enviárselo al maestro.

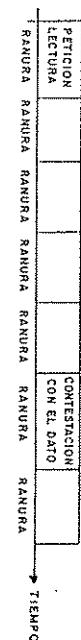


Fig. 8-5. Lectura síncrona en bus de ciclo partido.

Generalmente, el maestro dispone de una función de espera, de forma que, pasado un tiempo prefijado sin contestación, genera una señal de error.

- *Escrutura síncrona:* Se trata de la transferencia más sencilla, puesto que sólo consta de una ranura en la que el maestro envía un dato y la dirección. Figura 8-6. Se supone que el esclavo tomará correctamente esta información porque no existe mecanismo alguno para comprobarlo.

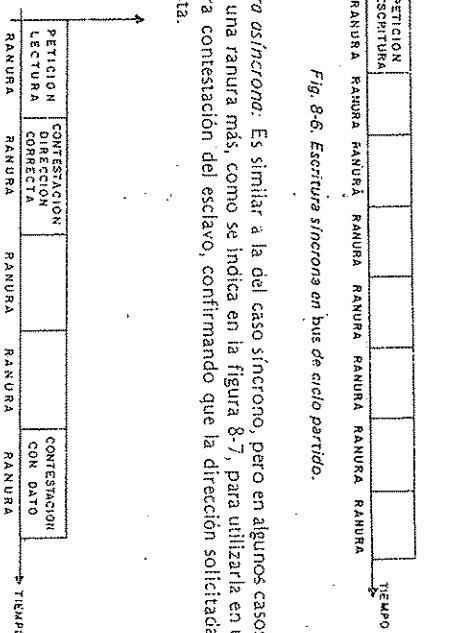


Fig. 8-6. Escritura síncrona en bus de ciclo partido.

Lectura asíncrona: Es similar a la del caso síncrono, pero en algunos casos se añade una ranura más, como se indica en la figura 8-7, para utilizarla en una primera contestación del esclavo, confirmando que la dirección solicitada es correcta.

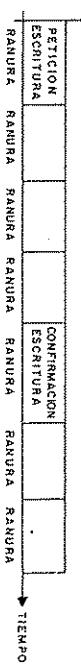


Fig. 8-7. Lectura asíncrona en el bus de ciclo partido.

- *Escrutura asíncrona:* Ocupa dos ranuras, una en la que el maestro envía la dirección y el dato a escribir, y otra para que el esclavo conteste confirmando la terminación de la operación de escritura. Figura 8-8.

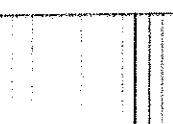


Fig. 8-8. Escritura asíncrona en bus de ciclo partido.

8.7. CONTROL DEL BUS

Según el control que ejercen sobre el bus los elementos a él conectados, se clasifican en *maestros* y *esclavos*. Los primeros son capaces de obtener el uso del bus, iniciando sus ciclos y determinando su tipo y temporización. Los segundos sólo realizan la transferencia de información, de acuerdo a los ciclos generados por los maestros.





El maestro puede ser *permanente* o *temporal*. Por ejemplo, la UCP en un sistema microprocesador será un maestro permanente, mientras que el controlador de periférico con acceso directo a memoria, será un maestro temporal.

En un sistema multiprocesador, pueden existir varios maestros principales que se reparten el uso del bus. En estas situaciones hay que diseñar la lógica de asignación para que no accedan al bus dos maestros simultáneamente.

8.8 LONGITUD DEL BUS

Es una de sus principales características. En principio, interesa que el bus pueda ser largo para poder conectarle todos los periféricos que se deseen, sin problemas. Sin embargo se puede afirmar, en forma general, que "a mayor longitud del bus, menor velocidad". Hay que encontrar un compromiso entre la longitud y la velocidad.

La figura 8.9 recoge una gráfica de Borrell donde se representa la máxima distancia correspondiente a buses reales en función de su tipo. La recta que aparece en el dibujo define los tamaños más adecuados para los distintos tipos.

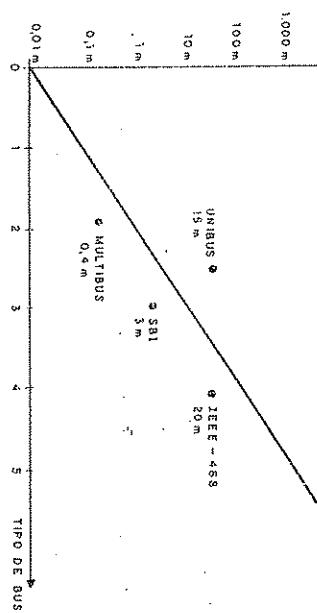


Fig. 8.9. Gráfico que representa la longitud del bus en función de su tipo.

El bus UNIBUS del PDP-11 y del VAX tiene un tamaño excesivamente grande, mientras que el IEEE-488, diseñado para interconectar instrumentos de laboratorio, es insuficiente, puesto que las necesidades de espacio en estos entornos superan fácilmente los 20 m.

8.9. JERARQUÍA DE BUSES

En todo computador existe toda una amplia gama de necesidades de comunicación, que comprende desde los elementos internos de las pastillas que contiene hasta las de los grandes módulos o unidades en que se dividen. Se pueden establecer 6 tipos de buses, que son los siguientes:

- 0: Buses internos de una pastilla

- 1: Buses para la conexión de componentes

- 2: Panel posterior

- 3: Sistema

- 4: Entrada/Salida

- 5: Serie

8.9.1. Buses tipo 0

Son los que utilizan internamente las pastillas de circuito integrado. El usuario no tiene acceso a ellos y sólo tiene cierta importancia la anchura de los mismos, puesto que es la determinante de la capacidad de trabajo en paralelo de las pastillas correspondientes.

8.9.2. Buses tipo 1

Corresponden a los encargados para la interconexión de los componentes de una placa o circuito impreso. Sus características más importantes son:

1. Al estar restringidos a una placa, las dimensiones de estos buses se reducen a decenas de cm, por lo que no es necesario considerar al bus como una línea de transmisión, ni emplear dispositivos especiales para hacer terminar el bus con su impedancia característica.
2. Los valores de fan-out de los circuitos integrados son suficientes para conectarles directamente al bus sin ayuda de circuitos de mayor poder deaccionamiento o amplificadores. Solamente cuando las señales han de salir de la placa, se requiere el uso de un buffer.
3. Los buses de tipo 1 suelen ser sincronos y disponen de un único maestro que establece la temporización en las transferencias. En los microprocesadores existen dos sistemas para llevar a cabo la temporización. Uno emplea las señales RD y WR, además de definir su temporización.





4. El diseño de los buses viene casi totalmente determinado por los circuitos VLSI empleados.

8.9.3. Buses tipo 2

Sirven para interconectar las distintas placas de un módulo, formando lo que se llama el *panel posterior*. Las dimensiones de estos buses son del orden de un metro y el número de señales que los componen se acercan al centenar.

Ejemplos de este tipo son el Multibus de Intel, el STD y el S-100.

8.9.4. Buses tipo 3

Interconectan diversos módulos del computador. La longitud de este bus es del orden de 10 m, lo que obliga a tratar el bus como una línea de transmisión, motivo por el que hay que colocar dispositivos terminales, como los que se muestran en la figura 8-10. También se precisan buffers para conectar un panel posterior con otro.

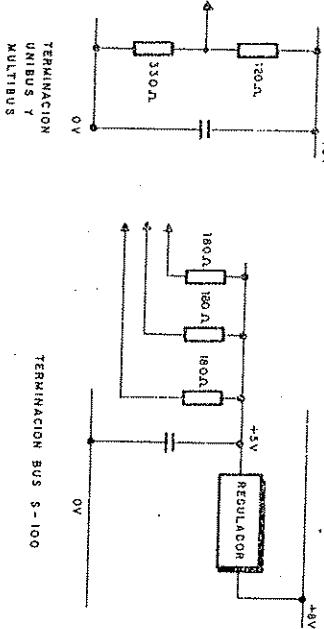


Fig. 8-10. Circuitos que se colocan en la terminación de los buses del tipo 3.

Para evitar problemas de ruido en estos buses de gran longitud, se apantallan los cables y se incrementa la corriente de las señales.

La temporización del bus debe tener en cuenta los tiempos de propagación de las señales en él.

Ejemplos de este tipo son el UNIBUS de DEC y el Eurobus de Ferranti.

8.9.5. Buses tipo 4

Son los buses paralelo para la conexión de periféricos. Admiten 4 situaciones:

- Los controladores de los periféricos se conectan directamente al bus del sistema (bus tipo 3).
- Existe un bus especial de E/S, como sucede con el UNIBUS dentro de la arquitectura de los minicomputadores VAX. Figura 8-11.

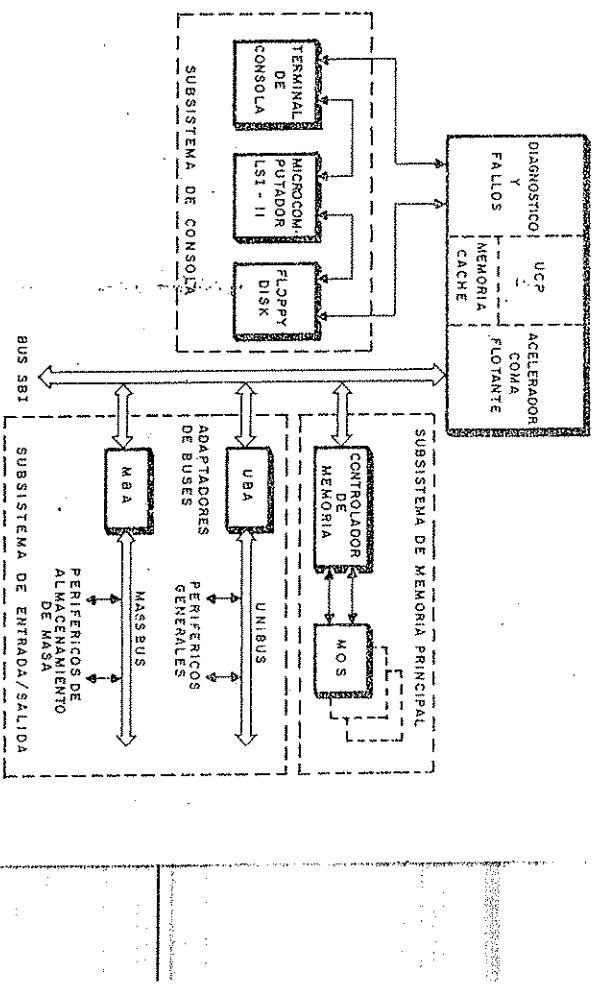


Fig. 8-11. Organización de los buses en la familia VAX.

Para evitar problemas de ruido en estos buses de gran longitud, se apantallan los cables y se incrementa la corriente de las señales.

La temporización del bus debe tener en cuenta los tiempos de propagación de la señales en él.

Ejemplos de este tipo son el IEEE-488, previsto para la

- Se diseña un bus reducido, derivado del bus del sistema, que permite la conexión en paralelo de una serie de periféricos específicos.
- Se emplea un bus paralelo de conexión de periféricos, de acuerdo con una norma.

En este tipo de buses se puede incluir el IEEE-488, previsto para la interconexión de instrumentación de laboratorio.

8.6. Buses tipo 5

Los buses serie se emplean para cubrir grandes distancias, pero soportan la menor velocidad de transmisión. No se van a considerar los enlaces con portadora.

El bus serie puede ser simple o doble. El simple admite la transmisión en un solo sentido o en ambos de forma alternativa (duplex). El doble permite la transmisión en los dos sentidos simultáneamente (transmisión full-duplex).

Los buses serie normalizados más difundidos son el RS-232-C y el RS-422/3. El RS-422/3 soporta comunicar distancias de hasta 1.000 m y se compone de unas normas que sólo especifican el nivel eléctrico, quedando los restantes niveles establecidos por otras normas o por el propio usuario.

La norma RS-232-C se aplica en la conexión de terminales a modems y especifica una serie de señales lógicas, así como los mecanismos de sincronismo utilizados. Esta norma, sin embargo, está siendo muy empleada en la interconexión de equipos digitales directamente, sin modems, en cuyo caso se sigue la especificación eléctrica mecánica de la norma, adecuándose la especificación lógica a las necesidades de cada usuario. La temporización básica de la norma RS-232-C viene definida por la velocidad de transmisión, la cual puede tomar una serie de valores predefinidos, comprendidos entre 110 y 19.200 baudios.

Con frecuencia se transmite la información en códigos ASCII, empaquetándose los bloques de información a transmitir con unas cabeceras y colas, que identifican al destinatario y el tipo de información enviada, así como detecta posibles errores en la transferencia.

Más adelante se describen los aspectos más relevantes de la norma RS-232-C.

En la figura 8-12 se presenta la aplicación de un bus serie que conecta varios terminales a una UCP, mientras que en la 8-13 aparece el esquema de aplicación de un bus serie en la conexión de varias UCP en forma de red en anillo. En la red en anillo cada dispositivo tiene asociado un elemento de conexión por el que entra y sale el bus, de manera que la información va pasando de uno a otro elemento de conexión en forma circular.

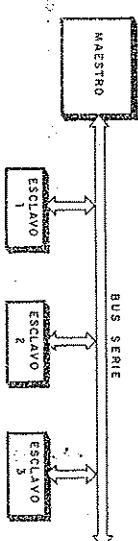


Fig. 8-12. Enlace de un maestro con varios esclavos mediante un bus serie.

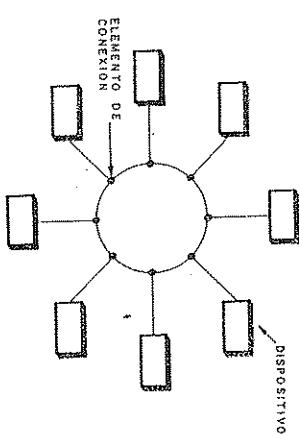


Fig. 8-13. Conexión de varias UCP en forma de red local en anillo.

Un ejemplo de red local es la Ethernet. Las velocidades de transmisión son del orden de 10 Mbit/s y las longitudes de unos pocos km, permitiendo la interconexión de clientes de dispositivos.

Como el bus serie no dispone de señales especiales para establecer la dirección del dispositivo destino de la información, en el caso de conectar varios receptores a un mismo bus serie, debe arbitrarse un mecanismo de dirección que permita seleccionar al receptor deseado. La solución consiste en empaquetar la información con una cabecera y una cola, como se indica en la figura 8-14. La cabecera contiene una indicación del dispositivo al que va destinada la información. La cola, además de indicar el fin del mensaje, suele contener un código de detección y/o corrección de error.

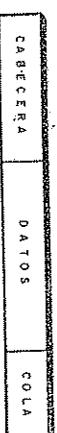


Fig. 8-14. Empaqueamiento de la información con una cabecera que identifica el dispositivo para el que va destinada la información.



La conexión de varios emisores al mismo bus es algo más delicada, puesto que hay que evitar que dos de ellos intenten tomar el bus a la vez. Si sólo hay un maestro, éste debe controlar todas las transmisiones usando un método de escrutinio o polling. Si existen varios maestros, como sucede en una red local, se suele utilizar un testigo que va pasando por la red en anillo.

3.10. DETECCION Y TRATAMIENTO DE ERRORES

Un buen bus debe ser capaz de detectar y tratar de corregir los errores más comunes que se pueden producir, entre los que se encuentran los siguientes:

- Intento de acceso a una posición de memoria no existente.
- Intento de escritura en una posición protegida.
- Intento de ejecución de instrucciones críticas, como pueden ser HALT o RESET.
- Bloqueo del sistema (dead lock).
- Error en la transmisión por ruido u otras causas.

APENDICE

La norma de conexión RS-232-C

A&1 INTRODUCCION

El modelo de conexión RS-232-C describe la forma de conectar un terminal a un modem, para la transmisión de datos en serie.

La norma RS-232-C proviene de la EIA RS-232, que fue propuesta en 1969 por la EIA (Asociación de Industrias Electrónicas).

Aunque la norma se estableció para la conexión de equipos informáticos (Equipos Terminales de Datos o DTE) a modems (Equipos de Comunicación de Datos o DCE),

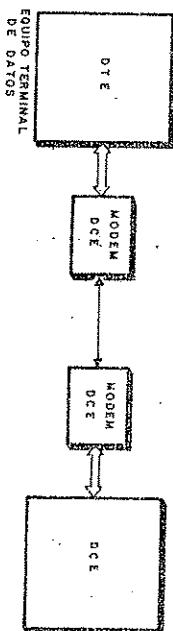


Fig. A&1. La norma de conexión RS-232-C se estableció para interconectar equipos informáticos (Equipos Terminales de Datos o DTE) a modems (Equipos de Comunicación de Datos o DCE).

o DCE), tal como se refleja en la figura A&1, la realidad es que se está empleando masivamente para interconectar equipos informáticos entre sí, sin la inclusión del modem. Por este motivo, no puede aplicar directamente la norma, y la interpretación que se da a sus diferentes señales, puede diferir de un fabricante a otro. Esta norma está especialmente difundida a nivel de microcomputadores.

3A.2. ESPECIFICACIONES GENERALES

Se pueden dividir en los tres siguientes apartados.



8A.2.1. Especificación eléctrica

En la figura A8-2 se muestran los circuitos equivalentes de emisores y receptores, que se aplican a todas las señales que componen la norma.

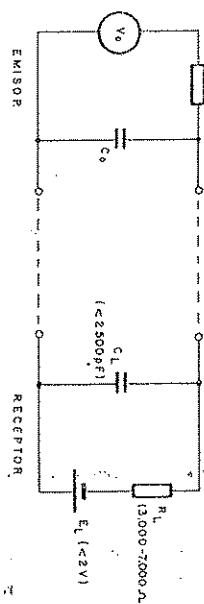


Fig. A8-2. Circuitos equivalentes de un emisor y un receptor en la norma RS-232-C.

Las señales están dadas en tensión y comprenden los siguientes valores:

- a) $+15 \text{ V}$ a $+3 \text{ V}$ de señal, se considera como un nivel bajo
- b) -15 V a -3 V de señal, se considera como un nivel alto.

8A.2.2. Especificación lógica

La norma RS-232-C contempla 25 señales de las que se suelen emplear menos de 10. Las más importantes se presentan en la figura A8-3 y son:

- Pata 2: Transmisión de datos de DTE a DCE
- Pata 3: Transmisión de datos de DCE a DTE
- Pata 20: DTR, el equipo DTE está en estado operativo
- Pata 6: DSR, el equipo DCE está en estado operativo
- Pata 4: RTS, petición de DTE para iniciar el envío.
- Pata 5: CTS, DCE está dispuesto a recibir.
- Pata 8: DCD, el DCE detecta portadora
- Pata 7: Masa
- Pata 1: Tierra de protección

Dado que esta norma se aplica para interconectar equipos directamente, el asignar a éstos el tipo DTE o DCE es totalmente arbitrario, lo que puede ocasionar problemas a la hora de conectar equipos de diferentes fabricantes.

350 / Buses

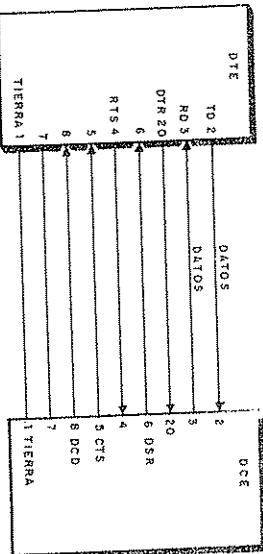


Fig. A8-3. Conexión lógica de las señales más importantes de la norma RS-232-C.

8A.2.3. Especificación mecánica

Esta norma no establece especificaciones mecánicas, pero lo más habitual es usar conectores del tipo DB-25, que constan de 25 paltas, repartidas de la forma que indica la figura A8-4.

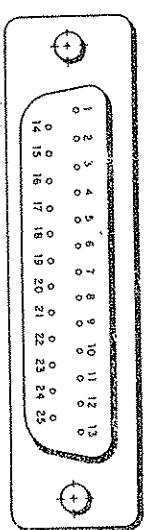


Fig. A8-4. Conector DB-25, usado normalmente para la aplicación de la norma RS-232-C.

El conexionado puede realizarse con cable plano, hilos trenzados o cable coaxial, según la distancia entre emisor y receptor

8A.3. METODO DE TRANSMISION

El método de transmisión es el que define la transferencia elemental del bus serie.

Aunque el método de transmisión no forma parte de la norma RS-232-C, generalmente se emplea el tipo asíncrono con formato start/stop. El envío se hace por bytes individuales, de acuerdo con el formato que aparece en la figura A8-5. Además, la información se suele codificar en ASCII.

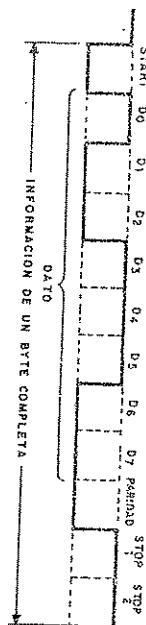


Fig. A8-5. Formato con el que se envía la información en la comunicación asincrónica con 8 bits de datos.

Como se refleja en la figura A8-5, la línea de comunicación está, normalmente, en estado 1, empleándose su paso a 0 (flanco descendente), como señal START o de aviso de comienzo del envío de un byte. Seguidamente vienen los bits de datos, que pueden ser de 5 a 8, aunque lo más frecuente es que sean 7 u 8. A continuación viene un bit de paridad, que es opcional y, finalmente, uno o dos bits de STCOP que avisan el fin del byte. Si se desea enviar otro byte, se repite el proceso y, en caso contrario, queda la línea con nivel 1.

La información no se codifica para la transmisión, siendo el estado 0 o 1 de la línea, el valor directo de cada bit. Este hecho obliga a que el receptor esté sincronizado con la señal recibida, lo que se realiza con el flanco descendente del bit de START.

Para elevadas velocidades de transmisión se usa el modo síncrono, cuyo formato se representa en la figura A8-6 y que está formado por uno o dos bytes de sincronismo (en el caso del código ASCII este byte es 0010110), seguidos por los bytes de datos. La transmisión debe ser continua, por lo que el emisor deberá intercalar, automáticamente, bytes de sincronismo cuando sean necesarios. El receptor ha de autosincronizarse con la cadena de bits que recibe.

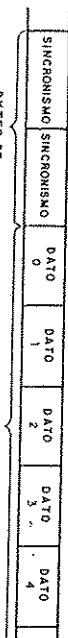


Fig. A8-6. Formato de la comunicación síncrona en la norma RS-232-C.

8A.4 PROTOCOLO DE COMUNICACIÓN

Es el que establece la sincronización a nivel de transferencia de bloques de información. Se emplean dos técnicas: protocolo por señales y protocolo por códigos.

352 / Buses



En el protocolo por señales se utilizan algunas señales de la norma RS-232-C para que los dispositivos indiquen si pueden o no recibir información. La pareja de señales DTR/DSR, o bien, RTS/CTS se suelen emplear con este fin. En este caso, la señal DTR o RTS indica que el dispositivo DTE está conectado o dispuesto, mientras que la señal DSR o CTS indica que lo está el DCE.

En el protocolo por códigos se emplean diversas técnicas basadas en códigos, en ASCII para que el receptor indique si puede recibir información. El proceso es el siguiente: En principio se supone que el receptor está en disposición de recibir información, por lo que el emisor comienza a enviarla. Si el receptor se aproxima a la saturación de su memoria de recepción, envía un XOFF para detener la transmisión. Una vez que ha vaciado su memoria, envía un XON para que el emisor reanude el envío de información.

Otro protocolo por códigos muy empleado es el que utiliza los caracteres ASCII ETX y ACK. Suponiendo que está disponible el receptor, el emisor inicia la transmisión enviando una línea de información, que finaliza con el código ETX. Una vez que el receptor ha asimilado la línea y está en disposición de recibir más información, envía el código ASCII ACK, con el que informa al emisor de su situación. La máxima longitud de las líneas de información es de 80 a 132 bytes.

8A.5. DISEÑO DE UN INTERFAZ RS-232-C

Gracias a los circuitos integrados VLSI, disponibles, el diseño de un interfaz RS-232-C es muy sencillo. En concreto, se puede usar una pastilla denominada 8251 USART (Universal Synchronous/Asynchronous Receiver/Transmitter), que realiza todas las operaciones necesarias para serializar la información enviada y transformar en paralelo la que recibe. También genera las formas de onda correspondientes a la velocidad de transmisión seleccionada y la que sincroniza la recepción.

En la figura A8-7 se muestra el diseño de una comunicación doble, en la que se usan las señales CTS, RTS, DTR y DSR para indicar el estado de los equipos conectados. Los relojes de transmisión y recepción se derivan del reloj principal del microprocesador (que se ha sugerido funciona a 3,072 MHz), mediante un divisor por 10. Si, además, se programa al USART para que divida por 64 esta frecuencia, se obtiene una velocidad de transmisión y recepción de 4.800 baudios.

Las pastillas 1488 y 1489 de la figura A8-7 realizan la recepción y adaptación de las líneas de transmisión, de acuerdo con la norma RS-232-C, puesto que la USART trabaja con niveles TTL (0 y 5 V) diferentes.

Periféricos

9.1. DEFINICION Y CLASIFICACION DE LOS EQUIPOS PERIFERICOS

La UCP es la encargada de realizar el procesamiento de la información. Se halla rodeada de diferentes dispositivos, cuya labor consiste en la introducción de datos y la extracción de resultados del mundo exterior, en donde se encuentran los usuarios de la máquina. Todo este conjunto de elementos que auxilian a la UCP y que, a través de los buses o canales, introducen y obtienen información, reciben el nombre de *Equipos Periféricos*, por su posición relativa dentro del sistema. También se les llama *Dispositivos de Entrada y Salida*, teniendo en cuenta su *función*. Figura 9-1.

Dada la gran cantidad de periféricos existentes, con características, funciones y formas de empleo distintas, se hace imprescindible su clasificación, como primer paso para su estudio.

Atendiendo a su *localización física*, los periféricos se clasifican en:

- Locales*: Están situados cerca de la UCP y su conexión con ella se realiza mediante cables eléctricos, que constituyen la prolongación de las líneas de los buses internos del procesador.
- Remotos*: Se ubican en los lugares adecuados para dar servicio a los posibles usuarios del computador, por lo que la distancia a la UCP, puede variar entre unos pocos metros y muchos miles de kilómetros. En su conexión con la UCP intervienen líneas telefónicas y telegráficas y otros tipos de enlaces; por este motivo, a estos periféricos se les llama *terminales de teletipo*.

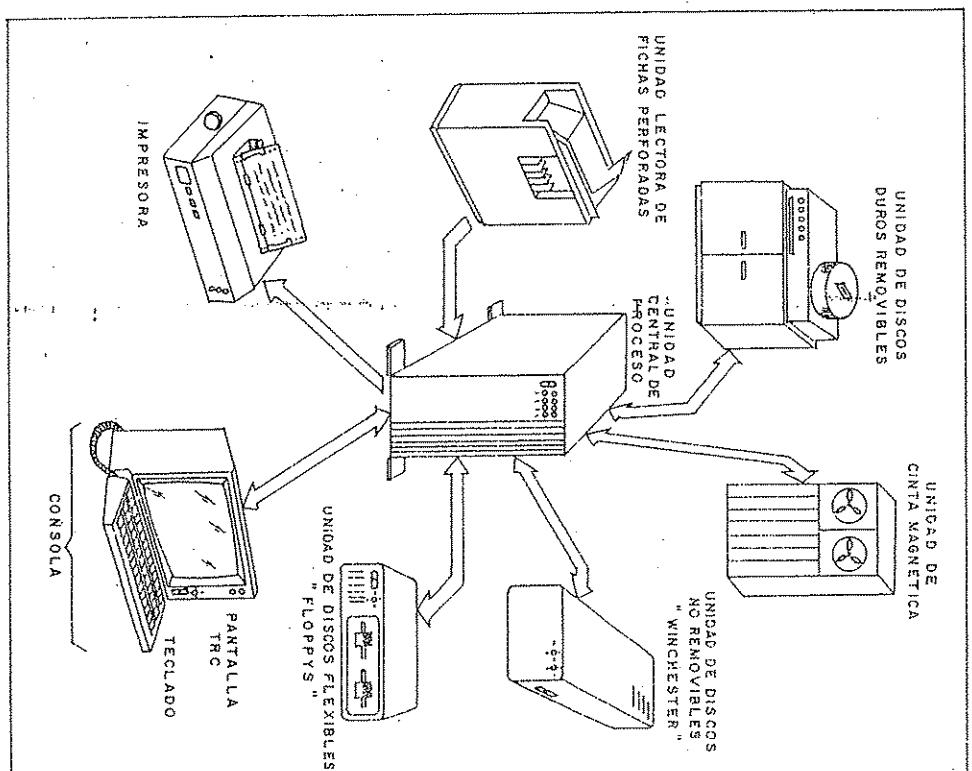


Fig. 9-1. Diagrama general de la configuración de un sistema computador. La UCP se relaciona con el exterior a través de diferentes tipos de periféricos.



Atendiendo a su función, los periféricos se clasifican en:

1) Dispositivos de entrada

Sólo son capaces de suministrar o introducir información al sistema computador. Dentro de este grupo de periféricos, destacan los siguientes:

- Lectora de marcas ópticas
- Lectora de caracteres ópticos y magnéticos
- Sensores de contacto
- Recognedor de voz
- Cámara de TV
- Digitalizador de dos y tres dimensiones
- Ratón

2) Dispositivos de salida

Pertenecen a este grupo de periféricos aquellos que están preparados, únicamente, para sacar al exterior los resultados obtenidos en el procesamiento. Destacan los siguientes:

- Impresora
- Perforadora de fichas
- Pantalla de rayos catódicos
- Trazadores gráficos o plotters
- Salida para microfilm
- Sintetizador de voz
- Salida en tres dimensiones.

3) Dispositivos de entrada y salida

Capaces de introducir y extraer información del sistema. Caben citar:

- Terminal teclado-pantalla (combinación de teclado y pantalla)
- Unidad de fichas de banda magnética.
- Unidad de almacenamiento intermedio, que cede o recibe información a/ desde la UCP, en lugar de actuar como elementos intermedios entre el hombre y la máquina. Actúan como memorias de gran capacidad que envían a la me-

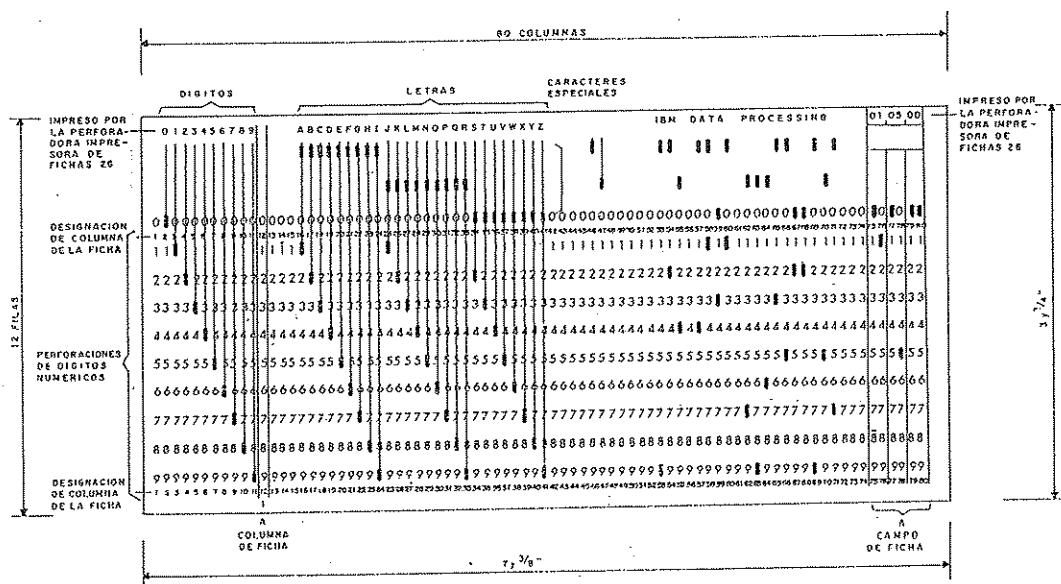


Fig. 9-2. Estructura y organización de una tarjeta perforada. Configuración estándar de perforaciones sobre una tarjeta de IBM.



moria principal, más pequeña, la información que necesita. Pertenecen a esta categoría las unidades de:

- Disco magnético
- Cinta magnética
- Tambor magnético

9.2. PERIFERICOS DE ENTRADA

9.2.1. Tarjetas y cintas perforadas

Estos dos sistemas, destinados a suministrar información al computador están completamente anticuados, por lo que resulta difícil encontrar instalaciones que dispongan de perforadoras y lectoras de tarjetas y cintas perforadas.

Cada tarjeta constituye un registro de un fichero, una sentencia de un programa, un comando, etc. Cada columna de perforaciones equivale a un carácter. El formato más popular fue el de 80 columnas, introducido en 1928 por IBM, que evolucionó hasta alcanzar 96 columnas con un tamaño menor, en 1970. Figura 9-2.

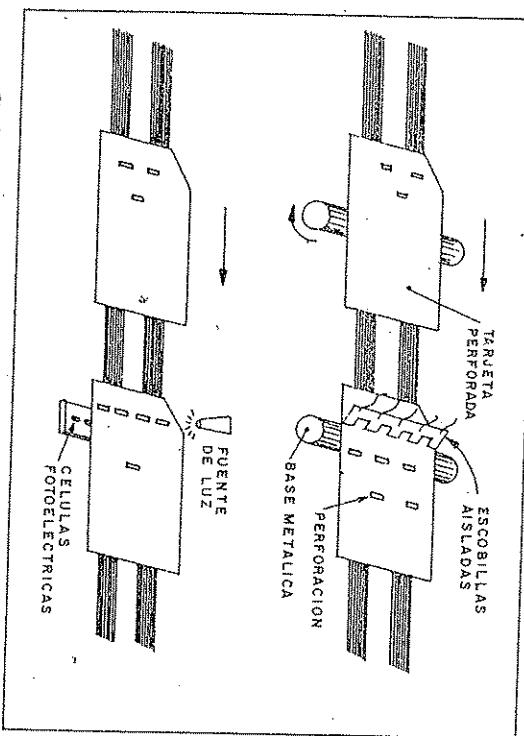


Fig. 9-3. Esquemas de funcionamiento de lectores de fichas perforadas.



La información se registra en la tarjeta y lo mismo en la cinta de papel continuo, mediante perforaciones rectangulares realizadas por máquinas perforadoras. La recogida de la información de la tarjeta se lleva a cabo mediante *lectores de escobillas* o *lectores de células fotoeléctricas*, de los que se muestra un esquema de su funcionamiento en la figura 9-3.

9.2.2. Sensores por contacto

La penetración, cada vez más intensa, de los computadores en todas las áreas de trabajo e, incluso, en la vida cotidiana, impulsa el desarrollo de nuevos sistemas de

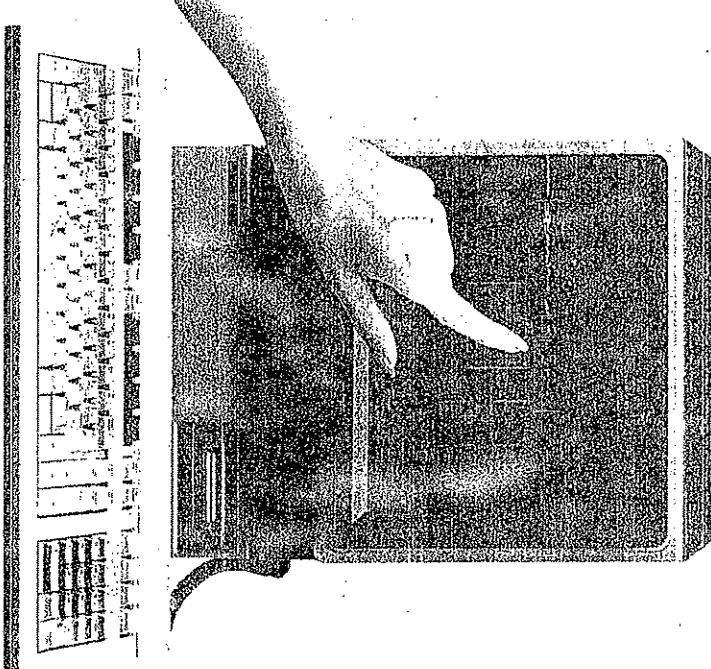


Fig. 9-4. Pantalla táctil. Cortesía de Hewlett Packard.



entrada de datos, que faciliten esta labor y la hagan más asequible al público no especializado.

Con esta orientación ya existen sistemas que permiten elegir opciones y programar en un computador tocando simplemente la pantalla con el dedo, un bolígrafo o un objeto similar. El número de opciones posibles que ofrece la máquina con este procedimiento está restringido, puesto que no se puede alcanzar mucha precisión con el dedo y podrían producirse errores entre opciones contiguas. Figura 9.4.

La pantalla táctil es un cómodo procedimiento para elegir entre diferentes opciones de menús. También resulta muy útil en aplicaciones con gráficos para indicar secciones a ampliar, etc.

Dentro de los sensores por contacto, además del tacto directo, hay otros tres tipos que requieren un elemento auxiliar:

1. Tacto indirecto: Permite mover un cursor en pantalla para seleccionar una opción mediante un mando o palanca manual (joystick).
2. Lápiz óptico: Se utiliza de forma semejante a la del tacto directo. El lápiz recibe la luminosidad del carácter de la pantalla, enviando la posición de éste al computador.
3. Tableta gráfica: La tableta representa la pantalla, de forma que, tocando un punto con un instrumento similar a un lápiz, aparece una marca en el punto correspondiente de la pantalla. La tableta puede disponer de una sección de comandos para el computador.

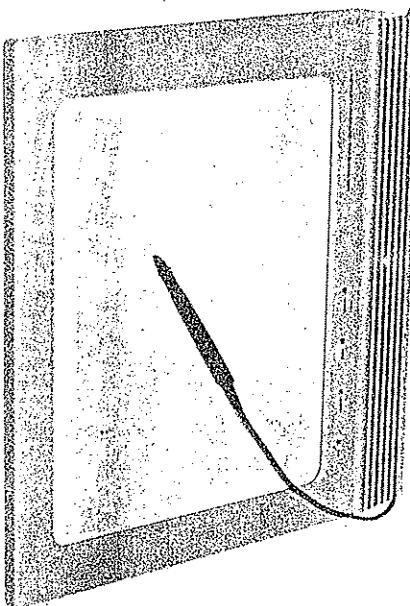


Fig. 9.5. Digitalizador de dos dimensiones. Cortesía de Hewlett Packard.

9.2.3. Reconocedores de voz

Analiza órdenes orales y las convierte en datos que transmite al computador. Su interés es extraordinario y sus aplicaciones, inmensas. Por ejemplo, un cirujano puede controlar los dispositivos de que dispone en el quirófano mediante un reconocedor de voz. También se puede emplear en sistemas de seguridad para la identificación de personas.

Cuando el reconocedor de voz recibe una orden oral, la compara con una lista de órdenes programadas anteriormente; si coincide con alguna de ellas, obrará en consecuencia. En caso contrario, hará caso omiso de la orden.

En la actualidad se fabrican reconocedores de voz que alcanzan una probabilidad de acierto de un 98%. Sin embargo, el reconocimiento de la voz humana, con todos sus matices, es una de las áreas de investigación que más problemas pendientes tiene.

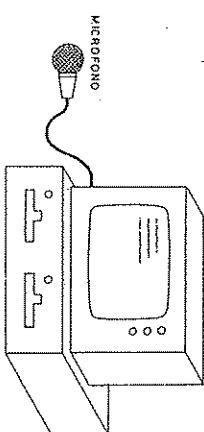


Fig. 9.6. En muchas situaciones, un micrófono y un reconocedor de voz pueden sustituir ventajosamente al inconveniente teclado.

9.2.4. Cámaras de TV

Permiten obtener imágenes digitales procesables por el computador. Se utilizan en controles de calidad, predicción de resultados en cosechas mediante imágenes aéreas, sistemas de seguridad, líneas de producción, etc.

Existen dos cámaras de TV acopiables a los computadores:

1. Cámara de estado sólido: Basada en el uso de chips, similares a los de las memorias electrónicas, pero sensibles sus células a la luz. Cada uno de sus elementos, que reciben el nombre de pixels, proporciona un código digital proporcional a la luminosidad de cada uno de sus puntos. Los pixels se agrupan en forma matricial, por filas y columnas. Véase la figura 9.7.
2. Cámara vidicon: Es el tipo de cámara que se usa habitualmente en la TV comercial y proporciona una señal analógica proporcional con la luminosidad que capta cada uno de sus puntos. Necesita que se digitalice la señal, para poder ser procesada por el computador.

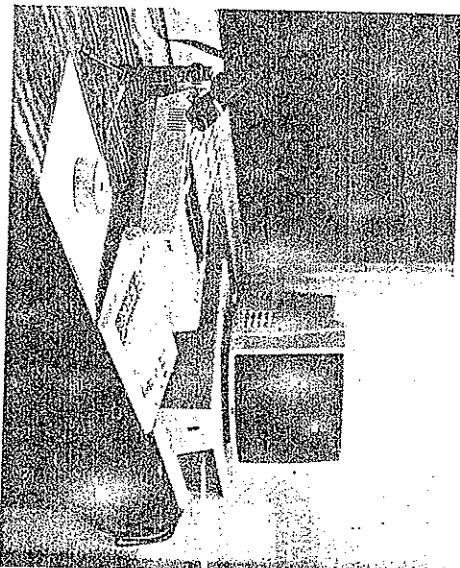


Fig. 9.7. Fotografía de un microcomputador equipado con una cámara de estado sólido.

9.2.5. Digitalizadores de dos y tres dimensiones

El digitalizador de dos dimensiones o 2D, como el de la figura 9.5, puede ser de tipo acústico o electrostático. Y se basa en un lápiz que se desplaza sobre una figura y van obteniéndose las coordenadas de sus puntos. Se emplea para generación de planos, circuitos impresos, etc.

El de 3D se trata de un dispositivo innovador, que consta de una sonda que desplaza el usuario por la superficie de un objeto de tres dimensiones. Así se introduce en el computador la información precisa para que el objeto pueda ser visualizado en la pantalla de la máquina desde cualquier ángulo y perspectiva.

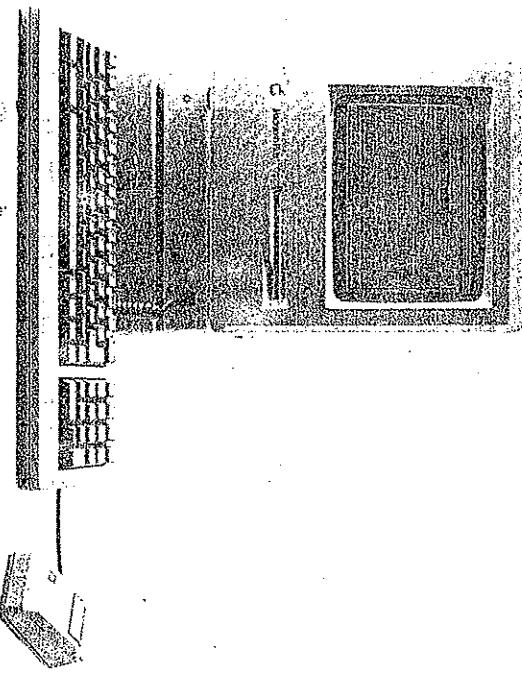


Fig. 9.8. El microcomputador Macintosh ha potenciado extraordinariamente las posibilidades del ratón que lleva incorporado.

9.2.7. Sistemas para automatizar la entrada de datos al computador

El proceso de entrada de datos de documentos escritos en el computador es muy lento y laborioso, si se realiza con los procedimientos tradicionales. Técnicas de datos es un trabajo que ocupa mucho tiempo, en especial si se compara con la velocidad de funcionamiento de los computadores.

En la parte inferior del ratón hay una bola que rueda según el sentido de desplazamiento. Así reconoce el computador la dirección en que debe mover el cursor por la pantalla.

Otros tipos de ratón, en vez de bola, disponen de un sistema óptico capaz de detectar líneas. Al desplazarse por una superficie cuadrículada, envía señales al computador cada vez que se corta una línea. Figura 9.8.



9.2.7.1. Reconocimiento de caracteres en tinta magnética (MICR)

Es un sistema promovido por la banca para agilizar el proceso de talones o cheques. Se utiliza una tinta magnética que es detectada por las lectoras MICR. Los sensores de una lectora MICR reconocen cada carácter por comparación con muestras o modelos cuadruplicados.

En la actualidad, la mayoría de los cheques y tarjetas de crédito disponen de líneas o bandas de caracteres magnéticos.

El juego de caracteres MICR consta de los 10 dígitos decimales (0-9) y cuatro símbolos especiales, que sirven para separar e identificar las diferentes partes de la información. Figura 9-9.

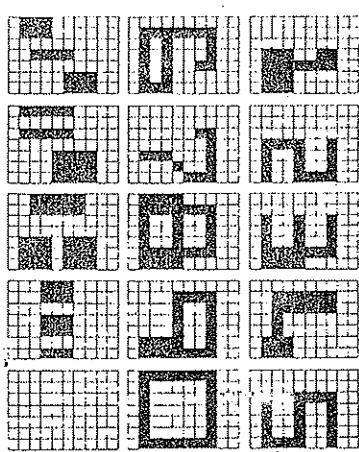


Fig. 9-9. Conjunto de caracteres con los que operan las lectoras MICR.

Los grandes bancos utilizan las lectoras MICR de alta velocidad, capaces de tratar hasta 100.000 caracteres por hora, lo que supone la manipulación de 30 cheques por segundo, con exactitud y sin causarles deterioro. Además, se suelen microfilmrar las dos caras del cheque al mismo tiempo que son leídas por la lectora MICR.

9.2.7.2. Reconocimiento óptico de caracteres (OCR)

Los dispositivos de reconocimiento óptico de caracteres escriben a máquina o a mano, leen dichos caracteres y los convierten en códigos directamente procesables por el computador.



S. La lectora es de caracteres escritos a mano, éstos deben ajustarse a una serie de normas, tales como: los caracteres deben estar encuadrados; deben ocupar el cuadro lo máximo posible, pero sin salirse de él; no deben existir discontinuidades en las líneas, etc.

Las lectoras de caracteres impresos con máquina o impresora pueden ser de diversos tipos, según su capacidad:

- **Lectoras de fuente única.** Utilizadas para propósitos especiales, tales como lectura de vales de compra de gasolina.

- **Lectoras multifuente.** Para aplicaciones en las que se deben leer diferentes tipos de caracteres OCR.

- **Lectoras de todos los fuentes.** Son las más versátiles. Permiten leer cualquier tipo de carácter OCR. Viene acompañadas de un potente software que las facilita para el reconocimiento de nuevas fuentes.

La fuente OCR más utilizada en EE.UU. es la OCR-A, aunque internacionalmente la más usada es la OCR-B. Ambas son dos formas diferentes de escritura de caracteres.

El empleo de dispositivos OCR abarca desde lectoras de cintas, procedentes de casas registradoras y sumadoras, hasta lectoras de textos, pasando por lectoras de pequeños documentos, como vales de gasolina.

En la figura 9-10 se muestra un esquema de funcionamiento de las lectoras OCR.

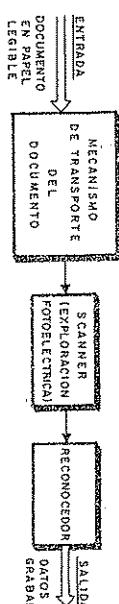


Fig. 9-10. Esquema gráfico sobre el funcionamiento de las lectoras OCR.

Una vez transportado el documento hasta el sistema de proceso, se analiza mediante la proyección de un rayo de luz. Un sistema fotosensible recoge la luz reflejada, obteniéndose una matriz de bits, que indican con "unos" la figura de cada letra. El software del computador examina esta matriz y deduce el carácter a que corresponde. Figura 9-11.

El reconocimiento de caracteres manuscritos no puede alcanzar el 100% de exactitud. Por ejemplo, el número 0 es fácil confundirlo con la letra O; la S con el 5, la Z con el 2, etc. Aun así, ya se han conseguido sistemas con un 99% de aciertos.

Una aplicación típica de OCR es la lectura de etiquetas de productos en venta, preimpresas con caracteres OCR. Así se evita el teclear el precio, el tipo de productos, etc.

60

Fig. 9-11. Matriz que se obtiene por la exploración de las fotocélulas de la lectora OCR, si leer el número 4.

9.2.1.3. Reconocimiento óptico de marcas (OMR)

La única desviación del sistema OCR, en la que se marca con un lápiz una de las respuestas posibles a una pregunta. La lectora OMR detecta la posición de la marca, conociendo así la respuesta. Figura 9-12.

Es una descripción real de la cultura Gobernante

J-12. En el reconocimiento óptico de marcas (OMR) la opción debe quedar bien marcada, ennegociendo el círculo completamente con el lápizco, sin salirse de él.

	PR.	OPCIONES
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
	PR.	OPCIONES
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Los vectores de código de barras reconocen la información codificada en una serie de barras anchas y estrechas. Una etiqueta de código de barras se lee mediante un dispositivo manual que se desplaza por ella, o bien, el artículo se desliza por la estación de lectura. El dispositivo manual o la estación de lectura emiten luz que es

9.2.7.4. Lectura del código de barras

... tal que la información codificada en una serie de barras anchas y estrechas, una etiqueta de código de barras se lee mediante un dispositivo manual que se desplaza por ella, o bien, el artículo se desliza por la estación de lectura. El dispositivo manual o la estación de lectura emiten luz que es

Los fabricantes de productos alimenticios han adoptado un código de barras especial, llamado *Código de Productos Universal* (UPC), que se emplea en todos los productos destinados a supermercados y establecimientos similares.

9.2.7.5. Código de Productos Universal (UPC)

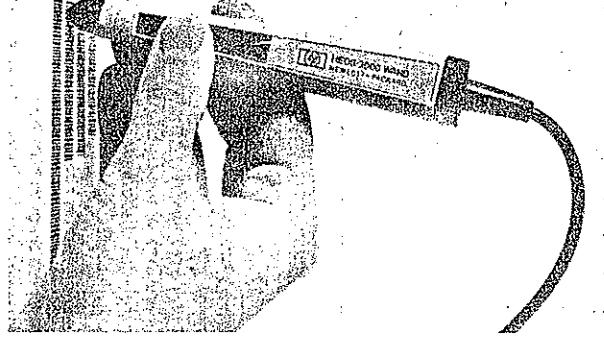


Fig. 9-13. La utilización del código de barras permite manejar la información de forma rápida, cómoda y segura. Cortesía de Hewlett Packard.

reflejada por las fotocélulas. Así, el computador averigua el número de barras y su anchura. Debe mantenerse constante la velocidad a que se mueve el artículo o la lectora para poder distinguir bien las barras anchas y las estrechas. Figura 9-13.

su

368 / Banting



Se alcanza una fiabilidad de un error único por cada varios millones de dígitos, algunas lectoras de códigos de barras están acopladas a sintetizadores de voz, que expresan oralmente el producto y su precio.

9.3. PERIFERICOS DE SALIDA

9.3.1. Impresoras

Las impresoras son los periféricos de salida más utilizados, puesto que de ellos se obtienen informes escritos, también denominados *copies duras* o *hard copies*. Estos informes pueden ser:

- Informes de detalle, con datos detallados de los ítems.
- Informes sumarios, con totales o información resumida de los anteriores.
- Informes de excepción, destacando algún apartado de interés.

9.3.1.1. Impresoras de impacto

En este tipo de impresoras, generalmente existe un martillo que golpea una varilla, la cual impresa el papel. El papel continuo se maneja en rollos o plegado en páginas. El papel, que dispone de unas perforaciones laterales, avanza por uno de los dos sistemas siguientes:

- *Fricción*: Un rodillo giratorio aprieta el papel y lo hace avanzar.
- *Tracción*: En los extremos del rodillo giratorio hay unas ruedas con salientes que enganchan en las tiras laterales perforadas del papel continuo, arrastrándole.

Impresoras de matriz de puntos

Disponen de un vector de martillitos o pines, de forma que, al golpear diferentes combinaciones de los mismos, se conforman los diversos caracteres. Este vector se desplaza rápidamente por la línea y a la altura de cada carácter se golpean los pines que lo configuran. Figura 9-14.

Esta impresora puede imprimir, además de caracteres, gráficos, gracias a las posibilidades de combinación de los puntos de la matriz. Sin embargo, la calidad de la escritura de caracteres no suele ser buena, aunque se ha obtenido una mejora de la definición de dos maneras:

- Aumentando los puntos de la matriz.

370 / Periféricos

— Realizando dos pasadas, la segunda de las cuales se hace desplazando ligeramente el cabezal de impresión, para que se intercalen los puntos con los de la primera pasada. También se realizan más de 2 pasadas.

Las velocidades de impresión varían desde 30 cps, (caracteres por segundo) hasta más de 300 cps.

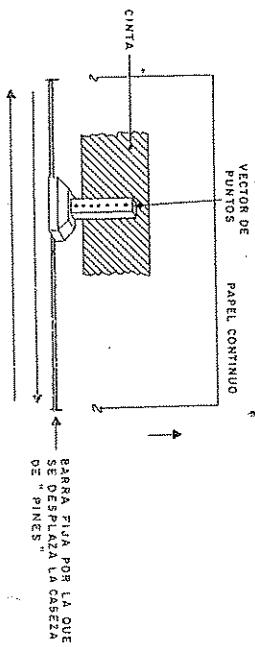


Fig. 9-14. Funcionamiento de una impresora de matriz de puntos y representación de la letra A, en una matriz de 5 X 7.

Impresoras de margarita

El cabezal está formado por una serie de varillas (una por carácter) alrededor de un disco giratorio, tal como se muestra en la figura 9-16.

El cabezal gira hasta que la varilla que tenga el carácter deseado se sitúe frente al papel; en ese momento, un martillo golpea la varilla imprimiendo el carácter.

Con estas impresoras se obtienen calidades de impresión similares a las de una máquina de escribir corriente. Sin embargo, su velocidad es bastante inferior a las impresoras de matriz de puntos. Las impresoras de margarita se utilizan para imprimir documentos en los que la calidad de impresión es primordial.

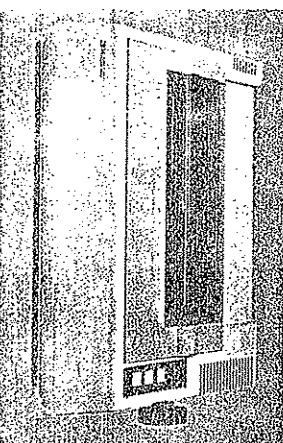


Fig. 9-15. Impresora gráfica IBM. Tiene una velocidad máxima de impresión de 80 caracteres, admitiendo 40, 80 y 132 caracteres por línea. La anchura del papel que emplea puede oscilar entre 10 y 25 centímetros. Incluye un microprocesador que controla todas sus funciones, además de encargarse de supervisar su operatividad.

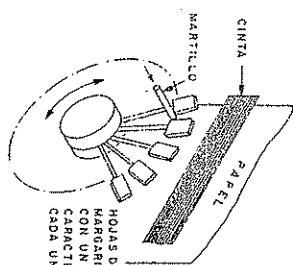


Fig. 9-16. impresión de un carácter en una impresora de margarita.

Impresoras de banda

Este tipo de impresoras corresponde al grupo denominado *impresoras de línea*. Una cinta que soporta todo el conjunto de caracteres gira en un plano perpendicular al papel. Figura 9-17. Cuando un carácter se encuentra a la altura de una posición que necesita ese carácter, un martillo golpea al carácter, que, al golpear una cinta entintada, lo imprime en el papel.

372 / Periféricos

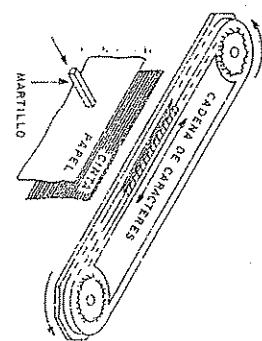


Fig. 9-17. Funcionamiento de una impresora de banda.

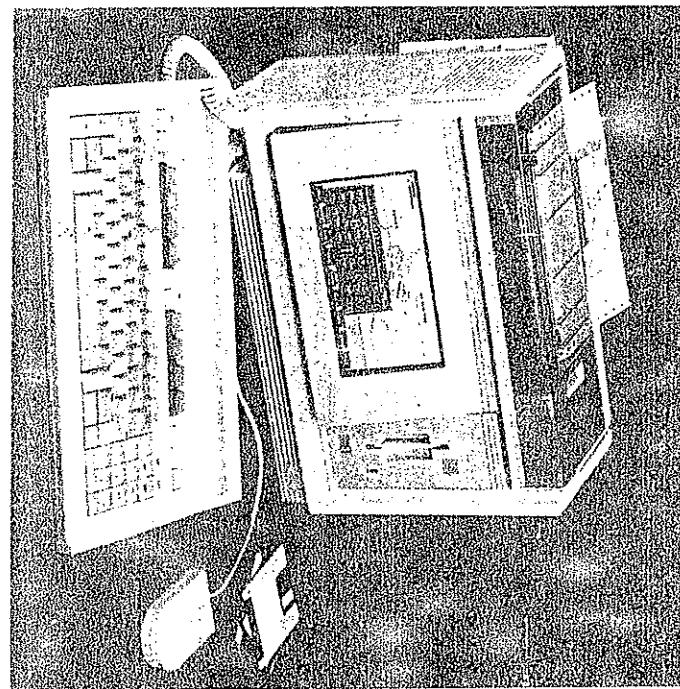


Fig. 9-18. Ordenador portátil, con impresora incorporada de 150 cps. Pantalla electroluminisciente, unidad de disco de 710 Kbytes y ratón. Cortesía de Hewlett Packard.



Impresoras de tambor

Pertenecen también al grupo de las impresoras de línea. El tambor es un cilindro. El cilindro está dividido en tantos sectores como letras tiene la línea y gira a velocidad constante. Existen tantos martillos como letras. El golpe de los martillos se produce cuando se sitúa la letra deseada delante del correspondiente martillo.

9.3.1.2. Impresoras sin impacto

Son impresoras que no utilizan el impacto para provocar la impresión de los caracteres, sino técnicas fotográficas, electrónicas, de impulsión de tinta, etc. SueLEN ser más silenciosas y rápidas.

Impresoras térmicas

Son impresoras de matriz de puntos, cuyos pines golpean directamente sobre el papel. Este papel tiene un tratamiento especial de forma que al golpear los pines calientes de la matriz, se coloreá, configurando el carácter.

Impresoras electrostáticas

Son impresoras de página, que utilizan rayos de luz o rayos láser para inducir cargas eléctricas en papel o un tambor de metal. Las áreas cargadas eléctricamente atraen la tinta química, para formar los caracteres o figuras.

Impresoras láser

Es el tipo más rápido de impresoras sin impacto. La luz precisa del láser hace posible guiar puntos de luz a posiciones exactas en una página. Los haces de luz pueden tratarse para imprimir caracteres, símbolos y gráficos de muy buena calidad y una variedad casi ilimitada.

Con este tipo de impresoras pueden obtenerse velocidades de impresión de dos páginas por segundo, o unas 18.000 líneas de texto por minuto. Algunas impresoras láser pueden imprimir el papel por las dos caras.

9.3.2. Impresoras gráficas y "plotters"

Sirven para reproducir curvas, gráficos, esquemas o cualquier dibujo artístico o lineal. Figura 9-19.

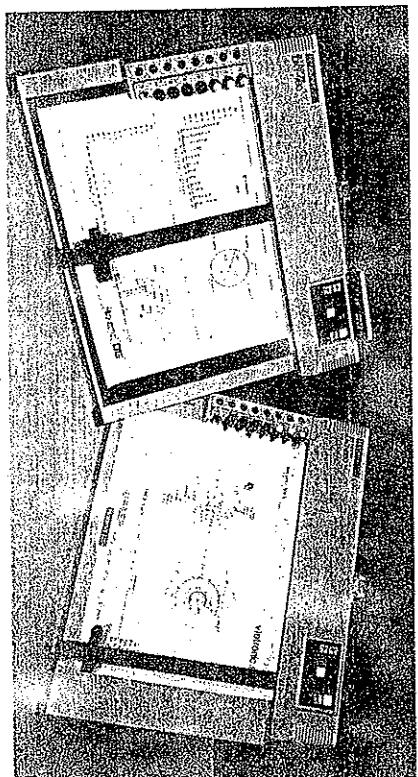


Fig. 9-19. Periféricos de salida de gráficos o plotters. Cortesía de Roland DG.

9.3.3. Pantallas de tubo de rayos catódicos

Las pantallas o terminales CRT son los periféricos de salida más populares. Suelen utilizarse como terminales de tiempo compartido y consolas para pequeños ordenadores. En la figura 9-20 se muestra un esquema interno de su estructura.

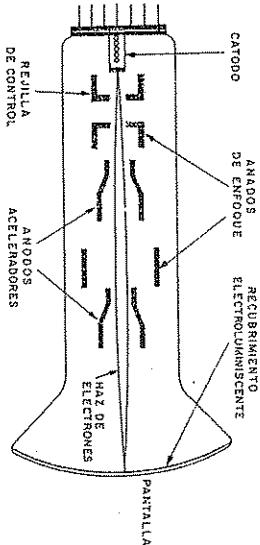


Fig. 9-20. Estructura interna de un tubo de rayos catódicos (CRT).

El funcionamiento básico de un tubo de rayos catódicos, como el mostrado en la figura 9-20, es el siguiente: debido a la temperatura que alcanza el catodo, calienta do por el filamento, emite un flujo de electrones que se concentran al pasar por la



rejilla de control. Los ánodos arrojan electrones que, desviados por los ánodos de enfoque, chocan a gran velocidad por la cara interna de la pantalla, que, al estar recubierta de un material fosforecente, se ilumina en los puntos así excitados.

Las características más importantes de un CRT son:

- **Scrolling:** Cuando se incluye una nueva línea en la pantalla, las líneas que ya estaban en ella se desplazan una línea hacia arriba, desapareciendo la superior y dejando un espacio libre abajo, para la nueva línea. Algunos sistemas permiten el scrolling hacia arriba y hacia abajo. Figura 9-21.

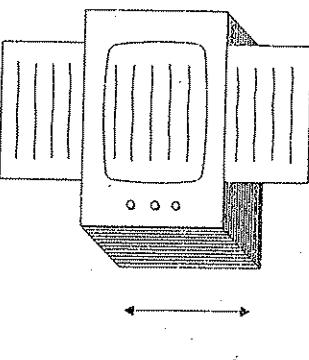


Fig. 9-21. Representación esquemática del funcionamiento del scrolling.

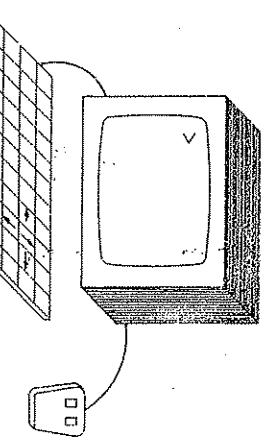


Fig. 9-22. Esquema del funcionamiento de la paginación de la pantalla.

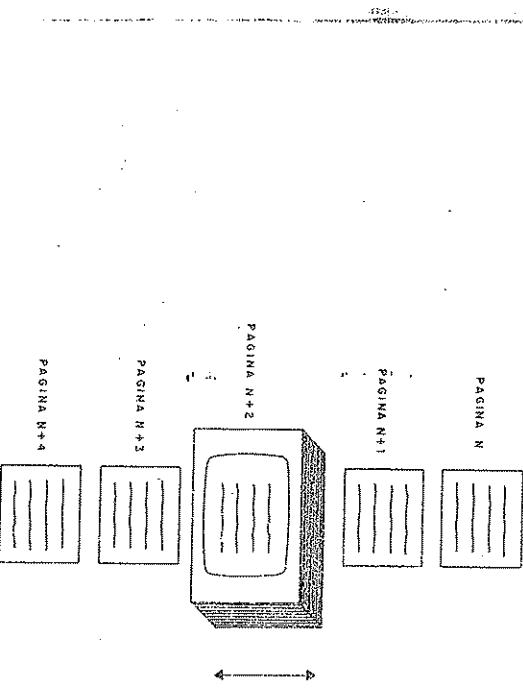


Fig. 9-23. Dos formas típicas de desplazar el cursor por la pantalla del computador. Mediante el teclado o con el ratón.

- **Doble página:** Algunos terminales de pantalla permiten almacenar en una memoria (que puede pertenecer al terminal o al computador), imágenes complejas de la pantalla, de manera que es posible visualizar una de ellas en cualquier momento, sin hacer scrolling. Figura 9-22.
- **Pantalla dividida:** Dentro de la imagen de la pantalla existen diferentes zonas que se pueden manejar independientemente. Por ejemplo, en una zona se puede efectuar scrolling y en las demás, no. En otra zona se puede ampliar o reducir su contenido, etc.
- **Campos reservados:** Los campos de datos aparecen, normalmente, luminosos sobre un fondo oscuro. Los campos reservados aparecen invertidos, es decir, sobre un fondo lumínoso se representan los caracteres oscuros, destacando su información.
- **Caracteres parpadeantes:** Para destacar ciertas informaciones, se hacen parpadear los caracteres.



- **Control de cursor:** Los controles normales del cursor permiten desplazamientos hacia arriba, hacia abajo, hacia la derecha y hacia la izquierda. También hay terminales, que, pulsando una de sus teclas ("HOME"), pueden enviar el cursor al extremo superior izquierdo. Figura 9.23.
- Un perfeíto muy popular es el terminal "teclado-pantalla", que proviene de la conjunción de un teclado de entrada y una pantalla de salida, entre los que produce el efecto "echo" apareciendo en la pantalla los caracteres que se van tecleando.

9.3.4. Salidas para microfilm

Para conseguir una salida rápida de impresos puede utilizarse el microfilm, con el que se consiguen velocidades de hasta 20.000 líneas o más por minuto. Además, si se dispone de procesamiento de imágenes Asistido por Computador, estas imágenes pueden visualizarse rápidamente por la pantalla.

Suelen obtenerse fichas de 4 X 6 pulgadas, conteniendo cientos de páginas cada una. Por tanto, ocupan muy poco espacio. Además, se pueden conseguir muchas copias con facilidad.

Entre los inconvenientes de este procedimiento figurarán: la necesidad de usar dispositivos especiales de lectura, la necesidad de renovación cuando se modifican los ficheros desde donde proviene la información, etc.

9.3.5. Sintetizadores de voz

Son dispositivos capaces de reproducir palabras sonoras, que se están haciendo cada día más populares. El primer producto comercial producido en gran escala, que incorporó la síntesis de voz fue el juguete Speak and Spell de Texas Instruments. Pero la síntesis de voz ya se ha empleado en mensajes telefónicos, automóviles, enseñanza de idiomas, interacción con computadores, etc. Sus limitaciones más importantes provienen de la reducción de su vocabulario y la desviación del sonido producido con respecto a la voz humana. Figura 9.24.

La voz humana está compuesta por un conjunto de ondas analógicas. El computador deberá digitalizar esta información con lo que se perderá parte de la misma.

El número de fonemas utilizados en un idioma son finitos, pudiéndose almacenar en código digital en la memoria de un computador la información de cada uno. Concatenando los fonemas y convirtiéndolos en una señal analógica, se reproducen palabras o frases completas.

Otra manera de guardar la información de los sonidos en el computador es mediante ecuaciones matemáticas. Estas se resuelven para producir ondas analógicas parecidas a las del sonido original.

9.3.6. Salida en tres dimensiones

Algunos computadores pueden generar hologramas mediante el empleo de rayos láser. Estos hologramas son imágenes tridimensionales, alrededor de las cuales se puede situar el usuario y verlas desde cualquier posición. Los hologramas contienen una enorme cantidad de información descriptiva.

Con la aparición de los supercomputadores se podrán generar imágenes tridimensionales a una velocidad de 30 imágenes por segundo, con lo que se logra visualizar imágenes tridimensionales en movimiento.

9.4. PERIFÉRICOS DE ALMACENAMIENTO

Dentro de este grupo de periféricos de entrada y salida, se dedica una atención especial a los dispositivos de almacenamiento masivo de información de tipo magnético, dada su gran utilización. Como estos periféricos se relacionan directamente con la UCP y no con el usuario, se estudian en profundidad las características técnicas que les rodean, además de las descriptivas que hacen referencia a su manejo.

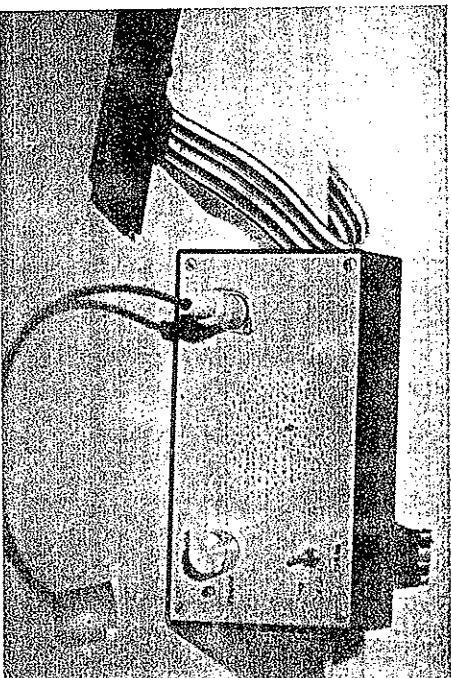


Fig. 9.24. Fotografía de un sintetizador de voz, conectable y controlado desde un microcomputador.



9.5. MEMORIAS MAGNÉTICAS. FUNDAMENTOS DE LA GRABACIÓN MAGNÉTICA

La práctica totalidad de los dispositivos de memoria secundaria y auxiliar son de tipo magnético, por lo que es conveniente conocer los fundamentos en los que se basa el principio de grabación magnética.

9.5.1. Medio de grabación magnético

Existen dos medios típicos para grabación magnética: los medios *flexibles* (cintas, diskettes, etc.) y los medios *duros* (discos y tambores). En ambos se aplica el mismo principio: se deposita una pequeña capa de material magnetizable (óxidos o metales) sobre un soporte, que en un caso es de tipo flexible y en otro, rígido o duro.

El almacenamiento se hace creando dominios magnéticos de polarización inversa. Para ello, "hay que dotar a la película magnética de una dirección preferente de polarización". Normalmente, esta dirección es horizontal (en dirección de la cinta magnética) y los dominios magnéticos se forman tal como se indica en la figura 9-25 a). Se están desarrollando sistemas cuyo medio tiene la dirección de magnetización preferente perpendicular a la capa (figura 9-25 b)), lo cual permite elevar considerablemente la densidad de grabación.

Las características del medio vienen determinadas por las propiedades de la capa magnetizable y del soporte. Estas características determinan la densidad de grabación y la relación señal/ruido de lectura. Entre las más importantes se citan las siguientes:

- Tamaño de la partícula de óxido o del grano en los medios metálicos. Cuanto más pequeño, mayor puede ser la capacidad de almacenamiento, puesto que los dominios magnéticos son más pequeños.
- Fuerza coercitiva H_c y la densidad de flujo residual B_r . La fuerza coercitiva H_c es la que hace que el material permanezca magnetizado. Por un lado, la tendencia a desmagnetizarse es mayor cuanto más pequeños sean los dominios. Por otro lado, B_r crece con H_c . Como la señal de lectura es directamente proporcional a B_r , interesa que, tanto la fuerza coercitiva como la densidad de flujo residual, sean grandes.
- Espesor de la capa metálica. Cuanto más fina sea, menores pueden ser los dominios y mayor la densidad. Sin embargo, las señales leídas son más débiles. Los espesores son del orden de 10^{-2} mm.

— Uniformidad de la superficie y estabilidad del soporte. Es importante que la superficie del soporte no tenga rugosidades, para que la capa magnética sea uniforme y lisa. Además, conviene que no existan variaciones de tamaño en el soporte con el tiempo, pues se modificaría la posición de las pistas, lo que crearía problemas de lectura.

Los materiales magnéticos más empleados son los siguientes:

1. Oxido de hierro.
2. Oxido de hierro-cobalto.
3. Oxido de cromo.
4. Película metálica de hierro, hierro-cobalto, cobalto-níquel, etc.

9.5.2. Grabación y lectura. Códigos

Los transductores de lectura y escritura para grabación magnética se componen de unas *cabezas* de estructura toroidal, con un pequeño entrehielro. La figura 9-26 representa el transductor, que tiene un arrollamiento por el que circulan las corrientes de escritura o lectura. Aunque en ocasiones se emplean dos transductores, es usual que solo haya uno, tanto para la lectura como para la escritura.

El núcleo de la cabeza o toro se construye de ferrita o laminado de hierro.

Para efectuar la operación de escritura se hace pasar una corriente por el arrollamiento de la cabeza. Esta corriente produce un campo magnético en el núcleo de la cabeza, el cual se dispersa en el entrehielro como se muestra en la figura 9-27.

Fig. 9-25. A la película magnética se le dota de una dirección preferente de magnetización.

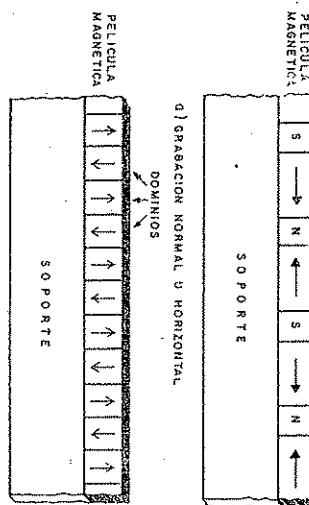




Fig. 9-26. Cabeza transductora de lectura y escritura para grabación magnética.

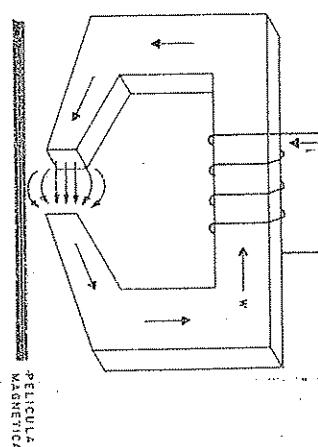


Fig. 9-27. Para escribir sobre la película magnética se hace pasar una corriente por la bobina de la cabeza transductora.

Si se mueve una película de material magnético en las proximidades del campo del entrehierro, la zona afectada quedará magnetizada. Invertiendo el sentido de la corriente se invierte el sentido del campo y, por tanto, el sentido de magnetización del soporte.

Si se mueve delante de la cabeza un soporte magnetizado, los cambios de magnetización del soporte (bordes de los dominios) producen unos cambios magnéticos en el toro, que, a su vez, inducen corrientes en el arrollamiento. Figura 9-28. Las corrientes producidas son proporcionales a la densidad de flujo residual B_r . Obsérvese que el motivo que produce la corriente en el arrollamiento es el cambio de magneti-

zación que existe en los bordes de los dominios y no la magnetización de los propios dominios.

Hay dos formas básicas de grabar un soporte magnético:

1. Por impulsos de corriente.
2. Por commutación de corriente.

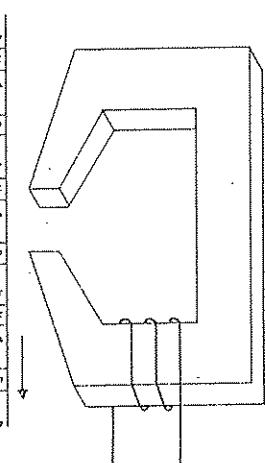


Fig. 9-28. Al moverse la película magnética frente a la cabeza generan una corriente en el arrollamiento como consecuencia del cambio de magnetización existente en los bordes de los dominios.

La grabación por impulsos está representada en la figura 9-29 a). Se parte de un medio desmagnetizado, $M = 0$, en el cual se pueden grabar dominios de magnetización inversa M_r y $-M_r$, mediante *impulsos de corriente en sentidos contrarios*. El tamaño de los dominios viene dado por el ancho del impulso y por la velocidad con la que se desplace el medio.

En la grabación por commutación de corriente, como puede apreciarse en la figura 9-29 b), siempre se está aplicando corriente a la cabeza grabadora, pero se *commuta entre los valores +I y -I*, lo que da lugar a ir grabando dominios de magnetización positiva $+M_r$ y negativa $-M_r$. En principio, dado que siempre está magnetizado el medio, no es necesario que esté desmagnetizado antes de la grabación.

La lectura del soporte magnético produce un impulso de corriente por cada cambio de polaridad en su magnetización. La figura 9-30 muestra los impulsos leídos en un medio grabado por el procedimiento de impulsos de corriente (caso a) y por el de commutación de corriente (caso b).

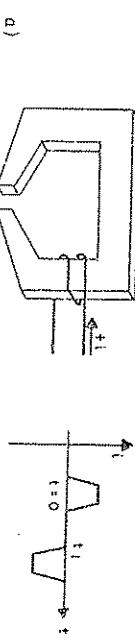
En el primer caso de la figura 9-30, cada dominio magnético puede hacerse corresponder a un punto de memoria o bit. En efecto, si se asigna a $+M_r$ el valor "1" y a $-M_r$ el valor "0", y se parte de una zona con M_0 , al encontrar un impulso posi-



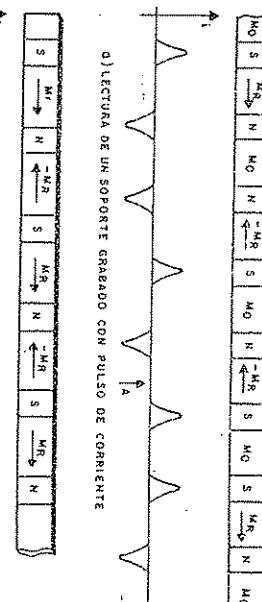
tivo seguido de otro negativo, quiere decir que ha pasado un dominio con $-M_r$ y por tanto un "0". Si, por el contrario, se obtiene un impulso negativo seguido de uno positivo, significa que hay $+M_r$ y por tanto un "1".

Si no se lee de forma sincronizada, se producen errores. Así, si se comienza a leer a partir de la marca A (figura 9-30) se lee un impulso negativo seguido de otro positivo, o sea, un "1", cuando en realidad existe una zona de separación entre bits. A continuación se leen dos impulsos negativos seguidos, lo que significa error de sincronismo. Además, permite sincronizar la lectura, tomando la zona entre esos dos impulsos como una zona M_0 .

Si no se lee de forma sincronizada, se producen errores. Así, si se comienza a leer a partir de la marca A (figura 9-30) se lee un impulso negativo seguido de otro positivo, o sea, un "1", cuando en realidad existe una zona de separación entre bits. A continuación se leen dos impulsos negativos seguidos, lo que significa error de sincronismo. Además, permite sincronizar la lectura, tomando la zona entre esos dos impulsos como una zona M_0 .



a) LECTURA DE UN SOPORTE GRABADO CON PULSO DE CORRIENTE



b) LECTURA DE UN SOPORTE GRABADO CON CONMUTACIÓN DE CORRIENTE

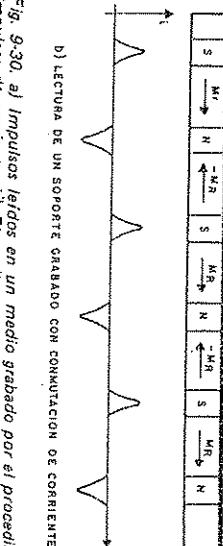


Fig. 9-30. a) Impulsos leídos en un medio grabado por el procedimiento de impulsos de corriente. b) El medio ha sido grabado por el procedimiento de conmutación de corriente.

Esta forma de grabar y leer se llama *de retorno a cero* (RZ: return to zero), y cumple todos los requisitos para ser empleada:

- Tiene grabado el reloj de lectura, ya que los impulsos de corriente sirven de reloj al indicar dónde se encuentran los puntos de memoria.
- Permite sincronizarse con la señal grabada.
- Permite diferenciar entre el "1" y el "0".

El grave inconveniente de la forma de grabar RZ es que desperdicia mucho espacio, pues las zonas $M = 0$ no almacenan información.

Se han diseñado otros códigos de grabación, basándose en el método de conmutación de corriente. Dichos códigos introducen la necesaria información adicional (figura 9-30 b)) no permite detectar los puntos de memoria.

La tabla de la figura 9-31 recoge los códigos más frecuentes.

CÓDIGOS	RELOJ EXTERNO	NRZ	
		SIN RETORNO A CERC	NRZ, 1 RB
RELOJ	INTERNO	SIN RETORNO	
		A CERO	FM PE
INTERNO	CON RETORNO	A CERO	
		RZ	

Fig. 9-31. Tabla con los códigos de grabación más frecuentes.

Los códigos se han clasificado en dos grandes apartados, aquellos que necesitan un reloj externo, por ejemplo, grabado en otra pista, y los que contienen su propio reloj y son autosuficientes.

Se exponen las características más relevantes de los códigos de grabación de la tabla de la figura 9-31.

Código RZ (Return to zero)

- Presenta dos puntos por bit almacenado.
 - Se detecta un uno al obtener un impulso — seguido de uno +.
 - Se detecta un uno al obtener un impulso — seguido de uno +.
 - Tiene reloj y permite sincronizar la lectura.
 - Emplea mucho espacio y tiene poca densidad.
- En la figura 9-32 a) se muestra la estructura de este código.

Código NRZ (Non return to zero)

Se obtiene del RZ eliminando las zonas de $M = 0$ y las parejas NS correspondientes.

- Presenta señal sólo para los cambios de 0-1 y de 1-0; cambios que representan un 1.
- Necesita sincronización y reloj externos. La pérdida de sincronización no se detecta y produce información errónea.
- En el caso de producirse un error, quedan afectados todos los bits posteriores.
- Permite una gran densidad de grabación.

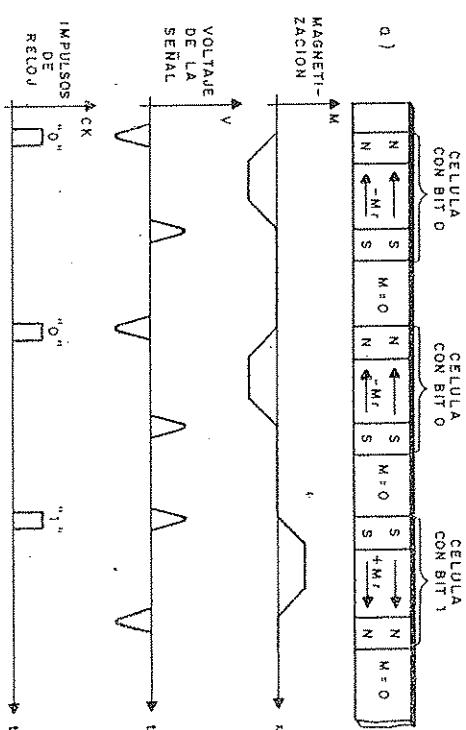


Fig. 9-32. a) Estructura del código RZ; b) estructura del código NRZ.



Código NRZI (Non return to zero inverted)

Es una variación del código anterior, donde el cambio de polaridad de la magnetización representa un 1.

- Presenta impulsos sólo para los 1.
- Necesita sincronización y reloj externos. No se detecta la pérdida de sincronización.
- No produce propagación de error.
- La inexistencia de señal se interpreta como 0.
- Permite una gran densidad de grabación.
- Es muy utilizado en bandas magnéticas, obteniéndose el reloj y la sincronización mediante una pista auxiliar, que sirve, además, de bit de paridad.

En la figura 9-33 se muestra la estructura del código NRZI.

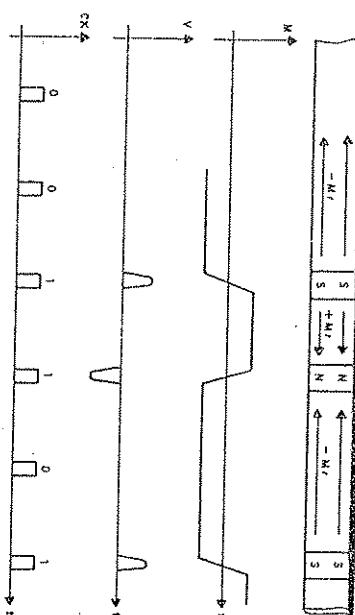


Fig. 9-33. Estructura del código NRZI.

Código PE o FM (Frequency modulation)

Es una modificación del anterior, donde se ha introducido un impulso auxiliar de reloj por bit.

- Reloj autocontenido, pero que ha de sincronizarse.
- Permite utilizar una sola pista.
- La falta de señal se toma como 0.
- La falta de un impulso de reloj conlleva la pérdida de sincronismo.

388 / Periféricos



- Permite una densidad mitad que el código NRZI por el espacio necesario para los impulsos de reloj.
- Se emplea en los disketes de simple densidad y en muchas unidades de disco.

Código PE (Phase encoding)

Un cero se representa de tal forma que genera un impulso negativo, mientras que el uno genera un impulso positivo.

- Reloj autocontenido, pues cada bit representa un impulso.
- Hay que eliminar los impulsos auxiliares, pero la sincronización es muy fácil.
- Teóricamente presentaría una densidad mitad que el código NRZI, pero en la práctica se puede comprimir más la información por su propiedad de reloj autocontenido.
- La falta de señal no se toma como 0.
- Se utiliza en las bandas magnéticas de mayor densidad de grabación.

Código RB (Return to bias)

Sólo deriva del RZ haciendo que las zonas no magnetizadas se conviertan en —M. De esta forma los valores "1" se almacenan como en RZ, mientras que los valores "0" desaparecen como en el NRZI.

- Requiere reloj externo, pero el sincronismo es automático.
- Tiene una densidad de grabación mitad que el NRZI.
- La falta de señal se toma como 0.

Código MFM (Modified frequency modulation)

Se deriva del FM, eliminando los impulsos de reloj innecesarios. Tiene propiedades parecidas al FM, pero permite doble densidad de grabación.

- Se emplea en los diskettes de doble densidad.

9.6. BANDA MAGNETICA

Las unidades de banda magnética utilizan como soporte una cinta de Mylar de $1/2$ pulgadas de ancho y del orden de 3×10^2 mm de espesor, en la que se ha depositado una capa de óxido de hierro o de cromo de unos $1,5 \times 10^{-2}$ mm de espesor. Estas bandas se fabrican en carretes de unos 800 m de longitud.

- Se emplean unidades de 9 pistas, lo que significa que los $0,5''$ de ancho se dividen en nueve zonas, cada una asignada a su correspondiente cabeza de lectura-escritura.
- Se leen 9 bits en paralelo: 8 de datos y uno de paridad. La figura 9-34 muestra la

388 / Periféricos

división de la cinta, así como el significado de los distintos bits para grabaciones en el código EBCDIC. Se utiliza una palabra de control de error horizontal, además de los bits de paridad verticales. La existencia del bit de paridad vertical hace que todo byte tenga al menos un "1", con lo que se garantiza la generación de reloj en el caso de usar el código NRZI.

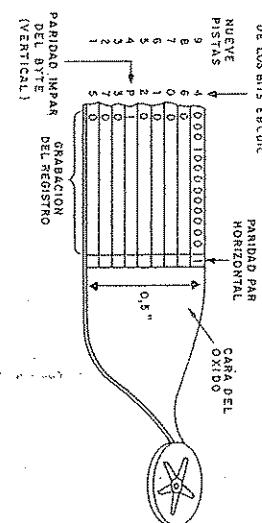


Fig. 9-34. Organización de la cinta magnética dividida en 9 pistas.

La tabla de la figura 9-35 recoge las principales características de algunos sistemas de banda magnética.

VELOCIDAD CINTA (PULGADAS/SEGUNDO)	VELOCIDAD MANTENIDA (K BYTES/SEGUNDO)	DENSIDAD LINEAL (BPI)	TIEMPO REPRODUCCION (SEGUNDOS)	CÓDIGO DE GRABACIÓN
18,75	15	800	MINUTOS	NRZI
37,5	30	800	MINUTOS	NRZI
100	150/750	1600/800	72	PE/NRZI
125	200/1000	1800/800	55	PE/NRZI
200	320/160/11,2	1600/800/556	45-50	PE/NRZI/NRZI
250	800	35,00	4,5	PE

Fig. 9-35. Tabla con las características fundamentales de algunos sistemas de banda magnética.

El dispositivo de banda magnética completo se ofrece en la figura 9-36. Hay dos bobinas de banda con sus correspondientes mecanismos de arrastre. También hay dos poleas encargadas de arrastrar la banda a una velocidad constante frente a las cabezas de lectura-escritura. Dichas poleas poseen un bucle de banda.

cuyo objeto es amortiguar los tirones de bobinado de los motores. Además de las cabezas de lectura-escritura, hay un circuito electrónico para la amplificación de las señales y el sistema de control de velocidad de las poleas y de control de bobinado.

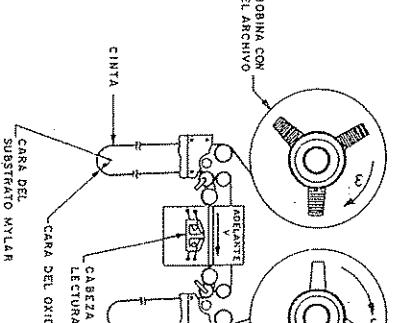


Fig. 9-36. Dispositivo completo de banda magnética.

No toda la longitud de la cinta puede emplearse para almacenar datos, ya que también hay que guardar en ella información de dirección para poder acceder a la zona de datos deseada y las poleas necesitan un cierto tiempo para parar y alcanzar la velocidad de régimen de la cinta, por lo que hay que dejar unos claros o zonas muertas entre los registros independientes. Estos claros reciben el nombre de IRG (Inter register gaps) y tienen 1,5 cm aproximadamente.

Para reducir este último inconveniente, se unen varios registros en un bloque, eliminando los claros entre ellos. La figura 9-37 representa una cinta grabada, que tiene organizada la información en ficheros, cada uno con varios registros, que pueden formar bloques o no. Cada fichero tiene su directorio que especifica los registros contenidos.

En la figura 9-38 se aprecia cómo cada registro tiene una cabecera con su dirección y zona de sincronismo, una cola con un byte de detección de error y una zona de fin de registro. La unidad sólo permite leer y grabar en un sentido, pero puede retroceder un número determinado de registros, detectando el fin de los mismos.

Las características principales de los dispositivos de banda magnética son:

- Memoria dinámica, pues la cinta ha de moverse delante de las cabezas.

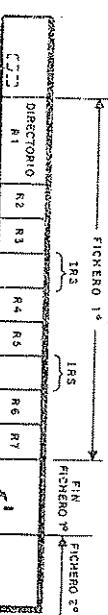


Fig. 9-37. Representación de una cinta grabada en la que se han agrupado los registros para eliminar los claros.

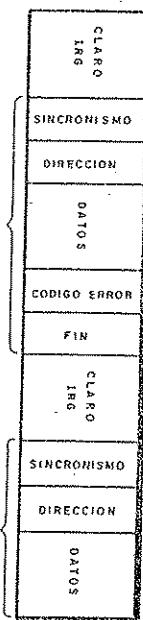


Fig. 9-38. Estructura de los registros en la cinta magnética.

- Hay contacto físico entre la banda y las cabezas.
- Funcionan con 9 pistas, pudiéndose leer o grabar caracteres en paralelo.
- El soporte es muy barato.
- La grabación es horizontal y los códigos que se emplean son el PE y el NRZI.
- El acceso es puramente secuencial, teniendo un tiempo de acceso elevado.
- No se puede intercalar información adicional; hay que regrabar todo el resto de la cinta.
- Memoria no volátil. La lectura no es destructiva.

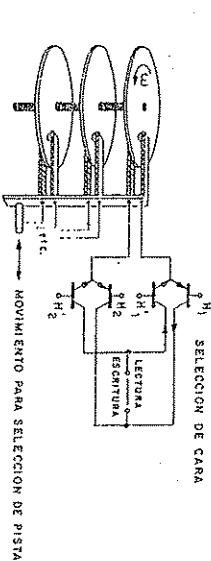
Además de las unidades de banda de 1/2 pulgada hay diversos tipos de cartucho, pero no han tenido gran difusión. Se basan en los mismos principios descritos, pero las prestaciones son menores y su mecanismo de arrastre suele ser mucho más sencillo.

El uso de las cosetas de audio como almacenamiento en sistemas domésticos, se basa en grabar la información en serie, bit a bit. La grabación se hace modulando la información en FSK. El método es lento, de poca capacidad y poco fiable, pero muy económico.

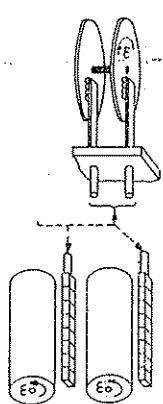
9.7. TAMBORES Y DISCOS

Los discos y los tambores se usan como almacenamiento auxiliar o secundario. Se llaman sistemas de acceso directo, porque se puede acceder a la información de forma más directa que en las bandas magnéticas. Su funcionamiento se esquema tiza en la figura 9-39.

El soporte es un disco o tambor recubierto por una fina película magnética, que gira a gran velocidad. El sistema puede tener una sola cabeza de lectura-escritura por superficie, como se muestra en la figura 9-39 a) o varias, tal como aparece en la figura 9-39 b). En el primer caso se asigna el nombre al sistema de *brazo o cabeza móvil* y en el segundo, de *cabeza fija o cabeza por pista*. El sistema puede tener una o varias superficies formadas por discos o tambores.



a) SISTEMA DE CABEZA MÓVIL



b) SISTEMA DE UNA CABEZA POR PISTA

Fig. 9-39. Esquema de funcionamiento de los discos y tambores.

La capacidad de almacenamiento se compone de:

1. Pistas.
2. Sectores.
3. Cilindros.



Una pista es la tira del soporte de almacenamiento que gira delante de una cabeza móvil. En los sistemas de cabeza fija, cada una de ellas define una pista. En los sistemas de cabeza móvil, el brazo puede adoptar una serie de posiciones y cada posición de cada cabeza constituye una pista. La figura 9-40 presenta la división de un disco en pistas. Como se indica en la figura 9-40 b), cada byte se graba en serie en una pista determinada, al contrario de como se hace en las cintas magnéticas.

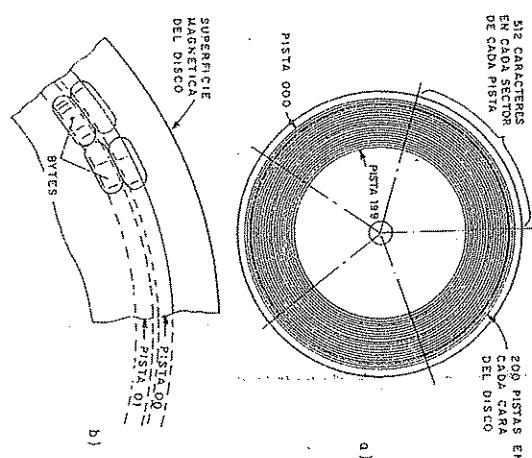


Fig. 9-40. División de un disco en pistas.

Las pistas se dividen en sectores. Cada sector constituye la unidad de información que es transferida en un acceso.

Finalmente, en los sistemas de varias superficies y brazo móvil, aquellas pistas que se acceden en una posición del brazo, constituyen un cilindro. Notese que la selección de la pista dentro del cilindro es un direccionamiento de tipo cableado y se hará por medios electrónicos.

El direccionamiento de la información exige la selección de la cabeza correspondiente (direcciónamiento cableado), del posicionamiento del brazo en caso de ser móvil (direcciónamiento necesario) y la interpretación de la información leída de la pista para seleccionar el sector deseado.



En general, las cabezas de lectura-escritura no tocan el soporte, para evitar su desgaste. Por el contrario, se desplazan a una distancia de unos 10^{-4} mm de la superficie del disco. Esto se consigue gracias al aire que desplaza el disco al girar con velocidades lineales del orden de 100 km/h . Un problema importante de estos dispositivos son los cortes de alimentación, pues, al disminuir la velocidad del disco o tambor, disminuye la fuerza ascendente de la cabeza, que tiende a chocar con la superficie del disco o tambor. Para evitar este choque, se suele disponer de un sistema que separa las cabezas antes de que el disco o tambor pierda velocidad.

En los dispositivos más económicos, tipo diskette, la cabeza apoya sobre el soporte, por lo que la vida de ambos es reducida.

Las velocidades de giro son del orden de 3.000 rpm , aunque se han alcanzado 24.000 rpm en tambores.

9.7.1. Formatos

Para poder reconocer la información del dispositivo hay que añadir una información de direccionamiento y, a veces, de sincronismo. El formato de grabación especifica esta información. Al igual que en las bandas magnéticas, la información se engloba en registros, siendo un registro la mínima cantidad de información que se transfiere en una operación de lectura o escritura.

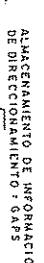


Fig. 9-41. Una posible disposición de las marcas, que se detectan mediante una fotocélula.



El formato puede ser de tipo hardware o software. En el primer caso, existen unas marcas fijas que indican el comienzo de cada registro, que, por lo tanto, son de tamaño fijo. La figura 9-41 indica una posible disposición de las marcas que se detectan mediante una fotocélula.

En el segundo caso, (figura 9-42), sólo hay una marca que indica el comienzo de las pistas. Los distintos registros, que pueden ser de longitud variable, se identifican por su cabecera. A modo de ejemplo, la figura 9-43 representa el formato IBM 3740 para diskettes de 8" de simple densidad con sectores de 128 bytes. Notense las zonas de sincronismo, de códigos de error, de dirección, etc. De un total de 5.208 bytes, sólo se destinan para datos 3.328, lo que da una relación capacidad/neta/capacidad bruta del 64%.

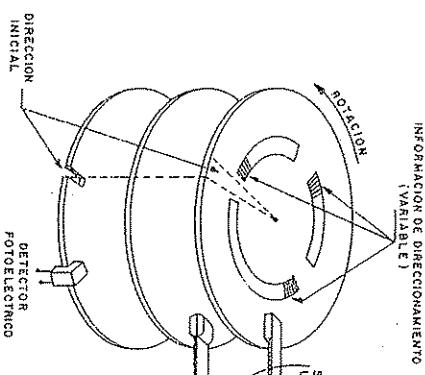


Fig. 9-42. En el formato tipo software, sólo existe una marca al comienzo de las pistas. Los registros se identifican por su propia cabeza.

9.7.2. Tiempo de acceso

El tiempo de acceso de estos dispositivos viene dado por el tiempo que tardan en posicionar el brazo en la pista deseada (*tiempo de búsqueda*), más el tiempo que tarda la información de la pista en pasar delante de la cabeza (*tiempo de latencia*). Ambos tiempos dependen de la posición de partida y de la posición deseada. Así, no se tarda lo mismo en mover el brazo de la pista 1 a la 2, que de la 1 a la 75. Por este motivo se proporcionan los valores medios.



O AGUJERO PRINCIPAL DE PISTA								
SECTOR N°	GAP	GAP	SYNC	GAP	SECTOR 1	SECTOR 2	-----	SECTOR 25
4	5	5	SYNC	2	SECTOR 1	SECTOR 2	-----	SECTOR 25

M.D	IDENIFICADOR	GAP	SYNC	CAMP. DE DATOS	GAP	M.D	DATOS EFECTIVOS	CRC 1	CRC 2
1	1	2	2	128 BYTES	3	1	128 BYTES	1	1

a) ESTRUCTURA DE CADA PISTA DEL FORMATO
b) TAMAÑO Y CONTENIDO DE LOS ELEMENTOS DEL FORMATO.

Fig. 9-43. Formato IBM 3740 de 25 sectores de 128 bytes.

El tiempo de rotación a veces debe incluir un tiempo de sincronización hasta que el transductor sea capaz de interpretar la información.

Por ejemplo, una unidad de diskettes de 8" en formato software requiere las siguientes operaciones antes de alcanzar un registro determinado:

1. Mover el brazo a la pista deseada (tiempo medio, 100 ms).
2. Esperar a que se establezca la cabeza (de 0 a 20 ms).
3. Actuar sobre la cabeza para apoyarla sobre el diskette, ya que suele estar levantada para minimizar el desgaste (de 25 a 60 ms).
4. Esperar la detección del agujero de comienzo de pista (tiempo medio = 1/2 revolución = 83,3 ms). En muchos dispositivos el sincronismo se alcanza con la primera cabecera de sector, por lo que este tiempo queda eliminado.
5. Esperar a que llegue el sector deseado (tiempo medio = 1/2 revolución = 83,3 ms).



El tiempo total de acceso es del orden de los 430 ms. En contraposición, un sistema de cabeza fija sólo requiere el último de los tiempos indicados. Por tanto, si gira a 3.000 rpm, tendrá un tiempo medio de acceso equivalente a 1/2 revolución, o sea, 10 ms.

9.7.3. Velocidad de transferencia

La velocidad a la que se transmiten los bits de un registro, viene dada por la *velocidad de giro*, por la *densidad de grabación* y por el *radio del medio*.

Si la densidad de grabación es $d = 2.000 \text{ bits/cm}^2$, con un radio $R = 20 \text{ cm}$ y una velocidad de 3.600 rpm, el número de bits en la pista será $2\pi R d = 126.000$. Luego pudiendo bytes de 8 bits más paridad, se obtiene una velocidad de transferencia de 0,84 Mbytes/s.

En una unidad de disco determinado, el número de bits grabados por pista es fijo, resultando que las pistas interiores disponen de mayor densidad de grabación que las exteriores, al ser más cortas.

Un resumen de las principales características de las memorias de disco y tambor es el siguiente:

- Memorias dinámicas de acceso directo.
- No volátiles.
- Permiten lectura (no destructiva) y escritura.
- Graban los caracteres en serie, pues sus pistas son de 1 bit de ancho.
- Tiempos de acceso de pocos ms en las unidades más rápidas.
- Capacidad de almacenamiento de hasta 2 Gbytes por unidad.
- Modelos de soporte fijo y modelos de soporte intercambiable.
- En la actualidad han caído en desuso los tambores.
- Los códigos usados son el FFM y el MFM.

Recientemente han surgido unidades con magnetización vertical, lo que aumenta la densidad de grabación.

Como ejemplo de una unidad moderna de disco se puede citar el IBM 3780 con una capacidad de 2,2 Gbytes, un tiempo de acceso medio de 16 ms y una cadencia de transferencia de 3 Mbytes/s.

9.8. MEMORIAS DE BURBUJAS MAGNETICAS

Este tipo de memoria constituye una de las posibles alternativas para la sustitución de los discos.



Los avances tecnológicos en la fabricación de circuitos integrados LSI y VLSI han permitido mejorar las características de las memorias de semiconductores en pocos años. No obstante, las memorias RAM son volátiles y de limitada capacidad.

Por otra parte, las memorias periféricas de alta capacidad requieren largos tiempos de acceso, elementos electromecánicos móviles e interfaz complejo.

En el futuro solamente podrá extenderse un nuevo modelo de memoria, que elimine la volatilidad de las RAM y los dispositivos electromecánicos móviles. Este tipo de memoria puede ser el de burbujas magnéticas, (abreviadamente MBM) cuyas principales características son:

1. No ser volátiles.
2. Ser estáticas, no precisando partes móviles.
3. De fabricación parecida a la de los circuitos integrados.
4. Su coste tiende a la baja, dado el constante crecimiento de su consumo.
5. Ofrecer mayor fiabilidad que los restantes tipos de memorias.
6. Presentar alta resistencia a los ambientes hostiles (campos magnéticos, vibraciones, temperatura, ruidos, etc.).

VENTAJAS DE LAS MBM

MEMORIAS DE SEMICONDUCTORES		DISCOS, CINTAS Y TAMBORES MAGNETICOS	
RAM	PROM Y REPROM	NO VOLATILES	TAMBORES SIN PARTES MOVILES
MUY RESISTENTES	FACILES DE GRABAR Y BORRAR	MUY RESISTENTES	MUY RESISTENTES
BAJO CONSUMO	MAYOR CAPACIDAD	BAJO CONSUMO	MAYOR FIABILIDAD
MAYOR CAPACIDAD	MAS BARATAS	MAS RAPIDAS	
MAS BARATOS			

DESVANTAJAS DE LAS MBM

MEMORIAS DE SEMICONDUCTORES		DISCOS, CINTAS Y TAMBORES MAGNETICOS	
RAM, PROM Y REPROM	INTERFAZ COMPLEJA	TAMBORES MAGNETICOS	MENOS CAPACIDAD
INTERFAZ COMPLEJA	MAYOR TIEMPO DE ACCESO	MENOS CAPACIDAD	
MAYOR TIEMPO DE TRANSFERENCIA	MAS CARAS		

Fig. 9-44. Cuadros con las ventajas e inconvenientes de las MBM frente a las memorias de semiconductores y las de tipo magnético.



En la tabla de la figura 9-44 se presentan las ventajas y desventajas de las MBM diferentes a los tipos de memorias más importantes.

Tengase en cuenta que, aunque las primeras noticias sobre las MBM surgieron en 1967, no fueron que transcurri más de 10 años para que apareciese en el mercado el primer modelo comercial: el TIB 103 de Texas Instruments, que tenía una capacidad de 92.304 bits, un tiempo de acceso de 4 ms y un coste cercano a los 40 \$.

Actualmente, las MBM se aplican en el campo de los microordenadores, de los aparatos portátiles y de las máquinas industriales que trabajan en ambientes hostiles.

En los microordenadores, las MBM sustituyen a las memorias de semiconductores y, especialmente, a los discos flexibles. En el mercado existen bastantes modelos que incorporan este tipo de memoria a precios realmente bajos.

En los ordenadores de cierta envergadura, las MBM se presentan como una alternativa a los discos, por su velocidad y reducido tamaño, aunque, quizás, en una primera fase coexisten ambos tipos de memorias.

Más del 15% de la producción de MBM se consume en aplicaciones industriales, en especial, en maquinaria-herramienta y robótica. Las MBM resultan muy idóneas en ambientes duros, por lo que son muy eficaces en el control parcial o total de procesos y en el mando de maquinaria bajo control de sistemas basados en microprocesadores.

9.8.1. Principios y tecnología

Una burbuja magnética consiste en un diminuto dominio magnético, de forma cilíndrica, con un diámetro de muy pocas micras, creado sobre una película de granate magnético de algunas micras de espesor. Al aplicar sobre dicha capa un campo magnético exterior, perpendicular a la superficie, el tamaño de los dominios magnéticos de popularidad opuesta al campo va disminuyendo a medida que aumenta el valor del mismo. Figura 9-45.

La presencia o ausencia de burbuja en un punto de memoria define el bit 1 o 0. El principal inconveniente de las MBM es que, al disponer de una sola cabeza lectora/detectora fija de burbujas, hay que desplazar o propagar a las burbujas hasta ella, a través de unas pistas formadas por unos pequeños elementos de Permalloy con diferentes formas (gálones, semidiscos, etc.).

En la figura 9-46 se muestra la construcción de un dispositivo de MBM y su estructura interna. La película magnética donde se forman las burbujas queda aislada con óxido de silicio y sobre ella se depositan unas pistas que sirven para generar,

detectar y borrar burbujas. Las pistas mencionadas, que son atravesadas por corrientes eléctricas, quedan aisladas por una nueva capa de óxido de silicio, sobre la que se distribuyen los "motivos" de Permalloy, que constituyen los puntos de memoria que alojan a las burbujas y forman el camino por el que se mueven hasta los centros de lectura/escritura/borrado. El movimiento de las burbujas se consigue mediante un campo magnético rotatorio producido por dos bobinas perpendiculares que envuelven la pastilla. Externamente y por encima de las bobinas, se dispone un blindaje metálico de protección.

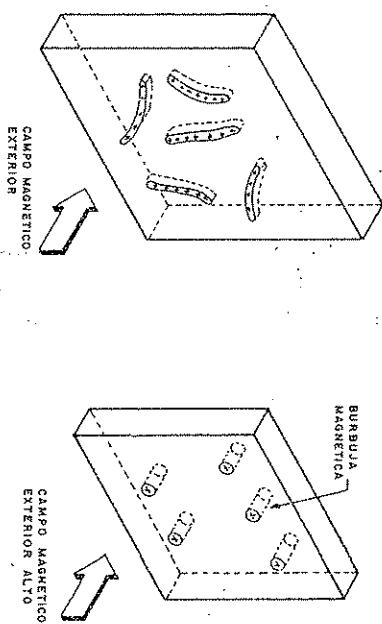


Fig. 9-45. Al aumentar el valor del campo magnético exterior perpendicular a la superficie, disminuye el tamaño de los dominios magnéticos de polaridad opuesta, hasta que se convierten en diminutos cilindros de muy pocas micras de diámetro.

En contraposición a las memorias RAM, que son de acceso aleatorio, las MBM son de acceso secuencial y es preciso desplazar, ordenadamente, las burbujas hasta el detector de las mismas. La forma más generalizada de organizar la arquitectura interna de una MBM, para obtener la máxima capacidad y velocidad, es la denominada *serie/paralelo/serie*. Dicha estructura, ofrecida en la figura 9-47, almacena la información, es decir, las burbujas, en las posiciones de memoria (motivos de Permalloy) que conforman una serie de bucles secundarios, interconectados todos ellos mediante "puertas de transferencia" con el bucle principal, sobre el que se encuentra el generador/detector/borrador con burbujas.

La manipulación de las burbujas, o sea, el desplazamiento de sus posiciones a los elementos fundamentales del bucle principal y su posterior regreso a las posiciones de partida, exige un completo controlador, que lleve la cuenta de todos los movimientos.

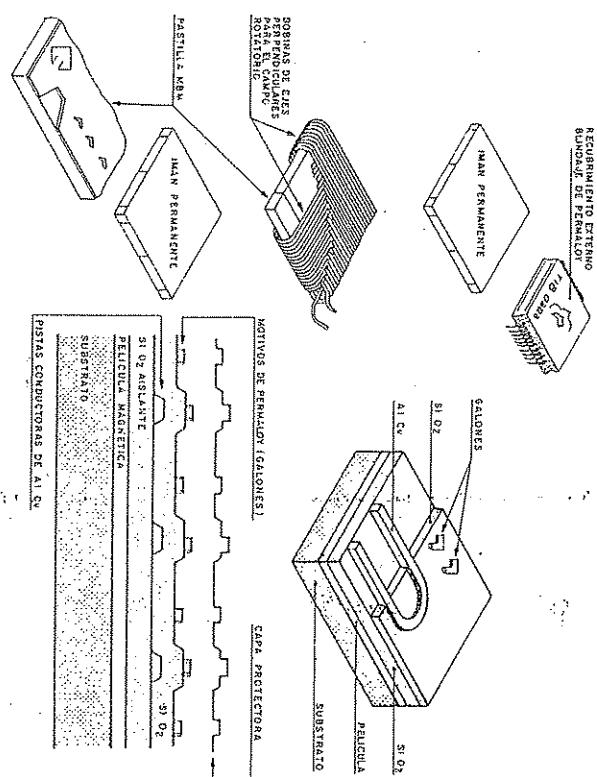


Fig. 9-46. Estructura interna y composición de un dispositivo MEM.

9.9. MEMORIAS ÓPTICAS

Estas memorias ofrecen un gran potencial de almacenamiento, pero no han tenido gran éxito comercial, al no haberse encontrado un método económico que permita escribir y borrar lo escrito. Su aplicación principal es sustituir a las memorias ROM.

Dos son las tecnologías a emplear. La primera consiste en grabar mediante un láser pequeñas marcas, que son leídas por el mismo láser, empleando una potencia menor. La segunda, consiste en generar un holograma de un conjunto de bits. La ventaja del segundo procedimiento es que es menos sensible al polvo y a maderos maciones del soporte.

El disco óptico almacena los bits en forma de huecos microscópicos producidos por el láser, que quema pequeños puntos de la superficie. La ausencia o presencia del hueco determina si se trata de un bit 1 o 0.

Los huecos tienen una dimensión de una micra, pudiéndose grabar hasta 5.000 huecos por pulgada. Como se ha dicho, desgraciadamente, los huecos no pueden borrarse, por lo que, una vez grabados los discos, sólo pueden actuar como dispositivos de lectura o de entrada de información.

Como el rayo láser accede muy rápidamente a cualquier zona del disco, este sistema añade, a una alta capacidad, un tiempo de acceso muy pequeño.

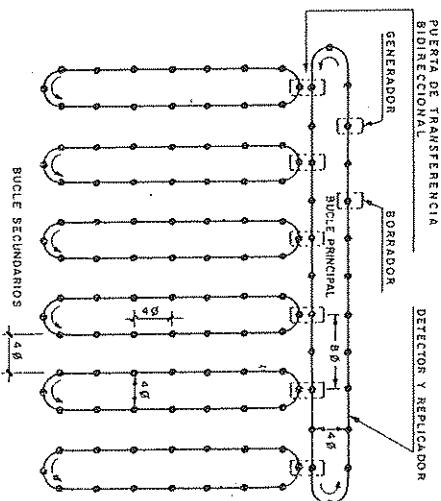


Fig. 9-47. Organización "serie-paralelo-serie" de una MBM.



EJERCICIOS

Ejercicio 9.1

A su juicio, ¿qué características hay que tener en cuenta en la elección de un periférico?

Ejercicio 9.2

Indicar los diferentes periféricos por sensor que se conozcan, comentando la principal característica de cada uno de ellos.

Ejercicio 9.3

Realizar un estudio comparativo de los tipos de impresoras existentes.

Ejercicio 9.4

Indicar los principales usos de la cinta magnética.

Ejercicio 9.5

Establecer posibles modos de verificación en cinta.

Ejercicio 9.6

¿Qué son los espacios inter-registros? ¿Para qué se emplean? ¿Cuándo se crean?

Ejercicio 9.13

¿Por qué se caracterizan los discos multicabeza?

¿Cuántas cabezas suelen existir en un disco?

¿Qué se entiende como cilindro en un disco?

Ejercicio 9.7

Determinar los posibles formatos de un registro en cinta, valiéndose de una representación gráfica para cada uno.

Ejercicio 9.14

¿Qué se entiende por densidad de grabación en un disco? ¿Existe la misma densidad de grabación en todas las pistas de un disco? ¿Y entre diferentes unidades de disco? ¿Por qué?

¿Qué es más normal el uso de volúmenes multifichero en cinta, o el uso de ficheros multivolumen?

Dada una unidad de disco cuya densidad de grabación es de 1.600 bits/cm y un radio de 15 cm, obtener la velocidad de transferencia VT, si la velocidad a la que gira el disco es de 2.400 r.p.m.



Ejercicio 9.9

Representar la señal que se emite en la escritura de la información 11010, usando los siguientes códigos de grabación:

- a) RZ (Return to zero).
- b) NRZ (no return to zero).
- c) PE (Phase encoding).

Ejercicio 9.10

Representar el literal "MOVE AX,BX" en una banda magnética, utilizando el código EBCDIC y el bit de verificación vertical para paridad par.

Ejercicio 9.11

¿Cuáles son, a su juicio, aquéllas características que se han de tener en cuenta en toda unidad de disco?

Ejercicio 9.12

¿Por qué se caracterizan los discos multicabeza?

¿Cuántas cabezas suelen existir en un disco?

¿Qué se entiende como cilindro en un disco?



Ejercicio 9.15

Se sabe que la unidad de disco 3330 de IBM dispone de las siguientes características:

- Un volumen consta de 404 cilindros.
- Un cilindro tiene 19 pistas (ó 19 cabezas).
- Una pista tiene 13030 octetos.

Se pide:

- a) Capacidad por cilindro.
- b) Capacidad por volumen.

Ejercicio 9.16

Se quiere introducir en un computador VAX 11/750 la información referente a las nóminas del personal de la Universidad. La plantilla activa consta de 1.000 empleados y se dispone de un terminal inteligente con dos unidades de diskette, con las siguientes características:

- 25.756 bytes por pista.
- 10 pistas por superficie (2 por diskette).
- 256 bytes por sector con la siguiente estructura:

NS	NR	LR	REGISTRO	NR	LR	REGISTRO
1 byte 2 bytes cada uno						

Se pide:

- a) Número de sectores utilizados en la grabación de la información.
- b) Número de pistas ocupadas.
- c) Número de superficies ocupadas y diskettes utilizados si el registro nómina consta de 87 bytes.

Ejercicio 9.17

Se desea almacenar 300.000 registros de 200 bytes y clave 10 octetos empleando

un disco 3330 (especificaciones indicadas en un ejercicio anterior) y sabiendo que el tamaño máximo del buffer es de 2K y que el factor de bloqueo es de 10.

Se pide:

- a) Número de bloques por pista.
- b) Número de pistas.
- c) Número de cilindros.

Nota: Cada bloque lleva una etiqueta de 11 bytes.

Descripción de la arquitectura de los computadores de la familia VAX

10.1. INTRODUCCIÓN

Como aplicación práctica de todos los conceptos estudiados hasta aquí, se ha elegido a la familia de minicomputadores "VAX-11" de Digital Equipment para su descripción detallada, por las siguientes razones:

1. Sus excelentes recursos y características técnicas.
2. Su frecuente utilización en los medios universitarios e industriales y su alto grado de introducción en todo el mundo.
3. La excelente documentación técnica que acompaña a las máquinas y la que hemos obtenido de sus fabricantes y distribuidores.
4. Su total compatibilidad con los modelos de las familias precedentes, como la PDP-11, y otras venideras.

Todos los procesadores de la familia VAX tienen una arquitectura interna de 32 bits, que se complementa con el sistema operativo VAX/VMS, que admite multiprogramación, multisusuario, trabajo en tiempo real y aplicaciones por lotes o en batch. Los computadores VAX ("Virtual Address Extension") trabajan con memoria virtual, como lo indica su propio nombre.

Los tres sistemas iniciales y básicos de esta familia son:

1. VAX 11/730: Es el modelo más barato, pero que puede conectarse a todo un conjunto de periféricos mediante un adaptador UNIBUS.
2. VAX 11/750: Es el modelo medio, construido con tecnología bipolar LSI. En este capítulo se hace referencia expresa a esta configuración.
3. VAX 11/780: Es el más potente y está destinado al manejo de grandes bases de datos y estrictas exigencias en los procesos.

408 / Descripción de la arquitectura

Como se indica al final de este capítulo, la familia de computadores VAX se ha ampliado recientemente con nuevos y más potentes modelos..

10.2. ARQUITECTURA BÁSICA DEL VAX 11/750

En la figura 10-1 se muestra la arquitectura fundamental de este minicomputador, compuesto por tres subsistemas interrelacionados, que son:

- UCP
- Memoria principal
- Subsistema E/S

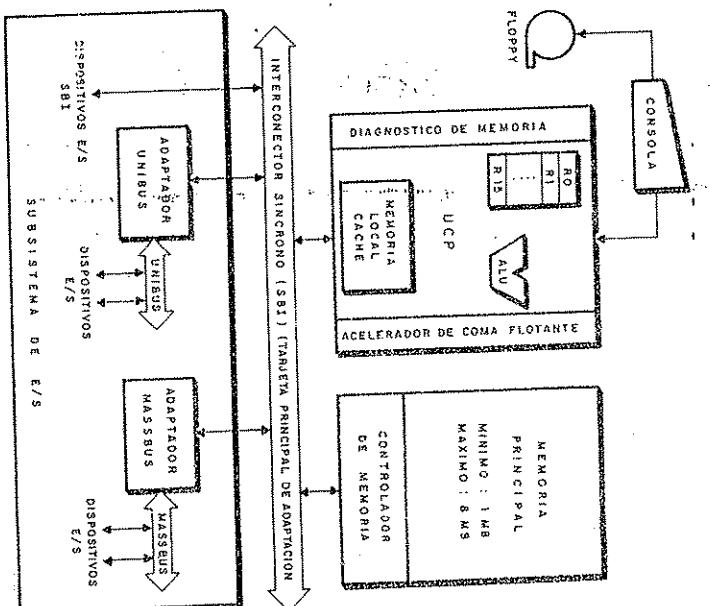
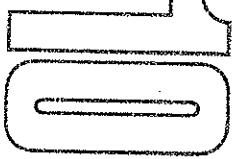


Fig. 10-1. Estructura general del minicomputador VAX 11/750 formada por tres subsistemas: UCP, memoria principal y Entradas/Salidas.

Los tres subsistemas mostrados en la figura 10-1 se comunican a través de un interconector síncrono SBI. También se pueden conectar al sistema diversos periféricos mediante su adaptación al bus UNIBUS o al MASSBUS.

La UCP está constituida por los siguientes elementos:

- 16 registros generales (R0, R1, ..., R15) de 32 bits cada uno, entre los que se encuentra el Contador de Programa.
- La ALU, que puede estar apoyada por un "Acelerador de coma Flotante", opcional.
- Una memoria cache local y una Unidad de Diagnóstico de Errores, opcional.

10.3. EL PROCESADOR CENTRAL

Las características principales del procesador central del VAX 11/750 son:

1. Procesador microprogrammado, que trabaja con palabras de 32 bits.
2. Circuitos de control para manejo de memoria, que alcanza un direccionamiento de hasta mil millones de bytes de memoria virtual, empleando una memoria principal física mucho más pequeña.
3. Tecnología de matriz de puertas (gate array), que permite la reducción del espacio físico que ocupa la circuitería y aumenta la fiabilidad.
4. Memoria "cache" de 4K bytes, que mejora notablemente el tiempo de acceso a la memoria principal.

5. Modo compatible con la familia PDP-11, que facilita al usuario de este sistema la familiarización con el sistema operativo VAX/VMS.

6. Una Memoria de Control del Usuario, opcional, que permite crear rutinas para aplicaciones específicas del usuario, en microcódigos. No existe para el modo VAX 11/750.

7. Acelerador de coma Flotante opcional, que realiza las funciones en las que se requiere coma flotante y algunas con datos de tipo entero, con mucha mayor rapidez que la ALU.

La UCP del VAX 11/750 está construida con tecnología avanzada gate array y sus cuartas partes de sus circuitos integrados son de alta escala de integración (LSI).

La UCP básica está construida con 55 pastillas de circuitos integrados LSI, cada una de los cuales consta de un promedio de 488 puertas lógicas. Con esta tecnología, el espacio y el consumo se reducen a la mitad si se comparan con los correspondientes a los de la tecnología convencional.

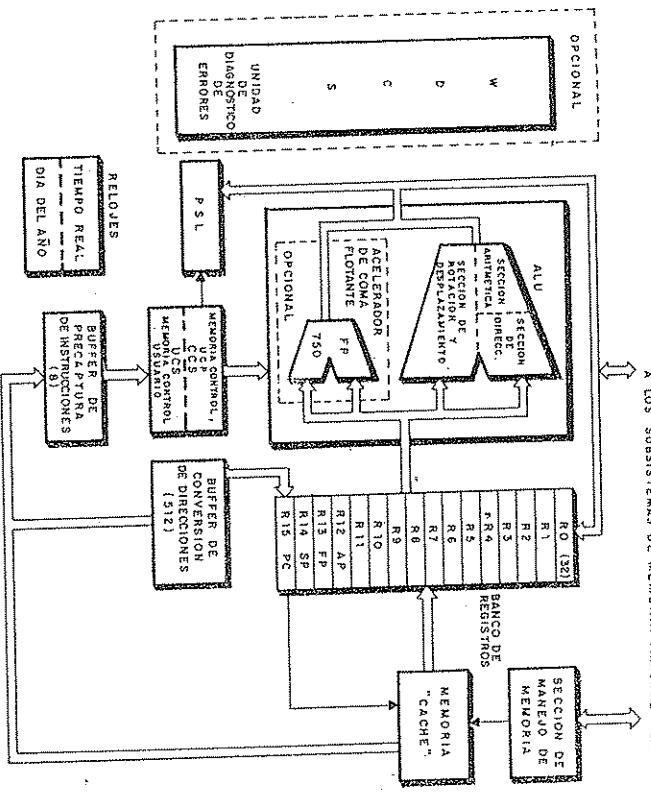


Fig. 10-2. Organización interna, simplificada, de la UCP del VAX 11/750.

El bus de direcciones de la UCP está formado por 32 líneas, lo que supone dirigir hasta 4 Gbytes (2^{32}) de memoria virtual. El hardware de la unidad de manejo de memoria convierte las direcciones virtuales en direcciones físicas.

El VAX 11/750 puede trabajar en 4 modos diferentes, cada uno con distintos privilegios.

MODO 0: KERNEL

Usado por el núcleo del sistema operativo para el manejo de páginas y de E/S. Es el de máxima prioridad.

MODO 1: EJECUTIVO

Usado por la mayoría de las llamadas al sistema operativo, también se encarga del tratamiento de los registros.

MODO 2: SUPERVISOR

Para servicios tales como la interpretación de comandos.

MODO 3: USUARIO

Se emplea para trabajar a nivel de códigos de utilidades, compiladores, depuradores, etc. Es el de menor privilegio.

En la figura 10-2 se muestra un esquema simplificado de la estructura principal de la UCP del VAX 11/750.

A continuación se describen los diversos bloques que conforman el esquema de la figura 10-2.

10.3.1. Unidad operativa

La ALU de la UCP consta de tres secciones paralelas para el procesamiento de datos y direcciones:

1. *Sección aritmética*: Ejecuta las operaciones aritméticas y lógicas sobre el conjunto de datos y direcciones.

2. *Sección de rotación y desplazamiento*: También se encarga de empaquetar y desempaquetar datos en coma flotante y cadenas de números decimales.

3. *Sección de direcciones*: Calcula las direcciones virtuales que envía al buffer de traducción de dirección.

El acelerador de coma flotante (FP 750) es opcional y con él se pueden ejecutar sumas, restas, productos y divisiones con operandos en coma flotante de simple precisión (32 bits) y de doble precisión (64 bits, 56 para la mantisa y otros 8 para el exponente). Realiza la Multiplicación Extendida (EMOD) y la Evaluación Polinómica (POLY) y convierte datos de enteros a coma flotante y de simple a doble precisión y viceversa.

10.3.2. Memoria "cache"

Es una pequeña memoria de 4K bytes de alta velocidad, que guarda una copia de aquellas partes de la memoria que, posiblemente, van a ser necesarias en cada fase del procesamiento. Así se consiguen accesos más rápidos a datos e instrucciones.

La sección de manejo de memoria se encarga de mantener actualizado el contenido de la cache, de manera que los datos que se van a necesitar se encuentren en ella. El procesador siempre comienza buscando las informaciones en la memoria cache, y, si no los encuentra, acude a la memoria principal.

Con la inclusión de esta memoria se reduce el tiempo de acceso a memoria aproximadamente a la mitad.

10.3.3. Buffer de conversión de direcciones

Está constituido por una memoria cache de conversión a direcciones físicas que se usan frecuentemente con objeto de disminuir el tiempo empleado por la UCP en traducir direcciones virtuales a reales. Esta memoria sólo consta de 256 conversiones de direcciones de páginas virtuales a reales para el espacio del sistema y otras 256 para el espacio del proceso. Cada entrada utiliza código de paridad para el control de los datos.

10.3.4. Relojes

El procesador contiene un reloj programable de tiempo real y un reloj que señala el día y la hora Smithsonians. El primero está basado en un oscilador de cristal con una exactitud de 0,01 % y una resolución de 1 microsegundo. El reloj del día del año es utilizado por el software y está equipado con una batería de automantenimiento para no ser afectado, en caso de producirse una caída del sistema.

10.3.5. Buffer de prebúsqueda de instrucciones

Se encarga de precapturar información de la secuencia de instrucciones.Consta de 8 bytes y la lógica de control se encarga de recoger datos de doble palabra de longitud (32 bits) para mantener completo el buffer.

10.3.6. Unidad de control de diagnóstico (WDCS)

Se trata de un equipo físico opcional dedicado a soportar los programas especiales para el diagnóstico y la detección de errores del sistema.

10.3.7. Registros generales

La UCP posee un banco de 16 registros de 32 bits cada uno, que se denominan R0, R1, ..., R15 y cuyas misiones se comentan a continuación.

R0-R5. Son registros de propósito general que pueden ser utilizados libremente por los programas. También pueden ser utilizados por ciertas instrucciones, cuya

ejecución puede interrumpirse. En este caso, son cargados por datos significativos que se manejan en la ejecución de dichas instrucciones.

R6-R7: Son registros de propósito general, que pueden ser utilizados libremente.

R12: Recibe el nombre de *puntero de Argumento* (AP) e indica la dirección de comienzo de una lista de argumentos que se van a utilizar después de la llamada a un procedimiento. Actúa conjuntamente con el registro FP.

R13: Se llama *Puntero de Región* (FP) y se emplea en las llamadas a procedimientos.

R14: Es el *Puntero de la Pila o Stack*. Señala la última dirección utilizada por la pila.

R15: Es el *Contador de Programa* (PC) y contiene la dirección de la siguiente instrucción o dato del programa en ejecución.

10.3.8. Doble Palabra de Estado del Procesador (PSL)

La UCP dispone de un registro de 32 bits, que informa del estado en que se encuentra. Como se refleja en la figura 10-3, de los 32 bits del PSL, los 16 de menos peso forman la *palabra de estado del procesador* (PSW), a la que puede acceder cualquier programa. Los 16 bits de más peso tienen un carácter *privilegiado*.

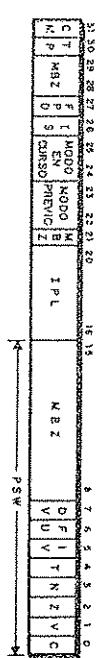


Fig. 10-3. Organización de la doble palabra de estado del procesador PSL.

A continuación se describe la función de cada campo de bits de PSL.

Bits 0-3: Código de condición (acarreo, sobrepasamiento, cero y signo).

Bits 4-7: Son los bits destinados a la habilitación de excepciones o traps.

Bit DV: Cuando está activo (1) permite que, si después de una instrucción se produce un resultado decimal cuyo valor absoluto no es representable en su destino, se produzca una excepción. Si DV = 0, no se origina la excepción y el resultado se trunca por los dígitos más significativos.

Bit FV: Cuando FV = 1, se produce una excepción después de la ejecución de una instrucción que produzca un resultado en coma flotante, demasiado pequeño para ser representado. Si FV = 0, no hay excepción y se guarda un cero como resultado.

Bit IV: Si IV = 1, se produce una excepción después de una instrucción, cuyo resultado entero no sea representable en el espacio destinado para ello.

Bit T: Si T = 1, al ejecutar una instrucción se genera una excepción de trazado. Se utiliza generalmente para la depuración de los programas.

Bits 8-15: No se utilizan. Reservados.

Bits 16-20: Forman el campo IPL, que contiene el nivel de prioridad mínimo que debe ser superado por una petición de interrupción para que sea aceptada. Hay 31 niveles de prioridad, desde el 01 al 1F. Todo el software se ejecuta a un nivel del IPL igual a cero, por lo que cualquier petición de interrupción es reconocida y aceptada por la UCP. Los niveles desde el 01 al 0F se emplean para interrupciones generadas por software. Los niveles comprendidos entre 10 y 1F, se dedican a situaciones urgentes, errores graves, fallos de energía, interrupciones de los periféricos, etc.

Bit 21: Debe ser cero.

Bit 22-23: Es el campo de Modo Previo, que indica el valor del nivel de la última excepción ocurrida del modo en curso.

Bit 24-25: Este campo expresa el modo en curso (Kernel, Ejecutivo, Supervisor o Usuario).

Bit 26: Este bit, denominado IS (Pila de Interrupciones), si está activo indica que el procesador está haciendo uso de la pila de Interrupciones.

Bit 27: Este bit, llamado FPD (Primera Parte Hecha) si se encuentra a 1 a la vuelta de una excepción, significa que la instrucción interrumpida se quedó a medio hacer, por lo que no se ejecutará desde el principio.

Bit 30: Es el señalizador de Traza Pendiente (TP) y lo usa la UCP para asegurarse de que sólo se ejecuta una excepción de trazado ($T = 1$).

Bit 31: Este bit, llamado abreviadamente CM, indica el Modo de Compatibilidad. Si CM = 1, la UCP está en modo compatible con el PDP-11 y puede ejecutar las instrucciones de ese procesador.

10.3.9. Memoria de Control del Usuario

Esta memoria, opcional, tiene una capacidad de 1 K palabras de 80 bits cada una, o sea, 10 K bytes. Grabando en ella los microcódigos convenientes, el usuario puede disponer de instrucciones específicas.

Esta memoria, llamada abreviadamente UCS, es como una ampliación de la Memoria de Control de la UCP (CCS), donde radican los microcódigos de las instrucciones del repertorio general.

10.4. ESTRUCTURA DE LOS PROCESOS

se utilizan los *Bloques de Control de Proceso PCB*. Cuando un programa es interrumpido, se guarda en memoria un PCB con una serie de datos del proceso para que, cuando vuelva a ser llamado, continúe en el mismo punto en que se interrumpió. Las informaciones que guarda un PCB se muestran en la tabla de la figura 11-1.

Señalizaremos en la parte posterior del PCB una pista especial, rCBB (Base del PCB), que contiene la dirección del sij.

PCB: Se hace una breve descripción de los registros que componen el

PSL, es 0 y el bit señalizador de la Pila de Interrupción está a 0.

SSP: Puntero de la pila utilizado con el modo de acceso 2.

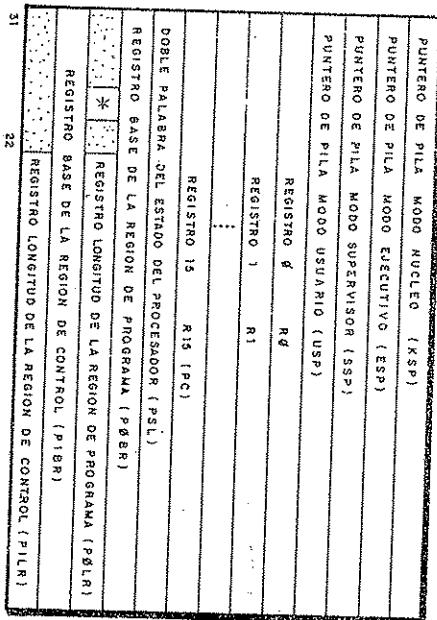


Fig. 10-5. Cada proceso tiene su pila de cada tipo, excepto la Pile a Interrupción que es común para todos los procesos.

Los elementos que faltan por describir y que están contenidos en el PCB son:

RO-R17: Registros generales de la UCP

873: Puntero de Región

R15: Contador de Prog

PSL: Doble Palabra de Estado del Procesador

POBR: Es el registro base para la tabla de p

nes virtuales (de 0 a 255 = 1).

POLAR: Es el registro definitivo para la lab
POBR:

ASTLVL: Excepción o trap asincróna del sistema que aún está pendiente. In-

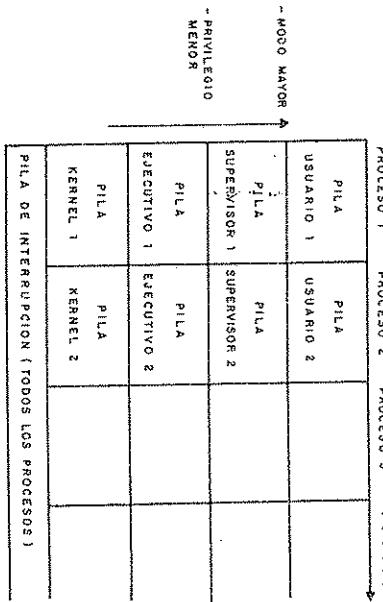
dica el modo de acceso más privilegiado para el que está pendiente una excepción al sistema (AST).

۱۰۷

416 / Descripción de la arquitectura

... "Estructura de las informaciones que contiene un PCB (Bloque de Control de Proceso).

descripción de la arquitectura



Existen 5 tipos de pilas, guardándose el puntero de cada una en un registro interno. Cuando se utiliza algún registro que no está en el SP, se guarda el que estaba en un registro interno y se saca el puntero que se deseaba de otro registro interno, poniéndolo en el SP. Además, cada proceso tiene su pila de cada tipo, excepto uno, que es común para todos los procesos y que se trata de la pila de interrupción. Véase la figura 10-5.

	SIGNIFICADO
0	AST pendiente para el modo de acceso 0 (Kernel)
1	AST pendiente para el modo de acceso 1 (Ejecución)
2	AST pendiente para el modo de acceso 2 (Supervisor)
3	AST pendiente para el modo de acceso 3 (Usuario)
4	No hay AST pendiente
5	Reservado a Digital Equipment

P1BR: Registro base para la tabla de páginas dentro del rango de direcciones virtuales (de 2^{30} a $2^{31}-1$).

P1LR: Registro de longitud de la tabla de páginas, referido a P1BR.

10.4.1. Excepciones asíncronas del sistema (AST)

Cuando ocurre una situación que provoca una excepción hay que saltar a su rutina correspondiente de tratamiento. Esta excepción tendrá asignada por software, un modo de acceso que se indica en el campo ASTVL del PCB.

Si el modo de acceso en curso es mayor que el asignado a la excepción, el proceso en curso continúa, ya que no puede interrumpirse un proceso por otro con modo de acceso menos privilegiado. En este caso, hay que esperar a la ejecución de una instrucción REI (retorno de Interrupción). Si REI detecta un modo de acceso nuevo, igual o mayor que el pendiente, se produce una interrupción IPL2 para el tratamiento de AST.

10.5. LA MEMORIA PRINCIPAL

Las principales características del subsistema de memoria principal de la familia VAX-11, son:

- a) *Memoria expansiónable*. Permite añadir tarjetas de 1 M byte, hasta alcanzar un máximo de 16 M byte, en el caso del VAX 11/750.
- b) *Controlador para la corrección de errores de memoria*. Corrige todos los errores de un solo bit y detecta los errores producidos por parejas de bits en la memoria del sistema.
- c) *Memorias ROM para el control de periféricos*. Sirven para controlar todos los periféricos, incluido el controlador de cinta TU58.

4.18 / Descripción de la arquitectura

d)

Batería de seguridad opcional. Mantiene la alimentación para salvar los datos durante un mínimo de 10 minutos de una memoria de hasta 8 M byte.

El módulo de control de la memoria (controlador) contiene toda la lógica necesaria para conectar las tarjetas ampliables del sistema. Estas tarjetas están construidas con pastillas de memoria MOS de 64 K bits de capacidad. La capacidad mínima del sistema es de 1 M byte pudiendo alcanzar los 8 ó 16 byte en incrementos de 1 M byte. Figura 10-6.

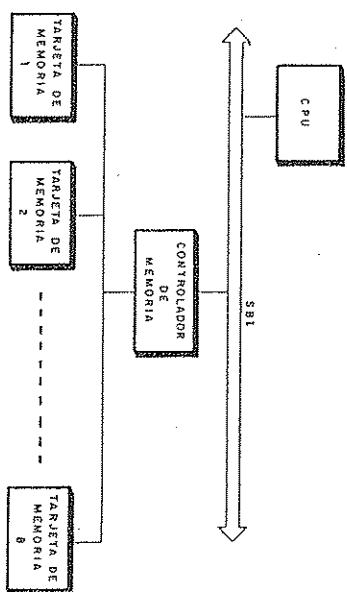


Fig. 10-6. Configuración del subsistema de memoria principal.

El controlador de memoria dispone de la lógica de direccionamiento, la de corrección de errores y la circuitería de refresco. También contiene tres registros de doble palabra, accesibles por el espacio de E/S, que sirven para determinar el tamaño de memoria, grabar posiciones defectuosas, asistir un fallo en una pastilla de la memoria y verificar que la lógica de corrección de errores trabaja correctamente.

10.5.1. Memorias ROM para el control de los periféricos

El controlador de memoria contiene cuatro pastillas ROM para permitir el gobierno de los periféricos. Cada memoria ROM contiene hasta 256 bytes de código, que son accesibles en el espacio de E/S, en las siguientes direcciones físicas:

ROM	DIRECCIONES FÍSICAS (HEX)
1	F20400 - F204FC
2	F20500 - F205FC
3	F20600 - F206FC
4	F20700 - F207FC

Descripción de la arquitectura / 419

La UCP lee los datos desde esas ROM y los ejecuta en el tiempo de proceso. Además, cada ROM incluye un espacio para comprobar su contenido.

10.6. OPERACIONES BASICAS DE LA MEMORIA PRINCIPAL

El espacio de direcciones físicas regulado por el controlador de memoria está dividido en dos zonas: la de las direcciones de la memoria física y la de direcciones de E/S. Figura 10-7.

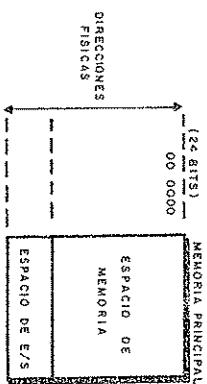


Fig. 10-7. Distribución de la memoria física en dos zonas.

Para acceder al espacio de la memoria principal existen tres tipos de ciclos de memoria, que se pasan a describir.

10.6.1. Lectura

Un ciclo de lectura de memoria, en el que el Controlador-Corrector de Errores (ECC) no haya detectado errores, consta de 4 etapas:

1. El dispositivo que solicita los datos de la memoria, proporciona la dirección y la información de control al controlador de la memoria.
 2. El controlador de la memoria inicia el ciclo seleccionando la tarjeta correspondiente.
 3. La tarjeta escogida presenta el dato a la lógica ECC.
 4. El controlador entrega el dato al peticonario, notificándole que tiene disponible la información.
- Si, por cualquier circunstancia, se produce un error capaz de ser corregido, esta acción la realiza el ECC, presentando el dato correcto al dispositivo y generando una interrupción para avisar al sistema del error producido. Si la orden de lectura, a partir de un adaptador de E/S, este recogerá y almacenará el mensaje de error. Si la orden de lectura salió de la UCP, entonces se originará un check abort.

10.6.2. Escritura de doble palabra

La escritura de una doble palabra se realiza en tres etapas:

1. El dispositivo requerido presenta la dirección y la información de control, al controlador de la memoria.
2. El controlador de la memoria inicia el ciclo de memoria, seleccionando la tarjeta correspondiente.
3. Mientras la tarjeta inicia su ciclo, el controlador de memoria genera 7 bits de comprobación, presentandolos, junto a la doble palabra de 32 bits, a las tarjetas de memoria. Así se completa el ciclo.

10.6.3. Escritura de byte y palabra

Para estos tipos de ciclos de escritura, la secuencia es la misma que en las dos primeras etapas del ciclo de lectura, pero se forma una nueva palabra de 32 bits que contiene los 7 bits de comprobación.

Si el ECC detecta un error corregible, él se encarga de modificarlo. Si el error se produce en un byte o en una palabra que se están escribiendo, se ignora. Si el ECC detecta un error no corregible durante la parte de lectura del ciclo, la doble palabra original será reescrita en la memoria, sin variación alguna. Esto puede originar un error redundante durante una orden de lectura posterior sobre esta posición.

10.7. REGISTROS DE CONTROL Y ESTADO

Las máquinas de la familia VAX-11 disponen de 3 registros de control y estado; el CSRO, situado en la dirección F20000, que proporciona la información necesaria para permitir la grabación de los errores corregibles y no corregibles; el CSR1,

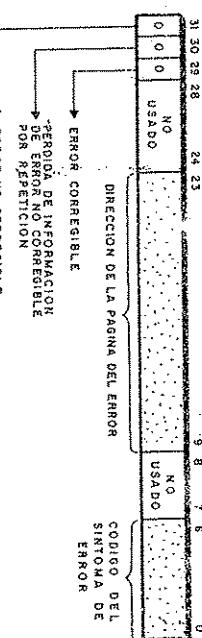


Fig. 10-8. Función de los diversos campos de que consta el registro CSRO, encargado del tratamiento de los errores corregibles y no corregibles.

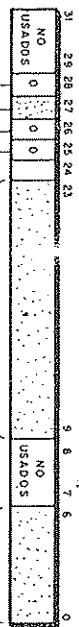


Fig. 10-9. Función de los bits del registro CSR1, dedicado a labores de verificación y diagnóstico del sistema de corrección.

ubicado en la dirección F20004₁₆, que colabora en la verificación y diagnósticos del hardware de corrección y, por último, el CSR2, que se halla en la dirección F20008₁₆ y que proporciona el tamaño de la memoria y la información adecuada sobre la configuración de la misma.

En la figura 10-8 se muestra la función de los diversos campos de que se compone el registro CSR0. La figura 10-9 realiza el mismo cometido para el registro CSR1 y la figura 10-10 está destinada al registro CSR2.

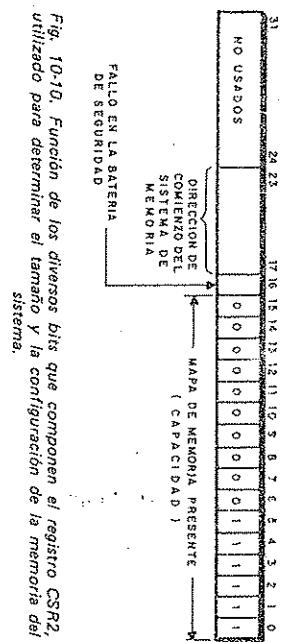


Fig. 10-10. Función de los diversos bits que componen el registro CSR2, utilizado para determinar el tamaño y la configuración de la memoria del sistema.

10.8. ESPACIO DE LA MEMORIA:

El espacio direccional de los minicomputadores VAX-11, es de 2³² bytes. La mitad de este espacio (la que está caracterizada por tener el bit de más peso de la dirección a 1), es el espacio del sistema. En esta zona se ubica el sistema operativo y los datos del sistema.

4.22./ Descripción de la arquitectura

La otra mitad es el verdadero espacio de memoria, que, a su vez, se divide en dos partes, la de las direcciones más bajas, llamada ESPACIO P0, que es donde residen los programas y la mayoría de sus datos. En la otra parte, denominada ESPACIO P1, como se muestra en la figura 10-11, se encuentra la pila y los datos de procesos específicos. Cada proceso ocupa una parte del ESPACIO 0 y otra del ESPACIO 1.

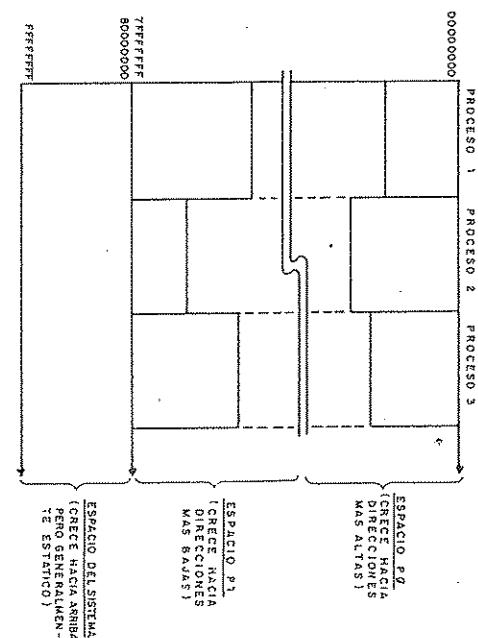


Fig. 10-11. Distribución del espacio de memoria del VAX-11.

La figura 10-12 muestra la distribución de los 32 bits de la dirección virtual de la memoria, que se divide en páginas de 512 bytes.

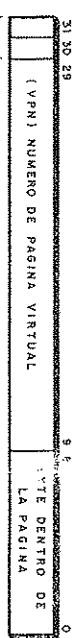


Fig. 10-12. Distribución de los 32 bits de que se compone una dirección virtual del VAX-11.

Descripción de la arquitectura /423

Con el número de la página virtual (VPN) ya se puede acceder a la Tabla de Páginas (PT), que contiene las Entradas de PT (PTE). Las PTE constan de 32 bits, que soportan la información mostrada en la figura 10-13.

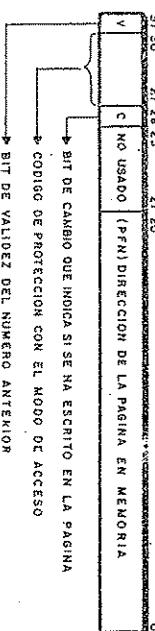


Fig. 10-13. Asignación de funciones de los 32 bits que componen cada Entrada de la Tabla de Páginas (PTE).

10.8.1. Espacio de direccionamiento virtual

Los más de 4 Gigabytes que pueden direccionar los componentes de la familia VAX-11, se distribuyen de acuerdo con el esquema de la figura 10-14.

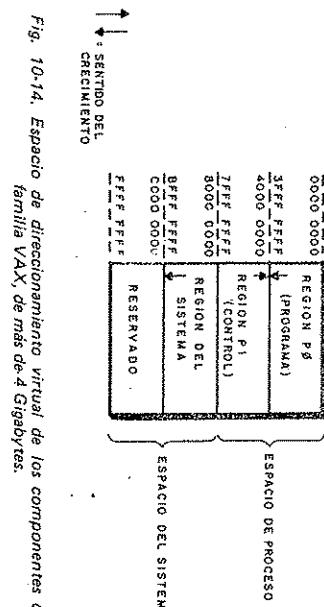


Fig. 10-14. Espacio de direccionamiento virtual de los componentes de la familia VAX, de más de 4 Gigabytes.

Teniendo en cuenta que un programa puede generar cualquier dirección, se pude evitar que modifique e, incluso, que acceda a partes del espacio de memoria. También puede prevenirse el acceso a zonas del espacio de proceso.

Cada página tiene asociado un código de protección, que especifica, para cada tipo de acceso a memoria, si se le permite leer y/o escribir o no. Además, cada dirección se comprueba para asegurar que está dentro de los espacios P0 a P1.

4.2.4 / Descripción de la arquitectura

10.8.2. Código de protección

Cada página tiene en su correspondiente Entrada en la Tabla de Páginas (PTE) un código de protección, según el cual se puede leer, escribir ambas cosas a la vez y no permitir el acceso. Si un modo tiene acceso a la lectura, los modos más privilegiados también lo tendrán; lo mismo ocurre con la escritura.

A continuación se indican los códigos de protección y las posibilidades que existen en cada uno de los 4 modos.

Código	K	E	S	U
0000	-	-	-	(ningún acceso)
0001	Código reservado			
0010	RW	-	-	
0011	R	-	-	
0100	RW	RW	RW	RW (todos los accesos)
0101	RW	RW	-	-
0110	RW	R	-	-
0111	R	R	-	-
1000	RW	RW	RW	-
1001	RW	RW	R	-
1010	RW	R	R	-
1011	R	R	R	-
1100	RW	RW	RW	R
1101	RW	RW	R	R
1110	RW	R	R	R
1111	R	R	R	R

K = Kernel
L = Ejecutivo
S = Supervisor
U = Usuario
R = Permiso de lectura
W = Permiso de escritura

10.9. LA TRADUCCIÓN DE LAS DIRECCIONES

Cuando el bit de *habilitación del mapa de memoria* (MME) está a cero, los bits de menos significado de la dirección virtual son directamente los que expresan la dirección física de la memoria y no existe tipo alguno de protección de página. Sin embargo, si MME = 1, hay que realizar la traducción de la dirección virtual a la dirección física o real. En la figura 10-15 se presenta el cometido de los diversos bits que componen la PTE para esta situación.

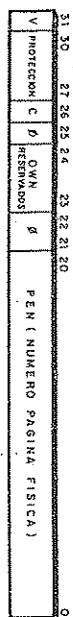


Fig. 10-15. Cometido de los diversos campos de bits que componen una PTE, cuando el bit MME = 1.

Aclaración BIT MME del VAX II:750

El bit MME se encuentra en un registro adicional y privilegiado, denominado MAPEN (Habilitación del mapa de memoria), y es uno de los 3 registros que dispone el hardware dedicado al manejo de la memoria.

Si MME está a "0", los bits menos significativos de la dirección virtual son directamente los que expresan la dirección física de la página y no hay ningún tipo de protección de página. Mientras que si MME = 1, hay que realizar la traducción de dirección virtual a física.

En el caso de que MME esté a 1, si en la PTE el bit de validez V = 1, entonces el campo PFN corresponde a los bits de más peso de la dirección física de la base de la página.

PTE: Entrada Tabla de Páginas.

En la figura 10-15, los bits 23 y 24 (OWN), están reservados y el sistema operativo VAX/MS los emplea como modo de acceso del propietario de la página. El campo PFN corresponde con los 21 bits de más peso de la dirección física de la base de la página. Son utilizados por el hardware sólo en el caso de que V = 1 (bit válido).

4.2.5 / Descripción de la arquitectura

10.9.1. Traducción de direcciones del espacio del sistema

Una dirección virtual hace referencia al espacio del sistema, cuando sus bits 31 y 30 tienen la configuración 10. El espacio de direcciones virtuales del sistema está definido por la *Tabla de Páginas del Sistema* (SPT), que es un vector de entrada a la Tabla de Páginas. La dirección física base de la SPT está contenida en el *Registro Base del Sistema* (SBR). La PTE direccionada por el SBR contiene el Número de Página Física (PFN) de la primera página del espacio del sistema (la que comienza en la dirección 80000000₁₆).

El mecanismo de traducción de una dirección virtual a real se indica en la figura 10-16.

El procedimiento empleado en la traducción mostrada en la figura 10-16 comienza multiplicando por 4 el VPN (Número de página virtual), para que, al sumarlo al contenido del SBR, se obtenga la dirección de la PTE que está compuesta por 4 bytes.

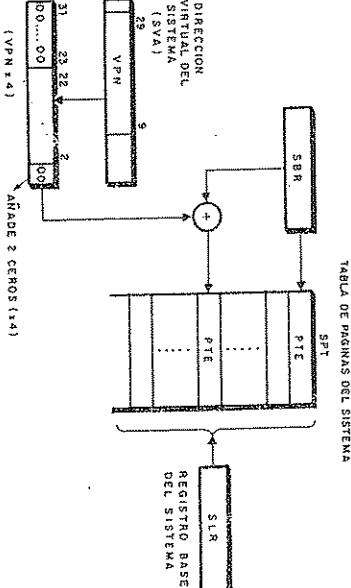


Fig. 10-16. Mecanismo de traducción de una dirección virtual a real en el espacio del sistema.

10.9.2. Traducción de direcciones del espacio del proceso

Cuando el bit de más peso (31) de la dirección virtual está a 0, se hace referencia al espacio del proceso. Además, si el bit 30 es cero, el espacio direccional es 0, mientras que, si el bit 30 es 1, se dirige a la dirección del espacio P₁.

La Tabla de Páginas de este espacio no tiene una dirección física, sino que se accede a ella con una dirección virtual, dentro del espacio del sistema. Por lo tanto, la dirección de la PTE es también una dirección virtual en el espacio del sistema.

Direccionalamiento del espacio P0

La Tabla de Páginas del espacio P0, se denomina POPT, y está definida por el Registro base POBR y el Registro de Longitud POLR.

La lógica empleada en la traducción de direcciones es similar a la usada en el espacio del sistema y se representa gráficamente en la figura 10-17.

Su Tabla de Páginas asociada se denomina P1PT, que queda definida por el Registro Base P1BR y el Registro de Longitud P1LR.

El esquema gráfico de traducción de direcciones responde al que se muestra en la figura 10-17, sustituyendo POBR, POLR y POPT por P1BR, P1LR y P1PT, respectivamente.

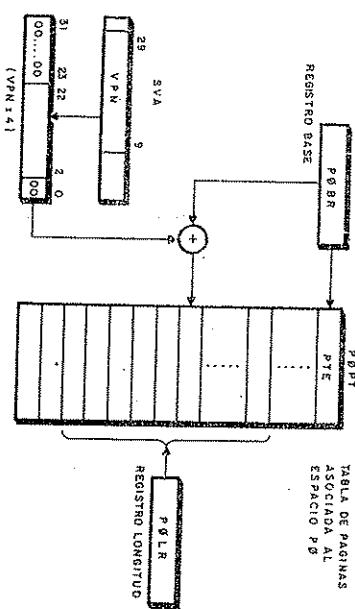


Fig. 10-17. Mecanismo de traducción aplicado al espacio P0.

10.10. SUBSISTEMA DE ENTRADAS Y SALIDAS.

EL BUS "UNIBUS"

Como se indicó al principio de este capítulo, el subsistema de E/S de la familia VAX-11 se encarga de la conexión con los periféricos del mundo exterior y esto lo

consigue mediante el bit interno SBI y, sobre todo, a través de los buses normalizados UNIBUS y MASSBUS.

Las principales características del UNIBUS son:

- Transferencia de datos con acceso directo a memoria (DMA).
- Comunicación entre los dispositivos conectados al UNIBUS. Permite la transferencia directa de datos entre los dispositivos colgados del UNIBUS sin la intervención de la UCP.
- Total compatibilidad con el formato UNIBUS del PDP-11.
- Interrupciones vectorizadas por hardware para el procesamiento de rutinas.
- Direccionalamiento de los registros de los dispositivos conectados al UNIBUS como direcciones en el espacio de memoria.
- Capacidad para soportar una amplia gama de periféricos.
- Possibilidad de conectar un segundo adaptador de UNIBUS.

El UNIBUS soporta todos los dispositivos, con excepción de los que operan a elevada velocidad tales como controladores de disco y cinta.

El bus UNIBUS está compuesto por 56 líneas a las que se conectan los periféricos. La figura 10-18 muestra la configuración de dichas líneas, de las que 51 funcionan en paralelo y 5 en serie. Por otra parte, 42 líneas son bidireccionales y 14 unidireccionales.

La comunicación entre dos dispositivos conectados al bus se establece mediante una relación maestro-esclavo. El maestro goberna al bus durante la comunicación con el otro dispositivo que actúa como esclavo. Por ejemplo, si la UCP actúa como maestro, puede capturar datos de un periférico que actúe como esclavo; en caso de que un controlador de disco actúe como maestro, se pueden transferir datos a la memoria, que en este caso soportaría la función de esclavo.

Cuando varios dispositivos solicitan el control del bus al mismo tiempo, los circuitos de prioridad se encargan de decidir al elemento que toma el cargo de maestro, quedando los restantes en una cota de espera. La prioridad de cada dispositivo se fija en la instalación del sistema. Si hay más de un dispositivo con el nivel máximo de prioridad, se atiende la petición simultánea cediendo el control al más cercano eléctricamente al procesador.

Petición de bus

Cuando se toma el control del bus, se puede operar en dos modos de trabajo:

- Petición sin procesador (NPRI): Este tipo de petición se usa para realizar transferencias de datos sin la participación de la UCP. El trabajo del DMA puede ser paralelo con el de ejecución de instrucciones por parte de la UCP..

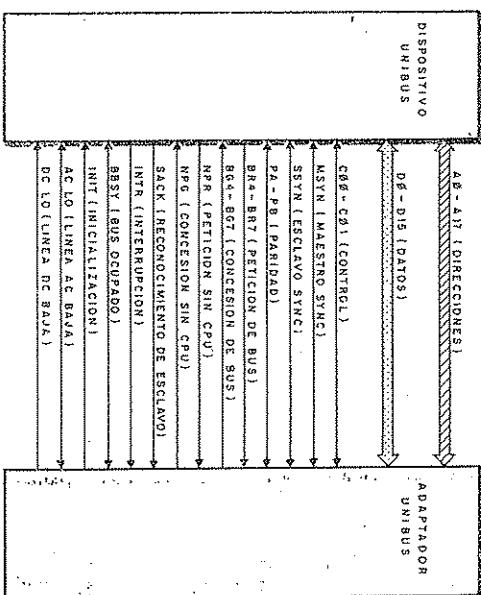


Fig. 10-18. Esquema de la configuración de las líneas del bus UNIBUS.

2. Petición con procesador (BR):

El dispositivo necesita interrumpir a la UCP para ser atendido.

Interrupciones

No se utiliza la exploración o polling para tratar las peticiones, por lo que el dispositivo que necesita ser atendido provocará una interrupción suministrando un vector que direcciará una posición de la memoria, en la que se halla la dirección de comienzo de la rutina de servicio para dicho dispositivo.

10.10.1. Adaptador de UNIBUS

Hay tres características de la arquitectura del VAX-11/750 que impiden la conexión directa del UNIBUS con el sistema.

- 1) Las peticiones tipo NRP producen direcciones virtuales secuenciales que no tienen por qué estar situadas consecutivamente en el espacio físico al que pertenecen las diferentes páginas. El adaptador realiza la función de indicar la página física a la que corresponde cada dirección.

2) Los periféricos no pueden proporcionar una dirección impar de memoria para depositar sus datos. El adaptador, cuando es necesario, desplaza una unidad de dirección por del dispositivo para obtener una dirección impar.

- 3) Debido a que los dispositivos conectados al UNIBUS sólo pueden direccionar un máximo de 256K bytes, el adaptador UNIBUS se encarga de la expansión de esta capacidad, con lo que se puede alcanzar una totalidad de 16 M bytes de direcciones físicas, para el caso del VAX-11/750.

10.10.1.1. El direccionamiento en el UNIBUS

El UNIBUS ocupa las direcciones físicas más significativas del sistema, concretamente, desde la dirección FC0000 a la FFFF, o sea, 256 K, que son direcciones con las 18 líneas de direcciones. Las direcciones comprendidas entre la FFEOOO y la FFFFFE (8 K), se reservan para guardar la información de estado de los periféricos.

Hay 4 modos de transferencia a la memoria; uno de ellos realiza la transferencia de forma directa y los otros 3 a través de un buffer.

En el modo directo, el dato se almacena directamente en la memoria, realizándose un ciclo de acceso a memoria exclusivo para él.

En las modalidades por buffer, existen 3 buffers, tipo cache, cada uno de los cuales contiene 4 bytes para el almacenamiento de los datos, 16 bits para guardar la dirección y 5 bits para los señalizadores. Cuando están completos los 4 bytes del buffer, se realiza la transferencia a memoria, con el consiguiente ahorro de ciclos de acceso.

10.10.1.2. Traducción de direcciones

La dirección que maneja el UNIBUS consta de 18 bits, de los cuales los 9 de más peso especifican una página de 512 bytes y los 9 restantes, el desplazamiento dentro de ella.

Existe un mapa de traducción de direcciones que está estructurado en forma de tabla. La página seleccionada por la dirección del UNIBUS se utiliza como índice de entrada al mapa. Así se obtiene la dirección física de la página, de la que, añadiendo el desplazamiento, se calcula la dirección física absoluta. Este proceso de traducción se representa gráficamente en la figura 10-19.

Cada entrada del mapa de traducción de la figura 10-19, contiene la información que se indica en la figura 10-20.

$$\text{DIRECCION VECTOR} = \text{Base del SCB (SCBB)} + 400_{16} + \text{VECTOR DEL DISPOSITIVO (UNIBUS OPCIONAL)}$$

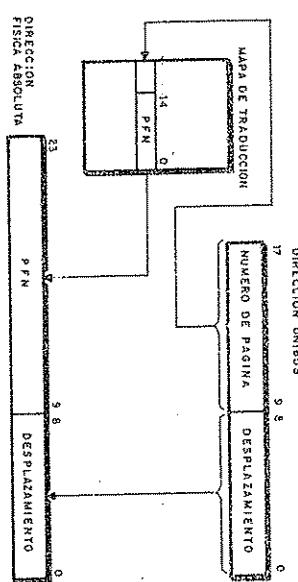


Fig. 10-19. Proceso de traducción para obtener la dirección física absoluta en el bus UNIBUS.

Con respecto a la figura 10-20, el campo de modo de transferencia puede tomar los siguientes valores:

- 0: Modo de transferencia directo
1, 2 ó 3: Modo de transferencia por buffer

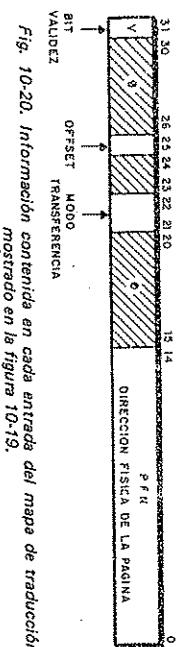


Fig. 10-20. Información contenida en cada entrada del mapa de traducción mostrado en la figura 10-19.

El offset, si se encuentra a 1, significa que la dirección entregada por el UNIBUS se incrementa en una unidad. El bit V de validez si se encuentra a 1, indica que el adaptador UNIBUS realiza el proceso de traducción. Si se encuentra V a 0, se ignora la petición del UNIBUS.

Finalmente, conviene resaltar que las interrupciones de los dispositivos están vectorizadas en el Bloque de Control del Sistema (SCB). La dirección donde se encuentra el vector se calcula según la expresión siguiente:

$$\text{DIRECCION VECTOR} = \text{Base del SCB (SCBB)} + 400_{16} + \text{VECTOR DEL DISPOSITIVO (UNIBUS)}$$

10.11. EL MASSBUS
Las características sobresalientes del bus normalizado MASSBUS son las siguientes:

- a) Transferencia de datos con acceso directo a memoria (DMA).
- b) Buffer de datos de 32 bytes. Permite alcanzar velocidades que llegan hasta los 2 Mbytes/s.
- c) Incorpora elementos de diagnóstico.
- d) Los registros de los dispositivos conectados al MASSBUS se direccionan como posiciones de memoria.

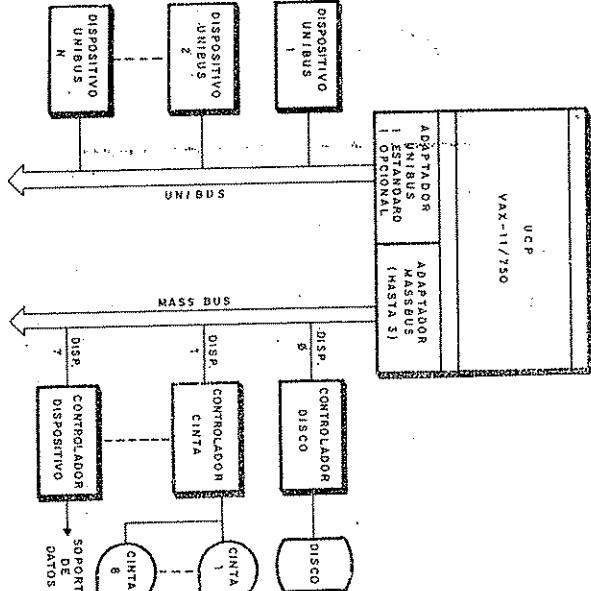


Fig. 10-21. Adaptación de periféricos al UNIBUS y MASSBUS.

El MASSBUS está compuesto por dos buses:

Bus de control. Para realizar el acceso a los registros internos de los dispositivos conectados al MASSBUS.

El adaptador del MASSBUS (MBA) puede controlar periféricos cuyo período en la transferencia de datos de 16 bits no supere los 900 ns, y un ancho de banda de 2 Mbytes/s. La UCP envía comandos que puede ejecutar el MBA, que devuelve información sobre la forma en que se ha realizado la operación.

De forma semejante al UNIBUS, existe un mapa de 256 registros de 32 bits, en donde se indica la dirección física de la página (PF) correspondiente a la página virtual.

Para facilitar la transferencia de información entre los dispositivos y el MASSBUS se utiliza el buffer de 32 bytes, ya comentado. Los datos de memoria se envían divididos en dos palabras de 16 bits, hacia los dispositivos.

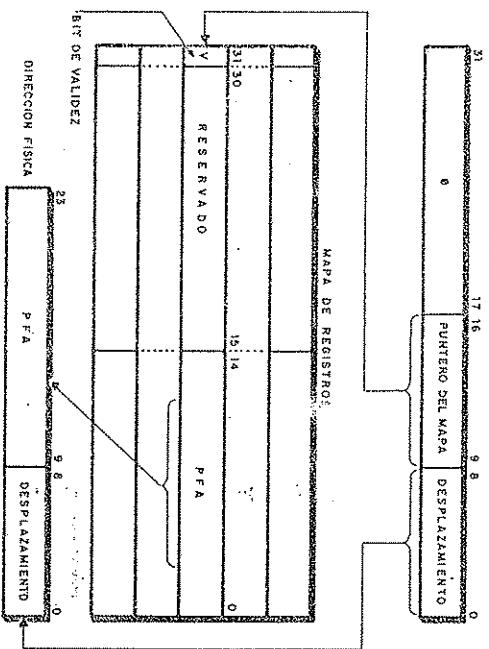


Fig. 10-22. Representación gráfica sobre la forma en la que se efectúa la traducción de direcciones en el MASSBUS.

4.34 / Descripción de la arquitectura

El MBA soporta las siguientes funciones:

1. Traducción de direcciones virtuales a reales.
2. Memorización de datos para realizar las transferencias entre la memoria principal y el MASSBUS.
3. Transferencia de las interrupciones de los dispositivos conectados al MASSBUS y el sistema.

El VAX-11/750 admite hasta 3 MBA (2 en caso de existir el UNIBUS opcional), cada uno de los cuales puede soportar hasta 8 controladores de periféricos. Alguno de estos controladores pueden manejar más de un periférico; por ejemplo, cada controlador de cinta magnética soporta hasta 8 cintas. El controlador de disco sólo admite uno.

El esquema de la figura 10-21 muestra la conexión de los periféricos a los buses normalizados UNIBUS y MASSBUS del procesador VAX-11/750.

TIPO DE DATO	TAMAÑO	LÍMITES	
		CON SIGNO	SIN SIGNO
ENTERO BYTE	(BITS)		
PALABRA	16	-128 a +127	0 a 255
DOBLE PALABRA	32	-32768 a +32767	0 a 65535
CUADRUPLE PALABRA	64	-231 a +231 -1	0 a 268 -1
COTA PALABRA	128	-2127 a +2127 -1	0 a 2128 -1
<i>COLA FLOTANTE</i>			
TIPO F	32	APPROXIMADAMENTE SIETE DÍGITOS DECIMALES DE PRECISIÓN	
TIPO D	64	APPROXIMADAMENTE 16 DÍGITOS DECIMALES DE PRECISIÓN	
TIPO G	64	APPROXIMADAMENTE 15 DÍGITOS DECIMALES DE PRECISIÓN	
TIPO H	128	APPROXIMADAMENTE 35 DÍGITOS DECIMALES DE PRECISIÓN	
CADENA DE CARÁCTERES	0 a 16 BYTES	DOS DÍGITOS POR BYTE. EL SIGNO EN LA DÉCIMALES ENPACO. 131 DÍGITOS) MITAD BAJA DEL ÚLTIMO BYTE	
CAMPOS DE BITS	0 a 32 BITS	SEGUN SE INTERPRETE	
CADENA NUMÉRICA	0 a 65536 BYTES	UN CARÁCTER POR BYTE	
CARACTERES	0 a 31 BYTES	10 ³¹ - 1 a +10 ³¹ - 1	
COLAS	2 DOBLES PALABRAS DE ENTRADA A LA COLA QUE DESPLAZA EN MEMORIA		
DIRECCIÓN FÍSICA	25		
PF A	9		
DESPLAZAMIENTO	0		

Fig. 10-23. Relación de los tipos de datos admitidos por los VAX-11, junto al tamaño y los límites con y sin signo.

Descripción de la arquitectura / 4-35

10

10

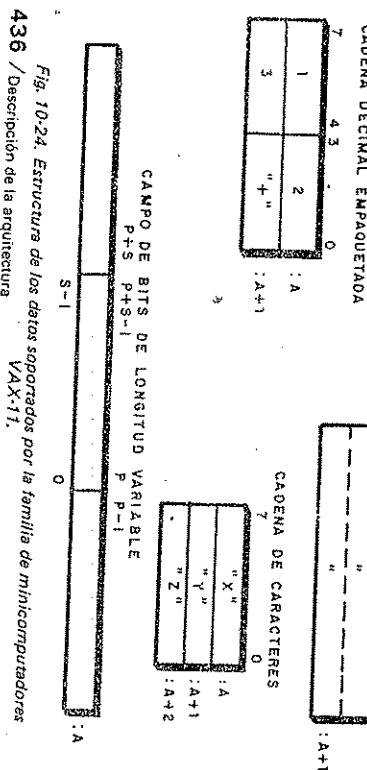
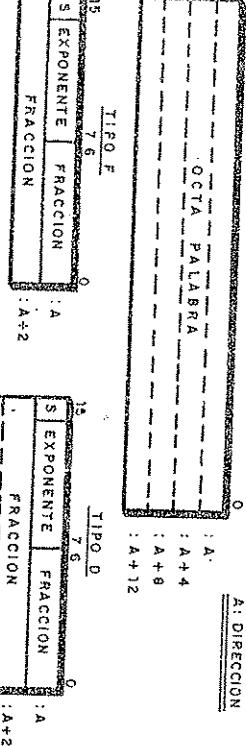
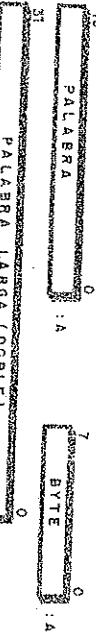


Fig. 10-24. Estructura de los datos soportados por la familia de minicomputadores VAX-11.

436 / Descripción de la arquitectura

Finalmente, en la figura 10-22 se representa gráficamente la forma en la que se realiza la traducción de direcciones en el MASSBUS, que es muy parecida a la empleada en el UNIBUS.

10.12. TIPOS DE DATOS ADMITIDOS POR LAS INSTRUCCIONES DEL VAX-⁴¹

El procesador, en modo nativo, reconoce 5 tipos de datos fundamentales:

1. Enteros.
2. Números expresados en coma flotante.
3. Decimales empacados.
4. Cadenas de caracteres.
5. Campos de bits.

Asimismo, también soporta dos tipos de estructura de colas. En las figuras 10-23 y 10-24 se muestran de forma gráfica y resumida, las principales características de los datos admitidos por los VAX-11.

10.13. EL FORMATO DEL CODIGO MAQUINA DE LAS INSTRUCCIONES

La estructura que debe tener el código máquina de una instrucción capaz de ser interpretada y ejecutada por las UCP de la familia VAX-11 responde a la que se presenta en la figura 10-25.

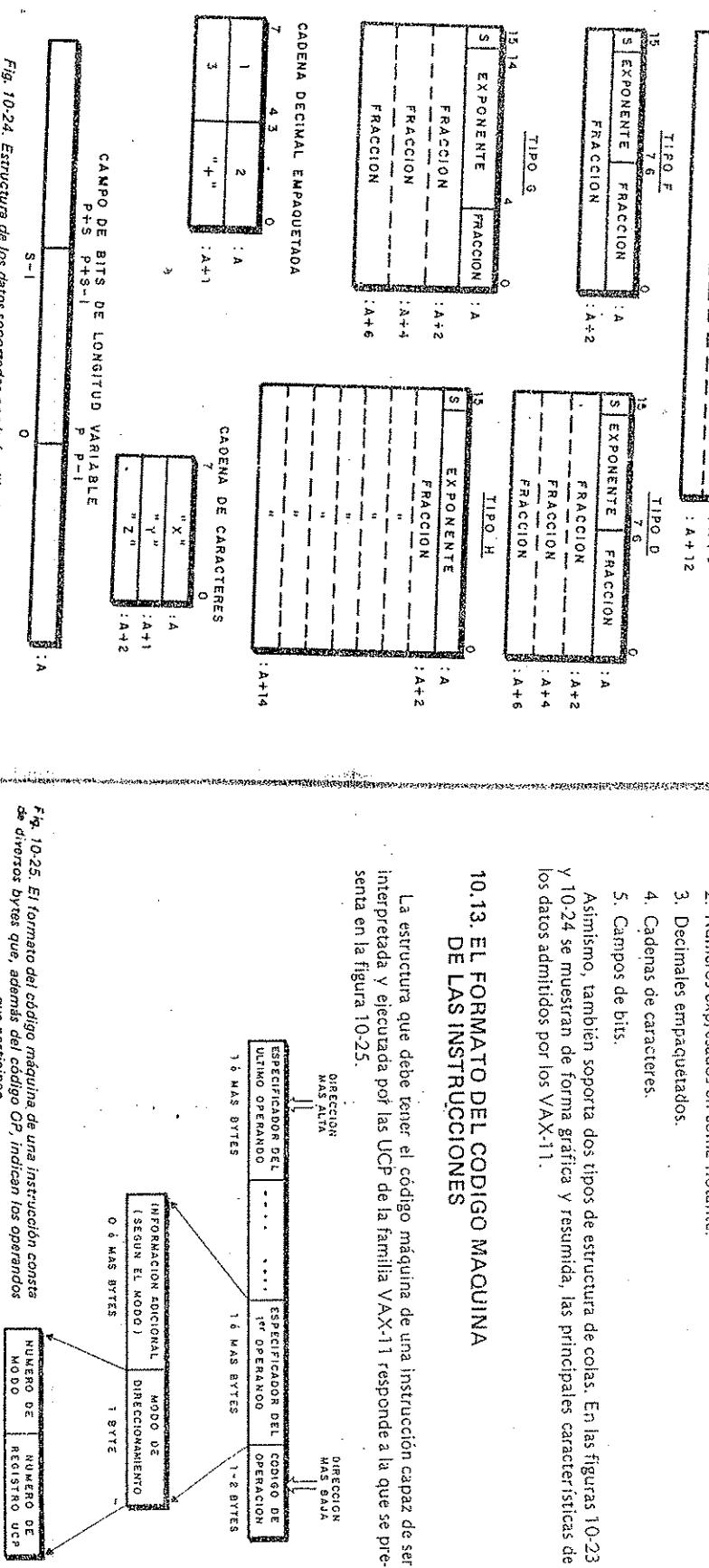


Fig. 10-25. El formato del código máquina de una instrucción consta de diversos bytes que, además del código OP, indican los operandos que participan.

Descripción de la arquitectura / 437

Como se aprecia en la figura 10-25, delante del código de operación de la instrucción, situado en la dirección más baja de memoria de las que ocupa la instrucción, están los especificadores de los operandos que se emplean. Cada especificador indica el modo de direccionamiento particular del operando y, si es necesario, información complementaria. El byte de modo dedica los 4 bits de más peso a expresar el número del modo a utilizar y los 4 bits de menos peso, a señalar el registro interno de la UCP que participa en la instrucción.

Una instrucción que en lenguaje Ensamblador se expresa:

CLRW –(R5)

se transforma, cuando pasa a código máquina, en:

75 B4

en el que cada uno de estos dígitos hexadecimales significa:

7: Modo de direccionamiento.

5: Registro de la UCP.

B4: Código de operación de CLRW.

La instrucción, que en lenguaje Ensamblador se representa

ADDL2 (R2) +, R7

en código máquina se transforma en:

57 82 C0

Siendo

5: Modo de direccionamiento.

7: Registro de la UCP.

8: Modo de direccionamiento.

2: Registro de la UCP.

C0: Código de operación de ADDL2.

Una instrucción que en lenguaje Ensamblador se expresa:

10.14. LOS OPERADORES

Se llaman operadores a los elementos de una instrucción que indican la acción que debe ejecutarse. En lenguaje Ensamblador hay tres tipos: *instrucciones máquina, directivas y macros*. Las primeras las traduce a código máquina el programa Ensamblador. Las directivas son instrucciones u órdenes para el Ensamblador y, por último, las macros son pseudoinstrucciones que están en las librerías y que, a su vez, pueden contener instrucciones, directivas u otras macros.

Detrás del operador se indica con una letra y un número, la longitud de los operandos. Y el número de ellos que participan en la instrucción. La longitud se expresa con las siguientes letras:

B – Byte

W – Palabra

L – Doble palabra

Q – Cuádruple palabra

P – Decimal empaquetado

F – coma flotante y simple precisión

D – coma flotante y doble precisión

G – coma flotante, tipo G.

H – coma flotante, tipo H

C – Carácter

V – Campo de longitud variable

Algunas representaciones típicas de operadores son:

MULW3: Multiplica los dos primeros operandos, con longitud de palabra, dejando el resultado en el tercer operando.

CVTLF: Convierte una doble palabra a coma flotante en simple precisión.

INCW: Incrementa una unidad un operando de longitud una palabra.

10.15. MODOS DE DIRECCIONAMIENTO

En términos generales, en los procesadores de la familia VAX, cualquier *direccionamiento* puede ser usado en todas las instrucciones. Por otra parte, el operando puede estar en un registro general en la memoria o puede ser una constante que acompaña a la propia instrucción.

En los procesadores de la familia VAX se admiten los 6 modos de direccionamiento básicos:

1. Modo inmediato o literal.
2. Modo absoluto.
3. Modo bifurcación.
4. Modo registro.
5. Modo indexado.
6. Modo indirecto.

10.15.1. Modo inmediato o literal

El operando se expresa en la misma instrucción y puede ser un entero o un valor en coma flotante. En lenguaje ensamblador, debe ir precedido del signo #.

El modo literal sólo admite operandos de 6 bits, mientras que el inmediato admite más bits.

Ejemplo

MOVW #25, R8

Mueve el valor 25 a la palabra de menos peso de R8. Como $25_{10} = 19_{16}$, si R8, antes de esta instrucción, contiene AABBB3210, después de la instrucción contendrá AABB0019.

Para usar un literal como un carácter se emplea el formato: # ^ l char l (código ASCII).

Ejemplo

*MOVW # ^ l * \ NOM*

Mueve el código ASCII del carácter * a NOM, con longitud de byte. En el modo literal, los dos bits de más peso son cero, porque sólo se emplean 6 bits <64 caracteres>.

10.15.2. Modo absoluto

Con este modo se especifica la dirección donde se halla almacenado el operando, o bien, la dirección donde hay que depositar el resultado.

440 / Descripción de la arquitectura

Ejemplo

MOVC #61, TALLA, COLOR

Mueve una cadena de 61 caracteres desde TALLA a COLOR.

10.15.3. Modo bifurcación

Es una instrucción de bifurcación que indica a qué dirección del programa se cede el control. Normalmente usa una etiqueta. Si el operador es un salto condicional, o bien, condicional y se cumple la condición, el Contador de Programa se carga con la dirección a donde se produce el salto.

Ejemplo

BRB SALTO

Se salta a la instrucción de programa que va etiquetada con SALTO.

Este modo es implícito y el especificador de operando sólo contiene el desplazamiento, es decir, el número de bytes que se salta hacia adelante o hacia atrás. Este número se codifica en complemento a dos.

10.15.4. Modo registro

Hay 7 modos de direccionamiento que hacen uso de los registros generales de la UCP.

Ejemplo

1. Por registro

Un registro general Rn contiene el operando. Si el operador va seguido de B o W, la parte de menos peso del registro (8 ó 16 bits) es la que se utiliza. Si, por el contrario, la longitud del operando es mayor de 32 bits, éste estará formado por Rn y R(n+1).

Ejemplo

MOVW R6,R9

Si antes de ejecutarse esta instrucción, R6 = 00552245 y R9 = BBBBAA, después quedará con R6 = 00552345 y R9 = BBBB2345.

Descripción de la arquitectura / 441

2. Por registro indirecto

El registro contiene la dirección del operando. En este caso, el registro se representa entre paréntesis (R_n).

Ejemplo

$ADDW2 R5, (R8)$

Suma la palabra de menor peso de $R5$ a la palabra situada en memoria en la dirección contenida en el registro $R8$.

3. Con autoincremento

El contenido del registro primero se decrementa según el tamaño del operando (una unidad por byte) y luego se usa como dirección del operando.

Ejemplo

$MOVW -(R8), R5$

El contenido de $R8$ se decremente una unidad y apunta a una dirección de la memoria cuyo valor almacenado en ella se traslada, a nivel de byte, a $R5$.

4. Con autoincremento

En este caso, primero se halla la dirección del operando que es la que indica el contenido del registro Y , una vez realizada la instrucción, dicho registro se incrementa en tantas unidades como número de bytes contenga el operando.

Ejemplo

$ADDL2 (R9)+, R3$

Si se supone que esta instrucción se incluye en un bucle, que se ejecuta tantas veces como entradas hay en una tabla, sumará todas las entradas de la tabla (diseñada por $R9$) y depositará el resultado en $R3$.

$$R_3 = R_3 + \sum \text{Entradas de la Tabla}$$

$R9 = R9 (\text{initial}) + n.º \text{ entradas} \times \text{largo del operando} (B, W, ...)$

En este modo, si un mismo registro se usa más de una vez en una instrucción, se incrementará cada vez que se encuentre un operando que lo utilice.

4.4.2 / Descripción de la arquitectura

5. Por autoincremento indirecto

El contenido del registro se usa como dirección de una posición de memoria que contiene la dirección del operando; luego, se incrementa el registro manejado en tantas unidades como número de bytes contenga el operando.

6. Por desplazamiento

La dirección del operando se obtiene sumando al contenido de un registro (base de direcciones) un número que actúa como desplazamiento.

Ejemplo

$MOVL 34(R6),(R2)+$

En esta instrucción, 34 es el desplazamiento que se suma al contenido del registro $R6$ para encontrar la dirección del operando.

Hay tres modos de desplazamiento, según la cantidad que indica el desplazamiento se encuentre en 1, 2 ó 4 bytes:

TIPO MODO DE DESPLAZAMIENTO	NUMERO USADO
B, desplazamiento (R_n)	A
W, desplazamiento (R_n)	C
L, desplazamiento (R_n)	E

Cuando se omite el tipo (B, W, L), el Ensamblador determina el más idóneo, procurando que ocupe lo mínimo posible. Si no tiene datos para averiguarlo, asume el tercer tipo (número de modo E).

7. Por desplazamiento indirecto

El contenido del registro es una base a la que se suma un desplazamiento para obtener la dirección donde se encuentra la dirección del operando.

Ejemplo

$MOVW @8 (R7), R10$

La dirección del primer operando es el contenido de la dirección $(R7) + 8$.

10.15.5. Modo indexado

En todos los modos, excepto por registro, se puede emplear un registro INDICE para modificar el valor del operando. El especificador del operando se expresa de esta forma:

Especificador de base [Rx]

Rx actúa como registro índice. El operando será el contenido de la dirección:

Dirección base + (Rx) × longitud del dato

El modo de direccionamiento de la base puede ser cualquiera excepto literal, inmediato, por registro, bifurcación o indexado.

Ejemplo

TSTW LISTA [R6]

El operando es el contenido de la dirección:

LISTA + (R6) × 2 (por ser longitud W)

10.16. UN BREVE RECORRIDO POR EL REPERTORIO DE INSTRUCCIONES

Explicar con detalle cada una de las instrucciones máquina del repertorio usado en los procesadores de la familia VAX-11, sobre todo los límites de este libro, por lo que sólo se hace una sucinta descripción de las características más importantes de las instrucciones representativas.

10.16.1. Instrucciones con números enteros y coma flotante

La mayor parte de las instrucciones que admiten números enteros, también aceptan números expresados en el formato de coma flotante. Hay algunas excepciones como sucede con las instrucciones Bit Clear, Complement, Add/Subtract With Carry, etc.

Las instrucciones con números enteros admiten operandos de 1, 2 ó 4 bytes de longitud y las que hacen referencia a valores expresados en coma flotante, en más. Afectan a los bits de la palabra de estado PSW del procesador, que expresan el signo, el acarreo, el sobrepasamiento y el valor cero del resultado.

Cuando se trabaja con los registros generales, se emplea el byte o la palabra de menor peso del registro sin alterar el resto de los bytes. Además, el bit de signo no se extiende hasta el bit más a la izquierda del registro, como ocurre en otras UC.P.

10.6.2. Instrucciones aritméticas

Admiten 2 y 3 operandos, lo que evita mover datos en registros temporales. Con dos operandos, en uno de ellos se almacena el resultado y, si son tres, las instrucciones se asemejan mucho a las típicas de los lenguajes de alto nivel.

Hay una instrucción específica para cada operación aritmética básica:

ADDx2	op1,op2	op1 + op2 → op2
ADDx3	op1,op2,op3	op1 + op2 → op3
MULx2	op1,op2	op1 * op2 → op2
MULx3	op1,op2,op3	op1 * op2 → op3
SUBx2	op1,op2	op2 - op1 → op2
SUBx3	op1,op2,op3	op2 - op1 → op3
DIVx2	op1,op2	op2/op1 → op2
DIVx3	op1,op2,op3	op2/op1 → op3

La dirección del operando origen se halla en la dirección de memoria dada por $(R5) + 5$.

siendo x el tamaño del operando, que puede ser B, W, L, F; etc.

Otras instrucciones de alguna manera relacionadas con las de tipo aritmético son:

C/LRx operando ($x = B, W, L, Q \circ O$)

Rellena el operando con ceros.

INCx operando ($x = B, W, L$)

Incrementa en una unidad el operando.

DECx operando ($x = B, W, L$)

Decrementa en una unidad el operando.

MNEG operando fuente, operando destino ($x = B, W, L$)

Mueve el complemento del operando fuente al operando destino.

EMUL (Multiplicación Extendida)

Toma argumentos de doble palabra y produce resultados de cuádruple palabra.

EDIV

Divide una cuádruple palabra por una palabra larga y produce un cociente de doble palabra y un resto de palabra larga.

EMOD (Módulo Extendido)

Multiplica un número en coma flotante por otro número en coma flotante (F o D) y devuelve separadamente la porción entera y la fraccionaria.

POLY (Evaluación Polinómica)

Evaluá un polinomio desde una tabla de coeficientes usando el método de Horner.

En la figura 10-26 se muestra una relación de las instrucciones aritméticas con enteros y coma flotante. Existe también un juego de instrucciones para el manejo de operandos en decimal empaquetado, que no se comenta dada su similitud con las instrucciones expuestas.

10.16.3. Instrucciones lógicas y de movimiento

En este grupo se han incluido instrucciones de la importancia, dado su frecuente uso, de *MOV*, que copia el contenido del operando fuente en el destino y actúa sobre el registro PSW.

446 / Descripción de la arquitectura

INC	: INCREMENTAR (B,W,L)
DEC	: DECREMENTAR (B,W,L)
ASH	: DESPLAZAMIENTO ARITMÉTICO (L,O)
ADD	: SUMA (B,W,L,F,D,G,H) CON DOS OPERANDOS
ADD	: SUMA (B,W,L,F,D,G,H) CON TRES OPERANDOS
ADWC	: SUMA CON ACABREO
ADAWL	: SUMA ALINEADA DE PALABRAS "INTERLOCKED" (PROPIEDAD DE UNA LECTURA SEGUIDA DE ESCRITURA DEL MISMO DATO, QUE NO ES POSIBLE REFERENCIAR POR UN SEGUNDO PROCESADOR O ELEMENTO I/O)
SUB	: RESTA DE 2 OPERANDOS
SUB	: RESTA DE 3 OPERANDOS
SWC	: RESTA CON ACARRO
MUL	: MULTIPLICACIÓN (B,W,L,F,D,G,H) 2 OPERANDOS
MUL	: MULTIPLICACIÓN (B,W,L,F,D,G,H) 3 OPERANDOS
DIV	: DIVISIÓN (B,W,L,F,D,G,H) 2 OPERANDOS
DIV	: DIVISIÓN (B,W,L,F,D,G,H) 3 OPERANDOS
EDIV	: DIVISIÓN EXTENDIDA
EMOD	: EXTENSIÓN DE MÓDULO (F,D,G,H)
POLY	: EVALUACIÓN POLINÓMICA (F,D,G,H)

Fig. 10-26. Instrucciones aritméticas con enteros y con números en coma flotante.

CVTxy op fuente, op destino

La longitud de op. fuente (x) y la del destino (y), pueden ser diferentes y también es una instrucción de movimiento, pero con la posibilidad de afectar a distintas longitudes en los operandos. Si el operando fuente es más corto que el destino, no se pierde información y se efectúa la extensión del bit de signo.

Ejemplo

CVTBW RS,R2

Cuando el operando fuente es más largo que el destino, se puede perder información y, además, alterarse el signo.

MVAX op fuente, op destino

Mueve la dirección indicada en el operando fuente (no su contenido) al destino, que debe ser de 32 bits.

MOVCA longitud, op fuente, op destino

Mueve el número de bytes expresados por longitud, entre la dirección inicial indicada en el operando fuente y la dirección final expresada por el operando destino.

10

CVTSP

Convierte el formato numérico a decimal empaquetado.

CVTPL

Convierte un número decimal empaquetado a complemento a dos.

CVTLP

Es la inversa de CVTPL.

CVTPS

Es la inversa de CVTSP.

B/Cx2 máscara, cadena

Los bits correspondientes indicados en la máscara se ponen a cero.

XORx3 máscara, fuente, destino

Realiza la función OR exclusiva.

MOV_	: MOVER (B,W,L,F,D,G,H,O)
MVDA_	: MOVER DIRECCIONES (B,W,L,D,O)
MNEG_	: MOVER NEGANDO
PUSHA_	: NETE UNA DIRECCION EN LA PILA A UNO O "STACK"
MCNA_	: MOVER CON COMPLEMENTO
MNZZ_	: MOVER CON EXTENSION DE CEROS
CRR_	: BORRAR (B,W,L + F,Q + G,H)
CVT_	: CONVERTIR FORMATOS (B,W,L,F,D,G,H) (B,W,L,F,D,G,H)
CVTR_	: CONVERSION CON REDONDEO A PALABRA LARGA
CMPL_	: COMPARACION
TST_	: TESTADO
BIS_2	: BIT SET (B,W,L) 2 OPERANDOS
BIS_3	: BIT SET (B,W,L) 3 OPERANDOS
BIC_2	: BIT CLEAR (B,W,L) 2 OPERANDOS
BIC_3	: BIT CLEAR (B,W,L) 3 OPERANDOS
BYR_	: BIT TEST
XOR_2	: EOR CON 2 OPERANDOS
XOR_3	: EOR CON 3 OPERANDOS
ROTLL_	: ROTACION DE PALABRA LARGA
ASH_	: DESPLAZAMIENTO ARITMETICO (L,O)

Fig. 10-27. Instrucciones lógicas y de movimiento.

10

MCUMX fuente, destino

Halla el complemento a uno.

ROTL número, op fuente,,op destino

Produce una rotación de bits en el operando fuente que se mueven a lo largo del destino, de forma que, si número > 0, la rotación es a la izquierda y si número < 0, es a la derecha.

10.16.4. Instrucciones de cadenas de caracteres

Este grupo de instrucciones permite operar con cadenas de caracteres, realizando las siguientes funciones:

1. Mover cadenas, con diversas opciones.
2. Comparar cadenas.
3. Buscar un carácter en una cadena.
4. Buscar una subcadena.

MOVCL longitud, op fuente, op destino

Copia la cadena que comienza en el operando fuente a las posiciones que comienzan en el operando destino, con la longitud que se indica en la propia instrucción.

La instrucción LOCC busca el primer carácter de una cadena que sea igual a un carácter dado.

LOCC carácter, longitud, cadena

La instrucción SKPC busca el primer carácter de una cadena, distinto de uno dado.

SKPC carácter, longitud, cadena

Estas instrucciones devuelven el resultado de su actuación en los registros R0 y R1, así como en el señalizador Z. En R0 dejan el número de bytes no utilizados en la cadena, incluyendo el encontrado. En R1 direcciona el byte encontrado, o si no se encontró, la dirección del byte siguiente a la cadena. Finalmente, Z = 0, si se encuentra el byte deseado.

Las restantes instrucciones que manejan cadenas de caracteres se muestran en la figura 10-28.

MVC 3	MOVER CADENAS DE CARACTERES
MOVTC	MOVER Y TRADUCIR (CON LISTA DE OFFSETS) CARACTERES
MOVTC	MOVER Y TRADUCIR HASTA UN CARACTER
MOVTC	MOVER Y TRADUCIR HASTA UN CARACTER
CMPC 3	COMPARAR CARACTERES
CMPC 3	COMPARAR CARACTERES
LOC C	LOCALIZAR CARACTER DADO
SXPC	SALTAR A CARACTER DISTINTO DEL DADO
SCANC	EXPLORAR CARACTERES
SPANC	EXPANDIR CARACTERES
MATCH C	LOCALIZAR MULTIPLES CARACTERES O SUBCADENAS

Fig. 10-28. Instrucciones encargadas del tratamiento de cadenas de caracteres.

10.16.5. Instrucciones para el tratamiento de campos de bits de longitud variable

INSV fuente, posición, número, base
Inserta los bits (número - 1) hasta cero de la doble palabra del operando fuente, en las posiciones indicadas por los tres últimos operandos.

EXTV posición, número, base, destino

Extrae los bits indicados por los tres primeros operandos en la doble palabra señalada por destino, rellenando los bits sobrantes a la izquierda con el mismo valor que el bit de signo.

En la figura 10-29 se presentan las instrucciones correspondientes a este grupo.

EXTV	TRAE UN CAMPO DE BITS Y LO ALMACENA EN LOS BITS DE MENOS
EXTV	RESO DE UNA PALABRA LARGA PUEDE SER CON SIGNO O SIN SIGNO
INSV	INSERTA BITS
CMPIV	PERMITE AL USUARIO TESTAR O COMPARAR UN CAMPO.
CMPIZV	PUEDE SER CON O SIN SIGNO.
FFS	LOCALIZA EL PRIMER BIT A UNO EN UN CAMPO.
FFC	LOCALIZA EL PRIMER BIT A CERO EN UN CAMPO.

Fig. 10-29. Instrucciones que actúan sobre campos de bits

4.50. Descripción de la arquitectura

10.16.6. Instrucciones de manejo de registros

Se encargan de cargar y descargar datos y registros en la pila o stack, así como manejar la doble palabra de estado PSW y la palabra de estado PSW. En la figura 10-30 se muestra una breve descripción de este grupo de instrucciones.

La PSW contiene en los 4 bits de menos peso, los señalizadores N, Z, V y C, y en los siguientes 4 bits, los señalizadores de habilitación de excepciones:

DV: Desbordamiento o sobre pasamiento decimal.

FV: Desbordamiento en coma flotante.

IV: Desbordamiento con números enteros.

T: Trazado.

PUSHL	CARGA EN LA PILA O "STACK" UNA PALABRA LARGA
PUSHR	CARGA EN LA PILA O "STACK" UN REGISTRO
POPA	SACA DEL "STACK" UN REGISTRO
MOVPSL	MUEVE UNA PALABRA LARGA DESDE EL REGISTRO ESTADO
BIS/SW	BIT A UNO EN PALABRA DE ESTADO
BIC/SW	BIT A CERO EN PALABRA DE ESTADO

Fig. 10-30. Grupo de instrucciones encargadas de la manipulación de registros diversos.

Cuando se produce alguno de estos 4 últimos casos, se origina una excepción, sólo si el señalizador de habilitación de la excepción correspondiente está a 1.Inicialmente están a 0 y los señalizadores IV y DV deben habilitarse o inhabilitarse al principio del programa principal. El resto de los bits pueden modificarse con las instrucciones B/S/S/W Y B/C/C/W.

10.16.7. Instrucciones para las bifurcaciones y los bucles

La mayoría de las instrucciones de bifurcación son de tipo condicional, es decir, el PC se modificará si se cumple una determinada condición, que proviene de la comparación del estado de los señalizadores N (signo), V (desbordamiento), (cero) y C (acarreo).

El formato general de estas instrucciones es:

Bxyz op destino

MÉMORICO	SALTA SI (TRAS TESTEo TST)	SALTA SI (TRAS COMPARAC. CMP)	SALTA SI (SEñALIZADORES)
<u>DATOS CON SIGNO</u>			
BZOL	OPERANDO = 0	OP \neq 0	OP \neq 0
BNEO		OP > 0	OP \neq OP 2
BGTR		OP > 0	OP $>$ OP 2
BLEQ		OP \leq 0	OP \leq OP 2
BGEQ		OP \geq 0	OP \geq OP 2
BLSS		OP $<$ 0	OP $<$ OP 2
<u>DATOS CON SIGNO</u>			
BXYZ	(x y z IGUAL QUE EN EL GRUPO ANTERIOR)	(IGUAL AL GRUPO ANTERIOR)	(IGUAL AL GRUPO ANTERIOR SUSTITUYENDO N POR C)
<u>PARA DESBORDAMIENTO Y ACARREO</u>			
BVS	DESBORDAMIENTO	V \neq 1	
BVC	NO HAY DESBORDAMIENTO	V = 0	
BCS	A ACARREO	C = 1	
BCC	NO HAY ACARREO	C = 0	
<u>CONDICIONALES AL BIT 0 DE MENOS PESO</u>			
BLBS	OP, DEST.	BIT 0 DE OP \neq 1	
BLBC	OP, DEST.	BIT 0 DE OP = 0	
<u>SALTOS INCONDICIONALES</u>			
BRA	OP SE CONSIDERA PALABRA LARGA	EL OPERANDO QUE INDICA LA DISTANCIA ES DE 1 BYTE.	
BRW		EL OPERANDO QUE INDICA LA DISTANCIA ES DE 2 BYTES.	
JMP		ADmite todos los DIRECCIONADOS EXCEPTO EL INMEDIATO Y POR REGISTRO.	

Fig. 10-31. Diferentes tipos de instrucciones de bifurcación y salto.

452 / Descripción de la arquitectura

siendo x,y,z quienes especifican la condición del salto y el operando destino, la dirección donde se bifurca el programa si se satisface la condición.

Normalmente la instrucción de bifurcación va detrás de una del tipo TST (comprobación) o del tipo CMP (comparación). La TST examina enteros en complemento a dos Y, según el resultado, modifica los señalizadores Z y N, dejando a cero el C y el V. La instrucción CMP resta sus dos operandos sin generar resultado, afectando únicamente a los señalizadores Z, N y C, dejando a cero a V.

En la figura 10-31 se muestran las diversas instrucciones de bifurcación junto a sus condiciones de salto.

Hay instrucciones destinadas a la creación de bucles del tipo "REPEAT UNTIL", es decir, bucles por los que el control del programa pasa al menos una vez y que se representan de manera gráfica en la figura 10-32.

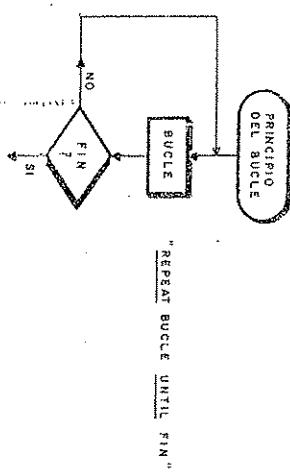


Fig. 10-32. Ordinograma de funcionamiento de un bucle del tipo "REPEAT UNTIL".

Para implementar la estructura "WHILE DO", mostrada en la figura 10-33 a), se simula con la estructura "REPEAT UNTIL", incluyendo una pregunta inicial para verificar si el valor que indica el fin ya se ha sobrepasado, tal como se representa en la figura 10-33 b).

Hay tres tipos de instrucciones para construir bucles, que admiten los procesadores VAX-11:

- 1) SOBGTR y SOBGEQ

Responden al formato

SOBGxy Índice, destino

El índice de esta pareja de instrucciones debe ser una doble palabra y el destino debe estar situado a una distancia inferior a 127 bytes por encima o por debajo.

2) AOBLEQ Y AOBLS

Responden al formato:

AOBLx y límite, índice, destino

Su funcionamiento se representa en los organigramas de la figura 10-35.

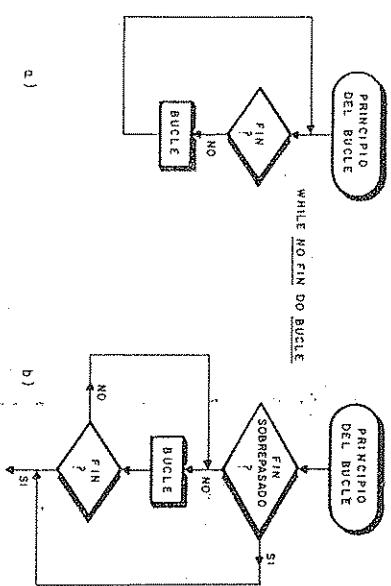
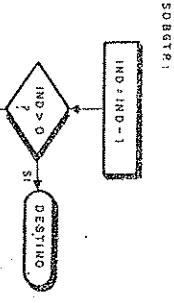


Fig. 10-33. A la izquierda, el organigrama de la estructura "WHILE DO" y a la derecha, simulación de dicha estructura mediante el bucle "REPEAT UNTIL".

Actúan de acuerdo con los organigramas mostrados en la figura 10-34.



Los operandos índice y límite deben ser de 32 bits.

Con ayuda de estas instrucciones se pueden construir bucles del tipo "FOR" o "DO", típicos en los lenguajes de alto nivel.

3) ACBX

Permite crear bucles del tipo "FOR" y "DO" con incrementos o decrementos distintos a la unidad. Su formato es:

ACBX límite, incremento, índice, destino

En donde:

$x = B, W, L, F, D, G \text{ y } H$

Límite, incremento e índice han de ser de la misma longitud que x .

Destino indica un salto de hasta 32.767 bytes por encima o por debajo.

En la figura 10-36 se muestra el organigrama de funcionamiento de esta instrucción.

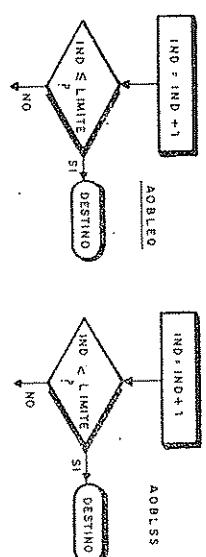


Fig. 10-35. Organigramas del funcionamiento de las instrucciones AOBLEQ y AOBLS.

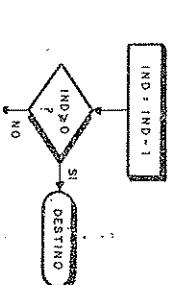


Fig. 10-34. Organigramas de funcionamiento de las instrucciones SOBGEQ y SOBGTG.

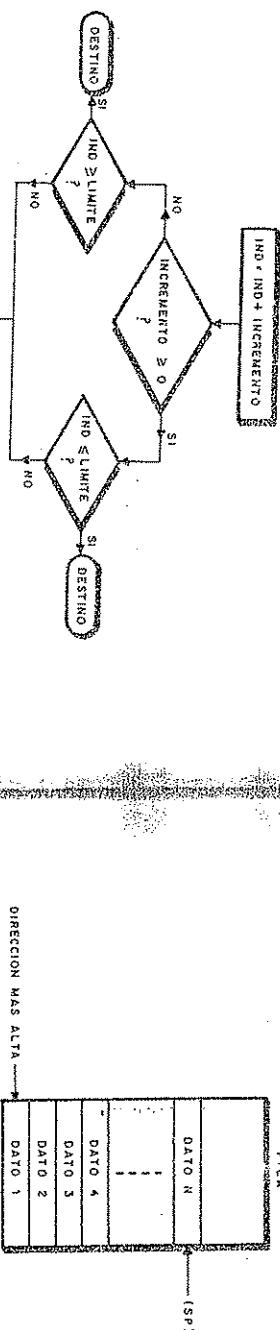


Fig. 10-36. Organigrama que representa la operatividad de la instrucción ACBX.

La instrucción CASE, al ejecutarse, permite al usuario disponer de una lista de desplazamientos para generar diferentes desplazamientos tendentes a seleccionar entre varios caminos de bifurcación. Es una especie de instrucción del tipo "ON n GOTO".

10.16.8. Instrucciones de manejo de subrutinas

En este grupo existen tres instrucciones:

BSB: Bifurcación a subrutina con desplazamiento.

JSB: Salto a subrutina.

RSB: Retorno de subrutina.

10.16.9. Instrucciones de llamada y retorno a procedimientos

El manejo de los procedimientos implica el uso de las pilas o stacks, tipo LIFO. A cada usuario se le asigna una pila, que tiene un puntero (SP), el cual, inicialmente, apunta a la dirección más alta de la pila. Según se van sacando o introduciendo datos, el puntero SP se incrementa o decremente, respectivamente. Figura 10-37.

Con la instrucción PUSH se pueden introducir datos en la pila y con la POP, se pueden sacar.

La UCP utiliza la pila para *salvar* el contenido del Contador de Programa, cuando se produce un salto a una subrutina (BSB o JSB). Al final de la subrutina existirá una instrucción de retorno (RSB), que recupera de la pila el valor del PC que se almacenó al ejecutarse BSB. Por este motivo, si se usa un número de instrucciones

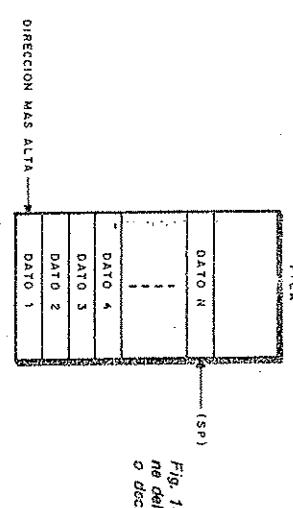


Fig. 10-37. La pila, tipo LIFO, dispone del puntero Sp que se incrementa o disminuye según se saquen o se metan datos de la pila.

PUSH durante una subrutina, hay que realizar el mismo número de instrucciones POP, para que al final, el SP dirccione la posición adecuada de la pila.

Cuando se llama a un procedimiento, se utiliza una lista de argumentos que necesita el desarrollo del procedimiento. Dicha llamada se materializa con las instrucciones CALL y CALLS, que, entre otras cosas, colocan la dirección de comienzo de la lista de argumentos en el Puntero de Argumentos (AP) y cambian el PC por la dirección en que comienza el procedimiento.

El procedimiento dispone de una máscara de entrada, que indica los registros que se usarán. Las instrucciones CALL emplean esta información para averiguar los registros que se han de salvar en la pila. Al final del procedimiento, habrá una instrucción RET que repone los registros salvados y coloca en el PC la dirección de la instrucción que sigue a CALL.

En lenguaje Ensamblador existe un directivo (.ENTRY) que facilita el comienzo de un procedimiento. Está compuesto, además del nombre del procedimiento, de la máscara de entrada, que señala los registros que deben salvarse y los señalizadores (IV y DV) que deben estar activos para generar una excepción en caso de error. El Ensamblador crea una máscara de 16 bits, poniendo a 1 los bits correspondientes a los registros y señalizadores a emplear. Los registros que se indican en la máscara sólo pueden ser los numerados del 2 al 11, puesto que R0 y R1 se usan para devolver valores de la función.

Cuando se llama a un procedimiento hay que proporcionar una lista de datos que serán procesados en él. A su vez, el procedimiento devuelve unos resultados. Todos estos datos configuran la lista de argumentos, que toma la forma que se muestra en la figura 10-38.

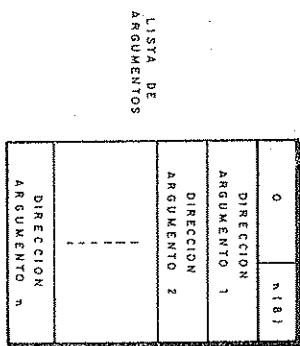


Fig. 10-38. Estructura de la lista de argumentos. El byte n indica el número de argumentos, que será inferior a 255 y que contiene la dirección del argumento, realmente.

La instrucción CALL crea en la pila la *Región de llamada*, donde se guardan los datos necesarios para regresar al programa desde el que se ha llamado al procedimiento. Figura 10-40.

Cuando se utiliza CALLS, se sobreentiende que, debajo del número de argumento, se han introducido los argumentos con instrucciones PUSH.

La zona de la pila que usa el procedimiento queda por encima del FP (Punto de Región).

Las listas de argumentos también pueden enviarse a la pila, con el siguiente ahorro de memoria. Para formar las listas en la pila se usan las instrucciones:

PUSHL operando

Con un parámetro de la instrucción CALL se indica el número de valores enviados a la pila, siendo la ejecución de la propia CALL, quien se encarga de colocar, encima de la pila, el número de argumentos del procedimiento.

El Puntero de Argumentos (AP) señala el comienzo de la cadena de argumentos. Cuando la cadena contiene las direcciones de los operandos, se pueden recuperar éstos desde el procedimiento haciendo uso del modo indirecto de desplazamiento. Figura 10-39.

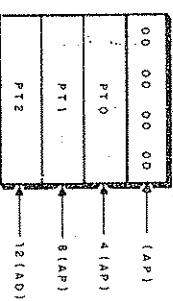


Fig. 10-39. Se puede acceder a los argumentos mediante el direccionamiento indirecto de desplazamiento. Así, si el operando 3 de la figura, se puede acceder con @I2(AP).

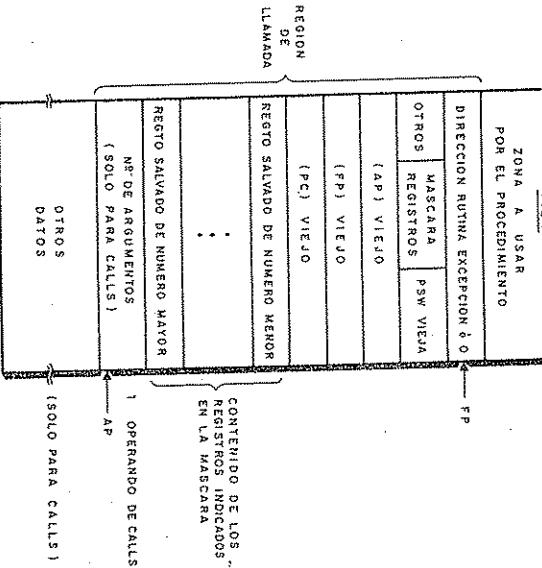


Fig. 10-40. La instrucción CALL salva en la pila los datos necesarios para poder regresar al programa desde el que se ha llamado al procedimiento. Obsérvense las peculiaridades de la instrucción CALLS.

Fig. 10-39. Se puede acceder a los argumentos mediante el direccionamiento indirecto de desplazamiento. Así, si el operando 3 de la figura, se puede acceder con @I2(AP).

Fig. 10-39. Se puede acceder a los argumentos mediante el direccionamiento indirecto de desplazamiento. Así, si el operando 3 de la figura, se puede acceder con @I2(AP).

La instrucción de retorno de procedimiento RET, hace desaparecer los datos incluidos encima del FP, o sea, que el SP baja hasta situarse en el FP. Seguidamente, se recuperan los PSW, AP, FP, PC y los registros salvados previamente. Además, RET elimina también la lista de argumentos si se ha usado la instrucción CALLS.

10

La última doble palabra que existe en la pila en la zona del procedimiento se refleja con ceros, pero puede hacerse con una dirección desde el procedimiento. Esta dirección se emplea para señalar una rutina que se ejecutará en caso de error de excepción.

Los resultados de un procedimiento se mandan al registro R0, que, si es insuficiente, se amplía con el R1.

Finalmente, en la figura 10-41 se muestran algunas instrucciones especiales del repertorio de los procesadores de la familia VAX-11.

INSTRUCCIONES DE ÍNDICE Y COLA	
INDEX	COMPUTA EL ÍNDICE
INSCUE	INSERTA ENTRADA EN COLA
INSPCHI	
INSPCHI	
INSPCHI	
RENUKE	TRAE ENTRADA DESDE LA COLA
REMCHI	
REMCHI	
REMCHI	
INSTRUCCIONES DE CONTROL PRIVILEGIADO	
SVPCTX	SALVA CONTEXTO DEL PROCESO
LOPCTX	CARGA CONTEXTO DEL PROCESO
MTPR	MUEVE AL REGISTRO PRIVILEGIADO
MFPTR	MUEVE DESDE REGISTRO PRIVILEGIADO
INSTRUCCIONES CON FUNCIONES ESPECIALES	
CRC	CHECKEO CÍCLICO DE REDUNDANCIA
BPT	FALLO DE PUNTO DE PARADA
XPC	LLAMADA A FUNCIÓN EXTERNADA
NOP	NO OPERA
HALT	PARO

Fig. 10-41. Instrucciones especiales del repertorio de los procesadores VAX-11.

10.17. AMPLIACION DE LA GAMA DE COMPUTADORES VAX

Al modelo VAX 8600, presentado en 1985, DEC ha añadido el VAX 8650, que

es un 44% más potente que el primero. Figura 10-42.

460 / Descripción de la arquitectura

10

Los sistemas 8600 y 8650 alcanzan una capacidad de memoria principal de 68 Mbytes. La memoria es ampliable con incrementos de 4 y 16 Mbytes. Un VAX 8650, con una configuración típica, puede direccionar directamente hasta 4.000 millones de bytes de memoria virtual.

En la construcción del 8650 se han utilizado nuevas técnicas de diseño para conseguir mejoras del rendimiento, tales como:

— Bus de memoria dedicado.

— Memoria cache con realimentación.

— Operación simultánea en cadena.

— Ciclo de UCP de 55 ns.

El espectacular aumento de la capacidad de memoria principal es el resultado de una nueva tecnología de *montaje superficial*, que permite empaquetar más chips de memoria en un área concreta. La circuitería de control de las tarjetas de memoria está montada en submódulos sujetos a la tarjeta principal. El nuevo subsistema de memoria permite colocar una placa de 16 Mbytes en el espacio requerido antes por dos tarjetas de 4 Mbytes, pudiéndose instalar hasta 4 placas de 16 Mbytes y una de 4 Mbytes, alcanzando un máximo de 68 Mbytes.

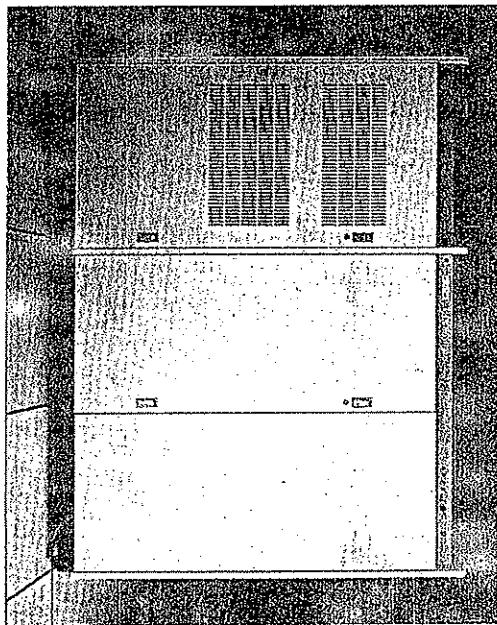


Fig. 10-42. Fotografía del computador VAX 8650. Cortesía de DEC.

Antes de 1987, Digital Equipment Corporation ya ofrecía en el mercado tres nuevos modelos de la familia VAX, incluyendo al más potente, hasta dicha época, el VAX 8800, que proporciona hasta 12 veces la potencia del VAX 11/780. Los otros dos nuevos modelos, VAX 8200 y VAX 8300, son sistemas de rango medio y ofrecen sustanciales mejoras en cuanto a precio y rendimiento, respecto al VAX 11/780.

VAX 8800

Diseñado para resolver las más complejas aplicaciones en organizaciones científicas de ingeniería y comerciales. Se aplica en áreas como Inteligencia Artificial, Diseño de circuitos integrados, y Modelos financieros.

VAX 8300

Proporciona hasta 1.9 veces la potencia del 8200 y está orientado a soportar aplicaciones que requieren alto nivel de cálculo en mercados comerciales y técnicos, donde el costo y el volumen son criterios importantes. Figura 10-43.

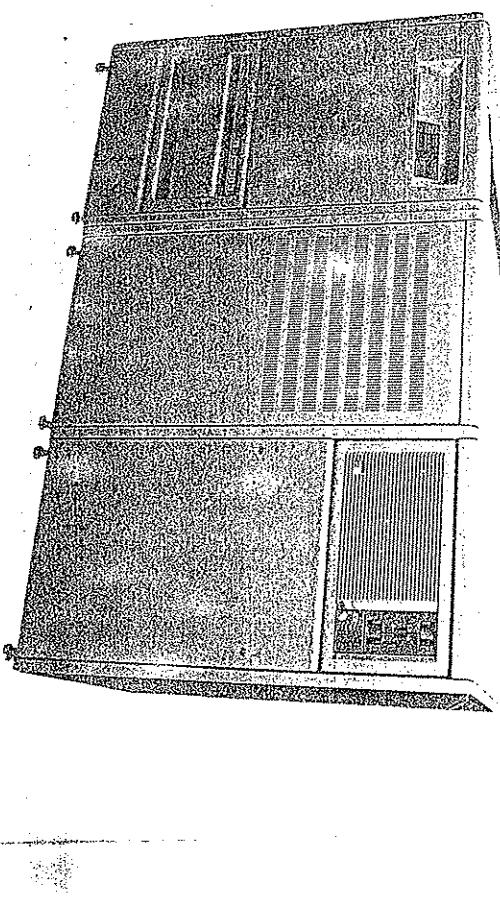


Fig. 10-43. Fotografía del computador VAX 8300. Cortesía de DEC.

VAX 8200

Tiene la potencia de un VAX 11/780 a un precio muy reducido. Es el sistema idóneo para aplicaciones de propósito general.

Estos tres nuevos modelos utilizan el mismo sistema operativo VMS que el resto de los computadores VAX, pudiendo así ejecutar más de 3.000 programas de gestión, de diseño y científicos.

En cuanto al VAX 8800, dispone de dos UCP acopladas que comparten 32 Mbytes de memoria principal. Cada UCP tiene 64 Kbytes de memoria cache y un acelerador de coma flotante.

El VAX 8800, con dos canales VAXBI de E/S y la máxima memoria, sólo ocupa 4,7 m², o sea, similar a los otros modelos de la familia. Para alcanzar estas características se han empleado más de 20 chips especiales y casi 200 matrices de puertas, con lo que se han integrado muchas más funciones que en otros modelos. Además, las tarjetas de circuito impreso están montadas con más niveles.

En el cuadro de la figura 10-44 se muestran las principales características técnicas del VAX 8800.

10.17.1. Implementaciones de la arquitectura VAX en tecnología VLSI

Existen diversas configuraciones de la arquitectura de los procesadores de la familia VAX construidas con tecnología VLSI. Mereden destacarse las tres siguientes:

1. Procesador VLSI "MicroVAX-32"

Admite la ejecución de un subconjunto de las instrucciones de la familia VAX con todos sus modos de direccionamiento. La memoria de control que contiene los microcódigos se ha reducido a 64 K palabras de 40 bits.

2. Configuración VAX VLSI

Se trata de un grupo de 9 circuitos integrados VLSI que soportan una arquitectura similar a la de la UCP del VAX 11/780.

10

3. "MicroVAX-1"

Utilizando diversos componentes MSI/LSI para la construcción del secuenciador de microcodigo y la unidad de búsqueda de instrucciones, completa el procesador un circuito VLSI que integra la ruta de datos.

EJERCICIOS

Ejercicio 10.1

¿Qué campos componen una instrucción en el VAX-11/750? ¿Cuál es la máxima longitud que puede tener una instrucción? ¿Y la mínima?

Ejercicio 10.2

Realizar un estudio comparativo de la memoria física y la memoria virtual en el procesador VAX-11/750 y el microprocesador 8086.

Ejercicio 10.3

Realizar un estudio comparativo de la arquitectura interna del 8086 y del 80386.

Ejercicio 10.4

¿Qué diferencias y semejanzas fundamentales existen entre el massbus y el unibus?

Ejercicio 10.5

Ventajas e inconvenientes de los modos de direccionamiento del VAX-11/750, frente a los del 8086/8088.

Ejercicio 10.6

Sobre el VAX-11/750 desarrollar las siguientes cuestiones:

- Diferencia de la escritura y lectura en el VAX-11/750 frente al microprocesador 8086/88.
- Manejo de memoria virtual. Dirección de tabla de páginas y mecanismos de traducción.
- Tipos de pilas utilizadas y forma de direccionarlas.
- Manejo de subrutinas en ensamblador. Contenido de la pila para cada salto a subrutina. Retorno de subrutina.

CARACTERÍSTICAS TÉCNICAS	
Arquitectura	Unidad de búsqueda de instrucciones completa integrada en el VLSI
Sistema Operativo	VMS Version 4.0
Tecnología	Logic ECL, DRAM, TTL de 28
Rendimiento relativo	Hasta 12 veces el VAX-11/750
Comunicación	Proporciona datos de 24 bits y 12 bits conectados a la memoria de intercambio.
Memoria principal	32 MB mediante comandos de memoria de intercambio.
Codificación	Un solo byte de dirección de memoria.
Bus CPU/Memoria	80MB/S. Unidireccional.
Buses de E/S	VAX-11/750/UNIBUS
Espacio	Unidad de VAX-11/780, 785, 800, 860.
Interconexiones	a) VMEbus b) Ethernet c) DECnet d) DMA
Programas de aplicación	Todas las aplicaciones basadas en MS-DOS.

Fig. 10-44. Principales características técnicas del VAX-8000. Cortesía de DEC.

464 / Descripción de la arquitectura

Ejercicio 10.7

- a)* Dada la siguiente palabra de estado (PSL):
00110011000000000000011110001001
especificar el estado en el que se queda la CPU
instrucción.
b) Si el contenido de los registros R6 y R8 es:

efectuar la instrucción ADDP2 R6, R8, indicando los cambios producidos en la PSL. ¿Se producirá alguna excepción?

R8: 533429011257643D

Edición 10.8

Coñece un programa en ensamblador que efectúa la transferencia de 10 palabras de memoria de la posición 000010F0 a la 000010A08, multiplicándolas previamente por el contenido del registro R8.

- D) Si el contenido de los registros R6 y R8 es:
R6: 820157642941027D
R8: 533429011257643D
efectar la instrucción ADDI12 R6, R8, in
ia PSL. ¿Se producirá alguna excepción?

456 / Descripción de la arquitectura

Computadores de altas prestaciones

11.1. EL REFORZAMIENTO DEL PARALELISMO Y EL AUMENTO DE LA VELOCIDAD

DE PROCESAMIENTO

La ampliación de las áreas de aplicación de los computadores a tiempos, que precisan efectuar un enorme número de operaciones sobre grandes masas de datos estructurados, como el procesamiento de imágenes en tiempo real, la meteorología, el cálculo y control de trayectorias de robots, el manejo de grandes bases de datos y la Inteligencia Artificial, entre otros, exigen la mejora constante de sus prestaciones. El incremento de la potencia de las máquinas programadas no sólo se consigue con la utilización de los últimos avances tecnológicos, sino, también, mejorando la arquitectura interna y los recursos del sistema lógico (lenguajes, técnicas de programación, etc.).

El reforzamiento del paralelismo y el aumento de la velocidad de procesamiento, están motivados por tres causas:

- a) Necesidad de un *incremento continuo de la potencia de cálculo*. Los monoprocesadores pueden alcanzar un máximo de potencia de 1 GFLOP (mil millones de operaciones en coma flotante por segundo), considerando las limitaciones de la velocidad de transmisión en el silicio. Dicha potencia es ya insuficiente para determinadas aplicaciones.

b) El *desfase entre la arquitectura clásica, propuesta por von Neumann, y los requerimientos de los nuevos sistemas lógicos y lenguajes*.

c) Las consideraciones que, respecto al *coste*, existían en el pasado, puesto que la tecnología actual VLSI proporciona elementos baratos especialmente cuando se usan de forma repetitiva, como es el caso de arquitecturas paralelas. Debe



esta forma, el aumento de la potencia del computador deja de ligarse al coste

del mismo mediante una función lineal.

Para elevar la velocidad de procesamiento de los computadores existen diversas alternativas, entre las que destacan:

- 1) Empleo de tecnologías más rápidas. Para poder disponer de máquinas que trabajan a más de 100 Megallops (millones de operaciones en coma flotante por segundo), se precisan memorias con tiempos de acceso de muy pocos ns.

Las tecnologías basadas en el Arseniuro de Galio, en el efecto Josephson y otras, consiguen retardos inferiores al ns, pero, teniendo en cuenta el tiempo imprescindible para circular las señales entre las distancias a las que se sitúan los componentes de la máquina, se llega a la conclusión de que no es suficiente con mejorar las características del equipo físico para elevar la velocidad continuamente.

- 2) Reducir el número de niveles de puertas para realizar algoritmos, como sucede en los sumadores con acarreo anticipado.

- 3) Aumentar la complejidad de los circuitos combinatorios. En este aspecto se puede hacer referencia al uso de PAL en el proceso de decodificación de las instrucciones.

- 4) Diseño de nuevas organizaciones y estrategias de funcionamiento de las memorias. Ejemplos de esta técnica son la memoria virtual, la memoria cache y la memoria entrelazada.

- 5) Sustitución de sistema lógico por equipo físico. Se tiende a construir mediante circuitería ciertas funciones repetitivas que hasta ahora se realizaban por sistema lógico.

- 6) Aumento del grado de paralelismo o concurrencia, desde el nivel de las instrucciones hasta el de los programas.

- 7) Utilización de estructuras segmentadas en las que, mientras se procesan unos datos, se capturan los próximos a procesarse, consiguiendo un mayor aprovechamiento de los componentes del sistema.

En este capítulo, se centra el tema de los supercomputadores, haciendo referencia, especialmente, a las alternativas sobre su estructura, que se basan, fundamentalmente, en potenciar el *procesamiento en paralelo* y propiciar la técnica de *segmentación*.

El paralelismo ha ido aumentando en la forma de operar de los sistemas; desde el *tratamiento por lotes* (batch), en donde los programas se agrupaban en lotes para irlos realizando uno tras otro, se ha llegado al *multiprocesamiento*, en el que se ejecutan varios programas simultáneamente, con el concurso de sistemas dotados de varios procesadores. Entre estas etapas límites, se han sucedido otras técnicas como la de *tiempo compartido*, en donde diversos procesos comparten la UCP, y la de *multiprogramación*, que ejecuta, de forma concurrente, varios programas residentes en la memoria principal.



La potenciación del paralelismo también se puede dirigir hacia el grado de concurrencia de los procesos, que puede alcanzar desde la ejecución simultánea de diferentes programas, hasta la que se consigue entre las diversas partes en que se descompone una instrucción.

En los sistemas *uniprocesadores* hay dos arquitecturas típicas dirigidas al *reforzamiento del paralelismo*:

- Pipeline o de segmentación.

- Matricial o array.

La arquitectura de segmentación favorece el encadenamiento del proceso y la superposición o solapamiento en la ejecución de las partes en que se descompone una instrucción. Los procesadores matriciales están dirigidos al uso sincronizado de múltiples unidades funcionales del tipo lógico-aritmético.

Los sistemas *multiprocesadores* enfatizan el nivel de paralelismo, funcionando asincrónicamente para ejecutar varios procesos al mismo tiempo.

Como resumen, puede decirse que los modernos supercomputadores disponen de tres recursos que explotan el paralelismo y que dan lugar a tres tipos de computadores:

1.) "De segmentación" o pipeline: Enfatizan la ejecución soñápa (en cadena) de instrucciones.

2.) "Matricial" o array: Explotan el paralelismo espacial mediante el empleo de diversas unidades aritmético-lógicas.

3.) "Multiprocesador": Admiten diferentes flujos de instrucciones que son tratados por un conjunto de procesadores que comparten los recursos principales (periféricos, memoria, base de datos, etc.).

Finalmente, existen computadores mixtos que combinan la tres técnicas de explotación del paralelismo, antes comentadas.

11.2. CLASIFICACIÓN DE LAS ARQUITECTURAS DE COMPUTADOR PROPUESTA POR FLYNN

Cuando se analizan las características de las arquitecturas típicas de computadores y se estudia el grado de paralelismo, es conveniente conocer la clasificación debida a Flynn (1966), basada en la forma de procesar los flujos de instrucciones y de datos. Dicha clasificación divide a los computadores atendiendo a su paralelismo explícito en 4 grupos:

SISD: Flujo único de instrucciones-flujo único de datos.

SMD: Flujo único de instrucciones-flujo múltiple de datos.

MISD: Flujo múltiple de instrucciones-flujo único de datos.

MIMD: Flujo múltiple de instrucciones-flujo múltiple de datos.

La arquitectura SISD, que esquemáticamente se representa en la figura 11-1, corresponde a la mayoría de los sistemas uniprocesadores actuales.

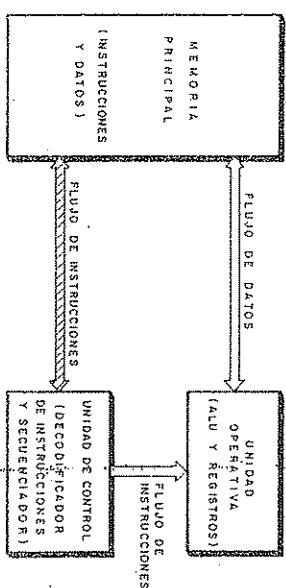


Fig. 11-1. Esquema simplificado de la arquitectura SISD.

En la figura 11-1, la memoria principal genera el flujo de instrucciones a la Unidad de Control, que, tras decodificarlas e interpretarlas, ordena su ejecución en la Unidad Operativa, compuesta por la ALU y los registros internos de trabajo. El flujo de datos bidireccional comunica la memoria principal y la Unidad Operativa. En el funcionamiento de los computadores SISD sólo hay una corriente de instrucciones unidireccional y otra de datos bidireccional.

En los computadores con arquitectura SIMD, la única Unidad de Control existe para supervisar el funcionamiento del conjunto de Unidades Operativas disponibles. La estructura corresponde a los llamados *procesadores matriciales*. Como se muestra en la figura 11-2, la Unidad de Control extrae e interpreta las instrucciones y envía las correspondientes señales de control a las Unidades Operativas encargadas de su ejecución. La Unidad de Control comienza la búsqueda de una nueva instrucción, nada más comenzada la ejecución de la anterior, siendo posible de esta forma, realizar varias instrucciones simultáneamente.

Cada Unidad Operativa de la figura 11-2 trabaja con un flujo de datos concreto, que se obtiene de una memoria compartida compuesta por varios módulos.

La estructura MISD consta de varias unidades de Control que reciben diferentes flujos de instrucciones que son ejecutadas en las correspondientes Unidades Operativas, las cuales se alimentan de un único flujo de datos. Figura 11-3. Un computador MISD se asemeja a una superestructura segmentada en la que cada procesador realiza una parte del procedimiento sobre el flujo de datos.

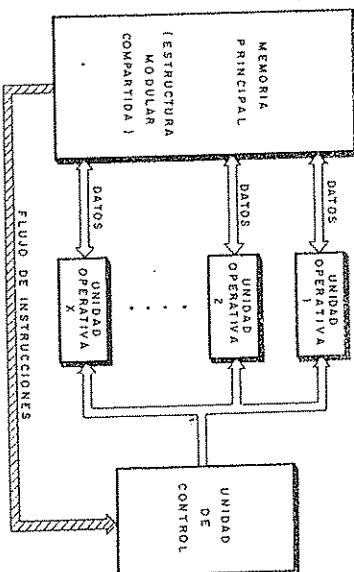


Fig. 11-2. Arquitectura SIMD de flujo único de instrucciones y flujos múltiples de datos.

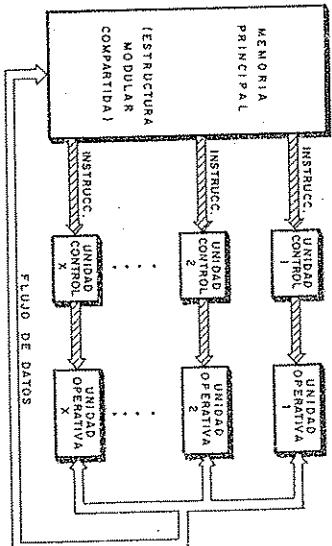


Fig. 11-3. Arquitectura MISD de múltiple flujo de instrucciones y flujo único de datos.

Finalmente, en la figura 11-4 se muestra un esquema general de la arquitectura MIMD con flujo múltiple de instrucciones y flujo múltiple de datos. Se trata de la estructura típica de un sistema multiprocesador, que puede definirse como un conjunto de procesadores serie, cuyas unidades operativas soportan un flujo de datos y las unidades de control, un flujo de instrucciones.

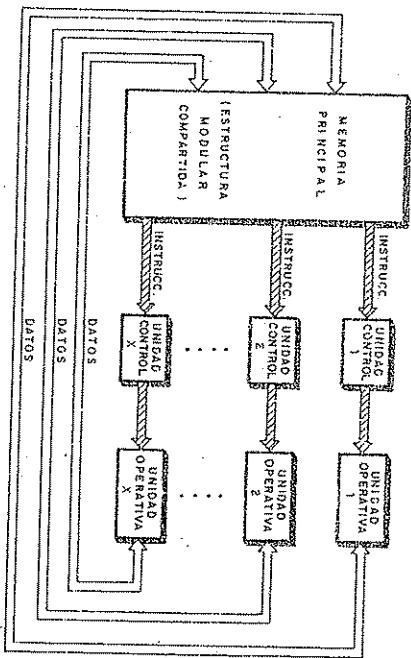


Fig. 11.4. Arquitectura MIMD de flujo múltiple de instrucciones y flujo múltiple de datos.

11.3. CLASIFICACION COMERCIAL DE LOS COMPUTADORES

Teniendo en cuenta los modelos existentes en el mercado, es decir, las realizaciones concretas, se pueden clasificar los computadores de la manera siguiente:

A) Procesadores segmentados

Dividen los procesos en subprocesos, con los que forman cadenas secuenciales de trabajo. Hay dos tipos fundamentales:

- 1) *Vectoriales*: Se emplean en el procesamiento de vectores "en serie". No es necesario terminar completamente el proceso de un elemento para iniciar el del siguiente.
- 2) *Array*: Son procesadores segmentados de dos dimensiones con varias cadenas de flujos de datos.

B) Procesadores de arquitectura SIMD

Dentro de este grupo destacan los siguientes tipos:

- 1) *Matriciales*: Formados por una matriz síncrona de procesadores paralelos.
- 2) *Computadores de altas prestaciones*

2) *Asociativos*: Son procesadores en los que la memoria de acceso es asociativa.

Se accede por contenido y no por dirección.

C) *Multiprocesadores*

Estos procesadores están formados por una serie de elementos de proceso de instrucciones, que se conectan con los módulos de memoria a través de una red. A este grupo de computadores MIMD pertenecen los *procesadores sistólicos*.

D) *Procesadores de la quinta generación*

Con las nuevas estructuras VLSI y los nuevos desarrollos del sistema lógico se desarrollan:

- 1) *Procesadores para tratamiento de bases de datos*.
- 2) *Procesadores de flujo de datos*. Carecen de flujo de control y de Contador de Programa. Las instrucciones se activan según la disponibilidad de los operandos.
- 3) *Procesadores inteligentes*. Procesan bases de conocimientos y disponen de un interfaz natural para el interfaz con el operador humano.

11.4. INTRODUCCION Y CONCEPTO DE LA TECNICA DE SEGMENTACION

Una de las técnicas empleadas para acelerar el funcionamiento de un sistema digital es la del *tratamiento en cadena* de la secuencia de operaciones que debe realizar. El concepto es muy parecido al que se aplica en el trabajo en cadena de la producción en serie, como sucede con las factorías de automóviles donde existen líneas de fabricación con células operativas específicas entre las que se reparten las tareas.

El tratamiento en cadena, al que se denomina técnicamente *segmentado o pipeline*, consiste en dividir la función F a realizar, en una serie de subfunciones ($F_1, F_2, F_3, \dots, F_n$), que se pueden ejecutar de forma independiente. Una unidad computadora, capaz de ejecutar toda una función F , sólo puede atender un proceso en cada instante como se representa en la figura 11.5.

Si se dispone de unidades individuales para procesar cada subfunción F_i , se puede configurar una cadena, que soporta el tratamiento simultáneo de tantos procesos como subfunciones existen. En la figura 11.6 se ofrece una representación gráfica de una cadena de 4 etapas, en la que los procesos circulan como en una tubería.



OPERADOR COMPLETO



Fig. 11-5. Un operador completo sólo atiende un proceso hasta que lo realiza. Los demás, esperan su turno.

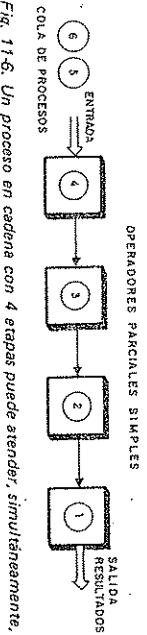


Fig. 11-6. Un proceso en cadena con 4 etapas puede atender, simultáneamente, 4 procesos.

Dado que las subfunciones F_i son más simples que la total F , se pueden efectuar en una fracción del tiempo T , que se tarda en ejecutar F . Si se parte de que los tiempos de las n subfunciones son iguales, cada una tendrá una duración de T/n . De esta manera, aunque el tiempo empleado en resolver cada proceso es T , el solapamiento a que da lugar esta técnica origina que en dicho tiempo se calculen n procesos.

En un computador, el flujo de instrucciones circula por una serie de unidades elementales en cadena, que realizan una operación simple, cada una, y cuyo conjunto completa la ejecución de las instrucciones. Dichas unidades fundamentales, en términos generales, tienen las siguientes funciones:

- 1) Búsqueda de la instrucción.

- 2) Decodificación.

- 3) Búsqueda de operandos.

- 4) Ejecución.

En la figura 11-7 se muestran las diversas unidades que se encargan de ir realizando las fases en que se descompone cada instrucción.

En la figura 11-8 se representa simbólicamente el proceso en cadena, seguido por la máquina programada, para realizar la secuencia de instrucciones.

Existe un reloj que genera impulsos de sincronismo de período t , que es el tiempo que dura cada fase. Si hay que procesar 5 instrucciones {1, 12, 13, 14 e 15}, en el primer impulso de reloj t_1 se efectúa la fase de búsqueda de la instrucción 11. Con el segundo impulso de reloj t_2 , la 11 pasa a la unidad encargada de la decodificación (FD), mientras que comienza el tratamiento de la primera fase de 12. Al cabo de 4 impulsos de reloj, se habrá completado la ejecución de 11; además, la 12 estará en la fase de búsqueda de los operandos, la 13 en la de decodificación y la 14 en la de búsqueda del código OP. Véase la figura 11-9.

4.74 / Computadores de altas prestaciones

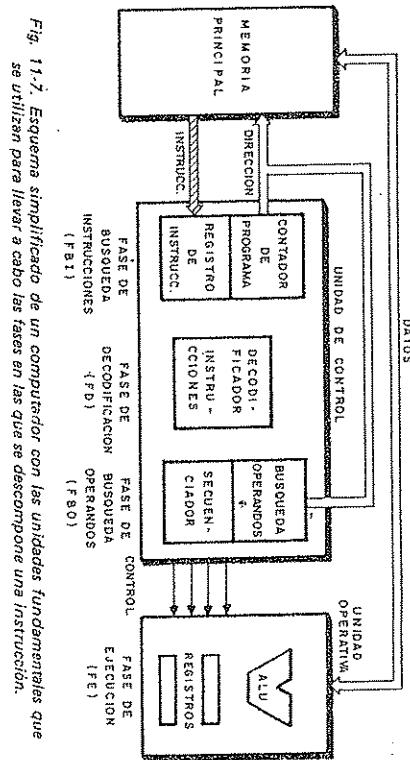


Fig. 11-7. Esquema simplificado de un computador con las unidades fundamentales que se utilizan para llevar a cabo las fases en las que se descompone una instrucción.

Para poder establecer el tratamiento segmentado de una función general F , se deben cumplir las cinco condiciones siguientes:

1. La evaluación de la función básica F debe poder descomponerse en la evaluación secuencial de una serie de subfunciones F_1, F_2, \dots, F_n .
2. Las entradas de cada subfunción F_i deben quedar determinadas exclusivamente por las salidas de la subfunción anterior en la secuencia de evaluación, es decir, la F_{i-1} .
3. No debe existir interrelación cruzada entre subfunciones, las cuales sólo deben comunicarse de salida de F_i a la entrada de F_{i+1} .
4. Cada subfunción debe ser calculable mediante un circuito específico y de forma más rápida que toda la función F . Cada uno de estos circuitos formará una etapa de la cadena.
5. Los tiempos requeridos para el cálculo de cada subfunción deben ser parecidos.

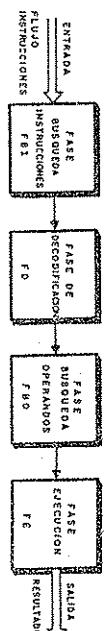


Fig. 11-8. Procesamiento segmentado o en cadena de la secuencia de instrucciones.

Esta última condición es importante desde el punto de vista de la utilidad de la cadena. Como se desprende de la figura 11.9, el tiempo de cálculo asignado a todas las subfunciones debe ser el mismo e igual al de la etapa que requiere la máxima duración. Esto es debido a que todos los procesos deben cambiar de subfunción de manera simultánea. Por todo ello, si los tiempos de las etapas son dispares, se desaprovecha la velocidad de trabajo de las que son rápidas.

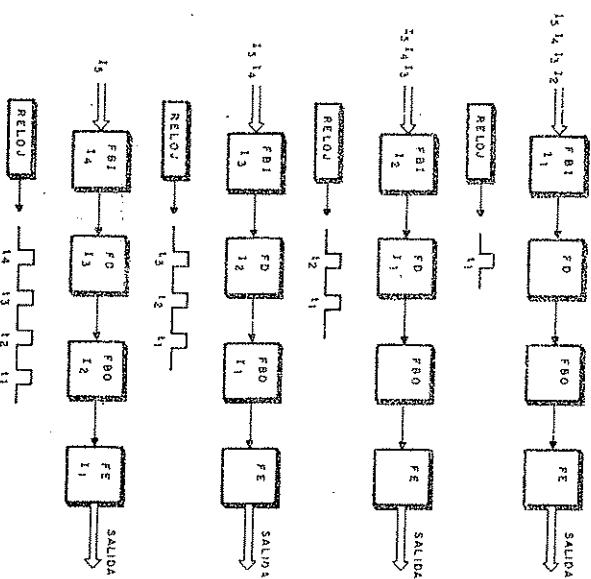


Fig. 11.9. Procesamiento de las instrucciones a través de los bloques funcionales.

La técnica de segmentación es muy apropiada en los computadores y puede aplicarse a sus tres grandes bloques: Unidad de Control, Unidad Lógico-Aritmética y memoria principal.

Como ya se ha indicado, la Unidad de Control es una buena candidata para el tratamiento en cadena, puesto que la ejecución de las instrucciones se divide en una serie de etapas.

Las unidades operativas complejas, como las que manejan datos en coma flotante, también se adaptan al tratamiento segmentado. Las distintas fases en las que se

dividen estas operaciones se pueden realizar por circuitos específicos, que configuran una cadena.

En el caso de la memoria principal, el concepto de cadena que se aplica es ligeramente distinto, puesto que su funcionamiento se basa en un conjunto de accesos independientes en módulos de memoria distintos, no existiendo interrelación ni comunicación entre dichos módulos.

11.4.1. Estructura y tipos de cadenas

Las cadenas suelen ser de funcionamiento síncrono y utilizan unos registros intermedios entre las distintas etapas que las componen. Cada registro almacena la información de salida de una etapa y de entrada a la siguiente; así la información de un proceso determinado va pasando registro a registro, sufriendo la transformación correspondiente a la subfunción de cada etapa. Todos los componentes están gobernados mediante un único reloj, cuyos flancos hacen que todas las informaciones avancen una posición simultáneamente. Figura 11.10.

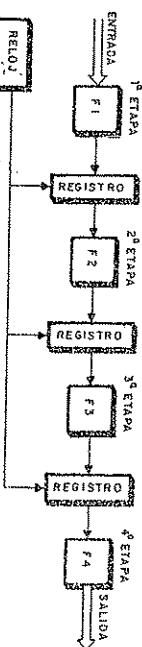


Fig. 11.10. Estructura típica de las cadenas.

De acuerdo con la filosofía de diseño empleada, las cadenas se clasifican en distintos tipos, entre los que destacan:

Cadena unifunción: Es una cadena diseñada para realizar una sola función.

Cadena multifunción: Permite realizar varias funciones, por lo que, además de la entrada de los procesos, debe tener unas entradas de control, que definan la función a realizar en cada situación, dependiendo del sistema con el que se pueden reconfigurar para realizar las distintas funciones, se denominan *dinámicamente reconfigurables* y *estáticamente reconfigurables*.

Las cadenas estáticamente reconfigurables están diseñadas para ser reconfiguradas en muy pocas ocasiones. La cadena actúa como si fuese unifunción entre cada cambio y trata todo un lote de procesos sin alterar su funcionamiento. Las cadenas dinámicamente reconfigurables pueden reconfigurarse

para cada entrada y se emplean en las Unidades de Control, puesto que cada instrucción requiere un tratamiento ligeramente diferente a las demás.

Cadena lineal: Se llama así a la cadena en la que a cada etapa sólo le sigue otra.

Cadena no lineal: Detrás de cada etapa hay diversos caminos e, incluso, realimentaciones para formar bucles.

11.4.2. Parones y choques en la cadena

Los parones o hazards representan, posiblemente, el mayor problema del tratamiento en cadena. Surgen cuando se produce alguna causa que impide se sigan introduciendo elementos en la cadena, quedando ésta vacía.

Las instrucciones de bifurcación son un ejemplo clásico de parón en Unidades de Control con tratamiento en cadena. Si existe una instrucción de bifurcación condicional sobre la operación realizada por la instrucción anterior, la instrucción que sigue a la de bifurcación no puede comenzar a procesarse hasta que quede especificada la alternativa de bifurcación. Este hecho provoca un 'hueco' en la cadena.

Los huecos que introducen los parones en la cadena hacen que no se pueda aprovechar al máximo su capacidad de proceso y que, por tanto, sus prestaciones reales sean inferiores.

Cuando los parones se deben a que distintos procesos desean utilizar la misma etapa a la vez, reciben el nombre de *choques*. Esto sucede, por ejemplo, cuando se desean realizar dos o más accesos sobre el mismo elemento de memoria.

11.4.3. Memorias entrelazadas

La velocidad de las memorias principales es bastante inferior que la de la lógica empleada en las Unidades de Control y en las Aritmético-Lógicas. Para solucionar este problema, se divide la memoria principal en varios módulos autónomos. Cada módulo funciona independientemente, por lo que se puede acceder al mismo tiempo a tantas posiciones de memoria principal como módulos tenga.

Esta organización modular de la memoria recibe el nombre de *entrelazada*, porque permite ir entretejiendo o entrelazando los accesos entre los diversos módulos que constituyen la memoria principal.

Se da el nombre de *entrelazado simple*, cuando se puede acceder a todos los módulos de memoria al mismo tiempo. En la figura 11-11 se aprecia cómo se accede, simultáneamente, a todos los módulos de la memoria, empleando la misma dirección. La información accedida de cada módulo se almacena en su registro.

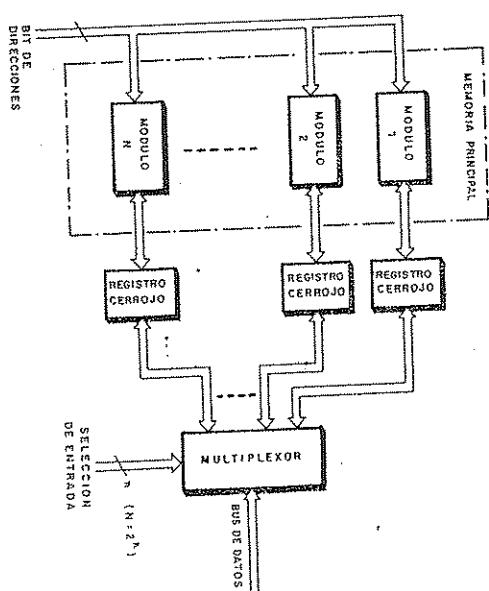


Fig. 11-11. Estructura de la memoria principal para configurar el entrelazado simple.

cerrojo correspondiente, de donde se van extrayendo mientras se realiza un nuevo acceso a todos los módulos. Los registros permiten simultaneamente los accesos con la distribución de las palabras leídas anteriormente.

Cuando se realiza el primer acceso a la memoria entrelazada de la figura 11-11, se obtienen N palabras, una de cada módulo. En los siguientes accesos se van obteniendo el mismo número de informaciones simultáneas, tal como se muestra en la figura 11-12.

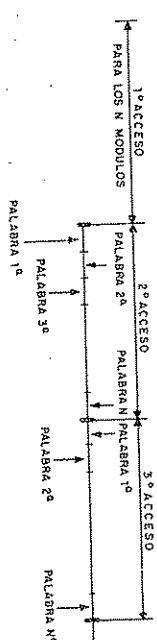


Fig. 11-12. En cada acceso a una memoria con entrelazado simple de N módulos, se obtienen N palabras.

El entrelazado simple funciona de forma satisfactoria en el caso de accesos secuenciales, como ocurre en la ejecución de instrucciones o en el tratamiento de vectores. Sin embargo, esta estructura no es demasiado práctica para el caso de accesos no secuenciales, como el tratamiento de bucles y de bifurcaciones o de datos no ordenados secuencialmente. En estos casos es más frecuente el uso de entrelazado complejo, que se describe a continuación:

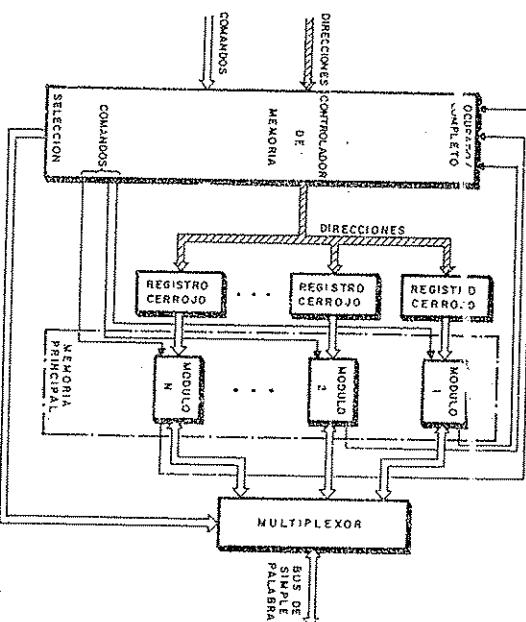


Fig. 11-13. Organización de una memoria principal con entrelazado complejo.

Se dice que el entrelazado es complejo cuando los accesos están solapados en el tiempo, tal y como se presenta en la figura 11-13. Los registros almacenan la dirección a la que se desea acceder en cada módulo, por lo que su distribución no tiene por qué ser secuencial. Si se divide el tiempo de acceso de los módulos n^{mo} y $n+1^{\text{mo}}$, se obtiene el tiempo que cada uno de ellos puede disponer del bus de direcciones para recibir su dirección y del bus de datos para enviar o recibir su información. Los comienzos de los accesos a estos módulos se solapan en el tiempo, como se muestra en la figura 11-14, de forma que los tiempos de ocupación de los buses por cada módulo no se superpongan. Este tipo de entrelazado exige que la memoria disponga de una unidad de distribución de accesos que se encargue de enviar a cada módulo las peticiones de accesos en el momento pertinente y que resuelva los choques, es

dicho, que haga esperar aquellos accesos que se vayan a realizar sobre módulos ocupados. Sin embargo, esta mayor complejidad se ve compensada por unas buenas prestaciones en los accesos no secuenciales.

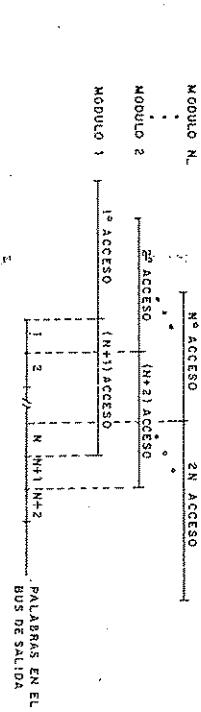


Fig. 11-14. Los comienzos de los accesos se solapan en el tiempo en la memoria entrelazada compleja para evitar que los tiempos de ocupación de cada módulo no se superpongan.

El rendimiento óptimo de la memoria entrelazada se obtiene cuando los accesos que se solicitan a la memoria principal, en cada ciclo, corresponden a módulos distintos. Sin embargo, con frecuencia surgen peticiones simultáneas sobre el mismo módulo, en cuyo caso, aparece un problema de choques, teniendo que atender a una de las peticiones y hacer esperar a las demás.

La forma en que se ordenan las peticiones sucesivas de memoria tiene una influencia importante sobre la cantidad de choques. Se denomina entrelazado de orden inferior cuándo las posiciones sucesivas de memoria principal, 0, 1, 2, 3, 4, ... se asignan a módulos consecutivos. Si, por ejemplo, la memoria tiene 4 módulos, el primer módulo estaría ocupado por las posiciones de memoria 0, 4, 8, 12, ..., mientras que el segundo ocuparía las posiciones 1, 5, 9, 13, etc. Por el contrario, si empleando módulos de 2^n posiciones, se asignan las 2^n primeras posiciones al primer módulo, las posiciones de la 2^n a la $2^{2n} - 1$ al segundo módulo y así sucesivamente, se dice que el entrelazado es de orden superior.

11.4.4. Los parones en la secuencia de instrucciones

El problema de los parones en la secuencia de instrucciones ocurre con las bifurcaciones condicionales, con las interrupciones y con las dependencias entre datos.

Los parones de dependencia entre datos se producen cuando una instrucción requiere como dato un resultado que debe generar una instrucción anterior. La figura 11-15 muestra cómo la necesidad del dato A, que se genera en la instrucción de suma (ADD), obliga a esperar a la instrucción de incremento, originando un hueco.



Fig. 11-15. Parón originado por la dependencia de datos entre una instrucción y la siguiente.

Los parones por bifurcación condicional se producen al reconocerse la dirección de bifurcación hasta que la instrucción de bifurcación pueda tratar la condición que, normalmente, genera la instrucción anterior. La figura 11-16 presenta este caso. Obsérvese, que la instrucción de bifurcación B2 introduce un hueco detrás de ella, produciendo el consiguiente retraso en la ejecución de la cadena.

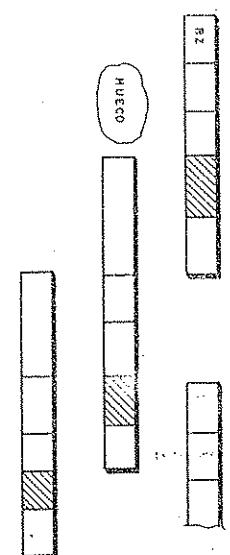


Fig. 11-16. El parón por bifurcación condicional se produce si no conoce la dirección de bifurcación hasta que no se averigüe la condición que, normalmente, genera la instrucción anterior.

Para evitar el problema de los huecos y retardos introducidos por las bifurcaciones condicionales, se pueden emplear las técnicas de *cadena de predicción* y la de *bifurcación retardada*, que describiremos a continuación.

Cadena de predicción

Se evita el hueco que causa el parón tomando a priori una de las dos alternativas de bifurcación. En el momento que se conoce la condición, si ésta corresponde con la alternativa supuesta, se sigue el procesamiento, con el consiguiente ahorro de tiempo. Si, por el contrario, la condición es distinta a la tomada a priori, se desecha el proceso realizado desde la instrucción de bifurcación y se inicia la rama real, perdiendo el tiempo empleado en esta rama.

El gran inconveniente de esta técnica es que el computador tiene que disponer de una serie de registros adicionales para almacenar todo el contenido de la máquina en el momento de la bifurcación y poder así volver hacia atrás cuando se falle en la predicción.

Bifurcación retardada

Otra técnica, más económica, para aprovechar el hueco que dejan las bifurcaciones condicionales, consiste en rellenarlo con instrucciones anteriores a la bifurcación y que, por tanto, deben ejecutarse independientemente del camino seguido por ésta. El mayor obstáculo de este procedimiento consiste en que no siempre se pueden retardar instrucciones para rellenar el hueco y en el esfuerzo adicional de programación que ello supone.

Se transfiere el aprovechamiento de los huecos al programador y al compilador, pero su gran ventaja es que no requiere dispositivo adicional alguno de equipo físico.

11.4.5. La segmentación en la unidad de control microprogramada

Las Unidades de Control microprogramadas suelen emplear el concepto de encadenamiento o segmentación para acelerar su funcionamiento. Si se analizan las subfunciones que comprenden el ciclo básico de ejecución de las microinstrucciones, se puede establecer la siguiente división:

1. Cálculo de la siguiente microdirección. Al tiempo empleado en realizar esta subfunción, se le llama TCD.
2. Acceso a la memoria de control, cuyo tiempo recibe el nombre simplificado de TMC.
3. Activación de los órganos del computador con las señales obtenidas de la memoria de control. Este tiempo se llama TOC.

La existencia de los registros apropiados en la UCP permite el almacenamiento de la dirección de la microinstrucción y, también, de la palabra de control, pudiendo así soportar un tratamiento segmentado.

El *tratamiento en serie*, mostrado en la figura 11-17, se produce cuando no se dispone de registro de almacenamiento intermedio.

El *tratamiento en cadena con secuenciamento explícito* se origina cuando hay dos etapas en la Unidad de Control, puesto que el registro de microinstrucción permite simultaneamente el acceso a la memoria de control con el uso de las señales cargadas anteriormente en dicho registro. El tiempo TCD se reduce al retraso del multiplexor, por lo que en la figura 11-18 se ha considerado muy pequeño, pero en serie con el TMC, puesto que no hay un registro más.



El tercer método es bastante usado, puesto que no supone coste de equipo, aunque exige un mayor esfuerzo en la microcodificación.

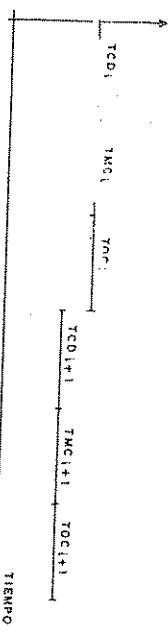


Fig. 11-17. Cuando no hay registros intermedios de almacenamiento en la microinstrucción, el tratamiento de las subfunciones de la microinstrucción se realiza en serie.

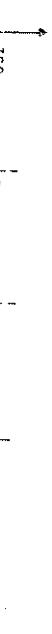


Fig. 11-18. Solape en el tratamiento de las microinstrucciones en cadena con secuenciamiento explícita.

En la figura 11-18 se ha supuesto que la suma $TCD + TMC$ es menor que TOC , tiempo de ejecución de las microinstrucciones.

Finalmente, en la figura 11-19 se presenta, gráficamente, el funcionamiento de la *Unidad de Control microprogramada trabajando en cadena con secuenciamiento implícito*. En este caso, la cadena tiene tres etapas, ya que existen dos registros intermedios, uno para las microinstrucciones y otro para las direcciones, lo que permite simultáneamente las tres bifurcaciones de la microinstrucción. Dado que en la figura 11-19 se ha supuesto que $TCD + TMC < TOC$, esta solución no reduce el tiempo total frente a la anterior.

La secuencia en cadena de las microinstrucciones presenta el problema de los parones debidos a las microbifurcaciones condicionales. Estas situaciones se pueden resolver por cualquiera de las tres alternativas siguientes:

- Espera.
- Predicción.
- Microbifurcación retardada.

484 / Computadores de altas prestaciones

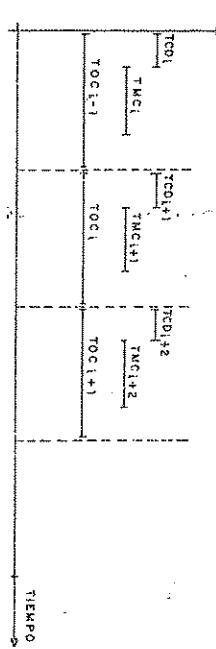


Fig. 11-19. Solape producido en el tratamiento de las microinstrucciones en cadena con secuenciamiento implícito.

En la figura 11-20 se muestra el esquema de una Unidad de Control microprogramada con cadena de dos etapas y predicción. Los elementos adicionales para realizar la marcha atrás, en caso necesario, son dos: el detector de secuencia errónea y el multiplexor de microinstrucciones.

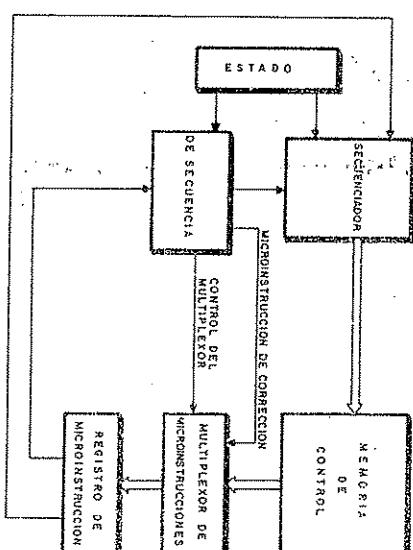


Fig. 11-20. Esquema de una Unidad de Control microprogramada con cadena de predicción.



11.5. COMPUTADORES VECTORIALES

Un operando vectorial o vector está formado por una lista ordenada de n elementos. Cada elemento es una cantidad escalar que puede ser: un número en coma flotante, un entero, un valor lógico o un carácter. Las operaciones vectoriales pueden ser de cuatro tipos:

$$1) f_a: V \rightarrow E$$

$$2) f_b: V \rightarrow V$$

$$3) f_c: V \times V \rightarrow V$$

$$4) f_d: V \times E \rightarrow V,$$

siendo V un operando vectorial y E un operando escalar.

El primer tipo de operaciones vectoriales lo componen aquéllas en las que, operando con los elementos de un vector, se obtiene un resultado escalar. Es el caso, por ejemplo, de hallar la suma de los elementos del vector.

Dentro del segundo tipo, f_b , se encuentran las operaciones que afectan independientemente a cada elemento del vector, formándose un vector resultado, donde cada elemento es el resultado de aplicar la operación al elemento correspondiente del primer vector. Por ejemplo, son operaciones de este tipo, las de raíz cuadrada del vector, en las que a cada elemento se aplica la raíz cuadrada.

Las operaciones del tercer tipo, f_c , son las que obtienen un vector a partir de otros dos. La suma de vectores es una operación que corresponde a este tipo de operación.

En las operaciones del tipo f_d se opera aplicando un escalar a cada elemento, para obtener un vector resultado. Una operación de este tipo es la suma de un escalar con un vector.

Las *instrucciones vectoriales* están disponibles en el lenguaje máquina de los procesadores vectoriales y se especifican mediante los siguientes campos:

1. *Código de operación*. Se encarga de seleccionar la unidad funcional o reconfigurar una unidad multifuncional de manera que ejecute la operación indicada.
2. *Direcciones base*. Son las direcciones de comienzo de los vectores operandos y de resultado.
3. *Incremento de dirección*. Incremento entre los elementos de los vectores.
4. *Desplazamiento*. Sumándolo a la base se obtiene la dirección del elemento buscado dentro de un vector.
5. *Largo del vector*.

La técnica de segmentación es muy aplicada a instrucciones vectoriales puesto que, mientras con procesadores escalares convencionales, cada elemento debe ejecutarse completamente para dejar paso al siguiente, en la técnica de segmentación para empezar a ejecutarse un elemento, basta con que esté libre el primer paso del proceso completo.

El gráfico de la figura 11.2.1 muestra el grado de paralelismo, según las siguientes formas de programación:

— Algoritmo paralelo (AP).

— Lenguaje de Alto Nivel (LAN).

— Código objeto (CO).

— Código máquina (CM).

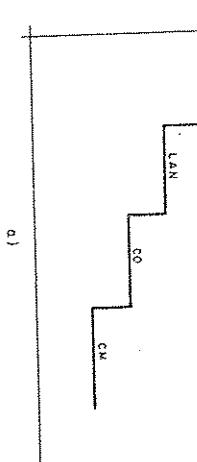
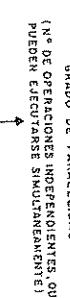


Fig. 11.2.1. a) Grados de paralelismo empleando un LAN especializado para procesos paralelos. b) Grado de paralelismo empleando un LAN convencional y un compilador de vectorización.

Existen grandes computadores que disponen de procesadores vectoriales con múltiples unidades de ejecución o pipes, según se refleja en la figura 11-22.

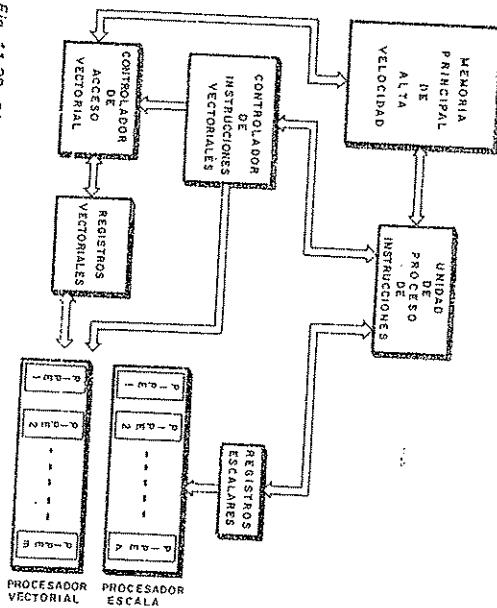


Fig. 11-22. Diagrama de un procesador vectorial con múltiples unidades de ejecución o pipes.

La Unidad de Proceso de Instrucciones de la figura 11-22, recoge las instrucciones y las decodifica, trasladándolas a la sección de instrucciones vectoriales o escalares, cada una de las cuales consta de un procesador multi-pipe.

11.5.1. Arquitectura del "Cray-1"

El Cray-1 es uno de los procesadores vectoriales más modernos, cuya arquitectura se describe seguidamente. Emplea tecnología ECL de 0,7 ns y MOS de 70 ns. Funciona a una frecuencia de reloj de 12,5 ns y no puede trabajar de forma autónoma. Se necesita un computador principal o host para que dirija el sistema, tal como se muestra en la figura 11-23.

La sección de memoria se organiza en 8 ó 16 bancos con 72 módulos cada uno.

La memoria principal está configurada por circuitos integrados, tipo RAM, de tecnología bipolar, que conforman un mínimo de un millón de palabras de 72 bits, 8 de las cuales se utilizan para la detección de errores dobles y corrección de los

488/ Computadores de altas prestaciones

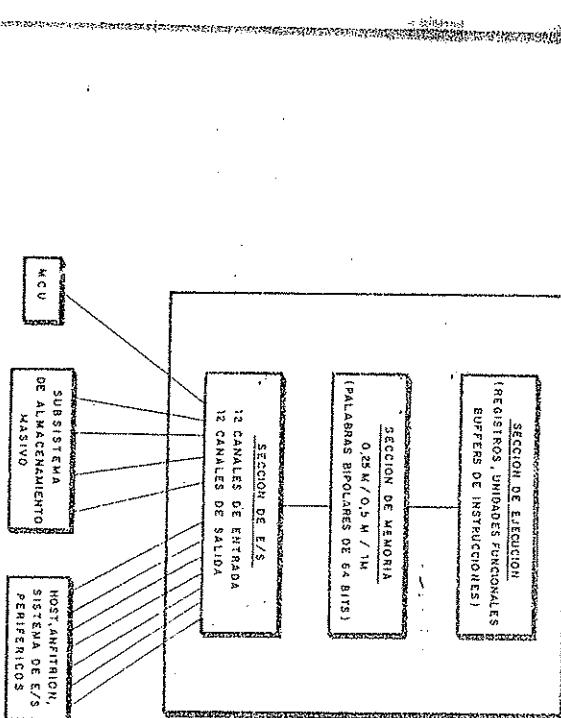


Fig. 11-23. Diagrama general de conexiónado de un Cray-1 a un computador principal o host.

simples. La memoria tiene un ciclo de 50 ns, o sea, 4 períodos de reloj. Se pueden transferir a la Unidad de Ejecución de la figura 11-23 una, dos o cuatro palabras por período de reloj.

La sección de E/S contiene 12 canales de entrada y otros 12 de salida. Cada canal tiene una frecuencia de transferencia de 80 Mbytes/s como máximo.

La Unidad de Control de Mantenimiento (MCU) se encarga de la inicialización del sistema y la supervisión de su funcionamiento.

En la figura 11-24 se representa detalladamente la sección de ejecución. Contiene 64 X 4 buffers de instrucciones y 800 registros con diferentes funciones. Tiene 12 unidades funcionales agrupadas en unidades vectoriales, de coma flotante, de tipo escalar y de direcciones, tal como se indica en la tabla de la figura 11-25. Las operaciones aritméticas pueden efectuarse con datos de 24 bits, mientras que las de coma flotante trabajan con datos de 64 bits.



REGISTROS VECTORIALES
MEMORIA
INSTRUCCIONES
DESEPLAZAMIENTO
LOGICA
SUMA
UNIDADES FUNCIONALES VECTORIALES

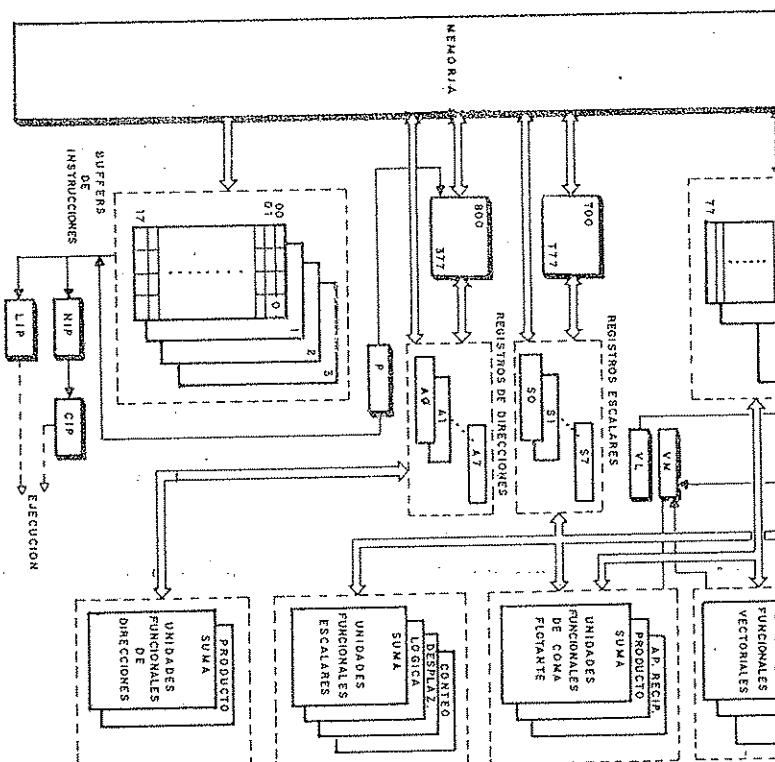


Fig. 11-24. Esquema general interno de la Unidad de Ejecución del computador Cray-1.

Los registros escalares Y de direcciones pueden acceder directamente a la memoria a través de los 64 registros T y B, respectivamente. Estos registros (T, S, B y A) son de 24 bits, y los registros T y B contienen los datos escalares y de direcciones más usados.

490 / Computadores de altas prestaciones

El registro CIP contiene la instrucción en curso, mientras que el registro NIP almacena la próxima instrucción a ejecutar. Ambos registros constan de 16 bits. Si la instrucción es de 32 bits, se utiliza también el registro LIP para contener los 16 bits de menor peso de la instrucción.

En instrucciones de bifurcación se emplea el registro P para contener la dirección de bifurcación.

En la figura 11-25 se presentan las diversas unidades funcionales del Cray-1.

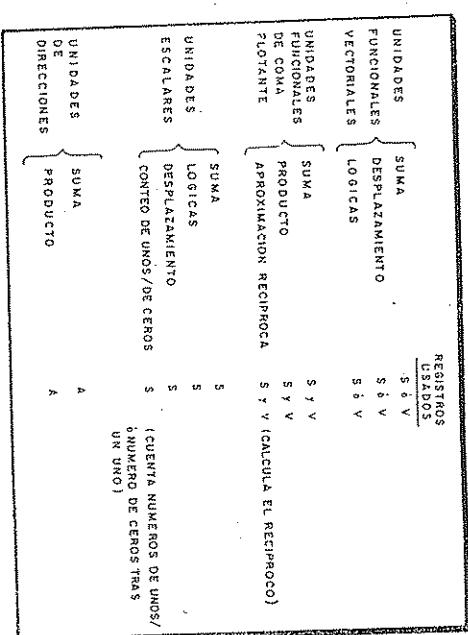


Fig. 11-25. Unidades funcionales del Cray-1.

11.6. COMPUTADORES "ARRAY"

Los procesadores segmentados array tienen una estructura de dos dimensiones, con múltiples cadenas de flujo de datos, para operaciones aritméticas de alto nivel, tales como producto de matrices, cálculo de la matriz inversa, etc. La arquitectura segmentada se basa en tablas celulares de unidades aritméticas.

En la figura 11-26 se presenta un ejemplo de diseño de un procesador pipeline array. Este array tiene una estructura pipeline de tres direcciones de flujo de datos, para la multiplicación de dos matrices. Cada célula P efectúa una operación de producto interno aditivo.



pero habiendo entre los dos únicos registros que facilitan el interfaz. Estos registros tienen las siguientes funciones:

- Registro de funciones, que sirve para los comandos típicos, tales como *start*, *stop*, etc.
- Registro de comutadores, que se utiliza para enviar datos de control, parámetros o direcciones del principal al AP-120B.
- Registros display o presentadores, que visualizan los contenidos de los registros del procesador array.
- Dirección de memoria del principal.
- Dirección de memoria del AP-120B.
- Recuento de palabras.

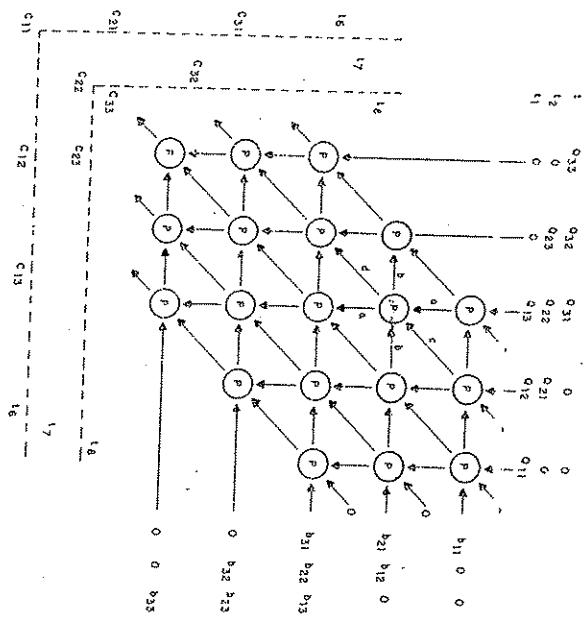


Fig. 11-26. Procesador segmentado array utilizado para el producto de matrices.

Los registros de los terminales de entrada y de salida están sincronizados por el mismo reloj. El array de la figura 11-26 efectúa el producto de dos matrices de orden 3 X 3:

$$A \cdot B = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} \cdot \begin{vmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{vmatrix} = \begin{vmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{vmatrix} = C$$

Las matrices de entrada son introducidas al procesador por las direcciones horizontales y verticales. Para multiplicar dos matrices de orden n X n se requieren 3 · n² - 4 · n + 2 celdas P, y, para completar el proceso, se requieren 3 · n - 1 períodos de reloj.

11.6.1. Arquitectura del computador AP-120B

La conexión de un AP-120B con un computador principal está representada en la figura 11-27, en la que el AP-120B parece como si se tratase de un terminal más, de reloj.

492 / Computadores de altas prestaciones

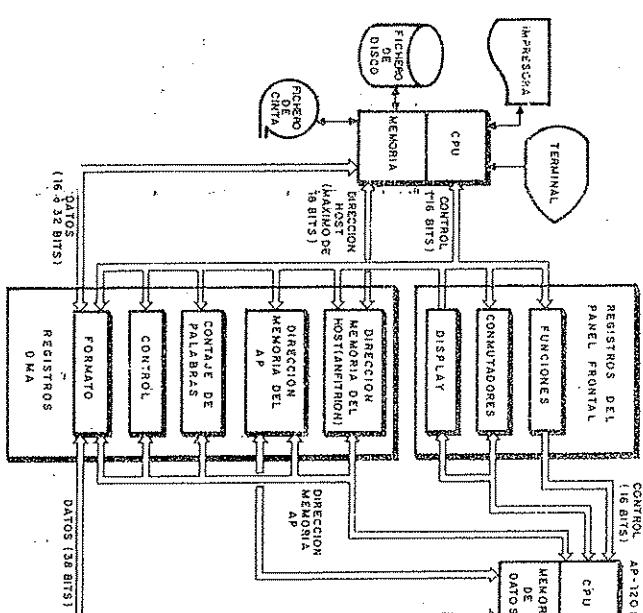


Fig. 11-27. Interconexión del computador AP-120B al principal o host.

— Registros de control.

- Registros de formato, que convierten el formato F_LP del computador principal del AP-120B y viceversa.

En la figura 11-28 se muestra el diagrama funcional del procesador AP-120B. Está formado por seis secciones: memoria de control, unidad de control, sección de memoria, sección de E/S, bus de 38 bits y unidades aritméticas. Dentro de estas secciones se incluyen a los bloques siguientes:

Memoria de control

- Memoria de programa (PM): Contiene las instrucciones a ejecutar por la Unidad de Control. Tiene un máximo de 4 K palabras, pudiéndose hacer ampliaciones de 256 palabras. Cada palabra es de 64 bits.

Sección de memoria

- Memoria principal de datos (MD): Constituye el almacenamiento principal de datos con palabras de 38 bits. Se pueden hacer ampliaciones de 2 u 8 K palabras, hasta un máximo de un millón.
- Memoria de Tablas (TM): para guardar constantes muy utilizadas. Puede ser de tipo RAM o ROM.
- Tablas de registros X, Y, DPX y DPY: Forman dos bloques acumuladores de 38 bits. Cada bloque contiene 16 acumuladores, pudiendo ser accedido por el AP-120B directamente.

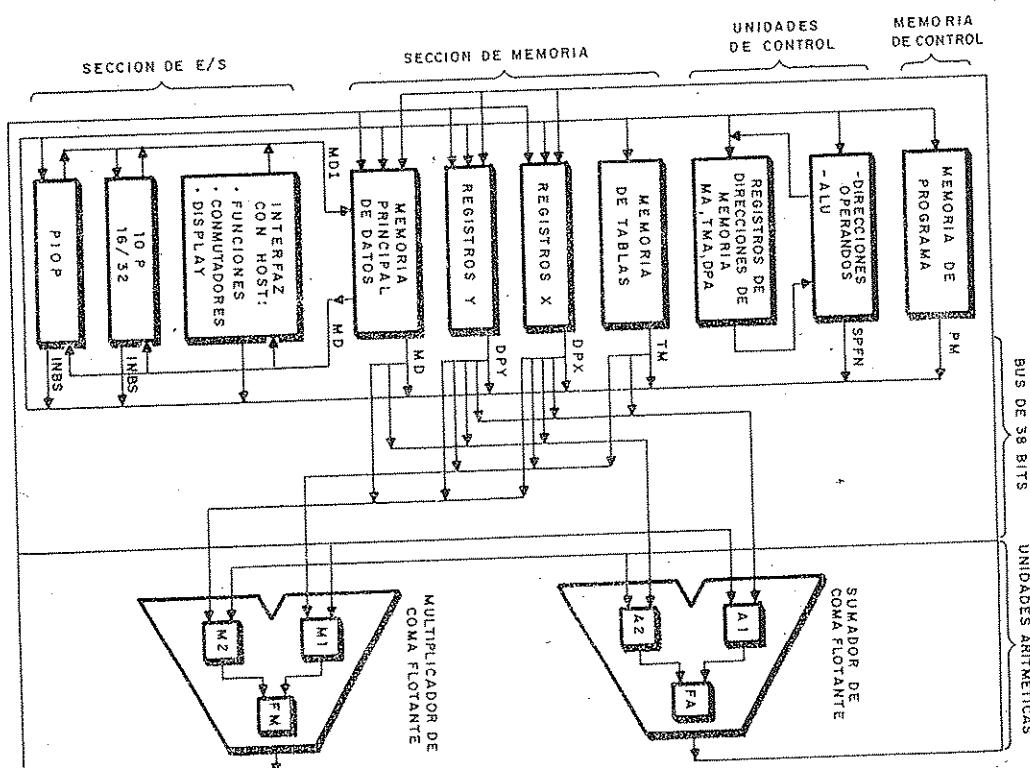
Unidad de Control

- Direcciones de operandos y ALU (SPFN): Contiene 16 registros con las direcciones de los operandos para la ALU, la cual efectúa operaciones aritméticas con enteros de 16 bits.
- Registros de direcciones (MA, TMA, DPA): La ALU daña las direcciones de los resultados en uno de estos tres registros, según sea la dirección de memoria, de memoria de tablas o de registros X e Y.

Sección de E/S

- Contiene los registros funciones, comutadores Y display. Además, contiene otros dos bloques IOP y PIO, para entradas y salidas.

Fig. 11-28. Diagrama funcional del computador AP-120B.





Unidades aritméticas

- Sumador de coma flotante (FA): Suma dos números en coma flotante, A1 y A2, que pueden provenir de diferentes registros, como se refleja en la figura 11-29, dejando el resultado en uno de los registros señalados en la misma figura.

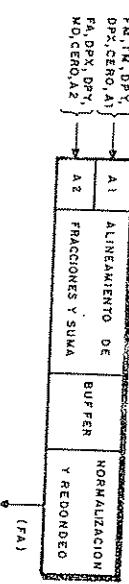


Fig. 11-29. Funcionamiento del sumador de coma flotante y registros empleados.

- Multiplicador de coma flotante (FM): Multiplica dos números M1 y M2, como se muestra en la figura 11-30.

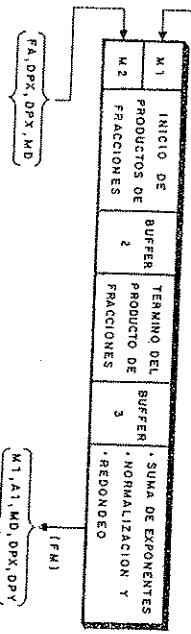


Fig. 11-30. Funcionamiento del multiplicador en coma flotante y registros empleados.

11.7. ARQUITECTURA SIMD: PROCESADORES MÁTRICIALES Y ASOCIATIVOS

Dentro de la tendencia a dotar a los procesos del factor de concurrencia, ya se comentó que existían computadores uniprocesadores matriciales que respondían a la categoría SIMD (flujo simple de instrucciones-flujo múltiple de datos). Constaban de N elementos de procesamiento (ALU y registros) y M módulos de memoria principal, que funcionan bajo el gobierno de una sola Unidad de Control que ejecuta las instrucciones.

Un procesador matricial, normalmente, está conectado a un computador principal a través de la Unidad de Control. El computador principal es una máquina de propósito general que dirige las operaciones de todo el sistema. Entre sus funciones, incluye la supervisión de los periféricos de entrada y salida. La Unidad de Control del computador matricial supervisa la ejecución de los programas, cuyos resultados son enviados al mundo exterior por el computador principal.

La arquitectura SIMD comprende, además de los computadores matriciales con memoria de acceso aleatorio, a los *procesadores asociativos*, caracterizados por el empleo de memorias direccionalables por contenido.

Junto a los bloques habituales de toda máquina programada, los sistemas con varias unidades operativas, necesitan una red de interconexión que las relacione y coordine. Las redes de interconexión constituyen uno de los aspectos más significativos dentro de los factores que se aplican en la selección de una arquitectura.

11.7.1. Procesadores matriciales

Un procesador matricial es una matriz sincrónica de procesadores paralelos, formada por múltiples elementos de proceso (PE) supervisados por la Unidad de Control (CU).

En la figura 11-31 se representan los dos tipos de configuraciones básicas para esta clase de procesadores. En la primera, cada PE se compone de una unidad lógico-aritmética (ALU) con sus registros de trabajo y una memoria local (PEM) para guardar los datos. La unidad de control tiene, además, su propia memoria de almacenamiento del programa que se ejecuta. La CU decodifica las instrucciones y determina dónde han de ejecutarse.

Todos los PE ejecutan la misma función sincrónicamente. Los operando vectoriales se sirven a los PE antes de la ejecución, en paralelo. Hay un sistema de máscaras para habilitar o deshabilitar cada uno de los PE, de forma que sólo trabajen los necesarios.

Para el intercambio de datos entre los diferentes PE, se emplea la red de interconexión controlada por la CU.

El segundo tipo de configuración difiere del primero, en que no dispone de bloques de memoria para cada PE, sino de un grupo de bloques de memoria (M) que son compartidos por el grupo de PE. Por otra parte, no existe red de interconexión, pero sí red de alineamiento que conecta a los PE con las memorias. Esta red es controlada, directamente, por la unidad de control.



11.7.1.1. Arquitectura del MPP

El MPP (*Massively Parallel Processor*) es un procesador de alta integración, desarrollado por la NASA para procesar las imágenes enviadas por los satélites artificiales. Contiene una matriz de $128 \times 128 = 16,384$ microprocesadores en paralelo. Además, el MPP tiene una unidad de control microprogramada (ACU) y cada PE tiene asociados 1 Kbits de memoria de acceso directo.

Dispone de 132 columnas, 128 normales y 4 de reserva para sustituir alguna columna defectuosa. Las funciones aritméticas de cada PE se ejecutan mediante un sumador serie y un registro de desplazamiento.

La unidad de manejo de programas y datos es un miniordenador que controla el flujo de datos en la matriz, carga programas en el controlador, ejecuta rutinas de autodiagnóstico y facilita el desarrollo de programas.

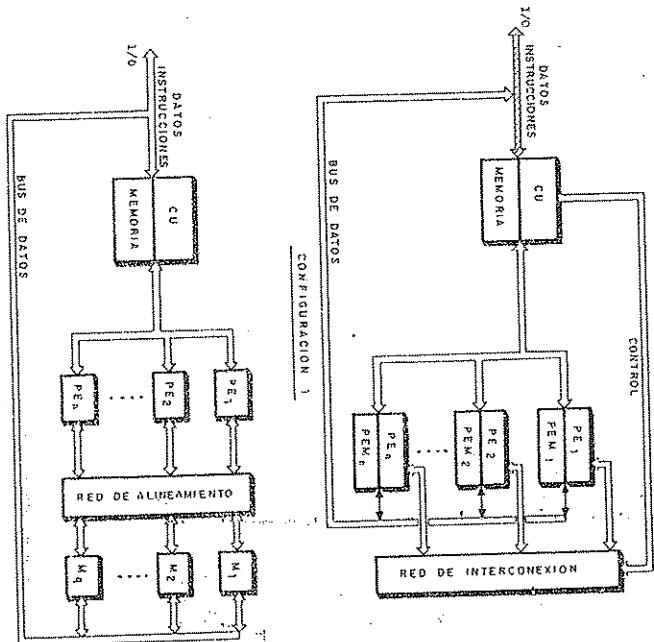


Fig. 11-31. Configuraciones de los procesadores matriciales.

Según lo expuesto, un procesador matricial se caracteriza por los siguientes parámetros:

- $C = \{N, F, I, M\}$; donde
- N : Número de PE.
- F : Funciones de transmisión de datos por las redes de interconexión o alineamiento.
- I : Instrucción máquina.
- M : Máscara para habilitar y deshabilitar los PE.

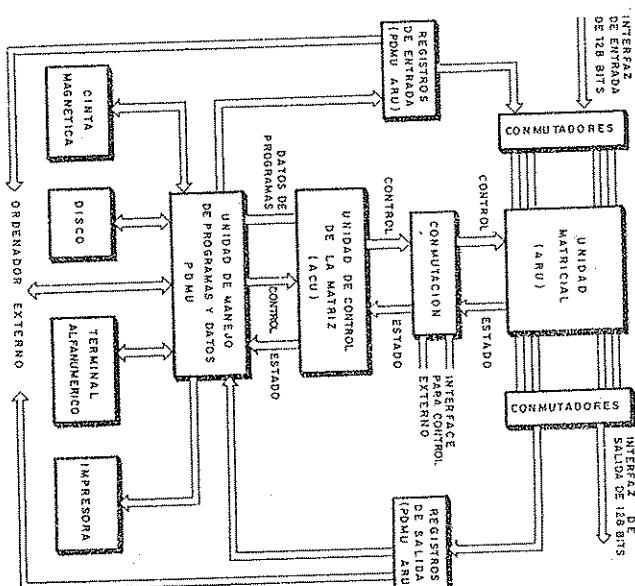


Fig. 11-32. Arquitectura general del sistema MPP.



Como ordenador externo de control se usa un PDP-11/34. El MPP incluye periféricos como controlador de cinta magnética, impresora, terminal y dos discos de 67 Mbytes.

Puede trabajar independientemente, sin ordenador externo, en modo stand-alone, mediante el terminal y los comandos propios del sistema MPP. En el modo on-line el ordenador externo le suministra datos, programas y petición de trabajo; también recibe los datos generados por el MPP e información sobre su estado.

La frecuencia de reloj para el funcionamiento de la matriz es de 10 MHz. La figura 11-32 muestra la arquitectura del sistema MPP.

11.7.2. Procesadores asociativos

Son del tipo matricial, pero sustituyendo la memoria de acceso directo (RAM) por otra de tipo asociativo (CAM). Mientras que en la RAM se precisa de la dirección antes del acceso al dato, en las CAM los datos son direccionables por contenido, permitiendo el acceso a numerosas palabras de la memoria. Además, se caracte-

riza porque las operaciones lógicas y aritméticas se efectúan sobre muchos conjuntos de argumentos en una instrucción simple.

11.7.2.1. Arquitectura del procesador STARAN

Se trata de un procesador *asociativo-bit-serie* por lo que su coste es reducido, en comparación con los de estructura paralela. El STARAN se compone de 32 módulos asociativos de matrices, como máximo. Cada módulo dispone de una memoria de 256 palabras de 256 bits cada una, de acceso multidimensional, una red de permutación y un selector, tal como se muestra en la figura 11-33.

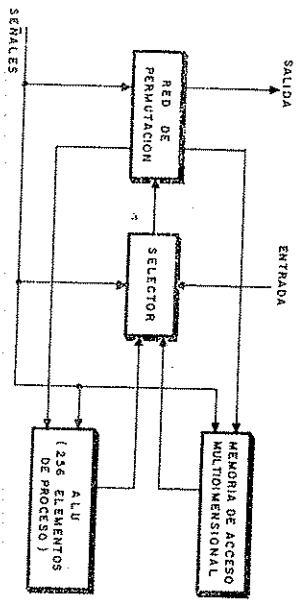
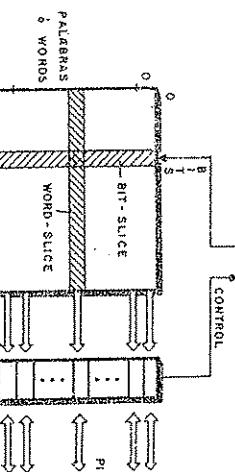


Fig. 11-33. Módulo asociativo de matriz del computador STARAN.

500 / Computadores de altas prestaciones



Fig. 11-34. Diferenciación entre operaciones paralelas en el STARAN.



Cada elemento de proceso opera en serie, bit a bit, en los datos de todas las palabras de memoria de acceso multidimensional (MDAM). En la figura 11-34 se hace una distinción entre trabajos con bit slice o word slice.

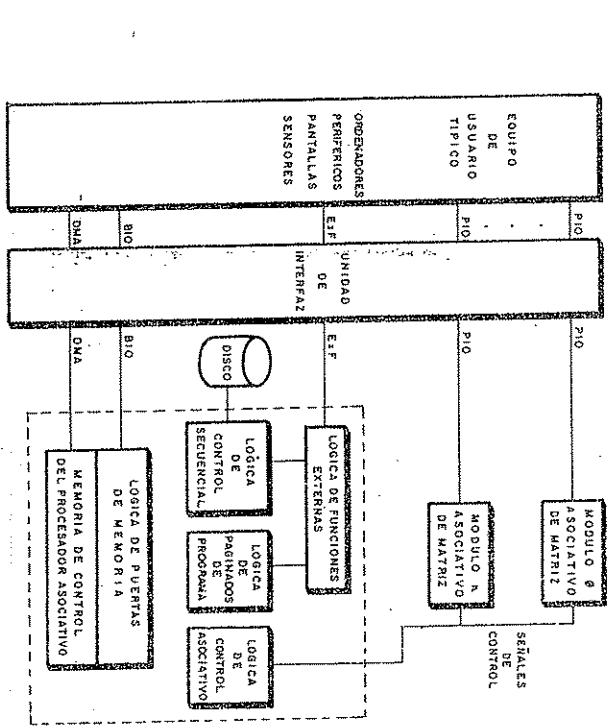


Fig. 11-35. Arquitectura general por bloques del sistema STARAN.

Computadores de altas prestaciones / 501

Utilizando la red de permutación, los datos almacenados en la M-DAM pueden ser accedidos por los canales de E/S en bit-slices, word-slices o una combinación de ambos. También se emplea la red de permutación para desplazar o manipular datos, para permitir búsquedas paralelas, operaciones aritméticas o lógicas entre palabras de la M-DAM.

Para localizar un dato particular, el STARAN inicia una búsqueda con un modo de comparación, por medio de la lógica de control asociativa. En la ejecución de una instrucción, los datos de todas las memorias seleccionadas se procesan simultáneamente por el elemento de proceso de cada palabra.

La unidad de interfaz, mostrada en la figura 11-35, incluye conexión con sensores, computadores convencionales, pantallas interactivas y dispositivos de almacenamiento masivo. Dentro de las opciones de E/S se contempla el acceso directo a memoria (DMA), canales de E/S (BIOS), canales de funcionamiento externo (EXF) y E/S paralelo (PIO).

Cada módulo asociativo de matriz puede contener hasta 256 entradas y 256 salidas en la unidad de interfaz. Estas pueden utilizarse para aumentar la velocidad en la comunicación de datos entre módulos.

11.8. MULTIPROCESADORES O COMPUTADORES MIMD

Un sistema multiprocesador es un computador compuesto por múltiples procesadores de instrucciones, que cooperan para obtener un fin común. Este concepto hace que difiera del correspondiente al computador múltiple o multicomputador, que es un sistema formado por varios computadores autónomos que pueden o no trabajar coordinadamente.

La arquitectura MIMD comprende N elementos de proceso de instrucciones, que se conectan con M módulos de memoria principal a través de una red de interconexión. Además, existe una unidad de coordinación que controla y sincroniza los procesos en ejecución, aunque no ejecute el código objeto. Figura 11-36.

Las arquitecturas de los multiprocesadores se clasifican en dos grandes grupos:

1. *Sistemas débilmente acoplados*. Cada procesador tiene un dispositivo de E/S y una memoria local. Los mensajes entre los procesadores se realizan a través de un sistema de transferencia de mensajes global (STM) a grandes velocidades (1 Mb/s). Estos sistemas son eficientes cuando la interacción es mínima. Figura 11-37. El acoplamiento débil sólo interesa cuando hay muy poca interacción entre los procesadores.

502 / Computadores de altas prestaciones

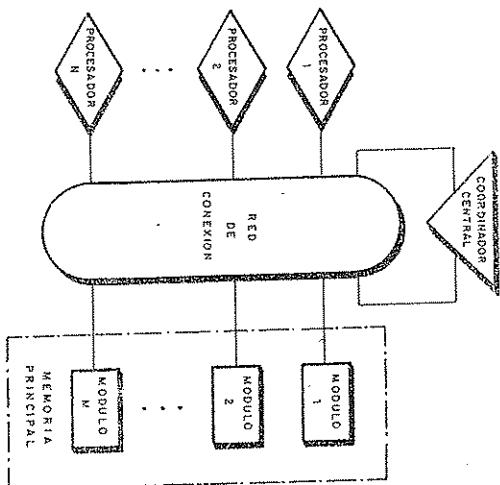


Fig. 11-36. Esquema general de la arquitectura MIMD.

2. *Sistemas fuertemente acoplados*. Se comunican a través de una memoria principal compartida, disponiendo cada procesador de una memoria cache propia. Admiten un alto nivel de interacción entre las tareas, sin que se produzca un detitorio importante de la respuesta. Sin embargo, se originan conteniciones cuando varios procesadores intentan acceder a la misma unidad de memoria a la vez. Figura 11-38. En resumen, todos los procesadores acoplados fuertemente pueden utilizar todos los recursos del sistema.

Un multiprocesador exige un potente sistema operativo que proporciona todas las necesidades que surgen en relación con los procesadores, los procesos, los programas y los datos.

Existen tres sistemas operativos clásicos:

- A) *Sistema operativo maestro-esclavo*.

Las rutinas del supervisor se ejecutan desde un único y exclusivo procesador, denominado maestro. Los restantes procesadores, esclavos, deben solicitar ser atendidos.

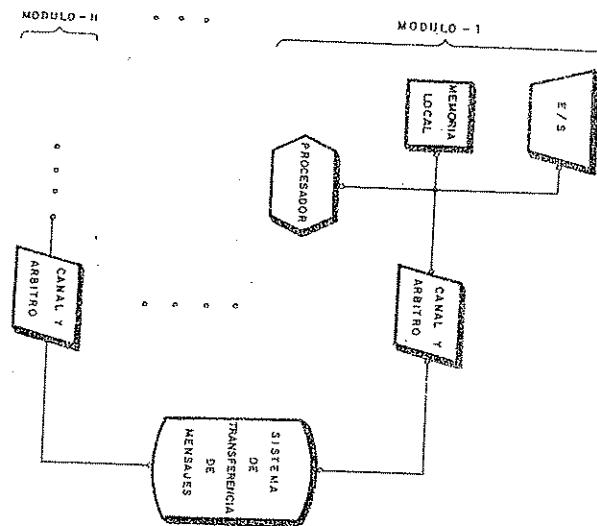


Fig. 11-37. Estructura de un sistema débilmente acoplado.

didos por el maestro. Dado que todo el funcionamiento depende del maestro, esta estructura implica que estén asociados el equipo físico y el sistema lógico.

B) Sistema operativo con supervisores separados en cada procesador

Existe duplicación de ciertas rutinas del supervisor en cada procesador, por ejemplo, las encargadas del tratamiento de las operaciones de E/S. También se dispone de tablas comunes que comparten ciertos problemas de control de acceso.

C) Sistema operativo de supervisor flotante

El "maestro" se traslada procesador a procesador, pudiéndose ejecutar varias rutinas del supervisor a la vez e, incluso, la misma rutina en varios procesadores. Los conflictos en las peticiones de rutina de supervisor se resuelven mediante el método de las prioridades. En este modelo un fallo en un procesador no afecta, especialmente, al sistema.

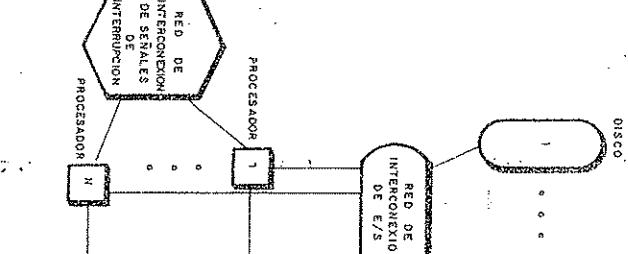


Fig. 11-38. Sistema fuertemente acoplado.

La capacidad que ofrecen los procesadores que comparten los módulos de memoria y, a veces, los dispositivos de E/S, se materializan en diversas estructuras, entre las que destacan:

1. Red de interconexión de bus común

Es la topología más simple y económica. Se basa en un bus común compartido por todos los elementos del sistema. Como los componentes de la red de comunicación son pasivos, el sistema debe poseer mecanismos para resolver las posibles conflictos. Los métodos más utilizados para este fin son las colas FIFO y el método de las prioridades fijas o variables. De este modo, cada procesador o unidad de E/S deberá atenerse al protocolo vigente cuando deseé hacer uso del bus. Figura 11-39.

La reestructuración y expansión de la organización del equipo físico es sencilla, ya que únicamente habrá que añadir unidades en el bus.

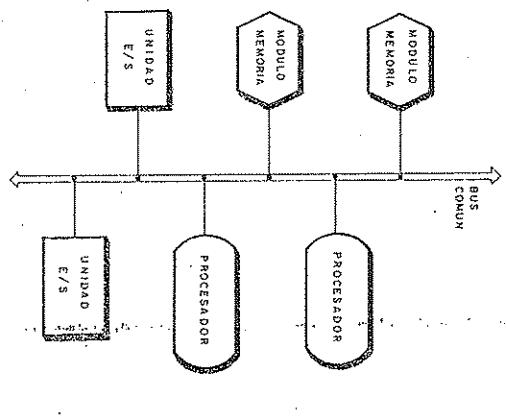


Fig. 11-39. Topología de la red de interconexión de bus común.

2. Red de interconexión "crossbar switch"

El control de las comunicaciones recae en una unidad especial de commutación, que dirige el flujo de información de los buses. Las expansiones son posibles y sólo están limitadas por el tamaño de la matriz de commutación.

Como puede deducirse de la figura 11-40, esta arquitectura simplifica el diseño del resto de los módulos.

3. Red de interconexión con memoria multipuerto

Un sistema de memoria multipuerto distribuye las funciones de control, comunicación y prioridades entre las unidades controladoras de los módulos de memoria.

La reconfiguración del sistema viene limitada por el tipo y número de unidades de memoria utilizadas en cada diseño. Figura 11-41.

505 / Comunicadores de alta prestaciones

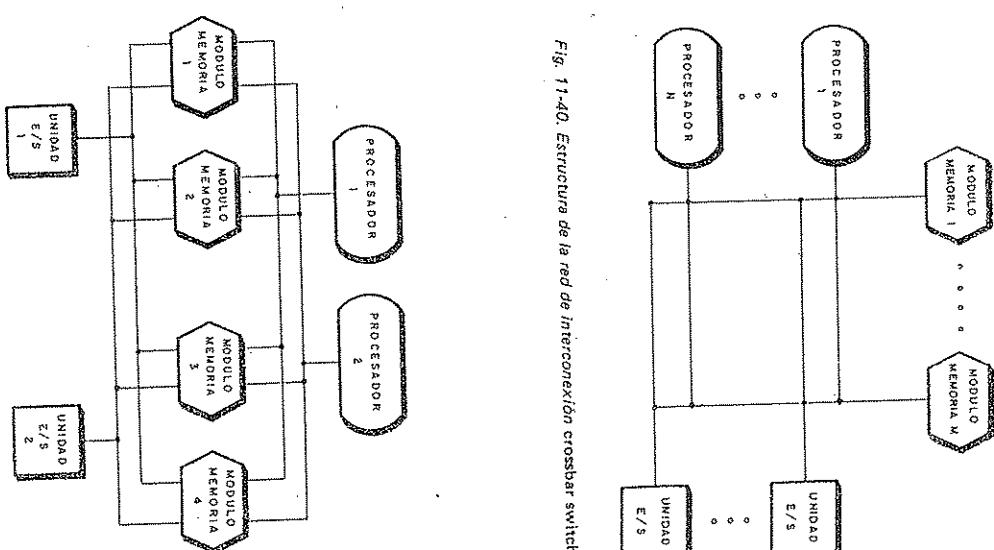


Fig. 11-40. Estructura de la red de interconexión crossbar switch.

Fig. 11-41. Red de interconexión multipuerto.

11.8.1. Multiprocesador "S-1"

Al "S-1" se le puede describir como un procesador de propósito general de alta velocidad. Está construido con los uniprocesadores S-1, llamados Mark II A. En la figura 11-42 se muestra la estructura de un S-1. Incluye 16 uniprocesadores Mark II A independientes que comparten 16 bancos de memoria. Cada banco puede contener hasta 2^{30} bytes, es decir, entre los 16 bancos se pueden direccionar hasta 16 Gigabytes (2^{34}).

Para la resolución eficiente de grandes problemas es crucial el empleo de grandes memorias. La capacidad de direccionamiento del S-1 elimina el costo de programación para manejar muchos sistemas de almacenamiento.

Entre procesador Y memoria se puede hacer una transferencia de una palabra en 50 ns, alcanzando una frecuencia en la transferencia de datos de 320 Mpalabras/s. El computador entrercruzado tiene dos funciones. Por una parte, controla las peticiones de acceso a memoria por parte de los procesadores; de tal manera que un procesador no puede acceder dos veces seguidas a un mismo banco si hay otro que está esperando hacerlo. Por otra parte, se encarga del manejo de las comunicaciones entre procesadores.

Cada procesador tiene una memoria cache privada, que es transparente al usuario. El S-1 dispone de un subsistema de E/S que consiste en muchos canales de E/S con microcódigo. Cada canal está gobernado por un procesador de E/S. El subsistema de E/S contiene un buffer de E/S o memorias accesibles dentro del espacio direccional por el procesador. Para cada canal existe un buffer de 2 K palabras. Estas memorias de E/S son compartidas entre un procesador S-1 y un procesador de E/S.

En la figura 11-43 se representa, con más detalle, la estructura de un uniprocesador Mark II A.

En la figura 11-43 se distinguen claramente las cinco unidades que componen el uniprocesador:

- Unidad de captura de instrucciones, guardadas en la memoria cache.
- Unidad de decodificación de la instrucción capturada, por medio de la RAM de decodificación.
- Unidad de preparación de los datos a ejecutarse y provenientes de otra memoria cache.
- Unidad aritmética del tipo pipeline.
- Unidad de interfaz con la memoria de S-1.

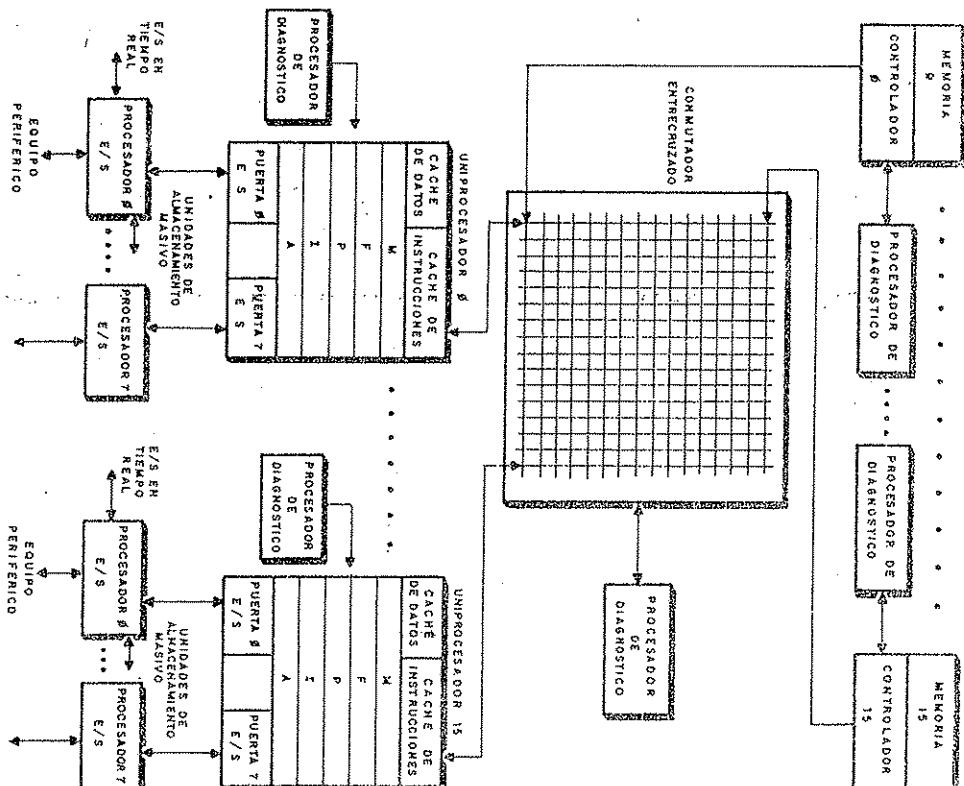


Fig. 11-42. Estructura del multiprocesador S-1.

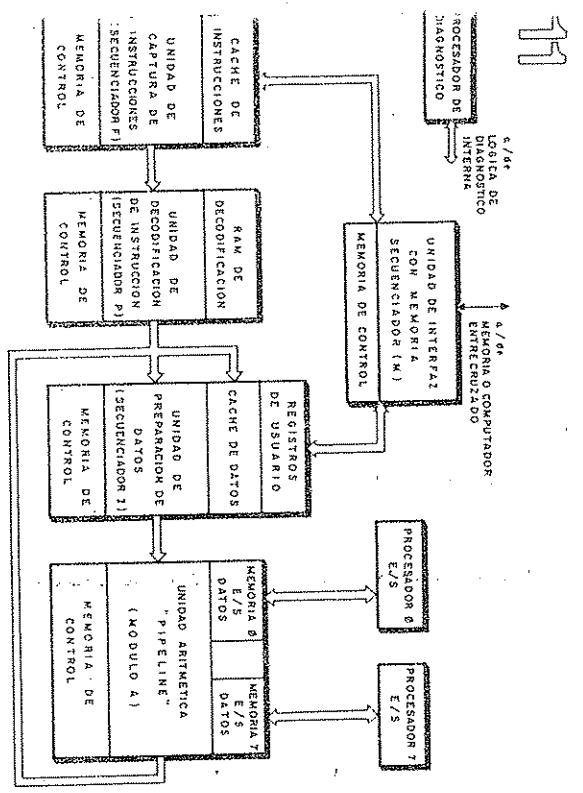


Fig. 11-43. Estructura interna del uniprocesador Mark IIA.

11.8.2. Procesadores sistólicos

El concepto de arquitectura sistólica fue desarrollado por Kung y utilizado en la Universidad de Carnegie-Mellon. Un sistema sistólico consiste en un conjunto de celdas interconectadas, cada una de las cuales es capaz de ejecutar alguna instrucción simple. Las celdas se interconectan en forma de matriz o de árbol. La información en un sistema sistólico fluye entre celdas en una estructura segmentada y la comunicación con el exterior sólo es posible en las celdas fronterizas.

El funcionamiento de un procesador sistólico se representa en la figura 11-44. La memoria impulsa los datos al elemento de proceso (PE), de forma semejante al "corazón". En la figura 11-44 a) se aprecia cómo un procesador convencional procesa una bifurcación mientras que el procesador sistólico de la figura 11-44 b) ejecuta 6 veces más instrucciones a la vez. Los datos se introducen por un lado de la red a manera de oleadas. El procesamiento sistólico opera de forma similar a las contracciones del corazón. Los datos se procesan rítmicamente.

El problema de este sistema radica en asegurarse, que, una vez tomado el dato de la memoria, se use efectivamente en cada celda por la que pasa.

510 / Computadores de altas prestaciones

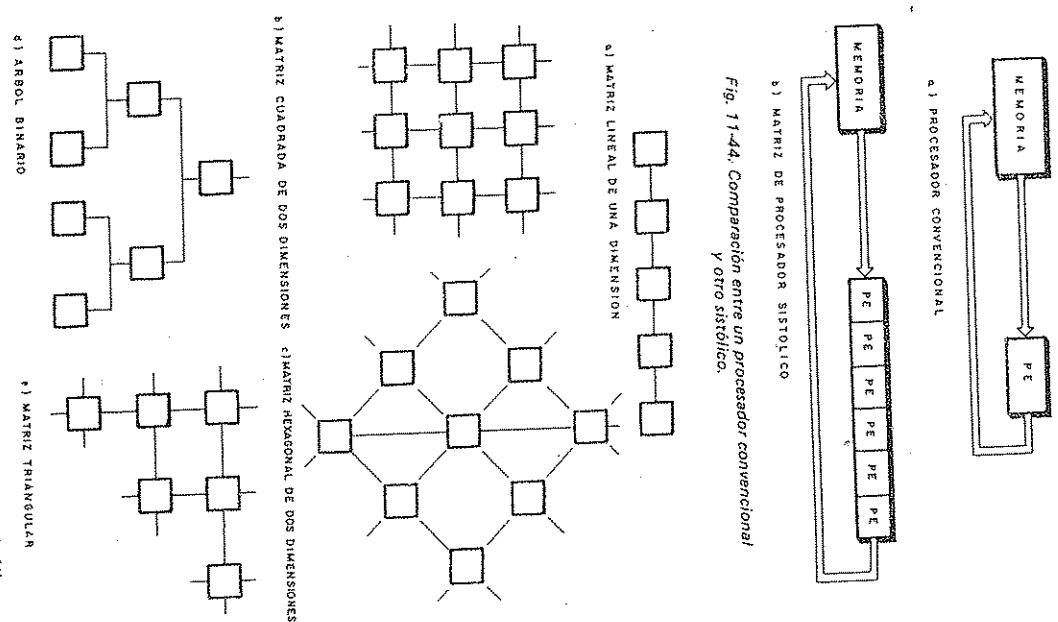
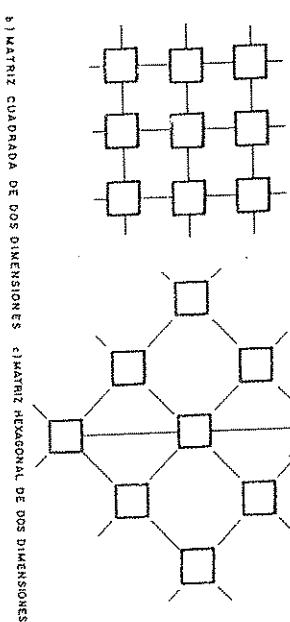
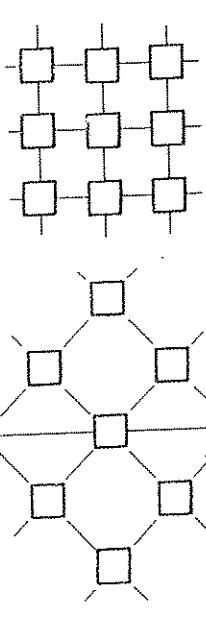


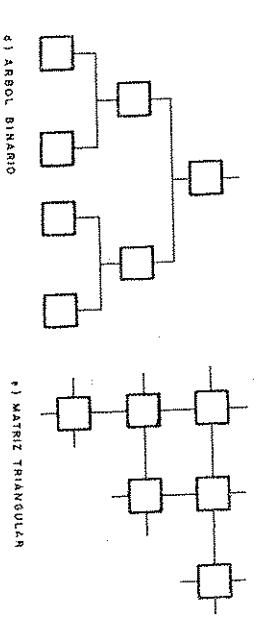
Fig. 11-44. Comparación entre un procesador convencional y otro sistólico.



b) MATRIZ CUADRADA DE DOS DIMENSIONES



c) MATRIZ HEXAGONAL DE DOS DIMENSIONES



d) MATRIZ TRIANGULAR

e) ARBOL BINARIO



Las celdas de proceso básicas, derivadas en la construcción de matrices aritméticas (a , b y c) y tres salidas ($-a = a$, $b = b$ y $d = c + a \cdot b$). Todos los registros de entradas y salidas están controlados por un reloj para conseguir una transferencia síncrona de datos entre celdas adyacentes.

Las matrices sistólicas VLSI pueden asumir muchas estructuras diferentes, según los algoritmos de cómputo. Su topología se adapta a la del algoritmo implementado. La figura 11-45 muestra varias configuraciones matriciales sistólicas y sus más frecuentes usos se indican en la tabla de la figura 11-46.

CONFIGURACION SISTOLICA	FUNCIONES DE COMPUTACION
MATRICES LINEALES DE UNA DIMENSION	CONVOLUCION, TRANSFORMADA DE LA PLACA DISPERSA, PRODUCTO CARTESIANO, COLAS DE PRIORIDAD EN TIEMPO REAL, UNIDADES ARITMETICAS TUBULARES
MATRICES CUADRADAS DE DOS DIMENSIONES	PROGRAMACION DINAMICA, ALGORITMOS DE GRAFICOS
MATRICES HEXAGONALES DE DOS DIMENSIONES	ARITMETICA DE MATRICES, COMPARACION DE MODELOS, TRANSFORMADA DE LA PLACA DISCRETA, OPERACIONES DE BUSES DE DATOS RELACIONALES
ARBOLLES	ALGORITMOS DE BUSQUEDA, EVALUACION DE FUNCIONES PARALELAS, EVALUACION DE RECURRENCIA
MATRICES TRIANGULARES	INVERSIÓN DE MATRICES TRIANGULARES, RECONOCIMIENTO DE LENGUAJE FORMAL

Fig. 11-46. Tabla que presenta las principales funciones de cada configuración sistólica.

La implementación de matrices sistólicas en un chip VLSI ofrece muchas dificultades prácticas. El mayor problema es el del ancho de banda de las E/S. Con la tecnología normal de encapsulado de circuitos integrados, sólo se pueden usar un reducido número de pines de E/S en un chip VLSI. Para aliviar esta dificultad, se recurre a la compartición de las pueras de E/S y a la multiplexación de éstas en el tiempo.

11.9. COMPUTADORES INTELIGENTES DE LA QUINTA GENERACIÓN

Entre los computadores de la quinta generación se encuentran los denominados *inteligentes*, diseñados para procesar conocimientos en vez de datos, y los *computadores de flujo de datos*, que se describen en el siguiente apartado.

512 / Computadores de altas prestaciones



En octubre de 1981, Japón anunció la puesta en marcha de un proyecto destinado a desarrollar los *computadores de la quinta generación*. El proyecto se dividía en tres etapas (1982/85, 1985/89 y 1989/92). En la primera etapa se ha desarrollado el *equipo físico básico* y el *sistema lógico fundamental*. Además, se han construido modelos pilotos para soportar el desarrollo del *sistema lógico*.

Con esta orientación se han construido los procesadores PSI. Estas máquinas trabajan a una velocidad de 30 K LIPS (1 LIP es una inferencia lógica por segundo). Otros diseños de máquinas inteligentes alcanzan un estado avanzado, como sucede con el sistema DELTA.

Los computadores fabricados hasta el momento presente están basados en la arquitectura típica de von Neumann, favoreciendo esta filosofía de funcionamiento los lenguajes máquina que disponen. Igualmente, las estructuras de los lenguajes de alto nivel (LAN) están influenciadas por el lenguaje máquina.

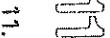
Los computadores inteligentes pretenden desviarse del procesamiento secuencial, sustituyéndolo por un modo de trabajo paralelo. También deberán soportar una búsqueda asociativa (busqueda por contenido y no por dirección), puesto que la operación básica que realizan es la *inferencia lógica*.

Por otra parte, el lenguaje máquina empleado es el denominado *lenguaje náutico o lenguaje kernel*, basado en la lógica de predicados. Aunque este lenguaje es del tipo máquina, es considerado como lenguaje de muy alto nivel y facilita el desarrollo de funciones de ayuda al usuario más cómodamente, intentando un acercamiento a los lenguajes naturales.

11.9.1. Requerimientos de los computadores inteligentes

Las máquinas de la quinta generación están enfocadas a procesar conocimientos, razón por la que se las denomina KIPS (Knowledge Information Processing Systems o Sistemas de Proceso de la Información del Conocimiento). Son capaces de realizar inferencias o deducciones lógicas a partir de hechos y reglas contenidos en una base de conocimientos y están facultados para soportar las siguientes funciones:

- Resolución de problemas mediante inferencias, que pueden ser deductivas o inductivas.
- Gestión de la base de conocimientos. Las inferencias se basan en los conocimientos acumulados por el sistema.
- Interfaz hombre-máquina que utilice el lenguaje natural, gráficos, etc.
- Mayor inteligencia de la máquina para obtener programas eficientes con lenguajes más próximos al usuario.



11.9.2. Estructura de los computadores de la quinta generación

En la figura 11-47 se muestra el diagrama por bloques de un prototipo de este tipo de máquinas, que, básicamente, está formado por tres secciones:

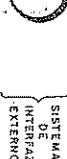
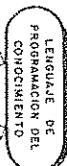
1. Interfaz externo.
2. Sistema de software.
3. Sección de hardware.

La sección destinada al sistema lógico consta de los bloques siguientes:

- a) Módulo de sistema lógico para inferencias.

Se encarga de efectuar inferencias complejas con ayuda de la máquina de inferencia. Es equivalente al sistema operativo de los computadores convencionales.

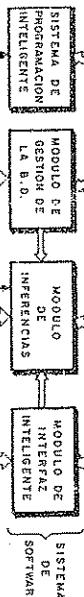
b) SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



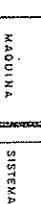
SISTEMA DE INTERFAZ EXTERNO



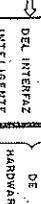
SISTEMA DE INTERFAZ EXTERNO



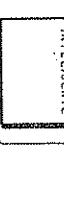
SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



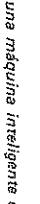
SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO



SISTEMA DE INTERFAZ EXTERNO

11.10. MAQUINAS DE FLUJO DE DATOS

Se basan en un nuevo concepto de ejecución de las instrucciones de un programa, radicalmente diferente al propuesto por von Neumann. Las instrucciones son activadas por la disponibilidad de los operandos o datos que requieren, no existiendo flujo de control ni contador de programación. La red de conexión distribuye la carga de trabajo entre todos los procesadores.

Los computadores de flujo de datos tienen una organización de conducción de datos que se caracteriza por un estado de muestreo pasivo. Las instrucciones son examinadas para detectar si los operandos están disponibles. Si es así, son ejecutadas inmediatamente cuando la unidad correspondiente se halla libre. De esta mane-

ra, el sistema de gestión de la base de datos de los sistemas clásicos, pero con funciones más inteligentes, puesto que en las bases de conocimiento, se almacenan hechos y reglas que permiten obtener nuevas deducciones. Es conveniente el uso de memorias asociativas que se detectan por su contenido y el tiempo de acceso no depende del número de posiciones consultadas.

c) Módulo de sistema lógico de interfaz inteligente.

Gestiona la comunicación entre el computador y el usuario, que puede utilizar lenguajes naturales.

d) Módulo de sistema lógico para programación inteligente.

Pretende facilitar la labor de los programadores con la finalidad de conseguir una programación automática.

e) Máquina de inferencia.

Corresponde con la UCP de los sistemas convencionales, ya que se encarga de la ejecución de las instrucciones del lenguaje náutico o kernel. Está constituida por un mecanismo de inferencia en paralelo, un mecanismo para el flujo de datos y un mecanismo para el tratamiento de los tipos abstractos de datos.

f) Máquina de la base de conocimientos.

Procesa los conocimientos expresados en formas complejas. Controla las bases de conocimientos y la base de datos relacional, donde están grabados los hechos y las reglas.

Todos los elementos que configuran la sección de equipo físico están fabricados con tecnología VLSI (Escala de integración muy alta).

Fig. 11-47. Estructura típica simplificada de una máquina inteligente de la quinta generación.

514 / Computadores de altas prestaciones



ra se alcanza un alto grado de paralelismo, puesto que varias instrucciones pueden ser ejecutadas simultáneamente y asincrónicamente.

Estas máquinas precisan de un lenguaje que permita la representación del paralelismo, como puede ser el *Value Algorithmic Language* (VAL) o el *Linear Dataflow* (LD). En ellos se ha introducido un formalismo que permite al programador sacar el mayor rendimiento a la capacidad de paralelismo de la máquina, lo que requiere innovaciones importantes tanto en el sistema lógico como en el equipo físico. Un programa de flujo de datos se representa mediante un diagrama de flujo, donde los operadores son los "nodos" y los datos son las "conexiones". Un ejemplo de cómo se calcula las raíces de una ecuación de segundo grado. Un ejemplo de cómo se calcula las raíces de una ecuación de segundo grado. Los nodos son los operadores: producto, resta, raíz cuadrada, suma y división.

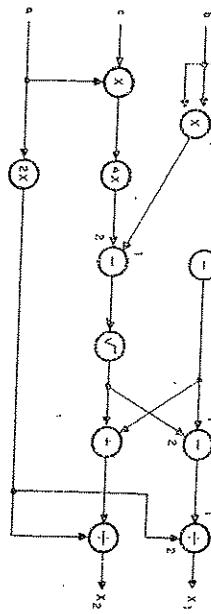


Fig. 11-48. Ejemplo de programa de flujo de datos que resuelve la ecuación $a \cdot x^2 + b \cdot x + c = 0$.

Dependiendo del modo en que se manipulan los datos capturados, existen dos arquitecturas de computadores de flujo de datos:

Máquinas de flujo de datos estáticas

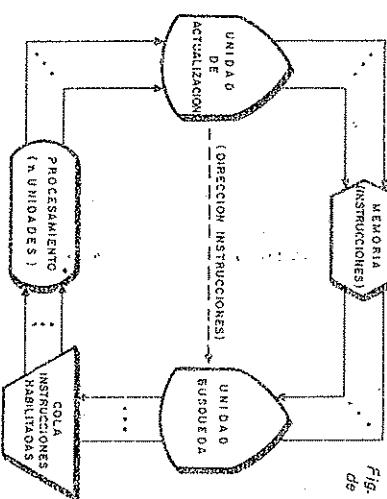
Los datos se desplazan a lo largo de los arcos del grafo hacia los operadores. Sólo está permitida la existencia de un dato en cualquier arco en cada instante. Existe un controlador que transfiere los datos de un nodo a otro. Un nodo se activa cuando están listos los datos de entrada y no hay alguno en la salida. Como se aprecia en la figura 11-49, constan de varias secciones unidas por canales, por lo que la información circula en forma de "paquetes".

La sección de memoria, mostrada en la figura 11-49, está formada por posiciones ten n unidades de proceso, encargadas de realizar las operaciones precisas. Finalmente, las unidades de control y distribución transportan los paquetes de operandos de una sección a otra.

516 / Computadores de altas prestaciones



Fig. 11-49. Estructura general de una máquina de flujo de datos estática.



Máquinas de flujo de datos dinámicos

Con este sistema se marcan los datos, los cuales pueden aparecer simultáneamente en cualquier punto del grafo. Se precisa de circuitería adicional para marcar los datos, pero se prescinde de las unidades de control de flujo. Figura 11-50.

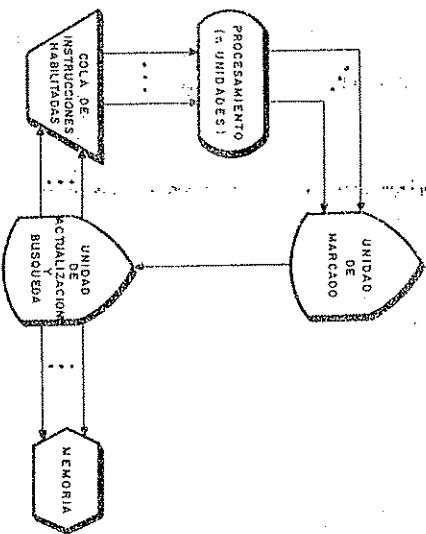


Fig. 11-50. Estructura simplificada de una máquina de flujo de datos dinámica.



EJERCICIOS

Ejercicio 11.1

Contesta brevemente a las siguientes cuestiones:

- Motivos del reforzamiento del paralelismo y el aumento de la velocidad de procesamiento.
- Clasificación de los computadores atendiendo a su paralelismo.

- Esquema resumen de la clasificación comercial de los computadores.
- Concepto de segmentación. ¿Cuáles son las condiciones para el tratamiento segmentado?

- ¿Qué es un parón? Y un choque? ¿Cuándo surgen los problemas de parones en la secuencia de instrucciones? ¿Cómo se pueden evitar?
- ¿Qué son los procesadores vectoriales? Ejemplo de procesador vectorial. ¿Qué es un operando vectorial? ¿Cómo se especifican las instrucciones vectoriales?
- ¿Qué es un computador "array"? Ejemplo.
- ¿Qué tipos de computadores comprende la arquitectura SIMD? Ejemplos.
- ¿Qué son los multiprocesadores? Ejemplo. ¿Qué tipos de sistemas operativos pueden admitir?
- ¿Qué es un procesador sistólico?
- ¿Qué funciones pueden soportar los computadores inteligentes?
- ¿Qué es una máquina de flujo de datos?
- ¿Qué tipo de arquitecturas de computadores de flujo de datos conoce? ¿En base a qué se distinguen?

Ejercicio 11.2

De las alternativas existentes para elevar la velocidad de procesamiento, indicar cuál de ellas le parece más conveniente.

Ejercicio 11.3

Responer verdadero o falso a las siguientes afirmaciones; razonar la respuesta:

- Al aumentar el n.^o de procesadores de un computador aumenta el rendimiento y el coste por lo que se produce un incremento de la relación coste-rendimiento.



2. Las funciones aritméticas de cada PE en el procesador MPP, se ejecutan mediante un sumador serie y un registro de desplazamiento.

3. La principal característica de los computadores de flujo de datos es la presencia de un potente flujo de control.

4. Las unidades de control microprogramadas emplean el concepto de encadenamiento gracias a la existencia de registros apropiados en la UCP.

5. El principal problema de las memorias entrelazadas surge con las peticiones simultáneas sobre un mismo modelo.

Ejercicio 11.4

¿Qué características inducen a la utilización de los microprocesadores en sistemas multiprocesadores?

Clasificar los sistemas con múltiples procesadores según el acoplamiento, especificando, además, las características de cada uno.

Ejercicio 11.5

Inconvenientes de la utilización de máquinas de flujo de datos.

Ejercicio 11.6

Tendencias actuales en la arquitectura de microprocesadores

12

12.1. EVOLUCIÓN DE LOS MICROPROCESADORES

Este capítulo está dedicado a presentar las características más relevantes que se aplican a las arquitecturas de los microprocesadores actuales. En él se describe el equipo físico y el sistema lógico del microprocesador 80286, que puede considerarse como el precursor de los modernos 80386, 80486 y otros. Se supone al lector conocedor del popular 8088/8086, corazón del microcomputador PC y XT de IBM.

En menos de un cuarto de siglo, en que se han desarrollado los microprocesadores, ya se pueden distinguir 5 fases o etapas, claramente diferenciadas y que ya fueron comentadas en el 1.º capítulo.

1.º Fase: Microprocesadores de 4 bits (1971-1974)

Con una estructura interna muy sencilla, estos elementos trabajaban con una palabra de 4 bits y estaban implementados por unos pocos miles de transistores. Su juego de instrucciones era reducido y poco potente. El modelo 4004, fabricado por INTEL con tecnología LSI, es uno de los más representativos de esta etapa inicial.

2.º Fase: Microprocesadores de 8 bits (1974-1975)

La palabra de trabajo se amplió hasta 8 bits, lo que supuso un incremento en la velocidad, la potencia de cálculo y en el repertorio de instrucciones. Con estas prestaciones, Intel comercializó el modelo 8008, diseñado en un principio como controlador de TRAC. El éxito que obtuvo animó al mismo fabricante a introducir una nueva versión, denominada 8080, y que pronto se convirtió en un estándar. Fue en esta época cuando aparecieron los primeros *microcomputadores personales*.

5.20 / Tendencias actuales en la arquitectura de microprocesadores

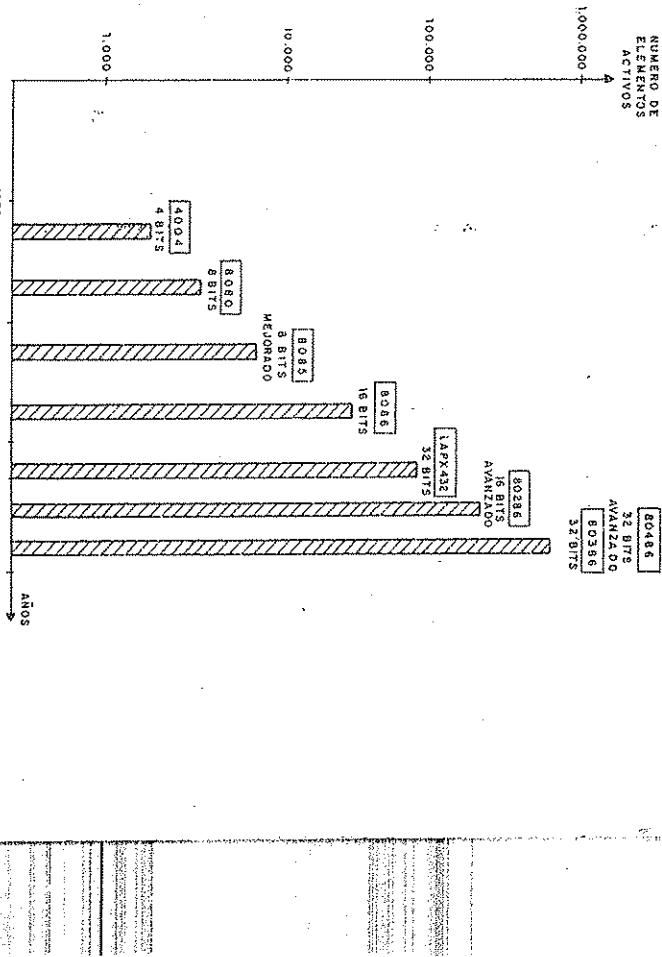


Fig. 12.1. Evolución histórica de los microprocesadores de Intel.

3.º Fase: Microprocesadores de 8 bits mejorados (1976-1978)

Aumenta considerablemente la velocidad de procesamiento; debido a las siguientes causas:

1. Por la implementación de nuevos recursos físicos, gracias a la integración de un mayor número de transistores en el chip.

* Tendencias actuales en la arquitectura de microprocesadores / 521

12

2. Por la ampliación del número de modos de direccionamiento.

3. Por la potenciación del juego de instrucciones.

A esta etapa pertenecen los microprocesadores 8085 de Intel, 6800 de Motorola, 6502 de Mos Technology y el Z-80 de Zilog.

4.º Fase: Microprocesadores de 16 bits (1978-1980)

Con objeto de disponer de unas instrucciones y unos recursos físicos adecuados a los lenguajes de alto nivel, en un intento por mejorar la productividad en el desarrollo de programas y reducir los costes de las aplicaciones, se introducen mejoras a la arquitectura de von Neumann. Con esta orientación aparece el 8086 y el 80286 de Intel, el 68000 de Motorola y el 16000 de National Semiconductor.

Este capítulo está dirigido a quienes conocen teórica y prácticamente los clásicos microprocesadores de 16 bits y desean introducirse y conocer las extraordinarias prestaciones de los modernos microprocesadores de 16 y 32 bits.

En resumen, las mejoras técnicas sustanciales que aportaron los micros de 16 bits fueron:

1. Aumento considerable en la velocidad de funcionamiento. Se hizo frecuente el uso de 5 y 10 MHz.
2. Aumento de la capacidad de memoria, puesto que el bus de direcciones típico de estos UCP sobrepasaba las 20 líneas.

5.º Fase: Microprocesadores de 32 bits (1987-1988)

Están constituidos para poder soportar con facilidad los lenguajes de alto nivel y los sistemas operativos que admiten la multitarea. El campo de aplicación de los microprocesadores se ha visto ampliado enormemente con estos nuevos componentes, ya que características tan importantes como la velocidad de procesamiento, capacidad de memoria y niveles de protección, han permitido que las prestaciones de los sistemas que los contienen se aproxime, e incluso sobreponga, las de los minicomputadores.

Así, por ejemplo, la velocidad de procesamiento de instrucciones está controlada por relojes cuya frecuencia supera los 20 MHz.

Dos nuevos conceptos informáticos adoptan estos microprocesadores:

1. **MEMORIA VIRTUAL:** Que pone a disposición del programador de una gran memoria, aunque la memoria principal "realmente implementada" es mucho más pequeña.

5.22. / Tendencias actuales en la arquitectura de microprocesadores

2. **MULTITAREA:** La UCP distribuye su tiempo entre diversas tareas independientes, que se manejan convenientemente protegidas, para evitar interferencias externas.

Desde su aparición, los microprocesadores se han venido utilizando en aplicaciones indirectamente ligadas a la Informática, en las que actuaba como un componente electrónico más que simplificaba la circuitería lógica. Se usaba, con frecuencia, el lenguaje máquina. El constante crecimiento del diseño del sistema lógico favoreció la construcción de microprocesadores capaces de soportar fácilmente los lenguajes de alto nivel. Por otra parte, la diversificación de las aplicaciones en las que se usa el microprocesador, exige, en muchas ocasiones, la disponibilidad de sistemas multiusuario con una gran capacidad de memoria.

La evolución que han seguido los microprocesadores en las 5 fases descritas, ha estado motivada por diversas circunstancias, entre las que destacan:

- a) El incesante aumento de la densidad de integración debido al progreso de la tecnología VLSI, que ha permitido disponer de pastillas de circuito integrado capaces de realizar funciones similares a las de un minicomputador.

b) La incorporación de un completo juego de instrucciones (CISC) con vistas al empleo de lenguajes de alto nivel. También existen versiones RISC, de juego de instrucciones reducido.

c) Ampliación gigantesca del campo de aplicación del microprocesador, que en algunas áreas hace preciso la posibilidad de contar con grandes espacios de memoria y disponer de varios niveles de protección para soportar diversas tareas.

6.º Fase: Microprocesadores de 64 bits (1989 hasta nuestros días)

12.2. ORIENTACION DE LOS MICROPROCESADORES DE 32 BITS

Incorporando todos los avances de la tecnología VLSI y de la Informática moderna, los microprocesadores avanzados han alcanzado cotas importantes en la velocidad de procesamiento, del orden de varios MIPS (millones de instrucciones por segundo), situándose en este concepto a la altura de los minicomputadores. Se ha superado el valor de 25 MHz como frecuencia de reloj Y, con buses de 32 bits, se superan velocidades de transferencia de información de 40 Megabytes/s.

La arquitectura interna de los modernos microprocesadores también ha quedado afectada Y, de esta forma, se ha logrado usar la técnica del *procesamiento segmentado*, con la cual se descomponen las instrucciones en varios elementos de procesamiento, alcanzando un alto grado de concurrencia.

En cuanto a la estructura del sistema general, junto a la UCP de 32 bits existen diversos coprocesadores que potencian considerablemente la capacidad operativa del conjunto.

Sin embargo, el esfuerzo principal realizado en la construcción de estos microprocesadores, se ha dirigido en las 3 direcciones siguientes:

1. *Capacidad para soportar lenguajes de alto nivel, de forma óptima.*
2. *Espacio de memoria de grandes dimensiones.*
3. *Posibilidad de implantación de sistemas multitarea y multiusuario.*

12.2.1. Capacidad para soportar lenguajes de alto nivel

Durante las dos primeras décadas de uso de los microprocesadores, la programación se basaba fundamentalmente en los lenguajes máquina. Los constructores de sistemas microcomputadoras han puesto cada vez más interés en desarrollar sus portabilidad y reducir el tiempo de diseño.

Hasta hace poco, el número de microprocesadores capaces de soportar eficazmente lenguajes de alto nivel era muy escaso. En los microprocesadores de 32 bits sus repertorios de instrucciones máquina han sido reforzados, lo que ha incidido en modificaciones en la arquitectura interna de la UCP.

También se ha permitido trabajar con datos de diversos tipos (cadenas de caracteres, campos de bits, etc.) para implementar de forma sencilla las instrucciones de alto nivel.

12.2.2. Espacio de memoria de grandes dimensiones

Para alcanzar este requisito se ha adoptado un concepto tradicional en las grandes máquinas, conocido por el nombre de *memoria virtual*. La UCP dirige una gran espacio de memoria enorme, al que se conoce como "virtual", aunque la memoria principal o física sea mucho más pequeña. Recuérdese que los datos e instrucciones que procesa la UCP deben residir en la memoria principal, implementada por circuitos integrados de memoria, RAM. Como disponer de inmensas memorias principales es física y económicamente imposible, de momento, se utiliza la filosofía de memoria virtual, que consiste en poner a disposición del programador una gran memoria virtual (disco) que transfiere la información necesaria a la memoria principal MMU, traducen las direcciones virtuales a direcciones físicas, seleccionando posiciones de la memoria principal en las que se había depositada la parte de la memoria virtual que en ese momento se procesa.

5.2.4. Tendencias actuales en la arquitectura de microprocesadores

Como toda la memoria virtual no cabe en la principal, cada vez que se referencia una dirección que no está contenida en la memoria principal, la MMU pone en marcha un mecanismo para reemplazar un bloque de información residente en la memoria física por otro de la memoria virtual que contenga la dirección referenciada y las próximas a ella, que, probablemente, serán las que se usan después.

El espacio de memoria con el que trabaja el programador es el virtual, que es mucho mayor que el disponible en la memoria principal. Consecuentemente, las direcciones virtuales o lógicas tienen una longitud mayor que las direcciones físicas. La MMU gestiona la traducción de direcciones y la carga de la memoria principal con las partes de la virtual que en cada momento sean precisas. Las dos funciones principales de la MMU son:

1. *Reduce la dirección lógica correspondiente a la memoria virtual a la dirección física de la memoria principal. Caso de que la dirección lógica a la que se referencia tenga cargado su contenido en una dirección física, una vez realizado el proceso de traducción de direcciones, se accede a dicho contenido.*
2. *Si el contenido al que referencia la dirección lógica no se halla almacenado en la memoria principal, se ejecuta una rutina de excepción que trastoca el bloque de información apuntado por la dirección lógica, desde la memoria virtual a la principal. Posteriormente se realiza la traducción de direcciones y se accede a la información referenciada.*

Para llevar a cabo la traducción de direcciones, la MMU utiliza dos técnicas: *Paginación y Segmentación*. También se suele emplear una combinación de ambas técnicas, que se denomina *Segmentación paginada*.

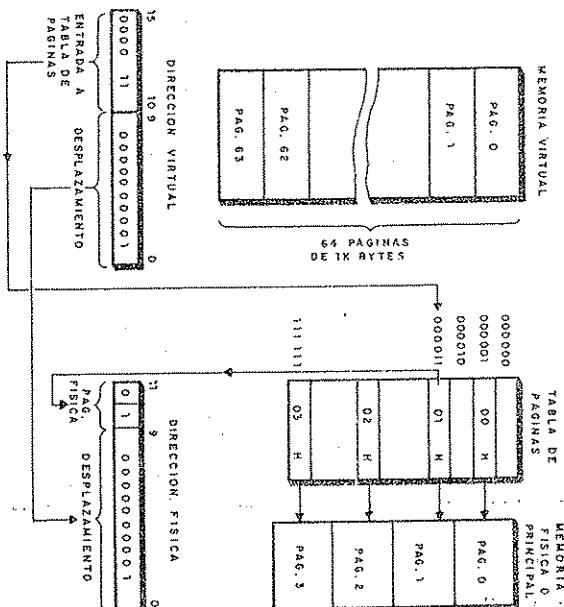
12.2.2.1. Paginación

Con este método el espacio virtual y el físico se dividen en bloques de igual tamaño, llamados *páginas*. En el espacio virtual existen muchas más páginas que en el físico correspondiente a la memoria principal.

Cuando la UCP genera a una dirección virtual, es decir, que hace referencia a una posición de la memoria virtual, con una parte de su código binario se accede a una entrada de una tabla, denominada *Tabla de Páginas*, en la que está guardada la dirección física de la página que contiene dicha posición. Con la información restante de la dirección virtual o *dirección lógica* se determina el *desplazamiento* dentro de la página, cuyo valor se suma al de inicio de la página para apuntar la posición deseada. El desplazamiento es él mismo en la página virtual que en la física.

Para comprender mejor el mecanismo de la paginación se propone el ejemplo de la figura 12-2. Se trata de una pequeña memoria virtual de 64 K bytes, dividida en 64 páginas de 1 K bytes cada una, lo que implica que la dirección lógica tenga 16 bits, que se dividen en dos "campos". El campo de menor peso consta de 10 bits y proporciona el valor del desplazamiento dentro de la página, mientras que el campo de más peso, de 6 bits, determina cuál de las 64 páginas de la memoria virtual se selecciona. Este mecanismo ya fue comentado en el 4.º capítulo.

La memoria principal tiene una capacidad de 4 K bytes que se descompone en 4 páginas de 1 K bytes cada una. En la memoria principal existe un espacio reservado para guardar la Tabla de Páginas, que ocupa 64 posiciones de 1 byte. Cada



"entrada" de la Tabla de Páginas se corresponde con una de las 64 páginas virtuales, y su contenido expresa en cuál de las páginas de la memoria física se halla almacenada la página virtual, caso de que lo esté, puesto que en la memoria principal sólo caben 4 páginas. Por este motivo, uno de los bits de cada posición de la Tabla de Páginas se destina a señalar si la página virtual correspondiente está cargada o no. En la memoria principal, Los restantes bits de dicha posición, aparte de indicar la página concreta de la memoria principal, sirven para expresar ciertos *atributos* de dicha página.

En el caso de la figura 12-2, en la dirección virtual 0000 1100 0000 0001 se referencia la cuarta página virtual.

El valor 000011 actúa como entrada a la Tabla de Páginas, en donde está almacenado el valor 01, que significa que dicha página virtual está depositada en la página 01 de la memoria principal. La posición relativa dentro de la página física se corresponde con la de la página virtual y viene expresada por el valor del desplazamiento 0000 0000 001. En consecuencia, la dirección física será 0100 0000 0001.

En caso de recibir la MMU la dirección de una página virtual que no se halle cargada en la memoria principal, se ejecuta una rutina que sustituye una de las páginas de la memoria principal por la referenciada del espacio virtual al mismo tiempo que se actualiza la Tabla de Páginas. Para señalar si una página virtual está cargada en la memoria principal se suele emplear el valor del bit de más peso de la entrada de la Tabla de Páginas; si es un 0 se halla cargada y si es un 1 no lo está.

12.2.2.2. Segmentación

Como sucede en el microprocesador de 16 bits 8086, el espacio direccionalible por la UCP o lógico se divide en *segmentos*, que son bloques de información de tamaño variable. Los segmentos lógicos se cargan en la memoria principal exactamente igual como ocurría con las páginas. También existe una *Tabla de Segmentos* dentro de la memoria física, que proporciona la situación del segmento en la memoria principal, así como algunos *atributos* que caracterizan a dicho segmento y le confieren la protección adecuada.

La técnica de la segmentación se ajusta mejor a la *programación estructurada* por la flexibilidad en la capacidad de los segmentos y en su sencilla reubicación.

12.2.2.3. Segmentación paginada

El mecanismo de transferencia entre los bloques de la memoria virtual y la física cuando se usa la paginación, queda simplificado porque al ser de tamaño fijo, los algoritmos de intercambio son más sencillos.

Fig. 12-2. Ejemplo de uso de la técnica de la paginación. La página referenciada en la dirección virtual (000011) está cargada en la página 01 de la memoria principal; según se desprende del contenido de la entrada 000011 de la Tabla de Páginas. La Tabla de Páginas está contenida en la memoria física.

La segmentación maneja bloques de tamaño variable con igual tipo de contenido, haciendo más apropiado su empleo en la moderna programación estructurada.

La Segmentación paginada es una técnica usada por la MMU para aprovecharse de las ventajas de la paginación y la segmentación. Esencialmente se basa en dividir el espacio virtual en segmentos de diferente número de páginas.

El 80386 dispone de *Unidad de Paginación* y *Unidad de Segmentación* y es capaz de direccionar hasta 64 Terabytes (trillones de bytes) de memoria virtual y 4 Gigabytes (miles de millones de bytes) de memoria principal.

12.2.3. Posibilidad de implantación de sistemas multitarea

y multiusuario

En los sistemas *multitarea* la UCP y los recursos existentes son compartidos por todas las tareas, aunque en cada instante la UCP sólo atiende a una de ellas.

El Sistema Operativo ha de controlar el acceso de las tareas a los recursos. Uno de los procedimientos para determinar el tipo de acceso permitido a los segmentos o páginas de cada tarea (lectura, escritura, lectura-escritura o posibilidad de acceso), consiste en indicarlo en las entradas de las tablas de traducción del espacio lógico al espacio físico, mediante un campo de bits específico que asigna las operaciones que pueden realizarse sobre el segmento al que referían.

En algunos microprocesadores de 32 bits se han elegido un par de *niveles de protección*, el de *Supervisor* y el de *Usuario*. En el primero se pueden ejecutar todas las instrucciones, mientras que en el segundo sólo un grupo restringido.

12.3. UN DIGNO PRECURSOR: EL MICROPROCESADOR 80286

El microprocesador de 16 bits 8086/8088 fue elegido por IBM a principios de la década de los años 80 de nuestro siglo para la construcción de sus *computadores personales "PC"*. El éxito de estas máquinas se extendió por todo el mundo y ayudó a popularizar el computador y algunas prestaciones informáticas, a nivel de empresas, de profesionales y de personas particulares. Miles de programas de aplicación específicos concretos e incluso máquinas compatibles, que convirtieron a este computador en un estándar de su década.

Pronto, el interés puesto en el PC desbordó sus posibilidades ya que sus limitadas características impedían soportar aplicaciones en los campos de la Inteligencia Artificial, la Robótica, el CAD/CAM, la Visión Artificial, etc.

528 / Tendencias actuales en la arquitectura de microprocesadores

Introducción al 80286 con el nuevo microprocesador 80286, que aunque también trabajaba con palabras de 16 bits, era capaz de manejar la memoria virtual y la multitarea. IBM lo adaptó para su gama AT y constituyó un auténtico "puente" hacia el 80386.

Dado que IBM ha seguido siempre una línea de continuidad en sus productos es muy aconsejable conocer las innovaciones surgidas con el 80286.

Las principales novedades del 80286 respecto al 8086 son tres:

1. Memoria virtual.

2. Multitarea.

3. Capacidad de asignar diferentes niveles de protección en los programas.

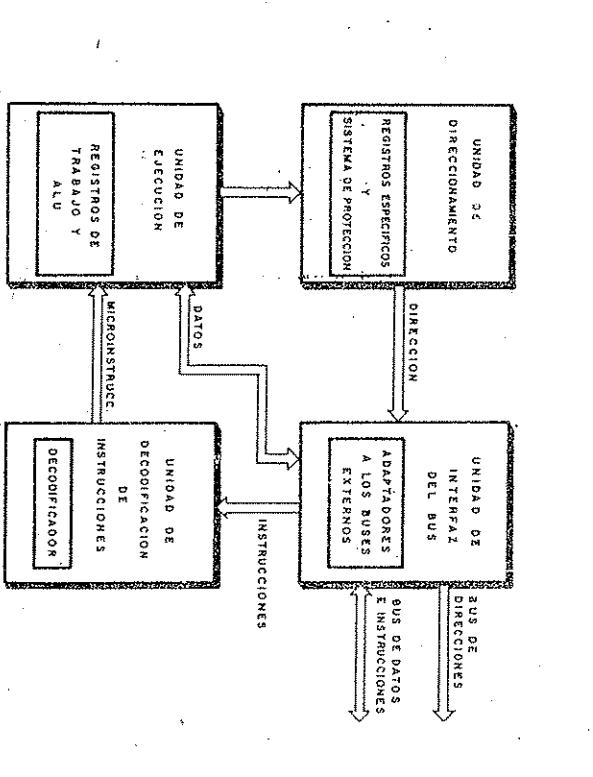


Fig. 12-3. La descomposición de la UCP en 4 secciones independientes favorece la "segmentación", puesto que cada una de ellas atiende a una de las fases en que se divide una instrucción.

No obstante, el 80286 puede o no emplear estas innovaciones y así puede funcionar también de manera similar a la del 8086. Es decir, tiene dos modos de funcionamiento:

a) *Igual que el 8086*, manejando sólo direcciones reales.

b) *En modo protegido*, utilizando todas sus características (memoria virtual, más instrucciones y mecanismos de protección).

Cuando trabaja en el "modo de direcciones reales" el "software" es totalmente compatible con el 8086, pero a mayor velocidad de ejecución, debido a que parte de los 120.000 transistores que conforman el 80286 se destinan a la mejora de la velocidad de proceso. Para conseguir esta mejora se ha aplicado a la arquitectura de este componente la *técnica de segmentación o "pipeline"*, que consiste en dividir cada instrucción matizada en varias partes o microinstrucciones, totalmente independientes entre sí a la hora de ejecutarse. Esta independencia permite, por ejemplo, que mientras una instrucción se está ejecutando, otra esté en la fase de decodificación, en la de búsqueda o en la de terna de operandos. En el mismo tiempo que se procesa una instrucción, con la segmentación se pueden procesar varias. En la figura 12-3 se muestra la estructura interna del 80286, basada en 4 bloques que favorecen la segmentación: Unidad de Decodificación de Instrucciones, Unidad de Ejecución, Unidad de Dirección y Unidad de Interfaz con el Bus.

12.3.1. Diagrama de conexiones

Existen versiones del 80286 que trabajan a 6, 8, 10, 12,5 y frecuencias de reloj superiores, con las que se puede sobrepasar una velocidad de procesamiento de 2 MIPS.

La figura 12-4 muestra el tipo de encapsulado y la distribución de las 68 patillas que dispone el 80286.

A diferencia del 8086, el bus de datos y el bus de direcciones no están multiplexados en el tiempo, lo que elimina una parte de circuitería auxiliar. Dispone de 16 patillas para el bus de datos (D0-D15) y 24 para el de direcciones (4 más que el 8086), que permiten direccionar hasta $2^{24} = 16$ Megabytes de memoria principal. Las líneas del bus de direcciones se denominan por A0-A23.

Las restantes patillas asumen las siguientes funciones:

BHE: Indica si por el bus de datos se transfieren los 8 bits de menor peso y/o los 8 de más peso. Esta señal funciona conjuntamente con el valor de la línea A0.

M/IO: Señaliza si se realiza un acceso a memoria o al mapa de Entrada-Salida.

COD/INTA: Distingue los ciclos de búsqueda de instrucción de los de lectura de datos en la memoria. También indica los ciclos de reconocimiento de interrupción (INTA).

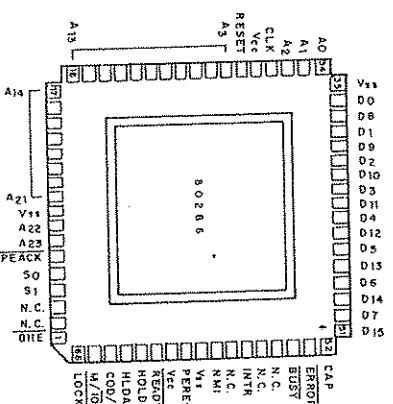


Fig. 12-4. Tipo de encapsulado en el que se presenta el 80286 y distribución de sus 68 patillas.

LOCK: Avisa cuando el procesador tiene el control del bus del sistema, impidiendo que sea utilizado por otros procesadores del sistema.

READY: Señal de entrada al 80286 que permite aumentar la duración de un ciclo de bus.

HOLD Y HOLD#: Petición del bus por otro maestro existente en el sistema y reconocimiento de la cesión del bus, respectivamente.

INTR: Petición de interrupción mascarable.

NMI: Petición de interrupción no mascarable.

S0 y S1: Definen el estado del ciclo del bus, conjuntamente con las señales COD/INTA y M/IO.

PREQ y PEACK: Señales de petición y reconocimiento de una solicitud de servicio por parte del coprocesador numérico, respectivamente.

BUSY y **ERROR**: Indican, respectivamente, que el coprocesador numérico está ocupado y que en el mismo se ha producido un error. Cuando **BUSY** está activa el 80286 permanece detenido. **ERROR** activa una interrupción de control del procesador.

RESET: Inicializa al microprocesador.

VSS y **VCC**: Tierra y +5 V, respectivamente.

CAP: Patita en la que hay que colocar un condensador de filtro con el sustrato de 0,047 μF y 12 V. El otro extremo del condensador se une a tierra.

CLOCK: Señal de entrada de la frecuencia de reloj, que internamente se divide por 2.

12.3.2. Registros

En líneas generales, el bloque de registros del 80286 sigue la misma disposición y nomenclatura que los del 8086; en concreto, dispone de todos los registros del 8086 y 5 nuevos (MSW, GDTR, LDTR, TR e IDTR) que se han añadido para soportar la memoria virtual y la multitarea. En la figura 12-5 se muestra el conjunto de registros similar a los del 8086.

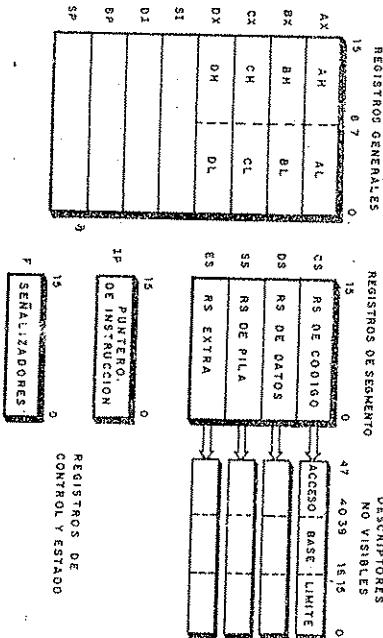


Fig. 12-5. Conjunto de registros del 80286 que coinciden con los del 8086. Aquel dispone de 5 registros más para soportar la memoria virtual y la multitarea, denominadas "registros del sistema".

532 / Tendencias actuales en la arquitectura de microprocesadores

Los 8 registros generales de 16 bits cada uno y denominados AX, BX, CX, DX, SI, DI, BP y SP son capaces de almacenar todo tipo de datos. Además, los 4 registros pueden trabajar con información de 8 bits al estarles permitido operar de forma independiente con sus dos mitades (H y L). Los registros BX, SI, DI, BP y SP se encargan de contener información complementaria para el cálculo de las direcciones. Así, el SP se emplea en el direccionamiento de la Pila, al igual que BP. Los registros BX, SI y DI guardan desplazamientos a efectuar sobre el segmento de datos (a veces el extra).

Los 4 registros de segmento guardan la dirección base o de inicio de los diferentes segmentos que se utilizan en una tarea:

- CS: Segmento de Código donde se guardan las instrucciones.
- DS: Segmento de Datos.
- SS: Segmento de la Pila.
- ES: Segmento Extra.

Estos registros de segmento tienen asociado cada uno, un descriptor caché no visible, que se emplea en el manejo de la memoria virtual. El contenido de estos 4 registros es interpretado de diferente manera según el microprocesador esté trabajando en modo real, como el 8086, o en modo protegido.

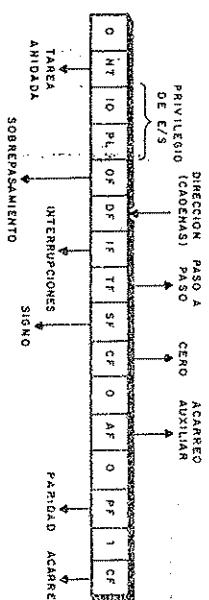


Fig. 12-6. Distribución de los señalizadores en el registro F.

El registro Puntero de Instrucciones, IP, actúa como puntero de instrucciones dentro del segmento en código que se está ejecutando.

El Registro de Señalizadores, F, dispone de la misma información que su homólogo del 8086. Existen indicadores de acarreo, paridad, cero, signo, sobrepasamiento, acarreo auxiliar, habilitación de la ejecución paso a paso, el de interrupciones y el señalizador de la dirección de incremento en las instrucciones sobre cadenas. Además se han añadido dos informaciones:

Tendencias actuales en la arquitectura de microprocesadores / 533

NT: Indica si la tarea que se está ejecutando se halla *anidada* a otra, en multitarea,

IO-PL: Son 2 bits que indican el nivel de privilegio mínimo que se precisa para realizar operaciones de Entrada-Salida.

En cuando a los 5 nuevos registros, propios del 80286 se presentan en la figura 12.7 y reciben el nombre de "registros del Sistema", entrando a funcionar sólo en el modo protegido.

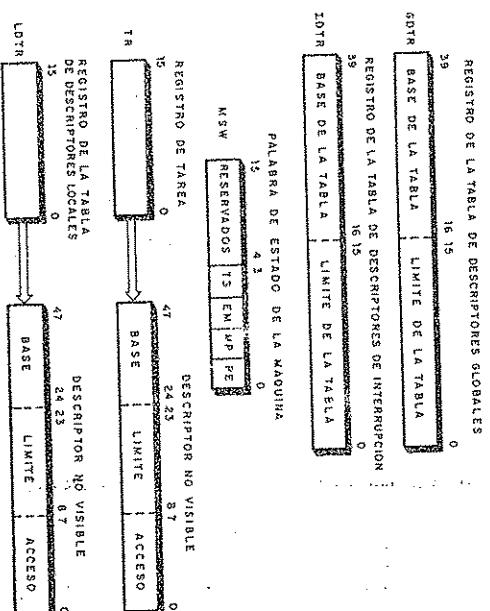


Fig. 12.7. Los 5 nuevos registros del 80286 destinados a soportar la memoria virtual y la multitarea.

La *Palabra de Estado de la Máquina*, MSW, sólo utiliza 4 bits de información, quedando reservados los 12 restantes para futuras ampliaciones en otras UCP (80386). La misión de dichos bits significativos es la siguiente:

PE: Se coloca a 1 para pasar a trabajar en el modo protegido y no vuelve a 0 más que al producirse un RESET.

5.3.4 / Tendencias actuales en la arquitectura de microprocesadores

TS: Bit que indica que ha ocurrido una *commutación entre tareas*. El coprocesador 80287 aún posee información de la tarea anterior. Este bit se pone a 0 cuando la nueva tarea vaya a utilizar el coprocesador.

EM y MP: Bits de *emulador matemático y matemático presente*. Sólo puede estar a 1 uno de los dos, el MP si existe coprocesador en el sistema, y el EM si no está presente y se usa un programa emulador.

Los 4 registros restantes *Registros del Sistema*, GDT, IDTR, TR y LDTR, se emplean para el manejo de la memoria virtual y el soporte de la multitarea. Se explican más adelante.

12.4. MODOS DE TRABAJO

El 80286 puede aceptar dos posibles maneras de funcionamiento:

1. Modo de direcciones reales o compatible con el 8086
2. Modo protegido, en el que se utiliza la memoria virtual y los niveles de privilegio para soportar la multitarea.

12.4.1. Modo real, compatible con el 8086

En este modo el 80286 trabaja con direcciones físicas de la misma forma que el 8086. Sólo se diferencia en que la velocidad de ejecución es más alta debido a la estructura segmentada. También se han añadido algunas instrucciones más.

En este modo no son posibles la memoria virtual y la multitarea.

El direccionamiento Y, por lo tanto, la utilización de los registros de segmento (RS) es similar al del 8086. Es decir, la dirección de una posición se encuentra multiplicando por 16 al registro segmento (añadiéndole 4 ceros por la derecha) y sumándole el desplazamiento. En el caso de buscar en el segmento de código una instrucción, la dirección vendrá dada por:

$$\text{Dirección instrucción} = (\text{CS} \times 16) + \text{IP}$$

12.4.2. Modo protegido

La entrada del 80286 en el modo protegido, implica el uso de toda la potencia de la MMU en el manejo de la memoria virtual. El mapa de memoria se amplía des-

de $2^{24} = 16$ Megabytes, hasta $2^{30} = 1$ Gigabyte. En este modo, el contenido de los registros de segmento es interpretado de forma diferente al modo real, pasando a emplear también los descriptores cache no visibles que tienen asignados.

La memoria virtual de 1 Gigabyte, se divide en dos espacios iguales de 0,5 Gigabytes cada uno. Medio Gigabyte corresponde al espacio de *Memoria Global*, donde están ubicados los programas y datos de uso general y del Sistema Operativo. El otro medio Gigabyte, corresponde al espacio de *Memoria Local*, donde se almacenan el código y los datos de la tarea que se está ejecutando en un momento dado.

La problemática que entraña la memoria virtual se centra en la forma de combinar el hecho de que los programadores manejan un espacio de memoria de 1 Gigabyte, cuando en realidad sólo existe un máximo de 16 Mbytes de memoria principal soportada físicamente con circuitos integrados. Para el tratamiento de esta situación el 80286 utiliza una *TABLA DE DESCRIPTORES GLOBALES* que siempre reside en la memoria principal y una serie de *TABLAS DE DESCRIPTORES LOCALES* que no requieren estar almacenados en la memoria principal permanente.

Cada descriptor controla un espacio de memoria virtual que puede alcanzar un máximo de 64 Kbytes. Existe una *Tabla de Descriptores Globales*, *GDTR*, que define el espacio de memoria global y una *Tabla de Descriptores Locales*, *LDT*, para cada tarea individual que se pueda ejecutar en el espacio de memoria local.

Tal como se muestra en la figura 12-8, cada descriptor consta de 8 bytes que definen perfectamente ese espacio de memoria que representan y que puede alcanzar un total de 64 Kbytes.

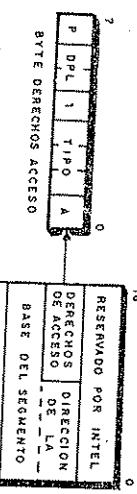


Fig. 12-8 Código descriptor consta de 8 bytes, que definen perfectamente el espacio de memoria que representan y que puede alcanzar una capacidad máxima de 64 K.

Tanto la Tabla de Descriptores Globales, como las de Descriptores Locales, tienen un tamaño máximo de 64 Kbytes, con lo que se pueden definir 8.192 descriptores. Como cada descriptor controla un espacio de memoria de 64 Kbytes, con cada una de las dos tablas se hace referencia a $8.192 \times 64\text{ K} = 0,5$ Gigabyte, alcanzando el control a toda la memoria virtual.

5.36 / Tendencias actuales en la arquitectura de microprocesadores

Una vez explicado cómo se gobierna toda la memoria virtual por medio de los descriptores, queda por conocer cómo se transforma una dirección correspondiente a un espacio de memoria de 1 Gigabyte a otra equivalente situada sobre un espacio de sólo 16 Mbytes.

Como se ha dicho anteriormente, el contenido de los registros de segmento (*RS*) tienen un significado diferente cuando se trabaja en modo protegido. Ahora su contenido, multiplicado por 16 ya no representa la dirección base de un segmento de memoria, como sucedía con el 8086, sino que se asume el papel de selector sobre las tablas de descriptores. La figura 12-9 muestra el contenido de un registro segmento cuando actúa como selector en el modo protegido.

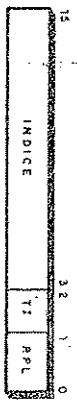


Fig. 12-9 Asignación de los bits de un registro de segmento (*RS*) cuando actúa como selector sobre las tablas de descriptores al trabajar en modo protegido del 80286.

Los dos bits de menor peso, *RPL*, del registro segmento *RS*, actuando como selector, que aparecen en la figura 12-9, indican el *nivel de protección*. El bit *T/I* señala si se hace referencia a la Tabla de Descriptores Globales (*GDT*) o a la *LDT*. El *Índice*, que ocupa 13 bits, indica el número de descriptor seleccionado ($2^{13} = 8.192$). Dicho índice se multiplica por 8 para cubrir los 64 K de *GDT* y *LDT* y direccionar las entradas que son múltiplo de 8.

Para poder acceder a un descriptor concreto, sólo se necesita conocer la dirección base o de inicio de la *GDT* o de la *LDT*. De este cometido se encarga el contenido de un par de registros del sistema, el *GDTR* y el *LDT*. La figura 12-10 describe gráficamente la distribución de los bits de dicha pareja de registros. *LDT* actúa como un selector de un descriptor de la *GDTR*, que contiene la base, el límite y el acceso de la *LDT*, cuyos valores se cargan en el descriptor caché oculto, asociado a *LDT*.

El valor del *GDTR* no se altera prácticamente, puesto que los programas y los datos del Sistema Operativo mantienen constantes sus posiciones. Por el contrario, el valor de *LDT* cambia automáticamente, por acción directa de la MMU, cada vez que se produce una *comunicación de tarea*, como más adelante se explicará.

En resumen, el acceso a un descriptor se lleva a cabo según los siguientes algoritmos:

- Dirección de un Descriptor Global* = $8 \times \text{ÍNDICE} (\text{RS}) + \text{BASE} (\text{GDTR})$
- $\text{TI} = 0$.

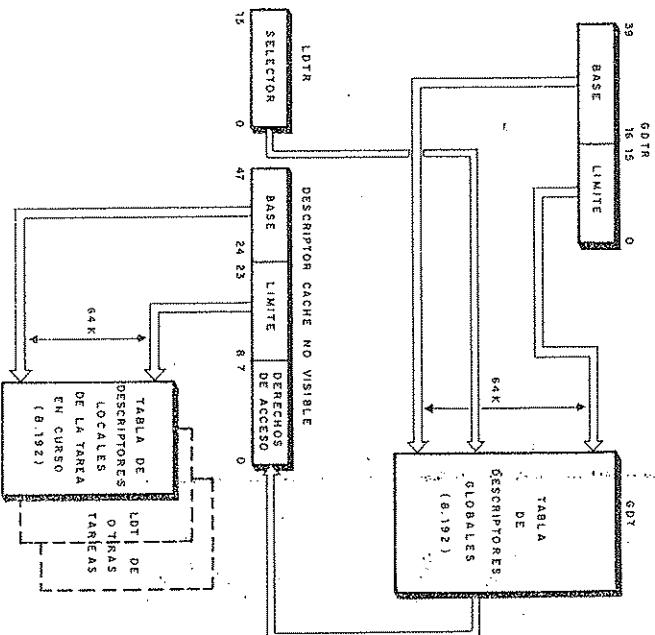


Fig. 12-10. Los registros del sistema GDT y LDT se encargan de contener el valor de la dirección base de las tablas GDT y LDT, respectivamente. LDT actúa como selector de un descriptor que contiene la base, el límite y los derechos de acceso de LDT, que se cargan en el descriptor caché no visible asociado a LDT.

b) Dirección de un Descriptor Local = $8 \times \text{INDICE} (\text{RS}) + \text{BASE} (\text{LDT})$

Recuérdese que el contenido del descriptor proporciona la dirección base del segmento en la memoria real, el límite o tamaño del segmento y los derechos de acceso que tiene asignados dicho segmento. Toda esta información se guarda en el descriptor cache del registro segmento utilizado.

El proceso de conversión de dirección virtual a dirección física sólo se realiza una vez, puesto que las restantes direcciones de ese segmento quedan definidas de forma inmediata por el descriptor cache. Como consecuencia de este proceso de traducción de direcciones, que dura un par de microsegundos, se consigue programar en un espacio virtual de 1 Gigabyte, aunque físicamente la memoria principal sólo disponga de un espacio de 16 Mbytes como máximo.

5.3.3 / Tendencias actuales en la arquitectura de microprocesadores

En la figura 12-11 se muestra gráfica y esquemáticamente la conversión de una dirección virtual a física y en la figura 12-12 el mecanismo que se sigue en el proceso de traducción de direcciones.

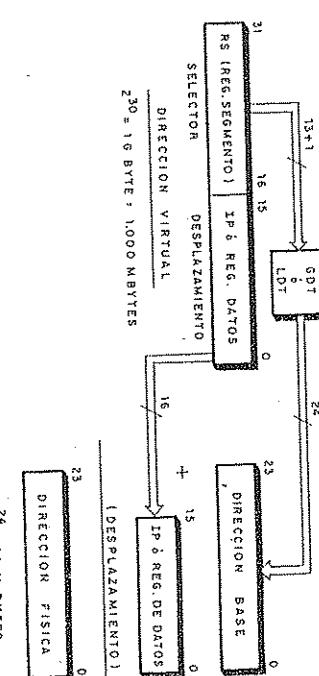


Fig. 12-11. Conversión de una dirección virtual de 32 bits a otra física de 24. A la dirección base contenida en el descriptor de la GDT o LDT, direccionalizada por los 14 bits formados por el índice (I31) y TI (1) de RS, se suma el desplazamiento de 16 bits.

Todo el proceso de conversión es transparente para el programador, el cual le parece que tiene a su disposición 1 Gigabyte de memoria. Esta "transparencia" es posible gracias a la labor de la MMU, que se encarga de las siguientes operaciones:

1. Se accede a la tabla correspondiente (GDT o LDT de la tarea en curso) para localizar la entrada asociada al segmento al que hace referencia la dirección virtual. En dicha entrada se halla el descriptor del segmento.
2. Se comprueba que la dirección está dentro de los límites tolerados y que está permitido el tipo de acceso (lectura, escritura, etc.) que establece el byte con los derechos de acceso.
3. Si el segmento referenciado se halla cargado en la memoria principal, se genera su dirección física de acuerdo con el contenido del descriptor. La información sobre la presencia o no del segmento en la memoria principal queda determinada por un bit de presente situado dentro del byte de los derechos de acceso.
4. Si el segmento referenciado no se halla cargado en la memoria principal, se produce una excepción de página o de segmento y se ejecuta una rutina que traslada dicho segmento desde la memoria virtual (disco) a la memoria principal (RAM).

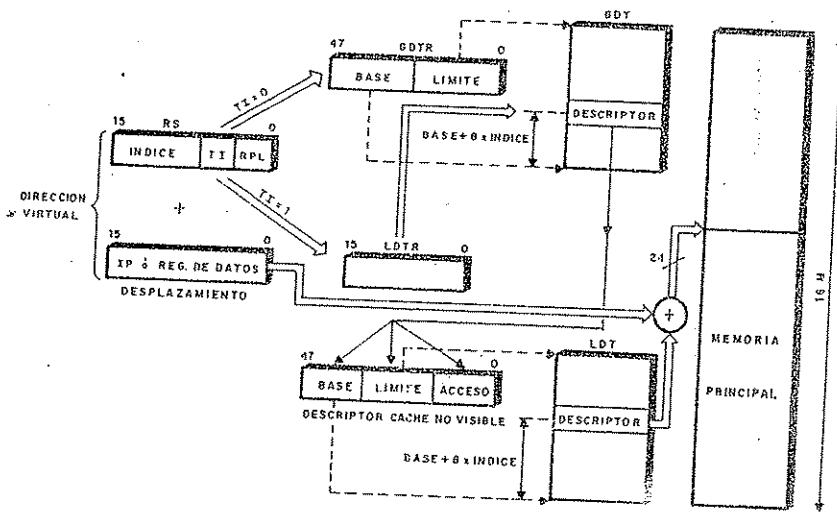


Fig. 12-12. Mecanismo seguido en la traducción de direcciones. La dirección virtual está formada por 13 bits del índice, 16 bits de desplazamiento y 2 bits de menor peso, RPL, del registro de segmento sólo especifican el nivel de privilegio.

12.4.3. Paso del modo real al modo protegido

El bit PE, que ocupa la posición extrema derecha de la "palabra de estado de la máquina" (MSW), sirve para habilitar el modo protegido, cuando se pone a 1. Al iniciarse el funcionamiento del 80286, después de un reset, el bit PE = 0 y el microprocesador trabaja en modo real, como si se tratase de un 8086 mejorado. Si operando en modo real se realizan las operaciones que luego se describen para pasar a modo protegido, la única manera de volver a regresar al modo real es haciendo un reset.

Antes de poner a 1 el bit PE de MSW, se deben cargar los registros del sistema GDTR e IDTR con dos direcciones físicas de 24 bits, que conforman el campo de "base" de ambos registros. Dichas direcciones apuntan el inicio de las tablas de descriptors globales y de descriptors de interrupción en la memoria principal. Los valores del campo "límite" en los registros GDTR e IDTR tienen una longitud de 16 bits, que permite una capacidad máxima de 64 Kbytes para cada una de las tablas. No obstante, como sólo hay 256 vectores de interrupción que controlan las 256 posibles rutinas de atención a interrupción, la tabla que dirige la dirección IDTR puede alcanzar una capacidad máxima de $256 \times 8 = 2$ Kbytes, dado que cada entrada de la tabla o descriptor está formado por 8 bytes.

Una vez cargado GDTR e IDTR y puesto a 1 el bit PE, se activa el modo protegido en el 80286 y, posteriormente, se suele realizar un JMP para borrar la cola de instrucciones precapturadas, seguido de una instrucción de habilitación del procesador numérico (80287) al modo protegido. Se trata de la instrucción FSENTPM.

Si se va a trabajar en un ambiente multitarea, hay que cargar, tras las operaciones anteriores, al Registro de Tarea TR con el valor inicial que señala el segmento de la memoria donde se guarda el contenido de los registros de la tarea en curso, o sea, el segmento del "estado de la tarea". Cada vez que se produce una commutación de tareas, se modifica automáticamente el valor de TR. El descriptor que selecciona TR está contenido dentro de la Tabla de Descriptors Globales.

Finalmente, hay que cargar el registro LDTR, que apunta de forma indirecta (a través de un descriptor de GDTR), a la Tabla de Descriptores Locales. Con objeto de aislar a cada tarea se asigna una zona de memoria virtual a cada una, la cual es映射 mediante la Tabla de Descriptores Locales, propios de cada tarea.

12.5. MULTITAREA

Se define la multitarea como la posibilidad de ejecutar diferentes tareas con una UCP, sin que se interfieran unas con otras. Mediante esta técnica se puede optimizar

el rendimiento de la UCP, al poder elegir otra tarea cuando la que se halla en curso se bloquee, por ejemplo, al entrar enfuncionamiento un periférico o producirse un error. También protege adecuadamente a las tareas autónomas.

Una sola UCP atiende diversos programas con diferentes contextos, en un mismo periodo de tiempo, aunque no simultáneamente. Para poder desarrollar la multitarea el 80286 emplea una *memoria virtual segmentada*, que favorece la *modularidad* y la *reubicabilidad* de los programas.

Recibe el nombre de *commutación de tareas* el proceso de transferencia desde una tarea a otra. Uno de los mayores obstáculos de la multitarea es la frecuente commutación de tareas, que en algunos casos, como en los sistemas multiusuario, se produce cada pocas centésimas de segundo. Sin embargo, gracias a la arquitectura del 80286, la commutación de tareas puede producirse con una sencilla instrucción del tipo *JMP* o *CALL*.

La commutación de tareas incluye un cambio de contexto, o sea, del estado de los registros. Todos los elementos que se alteran al cambiar de tarea están contenidos en el Segmento de Estado de la Tarea, que guarda toda la información necesaria para que la tarea interrumpida pueda reanudarse posteriormente. Cada tarea dispone de un *Segmento de Estado de la Tarea*, *TSS*, que almacena la información de la mayoría de los registros de la UCP. Cuando se produce una commutación de tareas, se salva el estado de la tarea saliente y se carga el de la entrada, iniciándose su proceso.

Todas las tareas inactivas disponen, además de sus segmentos de código y de datos, del segmento *TSS*, que guarda el estado de los registros, mientras permanece inactiva la tarea. Para poder dirigir la *TSS* se utiliza el registro *TR* (*Registro de Tarea*), que es un selector del *TSS* correspondiente. Dicho selector se encuentra en la *GDT* y se identifica con uno de los bits del byte de derechos de acceso. Cuando el *TR* se modifica, cambia inmediatamente el contenido del descriptor cache asociado al mismo, con los valores de la base, los derechos de acceso y el límite de desplazamiento desde el descriptor *TSS* indexado por el nuevo selector. Véase la figura 12-13.

A continuación se describe el proceso que se lleva a cabo cuando se produce una commutación de tareas:

1. Se ejecuta una instrucción *JMP* o *CALL* de salto o llamada a otro segmento, caracterizado por un *Segmento de Estado de Tarea (TSS)* de otra tarea diferente a la que se halla en curso.
2. El microprocesador reconoce la commutación de tareas y guarda los valores de los registros en el segmento *TSS* que direcciona el descriptor cache (parte oculta) del *TR* actual, es decir, salva el estado de la tarea que se está ejecutando. El nuevo selector de *TR* pasa a ser el indicado por el selector propuesto por el salto de *JMP* o de *CALL*.

54.2 / Tendencias actuales en la arquitectura de microprocesadores

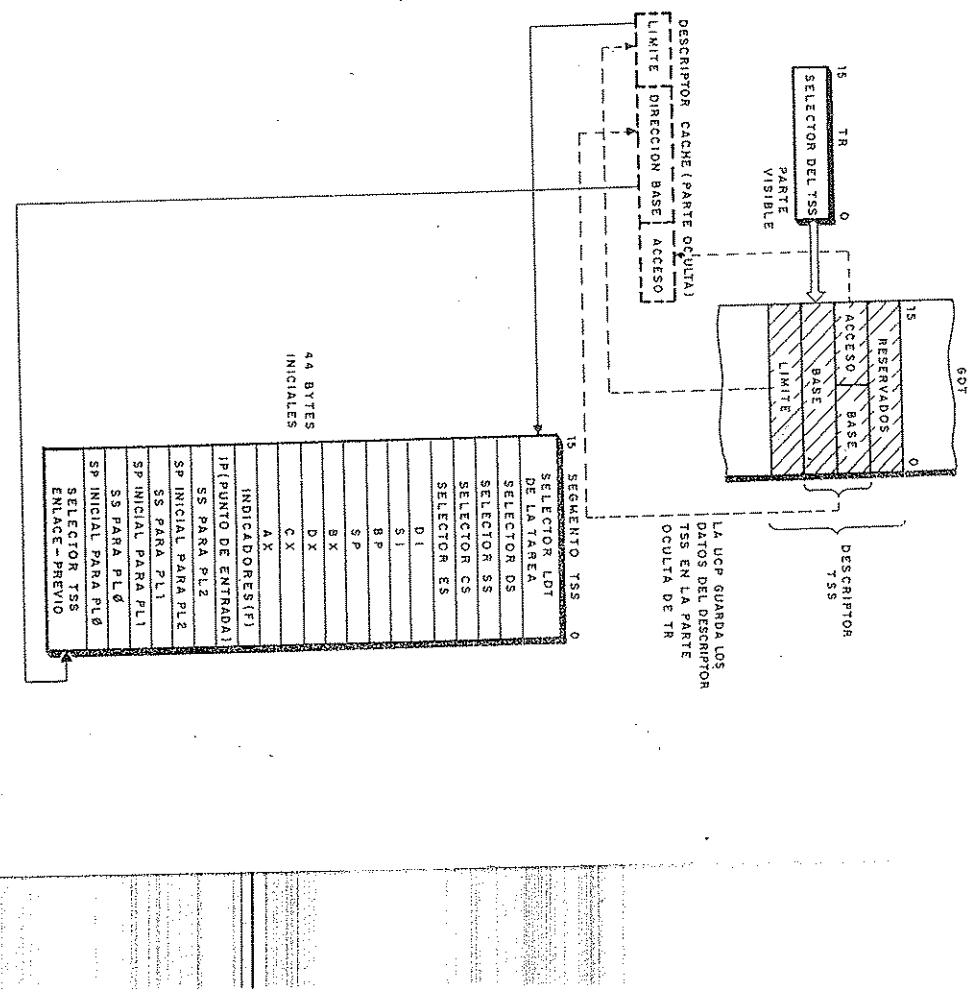
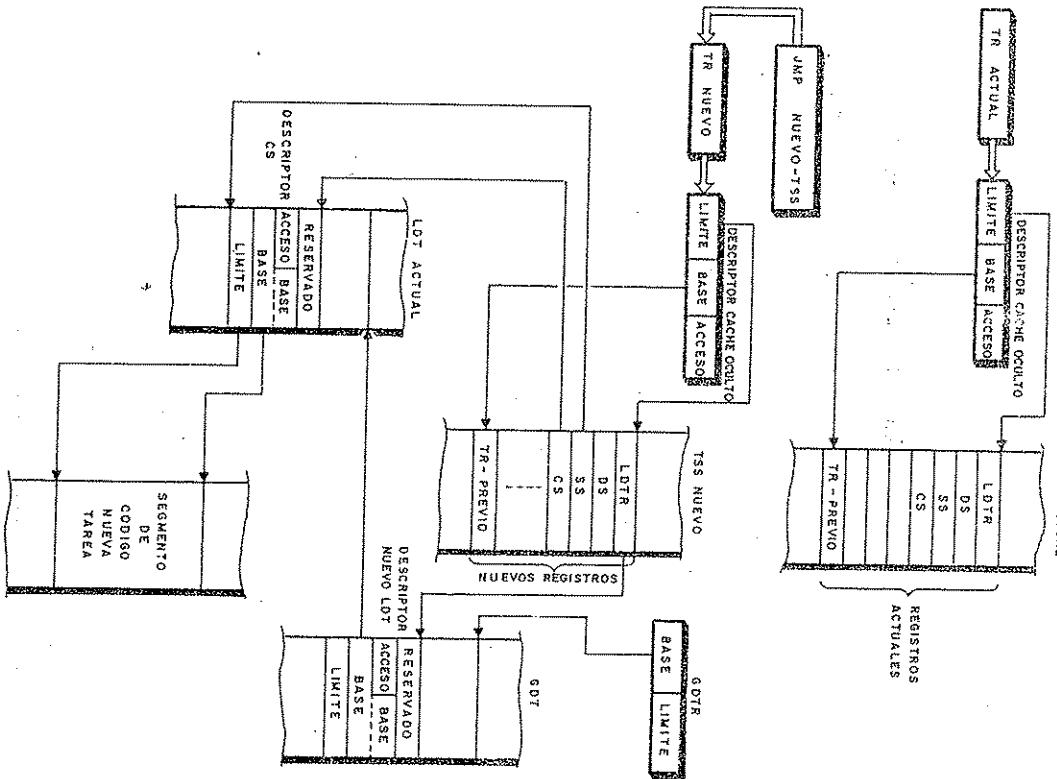


Fig. 12-13. El registro *TR* actúa como selector del descriptor del *TSS*, que se halla en la *GDT*. La parte oculta del *TR* se carga con los valores de la base y el límite del segmento *TSS* que corresponde con la tarea seleccionada.



Se cargan los registros con los valores correspondientes a la nueva tarea, según el nuevo descriptor TSS seleccionado. LDTIR será un selector a un descriptor global en la GDT, que se almacenará en el descriptor cache del LDTR y que direcciona el inicio de la LDT de la tarea actual.

Cuando se produce un *anidamiento* de tareas, queda señalado con el bit *N/T* de la palabra de serializadores (*F*). Cuando termina una tarea anidada se puede retornar a la tarea que la ha llamado, gracias a la *Palabra de Enlace Previo* que contiene el selector *TR* de la tarea que la ha llamado.

Finalmente, conviene indicar que existe un bit dentro del byte de derechos de acceso del TSS, que señala cuando está ocupado una tarea. Teniendo en cuenta dicho bit, se puede evitar la *recursividad de tareas*, aunque siempre es posible que se produzca la recursividad de programas. Figura 12-15.

12.6. PROTECCIÓN Y NIVELES DE PRIVILEGIO

El 80286 soporta una serie de mecanismos de protección en modo real, a los que se añaden otros más potentes en el modo protegido. Entre estas posibilidades se encuentra una muy importante que permite la determinación de distintos niveles de privilegios a las diferentes partes de un sistema.

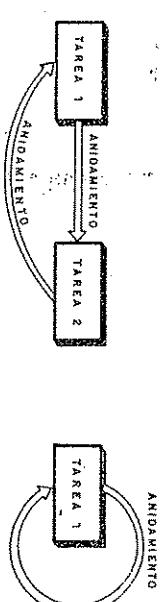


Fig. 12-14. Esquema gráfico que representa el proceso de "comunicación de tareas".

544 / Tendencias actuales en la arquitectura de microprocesadores

La UCP debe ser capaz de detectar cuando se sobrepasan los límites de un segmento definido; también debe conocer si se intenta ejecutar un grupo de datos en vez de instrucciones, etc. Para saber la UCP si se sobreponen los límites del segmento, solo hay que comparar el desplazamiento con el campo *límite* del descriptor. Por otra parte, dentro del byte de derechos de acceso del descriptor hay información sobre el tipo de segmento que se trata (datos o instrucciones), así como si se puede leer, leer y escribir, etc. La UCP al detectar cualquiera de estos posibles errores generará una interrupción que protegerá al sistema.

En el modo protegido el 80286 dispone de 4 *niveles de privilegio*, que van numerados del 0 al 3, de mayor a menor privilegio. Se puede usar un sólo nivel de privilegio, varios o todos. Cuando sólo se usa un nivel se elige el mayor, o sea, el 0. Normalmente se usan dos niveles, el 0 para el *supervisor* y el 3 para el *usuario*. En sistemas muy complejos se usan 3 y 4 niveles.

Estos niveles de privilegio permiten, por ejemplo, que no se bloquee el sistema aunque se produzcan errores en los programas de aplicación del usuario. Existen 4 reglas para la aplicación de los niveles de protección con objeto de que no se bloquee el sistema:

1. En cualquier momento el *nivel de privilegio actual (CPL)* debe coincidir con el del segmento de código que se está ejecutando (*DPL*). El valor *DPL* viene expresado en el byte de derechos de acceso y *CPL* en el registro *CS*.

2. El *nivel de privilegio del segmento de la pila* debe coincidir con el del segmento de código. Cada nivel de privilegio tiene su pila independiente.

3. El *nivel de privilegio* de los segmentos de datos será igual o menor que el del segmento de código.

4. Para acceder a una rutina de un segmento de código de nivel de privilegio superior hay que hacerlo a través de una puerta de llamada.

12.5.1. Puertas de llamada

Para acceder a una rutina de mayor privilegio al actual se dirige a una puerta de llamada, que redirecciona a esa rutina.

Las puertas de llamada son similares a los descriptores y están contenidos en la GDT. El acceso al segmento con nivel de privilegio superior se efectúa con el desplazamiento indicado en la puerta de llamada. Véase la figura 12-16. El "selección de destino" de la puerta de llamada sirve para localizar el descriptor del segmento de código con nivel superior.

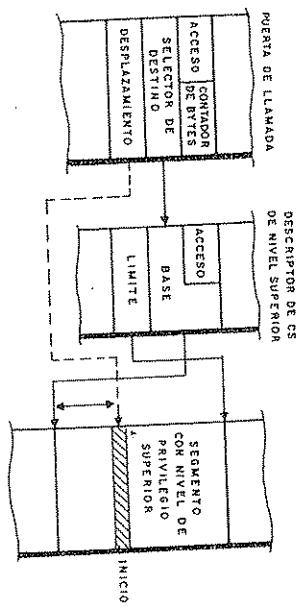


Fig. 12-16. Para acceder a una rutina con nivel de privilegio superior al actual, se utiliza una puerta de llamada, que redirecciona un descriptor de nivel superior CS.

El byte *contador de bytes*, que hay en la puerta de llamada, indica el número de parámetros que pasan de la pila de nivel de privilegio inferior a la superior, y puede alcanzar un máximo de 31 bytes.

Para realizar operaciones de E/S es necesario, al menos, igualar el nivel de privilegio mínimo, que viene especificado por la pareja de bits *IOPL*, contenidos en la palabra de los señalizadores de la UCP.

12.7. INTERRUPCIONES Y EXCEPCIONES

Tanto las interrupciones como las excepciones sirven para traspasar el control del programa que se está ejecutando a otra rutina que adquiere, en ese momento, una prioridad máxima.

Las interrupciones se producen desde el exterior del microprocesador, cuando se activan sus patitas NMI e INTR.

Sin embargo, las excepciones se generan dentro de la UCP, al detectarse algún tipo de error o condición especial. El 80286, además de las excepciones soportadas por el 8086 (error de división, error coprocesador y paso a paso), admite muchas más, tales como: Falta doble, código OP no válido, segmento no presente, violación del segmento de stack, violación de privilegio, etc.

En el modo protegido, las excepciones e interrupciones provocan la consulta de una serie de puertas que redirigen hacia una determinada rutina, que se encarga del tratamiento de la excepción o interrupción peticonaria. Estas puertas, cuyo número puede llegar hasta 256, tienen un tamaño de 8 bytes. La dirección base de

La Tabla de Puertas o Descriptores de Interrupción, IDT, se halla cargada en el registro del sistema IDTR, cuyo contenido se actualiza una sola vez, generalmente, cuando se entrega en el modo protegido, a partir de ese momento la UCP dirige la puerta correspondiente.

Las excepciones tienen reservadas las 32 primeras puertas de la IDT y su selección es automática. Cada tipo de excepción tiene su puerta correspondiente.

Las interrupciones generadas desde el exterior del microprocesador admiten dos variantes:

1. No mascarables, solicitadas a través de la puerta NMI, que producen un direccionamiento hacia la puerta número 2.
2. Mascarables, cuya solicitud se hace por la activación de la puerta INTR. Este tipo de interrupción puede no ser atendido, desactivando el bit IF de habilitación de las interrupciones, ubicado en la palabra de los señalizadores. Cuando se activa INT y están habilitados las interrupciones, la UCP lee el contenido del bus de datos en el que se ha depositado un byte desde el periférico solicitante, cuyo valor expresa el número de puerta seleccionada.

Las rutinas de interrupción se diseñan de acuerdo con el funcionamiento del sistema. El direccionamiento de dichas rutinas se expone gráficamente en la figura 12-17.

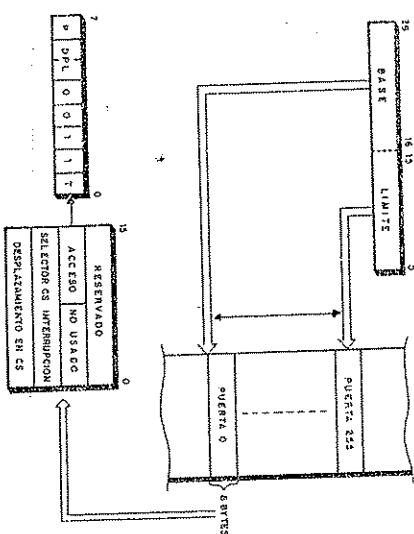


Fig. 12-17. Direccionamiento de las 256 puertas que dirigen a las rutinas de interrupción. Formato de una puerta de interrupción.

Las puertas pueden ser de tres tipos:

Puertas de interrupción

Se utilizan para atender las interrupciones provocadas desde periféricos externos. Deshabilita las interrupciones mascarables mientras se trata la interrupción en curso, poniendo el bit IF de la palabra de señalizadores a 0. Se salva en la pila el CS:IP y la palabra de señalizadores; además, si se ha producido transición de nivel, se salva el SS:SP.

Puertas de excepción, trampa o "trap"

Se usan para el tratamiento de las excepciones. Su funcionamiento es similar a las puertas de interrupción, salvo que éstas no deshabilitan las interrupciones mediante el bit IF.

Puertas de tarea

Ya se han citado al hablar de la "multitarea". Sirven para tratar a la interrupción como si se tratase de una tarea independiente, pero anidada a la que se trataba cuando se produjo la interrupción. Antes de ejecutar la tarea de interrupción se procede a realizar todos los pasos seguidos en la comutación entre tareas; también se activa el bit NT de anidamiento. Frente a la ventaja que supone el aislamiento de la tarea de interrupción respecto a la interrumpida, tiene el inconveniente de que es más lento el proceso de interrupción.

En el modo real las interrupciones y las excepciones direccionan también a la IDT, pero cada elemento que la compone sólo dispone de 4 bytes, que proporcionan la dirección de la rutina correspondiente que atiende a la interrupción o excepción peticionaria.

12.8. ANÁLISIS RESUMIDO DEL SISTEMA LÓGICO DEL 80286

Como este microprocesador es una continuación de los predecesores de INTEL, sólo se comentan brevemente sus características específicas.

12.8.1. Modos de direccionamiento

Además de los dos modos habituales de *direcciónamiento inmediato* (el operando está contenido en la propia instrucción) y *direcciónamiento por registro* (el ope-

rando está contenido en uno de los registros de la UCP), el 80286 dispone de otros 6 modos más para el direccionamiento a la memoria.

Un puntero de la memoria consta de dos palabras de 16 bits: La primera es el selector del segmento, que hace referencia a un espacio fijo de 64 K bytes en modo de direcciones reales, o un espacio variable en modo protegido, determinado en este caso por el descriptor correspondiente. La segunda palabra determina el desplazamiento dentro del segmento y se calcula de 6 formas diferentes que hacen referencia a los 6 modos de direccionamiento antes comentados.

Mientras el selector de segmento (CS, SS, DS o ES) suele quedar fijado implícitamente en la propia instrucción, el desplazamiento se calcula de acuerdo con uno de los 6 posibles modos de direccionamiento.

Modo de direccionamiento directo

El desplazamiento se proporciona directamente en la propia instrucción y debe tener una longitud de 16 bits.

Modo de registro indirecto

El desplazamiento está contenido de forma indirecta en uno de los registros SI, DI o BX.

Modo base

El desplazamiento se calcula sumando el contenido de un registro base (BX o BP) a un desplazamiento adicional que puede ser de 8 o 16 bits. Se aplica en el manejo de estructuras repetitivas de datos con elementos de diversos tamaños.

Modo indexado

El desplazamiento se calcula sumando un desplazamiento adicional y el contenido de un registro índice (SI o DI). Es útil para manejar tablas de datos ubicadas en direcciones fijas. El desplazamiento adicional indicará la dirección base de la tabla y el registro índice se irá incrementando según el tamaño de los elementos.

Modo indexado base.

El desplazamiento se obtiene de la suma de un registro base (BX o BP) con un registro índice (SI o DI). Es muy adecuado en el manejo de "arrays" de posición variable.

variable, donde el registro base indica la dirección inicial y el registro índice apunta a sus elementos.

Modo indexado base con desplazamiento

Sirve para manipular elementos de tablas incluidas dentro de otras estructuras. El desplazamiento se calcula sumando al contenido de un registro base, el de un registro índice y un desplazamiento adicional. El registro base apunta al comienzo de la estructura, el desplazamiento adicional la distancia al comienzo de la tabla dentro de la estructura y el registro índice señala los elementos correspondientes.

12.8.2. Juego de instrucciones

Todo el software de los microprocesadores 8086/8088 y 80186 es compatible con el 80286. Sin embargo, se pueden diferenciar tres grandes grupos de instrucciones:

1. Instrucciones básicas: iguales a las del 8086/8088 y 80186, sólo que potenciales en cuanto a velocidad de ejecución y modos de direccionamiento.
2. Instrucciones extendidas: Son propias del 80186 y del 80286, pero no están disponibles en el 8086/8088.
3. Instrucciones de control del sistema: propias del 80286 y que sólo son ejecutables en modo protegido.

Instrucciones básicas

a) De transferencia

MOV, PUSH, POP, PUSHA, POPA, LEA, LDS, LES, IN, OUT, XCHG Y XLAT.

b) De aritmética binaria

ADD, ADC, SUB, SBB, MUL, IMUL, DIV, IDIV, INC, DEC, NEG, CGW Y CXD.

c) De aritmética BCD

AAA, AAS, AAM, AAD, DAA Y DAS.

d) Lógicas

NOT, AND, OR Y XOR.

e) De desplazamiento y rotación

SHL, SHR, SAL, SAR, ROL, ROR, RLC y RCR.

f) De control del programa

CMP, TEST, LOOP, LOOPE, LLOOPZ, LOOPNE, LOOPNZ, JMP, JE, JO, JNZ, ..., JCXZ, CALL y RET.

g) De control del sistema

INT, INTO, IRET, CLI, STI, HLT, LOCK, WAIT, ESC, NOP, CLC, STC, CMCI, SAHF, LAHF, PUSHF y POPF.

h) De manipulación de cadenas

MOV\$, MOVSB, MOVSW, CMPS, CMPSB, CMPSW, SCAS, SCASB, SCASW, LODS, LODSB, LODSW, STOS, STOSB, SOTISW, CLD, STD, REP, REPE y REPNE.

Instrucciones extendidas

a) Instrucciones de E/S de cadenas

IN\$: Transfiere un elemento de 8 bits de una cadena (*IN\$B*) o de 16 bits (*IN\$W*), desde una puerta de entrada direccionada por el registro DX a la posición de memoria señalada por ES : DI.

OUT\$: Transfiere un elemento de 8 ó 16 bits de una cadena, desde una posición de memoria direcciónada por DS : Si a una puerta de salida apuntada por el registro DX.

Si a estas dos instrucciones se añade el prefijo REP se pueden transferir bloques de información entre memoria y una puerta. El número de bytes o palabras que se transfieren se indica previamente en el registro CX, el cual se decremente cada vez que se realiza una transferencia. Cuando CX = 0, finaliza la transferencia.

b) Instrucciones de alto nivel

Son instrucciones máquina del 80286 que poseen las características de algunas propias de los lenguajes de alto nivel.

ENTER: Sirve para crear la base de la pila que necesitan los módulos de los lenguajes de alto nivel; estructurados en bloques.

Posé dos operandos, el primero determina el espacio que se reserva a la pila a crear y a cuya base apunta el registro BP.

El segundo operando señala el nivel de anidamiento, o sea, el número de páginas a bases de otras pilas que se van a copiar de la base de la pila de procedimiento que llama al módulo actual. Si el valor del segundo operando es 0, significa que no hay anidamiento.

LEAVE: Es la instrucción que complementa a ENTER. Se coloca delante de una instrucción de retorno de procedimiento RETURN y los efectos son inversos. Los que provoca ENTER, el registro BP se carga con el valor de la base de la pila del procedimiento que llamó al saliente. Posteriormente, se elimina la información de la pila del procedimiento acabado.

BOUND: Se emplea para comprobar que el valor de un determinado registro se halla dentro de ciertos límites. Si dicho valor está fuera de esos límites se produce una interrupción del tipo 5.

El primer operando es el 'registro a comprobar' y el segundo contiene la dirección relativa de los dos valores límites.

Instrucciones de control del sistema

Este grupo de instrucciones sirve para manejar el sistema en modo privilegiado.

ARPL: El primer operando es una palabra (registro o variable en memoria) que contiene el valor de un selector. Y el segundo operando se trata de un registro de 16 bits. Con esta instrucción se evita el acceso de un programa a subrutinas con nivel de privilegio superior al suyo.

Si el campo RPL del primer operando es menor que el del segundo, el señalizador de cero se pone a 1 y se ignora el RPL al que tiene el segundo operando. En caso contrario, sólo se pone a 0 el señalizador cero.

CLTS: Pone a 0 el bit TS de la WSW. Dicho bit señala la realización de una comutación en tareas. Es una instrucción privilegiada que sólo puede ejecutarse en el nivel 0.

LAR: Cuando el descriptor asociado a la palabra indicada por el segundo operando (registro o memoria) es visible en el nivel de privilegio actual, se carga en los 8 bits de más peso del registro determinado por el primer operando, el byte de "derechos de acceso", mientras que en los 8 bits de menos peso se ponen en 0, pasando a 1 el señalizador cero.

LGDT - LDTR: Cargan el GDTR o al IDTR, respectivamente, con los 6 bytes siguientes a la dirección que señala el operando.

LLDT: El operando (registro o palabra de memoria) apunta a una entrada en la GDT, que hace referencia a una LDT. Cuando esto sucede, se carga el registro LDTR con el valor de la entrada de la GDT apuntada por el operando.

LMSSW: Carga en la MSW el valor del operando (registro o posición de memoria). Sirve para introducirse en el modo protegido.

LSL: Consta de dos operandos, el primero es un registro en donde se desea cargar el límite del segmento y el segundo hace referencia a un selector. Realiza la carga del límite del segmento cuando el selector del segundo operando referencia a un descriptor accesible en el nivel de privilegio actual; en este caso el señalizador Z se pone a 1.

LTR: Carga al registro TR con el valor indicado por el operando.

SGDT - SIDT: Cargan el valor de GDTR o IDTR, respectivamente, en los 6 bytes de la memoria señalados por el operando.

SLDT: Carga el valor del registro LDTR en la posición indicada por el operando. Esta posición debe ser una entrada de la GDT correspondiente a una LDT.

SMSW: El valor de MSW se carga en el operando de la instrucción.

STR: Carga el valor de TR en el operando que acompaña a la instrucción.

VERR - VERW: Son dos instrucciones que verifican si un segmento se puede leer o escribir, respectivamente desde el nivel de privilegio actual. Si el segmento es accesible al señalizador cero (Z) pasa a 1. Así se evitan errores en el acceso a dichos segmentos.

EJERCICIOS

Ejercicio 12.1

Dentro de las fases en las que se han desarrollado los microprocesadores, indicar cuál es a su juicio aquélla que más importancia ha tenido.

Ejercicio 12.2

¿Cuáles son las ventajas, con referencia a los registros de trabajo, del microprocesador 80286 frente al 8086?

Ejercicio 12.3

A un sistema basado en el 80286 se le quiere implantar una RAM de 2 K X 16, una ROM de 16 K X 16 y, dentro del mapa de E/S, 256 puetas destinadas al conexionado con los periféricos, indicar la memoria principal y de E/S que aún queda sin usar, expresada en bytes.

554 / Tendencias actuales en la arquitectura de microprocesadores

Ejercicio 12.4

Explicar en 5 pasos el mecanismo de funcionamiento de la MMU.

Ejercicio 12.5

Explicar razonadamente la función de la Tabla de Páginas, cuando se usa la paginación en el manejo de la memoria.

Ejercicio 12.6

¿Qué partición del microprocesador 80286 está destinada a prevenir a otros procesadores del sistema, que tomen el control del bus, cuando actúa como maestro del 80286?

Ejercicio 12.7

¿Por qué valor divide el 80286 la frecuencia que se le aplica desde el exterior a la patita CLOCK?

Ejercicio 12.8

¿Qué otros señalizadores existen en el Registro de Señalizadores, R, del 80286, diferentes a los que tiene el 8086? ¿Cuál es su misión?

Ejercicio 12.9

Explicar, razonadamente, la forma con la que se determina una dirección de memoria en el 80286, cuando trabaja en modo real.

Ejercicio 12.10

Explicar el cometido de la Tabla de Descriptores Globales y el de la Tabla de Descriptores Locales.

Tendencias actuales en la arquitectura de microprocesadores / 555

12

Ejercicio 12.11

Explicar el desarrollo de una commutación de tareas y las causas que pueden originar la misma.

Ejercicio 12.12

¿Qué relación debe existir entre el nivel de privilegio de la Pila y el del segmento de código?

Ejercicio 12.13

¿Qué diferencia existe, en el cálculo del desplazamiento, cuando se emplea el modo de direccionamiento indexado y el indexado base?

Ejercicio 12.14

El 80286 dispone de tres instrucciones de corte similar a algunas típicas en los lenguajes de alto nivel: ENTER, LEAVE y BOUND. Buscar las analogías correspondientes con lenguajes como el FORTRAN, COBOL, PASCAL, C y BASIC.

Teleinformática

13

13.1. INTRODUCCIÓN

La Teleinformática es la combinación de dos técnicas: la Informática y las Telecomunicaciones.

El fin que persigue la Informática es la automatización del tratamiento y del almacenamiento de la información.

La Telecomunicación trata de la difusión de información a través de ondas y electromagnetismo, haciendo posible la radio, la televisión, el teléfono, etc.

El objetivo de la Teleinformática es aunar las posibilidades de las dos técnicas mencionadas para poner en comunicación los sistemas informáticos.

Una red de transmisión de datos es una estructura formada por los medios físicos y lógicos requeridos para soportar las necesidades de la Teleinformática, es decir, proporciona el soporte para establecer la conexión entre diferentes equipos informáticos.

Se pasan a comentar tres tipos de redes:

13.1.1. Redes de Área Local (LAN)

Comunican e integran dispositivos de tratamiento de datos entre sí, en un área acotada, para conseguir el intercambio de información y la compartición de los recursos.

Una red de área local es un sistema de comunicaciones que permite el diálogo entre diversos usuarios a gran velocidad (más de 10 Mbit/s) en un entorno limitado (pocos kilómetros).

Recibe el nombre de Ofimática la técnica que intenta integrar los recursos disponibles en los centros de trabajo y oficinas.

13.1.2. Redes Públicas o de Largo Alcance

Son redes de transmisión de datos de servicio público y, consecuentemente, compuestas en régimen de alquiler. En general este tipo de redes exigen que la señal esté modulada sobre una portadora sinusoidal. Las redes más importantes de este grupo son la RTC (Red Telefónica Comunitada) y la RETD (Redes Especiales para Transmisión de Datos).

La RTC conecta a los equipos informáticos a través de un modem y soporta señales analógicas, que deben "modularse" y "demodularse", la modulación convierte las señales digitales en analógicas, tarea que lleva a cabo el elemento "transmisor". La demodulación es la operación inversa y es soportada por el "receptor". Para possibilitar la transmisión y recepción conjuntamente se utiliza el modem, que realiza ambos tipos de conversión de señales.

Las Centrales de Comunicación de la RTC controlan las conexiones entre origen y destino, la tarificación, el control de tráfico, etc.

13.1.3. Red Digital de Servicios Integrados (RDSI)

La RETD evolucionó hacia las RDSI, gracias a los logros tecnológicos referidos hacia la codificación digital de señales analógicas, como la voz y la imagen, y al uso de tecnologías avanzadas en la transmisión de señales (fibra óptica y centrales digitales).

13.2. EL MODELO OSI

La intercomunicación de sistemas informáticos conlleva la interconexión entre éstos y las redes de transmisión de datos. Es imprescindible estandarizar dicha interconexión para evitar problemas técnicos de adaptación entre diferentes sistemas. Con este fin se crearon organismos internacionales, como el CCITT (Comité Consultivo Internacional de Telegrafía y Telefonía), el ISO (International Standards Organization) y otros.

En 1983 se sintetizaron las especificaciones que tanto el CCITT como ISO habían propuesto para la conexión de sistemas informáticos dando lugar a la recomendación OSI (Interconexión de Sistemas Abiertos).

Un sistema abierto es aquél que permite la interconexión de otro sistema que cumpla las mismas normas. El problema de la interconexión se desarrolló sobre una estructura compuesta por 7 niveles, que permite dividirlo en partes. Figura 13.1.

El modelo OSI separa las aplicaciones de los usuarios de las funciones precisas para establecer la comunicación entre ellos. Dichas funciones se organizan en una pirámide jerárquica.

Los 7 niveles que existen entre el "usuario final" y el "medio físico", cumplen los siguientes requisitos:

NIVEL	FUNCION	SUBFUNCION	
7	APLICACION		APLICACION
6	PRESENTACION		
5	SESION		SISTEMAS INFORMATICOS
4	TRANSPORTE		
3	RED		
2	ENLACE	CONTROL DE ENLACE LOGICO TECNICAS DE ACCESO AL MEDIO	MEDIOS DE COMUNICACION
1	FISICO	CODIFICACION DE DATOS ACCESO AL MEDIO DE TRANSMISION	

Fig. 13.1. Los 7 niveles en que se dividió la problemática de la interconexión de sistemas abiertos, de acuerdo con la recomendación OSI.

- 1.º En cada nivel hay una serie de funciones que hacen uso de los servicios del nivel inferior, al mismo tiempo que dan servicio al nivel superior.
- 2.º En cada nivel existe un protocolo que fija las reglas de intercambio (convenios en el formato de la información, gestión de errores, etc.).

De forma general, los 7 niveles pueden clasificarse de la siguiente forma:

- Los tres niveles inferiores se destinan al medio de comunicación, cubriendo los aspectos físico y lógico.
- Los 3 siguientes niveles se refieren a los sistemas informáticos.
- El último nivel soporta las relaciones con las aplicaciones de los sistemas.

13.2.1. Nivel 1º: Físico

Controla la transmisión de bits sobre el canal de comunicación, activando y desactivando las conexiones físicas. Además, define las características eléctricas mecánicas y funcionales de la conexión con el medio físico.

La misión de este nivel es conseguir la máxima independencia del medio físico de transmisión.

Este nivel puede subdividirse en dos subniveles:

- A) Codificación de datos.
- B) Acceso al medio físico de transmisión.

୩

El nivel físico soporta los siguientes servicios:

- Conexiónado físico.
 - Puntos terminales.
 - Identificación del circuito de datos.
 - Sincronización.
 - Comunicación de errores.
 - Parámetros sobre el nivel de calidad de la comunicación

3.2.2. Nivel 2.º: Enlace

Controla el establecimiento del enlace lógico de los datos, así como su liberación; también regula el control de errores, el flujo entre extremos del enlace lógico y la sincronización.

1200 JOURNAL OF CLIMATE

- ### B) Técnicas de acceso al medio.

Los protocolos de este nivel establecen los enlaces de dirección entre los nodos.

los códigos de dirección, a la dirección y recuperación de fallos y al orden de los datos transmitidos.

Así como en el nivel anterior la unidad de información era el "bit", en este es la "trama" o el "byte", según el protocolo empleado. La trama es una estructura de información que está compuesta por varios campos.

3.2.3. Nivel 3.º: Red

Se encarga del control de la circulación de los mensajes a través de la red. Realiza el multiplexado de las conexiones, el direccionamiento de los mensajes y controla las congestiones de información.

Los protocolos ubicados en este nivel se encargan

- 1) Control del tráfico interno de la red.
 - 2) Comunicación adecuada entre los módulos de la red.

Asegura la correcta transmisión de los datos. Se encarga del establecimiento y liberación de la conexión de transporte, la segmentación del mensaje en paquetes y la optimización de la red.

560 / Teleinformática

୩

El protocolo correspondiente a este nivel es el primero que se dirige de extremo a extremo en redes extendidas, razón por la que deben encargarse de la detección y recuperación de errores y del control del tráfico de la información.

13.2.3. Nivel 3: Session

Determina el acceso a la red por los usuarios. Se encarga de la identificación de interlocutores para conexión-desconexión y de la definición del tipo de transmisión.

13.2.6. Nivel 6.º: Presentación

Controla la conversión de los códigos de presentación y la interpretación de la información intercambiada. Se encarga de la compresión, expansión, cifrado y descifrado de los textos.

13.2.7. Nivel 7.9: Aplicación

Este nivel es el encargado de soportar la adaptación entre el usuario y el sistema telemático. Gestiona el contador del uso de los recursos para confeccionar la factura de cada usuario.

El usuario final situado en este nivel se desentiende de las funciones atribuidas a los niveles inferiores.

Los niveles más inferiores son los más dependientes del contexto en el que se desarrolla la intercomunicación y su normalización es mucho más importante que en los restantes. Por este motivo la normalización del modelo OSI está más consolidada en los 3 primeros niveles y es respetada por todos los fabricantes y diseñadores.

13.3. MEDIOS PARA LA TRANSMISIÓN DE DATOS

Se encargan de transportar la información de un punto a otro, es decir, constituyen el soporte físico de la comunicación.

LOS MEJORES LIVROSPERMITAS DE HISTORIA SON:

- El par de hilos trenzados.
 - El cable coaxial.

En un futuro también se incluirán entre los medios de transmisión en Teleinformática las ondas de radio, infrarrojos, microondas, etc.

13.2.4. Nivel 4.2: Transporte

liberación de la conexión de transporte, la segmentación del mensaje en paquetes y la optimización de la red.

13

El par de hilos trenzados

Es el medio más común usado en las transmisiones telefónicas por ser el primero que se emplea. Consta de dos hilos de cobre recubiertos de material aislante, que se trenzan para reducir las interferencias.

Está diseñado para la transmisión de frecuencias bajas (voz), puesto que aunque es económico y de fácil instalación, es muy sensible a las interferencias externas.

Las velocidades normalizadas de transmisión de datos en las líneas telefónicas suelen ser 300, 900, 1200 y 2400 bits/segundo.

Más suele aplicarse en las redes locales de bajo coste, alcanzando velocidades de 10 Mb/s.

El cable coaxial

Está formado por un hilo central de cobre, recubierto de un aislante y una malla metálica exterior que actúa como pantalla protectora ante las interferencias externas.

Permite la transmisión a frecuencias superiores a la del par, siendo más elevada su inmunidad ante las interferencias.

La aplicación más usual del cable coaxial es en televisión. También se usa en teléfono para transmitir varias conversaciones por un mismo cable.

En Teleinformática se usan cables coaxiales de 50 ohmios en la red Ethernet.

La fibra óptica

Sirve para transmitir la "luz", de características especiales, para su posterior tratamiento en comunicaciones. Se trata de un delgado filamento de un material que permite el paso de la luz por su interior, estando recubierto por un material aislante.

Entre las ventajas más destacadas de este medio se encuentran:

1.a) La seguridad en la transmisión de datos, puesto que no queda afectada por las interferencias externas.

2.a) Gran velocidad de transmisión (100 Gigabits/s).

3.a) Grandes distancias sin atenuación (100 km).

El problema del precio y la optimización de la tecnología impiden su empleo masivo en Teleinformática. No obstante es uno de los recursos más valiosos para el desarrollo de la RDSI y de las LAN de alta velocidad.

13.4. LA TRANSMISIÓN

Para poder transmitir la información, ésta debe traducirse a señales capaces de ser transportadas por el medio. Así los mensajes se traducen en información analógica o

562 / Teleinformática

digital en forma de señales eléctricas, ópticas, etc. La estandarización se dirige hacia dicha traducción, así como a las características de los medios de comunicación, dentro del nivel 1 (físico) del modelo OSI.

Las señales digitales sólo admiten un número discreto de estados, generalmente dos (información binaria).

Las señales analógicas admiten un número infinito de estados, en cuyo caso interesa conocer la frecuencia con la que se repite cada ciclo. Se mide en Herzios.

13.4.1. Banda Base, Banda Ancha y Banda Portadora

La transmisión en Banda Base codifica la información y las señales resultantes (corrientes eléctricas o pulsos de luz) se envían directamente al medio de transmisión.

En Banda Ancha se modula una señal portadora con la información a transmitir. Utilizando diferentes bandas de frecuencia se forman canales independientes que permiten transmitir varios mensajes simultáneamente por el mismo medio. Así pueden enviarse conjuntamente voz, imagen, datos, etc.

En la transmisión en Banda Portadora se envían los datos modulados sobre una señal portadora de elevada frecuencia. Sólo soporta un canal con una mayor inmunidad a las interferencias que con la Banda Ancha.

13.4.2. Modulación

En la modulación se manejan dos señales: la portadora que es una señal periódica con frecuencia y amplitud constantes y la moduladora, que contiene la información que se desea transmitir y que sirve para "modular" la portadora, es decir, para cambiar alguna de las características de la portadora. El proceso de demodulación es inverso al de modulación, en él a partir de la señal modulada recibida se obtiene la señal moduladora, extrayéndola de la portadora.

En Teleinformática interesa trabajar con modulación digital, lo cual provoca cambios bruscos de la portadora, por lo que recibe el nombre de "modulación por desplazamiento". Hay varios tipos.

Modulación por desplazamiento de amplitud (MDA)

A cada uno de los dos estados digitales se le asigna una determinada amplitud. En general al estado 1 se le asigna la amplitud de la portadora y al estado 0 una amplitud nula. Figura 13-2.

Modulación por desplazamiento de frecuencia (MDF)

Según el estado (binario) de la modulación varía la frecuencia de la portadora. Para un estado la portadora tiene una frecuencia y para el otro estado otra. Figura 13-3.

13

SEÑAL PORTADORA
— — — — —

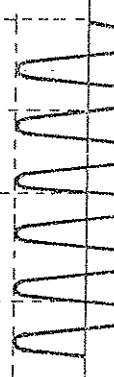


Fig. 13-2. Modulación por desplazamiento de amplitud (MDA).

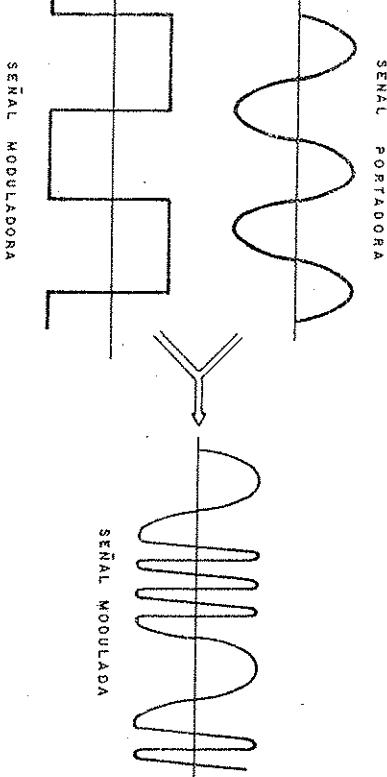


Fig. 13-3. Modulación por desplazamiento de frecuencia (MDF).

13

Como la señal enviada puede descomponerse en trozos de distintas frecuencias, es posible enviar dos señales MDA con distinta frecuencia portadora.

Como la amplitud en MDA es constante, se pueden eliminar fácilmente ciertas perturbaciones que afectan a la señal.

Modulación por desplazamiento de fase (MDP)

En este caso la fase de la portadora toma diversos valores en función de la forma de la moduladora. Figura 13-4.

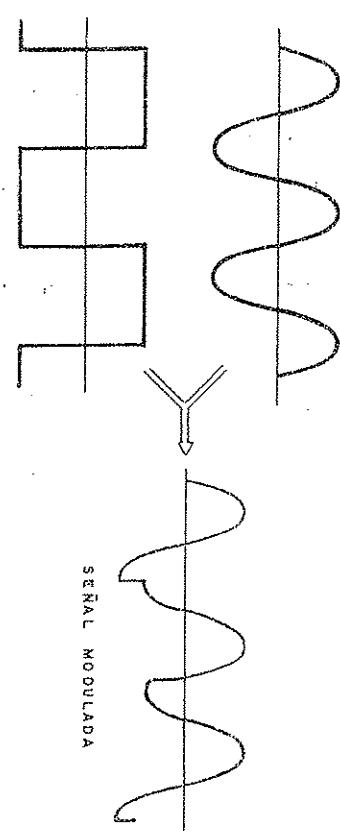


Fig. 13-4. Modulación por desplazamiento de fase (MDP).

En la transmisión de la información hay ciertos problemas que deben ser corregidos en la medida de lo posible: el ruido y la attenuación.

El ruido está provocado por perturbaciones externas (aleatorias) que suman su energía a la de la señal distorsionándola. La attenuación se produce siempre que se absorbe una señal en un medio de transmisión; la señal lleva cierta energía y el medio

13.5. CODIFICACION DE DATOS Y SINCRONIZACION

Codificar los datos es traducir los niveles 1 y 0 de la información binaria a señales que tengan los niveles necesarios para ser entregadas a los medios de transmisión directamente o modulados.

El receptor debe conocer exactamente cuando comienza y finaliza la transmisión, a nivel de bit. La sincronización de cada bit puede realizarse con o sin reloj.

Cuando existe un reloj incorporado, se codifica la información de tal forma que el receptor extrae la señal de reloj a partir de la línea de datos. Recibe el nombre de forma asincrónica, porque no hacen falta relojes en el emisor y receptor.

La transmisión sin reloj precisa de un reloj en el emisor y otro en el receptor, ambos sincronizados. También puede obtenerse mediante una línea, independiente de la de datos, por la que se envía una señal de reloj.

Se describen someramente algunos métodos de codificación.

NRZ

Hay dos niveles de tensión, el de mayor nivel representa el 1 y el otro el 0. Cuando el segundo nivel es nulo, se habla de NRZ unipolar y cuando es igual al primero, pero negativo se habla de NRZ polar.

RZ

Es parecido al método anterior, pero con una vuelta al nivel nulo de tensión en la mitad del intervalo. Así se evitan series largas con el mismo estado, que hacen perder la sincronización. Hay RZ polar y unipolar.

AMI (RZ bipolar)

Los niveles 0 se representan por una tensión nula y los 1 por una tensión alternativamente positiva y negativa. Se produce error si hay dos tensiones seguidas con el mismo signo.

NRZI

Los 1 producen cambio o transición de tensión y los 0 no. La transición provoca el paso de tensión positiva a nula o viceversa.

13.6. REDES

Una red está formada por nodos y enlaces. Como nodos pueden actuar los computadores y como enlaces las conexiones entre ellos.

El enlace físico hace referencia a los elementos directos que existen entre los nodos, mientras que el enlace lógico puede establecer una conexión indirecta a través de otros nodos, simulando así al enlace físico.

La forma geométrica que resulta de la conexión de los nodos con los enlaces recibe el nombre de topología de la red.

13.6.1. Red Punto a Punto y Red Multipunto

En la Red Punto a Punto hay una conexión directa entre un punto y otro, manteniendo una velocidad constante. En la Red Multipunto hay una línea a la que se conectan los distintos puntos.

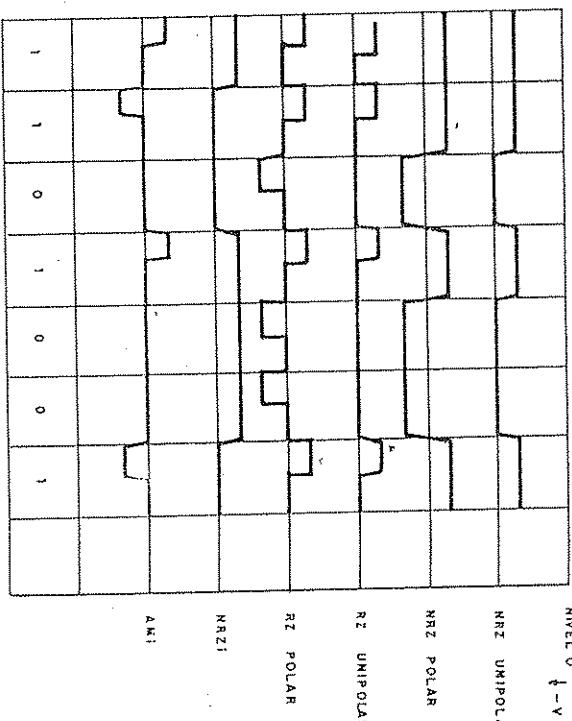


Fig. 13-5. Diversos métodos de codificación de datos.

Manchester

La señal de reloj está incorporada y consiste en una transición en flanco ascendente, cuando hay un 1 y un flanco descendente cuando hay un 0. Figura 13-5.

La primera red es muy sencilla y fiável, pero tiene un alto coste cuando se supera un determinado número de nodos y, por otra parte, se obtiene un bajo rendimiento. Figura 13-6.

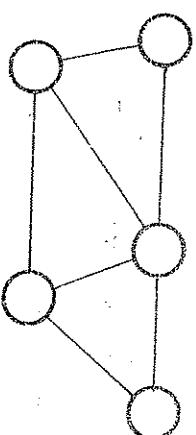


Fig. 13-6. Red Punto a Punto.

La Red Multipunto es económica en lo que respecta al medio físico empleado, pero plantea problemas cuando dos o más nodos intentan acceder a la línea simultáneamente. Figura 13-7.

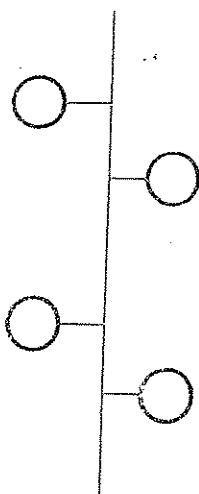


Fig. 13-7. Estructura de la Red Multipunto.

13.6.2. Escrutinio o "polling"

Se usa el método de escrutinio o "polling" cuando se conectan todos los nodos a uno central, que generalmente suele ser el encargado de controlar las comunicaciones. El nodo central va atendiendo alternativamente a cada nodo, consultando si está ocupado o no y si dispone de algún mensaje a enviar. Figura 13-8.

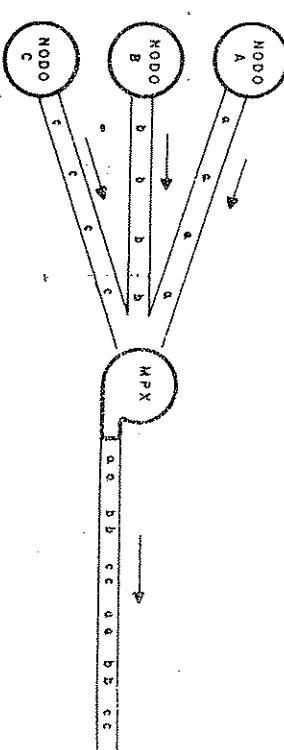


Fig. 13-8. Estructura de red que permite la técnica de escrutinio o "polling".

2.9) Multiplexores por división en el tiempo (TDM)

La información de cada línea de baja velocidad se manda o recibe a intervalos regulares de tiempo. Cada intervalo se corresponde con una línea de baja velocidad, formando un ciclo continuo. Figura 13-9.

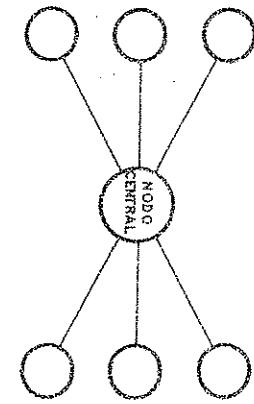


Fig. 13-9. Multiplexado por división en el tiempo.

13.7. MULTIPLEXADO Y CONCENTRACIÓN

Los multiplexores son dispositivos que establecen la conexión entre varias líneas de baja velocidad por un lado, y una línea de alta velocidad por el otro.

Se describen tres tipos de multiplexores.

2.9) Multiplexado por división estadística en el tiempo (SDTM)

En este caso, y con objeto de disminuir los tiempos muertos que existían en el multiplexado anterior, el tiempo de utilización de la línea de alta velocidad se ajusta "dinámicamente" según la actividad de las líneas de baja velocidad. Se llaman multiplexores inteligentes.

3.º Multiplexado por división en frecuencia (FDM)

Los multiplexores de este tipo asignan distintas frecuencias a cada línea de baja velocidad, haciendo preciso el uso de un modem. Figura 13-10.

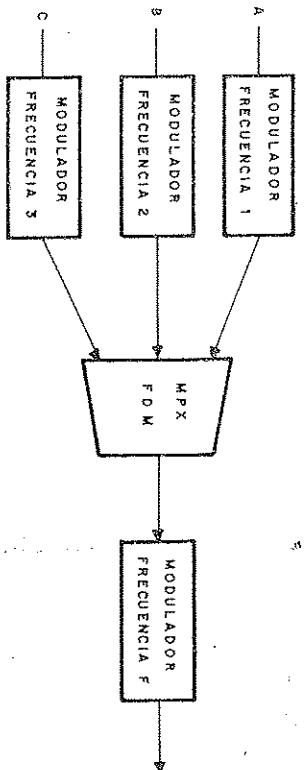


Fig. 13-10. Multiplexado por división en frecuencia (FDM).

Los concentradores son dispositivos similares a los multiplexores, con inteligencia y memoria propia. Se encargan de la gestión del tráfico procedente de muchas líneas de baja velocidad, solucionando las congestiones y regulando el flujo de información entre los terminales.

13.8. TECNICAS DE ACCESO AL MEDIO

Las principales técnicas de acceso al medio son:

1. Punto a punto.
2. Protocolo de ranuras fijas. Se comenta como ejemplo el protocolo ALOHA.
3. Protocolo CSMA/CD (Carrier Sense Multiple Access with Collision Detect).
4. Protocolo de paso de testigo o "Token Passing".

La topología condiciona el protocolo de acceso al medio y así, el protocolo CSMA/CD se implementa sobre una topología en anillo, mientras que el de paso de testigo lo hace sobre una topología en bus.

El protocolo ALOHA se denomina así porque comenzó a usarse en las redes de radio Aloha que comunicaban ordenadores de las islas Hawái. También se llama técnica de acceso por ranuras, aunque ya no se utiliza.

En este protocolo se divide el tiempo en "ranuras", que se asignan una para cada usuario. Se desaprovecha mucho tiempo cuando hay usuarios que no usan sus ranuras.

Una variante del protocolo Aloha, diseñada para evitar las pérdidas de tiempo mencionadas, es la de ranura vacía o anillo de Cambridge. Por una estructura en anillo circula un código previamente establecido que indica "cabecera de ranura vacía". Cuando un nodo recibe dicho código y tiene información a enviar, lo cambia por otro de "cabecera de ranura llena" y procede al envío de la información. La información circula por todo el anillo hasta que vuelve al nodo emisor, quien cambia a "cabecera de ranura vacía", pasando el sistema a quedar a disposición de los restantes nodos.

El IEEE (Institute of Electrical and Electronics Engineers) es un organismo internacional que dispone de normas que afectan a las redes locales. Su documento 802 hace referencia a los niveles más bajos de las comunicaciones, similares a los niveles 1 y 2 del modelo OSI.

El punto 802.3 se destina al protocolo CSMA/CD, en el cual cualquier nodo que detecte la señal de "línea libre" puede transmitir un mensaje. El acceso es muy rápido.

Se denomina colisión al problema que surge cuando varios nodos intentan acceder al mismo tiempo a la red. Su solución se consigue dejando de emitir todos los nodos cuando se detecta una colisión. Tras un periodo de tiempo aleatorio de espera se hace un nuevo intento, tratando de espaciar el máximo los reinicios de los diversos nodos. Este proceso es adecuado cuando el tiempo de transporte máximo es menor que la longitud del paquete.

Xerox ha desarrollado la red Ethernet, basada en el protocolo CSMA/CD, que obtiene un aprovechamiento de la línea cercano al 90%.

Las normas 802.4 y 802.5 se aplican al protocolo de paso de testigo o "Token Passing". En este caso se hace circular un testigo (grupo de bits que representan el derecho a usar la línea) continuamente. Solamente el nodo que posea el testigo le será posible enviar mensajes a la red. Se suele designar un nodo "monitor" encargado de controlar la presencia y circulación del testigo.

El nodo que deposita un mensaje en la red, retira el testigo cuando nuevamente lo vuelve a recibir. Los nodos son capaces de leer la dirección del destino del mensaje y recogerlo en caso de correspondientes.

13.9. NIVEL DE DATOS. LAS TRAMAS

Dentro del nivel de enlace debe existir una estructura de datos. La trama es una estructura que contiene ciertos datos, tales como la dirección de destino, verificación de errores, etc. Se puede considerar a la trama como la unidad de información para el nivel de enlace.

13

Para el nivel de enlace la recomendación OSI destina el protocolo HDLC (High-Level Data Link Control). Dicho protocolo posee un único formato para la trama. Que consta de los campos siguientes:

- Flag de inicio.
- Dirección de destino.
- Identificación del tipo de trama (de comandos y respuestas, de información o de supervisión).
- Deteción-verificación de errores.
- Flag final.

13.9.1. Deteción y control de errores

Entre los diversos métodos existentes para implementar los campos de detección y verificación de errores en la trama, se comentan los más importantes.

A) VRC (Vertical Redundancy Check)

A este método también se llama "bit de paridad", porque consiste en añadir un bit de paridad por cada byte transmitido.

El bit que se añade depende de los bits del byte al que acompaña. Así cuando se emplea la "paridad par", se hace que el conjunto de bits "1" que hay en el byte junto al bit de paridad den una cifra par. En la paridad impar, el conjunto de los bits 1 del total de los 9 bits transmitidos, debe ser impar.

Este método no es capaz de determinar cuál ha sido el bit erróneo y no es válido cuando se alteran 2 bits.

B) LRC (Longitudinal Redundancy Check)

Forma caracteres de bits de paridad siguiendo el método anterior, pero los envía como un byte más de información.

C) CRC (Cyclic Redundancy Check)

La información binaria se considera como un polinomio. Dicho polinomio se divide por otro previamente establecido y el resto que resulta se añade a la información. El receptor de la información realiza la misma operación y compara el resto que obtiene con el que recibe. Se produce error cuando ambos restos no coinciden.

Se trata de un método bastante fiable puesto que el error se detecta siempre si el número de bits erróneo es inferior al número de bits del resto.

13.9.2. Corrección de errores

Los métodos de corrección de errores suelen estar en función de los métodos empleados en la verificación así como en la economía, puesto que la mejor corrección consiste en repetir el proceso de envío y recepción.

Según los métodos las acciones que se pueden tomar son las siguientes:

- a) Corrección a partir de los datos disponibles en el nodo receptor (en la información pueden existir datos repetidos).
- b) Si la pérdida de información que ocasiona el error no es grave, se ignora éste último.
- c) Corrección del error mediante la información contenida en los campos de verificación.
- d) Solicitar al emisor que repita el envío.

13.10. NIVEL DE RED: TOPOLOGIAS

Se denomina topología a la estructura geométrica que forman los nodos y las líneas que los unen.

Las principales topologías utilizadas en las redes locales son:

- En estrella.
- En bus.
- En anillo.

Topología en estrella

Se trata de una red punto a punto en la que todos los nodos se unen a uno central que gestiona de la circulación de información entre todos ellos. Figura 13-11.

Los nodos pueden poseer características diferentes y existe una gran flexibilidad en la reconfiguración de la red. En caso de avería de un nodo, la misma queda aislada y limitada, con excepción del fallo en el nodo central que detiene al sistema.

El nodo central consulta a los exteriores mediante la técnica de escrutinio o "polling".

Topología en bus

Se trata de una red multipunto, en la que todos los nodos se conectan a una línea que permite la conexión entre ellos. Figura 13-12.

Es una red muy flexible y barata, pudiendo conectar muchos nodos, según el tipo de línea de transmisión que se emplee. El mecanismo de conexión entre los nodos se complica cuando éstos poseen diferentes características.

13

mismo tiempo que la regenera. Este proceso disminuye la tasa de errores. Figura 13-13.

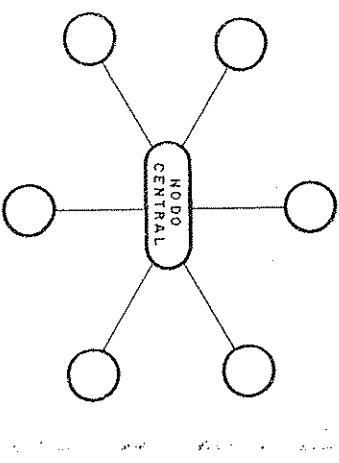


Fig. 13-11. Topología en estrella de una red.

avería en el medio de transmisión supone la parada total del sistema.
La topología en bus suele emplear como método de acceso al medio el CSMA/CD.

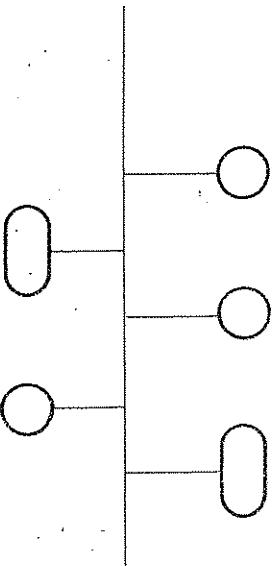


Fig. 13-12. Topología en bus.

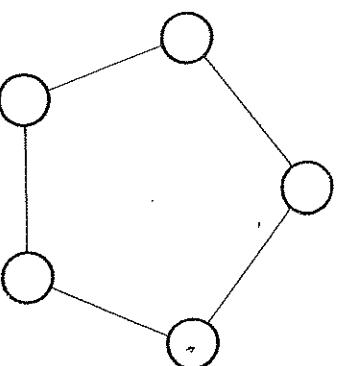


Fig. 13-13. Topología en anillo.

El gran problema de la topología en anillo radica en el hecho de que cuando un nodo se avería se detiene todo el sistema. Para evitar este inconveniente aparecieron las redes de doble anillo, que constan de una red en anillo más una línea cerrada que conecta con la red en un punto entre cada nodo.

Esta topología no es recomendada cuando existen muchos nodos, puesto que el paso de la información por cada nodo supone un retraso de tiempo.

13.1.1. LA NORMA X.25

Como conclusión de este capítulo se describe someramente la recomendación X.25, que elaborada por el CCITT, sirve para estandarizar el acceso a la RTC por parte de un usuario (terminal, ordenador personal, etc.).

Afecta a los tres primeros niveles del modelo OSI.

- Para el nivel 1 o físico se proporcionan las recomendaciones X.21 y X.21 bis del propio CCITT.
 - Para el nivel 2 (de enlace) se recomienda el empleo del protocolo HDLC.
 - Para el nivel de red, o sea, el 3 se establece el protocolo de control de red o de paquetes. El paquete consiste en un conjunto de tramas junto a unos campos específicos (cabecera, final, etc.).
- Los nodos se conectan uno tras otro formando un círculo cerrado, de manera que la información que deposita el emisor pasa por todos los nodos hasta encontrar al destino. En consecuencia, la información lleva la dirección del nodo destino, de forma que cada nodo reenvía la información recibida que no le corresponde, al

Topología en anillo

Los nodos se conectan uno tras otro formando un círculo cerrado, de manera que la información que deposita el emisor pasa por todos los nodos hasta encontrar al destino. En consecuencia, la información lleva la dirección del nodo destino, de forma que cada nodo reenvía la información recibida que no le corresponde, al

Bibliografía

- Andrews, M., *Principles of Firmware Engineering in Microprogram Control*, Rockville, Md., Computer Science Press, 1980.
- Angulo, J. M., *Microprocesadores 8086, 80286 y 80386*, Ed. Paraninfo, 1990.
- Angulo, J. M., *Microprocesadores de 32 bits. El gran salto*, Ed. Paraninfo, 1986.
- Angulo, J. M. y Zapater, C. E., *Introducción a la Informática*, Ed. Paraninfo, 3.ª edición, 1988.
- Angulo, J. M., *Electrónica Digital Moderna*, Ed. Paraninfo, 9.ª edición, 1988.
- Angulo, J. M., *Memorias de Burbujas Magnéticas*, Ed. Paraninfo, 1982.
- Angulo, J. M., *Microprocesadores. Diseño práctico de sistemas*, Ed. Paraninfo, 3.ª edición, 1986.
- Asitley, D. W., *A Method for Large Systems Performance Prediction*, TR00 1773, IBM System Development Division, Poughkeepsie, N. Y., Set 1968.
- Baase, S., *VAX-11 Assembly Language Programming*, Prentice Hall, 1983.
- Baer, J. L., *Computer Systems Architecture*, Pitman, 1980.
- Booth, T. L., *Digital Networks and Computer Systems*, 2.ª edición, John Wiley & Sons Inc., 1978.
- Borrell, P. L., *Microprocessor Bus Structures and Standards*, IEEE Micro, vol. 1, febrero, 1981.
- Boyce, J. C., *Digital Computer Fundamentals*, Prentice Hall, 1977.
- Chen, T. C., *Overlap and Pipeline Processing (Introduction to Computer Architecture)*, 2.ª edición, Science Research Associates, 1980.
- Deitel, H. M. y Deitel, B., *Computers and Data Processing*, Academic Press, 1985.
- Deschamps, J. P. y Angulo, J. M., *Diseño de Sistemas Digitales*, Paraninfo, 1989.

- De Miguel Anasagasti, P., *Fundamentos de Computadores*, Ed. Paraninfo, 1988.
- Digital Equipment Corporation, *Hardware Handbook VAX*, 1982.
- Doran, R. W., *Computer Architecture: A Structured Approach*, Academic Press, 1979.
- Fairclough, D. A., *A Unique Microprocessor Instruction Set*, IEEE Micro, págs. 8-18, mayo 1982.
- Fernández, G. Y Sáez Vacas, F., *Fundamentos de los Ordenadores*, Depto. de Publicaciones de la E. T. S. I. T., Madrid, 1978.
- Fisher, W. P., *Microprocessor Assembly Language Draft Standard*, IEEE P694. Computer, págs. 96-109 diciembre 1979.
- Flynn, M. J., *Trends and Problems in Computer Organization*, Proc. IFIP Congress, págs. 3-10, 1974.
- Foster, C. C. Y otros, *Measures of Op. Code Utilization*, IEEE T. on Computers, págs. 582-584, mayo 1971.
- Foster, C. C. e Iberall, T., *Computer Architecture*, 3.ª edición, Van Nostrand Reinhold Company, 1985.
- Galan, C. Y Cordero, F., *Teleinformática. Introducción, panorámica y perspectivas*, Ed. Paraninfo.
- Garside, R. G., *The Architecture of Digital Computers*, Clarendon Press, 1980.
- Gibson, J. C., *The Gibson Mix*, TR00 2043. IBM System Development Division, Poughkeepsie, N. Y., junio, 1970.
- Gómez, S.; Alvarez, E. Y Angulo, J. M.ª, *Sistemas Multiprocesadores*, Ed. Paraninfo, 1988.
- Gutiérrez, J. L., Echevarría, M., Calvo, J. C., Pérez, A. Y Gutiérrez, J. J., *Diseno práctico de una UCP de propósito general*, Departamento de Arquitectura de ordenadores de la Facultad de Informática de la Universidad de Deusto, 1986.
- Hamacher, V. C., Vranesic, Z. G. Y Zaky, S. G., *Computer Organization*, 2.ª edición, McGraw Hill, 1984.
- Hamming, R. W., *Coding and Information Theory*, Prentice Hall, 1980.
- Hewlett Packard, *Arquitectura de Alta Precisión*, Documentación técnica, 1985.
- Hockney, R. W. Y Jeshope, C. R., *Parallel Computers, Programming and Algorithms*, Adam Hilger Ltd, 1981.
- Hoste, F., *Redes locales para empresas*, Ed. Arcadia.
- Huffman, D. A., *A Method for the Construction of Minimum Redundancy Codes*, Proc. IRE, págs. 1098-1101, setiembre, 1952.
- Hwang, K., *Computer Arithmetic*, John Wiley & Sons Inc., 1979.
- Hwang, K. Y Briggs, F. A., *Computer Architecture and Parallel Processing*, McGraw Hill, 1984.
- Matick, R., *Computer Storage System & Technology*.
- Kogge, P. M., *The Architecture of Pipelined Computers*, McGraw Hill, 1981.
- Langley, G., *Telecomunicación básica*, Ed. Paraninfo.
- Lippmann, A. G. Y Wright, G. S. L., *The Architecture of Small Computer Systems*, 2.ª edición, Prentice Hall, 1985.
- Mehradier, J. P., *Estructura y Funcionamiento de los Computadores Digitales*, Editorial AC, 1975.
- Minsky, M. L., *Computation of Finite and Infinite Machines*, Prentice Hall, 1967.
- Morris Mano, M., *Arquitectura de Computadores*, Editorial Dossat-Prentice Hall, 1983.
- Myers, G., *Advances in Computer Architecture*, John Wiley & Sons Inc., 2.ª edición, 1981.
- Rábago, J. F., *Guía práctica de redes locales*, Ed. Anaya Multimedia.
- Sabesta, R. W., *VAX-11. Structured Assembly Language Programming*, Editorial Benjamin/Cummings, 1984.
- Satyannarayanan, M., *Multiprocessors. A Comparative Study*, Prentice Hall, 1981.
- Stevorek, D. P., Bell, C. G. Y Newell, A., *Computer Structures: Principles and Examples*, McGraw Hill, 1981.
- Tanenbaum, A. S., *Organización de Computadores: Un enfoque estructurado*, 2.ª edición, Prentice Hall, 1985.
- Valero, M.; Llacería, J. M.ª Y Beovide, R., *Supercomputadores*, Mundo Electrónico, noviembre, 1986.
- Wulf, W. A., *Compilers and Computer Architecture*, Computer, vol. 14, julio 1981.

Índice Alfabético

A	B	C
Acceso directo a Memoria, 7.2, 7.7		Cadena, 11.4.1
Memoria Multipuerta, 7.7.1		Cadena de predicción, 11.4.4
Robo de ciclo, 7.7.2, 7.2.0		Cámara de TV, 9.2.4
Acoplamiento fuerte, 11.8		Cambio de signo, 3.6.1
Acoplamiento débil, 11.8		Canal de entrada/salida, 7.21
Acumulador, 3.1, 6.1.2		Flotante, 7.21
Adición (ver suma)		Multiplexor, 7.21
Alfanumérico, 2.4		Simplic., 7.21
Am 20203, 3.9.2		Carácter, 2.2
AP-120B, 11.6.1		Cepo, 6.13
Anticipación de acarreo, 3.6.3.9		Choques, 11.4.2
ASCII, 2.4		Cerrojo y clave, 4.20.2
	Ciclo de máquina, 6.2	Ciclo de bus, 8.1
	Cinta perforada, 9.2.1, 7.10	Código de barras, 9.2.7.4
Banco de registros, 3.1, 6.3.3, 10.3.7	CISC, 1.3.2, 5.2.1	Corrección, 2.18.3, 7.21
Banda magnética, 9.6	Codificación de campos, 6.10	Hamming, 2.17
BCD, 2.11	Código detector, 2.17 a 2.21	Periodo, 2.18
BCDIC, 2.4	de Barras, 9.2.7.4	Polinómico, 2.20
Bifurcación retardada, 11.4.4	Corrector, 2.18.3, 7.21	Coma flotante, 2.12 a 2.15
Binario Puro, 2.7, 2.7.1	Hamming, 2.17	Mantisa entera, 2.13
Bit implícito, 2.14	Periodo, 2.18	Mantisa fracción, 2.14
Booth, 3.7.3	Polinómico, 2.20	Normalización, 3.6.3.11 y 2.14
Bus, 1.1, 7.3.1, 7.5, 8	Coma flotante, 2.12 a 2.15	Complemento Restringido, 2.8
Ciclo partida, 8.5	Mantisa entera, 2.13	Computadores
Control, 8.7	Normalización, 3.6.3.11 y 2.14	de Juego de Instrucciones Com-
Jerarquía, 8.9	Complemento Restringido, 2.8	plejo, 1.3.2
Longitud, 8.8	Computadores	de Juego de Instrucciones Redu-
Niveles de especificación, 8.8	de Juego de Instrucciones Redu-	cido, 1.3.4
Normalizado, 8.3	Vectoriales, 11.5	Paralelo, 8.4
Paralelo, 8.4		Byte, 2.2

Computadores array, 11.6
A dos, 2.9
A la base, 2.9
A uno, 2.8

Contador de programas, 6.1.3
Cray-1, 11.5.1
Cronogramas, 6.5, 7.7.2
CRT, 9.3.3

D

Daisy-chain, 7.9.1, 7.24.1, 7.24.2
Datos Numéricos, 2.5 a 2.15
Decodificación de instrucción, 6.5.1
Desempaqueado, 2.11
Desplazamiento, 3.4
Aritméticos, 3.4.3
Circular, 3.4.2
Concatenados, 3.4.4
Lógicos, 3.4.1
Desplazamiento de dirección, 5.6
Detección de errores, 2.17 a 2.21
Digitalizador, 9.2.5
Direccionamiento
 Directo absoluto, 5.5, 10.15.2
 Directo relativo, 5.6, 10.15.4
 al contador de programa, 5.6.1
 a registro base, 5.6.2
 a registro índice, 5.6.3, 10.15.5
 Indirecto, 5.7, 10.15.6
 Inmediato, 5.4, 10.15.1
 Modos de, 5.3, 10.15, 12.8.1
 Modos de, 4.14, 4.4, 4.5.3
 Dirección física, 4.14, 4.4, 4.5.3
 Dirección lógica, 4.14
Discos, 9.7
Cabeza fija, 9.7
Cabeza móvil, 9.7
Formato, 9.7.1
Tiempo de acceso, 9.7.2
Velocidad de transferencia, 9.7.3
División, 3.8
Con signo, 3.8.3
Con restauración, 3.8.1
Sin restauración, 3.8.2

EBCDIC, 2.4, 2.11
Empaquetado, 2.11
Encadenamiento, 7.9.1.1
Ensamblador, 5.22

Entrada/salida programada, 7.2, 7.3, 7.17
EPROM y EEPROM, 4.10.4
Estado
 Palabra, 7.24.2, 10.3.8, 10.3.7
 Señalizadores de, 3.1, 6.1.2
Excepción, 6.13, 10.4.1
Exceso Z, 2.10
Exponente, 2.12
Extensión de signo, 3.6.2

F

Fábrica, 7.24.2, 10.3.8, 10.3.7
Fase, 6.4
Fieldata, 2.4
FM, 9.5.2

H

G

I

L

M

N

O

P

R

S

T

V

W

X

Z

Tambor, 9.3.1.1
Instrucciones, 5
Aritméticas, 5.13, 10.6.2

Dinámicas, 4.5.3.2
Direccionamiento, 4.5.3
Entrelazada, 11.4.3
EPROM y EEPROM, 4.10.4

Ferrita, 4.8
Hilo plateado, 4.9
Intermedia, 4.3
Jerarquía de, 4.3
Lectura destructiva, 4.6.1.3
Línea de retraso, 4.11

Magnéticas, 9.5
Banda magnética, 9.6
Burbujas, 9.8

Códigos, 9.5.1
Discos (ver discos)
Medio de grabación, 9.5.1

Tambores (ver disco)
Medio o soporte, 4.5.1

Modos de acceso, 4.6.2

Opática, 9.9

Paginada, 4.15, 12.2.2.1

Correspondencia directa, 4.15.1

Correspondencia asociativa, 4.15.2

Película delgada, 4.9

Permanente, 4.6.1

Principal, 4.3, 4.7, 6.1.1, 6.3.1, 10.5,

Propagación, 4.5.3.2, 4.11

Protección, 4.20, 10.8.2

Punto de, 4.1

RAM estática, 4.10.1

RAM dinámica, 4.11.2

Refresco, 4.6.7.4, 4.11.2

ROM, 4.10.3

Segmentada, 4.16, 12.2.2

Segmentos paginados, 4.17, 12.2.2.3

Semiconductores, 4.10

Tambor de silicio, 4.12

Transductor, 4.5.2

Velocidad, 4.6.3

Virtual, 4.14, 10.8.1, 12.2.2

Volátil, 4.6.1.2

NFM, 9.5.2

NICR, 9.2.7.1

Microfilm, 9.1.4

Microinstrucción, 6.13, 6.2, 6.8, 6.10

Horizontal, 6.10

- Vertical, 6.10
 Microprocesador, 12
 de 32 bits, 12.2
 Evolución, 12.1
 Unidad aritmética lógica, 10.3.1
 Microprograma, 6.8.1
 MILID, 11.2, 11.8
 MIMD, 11.2
 MMW, 12.2.2
 Modelo de ejecución, 5.20
 MFP, 11.7.1
 Multiplicación, 3.7
 Booth, 3.7.3
 Por suma y desplazamiento, 3.7.1
 Por sumas y restas, 3.7.2
 Microinstrucción, 1.3.2
 Microinterfaz, 6.11.1
 Microburdo, 6.11.2
 Microsubrutina, 6.11.2
 Multiplicación, 3.7
 Booth, 3.7.3
 Por suma y desplazamiento, 3.7.1
 Por sumas y restas, 3.7.2
 Rápida, 3.7.4
 Multiprocesador, 11.1, 11.8
 M68000, 5.9.2
 N
 Neumann, 1.2
 Normalización de montaje, 3.5.3.11
 NRZ, 9.5.2
 NRZI, 9.5.2
 O
 OCR, 9.2.7.2
 OMR, 9.2.7.3
 Operación
 Cambio de signo, 3.6.1
 de desplazamiento (ver desplaza-
 miento)
 Diádica, 3.5
 División (ver división)
 Elemental, 5.2
 P
 Palabra, 2.2
 Paralelismo, 11.1
 Pares, 11.4.2, 11.4.4
 PDP-11, Entrada/Salida, 7.2.4.1
 PE, 9.5.2
 Períodos, 6.4
 Acceso directo, 9.7
 Almacenamiento, 9.4
 Control, 7.8
 Entrada, 9.1, 9.2
 Entrada y Salida, 9.1
 Salida, 9.1, 9.3
 Periodos, 6.4
 Pipe-line, 1.3.4, 11.1
 Concepto, 11.4
 Estructura, 11.4.3
 Pipe-line, 1.3.4, 11.1
 Plotter, 9.3.2
 Prioridades, 7.9
 Centralizada, 7.9.2
 Gestión distribuida, 7.9.1
 Híbrida, 7.9.2.1
 Procesadores matriciales, 11.1, 11.2,
 11.7.1
 Procesadores vectoriales, 11.3
 Asociativos, 11.7.2
 S
 \$1, 11.8.1
 SBI, 7.24.1.1, 10.2
 Scroll, 9.3.3
 Secuenciador, 6.1.3
 Sistólicos, 11.8.2
 Programa almacenado, 1.2
 Programa canal, 7.2.1
 PROMI, 4.10.3
 R
 RAM, 4.10.1 y 4.10.2
 Ratón, 9.2.5
 Red de interconexión, 11.8
 Registros de borde, 4.20.1
 Repetitorio de instrucciones (ver ins-
 trucciones)
 Representación de la información, 2
 Alfanumérica, 2.4
 8CD, 2.11
 Binario puro, 2.7, 2.7.1
 Capacidad, 2.2
 Coma flotante (ver coma flotante)
 Complemento (ver complemento)
 Datos Numéricos, 2.5 a 2.15
 Estructuras de datos, 2.22
 Exceso Z, 2.10
 Métodos posicionales, 2.6
 Resolución, 2.2
 Sistema de residuos, 2.16
 Tamaño privilegiado, 2.2
 Tipos, 2.3
 Residuos, 2.16
 Resta (ver suma)
 RISC, 1.3.4, 5.21
 Robo de ciclo, 7.7.2, 7.20
 ROM, 4.10.3
 RS-232-C 8A
 RZ, 9.5.2
 T
 Tabla de páginas, 4.15.1
 Tableta gráfica, 9.2.2
 Tambor de silicio, 4.12
 Tarjeta perforada, 9.2.1
 Terminales de teleproceso, 9.1
 Test and Set, 4.20.1
 TLB, 4.17
 Transferencia elemental de E/S, 7.1
 Asíncrona, 7.5.2, 8.5
 Ciclo completo, 8.6.1
 Ciclo parido, 8.6.2
 Síncrona, 7.6.1, 8.6
 Trap, 6.13
 Trasductor, 4.5.2
 Tubo de rayos catódicos, 9.3.3
 U
 UCP, 1.2, 6.1
 UNIBUS, 7.2.4, 7.7.2-10.10, 10.2
 Unidad Aritmética/Lógica, 3, 6.1.2,
 Segmentación (ver pipe-line)