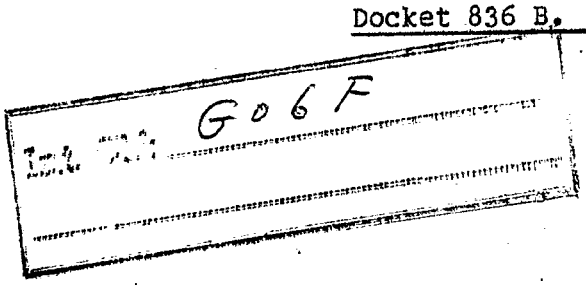
433.381

PATENTE DE INVENCION

Docket 836 B.



Memoria Descriptiva

sobre:

PERFECCIONAMIENTOS EN CALCULADORAS ELECTRONICAS.

==========

Solicitante: Ing. C. OLIVETTI & C., S.p.A., entidad italiana, residente en Via G. Jervis 77, 10015 Ivrea, Italia.

==========

La presente invención se refiere a una calculadora electrónica con equipo para poner a punto programas ejecutables que comprende una primera memoria para almacenar instrucciones y datos de dichos programas ejecutables, una unidad central

M - 2

para tratar dichos programas, un elemento de conmutación para definir selectivamente un primer modo de operación de dicha calculadora para tratar los citados programas ejecutables y un segundo modo de operaciones para tratar un programa de puesta a punto para la puesta a punto de los citados programas ejecutables, y un teclado para introducir en la citada primera memoria información relativa a los programas ejecutables durante el primer modo de operación.

En el campo de la elaboración de datos se conoce la necesidad de probar un programa antes de su utilización con el fin de eliminar los errores y las contradicciones originados durante la preparación del borrador de dicho programa.

En los elaboradores de gran capacidad de cálculo y de memoria, existen normalmente programas de prueba o de puesta a punto ya preparados y adaptados para proporcionar al operador todas las facilidades que le permiten seguir la elaboración del programa que se prueba y comprobar visualmente los posibles errores.

Es evidente que estos programas de puesta a punto, el actuar en el programa de prueba, deben existir simultáneamente a este último en la memoria del elaborador, por lo que sólo los elaboradores de grandes dimensiones ofrecen esta posibilidad.

Por otra parte, en el caso de elaboradores de pequeñas dimensiones en los que la capacidad de la memoria está dimensionada para contener el programa de dimensión máxima entre los que corresponden a la aplicación específica del elaborador (por ejemplo, la aplicación a problemas contables), el programador no puede probar el programa preparado.

En efecto, la tendencia de los fabricantes de estos elaboradores es la de proporcionar los programas necesarios al usuario, puestos a punto previamente.

Esta tendencia encuentra su justificación en el hecho de que un aumento de la memoria para albergar los programas de puesta a punto incidiría de modo negativo en el coste del elaborador; por otra parte, el usuario compraría un elaborador con un bajo rendimiento a causa de la parte de memoria inutilizada.

La evidente desventaja para el usuario, que se deriva de esta tendencia, es la absoluta falta de flexibilidad del elaborador adquirido ya que el usuario no puede aportar por sí solo la más pequeña modificación a los programas suministrados con el elaborador.

Esta rigidez de prestaciones del elaborador pone el usuario en la situación de depender totalmente del proveedor del elaborador para cualquier evolución de las prestaciones vinculadas a una modificación de programas.

Por consiguiente, el usuario se ve obligado a solicitar al proveedor la modificación de los programas, cosa que supone largos tiempos de espera y elevados costes. Para eliminar estos inconvenientes, el usuario se ve obligado a modificar por sí mismo los programas y solicitar a los centros externos de cálculo la puesta a punto de las modificaciones introducidas. También en este caso el usuario tiene que esperar mucho tiempo y pagar un coste suplementario.

Se conoce además una minicalculadora que tiene un aparato para poner a punto programas ejecutables. Este aparato comprende un panel de puesta a punto que no puede accionar el operador sino solo el programador. Este panel

incluye un conmutador para conmutar las operaciones de la minicalculadora del modo normal al modo de puesta a punto. Se proporciona también un grupo de conmutadores, cada uno de ellos asociado a una operación particular de puesta a punto, como por ejemplo la visualización de los registros de la memoria de trabajo, la escritura en la memoria, el funcionamiento paso a paso, etc.

Además de este panel hay otro panel para introducir los datos y las direcciones en la memoria, que incluye también una visualización para visualizar únicamente dos bytes de 8 bits de la memoria en código binario.

Este aparato de puesta a punto tiene fundamentalmente dos desventajas, la primera de las cuales es que exige unos dispositivos específicos que se utilizan únicamente en el modo de puesta a punto y no durante el modo normal. Estos dispositivos específicos aumentan el coste del aparato de puesta a punto, que por consiguiente resulta muy caro.

La segunda desventaja de este aparato es la de tener una visualización que expone únicamente dos bytes cada vez, con lo que es dificil para el programador tener una visualización completa del registro deseado de la memoria. Debido al hecho de que los dos modos de funcionamiento, normal y de puesta a punto, se excluyen mutuamente, existe pues el problama técnico de contar con un equipo de puesta a punto que utiliza todos los dispositivos de entrada-salida tales como el teclado, el pupitre, la visualización y el lector de tarjetas magnética que normalmente equipan la calculadora.

Este problema técnico se resuelve con el aparato de puesta a punto según la invención, que se caracteriza por unos medios (PI-314) controlados por dicho elemento

de conmutación (tecla 100) para condicionar la citada unidad central (CU-3), interrumpir la ejecución de los citados programas ejecutables para activar el segundo medio de operación, y permitir que el citado teclado introduzca la información que debe tratarse por el citado programa de puesta a punto.

Estas y otras características de la invención aparecerán claramente en la descripción que sigue y en las figuras adjuntas en las que:

La figura 1a representa un esquema de bloques del elaborador que utiliza el sistema de puesta a punto de los programas según la invención;

La figura 1b representa una vista en perspectiva del elaborador;

Las figuras 2a, 2b, 2c, representan un esquema detallado de bloques de la unidad central del elaborador;

La figura 2 representa la composición de las figuras 2a, 2c; la figura 3 representa unas señales de temporización de la unidad central;

La figura 4 representa el registro de estado SO;

La figura 5 representa señales de temporización del registro de estado SO;

La figura 6 representa señales que se operan en el registro SO durante la lectura de una microinstrucción;

La figura 7 representa los registros operativos 30;

La figura 8 representa un diagrama de bloques de los registros 30;

La figura 9 representa un esquema ZRM de la RAM 1;

La figura 9a representa el byte de condiciones de programa;

La figura 9b representa el byte del código de interrupción;

La figura 9c representa el byte de preinscripción de interrupción;

La figura 9d representa el byte de servicio de puesta a punto;

Las figuras 10a, 10b y 10c representan el flujograma del microprograma intérprete;

La figura 10d, representa los formatos de las instrucciones;

Las figuras 11a-11g, representan los flujogramas del programa de DBG;

Las figuras 12a-12f representan un ejemplo de visualización de una instrucción.

La figura 13 representa el flujograma del microprograma de identificación de las barras;

La figura 14 representa el flujograma que resume las operaciones del sistema de puesta a punto según la invención.

LISTA DE ABREVIATURAS

DI = Registro 41, indica los ocho desviadores

DEV = Un desviador del registro 41, especificado por tres bits

CRT = Carácter, corresponde a ocho bits de la memoria.

MEM = Memoria RAM 1

IND = Dirección

MLS = Matriz lógica de secuencia 28
RB = Registro base
P1 = Indicador 1
P2 = Indicador 2
CP = Condiciones de programa
CI = Código de interrupción
MI = Modificación de instrucción
DBG = Puesta a punto
BSD = Byte de servicio DBG
C.M. = Tarjeta magnética
OREX = O exclusivo
CI = Código de interrupción
IP = Dispositivo de direccionamiento de programa (207)
AI = Habilitación de interrupción
PSR = Programa en curso (Registro 300 de lá RAM 1)
IPSR = Programa de interrupción (Registro 302 de la RAM 1)
OPSR = Programa interrumpido (Registro 301 de la RAM 1)
ZRM = Zona reservada de la RAM 1
RC = Referencia de la corriente
CC = Código de condición
UC = Unidad central 3
UP = Unidad periférica 4
IR = Dirección de retorno (Reg. 327, Fig. 9)
II = Direcciones del programa de interrupción (Reg. 335 de la Fig. 9)
IS = Dirección de stop (Reg. 350, Fig. 9)
BSD = Byte del servicio de puesta a punto (Reg. 351 de la figura 9&
RL = Registro de trabajo (Reg. 352 de la fig. 9)
AB = Habilita/Barras

ITR = Registro de dirección de la Tabla de Referencias.

Con referencia a las figuras 1a y 1b se procede a continuación a una breve descripción del elaborador que utiliza el sistema de puesta a punto de los programas según la invención.

Naturalmente se hará referencia a una forma particular de realización del elaborador sin limitar por ésto las posibilidades de aplicación del sistema según la invención a otros tipos de elaboradores.

En particular, el elaborador de la figura 1a y 1b es del tipo microprogramado, es decir, a cada instrucción del programa corresponde un microprograma grabado en una memoria fija por lo cual la ejecución de una instrucción de programa se realiza mediante la ejecución secuencial de las microinstrucciones del microprograma correspondiente.

El elaborador de las figuras 1a y 1b comprende una memoria RAM 1 apta para contener las instrucciones y los datos del programa en curso de ejecución; una memoria ROM2 apta para contener tanto los microprogramas que realizan las instrucciones de los programas como los programas utilizados por el sistema de puesta a punto según la invención, como se explicará mejor más adelante.

La RAM1 y la ROM2 pueden ser de cualquier tipo, comercialmente conocido, y por consiguiente no se describirán con detalle, exigiéndose únicamente que cada una de las posiciones de ambas memorias pueda contener 16 bits.

La RAM 1 y la ROM2 se encuentran conectadas a una unidad central 3 de elaboración, la cual se explicará detalladamente a continuación, y que a su vez va conectada a un grupo de unidades periféricas 4.

Las unidades periféricas 4 pueden ser de diverso tipo según la aplicación a la que se destina el elaborador. En este caso particular se describirán y mostrarán a continuación sólo las unidades periféricas utilizadas por el sistema de puesta a punto según la invención. En particular, las unidades periféricas mostradas son: un teclado alfa numérico 5, un visualizador 6, una consola de mandos 7, una impresora 8, y una unidad de lectura y escritura 9' que sirve para registrar y leer datos en una tarjeta magnética 9.

Con referencia a las figuras 1a y 1b se describen ahora brevemente las operaciones que debe efectuar el programador durante la fase de puesta a punto de un programa anteriormente registrado en la RAM1. Naturalmente, estas operaciones se describirán más adelante, con mayor detalle.

Supongamos ahora que el programa registrado en la RAM1 no se ha efectuado correctamente por el elaborador a causa de los errores de diverso tipo que puede haber cometido el programador durante su preparación.

En este punto, el programador pretende realizar un control de las instrucciones del programa que el elaborador no puede efectuar, y supone que una de ellas esté equivocada. Para corregir directamente en la RAM1 esta instrucción, el programador actua en la consola 7 y colocando un conmutador de llave 100 (figura 1b) de la posición normal en la posición de puesta a punto, escribe en la parte numérica 101 del teclado 5 la dirección correspondiente de la instrucción y acciona después una barra de servicio S1 perteneciente a un grupo de barras 102.

Al mismo tiempo que esta maniobra, se genera una interrupción del programa que hay que corregir (provo-

cada por el conmutador 100 accionado en la consola 7) y se realiza uno de los programas de puesta a punto registrados en la ROM2 asociado a la barra particular accionada en el teclado 5. Este programa, por ejemplo, puede tener como efecto la visualización de la instrucción correspondiente a la dirección escrita en el teclado, la detención de la elaboración con habilitación del teclado 5. De esta manera el programador puede colocar en el teclado 5 la instrucción que considera correcta. Posteriormente, el operador acciona otra barra de servicio S6, a la que va asociado otro programa de puesta a punto que registra en la memoria RAM1 la instrucción correcta correspondiente a la dirección colocada anteriormente.

Cuando el programador desea realizar un programa de puesta a punto distinto de los registrados en la ROM2, inserta en el lector 9 la tarjeta magnética 9' en la que está registrado un programa de puesta a punto y acciona la barra de servicio S2. Esta última exige un particular programa de la ROM2 que provoca la lectura del programa registrado en la tarjeta 9, su transferencia a una zona fija (ZRM) de la RAM1 y la ejecución inmediata de dicho programa.

Se puntualiza, y ésto se explicará detalladamente a continuación, que la zona fija de la RAM1 en la que se transfiere el programa de tarjeta no contiene informaciones significativas para la reanudación del programa en prueva, por lo cual no se tiene pérdida de informaciones en la realización del programa de puesta a punto registrado en la tarjeta magnética 9. De todo lo dicho se desprende evidentemente una de las ventajas del sistema según la invención, es decir, la posibilidad de probar los programas simplemente accionando un conmutador y utilizando los mismos órganos (te-

clado, visualización, tarjeta magnética) utilizados durante el funcionamiento normal.

A continuación se procede con referencia a la figura 2 a una descripción detallada de la unidad central 3.

La unidad cent ral 3 es un conjunto de circuitos lógicos que dirigen y efectuan los diversos microprogramas contenidos en la ROM2.

Está compuesta de cuatro bloques principales:

Un temporizador 20 que temporiza el desarroolo de la elaboración de datos en el interior de la unidad cent ral 3. Se compone de un oscilador 21 y de un conjunto de circuitos generadores de señales 22.

Una red de la matriz lógica de secuencia 25, la cual toma en consideración e interpreta los códigos de las microinstrucciones leídas por la ROM2 y genera los mandos necesarios para su ejecución. Está compuesta por un registro de la microinstrucción 26 (RO), un registro de estado 27 (SO) y una matriz lógica de secuencia 28 (MLS).

Una red operativa la cual realiza la elaboración de datos con modalidades impuestas por la matriz lógica de secuencia 28. La red operativa comprende: los registros operativos 30 ("block de notas") los cuales están divididos en dos grupos RA-31 y RB-32, cada uno de los cuales está compuesto por 16 registros de 8 bits indicados a continuación con la referencia A0-A15 y B0-B15, respectivamente; una unidad aritmética 35, formada por tres bloques con paralelismo; ocho bits; UA-37, UB-36, UC-38; los desviadores DI-40; una red de desviación ND-41; una red de entrada

a los registros operativos que comprende los nudos NA y NB, y dos registros NA y NB, y dos registros BA-42, BB-43, y una red de conexión con la RAM1 compuesta por nudosNO y NC; una lógica de canal 45, la cual controla la interfase de conexión de las unidades periféricas que controla la simultaneidad operativa de la unidad central 3.

El oscilador 21 genera impulsos periódicos que definen un período de tiempo fijo denominado ciclo de máquina que dura el tiempo necesario para la ejecución de una operación elemental (por ejemplo: lectura de un registro operativo 30, su incremento y nueva escritura en el registro operativo 30).

Durante el ciclo de máquina se generan por el circuito 22 unas señales cuya duración y colocación son fijos en el ciclo de máquina.

La función de tales señales está preestablecida y, el hecho de que actuan o no sobre los circuitos de la unidad central 3 viene determinado por las condiciones generadas por la matriz de secuencia 28 de la manera que se describirá a continuación.

El funcionamiento de la unidad central 3 es completamente síncrono con esta temporización, como tambien la conversación con las unidades periféricas.

Las señales generadas por el circuito 22 son diez y su utilización se ilustrará a continuación. Son:

T0 que actua en el registro de estado 27,

T1 que temporiza la lectura ee la ROM2

T2 que temporiza la TAM1

T3A que actua en el registro RO-26

T3N que actua también en el registro RO26,

T4A que actua en los registros BA 42, BB 43 y sus desviadores 40,

T4N que actua en los registros BA 42, BB 43 y sus derivadores 40,

T5 que actua en los registros operativos 31 y 32,

T6 y T7 que actuan en la lógica de canal 45.

En la figura 3 se muestra un diagrama de temporización en el que aparecen las señales citadas.

Naturalmente, el oscilador 21 y los circuitos 22 no se describen detalladamente porque son conocidos en el campo del diseño de circuitos.

Antes de proceder a la descripción de los demás bloques de la unidad central 3, nos referiremos ahora brevemente a las microinstrucciones utilizadas por la unidad central 3 en el sistema de puesta a punto según la invención y a su ejecución.

La ejecución de una microinstrucción puede subdividirse en dos fases: una fase interpretativa, común a todas las microinstrucciones, que lee en la ROM2 la microinstrucción referenciada, predispone su realización e incrementa el dispositivo de accionamiento de la ROM2. Esta fase, independientemente, es independiente del código de la microinstrucción leída. Una fase ejecutiva, durante la cual se realiza la elaboración de los datos según las modalidades indicadas en la microinstrucción leída en la fase interpretativa precedente. La fase interpretativa se desarrolla siempre en un solo ciclo de máquinas y la configuración de las señales (que en lo sucesivo se denomonarán mandos) es estable en el ámbito del ciclo. La configuración de tales mandos define las operaciones que hay que desarrollar y se denomina

"Estado Interpretativo".

La presencia del Estado Interpretativo es identificada por un biestable del registro 27 denominado S000 (Figura 4).

La fase ejecutiva se desarrolla en uno o varios ciclos de máquina a los que corresponden otros tantos estados definidos cada uno por un biestable correspondiente del registro 27.

Durante toda la fase ejecutiva el código de la microinstrucción en cuestión permanece estable en el registro 26, mientras evoluciona la situación de los biestables del registro 27 que definen el estado en curso.

Cada estado define el sucesivo en función del código de la microinstrucción leída.

Al terminar la ejecución de cada microinstrucción se vuelve al estado interpretativo S000 para leer en la ROM2 la microinstrucción sucesiva.

Durante ambas fases, interpretativa y ejecutiva, la red combinatoria 28 (MLS) que tiene como entradas los registros 26 y 27, genera unos mandos C que habilitan determinados flujos de información a través de la red operativa o los demás bloques de la unidad central 3.

Las informaciones circulan acto seguido entre los bloques de la unidad central 3 a través de una serie de puertas Y de diversos tioos que son controladas por las señales de mando C generadas por la red combinatoria 28. En la figura 2, estas puertas se representan simbólicamente divididas en tres zonas. La zona central contiene la señal de control de la puerta generada por la red 28 (MLS). Cuando esta señal de mando está presente las señales en la entrada

de la puerta se transfieren al bloque suvesivo. Los pares de números variables de 00 a 15 que se encuentran en la zona superior y en la zona inferior de las puertas indican el número de bits que dejan pasar, y mas concretamente, las posiciones en las que se encuentran a la entrada y a la salida estos bits. Por ejemplo, una puerta que tenga tanto en entrada como en salida los pares de números 07, 00, es una puerta que transfiere un carácter de 8 bits en directo paralelo, Por el contrario, una puerta que tiene en la zona superior, es decir, en entrada, el par de números 03, 00 y en la zona interior, esto es en salida, el par de números 07, 04 es una puerta que transfiere cuatro bits escalándolos hacia la izquierda en cuatro posiciones. Si 07, 04 se encuentran en entrada y 03, 00 en salida, el escalamiento es de cuatro posiciones hacia la derecha. Finalmente, si la zona en entrada está vacía significa que los bits son forzados a la puerta desde el exterior.

A continuación, y con referencia a la tabla A, se describe el grupo de microinstrucciones utilizadas por el sistema de puesta a punto según la invención, dejando a un lado las otras microinstrucciones que puede efectuar la unidad central. Las microinstrucciones que se dan en la Tabla A tienen un formato fijo de 16 bits que corresponde a una palabra de la ROM2. El formato de las microinstrucciones es el siguiente:

F	X	Y	Z
15 11	07	03	00

Los campos, cada uno de cuatro bits, tienen el significado siguiente:

F es el código operativo de la microinstrucción;

X indica el primer operando;
Y indica el segundo operando;
Z es un extensor de uno de los campos anteriores.

Cuando los campos Z e Y especifican como operandos los registros A, B ó L de los registros operativos 30, se indicarán en las microinstrucciones con los símbolos Ax, Bx, Lx, Ay, By, Ly respectivamente.

Las microinstrucciones se subdividen en grupos, que se distinguen por el distinto código de función, es decir, por la diversa configuración binaria del campo F de la microinstrucción.

Las microinstrucciones que tienen el mismo código de función se realizan con la misma secuencia de estados.

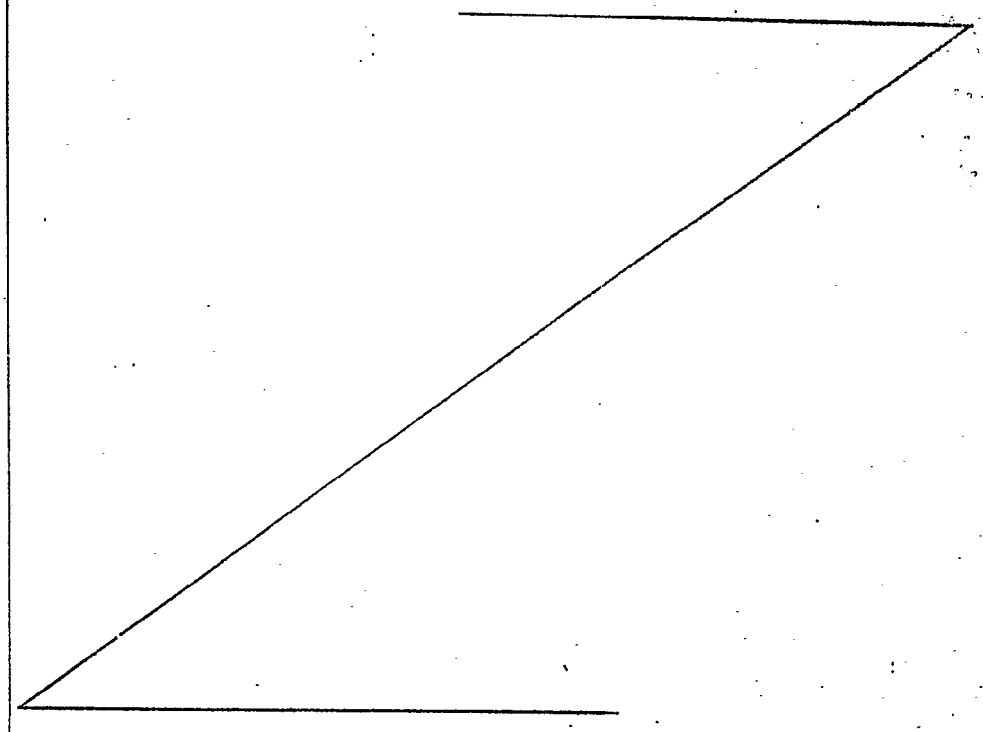


TABLA A

Nombre	F	X	Y	Z	FUNCION
					ARITMETICAS LOGICAS
ADDB	0110	A	B	0101	B ⟵ (A+B)
ANDA	0110	A	B	1000	Si (A ⟵ A AND B)=0
ANDB	0110	A	B	0100	Si (B ⟵ A AND B)=0
AND	0110	A	B	0000	si(A AND B) =0 } PONE DO1=1
ORA	0110	A	B	1110	Si (A ⟵ A OR B) =0
ORE	0110	A	B	0111	Si (A OR EX B) =0
SOT	0110	A	B	0010	Si (A - B) >0 DO0=1
					TRANSFERENCIA
TAB	0101	A	B	1100	B ⟵ A
TBA	0101	A	B	0011	A ⟵ B
					CONMUTACION
SLL	0100	L	L	1111	Ax ⟷ By ; Bx ⟷ Ay
					DECREMENTO
DCA	1010	A	0100	1010	Si (A ⟵ A - 1)=0 pone DO1=1
					CARGA DE DESVIADORES
TADI	1011	A	1110	0111	DI ⟵ A

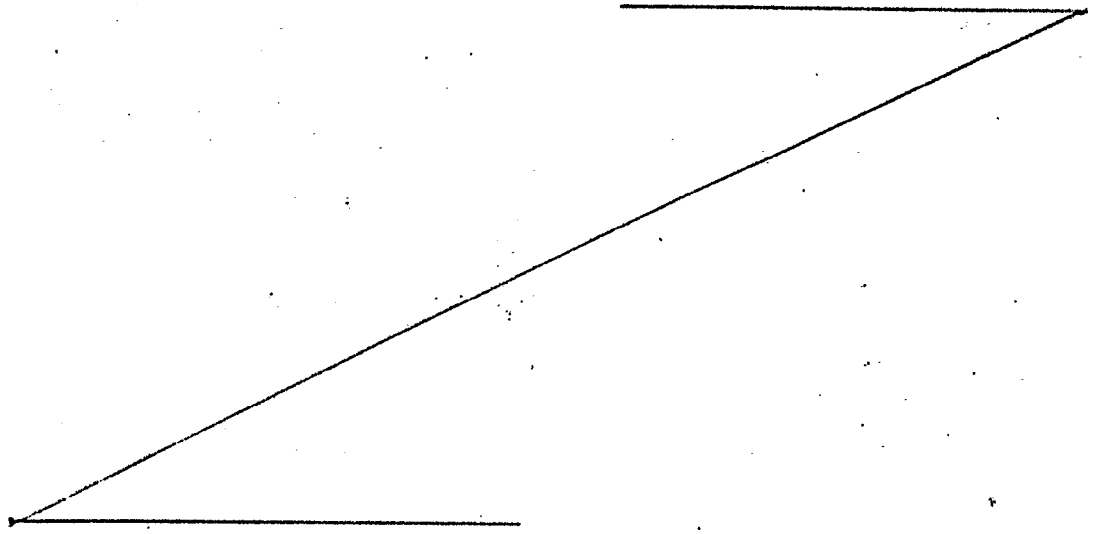
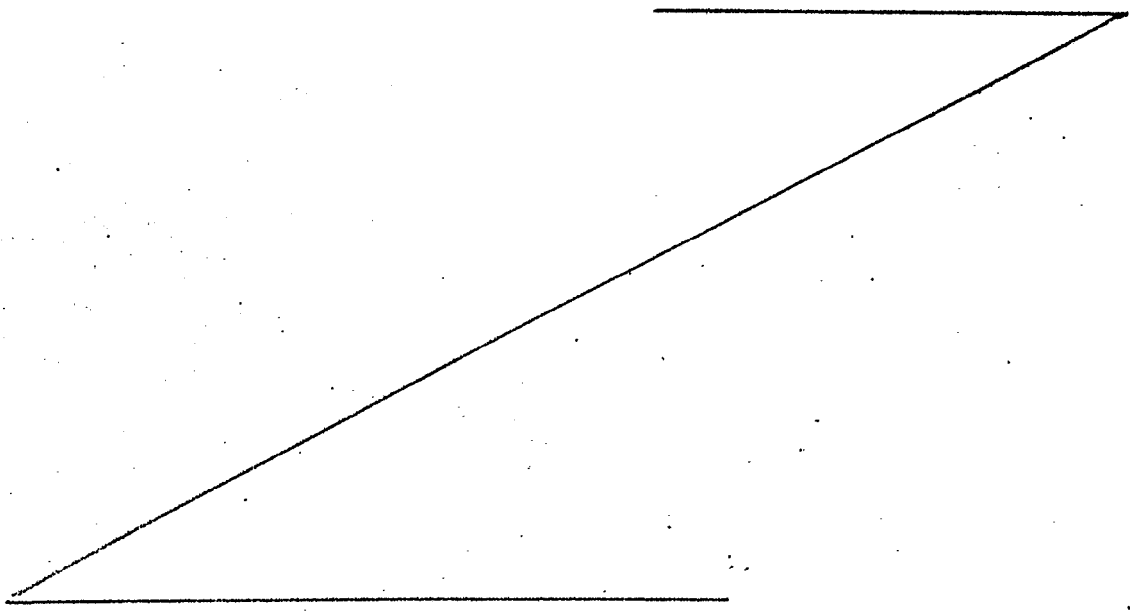


TABLA A

Nombre	F	X	Y	Z	FUNCION
TBDI	1011	B	1111	0111	DI ⟵ B
REDI	1011	0 DESV	0110	0110	DESV ⟵ '0'
SEDI	1011	1 DESV	0110	0110	DESV ⟵ '1'
SHSB	1011	B	0001	0101	DESVIA B 1 bit a la izquierda
ROTB	1011	B	0001	0110	CONMUTACION DE SEMIBYTE
AZAP	1011	A	0010	0111	PONE A CERO EL SEMIBYTE IZQUIERDO
					SALTO
SAI	000	I			Salto IND. I
SADO	0010	ODESV	I		Salto a I si DEV = 0
SADI	0011	.0 DESV	I		" " " " DEV = 1
					ESCRIBE/LEE RAMI
MAD	1100	A	I		A ⟵ MEM.DIR. I
AMD	1101	A	I		MEM. DIR. I ⟵ A
AMI	1110	L	A	1011	MEM DIR. L ⟵ A
BMI	1110	L	B	0011	MEM.DIR. L ⟵ B
AMIP	1110	L	A	1001	MEM.DIR. L ⟵ A; L ⟵ L + 1
BMIP	1110	L	B	0001	MEM.DIR. L ⟵ B; L ⟵ L + 1
MAIP	1110	L	A	1101	A ⟵ MEM.DIR. L; L ⟵ L + 1

TABLA A

Nombre	F	X	Y	Z	FUNCION
MBIP	1110	L	B	0101	B ⟵ MEM.DIR.L; L ⟵ L + 1
					FUERZA REGISTROS 30
CRTA	1000	A	CRT		A ⟵ CRT
CRTB	1001	B	CRT		B ⟵ CRT
					LEE ROM2
ROMA	0111	A	0000	0000	A ⟵ MEM.DIR.L2; b07=0 pone 8 bits más significativos si b07=1 pone 8 bits más significativos b07=bit más significativos del registro B2. L2 ⟵ L2 + 1
TCCA	1010	A	1000	1000	A ⟵CRT. DE PUPITRE



El registro 27 está formado por ocho biestables (figura 4) que diferencian los diversos ciclos de máquina. Son: S000-S001-S002-S003-S004-S042-S043-S010.

Su colocación es controlada por la matriz lógica 28 analizando directamente el campo F de la microinstrucción presente en el registro 26 (RO). El cambio de la configuración del registro 27 se realiza con el frente de subida de la señal TO y es ésta la primera operación de la matriz 28 efectua en el ámbito de un ciclo de temporización.

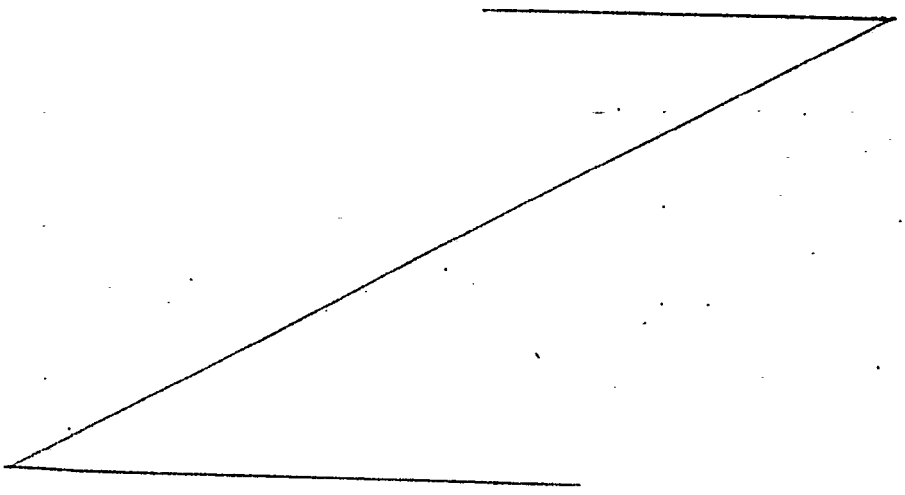
Las señales S042, S043, S010 son obtenidas por el 0 de los siguientes estados:

S042 = S004 + S002

S043 = S004 + S003

S010 = S000 + S001

En la figura 5 se muestra el diagrama de temporización correspondiente al estado de S010 a partir de los estados S000 y S001. Naturalmente, de igual modo se generan los estados S042 y S043. Hay que observar, por consiguiente, que la matriz 28 genera sólo 5 estados, a saber S000-S004, mientras que los otros tres estados se derivan de éstos.



En la Tabla B se da ahora la sucesión de los estados correspondientes a la microinstrucción de la Tabla A.

TABLA B

F	SECUENCIA DE EJECUCION	TIPO DE MICROINSTRUCCION
000	S001	SALTO
0010	S001	
0011	S001	
0100	S002 S003	TRANSFERENCIA
0101	S002	ARITMETICAS Y LOGICAS
0110	S002	
0111	S002 S001 S004	LECTURA
1000	S004	ROM2 en RA/RB
1001	S004	
1010	S004	± COMPRUEBA; VARIOS PUPITRES
1011	S004	DESVIACION Y OPERACION EN
		DESVIADORES 40
1100	S004 S002	MEMORIA RAM1
1101	S004 S002	
1110	S004 S002	
1110	S004 S003	
1111	S004 S003	

Hay que observar, por último, que todas las secuencias están procedidas por el estado interpretativo S000. Las señales de mando generadas por la matriz 28 en cada uno de los estados se describirán más adelante.

El registro 26 comprende diez y seis biestables que toman en consideración el código de la microinstrucción o la información leída por la ROM en la dirección especificada por unos determinados registros operativos 30.

Los diez y seis biestables se dividen en dos grupos de ocho: los menos significativos son controlados por la señal T3N, y los demás por la señal T3A.

La generación de las señales T3N y T3A se efectua únicamente en los dos estados en los que se efectua una lectura de la ROM, a saber: en el estado S000 interpretativo de todas las microinstrucciones, en el estado S001 de la microinstrucción ROMA.

Con el frente de subida de la señal T3N y T3A los diez y seis bits leídos por la ROM2 son tomados en consideración en el registro RO-26 y constituyen el código de la microinstrucción que debe realizarse.

La información permanece estable en el registro durante todas las fases ejecutivas siguientes, como se muestra en la figura 6.

En el estado S001 de la microinstrucción ROMA, como se ha dicho, se realiza una segunda lectura de la ROM. Los ocho biestables menos significativos del registro 26 son colocados con la señal T3N con los ocho bits más o menos significativos leídos. Esto depende del valor del bit 07 del registro B2 (Véase tabla A.).

Los registros operativos 30 se encuentran organizados en dos series denominadas A y B de diez y seis registros cada una con una capacidad de ocho bits (figura 8). Los bits del mismo peso de los registros de cada una de las series, por ejemplo la serie A, se encuentran organizados en

una matriz 4 x 4 (figura 7). Por lo cual se tiene ocho matrices 4 x 4 en las que el primer bit de cada una de ellas forman el registro A0, el segundo bit el registro A1, y así sucesivamente.

Para seleccionar un registro, por ejemplo el registro A15, basta con enviar a los ocho hilos de selección mostrados en la figura 7 ocho señales de mando C024, C031 que tienen la siguiente configuración: 10000001.

Naturalmente, las señales de mando C024-C031 son generadas por la matriz de secuencia 28 la cual tiene en cuenta los campos Z e Y de las microinstrucciones para generar tanto las señales de mando de selección (C024-C031) como el estado asociado (formado en S0) a una de las dos series de registros. En particular al estado S043 selecciona uno de los registros de la serie B, mientras que el estado S042 selecciona un registro de la serie A. El estado S010, por el contrario, va asociado a un registro de diez y seis bits de largo formado por los registros A y B análogos, denominado "registro largo" e indicado L. La escritura de una información en uno de los registros 30 con las informaciones ya registradas en los registros 42 y 43, es temporizada por la señal T5 como se ha dicho (figura 2). En este instante las señales de mando CT04-CT07, generadas por la matriz lógica 28, seleccionan los datos que hay que transferir a los registros 31 y 32 al nivel de cuatro bits cada vez, es decir, se pueden modificar uno de los registros A ó B en una de sus partes, dejando la otra parte sin cambios.

La unidad aritmética 35 efectúa operaciones aritméticas y lógicas sobre el contenido de los registros operativos 30.

Está constituída por dos sumadores con paralelismo 8 AU-34 y UB-36 y por una red lógica UC-38. Los dos sumadores 34 y 36 (UA y UB) están conectados entre sí de forma que se obtenga un sumador único con paralelismo 16. No obstante, sólo en operaciones particulares, es decir, cuando se actúa en un registro largo (L), son significativas las diez y seis salidas del sumador.

La red UC-38 que puede entrar como primer operando en AU, realiza las funciones lógicas O, Y y O exclusivo.

La unidad aritmética 35, por medio de un decodificador 50 (figura 2b) suministra además informaciones sobre el resultado de las operaciones aritméticas y lógicas que son memorizadas en el desviador DO como consecuencia de las señales de mando CD11 y CD12 generadas por la matriz lógica MLS-28. Este desviador es posteriormente examinado por las instrucciones SADO y SADI para efectuar saltos a condición.

En la Tabla C siguiente, se da la lista de las microinstrucciones que interesan a la unidad aritmética 35, en la que aparece el nombre simbólico de las señales de mando CU00-CU09 generadas por MLS26 que realiza la transferencia de los datos y los estados de validez de las señales de mando.

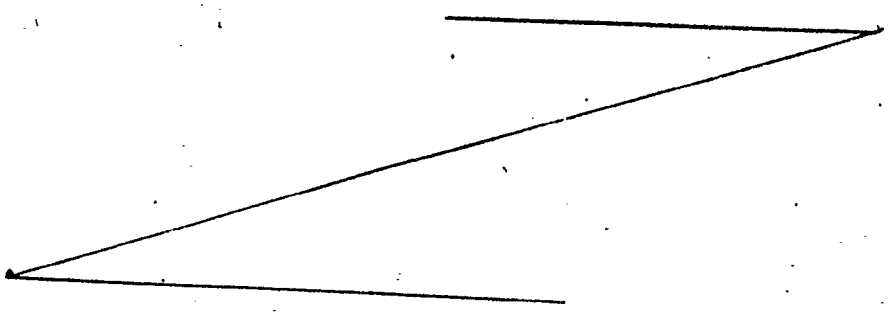
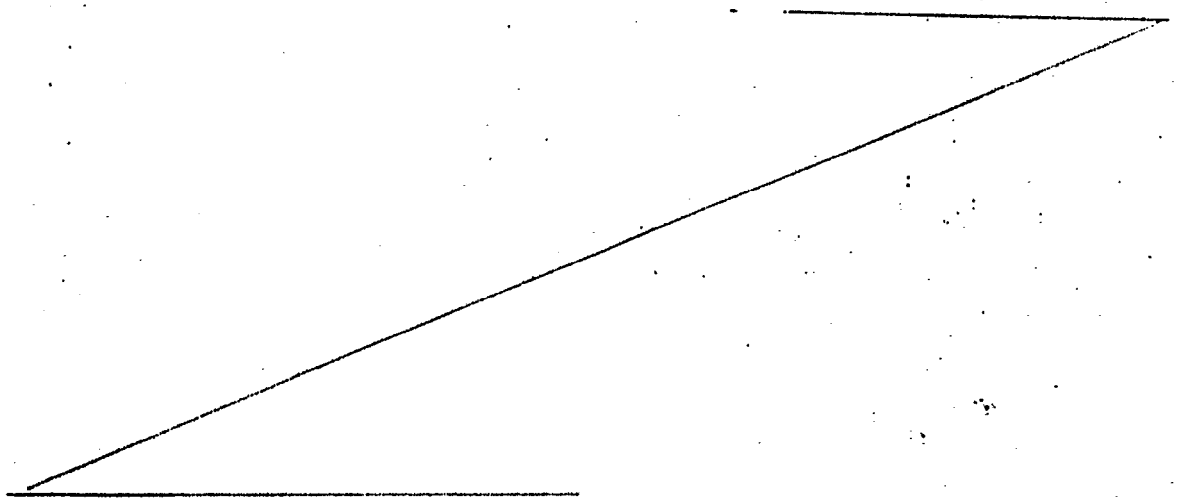


TABLA C

MINSTR.	CU00	CU01	CU02	CU04	CU05	CU06	CU07	CU08	CU09	ESTADO DE VALIDEZ
ADDB	1	0	0	0	1	X	X	X	X	S002
DCA	0	0	0	1	0	X	X	X	X	S004
AND	1	1	1	1	0	X	0	1	0	S002
ANDA	1	1	1	1	0	X	0	1	0	S002
ANDB	1	1	1	1	0	X	0	1	0	S002
ORA	1	1	1	1	0	X	1	0	0	S002
ORE	1	1	1	1	0	X	0	0	1	S002
ROMA	1	1	0	0	0	0	X	X	X	S001
TAB	1	1	0	1	0	0	X	X	X	S002
TBA	1	1	0	1	0	0	X	X	X	S002
MAIP	1	1	0	0	0	0	X	X	X	S004
AMIP	1	1	0	0	0	0	X	X	X	S004
MBIP	1	1	0	0	0	0	X	X	X	S004
BMIP	1	1	0	0	0	0	X	X	X	S004
MBI	1	1	0	1	0	0	X	X	X	S004
AMI	1	1	0	1	0	0	X	X	X	S004
BMI	1	1	0	1	0	0	X	X	X	S004



Los desviadores 40 están constituidos por ocho biestables (D00-D07) que toman en consideración acontecimientos que ocurren durante la ejecución de algunas microinstrucciones. Su contenido es sondeado durante la ejecución de los microprogramas para condiconar la realización de saltos de dirección en los dispositivos de direccionamiento de la ROM2. Las microinstrucciones lógicas (Y,O etc) se relacionan automáticamente con ellos para depositar el resultado de la operación lógica efectuada.

Cada uno de los desviadores puede colocarse además en CERO o bien en UNO gracias a la microinstrucción REDI y SEDI, respectivamente (Tabla A).

En el formato de esta microinstrucción (Tabla A) los tres bits menos significativos del campo x constituyen la dirección binaria (00 ÷ 07) del desviador interesado.

Algunas microinstrucciones (TADI-TBDI-SADI) fuerzan en los ocho desviadores los ocho bits del registro A ó B seleccionado (Véase Tabla A).

Algunas microinstrucciones aritméticas y lógicas (Y,O, ORE, ADD), por el contrario, colocan los desviadores con su resultado cualitativo; en particular el desviador D0I toma en consideración la presentación de un resultado nulo en salida por la unidad aritmética 35.

Los desviadores 40 cambian su estado en dos tiempos diferentes. Los desviadores D00-D03 se conmutan con la señal 74N mientras que los desviadores D04-D07 se conmutan con la señal T4A. A continuación se da la tabla D, en la que se contienen las microinstrucciones que corresponden a los desviadores 40 y las señales de mando de los mismos desviadores generadas por la MLS28.

TABLA D

Microinstr.	CDRR	CU05	CD11	CD13	CD14
REDI	1	0	0	0	0
DCA	0	0	1	0	0
AND/A/B	0	1	1	0	1
OR/A/B	0	0	1	1	0
ORE	0	0	1	0	1
ADD/A/B	0	1	1	0	0
TADI	0	0	0	1	1
SADI	0	0	0	1	0
TBDI	0	0	0	0	1

A través de esta red formada por circuitos del tipo Y-0 se puede abrir una circulación de información entre todas las circulaciones posibles hacia la red de entrada (NA, NB) a los registros operativos (31, 32). La red de desplazamiento 41 está formada por un grupo de ocho puertas dividido en dos subgrupos conectados a los registros operativos RA-31 y RB-32, respectivamente. Cada uno de tales subgrupos es capaz de operar un escalamiento (desplazamiento) o una rotación sobre los datos procedentes de los registros operativos 30, como se muestra sinbólicamente en la figura 2b. Cada puerta de los dos subgrupos es direccionada por una combinación de tres bits de las microinstrucciones SHSB y ROTB que actuan sobre dicha red. Estas combinaciones se indican sinbólicamente en la figura 2b con los símbolos CZ00-CZ07, mientras que las otras dos puertas de la red 41 son controladas de modo directo y sirven para forzar las condiciones de los desviadores o bien

cero. Una entrada a la red de desplazamiento 41 está constituida además por una puerta 70, la cual va conectada a la lógica de canal 45 mediante el canal de introducción de datos.D. Esta puerta 70 permite la introducción de los datos procedentes de las unidades periféricas a través de la lógica 45 en los registros operativos 30 a través de los nudos NA ó NB.

Es una red en la que terminan los registros operativos 31 y 32; permite seleccionar el byte que debe enviarse y escribirse en los registros operativos 31 y 32.

Esta red está formada por los nudos NA y NB y por los registros BA-42 y BB-43.

Los nudos NA y NB son dos redes, cada una con paralelismo de 8 bits que seleccionan por medio de las señales de mando CA00-CA07, generadas por la MLS28, las ocho posibles circulaciones de información hacia los registros operativos 31 y 32.

Las informaciones que se seleccionan pueden proceder de hecho de las unidades siguientes:

- de la unidad aritmética 35 (dos circulaciones)
- de la red de desplazamiento 41 (ND),
- de la ROM2,
- de la RAM1,
- del pupitre 7 (dos circulaciones),
- de la lógica del canal 45.

Los registros BA-42 y BB-43 toman en consideración la información presente en los nudos NA ó NB seleccionada por una de las señales de mando CA00-CA07 en presencia de la señal T4. El contenido de BA-42 y BB-43 puede escribirse o no en los registros operativos 31 y 32 según que sean o no accionadas las señales de mando CT04-CT07 descritas

anteriormente.

La unidad central 3 está conectada a la entrada de la memoria RAM1 a través de un nudo NO con paralelismo de 16 bits (NO0÷15). Este nudo se activa durante la ejecución de las microinstrucciones de escritura en memoria y de lectura de la memoria.

En ambos casos, el nudo NO proporciona la dirección a la que se quiere acceder y sólo en las microinstrucciones de escritura envía el carácter (ocho bits), que debe memorizarse.

La salida de la RAM1 está constituída por un nudo NC de paralelismo 8 (NC00 ÷ 07) y se utiliza únicamente en caso de lectura.

Todas las microinstrucciones que preveen lectura o escritura en la RAM1 se realizan en tres ciclos de máquina: en el primer ciclo S000 se desarrolla el estado interpretativo, en el segundo ciclo S004 se envía a través del nudo NO la dirección de la RAM2 en la que actua la microinstrucción.

Los registros que pueden conectarse al nudo NO como direccionadores, son el registro RO-26 si se quiere acceder a una dirección menor de 255 (es decir, a la zona reservada de la RAM1) o bien un par de registros (AB ó BA) si se quiere acceder a una dirección cualquiera de la RAM1.

En la figura 2 las señales de mando de direccionamiento de la memoria RAM1 están representadas por las señales de mando CM03 ÷ CM07. La señal de mand9 CM03 habilita el registro RO, mientras que las señales de mando CM04 y CM05 habilitan los registros RA-31 y RB-32.

Del estado S004 se pasa, según el tipo de

microinstrucción en desarrollo, a S002 o bien a S003.

Se pasa al estado S002 para todas las microinstrucciones en las que es un registro B el que proporciona el dato que hay que escribir o que recibe la información leída.

Se pasa al estado S003, por el contrario, cuando es un registro A el que se dedica a la lectura o escritura.

En el ámbito de los estados S002 y S003 es necesario distinguir dos funcionamientos diferentes: 1) en las microinstrucciones de escritura se envía, acompañado por la señal T2, el dato que hay que escribir en la memoria (a la dirección ya especificada en el estado S004) a través de los ocho primeros bits del nudo NO (N000 a 07). La salida en eje de la memoria no es significativa y no se utiliza.

Las informaciones que pueden escribirse pueden proceder de los registros RA-31, RB-32 o bien de las unidades periféricas a través de la lógica de canal 45 cuando son generadas por la MLS28 las señales de mando CM04, CM06 y CM07, respectivamente. 2) En las microinstrucciones de lectura, por el contrario, el nudo NO no es significativo y no es utilizado por la RAM1. Por el contrario, tiene valor de salida NC que puede enviarse a un registro B si está presente el estado (S002) y la señal de mando CA05 o bien a un registro A si está presente el estado S003 y la señal de mando CA05.

En la tabla E se enumeran las microinstrucciones que utilizan la RAM1 con las respectivas señales de mando y estados generados por la MLS28.

TABLA E

MICRO-INSTRUCCION	Estado S004			Estado S002			Estado S003		
	CM03	CM04	CM05	CM04	CM06	CM07	CM04	CM06	CM07
AMD	.1	0	0	1	0	0			
MAD	1	0	0	X	X	X			
MAIP	0	1	1				X	X	X
MBIP	0	1	1	X	X	X			
AMI	0	1	1				1	0	0
AMIP	0	1	1				1	0	0
BMI	0	1	1	0	1	0			
BMIP	0	1	1	0	1	0			

En la descripción anterior se han mostrado todos los bloques de la unidad central 3, así como todas las señales de mando generadas por la MLS28 para dirigir el flujo de las informaciones entre dichos bloques. No obstante, no se ha mostrado detalladamente la MLS28; esta última no es otra cosa que una matriz que tiene como filas las salidas de los registros 26 y 27 y como columnas los conductores sobre los que se generan las señales de mando C. La MLS28 está condicionada además por el temporizador 20 para generar las señales de mando en la sucesión requerida.

Para mayores detalles sobre la MLS 28 se hace referencia al volumen "Microprogramming", Principles and Practices" de Samir S. Husson editado por Prentice-Hall Inc. Englowood Cliff., Estados Unidos de América de 1970. En el capítulo 2, y con referencia a ejemplos concretos, se explica el principio con el que se generan una secuencia de mandos aptos para efectuar microinstrucciones.

La lógica del canal 45 es un conjunto de circuitos aptos para controlar y coordinar la conmutación de datos y señales de mando entre la unidad central 3 y las unidades periféricas 4 conectadas a la misma, excluyendo el pupitre 7, el cual tiene el acceso directo a la unidad central 3 a través del nudo NA-NB.

En la patente italiana nº 916.415, publicada el 15 de abril de 1972 a nombre de la solicitante, se da una descripción detallada de la lógica de canal 45.

Aquí sólo interesa puntualizar que la lógica de canal 45 controla las microinstrucciones entre los diversos niveles prioritarios presentes en el eleborador sobre la base de un orden prioritario preestablecido.

La razón de haber insertado la lógica de canal 45 es pues la de permitir la interrupción del microprograma en curso para realizar un microprograma de interrupción más prioritario.

En el caso particular se tienen cuatro niveles prioritarios de microprogramas, a saber:

-El microprograma principal o de prioridad 4 que tiene normalmente la función de interpretar y realizar las instrucciones del programa elaborando los datos y poniendo en marcha las operaciones de entrada y de salida;

-Un microprograma de prioridad 3, destinado normalmente a realizar operaciones que no entran en la secuencia temporal predeterminada del programa, por ejemplo predisposiciones de interrupciones del programa, y control microprogramado de operaciones de entrada-salida;

-Microprograma de prioridad 2 y 1; destinados normalmente a efectuar la transferencia de datos de una

unidad periférica a la mekoria o viceversa.

A cada microprograma va asociado un registro direccionador según se indica en la figura 8.

En particular, el nivel 4 está direccionado por el registro L00, el nivel 3 p or el nivel L01, el nivel 2 por el registro A13 y el nivel 1 por el registro A12.

La transferencia de datos de las unidades periféricas a la unidad central 3 puede efectuarse de dos modos. El primero es controlado por la puerta 39 que, a través del nudo N0 (Figura 2c) permite el acceso directo a la RAM1. Esta puerta es controlada por las microinstrucciones de acceso directo a la RAM1 anteriormente descritas. El segundo modo es controlado por la puerta 70 del nudo ND-41 y permite el acceso a los registros operativos 31 y 32 a través de los nudos NA y NB. Los datos y las señales de mando que procedentes de las unidades periféricas se registran en los registros operativos 30 y 31 son elaborados directamente por el grupo de microinstrucciones que trabajan sobre los registros.

A continuación se procede a una descripción de la parte de la RAM1 utilizada por los programas DBG con referencia a la figura 9. La RAM1 se divide en dos zonas. La primera zona, denominada zona reservada (ZRM) está a disposición del micrprograma intérprete y de los microprogramas que controlan las unidades periféricas y de los programas de DBG.

La segunda zona, por el contrario, está destinada a registrar los programas que hay que realizar, los datos sobre los que actuan estos programas y los resultados de las elaboraciones.

Antes de describir detalladamente la RAM1 es necesario que apuntemos, siquiera brevemente, las operacio

nes desarrolladas por un microprograma particular, residente en la ROM2, y denominado intérprete. Este micrpprograma particular, residente en la ROM2, y denominado intérprete. Este micropograma, que se describirá con mayor detalle a continuación, realiza las operaciones siguientes:

- Interpreta la instrucción en curso (fase ALFA)
- Reconoce las interrupciones de programa. Lanza el programa de interripción reconociendo si está registrado en la RAM1 o en la ROM2.
- Deshabilita todas las interrupciones, incluída la del programa en fase de lanzamiento.
- Habilita la lectura por parte de la RAM1 ó de la ROM2 según que el programa de interrupción se encuentre en la RAM1 ó en la ROM2.
- Efectua la lectura de las instrucciones por la RAM1 ó la ROM2.
- Reconoce los formatos de las instrucciones.
- Extrae los operandos.
- Lleva a la práctica las instrucciones lanzando los microprogramas asociados a las mismas (Fase BETA).
- La ZRM comprende en particular un registro PSR-300 (figura 9) que contiene los parámetros del programa en curso de elaboración, y está constituído por los registros siguientes: (véase Tabla F).

Un registro base RB-310 que contiene la dirección de comienzo de la zona de memoria disponible para los programas normales. El registro RB-310 es utilizado por el intérprete para calcular las direcciones de los operandos expre

sados en las instrucciones. Es modificado por instrucciones adecuadas durante la ejecución de un programa.

TABLA F

REG. Nº	NOMBRE	SIGLA	NºBYTE	DIRECCION DE	DIRECCION A
310	REG. BASE	RB	2	00B0	00B1
311	INDICADOR 1	P1	2	00B2	00B3
312	INDICADOR 2	P2	2	00B4	00B5
313	CONDIC.DE PROGRAMA	CP	1	00B6	—
314	PRENOTACION INTERR.	PI	1	00B7	—
315	MODIFIC.DE LA INSTRUCCION	MI	1	00B8	—

REG. Nº	NOMBRE	SIGLA	NºBYTE	DIRECCION DA	DIRECCION A
320	REG. BASE	RB	2	00D0	00D1
321	INDICADOR 1	P1	2	00D2	00D3
322	INDICADOR 2	P2	2	00D4	00D5
323	COND.DE PROGRAMA	CP	1	00D6	—
324	COD. INTERR	CI	1	00D7	—
325	MODIFIC.DE LA INSTRUCCION	MI	1	00D8	—
327	DIRECCIONAMIENTO OPSR	IR	2	00DA	00DB
333	CONIDC. PROGR.	CP	1	00BC	—
334	HABILIT. INT.	AI	1	00BD	—
335	DIRECC. INTERR	II	2	00BE	00BF
350	DIRECC. STOP	IS	2	00EC	00ED
351	BYTE DE SERV. DBG	BSD	1	00C7	—
352	REGISTRO DE TRABAJO	RL	8	00A8	00AF
353	DIRECCION DE LA TABLA DE REFERENCIA	ITR	3	00D4	00D6

Registros indicador P1-311 e indicador P2-312; son registros utilizados por instrucciones particulares para calcular las direcciones absolutas de los operandos. Esas direcciones se obtienen sumando P1-311 ó P2-312 con RB-310. Su contenido puede ser modificado por instrucciones particulares.

El byte Condiciones de programa que se da en la figura 9a, tiene el significado siguiente:

Los bits 00,01 se denominan código de condición (CC) y son recogidos por las instrucciones aritméticas y lógicas para memoriar los resultados significativos. Estas condiciones son probadas después por otras instrucciones para realizar saltos en condición. El bit 03 es utilizado por el microprograma intérprete para determinar si la instrucción que hay que llevar a cabo debe ser leída por la RM1 (bit 03= 1) o por la ROM2 (bit 03= 0). Este bit es normalmente de "uno" y sólo expuesto a "cero" por el microprograma intérprete cuando este último reconoce una interrupción generada por el accionamiento de la llave 100, la cual exige un programa de DBG que reside en ROM2 para indicar que las instrucciones de dicho programa deben ser leídas en la ROM2. El bit 05 es normalmente de uno y se utiliza para habilitar las interrupciones del programador para peticiones de DBG, y es puesto a cero por el intérprete cuando se realiza la interrupción. Los bits 02, 04, 06, 07 no son utilizados por los programas de DBG, y sirven para habilitar otras causas de interrupción.

Byte de prenotación de intemupciones (PI-311) de la figura 9).

Es utilizado por el intérprete para poner en marcha una petición de intemupción contenida en el mismo.

Se acciona una interrupción si el AND entre PI y CP es distinto de cero, como se explicará más adelante (párrafo sobre el intérprete).

Es preparado por los microprogramas asociados a causas de interrupción tanto procedentes de la U.C.-3 como de la U.P.-4. El bit 05 indica en particular una interrupción por DBG.

El modo con el que se fuerza a "uno" el bit 05 del byte de Prenotación de Interrupción, se describirá con mayor detalle más adelante, en el párrafo que se refiere al byte de servicio de puesta a punto (Tabla F).

Byte de modificación de instrucción (MI-315 de la figura 9).

Es utilizado por el microprograma intérprete para modificar el segundo byte de la instrucción que debe realizarse y puede ser preparado por el programador en función de los resultados de las instrucciones precedentes.

Los byte 316, 317 y 318 son utilizados para otros fines que no interesan a la invención, y por consiguiente no se describen.

La ZRM comprende además otro registro OPSR-301 que sirve para contener los parámetros del programa interrumpido. El OPSR-301 es preparado por el intérprete extrayendo los registros y los bytes correspondientes del PSR-300. Cuando termina el programa de interrupción, la última instrucción es siempre la de reposición del programa interrumpido, es decir, es una instrucción que transfiere OPSR-301 a PSR-300. En particular el registro OPSR-301 comprende:

Los registros RB-320, P1-321, P2-322, CO-323, MI-325, 326, los cuales son preparados con el contenido

de los registros correspondientes 310-316 del PSR-300.

El registro 324 contiene el código de interrupción CI/figura 9b) es decir, el código de la causa de interrupción en curso de elaboración en el programa en ejecución. Es preparado por el intérprete antes de activar el programa de interrupción. Las causas de interrupción especificadas por el C.I. han sido divididas en cinco clases, cada una de ellas homogénea, y tratada por un microprograma distinto. A cada clase corresponde un bit de C.I. en particular las clases 1 y 2 corresponden cada una a una sola causa de interrupción y se identifican por el bit 01 y 02, respectivamente. Las clases 3, 4 y 5 son identificadas por el bit 05, 06, 07, respectivamente, y cada una de ellas comprende varias causas de interrupción (como máximo 16 causas) identificadas por los bits 00-03.

La finalidad de registrar en OPSR-301 el C.I. de la causa de interrupción se debe al hecho de que la reposición o no del programa interrumpido depende precisamente del tipo de interrupción. Por ejemplo, si la causa de interrupción es tal que el programa interrumpido no puede ser reanudado, entonces el programa de interrupción termina llamando al operador. Sólo después de la intervención del operador podrá reanudarse el programa interrumpido.

El registro IR-327 contiene la dirección de retorno de PSR-300 a la que corresponde la instrucción que debe realizarse en el momento del retorno.

Es preparado por el intérprete transfiriendo el contenido del registro operativo L07 (direccionador de programa) en el momento de la interrupción.

La ZRM comprende además un registro IPSR-

302 que sirve para contener los parámetros del programa de interrupción. Está formado por un byte CP-333 que indica las condiciones de programa asociadas al mismo. El byte CP tiene el significado que se describe en la figura 9a y es transferido por el microprograma intérprete al registro CP-313 en el momento de la habilitación del programa de interrupción.

El registro IPSR-302 comprende además la dirección del programa de interrupción II-335 (figura 9) el cual es cargado por el microprograma intérprete en el registro L07 de los registros 30 de la figura 2b si se registra en la RAM1; el programa de interrupción.

El registro 302 comprende asimismo el byte de habilitación de interrupciòn AI-334 que se da en la figura 9c en el cual los bits 01-02-05-06 y 07, si se encuentran en el nivel 1, indican que los programas correspondientes a las clases respectivas de interrupción son registrados en la RAM1, y si se encuentran a nivel cero, indican que los programas son registrados en la ROM2.

Más particularmente, el micrprograma intérprete realiza el "AND" lógico entre el código de interrupción CI y el byte de Habilitación de la interrupción AI. Si el AND lógico es cero, significa que el programa asociado a la interrupción se registra en ROM2; si es uno se registra en RAM1.

En el primer caso, el intérprete fuerza el contenido del registro II-335 al registro operativo L07; en el segundo caso, fuerza en el mismo la dirección de la ROM2 de comienzo de la zona B reservada a los programas de DBG.

La ZRM comprende además un registro IS-350 de la figura 9 que contiene la dirección de STOP en la que el operador desea detener la elaboración del programa como se ha

indicado en la parte inicial, y como se explicará mejor más adelante. Este registro es preparado por un programa de DBG utilizando los datos formados en el teclado por el programador.

LA ZRM comprende además un byte de servicio del DBG (BSD-351):

EL BSD-351 se representa con detalle en la figura 9d. Los bits utilizados son el bit 01, que indica (como se explicará mejor en el capítulo del intérprete) si debe o no llevarse a cabo la instrucción presente en el momento de la interrupción. Si dicho bit es igual a cero, se efectua la instrucción; de lo contrario se realiza el programa de interrupción.

El bit 02 indica se ha sido prenotado un stop y es preparado por el programa de DBG asociado al stop direccionado. El bit 03 indica si el conmutador de llave 100 se encuentra en posición normal (bit 03 = 0) o en posición de puesta a punto (bit 03 = 1). El preparado por el microprograma que se muestra en la tabla G utilizando la posición de la llave 100.

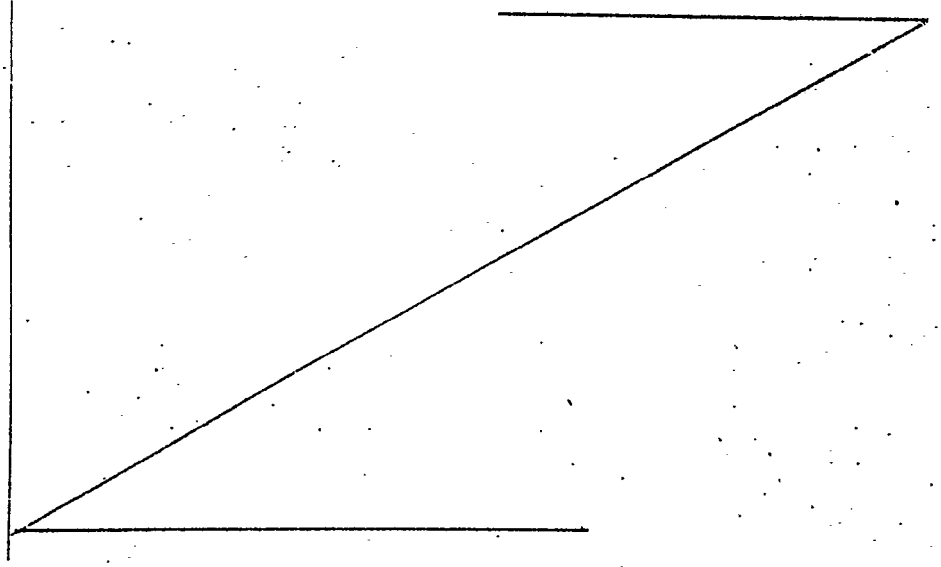


TABLA G

IND.SIMB.	MICRO INSTRUC	CODIGO	I OPERANDO	II OPERANDO
IDISO	C A C 7	MAD	A10	CC7
	9 2 F 7	CRTB	B02	CF7
	S A 2 8	ANDA	A10	B02
	A 8 8 8	TCCA	A08	
	B 8 E 7	TADI	A08	
	9 2 0 0	CRTB	B02	C00
	2 2 3 6	SADO	D02	IDISAO
	9 2 0 8	CRTB	B02	C08
IDISAO	6 A 2 E	ORA	A10	B02
	D A C 7	AMD	A10	CC7
	C 2 B 7	MAD	A02	CB7
	9 2 D 0	CRTB	B02	CD0
	6 2 2 8	ANDA	A02	B02
	B A E 7	TADI	A10	
	9 2 3 0	CRTB	B02	C20
	3 2 4 0	SADI	D02	IDISA1
	3 3 4 0	SADI	D03	IDISA1
IDISA1	6 2 2 E	ORA	A02	B02
	D 2 B 7	AMD	A02	CB7

Con referencia a la figura 2c, el pupitre 7, a través de la llave 100 activa un interruptor 60 el cual es conectado directamente al nudo NA a través del hilo 61 que forma parte del canal 62.

Si la llave 100 se encuentra en la posición NORMAL (N), el interruptor 60 está abierto y por consiguiente

el hilo 61 se encuentra a nivel cero. Esto corresponde, como se ha dicho ya, al funcionamiento normal de la máquina. Cuando por el contrario, la llave 100 se encuentra en posición de PUESTA A PUNTO (DBG), se cierra el interruptor 60, por lo que el hilo 61 está en el nivel 1.

El temporizador 20 genera cada 60 milisegundos una señal TM que provoca una interruptor del micrprograma de nivel 4 (es decir, el microprograma intérprete) y lleva a cabo la microinstrucción direccionada por el registro L01 (direccionador del micrprograma de nivel 3). Esta microinstrucción forma parte de una seguencia predefinida de microinstrucciones que corresponde a las distintas condiciones externas que deben controlarse durante el desarrollo de un programa.

En el instante TM, en particular, está presente en él el 01 la dirección que corresponde al microprograma IDISO que se muestra en la tabla G.

Mediante una microinstrucción MAD, se carga en A10 la célula de RAM2 a la dirección ØØC7, es decir, el byte de servicio DBG, BSD-351. Acto seguido se carga en el registro B02 el número "F7" = 1110111 a través de una CRTA. A través de una ANDA se transfiere a A10 el AND LOGICO entre BSD y "F7", es decir, a A10 si tiene el BSD con el bit 03 = 0.

A continuación, mediante una microinstrucción TCCA se transfiere a A08 el byte presente en el canal 62 (véase figura 2). En particular, si la llave 100 se encuentra en posición DBG, el hilo 61 (que corresponde al bit 02 del canal 62) está en el nivel "1". A través de una TADI, el byte del canal 62 se transfiere a los desviadores 40. Acto seguido, mediante una CRTB se fuerza al registro B02 el número

"00". Posteriormente se prueba el nivel lógico del desviador D02 (en donde está presente la condición del hilo 61 que corresponde a la posición DBG de la llave 100). Si este bit se encuentra en el nivel 1, mediante una CRTB se fuerza en el registro B02 el número "08" = 00001000, y a continuación se lleva a cabo el OR lógico (microinstrucción ORA) entre el contenido del registro A10 y el número "8". Hay que observar que en el registro A10 estaba registrado el BSD con el bit 03 = 0, por consiguiente el resultado de la microinstrucción ORA es el de formar a 1 el bit b03.

Sí, por el contrario, el bit en el hilo 61 se encontraba en el nivel cero, no se realizaba la CRTB que forzaba 00001000 en b02, por lo que no se realizaba el OR lógico entre el contenido del registro A10 y el número "00000000"; esto corresponde a dejar en cero el bit 03 de BSD.

Después de estas operaciones, mediante una AMD se vuelve a escribir en la posición ØØC7 (BSD-351) el BSD así modificado. En definitiva, si la llave 100 se encuentra en posición NORMAL, se tiene el bit 03 = 0; si por el contrario está en posición DBG, el b03 = 1. El mismo microprograma procede además a preparar el bit 05 del byte de prenotación de interrupción (PI). Mediante una microinstrucción MAD se transfiere al registro A02 el byte registrado en la dirección ØØB7 de la RAM1, es decir, el byte PI - 314.

Posteriormente, mediante las dos microinstrucciones CTRB y ANDA se pone a cero el bit 05 de PI. Mediante una TADI se transfiere a los desviadores 40 el contenido del registro A10, es decir, el BSD modificado anteriormente según la condición de la llave 100 del pupitre.

Más tarde se escribe con una CRTB el número

"20" = 00100000 en el registro B02. Posteriormente, mediante dos SADI, se prueban los bits b02 y b03 del BSD anteriormente cargados en los desviadores, que corresponden a la prenotación de una dirección de STOP y al accionamiento de la llave 100 respectivamente. Si al menos uno de los bits probados se encuentra a nivel 1, se coloca en 1, mediante una ORA, el bit b05 del byte PI, el cual, mediante una AMD se vuelve a escribir en la RAM1. Si, por el contrario, ambos bits b02 y b03 de BSD se encontraban a nivel cero, mediante una CRTB se forzaba en el registro B02 el número "10" = 00010000, que corresponde a otra causa de interrupción que no interesa al DBG, y que por consiguiente no se describe.

La ZRM comprende además un registro de 8 bytes denominados registro de trabajo (RL - 352 en la figura 9) que se utiliza como zona de trabajo para acumular los resultados parciales durante el desarrollo de algunas instrucciones ypara proporcionar, a efectos de instrucción, un resultado que no puede contenerse en los registros de los operandos (por ejemplo el resto de la división).

La ZRM comprende además un registro 359 de 8 bites denominados registro de condiciones RC. Cada byte se divide en dos semibytes, los cuales identifican condiciones particulares del programa. En efecto, el registro 359 se utiliza para recoger todas las condiciones significativas del programa que surgen durante la ejecución de instrucciones internas o externas y que, dado su elevado número, no pueden expresarse en el código de condición o que conviene memorizar independientemente del mismo.

De todos los semibytes sólo se explica el contenido del noveno porque es el utilizado por los programas

de DBG como se explicará más adelante. El noveno semibyte ocupa los cuatro primeros bits de la célula ØØCB y es utilizado por las instrucciones de introducción por teclado para preparar el código de la barra 102 que ha concluído una introducción de datos de teclado.

La ZRM comprende además un registro de un byte AB-370 que identifica las barras 102 habilitadas por el programa. En particular este registro es preparado, como se verá más adelante, por los programas de DBG para habilitar las barras S0, S1, S2, S6, ya que solamente éstas tienen significado durante el desarrollo del DBG.

La ZRM comprende asimismo un grupo de ocho registros 360-367 (figura 9) que normalmente son utilizadas por los programas del siguiente modo. Los registros 360 a 363 son utilizados junto con el registro de trabajo 352 para contener los resultados intermedios durante las operaciones de multiplicación y división y los resultados que no pueden ser contenidos en los registros de los operandos. Mas concretamente, las instrucciones de multiplicación y división son realizadas por microprogramas que actuan en dichos registros. Hay que observar que el contenido de tales registros no es significativo a efectos de la instrucción que los ha utilizado, ya que todas las condiciones y los resultados significativos se transfieren a zonas de memoria externas a la ZRM direccionadas por los operandos de las instrucciones específicas.

Los registros 364 y 365 son utilizados por las instrucciones de EDITING (edición) de un registro, es decir, contienen todos los caracteres correspondientes a la puntuación, los caracteres algebráicos (+, -), los espacios, etc, necesarios durante la impresión de una superficie de

memoria.

Son solicitados por los operandos de dichas instrucciones, y su contenido no es significativo a efectos de dicha instrucción. Los registros 366 y 367 son utilizados por el programa de DBG como extensión del registro OPSR-301. Sirven para cargar condiciones significativas del programa interrumpido que no pueden contenerse en el registro OPSR-301. Hay que observar que, si bien los registros 360 a 365, una vez terminada la instrucción que los utiliza, no contienen datos significativos, los registros 366 y 367 contienen datos significativos a efectos de la reanudación del programa interrumpido y por consiguiente pueden ser utilizados por los programas DBG únicamente en casos particulares que se precisarán más adelante.

Debe observarse igualmente que los registros 360 a 367 no deben situarse necesariamente en las posiciones de RAM1 indicadas en la figura 9, sino que pueden encontrarse en cualquier zona de la memoria. Una de las características de la invención consiste, en efecto, en determinar en la RAM1, mediante las direcciones correspondientes, un cierto número de registros (en nuestro caso 8) que no contienen datos significativos a efectos de la ejecución de las instrucciones y utilizar tales registros como registros de apoyo de los programas de puesta a punto. Todo esto se hace naturalmente de manera automática y sin intervención del programador, el cual sólo debe accionar la llave 100 y las barras 102.

Se debe precisar igualmente que como registros de apoyo de los programas de DBG no deben utilizarse necesariamente los registros 360-367, sino que pueden utilizarse

registros reservados exclusivamente a los programas de DBG, que pueden situarse tanto en ZRM, como en la memoria libre, o que incluso pueden ser registros externos a la momoria.

La zona libre de memoria, a saber, la zona inmediatamente posterior a la ZRM contiene asimismo una zona denominada tabla de referencia cuyo emplazamiento es determinado por un registro a ZRM. Este registro ITR - 353 está compuesto por tres bytes los dos primeros de los cuales definen la dirección de comienzo de la tabla y el tercero la longitud de dicha tabla (como máximo 256 bytes). La tabla de referencias es direccionada por algunas instrucciones para calcular las direcciones de los operandos. La zona libre de memoria, inmediatamente posterior a la tabla de referencias, contiene 16 registros de 8 bytes cada uno, denominados registros privilegiados. En efecto, estos últimos pueden ser direccionados directamente por las instrucciones citando en exadecimal su número de referencia. La parte restante de la RAM1 puede ser direccionada además de manera libre.

La ROM2 está dividida en dos zonas A y B (figura 9). La zona A comprende todos los microprogramas necesarios para el funcionamiento del elaborador, la zona B comprende de los programas de DBG.

Como se ha dicho anteriormente, los programas registrados en la RAM1 se realizan instrucción por instrucción. Cada instrucción se lleva a cabo a su vez en dos fases: una fase interpretativa (fase ALFA) y una fase ejecutiva (fase BETA). La base interpretativa es común a todas las instrucciones y es realizada por un microprograma adecuado, denominado intérprete, registrado en la zona A de la ROM2. Esta fase termina con el reconocimiento del formato de la instrucción

que el mismo microprograma intérprete ha leído en la RAM1 ó RAM2 y con la preparación de los operandos en los registros - operativos 30. Este microprograma es solicitado por consiguiente al comienzo de cada instrucción por la microinstrucción de fin de ejcución de la instrucción que acaba de realizarse.

En particular, pues, la ejecución de cualquier instrucción del programa que realiza el elaborador se lleva a cabo del siguiente modo:

La última microinstrucción del microprograma que ha realizado la instrucción precedente es una microinstrucción SAI de salto incondicionado a la dirección IALFA (Tabla H) es decir, a la primera microinstrucción del microprograma intérprete.

A continuación se da la tabla H, en la que se enumeran las microinstrucciones correspondientes al microprograma intérprete.

La primera columna indica el nombre simbólico de las direcciones de salto que se utilizarán como operando en las microinstrucciones de salto. La segunda columna contiene la microinstrucción en código exadecimal; las tercera, cuarta y quinta, indican la instrucción en forma simbólica, es decir, la función realizada, el primer operando y el segundo operando, respectivamente.

Se hace notar que si un operando es indicado con la letra C seguida por dos caracteres alfanuméricos, ésto significa que el operando es el número exadecimal (1 byte) que sigue a la letra C.

Hacemos ahora referencia a la tabla H, a la figura 9 y a las figuras 10a, 10b y 10 C.

Mediante las dos primeras microinstrucciones

CRTA y AMD, se fuerza a la dirección ØØB8 de la RAM1 el carácter CRT '00', es decir, se pone a cero el byte de modificación de instrucción (bloque 200 de la figura 10a); ésto es necesario cuando se quiere iniciar una mueva instrucción. Mediante las dos microinstrucciones sucesivas, es decir MAD y TAB, se transfiere al registro operativo B15 del grupo de registro 32 de la figura 2b el contenido de la célula ØØB6, la cual, como se ha dicho anteriormente, contiene el byte de las condiciones de programa CP-313 (bloque 201).

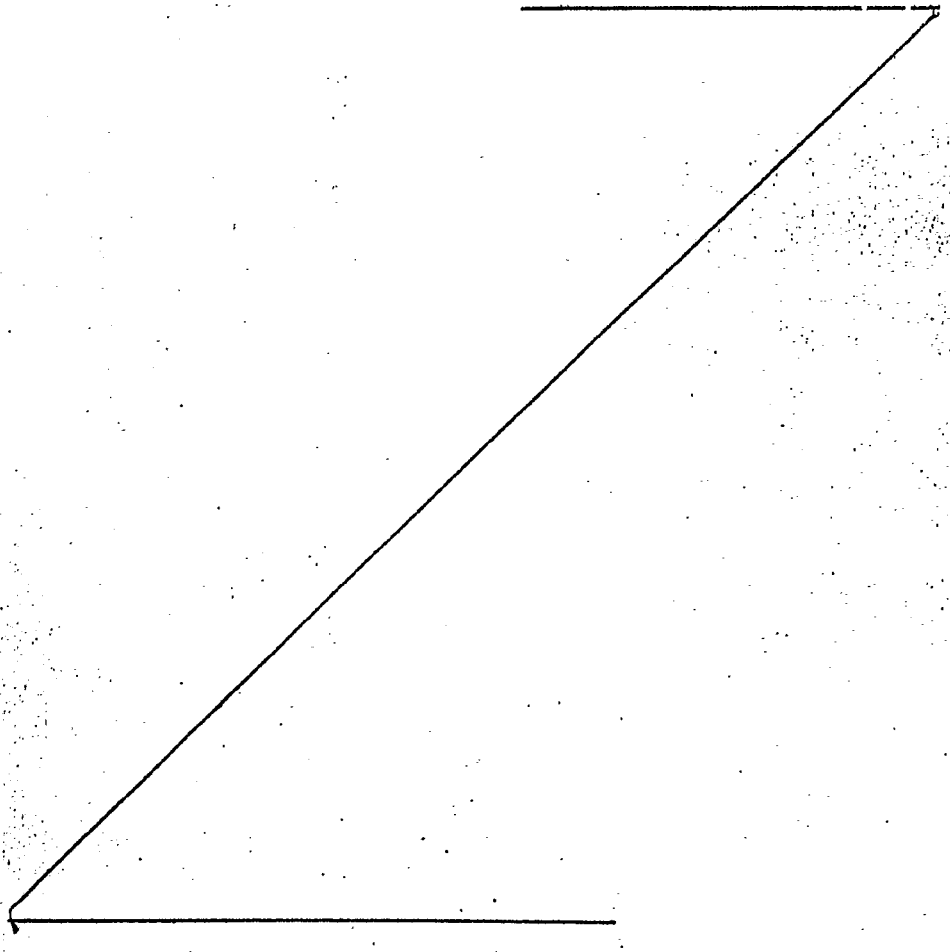


TABLA H

IND.SIMB	MICRO INSTRUC	CODIGO	I OPERANDO	II OPERANDO
IALFA	8 8 0 0	CRTA	A08	C00
	D 8 B 8	AMD	A08	CB8
IALFAJ	C E B 6	MAD	A14	CB6
	5 E F C	TAB	A14	B15
	C E B 7	MAD	A14	CB7
	6 E F 0	AND	A14	B15
	2 1 E 1	SADO	D01	IINTE
IALFAR	C E B 6	MAD	A14	CB6
	B E E 7	TADI	A14	
	3 3 1 3	SADI	D03	IALFA1
	4 7 2 F	SLL	L07	L02
	7 8 0 0	ROMA	A08	
	5 8 E C	TAB	A08	B14
	7 8 0 0	ROMA	A08	
	5 8 F C	TAB	A08	B15
	4 2 7 F	SLL	L02	L07
	0 2 1 5	SAI	IALFA2	
IALFA1	E 7 E 5	MBIP	M07	B14
	E 7 F 5	MBIP	M07	B15
IALFA2	C 8 B 8	MAD	A08	CB8
	B 0 6 6	REDI	D00	
	6 8 F 5	ADDB	A08	B15
	9 2 0 2	CRTB	B02	C02
	5 2 E 3	TBA	A02	B14
	7 B 0 0	ROMA	A11	
	7 2 0 0	ROMA	A02	
	5 B 2 C	TAB	A11	B02

El byte de prenotación de interrupción PI contiene en forma codificada, como hemos visto ya, las causas que pueden provocar una interrupción del programa en ejecución. A continuación la microinstrucción AND efectua el y lógico entre C.P.-313 y P.I.-314 (alternativa lógica 202).

Como se ha dicho anteriormente (véase tabla A), la microinstrucción de AND envía a uno el desviador D01 del grupo de desviadores 40 si el resultado del AND es cero. La microinstrucción sucesiva SADO se prueba el contenido del desviador D01, y si éste último está en el nivel lógico cero (es decir, si ha sido prenotada una interrupción), se efectua un salto a la dirección dada por el nombre simbólico IINTE (bloque 250) en donde se registra la primera microinstrucción de un microprograma que trata las interrupciones (véase tabla J). Si por el contrario, el desviador está en el nivel lógico 1 se prosigue con la fase ALFA de lectura de la RAM1.

TABLA J

IND.SIMB.	MICRO INSTRUC.	CODIGO	I OPERANDO	II OPERANDO
IINTE	6 E F 4	ANDB	A14	B15
	C A C 7	MAD	A10	CC7
	B F F 7	TBDI	B15	
	B A E 7	TADI	A10	
	2 1 6 1	SADO	D01	IINTE1
	2 2 6 7	SADO	D02	IINTE2
	3 3 6 7	SADI	D03	IINTE2
	C 8 E D	AMD	A08	CED
	5 8 8 C	TAB	A08	B08
	C 8 E C	MAD	A08	CEC
	6 8 7 3	ORE	A08	B07

(Tabla J continuación)

		2 1 6 1	SADO	D01	IINTE1
		6 7 8 3	ORE	A07	D08
		3 1 6 7	SADI	D01	IINTE2
5	IINTE1	9 A 0 2	CRTB	B10	C02
		6 A A E	ORA	A10	B10
		D A C 7	AMD	A10	CC7
		C A B 6	MAD	A10	CB6
		5 A F C	TAB	A10	B15
10		0 2 0 7	SAI	IALFAR	
	IINTE2	9 A F D	CRTB	B10	CFD
		6 A A 8	ANDA	A10	B10
		D A C 7	AMD	A10	CC7
		9 D 0 4	CRTB	B13	C04
15	IERR01	8 E B 0	CRTA	A14	CB0
		9 8 0 0	CRGB	B08	C00
		8 8 D 0	CRTA	A08	CD0
		8 B 0 A	CRTA	A11	C0A
	INTER1	E E 9 D	MAIP	A14	A09
20		E 8 9 9	AMIP	M08	A09
		A B 4 A	DCA	A11	
		2 1 7 4	SADO	D01	INTER1
		E 8 7 1	BMIP	M08	B07
		E 8 7 B	AMI	M08	A07
25		8 8 D 7	CRTA	A08	C07
		E 8 D 3	BMI	M08	B13
		D B B 8	AMD	A11	CB8
		C 9 B D	MAD	A09	CBD
		5 B D 2	TBA	A11	B13
30	INTER5	6 B 9 0	AND	A11	B09

TABLA J (continuación)

	2 1 8 E	SADO	D01	INTER2
	8 B 0 0	CRTA	A11	C00
INTER4	D B B 6	AMD	A11	CB6
INTER4	9 7 1 7	CRTB	B07	C17
	8 7 0 0	CRTA	A07	C00
	0 2 0 0	SAI	IALFA	
INTER2	C 9 B C	MAD	A09	CBC
	D 9 B 6	AMD	A09	CB6
	C 7 B E	MAD	A07	CBE
	5 7 7 C	TAB	A07	B07
	C 7 B F	MAD	A07	CBF
	0 2 0 0	SAI	IALFA	

Examinemos a continuación las condiciones que determinan el salto a IINTE o la prosecución del microprograma intérprete.

Si el byte P.I. es 0000 0000, el AND, con el byte C.P. será sin duda 0000 0000, por lo que el desviador D01 se encuentra en el nivel 1 y por lo tanto no se efectua el salto. La condición P.I. 0000 0000 indica que no ha sido prenotada ninguna interrupción. Si al menos un bit de P.I. está en el nivel 1, y el bit correspondiente de C.P. es también 1, es decir, si queda habilitada esa particular microinterrupción, entonces el AND entre C.P. y P.I. pondrá en 1 el bit del mismo peso que el del P.I. Si, por ejemplo, ha sido prenotada una interrupción a través de la llave 100 de pupitre (figura 1b) el byte P.I., como se ha visto anteriormente, tiene la configuración 0010 0000. El byte C.P., a su vez, ha si-

do colocado por el programador con la conformación 1X1X XXXX, dado que el programa en curso preve la habilitación de la interrupción de DBG, por lo cual el AND entre C.P. y P.I. será 0010 0000. Por consiguiente D01 será obligado a pasar a cero por la microinstrucción AND y en consecuencia se tendrá el salto a IINTE.

Si no ha habido interrupdión, mediante las microinstrucciones MAD y TADI se transfiere a los desviadores 40 todo el byte C.P. (bloque 203). Acto seguido se controla con una microinstrucción SADI si el desviador D03 se encuentra a nivel 1 ó a nivel 0 (alternativa lógica 204).

Como se ha dicho, el bit 03 del byte C.P. indica si la lectura de la instrucción debe hacerse por parte de la RAM1 o de la ROM2.

Si la lectura de la instrucción la debe hacer la RAM1 (caso de los programas normales), con las dos microinstrucciones MBIP se leen el primero y el segundo byte de la instrucción en la dirección contenida en el registro largo L07 (bloque 205). El registro 207 se incrementa en 1 en cada lectura (MBIP). Los dos bytes se transfieren además a los registros B14 y B15 de los registros operativos 32. Por todo lo expuesto se comprueba que el registro L07 funciona de direccionador de programa para las instrucciones, ya que indica siempre la dirección de la instrucción sucesiva después de cada lectura. Además, en caso de que la instrucción precedente a la instrucción en curso fuese una instrucción de salto, habría efectuado ya en el registro L07, a través de un microprograma adecuado realizado en la fase BETA, la dirección de la RAM1 de la instrucción a la que saltar. En conclusión, tanto para el direccionamiento secuencial como para el direcciona-

miento con salto, el registro L07 contiene siempre la dirección de la RAM1 de la instrucción sucesiva.

Si la instrucción en curso debe ser leída por la ROM2 (caso de los programas de DBG) el bit 03 de las condiciones de programa habría sido forzado a cero por lo cual el salto en IALFA1 no se lleva a cabo. Se realiza sin embargo la microinstrucción SLL que cambia el contenido del registro L07 por el del registro L02 (bloque 206). Por consiguiente, con dos pares sucesivos de instrucciones ROMA y TAB se transfieren a los dos registros B14 y B15 los dos primeros bytes de la instrucción leída (bloque 206).

Hay que observar que el cambio de L07 y L02 se hace necesario por el hecho de que la microinstrucción ROMA que realiza la lectura de la ROM2 es direccionada ezclusivamente por el registro L02. Además, esta microinstrucción incrementa en 1 el contenido de dicho registro después de cada lectura. Después de la lectura del primero y segundo byte de la instrucción se repone el contenido del registro L07 efectuando una microinstrucción de conmutación SLL, mediante la cual el registro L02 se cambia con el registro L07. Hay que notar por lo tanto que en ambos casos, lectura de la RAM1 y de la ROM2, en los dos registros B14 y B15 de los registros 32 se encuentran registrados los dos bytes de la instrucción que hay que realizar y después de dicha lectura el registro L07 contiene ya la dirección de la instrucción sucesiva. Después de ésto con tres microinstrucciones MAD, REDI, ADD3 (bloque 207 de la figura 10a) se modifica el segundo byte de la instrucción leída según el byte de modificación de instrucción (MI)anteriormente registrado en la célula ØØB8.

Mediante las microinstrucciones CRTB y TBA

(bloque 208), se utiliza el primer byte de la instrucción para calcular una dirección de ROM a partir de la cual se escriben pares de bytes que corresponden a los formatos de las instrucciones (1º semibyte) y las direcciones de las fases ejecutivas (2º, 3º y 4º semibyte) asociados agrupos de instrucciones, constituyendo tales pares de bytes los elementos que caracterizan la instrucción. Por consiguiente, con dos microinstrucciones ROMA consecutivas, se leen los dos bytes asociados a la instrucción y se transfieren al registro operativo LO2 (figura 8). En este punto la instrucción contenida en LO2 es examinada por un microprograma que reconoce uno de los site posibles - formatos que se dan en la figura 10a.

Si la instrucción reconocida es del formato 1, está compuesta por un código de función F, los bits I1 e I2 que indican si los respectivos registroa R1 y R2 direccionan de modo directo o indirecto la RAM1 y dos campos de cuatro bits R1 y R2 que indican dos constantes. Si I1 - I2 = 0 las direcciones de los operandos se calculan multiplicando por 8 las constantes R1 y R2 y sumando al resultado el valor del registro base RB-310. De este modo la RAM1 puede ser direccionada por registros de 8 bytes. Si I1 = 1 e I2 = 0, R1 indica uno de los diez y seis registros privilegiados de la RAM1 y R2 tiene el mismo significado anterior. En este caso el primer operando se lee en la zona de la RAM1 direccionada por el contenido de R1, mientras que el segundo operando se calcula como se ha dicho anteriormente. Todos los casos: I1 = 0 e I2 = 1 o bien I1 = I2 = 1 pueden deducirse de los precedentes.

Si la instrucción es del formato 2 está compuesta por un código de función F, un bit I y dos campos R1 y L2. El primer operando se calcula como en el formato 1 (I1 =

0, I1 = 1) mientras que el segundo operando se calcula sumando a RB el indicador P1 ó P2 especificado por el código F. El campo L2 contiene el número de bytes del segundo operando que hay que leer a partir de la dirección calculada.

Si la instrucción es del formato 3 está compuesta por un código de función F y por dos campos L1 y L2. Las direcciones de los operandos se calculan como las del segundo operando del formato 2 y las longitudes de los operandos vienen especificadas por los campos Li y L2.

Si la instrucción es del formato 4 está compuesta por dos campos, uno de los cuales indica el código de función F, y el otro un campo E, MD, I y L que pueden asumir cuatro significados distintos según el contenido de F. El código F direcciona los dos operandos por medio del indicador P1 y P2 como para el formato 3 y además especifica el significado del segundo campo.

Si la instrucción es del formato 5 está compuesta por un código de función F en el que el primer operando se calcula como para el formato 3 y el segundo para el formato 1.

Si la instrucción es del formato 6 los dos bytes de la instrucción son utilizados directamente en la fase BETA sucesiva.

Si la instrucción es del formato 7 está compuesta por cuatro bytes para lo cual transfiere primero los bytes 1 y 2 a otros registros 30 y después pone en B14 y B15 los bytes 3 y 4 direccionados por el registro L07. Está compuesta por un código de función F, un campo E que indica el elemento de la tabla de referencias a partir del cual se calcula la dirección del operando en un campo LD que indica el

desplazamiento respecto a la dirección así calculada.

Después de estas operaciones, el microprograma intérprete termina su misión y comienza por lo tanto la fase ejecutiva BETA en la que se elaboran los operandos calculados anteriormente. Las instrucciones realizadas por los programas de DBG se dan en la tabla L en la que se indica el formato respectivo.

Si durante la fase ALFA se reconoce una interrupción (alternativa lógica 202 de la figura 10a) el microprograma intérprete realiza un salto a la dirección IINTE (Tabla 5 y figura 10b). La primera microinstrucción ANDB efectua de nueve el AND entre C.P. y P.I. y conserva su resultado en el registro B15 (bloque 251). A continuación dicho resultado se transfiere a los DEV-40 mediante la microinstrucción TBDI, y estos últimos se sondean para reconocer la causa de interrupción (bloque 252). No nos ocupamos de las posibles causas de interrupción, a excepción de las debidas a la intervención del programador para solicitudes de DBG. En este caso, mediante la microinstrucción MAD, la RAM1 lee el byte de servicio de DBG y posteriormente este último es transferido a los desviadores 40 mediante la microinstrucción TADI (bloque 256&. Después de ésto se sondean los desviadores DO1, DO2 y DO3, los cuales contienen el bit 01, 02 y 03 del BSD, respectivamente.

Este bit se encuentra normalmente a nivel cero por lo que se efectua un salto a la dirección IINTE1 en la que dicho bit se coloca en 1 y en la que se efectua un salto a la dirección IALFAR. Esto se realiza para hacer operante la interrupción no durante la instrucción en curso, sino el final de la misma. A continuación se prueba el desviador DO3 (bloque 259) del cual disñcrimina el STOP direccionado

por todos los demás programas de DBG.

El control del bit 03 se efectua eespués del bit 02 por lo siguientes motivos:

Supongamos que el programador prenota una dirección de STOP, es decir, quiere que se interrumpa la ejecución del programa a la instrucción correspondiente a la dirección de STOP. Como se describirá mejor más adelante, ésto hace que se interrumpa el bit 02 del BSD y se lleva a "uno" junto con el byte P.I. Esta disposición IINTE (alternativa lógica 202) y de aquí llega a la alternativa lógica 259.

La alternativa lógica 252 discrimina si además del STOP direccionado, el programador ha pedido otro programa de DBG. Esta posibilidad surge cuando el operador se da cuenta de que la petición de STOP que acaba de efectuar no le es de ninguna utilidad, y por el contrario quiere por ejemplo efectuar un programa especial de DBG grábado en la tarjeta magnética 9. Esta posibilidad corresponde a tener en "uno" en el BSD los bits 02 y 03 simultáneamente. Esto provoca un salto del microprograma intérprete a IINTE 2. Se comprende que IINTE2 se deshabilitará el STOP direccionado y se realizará el programa de DBG seleccionado por el programador. En el ejemplo citado, este programa provoca la lectura y la ejecución del programa grabado en la C.M-9.

En definitiva, resulta evidente que la búsqueda de una dirección prenotada de STOP se realiza únicamente si no hay otras peticiones alternativas de DBG, y por lo tanto es el programa de DBG menos prioritario; en efecto el bit 02 del STOP direccionado es deshabilitado por cualquier otra petición de DBG.

En caso de que la única petición de DBG sea

el STOP direccionado, se realizan las microinstrucciones MAD, TAB,)bloque 260) mediante las cuales se transfieren al registro L08 (30 de la figura 2b) los dos bytes de la dirección de STOP prenotado tomados de las células ØØEC y ØØED (Registro IS-350, figura 9). A continuación se realiza, a través de las microinstrucciones ORE, SADO, ORE, el OREX entre el contenido de los registros L07 y L08 (30 de la figura 2b), es decir, se efectua la comparación entre la dirección de programa y la dirección prenotada.

Si las dos direcciones son iguales, la microinstrucción ORE pone en 1 el desviador D01.

En este caso la microinstrucción SADI provoca un salto a IINTE2 (alternativa lógica 262) y, como se explicará más adelante, tendrá lugar la visualización de la instrucción registrada en la dirección prenotada.

En el caso de direcciones distintas se llevan a cabo las microinstrucciones CRTB, ORA y AMD, que provocan la escritura en la RAM1 a la dirección ØØC7 del BSD en el que, no obstante, el bit 01 ha sido colocado a nivel 1(bloque 265). A continuación se realizan las microinstrucciones MAD y TAB las cuales reponen en el registro B15 y byte CP (bloque 266). Posteriormente se realiza una microinstrucción SAI que provoca un salto incondicionado a la dirección IALFAR (bloque 203 de la figura 10a).

De este modo, tanto si el bit 01 del BSD se encuentra a nivel 0 (alternativa lógica 256), como si las direcciones son distintas, se coloca siempre en 1 el bit 01 de BSD y por lo tanto se continua hasta la interpretación de la instrucción, por lo cual, durante la siguiente instmcción, el microprograma intérprete realiza de nuevo la comparación

de las direcciones. En caso de que las direcciones sean iguales, o si hay una petición de DBG distinta de la prenotación STOP se salta a la dirección IINTE2.

Se realizan las microinstrucciones CRTA y ANDA las cuales ponen a 0 el bit 01 del BSD (bloque 270); es decir, que se repone, después de activada la interrupción, la condición normal de dicho bit. A continuación el BSD así modificado se vuelve a introducir en la RAM1 a la dirección ØØC7 mediante las microinstrucciones AMD y CRTB (Bloque 271).

De esta manera se anula la prenotación del STOP. Acto seguido se realiza la microinstrucción CRTB mediante la cual se escribe en el registro B13 el byte CI "0000 0100" el cual implica que la causa de la interrupción es un programa de DBG.

A partir de esta microinstrucción tiene lugar la interrupción efectiva del programa en curso; en efecto, se realizan las ocho microinstrucciones sucesivas a las contenidas en la dirección de ROM2, IERR01, las cuales provocan la transferencia de los diez primeros bytes de los parámetros del programa en ejecución registrados en el registro 300-PSR en las células correspondientes del registro 301-OPSR (bloque 273).

A continuación, con las microinstrucciones BMIP y AMI se registran las direcciones ØØDA y ØØDB de la RAM1 el contenido del registro L07, es decir, se salva en el registro OPSR-301 el direccionador de programa del programa interrumpido.

Más adelante, a través de la microinstrucción CRTA y BMI, se escribe en la dirección ØØD7 (registro

324 de la figura 9) el código de interrupción anteriormente preparado.

De esta manera se salvan en el registro OPSR-301 todos los parámetros que permitirán la reanudación del programa interrumpido una vez terminado el programa de interrupción.

Como veremos más adelante, todos los programas de interrupción terminan con las reposición del registro OPSR-301 en el registro PSR-300. Después de esta fase el microprograma intérprete prepara el registro 300 con los parámetros del programa de interrupción que debe llevarse a cabo en substitución del programa interrumpido. En particular, con las microinstrucciones AMD (bloque 280 de la figura 10a) pone a cero el byte M.I. del registro PSR-300 y con las microinstrucciones MAD y TBA, TAB, transfiere al registro B09 el byte A.I. registrado en la célula ØØBD de la RAM1 (registro AI-334) y al registro A11 el byte C.I. anteriormente preparado con "00000100" (bloque 281).

Como se ha dicho, el byte A.I. es preparado por el programador para definir si el programa de interrupción se encuentra en la ROM1 ó en la RAM2. En este caso particular, por las razones indicadas en la introducción, se ha preferido registrar los programas de DBG en la ROM2, pero ésto no cierra al usuario la posibilidad de registrar en la RAM1 los propios programas de DBG, si así lo precisan. Para hacer esto, basta con activar el bit 03 de AI-334, A continuación se realiza el AND lógico entre AI y CI que pone D01 = 1 si el AND es distinto de 0000 0000.

La microinstrucción SADO controla el desviador D01 y si este último se encuentra en el nivel 1 (progra-

ma registrado en ROM) realiza dos microinstrucciones CRTA y AMD, las cuales ponen a cero el byte CP del programa de interrupción en curso (bloque 283).

De este modo se observa que el microprograma intérprete, después de reconocer que la causa de la interrupción es un programa de DBG, procede a deshabilitar cualquier otra posible interrupción (CP = 0000 0000) en cuanto que estas últimas serían incompatibles con los programas de DBG.

Después de ésto, realiza las microinstrucciones CRTB y CRTA mediante las cuales fuerza la dirección 17ØØ de ROM2 en el registro L07 (bloque 284), y acto seguido efectua un salto incondicionado a la dirección simbólica IALFA (figura 10a) para interpretar la primera instrucción del programa de DBG.

La dirección 17ØØ corresponde al comienzo de la zona B de la ROM de la figura 9. Hay que notar que entre otras cosas, el intérprete ha forzado también a cero el bit 03 de CP-313 por lo cual, cuando vuelva a IALFA, la lectura de las instrucciones será realizada por la ROM2 y no por la RAM1.

Hay que observar además que el microprograma intérprete anula también el bit 05 de CP-313 correspondiente a las instrucciones de DBG porque de lo contrario, después de haber lanzado el programa de DBG, entraría en un circuito cerrado.

En efecto, al realizar la primera instrucción del programa de DBG, el microprograma intérprete, en la alternativa lógica 202 (figura 10a), encontraría habilitada la interrupción por DBG ya que PI no ha variado y el bit 05

de CP=1. Por consiguiente saltaría a IINTE y, a través de los bloques 250 a 284 (figuras 10 b y 10 c), saltaría de nuevo a la dirección IALFA y no saldría ya del mencionado círculo.

Finalmente, en caso de que el programa de interrupción sea registrado en RAM1 (alternativa lógica 282), el intérprete efectua las microinstrucciones MAD, AMD, MAD, TAB, MAD (bloque 285 de la figura 10c) mediante las cuales fuerza en el registro direccionadlr del programa en ejecución LO7 la dirección contenida en los bytes 3º y 4º del registro IPSR-302 y carga además en el registro CP-313 el byte CP-333 del registro IPSR-302, instaurando así en el registro PSR-300 las nuevas condiciones del programa de interrupción.

1. Instrucciones utilizadas.

Se ha visto anteriormente que el funcionamiento de la llave 100 o bien el reconocimiento de una dirección de STOP prenotada previamente tiene como consecuencia que el bit 05 de PI-314 y el bit 03 de BSD son forzados a un nivel 1. Se ha visto igualmente que si el programador ha registrado los programas de DBG en la ROM2 ha cuidado de colocar a "0" el bit 01 del byte AI-334 del IPSR-302.

Se ha visto, por último, que la presencia simultánea de los valores de los bits que se acaban de mencionar, condicoona el microprograma intérprete para que interrumpa la elaboración del programa en curso, para salvar sus parámetros significativos y para forzar en el direccicnador del programa LO7 la dirección "17ØØ" de ROM2.

Esta dirección es la dirección de la instrucción inicial del programa de DBG memorizado en la ROM2.

Antes de proceder a la descripción del pro-

grama de DBG conviene explicar con relación a la tabla K el significado de las instrucciones utilizadas por el mismo.

En la tabla K la primera columna indica para cada instrucción el formato descrito anteriormente y al que pertenece la misma instrucción, la segunda columna contiene una breve descripción de la operación efectuada por la instrucción y la tercera y cuarta columnas indican el código simbólico y el código de máquina, respectivamente, en exadecimal de la citada instrucción.

Tabla K

FORM.	DESCRIPCION	COD.SIMB.	CODIGO DE MAQUINA 1º BYTE	2º BYTE
1	R1 ←R1 + R2	AR	C $00X_1X_2$	R1 R2
1	R1 ←R1 - R2	SR	Ø $01X_1X_2$	R1 R2
1	R1 ←R2	LR	Ø $10X_1X_2$	R1 R2
1	R2 ←REG. DE TRABAJO	LAX	6 $110X_2$	B R2
2	EMPAQUETA	PK	5 $00X_1$1	R1 L2
3	INTR.VISUALIZ.PARA	YOP	6 E	L1 L2
3	DESEMP. Y TRANSC. HEXADEC.	YTX	D 6	L1 L2
4	P1 ←(P1+(MD+1))	AP,1	3 C	MD
4	P1 ←(P1-(MD+1))	SP,1	3 E	MD
4	P2 ←(P2+(MD+1))	AP,2	3 D	MD
4	P2 ←(P2-(MD+1))	SP,2	3 F	MD
4	P1 ←E	TL,1	B 8	E
4	P2 ←E	TL,2	B 9	E
4	INDICAD.1 ←INDICAD.2	MVC	C 5	L

(Tabla K continuación)

4	COST 'I'⟶MEM(P1)	MVI,1	C	6		I
4	COST 'I'⟶MEM (P2)	MVI,2	C	7		I
4	COMPAR.COST. 'I'-INDICAD.1	CBI,1	B	C		I
4	COMPARAR.COST 'I'-INDICAD.2	CBI,2	B	D		I
4	HABILIT.DE BARRAS	KES	A	7		I
4	ESPERA DEL FINAL DE LA TRACCION	WAIT	A	5		I
5	P1 ⟶R2	TRD,1	6	$110X_2$	Ø	R2
5	P2 ⟶R2	TRD,2	6	$110X_2$	1	R2
5	R2 ⟶P1	LPD,1	6	$110X_2$	2	R2
5	R2 ⟶P2	LPD,2	6	$110X_2$	3	R2
6	IP ⟶(IP+SD) (SKIP)	F	7	3	00	SD
6	IP ⟶(IP-SD) (SKIP)	R	7	3	01	SD
6	MOD.BIN.SEGUN P1	MBD,1	9	A	00	SD
6	MOD.BIN.SEGUN P2	MBD,2	9	B	00	SD
6	COMMUTA RB⟹ P1	YBP,1	A	E	Ø	2
6	COMMUTA P1⟹P2	TCP	3	8	Ø	2
6	CONVERS.DECIM ⟶BINAR	CVB	3	8	Ø	1
6	ENVIO COM.U.P.SEG.P1	STIO,1	A	Ø	M	Ø
6	ENVIO COM.U.P.SEG.P2	STIO,2	A	1	M	Ø
6	ENCIENDE LA LUZ DE "ERROR"	ON KBE	E	4	Ø	Ø

FORM.	DESCRIPCION	COD.SIMB.	CODIGO DE MAQUINA 1º BYTE		2º BYTE	
6	AND.COST.K(.)POS.P di RØ	NI	2	Ø	P	K
6	AND.COST.K(.)POS.P di RC	NIC	2	8	P	K
6	OR.COST.K(+)POS. di RO	OI	2	9	P	K
6	OPSR en PSR y realiza	YPS	3	8	Ø	9

FORM.	DESCRIPCION	COD.SIMB.	CODIGO DE MAQUINA 1º BYTE		2º	3º 4º
7	SALTO DIR.	BD	7	Ø	E	LD
7	SALTO COND.VERIF.F.	BDC	F	OOCC	E	LD
7	CARGA 'P1'DIR.F.LARGO	TLD,1	F	8	E	LD
7	CARGA 'P2'DIR.F.LARGO	TLD,2	F	9	E	LD
7	ACTIVA EL PROGR. C.M.	LAC	A	B	E	LD

Examinemos ahora brevemente los pasos que hay que realizar para elaborar cualquier instrucción de tabla K.

Como se ha visto, el microprograma intérprete lee los dos bytes de la instrucción en la RAM1 ó en la ROM2. Según el contenido e tales bytes, el microprograma intérprete reconoce el formato asociado a la instrucción, según el formato, calcula la dirección de los operandos, extrae de la RAM1 los operandos y los transfiere a los registros operativos B14 y B15. En este punto, según el contenido del campo F de la instrucción se extraen de la ROM2 los microprogramas que realizan la instrucción. Una vez terminada la ejecución de la instrucción siguiente.

Con referencia a la tabla K se explica a --

continuación la simbología utilizada en las columnas del código simbólico y en el código de máquina de las instrucciones. Con R1 y R2 se indica uno de los diez y seis registros privilegiados de la RAM1. La dirección del primer operando R1 o del segundo operando R2 se contiene en el registro indicado en la instrucción (direccionamiento directo), si el bit correspondiente X1 ó X2, respectivamente, se encuentra en cero. Si dicho bit está a nivel 1, el registro correspondiente direcciona una zona de la memoria en la que está registrada la dirección del operando.

L1 y L2 indican el número de bytes menos 1 de los operandos, direccionado por el índice P1 y P2.

MD es el valor menos 1 de la constante que debe ser sumada o restada al valor del índice P1 y P2.

E es el número de referencia que contiene la dirección que hay que forzar en P1 ó P2. I indica el operando inmediato equivalente al byte que debe utilizarse en la instrucción. SD indica el valor que debe añadirse o restarse al direccionador de programa para obtener la dirección de salto (SKIP).

A continuación se describe, con referencia a la tabla L y a las figuras 11 a 11g, el programa de DBG registrado en ROM2.

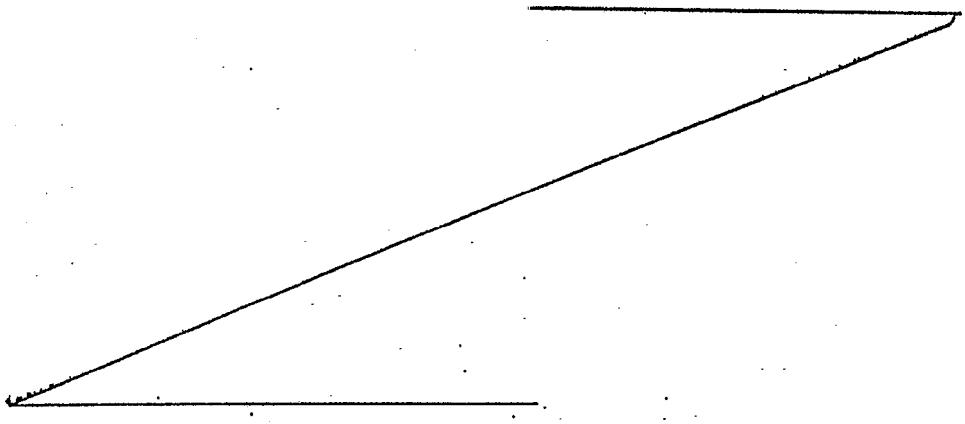
El programa de DBG registrado en ROM2 se divide en varios bloques B100, B0, B1, B2, B6, funcionales, el primero de los cuales B100, es común a todos los demás; los sucesivos, B0, B1, B2, B6 por el contrario, pueden ser solicitados selectivamente por las barras S0, S1, S2, S6 que se indican en la figura 1b en conjunto con el número 102.

Debe precisarse que todas las barras 102

(S0-S6) se utilizan también cuando la llave 100 se encuentra en posición NORMAL. En este caso asumen cada una un significado particular que les asigna el programa en curso de ejecución. Por ejemplo, las barras S0 y 56 pueden ser interpretadas por el programa como fin normal de introducción de datos por el teclado 5 y repuestas por las elaboraciones utilizando tales datos. Las restantes barras S1-S5, además de cerrar la introducción normal de los datos añaden a tales bloques de datos informaciones suplementarias establecidas por el programador.

Naturalmente no se describe el funcionamiento del elaborador cuando la llave 100 se encuentra en posición NORMAL, sino que se describirán las operaciones asociadas a las barras 102 cuando la llave 100 se encuentra en posición DBG.

A continuación se describe el bloque de comienzo B100 del programa común a todos los demás programas. Este vloque funcional salva otros parámetros del programa interrumpido que se contienen en OPSR-301 y prepara además la visualización de la dirección y de la instrucción que debía realizarse en el momento de la interrupción.



PROGRAMA DE PUESTA A PUNTO - TABLA L

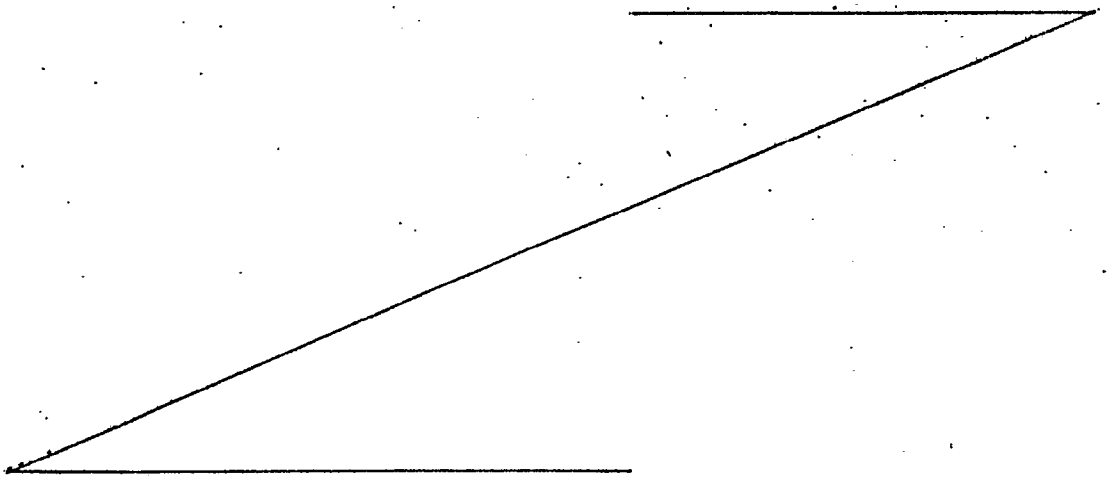
DIR. ROM2	CODIGO SIMBOLICO	INSTRUCCION	COMENTARIO
17 ØØ	TLD,2	F 9 Ø Ø Ø 1 6 Ø	CARGA Ø16Ø en P2
Ø4	YBP,2	A F Ø 2	CAMBIA P2 POR RB (RB=Ø16Ø)
Ø6	LAX	6 C B 4	DEPOSITA R.L. EN R4
Ø8	TL	B 8 Ø Ø	DEPONE A CERO EL INDIC.1
ØA	YBP,1	A E Ø 2	CAMBIA P1 POR RB (RB=ØØØØ)
ØC	TLD,1	F 8 Ø Ø Ø 1 8 8	CARGA Ø188 EN P1
1Ø	TLD,2	F 9 Ø Ø Ø Ø C A	CARGA ØØCA EN P2
14	MVC	C 5 Ø 1	TRANSF.2 BYTE DE ØØCA A Ø188
16	AP,1	3 C Ø 1	SUMA 2 A P1 (P1=Ø18A)
18	AP,2	3 D 1 9	SUMA 1A A P2 (P2= ØØE4)
1A	MVC	C 5 Ø Ø	TRANS. 1 BYTE DE ØØE4 A Ø18A RESERVA BYTE "HABILITACION BARRAS"
1C	SP,2	3 F Ø C	RESTA ØD DE P2(P2=ØØD7
1E	SP,1	3 E 2 1	RESTA 22 DE P1(P1=Ø168
2Ø	MVC	C 5 Ø Ø	TRANS. 1 BYTE ØØD7 A Ø168 (SALVA C.I.)
22	AP,2	3 D Ø 2	SUMA Ø3 A P2(P2=ØØDA)
24	SP,1	3 E B 5	RESTA B6 DE P1(P1=ØØB2
26	MVC	C 5 Ø 1	TRANS. 2 BYTE EN ØØDA A ØØB2 (SALVA IP DE OPSR EN P1 DE PSR)

PROGRAMA DE PUESTA A PUNTO - TABLA I

DIR.	CODIGO SIMBOLICO	INSTRUCCION	COMENTARIO
17 28	AP,2	3 D 8 5	SUMA 86 A P2(P2=Ø16Ø
2A	LPD	6 C 2 0	TRANS. P1 AN RO (RO= I.P.DI OPSR)
2C	YBP,2	A F Ø 2	CAMBIA P2 POR RB (RB=Ø16Ø)
2E	SP,2	3 F 8 F	RESTA 9ØDEP2 (P2=ØØDØ
30	TLD,1	F 8 Ø Ø Ø Ø B 2	CARGA EN P1
34	MVC	C 5 Ø 1	TRANS. 2 BYTE DE ØØDØ A ØØB2 (RB DE OPSR EN P1 DE PSR)
36	AP,2	3 D 8 F	SUMA 9Ø A P2 (P2=Ø16Ø
38	LPD,1	6 C 2 2	TRANS. P1 A R2 (RB DE OPSR A R2)
3A	SR	Ø 4 Ø 2	RO ⟵ R2 - RO (DIREC= RELATIVA DE OPSR EN RO)
3C	LR	Ø 8 3 Ø	TRANS. RO A R3
3E	TL,1	B 8 ØØ	PONE A CERO P1
40	TLD,2	F 9 Ø Ø Ø Ø Ø 5	CARGA ØØØ5 EN P2
44	YTX	D 6 7 3	DESEMPAQ. Y TRANSGO-DIFICA (DIR. RELATIVA DE OPSR)DAØ165 A Ø16Ø
46	MVI,2	C 7 2 Ø	FUERZA Ø IN Ø165
48	YBP,1	A E 0 / 2	CAMBIA RB POR P1 (RB=ØØØØ)
4A	AP,1	3 C Ø 5	SUMA Ø6 P1 (P1=Ø166)

PROGRAMA DE PUESTA A PUNTO - TABLA L

DIR. ROM2.	CODIGO SIMBOLICO	INSTRUCCION	COMENTARIO
17 4C	AP,2	2 D D Ø	SUMA D1 A P2 (P2=ØØD6)
4E	YTX	D 6 1 Ø	DESEMPAQ. Y TRANSCODIFICA (CP DE OPSR EN Ø166) DE ØØD6 A Ø166
5Ø	KES	A 7 4 7	HABILITA BARRAS: S0, S1, S2, (0100 0111)
52	SP,1	3 E B 1	RESTA B2 DE P1 (P1=ØØB4)
54	SP,2	3 F Ø 5	RESTA Ø6 DE P2 (P2=ØØDØ)
56	MVC	C 5 Ø 1	TRANSF.2 BYTE DE ØØDØ A ØØB4) (RB DE OPSR A P2 DE PSR)
58	AP,1	3 C 9 B	SUMA 9C A P1 (P1=Ø15Ø)
5A	YBP,1	A E Ø 2	CAMBIA RB POR P1 (RB=Ø15Ø)
5C	LPD,2	6 C 3 4	TRANSF. P2 A R4 (RB DE OPSR A R2; RB=Ø16Ø)
5E	LR	Ø 8 1 4	TRANSF. R2 A Ø158 (RB DE OPSR A Ø158)
6Ø	AR	C Ø 4 5	R2 ← R2+R3 (R2=DIR.ABSOLUTA DE OPSR)
62	LR	Ø 8 Ø 4	TRANSF. R2 EN Ø15Ø
64	TRD	6 C 1 4	TRANSF. R2 EN P2 (P2 CONTIENE DIR. ABSOLUTA DE OPSR)



PROGRAMA DE PUESTA A PUNTO -TABLA L

DIR. ROM2	CODIGO SIMBOLICO	INSTRUCCION	COMENTARIO
17 66	TL	B 8 Ø Ø	PONE A CERO P1
68	YBP,1	A E Ø 2	CAMBIA P1 POR RB (RB=ØØØØ)
6A	AP,1	3 C 1 B	SUMA 1C A P1 (P1=Ø16C)
6C	MVC	C 5 Ø 3	TRANSF. 4 BYTE (INSTRUCCION NO REALIZADA DE OPSR) EN Ø16C
6E	TLD,2	F 9 Ø Ø Ø 1 6 8	CARGA Ø168 EN P2
72	TCP	3 8 Ø 2	CAMBIA P1 POR P2 (P1=Ø168 E P2=Ø16C)
74	YTX	D 6 7 3	DESEMPAQ. Y TRANSCODIFICA (INSTRUCCION NO REALIZADA DE OPSR)
76	SP,1	3 E Ø 1	RESTA Ø2 DE P1 (P1=Ø166)
78	SP,2	3 F Ø 4	RESTA Ø5 DE P2 (P2=Ø167)
7A	MVC	C 5 Ø Ø	TRANSF.1 BYTE DE Ø167 A Ø166 (2º SEMIBYTE DE CP EN Ø166)
7C	MVI	C 7 2 Ø	FUERZA Ø a Ø167
7E	SP,2	3 F Ø 6	RESTA Ø7 DE P2 (P2=Ø16Ø)
8Ø	YBP,2	A F Ø 2	CAMBIA RB POR P2 (RB=Ø16Ø)
82	TLD,1	F 8 Ø Ø Ø Ø 1 Ø	FUERZA ØØ1Ø A P1 (P1=ØØ1Ø)

PROGRAMA DE PUESTA A PUNTO - TABLA L

DIR. ROM2	CODIGO SIMBOLICO	INSTRUCCION	COMENTARIO
17 86	YOP	& E 7 F	VISUALIZA INSTRUCCION DE OPSR QUE SE DEBE REALIZ. PARA 16 BYTE DIRECCIONADOS POR P1 E INTRODUCE DATOS DEL TECLADO PARA 8 BYTES DIRECCIONADOS POR P2 PONE CC=0 SI S0, CC=1 SI S1 CC=2 SI S6, CC=3 SI S2 O RUN
88	BDC	F Ø Ø Ø 1 7 A E	SALTA A DIREC.17AE SI CC=ØØ
8C	BDC	F 1 Ø Ø 1 7 C 4	SALTA A DIREC.17C4 SI CC=Ø1
9Ø	BDC	F 2 Ø Ø 1 7 E E	SALTA A DIREC.17EE SI CC=Ø2
94	YBP,2	A F Ø 2	CAMBIA RB POR P2 (RB=ØØØØ)
96	NIC	2 8 9 D	AND ENTRE (1101) Y SEMIBYTE EN POSIC.9 DE R.C. (CODIGO DE BARRA INTRODUCIDA) PONE CC=0 SI S2
98	FØ	6 8 Ø 4	SI CC=0, SALTA
9A	BD	7 Ø Ø Ø 1 8 Ø C	SALTO AL PRG. 'RUN '-DIREC. 18ØC

PROGRAMA DE PUESTA A PUNTO - TABLA L

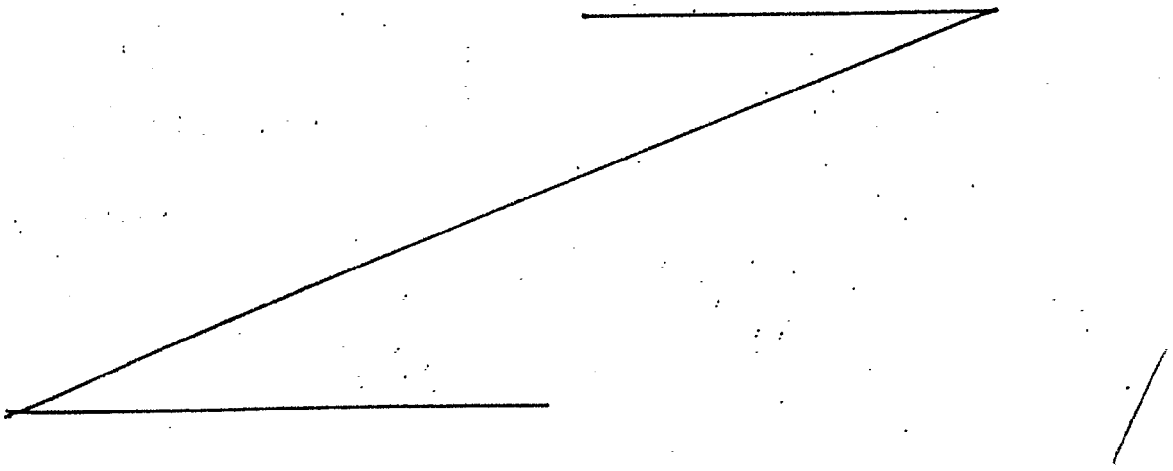
DIR. ROM2	CODIGO SIMBOLICO	INSTRUCCION	COMENTARIO
			PROGRAMA DE INSTRUCCION POR
			TARJETA MAGNETICA BARRA S2
17 9E	TLD,1	F 8 Ø Ø Ø 1 8 C	FUERZA Ø18C a P1
A2	SP,2	3 F 9 1	RESTA ?2 DE P2 (P2=ØØBE)
A4	MVC	C 5 Ø 1	SALVA I.I.EN EL REGISTRO 3Ø
A6	AP,2	3 D 8 1	SUMA 82 A P2 (P2=Ø15Ø)
A8	STIO,2	A F Ø Ø	PREP.LOS PARAMETROS DE
			TRANSFER.
			PERIFERICA - RAM1
			TRANSFIERE DE C.M.SEGUN.P2
AA	LAC	A B 8 8 Ø 1 5 Ø	LANZA POR DE DIREC.Ø15Ø RAM
			PROGRAMA DE LECTURA DE LA
			MEMORIA BARRA SO.
AE	AP,1	3 C A F	SUMA BØ A P1 (P1=ØØC7)
BØ	YBP,1	A E Ø 2	CAMBIA P1 POR RB (RB=ØØC7)
B2	NI	2 Ø Ø B	(BSD) ⟵(BSD)AND ('ØB ')
B4	YBP,1	A E Ø 2	CAMBIA P1 POR RB (RB=Ø16Ø)
B6	SP,1	3 E A F	RESTA BØ DE P1 (P1=ØØ1Ø)
B8	CBI,1	8 C 2 Ø	SALTA A S1 SI SE INTRODUCE
			UNA DIRECCION EN EL TECLADO

PROGRAMA DE PUESTA A PUNTO - TABLA L

DIR. ROM2	CODIGO SIMBOLICO	INSTRUCCION	COMENTARIO
17 BC	ARI	5 Ø 3 4	REG(365) ←—REG(365)+4
BE	LR	Ø 8 Ø 3	REG(362) ←—REG(365)
CØ	BD	7 Ø Ø Ø 1 7 3 E	SALTA A DIRECC. 173E
			PROGRAMA DE PRENOTACION DE
			STOP BARRA SI
C4	PK	5 1 Ø Ø	EMPAQU.EL REG(364)Y TRANSCOD
			TRANSFER.AL REG.(362)LA DIREC
			INTROCUCIDA POR EL TECLADO
C6	NIC	2 8 9 F	(0101)AND (POS.9 DI RC)
C8	BD	F Ø Ø Ø 1 7 3 C	SI AND=0 SALTA A 173C
CC	TLD,2	F 9 Ø Ø Ø 1 5 8	FUERZA Ø158 A P2
DØ	YBP,2	A F Ø 2	CAMBIA P2 POR RB (RB=Ø158)
D2	AR	C Ø Ø 1	DIR.ABSOLUTA DE PRENOT.EN 361
D4	CVB	3 8 Ø 1	361 DEC. ←— (361 BIN.)
D6	SP,2	3 F C 1	RESTA C2 DE P2 (P2=ØØ9E)
D8	TLD.1	F 8 Ø Ø Ø Ø C Ø	FUERZA CØ A P1
DC	YBP,1	A E Ø 2	CAMBIA RB POR P1 (RB=ØØCØ)
EØ	TLD,1	F 8 Ø Ø Ø Ø 2 C	FUERZA ØØ2C A P1
E4	MVC	C 5 Ø 1	TRANSF.EL 1º Y 2º BYTE DE
			361 A 350

PROGRAMA DE PUESTA A PUNTO - TABLA L

DIR. ROM2	CODIGO DIMBOLICO	INSTRUCCION	COMENTARIO
17 E6	OI	2 1 F 4	(BSD 351)OR (0100)
E8	AP,2	3 D C 1	SUMA C2 A P2 (P2=ø16ø)
EA	YBP,2	A F ø 2	CAMBIA P2 POR RB(RB=ø16ø) (P2=øøCø)
EC	BD	7 ø ø ø 1 7 B E	SALTO A 17BE
			PROGRAMA DE ESCRITURA EN LA RAM1 - BARRA S6
EE	AP,2	3 D 1 3	FUERZA 'øø14' A P2(P2=øø14)
Fø	TCP	3 8 ø 2	CAMBIA P1 POR P2 (P2=øø1ø)
F2	YTC	D 7 3 7	EMPAQ. Y TRANSCODIFICA DIRECCION INTRODUCIDA
F4	TLD,2	F 9 ø ø ø 1 5 ø	FUERZA ø15ø A P2 (P2=ø15ø)
FA	TRD,1	6 C ø ø	TRANSFIERE REG(362)=ID.EN EL INDICADOR P1
FC	TL,2	F 9 ø ø	FUERZA P2 A 'øøøø'
FE	YBP,2	A F ø 2	CAMBIA RB POR P2 (RB=øøøø)
18 øø	AP,2	3 D 2 3	SUMA '24'A P2 (P2=ø174)
ø2	MVC	C 5 ø 3	SUSTITUYE LA INSTRUCC. INSTRODUCIDA UNA VEZ VISUALIZADA LA INSTRUCC.



PROGRAMA DE PUESTA A PUNTO - TABLA L

DIR. ROM2	CODIGO SIMBOLICO	INSTRUCCION	COMENTARIO
18 Ø4	SP,2	3 F 1 3	RESTA '14' A P2 (P2=Ø16Ø)
Ø6	YBP,2	A F Ø 2	CAMBIA RB POR (RB=Ø16Ø)
Ø8	BD	7 Ø Ø Ø 1 7 B C	SALTA A 17BC
			PROGRAMA DE "EJECUCION PASO A PASO" TECLA RUN.
ØC	TLD,2	F 9 Ø Ø Ø 1 8 8	FUERZA P2 A Ø188'
1Ø	TLD,1	F 8 Ø Ø Ø Ø C A	FUERZA P1 A ØØCA'
14	MVC	C 5 Ø 1	REPONE E BYTE EN R.C. 359
16	AP,2	3 D Ø 1	SUMA 2 A P2 (P2=Ø18A)
18	AP,1	3 C 1 9	SUMA 1A A P1 (P1=ØØE4)
1A	MVC	C 5 Ø Ø	REPONE EL BYTE 'HABILITA BARRAS' (370)
1C	SP,2	3 F Ø 9	RESTA 'ØA' DE P2 (P2=Ø18Ø)
1E	SP,1	3 E 3 B	RESTA '3C' DE P1 (P1=ØØA8)
2Ø	MVC	C 5 Ø 7	REPONE 8 BYTES DE REG(366) EN EL REG. DE TRABAJO 352
22	YPS	3 8 Ø 9	REPONE 10 BYTE DE OPSR 301 EN EL PSR 300 Y ACTIVA PROGRAMA DE PRUEBA

En particular, mediante las dos primeras instrucciones TLD, 2 y YPB, 2 se lleva a Ø16Ø el registro RB-310 (bloque 400 de la figura 11a). A continuación se transfiere al registro 366 el contenido del registro de trabajo 352 mediante una instrucción LAX (bloque 401). Hay que observar que el registro 366 es direccionado por la LAX como cuarto registro (determinado por el último semibyte de la instrucción) a partir del registro base RB-310 de PSR-300 que contiene la dirección Ø16Ø.

Mediante las instrucciones TL y YBP,1 se pone a cero el registro base RB-310 (bloque 402). Posteriormente, mediante las instrucciones TLD,1, TLD,2 y MVC se salvan dos bytes del registro de condiciones RC-359 en las posiciones Ø188 y Ø189 (bloque 403) los cuales, como se ha dicho anteriormente, contienen las condiciones marcadas desde el exterior por el programador antes de la interrupción de DBG. Estos bytes se volverán a introducir en el RC-359 del programa de DBG en caso de que el programa interrumpido se reanude después de la ejecución del programa de DBG.

Hay que notar que la instrucción MVC transfiere a partir de la posición direccionada por la suma de los contenidos del registro P2-312 y RB-310 un campo de longitud L + 1 a partir de la posición direccionada de la suma de los contenidos de RB-310 y P1-311; L es el número que se contiene en el segundo byte de la instrucción.

A continuación, mediante las instrucciones AP1, AP2 y MVC, se transfiere el registro AB-370 (Habilita-Barras) a la posición Ø18A (bloque 404).

Mediante las instrucciones SP2, SP1 y MVC, se transfiere el registro CI-324 a la posición 0168 (bloque

405).

Mediante las instrucciones AP2, SP1, MVC se transfieren los dos bytes que indican la dirección absoluta de reposición del programa interrumpido (registro IR-327) en el registro P1-311 de PSR-300. Posteriormente, mediante las instrucciones AP2 y LPD, esta dirección es transferida por P1-311 al registro 362 (bloque 406).

Mediante las instrucciones YBP,2, SP2, TLD,1 y MVC se lleva el registro RB-310 al valor Ø16Ø y se transfiere el registro base RB-320 de OPSR-301 al registro P1-311 de PSR-300. Acto seguido, mediante las instrucciones AP2 y LPD1, el registro RB-320 es transferido al registro 364.

Mediante la instrucción SR se efectua la diferencia entre el contenido de los registros 364 y 362, y el resultado se coloca en el registro 362 (bloque 407). Esta diferencia es transferida posteriormente mediante una LR al registro 363 (bloque 408). Mediante las operaciones que se acaban de describir, en los registros 360 y 363 se registra la dirección relativa en código exadecimal de la instrucción que debe realizarse del programa interrumpido.

Mediante las instrucciones TL1 y TL2 se colocan los registros P1-311 y P2-312, respectivamente, en los valores ØØØØ y ØØØ5 que, sumados al registro RB-310 direccionan las posiciones Ø16Ø y Ø165 respectivamente.

Se efectua después una instrucción YTX la cual desalmacena y transcodifica cuatro bytes (especificados por el último semibyte de la instrucción) del registro P2-312 y los transfiere a ocho bytes (especificados por el tercer semibyre de la instrucción) direccionados por el regis-

tro P1-311, eliminando los ceros no significativos (bloque 409).

Se realiza después una instrucción MVI,2 la cual es direccionada por P2-312 y escribe un CRT"espacio" en la posición Ø165 (bloque 410).

Mediante las instrucciones YPB,1, AP1 y AP2 y YTX se pone a cero el registro RB-310 y el byte CP-313 leído por la RAM1, desempacado, transcodificado y transferido a las posiciones 0166 y 0167 (bloque 411 de la figura 11b).

Se efectua después la instrucción KES la cual prepara en la posición ØØE4 el código expresado por el segundo byte. Este código es "01000111" y cada bit va asociado a una barra correspondiente 102 (bloque 412). En particular, este código se interpreta como un código de habilitación de las barras S0, S1, S2 y S6 por la instrucción YOP para controlar que las barras accionadas por el programador hayan sido o no habilitadas.

Mediante las instrucciones SP1, SP2 y MVC se transfiere el registro RB-320 de OPSR-301 al registro P2-312 de PSR (bloque 413). Mediante las instrucciones AP1, YBPI, LPD2 se transfiere RB-320 de OPSR-301 al registro 364 (bloque 414). El contenido del registro 363 se transfiere después al registro 361 (bloque 415).

Mediante las instrucciones LR y AR se calcula en el registro 364 la dirección absoluta de la instrucción del programa interrumpido que debe realizarse (bloque 416). Hay que notar que el contenido del registro 365 era la dirección relativa del programa interrumpido. Esta dirección absoluta se transfiere después al registro 360 y al registro P2-312 mediante las instrucciones LR y TRD (bloque

416). Mediante las instrucciones TL y YBP1 se pone a cero el registro RB-310 y mediante las instrucciones AP1 y MVC se transfiere el contenido de la posición direccionada por P2-312 a la segunda mitad del registro 363. Este contenido se desalmacena y transcodifica mediante las instrucciones TLD2, TCP y YTX, por lo cual en el registro 363 queda registrada la instrucción que debía ser realizada por el programa interrumpido en código desalmacenado (bloque 417). Mediante las instrucciones SP1, SP2 y MVC se desplaza en una posición a la izquierda el segundo semibyte del byte CP, registrado en la posición Ø167, por lo cual dicho semibyte se encuentra ahora registrado en ambas posiciones Ø166 y Ø167. Posteriormente, mediante la instrucción MVI se escribe en CRT de "espacio" en la posición Ø167 (bloque 418).

En conclusión, las operaciones desarrolladas del bloque 406 al bloque 418 han predispuesto en los registros 362 y 363 en código desalmacenado la dirección correspondiente de la instrucción que debe realizar el programa interrumpido, un CRT espacio, el segundo semibyte del byte CP (que comprende entre otros el código de condición), un CRT "espacio" y el código exadecimal de la instrucción contenida en dicha dirección. Con referencia a las figuras 12a, 12f, se describe un ejemplo de preparación de los registros 362 y 363.

Supongamos que el programa haya sido interrumpido en la dirección absoluta 2423 expresada en exadecimal almacenado; el cual se transfiere al registro 362 (bloque 406) y se representa en el mismo en forma almacenada como se muestra en la figura 12, donde el último semibyte (carácter c) representa el signo positivo. Supongamos además que el

contenido de RB-320 de OPSR-301 sea "12ØØ"; dicho contenido se transfiere al registro 364 (bloque 407) y se representa en la forma almacenada que se muestra en la figura 12a. A continuación se efectua la diferencia entre los dos registros 364 y 362 y el resultado se momoriza en el registro 362 como se muestra en la figura 12b (bloque 408). A continuación se desalmacena el contenido del registro 362 (bloque 409) ocupando por lo tanto los cinco primeros bytes del registro 362 (figura 12c). Acto seguido se inserta un CRT de "espacio" en la posición 165 (bloque 410) y el byte C.P. (323) en forma desalmacenada en las posiciones 0166 y 0167. Se supone que dicho byte tiene el valor "4A" representado en la figura 12e. Posteriormente se lee la instrucción que debía ser realizada por el programa interrumpido y se transfiere en forma almacenada a la segunda mitad del registro 363. Supongamos que la instrucción tenga el código "F9000005" es decir, que sea una instrucción TLD,2 mediante la cual se carga el número "5" en el indicador P2-312 (figura 12b). Posteriormente esta instrucción vuelve a escribirse en el registro 363 en forma desalmacenada (figura 12 c y bloque 417). Acto seguido el byte CP se desplaza a la izquierda en una posición y se escribe en la posición Ø167 un CRT "espacio" (bloque 418).

En definitiva, la configuración de los dos registros 362 y 363 es la que se muestra en la figura 12d.

La figura 12f muestra con detalle la secuencia de las configuraciones de las posiciones Ø166 y Ø167 en el paso de la configuración de la figura 12c a la de la figura 12d.

La configuración predispuesta en los registros 363 y 364 se visualizará después como se verá mejor más

adelante.

Pasamos ahora a la descripción del programa de DBG (figura 11b).

Mediante las instrucciones SP2, YBP,2 y TLD,1 se lleva a Ø16Ø el registro RB310 y se lleva a ØØ1Ø el registro P1-31 (bloque 419 de la figura 11b). Después de esto se efectua una instrucción YOP (bloques 420 y 421 de la figura 11c) la cual envía al visualizador 103 los caracteres registrados en los registros 362 y 363 representados y transfiere los datos instroducidos por el operador en el registro 364 y coloca el código de condición según la barra 102 accionada por el operador.

La instrucción YOP puede considerarse dividida substancialmente en tres bloques funcionales: un primer bloque procede a enviar al visualizador 6 los datos preparados por las instrucciones anteriores en los registros 362 y 363; un segundo bloque procede a introducir en el registro 364 lo indicado por el indicador P1-311 los datos introducidos en el teclado; y un tercer bloque que procede a preparar el código de condición contenido en el byte de condiciones de programa 313 el noveno semibyte del registro de condiciones 359 según la barra que ha sido accionada.

El primer bloque funcional es un microprograma que utiliza la suma del indicador P2-312 y del registro base RB-310 para direccionar el primer byte del registro 362 para iniciar la lectura de la RAM1 de los datos contenidos en los diez y seis bytes sucesivos al direccionado.

Este microprograma, mediante microinstrucciones apropiadas, transfiere directamente los datos (a través del nudo NC y la lógica de canal 45) a las unidades peri-

féricas seleccionadas.

En este caso, la lógica de canal 45 a través de la instrucción YOP ha seleccionado la visualización 6 (figura 1b) por lo cual se visualizan los datos contenidos en ambos registros 362 y 363.

El modo con el que la lógica de canal 45 selecciona la visualización 6, ha sido ya apuntado anteriormente y se expone además en la mencionada solicitud de patente número 916.415. Además, esta selección puede ser efectuada de cualquier modo conocido; por lo tanto no se describe con detalle, incluso porque no constituye el objeto de la presente invención.

El segundo bloque funcional de la YOP es un microprograma que utiliza la suma del indicador P1-311 y RB-310 para direccionar la posición inicial del registro 364 y depositar en los ocho bytes sucesivos los datos introducidos en el teclado. Este microprograma utiliza unas microinstrucciones apropiadas (que no se incluyen en las tablas) que permiten cambiar los datos con la unidad periférica seleccionada. Estos datos, a través de la puerta 70 de la figura 2b, se introducen en el nudo ND y a partir de éste (mediante el nudo NB), se transfieren al registro B14 (figura 8). La introducción de los datos del teclado 5 al registro B14 es controlada por una microinstrucción que no se incluye en las tablas, la cual genera las señales de mando: CZ03, CA02, CT06, CT07 y las señales de mando de selección C032-C039, las cuales actuan en las puertas 70, 71, 72, 73 respectivamente, y en los circuitos de selección que se dan en la figura 7, por lo que los caracteres son transferidos desde el teclado 5 al registro B14. Se precisa en particular que esta transferencia tie-

ne lugar en paralelo para bit y en serie para carácter, es decir, que los caracteres se transfieren uno por uno.

Se describe a continuación con detalle el tercer bloque funcional con referencia al microprograma de la tabla M y al diagrama de circulación de la figura 13.

TABLA M

DIREC. SIMB.	INSTRUCC.	CODIGO	I OPERANDO	II OPERANDO
IGOTEO	B E 9 7	SDIB	B14	
	2 7 A 2	SADO	D07	IGOTEH
	3 6 A 2	SADI	D06	IGOTEH
	3 5 A 2	SADI	D05	IGOTEH
	8 4 A 2	SADO	D04	IGOTEH
	2 3 A 2	SADO	D03	IGOTEH
	B E 9 7	SDIB	B14	
	8 9 8 F	CRTA	A09	C0F
	6 9 E 3	ORE	A09	B14
	3 1 F 4	SADI	D01	IGOTUX
	8 9 0 7	CRTA	A09	C07
	6 9 E 4	ANDB	A09	B14
	8 9 0 0	CRTA	A09	C00
	9 2 0 1	CRTB	B02	C01
IGOTU4	B 8 6 6	SEDI	D00	
	6 9 E 2	SOT	A09	B14
	3 1 D 2	SADI	D01	IGOTU3
	A 9 4 9	ICA	A09	
	0 8 D 8	SHSB	B02	
	0 8 C C	SAI	IGOTU4	
IGOTU3	C 2 E 4	MAD	A02	CE4

TABLA M

DIREC. SIMB.	µINSTRUCC.	CODIGO	I OPERANDO	II OPERANDO
	6 2 2 0	AND	A02	B02
	0 8 D 4	SADI	D01	IGOTAJ
IGOTUY	C 2 C B	MAD	A02	CCB
	B 2 2 7	AZAP	A02	
	5 9 2 C	TAB	A09	B02
	B 2 1 6	ROTB	B02	
	6 2 2 E	ORA	A02	B02
	D 2 C B	AMD	A02	CCB
	9 2 0 6	CRTB	B02	C06
	6 9 2 3	ORE	A09	B02
	3 1 E E	SADI	D01	IGOTU6
	9 2 0 2	CRTB	B02	C02
	B 8 6 6	SEDI	D00	
	6 9 2 2	SOT	A09	B02
	3 0 E C	SADI	D00	IGOTU7
IGOTU8	C 2 B 6	MAD	A02	CB6
	9 2 F C	CRTB	B02	CFC
	6 2 2 4	ANDB	A02	B02
	6 9 2 E	ORA	A09	B02
	D 9 B 6	AMD	A09	CB6
	0 2 0 0	SAI	IALFA	

TABLA M

DIRC. SIMB	µINSTRUCC	CODIGO	I OPERANDO	II OPERANDO
IGOTU7	8 9 0 3	CRTA	A09	C03
	0 8 E 4	SAI	IGOTU8	
IGOTU6	8 9 0 2	CRTA	A09	C02
	0 8 E 4	SAI	IGOTU8	
IGOTUX	0 9 0 9	CRTA	A09	C09
	0 8 D 5	SAI	IGOTUY	

Como se ha dicho anteriormente, en el registro B14 ha sido memorizado por obra del segundo bloque funcional el último carácter procedente del teclado 5 (bloque 600 de la figura 13).

A través de la microinstrucción SDIB se intercambia el CRT contenido en B14 con el contenido en los desviadores 40.

A continuación se prueban los bits de 03 a 07 del último CRT introducido mediante las instrucciones SADO y SADI que actuan sobre los desviadores D03 a D07 (alternativa lógica 601).

Debe hacerse notar que si la última introducción del teclado es una barra del grupo 102, o la tecla RUN del pupitre 7 en el registro B14, el CRT registrado toma la siguiente configuración: 10011XXX. Los bits XXX indican el binario el número de 0 a 6 asociado a las barras 102, mientras que el número 7 va asociado a la tecla RUN.

Sí, por el contrario, el último carácter introducido es un código alfa-numérico, al menos uno de los bits de

b03 a b07 toma una configuración distinta de 10011, por lo que se realiza un salto a la dirección IGOTEH. El microprograma que comienza en IGOTEH es el que efectua la introducción de los caracteres del registro 364 de la RAM1, es decir, realiza el segundo bloque funcional apuntado anteriormente.

Si los bits b03-b07 tienen la configuración 10011, se reconoce la tecla RUN, a la que corresponde 10011111 mediante las microinstrucciones CRTA, ORE y SADI, que comprueban con una máscara su el contenido del registro B14 es igual a 10011111 (alternativa lógica 602). Si este contenido es distinto, el CRT es ciertamente una barra del grupo 102.

Mediante las microinstrucciones CRTA y ANDB se ponen a cero los bits 03 a 07 del registro B14, mientras que los bits 00-02 indicados con XXX que dan en binario el número asociado a la barra 102 introducida no quedan alterados (bloque 603). A continuación, mediante las microinstrucciones CRTA y CRTB y SEDI se predisponen los registros operativos A09 = 'ØØ' y B02 = 'Ø1' se coloca en 1 el desviador D00. Mediante las microinstrucciones SOT y SADI se controla si el contenido de B14 es igual al contenido de A09 (alternativa lógica 606). Si los dos registros tienen distinto contenido, mediante las microinstrucciones ICA y SHSB se incrementa en una unidad el registro A09 y se desplaza en una posición a la izquierda el contenido del registro B02 (bloque 607) a continuación, con un salto incondicionado SAI, se vuelve al bloque 604 y se repite el ciclo de los bloques 604, 606, 607. Este ciclo se repite tantas veces cuanto sea el número de orden de la barra 102 introducida.

Si, por ejemplo, se introduce la barra S6,

el ciclo se repite siete veces, y así sucesivamente, debe observarse que cuando el contenido del registro B14 es igual al del registro A09, el bit "1", registrado inicialmente en la primera posición del registro B02 ha sido desplazado a la izquierda tantas posiciones cuantos son los ciclos efectuados. En el ejemplo citado el bit "1" se encuentra registrado en la séptima posición del registro B02, es decir, que el byte registrado en B02 es 0100 0000.

Cuando B14 = A09, mediante las microinstrucciones MAD, V y SADI se controla si la barra introducida está o no habilitada por el programa (alternativa lógica 608). Como se ha dicho anteriormente, el programa de DBG había preparado la posición ØØE4 de la RAM1 (registro 370 de la figura 9) con un byte de habilitación de barras en el que los bits a nivel "1" identifican la barra habilitada.

En particular se habían habilitado las barras S0, S1, S2, S6 (bloque 412 de la figura 11b) por lo que en la posición ØØE4 de la RAM1 se encuentra registrado el byte 01000111.

Mediante la microinstrucción AND se controla por lo tanto si la posición del registro 370 correspondiente a la barra identificada anteriormente se encuentra registrado un bit "1", es decir, se controla si la barra ha sido habilitada. En el ejemplo citado, tanto en la sexta posición del registro B02 como en la sexta posición del registro 370 se encuentra registrado un bit "1".

Por el contrario, si la barra no hubiese sido habilitada se habría efectuado un salto a la dirección IGOTAJ. En esta dirección se encuentra registrada la microinstrucción inicial de un microprograma que controla las ope-

raciones consiguientes al accionamiento de una barra no habilitada, por ejemplo, el encendido de una lámpara del pupitre 7 y la espera de una nueva introducción. Naturalmente, este microprograma no se describe porque no constituye el objeto de la presente invención.

Si la barra accionada por el operador había sido habilitada por el programa de DBG se efectua la microinstrucción MAD que transfiere al registro operativo A02 el 5º byte del registro de condiciones RC-359, que se encuentra registrado en la posición ØØCB con barra. Como se ha dicho, en dicho byte se registran a partir de la derecha los semibytes de la octava y novena posición de RC-359 y la novena posición contiene el código de la barra introducida en última posición y anteriormente registrada también en la posición Ø188 (registro 365), como se describe con referencia al bloque 403 de la figura 11a.

Acto seguido se realiza una microinstrucción AZAP que pone en cero los bits de la posición 9 de RC.

Mediante las microinstrucciones TAB y ROTB, se transfiere al registro B02 el código de la barra accionada presente en A09 después de haber intercambiado los semibytes.

En el ejemplo considerado (accionamiento de la barra S6), en el registro A09 estaba presente el byte 00000110 por lo cual en el registro B02 se encuentra ahora registrado el byte 0110 0000.

A través de la microinstrucción ORA se efectua el OR entre el registro A02 y del registro B02, y el resultado se transfiere al registro A02. De este modo, en el primer semibyte del registro A02 se encuentra registrado el código binario de la barra accionada, y en el segundo -

semibyte el contenido inalterado de la posición 8 del RC-359.

Finalmente, mediante la microinstrucción AMD, se repone en la posición 9 de RC-359 el código binario de la barra accionada, mientras que la posición ocho se vuelve a escribir sin cambios. Estas operaciones se indican brevemente en el bloque 609 de la figura 13.

En caso de que la tecla accionada no sea una barra, sino mas bien la techa RUN, se realizaba como se ha dicho anteriormente un salto a la dirección IGOTUX en la cual se escribe mediante una CRTA (bloque 610) el carácter 0000 1001 en el registro A09. Este carácter corresponde a la tecla RUN, y mediante un salto incondicionado SAI a la dirección IGOTUY (bloque 609) se efectua el registro del carácter '1001' en la posición nueva del RC-359 como se ha descrito ya para las barras. En cualquier caso, después de la microinstrucción AMD del bloque 609, se controla mediante las microinstrucciones CRTB y ORE si el contenido del registro A09 es igual al carácter '06' que identifica la barra S6 (alternativa lógica 611).

Si la comparación tiene resultado negativo, se controla mediante las microinstrucciones CRTB, SEDI, SOT, si el contenido del registro A09 es mayor o igual a dos, es decir, si han sido accionadas las barras S0, S1, o bien la barra habilitada restante S2 (alternativa lógica 612).

Supongamos que la barra accionada sea S0 ó S1, por lo cual el registro A09 contiene el byte 0000 0000, o bien 0000 0001. Mediante la microinstrucción MAD, CRTB y ANDB, se extrae el byte CP-313 de la dirección ØØB6 y se ponen a cero los dos bits menos significativos correspondientes al código de condición. El byte C.P. así modificado se

coloca después en el registro B02.

Acto seguido, y mediante las microinstrucciones ORA y AND, se vuelve a escribir en la posición ØØB6 el byte CP-313 cuyos dos bits menos significativos contienen 00 ó 01, según que la barra accionada sea S0 ó S1. Estas operaciones se indican sintéticamente en el bloque 613.

Si por el contrario, se había accionado la barra S6 (alternativa lógica 611) o la barra S2 (alternativa lógica 612) se efectuaba un salto a la dirección IGOTU6, o a la dirección IGOTU7, respectivamente. En ambos casos, se efectuaba la microinstrucción CRTA (bloques 614 y 615) la cual forzaba al registro A09 el carácter 0000 0010 ó 0000 00011, respectivamente. A continuación, se efectuaba en ambos casos un salto incondicionado a la dirección IGOTU8 a partir de la cual se efectuan las microinstrucciones anteriormente descritas con referencia al bloque 613.

Después de esto, se salta a la dirección IALFA para realizar a través del microprograma intérprete el reconocimiento de la instrucción sucesiva como se ha expuesto anteriormente.

En conclusión, se ha visto como el tercer funcional de la instrucción YOP prepara la posición nueva de RC-359 con el código de la barra accionada, como coloca el código de condición en 00 si ha sido accionada la barra S0, 01 para la barra S1, 02 para la barra S6 y 03 para la barra S2 y la tecla RUN.

Las operaciones de los tres bloques funcionales de la instrucción YOP que acaban de describirse se indican simbólicamente en los bloques 420, 421, y 422 de la figura 11c.

Pasamos acto seguido a la descripción del programa de DNG con referencia a la tabla L de la figura 11c. Después de la ejecución de la instrucción YOP se analiza el código de condición mediante tres instrucciones BDC de salto condicionado (alternativa lógica 422, 423, 424).

Apuntamos ahora brevemente las operaciones asociadas a las barras S0, S1 y S6, que se explicarán más adelante con mayor detalle.

Si ha sido accionada la barra S0 se tiene CC =0 por lo cual se efectua un salto a la dirección 17AE en la que se registra el programa de lectura de la memoria que visualiza la instrucción sucesiva a la ya visualizada por la YOP, o bien en caso de que el operador haya introducido un número en el teclado, se visualiza la instrucción registrada en la dirección expresada por el número marcado o introducido aumentado con el contenido del RB-310.

Si ha sido marcado un número en el teclado y accionada la barra S1 (CC = 1) se efectua un salto a la dirección 17C4 en la que está registrado el programa de prenotación de la parada de las elaboraciones del programa que hay que corregir en la dirección de RAM1 expresada por el número marcado aumentado con el contenido de RB-320.

Si finalmente ha sido marcado un número de base exadecinal (D + F) de ocho CRT, y ha sido accionada la barra S6 (CC = 3) se efectua un salto a la dirección 17EE en la que se registra un programa que registra en la dirección de memoria visualizada en el visualizador 7 la instrucción o los datos introducidos en el teclado y que visualiza el contenido del campo registrado en la dirección visualizada más cuatro, permitiendo así escribir en secuencia en la RAM1.

De esta forma se pueden modificar cualquier instrucción del programa que se quiere corregir.

Consideremos ahora el caso de que haya sido accionada la barra S2 o la tecla RUN.

A través de una instrucción YBP2 se pone a cero el registro base RB-310 y después se compara mediante una instrucción NIC el contenido de la posición 9 del registro de condiciones 359 con la constante "1101". Como se ha dicho, en la posición 9 del registro de condiciones se había dispuesto previamente el código binario de la barra accionada. En particular, el código asociado a la tecla RUN es "1001". La instrucción NIC pone a cero el código de condición si el AND es igual a cero, y sin embargo lo pone en "1" si el AND es distinto de cero. Si en la posición 9 del registro de condiciones 359 se ha registrado 1001 (tecla RUN) el AND es distinto de cero por lo que CC = 1; sí, por el contrario, está registrado 0010 (barra S2) el AND es cero y CC = 0 (bloque 426 de la figura 11c).

Mediante la instrucción FO se controla (alternativa lógica 427) si el CC = 0. Si el CC = 0, es decir, si ha sido accionada la barra S2, se salta la instrucción siguiente y se procede con la instrucción TLDI, que corresponde al programa de "introducción de programa de DBG por CM y ejecución inmediata".

Si por el contrario CC = 1, mediante la instrucción BD, se salta a la dirección 18ØC en la que se encuentra registrado un programa que reanuda la ejecución del programa interrumpido, es decir, transfiere el registro 301-OPSR al registro 300-PSR.

En caso de accionamiento de la barra S2, se

realiza, como se ha indicado la instrucción TLD,1 y por consiguiente se escribe en el registro P1-311 la dirección Ø18C; y mediante la instrucción SP2 se fuerza en el registro P2-312 la dirección ØØBE. Acto seguido, mediante la instrucción MVC se transfieren dos bytes a partir de ØØBE a la dirección Ø18C. De esta forma se salva en la segunda mitad del registro 367 (figura 9) la dirección del programa de interrupción registrado en el registro OPSR-302(bloque 428 de la figura 11c).

Acto seguido se realiza la instrucción AP2 que fuerza la dirección Ø15Ø al registro P2-312. Mediante la instrucción sucesiva STIO se transfiere el contenido de la tarjeta magnética 9 (figura 1b) a los registros 360-367.

La instrucción STIO se puede considerar dividida en dos bloques funcionales. El primer bloque prepara en el registro 360 ocho CRT que especifican la unidad periférica de selección, el tipo de transferencia (lectura o registro), la dirección de RAM1 en la que deben introducirse o leerse los datos y la longitud del campo que interviene en la transferencia (máximo de 64 bytes). Estos caracteres son utilizados por la lógica de canal 45 para controlar la transferencia (bloque 426).

El segundo bloque funcional procede a la transferencia y al control de la exactitud de los datos recibidos de manera conocida (bloque 430) utilizando el indicador P2 como direccionador de la RAM1.

A continuación se efectua la instrucción LAC que lanza el programa de C.M. regisatrado de la forma que se describirá a continuación con referencia a la tabla N, la cual contiene el microprograma asociado a la fase BETA de la instrucción LAC.

La primera microinstrucción de dicho microprograma es una MAD que transfiere al registro A14 del byte CP-313 registrado en la dirección ØØB6. A continuación mediante las microinstrucciones TADI y SADI se sondea el bit 03 de CP-313, el cual, como se ha visto anteriormente (Figura 9a) indica si la instrucción debe leerse en RAM1 ó en ROM2. Este control se hace necesario porque la instrucción LAC tiene cuatro bytes de longitud (Tabla K) por lo cual se lee en dos fases sucesivas. Durante la primera fase, y como se ha explicado en el capítulo correspondiente al micróprograma intérprete, se transfieren a los registros B14 y B15 los dos primeros bytes, es decir, 'AB' y '88', respectivamente, y durante la segunda fase se leen el tercero y cuarto byte, es decir, 'Ø1' y '5Ø'. En el caso de programa de DBG la instrucción LAC se encuentra registrada en la ROM2 por lo que se hace necesario cargar el registro L02 con la dirección de programa registrado en el registro L07 (figura 8). Esto se efectua a través de las microinstrucciones SLL y SAB.

Finalmente, a través de dos microinstrucciones ROMA y una microinstrucción TAB se transfieren al registro L11 (figura 8) el tercer y cuarto byte de la instrucción, a saber, 'Ø15Ø '. En el caso de que la instrucción LAC perteneciere a un programa que se encuentra en la RAM1 se habría efectuado un salto a la dirección IPSRO por lo cual se habrían realizado las microinstrucciones MAIP y MBI direccionadas por el registro L07. En todo caso, en el registro L11 se encuentra registrada la dirección 'Ø15Ø '. Acto seguido se efectua la microinstrucción SLL con la que se transfiere la dirección Ø15Ø' al direccionador de programa L07. A continuación, mediante las microinstrucciones TBA y AMD,

se transfiere al registro CP-313 el byte contenido en el registro B14, a saber, el byte '88' que en código exadecimal-binario toma la configuración '1000010000'. Después se efectua un salto incondicionado a la dirección de ROM2 IALFA en la que se encuentra registrada la microinstrucción inicial del microprograma intérprete (figura 10a).

Las operaciones que se acaban de describir se indican simbólicamente en el bloque 431 de la figura 11c.

TABLA N

DIRC.SIM.	μ INSTRC.	CODIGO	I OPERANDO	II OPERANDO
IPSRI	C E B 6	MAD	A14	C'B6'
	B E E 7	TADI	A14	
	3 3 C 6	SADI	D03	IPSRO
	4 7 2 F	SLL	L07	L02
	5 2 2 F	SAB	A02	B02
	7 B 0 0	ROMA	A11	
	7 A 0 0	ROMA	A10	
	5 A B C	TAB	A10	B11
	1 5 C 8	SAI	IPSR1	
IPSRO	E 7 B D	MAIP	M07	A11
	E 7 B 7	MBI	M07	B11
IPSR1	4 7 B F	SLL	L07	L11
	1 5 C 9	TBA	A14	B15
	D E B 6	AMD	A14	CB6
	0 2 0 0	SAI	IALFA	

En conclusión, al término de la instrucción LAC se tiene que en el registro CP-313 se encuentra registrado el

byte 100010000 y que en el direccionador de programa está registrada la dirección 'Ø15Ø'.

Esto significa que cuando el intérprete llegue a la alternativa lógica 202 (figura 10a) no efectuará el salto a IINTE (bloque 250), dado que el bit 05 del byte CP (1000010000) se encuentra a nivel '0'. Cuando posteriormente el microprograma intérprete llegue a la alternativa lógica 204 encontrará el bit 03 de CP en el nivel 1 por lo que realizará una lectura de la RAM1 (bloque 205) en la dirección indicada por el registro L07, es decir, 'Ø15Ø'.

En esta dirección ha sido registrada por medio de la instrucción STIO (bloque 431) la primera instrucción del programa de tarjeta magnética el cual, como se ha dicho, se encuentra registrado en los registros 360 a 367. Esos últimos se encuentran asignados a las posiciones de la RAM1 comprendidas entre Ø15Ø y Ø18F (figura 9).

Acto seguido se realiza el programa de tarjeta magnética como cualquier otro programa que se encuentre en RAM1.

Hay que observar que forzando simplemente el byte '88' en el registro CP-313 ha sido posible cambiar el soporte del programa durante la ejecución. En efecto, se ha pasado de la lectura de ROM2, en la que se encuentran registrados los programas de DBG, a la lectura de RAM1 en la que se encuentra registrado el programa registrado en la tarjeta magnética 9.

El programa de C.M. puede ser cualquier programa de DBG (de longitud máxima = a 64 bytes) distintos de los presentes en la ROM2. El programador tendrá por lo tanto a su disposición un grupo de C.M. en los que están re-

gistrados otros tantos programas de DBG los cuales pueden comprender los programas proporcionados junto al sistema según la invención o bien programas preparados por él mismo para resolver problemas particulares relacionados con el tipo de programas utilizados por él. Por ejemplo, los programas de puesta a punto para programas orientados a problemas contables serán distintos de los programas de puesta a punto para programas orientados a problemas científicos.

Debe precisarse que los programas registrados en C.M. deben ser preparados de manera que se proporcione al término de su ejecución los parámetros del programa que deberá ser activado a continuación.

En particular pueden ocurrir tres casos:

a) el programa de reanudación por DBG es el programa anteriormente interrumpido (programa de prueba);
b) el programa de reanudación es otro programa de DBG;
c) el programa de reanudación es definido por el programa C.M., o bien el programa de C.M. cierra el trabajo de puesta a punto.

Los casos a) y b), el programa de C.M. no puede tener más de 48 bytes de longitud de manera que no ocupe los registros 366 y 367 los cuales, como se ha dicho, contienen parámetros que junto con los registros en el registro OPSR-301 son necesarios para la reanudación del programa interrumpido. Este último puede ser el programa que se prueba (caso a) o el programa de DBG (caso b)

En el caso c), por el contrario, el programa de C.M. puede tener hasta 64 bytes de longitud ya que los datos registrados en los registros 366 y 367 no serán utilizados para definir el programa sucesivo.

El caso c) se presenta, por ejemplo, cuando el programador necesita utilizar un programa de DBG que tenga más de 64 bytes de longitud.

En este caso, el programador debe preparar en una tarjeta magnética un programa de más de 64 bytes de longitud el cual realiza las funciones siguientes:

1) Transfiere una zona de memoria RAM1 definida por el programador en un soporte externo, por ejemplo, una unidad de disco;

2) Introducir un programa registrado en un soporte externo (tarjetas magnéticas o cinta magnética) en la superficie anteriormente liberada;

3) Cargar el byte de condición de programa 313 con el bit 05 de CP a nivel 1 (lectura RAM) y el direccionador de programa L07 con la dirección de la primera instrucción del programa cargado. Esta última operación ha sido descrita con detalle con referencia a la tabla N, y como se ha visto, es realizada por una única instrucción LAC.

Desde este punto en adelante, el programa cargado realiza sus funciones diagnósticas en el programa registrado en la parte restante de la RAM1.

De esta forma se pueden realizar programas de DBG de una cierta complejidad, y por consiguiente de una cierta longitud, sin alterar el programa que se quiere probar contenido en la RAM1.

En efecto, supongamos que el programa que se quiere probar y sus datos ocupen totalmente la zona libre de la RAM1, por lo que no existe memoria disponible para cargar un programa de DBG incluso de dimensiones mínimas. En este caso no sería posible cargar el programa de DBG destruir al menos parcialmente el programa que se prueba, y además no sería

posible hacer funcionar alternativamente los dos programas. Por consiguiente, como se ha dicho ya en la introducción, el programador debería probar el programa en un elaborador de capacidad superior de forma que contuviese simultáneamente el programa de prueba y los programas de DBG.

Mediante el sistema según la invención, por el contrario, llevando simplemente la llave 100 a la posición DBG, introduciendo la tarjeta magnética 9 en el lector de tarjeta magnética 9' y accionando la barra S2, el calculador se coloca en prueba de puesta al día y el programa contenido en la tarjeta magnética se carga en los registros 360-367 ó 360-365.

Esta operación no altera la zona de libre de mekoria con siguiente el programa de prueba. Desde este punto en adelante, como se ha visto ya, se realiza el programa introducido con tarjeta magnética.

En particular, supongamos que el programa de tarjeta magnética transfiere a un soporte externo una superficie de memoria que contiene una parte del programa que se quiere probar, por ejemplo, la segunda mitad. Supongamos además ue este programa introduce en dicha zona un programa de DBG, el cual es habilitado posteriormente para elaborar la primera mitad del programa de prueba que se encuentra en RAM1. Actuando después en la llave 100 se pueden obtener el funcionamiento alternativo de los dos PGM. El programa de DBG realiza pues las elaboraciones de diagnóstico y proporciona los resultados del control efectuado imprimiendo, por ejemplo, los errores cometidos en la preparación del programa de prueba. Este programa ha sidò además reparado de manera que reconazca si en la RAM1 no existe mas instrucciones del pro-

grama de prueba que hay que controlar.

En este punto el programa de DBG procede a transferir a un soporte externo la parte controlada y corregida del programa de prueba y a cargar en su lugar la segunda mitad del programa de prueba anteriormente transferido al soporte externo.

Acto seguido se controla la segunda mitad del programa de prueba y se imprimen de modo análogo a lo anteriormente dicho los resultados del control efectuado.

Posteriormente, el programador puede aportar las correcciones a las instrucciones basándose en el control efectuado por el programa de DBG; y volver a cargar el programa corregido en la RAM1. A continuación el programador mando la ejecución del programa así corregido. De todo lo dicho se deduce que el programador, utilizando el sistema de DBG según la invención, puede efectuar incluso los controles más complejos sin alterar el programa de prueba.

Como se ha dicho, el sistema de DBG según la invención permite por otra parte solicitar programas de DBG ya registrados en la ROM2 simplemente accionando la llave 100 y una de las barras 102 (S0, S1, S6).

En particular si la introducción por teclado finaliza con el accionamiento de la barra S0 (alternativa lógica 423 Figura 11c) se realiza un salto a la dirección 17AE. En esta dirección se registra una instrucción AP,1 la cual fuerza la dirección ØØC7 al registro P1-311; acto seguido se efectua una instrucción YBP,1 la cual fuerza la dirección ØØC7 al registro RB-310, Mediante la instrucción NI se pone a cero el bit 02 del byte BSD-351 (bloque 435 de la figura 11d). Acto seguido, mediante las instrucciones YBP,1 y SP,1

se lleva el registro base RB-310 a Ø16Ø y el registro 01 a ØØ1Ø (bloque 436). Posteriormente, y a través de la instrucción CBI,1 se controla si han sido introducidos datos en el registro 364 (alternativa lógica 437). Si han sido introducidos datos en el teclado se salta a la dirección 17C4, de lo contrario se efectua la instrucción ARI la cual suma la constante 4 al contenido del registro 365 (bloque 438). Como se ha visto, en este registro se contiene la dirección que corresponde al programa interrumpido (bloque 408 de la figura 11a). A continuación esta dirección así modificada se transfiere al registro 362 mediante la instrucción LR, y se realiza un salto incondicoonado a la dirección 173E (bloque 409 de la figura 11a). Como se ha visto ya, desde el bloque 409 en adelante, comienza la visualización de la instrucción contenida en la dirección cargada en el registro 362.

En este caso se visualizan los dos bytes sucesivos a la instrucción visualizada anteriormente. Por consiguiente, accionando la llave 100 y la barra S0 pueden visualizarse en el visualizador 7 y una a una, las instrucciones del programa que debe corregirse.

Si, por el contrario, el accionamiento de la barra S0 ha sido precedido por la instroducción de una dirección (alternativa lógica 437) o bien una introducción por teclado ha sido concluida con el accionamiento de la barra S1 (alternativa lógica 424 de la figura 11c) se salta a la dirección 17C4. Se almacena y transfiere al registro 362 el contenido del registro 364, el cual contiene la dirección marcada en el teclado (bloque 421 de la figura 11c). Estas operaciones son indicadas por el bloque 439 de la figura 11e.

Acto seguido, mediante las instrucciones NIC

y BD se controla si la barra accionada es S0 ó S1; esto se hace mediante un AND entre la posición 9 del registro de condiciones 359 anteriormente preparados según la barra accionada (figura 13) y la constante 1101 (bloque 440). Su el AND es cero se realiza un salto a la dirección 173C. Por lo tanto se realizan las operaciones descritas a partir del bloque 408 de la figura 11a; es decir, se visualiza la instrucción cuya dirección ha sido introducida por el teclado 5 y que se encuentra ahora registrada en el registro 362 (bloque 439 de la figura 11e). En conclusión, si se introduce en el teclado 5 una dirección y se acciona la barra S0, se visualiza la instrucción contenida en la dirección introducida.

Si, por el contrario, ha sido accionada la barra S1 mediante las instrucciones TLD,2 e YBP, el registro RB-310 se lleva a Ø158. A continuación se realiza la suma (AR) de los contenidos de los registros 361 y 362 y el resultado se registra en el registro 361 (bloque 442). El registro 361, como se ha dicho con referencia al bloque 415 de la figura 11b, contiene el registro base RB-320 del programa interrumpido, mientras que el registro 362 (bloque 439) contiene la dirección correspondiente introducida en el teclado 5. Por lo tanto, mediante las operaciones descritas por el bloque 442 se prepara en el registro 361 la dirección absoluta del programa interrumpido seleccionado por el programador mediante el teclado 5.

Mediante una instrucción CVB, se decodifica en decimal la dirección absoluta expresada en binario, y se transfiere al primero y segundo byte del registro 361 (bloque 443). Mediante las instrucciones SP,2; TLD,1; YBP,1; TLD,1; y MVC esta dirección se transfiere al registro IS-350 (bloque 444). A través de las instrucciones OI AP2; YBP,1; DB; se co-

loca en uno el bit 02 del byte BSD-351, posteriormente se fuerza la dirección Ø16Ø al registro RB-310 y más tarde se efectua un salto incondicionado a la dirección 17DE (bloque 438de la figura 11d). Desde este punto en adelante se repiten las operacuones descritas con referencia a los bloques 438' (figura 11d), 409 y siguientes (figuras 11a y 11b y c), por lo cual permanece en el visualizador 6 la instrucción anteriormente visualizada. Se ha visto por consiguiente que si el programador introduce una dirección relativa en el teclado 5 y acciona la barra S1, se escribe automáticamente en el registro IS-350 la dirección absoluta que corresponde a la introducidá y se coloca en el binario "1" el bit 02 del BSD-351. Como se ha explicado anteriormente, estos datos son utilizados por el microprograma intérprete para controlar al comienzo de la fase ALFA. De cada instrucción si la dirección de la instrucción en curso es igual a la prenotada (alternativa lógica 258 y 262 de la figura 10 b).

Después de haber prenotado la dirección de STOP, el programador devuelve la llave 100 a la posición NORMAL y acciona la tecla RUN. Reanuda pues la elaboración del programa que hay que corregir, interrumpido anteriormente a partir de la instrucción visualizada en el visualizador 6.

A continuación será el programa intérprete el que detenga elaboración y visualice la instrucción cuando la dirección prenotada es igual a la dirección de la instrucción que debe realizarse.

La utilidad de un programa de DBG de "prenotación de dirección de parada del programa" resulta evidente si se considera el caso de un programa que hay que corregir y que esté equivocado desde un cierto punto en adelante.

En este caso, conviene detener las elaboraciones directamente al comienzo del bloque de instrucciones en el que se supone que haya un error.

Si el programador, después de haber introducido ocho caracteres exadecimales acciona la barra S6 (alternativa lógica 425, figura 11c) el programa de DBG efectua un salto al bloque 446 de la figura 11f, que corresponde a la dirección 17EE de la tabla L.

A partir de esta dirección se registra el "programa de escritura en la RAM1" y a continuación los ocho caracteres exadecimales como se explicará acontinuación se consideran como una modificación del contenido de la RAM1.

La primera instrucción registrada en la dirección 17EE es una AP,2 que coloca el indicador P2-312 en el cuarto byte del registro 364, mientras que la instruccción TCP posterior coloca el indicador P1-311 en el primer byte del registro 364. Como se ha dicho con referencia al bloque 421 de la figura 11c, en el registro 364 se registran los caracteres introducidos desde el teclado.

Acto seguido se realiza la instrucción YTC, la cual almacena y transcodifica los caracteres introducidos presentes en el registro 364, estando representadas tales instrucciones simbólicamente por el bloque 446 de la figura 11f. A continuación, mediante las inst rucciones TLD,2; YBP,2 y TRD,1 se transfiere al registro P1-311 el contenido del registro 362, el cual, como se ha visto (bloque 420 de la figura 11c), contiene la dirección de la instrucción o datos presentes en el visualizador 6. A continuación se pone a cero mediante una TL,2 el registro P2-312 (bloque 447 de la figura 11f), Mediante las instrucciones YBP,2; AP,2 y MVC se transfie-

re el contenido del registro 364 al registro de memoria RAM1 direccionado por el registro P1-311 (bloque 448). Acto seguido se coloca el registro RB-310 en el valor Ø16Ø (bloque 449). Posteriormente se realiza un salto a la dirección 17BC (bloque 438 de la figura 11d) a partir del cual se visualiza la instrucción o los datos sucesivos a la modificada de la manera que se ha dicho anteriormente.

Se ha visto por lo tanto como introduciendo ocho CRT en el teclado 5 y accionando la barra S6, es posible modificar el contenido de la instrucción presente en el visualizador 6.

Si por último el operador acciona la tecla RUN de la consola 7 (alternativa lógica 427 de la figura 11c) se realiza como se ha dicho un salto a la dirección 18ØC. Por lo tanto se efectuan las instrucciones TLD,2 TLD,1 y MVC (bloque 450 de la figura 11g) para reponer en las posiciones CA y CB los dos bytes del registro RC-359 anteriormente transferidos al registro 367 (véase bloque 403 de la figura 11a); posteriormente, y mediante las instrucciones AP,2; AP,1 y MVC (bloque 451) se repone en el registro AB-370 el byte habilita-barras anteriormente transferido a la posición Ø18A (bloque 404 de la figura 11a).

Mediante las instrucciones SP,2; SP,1 y MVC (bloque 452) se repone también el registro de trabajo 352 anteriormente transferido al registro 366 (bloque 401 de la figura 11a).

Por último se realiza una instrucción YPS la cual repone el registro OPSR-301 en el registro PSR-300 y transfiere el registro IR-327 al direccionador de programa L07 (bloque 452). De esta manera se lee la instrucción del

programa en prueba posterior a la visualizada por la visualización. Mas concretamente, el programa intérprete:

1) efectua el salto a la dirección simbólica IINTE (alternativa lógica 202 y bloque 250, figuras 10a y 10b) ya que los bytes CP-313 y PI-314 tienen siempre el bit 05 al nivel 1, dado que la llave 100 se encuentra siempre en posición DBG.

2) dado que el bit 01 del BSD se encuentra al nivel cero (que corresponde a su posición normal) se efectua un salto a la dirección IINTE1 (alternativa lógica 257 y bloque 265 de la figura 10b); el bit b01 del BSD se coloca en 1 y se procede finalmente a la ejecución de la instrucción mediante el salto a IALFAR (bloque 265).

3) Una vez terminada dicha instrucción, el intérprete efectua las operaciones explicadas en el punto 1) sobre la instrucción que sigue, y en la alternativa lógica 257 procede en secuencia para todo lo dicho en el punto 2) además se coloca en cero el bit 01 de BSD, el cual,por consiguiente, es anulado siempre antes de la ejecución de un programa de DBG.

4) Como se ha dicho anteriormente (figuras 10b y 10c) se pasa a realizar el programa de DBG y acto seguido se visualiza la instrucción realizada.

Se ha visto, por lo tanto, que si el programador coloca la llave 100 en la posición de DBG y acciona la tecla RUN, se lleva a cabo la instrucción visualizada. Siempre que se acciona la tecla RUN el contador de programa se incrementa en cuatro unidades, como se ha dicho a propósito del intérprete, y el elaborador efectua la instrucción registrada en dicha dirección.

Mediante el accionamiento repetido de la tecla RUN es pues posible mandar la ejecución de una instrucción y la visualización de la instrucción sucesiva de un bloque del programa que se prueba. Esta secuencia alternada de ejecución de la instrucción con visualización de la sucesiva se llama ejecución "paso a paso" y su utilidad para el DBG de un programa que se prueba es evidente.

Con referencia a la figura 14, se procede ahora a una descripción conclusiva de las posibilidades ofrecidas al programador durante la fase de puesta a punto de un programa por el sistema de DBG según la invención.

Normalmente el programador carga en el calculador el programa que se quiere probar (bloque 700) y manda su ejecución.

Cuando se da cuenta de un error, gira la llave 100 a la posición DBG (alternativa lógica 701) y detiene de este modo la ejecución del programa de prueba, provocando después la visualización en el visualizador de la instrucción no realizada en el momento de la interrupción (bloque 702). El elaborador puede además pasar al funcionamiento en puesta a punto incluso cuando se ha reconocido la dirección de parada de las elaboraciones prenotadas anteriormente. En este caso el elaborador se comporta como si se hubiese accionado la llave 100 y por consiguiente también en este caso se atribuye a las barras S0, S6, el significado descrito anteriormente para el funcionamiento en DBG.

En este punto el programa de DBG habilita las barras S0, S1, S2, S6 y se pone en espera de introducción de datos por el teclado (bloque 703).

Según la barra accionada (alternativas lógi-

cas 704, 705) el elaborador efectua el programa correspondiente de DBG registrado en la ROM2.

Una vez terminados los programas de DBG acosiados a las barras S0, S1, S6 y a la tecla RUN, el elaborador vuelve al bloque 701 para solicitar la ejecución de otro programa de DBG.

Si, por el contrario, el operador ha mandado a través de la barra S2 la ejecución de un programa registrado en tarjeta magnética (bloque 706), la reanudación es controlada por el mismo programa de tarjeta magnética. En particular, se puede dar el retorno a la alternativa lógica 701 (programa de prueba o bien otros programas de DBG, o bien a la alternativa lógica 702 y por lo tanto directamente en DBG.

El programa registrado en tarjeta magnética ofrece al programador la posibilidad de activar otros programas de DBG registrados en otros soportes, y por consiguiente el programador tiene a su disposición todos los tipos de reposición previstos por él.

A continuación se exponen dos ejemplos simples de utilización del sistema de DBG según la invención.

Supongamos, como primer caso, que el programador debe buscar un error en un programa. Puede, por ejemplo, dividir el programa en bloques de un cierto número de instalaciones y buscar en el ámbito del bloque la instrucción equivocada. En este caso debe proceder de la manera siguiente.

- Gira la llave 100 a la posición DBG, y por lo tanto se visualiza la instrucción sucesiva a la de interrupción.
- Introduce en el teclado la dirección correspondiente de la

instrucción a la que quiere que se detenga el programa y acciona la barra S1, (bloques 710 y 711).

- Devuelve la llave 100 a la posición NORMAL y pulsa la tecla RUN.

El programa de prueba reanuda su ejecución hasta la dirección prenotada. En esta dirección el programa de prueba se detiene y se visualiza la instrucción registrada en la dirección prenotada (bloque 702). Dado que el programa se ha realizado hasta la dirección prenotada, esto indica que en dicho bloque no había errores formales. Supongamos también que no haya errores propios del programa de prueba. El programador prenota del modo visto anteriormente la dirección del bloque sucesivo, devuelve la llave 100 a la posición NORMAL y pulsa la tecla RUN.

Supongamos ahora que el programador detecta un error de programa en el segundo bloque, y que quiera descubrir la instrucción que debe corregir. En este caso - marca en el teclado la dirección inicial del bloque y acciona la barra S0 (bloques 712 y 713). El programa de DBG procede a visualizar en la visualización la dirección y la instrucción correspondiente volviendo acto seguido a los bloques 702 y 703. En este punto el programador acciona la tecla RUN activando el funcionamiento paso a paso. De esta manera el programador puede realizar instrucción por instrucción el bloque en el que ha supuesto que hay un error.

Cuando detecta que la instrucción visualizada es equivocada, en vez de accionar la tecla RUN prepara en el teclado la instrucción correcta y acciona la barra S6. De esta manera el programa de DBG substituye en la dirección visualizada la instrucción equivocada por la introducida en el teclado.

De este modo el programador tiene la posibilidad de controlar todas las instrucciones del programa y aportar las correciones que considere adecuadas.

Supongamos ahora, como segundo caso, que el programa que hay que poner a punto ocupe toda la parte disponible de la memoria.

Supongamos además que al final de todas las posibles operaciones de DBG efectuadas por los programas correspondientes que se encuentran en la ROM2, el programador no ha podido eliminar los errores.

Es evidente por lo expuesto que el programa que se prueba no podrá ser reanudado, y por consiguiente el programador deberá ver de nuevo su programa.

Para hacer esto, le conviene tener impreso un contenido tabulado de la RAM1. De esta manera, el programador, analizando dicho resultado, puede llegar hasta las instrucciones equivocadas. Naturalmente, para realizar la impresión del contenido de la memoria, es preciso que un programa adecuado de DBG controle su ejecución. Dado que este programa no forma parte de los registros en la ROM2, el programador puede utilizar una tarjeta magnética en la que se encuentra registrado dicho programa.

Para conseguir la impresión del contenido de la memoria, el programador debe actuar del siguiente modo:

- Introduce la tarjeta magnética en el introductor 9'.
- Coloca la llave 100 en DBG.
- Acciona la barra S2.

Como se ha visto anteriormente, esto provoca que el programa registrado en tarjeta magnética se transfiere a los registros 360-367 de la ZRM, y se efectua inmedia

tamente.

Se recuerda que los registros 360-367 no están reservados exclusivamente a los programas de DBG, sino que se utilizan durante el funcionamiento normal para contener resultados intermedios de algunas instrucciones.

Se hace observar además que tales registros 360-367 están de todas maneras presentes en cualquier elaborador, porque es siempre necesario memorizar resultados intermedios en algunas instrucciones (como por ejemplo en las instrucciones de multiplicación y división). Resulta evidente de todo lo expuesto que la utilización de tales registros de la ZRM permite cargar y realizar cualquier programa registrado en una tarjeta magnética sin alterar los parámetros significativos del programa que se prueba.

Las ventajas de esta posibilidad no se derivan exclusivamente del programa de DBG registrado en tarjeta magnética, el cual es limitado per se, sino que se derivan sobre todo de la posibilidad que tiene dicho programa de poder liberar una zona de memoria sin destruir su contento y de registrar en tal zona un programa de DBG de longitud y complejidad superior.

De todo lo expuesto aparece evidente que el programador, utilizando el sistema de DBG según la invención, es totalmente independiente por lo que se refiere a la preparación y puesta a punto de un programa o una modificación de un programa existente.

Aparece también claro que la gama de posibilidades ofrecida al programador del sistema según la invención, sea en efecto muy superior a todo lo descrito anteriormente. En efecto, estas posibilidades dependen esen-

cialmente de la complejidad del problema tratado, del número de programas de DBG a disposición y por último de la capacidad del programador.

Se comprende que pueden aportarse modificaciones, substituciones o adiciones de partes al sistema de puesta a punto de los programas que acaba de describirse sin salirse por ello del ámbito de la presente invención.

Por ejemplo, la llave 100 que sirve al programador para activar en los programas de puesta a punto puede ser substituída por una tecla que habilita un conmutador, siendo el resultado esencial que el nivel lógico de la señal presente en el hilo 61 de la figura 2c se conmute de cualquier manera.

El soporte externo de los programas de puesta a punto a disposición del programador, puede ser, aparte de una tarjeta magnética, una cinta magnética, un grupo de tarjetas perforadas, etc. El único vínculo es que el número de los caracteres registrados en dicho soporte sea como máximo igual al número de bytes a disposición en la ZRM.

Los programas de DBG que se encuentran en la memoria ROM2 y que se han descrito anteriormente, pueden variar tanto en número como en contenido, según la zona disponible de la ROM2 y las exigencias del calculador.

La función reservada a las barras S0-S6 durante la ejecución de los programas de DBG que se encuentran en ROM2, puede hacerse con otras teclas alfabéticas comprendidas entre G y Z, siendo utilizadas las que están entre A y F por el código exadecimal para modificar las instrucciones. También las teclas añadidas expresamente para solicitar los programas de DBG que se encuentran en la ROM2,

o desviadores o llaves, entran en las posibles variantes del sistema de puesta a punto de los programas ilustrados en las páginas anteriores.

Los registros 360-367 de la ZRM de la RAM1 que durante el funcionamiento en DBG memorizan el programa de DBG registrado en un soporte externo, pueden disponerse de manera distinta a la ilustrada en la figura 9. Mas particularmente, estos registros pueden ocupar cualquier posición de la ZRM, a condición de que su contenido sea no significativo al término de las instrucciones que las han utilizado. Además estos registros pueden estar dispuestos en la ZRM no secuencialmente, sino distribuidos en grupos separados. En este caso, el programador deberá encargarse de direccionar correctamente tales registros durante la lectura de la tarjeta magnética.

Finalmente, tanto los programas de DBG registrados en la ROM2, como la zona de RAM1 (registros 360-367) destinada a recibir otros programas de DBG registrados en soportes externos (por ejemplo tarjeta magnética), pueden ser distribuidos de manera distinta. En particular pueden ser distribuidos en una zona de la RAM1 predispuesta para recibir únicamente programas de DBG o bien en una tercera memoria de la unidad central 3 destinada exclusivamente a contener programas de DBG.

Finalmente, debe precisarse que el reconocimiento de las condiciones externas que interesan a los programs de DBG (llave 100 y barras 102) y la preparación sucesiva de los registros de ZRM destinados a recibir dichas condiciones, puede hacerse de cualquier otro modo distinto del que se ha descrito.

N O T A .-

Descrita suficientemente la naturaleza del invento, así como la manera de realizarlo en la práctica, debe hacerse constar que las disposiciones anteriormente indicadas, son susceptibles de modificaciones de detalle en cuanto no alteren su principio fundamental; también se hace constar, que el invento corresponde a una solicitud de patente, presentada en Italia, bajo el número 70879-A/73, de fecha de 28 de diciembre de 1.973, acogiéndose por lo tanto a los beneficios que conceden los Convenios Internacionales en vigor, siendo lo que constituye la esencia del referido invento, y por lo que se solicita Patente de Invención por 20 años en España, sobre. PERFECCIONAMIENTOS EN CALCULADORAS ELECTRONICAS; caracterizándose por lo siguiente:

1ª.- Perfeccionamientos en calculadoras electrónicas, del tipo que están dotadas con equipo para poner a punto programas ejecutables, que comprende una primera memoria para almacenar instrucciones y datos de dichos programas ejecutables, una unidad central para tratar dichos programas, un elemento de conmutación para definir selectivamente un primer modo de operación de la citada calculadora para tratar los programas ejecutables y un segundo modo de operaciones para tratar un programa de puesta a punto para poner a punto los citados programas ejecutables, y un teclado para introducir en la citada primera memoria información relacionada con los citados programas operativos durante el primer modo de operación, caracterizados porque la calculadora comprende medios controlados por el citado elemento de conmutación para condicionar la unidad central, para interrumpir los citados programas ejecutables, activar el segundo modo de ope-

ración y permitir que el mencionado teclado introduzca la información que debe ser tratada por dicho programa de puesta a punto.

2ª.- Perfeccionamientos según la reivindicación 1ª, caracterizados porque cuando la calculadora comprende además una unidad visualizadora para visualizar información relacionada con dichos programas ejecutables durante el primer modo de operaciones, el citado elemento conmutador hace que la unidad central asigne la unidad visualizadora al programa de puesta a punto durante el segundo modo de operación.

3ª.- Perfeccionamientos según la reivindicación 2ª, caracterizados porque el citado elemento de conmutación hace que durante el segundo modo de operación la unidad central transmita al visualizador la dirección de la próxima instrucción que debe ejecutarse del citado programa ejecutable y un código de la información almacenada en la primera memoria en dicha dirección.

4ª.- Perfeccionamientos según la reivindicación 3ª, caracterizados porque cuando la calculadora comprende además un registro de dirección de memoria para almacenar la dirección absoluta de la próxima instrucción que debe ejecutarse, y un registro de base para almacenar la dirección de base del citado programa ejecutable, la unidad de control es condicoonada por el elemento de conmutación para calcular la dirección relativa de la próxima instrucción restando el contenido del citado registro de base del contenido del registro de dirección de memoria y para enviar dicha dirección relativa a la unidad visualizadora.

5ª.- Perfeccionamientos según la reivindica-

ción 4ª, caracterizados porque cuando la calculadora comprende además un registro de programa de condición para almacenar resultados significativos de instrucciones anteriormente realizadas, el citado elemento de conmutación hace que la unidad central envíe durante el segundo modo de operación a la unidad de visualización el contenido del registro de condición de programa.

6ª.- Perfeccionamientos según cualquiera de las reivindicaciones anteriores, caracterizados porque el programa de puesta a punto se encuentra dividido en una serie de bloques y el teclado comprende un grupo de teclas predeterminadas controladas por el elemento de conmutación durante el segundo modo de operaciones para controlar selectivamente el tratamiento de la información introducida por el bloque de dicho programa de puesta a punto correspondiente a la tecla accionada de dicho grupo.

7ª.- Perfeccionamientos según las reivindicaciones 1ª, 2ª ó 6ª, caracterizados porque cuando la calculadora comprende además una unidad periférica para introducir información registrada en un soporte externo en la citada unidad central, comprende a su vez una segunda memoria para almacenar la información registrada en dicho soporte, por una tecla predeterminada incluida en dicho teclado para hacer que la unidad central transfiera el programa leído al citado soporte en la segunda memoria y para activar el programa almacenado en dicha segunda memoria.

8ª.- Perfeccionamientos según la reivindicación 7ª, caracterizados porque dicha segunda memoria se incluye en la primera memoria.

9ª.- Perfeccionamientos según la reivindica-

ción 8ª, caracterizados porque la citada unidad central registra en la segunda memoria datos no importantes al final de cada instrucción para dichos programas operativos durante el primer modo de operación.

10ª.- Perfeccionamientos según la reivindicación 9ª, caracterizados porque dicha unidad periférica comprende un lector de tarjetas magnéticas y el citado soporte de registro comprende una tarjeta magnética.

11ª.- Perfeccionamientos según cualquiera de las reivindicaciones anteriores, caracterizados porque el citado elemento de conmutación genera selectivamente una primera señal que tiene un primer nivel lógico asociado con el primer modo de operación y un segundo nivel lógico asociado al segundo modo de operación, y que los citados medios controlados están condicionados por el segundo nivel lógico de la primera señal para interrumpir el primer modo de operación y activar el segundo modo de operación.

12ª.- Perfeccionamientos según la reivindicación 11ª, caracterizados porque los citados medios controlados comprenden un primer registro de la primera memoria, para registrar la primera señal lógica, un segundo registro de la primera memoria para registrar una segunda señal que habilita a la primera señal, siendo controlada la unidad central por el segundo nivel lógico de la primera señal y por la segunda señal de habilitación para activar el segundo modo de operación.

13ª.- Perfeccionamientos según las reivindicaciones 4ª a 12ª, caracterizados porque cuando la calculadora comprende además una tercera memoria para almacenar microprogramas asociados a dichas instrucciones, caracterizada

por el hecho de que los citados medios activadores comprenden un tercer registro de la primera memoria para registrar una tercera señal que tiene un primer nivel lógico asociado a la primera memoria y un segundo nivel lógico asociado a la tercera memoria, siendo controlada la unidad central por el primero y el segundo nivel lógico de la tercera señal para registrar respectivamente en el citado registro de dirección de memoria la dirección de la primera y la tercera memoria de la instrucción siguiente y, en el segundo registro una cuarta señal que tiene un primero y segundo nivel lógico asociado a la primera y tercera memoria, respectivamente.

14ª.- Perfeccionamientos según la reivindicación 13ª, caracterizados porque dicha unidad central es controlada por el primer nivel lógico y por el segundo nivel de la cuarta señal para leer la siguiente instrucción en la primera y tercera memoria, respectivamente.

15ª.- Perfeccionamientos según la reivindicación 14ª, caracterizados porque el programa de puesta a punto se registra en una zona predeterminada de la tercera memoria y que la unidad central es controlada por el segundo nivel lógico de la tercera señal para registrar respectivamente en el registro de dirección de memoria la dirección inicial de dicha zona predeterminada y en el segundo registro el segundo nivel lógico de la cuarta señal.

16ª.- Perfeccionamientos según la reivindicación 15ª, caracterizados porque la calculadora comprende un cuarto registro de la primera memoria para registrar combinaciones de señales asociadas a dichas teclas predeterminadas, siendo activada la unidad central por las citadas combinacjones de señales para seleccionar dichos bloques corres-

pondientes.

17ª.- Perfeccionamientos según la reivindicación 16ª, caracterizados porque la unidad central es controlada por dicha combinación de señales para escribir selectivamente en el citado registro de dirección de memoria la dirección de los citados bloques asociados con las mencionadas combinaciones predeterminadas de señales.

18ª.- Perfeccionamientos en calculadoras electrónicas; tal y como queda sustancialmente descrito en la presente Memoria e ilustrado en los dibujos adjuntos.

Esta Memoria consta de 122 hojas escritas a máquina por una sola cara.

Madrid, 20 FEB. 1975

Ing. C. OLIVETTI & C., S.p.A.

A. SUAREZ ARIAS Y MODET
P. P. Firmado: L. Gasia Fernández

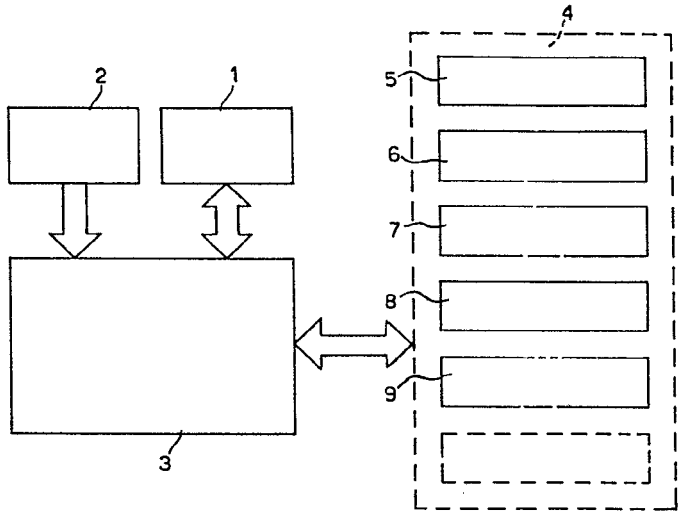


FIG. 1a

L00	
L01	
L02	
B03	A03
B04	A04
B05	A05
B06	A06
L07	
B08	A08
B09	B09
B10	A10
B11	A11
B12	
B13	
B14	A14
B15	A15

30

31

32

FIG. 8

Madrid 20 FEB. 1976

J. GÓMEZ ACEBO Y [illegible]
P. de Poderes L. Gach Ramón

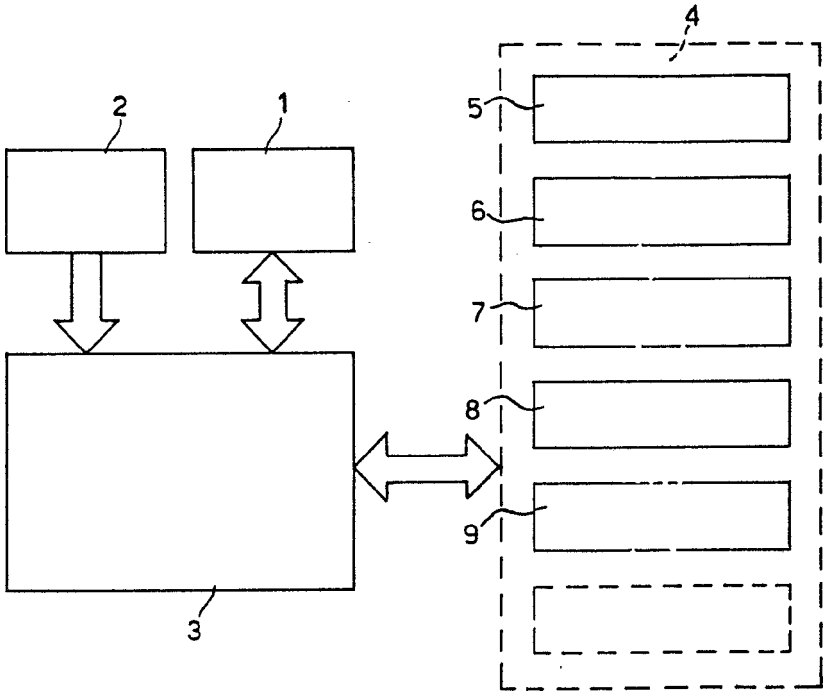


FIG. 1a

FIG.

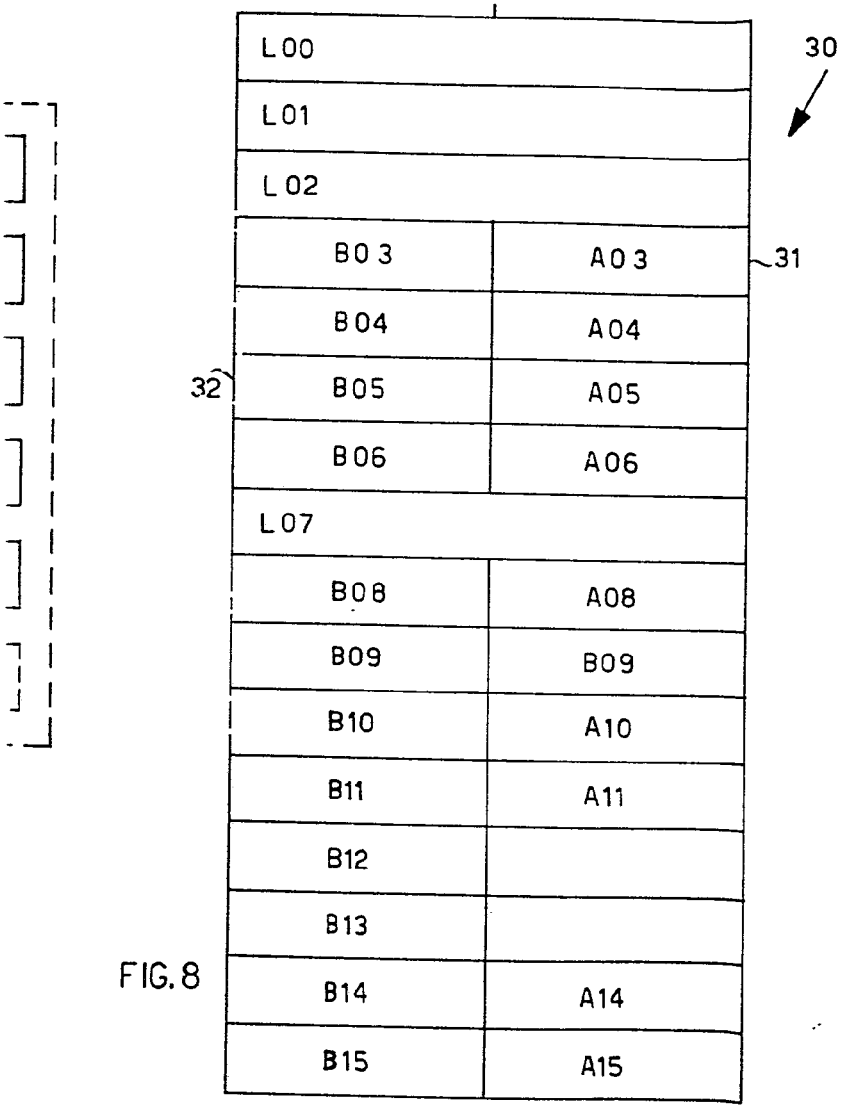


FIG. 8

Madrid 20 FEB. 1975

I. [illegible] ACCESO Y [illegible]
[illegible] Francisco L. Costa [illegible]

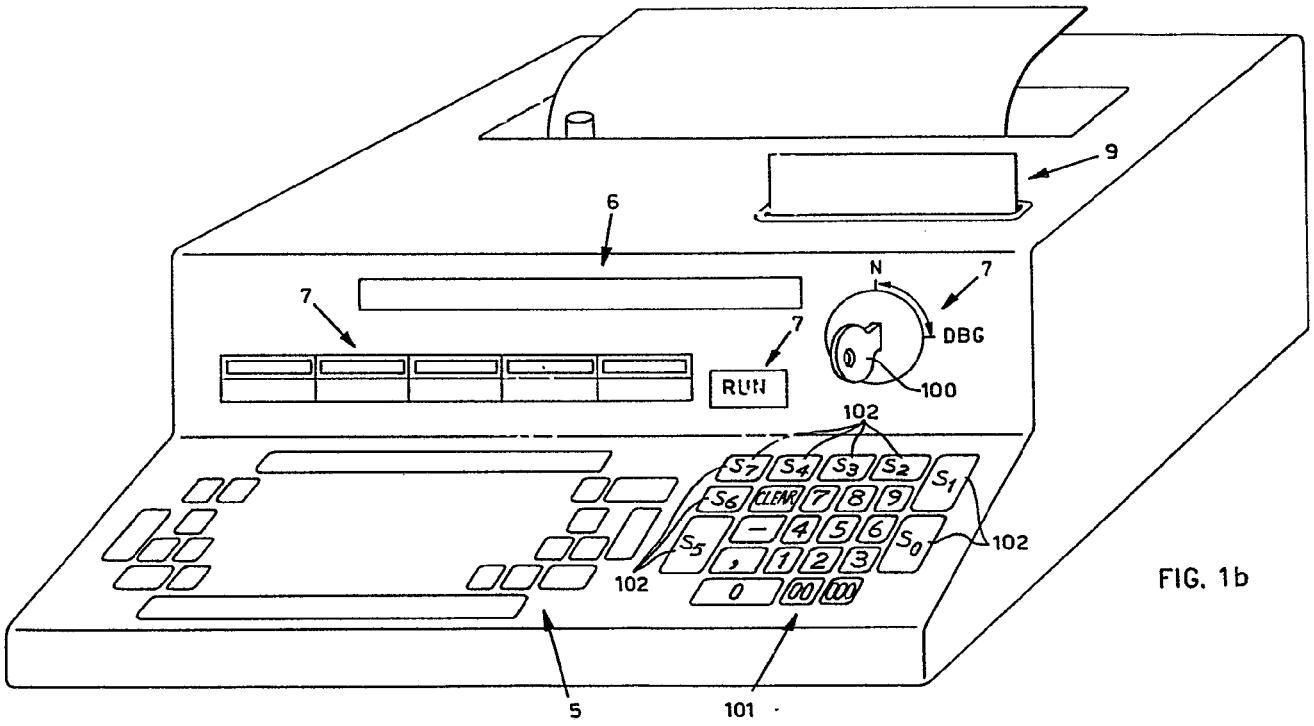
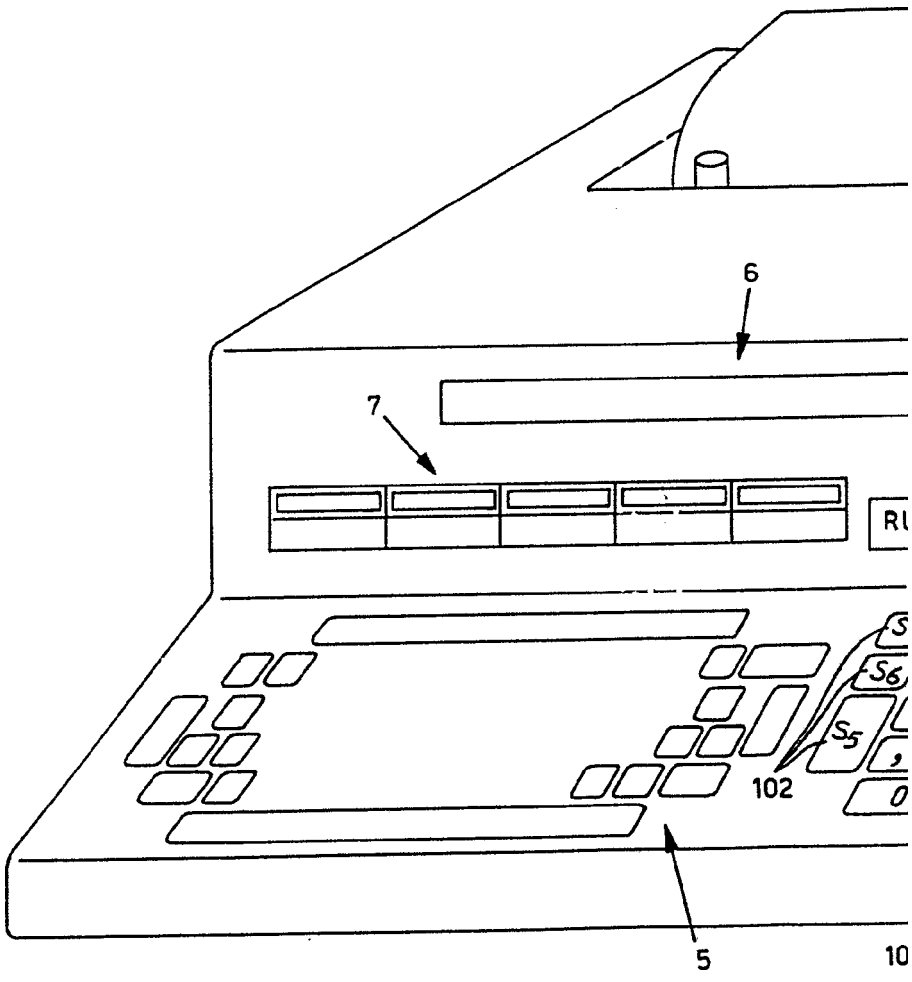


FIG. 1b

20 FEB. 1975

Madrid

J. GOMEZ ACEBO Y MODET
P. Fernandez L. Garcia Fernandez


6
7
S6
S5
102
5
10

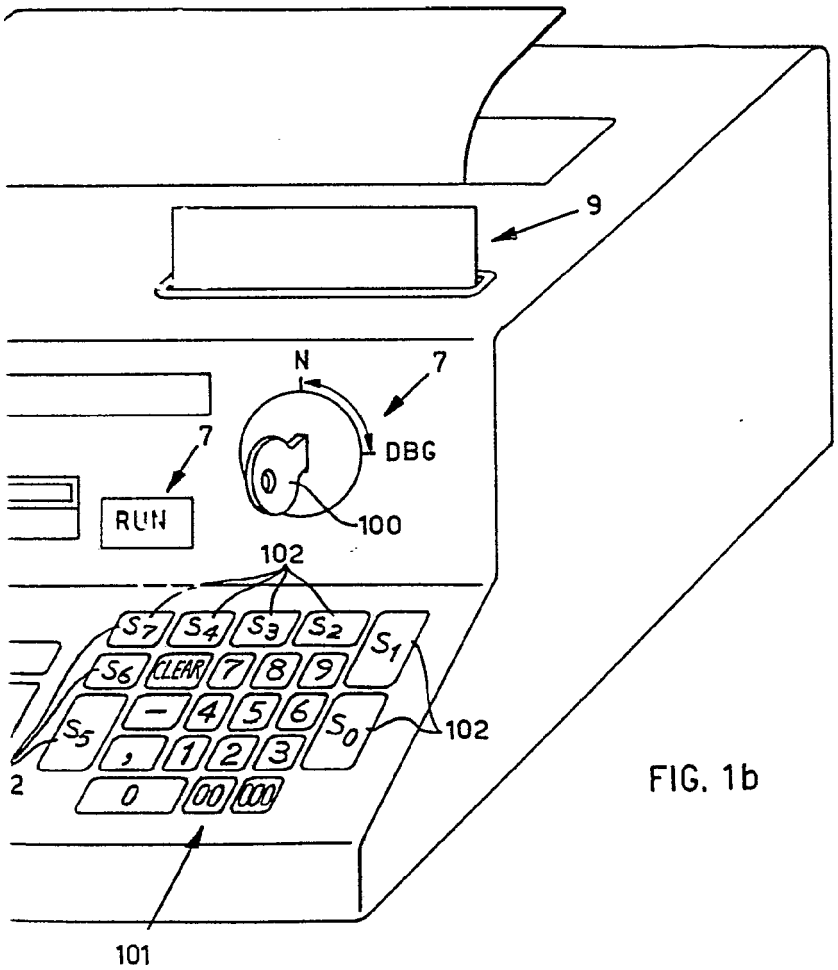


FIG. 1b

20 FEB. 1975

Madrid

I. GOMEZ ACEBO Y MODET

P. A. Firmado: L. Gaeta Fernández

Fig. 2a	Fig. 2b	Fig. 2c

Fig. 2

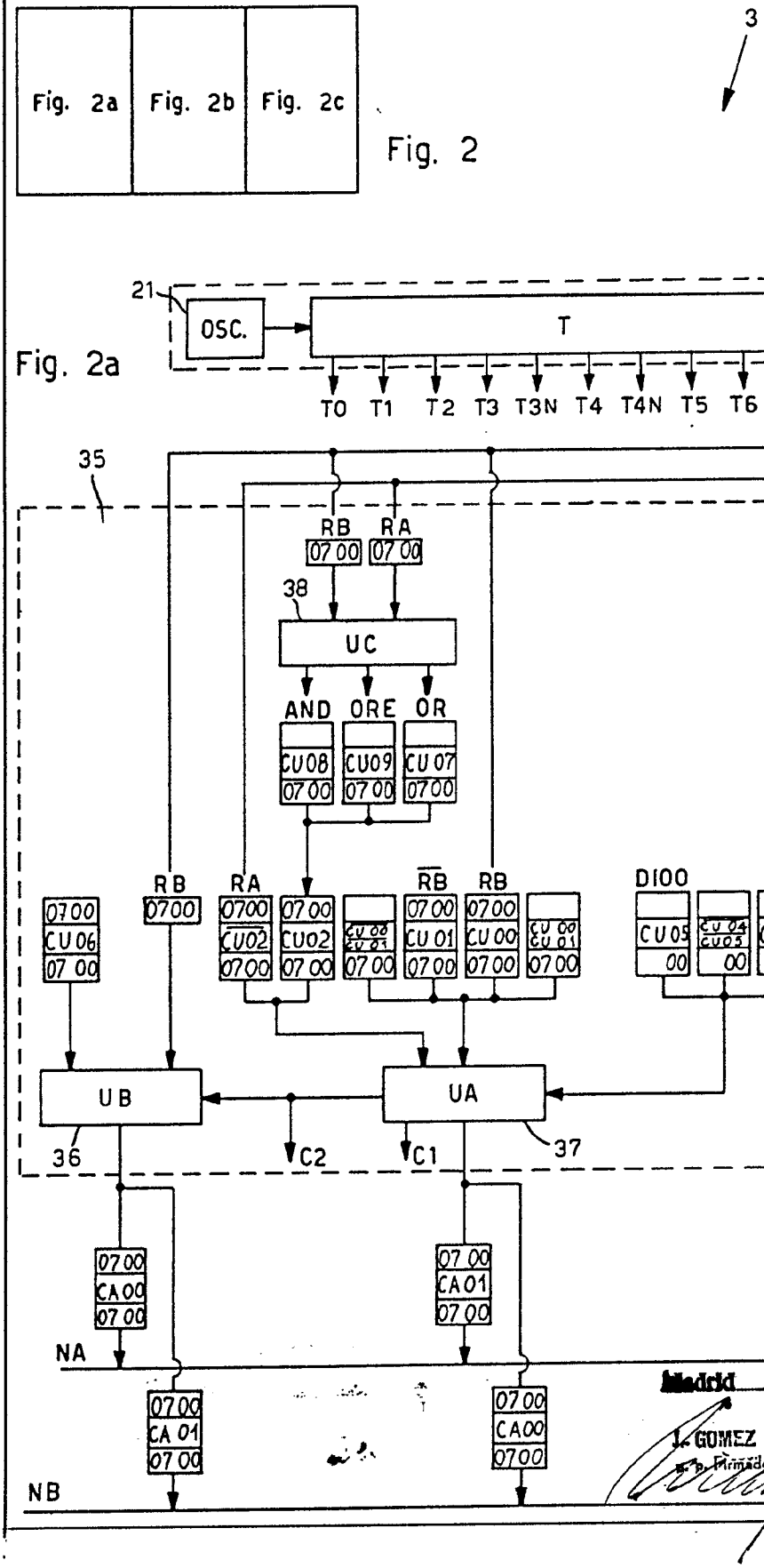


Fig. 2b

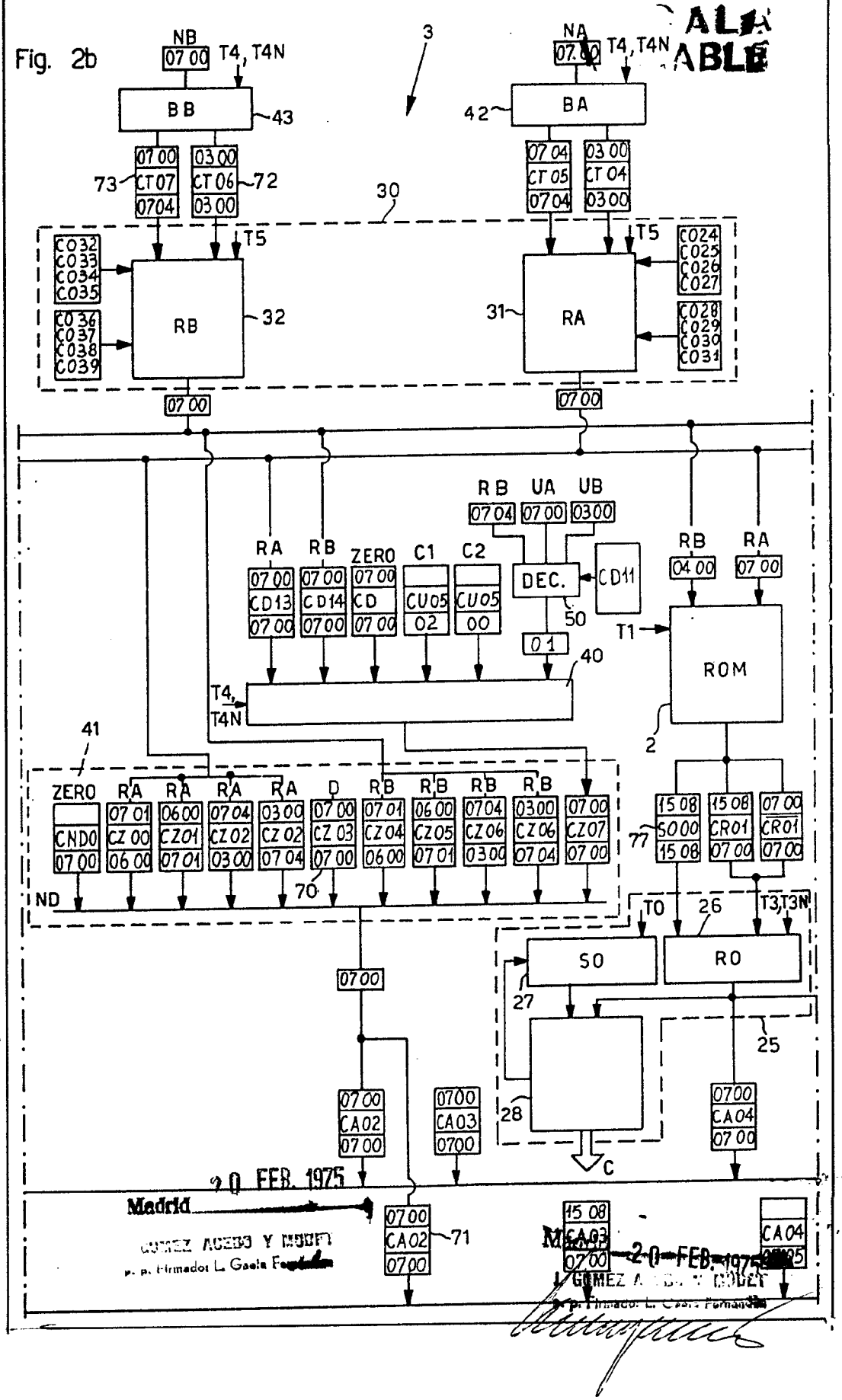
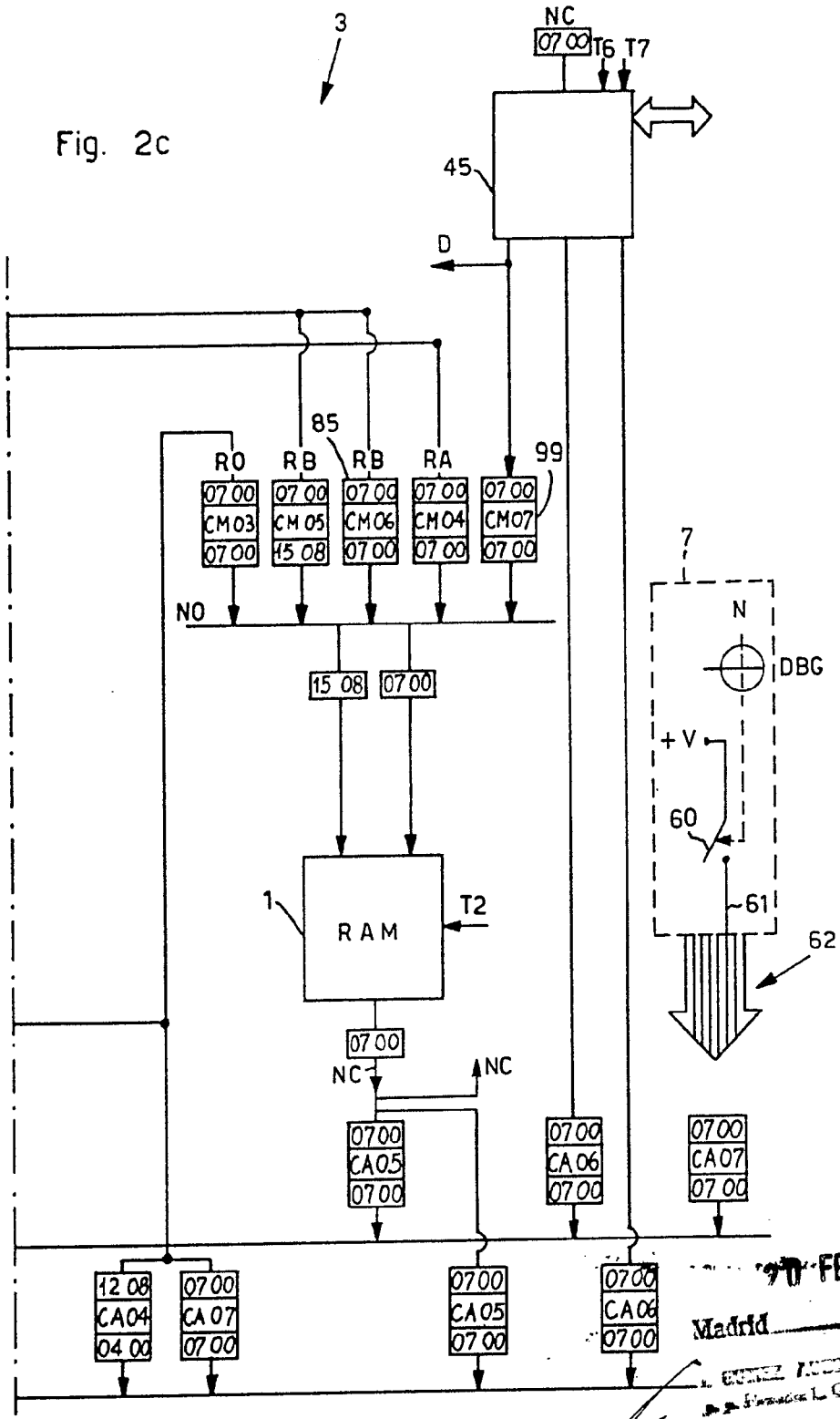

NB
07 00
T4, T4N
BB
43
3
NA
07 00
T4, T4N
42
BA
07 00
CT 07
07 04
73
03 00
CT 06
03 00
72
07 04
CT 05
07 04
03 00
CT 04
03 00
30
T5
T5
C032
C033
C034
C035
C036
C037
C038
C039
RB
32
31
RA
C024
C025
C026
C027
C028
C029
C030
C031
07 00
07 00
RB UA UB
07 04 07 00 03 00
RA RB ZERO C1 C2
07 00 07 00 07 00
CD13 CD14 CD
07 00 07 00 07 00
CU05 CU05
02 00
DEC.
CD11
50
01
40
T4,
T4N
RB RA
04 00 07 00
T1
ROM
2
41
ZERO RA RA RA RA D RB RB RB RB RB
CND0 07 01 06 00 07 04 03 00 07 00 07 01 06 00 07 04 03 00 07 00
07 00 CZ 00 CZ01 CZ02 CZ 02 CZ 03 CZ 04 CZ05 CZ 06 CZ 06 CZ07
06 00 07 01 03 00 07 04 07 00 06 00 07 01 03 00 07 04 07 00
70
ND
15 08 15 08 07 00
77
5000 CR01 CR01
15 08 07 00 07 00
T0
26
T3,T3N
SO
RO
27
25
28
C
07 00
CA02
07 00
07 00
CA03
07 00
07 00
CA04
07 00
20 FEB. 1975
Madrid
GOMEZ ACEBO Y MODET
p. p. Firmado L. Gaeta Fernández
71
15 08
CA03
07 00
20 FEB. 1975
CA04
GOMEZ ACEBO Y MODET
p. p. Firmado L. Gaeta Fernández

Fig. 2c



FEB. 1975

Madrid

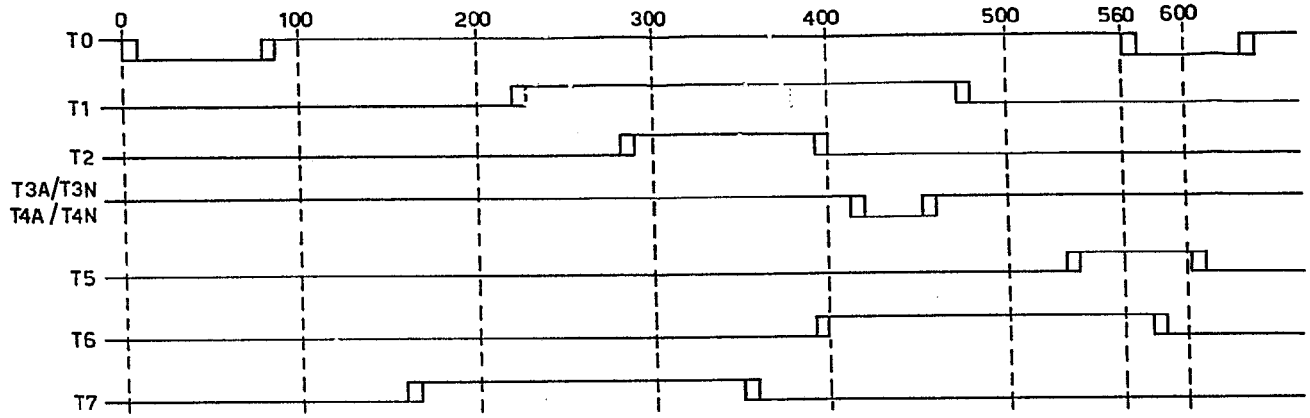


FIG. 3

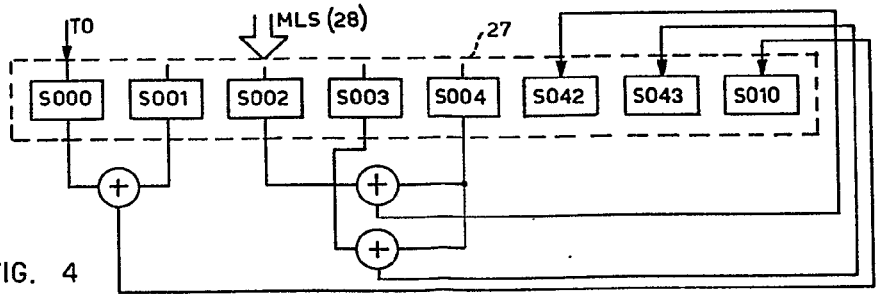


FIG. 4

Madrid 20 FEB. 1975

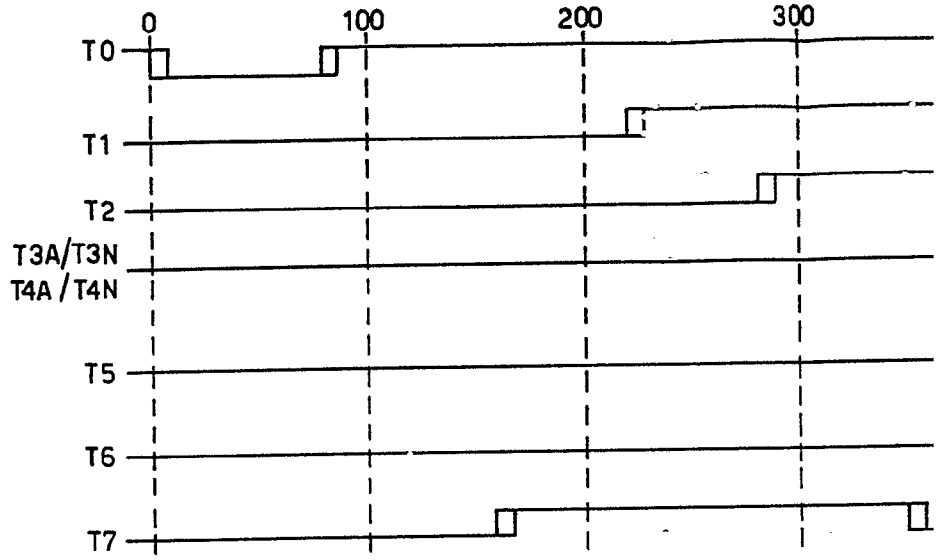

0
100
200
300
T0
T1
T2
T3A/T3N
T4A/T4N
T5
T6
T7

FIG. 3

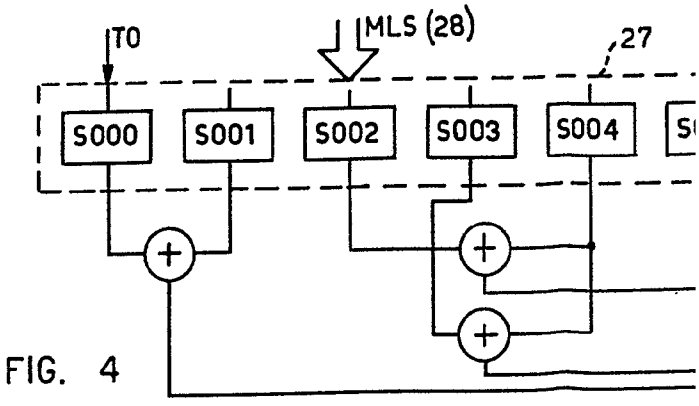
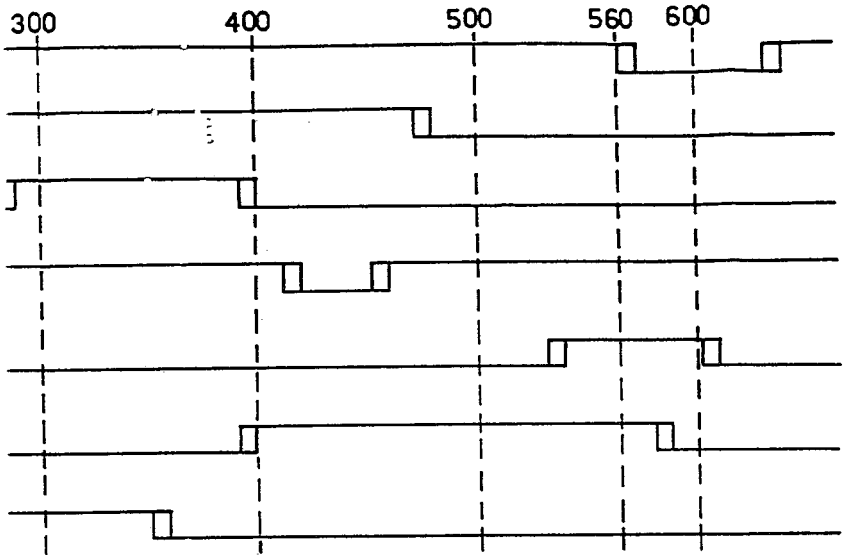
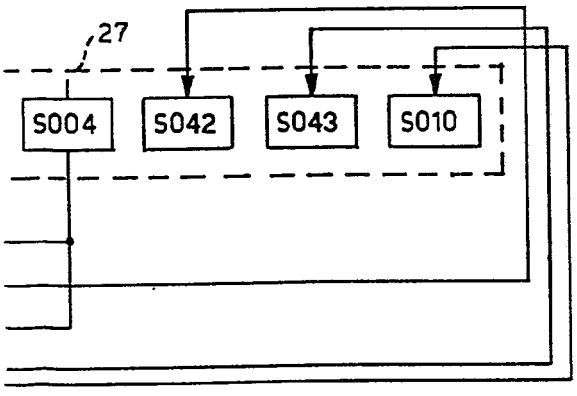

T0
MLS (28)
27
S000
S001
S002
S003
S004

FIG. 4





Madrid 20 FEB. 1975

p. p. Firmado L. Gesto Fernández

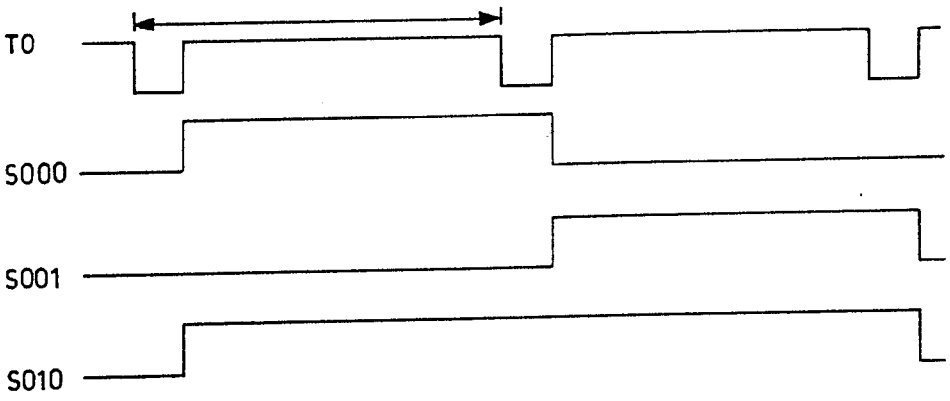


FIG. 5

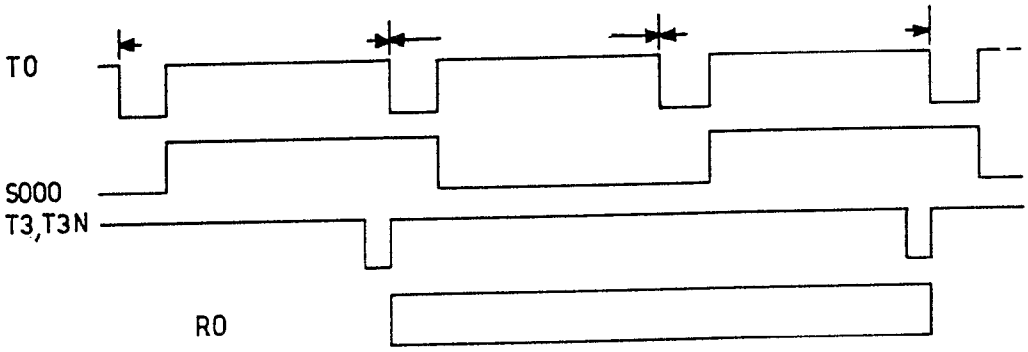
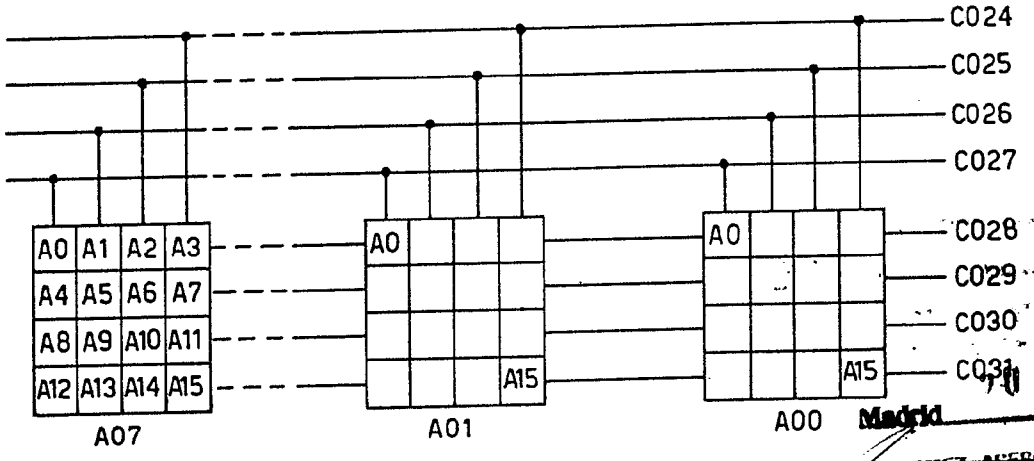


FIG. 6

FIG. 7



10 FEB. 1975
Madrid
I. GOMEZ ACEBO Y MODET
D. P. Firmador L. Gomez Fernández

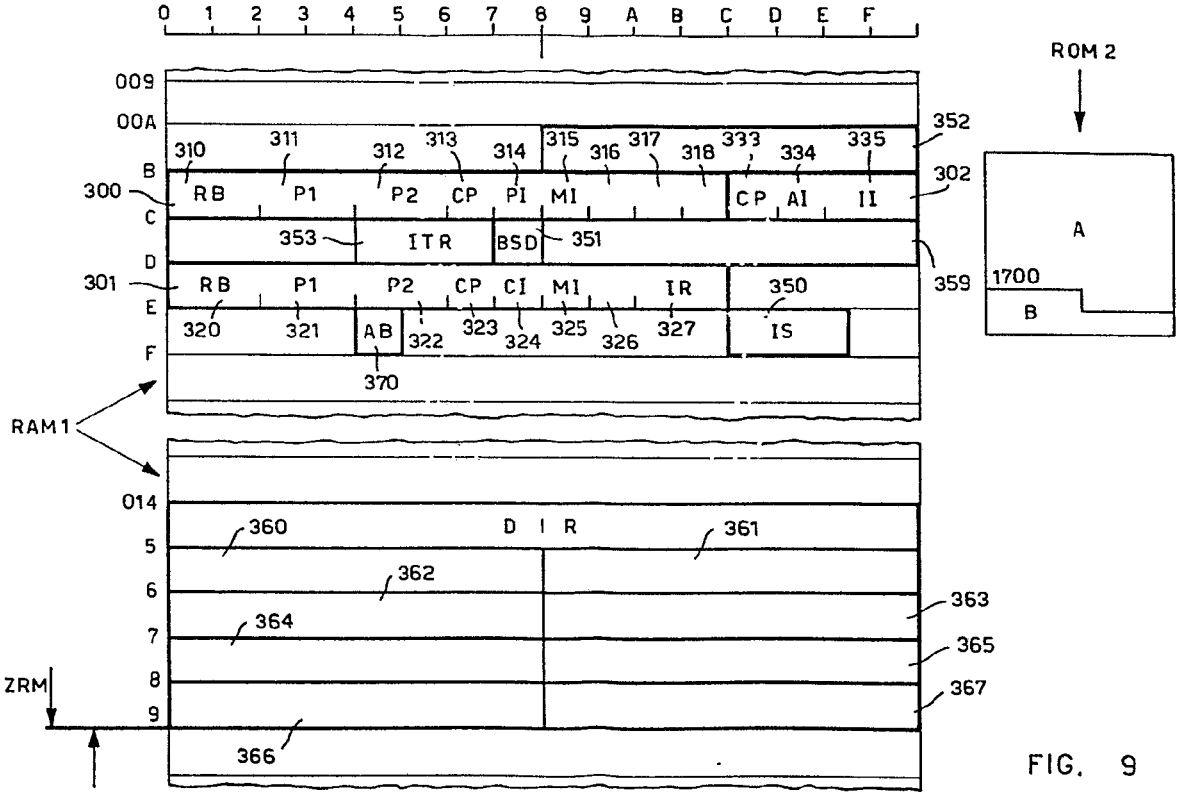
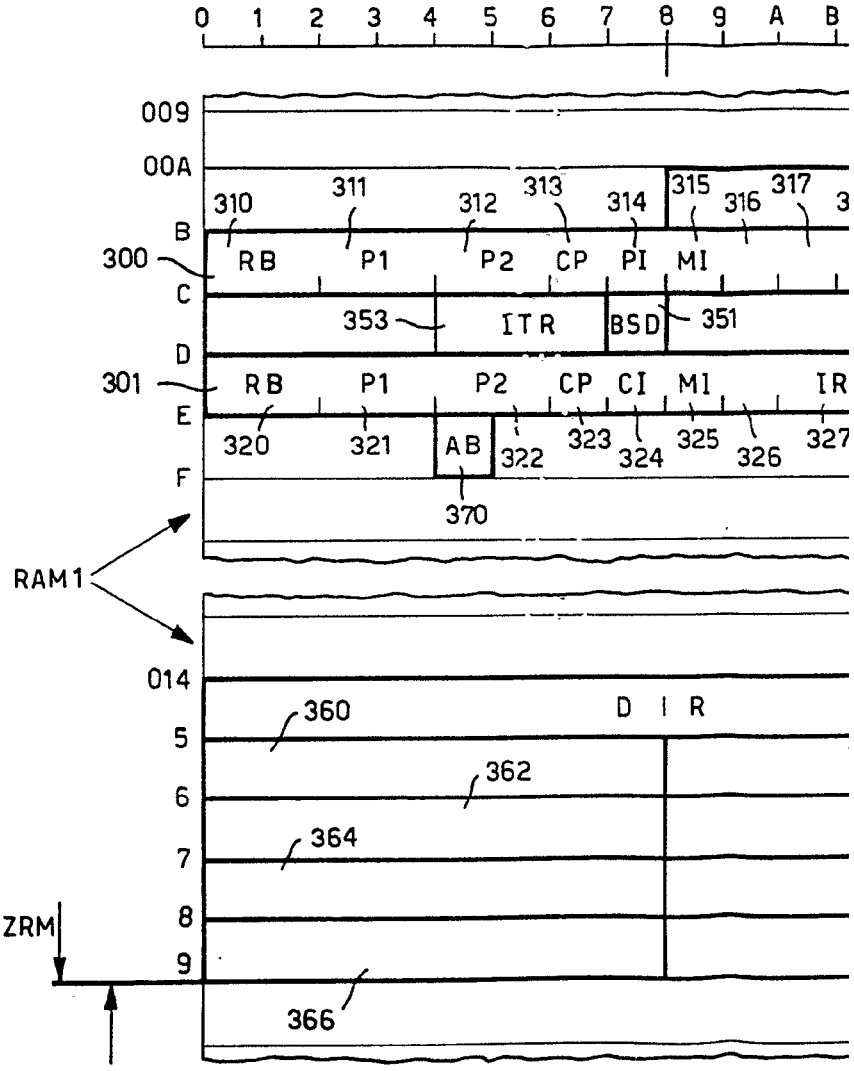


FIG. 9

Madrid 11 FEB. 1975

I. GOMEZ ACEBO Y MODET

p. p. Firmado: L. Gaeta Fernández


0 1 2 3 4 5 6 7 8 9 A B
009
00A
B
C
D
E
F
014
5
6
7
8
9
310
311
312
313
314
315
316
317
300
RB
P1
P2
CP
PI
MI
353
ITR
BSD
351
301
RB
P1
P2
CP
CI
MI
IR
320
321
AB
322
323
324
325
326
327
370
RAM1
360
D I R
362
364
ZRM
366

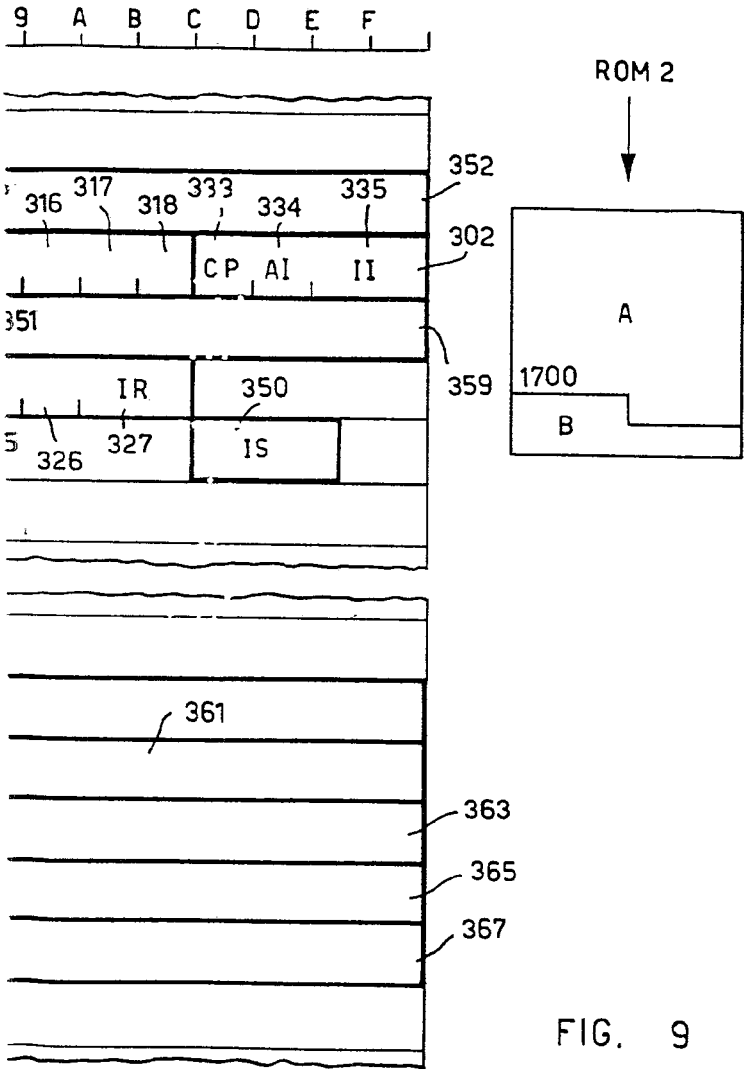


FIG. 9

Madrid 21 FEB. 1975

I. GOMEZ ACEBO Y MODET

p. p. Firmado L. Costa Fernández

(CP)

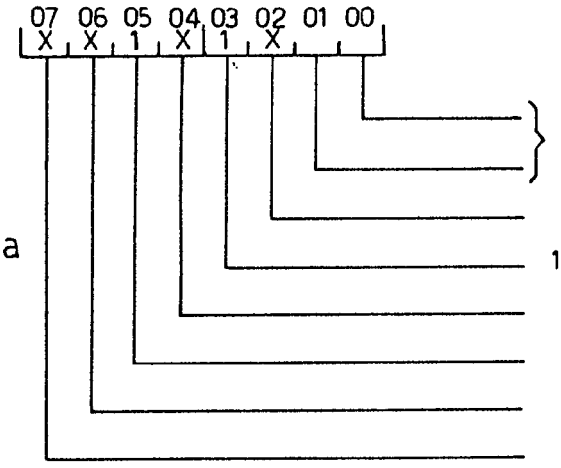


FIG. 9a

(CI)

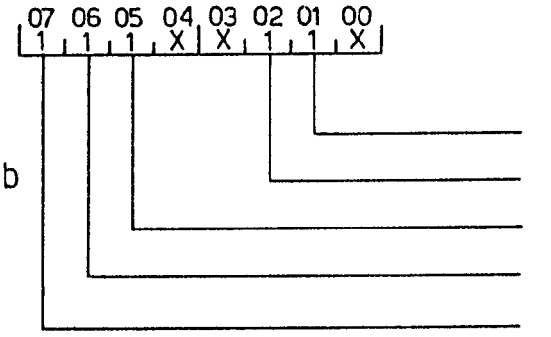


FIG.9b

(AI)

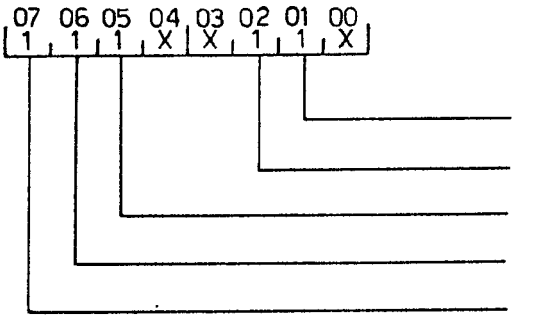


FIG.9c

(BSD)

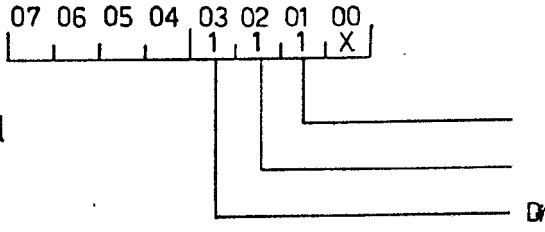


FIG.9d

20 FEB. 1975

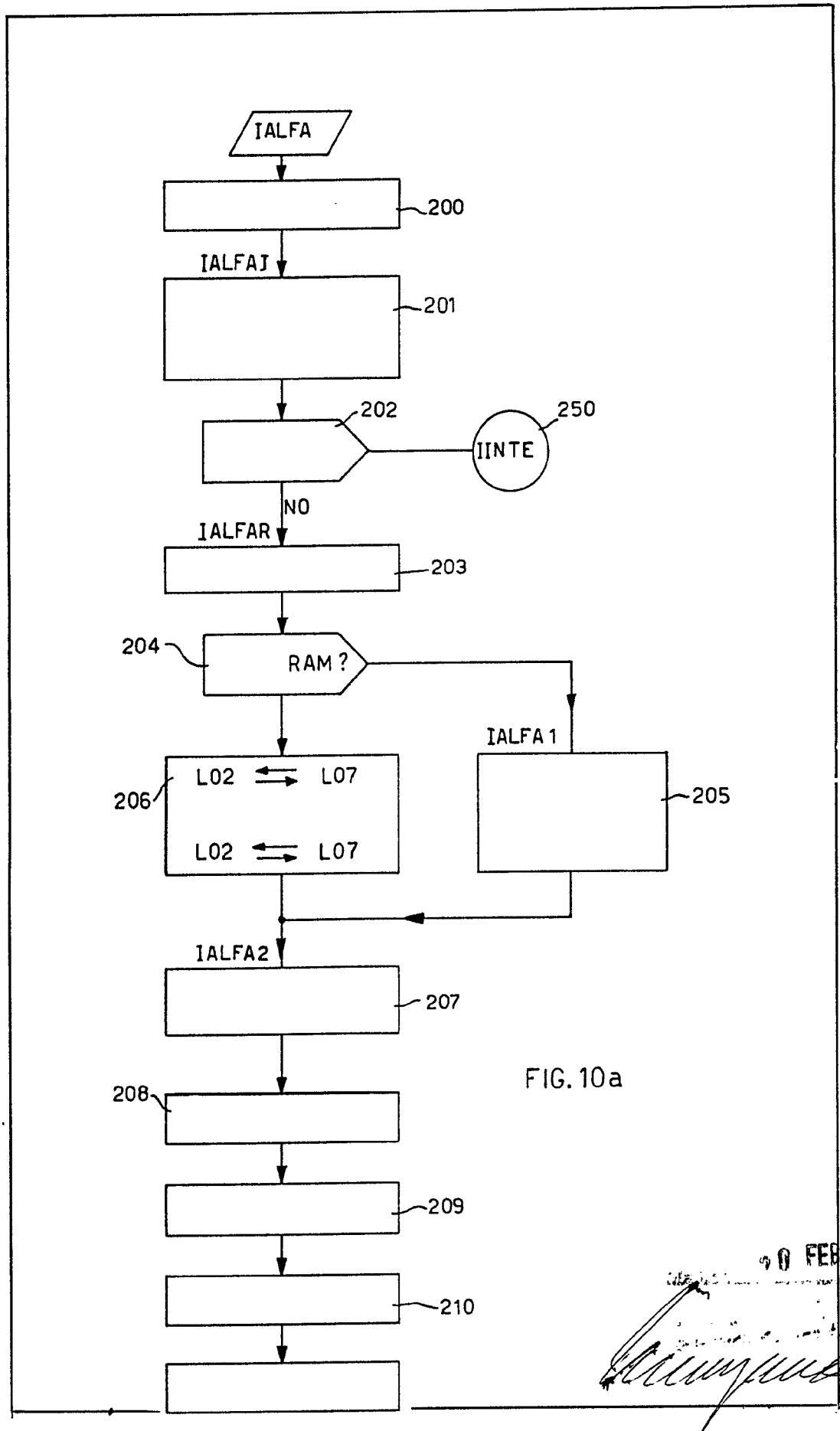


FIG. 10a

20 FEB. 1975

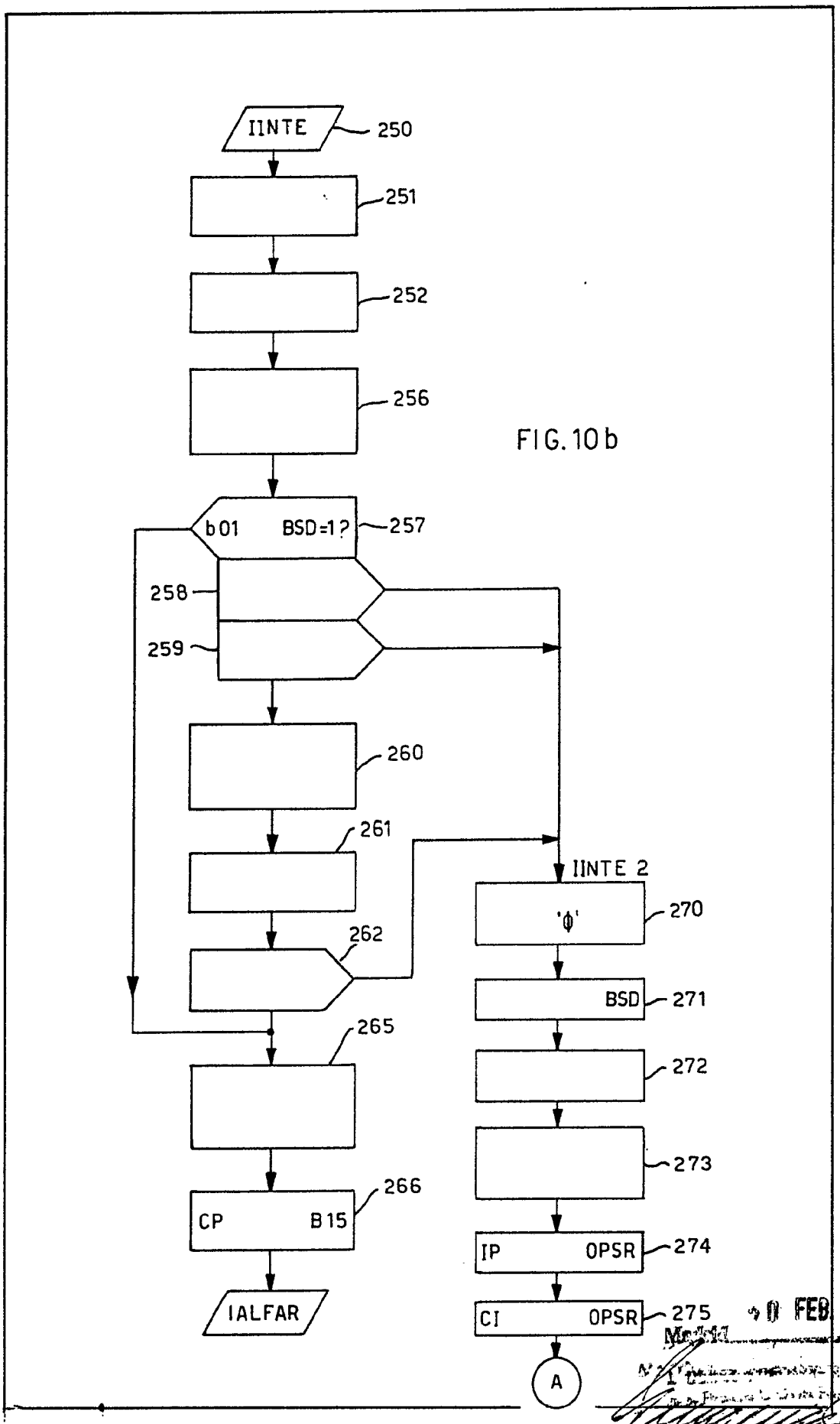


FIG. 10b

10 FEB 1975

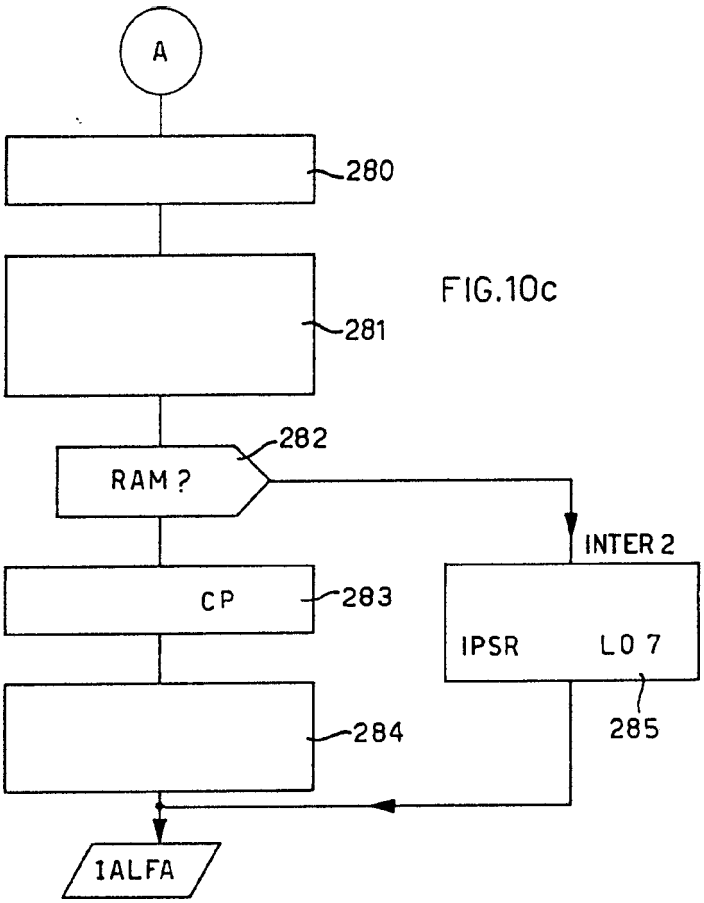


FIG.10c

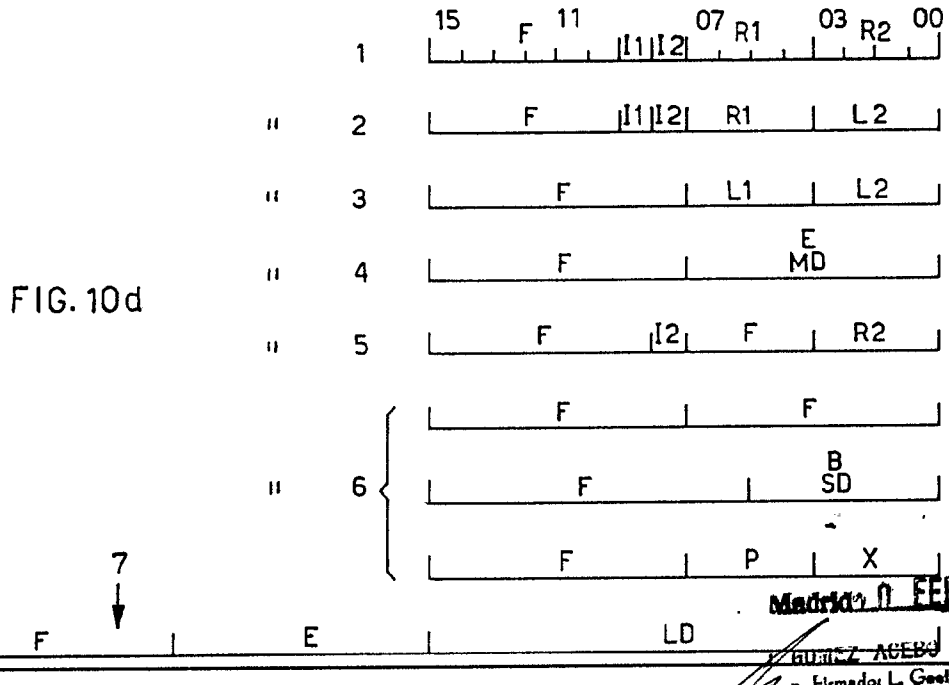


FIG. 10d

Madrid 11 FEB. 1975

GOMEZ ACEBO Y MODET

P. p. Firmado: L. Gesta

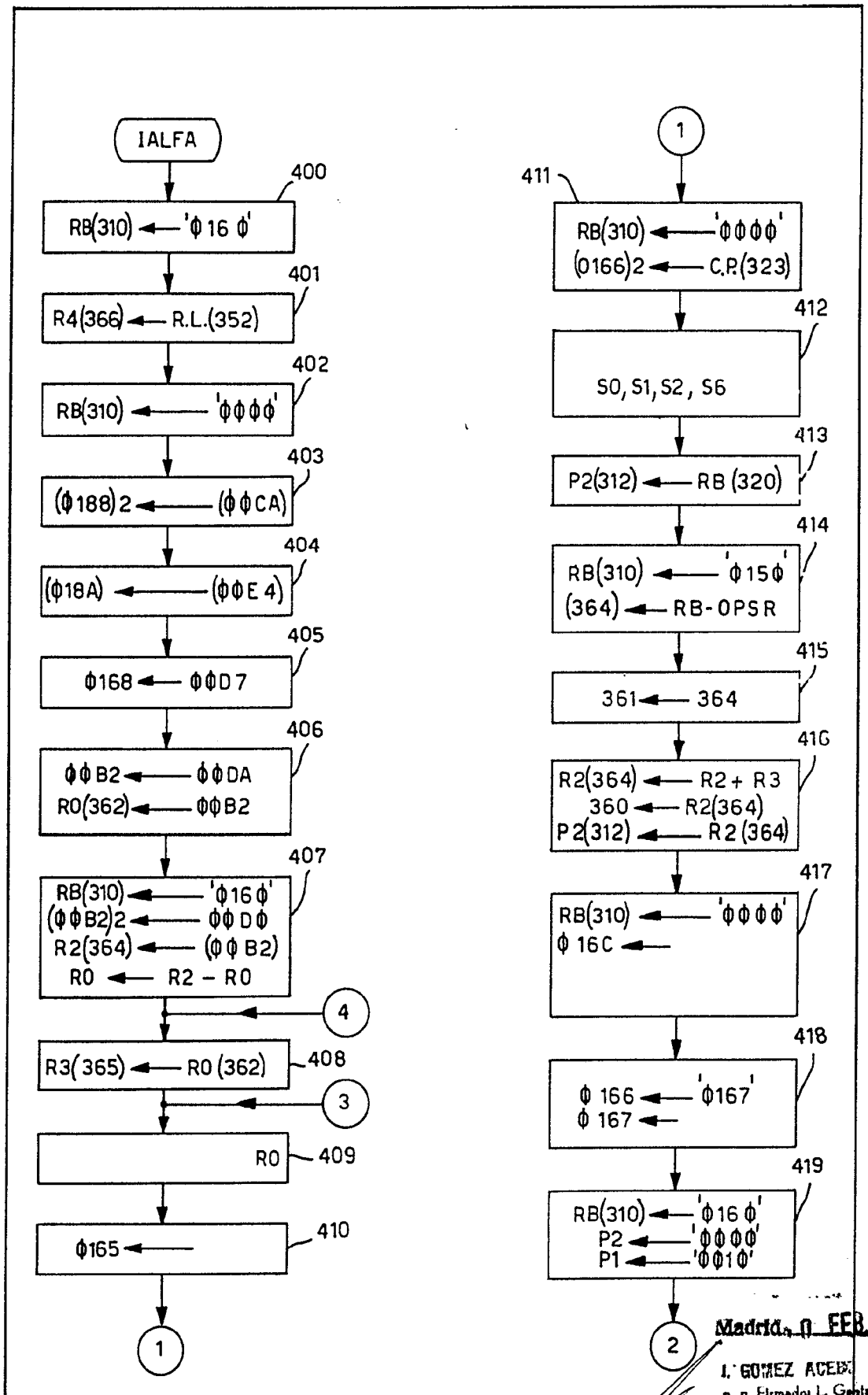


FIG. 11a

FIG. 11b

Madrid, 0 FEB. 1974

J. GOMEZ ACEBO Y ROMERO

P. P. Firmado: L. Gomez Fernández

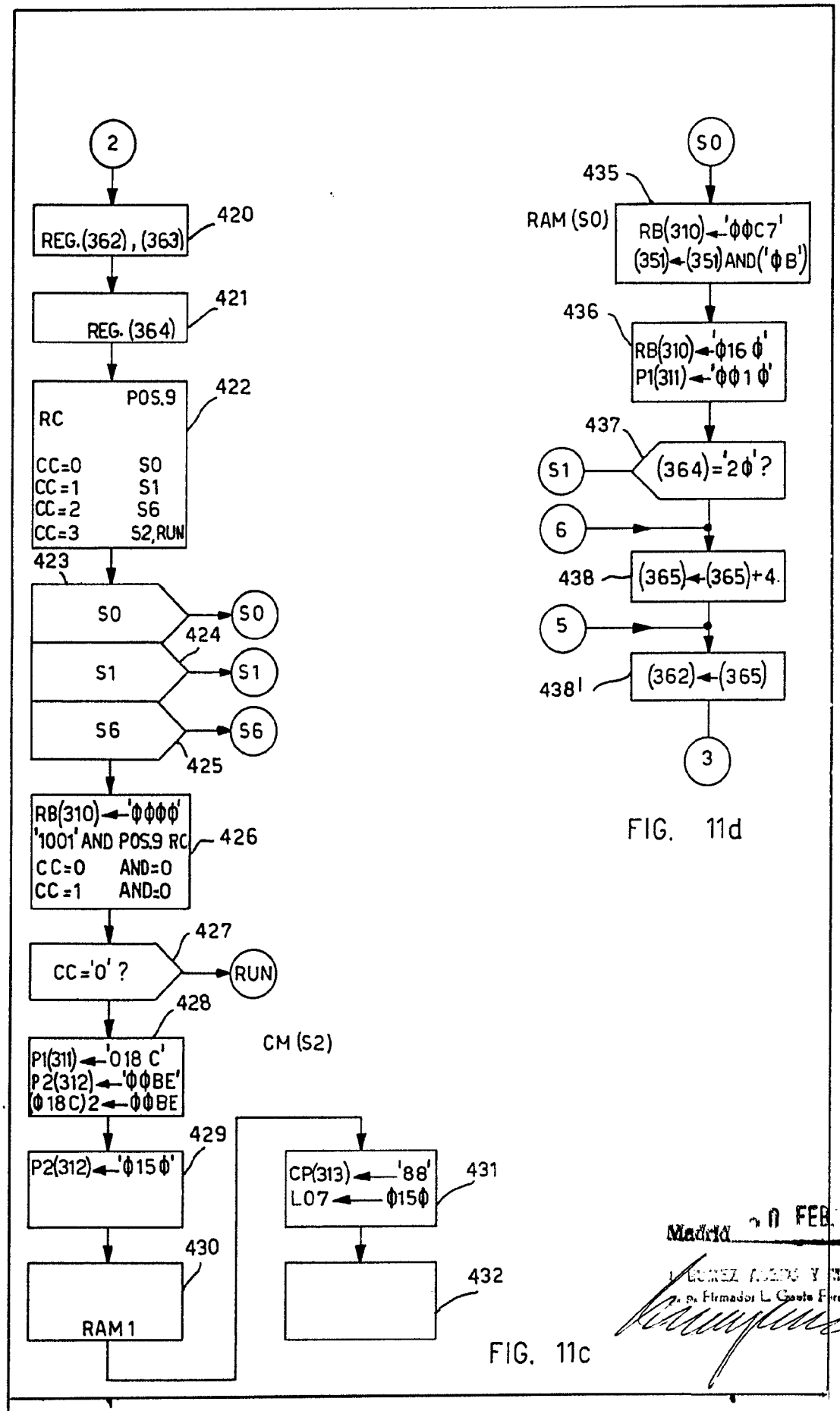


FIG. 11d

FIG. 11c

Madrid 20 FEB. 1975

EL INGENIERO JEFE DE LA OFICINA
P. p. Firmado: L. Gaeta Fernández

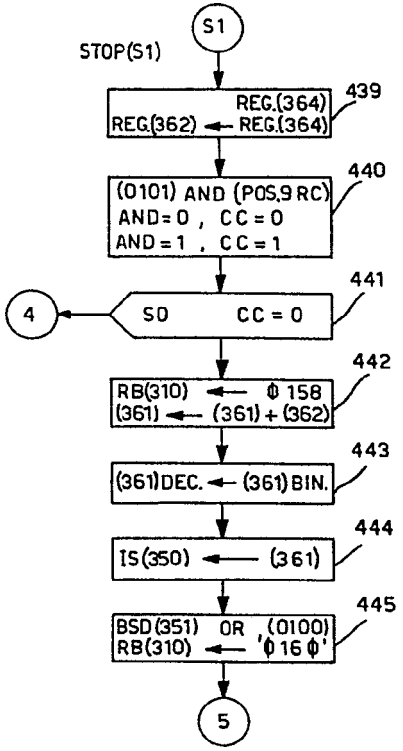


FIG. 11e

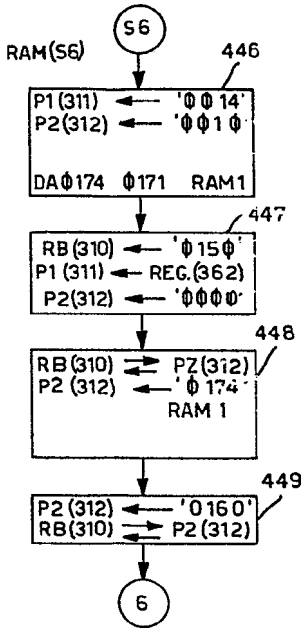


FIG. 11f

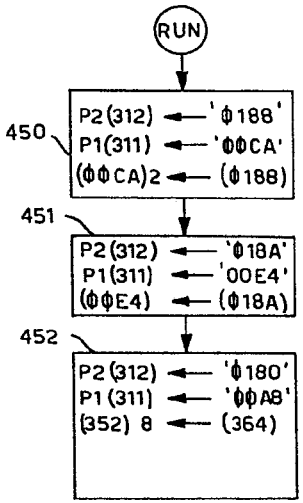


FIG. 11g

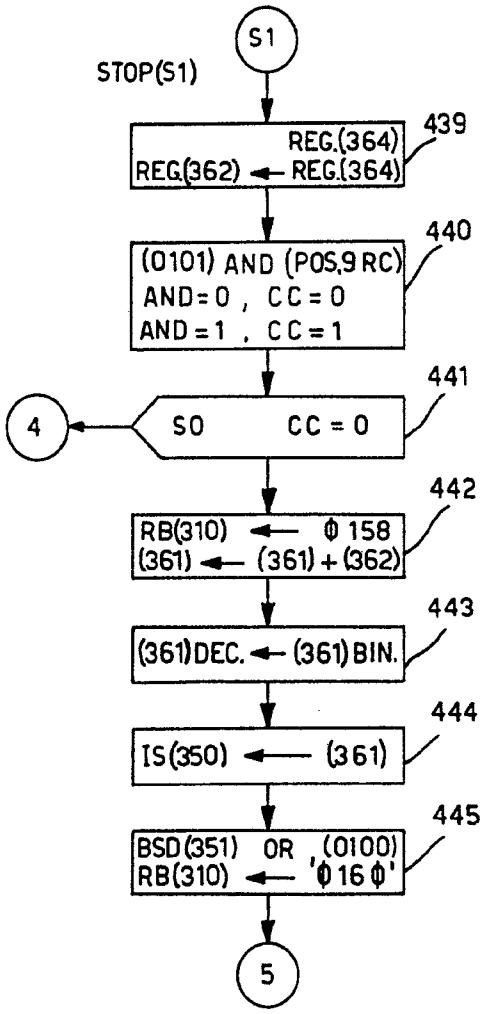


FIG. 11e

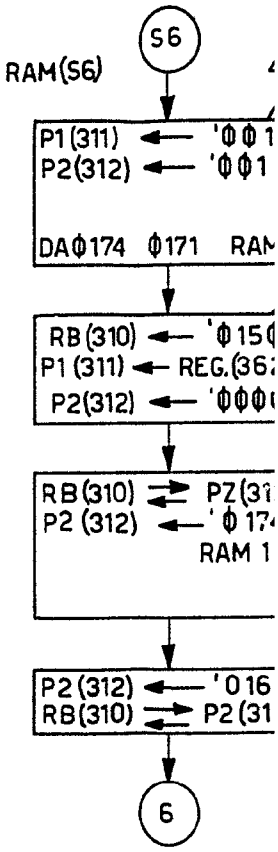


FIG. 11f

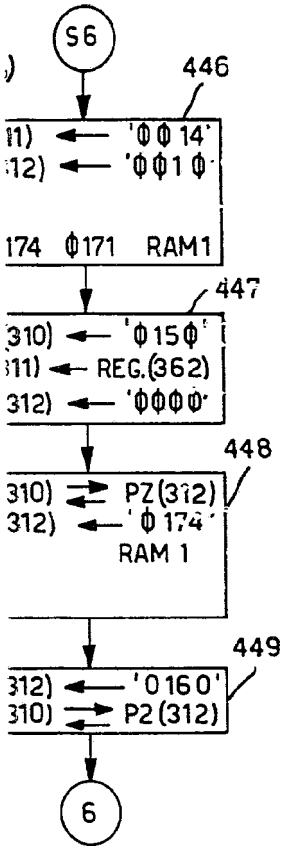


FIG. 11f

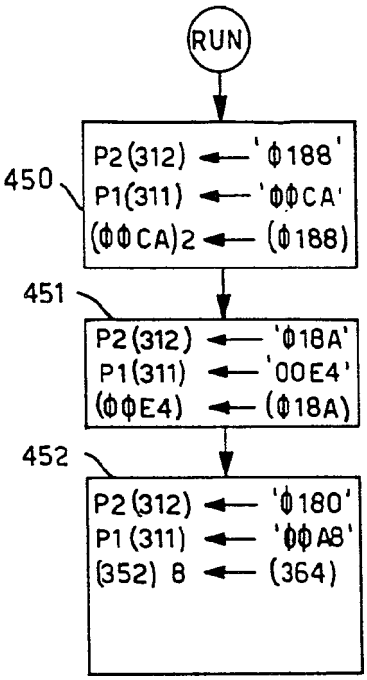


FIG. 11g

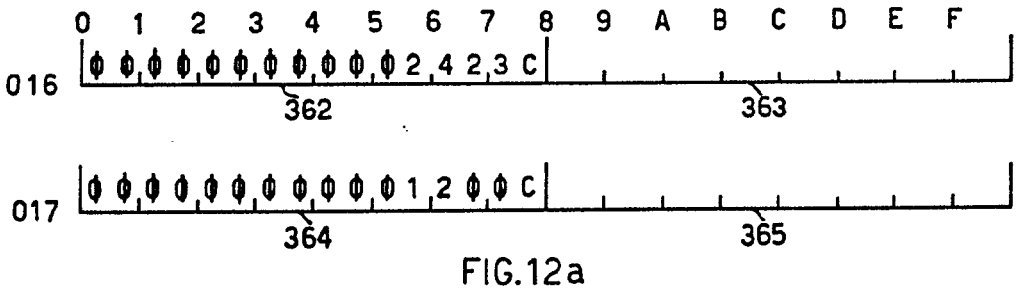


FIG.12a

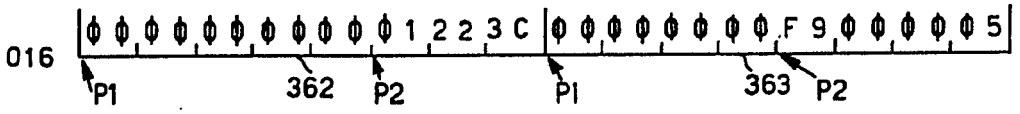


FIG. 12b

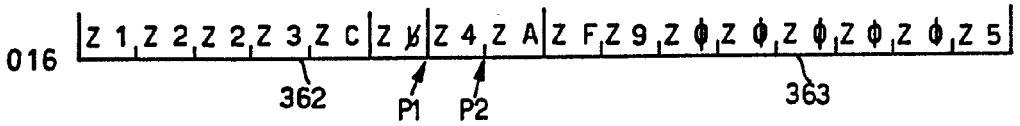


FIG.12c

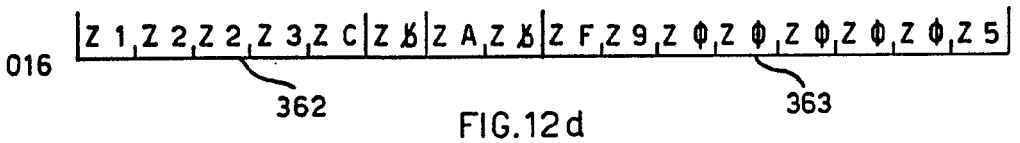


FIG.12d

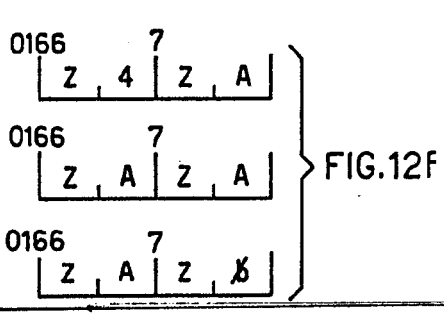


FIG.12F

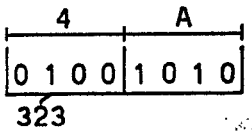


FIG.12e

11 FEB. 1975

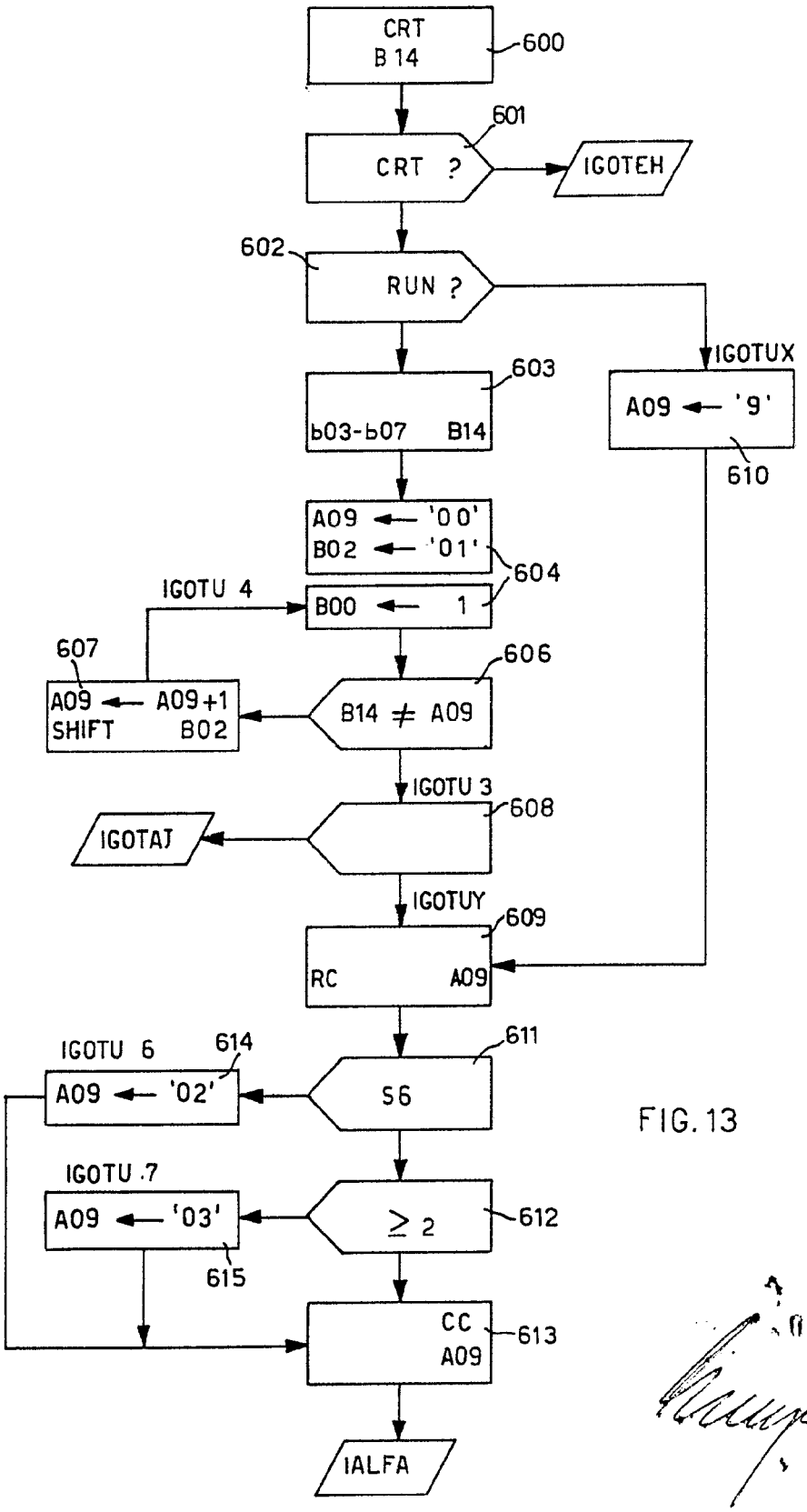


FIG. 13

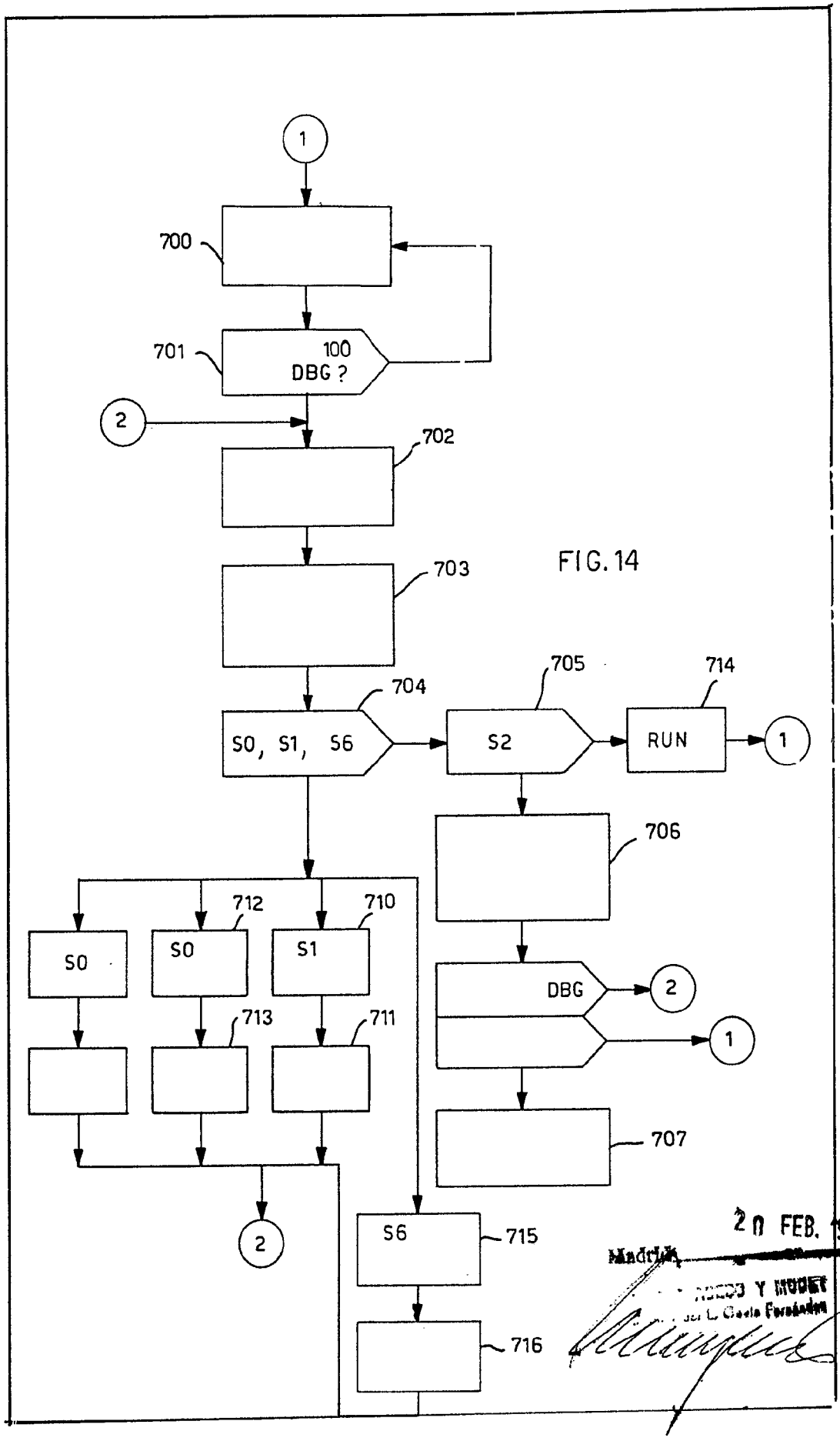


FIG. 14

20 FEB. 1975

Madrid,