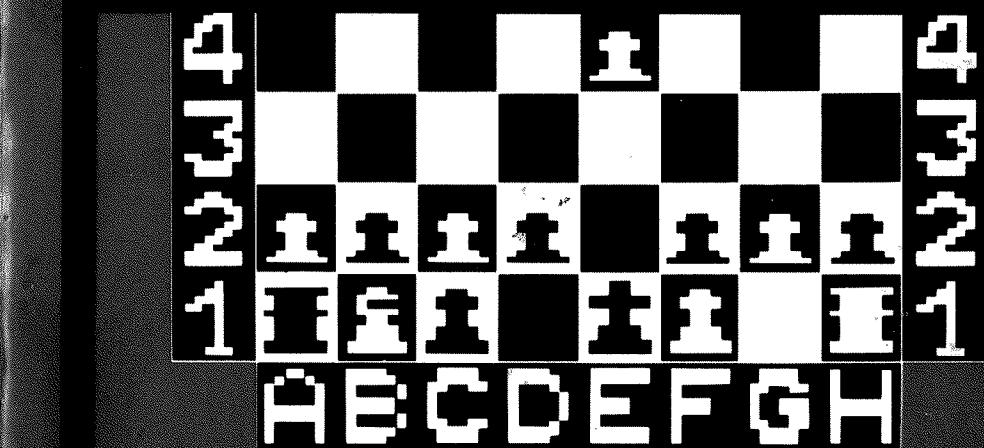


CIRCUITOS PRACTICOS DE ELECTRONICA

Juegos Electrónicos para TV

H. Bernstein



CEAC

Juegos Electrónicos para TV

P
JUEGOS NIMEL
CACIQUEZ Tel. 2-10-30
Calle 102
Nº 102
Col. 2do.
Calle 102
C.P. 711019-C
R.P.C. PNCR-711019-C

CIRCUITOS PRACTICOS DE ELECTRONICA

Juegos Electrónicos para TV

H. Bernstein



Perú, 164 - 08020 Barcelona - España

*en rigurosamente prohibidas, sin la autorización escrita
titulares del «Copyright», bajo las sanciones
previstas en las leyes, la reproducción parcial o total de
obra por cualquier medio o procedimiento, comprendidos
el fotocopia y el tratamiento informático y la distribución
de ejemplares de ella mediante alquiler o préstamo públicos.*

ucción autorizada de la obra
OMPUTERSPIELE
ado en lengua alemana por
n-Verlag, Stuttgart
80 FRECH-VERLAG
N 3-7724-0436-7

EDICIONES CEAC, S.A.
c. 164 - 08020 Barcelona (España)

Edición: Octubre 1991
N: 84-329-6806-4
Dósito Legal: B. 35.095-1991

Impreso por GERSA, Industria Gráfica
nbor del Bruc, 6
70 Sant Joan Despí (Barcelona)

Impreso en España
nted in Spain

CONTENIDO

PROLOGO	7
Juegos electrónicos en el televisor	7
Advertencia	7
1. Fundamentos de los juegos televisivos	9
1.1 El microprocesador 2650	21
1.2 Interface de video programable	81
1.3 Generador de sincronismos	122
1.4 Comutador analógico CMOS con multiplexor	130
1.5 Memoria fija borrable y programable	133
2. Estructura y conexión del computador de televisión	149
2.1 Sumador de video	151
2.2 Microprocesador. Interface de video programable, Generador PAL y memoria de instrucciones para el computa- dor TV	175
2.3 Circuito de audio	215
2.4 Modulador AF	222
2.5 Fuente de alimentación	231
2.6 Interconexión del computador TV	236
3. Programación del computador TV	239
3.1 Emulador para el desarrollo de programas	241
3.2 Programa de juego	246
3.2.1 Tenis individual	246
3.2.2 Tenis de dobles	251
3.2.3 Fútbol	251
3.2.4 Tenis de mesa	254
3.2.5 Balón volea (voleibol)	254

Baloncesto	254
Juegos de obstáculos	255
Observaciones relativas a los juegos	255
Programa para los juegos de pelota	256

Anexo

(Lista de componentes para los juegos televisivos)	285
--	-----

Prólogo

JUEGOS ELECTRONICOS EN EL TELEVISOR

A todo buen aficionado a la electrónica le encantaría organizar sus propios juegos conectando el televisor a un microprocesador, pues éste permite programar libremente su control; es decir, que de ese modo podemos desarrollar nuestros propios juegos televisivos.

En este libro se describen los distintos componentes y circuitos de un computador de televisión de programación libre.

Puesto que se trata de un libro para aficionados, el autor renuncia expresamente a exponer la compleja teoría de los microprocesadores.

En los casos en que aparecen vocablos técnicos especializados, el autor presupone que el lector los conoce.

ADVERTENCIA

Los circuitos descritos en este libro suelen funcionar sin problemas utilizando las placas propuestas por el autor.

Preste suma atención a las tareas de ajuste con el fin de que, por ejemplo, en el juego de fútbol los balones no se salgan de la pantalla.

1. Fundamentos de los juegos televisivos con microprocesador

El microprocesador 2650 es el cerebro de los juegos televisivos. Se emplea en la electrónica industrial, por lo que tal vez haya dificultades para conseguirlo. Además del microprocesador necesitamos otros componentes que se describirán con más detalle en los capítulos correspondientes.

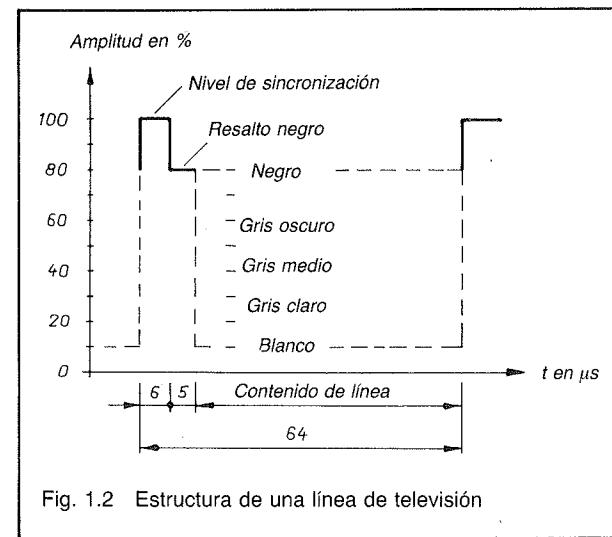
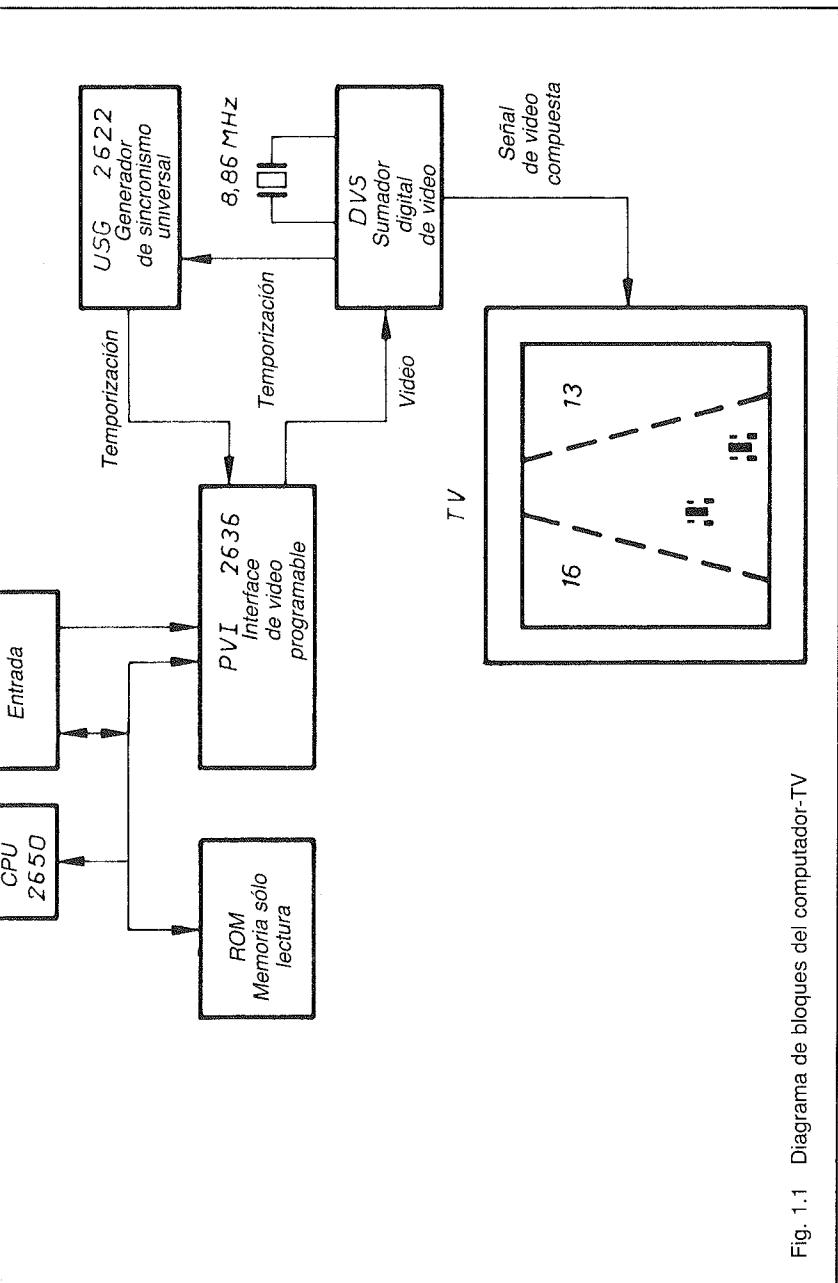
En la figura 1.1 se representa el esquema de conexión de un juego televisivo. El microprocesador 2650 es la unidad central CPU (*Central Processing Unit*: Unidad central de proceso). Mediante los buses (*) de datos, de direcciones y de control, se generan los datos, las direcciones y las funciones de control entre los diferentes bloques. En la ROM (Read Only Memory: memoria de lectura solamente) están almacenadas las instrucciones y los datos para su ejecución. El PVI (interface de video programable) es el elemento de unión entre el control y la parte de AF (alta frecuencia) del juego televisivo. Mediante el USG (generador universal de sincronismo) y el DVS (señal digital de video) se obtiene la interconmutación entre PVI y la señal de salida del juego.

Con el cuarzo de 8,86 MHz no suelen producirse problemas de sincronización y obtenemos una imagen estable. El juego se conecta al televisor mediante un cable coaxial.

El almacenamiento de la imagen del juego se encuentra en la ROM y en la interface de video programable (PVI). El microprocesador obtiene las informaciones relativas al tipo de representación en la pantalla «leyéndolas» en la ROM.

Veamos en detalle la composición de una línea de pantalla. La figura 1.2 representa la longitud de una línea. Primero tenemos una señal de borrado y de sincronización. Esta señal contiene un impulso de sincronización de $5\ \mu s$ aprox. y un resalte negro de igual duración, si se trata de una imagen en blanco y negro. A continuación le sigue el contenido de la

(*) N. T.: Se trata de una vía de transmisión. Dado el uso general de este vocablo inglés en la jerga electrónica, será el que empleemos en lo largo de todas nuestras explicaciones.

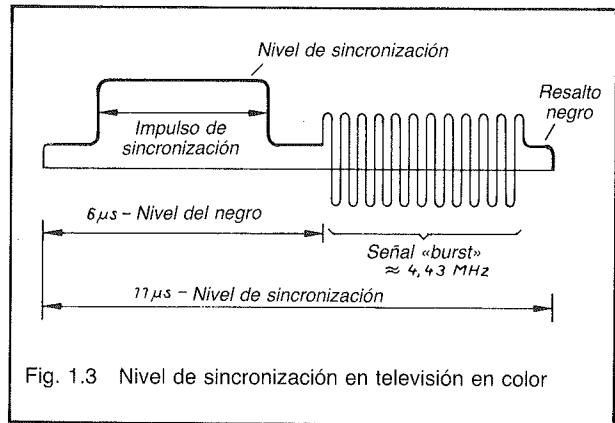


línea con una longitud de unos 52 μs . Después de ese contenido, el televisor recibe un impulso de sincronización con un resalto negro. De este modo, para media imagen se escriben 312,5 líneas.

Si se trata de televisión en color, además del impulso de sincronización, necesitamos una señal de «burst» (*), que se muestra en la figura 1.3. Para esta señal suele necesitarse una frecuencia de 4,43 MHz. En el esquema de la figura 1.1 disponemos exactamente del doble de esa frecuencia. Un divisor de frecuencia genera en el sumador digital de video la frecuencia de «burst» correcta. El mismo sumador produce, además, de 12 a 14 señales de «burst», cada una de ellas de unos 0,2 μs . A continuación viene un breve resalto negro de unos 2 μs .

Una imagen de televisión completa consta de 625 líneas. En ella distinguimos dos medias imágenes, cada una con 312,5 líneas. En el juego con mi-

(*) N. E.: Señal de sincronización utilizada en televisión en color que controla la fase y la frecuencia del oscilador del color en el receptor.

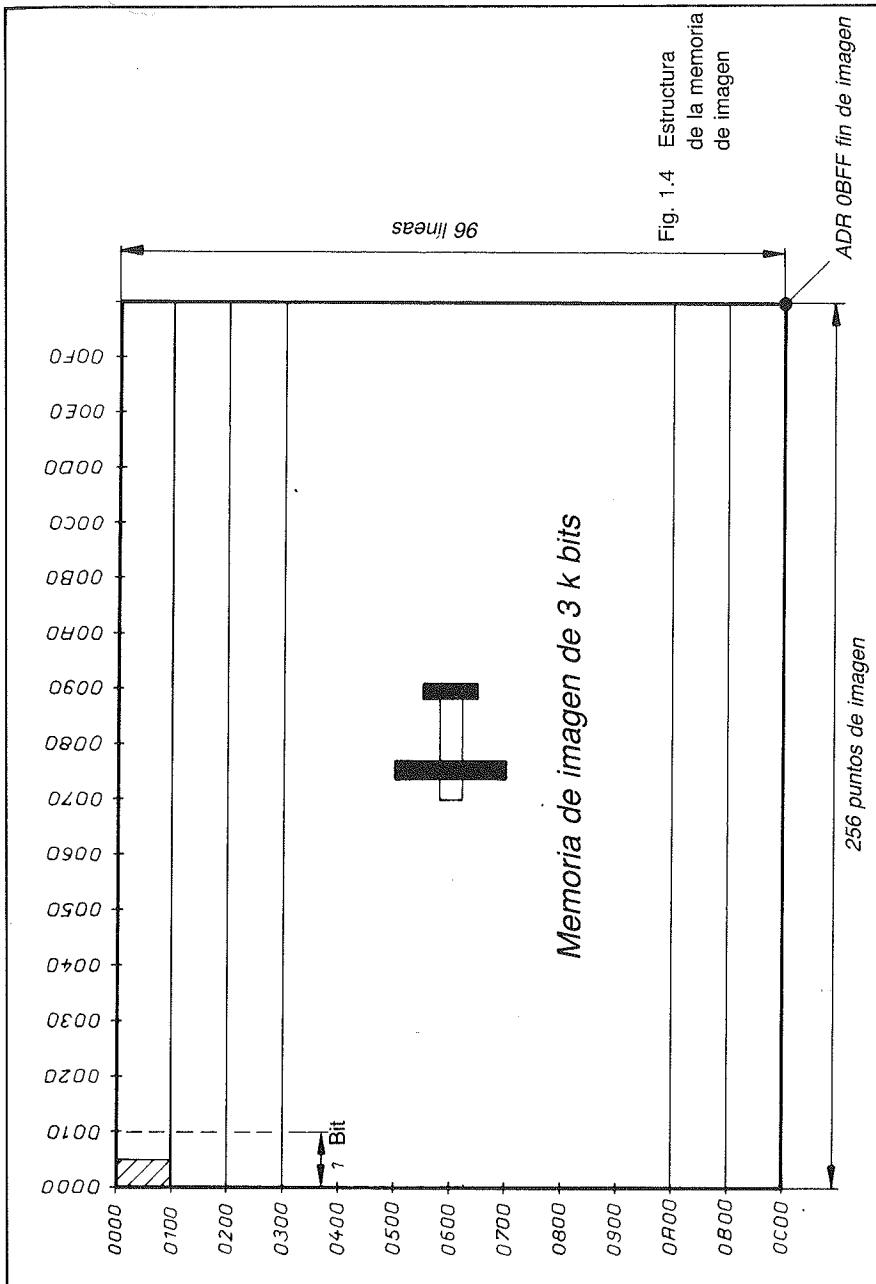


croprocesador escribimos 312 líneas y después sincronizamos. A continuación, el microprocesador representa otra vez en la pantalla el mismo contenido. De esta manera obtenemos en la pantalla una imagen sin centelleo. Una imagen completa dura 20 ms y consta de dos imágenes parciales iguales.

Cada 20 ms, y procedente del interface de video, el microprocesador recibe una señal de interrupción de retorno de imagen del sumador digital de video.

El elemento más importante del circuito de la figura 1.1 es la memoria de imagen. El microprocesador 2650 la dirige continuamente obteniendo así sus instrucciones e informaciones. A lo largo del proceso, obtenemos nuevas informaciones que después, a través de la interface de video programable, se leerán en el sumador digital de video (DVS). Con la interface de video se obtiene, a partir de una estructura de datos de 8 bits en paralelo, una secuencia de datos de 8 bits en serie.

El contenido de la imagen se almacena en la memoria de lectura solamente (ROM). Si descomponemos una imagen en color en distintos puntos, obtenemos aproximadamente unos $1,2 \cdot 10^6$. Esto quiere decir que la memoria de lectura solamente ha de tener la capacidad exigida por un gran volumen de al-



macenamiento. Por ello, la descomposición de imagen según este principio no parece razonable, ya que la memoria necesaria nos saldría bastante cara. Una buena solución intermedia sería aplicar aquí una descomposición de 3×3 puntos. Entonces, podemos emplear una memoria de lectura solamente con 16384 posiciones de almacenamiento. Se trata de una memoria de imagen organizada en 2048 direcciones de 8 bits de salida cada una.

En la figura 1.4 se representa la configuración de nuestra memoria de imagen. El microprocesador comienza en la dirección 0000. La dirección contiene una capacidad de 1 bit y, por tanto, pueden almacenarse 256 informaciones. Después, el microprocesador pasa a la dirección 0001 y se da entrada en paralelo a otro bit en la interfase de video. A continuación, el microprocesador pasa a las direcciones 0002, 0003, 0004, etc., en cada una de las cuales la interfase de video recibe un bit como información en paralelo.

En la memoria de imagen visual, el direccionamiento se realiza de acuerdo con el sistema de numeración hexadecimal. Después de la dirección 000F se produce un desbordamiento y obtenemos la dirección 0010. De esta manera, alineamos consecutivamente 256 puntos de imagen. Si una pantalla tiene una anchura de 52 cm, se deduce que cada punto tendrá una longitud de 2 mm aproximadamente. Con ello abarcamos tres puntos de color del tubo de imagen.

Después de los 256 puntos de imagen de la primera línea, el microprocesador pasa a la segunda línea de la memoria de imagen y lee las direcciones 0100 a 01FF. Después salta a la tercera línea y se introducen en la interfase de video las direcciones 0200 a 02FF. La dirección 02FF es la última información de la línea y después, el microprocesador salta a la 0300. De esta forma se va registrando línea a línea desde la memoria de imagen a la interfase de video.

La última dirección de la memoria es la OBFF. Aquí comienza ya el desbordamiento y el microprocesador retrocede inmediatamente a la dirección 0000 y comienza un nuevo direccionamiento de consulta a la memoria de imagen: Así obtenemos:

$$\begin{array}{r}
 \text{O} \text{ } \text{B} \text{ } \text{F} \text{ } \text{F} \\
 \boxed{\quad} \xrightarrow{\text{F}} \text{F} \cdot 16^0 = 15 \\
 \boxed{\quad} \xrightarrow{\text{F}} \text{F} \cdot 16^1 = 240 \\
 \boxed{\quad} \xrightarrow{\text{B}} \text{B} \cdot 16^2 = 2816 \\
 \hline
 3071 + \text{ADRO} = 3072
 \end{array}$$

Para poder almacenar todos los pasos de programa, necesitamos 3072 direcciones de memoria de 8 bits cada una ($\hat{=}$ 1 Bit). La memoria ROM tendría una capacidad total de 24576 Bits.

Aprovechando la capacidad de almacenamiento total, cada línea de imagen consta de tres líneas de televisión. Por tanto, tenemos:

96 líneas de imagen. 3 líneas de televisión = 288 líneas.

Las 24 líneas restantes de cada imagen parcial (de 312 líneas) no las introduce el microprocesador en la interfase de video, sino que tiene lugar una supresión, de color que es objeto de almacenamiento en una dirección aparte. La imagen de televisión completa está decalada por arriba y por abajo en 24 líneas al visualizar toda la imagen.

En la figura 1.5 se muestra la relación temporal entre los elementos de memoria y las informaciones cromáticas de los diferentes puntos. En el bit 7 se almacena el color negro, es decir, la pantalla se controla para negro si en este bit hay una señal «0» o un nivel L (bajo). Para la imagen de color normal, se necesita aquí una señal «1» o un nivel H (alto). El bit 6 contiene el color verde, el bit 5 el azul y el 4 el rojo. Combinando los tres colores básicos podemos obtener, con los cuatro bits, 16 colores mixtos. El bit más significativo, el bit 7, no lo empleamos para la mezcla de colores, sino para el control del brillo de la pantalla. En el control de la figura 1.1 sincronizamos

Dirección	0000	0001	0002	0003	0004	0005	0006	0007
BIT 7			0	0				
BIT 6			1	0				
BIT 5			0	1				
BIT 4			1	1				
BIT 3								
BIT 2								
BIT 1								
BIT 0								

Fig. 1.5 Disposición de bits de color en la memoria de imagen

de este modo la interfaz de video programable. Mediante los 3 bits de la figura 1.5 podemos obtener en la pantalla los colores siguientes:

Bit n. ^o	Verde	Azul	Rojo	Color
0	0	0	0	Blanco
1	0	0	1	Azul verdoso
2	0	1	0	Amarillo
3	0	1	1	Verde
4	1	0	0	Púrpura
5	1	0	1	Azul
6	1	1	0	Rojo
7	1	1	1	Negro

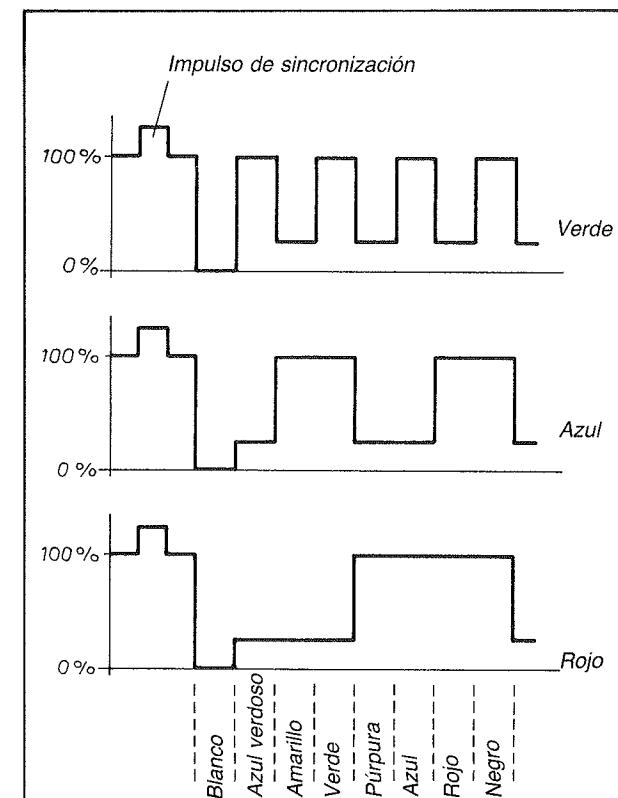


Fig. 1.6 Relación entre las distintas barras de color digitalizadas

La interfaz de video recibe este modelo cromático digitalizado y genera, a partir de dicho patrón, el color de cada punto. Sin embargo, la interfaz de video programable tiene tres entradas de color, pero con ello la programación del microprocesador sería muy compleja y daría lugar a errores de conversión de datos ya que falta una sincronización.

La interfase de video programable tiene una matriz-Y interna. Allí, se genera la señal Y de luminan-

cia a partir de los modelos cromáticos digitales. Si no existiera la matriz-Y, los puntos verdes aparecerían bastante más claros en la pantalla, ya que nuestro ojo es más sensible al color verde.

En la representación de la figura 1.5, aún disponemos de otros cuatro campos para transferirlos a la interface de video. En este caso, transferimos aquí las señales de tono para la componente de audio correspondiente del juego televisivo. Con ello, la componente de audio del programa puede contener hasta 16 tonos diferentes que estableceremos programando la memoria de lectura (ROM). Podremos dar, a través del altavoz del televisor, 256 tonos para cada línea de imagen. Con ello no se imponen límites al fondo acústico de la imagen, pudiendo obtener desde una sucesión melódica de tonos hasta el pavoroso arrastrar de cadenas de tanques.

Cuando consideramos esta breve introducción, deducimos, en principio, que un juego televisivo con microprocesador es una materia algo difícil, puesto que la técnica digital está asociada con la analógica. En tanto que la consulta del microprocesador se desarrolla de forma digital, la interface de video trabaja en salida analógica y ha de modular las señales del microprocesador sobre una portadora que después será recibida por el televisor.

El modo de trabajo del programa en el microprocesador en forma interconectada con la interface de video programable, permite, sin embargo, que el microprocesador disponga de tiempo suficiente para calcular las distintas posiciones de los símbolos del juego, realizar los desplazamientos que correspondan, identificar disparos o colisiones, mostrar visiblemente los destrozos así producidos, contar los impactos o los puntos del juego e interpretar las entradas de los jugadores mediante los potenciómetros correspondientes. Todo ello lo podemos modificar mediante el programa en memoria de lectura (ROM). Para estos sistemas de microprocesadores no existen, apenas, barreras si se cuenta con el software y un programa.

1.1 EL MICROPROCESADOR 2650

El microprocesador 2650 es muy adecuado para los juegos de televisión. El 2650 es un microprocesador estático de 8 bits en un solo chip de 40 patillas («40 pines»). En la figura 1.7 se muestra la disposición de conexiones del 2650.

El microprocesador 2650 se fabrica en un proceso de puertas de silicio de canal-N con implantación iónica. Debido a ello, necesitamos sólo una tensión de alimentación de $U_b = 5$ V, es decir, podemos utilizar el 2650 con la tensión TTL usual de alimentación. La potencia máxima es de unos 600 mW, tal como queda condicionada por ese proceso.

Las entradas y salidas del 2650 tienen una total compatibilidad TTL. Por eso no necesitamos ningún componente de adaptación especial. Tampoco aparecen dificultades de ningún tipo en los trabajos de equilibrado, ya que todo el sistema consta de componentes TTL (a excepción de un componente CMOS para la conexión de los potenciómetros

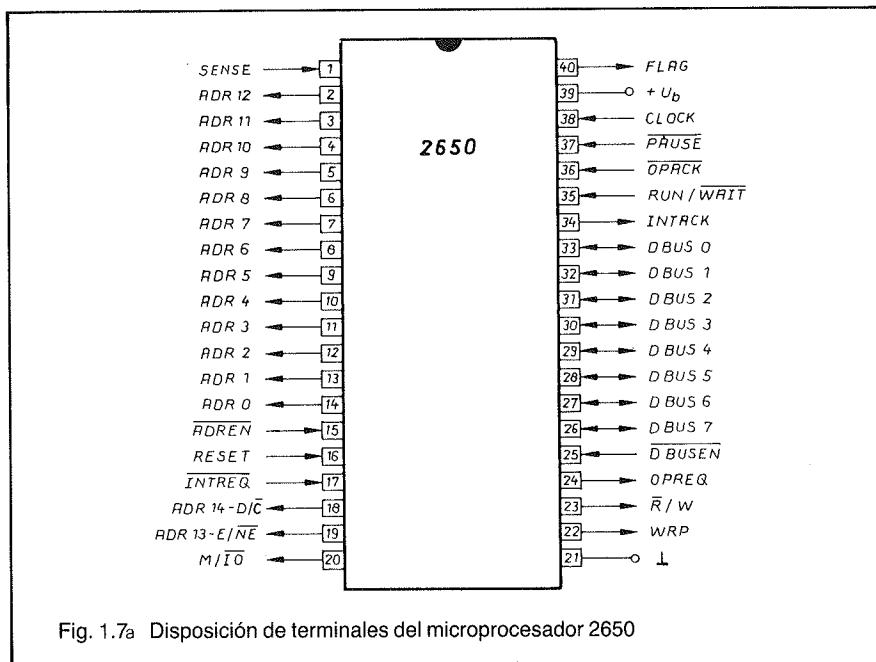


Fig. 1.7a Disposición de terminales del microprocesador 2650

externos de los jugadores y del amplificador operacional en la parte de audio).

La seguridad de servicio del sistema depende del modo de trabajo del microprocesador. Este trabaja en modo estático, esto es, la frecuencia de reloj (o de sincronización) puede estar entre 0 Hz (servicio estático) y 1,75 MHz. En nuestro caso, para el juego televisivo, empleamos como frecuencia de reloj a 1 MHz. El 2650 necesita aproximadamente 1 μ s como tiempo mínimo de ciclo en los procesos de cálculo y direccionamiento.

En la figura 1.7 b se muestra el esquema de conexiones de bloque del microprocesador 2650.

El 2650 recibe sus datos e instrucciones a través del bus de datos de 8 bits. El bus de datos trabaja en forma bidireccional, esto es, trabaja en dos direcciones y con su empleo podemos tanto leer como escribir datos.

El bus de direcciones tiene 15 líneas. De esta forma podemos direccionar $2^{15} \approx 32768$ direcciones. El modo de funcionamiento del bus de direcciones es unidireccional, es decir, con él sólo podemos dar salida a direcciones. La capacidad de almacenamiento de direccionamiento directo es de 32 K-bits.

El 2650 tiene 7 registros direccionables de uso general de 8 bits. Seis de ellos (R1 a R6) están reunidos en un banco de registros. El banco de registros, a su vez, está subdividido en dos partes: el banco 0 contiene R1, R2 y R3 y el banco 1 a los R4, R5 y R6. El registro R0 está fuera del banco.

La memoria de dirección de retorno de subprograma tiene 8 niveles. A esta memoria está conectado el puntero de pila («stack pointer») que puede captar la dirección antes de saltar a un subprograma. El puntero de pila tiene 8 posiciones de memoria para palabras de dirección de 15 bits. El microprocesador 2650, al retornar al programa principal, busca su última dirección en el puntero de pila y reanuda nuevamente allí su programa original.

Mediante la memoria de dirección de retorno de subprograma, junto con el «stack pointer», podemos

estructurar una intercalación de subprogramas como se indica a continuación.

La tabla de la página siguiente muestra las distintas funciones de la disposición de conexiones, designándose con E las entradas y con S las salidas.

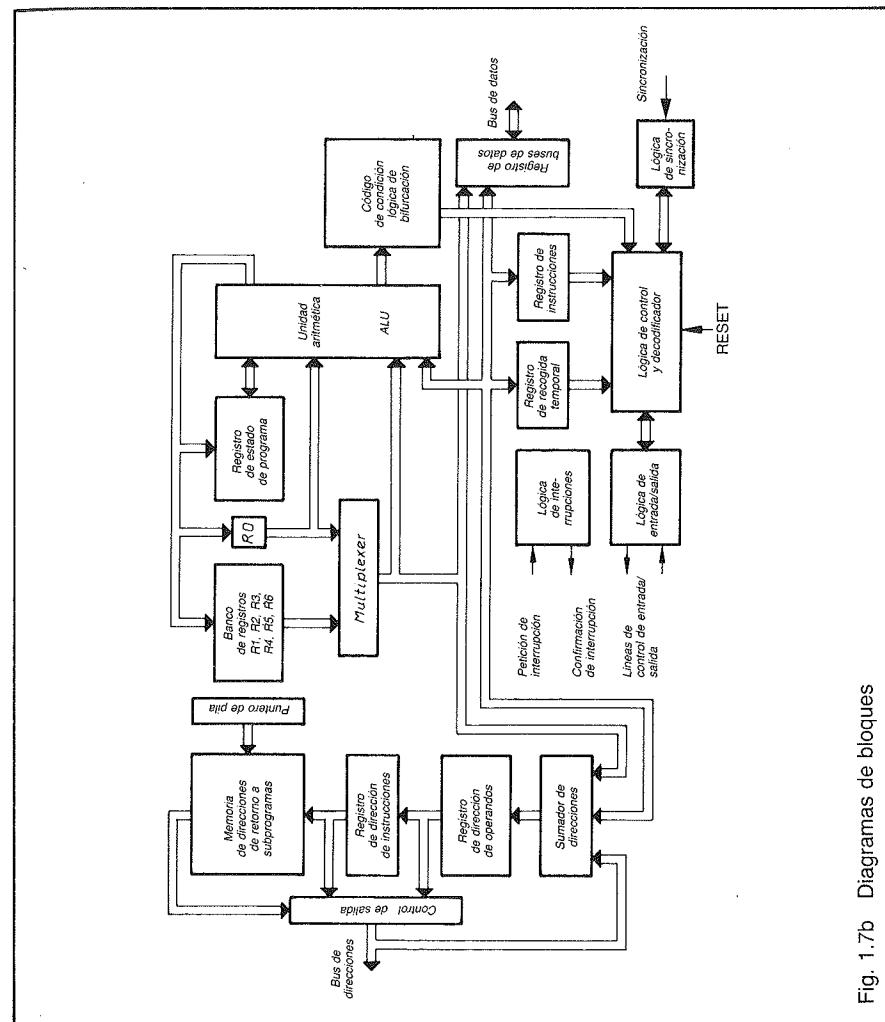


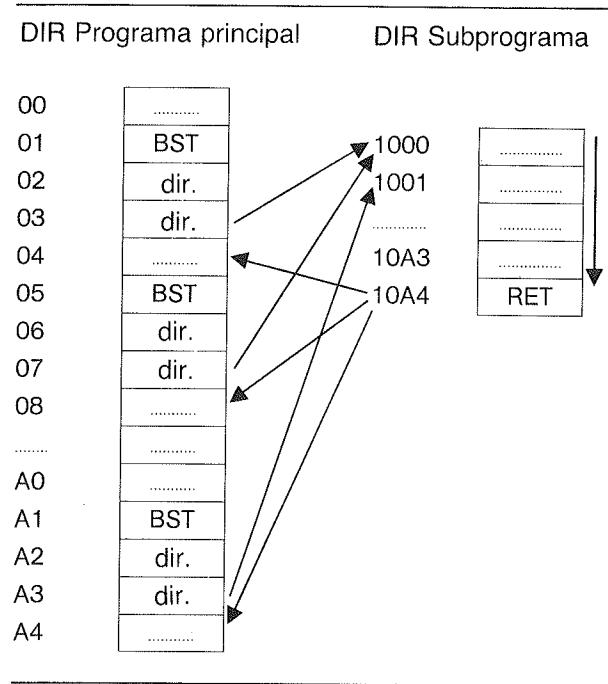
Fig. 1.7b Diagramas de bloques

Símbolo	Designación completa	Patilla	E o S	Nivel	Función
WRP	Impulso de escritura	22	S	H	Esta señal de sincronización del 2650 proporciona, para cada proceso de escritura en un acceso a memoria u operación E/S, un impulso positivo y en procesos de lectura un nivel H (alto) constante.
SENSE_Lectura	Lectura	1	E	H	Entrada de 1 bit, cuyo estado puede consultarse mediante el bit SENSE en el bit de estado de programa (PSU).
ADR0...ADR12_DIRECCIONES	Direcciones	2...14	S	H	Las 13 direcciones de memoria inferiores se emplean además para dirección de E/S.
ADR13-EÑE_DIRECCIONES	Dirección 13 -distinción de instrucciones	19	S	H/L	En instrucciones de E/S, el estado de la dirección de línea ADR13 indica si se trata de instrucciones de 1 bit (sin servicio E/S) o de 2 bits (con servicio E/S).
ADR14-D/C_Datos/Control	Dirección 14 -Datos/Control	18	S	H/L	Con instrucciones de E/S de 1 bit, la dirección ADR14 indica si como puerta de E/S debe emplearse la puerta C (control) o la puerta D (datos).
ADREN_Disponibles	Direcciones disponibles	15	E	L	Control de tres estados para las direcciones ADRO...ADR12. Las salidas son de alta impedancia.
DBUS0...DBUS7_Bus de datos	Bus de datos	33...36	E/S	H	Estas líneas del bus de datos (8-bits) hacen posible el intercambio de datos e instrucciones entre la ROM, la interfaz de video y el 2650.
DBUSEN_Bus de datos disponible	Bus de datos	25	E	L	Control de tres estados para las líneas del bus de datos DBUS0...DBUS7. Las salidas son de alta impedancia.
OPREQ_Solicitud de operación	Solicitud de operación	24	S	H	Indica para todas las unidades funcionales que se encuentran fuera del 2650, que son válidas todas las informaciones del bus de direcciones, del bus de datos y del bus de control (sincronización de bus).
OPACK_Confirmar operación	Confirmar operación	36	S	L	Señal de entrada que indica la terminación de una operación externa. Con ello es posible el servicio asíncrono con unidades funcionales exteriores.
M/0 Entrada/Salida en memoria	Entrada/Salida en memoria	20	S	H/L	Indica si debe realizarse un acceso a memoria o una operación de E/S.

Símbolo	Designación completa	Patilla	E o S	Nivel	Función
RW_Lectura/escritura	Lectura/escritura	23	S	L/H	Indica si debe realizarse un proceso de lectura o escritura.
RESET_Reset	Reset	16	E	H	Entrada con la que se pone a 0000 el registro de dirección de instrucciones y se borra el bit de inhibición de interrupciones del bit de estado de programa.
CLOCK_Reloj	Reloj	38	E	H	Entrada de reloj.
Ub_+V5	+V5	39			Tensión de alimentación.
OV_Masa	Masa	21			Masa
FLAG_Flag (indicador de estado)	Flag (indicador de estado)	40	E	H	Salida de 1 bit que acepta el estado del bit «FLAG» en el bit de estado de programa.
INTREQ_Solicitud de interrupción	Solicitud de interrupción	17	E	L	Con nivel L (bajo), esta entrada indica al 2650 que una unidad funcional externa desea recibir nuevos datos o instrucciones del 2650. El 2650 solo percibe este nivel, después de terminar la ejecución de la instrucción en curso, si el bit de inhibición de interrupciones del bit de estado de programa tiene un nivel L (bajo).
INTACK_Acuse de re-cibo de interrupción	Acuse de re-cibo	34	S	H	Esta salida indica que el 2650 está preparado para aceptar el vector de interrupción (bit con dirección relativa) procedente de la unidad funcional que provoca esa interrupción.
PAUSE_Pausa	Pausa	37	E	L	El 2650, después de terminar la ejecución de la instrucción en curso, se detiene durante el tiempo en que esta señal de entrada se mantiene en nivel-L (bajo).
RUNWAIT_Ejecución/espera	Ejecución/espera	35	S	H/L	Esta salida representa un indicador de estado para el 2650. Durante el proceso de ejecución normal tiene un nivel alto. Si el 2650 se detiene (por una instrucción HALT o por nivel bajo en la entrada PAUSE), la salida RUNWAIT pasa a nivel bajo.

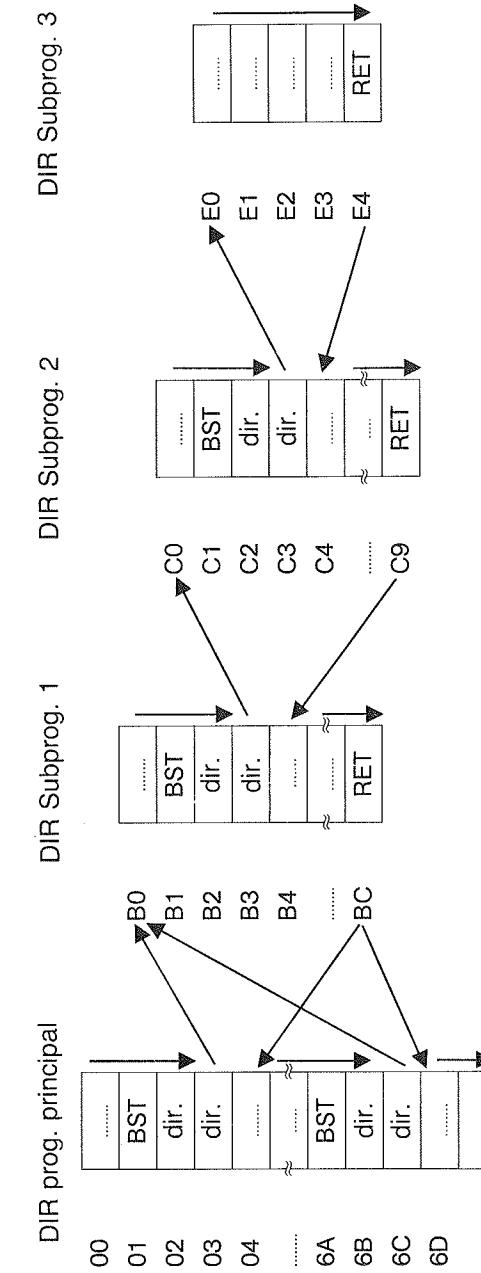
1 Nivel de subprogramas

El programa principal queda separado del subprograma mediante las direcciones. En cada instrucción-BST con su dirección adjunta, el 2650 salta al subprograma. Al saltar, la última dirección se almacena en el «stack pointer» con una palabra de 15 bits. Cuando el microprocesador recibe de su memoria la instrucción «RET», recupera la dirección del «stack pointer» y prosigue en ella su programa. El salto al subprograma puede realizarse tan frecuentemente como se deseé.



3 Niveles de subprogramas

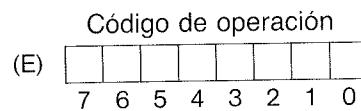
Al principio, el microprocesador se encuentra en su programa principal. Mediante una instrucción salta al subprograma 1 y deposita la última dirección (02 y 03) en el «stack pointer». En el subprograma 1



hay un salto al siguiente subprograma 2. La última dirección (B2 y B3) se lleva al «stack pointer» y comienza el subprograma 2. Un salto posterior lleva al microprocesador al subprograma 3. Cuando éste termina, el microprocesador salta al subprograma 2 y luego al subprograma 1. Sólo después de estos retornos podrá reanudarse el programa principal.

De este modo, en el «stack pointer» pueden introducirse hasta 8 direcciones de subprograma. El retorno sólo es posible mediante las direcciones de subprograma más altas, excepto si empleamos un retorno directo con dirección anexa.

El microprocesador 2650 trabaja con una dotación de 75 instrucciones que pueden tener una estructura de 1, 2 o 3 bits. El primer bit de una instrucción contiene un código de operación para el registro de instrucciones. Obtenemos la representación esquemática de la página anterior.

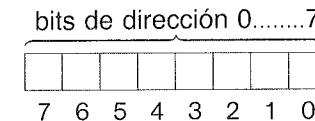
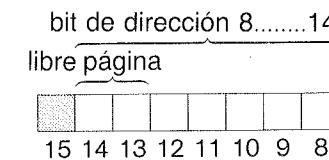


La denominación «E» señala el formato de la instrucción, es decir, trabajamos en «formato E». El primer bit de la instrucción contiene el código de operación para el registro de instrucciones. El código determina la operación a realizar y el tipo de direccionamiento empleado. Los otros dos bits contienen las constantes o las direcciones. Estos bits se adjuntan simplemente al bit de operación.

El código de operación del microprocesador 2650 es casi siempre de 6 bits de longitud, y los dos bits restantes están disponibles para designar registros o para los códigos de condición. Como veremos más adelante en la lista de instrucciones, algunas de ellas también tienen una longitud de 8 bits.

Gracias a la estructura de direcciones, el microprocesador 2650 puede direccionar 32768 posiciones de memoria agrupadas en 4 páginas con 8192

bits cada una. Obtenemos así la representación siguiente:



El bit inferior contiene los 8 bits de dirección de orden más bajo. En el bit superior, los 5 bits de la derecha son los bits de dirección de orden más alto. Con los bits 13 y 14 determinaremos la página. Se deduce, entonces, el direccionamiento siguiente:

ADR14	ADR13	Página	Posiciones de memoria
0	0	0	0000 8191
0	1	1	8192 16383
1	0	2	16384 24575
1	1	3	24576 32767

Mediante el concepto de paginación podemos simplificar el programa del juego en cuanto al direccionamiento.

Las salidas de tres estados existentes en el 2650 colaboran a una mayor sencillez en el desarrollo de este juego. El acceso a memoria así como la entrada/salida de datos, direcciones y variables pueden realizarse en forma síncrona o asíncrona (esto es, controladas mediante una confirmación o acuse de recibo). De este modo, pueden conectarse unidades y circuitos periféricos más lentos. Puede realizarse de una forma sencilla un acceso directo a memoria

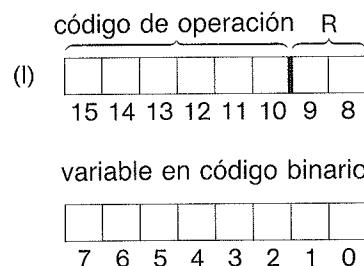
DMA (Direct Memory Access) mientras detenemos brevemente el reloj.

Para el direccionamiento en el ámbito del juego de televisión disponemos de las posibilidades siguientes:

a) Indicación inmediata del operando:

Trabajamos aquí en «formato I». En el primer bit se encuentra el código de operación y el operando está en el segundo bit. Así tenemos una instrucción de 2 bits. Mediante el primer bit, el microprocesador obtiene una instrucción, por ejemplo, «cargar el registro R2», con el valor que tenga el segundo bit. En el segundo bit podemos almacenar una variable o una constante que representa nuestro símbolo en un juego de televisión.

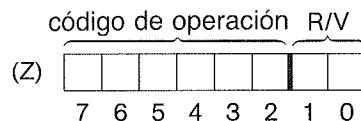
Para el «formato I» tenemos la representación siguiente:



Con la letra «R» designamos un registro.

b) Direccionamiento de registros:

En el direccionamiento de registros empleamos el «formato Z». Este formato corresponde a una instrucción de 1 bit como muestra la representación siguiente:

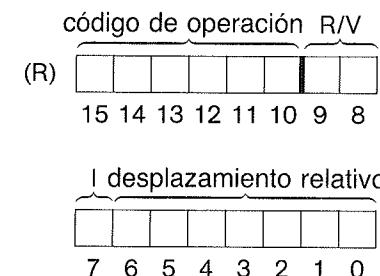


La notación R/V significa siempre que se alude a uno de los registros R y que V es un valor o una condición.

Los dos bits de la derecha se necesitan para indicar el registro, como es el caso de las instrucciones registro-registro o de las instrucciones de E/S de 1 bit o en las palabras de control (en instrucciones de retorno).

c) Direccionamiento relativo:

El direccionamiento relativo trabaja en «formato R», si el bit de direccionamiento indirecto I es igual a 0. Se deduce el siguiente esquema:



Con el código de operación se origina el direccionamiento relativo. Los 7 bits de orden inferior en el segundo bit contienen la diferencia de direcciones, que indica la diferencia entre la posición siguiente a la de la instrucción y la posición en la que se encuentra el operando o la dirección de salto. Esta diferencia puede estar comprendida entre -63 y +63. La representación de esa diferencia se hace en sistema binario y en la forma de complemento a 2.

Cuando el bit indirecto I es igual a 1 tenemos un direccionamiento indirecto relativo. La posición de memoria aludida por la diferencia de direcciones recibe el bit de mayor orden de la dirección del operando o de la dirección de salto (ambas direcciones de 15 bits). El bit de orden inferior se encuentra en la posición de memoria contigua.

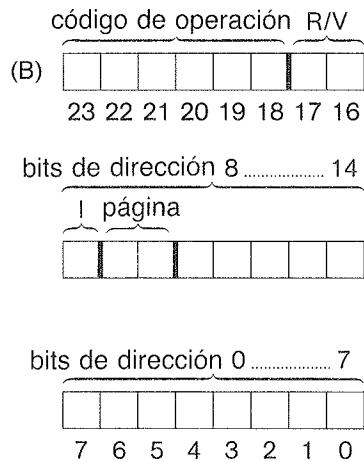
De este modo, en un juego de televisión, pueden calcularse muy rápidamente los desplazamientos.

Una desviación en el programa nos conduce inmediatamente a una nueva dirección en la que el símbolo puede hacerse desplazar una magnitud prefijada por el programa. Así, no hay límites prácticamente para el juego de televisión ni en la programación.

d) Direccionamiento absoluto:

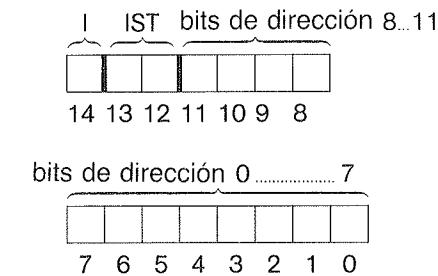
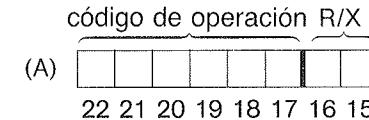
El direccionamiento absoluto trabaja en «formato B» o en «formato A». Con instrucciones de salto empleamos siempre el «formato B» y con las demás, el «formato A».

Obtenemos el «formato B» cuando el bit indirecto $I = 0$. Tenemos la estructura siguiente:



En esta instrucción de 3 bits, el primero de ellos contiene el código de operación, la identificación de registro R y el valor o la condición V. El segundo bit contiene los bits de orden más alto de la dirección, el control de índice y el bit indirecto. El bit 13 y el 14 nos indican la página. El tercer bit tiene los bits de orden más bajo de la dirección.

El otro tipo de direccionamiento absoluto se aplica en las restantes instrucciones del microprocesador 2650. Empleamos para ello el «formato A», cuando el bit indirecto $I = 0$ y el control de índice $IST = 00$. Tenemos el esquema siguiente:



En esta instrucción de 3 bits, el primero contiene el código de operación y la identificación de registro R y la identificación de registro-índice X.

La designación IST significa el control de índice.

e) Direccionamiento indirecto absoluto:

Aquí distinguimos, de nuevo, entre instrucciones de salto y las restantes instrucciones. La posición de memoria direccionada primeramente por la instrucción contiene el bit de orden superior de la dirección de 15 bits del operando o de la dirección de salto. El bit de orden inferior de esa dirección se encuentra en la posición contigua inmediata.

Para las instrucciones de salto emplearemos el «formato B», cuando el bit indirecto $I = 1$. Para las demás instrucciones, emplearemos el «formato A», si el bit indirecto $I = 0$.

f) Direccionamiento absoluto con indexación:

Para este tipo de direccionamiento, volvemos a distinguir entre instrucciones de salto y las demás instrucciones. La dirección del operando, o la de salto, se obtiene sumando a la dirección indicada en la instrucción el contenido del registro de índice. El contenido de este último sólo puede ser positivo y su valor, codificado en binario, puede variar entre 0... 255. Para las instrucciones de salto empleamos

el «formato B», usando los registros 3 y 6 como registros de índice, cuando el bit indirecto $I = 0$. Para las demás instrucciones, empleamos el «formato A» cuando el bit indirecto $I = 0$ y el control de índice $IST = 0$. Para este último tipo de direccionamiento se aplica la tabla siguiente:

control de índice		significado
0	0	no indexado
0	1	indexación con incremento automático de registro de índice
1	0	indexación con decremento automático del registro de índice
1	1	indexación

Mediante el control de indexación podemos decrementar en 1 (restar 1) o incrementar 1 (sumar 1) al contenido del registro.

g) Direccionamiento absoluto e indirecto con indexación:

A la dirección obtenida como en el apartado e), se le suma ahora también el contenido del registro de índice. Obtenemos, así una postindexación. Para las instrucciones de salto mantenemos el «formato B» y los registros R3 y R6 trabajan como registros índice, siendo el bit indirecto $I = 1$.

Para todas las demás instrucciones, empleamos el «formato A» cuando el bit indirecto $I = 1$ y el control de indexación $IST = 00$. Para el control de indexación se aplica la tabla mencionada en f).

Todos los microprocesadores trabajan según el ciclo de Neumann:

1. Buscar instrucción
2. Decodificar instrucción
3. Buscar operandos

4. Ejecutar instrucción
5. Incrementar contador de instrucciones

Vamos a considerar más detalladamente este ciclo de Neumann en el microprocesador 2650.

Al comenzar una operación de cálculo, lo primero es efectuar la reposición del microprocesador a la dirección 0 para que pueda encontrar su primera dirección en el programa.

En el 2650 disponemos de una entrada «RESET». Con un nivel alto, realizamos la reposición del 2650 a la dirección 0. En el juego televisivo, esta entrada se encuentra en la placa frontal.

Al activarse el microprocesador, la señal de reposición borra todos los registros del 2650 y los pone en su posición de comienzo.

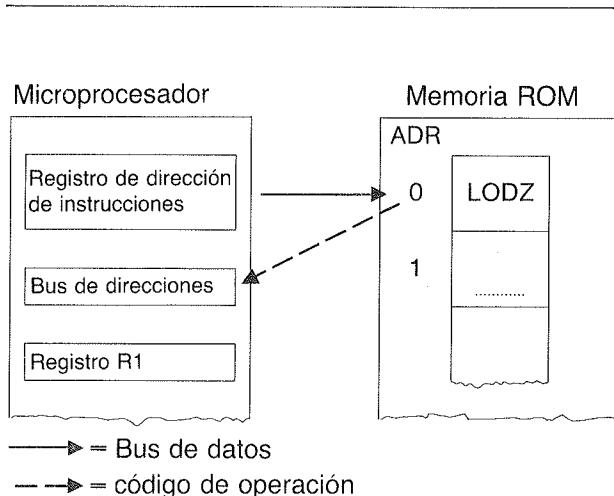
La figura 1.8 nos muestra la salida de la dirección de instrucción 0 (cero). Mediante un nivel alto en la entrada «RESET», el registro de dirección de instrucciones se pone a 0. El control de salida comuta inmediatamente esta dirección al bus de direcciones. El bus de direcciones, por su parte, dirige una posición de memoria en nuestro juego televisivo. Se trata de una ROM con sus instrucciones y variables.

La posición direccionada en la ROM contiene una instrucción.

En la figura 1.9, esta instrucción se encuentra en el registro de instrucciones después de haber pasado por el bus de datos y el registro del bus de datos. En el registro de instrucciones tiene lugar un almacenamiento temporal de la instrucción. Mediante la lógica de decodificación se procede a decodificar la instrucción y así nos atenemos al ciclo de Neumann. La lógica de control está ahora en condiciones de distinguir cuál es la estructura de la instrucción decodificada.

Como ya se ha dicho, existen instrucciones de 1, 2 y 3 bits.

Secuencia funcional para instrucciones de 1 bit:



Estas instrucciones pueden ejecutarse inmediatamente en el microprocesador. Una instrucción de 1 bit característica es la instrucción «LODZ» (Cargar registro cero).

Tenemos el siguiente código de instrucción:

(Z)	Código de operaciones						R/V
	7	6	5	4	3	2	
0	0	0	0	0	0	0	1

Nos encontramos con el «formato Z» con código de operación y la identificación de registro o bien el valor o la condición V.

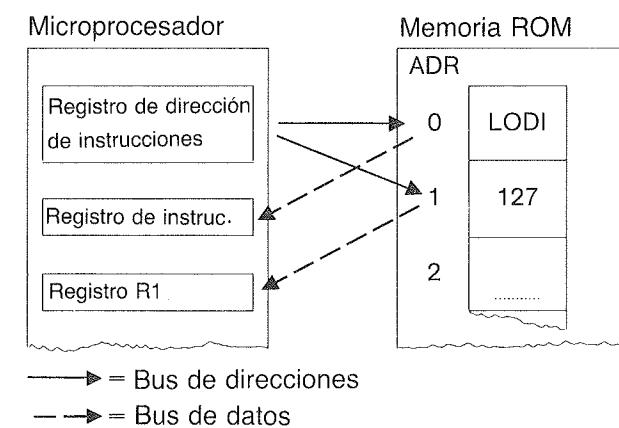
Esta instrucción en el lenguaje máquina del 2650 es «000 000» y en lenguaje de programación ASSEMBLER 2650 es «LODZ». Con ella, el contenido del registro R1 se transfiere al registro R0. Trabajando con las identificaciones de registros son posibles las transformaciones siguientes:

R/V	
0 1	Transferencia del registro R1
1 0	Transferencia del registro R2
1 1	Transferencia del registro R3

Por el contrario, el registro R0 no puede direccionarse mediante esta instrucción.

En una instrucción de 2 bits, el microprocesador necesita, además del código de operación, una cantidad binaria. Esta se encuentra en la ROM en la dirección siguiente.

Secuencia funcional de una instrucción de 2 bits:



De la secuencia funcional del programa, deducimos las diferencias respecto a las instrucciones de 1 bit. La expresión binaria está después de la instrucción. Además, el microprocesador necesita más tiempo para su ejecución.

En una instrucción de 2 bits trabajamos en «formato I», con lo que tenemos un direccionamiento inmediato. Una instrucción típica de esta clase es la «LODI» (cargar registro con el valor inmediato—

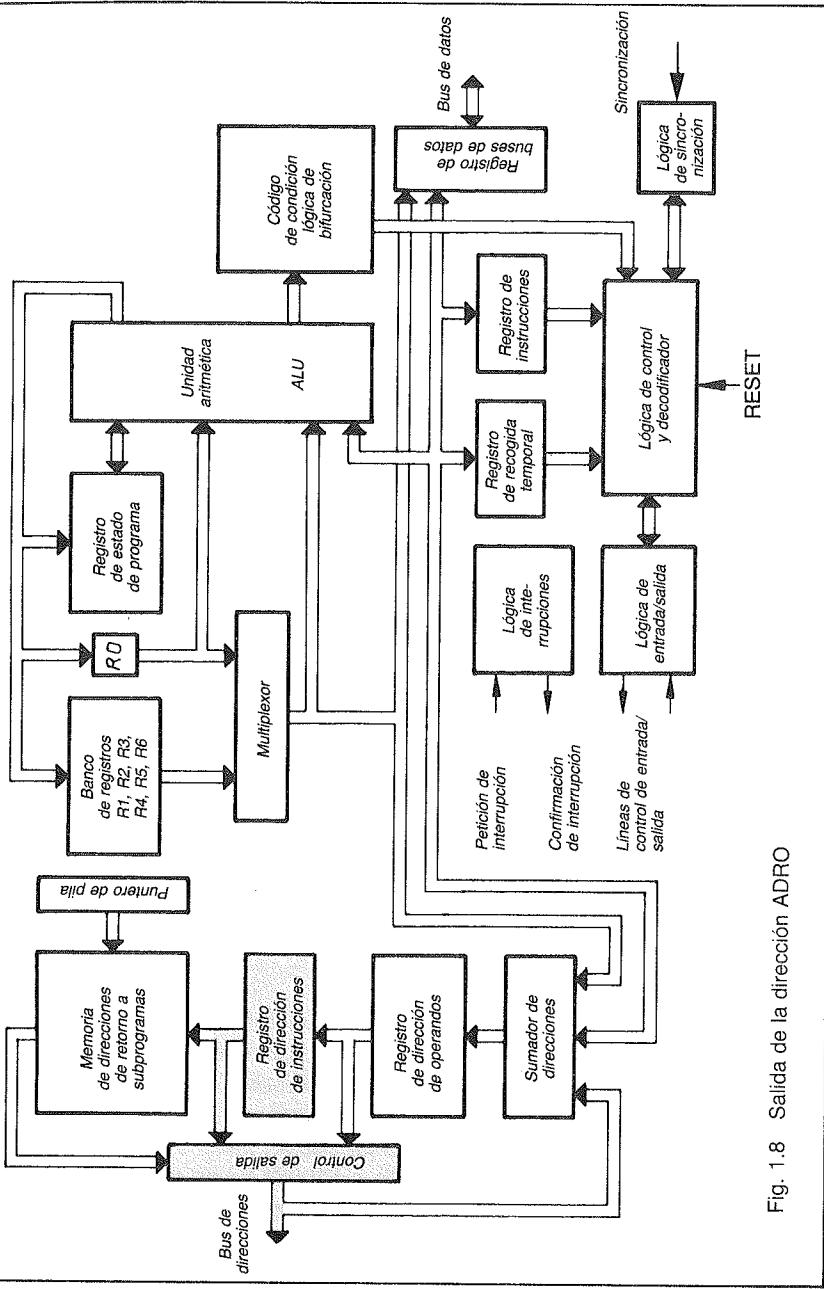


Fig. 1.8 Salida de la dirección ADRO

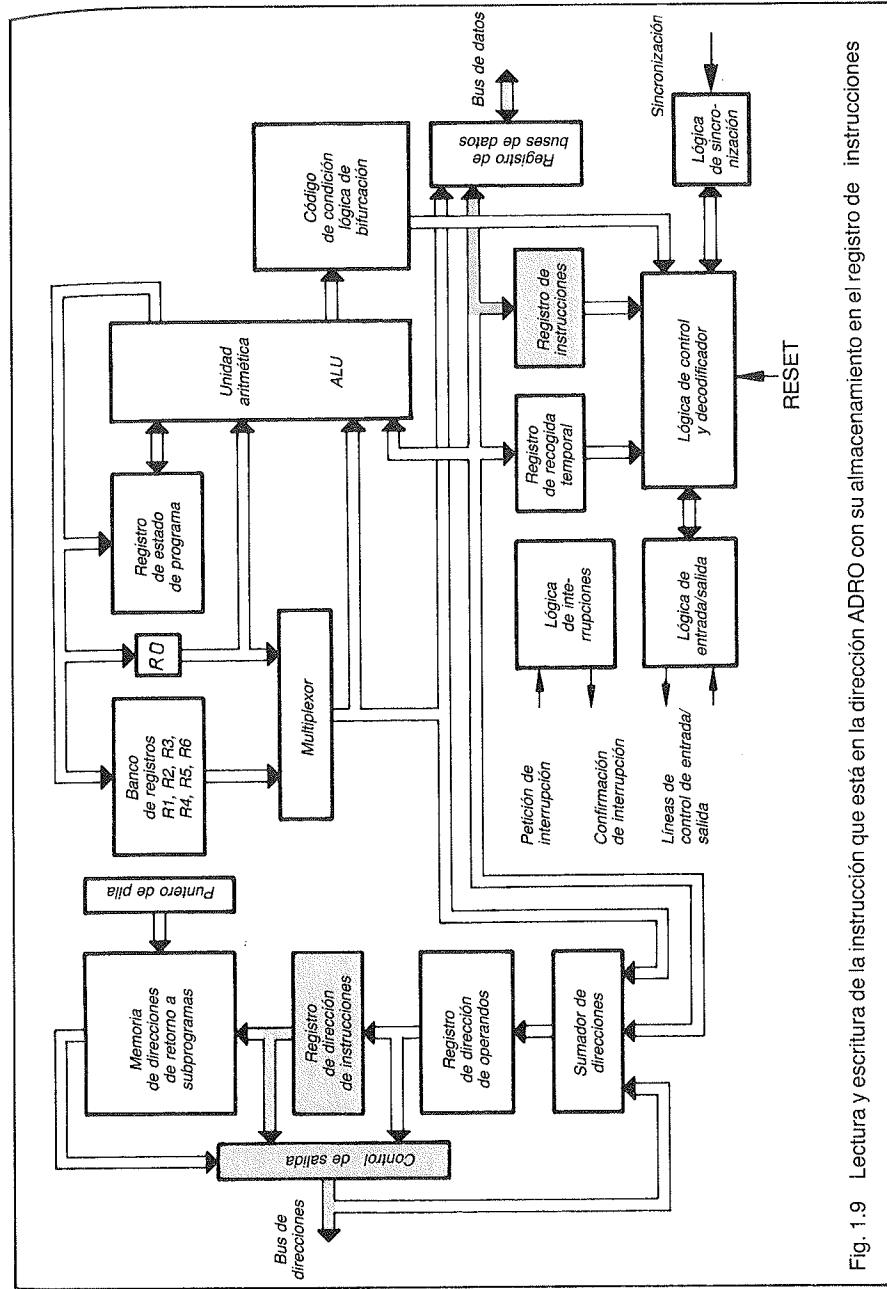


Fig. 1.9 Lectura y escritura de la instrucción que está en la dirección ADRIQ con su almacenamiento en el registro de instrucciones

«Load Register Inmmediate»). tenemos, pues, el siguiente código de operación y su expresión binaria anexa:

código de operación R							
(I) 0 0 0 0 0 1 0 1							
15	14	13	12	11	10	9	8
expresión binaria							
0	1	1	1	1	1	1	1
7	6	5	4	3	2	1	0

Tenemos el «formato I» con el código de operación y la identificación de registro R1. La expresión binaria equivale a la decimal 127.

La instrucción en el lenguaje máquina del 2650 es «000 001» o en lenguaje Assembler-2650 «LDI». Mediante ella, se carga el registro R1 con el valor de la expresión binaria anexa. Cambiando la identificación de registro, podemos cargar los siguientes:

0	0	0	0	0	1	0	0	Cargar registro R0
0	0	0	0	0	1	0	1	Cargar registro R1
0	0	0	0	0	1	1	0	Cargar registro R2
0	0	0	0	0	1	1	1	Cargar registro R3

R

El valor de la magnitud binaria puede estar entre 0 y 255.

Las figuras 1.8, 1.9 y 1.10 explican la interrelación de la operación total. En la figura 1.8, se da salida a la dirección 0 cuando efectuamos la reposición del microprocesador a su estado inicial mediante la entrada «RESET». El registro de dirección de instrucciones apunta a la posición 0 del sistema. Mediante el direccionamiento, se lee, sin destruirla, la

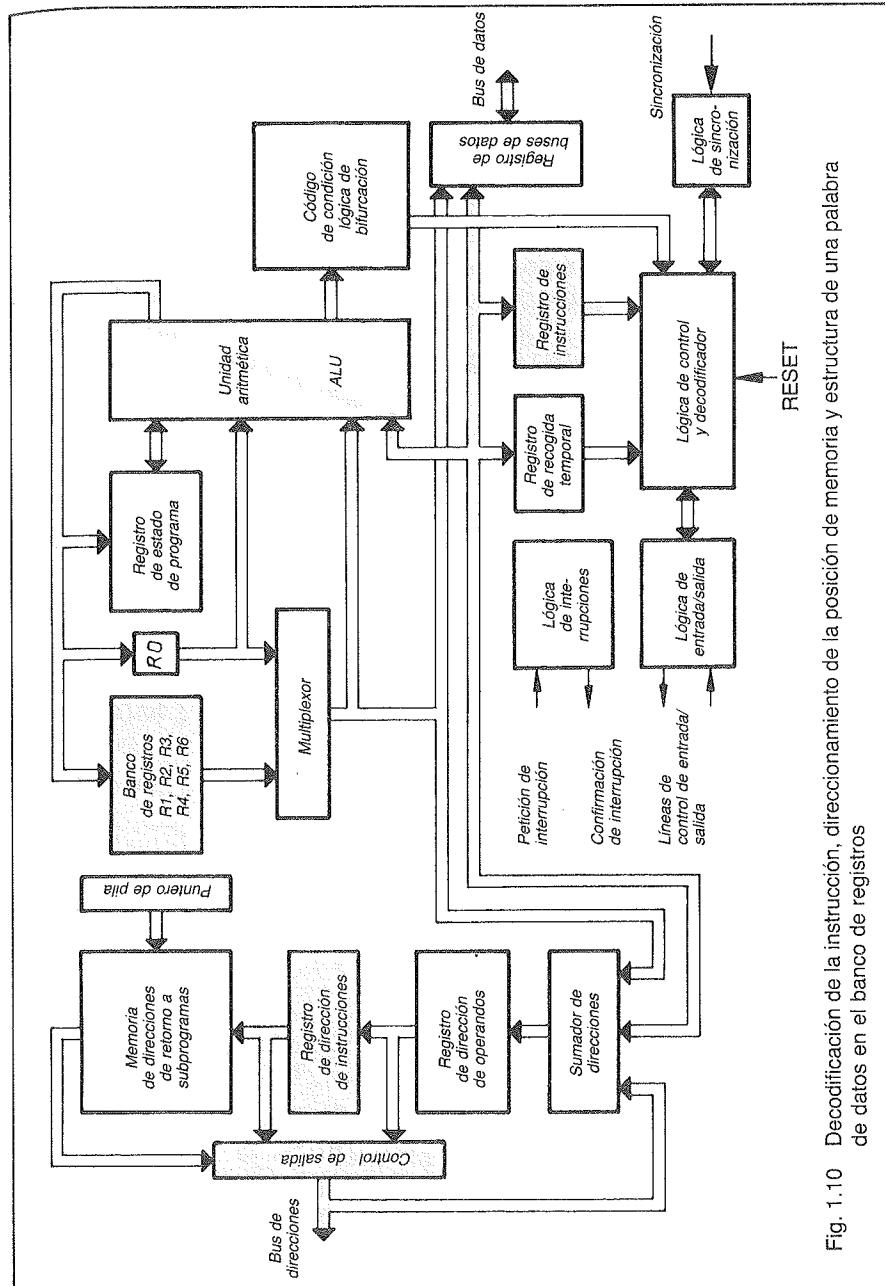
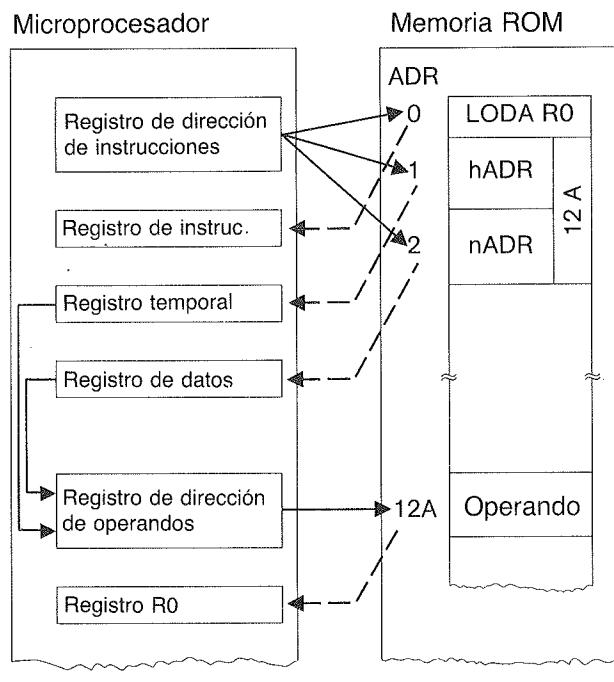


Fig. 1.10 Decodificación de la instrucción, direccionamiento de la posición de memoria y estructura de una palabra de datos en el banco de registros

posición 0 y se transfiere desde la ROM, a través del bus de datos, la instrucción «LODI» al microprocesador. Este proceso se expone en la figura 1.9. La lógica de instrucciones identifica la instrucción decodificada como una instrucción de 2 bits.

La figura 1.10 muestra la interrelación para la búsqueda de la palabra binaria anexa. El sumador de direcciones de la lógica de control, incrementa su estado de direcciones de la ADRO a la ADRI (dir. 0 a dir. 1). Después se transfiere la nueva dirección al registro de dirección de instrucciones y allí se almacena. El direccionamiento de la posición 1 se realiza a través del control de salida y del bus de direccio-



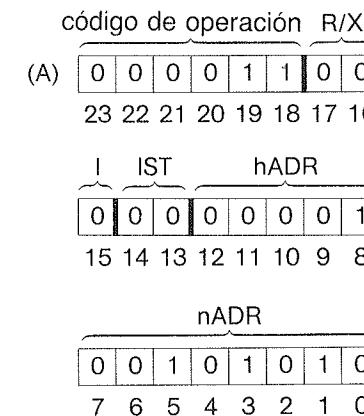
nes. El microprocesador obtiene, entonces, la palabra leída a través del bus de datos. Esta palabra puede atravesar, sin impedimento, la unidad aritmética y se carga en el registro R1. Con esto concluye la instrucción y la lógica de control puede incrementar en +1 el sumador de direcciones. Se termina el ciclo de Neumann para una instrucción de 2 bits y el 2650 obtiene una nueva instrucción.

El microprocesador 2650 dispone de un juego de 75 instrucciones distintas. Las instrucciones de 3 bits pertenecen a las instrucciones especiales. Elegiremos la instrucción «LODA» con una dirección anexa.

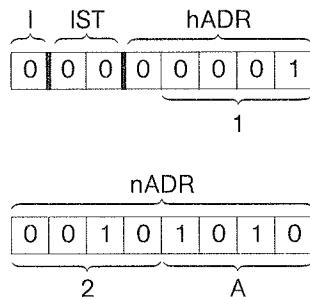
Secuencia funcional para una instrucción de 3 bits en la página anterior.

De la secuencia funcional del programa comprendemos ya las diferencias con las instrucciones de 1 y 2 bits. La dirección de los operandos sigue a la instrucción.

En las instrucciones de 3 bits trabajamos en «formato A» y tenemos la estructura siguiente:



La instrucción «LODA» en lenguaje máquina del 2650 es «000 011» y en Assembler-2650 es «LODA». Con ella se carga el registro R0 con la dirección 12A. Podemos codificar esa dirección del modo siguiente:



Tenemos una combinación del bit indirecto I, el control de indexación IST y las dos partes de dirección hADR y nADR. De la composición de ambas partes de dirección, el microprocesador 2650 puede encontrar las posiciones de memoria de sus operandos.

Consideremos el proceso observando las figuras 1.8, 1.9, 1.11, 1.12, 1.13 y 1.14. En la figura 1.8 mediante un nivel alto en la entrada «RESET» efectuamos la reposición del 2650 y obtenemos la dirección 0 en el bus de direcciones. Así se lee la posición 0 de memoria y se introduce su contenido en el registro de instrucciones. La lógica de decodificación decodifica la instrucción y con ello puede trabajar la lógica de control. Esta identifica que se trata de una instrucción de tres bits. El sumador de direcciones (un simple contador) recibe un impulso e incrementa su estado en una unidad. El estado del contador se transfiere al registro de dirección de instrucciones con lo que tenemos en el bus la dirección 1.

En la figura 1.9 vemos la salida de la dirección de instrucción 0 y la búsqueda del primer bit de instrucción. En la figura 1.11, el bus de direcciones tiene ya un «1» con lo que se leerá la posición de memoria correspondiente. La introducción, o escritura, de la parte de orden más alto de la dirección (hADR) se realiza a través del bus de datos. El valor de hADR se almacena en el registro de recepción.

Después de almacenar hADR en dicho registro, la lógica de control transmite un impulso al sumador

de direcciones. Este incrementa su cuenta de ADR1 a ADR2 y su nuevo estado se transfiere al registro de dirección de instrucciones, según se muestra en la figura 1.12. A través del control de salida, recibimos por el bus de direcciones la dirección ADR2. Así se direcciona la posición 2. Se lee esa posición y su contenido se transfiere al registro de bus de datos. Cuando esa transferencia concluye, la lógica de control emite un impulso.

Mediante un impulso de control, los datos procedentes del registro tampón y del registro de bus de datos se transfieren consecutivamente al registro de dirección de operandos. Con ello en ese registro se encontrará la nueva dirección en la que el microprocesador podrá localizar los operandos. En la figura 1.13 observamos este proceso.

Mediante otro impulso de control, el contenido del registro de dirección de operandos se conmuta al bus de direcciones. En este caso tendremos la dirección 12A. A través del bus de datos se lee el contenido de esa dirección. El registro de bus de datos transmite los datos al microprocesador. La unidad aritmética introduce los datos en el registro R0. En la figura 1.14 se muestra el proceso. Después de ese proceso, el sumador de direcciones recibe un impulso y el registro de dirección de instrucciones de salida a la dirección ADR3. Así finaliza el ciclo de Neumann.

Uno de los componentes más esenciales del microprocesador 2650 es el dispositivo de cálculo o unidad aritmética. En él pueden realizarse además de operaciones aritméticas también operaciones lógicas con los datos. Esa es la razón por la que se le denomina ALU (Unidad Aritmético-lógica: «Arithmetic Logic Unit»). Al realizarse las funciones y operaciones se modifican en el registro de estado de programa PSU (Program Status Register; U = Unit) los bits siguientes: el de acarreo (carry-bit), el de desbordamiento (over-flow-bit), el de acarreo entre dígitos (interdigit-carry-bit) y los bits de código de condición (condition-code-bits).

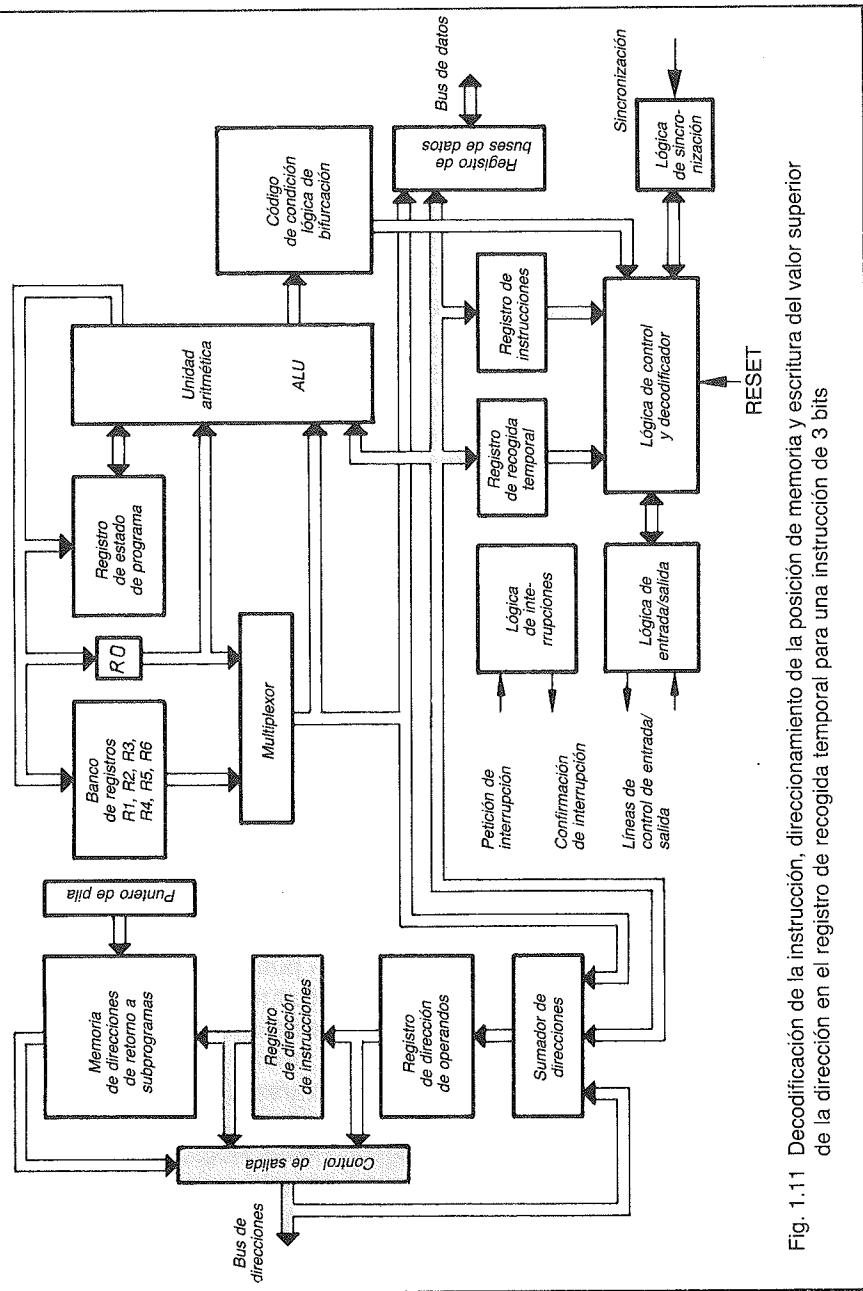


Fig. 1.11 Decodificación de la instrucción, direccionamiento de la posición de memoria y escritura del valor superior de la dirección en el registro de recogida temporal para una instrucción de 3 bits

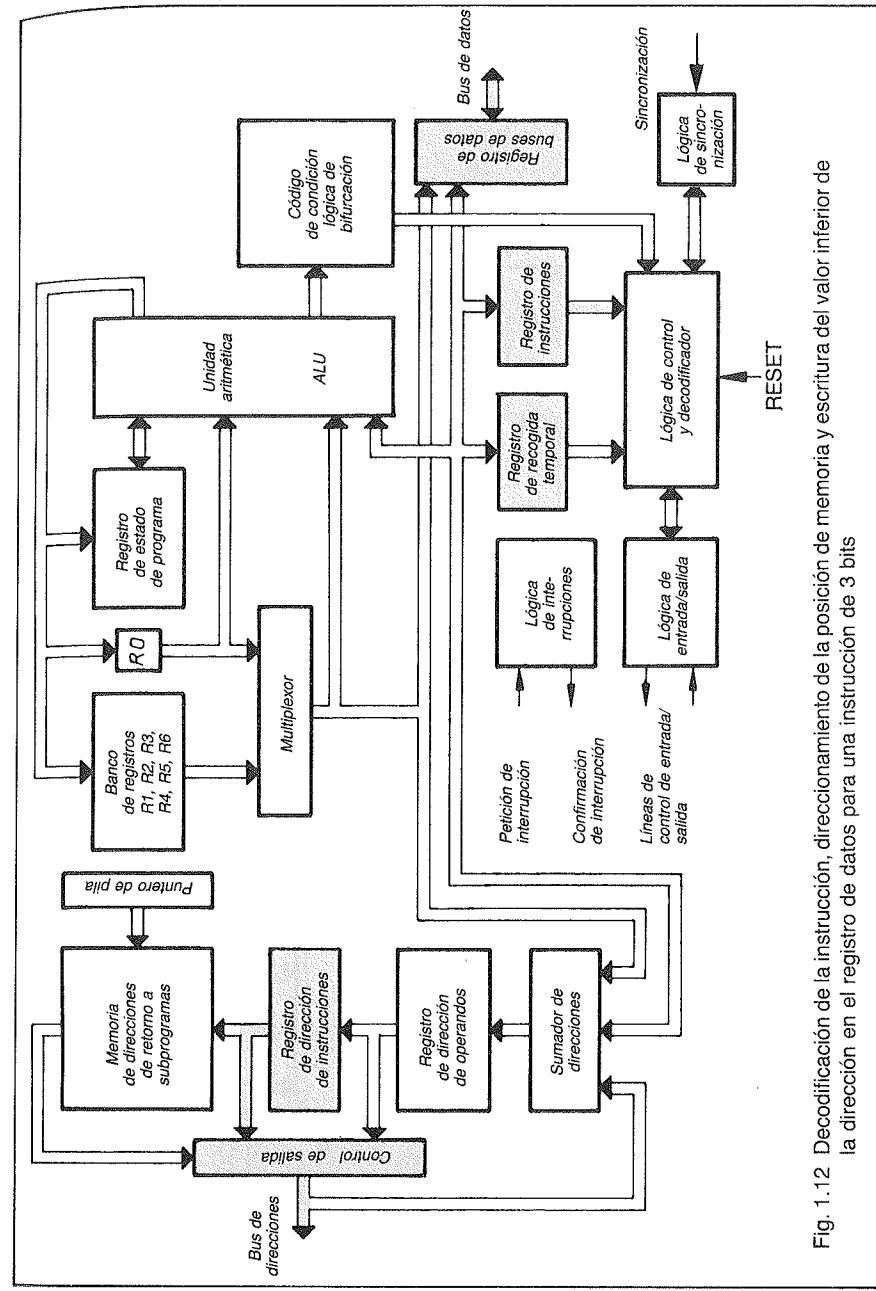


Fig. 1.12 Decodificación de la instrucción, direccionamiento de la posición de memoria y escritura del valor inferior de la dirección en el registro de datos para una instrucción de 3 bits

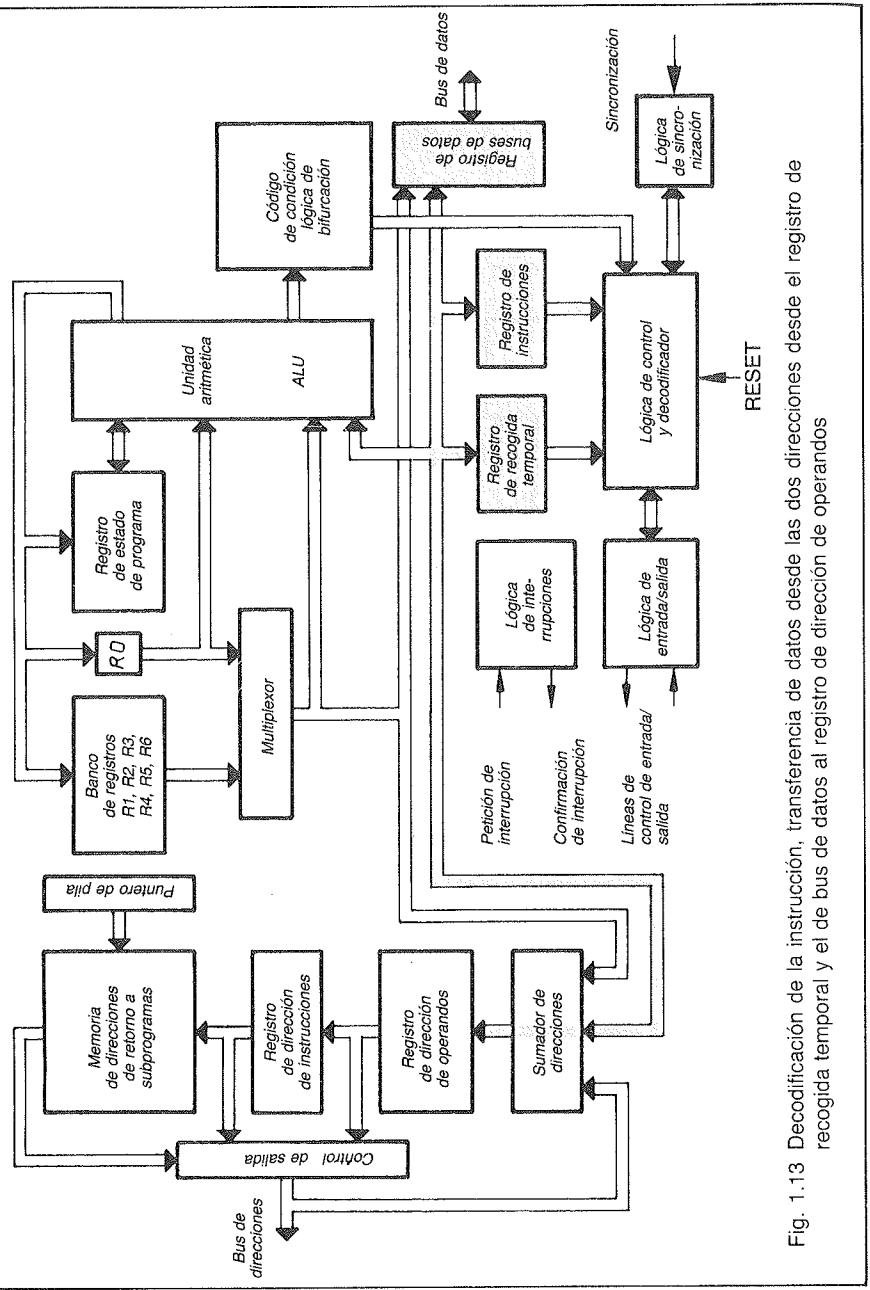


Fig. 1.13 Decodificación de la instrucción, transferencia de datos desde las dos direcciones desde el registro de recogida temporal y el de bus de datos al registro de dirección de operandos

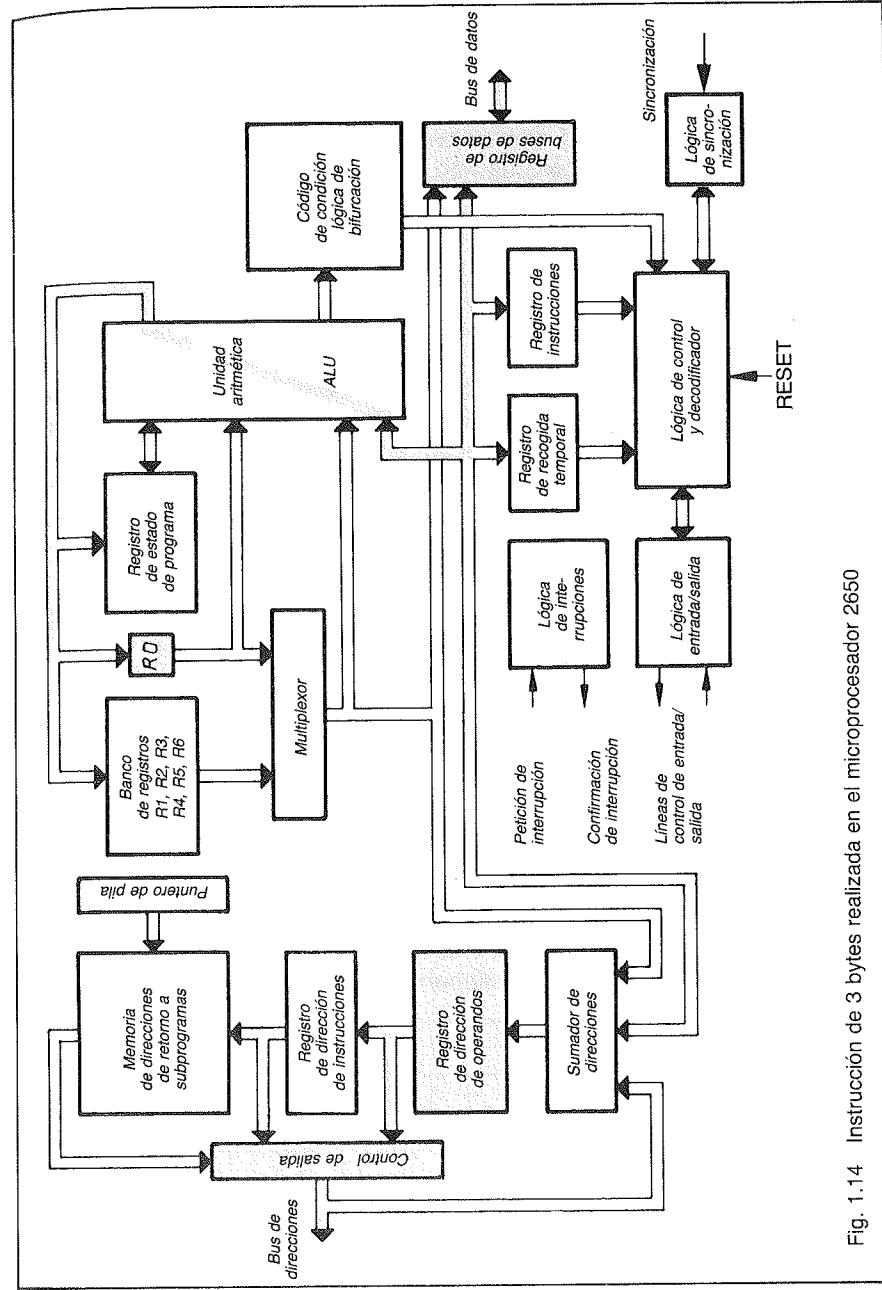


Fig. 1.14 Instrucción de 3 bytes realizada en el microprocesador 2650

Como ya se dijo, el microprocesador 2650 dispone de siete registros internos de uso general. Seis de ellos se encuentran en dos bancos de registros. El bit de selección de registro RS (Register-Select Bit), que se encuentra en el bit inferior de estado de programa PSL, determina a cuál de los bancos de registros se ha de acceder. El registro R0, con objeto de simplificar las instrucciones registro-registro, no se encuentra en esos bancos. De esa forma, el registro R0 puede seleccionarse siempre, además de cualquiera de los bancos en acción en cada momento.

El registro de estado de programa, PSU, contiene los bits PSU y PSL. El bit PSU (Program Status Upper) es el bit de mayor rango del PSU. El bit PSL es el de menor rango o bit inferior (Program Status Lower). Conjuntamente constituyen dos bits, de los que sólo se emplean 14 de los 16 bits. En los 14 bits están contenidas todas las informaciones de estado y de control. Con instrucciones de estado de programa especiales puede cargarse el registro R0 con el contenido de ambos bits, y viceversa, es decir, cargar los dos bits con el contenido del registro R0. También con ellas y contando con una máscara (palabra binaria en una instrucción de 2 bits) pueden comprobarse, activarse o borrarse los distintos bits de la palabra de estado. Esto es una problemática que se resuelve mediante el software del usuario, tal como haremos en el programa para el juego de televisión.

Los distintos bits de los dos bits de estado de programa tienen el significado siguiente:

PSU

7	6	5	4	3	2	1	0
S	F	II	frei		SP2	SP1	SP0

S $\hat{=}$ Sense (salida de lectura)
F $\hat{=}$ Flag (señalizador)
II $\hat{=}$ Inhibir Interrupciones

SP2 $\hat{=}$ Stack Pointer 2 (puntero de pila 2)
SP1 $\hat{=}$ Stack Pointer 1 (puntero de pila 1)
SP0 $\hat{=}$ Stack Pointer 0 (puntero de pila 0)

Para el bit superior de estado de programa tenemos la tabla siguiente:

Bit-PSU	Abreviatura	*	Significado
0, 1, 2	SP		Número de la posición de memoria de dirección de retorno de subprograma que contiene la dirección de retorno que se ha almacenado la última. El «stack pointer» trabaja según el principio de almacenamiento LIFO (Last in First out). Es decir, la última dirección escrita en él será la primera que se lea después. En total se tienen 8 posibilidades de almacenamiento.
3, 4			Estos bits están siempre a «0».
5	II	X	Inhibición de interrupciones que suprimirá cualquier solicitud posterior de una interrupción. El microprocesador 2650 queda bloqueado ante posteriores interrupciones mediante una señal «1» (nivel alto).
6	F		Salida de «flag» en el pin 40. Si hay un «flag», la salida recibirá un nivel alto, en caso contrario un nivel bajo.
7	S		Entrada Sense en el pin 1. Si hay un nivel alto puede consultarse el bit Sense; en caso contrario, este último estará bloqueado.

* Donde hay una X significa que estos problemas se resuelven mediante software.

PSL

7	6	5	4	3	2	1	0
CC1	CC0	IDC	RS	WC	OVF	COM	C

CC1 ≡ Código de condición 1

CC0 ≡ Código de condición 0

IDC ≡ Acarreo entre dígitos (Interdigit carry)

RS ≡ Selección de banco de registros (Register Bank Select)

WC ≡ Con/sin acarreo (With/Without Carry)

OVF ≡ Desbordamiento (Overflow)

COM ≡ Comparación lógico/aritmética (Logical/arithmetic Compare)

C ≡ Acarreo/Préstamo (Carry/Borrow)

Para el bit inferior de estado de programa tenemos la tabla siguiente:

Bit-PSL	Abreviatura	*	Significado
0	C		Si al efectuar una operación de cálculo se produce un bit que no cabe a la izquierda (acarreo), este bit (el de mayor rango) se almacena y en la salida «carry» aparece una señal (nivel alto).
1	COM	X	Comparación. Mediante este bit, el 2650 establece si, en una operación de comparación, los datos han de ser interpretados como cantidades binarias positivas de 8 bits (comparación lógica, COM = 1) o como cantidades binarias en la forma de complemento a 2 (comparación aritmética, COM = 0).

* Se sustituirá por software

Bit-PSL	Abreviatura	*	Significado
2	OVF		Desbordamiento. El 2650 establece, mediante este bit, si en una suma o resta de cantidades binarias en la forma de complemento a 2 se ha desbordado el margen de cantidades representables en 8 bits.
3	WC	X	Con acarreo. Aquí, el 2650 establece si en operaciones aritméticas o de desplazamiento de cantidad, ha de tenerse en cuenta el bit de acarreo (bit 0 en PSL). Si así es, entonces WC = 1, en caso contrario WC = 0.
4	RS	X	Selección del banco de registros. El 2650 establece, mediante este bit, cuál de los dos bancos de registros ha de seleccionar: RS = 0: banco 0 con los registros R1, R2 y R3 RS = 1: banco 1 con los registros R4, R5 y R6.
5	IDC		Acarreo entre dígitos. El 2650 activa este bit cuando en operaciones aritméticas, o de desplazamiento de cantidades, ha tenido lugar un acarreo del bit 3 al 4. Es importante para operaciones de cálculo en aritmética binaria BCD.
6,7	CC		Código de condición o código de servicio. Este bit se activa al ejecutar todas aquellas instrucciones que ocasionan una modificación en los 7 registros de uso general o que realizan una comparación.

Para los códigos de condición, o de servicio, tenemos cuatro posibilidades. En las respectivas tablas a), b) y c) se comprendían los significados de CC1 y CCO después de realizar determinadas instrucciones.

a) Instrucciones de carga, de almacenamiento, de entrada, aritméticas, lógicas y de desplazamiento:

$<R>$ es

CC1	CC0	
0	0	igual a cero
0	1	positivo
1	0	negativo
1	1	XX

$<R>$ = contenido del registro R

XX = esta combinación de códigos de condición no se produce

b) Instrucciones de comparación:

b1) $<R0> = <R>$ es

b2) $<R0> = [M]$ es

CC1	CC0	
0	0	igual a cero
0	1	positivo
1	0	negativo
1	1	XX

$<R>$ = contenido del registro R

$<R0>$ = Contenido del registro R0

[M] = Contenido de la posición de memoria con dirección M

XX = La combinación de códigos de condición no se produce

c) Instrucciones de comprobación (TPSU, TPSL, TMI):

De los bits que están a «1» en la máscara, los bits correspondientes del bit que se comprueba están:

CC1	CC0	
0	0	todos a «1»
0	1	XX
1	0	no todos a «1»
1	1	XX

XX = La combinación de códigos de condición no se produce.

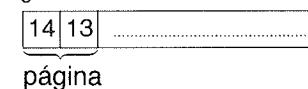
Para todos los registros del microprocesador podemos emplear las tablas siguientes:

Memoria de direcciones de retorno de subprograma.

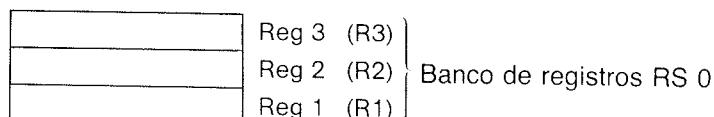
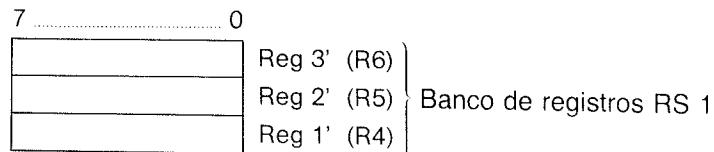
14	0
.	0
.	1
.	2
.	3
.	4
.	5
.	6
.	7

Memoria de lectura-escritura en modo de trabajo FILO (8 × 15 RAM).

Registro de dirección de instrucciones



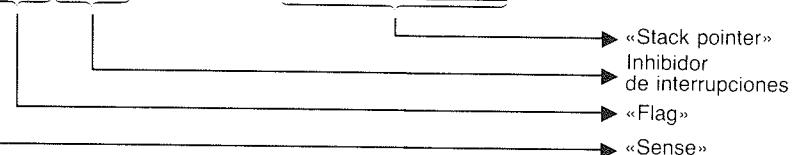
d) Registros de uso general.



Reg 0 (R0)

de estado de programa

6	5	4	3	2	1	0
F	II	libre		SP2	SP1	SP0



6	5	4	3	2	1	0	
1	CC0	IDC	RS	WC	OVF	COM	C

Comparación
Bit de desbordamiento
Con/sin acarreo
Selección de banco de registros
Acarreo entre dígitos
Códigos de condición

Clases de instrucciones para el microprocesador 2650

Las 75 instrucciones del microprocesador 2650 podemos agruparlas en diversas clases:

1. Instrucciones de carga y de almacenamiento:

Estas instrucciones ocasionan un intercambio de datos entre el registro R0 y uno de los 6 registros restantes. Trabajamos en «formato Z». Si el intercambio de datos ha de tener lugar entre un registro y una posición de la memoria, emplearemos el «formato A» o el «formato R». Las instrucciones permiten también cargar un valor de 8 bits, especificado en la propia instrucción, en uno de los 7 registros de uso general. Emplearemos, entonces, el «formato I». En esta clase, la instrucción «LOAD» (cargar) significa una transferencia de datos al registro y la instrucción «STORE» (almacenar) significa una transferencia de datos desde el registro.

2. Instrucciones de entrada/salida:

El microprocesador 2650 tiene dos posibilidades para las instrucciones de entrada/salida:

- Con instrucciones de 1 bit, puede procederse a intercambiar datos entre uno de los siete registros y una de las dos puertas de E/S de 8 bits (puertas C y D). Para ello adoptamos el «formato Z».
- Empleando instrucciones de 2 bits podemos realizar lo que se denomina «Extended I/O» (servicio de entrada/salida ampliado). En tal caso, el intercambio de datos se efectúa entre un registro y una de las 256 distintas puertas externas de E/S. Empleamos el «formato I» y el direccionamiento se realiza mediante los 8 bits inferiores del bus de direcciones. Este tipo de servicio no se emplea en el juego televisivo.

Adicionalmente, para una transmisión de 1 bit en serie, podemos utilizar el bit Sense (pin 1 de entrada) y el bit Flag (pin 40 de salida) del registro de estado de programa.

3. Operaciones aritméticas:

Con el microprocesador 2650 pueden efectuarse sumas y restas. Estas instrucciones dan lugar a la suma o resta entre el contenido de uno de los 6 registros y el contenido del registro R0. En este caso, tenemos el «formato Z». Con otras estructuras de instrucciones, lo que se efectúa es la suma o resta entre el contenido de una posición de memoria y el de un registro. Empleamos, entonces, el «formato A» o el «formato R». Con el «formato I», la suma o resta tiene lugar entre un registro y un valor de 8 bits especificado en la propia instrucción.

En el bit WC (con acarreo) del registro de estado de programa podemos especificar si en esas operaciones ha de tenerse en cuenta el bit de desbordamiento (bit 0 de PSL).

A estas operaciones aritméticas pertenece también la corrección decimal. La instrucción «RAR» sirve para realizar el acarreo decimal al emplear la aritmética BCD (decimal codificado en binario).

4. Operaciones lógicas:

Con estas operaciones, podemos llevar a cabo combinaciones lógicas entre dos operandos. Disponemos de las operaciones lógicas siguientes:

Función - AND (Y lógica)

Función - OR exclusiva (O exclusiva)

Función - OR inclusiva

Estas instrucciones dan lugar a la confrontación lógica entre el contenido de uno de los 6 registros y el contenido del registro R0. Empleamos el «formato Z». Si empleamos el «formato A» o el «formato R», las conexiones se efectúan entre el contenido de una posición de memoria y el de un registro. Además, mediante el «formato I» puede conectarse lógicamente el contenido de un registro con un valor de 8 bits especificado en la propia instrucción.

5. Funciones de desplazamiento:

Con una instrucción de desplazamiento puede desplazarse el contenido de un registro un bit a la

derecha o a la izquierda. Mediante el programa estableceremos si el bit C de desbordamiento (bit 0 en PSL) ha de ser tomado en cuenta como noveno bit.

6. Instrucciones de comparación:

Mediante estas instrucciones podemos comparar dos operandos entre sí. Como resultado de estas operaciones se activarán los dos bits de código de condición CC del registro de estado de programa.

CC1	CC0	
0	0	igual a cero
0	1	positivo
1	0	negativo
1	1	no se produce

Mediante el bit COM (comparación lógica/aritmética) establecemos en el registro de estado de programa si ambos operandos han de interpretarse como magnitudes con signo en la forma de complemento a 2 en sistema binario (COM = 0, comparación aritmética) o bien como magnitudes positivas sin bit de signo aritmético (COM = 1, comparación lógica).

7. Instrucciones de bifurcación y de salto:

Se distingue entre instrucciones de salto incondicional e instrucciones de salto condicionado.

Las primeras no dependen de ninguna condición (decisión lógica). Cuando el programa llega a una de éstas, el microprocesador las ejecuta inmediatamente. Para el microprocesador 2650 tenemos las instrucciones siguientes:

BCTR y BCTA: En la instrucción BCTR tenemos un direccionamiento relativo y, por tanto, trabajamos en «formato R». La instrucción BCTA tiene un direccionamiento absoluto y es de «formato B».

ZBRR: De ella se deriva un salto de programa a una información de dirección que es relativa a la dirección 0. Si acoplamos el «formato E» con el «for-

mato R» obtenemos el «formato ER». Se obtiene así una dirección entre -64 y +63 en la página 0 (Page 0), lo cual corresponde a las direcciones entre 1FC0 y 003F en el sistema hexadecimal.

BXA: Tenemos un salto de programa absoluto con indexación empleando para ella los registros índice R3 y R6. Para ello utilizamos el «formato EB», que es una combinación del «formato E» y el «formato B».

En las instrucciones de salto condicional, el curso posterior del programa depende de una condición (decisión). Trabajamos en los «formatos R o B». El salto a la instrucción que se encuentra en la dirección especificada en el salto tendrá lugar si el código de condición especificado en la instrucción de salto coincide con el estado de los bits del código de condición en PSL (instrucciones BCTR y BCTA) o bien, si no coincide con los bits de código de condición en PSL (instrucciones BCFR y BCFA). El salto a la dirección especificada en la instrucción se realizará sólo cuando el contenido de los registros especificados dependa de lo siguiente:

- a) El contenido después de incrementarlo en «1» no es cero, si se trata de las instrucciones BRNR y BRNA.
- b) El contenido después de incrementarlo en «1» no es cero, si se trata de las instrucciones BIRR y BIRA.
- c) El contenido después de decrementar en «1» (restar) no es cero, si se trata de las instrucciones BDRR y BDRA.

8. Instrucciones de saltos a subprogramas:

En estas instrucciones diferenciamos las de salto al subprograma y las de retorno al programa principal.

En las de salto a subprograma, diferenciamos nuevamente las instrucciones de salto condicional y las de salto incondicional. Lo mismo vale para las de retorno al programa principal.

Al producirse un salto a un subprograma, la dirección de la última instrucción en curso se deposita en el «stack pointer». En el retorno, el microprocesador recoge dicha dirección del «stack pointer» y puede proseguir inmediatamente el programa principal.

9. Instrucciones para seleccionar el registro de estado de programa:

Con estas instrucciones podemos consultar o activar el contenido de un bit del registro de estado de programa. Con la instrucción LPSU/LPSL se lleva el contenido del registro R0 a uno de los dos bits del PSR. Con la instrucción SPSU/SPSL hacemos lo contrario, es decir, llevamos al registro R0 el contenido de uno de los dos bits del PSR.

Con la instrucción CPSU/SPSL, o la PPSU/PPSL, todos los bits «0» o «1» de la máscara incluida en la instrucción se transfieren al correspondiente bit de estado de programa. Los bits restantes no se modifican.

Con la instrucción TPSU/TPSL se ponen los bits de código de condición a 00, si en los lugares en los que hay un «1» en la máscara hay también un «1» en el correspondiente bit de estado de programa. En caso contrario obtendremos CC = 10.

10. Restantes instrucciones:

Con la instrucción de «HALT», el microprocesador pasa a su estado «HALT» (parada). No podemos sacar al microprocesador de ese estado sólo mediante software, es preciso una manipulación física directa en las líneas «RESET» o «INTREQ».

Mediante la instrucción «NOP» (No Operation = ninguna operación) podemos introducir una demora de tiempo en el programa. La instrucción sólo consume tiempo. Con la instrucción «TMI», el microprocesador compara el contenido de un registro con una máscara dada en la propia instrucción. Los bits de código de condición se ponen a 00 cuando en las posiciones de la máscara hay un «1», en caso contrario CC = 10.

**Lista de instrucciones
del microprocesador 2650**

LODZ

0	0	0	0	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Load register zero: Cargar registro cero)

Cargar el contenido del registro especificado en el registro R0.

LODI

0	0	0	0	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Load immediate: Cargar registro con el valor inmediato)

Cargar el valor definido en los bits 0 a 7 en el registro especificado por los bits 8 y 9.

LODR

0	0	0	0	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Load relative: Cargar registro de direccionamiento relativo)

Cargar el contenido de la dirección de memoria definida por los bits 0 a 6 en el registro definido mediante los bits 8 y 9.

LODA

0	0	0	0	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(Load absolute: Cargar registro con direccionamiento absoluto)

Cargar el contenido de la dirección de memoria definida mediante los bits 0 a 12 en el registro definido con los bits 16 y 17.

STRZ

1	1	0	0	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Store register zero: Almacenar registro cero)

Almacenar el contenido del registro R0 en el registro definido por los bits 0 y 1.

STRR

1	1	0	0	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Store relative: Almacenar registro con direccionamiento relativo)

Almacenar el contenido del registro definido con los bits 8 y 9 en la dirección de memoria calculada efectiva que se encuentra en los bits 0 al 6.

STRA

1	1	0	0	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(Store absolute: Almacenar registro con direccionamiento absoluto)

Almacenar el contenido del registro definido con los bits 16 y 17 en la dirección de memoria definida con los bits 0 al 12.

ADDZ

1	0	0	0	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Add to register zero w/wo carry: Sumar al registro cero con/sin acarreo)

Sumar al contenido del registro definido con los bits 0 y 1 el registro R0, teniendo en cuenta el bit de estado WC.

ADDI

1	0	0	0	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Add immediate w/wo carry: sumar el valor inmediato con/sin acarreo)

Sumar al valor definido por los bits 0 a 7 el contenido del registro definido por los bits 8 y 9, teniendo en cuenta el bit de estado WC.

ADDR

1	0	0	0	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Add relative w/wo carry: Sumar con direccionamiento relativo con/sin acarreo)

Sumar al contenido de la posición de memoria, calculada con los bits 0 a 6, el contenido del registro indicado por los bits 8 y 9, teniendo en cuenta el bit de estado WC.

ADDA

1	0	0	0	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(Add absolute w/wo carry: Suma con direccionamiento absoluto con/sin acarreo)

Sumar al contenido de la dirección de memoria definida por los bits 0 a 12 el contenido del registro definido por los bits 16 y 17, teniendo en cuenta el bit de estado WC.

SUBZ

1	0	1	0	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Subtract from register zero w/wo Borrow: restar del registro cero con/sin acarreo negativo)

Sumar el complemento a 2 del contenido del registro indicado por los bits 0 y 1 con el contenido del R0, teniendo en cuenta el bit de estado WC.

SUBI

1	0	1	0	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Subtract immediate w/wo borrow: Restar inmediato con/sin acarreo negativo)

Sumar el complemento a 2 del valor definido por los bits 0 a 7 con el contenido del registro definido por los bits 8 y 9, teniendo en cuenta el bit de estado WC.

SUBR

1	0	1	0	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Subtract relative w/wo borrow: restar con direccionamiento relativo con/sin acarreo negativo)

Sumar el complemento a 2 del contenido de la dirección de memoria calculada por los bits 0 a 6 con el contenido del registro direccionado por los bits 8 y 9, teniendo en cuenta el bit de estado WC.

SUBA

1	0	1	0	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(Subtract absolute w/wo borrow: Restar con direccionamiento absoluto con/sin acarreo negativo)

Sumar el contenido de la posición de memoria calculada mediante los bits 0 a 12 con el contenido del registro definido por los bits 16 y 17, teniendo en cuenta el bit de estado WC.

DAR

1	0	0	1	0	1	.	.
---	---	---	---	---	---	---	---

 (Z)

(Decimal adjust register: Convección decimal de registro)

Corregir el contenido del registro definido por los bits 0 y 1, dependiendo de los bits de estado C y IDC, de modo que los dos valores hexadecimales se transforman en cantidades decimales.

ANDZ

0	1	0	0	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(AND to register zero, R0: Efectuar Y lógica con el registro R0)

Efectuar una operación lógica AND (Y lógica) entre el contenido del registro definido por los bits 0 y 1 y el contenido del R0. El resultado se almacena en R0. El otro registro no se modifica.

ANDI

0	1	0	0	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(AND Immediate: AND inmediato)

Efectuar una operación lógica AND entre el valor definido en los bits 0 a 7 y el contenido del registro definido por los bits 8 y 9. El resultado se almacena en ese registro.

ANDR

0	1	0	0	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(AND relative: AND con direccionamiento relativo)

Efectuar una operación lógica AND entre el contenido de la posición de memoria calculada con los bits 0 a 6 y el contenido del registro definido por los bits 8 y 9. El resultado se almacena en el registro.

ANDA

0	1	0	0	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(AND absolute: AND con direccionamiento absoluto)

Efectuar una operación lógica AND entre el contenido de la posición de memoria especificada por los bits 0 a 12 y el contenido del registro definido mediante los bits 16 y 17. El resultado se almacena en el registro.

IORZ

0	1	1	0	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Inclusive OR to register zero: OR inclusiva con el registro cero)

Efectuar una operación lógica OR inclusiva entre el

contenido del registro definido por los bits 0 y 1 y el contenido del registro R0. El resultado se almacena en R0.

IORI

0	1	1	0	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Inclusive OR Immediate: OR inclusiva inmediato)
Efectuar una operación entre el contenido del registro definido por los bits 8 y 9 y el operando inmediato que se encuentra en los bits 0 a 7. El resultado se almacena en el registro.

IORR

0	1	1	0	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Inclusive OR relative: OR inclusiva con direccionamiento relativo)
Efectuar una OR inclusiva entre el contenido de la posición de memoria calculada con los bits 0 a 6 y el contenido del registro especificado por los bits 8 y 9. El resultado se almacena en el registro.

IORA

0	1	1	0	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(Inclusive OR absolute: Or inclusiva con direccionamiento absoluto)
Efectuar una OR inclusiva entre el contenido de la posición de memoria especificada en los bits 0 a 12 y el contenido del registro definido mediante los bits 16 y 17. El resultado se almacena en el registro.

EORZ

0	0	1	0	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Exclusive OR to register zero: OR exclusiva al registro cero)
Efectuar una OR exclusiva entre el contenido del registro definido por los bits 0 y 1 y el contenido del R0. El resultado se almacena en R0.

EORI

0	0	1	0	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Exclusive OR immediate: OR exclusiva inmediato)
Efectuar una OR exclusiva entre el valor definido mediante los bits 0 a 7 y el contenido del registro

definido por los bits 8 y 9. El resultado se almacena en el registro.

EORR

0	0	1	0	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Exclusive OR relative: OR exclusiva con direccionamiento relativo)
Efectuar una OR exclusiva entre el contenido de la dirección de memoria calculada mediante los bits 0 a 6 y el contenido del registro definido por los bits 8 y 9. El resultado se almacena en el registro.

EORA

0	0	1	0	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(Exclusive OR absolute: OR exclusiva con direccionamiento absoluto)
Efectuar una OR exclusiva entre el contenido de la dirección de memoria especificada por los bits 0 a 12 y el contenido del registro definido por los bits 16 y 17. El resultado se almacena en el registro.

COMZ

1	1	1	0	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Compare to register zero Arithmetic/logical: Comparar contenido aritmético/lógico con el de registro cero)
Comparar el contenido del registro definido por los bits 0 y 1 con el del registro R0, teniendo en cuenta el bit de estado COM. Dependiendo del resultado, poner a «1» los bits de estado CCO y CC1.

COMI

1	1	1	0	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Compare immediate arithmetic/logical: Comparación aritmética/lógica del valor inmediato)
Comparar el valor inmediato especificado en los bits 0 a 7 con el contenido del registro definido por los bits 8 y 9, teniendo en cuenta el bit de estado COM. Poner a «1» los bits de estado CCO y CC1 dependiendo del resultado.

COMR

1	1	1	0	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Compare relative arithmetic/logical: Comparación aritmética/lógica en direccionamiento relativo)

Comparar el contenido de la dirección de memoria calculada con ayuda de los bits 0 a 6 con el contenido del registro definido por los bits 8 y 9, teniendo en cuenta el bit de estado COM. Poner a «1» los bits de estado CCO y CC1 dependiendo del resultado.

COMA

1	1	1	0	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(Compare absolute arithmetic/logical: Comparación aritmética/lógica en direccionamiento absoluto)

Comparar el contenido de la dirección de memoria especificada en los bits 0 a 12 con el contenido del registro definido por los bits 16 y 17, teniendo en cuenta el bit de estado COM. Poner a «1» los bits CCO y CCL dependiendo del resultado.

RRR

1	0	1	1	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Rotate register right w/wo carry: rotar registro a derechas con/sin acarreo)

Desplazar el contenido del registro definido por los bits 0 y 1 un lugar a la derecha, teniendo en cuenta el bit de estado WC. El bit que desborda por la derecha pasa a ocupar el primer lugar por la izquierda.

RRL

1	1	0	1	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Rotate register left w/wo carry: rotar registro a izquierdas con/sin acarreo)

Desplazar el contenido del registro definido por los bits 0 y 1 un lugar a la izquierda, teniendo en cuenta el bit de estado WC. El bit que desborda por la izquierda pasa a ocupar el primer lugar por la derecha.

BCTR

0	0	0	1	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Branch on condition true relative: Bifurcación en direccionamiento relativo si la condición es verdadera)

Cuando los bits de estado CC1 y CC0 coinciden respectivamente con los bits 8 y 9, bifurcar (saltar) a la dirección de memoria calculada mediante los bits 0 a 6. Si los bits 8 y 9 están a «1», bifurcar incondicionalmente.

BCTA

0	0	0	1	1	1	.	.
---	---	---	---	---	---	---	---

 (B)

(Branch on condition true absolute: Bifurcación en direccionamiento absoluto si la condición es verdadera)

Cuando los bits 16 y 17 coinciden con los bits de estado CC0 y CC1, bifurcar a la dirección especificada en los bits 0 a 14. Bifurcar incondicionalmente si los bits 8 y 9 están a 1.

BCFR

1	0	0	1	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Branch on condition false relative: Bifurcación en direccionamiento relativo si condición es falsa)

Si los bits 8 y 9 no coinciden con los bits de estado CC0 y CC1, bifurcar a la dirección calculada mediante los bits 0 a 6.

BCFA

1	0	0	1	1	1	.	.
---	---	---	---	---	---	---	---

 (B)

(Branch on condition false absolute: Bifurcación en direccionamiento absoluto si condición es falsa)

Si los bits 16 y 17 no coinciden con los bits de estado CC0 y CC1, bifurcar a la posición de memoria definida por los bits 0 a 14.

BRNR

0	1	0	1	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Branch on register non-zero relative: Bifurcación en direccionamiento relativo si registro no es cero)

Si el contenido del registro definido por los bits 8 y 9 no es cero, bifurcar a la dirección de memoria calculada con los bits 0 a 6.

BRNA

0	1	0	1	1	1	.	.
---	---	---	---	---	---	---	---

 (B)

(Branch on register non-zero absolute: Bifurcación en direccionamiento absoluto si registro no es cero)

Si el contenido del registro definido por los bits 16 y 17 no es cero, bifurcar a la dirección de memoria especificada en los bits 0 a 14.

BIRR

1	1	0	1	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Branch on incrementing register relative: Bifurcación en direccionamiento relativo al incrementar registro)

Incrementar el contenido del registro definido por los bits 8 y 9 y si ese contenido es cero, bifurcar a la dirección de memoria calculada mediante los bits 0 a 6.

BIRA

1	1	0	1	1	1	.	.
---	---	---	---	---	---	---	---

 (B)

(Branch on incrementing register absolute: Bifurcación en direccionamiento absoluto al incrementar registro)

Incrementar el contenido del registro definido por los bits 16 y 17 y si ese contenido es cero, bifurcar a la dirección especificada en los bits 0 a 14.

BDRR

1	1	1	1	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Branch on decrementing register relative: Bifurcación en direccionamiento relativo al decrementar registro)

Decrementar el contenido del registro definido por los bits 8 y 9, y si ese contenido no es cero, bifurcar a la dirección de memoria calculada mediante los bits 0 a 6.

BDRA

1	1	1	1	1	1	.	.
---	---	---	---	---	---	---	---

 (B)

(Branch on decrementing register absolute: Bifurcación en direccionamiento absoluto al decrementar registro)

Decrementar el contenido del registro definido por los bits 16 y 17 y si ese contenido no es cero, bifurcar a la dirección de memoria calculada mediante los bits 0 a 14.

ZBRR

1	0	0	1	1	0	1	1
---	---	---	---	---	---	---	---

 (R)

(Zero Branch relative, unconditional: Bifurcación incondicional a cero, en direccionamiento relativo)

Bifurcar incondicionalmente a la dirección de memo-

ria definida con el bit 0 a 6. La dirección de base es 0. No se direcciona ningún registro del 2650.

BXA

1	0	0	1	1	1	1	1
---	---	---	---	---	---	---	---

 (B)

(Branch indexed absolute, unconditional: Bifurcación incondicional con direccionamiento absoluto e indexado)

Bifurcar incondicionalmente a la dirección de memoria que se obtiene al sumar los bits 0 a 14 con el contenido del registro R3. El contenido del registro no se modifica.

BSTR

0	0	1	1	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Branch to subroutine on condition true, relative: Bifurcación subprograma si condición verdadera, en direccionamiento relativo)

Bifurcar a la dirección de memoria calculada mediante los bits 0 a 6 si los bits 8 y 9 coinciden con los bits de estado CC0 y CC1. Bifurcar incondicionalmente si los bits 8 y 9 están a «1». Almacenar también en el «stack pointer» la dirección de la instrucción en curso antes de bifurcar el subprograma.

BSTA

0	0	1	1	1	1	.	.
---	---	---	---	---	---	---	---

 (B)

(Branch to subroutine on condition true, absolute: Bifurcación a subprograma si condición verdadera, en direccionamiento absoluto)

Bifurcar a la dirección de memoria especificada en los bits 0 a 14, si los bits 16 y 17 coinciden con los bits de estado CC0 y CC1. Bifurcar incondicionalmente si los bits 8 y 9 están a 1. Almacenar en el «stack pointer» la dirección de la instrucción en curso antes de saltar al subprograma.

BSFR

1	0	1	1	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Branch to subroutine on condition false, relative: Bifurcación a subprograma si condición falsa, en direccionamiento relativo)

Bifurcar a la dirección de memoria calculada mediante los bits 0 a 6, si los bits 8 y 9 no coinciden con

los bits de estado CC0 y CC1. Almacenar en el «stack pointer» la dirección de la instrucción en curso antes de bifurcar al subprograma.

BSFA

1	0	1	1	1	1	.	.
---	---	---	---	---	---	---	---

 (A)

(Branch to subroutine on condition false, absolute: Bifurcación a subprograma si condición falsa, en direccionamiento absoluto)

Bifurcar a la dirección de memoria especificada en los bits 0 a 14, si los bits 16 y 17 no coinciden con los bits CC0 y CC1. Almacenar en el «stack pointer» la dirección de la instrucción en curso antes de bifurcar al subprograma.

BSNR

0	1	1	1	1	0	.	.
---	---	---	---	---	---	---	---

 (R)

(Branch to subroutine on non-zero register, relative: Bifurcación a subprograma si registro no cero, en direccionamiento relativo)

Bifurcar a la dirección de memoria calculada mediante los bits 0 a 6, si el contenido del registro definido por los bits 8 y 9 no es cero. Almacenar en el «stack pointer» la dirección de la instrucción en curso antes de bifurcar al subprograma.

BSNA

0	1	1	1	1	1	.	.
---	---	---	---	---	---	---	---

 (B)

(Branch to subroutine on non-zero register, absolute: Bifurcación a subprograma si registro no cero, en direccionamiento absoluto)

Bifurcar a la dirección de memoria especificada en los bits 0 a 14, si el contenido del registro definido por los bits 16 y 17 no es cero. Almacenar en el «stack pointer» la dirección de la instrucción en curso antes de saltar al subprograma.

ZBSR

1	0	1	1	1	0	1	1
---	---	---	---	---	---	---	---

 (R)

(Zero branch to subroutine relative: Bifurcación cero a subprograma, en relativo)

Bifurcar incondicionalmente a la dirección de memoria definida por los bits 0 a 6. La dirección base es 0. No se direcciona ningún registro. Almacenar en el

«stack pointer» la dirección de la instrucción en curso antes de bifurcar al subprograma.

BSXA

1	0	1	1	1	1	1	1
---	---	---	---	---	---	---	---

 (B)

(Branch to subroutine, indexed absolute unconditional: Bifurcación incondicional, en indexado y en relativo)

Bifurcar incondicionalmente a la dirección de memoria obtenida al sumar los bits 0 a 14 con el contenido del registro R3. El contenido de este registro no se altera. Almacenar en el «stack pointer» la dirección de la instrucción en curso antes de efectuar el salto.

RETC

0	0	0	1	0	1	.	.
---	---	---	---	---	---	---	---

 (Z)

(Return from subroutine, conditional: Retorno condicional de subprograma)

Si los bits 0 y 1 coinciden con los bits de código de condición CC0 y CC1, bifurcar a la dirección que se almacenó la última en el «stack pointer» y decrementar este último. Si los bits 8 y 9 están a «1», bifurcar incondicionalmente.

RETE

0	0	1	1	0	1	.	.
---	---	---	---	---	---	---	---

 (Z)

(Return from subroutine and enable: Retorno de subprograma y desinhibición)

Si los bits 0 y 1 coinciden con los bits de estado CC0 y CC1, bifurcar a la dirección última almacenada en el «stack pointer» y decrementarla. A la vez, borrar el bit de estado II. Si los bits 8 y 9 están a 1, saltar incondicionalmente.

WRTD

1	1	1	1	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Write data: Escribir datos, salida datos)

Escribir, por la puerta de salida D, el contenido del registro definido con los bits 0 y 1.

REDD

0	1	1	1	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Read data: Leer datos, entrada datos)

Recoger el contenido de la puerta de entrada D en el registro definido con los bits 0 y 1.

WRTC

1	0	1	1	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Write control: Escribir información de control, salida de control)

Escribir el contenido del registro definido por los bits 0 y 1 en la puerta de salida C.

REDC

0	0	1	1	0	0	.	.
---	---	---	---	---	---	---	---

 (Z)

(Read control: Leer información de control, entrada de control)

Recoger el contenido de la puerta de entrada C en el registro definido por los bits 0 y 1.

WRTE

1	1	0	1	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Write extended: Escritura extendida, salida extendida)

Escribir el contenido del registro definido por los bits 8 y 9 en la puerta de salida definida por los bits 0 a 7.

REDE

0	1	0	1	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Read extended: Lectura extendida, entrada extendida)

Recoger el contenido de la puerta de entrada definida por los bits 0 a 7 en el registro definido por los bits 8 y 9.

HALT

0	1	0	0	0	0	0	0
---	---	---	---	---	---	---	---

 (E)

(Halt, enter wait state: Parada, pasar a estado de espera)

Interrumpir el funcionamiento del microprocesador 2650 y reaccionar sólo ante RESET o INTREQ. Esta instrucción se da sólo por software y se anula nuevamente por hardware.

NOP

1	1	0	0	0	0	0	0
---	---	---	---	---	---	---	---

 (E)

(No operation: ninguna operación)

Instrucción de no operación. El microprocesador 2650 se mantiene en el programa. La demora en tiempo se consigue por software.

TMI

1	1	1	1	0	1	.	.
---	---	---	---	---	---	---	---

 (I)

(Test under mask immediate: Comprobar contenido de registro con una máscara, en inmediato)

Comprobar si cada bit del 0 al 7, que esté a «1», tiene su equivalente también a «1» en el registro definido por los bits 8 y 9. Dependiendo del resultado de la comparación, se pondrán a «1» los bits de estado CC0 y CC1.

LPSU

1	0	0	1	0	0	1	0
---	---	---	---	---	---	---	---

 (E)

(Load program status, upper: Cargar registro de estado de programa, bit superior)

Cargar el bit superior del registro de estado de programa PSU con el contenido del registro R0.

LPSL

1	0	0	1	0	0	1	1
---	---	---	---	---	---	---	---

 (E)

(Load program status, lower: Cargar registro de estado de programa, bit inferior)

Cargar el bit inferior del registro de estado de programa PSU con el contenido del registro R0.

SPSU

0	0	0	1	0	0	1	0
---	---	---	---	---	---	---	---

 (E)

(Store program status, Upper: Almacenar registro de estado de programa, bit superior)

Cargar el registro R0 el contenido del bit superior del registro de estado de programa PSU.

SPSL

0	0	0	1	0	0	1	1
---	---	---	---	---	---	---	---

 (E)

(Store program status, lower: Almacenar registro de estado de programa, bit inferior)

Cargar en el registro R0 el contenido del bit inferior del registro de estado de programa PSL.

CPSU

0	1	1	1	0	1	0	0
---	---	---	---	---	---	---	---

 (E)

(Clear program status, upper, masked: Borrar registro de estado de programa, bit superior, con máscara)

Borrar cada bit del bit superior de la palabra de esta-

do de programa cuyo bit equivalente del 0 al 7 está a «1».

CPSL 0 1 1 1 0 1 0 1 (E)

(Clear program status lower, masked: Borrar registro de estado de programa, bit inferior con máscara)

Borrar cada bit del bit inferior de la palabra de estado de programa cuyo bit equivalente del 0 al 7 esté a «1».

PPSU 0 1 1 1 0 1 1 0 (E)

(Preset program status, upper, masked: Poner a uno el registro de estado de programa, bit superior, con máscara)

Poner a 1 cada bit del bit superior de la palabra de estado de programa cuyo equivalente del bit 0 al 7 esté también a «1».

PPSL 0 1 1 1 0 1 1 1 (E)

(Preset program status, lower, masked:

Poner en 1 el registro de estado de programa, byte inferior, con máscara)

Poner a 1 cada bit del byte inferior de palabra de estado de programa cuyo equivalente del bit 0 al 7 esté también a «1».

TPSU | 1 0 1 1 0 1 0 0 (E)

(Test program status, upper masked: Comprobar el registro de estado de programa, bit superior, con máscara)

Comprobar si cada bit del 0 al 7 que está a «1» tiene su correspondiente bit en el bit superior de la palabra de estado de programa también a «1». Poner a «1» los bits de estado CC0, CC1 dependiendo del resultado.

TPSL

1	0	1	1	0	1	0	1
---	---	---	---	---	---	---	---

 (E)

(Test program status, lower, masked: Comprobar el registro de estado de programa, bit inferior, con máscara)

Comprobar si cada bit del 0 al 7 que esté a «1» tiene su correspondiente bit en el bit inferior de la palabra inferior de estado de programa también a «1». Poner a «1» los bits de estado CC0, CC1 dependiendo del resultado.

Con lo anterior, tenemos 245 instrucciones para el sistema del microprocesador. Prestese atención a la estructura de las instrucciones para saber si son de 1, 2 o 3 bits. Todo ello con el fin de garantizar que la ejecución de programas sea la correcta.

 es posible la indexación

■ direcccionamiento con 15 bits

código nemónico	código de operación			formato	bit	ciclo	bit PSW					
	R o CC						CC	IDS	C	OVF	SP	II
LOD	Z 03 02 01 0	12	1	2	2	2						
	I 07 06 05 04	2I	2	2	2	2						
	R 0B 0A 09 08	2R	2	3	3	3						
	A 0F 0E 0D 0C	3A	3	4	4	4						
STR	Z C3 C2 C1 -	1Z	1	2	2	2						
	R CB CA C9 C8	2R	2	3	3	3						
	A CF CE CD CC	3A	3	4	4	4						
DAD	Z 83 82 81 80	1Z	1	2	2	2						
	I 87 86 85 84	2I	2	2	2	2						
	R 8B 8A 89 88	2R	2	3	3	3						
	A 8F 8E 8D 8C	3A	3	4	4	4						
SUB	Z A3 A2 A1 A0	1Z	1	2	2	2						
	I A7 A6 A5 A4	2I	2	2	2	2						
	R AB AA A9 A8	2R	2	3	3	3						
	A AF AE AD AC	3A	3	4	4	4						
AND	Z 43 42 41 40	1Z	1	2	2	2						
	I 47 46 45 44	2I	2	2	2	2						
	R 4B 4A 49 48	2R	2	3	3	3						
	A 4F 4E 4D 4C	3A	3	4	4	4						
IOR	Z 63 62 61 60	1Z	1	2	2	2						
	I 67 66 65 64	2I	2	2	2	2						
	R 6B 6A 69 68	2R	2	3	3	3						
	A 6F 6E 6D 6C	3A	3	4	4	4						
EOR	Z 23 22 21 20	1Z	1	2	2	2						
	I 27 26 25 24	2I	2	2	2	2						
	R 2B 2A 29 28	2R	2	3	3	3						
	A 2F 2E 2D 2C	3A	3	4	4	4						

código nemónico	código de operación			formato	bit	ciclo	bit PSW					
	R oder CC						CC	IDS	C	OVF	SP	II
COM	Z E3 E2 E1 E0	1Z	1	2	2	2						
	I E7 E6 E5 E4	2I	2	2	2	2						
	R EB EA E9 E8	2R	2	3	3	3						
	A EF EE ED EC	3A	3	4	4	4						
RRR	Z 53 52 51 50	1Z	1	2	2	2						
	I 6E 6D 6C 6B	2I	2	2	2	2						
	R D3 D2 D1 D0	1Z	1	2	2	2						
BCT	Z 1B 1A 19 18	2R	2	2	2	2						
	I A 1F 1E 1D 1C	3B	3	3	3	3						
	R - 9A 99 98	2R	2	3	3	3						
	A - 9E 9D 9C	3B	3	3	3	3						
BRN	Z 5B 5A 59 58	2R	2	2	2	2						
	I A 5F 5E 5D 5C	3B	3	3	3	3						
	R DB DA D9 D8	2R	2	3	3	3						
	A DF DE DD DC	3B	3	3	3	3						
BDR	Z FB FA F9 F8	2R	2	2	2	2						
	I A FF FE FD FC	3B	3	3	3	3						
ZBRR	Z 9B - -	2ER	2	3	3	3						
	I BXA 9F - -	3EB	3	3	3	3						
BST	Z 3B 3A 39 38	2R	2	2	2	2						
	I A 3F 3E 3D 3C	3B	3	3	3	3						
BSF	Z R - BA B9 B8	2R	2	3	3	3						
	I A - BE BD BC	3B	3	3	3	3						
BSN	Z 7B 7A 79 78	2R	2	2	2	2						
	I A 7F 7E 7D 7C	3B	3	3	3	3						
ZBST	Z BB - -	2ER	2	3	3	3						
	I BSXA BF - -	3EB	3	3	3	3						
RET	Z C 17 16 15 14	1Z	1	3	3	3						
	I E 37 36 35 34	1Z	1	3	3	3						

1.2 INTERFACE DE VIDEO PROGRAMABLE

Después de ocuparnos del modo de trabajo y de las funciones del microprocesador 2650, vamos a referirnos a la interface de video programable. Este elemento es la interface, es decir, el elemento que sirve para acoplar el sistema microprocesador y la parte de televisión.

Con la interface de video programable podemos obtener dos variantes de la imagen de pantalla. La primera de ellas es una memoria RAM de regeneración de imagen y la designaremos como sistema RAM de regeneración de imagen. La segunda posibilidad es el sistema orientado al objeto, que es más fácil de realizar mediante la electrónica.

En la figura 1.15 se muestra el diagrama de bloques de un sistema controlado por microprocesador que cuenta con una memoria RAM de imagen. Una RAM (Random access memory: memoria de acceso aleatorio) es una memoria de lectura-escritura a cuyas posiciones puede accederse aleatoriamente. Al contrario de lo que sucede en las ROM (Read only memory = memoria de sólo lectura), en las RAM podremos introducir datos y volverlos a extraer más tarde, sin que por ello se destruyan los datos. En un sistema RAM de regeneración de imagen no tenemos una llamada a memoria a consecuencia de un programa fijo en una ROM, sino que el sistema recoge los datos introducidos, para la imagen televisiva, a partir de una memoria de escritura-lectura. Esta memoria ha de tener una gran capacidad de almacenamiento.

A parte de la RAM necesitamos también la memoria para el programa o memoria de instrucciones. Para ella emplearemos una ROM.

Los distintos puntos de la imagen, en un sistema de regeneración RAM, no dependen de un programa almacenado en una memoria fija de sólo lectura (ROM). Podemos determinarlos individualmente mediante un teclado externo. Obtenemos de este modo un «gráfico por ordenador».

El almacenamiento completo de la imagen de pantalla se realiza en una memoria de lectura-escritura.

tura. La capacidad de bits de esa memoria va a determinar el poder de resolución para la imagen de pantalla. En la figura 1.15, la memoria tiene una configuración de $4k \times 8$ bits, es decir, disponemos de 4096 direcciones y cada una de ellas corresponde a un bit. Esto proporciona una capacidad de almacenamiento total de 32 kbits. La memoria para imagen RAM puede ampliarse a $32k \times 8$ bits, ya que el microprocesador 2650 está diseñado para un direccionamiento completo de 32k posiciones. De esta forma tendríamos una capacidad de memoria de $32k \times 8$ bits = 256k (≈ 262144 posiciones de memoria).

Para una matriz de 128×124 puntos y 4 colores, necesitamos una memoria de 32k, como se indica en la figura 1.15. De esta forma, podemos descomponer la imagen televisiva en 128 columnas y 124 líneas, con lo cual obtenemos una buena resolución. Si ampliamos la capacidad a 64k, podemos descomponer en la imagen una matriz de 128×248 puntos.

Para dar servicio a una memoria de regeneración de imagen RAM, necesitamos un canal de DMA (direct memory access: acceso directo a memoria). Esto quiere decir que, mediante teclado o potenciómetro, tenemos que estar en condiciones de modificar directamente el contenido de la memoria de imagen RAM. Sin este canal DMA, tendríamos que hacer una consulta al microprocesador a través de la memoria de programa o de instrucciones. Como esa consulta dura bastante, debido a la estructura de las instrucciones y a la ejecución del programa, obtendríamos una imagen con centelleo. Mediante el acceso directo a memoria, los datos se introducen inmediatamente en la memoria de imagen RAM y quedan disponibles para la visualización en pantalla sin demora alguna. Obtenemos, así, una imagen televisiva sin centelleo.

Para estas memorias RAM de pantalla se aplican memorias dinámicas de lectura-escritura. Puesto que los elementos de estas memorias pueden con-

sultarse cada 20 ms, no necesitamos «refrescar» (regenerar) la imagen por separado.

El sistema de memoria de regeneración de imagen RAM, sin embargo, es difícil de resolver. No obstante, ofrece al técnico en electrónica muchas más posibilidades de juego. Al desconectar el equipo se borra el contenido de la memoria de imagen, a no ser que se trate de una memoria intermedia con baterías.

La figura 1.15 muestra también un convertidor A/D que convierte la entrada analógica del potenciómetro en un valor digital de 8 dígitos. Para el modulador de color y para todo el sistema microprocesador se necesitan la interfaz de video RAM y el generador de impulsos de sincronización. El cuarzo de 8,86 MHz nos genera la base de tiempos más importante para que tengamos un transcurso de tiempo continuo.

En los nuevos aparatos de televisión disponemos de una entrada directa de video. De ser así, podemos omitir el modulador de AF. Introduciremos directamente en la entrada de video el tono y la señal RGB.

La señal RGB es la portadora de color para el aparato de televisión (R=rojo, G=verde, y B=azul). La entrada de video puede generar directamente la señal RGB.

En los aparatos de televisión, sin entrada de video, necesitamos un modulador de color para generar la señal FBAS. En la figura 1.15, a partir de la señal RGB de la salida del bloque, obtenemos la señal AS para borrado de color y la sincronización. En el modulador de color se añade a esta señal AS la señal-B para la señal de imagen. Con ello el modulador genera la señal BAS a partir de las señales As y B. Mediante la suma de las señales se obtiene la señal BAS de frecuencia de líneas para el televisor. Mediante la señal de tipo de color F se obtiene la FBAS para cada televisor en color.

En muchos aparatos podemos controlar directamente la pantalla con la señal FBAS.

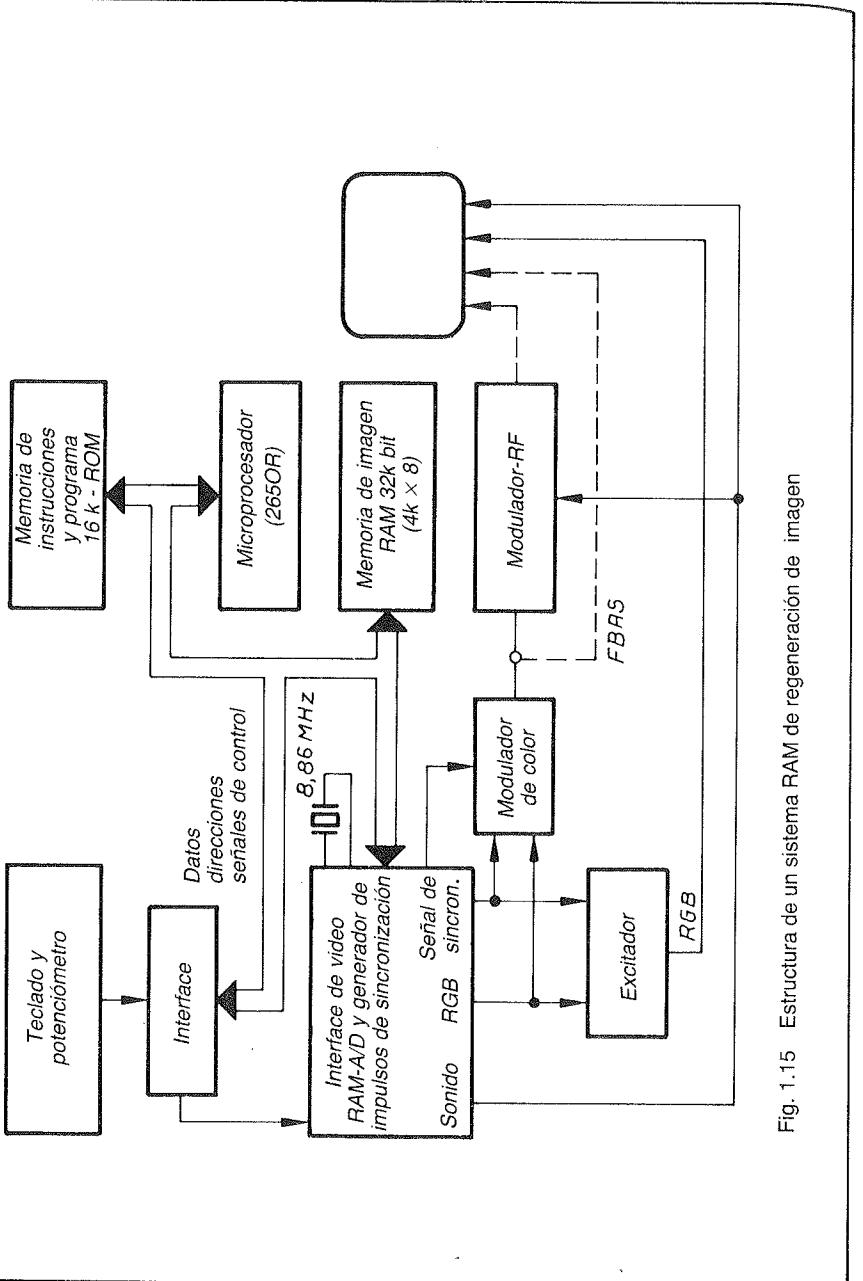


Fig. 1.15 Estructura de un sistema RAM de regeneración de imagen

Otro tipo de transmisión es con el modulador de AF. Suministramos la señal FBAS con un ancho de banda de 5,5 MHz a un modulador AZ que modula la señal en un canal de televisión. Como el modulador es muy sensible, se recomienda casi siempre seleccionar el canal 3 o el 4. En nuestro caso elegimos el canal 3.

En este libro nos ocuparemos sólo del sistema orientado al objeto, que puede programarse bastante más fácilmente. La figura 1.16 nos muestra este sistema y apreciaremos la sencillez de sus unidades funcionales.

El programa para el sistema orientado al objeto está almacenado en la memoria de programas. Aquí utilizaremos una EPROM (Erasable programmable read only memory: memoria sólo lectura, programable y borrable). Con un dispositivo de programación (programador) autoconstruido podemos: introducir un programa en la EPROM, borrarlo eventualmente y programar un nuevo contenido.

El microprocesador 2650 obtiene sus instrucciones de la memoria de programas y puede así trabajar con ellas. En conjunción con la interface de video programable tenemos ya el sistema de microprocesador completo para el juego de televisión. La interface entre el teclado y el sistema de buses es un simple conmutador analógico o un multiplexor analógico. La entrada puede hacerse o bien digital a través del teclado o bien analógica a través de los potenciómetros.

La interface de video se controla mediante el generador de impulsos de sincronismo USG 2621 (Universal Sync Generator). Mediante este dispositivo, la interface de video programable puede generar la señal RGB y el tono. Con una televisión en color que tenga entrada directa de video no necesitamos modulador de color ni de AF, sino tan sólo un excitador entre PVI 2636 y la entrada de video. El aparato de televisión podrá emitir el tono y la señal RGB.

En el aparato que describimos, necesitamos un modulador de color y de AF. El PVI 2636, mediante

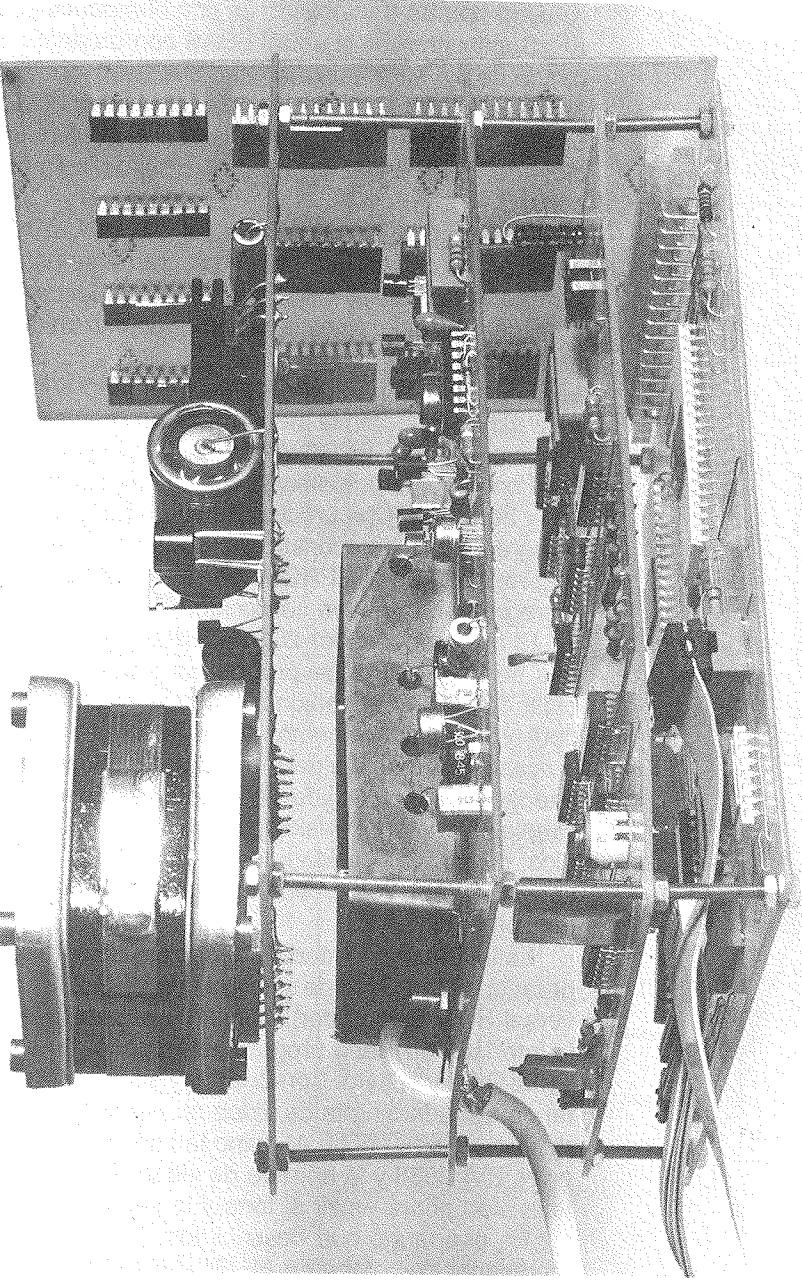


Fig. 1.16 Construcción de un sistema orientado a objetos

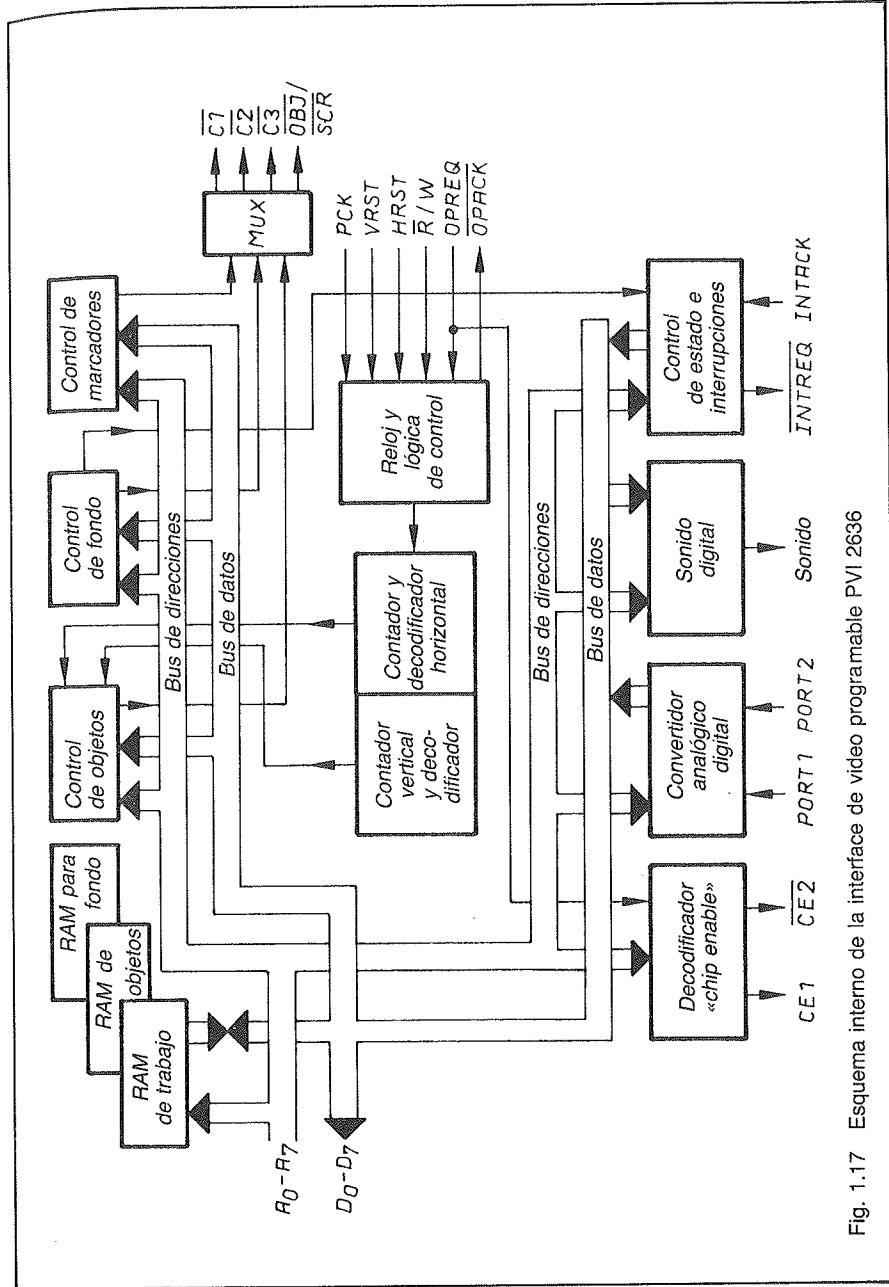


Fig. 1.17 Esquema interno de la interfaz de video programable PVI 2636

el USG 2621 y el modulador de color, consigue una frecuencia estable de servicio del cuarzo PAL, que oscila con una frecuencia de 8,86 MHz. La frecuencia ha de estar exactamente equilibrada a 8,867238 MHz \pm 20 Hz. No ha de medirse la frecuencia en un punto contiguo al cuarzo de PAL, sino en un punto de prueba que describiremos más adelante, con el fin de que la punta de prueba no la altere. Con algo de paciencia y habilidad no se necesita ningún frecuencímetro especial para el equilibrado. Nos bastará accionar un condensador variable giratorio hasta que aparezca una imagen correcta.

La frecuencia del sistema PAL de 8,86 MHz se reduce mediante el factor 2,5 y obtenemos 3,54 MHz, exactamente 3,546952 MHz. Con esta frecuencia se obtiene una resolución de líneas de aproximadamente 280 ns. Con el mismo cuarzo de PAL y mediante un simple divisor de frecuencias, generamos los 4,43 MHz (exactamente 4,433619 MHz) con los que se compone la señal FBAS completa. Para generar las distintas funciones necesitamos tan sólo unos pocos componentes de la serie TTL.

En la figura 1.17 se muestra las conexiones internas de la interface de video programable, PVI 2636. Gracias a los numerosos bloques funcionales internos podemos instalar PVI 2636 de forma universal para cualquier computador de pantalla. La PVI nos sirve para generar las señales que describirán la forma, tamaño, color y situación de cuatro objetos diferentes así como la situación de los duplicados de esos cuatro objetos. La interface de video programable nos sirve también para generar el aspecto y el color del escenario del juego así como el color del fondo («background»). Como escenario entendemos la delimitación de un campo de juego o los límites que suponen un obstáculo (nubes, campos de minas, bordillos, montañas, etc.).

Mediante la interface de video programable, obtenemos también nuestras señales acústicas de tono, la reproducción de colisiones entre dos objetos así como entre objeto y límites del escenario del jue-

go o el disparo y acierto en juegos de batallas de tanques o aéreas.

En la interface de video programable hay incorporados dos convertidores analógicos-digitales. Mediante ellos podemos consultar los potenciómetros de los jugadores y transformar el valor correspondiente analógico en una palabra digital de datos. Si simultáneamente tiene lugar el almacenamiento de la posición del potenciómetro dada por el jugador. Los potenciómetros se consultan cada 20 ms. De esta forma, podemos acelerar o frenar rápidamente los objetos.

El microprocesador 2650 obtiene sus datos para el cálculo del programa a partir de la interface de video programable. A tal fin, en la PVI se dispone de tres memorias de lectura-escritura (RAM). El 2650 podrá ejecutar su programa a partir de los distintos datos. Las RAM de PVI tienen una capacidad de memoria de 256 bits y de ellos, 37 se emplean para almacenamiento provisional de los datos. Los cuatro objetos posibles están contenidos en la parte de RAM.

La resolución del objeto es 280 ns. Mediante el programa en memoria puede controlarse y regularse el tamaño y situación de un objeto. Esos valores quedan depositados en la memoria de instrucciones del microprocesador. Igualmente, almacenamos en la ROM las señales de audio. Asimismo, en la memoria fija (ROM) está el color del fondo. Podemos elegir entre ocho colores con distintos grados de brillo y luminosidad.

El microprocesador 2650 y la memoria estática (ROM) están conectados con la interface de video mediante los buses de datos, de direcciones y de control. El 2650 lee los datos del juego programado a partir de la memoria estática e introduce los datos en la parte RAM de la interface de video. Esta parte de RAM de PVI consta de tres niveles. En el primer nivel tenemos la RAM de trabajo. Esta RAM la utiliza el microprocesador como memoria de trabajo para el almacenamiento provisional de datos e instruccio-

nes. El segundo nivel es la RAM objeto. Aquí se encuentran los datos para los objetos y mediante ellos el microprocesador identifica la dirección de dichos objetos en la pantalla. En el tercer nivel está la RAM de fondo. En ella están almacenados los colores y el escenario. Esta RAM sólo se modifica mediante la memoria de programas y el microprocesador al cambiar de juego. Después se mantiene constante el contenido.

La interface de video programable dispone de una memoria de trabajo que puede ser utilizada por el 2650 para almacenar provisionalmente datos e instrucciones. De esta forma, PVI recoge las informaciones de entrada y de estado en su parte-RAM, por ejemplo, informaciones relativas a colisión de objetos o la eliminación de uno de ellos. Después, el microprocesador obtiene esas informaciones de la RAM de trabajo de PVI, las procesa y toma decisiones de acuerdo con su programa.

En los sistemas orientados a objetos, éstos se componen de forma análoga a como sucedía en los sistemas RAM de regeneración de imagen. Sin embargo, ahora no necesitamos ningún canal de DMA para el acceso directo a la memoria. El almacenamiento de imagen tiene lugar, en este sistema, en la parte de RAM del componente 2636 que está dividido en tres niveles. Simultáneamente, se genera la señal de video a partir de la parte de RAM.

Mediante la interface de video programable, pueden resolverse fácilmente la obtención de movimientos rectilíneos, colisiones o eliminaciones de objetos y ampliaciones o reducciones de los mismos.

Mediante los potenciómetros externos de los jugadores pueden modificarse solamente las coordenadas de los objetos así como las instrucciones de control a PVI. De este modo se obtiene un ahorro considerable de tiempo en la ejecución del programa en todo el sistema. Dicho de otro modo, mediante el componente 2636 podemos realizar juegos de movimiento rápido sin que tengamos problemas a causa del tiempo de proceso. Ha de emplearse sólo el tipo

2650A-1 con el fin de conseguir un tiempo mínimo de máquina de 1,5 μ s. El tipo 2650-A, por el contrario, necesita 2,4 μ s. Puesto que en el computador de pantalla sólo están almacenadas las coordenadas de objetos y escenario del juego, no necesitamos ninguna memoria externa de lectura-escritura, como es el caso de los sistemas de regeneración de imagen de RAM. Además, el programa es bastante más sencillo y necesita sólo pocos pasos.

En el capítulo 3, en 2048 direcciones, se muestran no menos de 60 juegos de pelota. En esas 2048 direcciones pueden incorporarse muchos más juegos. Empleando una memoria de lectura-escritura en el marco del sistema del microprocesador, podemos incluso tener un ajedrez automatizado en el aparato de televisión. En ese juego, se han programado las instrucciones del ajedrez en 30 k-posiciones de memoria y en otras 2 k tiene lugar el almacenamiento provisional.

Las diferentes memorias RAM, unidades de control, decodificador, convertidor analógico-digital y los controles de interrupción y de estado se direccionan mediante los buses de direcciones A_0 a A_{11} . En la interface de video programable podemos llamar, en total, 4096 posiciones de memoria. La gama de direcciones, en sistema hexadecimal, está entre 000 y FFF:

000 $\hat{=}$ dirección 0

4096 lugares en la memoria

FFF $\hat{=}$ dirección 4095

Con el bus de datos conectamos el microprocesador con la memoria de programas y la interface de video. Dentro de PVI, los distintos bloques funcionales están interconectados entre sí. Es posible el intercambio de datos entre ellos, sin que haya que utilizar el 2650 como almacenamiento intermedio. Por tanto, dentro de la interface de video obtenemos un acceso directo a memoria. El programa, a partir de las informaciones introducidas, elabora los distintos elemen-

tos de control a los que se da salida a través de un multiplexor MUX.

Las salidas del multiplexor MUX son $\overline{C_1}$, $\overline{C_2}$, $\overline{C_3}$ y $\overline{OBJ/SCR}$. Las tres salidas \overline{C} controlan el sumador de video. A partir de esas tres salidas se forman en el sumador los distintos colores. La salida $\overline{OBJ/SCR}$ tiene un nivel bajo cuando el componente 2636 emite las señales de video a través de las tres salidas $\overline{C_1}$, $\overline{C_2}$ y $\overline{C_3}$. Pero podemos utilizar también esta salida para transferir un cuarto color al sumador de video. Sin embargo, de esa forma se complica la programación para el juego. En nuestro caso, empleamos la salida $\overline{OBJ/SCR}$ en conexión una lógica adicional que identifique colisiones de objetos.

El modo de trabajo de la interface de video programable, en unión del sistema microprocesador, puede controlarse adicionalmente por salidas CE_1 y CE_2 («Chip enable»). Con la salida CE_1 se tiene direccionamiento directo con la memoria de instrucciones y podemos bloquearla mediante un nivel bajo. Aunque esto sólo es preciso en programaciones muy complicadas como, por ejemplo, en el juego de ajedrez. Con la CE_2 bloqueamos el multiplexor analógico para las dos entradas del convertidor analógico-digital.

Con otras cinco entradas controlamos el temporizador y la lógica de control de la interface de video. De este bloque funcional, el contador y decodificador vertical, al igual que el contador y decodificador horizontal, obtienen sus impulsos para seguir la cuenta. La entrada PCK está unida al cuarzo de PAL y recibe de éste último la frecuencia de 3,56 MHz, con lo que obtenemos una resolución de línea de unos 280 ns. Así se tienen 227 impulsos por línea. Esta entrada se necesita para sincronizar el proceso de trabajo interno en el componente 2636. La entrada PCK está directamente conectada al componente USG 2621.

La entrada VRST corresponde a «Restart vertical» (reiniciación vertical) y junto con el componente USG 2621, está también conectada a la entrada

«Sense» del microprocesador. El componente USG genera una señal de retroceso vertical para sincronizar el contador vertical.

La entrada HRST es la «Horizontal restart» (reiniciación horizontal) y está conectado igualmente al componente USG 2621. Con ello, la interface de video programable recibe del USG una señal de retroceso horizontal para sincronizar el contador horizontal.

El componente USG 2621 aporta a nuestro juego la temporización correcta. La entrada HRST, cada 64 μ s, recibe del USG un impulso de sincronismo de una longitud de unos 11 μ s. El contador horizontal, al recibirla, se pone a 0. Despues de ello, el contador horizontal, con la frecuencia de 3,56 MHz, va contando en sentido creciente emitiendo los correspondientes impulsos con fines de control. Con el contador vertical se genera la frecuencia de cambio de imagen. Una vez que el contador horizontal ha efectuado el recuento de 0 a 227, se le pone a 0, a la vez que el contador vertical se incrementa en una unidad. De esta manera, el contador vertical pasa de 0 a 312 y así se introduce una imagen parcial con 312 líneas. Una imagen de pantalla completa consta, sin embargo, de 2 imágenes parciales.

Después de representar una imagen parcial, ambos contadores de deflexión vertical y horizontal de líneas se vuelven a reponer a cero para comenzar nuevamente un proceso de representación de imagen. De esta manera, se representan 50 imágenes parciales por segundo.

En el bloque funcional «temporizador y lógica de control» tenemos también la entrada R/W. Si hay un nivel bajo, los datos se leerán de PVI (read); si el nivel es alto, los datos se escribirán en PVI (write).

La entrada R/W está conectada, mediante el bus de control, al microprocesador 2650 y a las demás entradas de lectura-escritura del sistema. Como aquí se trata de un sistema sencillo, la línea de lectura-escritura sólo está conectada con el microprocesador y con el multiplexor analógico para los poten-

ciómetros. La entrada «OPREQ» (operation request) sirve sólo para la solicitud de operaciones para la interface de video y para el multiplexor analógico. El microprocesador 2650 tiene una salida «OPREQ». A través de esta línea de control, el microprocesador indica a sus unidades funcionales externas que es válida la información de las demás líneas de conexión, en este caso, del bus de datos o de direcciones. La entrada OPACK de la interface de video está conectada con la OPACK del microprocesador. La PVI notifica al microprocesador que está disponible. El 2650 acepta esa señal como confirmación y, sabe, así, que ha terminado una operación de almacenamiento o de entrada-salida.

El convertidor analógico-digital recibe sus informaciones a través de las dos entradas PORT1 y PORT2. Los potenciómetros para ambas entradas están entre la tensión de servicio $+U_b = 5\text{ V}$ y masa. La conexión del bucle está conectada directamente con un PORT (*) o bien, delante del PORT, tenemos un commutador analógico o un multiplexor analógico.

La salida «Sound» emite las diferentes frecuencias de tono del programa. Debido al modo de trabajo digital, las frecuencias de audio tienen una forma rectangular y se transforman, mediante una sección de audio aparte, en los distintos timbres del sonido.

Para la ejecución del programa necesitamos aún las conexiones INTREQ e INTACK entre el microprocesador y la interface de video. La salida INTREQ da al 2650 una solicitud de interrupción. Con ello, el 2650 queda incorporado por la PVI a la secuencia de interrupciones del programa. La entrada INTACK recibe la información del microprocesador mediante una señal y así sabe que ha terminado la secuencia de interrupciones de programa.

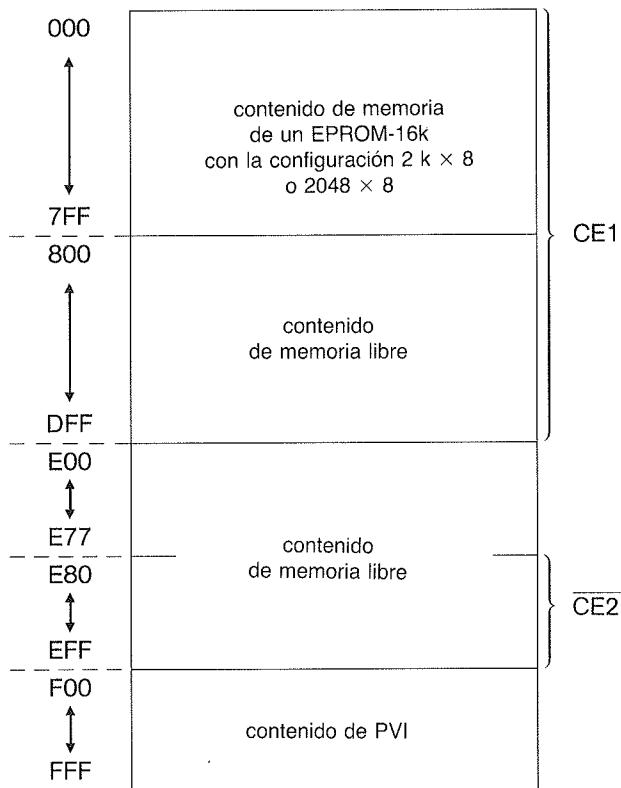
La zona de direcciones de la memoria de programa y de instrucciones abarca desde la 000 a la 7FF.

(*) N. del T.: «Port» es un conector de dispositivos de entrada/salida.

El direccionamiento se controla mediante la salida CE1 de la interface de video programable. De este modo, podemos elegir entre 2048 direcciones.

Entre las direcciones 800 y DFF disponemos de una capacidad de memoria adicional de 1536 direcciones. Las emplearemos bien para almacenar más instrucciones o programas o bien incorporaremos una memoria de lectura-escritura adicional. Esta zona de direcciones también queda controlada por la salida CE1 de la interface de video.

direcciones



La zona de direcciones desde la E80 a la EFF queda disponible a partir de la salida CE2 de la interface de video. Si entre las direcciones 800 y DFF tenemos más instrucciones o programas, utilizaremos la zona entre E80 y EFF como memoria de lectura-escritura. En caso contrario, se empleará esta zona para pequeños problemas de control como, por ejemplo, la conmutación del multiplexor analógico a PORT1 y PORT2.

Con las direcciones F00 a FFF podemos dirigir las 256 posiciones de memoria de la interface de video. El esquema de la página anterior nos muestra la estructura de RAM del componente 2636.

Tenemos cuatro campos objeto en la parte de RAM de la interface de video. Estos campos objeto son particularmente importantes e interesantes cuando estudiemos la programación en el capítulo 3.

En la figura 1.18 se muestra el esquema de conexiones de la interface de video programable 2636. Tenemos la siguiente disposición de conexiones:

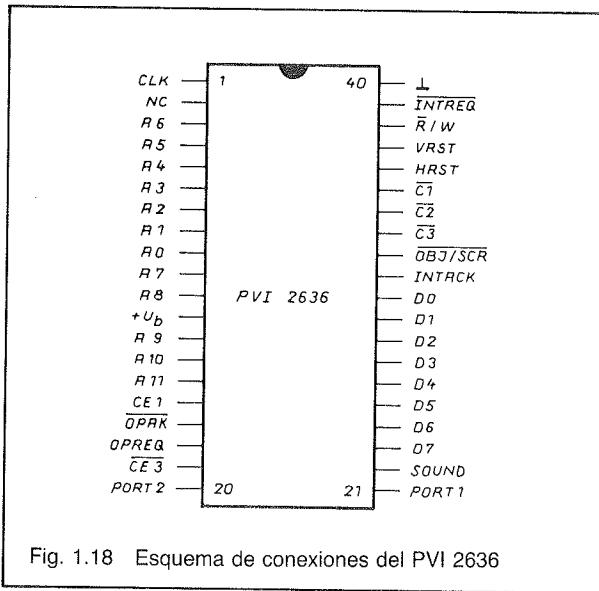


Fig. 1.18 Esquema de conexiones del PVI 2636

conexión	patilla	entrada/salida	función
A0 ... A11	3 ... 15	E	Bus de direcciones de 12 bits para tener un direccionamiento de 4 k direcciones.
D0 ... D7	23 ... 30	E/A	Bus de datos bidireccional de 8 bits.
OPREQ	18	E	Solicitud de operación. Si el nivel es alto son válidas todas las señales (estados de nivel) generadas por el microprocesador.
R/W	38	E	Entrada de lectura/escritura. Indica que PVI ha terminado un proceso de trabajo.
OPACK	17	A	Disponibilidad de operación. Indica al microprocesador que se necesita una interrupción en el programa.
INTREQ	39	A	Solicitud de interrupción. Indica al microprocesador que se necesita una interrupción en el programa.
INTACK	31	E	Confirmación de la solicitud de interrupción por el microprocesador.
PCK	1	E	Señal del cuarzo PAL. Esta entrada sirve para la sincronización de la ejecución interna en el PVI (3.54 MHz ≈ 280 ns produce 227 impulsos por línea).
C1, C2, C3	33 ... 36	A	Colores 1, 2 y 3. Con estas tres señales se indican los colores al sumador de video.
VRST	37	E	Señal de retroceso vertical generada por el USG para sincronizar el contador vertical de PVI.
HRST	36	E	Señal de retroceso horizontal generada por el USG para sincronizar el contador horizontal.
CE1, CE2	16, 19	A	Chip disponible o salida desbloqueada. Interface para la memoria de programa o de instrucciones perteneciente al sistema.
PORT1, PORT2	21, 20	E	Conexiones analógicas para los potenciómetros.
SOUND	22	A	Salida rectangular para la parte de audio.
OBJ/SCR	32	A	Si PVI emite señales de video para retículo y objetivo, a través de C1, C2 y C3, esta salida tiene nivel bajo. En general, se emplea para transferir un cuarto color al sumador de video. También, mediante una lógica adicional, puede emplearse para identificar colisiones entre objetos en aquellos sistemas que dan servicio a varios PVI.

Conexión	patilla	entrada/ salida	función
NC	2		No conectado
+U _b	12		Tensión de alimentación
—	40		Masa, 0V

direcciones función

F00	Campo objeto 1
FOE	Memoria intermedia de 2 bits
F10	Campo objeto 2
F1E	Memoria intermedia de 2 bits
F20	Campo objeto 3
F2E	Libre
F40	Campo objeto 4
F4E	Memoria intermedia de 32 bits
F6E	Libre
F80	Barras verticales de fondo 40 bits
FA8	Barras horizontales de fondo 5 bits
FAD	Memoria intermedia de 1 bit
FAE	Libre
FC0	Entrada/salida y controles
FDO	Entrada/salida y controles
FE0	Entrada/salida y controles
FF0	Entrada/salida y controles
FFF	Entrada/salida y controles

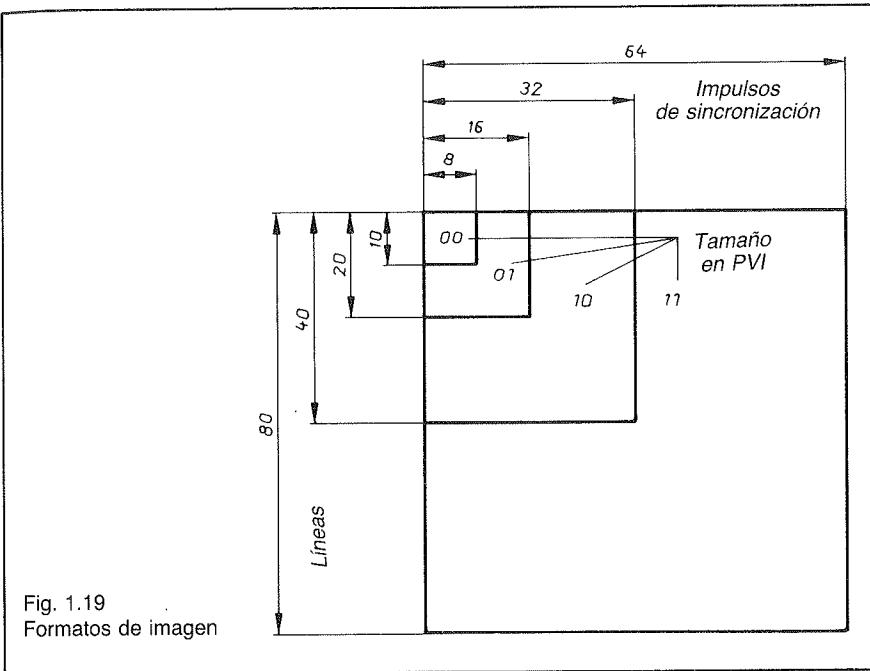
Un campo objeto se compone del modo siguiente:

		Bit	R/W	Descripción
Byte	7	6	5	4
FC0	Tamaño 4	Tamaño 3	Tamaño 2	Tamaño 1
FC1	C1 C2 Color 1	C3 C1 C2 C3	Colör 2	Escritura
FC2	C1 C2 Color 3	C3 C1 C2 C3	Color 4	Escritura
FC3	Forma	Posic.	Escritura	Formato de indicador visual y posición
FC4				Libre
FC5				
FC6	C1 C2 C3 BG Color de fondo	Color de pantalla Dispon.	C1 C2 C3	Escritura
FC7	Sonido		Escritura	Barraera de fondo y color
FC8	N1	N2	Escritura	Salida rectangular
FC9	N3	N4	Escritura	Zona de los cuatro dígitos indicadores
FCA	O Objeto/fondo	Objeto completo	Lectura	Composición de objeto y fondo, identificación de colisiones e indicador de objeto para componer un estado que irá a un registro de estado. Activar VRLE y demorar VRST. Leer o transferir todos los bits, hasta reposición a 0 por el VRST.
FCB	VRLE	1/2 1/3 1/3 1/4 2/4 3/4	Colisiones de objetos	
FCC	PORT1	PORT2	Lectura	PORT1 y PORT2 para la zona de conversión analógico/digital. Se borra por el VFSI.
PCD				Libre
FCE				
FCF				

Este esquema se aplica a los cuatro campos objeto, es decir, la estructura de esos campos es la misma. Con las 10 líneas podemos determinar el aspecto del objeto. Ya explicaremos más adelante la programación del aspecto de un objeto. Los cuatro bits restantes sirven para registrar las coordenadas de los objetos. Además, tenemos registros para sus duplicados.

En el marco del direccionamiento de la interface de video hay previstos, en otro lugar de la parte RAM, otros 2 bits para el tamaño y 3 bits para el color de cada objeto respectivamente. Esta programación puede hacerse en las cuatro zonas correspondientes a los campos de entrada-salida y de control del esquema anterior.

En la figura 1.19 se muestra las superficies disponibles para representar un objeto. Cada uno de ellos puede programarse en cuatro tamaños diferentes:



tes. Si elegimos el más pequeño, de los 227 impulsos de cada línea de pantalla, nos queda un máximo de 8 impulsos para el objeto. En dirección vertical abarcan 10 líneas de pantalla. Obtenemos una matriz de imagen de 8×10 .

En el bit FCO podemos establecer las superficies del siguiente modo:

Bit		Matriz
0	0	8×10
0	1	16×20
1	0	32×40
1	1	64×80

El tamaño del objeto queda determinado por la codificación de bits que se anota en el ángulo inferior izquierdo de cada campo objeto en la figura 1.19. Puesto que en la dirección FCO podemos componer cuatro objetos distintos y tenemos cuatro posibilidades de dirección en la interfaz de video, pueden programarse también colisiones empleando otros símbolos. De este modo, podemos simbolizar el disparo de un tanque o de un avión mediante «trozos» en movimiento a su alrededor. En este punto, no hay barreras a la imaginación del programador.

Los tamaños de objeto pueden almacenarse en las direcciones siguientes:

FCO, FDO, FEO y FFO

En total, tenemos 16 campos de objeto para los cuatro tamaños.

Designamos como tamaño de objeto la configuración, en uno de los cuatro campos de objeto, de los 10 bits que tenemos disponibles para describir su aspecto. Cada bit representa la señal de video para una línea, si elegimos el tamaño «00». Obtenemos 8 impulsos verticales y 10 líneas.

Observemos, a este respecto, las dos figuras 1.20 y 1.21. vemos la misma matriz pero con tamaño

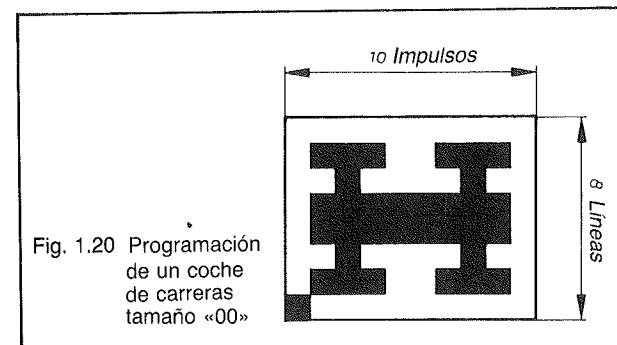


Fig. 1.20 Programación de un coche de carreras tamaño «00»

distinto. La figura 1.20 nos muestra el aspecto de un pequeño coche de carreras a tamaño «00». Un bit (abajo a la derecha) sirve como punto de referencia y a cada bit corresponde un impulso en una línea de pantalla.

La figura 1.20 puede ampliarse simplemente si en lugar del tamaño «00» programamos «01». Con ello, para cada bit, tendremos dos impulsos y dos líneas de pantalla. Así lo estableceremos mediante la programación. Para la figura 1.20 necesitamos programar el campo objeto tal como se muestra en la tabla adjunta.

BITS	BIT							
	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0
1	0	1	0	1	1	0	1	0
2	0	1	1	1	1	1	1	0
3	0	1	0	1	1	0	1	0
4	0	0	0	1	1	0	0	0
5	0	0	0	1	1	0	0	0
6	0	1	0	1	1	0	1	0
7	0	1	1	1	1	1	1	0
8	0	1	0	1	1	0	1	0
9	0	0	0	0	0	0	0	1

Punto de referencia

Cada señal 1 significa que la interface de video emite una señal de video y aparece un punto en la pantalla, si hemos programado el tamaño «00». Cada señal 0 significa una posición en blanco en la pantalla.

En la figura 1.21 se muestra cómo con el dato del tamaño podemos aumentar al doble el coche de carreras en la figura 1.20. Mediante la programación puede aumentarse el tamaño a cuatro u ocho veces. Así, tenemos innumerables posibilidades de variación en nuestro programa. La situación de cada objeto lo conservamos en los registros verticales y horizontales. Existen respectivamente cuatro registros verticales y cuatro horizontales y también cuatro registros de duplicados. Puesto que, en cada registro de 1 bit, tenemos 256 posibles combinaciones que podemos almacenar, quiere decir que podemos diseñar exactamente un objeto en la pantalla y programarle adecuadamente. En estos registros se establece el número de impulsos o líneas que han de saltarse antes de dar salida a la señal de video de un

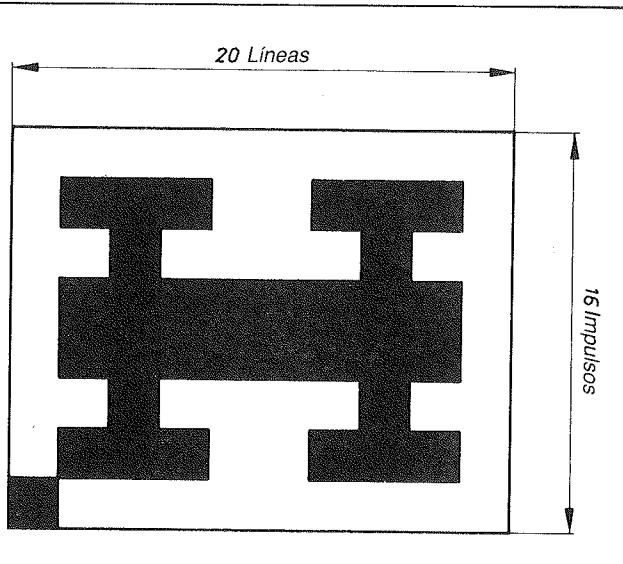


Fig. 1.21
Programación
de un coche de carre-
tas tamaño «01»

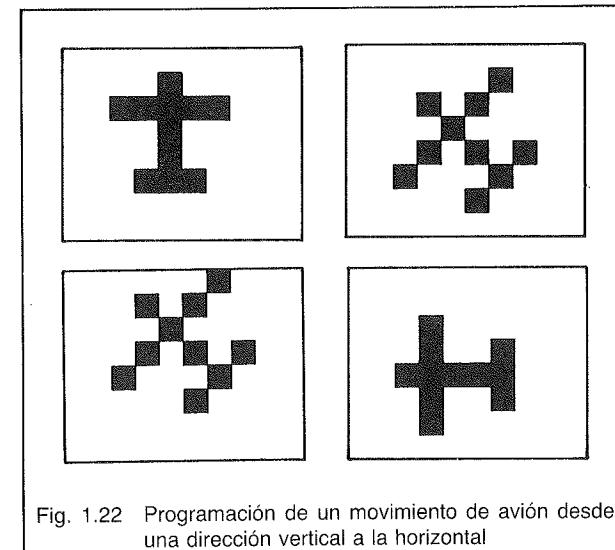


Fig. 1.22 Programación de un movimiento de avión desde una dirección vertical a la horizontal

objeto. Los registros se encuentran en conexión directa con el «Horizontal restart» y el «Vertical restart». Como, por otra parte, toda la lógica está controlada por una frecuencia del oscilador de cuarzo, obtenemos siempre una imagen exacta y estable. Si se produce un error, se corregirá durante el tiempo de formación de una imagen.

El microprocesador puede conseguir los movimientos de los distintos objetos incrementando o decrementando el contenido de los registros horizontales y verticales. Mediante las distintas funciones de la interface de video programable y las posibilidades de direccionamiento del microprocesador 2650, tenemos dos grados de libertad para cada objeto, es decir, podemos desplazarlos vertical y horizontalmente. Pero si han de desarrollarse funciones de rotación de un objeto, hemos de alterar el contenido del campo del objeto. Esto plantea problemas en algunos casos, por ejemplo, si un avión ha de describir un arco de curva (trayectoria sinuosa).

La figura 1.22 nos muestra para un avión las etapas de una trayectoria en forma de arco. En la parte

superior izquierda vemos un avión vertical. Mediante la programación y la referencia del potenciómetro podemos mover el avión sobre la pantalla lenta, o rápidamente, tanto tiempo como dure la tensión positiva que el potenciómetro vertical proporciona a la interface de video programable. Si reducimos la tensión, el avión volará muy despacio. Obtenemos así un movimiento de abajo a arriba.

Con una trayectoria lenta de abajo a arriba, ponemos el potenciómetro vertical a la tensión máxima. El microprocesador, mediante la referencia de los potenciómetros, obtiene nuevas coordenadas del convertidor analógico-digital y comienza el cálculo de la curva de vuelo. El avión describe, entonces, en la pantalla un arco pequeño y después sigue volando en dirección horizontal. Aparece una aceleración si volamos exactamente horizontal.

Mediante los potenciómetros horizontal y vertical, podemos hacer volar al avión alrededor de un punto, si fijamos mediante ambos una tensión de salida muy pequeña. Pero si giramos ambos potenciómetros a la tensión máxima obtenemos una curva descrita a gran velocidad y de radio grande.

Por efecto del arco de curva se produce una deformación del símbolo, pues, a fin de cuentas, en el juego sólo tenemos una descomposición de puntos. En los dibujos de la figura 1.22 queda indicada la deformación. Transcurrido aproximadamente un segundo, el avión sigue volando en dirección horizontal.

Una línea de pantalla de nuestro computador de televisión consta de 227 impulsos. Si disponemos de un tubo de rayos catódicos con una anchura de unos 50 cm, cada impulso tendrá una longitud de 0,22 mm. Por tanto, nuestro avión de la figura 1.22 tendrá una magnitud de 1,1 mm. Modificando el tamaño, mediante la combinación de bits para determinar la superficie, podemos representar el avión con magnitudes de 2,2 mm, 4,4 mm o 8,8 mm, sin necesidad de modificar el campo de objeto.

Como ya hemos visto al tratar de la interface de

video programable, podemos representar en la pantalla un duplicado, o más, de cada objeto activando convenientemente los campos HCB y VCB de la memoria RAM. En el HCB (coordenadas horizontales del duplicado) prefijamos a qué distancia del borde izquierdo de la pantalla (cantidad de impulsos) ha de empezar la representación del duplicado. En el VCB (coordenadas verticales del duplicado) prefijamos cuántas líneas han de saltarse por debajo del objeto o del duplicado.

La figura 1.23 indica la relación entre el objeto y los duplicados. El objeto 1 tiene almacenadas las coordenadas HC = 40 y VC = 35. La designación 40H se aplica para H = horizontal y 35V para V = vertical. Esas coordenadas determinan siempre el punto superior izquierdo del objeto. El objeto 2 tiene como coordenadas HC = 90 y VC = 30. Estas coordenadas están siempre almacenadas en HC y VC del campo objeto.

En el campo objeto tenemos también almacenadas las direcciones (impulsos y líneas) para los duplicados. El duplicado 1 del objeto 1 tiene las coordenadas HNC = 29 y VCB = 9. El duplicado tiene su punto superior izquierdo en el impulso de reloj trigésimo (29H) de la línea 60. Con el registro HCB prefiguemos el número de impulsos más 1.. El registro VCB nos da la distancia entre objeto y duplicado o entre duplicado y duplicado. La distancia entre líneas propiamente dicha es VCB + 1.

Para el objeto 1 tenemos HCR = 29 y VCR = 9 en los registros de duplicado. Puesto que hemos de sumar 1, obtendremos como siguiente punto de coordenadas el 30H y 60V. Entre el objeto 1 y el duplicado 1 tenemos una distancia de 10 líneas, estando desplazado el duplicado 5 impulsos a la izquierda en 30H.

El siguiente duplicado aparecerá representado en el punto de coordenadas 30H y 85V. De este modo, obtenemos 10 duplicados del objeto 1.

Para el objeto 2 tenemos, en los dos registros de duplicación, los valores HCB = 109 y VCB = 34. Por

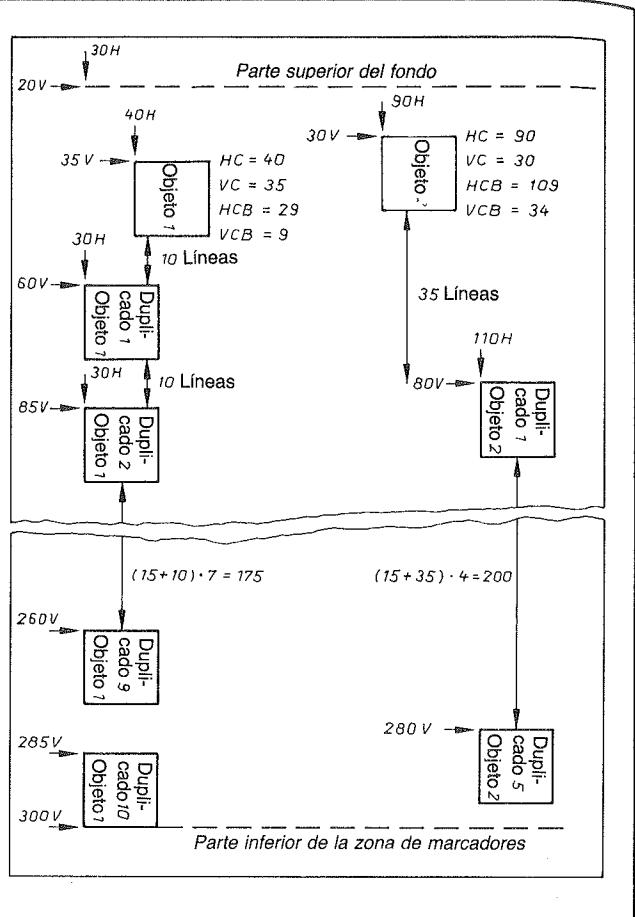


Fig. 1.23
Objetos y duplicados

tanto, el duplicado 1 del objeto 2 se situará en el punto de coordenadas 110H y 80V. Existe una distancia de 35 líneas entre el objeto 2 y su duplicado. Los duplicados siguientes se obtienen a partir del punto de coordenadas 110H y 130V, así pues en los 110H y 180V, 110H y 230H, 110H y 280V. A partir de la línea 300 no representamos ningún duplicado más.

Sin necesidad de componentes lógicos adicionales, podemos conseguir que la distancia entre objeto

y duplicado 1 sea la misma que entre los duplicados 1 y 2, 2 y 3, etc. No obstante, mediante la programación, podemos alterar convenientemente esta disposición rígida. No hay posibilidad de solape entre objeto y duplicado, ni tampoco entre duplicado y duplicado.

Como ya hemos visto, las señales de video para un objeto sólo se representan en la pantalla de televisión, cuando para él hemos programado una señal 1. Para tal señal tenemos disponibles en las salidas, C1, C2 y C3 las combinaciones cromáticas para ese objeto. En ese momento, el sumador de video realiza una adición de las tres señales para obtener un impulso FBAS. Pero si, por el contrario, para el objeto hemos programado una señal 0, aparecerán los colores del fondo, los del escenario del juego o los de otro objeto en las tres salidas de color de la interface de video.

Para generar las señales de video del escenario del juego, están previstos cuatro grupos de datos en la RAM de la interface de video:

Generación de barras verticales:	320 bits
Generación de barras horizontales:	80 bits
Disponibilidad de escenario de juego y fondo:	1 bit
Escenario del juego y color de fondo: Dos grupos de 3 bits cada uno	

Estos datos se almacenan en la memoria de programa e instrucciones del sistema microprocesador. El microprocesador los transferirá y almacenará en la parte de RAM de la interface de video programable.

En el juego televisivo podemos representar hasta 320 barras verticales, es decir, 320 líneas de televisión individuales. Estas se agrupan en 10 series adyacentes, una debajo de otras, de 16 pares de barras cada una. Cada par tiene, por tanto, 160 líneas y en total 320 barras. Cada par de barras tiene un ancho de un impulso, resultando una longitud de impulso de 280 ns debido al cuarzo. Este par de barras

está separado del siguiente en 7 impulsos y consta de barras de 2 líneas de altura, así como de barras de 18 líneas de altura. La cantidad máxima de barras representadas las podemos incluir en una superficie de pantalla de 128 impulsos con 200 líneas cada uno para formar una media imagen.

La figura 1.24 nos muestra las barras verticales en la pantalla indicando las posiciones de memoria en la RAM de video. De las 10 posibles series de 2 y 18 líneas, se representan en la figura sólo las dos series primeras. En dirección horizontal podemos programar, en la memoria de instrucciones, 16 barras de cada una de ellas.

Con los 40 bits de la parte de RAM en la interfaz de video prefijamos qué barras han de representarse. El contenido de estos bits proporciona las 10 series, adjudicándose a cada una 32 bits. Por tanto, para cada par de barras necesitamos 2 bits, estando previsto uno de cada 4 bits para una serie.

Las barras horizontales de la figura 1.24 se generan simplemente ensanchando en dirección horizontal cada una de las barras verticales. Como ya hemos visto, las barras verticales prefijadas con los 40 bits, tienen una anchura inicial de 1 impulso. Pero para la programación existen 5 bits adicionales desde la dirección FAB a la FAD, con el fin de obtener barras con anchuras de 2, 4 u 8 impulsos. A cada uno de estos 5 bits adicionales se asocian dos series, estando destinados los bits 6 y 7 de cada uno de ellos para indicar si la representación ha de hacerse con una anchura de 1, 2 o 4 impulsos. En la figura 1.24, mediante los bits 0 al 5, se establece una subdivisión de la dos series en 6 grupos horizontales. Si mediante la programación se ponen a «1» los bits 0 a 5 con una señal 1, todas las barras verticales del grupo asociado serán de 8 impulsos de anchura. Si éstos reciben una señal 0, la anchura se obtendrá de lo que indiquen los bits 6 y 7, como veremos en un ejemplo. La programación de este proceso se almacena en las direcciones entre FCO y FCF.

La figura 1.25 nos muestra la estructura comple-

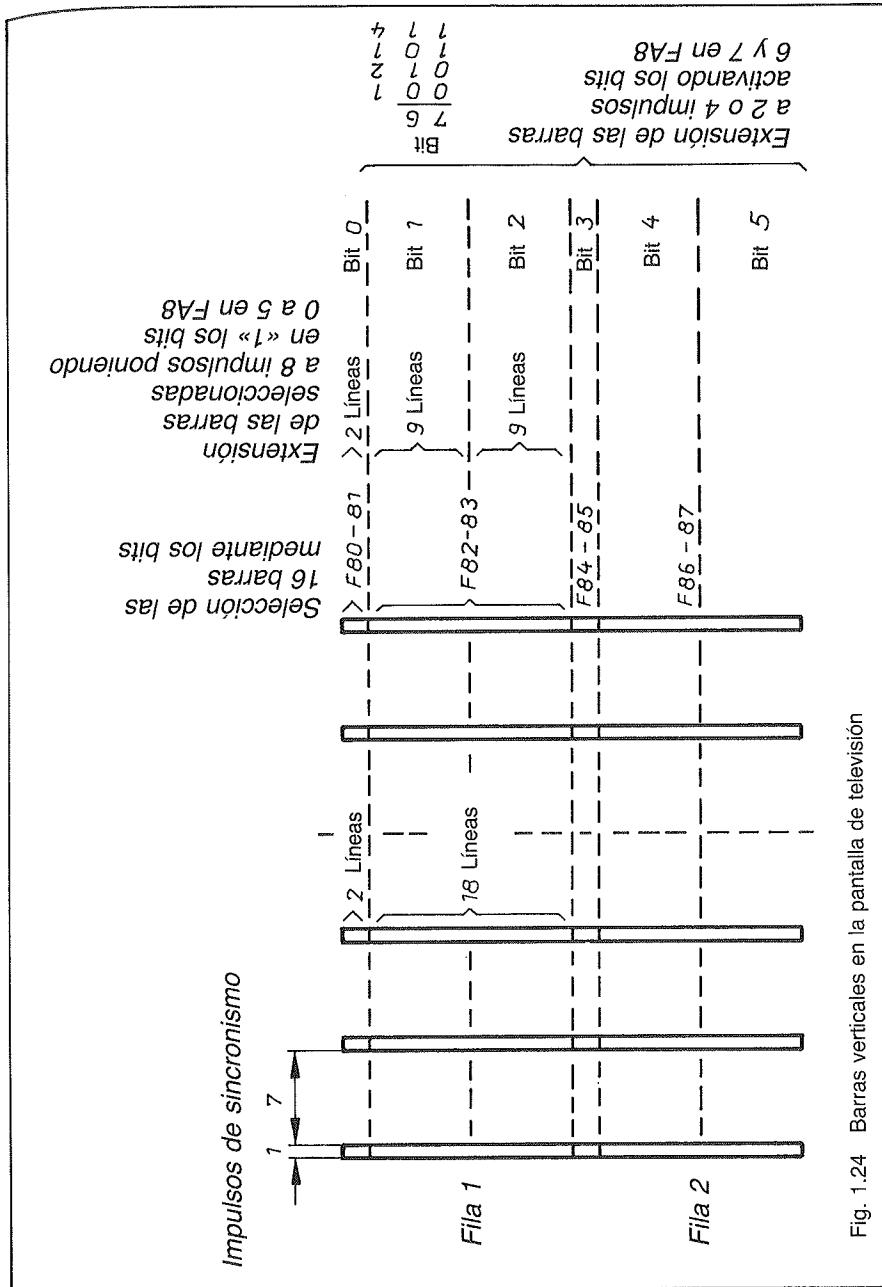


Fig. 1.24 Barras verticales en la pantalla de televisión

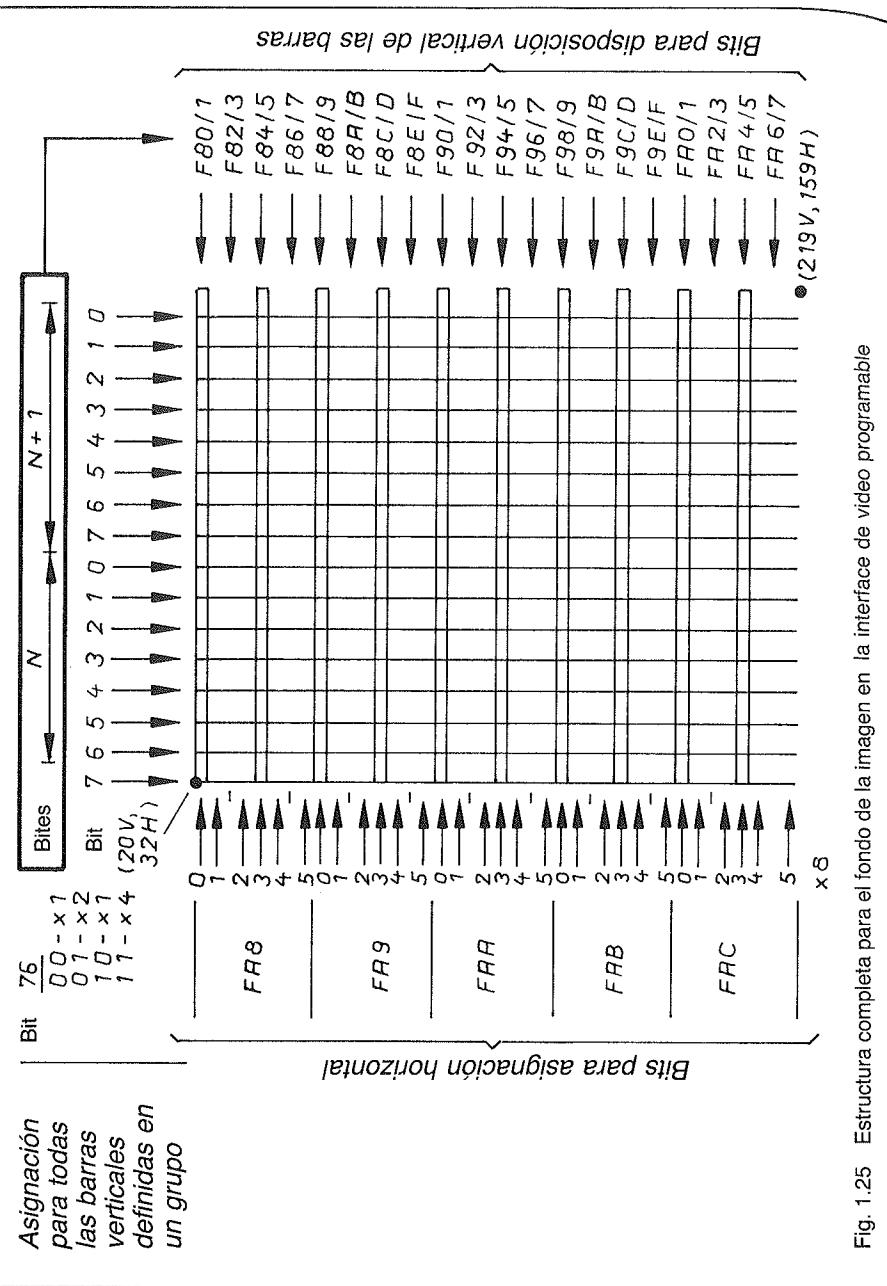


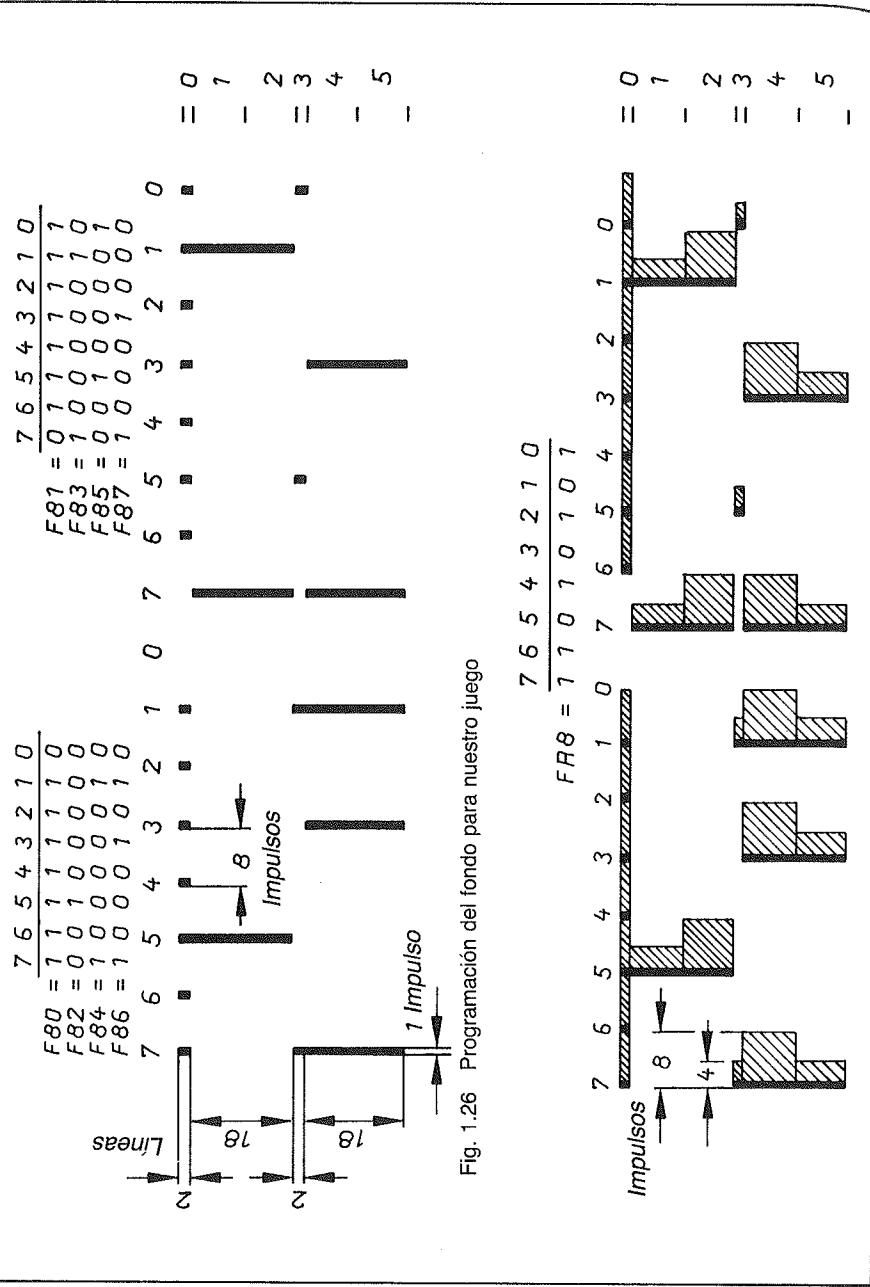
Fig. 1.25 Estructura completa para el fondo de la imagen en la interfaz de video programable

ta para el fondo de la imagen de pantalla en la interface de video. Podemos programar, así, hasta 360 barras verticales. Sin embargo, en nuestro sistema de computador de video, agrupamos siempre dos de estas barras para formar un par. De esta forma, se obtendrán más tarde en la programación 10 series de 16 pares de barras cada una y situadas unas debajo de las otras. En nuestra figura, cada par de barras tiene una anchura de un impulso, es decir, 280 ns. Para la generación de impulsos, necesitamos los ya mencionados 3,54 MHz para su control. Cada dos pares de barras están separadas por un intervalo de 7 impulsos.

Las barras en la interface de video programable quedan definidas mediante los 40 bits de la dirección F80 a la FA7. La configuración de estos bits reproduce nuevamente las 10 series con 32 bits cada una. Las barras horizontales se logran estirando las barras verticales en dirección horizontal. Las barras definidas con los 40 bits tienen una anchura inicial de 1 impulso. Con los cinco bits adicionales podemos hacer que las barras verticales tengan una anchura de 2, 4 o 6 impulsos. Estos 5 bits serán ordenados en dos series que aparecen a la izquierda de la figura 1.25. De ello se deducen las siguientes direcciones a tener en cuenta para la definición:

- aceptar FA8 y poner a «1» de 1 a 4
- aceptar FA9 y poner a «1» de 5 a 18
- aceptar FAA y poner a «1» de 9 a 12
- aceptar FAB y poner a «1» de 13 a 16
- aceptar FAC y poner a «1» de 17 a 20

De este modo, en la figura 1.25 obtenemos las 10 barras horizontales para la pantalla. Con los bits 6 y 7, en la dirección respectiva, definiremos si la interface de video ha de representar barras con una anchura de 1, 2 o 4 impulsos. Para la imagen tenemos la tabla siguiente:



Bit 7	Bit 6	Impulsos/barra
0	0	1
0	1	2
1	0	1
1	1	4

La figura 1.26 nos muestra un ejemplo de programación del fondo para nuestro juego televisivo. En los bits 0 a 5 tenemos subdivididas las dos series en 6 grupos horizontales. Si, mediante la programación, ponemos a «1» con la señal 1 uno de los bits 0 a 5, todas las barras verticales del grupo asociado tendrán una anchura de 8 impulsos. Si la señal es 0, la anchura depende de los bits 6 y 7. En nuestro ejemplo tenemos las funciones siguientes:

Bit	Función
0	Alargar las barras como corresponda, de 1 a 8 impulsos.
1	Alargar las 9 líneas superiores como corresponda, de 2 a 8 impulsos.
2	Alargar las 9 líneas inferiores como corresponda, de 2 a 8 impulsos.
3	Alargar las barras de 3 a 8 impulsos según los bits activados.
4	Alargar las líneas superiores de 4 a 8 impulsos según los bits activados.
5	Alargar las 9 líneas inferiores de 4 a 8 impulsos, según corresponda.
6, 7	Alargar las barras verticales seleccionadas conforme a la codificación, hasta 1, 2 o 4 impulsos.

Examinando la figura 1.26 distinguimos entre la definición de las barras horizontales y verticales. El primer par de barras en nuestra programación del fondo queda definido por las direcciones F80 y F81. Estas direcciones tienen los valores siguientes:

F80 =

1	1	1	1	1	1	1	0
---	---	---	---	---	---	---	---

F81 =

0	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---

Con cada señal 1 escribimos inicialmente un punto en la definición de barras horizontales. Con la señal 0, la pantalla permanece oscura. Así obtenemos las dos primeras líneas de pantalla, que hemos reunido en un par de barras.

Después del primer par de barras, o líneas, vienen las direcciones F82 y F83 para las 18 líneas siguientes. Estas direcciones tienen los valores siguientes:

F82 =

0	0	1	0	0	0	0	0
---	---	---	---	---	---	---	---

F83 =

1	0	0	0	0	0	1	0
---	---	---	---	---	---	---	---

Cada señal 1 siempre significa un punto para las 18 líneas de pantalla. De este modo, en la figura 1.26, obtenemos una barra vertical.

Entre las otras 18 líneas tenemos aún otro par de barras que podemos definir con las direcciones F83 y F85. Las direcciones tienen los valores siguientes:

F84 =

1	0	0	0	0	0	1	0
---	---	---	---	---	---	---	---

F85 =

0	0	1	0	0	0	0	1
---	---	---	---	---	---	---	---

Con cada señal 1, la interface de video programable escribe un punto de dos líneas en la pantalla.

Tras esta línea intermedia con el par de barras, siguen las 16 líneas de pantalla inmediatas y que están definidas por las direcciones F86 y F87. Las direcciones tienen los contenidos siguientes:

F86 =

1	0	0	0	1	0	1	0
---	---	---	---	---	---	---	---

F87 =

1	0	0	0	1	0	0	0
---	---	---	---	---	---	---	---

Cada señal 1 genera un punto en la pantalla. Debido a la forma en que se escriben horizontalmente los puntos, se obtiene una barra vertical, según indica la figura 1.26.

Con ayuda de la programación de nuestra memoria de instrucciones y de programa, podemos obtener, mediante las direcciones horizontales, un número cualquiera de puntos. Lo esencial es que después de un bloque de líneas correspondiente a un par de barras, tengamos siempre nueve pares de barras, y así sucesivamente. No podemos salirnos de este esquema de trabajo durante la programación, puesto que, de lo contrario, no funcionará correctamente la interface de video.

La definición de barra vertical, en la figura 1.26, tiene la dirección FA8 y contiene los valores siguientes:

FA8 =

1	1	0	1	0	1	0	1
---	---	---	---	---	---	---	---

I II

La dirección está dividida en dos segmentos. Con el segmento I se definen 1, 2 o 4 impulsos.

El segmento II identifica la definición de cada segmento. El bit 0 en la dirección FA8 extiende a exactamente 8 impulsos los puntos del primer par de barras, puesto que tenemos una señal 1. Si fuese una señal 0, la extensión sería a 4 impulsos. Por ejemplo, tenemos esta señal 0 en el bit 1 de la dirección FA8.

Según se deduce de la figura 1.26, el bloque de 18 líneas se ha subdividido en dos subbloques de 9 líneas cada uno atendiendo a la extensión de impulsos. El bloque superior con 9 líneas se ensancha a 4 impulsos, puesto que hay una señal 0 en el bit 1 de la dirección FA8. El bloque inferior, por el contrario, se ensancha a 8 impulsos, porque en el bit 2 de la dirección FA8 hemos programado una señal 1. Del mismo modo, obtendremos la extensión de impulsos de las demás direcciones horizontales de la figura 1.26.

En la figura 1.27 se indica una programación completa para una imagen de pantalla. En su parte izquierda tenemos los 5 bits para la prolongación de puntos y en su parte derecha los 40 bits para la programación de dichos puntos.

Para el bloque con dirección FA8 hemos codificado el valor siguiente:

$$FA8 = C2 \triangleq \begin{array}{|c|c|c|c|c|c|c|c|} \hline 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline \text{C} & & & & & & & \\ \hline \text{2} & & & & & & & \\ \hline \end{array}$$

Cuando en los bits del 0 al 5 tenemos una señal 0, obtendremos una extensión de 4 impulsos o bien una extensión de 8 impulsos, si en ellos hay una señal 1. Esto queda especificado mediante el segmento I de la dirección FA8.

Con esa dirección están asociadas las F80, F81, F82, F83, F84, F85, F86 y F87. Codificamos esas direcciones con los valores siguientes:

$$F80 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & & & & & & & & \\ \hline \end{array}$$

$$F81 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & & & & & & & & \\ \hline \end{array}$$

$$F82 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline \text{F} & & & & & & & & \\ \hline \text{F} & & & & & & & & \\ \hline \end{array}$$

$$F83 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ \hline \text{F} & & & & & & & \\ \hline \text{E} & & & & & & & \\ \hline \end{array}$$

$$F84 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline \text{F} & & & & & & & & \\ \hline \text{F} & & & & & & & & \\ \hline \end{array}$$

$$F85 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ \hline \text{F} & & & & & & & \\ \hline \text{E} & & & & & & & \\ \hline \end{array}$$

$$F86 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 8 & & & & & & & & \\ \hline 0 & & & & & & & & \\ \hline \end{array}$$

$$F87 = \begin{array}{|c|c|c|c|c|c|c|c|c|} \hline 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & & & & & & & \\ \hline 2 & & & & & & & \\ \hline \end{array}$$

Mediante esa programación, obtenemos las 40 primeras líneas de las 312. En la figura 1.27 se indica la programación de las restantes. Han de codificarse en el sistema binario las cantidades hexadecimales indicadas en la figura cuando se vaya a realizar la programación de la memoria de instrucciones.

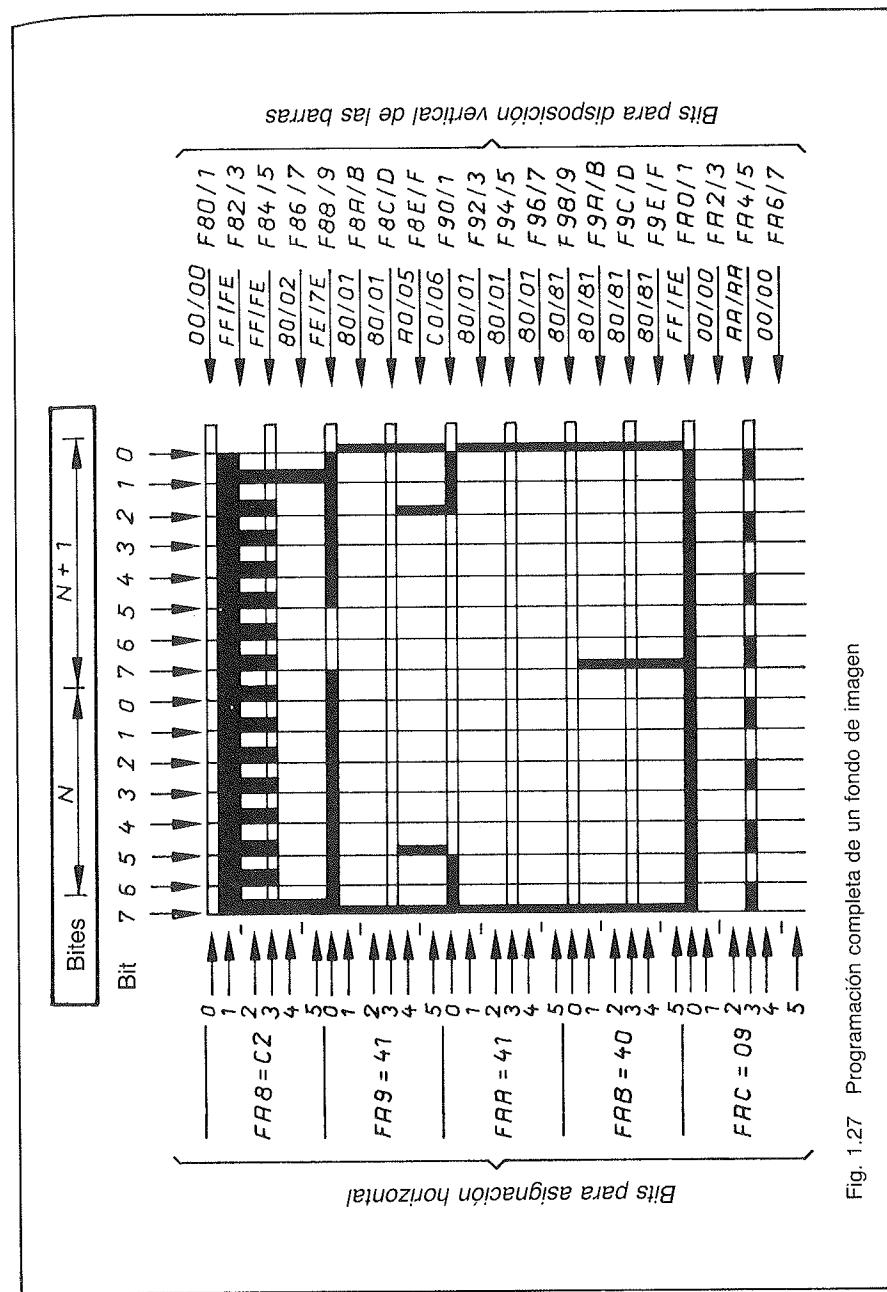


Fig. 1.27 Programación completa de un fondo de imagen

Para nuestro juego televisivo se ha prefijado ya el marcador del juego, mediante una posición de memoria en la interfase de video. En el bit 1 de la dirección FC3 establecemos la forma de nuestro marcador. Si en la programación hay una señal 0, tenemos un espacio intermedio de 20 impulsos, como muestra la figura 1.28. De esta manera, tenemos dos marcadores separados y podemos contar en cada uno desde 0 hasta 99. emplearemos esta solución cuando sean dos los jugadores. Por el contrario, si la señal es 1, no tenemos espacio intermedio y podremos contar desde 0 hasta 9999. Esto se empleará cuando el juego sea de un solo jugador.

Mediante el bit 0 de la dirección FC3 establecemos la situación de nuestro marcador. Con una señal 0, el marcador de datos comienza entre la línea 0 y la 19 del fondo de imagen. Si, por el contrario, programamos una señal 1 comenzará entre las líneas 180 y 199.

Los datos pueden introducirse y almacenarse en las direcciones FC8 y FC9 de la interfase de video. Las cifras posibles se representan en la figura 1.29. Cuando se trata de pseudocifras, es decir, cifras de la A a la F, el marcador permanece oscuro. Para el componente 2636 se produce un oscurecimiento automático. El aspecto de las cifras representadas es el de símbolos de 7 segmentos. Cada uno de ellos se encuentra en una matriz de 20 líneas con 12 impulsos cada una. El color de las cifras corresponde siempre al del fondo.

Los colores del escenario del juego y del fondo se establecen cada uno mediante 3 bits en el control de dicho escenario. La elaboración de este último se activa con el bit «enable» (activación de bit) en la dirección FC6. Si esa activación no tiene lugar, la señal de salida 111 aparecerá tanto para el color del escenario del juego como para el de fondo.

El bit VRLE en la dirección FCB es el bit de estado de retroceso vertical. Si el bit está activado (puesto en «1»), cuando se lea la dirección FCB se producirá un borrado automático.

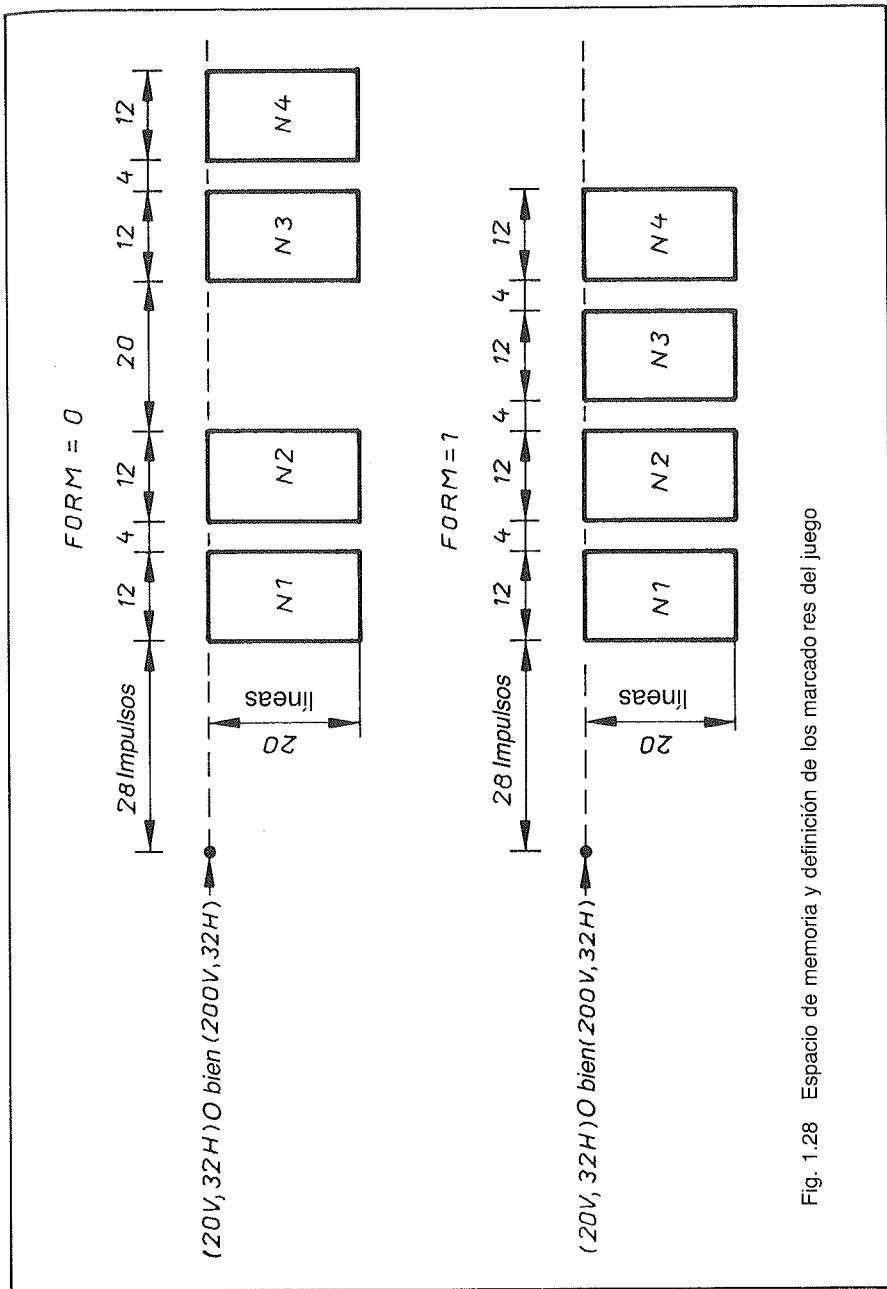


Fig. 1.28 Espacio de memoria y definición de los marcadores del juego

La secuencia de tonos en la salida «sound» depende de la programación. Se obtienen impulsos rectangulares con una frecuencia entre 30 Hz y 4 kHz.

La interface de video programable tiene dos convertidores analógico-digitales internos. Con ellos se convierte las señales de entrada analógicas, procedentes de las entradas de PORT, en valores digitales. El almacenamiento de ellos tiene lugar en las direcciones FCC y FCD del componente 2636. La gama de valores está entre 20 y 255. Mediante el 2650 podremos consultar ese valor y después procesarlo. Todo esto compete al programa para nuestro juego televisivo.

3 GENERADOR DE SÍNCRONISMO

Al ocuparnos del generador de sincronismo para nuestro juego de televisión, hemos de distinguir entre el sistema PAL y el NTSC. El primero se emplea en la República Alemana y el segundo en Estados Unidos.

El sistema NTSC (National Television System Committee) desarrolló, partiendo del principio de la televisión en blanco y negro, un sistema compatible de televisión en color. Ello tuvo lugar en 1954. El componente de generación de sincronismo 2622 se desarrolló para el sistema NTSC.

La figura 1.30 nos muestra el esquema de conexiones del generador síncrono.

Para nuestro video computador empleamos el sistema PAL (Phase Alternating Line). Para ello se diseñó el componente 2621. Ha de prestarse atención al número de componente cuando vaya a pedirse.

En la figura 1.31 tenemos el diagrama de conexión de bloques del componente USG-2621 (Universal Sync Generator: generador universal de sincronismos). La diferencia entre el 2621 y el 2622 es la relación divisoria en el contador vertical. Con el sistema PAL tenemos una resolución de 312 líneas y con el NTSC de 262 líneas. La disposición de los circui-



Fig. 1.29 Representación de las cifras para la programación de la interfaz de video

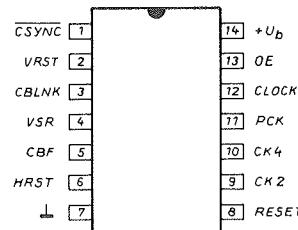


Fig. 1.30 Esquema de conexiones del 262/2622

tos lógicos produce también otras señales de salida con retardo.

Tenemos la disposición de terminales de la página siguiente.

La entrada «Clock» está conectada al contador vertical y horizontal mediante un dispositivo intermedio («buffer»). A partir de las distintas funciones obtenemos los diagramas de tiempo de las figuras 1.32 y 1.33.

La figura 1.32 nos muestra el diagrama de impulsos para el proceso horizontal en el video computador. En la primera línea tenemos la salida PCK para una línea de televisión. Mediante el decodificador, se transfiere a la salida la frecuencia de 3,546895 MHz de la entrada «Clock». De esa forma, cada impulso PCK tiene una longitud de 280 ns. El contador horizontal comienza en 0 y cuenta hasta 226. Inmediatamente después se repone a 0 y comienza un nuevo proceso de recuento.

A partir del recuento se generan en el decodificador horizontal los impulsos de control.

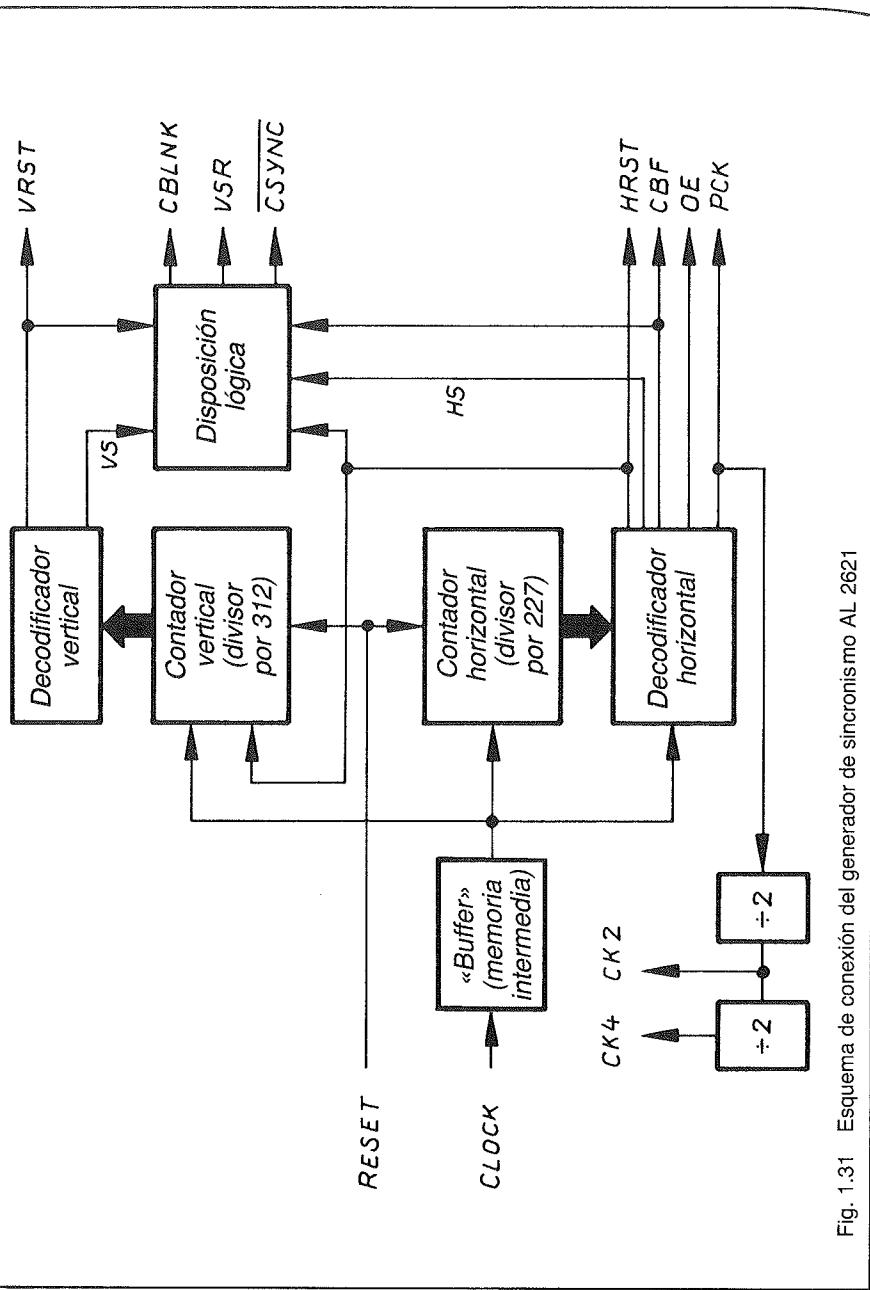


Fig. 1.31 Esquema de conexión del generador de sincronismo AL 2621

Conección	Term.	Entrada/salida	Funcióñ
CSYNC	1	A	Composite Sync (sincronismo compuesto); En esta salida obtenemos los impulsos de sincronización compuestos para el sistema PAL. La salida está directamente conectada con el sumador de video.
VRST	2	A	Vertical reset (reposición vertical); Salida para el impulso de reposición vertical. La entrada VRST está directamente excitada por la interfaz de video y la entrada Sense del microprocesador.
CBLNK	3	A	Composite blanking (borrado compuesto); Salida para impulsos de borrado compuestos que va al sumador de video.
VSR	4	A	Salida para comprobaciones.
CBF	5	A	Color Flag burst; Salida de la señal de «bursts» para el sumador de video.
HRST	6	A	Horizontal reset (reposición horizontal); Salida para el impulso de reposición horizontal.
Reset	8	E	Reset (reposición); Reposición para el contador horizontal y el vertical
CK2	9	A	Clock 2 (Reloj 2); Se divide por 2 la señal PCK.
CK4	10	A	Clock 4 (Reloj 4); Se divide por 4 la señal PCK.
PCK	11	A	Position clock (Reloj de posición); Impulso para las líneas de televisión. Tenemos 227 impulsos de reloj con una longitud de 280 ns. Después de 64 µs se ha terminado una línea, es decir, han pasado 227 impulsos de reloj por la salida PCK.
Clock	12	E	Clock (Reloj); Entrada para la frecuencia de 3.546895 MHz.
OE	13	A	ODDIEVEN (Impar/Par); Determinación de imágenes parciales para el sumador de video.
	1	7	Masa, 0 V.
+Ub	14		Tensión de alimentación +5V.

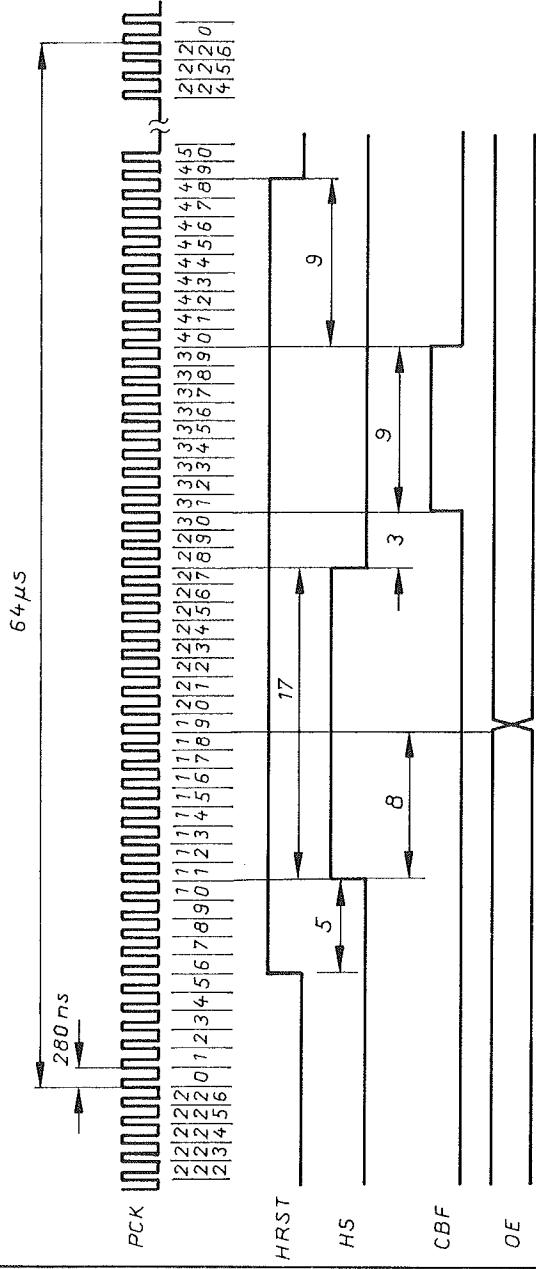


Fig. 1.32 Diagrama de impulsos para el proceso horizontal del computador-TV

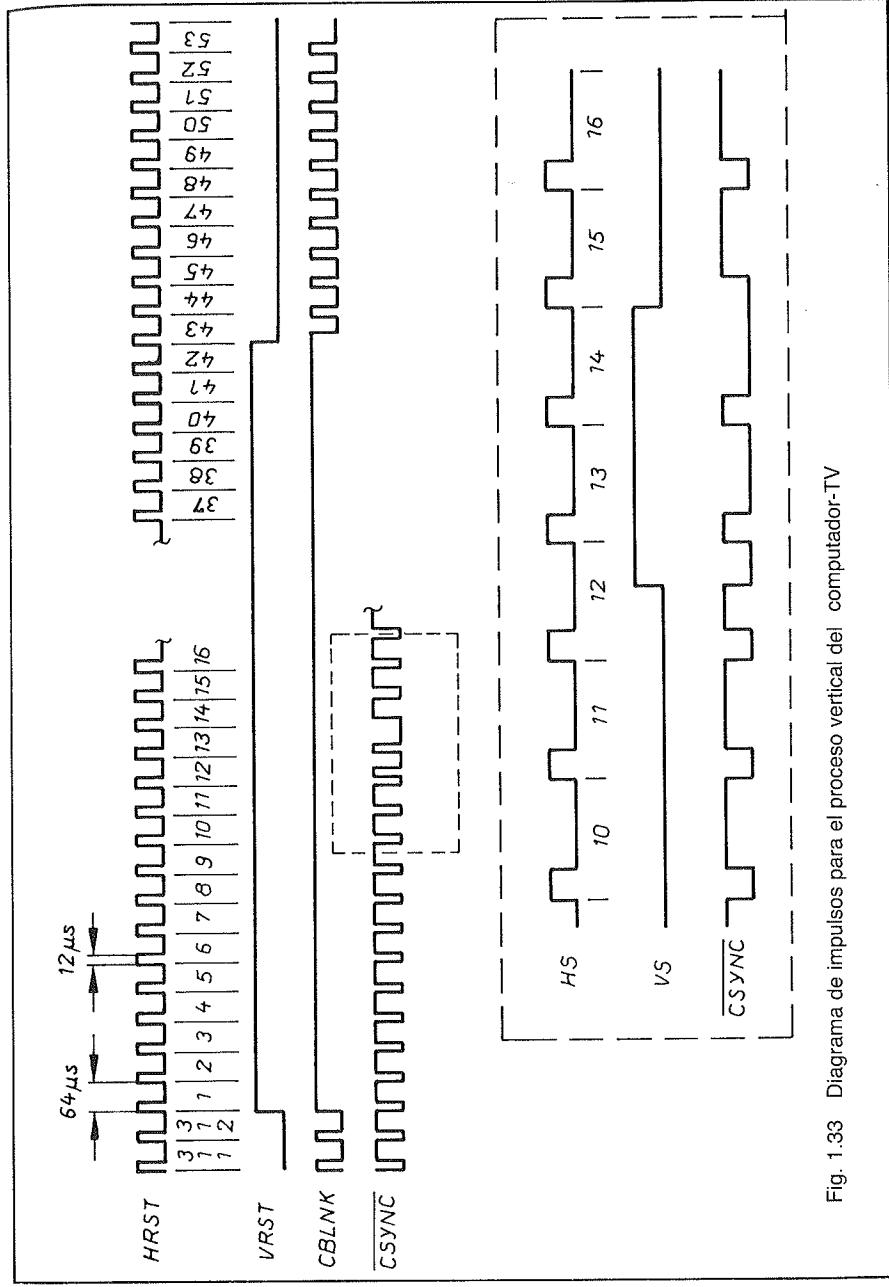


Fig. 1.33 Diagrama de impulsos para el proceso vertical del computador-TV

En la segunda línea tenemos el impulso de reposición horizontal. Después del 5.^º impulso de reloj, la salida HRST se pone a «1» al recibir la señal 1. Esta señal, después del 48 impulso, vuelve a reponerse a la señal 0. Este impulso de reposición es, por tanto, de unos 12 µs de duración y se necesita para la sincronización de la interface de video programable.

La tercera línea del diagrama de impulsos horizontales representa el impulso interno HS (sincronismo horizontal). Este impulso se pone a «1» después de transcurridos 5 impulsos a partir del momento en que HRST se activó la señal 1 y se repone a la señal 0, después de 17 impulsos. Estos 17 impulsos suponen un retardo en tiempo de unos 4,8 µs. Mediante el impulso interno HS se controla la disposición de los circuitos lógicos del componente 2621.

La salida CBF es responsable de la señal de «burst». Transcurrido un tiempo de 8,7 µs, es decir, 31 impulsos de reloj, tenemos una señal 1. Esta señal dura unos 2,5 µs, puesto que tenemos 9 impulsos.

La última línea de la figura 1.32 representa la conmutación entre ambas imágenes parciales de pantalla. Una imagen completa consta de 625 líneas. El generador de sincronismo produce una imagen parcial con 312 líneas y conmuta después la salida OE a la imagen parcial siguiente. La salida OE cambia su estado de salida cada 312 líneas.

La figura 1.33 nos muestra el diagrama de impulsos para el proceso vertical en el video computador. En la primera línea tenemos la reposición horizontal HRST. Entre cada dos flancos positivos consecutivos aparece un intervalo de tiempo de 64 µs, con una duración de impulso de 12 µs. Cuando el contador horizontal ha dado una línea de pantalla completa con 227 impulsos, el contador vertical recibe un impulso de recuento del HRST con un breve retardo de tiempo. De esta forma, el contador vertical puede incrementar sincrónicamente su estado en 1, mediante la entrada «clock» (reloj).

El contador vertical controla al decodificador vertical en el componente 2621. El decodificador vertical genera la señal de salida VRST y el impulso interno de control VS (sincronización vertical). La segunda línea de la figura 1.33 representa el impulso de salida VRST. Este implica 42 líneas de televisión y controla la entrada VRST de la interface de video programable. Simultáneamente, la entrada directa «Sense» del microprocesador recibe una señal 1. Como cada línea de televisión dura 64 µs, tenemos una señal 1 en estas entradas durante unos 1,7 ms. Durante este tiempo, el microprocesador podrá introducir datos en la interface de video o podrá recibirlas de esta última. Después de ese tiempo, ambas unidades quedan bloqueadas por la señal 0. Después de 312 líneas de televisión, la salida salta automáticamente a señal 1.

La tercera línea de la figura representa el impulso de sincronización para el sumador de video. Cuando la salida VRST tiene una señal 1, la salida CBLNK se conmuta también a 1 mediante la disposición de circuitos lógicos del componente y con ello el sumador de video no recibe ningún impulso de borrado para la sincronización de líneas. En el momento en que la salida VRST cambia a señal 0, los impulsos de borrado pueden pasar de nuevo a los circuitos lógicos del componente. La función lógica entre VRST y CBLNK es una relación funcional OR.

La cuarta línea de la figura representa la señal de sincronización compuesta para el sistema PAL. El impulso-CSYNC se compone a partir de la señal horizontal HS y la vertical VS mediante una función OR-exclusiva o una relación funcional parecida. A este respecto se aplica la tabla siguiente:

HS	VS	CSYNC
0	0	0
0	1	0
1	0	1
1	1	1

El impulso-CSYNC controla al sumador de video.

Para poder desviar horizontal y verticalmente los diferentes símbolos de nuestro juego, necesitamos potenciómetros. La interface de video programable puede consultar, mediante el conmutador analógico, el estado de los mismos y convertirlo después en un valor digital.

En la figura 1.34 se muestra la estructura del componente-CMOS 4053. Las 3 entradas lógicas A, B y C controlan un convertidor de niveles lógicos. Mediante la entrada CE (chip enable: chip disponible) puede bloquearse el componente. El convertidor tiene una tensión de servicio de $+U_b = 3\ldots 15$ V. La entrada U1 es la entrada a masa para la tensión de nivel. En nuestra aplicación, conectaremos la entrada U1 a masa. Las salidas del convertidor de niveles lógicos están conectadas a los decodificadores 2 a 1. En este caso, a partir de 2 entradas se genera una salida.

Con las 6 salidas del decodificador controlamos las puertas de transmisión TG. Para el componente 4053 tenemos la tabla siguiente:

Nr.	C	B	A	CE	canales		
					Q _C	Q _B	Q _A
0	L	L	L	L	X _C	X _B	X _A
1	L	L	H	L	X _C	X _B	Y _A
2	L	H	L	L	X _C	Y _B	X _A
3	L	H	H	L	X _C	Y _B	Y _A
4	H	L	L	L	Y _C	X _B	X _A
5	H	L	H	L	Y _C	X _B	Y _A
6	H	H	L	L	Y _C	Y _B	X _A
7	H	H	H	L	Y _C	Y _B	Y _A
8	X	X	X	H	bloqueado		

x = nivel bajo o alto

Con el nivel bajo (señal 0) comutamos las entradas X con las salidas Q y con un nivel alto (señal 1) las entradas Y. La entrada CE tiene que estar en nivel bajo para que las puertas de transmisión pue-

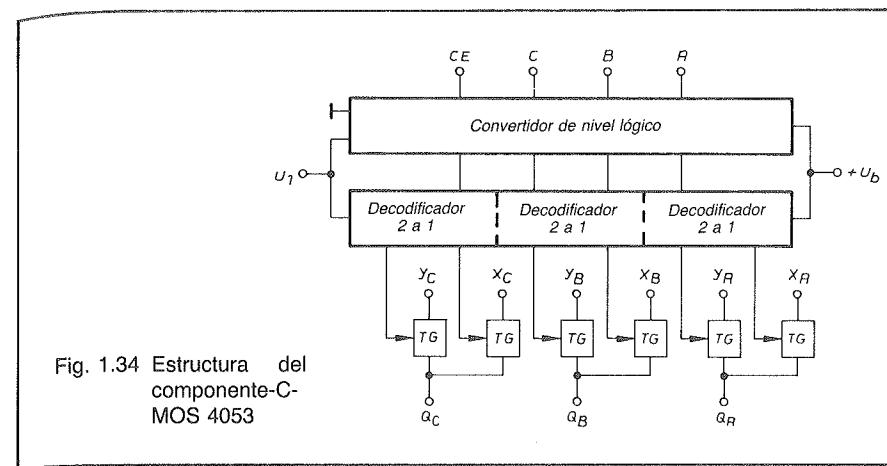


Fig. 1.34 Estructura del componente-CMOS 4053

dan efectuar la conmutación. La figura 1.35 muestra el esquema de conexiones.

Una puerta de transmisión TG es un conmutador analógico, es decir, sirve para comutar magnitudes en tensión alterna. Un conmutador analógico tiene las mismas funciones eléctricas que un conmutador mecánico de contacto. La figura 1.36 representa la conexión interna de un conmutador analógico.

El conmutador analógico de la figura 1.36 se ha realizado en técnica CMOS. El conmutador propiamente dicho consta de una conmutación en paralelo de un transistor de canal-p y otro transistor MOSFET de canal-n. Aquí tenemos la entrada X y la salida Q.

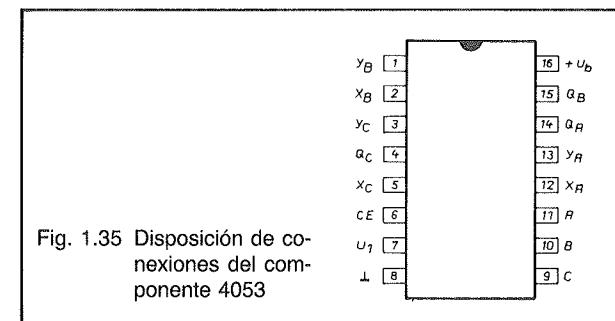


Fig. 1.35 Disposición de conexiones del componente 4053

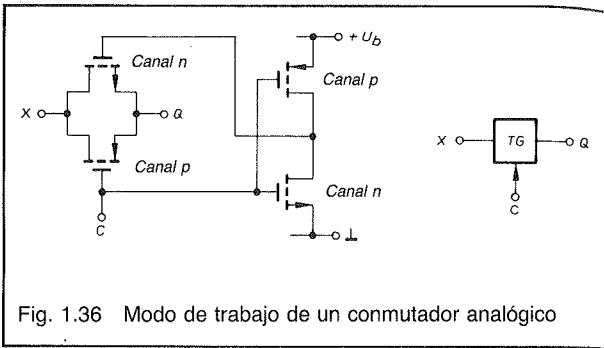


Fig. 1.36 Modo de trabajo de un conmutador analógico

Quien conozca la forma de trabajo de los transistores MOSFET sabe, que la fuente y el drenador se adaptan a la polaridad de la tensión aplicada. La entrada C para la puerta de transmisión es la entrada de control. El transistor MOSFET de canal-p recibe directamente su señal de la entrada C. La puerta del transistor de canal-n está conectada a la entrada C mediante un inversor-CMOS. Si aplicamos a la entrada C un nivel bajo, el transistor de canal-p será conductor. Las dos conexiones de puertas del inversor-CMOS están también en nivel bajo y el transistor de canal-p efectúa la conmutación, mientras que el de canal-n se bloquea. Con ello, el transistor-n de la puerta de transmisión recibe un nivel alto y realiza la conmutación.

Si ambos MOSFET efectúan la conmutación, tenemos en la puerta de transmisión una resistencia de paso de unos 80 ohmios para una tensión de servicio de $+U_b = 5$ V. Podemos, por tanto, transmitir una gama de frecuencias desde 0 Hz (corriente continua) hasta 10 MHz.

Si en la entrada C hay un nivel alto, el transistor-p recibe de la puerta de transmisión un nivel de bloqueo. El transistor se bloquea totalmente. En el inversor-CMOS, el transistor-p se bloquea y el transistor-n conduce. Tendremos en la salida del inversor-CMOS un nivel bajo. Este nivel bajo controla al transistor-n y le bloquea totalmente.

1.5 MEMORIA FIJA BORRABLE Y PROGRAMABLE

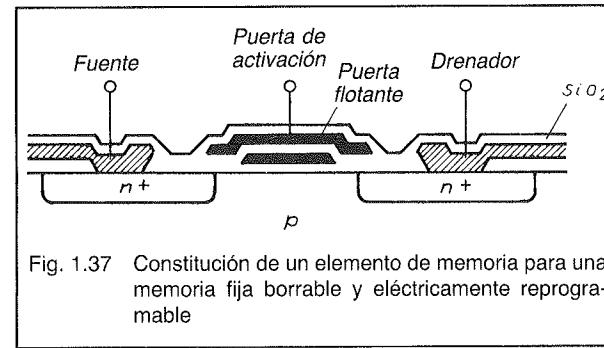


Fig. 1.37 Constitución de un elemento de memoria para una memoria fija borrable y eléctricamente reprogramable

Si ambos MOSFET están bloqueados, tendremos en la puerta de transmisión una resistencia de paso de unos 10^6 ohmios (con una tensión de servicio de $+U_b = 5$ V).

La figura 1.37 representa la constitución de un elemento de memoria borrable y programable por medios eléctricos. Una ROM (memoria fija) trabaja de forma muy distinta a la de una RAM (memoria de lectura-escritura). Si desconectamos la tensión de servicio de nuestro video-computador, el contenido de las RAM desaparecerá, mientras que se conservará el contenido de las ROM.

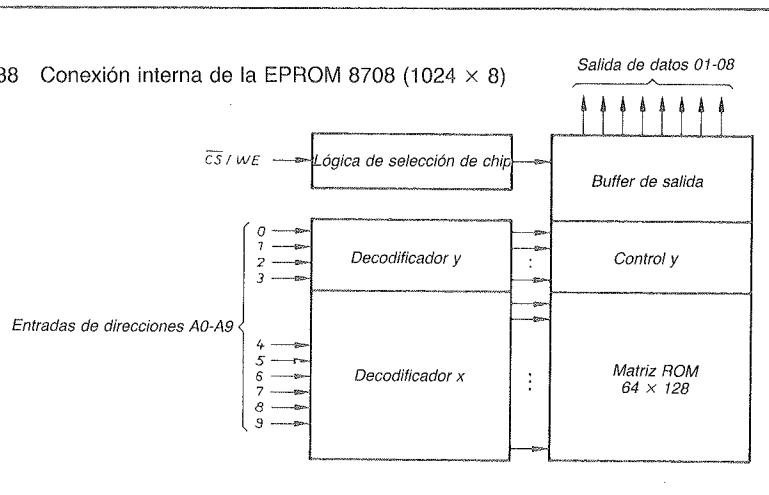
Entre las ROM distinguimos las de máscara y las memorias fijas programables. Las ROM de máscara se aplican sólo en la industria, ya que los costes de las máscaras son muy altos, aunque el precio del componente sea relativamente bajo. Quedan fuera de discusión también las memorias fijas sin dispositivos de borrado, ya que si bien las podemos borrar, su contenido permanece para siempre. Para nuestra aplicación son adecuadas las REPROM (RE-Programmable ROM) o las EPROM (Erasable ROM) puesto que podremos borrar su contenido, desarrollar nuevos programas e introducirlos en ellas sin necesidad de tener que adquirir otro componente nuevo.

Las REPROM o EPROM se fabrican casi exclusivamente con tecnología de puertas de silicio. Como muestra la figura 1.37, un elemento de memoria consta de un único transistor MOS que dispone de 2 capas de polisilicio. La capa superior es la puerta y la segunda es la puerta «flotante». En esta última podemos almacenar nuestra información. Al programarla introducimos una inyección de electrones de alta energía en la puerta flotante. Esa inyección queda allí almacenada, puesto que no hay conexión eléctrica alguna en su entorno. La puerta móvil está incorporada en una capa aislante de SiO_2 .

En estado no programado, un elemento de memoria tiene siempre un nivel alto. Al programarle le damos un nivel alto. Es preciso atenerse a las normas de programación siguientes. De no ser así puede destruirse el componente.

La figura 1.38 muestra la conexión interna de la EPROM 8708. En esta EPROM se contiene la mitad de las instrucciones de nuestro juego televisivo. Para almacenarlas todas necesitamos dos memorias EPROM 8708 o bien emplearemos la EPROM 2716.

Fig. 1.38 Conexión interna de la EPROM 8708 (1024 × 8)



La EPROM 8708 tiene una matriz-ROM interna de 64×128 , es decir, tenemos 64 columnas de 128 líneas cada una. Esto proporciona una capacidad de almacenamiento de $64 \times 128 = 8192$ posiciones de memoria. La EPROM 2716, por el contrario, tiene una matriz ROM interna de $128 \times 128 = 16384$ posiciones de memoria.

La figura 1.39 representa el esquema de conexiones de la EPROM 8708. Disponemos de las entradas de direcciones A0 a A9 y podemos, con ellas, elegir desde la dirección 0 hasta la 1023. La EPROM 2716 tiene las entradas de direcciones de A0 a A10 y, por tanto, direcciones desde 0 a 2047. Pero ocurre que, normalmente, la EPROM 2716 no se obtiene en el comercio y además es 8 veces más cara.

La matriz ROM de la figura 1.38 está dispuesta bajo una ventana de cuarzo a través de la cual podremos borrar el contenido de la EPROM 8708. Para ello necesitamos un radiador de esterilización ultravioleta del tipo HNS de la firma Osram. Este dispositivo hemos de colocarlo, sin filtro de onda corta, a una distancia de unos 3 cm de la ventana de cuarzo.

Advertencia

Al borrar la EPROM mediante un aparato de ese tipo, el componente a borrar y el aparato han de estar encerrados en una caja opaca. Si esto no se hace así, se destruirá inevitablemente la retina del ojo. Hay de tomarse en serio esta advertencia, ya que se produce ceguera incluso tras una corta irradiación (aproximadamente 1 ms).

Bajo el efecto de la luz ultravioleta muy intensa, la EPROM 8708 se borra en cuestión de 30 minutos.

El componente 8708 tiene, como ya sabemos, en estado no programado un nivel alto. Mediante «buffer» de salida, estos niveles altos se comután a

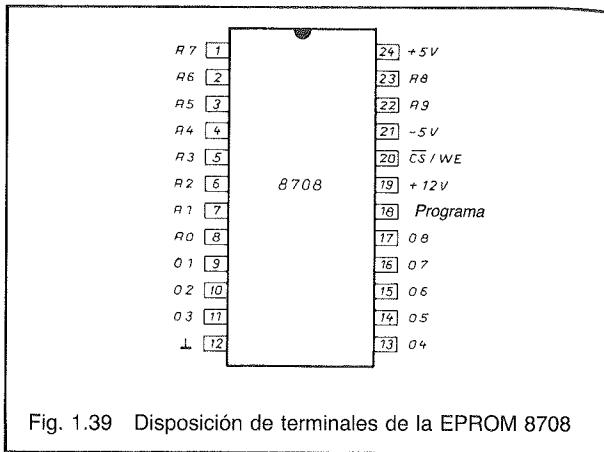


Fig. 1.39 Disposición de terminales de la EPROM 8708

la salida. Las informaciones se introducen con niveles bajos.

La programación de nuestro componente 8708 es relativamente simple si, previamente, nos hemos construido el dispositivo programador. La figura 1.40 nos muestra el diagrama de impulsos para el proceso de programación.

Inicialmente accedemos al 8708. Si la dirección se encuentra efectivamente en las entradas, se elevará a +12V la entrada CS/WE (terminal 20). Esta entrada debe estar a +5V en el estado normal (CS

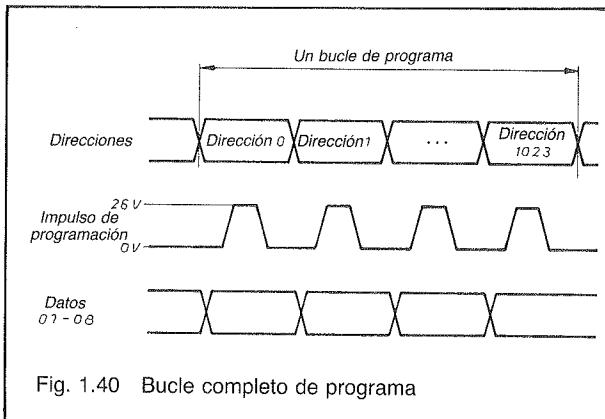


Fig. 1.40 Bucle completo de programa

= Chip Select, WE = Write Enable). Los datos a programar se dispondrán después en paralelo en las 8 salidas, teniendo en cuenta que señal 1 = nivel alto y señal 0 = nivel bajo. Después de determinar las direcciones, la disponibilidad (\overline{CS}/WE) y los datos, se da un impulso de programación por cada dirección en la entrada de programa (terminal 18). Cada vez programamos simultáneamente las 8 salidas. El impulso debe tener una duración de 0,5 ms. La intensidad de corriente no puede sobrepasar 15 mA. La tensión de programación es de +25V.

Si nos atenemos exactamente a estos valores, podremos borrar y reprogramar tantas veces como queramos el componente 8708.

En la figura 1.40 se representa un ciclo completo de programación para la programación. Comenzamos con la dirección 0 y terminamos el bucle en la dirección 1023. Esta pasada, en una sola vez, a través de todas las direcciones a programar, es lo que los técnicos denominan un bucle de programa. Para el componente 8708 necesitamos un total de 100 bucles de programa. Después de ello, el componente está programado segura y certamente. Con independencia de los datos, si bien el componente está programado, los datos, sin embargo, se volatilizan lentamente y nuestro programa se destruye con esta operación.

La figura 1.41 nos representa el esquema de un dispositivo simple para programar la EPROM 8708. Este esquema está comprobado y se emplea en muchos programas industriales.

En el programador (dispositivo de programación) distinguimos dos modos de servicio: la introducción en RAM (memoria de lectura-escritura) y la introducción de los datos almacenados en la EPROM (memoria fija).

Con el conmutador «programiere» (programación) determinaremos si han de introducirse los datos en la RAM o si el programador inicia un proceso de programación. Los conmutadores S1, S2 y S3 están

interconectados mecánicamente. Si el conmutador S1 está abierto, los S2 y S3 están cerrados.

En el momento de introducir los datos en el programador, el conmutador S1 está abierto y, por tanto, tenemos un nivel alto en G1 (puerta N0). De esa forma, en G2 (puerta N0) y en G3 (puerta Y) tenemos un nivel bajo. La puerta Y, G3, bloquea el generador de onda rectangular en la salida. La puerta Y, G4, así como las 8 puertas Y triestadio obtienen un nivel alto a través de la puerta N0, G3. Cuando el conmutador «speichere» (memorización) está abierto, la entrada CS de la RAM tiene un nivel alto. Con ello se impide que la RAM acepte datos. Si, por el contrario, pulsamos el conmutador, la entrada CS tendrá un nivel bajo y la RAM almacenará los datos. La entrada de datos tiene lugar a través de las 8 puertas Y triestadio. A este respecto, utilizamos dos componentes del tipo 74126. Tenemos la siguiente tabla de funciones:

Entradas		Salida
C	A	Y
L	X	Z
H	L	L
H	H	H

X ≡ Nivel alto o bajo

Z ≡ Alta impedancia

Si la entrada C de control tiene un nivel alto, la salida Y será de alta impedancia. Solamente cuando esta entrada tenga nivel alto se transmitirán los datos desde la entrada A a la salida.

Durante el proceso de escritura tenemos un nivel alto en la entrada de control C de la puerta Y triestadio y podemos, por tanto, introducir los datos mediante los conmutadores D1 a D8. Las señales 1 o los niveles altos introducidos se muestran mediante los LED.

El bus de datos tiene un comportamiento triestadio. Por esta razón, podremos conectar directamente

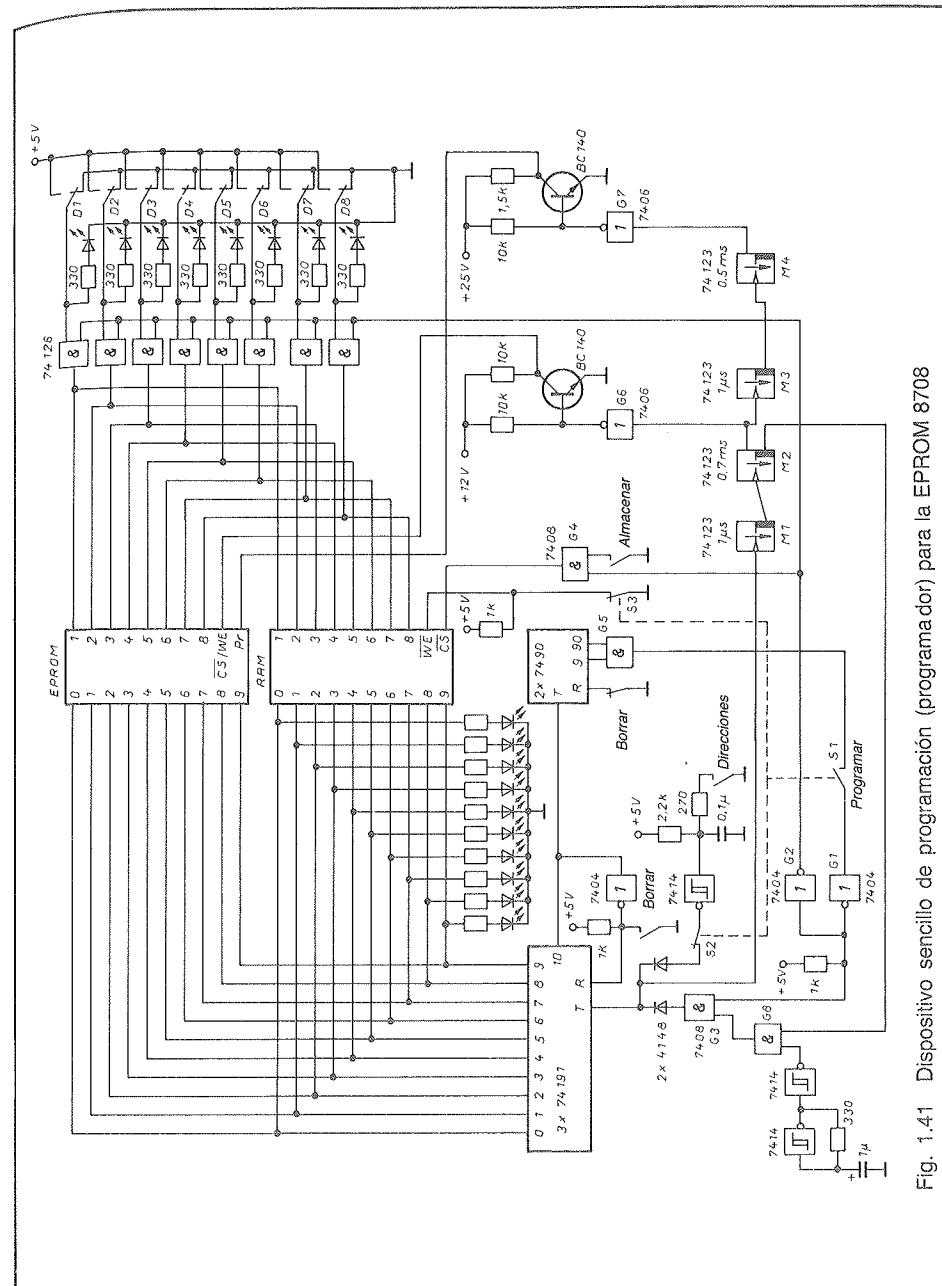


Fig. 1.41 Dispositivo sencillo de programación (programador) para la EPROM 8708

las entradas y las salidas de datos. Los datos de entrada se encuentran simultáneamente en la EPROM y en la RAM. La EPROM no puede aceptar datos porque la entrada CS/WE tiene un nivel bajo. La entrada CS de la RAM tiene nivel bajo durante la entrada de datos ya que el comutador S3 está cerrado. Después de finalizada la entrada de datos mediante los comutadores D1 a D8, los introduciremos en las RAM mediante el comutador «speichern» (almacenar).

Si en la RAM aún quedaran datos, éstos se recuperan y se pierden. Igual sucede si cortamos la tensión de servicio.

Si el comutador S3 está abierto no podemos registrar ningún dato y tan sólo podremos leerlos. Este es el caso de cuando en el proceso de programación transferimos a la EPROM los datos almacenados.

Las figuras 1.42 y 1.43 muestran el esquema de conexiones y el diagrama de bloques de la memoria de escritura-lectura 2114. Este tipo de memoria tiene una configuración de $1024 \times 4 = 4096$ posiciones de memoria. En nuestro dispositivo programador de la figura 1.41 necesitamos dos componentes del tipo 2114. Los conectamos simplemente en paralelo con lo que obtenemos una estructura de 1024×8 .

Mediante las direcciones A3 a A8 seleccionamos las líneas de la matriz de almacenamiento. Puesto que disponemos de 6 entradas, obtenemos $2^6 = 64$ líneas. Estas entradas están dotadas de puertas especiales al igual que sucede en los decodificadores.

Mediante las direcciones A0 a A2 y con la A9 seleccionamos las columnas de la matriz de almacenamiento. Con estas cuatro entradas obtenemos $2^4 = 16$ columnas. Cada columna consta de 4 subcolumnas, de modo que así conseguimos las 64 columnas de la memoria de lectura-escritura 2114.

En el esquema de la figura 1.41 tenemos una estructura de buses para los datos, lo que quiere decir que podremos desplazar los datos a un lado y otro arbitrariamente. Por esta razón, el componente 2114

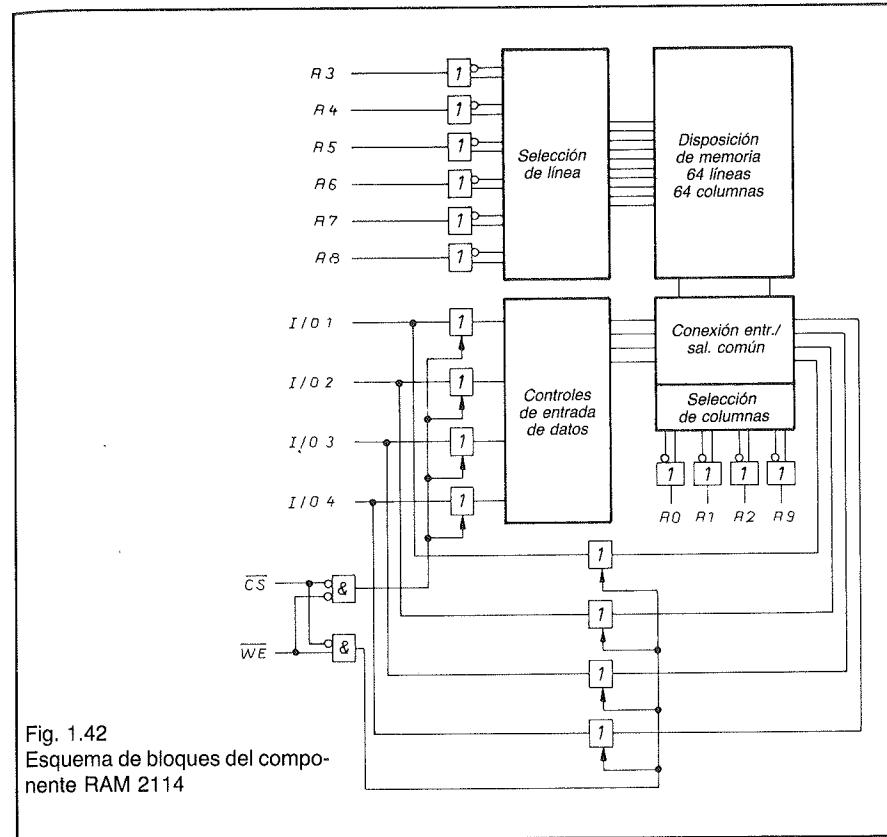


Fig. 1.42
Esquema de bloques del componente RAM 2114

tiene entradas (I = Input) y salidas (O = Output) de datos combinadas. Este modo de servicio E/S quedará determinado mediante las dos entradas CS y WE. Nos atendremos a la tabla siguiente:

CS	WE	Función
H	X	Componente bloqueado
L	L	Servicio de escritura
L	H	Servicio de lectura

Las dos puertas Y tienen a su cargo el control del modo de servicio entre desbloqueo, escritura y lectura.

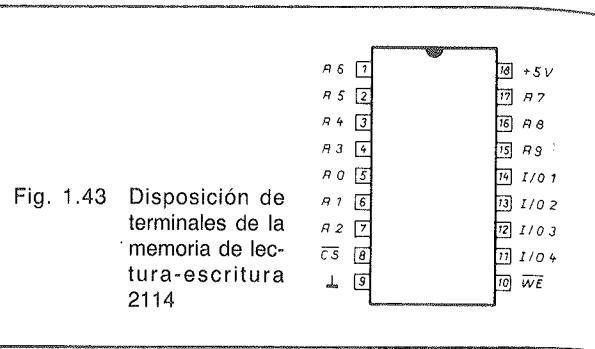


Fig. 1.43 Disposición de terminales de la memoria de lectura-escritura 2114

ra. Si el servicio es de escritura, los datos se encontrarán en los controles de entrada a través de los «buffer» triestado. Luego se escribirán en las correspondientes posiciones de memoria mediante la conmutación E/S común. Durante el servicio de lectura, los «buffer» triestado situados delante de los controles de entrada de datos estarán bloqueados, pero no así los restantes «buffer» triestado que se encontrarán disponibles. A través de ellos, se leerán, sin riesgo de destrucción, los datos almacenados a los que se dará salida por las conexiones comunes de E/S. De esta manera, nos ahorraremos numerosos componentes lógicos.

Mediante el conmutador «Adresse» (direcciones) determinamos cada una de éstas. Utilizamos aquí una sencilla lógica especial. Si el conmutador está abierto, el disparador Schmitt tiene en su entrada un nivel alto. Si pulsamos el conmutador, el condensador se descarga a masa a través de la resistencia. Si la tensión en el condensador queda por debajo de un valor determinado, el componente 7414 identifica un nivel bajo y en la salida aparece un flanco positivo para el contador que está conectado a continuación.

Entre la puerta Y y el conmutador de direcciones se tiene una simple puerta O de diodos (1). Si el conmutador S2 está abierto, el conmutador «Adresse» no tiene efecto.

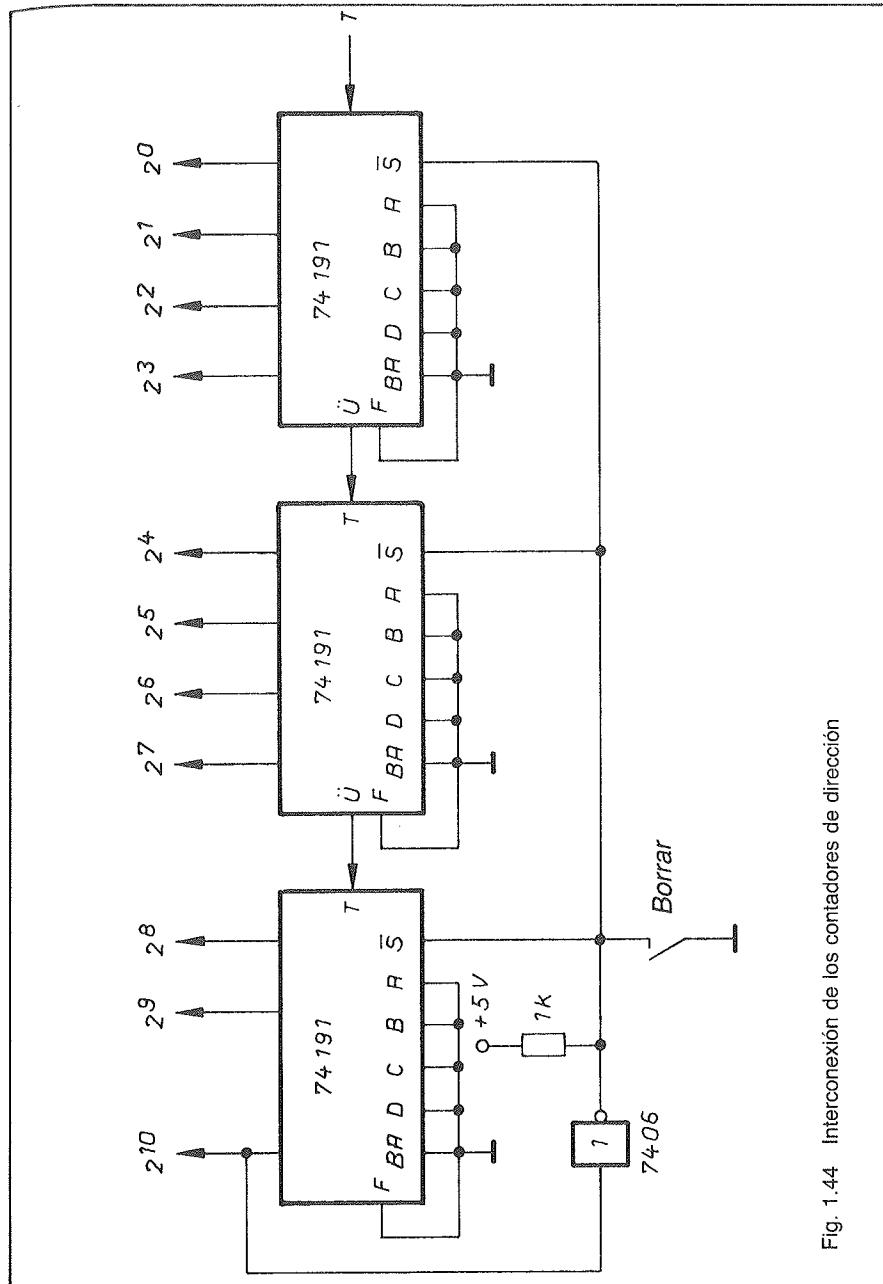


Fig. 1.44 Interconexión de los contadores de dirección

La figura 1.44 nos muestra la interconexión del contador de direcciones. Para ello necesitamos tres componentes contadores del tipo 74191. Estos componentes cuentan de 0 a 15 y se reponen automáticamente a 0. Al efectuarse la reposición emiten en la salida Ü un impulso positivo.

La señal para los contadores de la figura 1.44 la generamos bien sea mediante la señal generada como se ha explicado anteriormente, bien sea mediante el generador de onda cuadrada. Ello depende de la posición que tenga el conmutador «programmire». El contador de direcciones aumenta en 1 con cada flanco positivo cuando el conmutador S2 está cerrado; en caso contrario, mediante el generador de onda cuadrada.

Las entradas F (desbloqueo) están unidas directamente a masa. Por ello los 3 componentes contadores pueden entrar en servicio inmediatamente puesto que queda impedido el funcionamiento en modo de bloqueo. Mediante las entradas BA establecemos el modo de funcionamiento de los contadores. Al conectar a masa se produce un nivel bajo y los contadores trabajan en modo de recuento creciente.

Cuando se genera un impulso de reloj en el punto T de la figura 1.41, el estado de los contadores de tres décadas se incrementa en un 1. De esa forma, conseguimos un contador de 0 a 1023. Los distintos valores de las direcciones se nos muestran mediante LED. Si la salida $2^{10} = 1024$ alcanza un nivel alto, automáticamente los tres contadores de décadas se reponen a 0. El nivel alto queda neutralizado por la puerta N0 y permanece en las entradas S. Si en las entradas S hay nivel bajo aparecerán en las salidas correspondientes a las informaciones ofrecidas en las entradas A a D. Pero como estas entradas están unidas directamente a masa, tienen, por tanto, un nivel bajo, se responderán a 0 todos los circuitos «flip-flop» internos y el contador de direcciones comenzará de nuevo desde cero.

Las salidas de transferencia Ü están conectadas

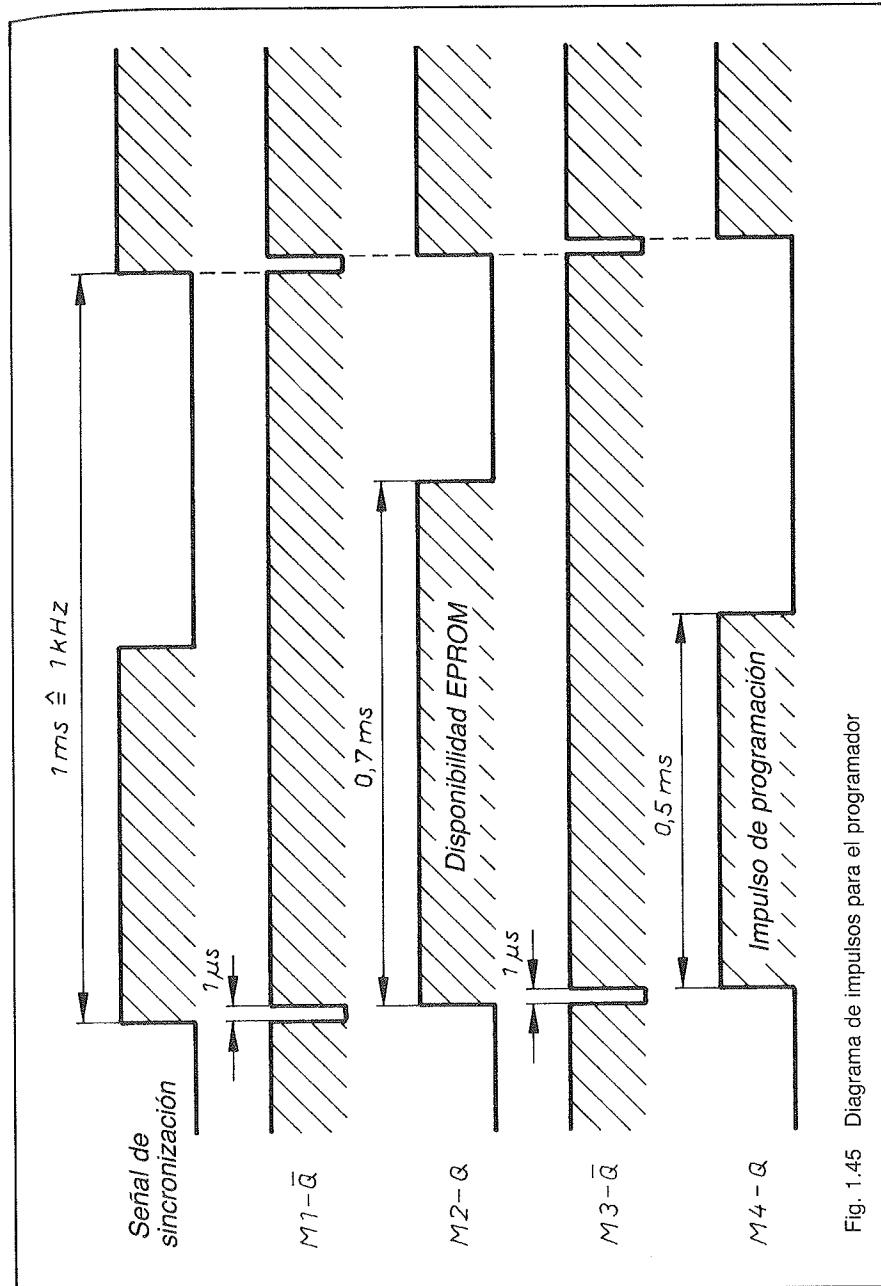


Fig. 1.45 Diagrama de impulsos para el programador

con las correspondientes entradas. El componente 74191 es un contador universal síncrono y, por tanto, es muy adecuado como contador de direcciones.

Podemos poner a cero también los contadores mediante el conmutador «losschen» (borrado).

La salida 10 del contador de direcciones está conectado a un contador de ciclos. Este cuenta los bucles de programa y trabaja de 0 a 99. Lo designamos, por tanto, como contador de 100. El primer contador trabaja de 0 a 9 y después se repone automáticamente a 0 ya que el componente 7490 es un contador BCD. Cuando se repone, el estado del siguiente contador se incrementa en un 1. Después de 99 impulsos, o de 99 bucles de programa, tenemos un estado de contador igual a 99. Las salidas 9 y 90 del contador de 100 están interconectadas mediante la puerta Y, G5. Si los contadores tienen conjuntamente el estado 99, se satisface la condición Y y el conmutador S1 adopta un nivel bajo. Pero, como durante la introducción de datos S1 está abierto, ese nivel no entra en consideración para la reposición en la puerta N0, G1.

Cuando el esquema de la figura 1.41 se utilice para programar la EPROM, se situará el conmutador en la posición «programmieren». El conmutador S1 está cerrado y los S2 y S3 abiertos. Con el conmutador de borrado se repone el contador de las centenas. Esto tenemos que hacerlo también con el contador de direcciones.

Cuando el contador de centenas se repone a 0, el conmutador S1 tiene un nivel alto. La puerta N0, G1, invierte ese nivel alto. Los impulsos de generador de onda cuadrada pueden atravesar la puerta G3 y llegan así a la entrada de señal de sincronización a través de la puerta 0 de diodos. Mediante la puerta Y, G8, obtenemos una sincronización adicional del dispositivo programador. Si se activara uno de los circuitos monoestables a causa de un impulso de ruido, se bloquearían los impulsos rectangulares.

Si en la salida de la puerta G2 hay nivel bajo, la entrada CS de la memoria de lectura-escritura se

ponen también a nivel bajo. La entrada WE está a nivel alto ya que el conmutador S3 está abierto. La RAM puede conmutar sus datos al bus. Si en la salida de la puerta G2 hay nivel bajo quedan bloqueadas también las puertas Y triestado de las entradas de datos. Con ello quedan sin efecto funcional estos ocho conmutadores y sus salidas quedan de alta impedancia.

El generador de onda cuadrada trabaja con una frecuencia de 1 kHz. Esta frecuencia se consigue mediante la resistencia de 330 ohmios en conexión con el condensador de 1 μ F. Este generador ofrece una gran seguridad de oscilación. Sus impulsos de pico se transforman en impulsos rectangulares mediante el disparador Schmitt conectado a continuación.

El contador de direcciones recibe sus impulsos a través de las dos puertas Y y de la puerta 0 de diodos. El contador de direcciones incrementa su estado con cada flanco positivo. En la gama de 130 ns disponemos de una nueva dirección en las salidas, en el momento en que el impulso ha cambiado de nivel bajo a alto.

Durante la programación de la EPROM, es esencial para el control del proceso su evolución en el tiempo. En la figura 1.45 tenemos el diagrama de impulsos para el dispositivo programador de la figura 1.41.

El impulso de sincronización (o de reloj) tiene una duración de 1 ms \approx 1 kHz. El monoestable M1 retarda este impulso en 1 μ s. Durante este tiempo, han quedado fijadas con seguridad las direcciones en la EPROM y en la RAM. Esto es totalmente necesario puesto que en la entrada WE de la RAM ya hay un nivel alto.

En la salida Q del monoestable tenemos un nivel bajo durante 1 μ s. Después de este tiempo metaestable aparece aquí un flanco de disparo positivo para el monoestable M2. El monoestable M2 obtiene en la salida Q un nivel alto que produce un disparo en la puerta N0 G6 y simultáneamente en el M3.

El monoestable M3, después del efecto de disparo provocado por M2, entra en un breve período de tiempo metaestable de 1 μ s y provoca, a su vez, un efecto de disparo en el M4. Este monoestable tiene un tiempo metaestable de 0,5 μ s.

En el diagrama de impulsos de la figura 1.45 observamos la secuencia temporal de las diferentes fases. Si la puerta N0, G6, recibe un impulso de nivel alto, la salida tiene, entonces, un nivel bajo. Esta condición vale también para el M4 y la puerta N0, G7.

El circuito integrado 7406 consta de 6 puertas N0 con «colector abierto». Los transistores de salida admiten una tensión de servicio de hasta 30 V. Con un nivel alto en la entrada, el transistor BC 140 queda bloqueado por una puerta N0 y ya no circulará ninguna corriente de base. El transistor se bloquea, en la entrada CS/WE hay +12V y en la entrada PR +25V. El impulso de +12V dura 0,7 ms y el de +25V dura 0,5 ms. Con ello tenemos las especificaciones correctas de programación.

Si los monoestables retornan de sus fases metaestables tenemos en ambas puertas N0 un nivel bajo. Entonces, el transistor de salida de las puertas queda bloqueado y circula una corriente de base a través de las resistencias de 10 k Ω . Los transistores conducen y tendremos nivel bajo en las entradas CS/WE o Pr respectivamente. La EPROM queda así bloqueada.

Los dos transistores BC 140 han de estar suficientemente fríos.

Para el transistor, con la corriente de programación tenemos una resistencia de trabajo de 1,5 k Ω con una tensión de servicio de +25V. Circula una corriente de programación de unos 17 mA.

2. Estructura y las interconexiones del computador de televisión

El montaje y las interconexiones del computador de televisión tienen lugar en 5 etapas. Pero se necesitan, además, algunas aclaraciones para aquellos casos en los que el aparato no trabaja correctamente o sufre algún deterioro mecánico.

Tenemos los apartados siguientes:

- 2.1 Sumador de video con oscilador de cuarzo y salida de video.
- 2.2 Microprocesador, interface de video programable, generador PAL y memoria de instrucciones. (Esta memoria se encuentra en una placa aparte.)
- 2.3 Sección de audio: Generación de la secuencia de tonos para el modulador.
- 2.4 Modulador: Generación de la tensión AF para la transferencia de informaciones al aparato de televisión.
- 2.5 Sección de red: Tensión de servicio para los distintos grupos de componentes.

Los diferentes grupos de componentes actúan recíprocamente, unos sobre otros, dependiendo de su estructura. Por esta razón, las distintas fases de la explicación expuesta son algo difíciles, a no ser que se sea un experto en televisión en color. De las diferentes etapas se deducirá, después, la conexión completa para el computador de televisión.

2.1 SUMADOR DE VIDEO

El sumador de video contiene el oscilador de cuarzo con una frecuencia de 8,86 MHz. Esta frecuencia queda dividida mediante dos divisores separados. Uno de ellos tiene la relación 1:2 y nos genera las distintas frecuencias para la modulación de color y el otro tiene una relación 1:2,5 y proporciona la frecuencia de sincronismo.

El sumador de video, a partir de las tres salidas C1, C2 y C3, modula, según lo programado, la señal cromática que va al modulador de AF. En la salida «Video» tenemos la señal de color completa pero sin señal de audio.

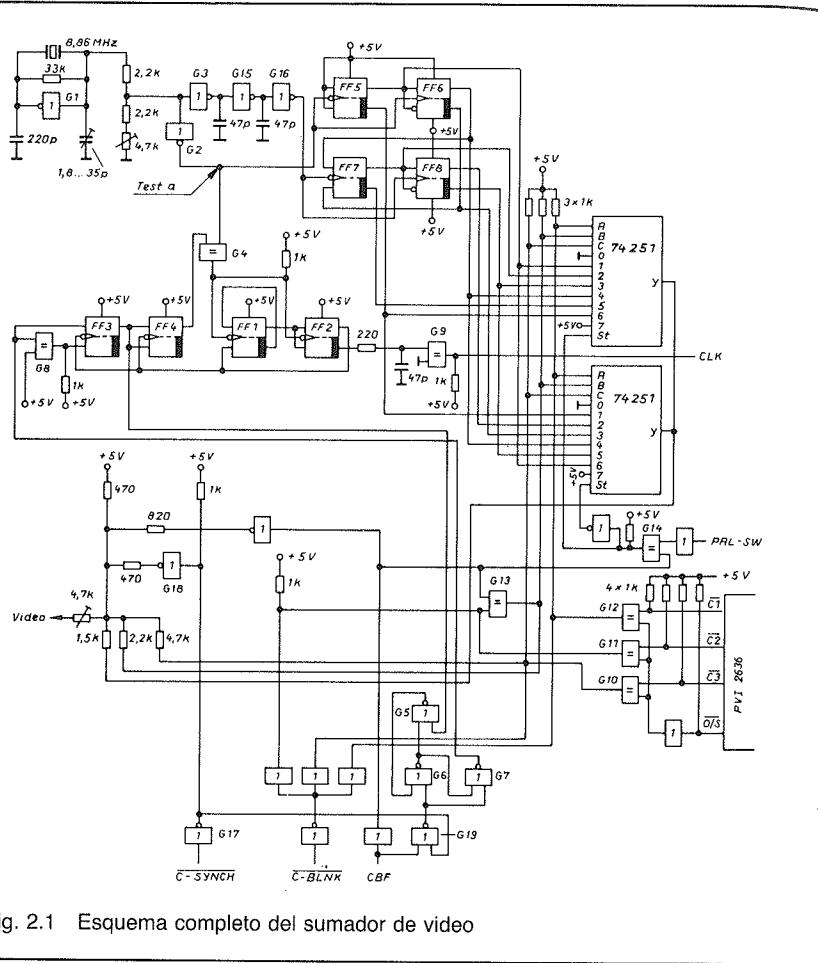


Fig. 2.1 Esquema completo del sumador de video

La figura 2.1 nos muestra la conexión completa del sumador de video. En nuestro computador de televisión generamos la frecuencia mediante un cristal de cuarzo de 8,867238 MHz exactamente y con una tolerancia de $\pm 20\text{Hz}$. Ha de procurarse que el cuarzo sea de muy buena calidad. ¡No emplear cristales de cuarzo de armónicos! En combinación con la puerta N0, G1, conseguimos, así, un oscilador de cuarzo estable.

Normalmente, los componentes TTL no están concebidos para conexiones analógicas. Los osciladores pertenecen también al grupo de circuitos analógicos. Sin embargo, mediante la resistencia de 33 k Ω logramos que una puerta TTL trabaje en modo analógico. A partir de los dos puntos de trabajo de la puerta N0 TTL conseguimos un punto de trabajo para el oscilador. La conexión de la figura 2.1 oscila con gran seguridad y su comportamiento es muy estable.

El punto de trabajo del oscilador queda mejorado mediante los dos condensadores que le están acoplados, ya que la relación de tensión continua se mantiene a potencial constante. Podemos influir sobre la frecuencia mediante el condensador variable. En paralelo con dicho condensador hay también un divisor resistivo de tensión. Con el condensador variable podemos regular la frecuencia del oscilador y con ese mismo ajuste, podremos amortiguar algo el oscilador.

Para el equilibrado de frecuencias conectamos el «punto de prueba a un contador de frecuencia. Hemos de medir en ese punto 8,867238 MHz. Mediante el condensador variable podemos modificar algo la frecuencia. Después, se intercambiará el contador de frecuencia con un osciloscopio de banda ancha y se tendrá que ajustar la relación del oscilador con el regulador. Cuando ésta sea del 50 %, el oscilador de cuarzo trabajará correctamente.

Si no dispone de contador de frecuencia o de un osciloscopio de banda ancha, el equilibrado puede ser enojoso puesto que el computador sólo podremos equilibrarlo con un aparato de televisión. En tal caso, el regulador y el condensador variable habrán de situarse en una posición media. Ha de ajustarse el sonido del televisor muy bajo, ya que el sonido del conmutador no está todavía ajustado.

Si tenemos una imagen estable, la frecuencia será correcta. La estabilidad se refiere tanto a las líneas como al color. Si la imagen fuera inestable, ajustaríamos, entonces, lentamente el condensador

variable hasta que el barrido de las líneas sea correcto. Las tolerancias menores serán reguladas por el receptor de televisión. Igualmente ajustaremos lentamente el ajuste del oscilador hasta que «aparezca» el color. Entonces, tenemos una relación del 50 % en el oscilador de cuarzo.

Normalmente, si empleamos un buen oscilador de cuarzo, el trabajo de equilibrado no representa problema alguno. El cuarzo no tiene un coste elevado.

Si obtuviéramos un efecto Moires (imagen con interferencias marginales) o una sobremodulación (distorsión), la causa puede estar también en el potenciómetro de la salida «Video».

La frecuencia del oscilador queda acoplada mediante las dos puertas N0, G2 y G3. Gracias a ello, la frecuencia de la salida se convierte en un nivel TTL estable y se mejora fundamentalmente la verticalidad de los flancos.

No ha de medirse nunca con el cuarzo directamente, puesto que los hilos de medida tienen siempre un efecto capacitivo e influyen con ello negativamente en la frecuencia del cuarzo. La frecuencia aumenta o disminuye. El punto de medida para el equilibrado está detrás de la puerta N0 G2.

Para el circuito integrado 2621 necesitamos una frecuencia de 3,54 MHz. Generamos esta frecuencia mediante los dos circuitos «flip-flop» FF1 y FF2 en conexión con otros dos FF3 y FF4. Obtenemos, así un divisor por 2,5 y una frecuencia de salida de

$$\frac{8,867238 \text{ MHz}}{2,5} = 3,546895 \text{ MHz}$$

El trabajo y función de este divisor por 2,5 es muy complicado. A este respecto, consideremos, en primer lugar, el modo de trabajo del circuito integrado 74113 que se suministra sólo en «Low Power Schottky» (abreviadamente «LS»). Tenemos la tabla de función siguiente:

Preset *	Reloj	Entradas		Salidas	
		J	K	Q	\bar{Q}
L	X	X	X	H	L
H	↓	L	L	Q_n	\bar{Q}_n
H	↓	H	L	H	L
H	↓	L	H	L	H
H	↓	H	H	Báscula	
H	H	X	X	Q_n	\bar{Q}_n

X ≈ Nivel «L» o «H» (bajo o alto)

* (ajuste previo)

Q_n ≈ Estado de memoria

En la figura 2.2 se muestra el esquema de conexión para el componente 74113.

Con cada flanco negativo de reloj podemos activar, reponer y bascular el flip-flop, según sean los niveles en las entradas de excitación J y K. Si no necesitáramos la entrada de preajuste (preset), se la conectaría necesariamente con la tensión de servicio positiva. Si esa entrada quedara abierta, es decir, no conectada, podrían recogerse impulsos de ruido debidos a la técnica «Low Power Schottky» que activarían el flip-flop de forma imprevisible.

En la figura 2.3 se muestra el diagrama de impulsos para el contador de división por 2,5. En la primera línea tenemos el impulso del oscilador de cuarzo con 8,8 MHz. El ciclo de esta frecuencia tiene una duración aproximada de 0,113 µs. Esta frecuencia queda en las dos entradas de los circuitos flip-flop FF1 y FF2 a través de la puerta O-exclusiva. Con el primer flanco negativo se activa el flip-flop FF1. Con

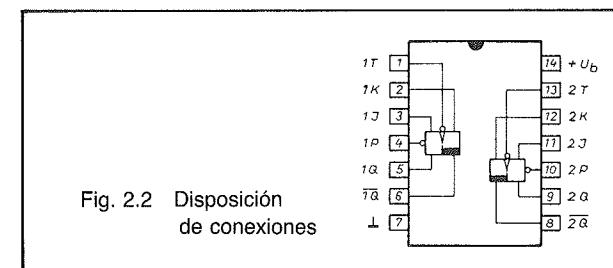
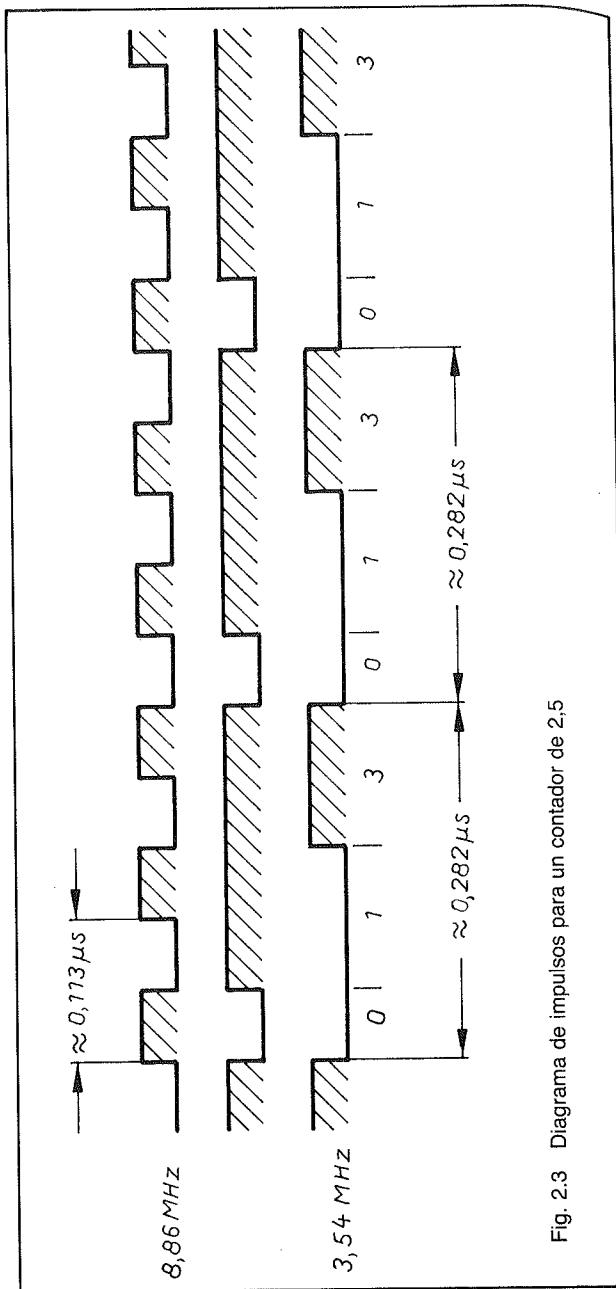


Fig. 2.2 Disposición de conexiones



ello tenemos el estado de contador 1. Cuando se activa el circuito flip-flop FF1 se activará también el FF2, ya que en las dos entradas de excitación hay un nivel alto.

A través de la realimentación, la entrada K del FF1 está conectada con la salida Q del FF2. En la salida Q tenemos un nivel bajo (L) con lo que el FF1 se encuentra en estado de almacenamiento pues $J = K = L$. Con el flanco negativo siguiente se activa el FF2, mientras que FF1 conserva su estado. Con ello tenemos el estado de contador 3. Mediante la realimentación, la entrada K del flip-flop tiene ahora nivel alto y éste puede reponerse.

El control propiamente dicho del contador por 2,5 tiene lugar mediante los circuitos flip-flop FF3 y FF4 en conexión con las puertas G5, G6 y G7. Si para ello hacemos referencia al diagrama de impulsos de la figura 2.3, quedará claro el modo de trabajo. Con un flanco positivo obtenemos el estado 0 de contador si el FF4 está activado. Pero si está desactivado, el estado del contador se alterará sólo con un flanco negativo, como sucede después de transcurridos $0,282 \mu s$. El control se realiza mediante el circuito flip-flop FF4 junto con la puerta 0 exclusiva.

Como puerta 0 exclusiva empleamos el componente 74136. La figura 2.4 nos indica el esquema de conexiones. Obtenemos, así, la tabla de función siguiente:

Entradas		Salida
B	A	X
L	L	L
L	H	H
H	L	H
H	H	L

La entrada A de la puerta 0 exclusiva está conectada a la salida Q del flip-flop FF4. Con ello conseguimos el control para el contador de división por 2,5 como muestra la figura 2.3.

La duración de impulso de un intervalo en la tercera línea en la figura 2.3 es de unos $0,282 \mu s$, que corresponde a 3,54 MHz.

El flip-flop FF3 tiene también delante de las puertas de excitación J y K una puerta 0 exclusiva G8. Puesto que una de ambas entradas está conectada a la tensión de servicio de +5V, obtenemos una función como puerta N0.

De la salida Q del flip-flop FF2 derivamos la salida de sincronización del contador de división por 2,5. Mediante una resistencia podrá cargarse y descargarse un condensador. Tenemos así un elemento integrador para un retardo muy breve. Como «buffer» (separador) para ese elemento se ha conectado la puerta G9, ya que la otra entrada está unida a masa. Así no se obtiene inversión alguna, sino un desacoplamiento. La salida CLK del sumador de video está unida directamente al generador PAL 2621 y se necesita una resistencia de trabajo. Las 4 puertas 0 exclusiva del componente 74136 están montadas en colector «abierto» y es posible obtener una puerta «Y-cableada».

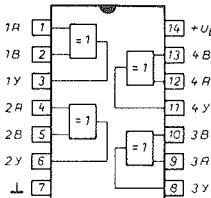
Para crear las señales cromáticas necesitamos una frecuencia de 4,43 MHz. Podemos generarlas mediante un simple contador, o un divisor por 2, sin que haya que realizar muchas conexiones. Dividiremos la frecuencia de 8,86 MHz por 2 y obtenemos así la señal de «burst».

Al explicar la interfase de video programable hemos encontrado ya el modulador de color. Con las salidas C1, C2 y C3 podemos programar los distintos colores mediante la memoria de instrucciones. Ahora, en el sumador de video, los diferentes estados binarios de la interface de video se convierten en señales cromáticas, según el método PAL y se combinan entre sí.

Para comprender la conversión y combinación de las señales cromáticas necesitamos algo de teoría.

Las señales de salida de la interface de video programable son de +5V para nivel alto, o señal

Fig. 2.4
Esquema de conexiones de la puerta 74136



«1», y 0 V para nivel bajo, o señal «0». Este nivel significa para el sumador de video de la señal de densidad luminosa. Pero puesto que hemos de atenernos al procedimiento PAL normalizado, necesitamos un circuito resistivo que genere la señal de video o señal Y. Consideraremos como 100 % la tensión de servicio de +5V, entonces, para los tres co-

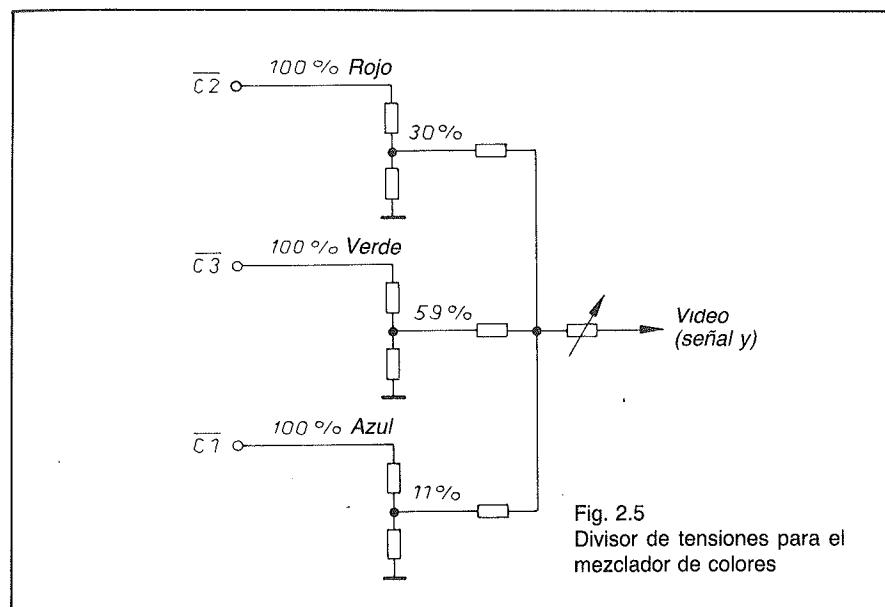


Fig. 2.5
Divisor de tensiones para el mezclador de colores

lores básicos tendremos la siguiente relación de tensiones:

$$\begin{aligned} \text{Rojo} &= 30\% \\ \text{Verde} &= 59\% \\ \text{Azul} &= 11\% \end{aligned}$$

La figura 2.5 nos muestra la red de resistencias para componer la señal Y. Si mezclamos los tres colores, obtenemos

$$30\% + 59\% + 11\% = 100\%$$

es decir, la mezcla «blanco». Si, por el contrario, en la salida Y el valor es 0 %, la pantalla queda oscura, pues 0 % da el negro.

Si iniciamos la realización técnica del procedimiento PAL en el sumador de video, se producen otras dificultades. De los 3 colores: rojo, verde y azul, se transmiten sólo 2 de ellos, el rojo y el azul. El color verde se inhibe y tan sólo en el aparato de televisión se le genera de nuevo. Necesitamos una conexión para generar ambas señales diferenciales de color V y U.

Si examinamos nuevamente el circuito de la figura 2.1, vemos que la salida C3 de la interfaz de video programable controla directamente la resistencia de 4,7 kΩ a través de la puerta 0 exclusiva G10. Simultáneamente, la entrada C de ambos multiplexores recibe un nivel de control.

Lo contrario ocurre con las dos salidas C1 y C2 de la interfaz de video programable. La salida C1 controla las entradas A del multiplexor y la salida C2, a través de una puerta 0 exclusiva adicional G13, controla la entrada B.

Los dos multiplexores de la figura 2.1 son sumadores digitales para generar la señal Y para la señal de rojo y de azul.

La figura 2.6 nos muestra de una forma sencilla el modo de funcionamiento.

En la matriz Y se incluyen conjuntamente los tres colores rojo, verde y azul. En nuestro caso, la matriz

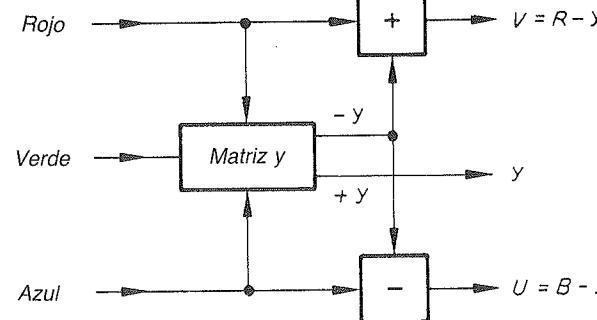


Fig. 2.6 Elaboración de las señales V, U e Y

Y son las tres resistencias de 1,5 kΩ, 2,2 kΩ y 4,7 kΩ situadas en la salida de video de la figura 2.1. En la figura 2.6 se ve como la salida de la matriz nos genera la señal + Y y la señal diferencial - Y. La señal + Y produce la salida Y. La señal diferencial - Y está en el sumador superior e inferior. Puesto que trabajamos en modo digital, empleamos el símbolo + como símbolo de sumador. Las salidas de los sumadores nos generan las dos señales vectoriales U y V:

$$\begin{aligned} U &= B - Y \\ V &= R - Y \end{aligned}$$

En el sumador de video se generan automáticamente estas dos señales cromáticas diferenciales. Pero, a este respecto, durante la programación de los colores no hemos de tener en cuenta este hecho, nos limitaremos tan sólo a introducir en la memoria de instrucciones la combinación de colores correspondiente.

Consideraremos primeramente el modo de trabajo de los dos selectores de datos/multiplexores (8 a 1) con salidas triestado. En la figura 2.7 se muestra la asignación de conexiones de este componente 74251. Tenemos la tabla de funciones siguiente:

Entradas				Salidas	
S	C	B	A	Y	W
H	X	X	X	Z	Z
L	L	L	L	D0	D0
L	L	L	H	D1	D1
L	L	H	L	D2	D2
L	L	H	H	D3	D3
L	H	L	L	D4	D4
L	H	L	H	D5	D5
L	H	H	L	D6	D6
L	H	H	H	D7	D7

X = Nivel bajo o alto

Z = Salida de alta impedancia

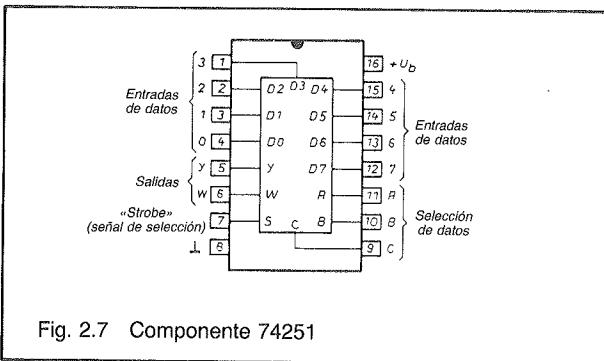


Fig. 2.7 Componente 74251

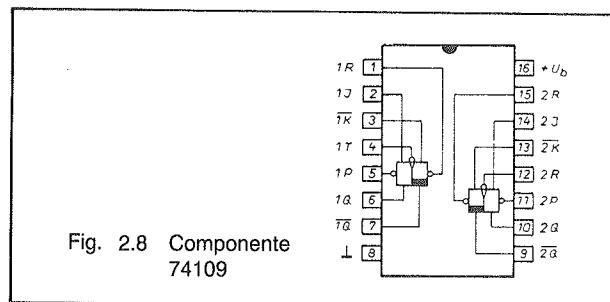


Fig. 2.8 Componente 74109

Pero antes debemos considerar el flip-flop 74109. En la figura 2.8 tenemos la asignación de conexiones. Tenemos la tabla de funciones siguiente:

Entradas					Salidas	
Preset	Borrado	T	J	K	Q	Q
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H*	H*
H	H	▼	L	L	L	H
H	H	▼	H	L	Q _n	Q _n
H	H	▼	L	H	H	L
H	H	L	X	X	Q _n	Q _n

X = Nivel bajo o alto

Q_n = Estado de memoria

H* = Este estado no es estable, es decir, se mantiene cuando «preset» y/o «clear» tienen un nivel alto.

Tenemos un flip-flop «disparado» con una entrada preset y otra para clear.

La serie superior de circuitos flip-flop para el mezclador de color consta de dos circuitos flip-flop de los tipos 74113 y 74109. Las dos entradas de sincronización de estos circuitos flip-flop están conectadas directamente a la frecuencia del cuarzo de 8,86 MHz a través de la puerta N0 G2. Por tanto, esa frecuencia se retarda sólo unos 9,5 ns.

La serie inferior de circuitos «flip-flop» del mezclador de color consta igualmente de los tipos 74113 y 74109. Las entradas de sincronización de ellos están reunidas, si bien obtienen su frecuencia en unos 20 ns. Las puertas N0 del tipo «Low Power Schottky» retardan una señal en 9,5 ns. Con ello se obtiene un retardo por puertas cuyo valor en tiempo es de 19 ns.

Advertencia: Los tiempos de retardo de las puertas G2 y G3 se compensan de modo que tan sólo las puertas N0, G15 y G16, retarden la señal de sincronización. Con ambos condensadores se obtiene un tiempo de retardo de unos 20 ns. Han de elegirse condensadores que no sean más pequeños pues, de lo contrario, fallará la función del modulador de color.

Si al equilibrar más tarde el computador surgen dificultades con el color, habrá que aumentar la capacidad de uno de los dos condensadores a 68 nF. No obstante, esta ampliación sólo deberá hacerse si surgen problemas.

Los dos condensadores de la línea de retardo se cargan a través de la resistencia interna de la puerta N0 74LS04. Esta resistencia tiene un valor de 130 Ω . Cuando un condensador se ha cargado hasta unos 2,4 V, la puerta N0 situada detrás admite un nivel alto. La salida de la puerta N0 se conmuta a nivel bajo y el condensador correspondiente se descarga instantáneamente.

El tiempo de proceso en la línea de retardo se ha establecido en 20 ns para que exista una gran seguridad. A la vez, si se producen anomalías en el equipo, el flip-flop FF6 deja bloqueado al FF7. De esta forma, la señal de sincronización retardada trabaja con un desfase de unos 180° respecto a la señal directa de la serie superior de circuitos flip-flop.

Consideramos más detenidamente el modo de trabajo de ambas series de circuitos flip-flop. En la figura 2.9 tenemos el diagrama de impulsos. En la primera línea tenemos los impulsos del oscilador de cuarzo de 8,86 MHz. El impulso para el nivel alto tiene una duración aproximada de 56 ns.

El impulso de la línea no retardada aparece directamente en los circuitos flip-flop FF5 y FF6 a través de la puerta G2. Con el flanco negativo se activa el flip-flop FF5 y la salida Q tiene un nivel alto (H). Con ese nivel alto, a su vez, se excita el FF6. Con el flanco positivo se activa el FF6. Con el flanco negativo siguiente, el FF5 bascula y la salida Q adquiere un nivel bajo (L). Este nivel bajo está en las entradas de excitación de FF6 y con un flanco positivo se desactiva y se repone FF6.

En el diagrama de impulsos de la figura 2.9 observamos en la 2.^a y 3.^a línea la forma de trabajo de este contador especial. Obtenemos una especie de contador en anillo. Los dos flip-flop FF5 y FF6 tienen ambos una frecuencia de salida de 4,43 MHz, pero

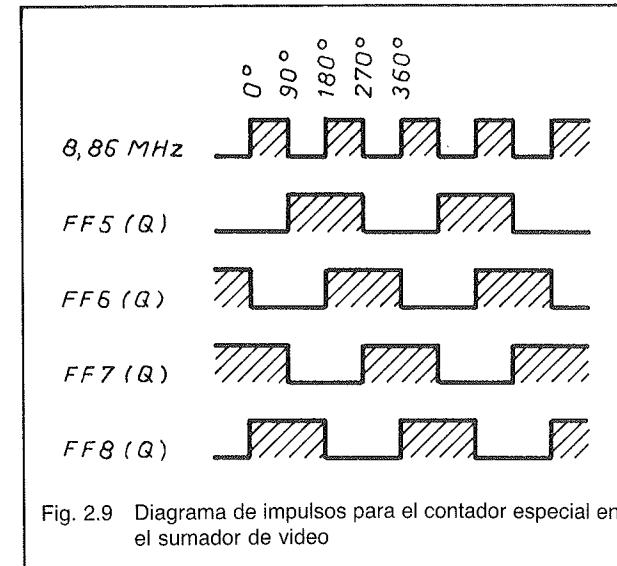


Fig. 2.9 Diagrama de impulsos para el contador especial en el sumador de video

el nivel alto del FF6 retrasa el nivel alto del FF5 en un ángulo exacto de 90°, cuando el oscilador de cuarzo tiene una relación del 50 %. Si no disponemos de esta relación se produce un error en el desfase digital y como consecuencia, un fallo de color en el aparato de televisión. Este fallo podemos corregirlo mediante el potenciómetro en el oscilador de cuarzo.

En el computador de televisión necesitamos este desfase digital para la frecuencia portadora de las dos señales diferenciales U y V.

La salida Q del FF6 está unida a la entrada J de activación del FF7, igualmente Q está unido con K. Si FF6 está activado, el FF7 se prepara para su activación; si se desactiva FF6 se prepara el FF7 para su desactivación. Mediante esta forma de conexión conseguimos un efecto de bloqueo entre la primera y la segunda serie de circuitos flip-flop. Normalmente podríamos omitir este boqueo, pero en tal caso ya no contaría con ningún dispositivo automático de corrección para el computador de televisión. Por

esta razón, el circuito para el color trabaja con una gran seguridad de servicio.

La señal de sincronización para la serie de flip-flop inferior llega retardada en unos 20 ns a las dos entradas de FF7 y FF8. Transcurridos unos 56 ns están preparadas las entradas de excitación para los flip-flops. Conseguimos, así, un ajuste de tiempo (timing) correcto para nuestro computador. El flip-flop FF7 podrá activarse cuando la señal de sincronismo retardada cambia de nivel alto a bajo. La salida Q de FF7 tiene un nivel alto y FF8 se activa. Con el flanco positivo retardado se activa el flip-flop FF8. La desactivación de ambos flip-flops se logra mediante el bloqueo antes mencionado y precisamente por un flanco negativo (74113) y el siguiente flanco positivo (74109). Obtenemos así las dos líneas últimas de la figura 2.9.

Antes de ocuparnos del circuito de la figura 2.1, hemos de analizar primeramente la composición de colores y la mezcla de ellos en el aparato de televisión.

La figura 2.10 nos muestra un círculo cromático con los 3 colores básicos azul, rojo, verde, junto con sus complementarios púrpura, amarillo y azul verdoso. Los colores básicos están dispuestos en una circunferencia.

El círculo comienza con 0° y el sentido de giro es contrario al de las agujas del reloj. El color en 0° corresponde a un azul tirando a violeta.

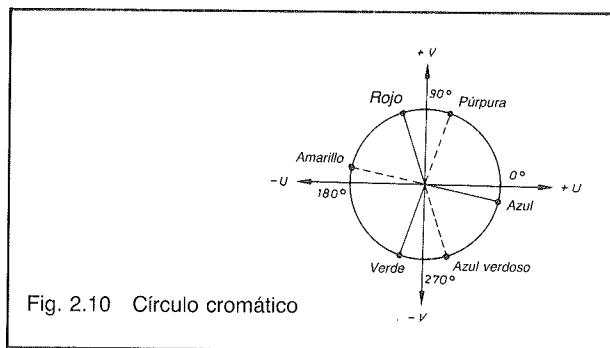


Fig. 2.10 Círculo cromático

Si nos movemos desde 0° a 100° obtenemos el primer color básico rojo. Después de otros 150° tenemos el color básico verde. Girando otros 90° tenemos el color básico azul. Con otros 20° más cerramos el círculo.

Este círculo cromático es recorrido por segundo con 4,43 MHz, es decir, tenemos cada 228 ns un círculo cromático completo.

Los colores básicos tienen un desfase mutuo de los ángulos siguientes:

Rojo-Azul:	120°
Rojo-Verde:	150°
Verde-Azul:	90°
	360°

Estos ángulos los genera el desfasador digital del computador de televisión.

Pero junto a los tres colores básicos hay también los complementarios. Se encuentran directamente enfrente de los básicos:

Color básico	Color complementario	Desfase
Rojo (100°)	Azul verdoso (280°)	180°
Verde (250°)	Púrpura (70°)	180°
Azul (340°)	Amarillo (160°)	180°

Los valores entre paréntesis están referidos al valor inicial 0° en el círculo cromático.

Mediante los colores básicos y los complementarios podemos determinar y programar los diferentes colores en nuestro computador de televisión. La programación se hace en la memoria de instrucciones. Mediante la interface de video programable conseguimos transformar la programación de color en una palabra de datos de tres posiciones que después llegarán a las tres salidas C1, C2 y C3. Con esa palabra de datos de tres posiciones comutaremos los colores a la salida Y de ambos multiplexores. Las salidas de ambos multiplexores están conectadas a

la salida de video a través de una resistencia de $1,5 \text{ k}\Omega$.

Las coordenadas de la figura 2.10 no se designan mediante x e y , sino con los símbolos de las señales diferenciales U y V : $x = U$, $y = V$.

Al estudiar el diagrama de impulsos de la figura 2.9, vemos el desfase digital de los diferentes impulsos que nos permitirán determinar los colores en nuestro computador de televisión. Con desfase de 90° tenemos un color entre rojo y púrpura, con 180° un amarillo con un poco de verde, con 270° un verde con un ligero porcentaje de azul y con $360^\circ = 0^\circ$ un azul oscuro tirando a violeta. A este respecto se conectan las salidas de los flip-flops a las entradas de ambos multiplexores, según se aprecia en la página siguiente.

En ambos multiplexores la entrada 0 está unida a masa, es decir, si las entradas de selección reciben una palabra de señal-LLL desde las salidas de la interface de video programable, tendremos en la pantalla de color blanco.

La entrada 7 de los multiplexores está unida a $+5V$ con lo que obtenemos el color negro, pantalla oscura, cuando en las entradas de selección recibimos de la interface de video una palabra de señal HHH.

La composición de colores tiene lugar en el sumador de video de una manera relativamente simple. La salida Q del flip-flop FF5 es la salida complementaria de la \bar{Q} . Puesto que este flip-flop se activa con un desfase de 90° obtendremos en la salida Q el color básico rojo. El complementario lo obtenemos de la salida \bar{Q} del flip-flop siendo un azul verdoso. Despues de 180° se activa el FF6. En su salida Q tendremos el color amarillo. Este es el complementario del básico azul que obtendremos en la salida \bar{Q} del flip-flop.

Después de 270° se activa el FF7. En la salida Q obtendremos el color básico verde y en la salida \bar{Q} el complementario púrpura.

Después de 360° se activa el FF8. Sin embargo,

	FF8	FF7	FF6	FF5	Entradas de multiplexor	Desfase
Nr.	Q	\bar{Q}	Q	\bar{Q}	Q	\bar{Q}
0	H	L	L	H	L	H
1	L	H	L	H	L	H
2	L	H	L	H	L	H
3	H	L	L	H	L	H
4	H	L	L	H	L	H
					3, 5 und 6	0° ... 90°
					1, 3 und 5	90° ... 180°
					1, 2 und 4	180° ... 270°
					2, 4 und 6	270° ... $360^\circ(0^\circ)$
					3, 5 und 6	0° ... 90°

Multiplexor superior

	FF8	FF7	FF6	FF5	Entradas de multiplexor	Desfase
Nr.	Q	\bar{Q}	Q	\bar{Q}	Q	\bar{Q}
0	H	L	L	H	L	H
1	L	H	L	H	L	H
2	L	H	L	H	L	H
3	H	L	L	H	L	H
4	H	L	L	H	L	H
					1, 2 und 3	0° ... 90°
					3, 5 und 6	90° ... 180°
					4, 5 und 6	180° ... 270°
					1, 2 und 4	270° ... $360^\circ(0^\circ)$
					1, 2 und 3	0° ... 90°

Multiplexor inferior

para el multiplexor superior este flip-flop no tiene función alguna.

Las salidas de los circuitos flip-flop están unidas al multiplexor superior. Por ello con la siguiente programación de la interface de video, obtenemos los colores:

Interface de video			Colores
C3	C2	C1	
0	0	0	Blanco
0	0	1	Rojo
0	1	0	Verde
0	1	1	Azul
1	0	0	Amarillo
1	0	1	Púrpura
1	1	0	Azul verdoso
1	1	1	Negro

El multiplexor superior puede trabajar cuando existe un nivel L en la entrada CBF y en la PAL-SW respectivamente. El multiplexor inferior durante ese tiempo queda bloqueado, ya que detrás de la puerta 0 exclusiva hay una puerta NO.

Aquí necesitamos hacer algunas consideraciones de tipo teórico: En el procedimiento PAL se escribe primero la línea a. Después la línea b. La salida OE del generador-PAL, teniendo en cuenta la entrada PAL-SW, determina qué línea ha de escribirse:

$$\begin{aligned} \text{PAL-SW} = L &: \text{línea a} \\ \text{PAL-SW} = H &: \text{línea b} \end{aligned}$$

Se designa como «phase alternating» (alternancia de fase) la conmutación de la línea a a la b y viceversa. En principio, la conmutación tiene lugar entre la línea a y la línea b. Precisamente la denominación PAL procede del hecho de alternancia entre las dos líneas:

PAL = phase alternating line

La figura 2.11 nos muestra la relación en la conmutación entre las dos líneas. La componente U de

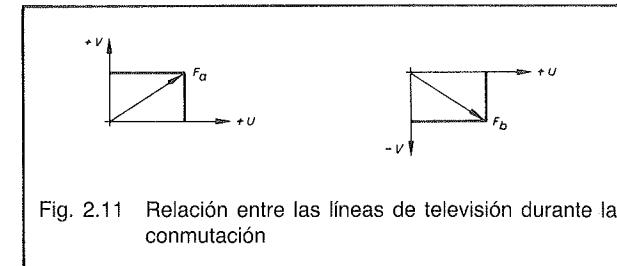


Fig. 2.11 Relación entre las líneas de televisión durante la conmutación

la señal de tipo de color es la misma, pero el signo de la componente V es distinto. En la línea a la componente V va adelantada 90° y en la línea b retrasada 90°. Esto da lugar a un desfase de 180° que podremos comutar fácilmente mediante la lógica de ambos multiplexores. El generador PAL-2621 comunica automáticamente este proceso.

La composición de color para la línea b es relativamente igual de sencilla. La salida \bar{Q} del flip-flop FF5 es la salida complementaria de la Q. Puesto que este flip-flop se activa después de 90°, la salida Q tendrá el color rojo. El complementario es azul verdoso. Pero puesto que ahora se produce una conmutación de línea tendremos que intercambiar adecuadamente las conexiones para el multiplexor inferior.

Después de 180° se activa el FF6. La salida Q genera el color amarillo. Este es el complementario del azul que obtenemos en la salida \bar{Q} del FF6.

Después de 270° se activa el FF7, pero éste no está conectado al multiplexor inferior.

Después de 360° se activa el FF8 y obtenemos en la salida Q el color básico verde y en la \bar{Q} el complementario púrpura.

Después de 360° se cierra el círculo cromático de la figura 2.10.

Las salidas de los circuitos flip-flop están conectadas al multiplexor inferior. Por esta razón, programando convenientemente la interface de video, obtenemos los colores siguientes:

Interface de video			Colores
C3	C2	C1	
0	0	0	Blanco
0	0	1	Rojo
0	1	0	Verde
0	1	1	Azul
1	0	0	Amarillo
1	0	1	Púrpura
1	1	0	Azul verdoso
1	1	1	Negro

Si comparamos las tablas funcionales, o de verdad, para programar ambos multiplexores, deducimos que la programación es la misma, con lo que ésta se simplifica notablemente puesto que no hemos de programar semi-imágenes separadas.

La figura 2.12 muestra la placa para el sumador de video y la figura 2.13 muestra la disposición de los componentes. La fotografía de la figura 2.14 corresponde a la parte de los componentes de la placa.

Lista de componentes para el sumador de video

- 1 Puerta NAND 74LS00
- 2 Puerta NO 74LS04
- 1 Puerta AND 74LS08
- 1 Flipflop 74LS109
- 3 Flipflop 74LS113
- 2 Puerta de equivalencia 74LS136
- 2 Multiplexor 74LS251
- 1 Cuarzo con 8,867238 MHz
- 1 Condensador de ajuste 1,8 pF...35 pF
- 2 Condensadores 22 μ F/16 V
- 3 Condensadores 47 pF
- 1 Condensador 220 pF
- 2 Regulador 5 k Ω
- 1 Resistencia 220 k Ω
- 2 Resistencia 470 k Ω
- 1 Resistencia 820 k Ω
- 12 Resistencia 1 k Ω
- 1 Resistencia 1,5 k Ω
- 3 Resistencia 2,2 k Ω
- 1 Resistencia 4,7 k Ω
- 1 Resistencia 33 k Ω

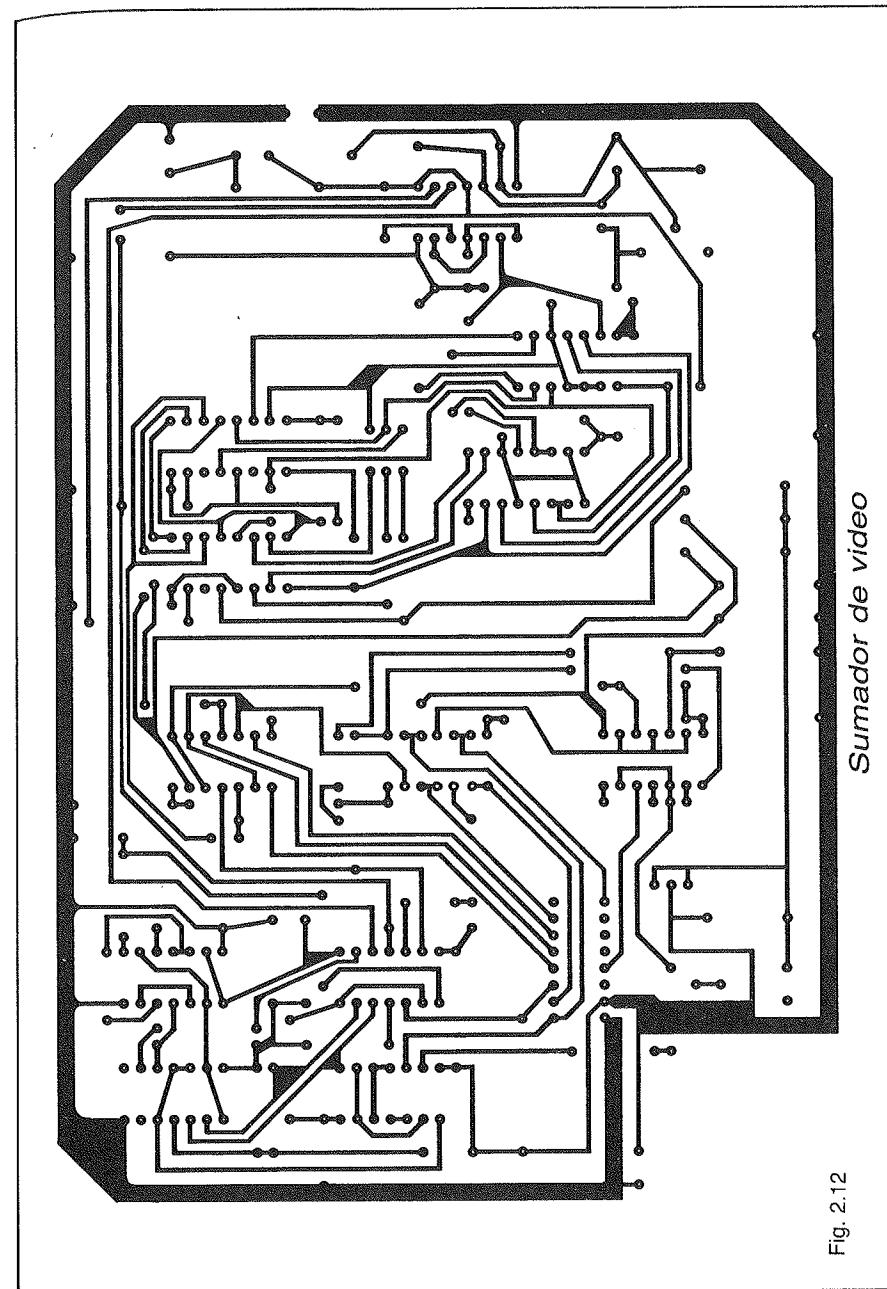


Fig. 2.12

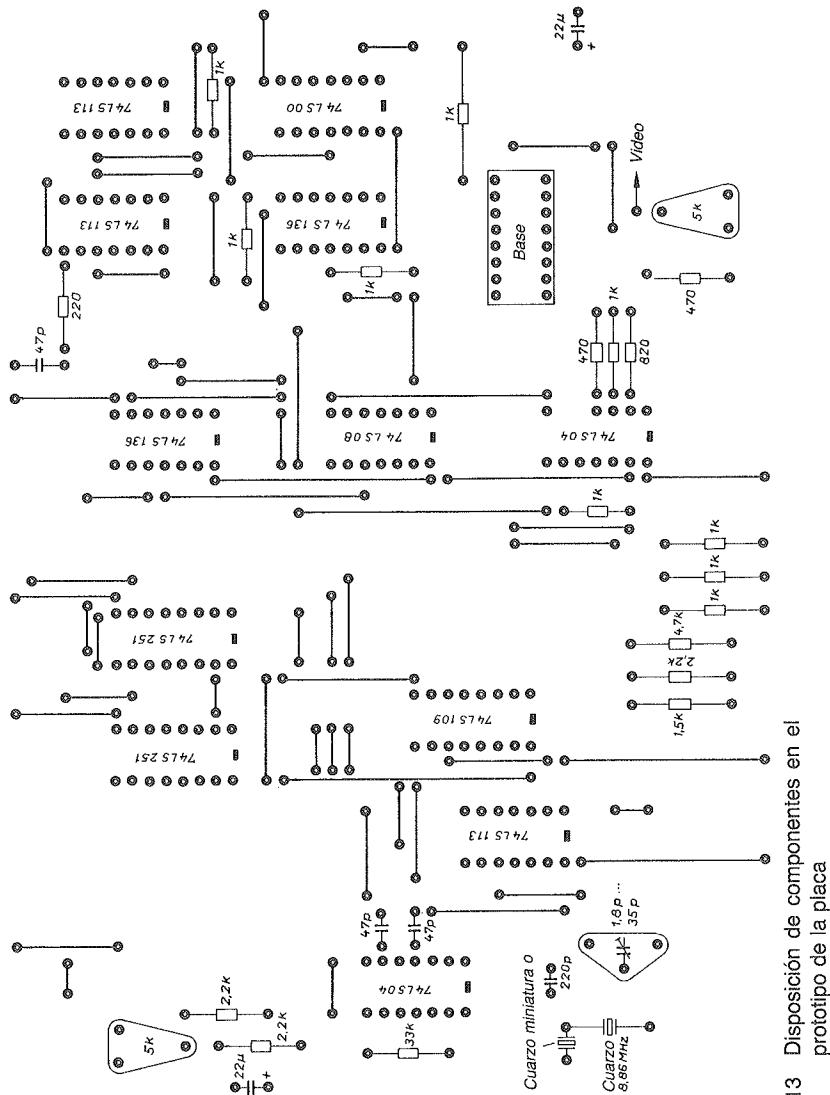


Fig. 2.13 Disposición de componentes en el prototipo de la placa

2.2 MICROPROCESADOR, INTERFACE DE VIDEO PROGRAMABLE, GENERADOR PAL Y MEMORIA DE INSTRUCCIONES PARA EL COMPUTADOR TV

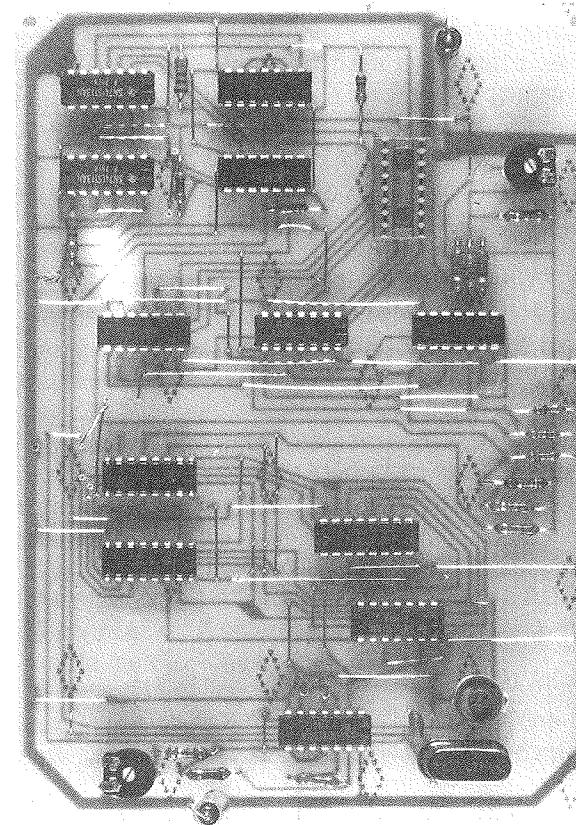


Fig. 2.14 Fotografía de la parte de componentes del sumador de video

Para esta unidad compleja, la figura 2.15 nos muestra las conexiones. En el siguiente capítulo, designaremos este esquema simplemente como placa del microprocesador. El esquema de la figura 2.15 es el cerebro de nuestro sistema computador.

La entrada «Sense» (terminal 1) del 2650 está unida a la entrada «VRST» de la interface de video programable 2636. Ambas entradas reciben sus señales de la salida «VRST» del generador PAL 2621. De esta forma, entre el 2650, el 2636 y el 2621, existe una línea para la sincronización de la reposición vertical.

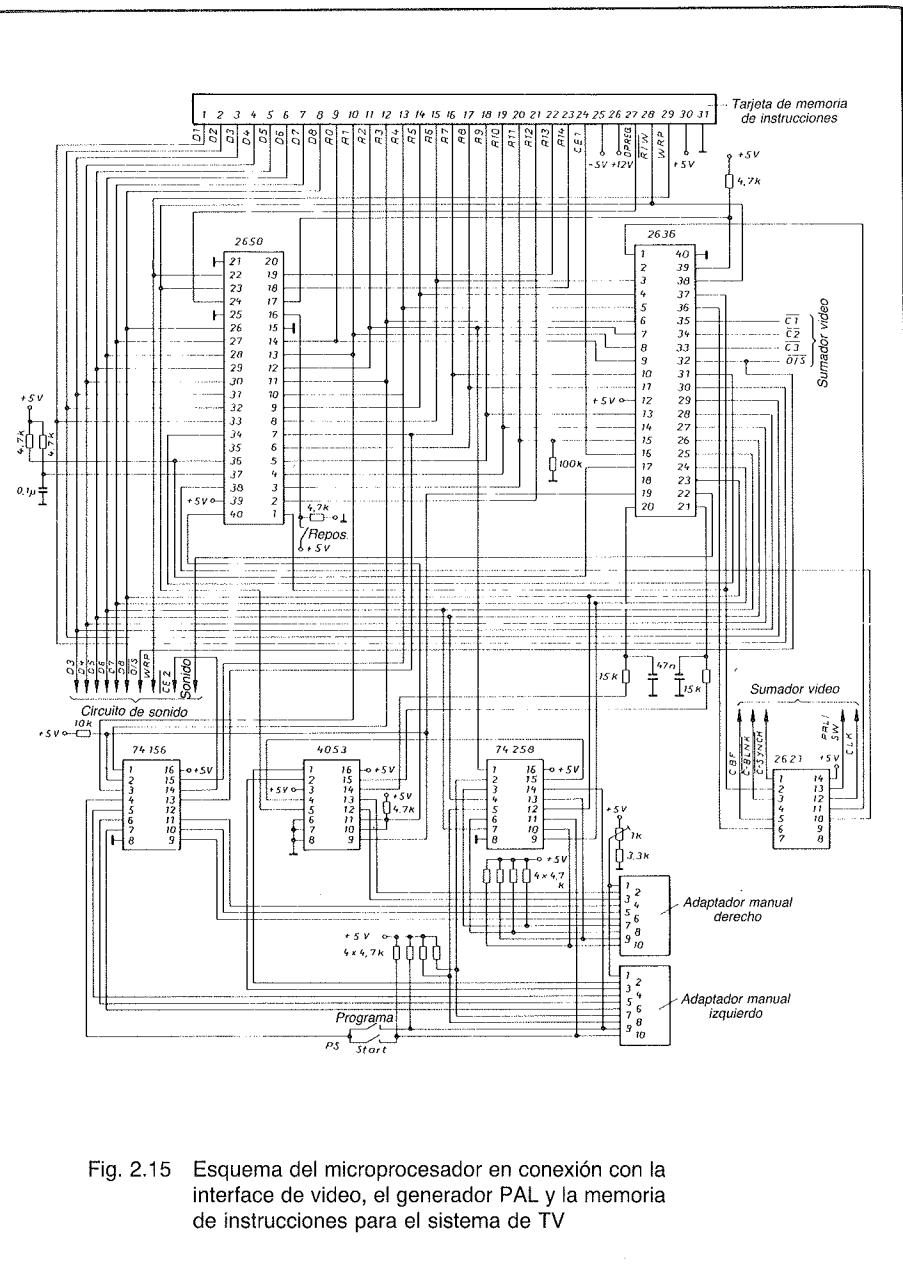


Fig. 2.15 Esquema del microprocesador en conexión con la interfaz de video, el generador PAL y la memoria de instrucciones para el sistema de TV

Del terminal 2 al 14 tenemos el bus de direcciones del 2650. Las direcciones salen del microprocesador y quedan disponibles para la interface de video programable, la tarjeta de la memoria de instrucciones, el componente TTL 74156 y el 74258.

El 2650 tiene en total 15 salidas de direcciones. Del terminal 2 al 14 obtenemos las direcciones ADRO a ADR12 y del terminal 18 al 19 las ADR13 y ADR14. Para nuestro sistema microprocesador interno necesitamos sólo las direcciones ADRO y ADR11 y las ADR13 y ADR14 son necesarias para un computador de ajedrez. ADR13 y ADR14 direccionan una tarjeta de memoria de instrucciones especial de un decodificador, para que sea posible elegir posiciones o direcciones adicionales de una memoria de lectura-escritura donde se contienen instrucciones complementarias. Lo trataremos más adelante.

La interface de video programable obtiene sus direcciones de ADRO a ADR11. La dirección ADR12 está conectada directamente con la tarjeta de memoria de instrucciones.

El circuito integrado TTL 74156 son dos decodificadores/demultiplexores binarios de 2 bits con colectores abiertos en las salidas. La figura 2.16 muestra la conexión interna y el esquema de este compo-

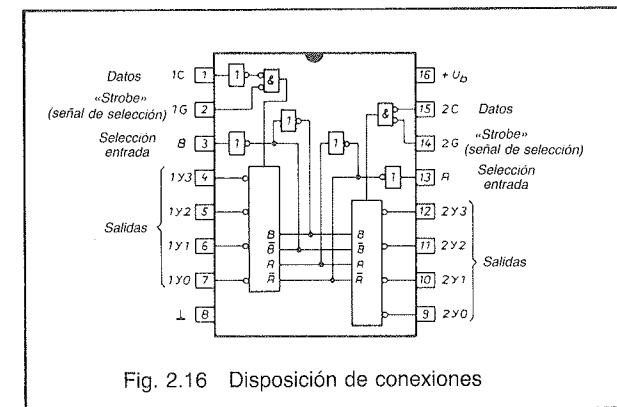


Fig. 2.16 Disposición de conexiones

nente. Disponemos de la tabla de funciones siguientes:

Entradas			Salidas				
«Select»	«Strobe»	Datos	C	Y0	Y1	Y2	Y3
X	X	H	X	H	H	H	H
L	L	L	H	L	H	H	H
L	H	L	H	H	L	H	H
H	L	L	H	H	H	L	H
H	H	L	H	H	H	H	L
X	X	X	L	H	H	H	H

X ≈ nivel bajo o alto

El circuito integrado TTL 74156 obtiene las señales de control del bus de direcciones del sistema microcomputador y de la interface de video programable. Para el bus de direcciones se deduce la disposición siguiente:

1C: ADR3
Select B: ADR1
2C: ADR4
Select A: ADR 7

Las dos entradas «strobe» del 74156 están conectadas entre sí y obtienen su señal de disponibilidad de la salida \overline{CE}_2 de la interface de video programable. Con la salida del circuito integrado 74156 controlamos los campos de servicio de nuestros dos adaptadores manuales. Trataremos en detalle, más adelante, estos adaptadores manuales.

El circuito integrado TTL 74258 es un cuádruple selector/multiplexor de datos 2 a 1 con salidas triestado. La figura 2.17 muestra el esquema de conexiones. Tenemos la tabla de funciones de la página siguiente.

Con nivel alto en los controles de salida podemos conmutar las salidas de alta impedancia y con-

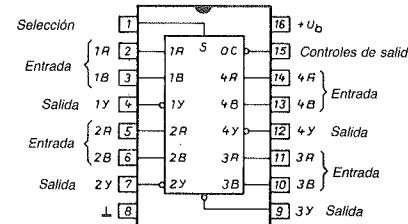


Fig. 2.17 Esquema de conexiones del 74258

Entradas				Salidas	
Controles de salida	«Select»	A	B	1Y, 2Y, 3Y, 4Y	
H	X	X	X		Z
L	L	L	X		H
L	L	H	X		L
L	H	X	L		H
L	H	X	H		L

X ≈ Nivel bajo o alto

Z ≈ Salida de alta impedancia

seguimos con ello una salida triestado bloqueada. Las salidas 1Y, 2Y, 3Y y 4Y están conectadas directamente al bus de datos del sistema microprocesador. Puesto que tenemos un bus, necesitamos para ello un comportamiento triestado en las correspondientes salidas.

La entrada «select» recibe su señal del bus de direcciones del microprocesador. Para ello utilizaremos la dirección ADR2. El control de salida del circuito integrado TTL 74258 está conectado al circuito integrado CMOS 4053. El 74258 recibe su señal de selección procedente del 4053 por el bus de direcciones a través de la salida Q_c . El circuito integrado 74258 puede trabajar sólo cuando el circuito integrado CMOS ha sido direccionado adecuadamente por el software de nuestro programa. Este direccionamiento parece aparentemente algo complicado, pe-

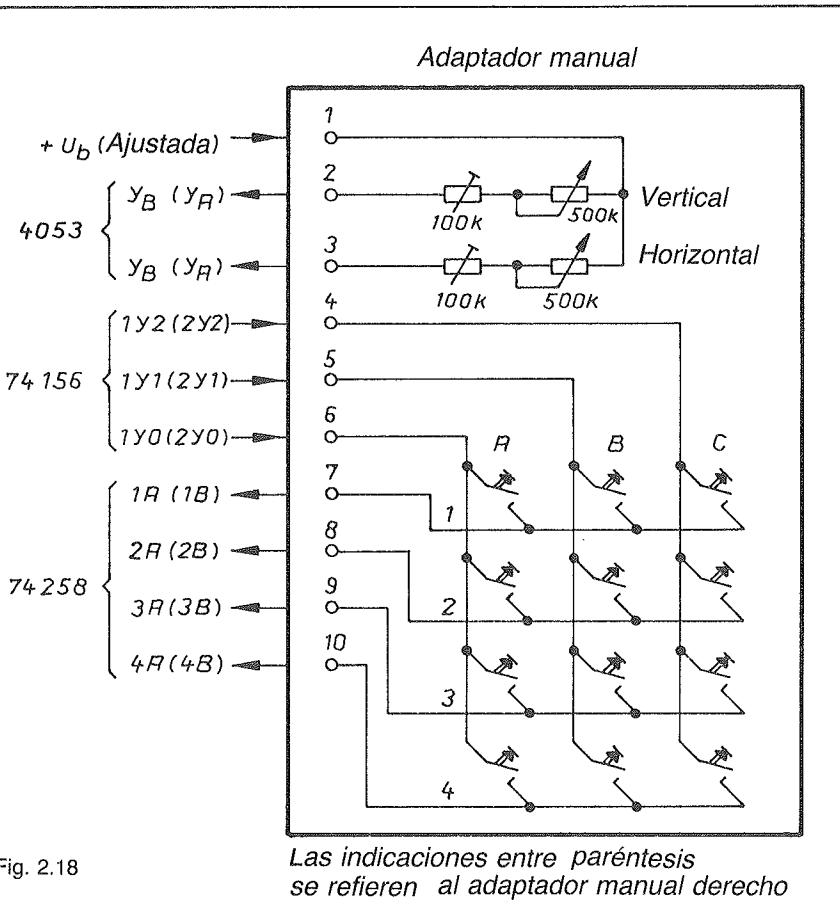


Fig. 2.18

ro así podremos ahorrar componentes y rebajar el precio de nuestro computador-TV.

El modo de trabajo de los componentes TTL 74156 y 74258 nos será comprensible si examinamos las conexiones internas de los adaptadores manuales. En la figura 2.18 se muestra la conexión interna de ambos adaptadores para nuestro computador.

En el terminal 1 de ambos adaptadores se dispone la tensión de servicio para los potenciómetros in-

ternos. Las salidas de éstos están unidas a los PORT1 y PORT2 de la interface de video 2623 a través de un conmutador analógico CMOS. De esa forma se deriva el modo de trabajo siguiente:

Cuando el microprocesador, o la interface de video programable, necesita conocer el ángulo de giro del potenciómetro correspondiente para calcular coordenadas o para determinar la ubicación de un objeto, el componente CMOS obtiene la selección a partir de la salida FLAG del 2650 y de la entrada CE de la interface de video 2636. Para la consulta disponemos de la tabla de funciones siguiente:

FLAG	CE2	Función
L	L	Potenciómetro horizontal y disponibilidad para el 74258
L	H	Potenciómetro vertical y disponibilidad para el 74258
H	L	Potenciómetro horizontal y bloqueo para el 74258
H	H	Potenciómetro vertical y bloqueo para el 74258

Hemos de tener en cuenta estas diferentes funciones en el momento de elaborar nuestro programa.

Los dos adaptadores manuales tienen, además de los potenciómetros, otras 12 teclas para introducir informaciones en el computador-TV. Estas teclas deberían ser sin tope elástico, en la medida de lo posible, con el fin de que no se produzcan informaciones erróneas durante la entrada. Simultáneamente con ésta tiene lugar una consulta de las teclas por parte del computador. Por esta razón, hemos de programar un bucle de consulta de teclas aparte. Despues de cada ejecución del programa tendrá lugar una consulta automática.

La consulta de las distintas teclas tiene lugar según el principio de multiplexación (8). El circuito integrado 74156 obtiene del bus de direcciones de

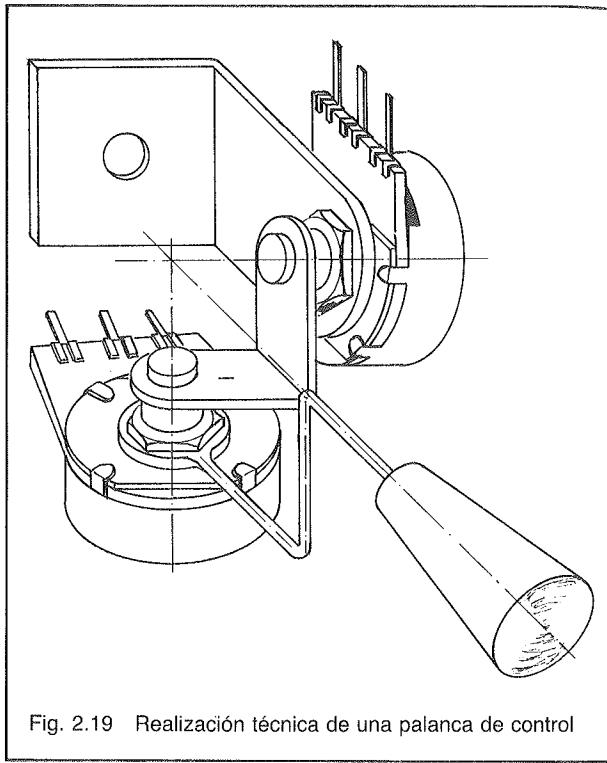


Fig. 2.19 Realización técnica de una palanca de control

nuestro microprocesador sus informaciones en forma de una dirección. Esta información sobre direcciones se convierte en un total de cuatro líneas mediante el multiplexor interno del circuito integrado TTL 74156. La salida Y0 del 74156 genera la columna A en el adaptador manual, la salida Y1 genera la columna B y la salida Y2 la columna C. Gracias a ello podremos establecer nuestro programa en esas tres columnas. La salida Y3 del circuito integrado 74156 selecciona las dos teclas de servicio «Program» y «Start».

Después de cada ejecución de programa, un bucle adicional selecciona continuamente las dos teclas de servicio «Program» y «Start». Con la tecla «Program» ponemos en acción los programas co-

rrespondientes. Estos programas se eligen uno tras otro.

Con la tecla «RESET» ponemos el 2650 apuntando a la dirección ADRO y comenzamos con el primer programa al que designaremos, por ejemplo, como Programa I. Cuando ha terminado la ejecución del mismo se produce una consulta automática de la tecla «Program». Si no está pulsada, el microprocesado acaba la programación del Programa I, pero si está pulsada, el 2650 salta automáticamente al Programa II. Tras una ejecución de dicho programa tiene lugar otra nueva consulta a la tecla «Program». Si no está pulsada, el Programa II permanece; si lo está, se procesa el Programa III.

Sin embargo, el programa se acepta, en principio, sólo si hemos pulsado la tecla «Start».

Para los movimientos horizontales y verticales de los objetos necesitamos dos potenciómetros, con valores de $500\text{ k}\Omega$ cada uno. Pero la disposición de ellos implica dificultades ya que han de conectarse como palancas de control. Para ello son adecuadas las palancas de control de los dispositivos de control remoto tales como los de los modelos a escala. Por desgracia, estos mandos tienen la mayoría de ellos valores óhmicos de unos $20\text{ k}\Omega$ y su coste es algo elevado.

En la figura 2.19 se indica una propuesta para la realización técnica de un mando de control. En un angular de sujeción se encuentra el potenciómetro para los movimientos horizontales. Si movemos la palanca de arriba a abajo, o al contrario, se modificará el ángulo que gira el potenciómetro. El eje del potenciómetro vertical está unido al del horizontal mediante otro angular. El mando de control propiamente dicho se sitúa bajo la rosca de fijación del potenciómetro horizontal. Si movemos la palanca de derecha a izquierda, o viceversa, se modificará el ángulo del potenciómetro vertical.

El mando de control puede estar provisto también de un commutador para la función de disparo si el computador-TV se va a emplear para juegos de

guerra. Para ello conectaremos un simple interruptor, o pulsador, en paralelo con el interruptor del adaptador manual en la línea I/Columna B. La consulta de ese interruptor tendrá lugar en paralelo con la del pulsador en el adaptador manual de la línea I/Columna B. Aquí no hay barrera para nuestra fantasía.

Podemos mejorar sensiblemente la estabilidad mecánica de nuestro diseño de la figura 2.19, si hacemos pasar el mando por una ventanilla rectangular. Con ello conseguimos limitar la posibilidad de esfuerzos mecánicos excesivos en caso de movimientos energéticos verticales y horizontales cuando disfrutemos de un juego apasionante.

La ventaja del diseño mecánico de la figura 2.19 es la posibilidad de desplazamiento de los objetos, simultáneamente en dirección vertical y horizontal. Esto es muy importante e interesante cuando hemos programado, por ejemplo, símbolos que sean aviones.

Si mediante la ventanilla rectangular limitamos mucho el ángulo de giro de los potenciómetros de $500\text{ k}\Omega$, los cambiaremos por otros de $750\text{ k}\Omega$ o $1\text{ M}\Omega$. En la posición de reposo del mando de control, los potenciómetros habrán de tener un ángulo de giro medio. Mediante el potenciómetro de $1\text{ k}\Omega$ puede ajustarse la tensión de servicio para ellos y con los potenciómetros de $100\text{ k}\Omega$ realizamos el ajuste.

La consulta de los cuatro potenciómetros se realiza mediante el circuito integrado CMOS 4053. Los dos potenciómetros del adaptador manual derecho están unidos a las entradas X_A y Y_A y los potenciómetros del adaptador izquierdo a X_B y Y_B . La regulación y el control se consiguen mediante la salida «FLAG» del microprocesador (para la salida en colector abierto se necesita una resistencia) y mediante la salida $\overline{\text{CE2}}$ de la interfaz de video programable.

Los potenciómetros verticales se reúnen y comutan a PORT2 y los horizontales a PORT1. Con ello, la interfaz de video programable podrá consul-

tar los ángulos de los potenciómetros. Cada ándulo de giro genera una tensión diferente que queda disponible como valor analógico. La interfaz de video obtiene el ángulo de giro como valor digital gracias a los dos convertidores analógico-digitales internos, pudiendo así elaborarlo por sí mismo o bien transferirlo al 2650 a través del bus de datos. Eso depende del caso de que se trate.

En lugar de potenciómetros pueden utilizarse micrófonos especiales. Con ellos pueden convertirse sonidos o ruidos en tensiones que más tarde son convertidores analógico-digitales) para nuestro computador-TV. Tampoco aquí hay barreras para nuestra fantasía.

El terminal 15 del 2650 es la entrada ADREN de la figura 2.15. Esta entrada está unida a la masa de la tensión de alimentación. Con ella consigue el microprocesador dejar disponible el bus de direcciones. Las direcciones ADRO a ADR12 quedan así conectadas. Si esta entrada de control no estuviera unida a masa, las salidas de direcciones del microprocesador tendrían un estado de alta impedancia. Mediante la conexión a masa nos ahorraremos una línea adicional en el sistema de control del bus y también varias instrucciones de programación.

El terminal 16 es la entrada RESET para el 2650. Esta entrada queda conectada a la tensión de servicio de $+5\text{V}$ a través de un interruptor, o pulsador, simple cada vez que se pulsa este último. Cuando pulsamos el interruptor, el microprocesador se sitúa inmediatamente en la dirección 0000 y allí permanece. De esa forma se interrumpe inmediatamente cualquier programa. En estado de trabajo, la entrada RESET está unida a masa mediante una resistencia de $4,7\text{ k}\Omega$. La caída de tensión en la resistencia basta para que la entrada RESET pueda admitir un nivel bajo. Si la reposición (reset) no fuera posible, se disminuirá el valor de la resistencia a $2,2\text{ k}\Omega$.

El terminal 17 es la entrada INTREQ para el microprocesador. Esta entrada se utiliza para que el 2650 pueda obtener de sus periféricos externos la

instrucción que inicie una secuencia de interrupción de programa. La entrada INTREQ está conectada directamente con la salida INTREQ de la interface de video 2636. Con ello esta última está siempre en situación de modificar el programa del microprocesador en caso de necesidad. El componente 2636 tiene un control interno de fondo para la pantalla, el cual a su vez dirige el control de interrupciones y el control de estado. Si el componente de interface necesita datos o instrucciones nuevas, con el fin de que la imagen permanezca en buen estado en el receptor de televisión, el 2636 emite un impulso corto en su salida INTREQ con lo que el microprocesador abandona su programa principal y salta a un subprograma, puesto que se encuentra ante una secuencia de interrupciones. Cuando el microprocesador ha finalizado esa secuencia, la interface de video recibe la señal correspondiente.

Los terminales 18 y 19 son las direcciones más altas de nuestro microprocesador. Se necesitan sólo para el juego de ajedrez. Las dos conexiones están directamente unidas a la tarjeta de instrucciones. Para juegos sencillos, ello no es necesario. La dirección ADR13 es una señal que se aprovecha en muchas ocasiones. Durante el servicio de tipo M/I/O, cuando el microprocesador se encuentra en la fase M, tenemos directamente la dirección ADR13. Con una instrucción I/O, la salida distingue, no obstante, entre una «extendida» o una «no-extendida», cuando está en la fase-I/O. Esto se aplica también para la dirección ADR14 o para Datos/Control. Durante la fase M de la conexión M/I/O significa directamente la dirección ADR14 y en instrucciones I/O distingue entre instrucciones de datos e instrucciones de control.

El terminal 20 es la salida-M/I/O de nuestro microprocesador. Indica a los componentes externos si tienen ante sí una función de almacenamiento o una función I/O. Esta entrada sólo se necesita en programas muy complicados como ocurre, por ejemplo, en los de ajedrez. Por esta razón no se ha conectado en la figura 2.15.

El terminal 21 es la conexión a masa del microprocesador 2650.

El terminal 22 es la conexión WRP del microprocesador 2650. Se trata de una salida de control para la lectura de la parte de sonido y de la memoria de instrucciones. En programas sencillos esta salida se utiliza sólo para la parte de sonido del computador TV. En programas de juegos más complejos la utilizaremos para lectura de la memoria de instrucciones.

El terminal 23 es la salida R/W del microprocesador 2650. En nuestro sistema computador los restantes elementos reciben, a través de esta salida, una señal durante una fase de lectura/escritura. Con ella identifican si el bus de datos, que trabaja en modo bidireccional, está comutado para entrada (I = Input) o para salida (O = Output). La interface de video programable, la tarjeta de memoria de instrucciones y el componente CMOS 4053 reciben, así, las informaciones correspondientes.

El terminal 24 es la salida OPREQ del microprocesador. Con una señal de petición, el microprocesador indica a sus unidades externas que son válidas todas las informaciones de las demás conexiones. Esto se aplica para los buses de datos y de direcciones. La interface de video programable y la tarjeta de memoria de instrucciones reciben esta información a través del bus de control.

El terminal 25 está directamente unido a masa. Así la entrada DBUSEN tiene un nivel bajo para que el bus de datos quede liberado y los buffers triestado ya no queden bloqueados, es decir, a alta impedancia. Los datos o informaciones pueden transferirse directamente.

Entre los terminales 26 y 33 tenemos el bus de datos de nuestro sistema computador. El bus de datos trabaja en modo bidireccional, es decir, pueden desplazarse los datos en una y otra dirección del bus. El microprocesador 2650 es del tipo de 8 bits y, por esta razón, tenemos 8 líneas en paralelo para los datos. El sistema de bus de datos está conectado

con la tarjeta de memoria de instrucciones, la interfaz de video programable, la sección de audio y el componente TTL 74258. De esa forma, podremos determinar el flujo de datos mediante nuestro programa almacenado en la tarjeta de memoria de instrucciones. Este flujo de datos no implica solamente a éstos, sino también a las instrucciones, direcciones recién calculadas y eventualmente también informaciones de control que pueden transmitirse a través del bus. Con ello se completa la estructura de nuestro programa tanto en lo referente al hardware como al software.

El terminal 34 es la salida INTACK del microprocesador. Mediante esta salida, el microprocesador puede informar a la interfaz de video programable de que se ha producido una secuencia de interrupción de programa. El microprocesador proporciona inmediatamente a su bus de datos las nuevas informaciones de interrupción que serán aceptadas por la interfaz de video programable en este mismo momento, pues la secuencia de interrupción y control contiene una instrucción que desactiva a la interfaz de video. Este procedimiento se denomina servicio de acuse de recibo para programas de interrupción entre interfaz de video y microprocesador.

El terminal 35 del microprocesador no se utiliza en nuestro sistema, ya que el 2650 no queda en estado de trabajo/espera por efecto de ninguna de las unidades externas.

El terminal 36 es una entrada especial de control del microprocesador. La interfaz de video programable da una señal a la entrada OPACK cuando ha finalizado sus fases de almacenamiento o de E/S. Podemos denominar ciclos, o ciclos de trabajo, a estas fases. De esa manera, el microprocesador podrá reanudar su programa principal una vez finalizado un ciclo de interrupción.

El terminal 37 es la entrada «Pause» de nuestro microprocesador. Esta entrada se encuentra a tensión de servicio debido a una resistencia de trabajo externa («pull-up») y tiene, por tanto, un nivel alto.

Por esta causa, no podemos realizar ningún acceso directo a memoria, lo que, sin embargo, necesitamos obligatoriamente en el sistema de regeneración de imagen de RAM. Pero puesto que la entrada está a nivel alto, no se derivan de ello graves problemas para nuestra programación del EPROM en la tarjeta de memoria de instrucciones; en resumen, nuestro programa se simplifica notablemente. En el caso de superjuegos, podremos utilizar esta entrada en juegos con regeneración de imagen de RAM, pero la imagen de televisión permanecerá sólo un corto tiempo debido a que no hay señales de control para acceso directo a memoria.

El terminal 38 es la entrada de sincronización para el microprocesador. El generador PAL 2621 genera en su salida CK4 una frecuencia de 886 kHz para el 2650. Esta señal se genera a partir de la señal de 3,546 MHz del sumador de video. La frecuencia de 3,546 MHz se divide por 4 y se obtiene así la frecuencia de 0,886 MHz. Con ello, la frecuencia de 8,86 MHz del cuarzo se subdivide primero en la relación 1:2,5 y después en la 1:4. Los 0,886 MHz corresponden, por tanto, a 1/10 de la frecuencia del cuarzo. Gracias a los distintos divisores se obtiene una relación del 50 % y con ello un modo de trabajo muy seguro para nuestro sistema microprocesador.

El terminal 39 es la conexión a la tensión de servicio.

El terminal 40 es la salida FLAG para nuestro microprocesador 2650. Esta salida es independiente de las restantes señales de E/S y representa una salida directa. La salida FLAG está conectada al componente CMOS 4053 y sirve para controlar los adaptadores manuales, es decir, para control de los potenciómetros asociados o del teclado.

Después de tratar el microprocesador 2650, debemos describir ahora el funcionamiento de la interfaz de video programable 2636 en la conexión del esquema de la figura 2.15.

El terminal 1 del componente 2636 es la entrada PCK. Aquí tiene lugar la sincronización por el gene-

rador PAL mediante la frecuencia de línea de 3,5 MHz o con un impulso de longitud 282 ns. Por cada línea obtenemos 227 impulsos de sincronización. La longitud de impulso de línea es de 64 μ s. La entrada PCK controla el reloj interno y la lógica de control.

El terminal 2 no tiene ninguna función.

Los terminales 3 al 15 son las patillas de conexión para el bus de direcciones, a excepción de terminal 12. El terminal 12 es la conexión para la tensión de servicio $U_b = +5V$. Las conexiones para las direcciones se unirán con el sistema de buses, de acuerdo con sus respectivos valores.

El terminal 16 de la interfaz de video programable tiene una función particular. Esta salida CE1 controla la tarjeta de memoria de instrucciones de nuestro microprocesador. Si la interfaz de video trabaja entre las direcciones 000 a DFF, la salida CE1 tiene un nivel alto y podrán trabajar las EPROM, las RAM y los generadores de caracteres. La dirección siguiente a la DFF es la E00. A partir de esa dirección E00, la interfaz de video emite un nivel bajo a la tarjeta de memoria de instrucciones, dejándola bloqueada. Ha de examinarse la ocupación de memoria para nuestro sistema en el apartado 1.2, con el fin de evitar dificultades de elaborar más tarde un programa de juego. Desde la dirección 000 a la DFF tenemos disponibles 3584 direcciones de memoria de 8 bits cada una. Podremos conseguir 3072 direcciones mediante tres EPROM del tipo 2078 con la conexión anterior de un simple decodificador TTL. Las 512 posiciones de memoria restantes podemos almacenarlas mediante una EPROM 2702, cuya constitución es análoga a la 2708. Así pues, podremos realizar toda la memoria de instrucciones con cuatro memorias EPROM.

El terminal 17 es la salida OPACK de la interfaz de video programable que está unida directamente con la entrada OPACK del microprocesador. Con ella la interfaz de video indica al microprocesador que ha finalizado un proceso de trabajo. Este puede ser una instrucción de escritura, lectura o de control.

El microprocesador acepta esta señal de confirmación y toma conocimiento de que ha concluido una fase de almacenamiento o de E/S con lo que podrá reanudar su programa principal. Esta conexión es especialmente importante, pues la interfaz de video puede realizar numerosas funciones independientemente del microprocesador. Con ello logramos un gran rendimiento de trabajo (velocidad de cálculo y de ejecución de instrucciones) en nuestro sistema.

El terminal 18 es una entrada de control para la interfaz de video. Si esta entrada OPREQ tiene nivel alto, todas las señales del 2650 serán válidas. Como muestra la figura 1.15, la entrada OPREQ está directamente unida a la salida OPREQ del 2650. Este procedimiento de la señal OPREQ es de gran importancia para garantizar un modo de trabajo seguro a nuestro sistema, ya que la interfaz de video admite datos y direcciones tan sólo si éstos se encuentran disponibles de una forma estable en los buses correspondientes. No se producen así errores de transmisión y nuestra imagen de pantalla queda estable.

El terminal 19 es la salida $\overline{CE2}$ de la interfaz de video. Si examinamos el esquema de la figura 2.15 veremos la conexión entre la salida CE2 y la entrada C del componente CMOS 4053. En esa conexión están también las dos entradas «strobe» del componente TT1 74156. Esta salida $\overline{CE2}$ tiene nivel bajo cuando el bus de direcciones del microprocesador se encuentra entre las direcciones E80 y EFF. Tendremos en cuenta, también, esta gama de direcciones cuando programemos nuestra tarjeta de memoria de instrucciones.

Los terminales 20 y 21 de la interfaz de video son las entradas a los dos convertidores analógicos/digitales internos. Con la dirección FCC direccionalmos PORT1 y con la FCD direccionalmos PORT2. Ambos convertidores trabajan con una frecuencia de 3,45 MHz. Como ambos convertidores trabajan según el principio de retardo de fase, obtendremos en el bus de datos una palabra de 8 posiciones inme-

diatamente después de su direccionamiento. Si no fuera ese el caso, activaríamos uno de los dos convertidores mediante una instrucción aparte y después de unos 1,5 μ s, tendríamos disponible una palabra de datos. Esos 1,5 μ s sólo pueden alcanzarse con sistemas convertidores muy rápidos, pero también muy caros. Gracias a los convertidores controlados en desfase no se originan grandes problemas y con ello se reduce al mínimo nuestro trabajo de programación.

El terminal 22 es la salida de audio de la interface de video. Esta salida está conectada a la sección de audio de nuestro sistema y genera una secuencia en serie de bits en forma rectangular. Para la programación de las secuencias de sonido utilizamos la dirección FC7.

Los terminales 23 al 30 son las entradas de datos para la interface de video programable. Esas entradas están conectadas en paralelo con el bus de datos de nuestro sistema. Tenemos un bus de datos de 8 bits.

El terminal 31 es la entrada INTACK. Examinando el esquema de la figura 2.15 vemos que está conectada a la salida INTACK del microprocesador 2650. Mediante esta señal de retorno, el componente 2650 notifica a la interface de video que se ha producido una secuencia de interrupción de programa. En el lenguaje especializado se habla de un «interrupt request» (petición de interrupción).

El terminal 32 es la salida OBJ/SCR para el sumador de video. En el esquema de la figura 2.15 empleamos la designación abreviada O/S. Cuando la interface de video dé salida a los datos de los objetos o al valor de los marcadores, la salida O/S tendrá nivel bajo. Tenemos que situar el valor de los marcadores en el correspondiente campo. Podemos utilizar también esta salida para el cuarto color, pero con ello hacemos la programación algo más complicada. Además habríamos de modificar el control del sumador de video. Utilizaremos esta salida para fines de

control y controlaremos la puerta de equivalencia de la salida C en el esquema de la figura 2.1.

Los terminales 33 a 35 nos generan los colores en el sumador de video. La mezcla de colores alimenta automáticamente al sumador de video como ya hemos visto en el apartado 2.1.

El terminal 36 es otra entrada de control para la interface de video. Esta entrada HRST recibe a la salida HRST del generador PAL 2621 los impulsos que servirán para la sincronización horizontal. Cuando la línea de televisión comienza, tenemos un impulso de reposición horizontal en esa conexión. Después de 226 impulsos PCK para la resolución de líneas llega este impulso HRST para la línea siguiente.

El terminal 37 es la entrada VRST para el componente 2636. Esta entrada está unida a la entrada SENSE del microprocesador y con la salida VRST del generador PAL. De esta forma, desde la línea 1 a la 44, las dos entradas reciben un nivel alto. Así podremos determinar, o calcular, desde arriba una cantidad de líneas de imagen mediante el microprocesador o la interface de video. Es importante el hecho de que el contador vertical se repone, o sea, se borra con el flanco positivo VRST. Esto se produce automáticamente también por el reloj interno y la lógica de control.

El terminal 38 es la entrada R/W de la interface de video. Con ello el microprocesador determina si después de un direccionamiento ha de escribirse o leerse datos en la interface. Con nivel bajo tenemos proceso de lectura y con nivel alto, de escritura. Sin embargo, esto depende del tipo de instrucciones en nuestra memoria de programa. El microprocesador determina mediante su lógica interna si se trata de instrucciones de lectura o de escritura.

El terminal 39 es la salida INTREQ de la interface de video. En el esquema de la figura 1.15, esta salida está unida a la entrada INTREQ del microprocesador. De esta forma, el microprocesador recibe una instrucción, o una petición, de la interface de video

para realizar una secuencia de interrupción de programa. En estado de trabajo, esto es en el programa principal, la salida INTREQ tiene un nivel alto. Pero si la salida pasa a nivel bajo, se inicia una secuencia de interrupción de programa que casi siempre lleva consigo un salto a un subprograma (subrutina).

El terminal 40 es la conexión a masa de la interface de video programable.

En la placa del microprocesador se encuentra, además del 2650 y la interface de video programable 2636, el generador PAL. Este generador representa el elemento de unión entre la placa del microprocesador y la del sumador de video. Por esta razón describiremos brevemente la función del componente 2621.

El terminal 1 es la salida C-SYNCH del generador PAL. Con esta salida, a través de dos puertas NO, controlamos la salida de video. Con ello conseguimos la sincronización correcta de la señal de salida de video de acuerdo con la norma PAL.

El terminal 2 es la salida VRST. Con ella la interface de video obtiene del generador PAL un impulso de sincronización vertical. Simultáneamente, la salida SENSE del microprocesador está unida a la salida VRST. Logramos, así, un modo de trabajo sincronizado.

El terminal 3 nos proporciona la salida C-BLINK para el sumador de video. La salida se obtiene a través de una puerta NO y tres puertas no inversoras del sumador de video y comunica la salida de video.

El terminal 4 sirve sólo para fines de comprobación.

El terminal 5 es la salida CBF del generador PAL. Genera la señal de «burst» de color para el sumador de video.

El terminal 6 genera el impulso de sincronización horizontal para la interface de video. La salida HRST sólo está unida al componente 2636 y efectúa la reposición del contador horizontal interno a 0.

El terminal 7 es la conexión a masa del componente 2621.

Los terminales 8 y 9 no se utilizan en nuestro caso de aplicación.

El terminal 10 suministra para el microprocesador la frecuencia de sincronización. La frecuencia de la salida PCK se divide en una relación 1:4 y obtenemos 0,886 MHz. Con esta frecuencia, el microprocesador puede trabajar tan rápido que, para cada barriado de imagen, podemos obtener nuevas coordenadas para la imagen de pantalla.

El terminal 11 es la salida PCK con una frecuencia de 3,54 MHz. Con esta señal controlamos la interface de video programable. El generador PAL, en este caso, sirve sólo como «buffer» (separador) de señal.

El terminal 12 es la entrada de sincronización del generador PAL. Aquí disponemos de los 3,54 MHz del sumador de video y, según vimos en el apartado 1.3, se transforman en el circuito integrado para realizar numerosas funciones.

El terminal 13 sirve para la conmutación entre líneas de televisión pares e impares.

El terminal 14 es la conexión a la tensión de alimentación.

La figura 2.20 nos muestra el diseño de la placa para el microprocesador y la 2.21 el esquema de disposición de componentes. La figura 2.22 reproduce el aspecto de esta placa del lado de los componentes.

La figura 2.23 muestra las conexiones de la memoria de instrucciones de 2 KB. Aquí podremos almacenar 2048 direcciones de instrucciones de 8 bits cada una (= 1 Bit) y obtener así una capacidad total de almacenamiento de 16 KB.

Las dos EPROM del tipo 2708 están conectadas en paralelo con los buses de datos y de direcciones. Con la dirección ADR10 se produce la conmutación entre la EPROMI y la EPROMII. Tenemos la tabla de funciones siguientes:

ADR1 = L: EPROM I gama de direcciones de 000 a 3FF.

ADR10 = H: EPROM II gama de direcciones de 400 a 7FF.

La conmutación entre ambas EPROM se realiza a través de una puerta NO, según se muestra en el esquema de la figura 2.23.

Además, la conmutación de ambas EPROM se realiza a través de las direcciones ADR11, ADR12 y mediante la salida «Chip enable» de la interface de video programable. Las EPROM podrán trabajar cuando en estas tres entradas haya las señales siguientes:

- A11: Nivel bajo
- A12: Nivel alto
- CE1: Nivel bajo

La dirección A11 es negada mediante una puerta NO e igualmente la entrada de control CE1. Con la dirección A12 estas tres líneas quedan unidas a una puerta Y de diodos. La salida de esta puerta está unida a otras dos puertas Y de diodos, con lo que se consigue una conmutación lógica con la dirección directa A10 o bien con la dirección negada A11. Seguimos, así, la conmutación correcta de ambas EPROM.

La figura 2.24 nos muestra el diseño de placa para la memoria de instrucciones y la figura 2.25 el esquema correspondiente de ubicación de componentes. La fotografía de la figura 2.26 nos reproduce esta placa vista desde el lado de los componentes.

La conexión de la figura 2.23 resulta más sencilla si empleamos el componente EPROM 2716. Pero una EPROM 2716 tiene un coste considerable y es difícil de conseguir por el aficionado.

La figura 2.27 nos muestra la conexión interna y el esquema de esta EPROM. El componente tiene una organización de 2048 direcciones y 8 salidas ($2048 \cdot 8 = 16384$ o sea 16 K). La EPROM 2716 precisa tan sólo una tensión de alimentación de $U_b = 5$ V. Con ello podemos omitir las dos tensiones $U_b = +12$ V y $U_b = -5$ V. La estructura de la tarjeta

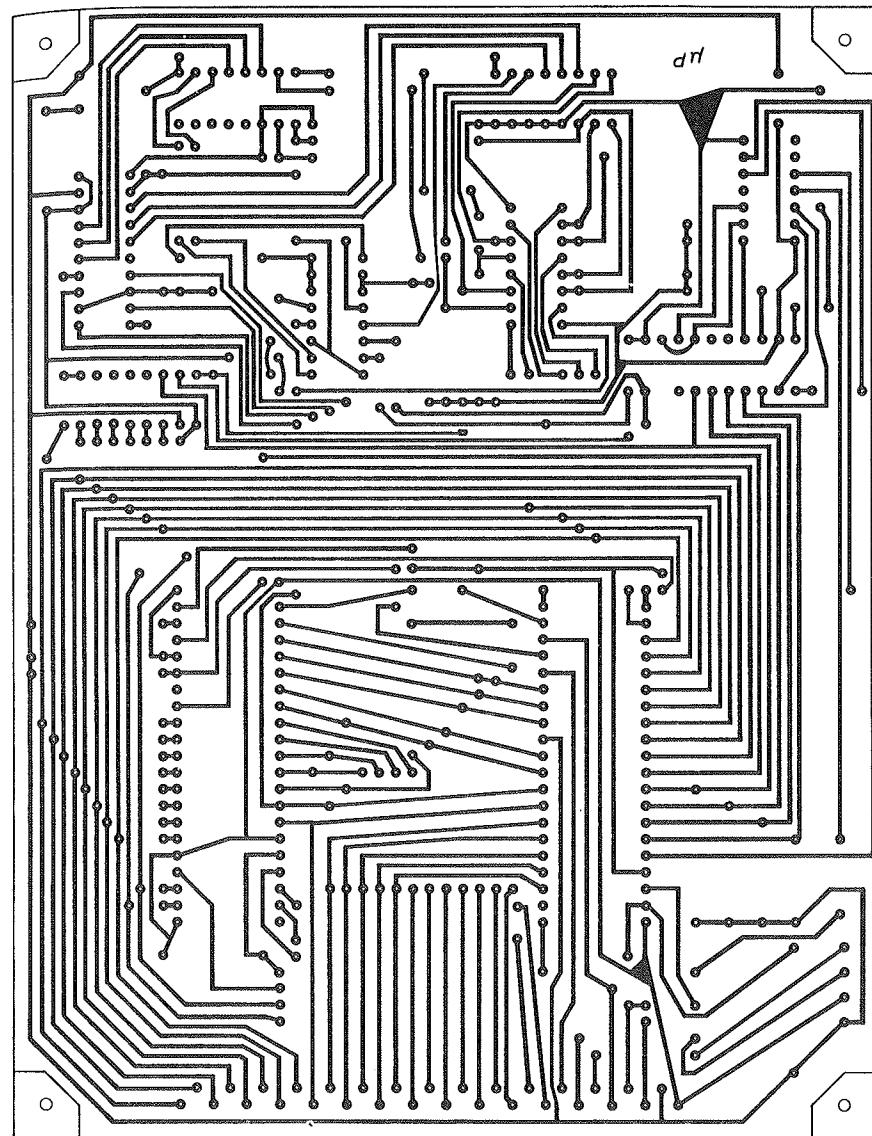


Fig. 2.20

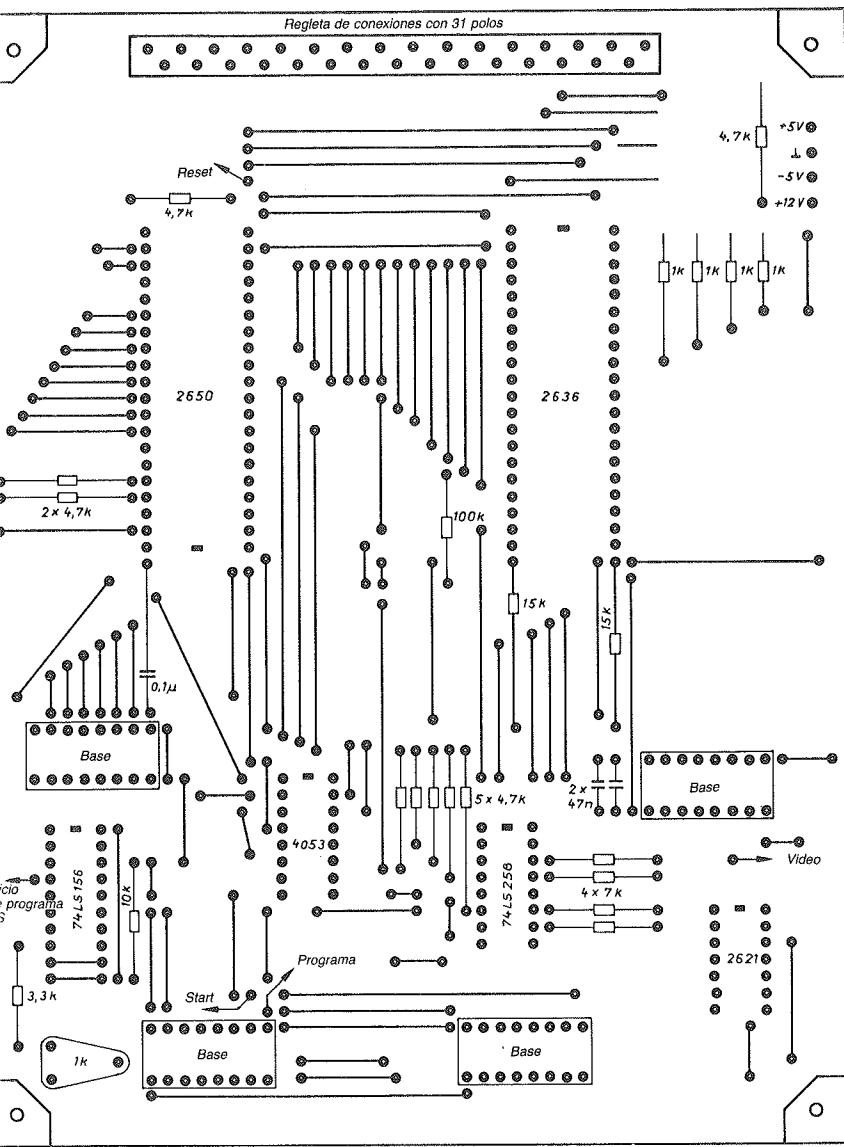


Fig. 2.21 Plano de disposición de componentes para el diseño de la placa

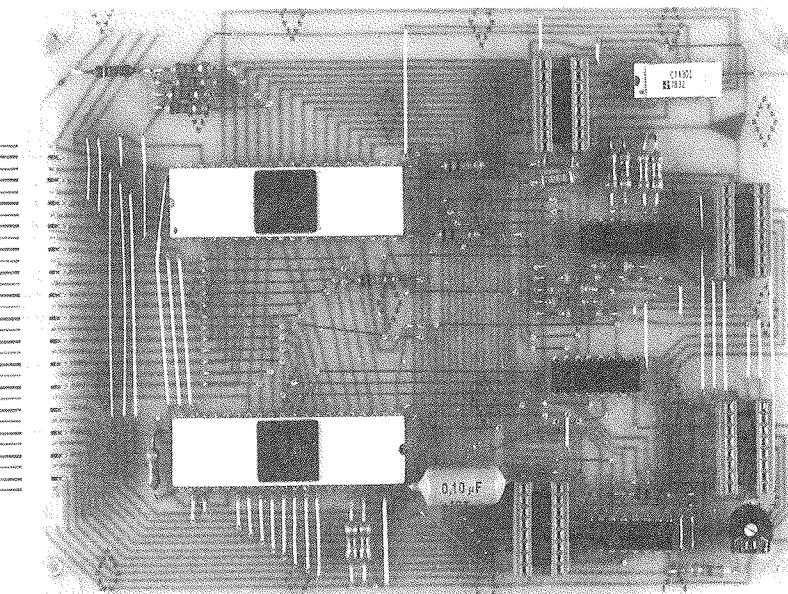


Fig. 2.22 Fotografía por el lado de los componentes del sistema microprocesador

de memoria de instrucciones de nuestro computador-TV se simplifica algo pero también se encarece.

Para la EPROM 2716 se aplica la tabla de funciones siguiente:

Modo de funcion.	PD/PGM	CS	U_{pp}	$+U_b$	Salidas
Lectura	L	L	+5V	+5V	Salida
Bloqueo	X	H	+5V	+5V	Alta resist.
Fuera de servicio	H	X	+5V	+5V	Alta resist.
Programación	L→H	H	+25V	+5V	Entrada
Prueba	L	L	+25V	+5V	Salida
Inhibición	L	H	+25V	+5V	Alta resist.

X ≈ nivel bajo o alto

Nota: Las tensiones indicadas directamente han de mantenerse necesariamente, pues, de no ser así, no podrá trabajar el componente correctamente. Los niveles bajo o alto podrán oscilar de acuerdo con los niveles TTL.

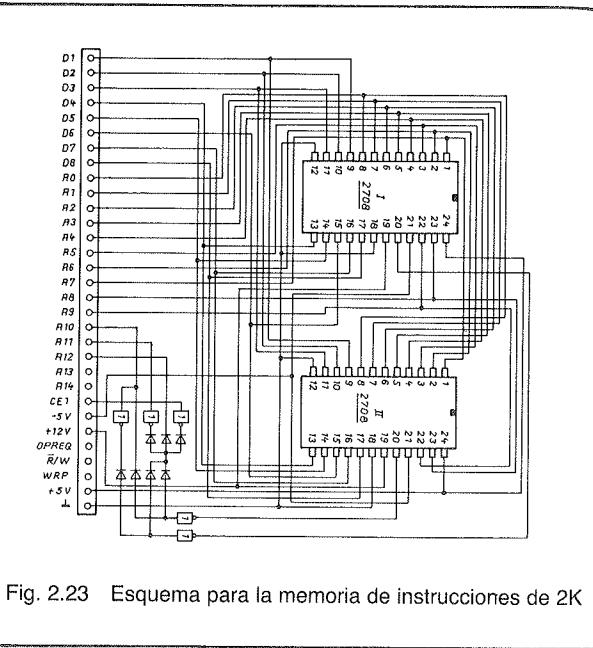


Fig. 2.23 Esquema para la memoria de instrucciones de 2K

En el modo de funcionamiento de «lectura», el contenido de la EPROM 2716 se conmuta directamente a las salidas. Puesto que tenemos 2048 direcciones, obtenemos 2048 palabras de salida diferentes si las condiciones de control se cumplen. No necesitamos en las salidas ninguna resistencia externa puesto que el circuito integrado 2716 tiene salidas triestado.

Si en la entrada CS disponemos un nivel alto, en vez de un nivel bajo, las salidas de la EPROM 2716 serán de alta impedancia y podremos conectar directamente al bus de datos las salidas del circuito integrado.

En el modo «fuera de servicio», aparece el caso de que en la entrada PD/PGM hay un nivel alto. Entonces, las EPROM también quedarán en estado de alta impedancia en sus salidas. Podremos utilizar esta entrada para indicaciones de control especiales

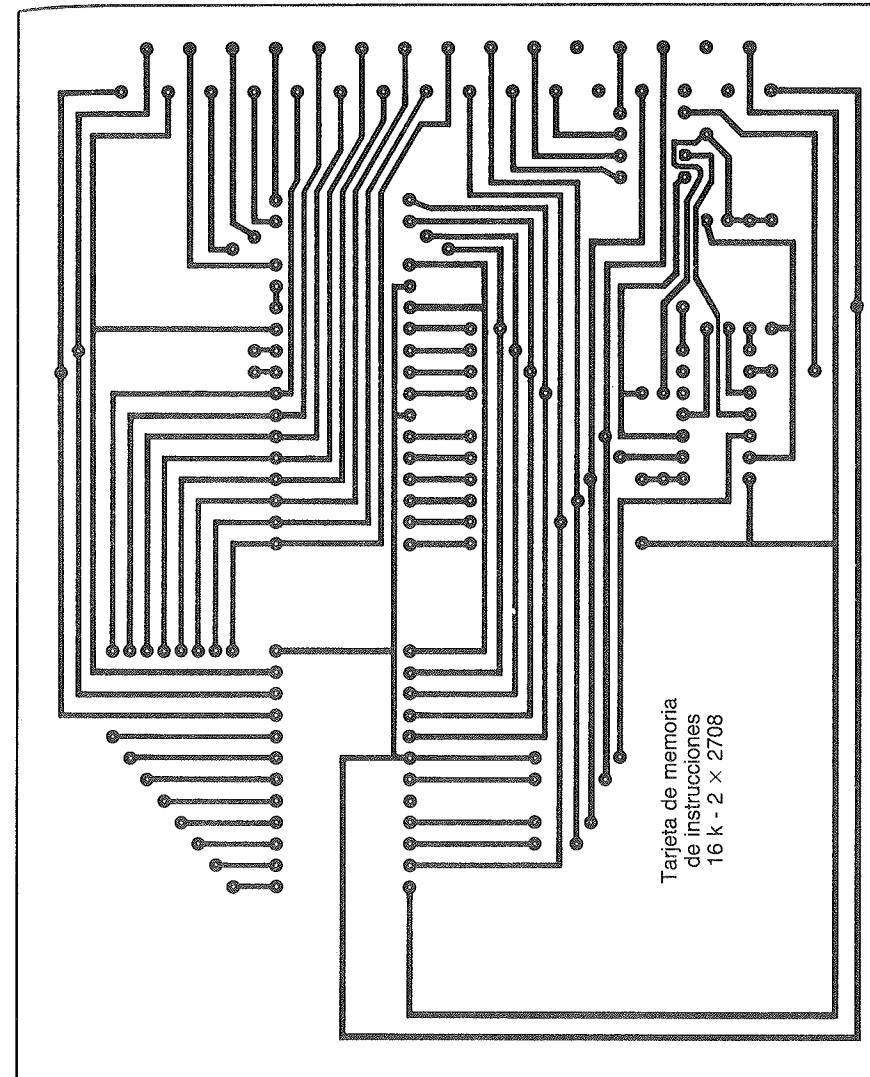


Fig. 2.24 Diseño de placa para la memoria de instrucciones de 2K

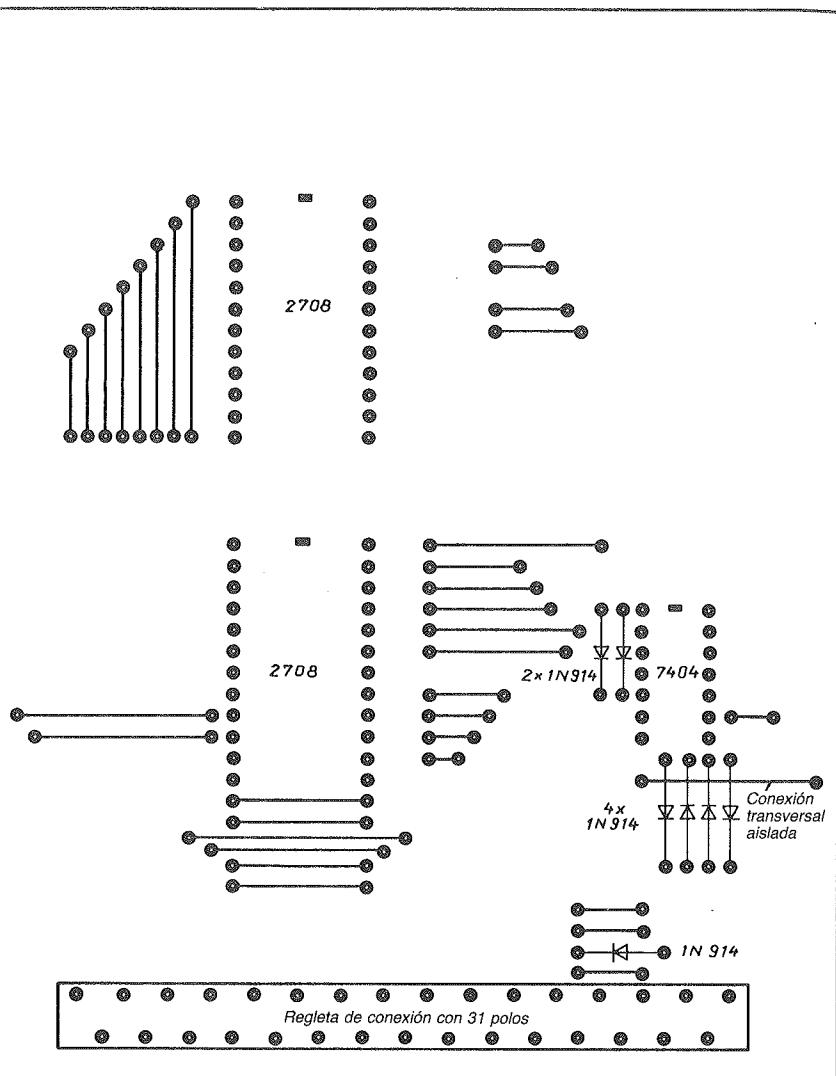


Fig. 2.25 Plan de disposición de componentes en la placa

Fig. 2.26 Fotografía desde el lado de los componentes de la memoria de instrucciones de 2K

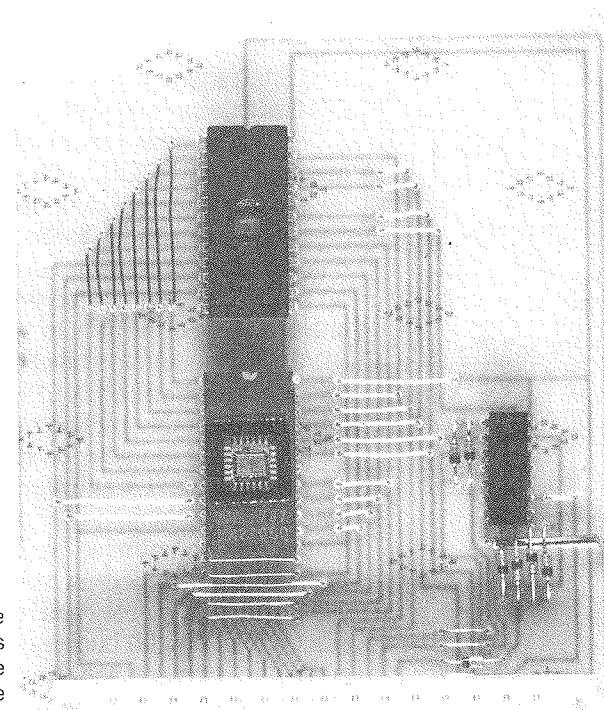
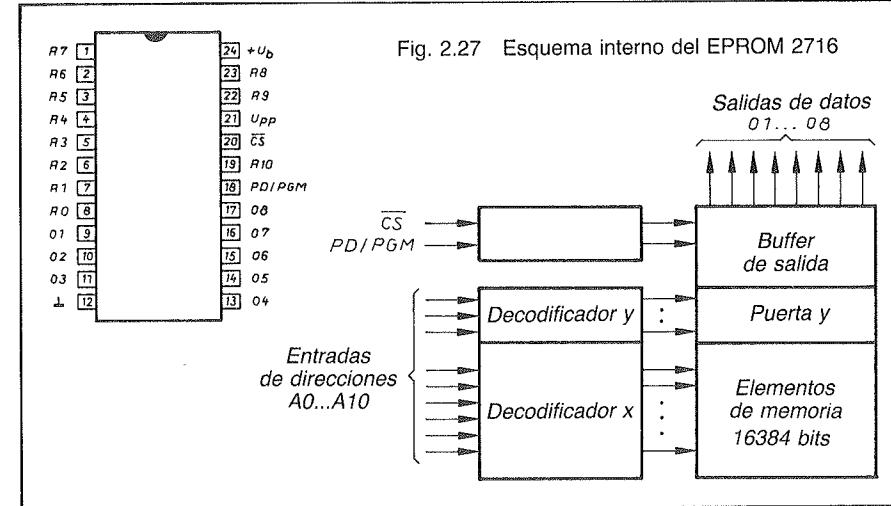


Fig. 2.27 Esquema interno del EPROM 2716



o bien la unimos a masa (nivel bajo), como así ocurre en el esquema de la figura 2.28. En esta conexión ahorrariamos tan sólo un diodo. El componente 2716 puede programarse con más sencillez.

Para ello podemos emplear básicamente el esquema de la EPROM 2708. La entrada U_{pp} se mantiene continuamente durante todo el proceso de programación a +25 V. Cuando las direcciones y los datos a programar deban quedar de manera estable en la EPROM, deberá comutarse primeramente la entrada \overline{CS} a nivel alto y después, la entrada PD/PGM cambiará de nivel bajo a alto. Este salto de nivel desencadena en la EPROM 2716 el proceso de programación de una línea completa, registrándose en las salidas tan sólo los niveles bajos que haya.

Advertencia: La amplitud del impulso de programación PD/PGM ha de ser de $t_{pp} = 40$ ms. Después tendrá que comutarse de nuevo a nivel bajo. Sólo después de esto podrán modificarse las direcciones y la palabra de datos. Para la EPROM 2716 se necesita un total de 100 bucles de programa, durante los cuales no siempre habrá que programar una dirección. Han de emplearse las especificaciones de programación de la EPROM 2708 cuando se vaya a realizar bucles de programas.

Los modos de funcionamiento «Prueba» e «Inhibición» sólo son importantes en conexión con dispositivos potentes de programación industrial. En el modo «Prueba» se dispone de la totalidad de la tensión de programación y la entrada PD/PGM o la \overline{CS} están a nivel bajo. De esa forma, podremos supervisar y controlar el proceso de almacenamiento propiamente dicho durante la programación, esto es, si durante el mismo los elementos se han modificado convenientemente.

Si la entrada \overline{CS} pasa a nivel alto, las salidas de la EPROM pasan a ser de alta impedancia.

La figura 2.28 muestra las conexiones para la placa. Las líneas de datos desde D1 a D8 se disponen conforme a las salidas de la EPROM 2716. Las líneas de direcciones de A0 a A10 están directamen-

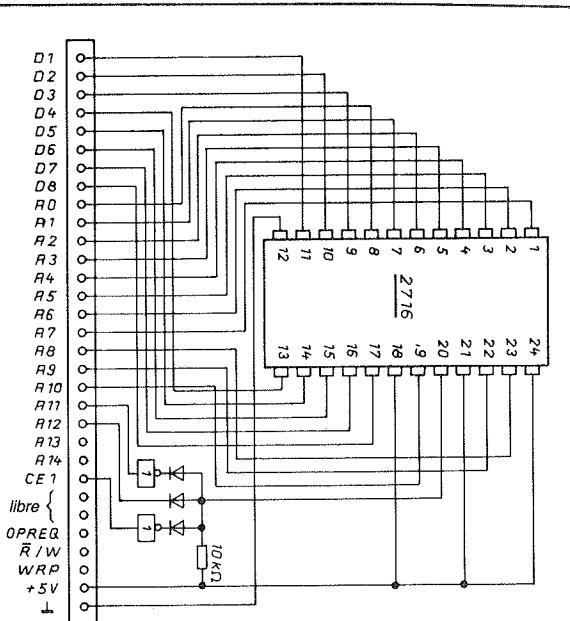


Fig. 2.28 Conexión de la memoria de instrucciones de 2K

te unidas a las entradas de direcciones. Para las direcciones A11 y A12, conectadas con la entrada \overline{CE} , tenemos una conexión de puerta AND de diodos precedida por una puerta NO. Para esta última empleamos el componente TTL 7404. Con las direcciones A0 a A10 abarcamos un espectro de direcciones desde 000 a 7FF, es decir, toda la capacidad de 2K de la EPROM 2716. El manejo de la EPROM tiene lugar mediante las dos direcciones A11 y A12 y mediante la salida \overline{CE} (Chip-enable) de la interfaz de video programable. La EPROM del esquema de la figura 2.28 sólo puede trabajar cuando en las tres entradas existan los siguientes estados de nivel:

- A11: nivel bajo
- A12: nivel alto
- $\overline{CE1}$: nivel bajo

Tarjeta de memoria
de instrucciones 16 k-2716

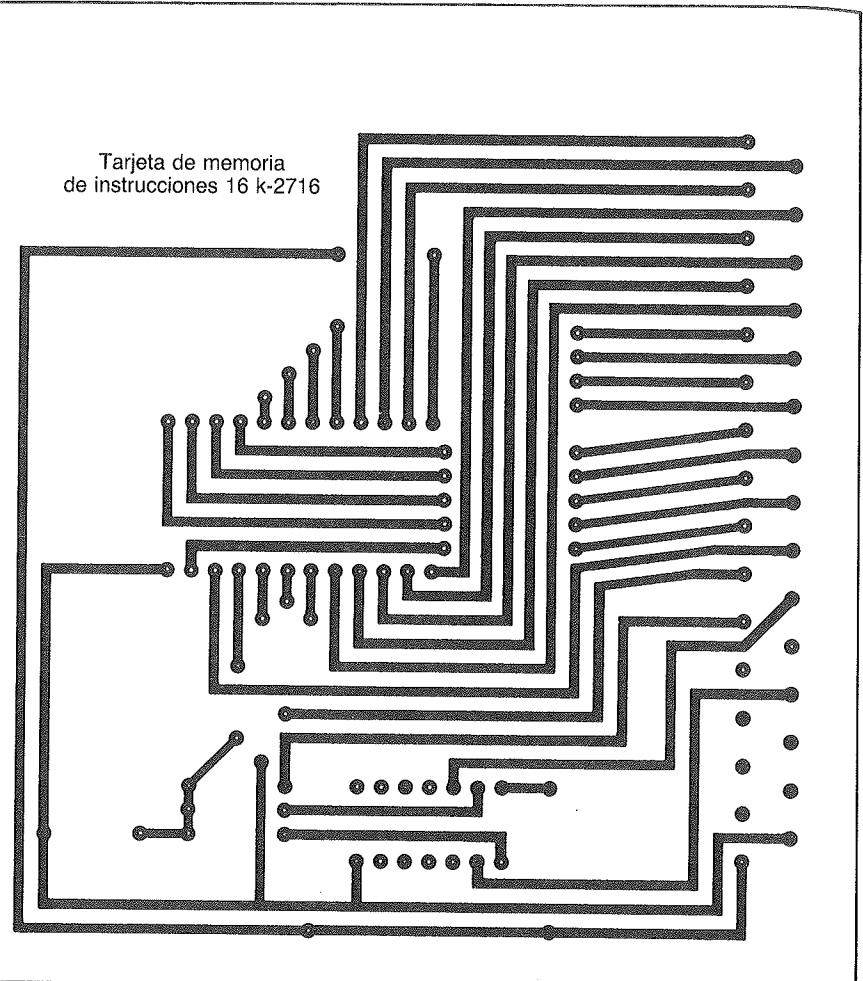


Fig. 2.29 Diseño de placa

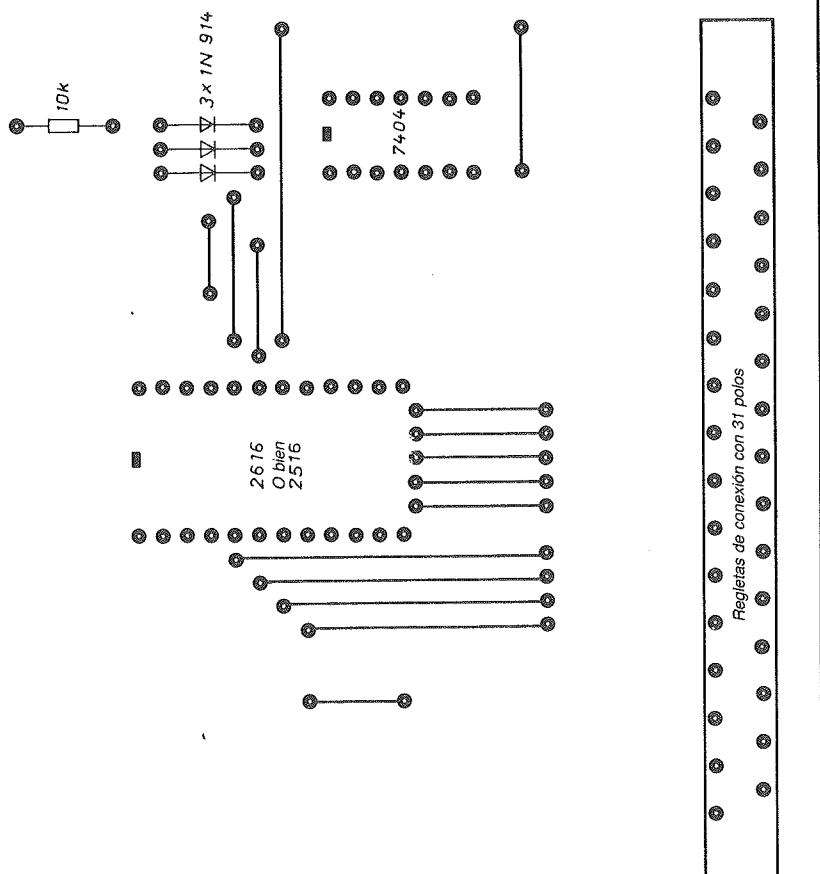


Fig. 2.30 Situación de los componentes

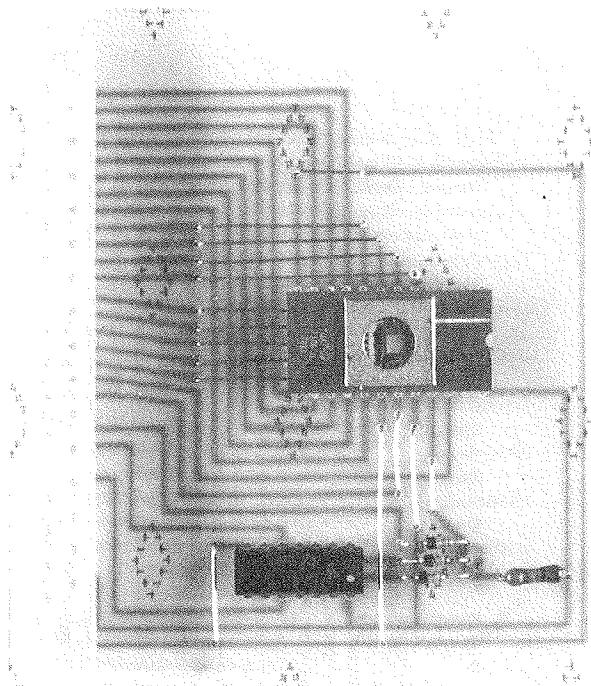


Fig. 2.31 Fotografía desde el lado de los componentes de la memoria de instrucciones de 2K

Los dos niveles bajos se invierten mediante dos puertas NO.

La figura 2.29 muestra un diseño de placa para memoria de instrucciones de 2K con la EPROM 2716. La figura 2.30 es el esquema de disposición de elementos para dicha memoria y la figura 2.31 muestra la fotografía de la placa vista desde el lado de los componentes.

Si ampliamos la tarjeta de instrucciones llegamos al esquema de la figura 2.32. Esta conexión contiene el programa para un pequeño computador de ajedrez. La figura 2.33 reproduce la imagen de pantalla para esa tarjeta de instrucciones.

El esquema de la figura 2.32 se une a las restantes conexiones del computador TV mediante unas

regletas de conexión. En esa placa hay seis EPROM del tipo 2708 y cuatro RAM del tipo 2114. Con ellos podremos almacenar 6144 instrucciones y variables en las EPROM y 2048 datos en las RAM. Esto no significa para un ajedrez una gran capacidad, por lo cual sólo podremos realizar juegos sencillos. En caso contrario, tendríamos que ampliar la capacidad en gran medida. A este juego pertenecen también un generador de números aleatorios y un procesador aritmético para calcular y evaluar las diferentes combinaciones del juego. El precio para esta placa de ajedrez supone unas 15.000 ptas. aproximadamente.

A través de la regleta se conecta el bus de datos a las seis EPROM y a las cuatro RAM. Respecto a estas últimas hay que observar que se encuentran conectadas dos a dos en paralelo, ya que las RAM 2114 tienen una organización de 1024×4 . Al conectarlas en paralelo se obtiene 1024×8 . Normalmente basta la potencia del microprocesador para activar correctamente los componentes de almacenamiento, es decir, para garantizar el nivel bajo correcto. Pero si se amplía la memoria han de conectarse necesariamente componentes TTL como activadores.

También se conecta el bus de direcciones a los elementos de memoria a través de la regleta. Las direcciones A0 a A10 se unen directamente a los diferentes elementos de memoria. Las direcciones A10, A11 y A12 se unen, sin embargo, a las entradas A, B y C del componente 7442. Este último se controla a través de la salida $\overline{CE}1$ de la interface de video programable. Las salidas del componente TTL 7442 están unidas directamente a los distintos elementos de memoria de modo que se obtiene un control de dicho circuito integrado. Para las conexiones de las memorias de instrucciones de la figura 2.32 tenemos la tabla de la página siguiente.

A partir de estas tablas podemos conocer el modo de acción operativa del decodificador BCD a decimal 7442, sobre los distintos elementos de memo-

Entradas del circuito integrado TTL 7442						
Nr.	(D)	(C)	(B)	(A)	A11	A10
0	L	L	L	L		
1	L	L	L	H	L	H
2	L	L	H	H	L	H
3	L	L	L	L	H	H
4	L	L	L	H	H	H
5	L	L	L	H	H	H
6	L	L	L	H	H	H
7	L	H	H	H	H	H
8	H	H	H	H	H	H
9	H	H	H	H	H	H
10	H	H	H	H	H	H
11	H	H	H	H	H	H
12	H	H	H	H	H	H
13	H	H	H	H	H	H
14	H	H	H	H	H	H
15	H	H	H	H	H	H

Elementos de memoria
EPROM 1
EPROM 2
EPROM 3
EPROM 4
EPROM 5
EPROM 6
RAM 7/1,7/2
RAM 8/1,7/2

Bloqueado
Bloqueado
Bloqueado
Bloqueado
Bloqueado
Bloqueado
Bloqueado

Entradas del circuito integrado TTL 7442

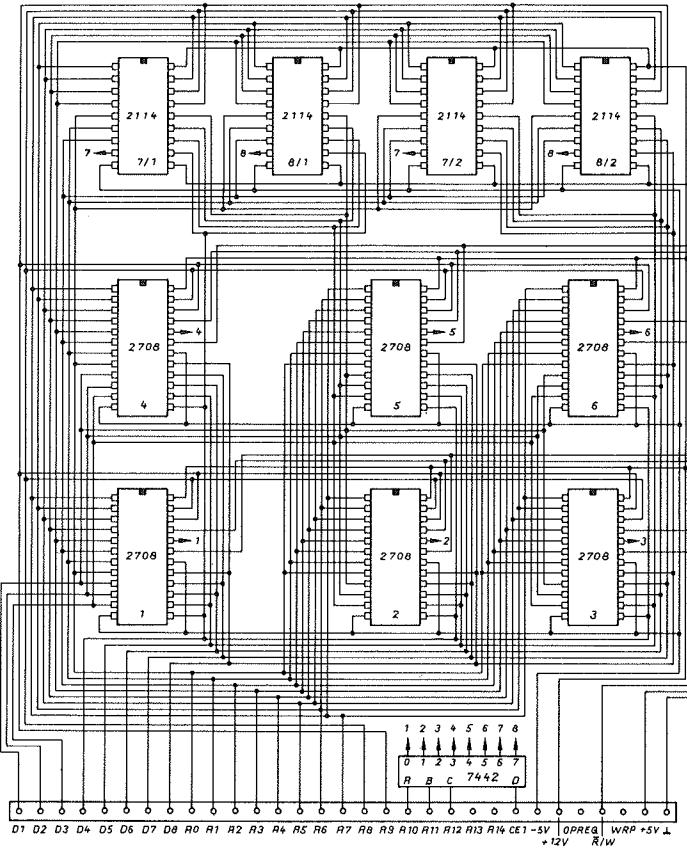


Fig. 2.32 Conexión de la memoria de instrucciones para un computador de ajedrez

ria. En estado «bloqueado», conectamos todas las salidas al nivel Z.

Las EPROMS obtienen su señal de disponibilidad directamente en las entradas CE mediante el decodificador 7442. Con un nivel bajo y para direcciones de A0 a A9, las direcciones y variables de las EPROMS pasan al bus de datos en el intervalo de 350 ns. No necesitamos para ello la línea de escritu-

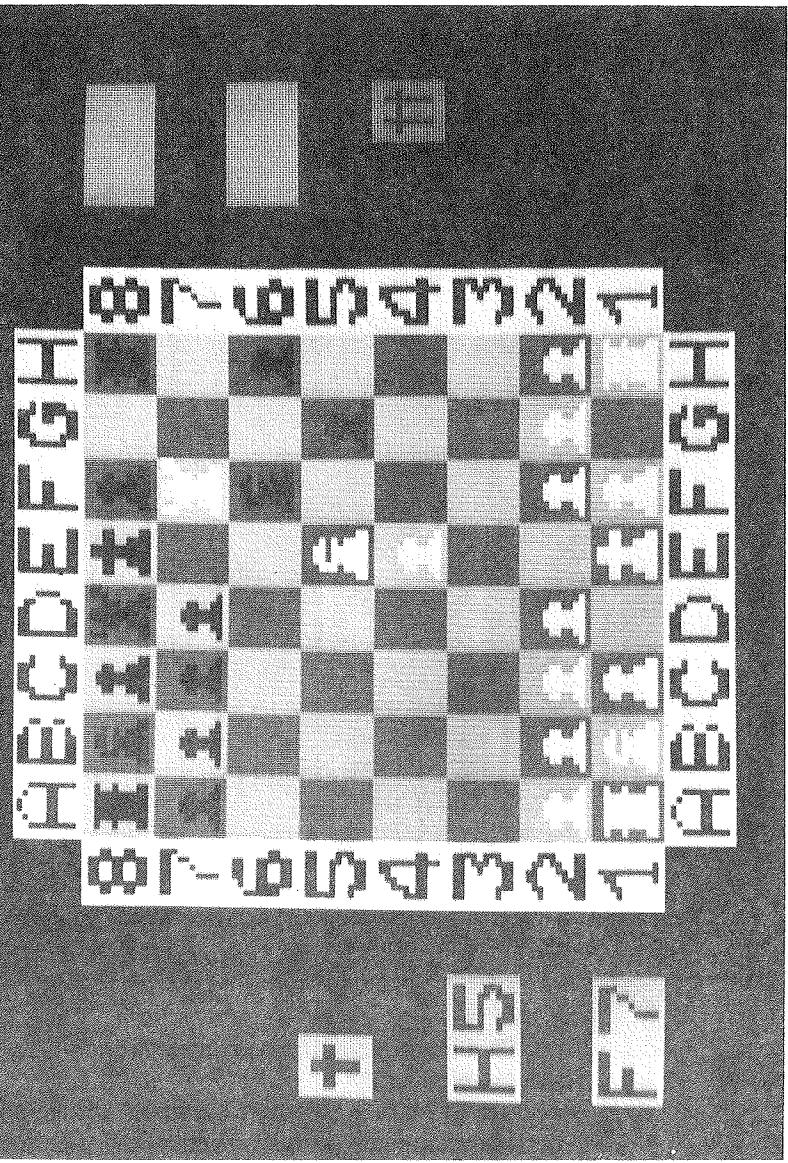
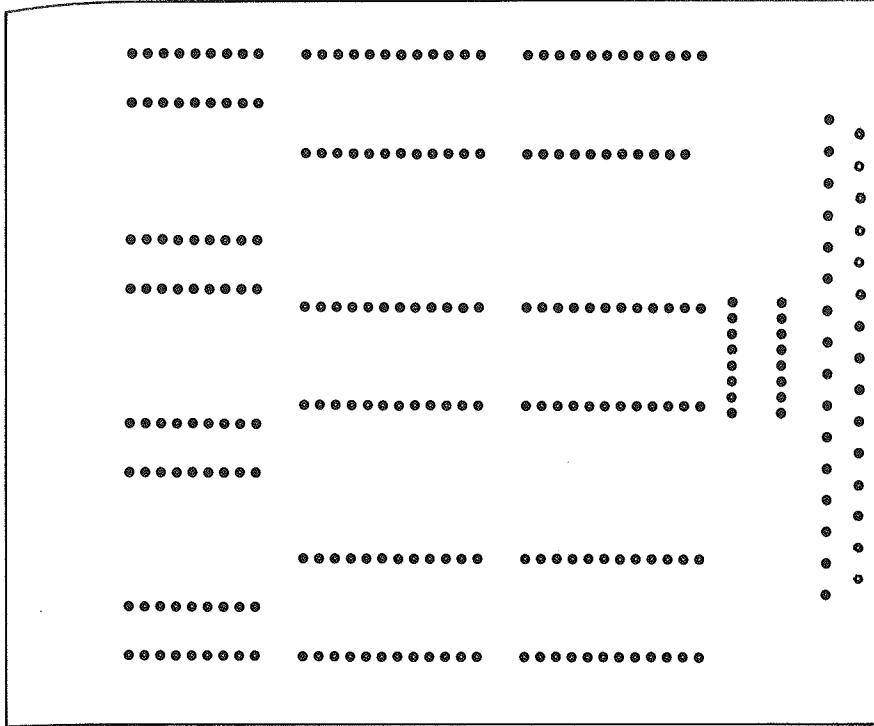


Fig. 2.33 Fotografía de la imagen de un juego TV de ajedrez

Lista de componentes para el microprocesador	
1	Microprocesador 2650A (sólo versión rápida)
1	Interface de video programable 26336
1	Generador PAL 2621
1	Multiplexor 4053
1	Decodificador binario/Demultiplexor 74LS156
1	Selector de datos/multiplexor 74LS258
1	Condensador 0,1 μ F
2	Regulador 47 nF
1	Resistencias 1 k Ω
4	Resistencia 1 k Ω
1	Resistencia 3,3 k Ω
13	Resistencias 4,7 k Ω
1	Resistencia 10 k Ω
2	Resistencia 15 k Ω
1	Resistencia 100 k Ω
2	EPROM 2708 o bien
1	EPROM 2716

Fig. 2.34 Prototipo de la placa, el cableado se realiza en térmica «Wire-wrap»



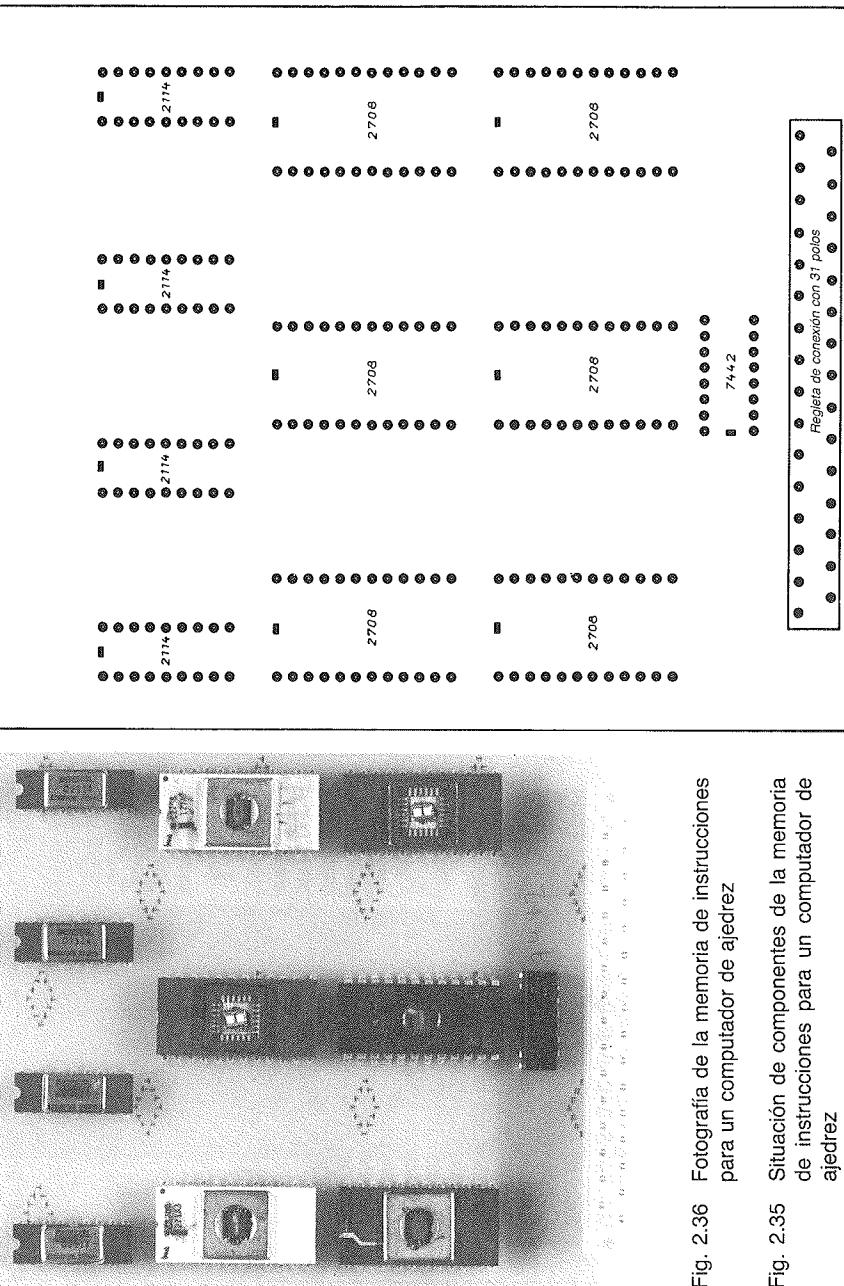


Fig. 2.36 Fotografía de la memoria de instrucciones para un computador de ajedrez

Fig. 2.35 Situación de componentes de la memoria de instrucciones para un computador de ajedrez

ra-lectura del microprocesador, no obstante, ello ha de tenerse en cuenta en el direccionamiento de nuestra tarjeta de memoria de instrucciones.

El proceso de lectura-escritura en las RAM se desarrolla de otra manera. Las RAM pueden trabajar sólo si la entrada \overline{CS} está a nivel bajo y si con la entrada WE está definido un trabajo de escritura o lectura. La entrada CS de las RAM recibe un nivel bajo del decodificador 7442 cuando el microprocesador ha dado salida a una dirección correcta. Cuando la entrada WE hay un nivel bajo, el microprocesador podrá establecer el trabajo de escritura mediante un nivel bajo o el de lectura mediante nivel alto. No se necesita un control especial para estas dos funciones, ya que el 2650 puede discernir en sus instrucciones si éstas son de escritura o de lectura. De acuerdo con ello, la línea WE o la R/W tiene un nivel bajo o alto.

La figura 2.34 muestra un diseño de placa para la memoria de instrucciones de un computador de ajedrez. En la figura 2.35 tenemos el esquema de disposición de componentes.

La figura 2.36 muestra una fotografía de esta placa para computador de ajedrez. El cableado no se hizo mediante placa de soldaduras, sino con el procedimiento «wired-wrap» (conexión enrollada). Para ello necesitamos emplear, sin embargo, una herramienta especial, pero reducimos el trabajo en gran medida.

2.3 COMPONENTES DE AUDIO

El esquema de la figura 2.37 muestra el diagrama para la sección de audio de nuestro computador-TV. En el apartado 2.4 se encuentra la placa para este esquema ya que en ella se encuentra también el modulador HF.

La sección de audio está unida a la placa del microprocesador mediante una conexión de 16 terminales. De las ocho líneas de datos se necesitan sólo 6 para la sección de audio; a saber, D2 a D7. Las 6 líneas están unidas al componente 74378.

La figura 2.38 muestra el esquema de conexiones y el circuito interno del componente TTL 74378, que se suministra sólo en tecnología «Low-Power-Schottky». El componente dispone de 6 registros internos de almacenamiento D con una conexión común de sincronización (terminal 9) y un bloqueo también común (terminal 1). Para el componente tenemos la tabla de funciones siguientes:

Entradas		Salidas	
No disponible	Sincr.	D	Q
H	X	X	Q_n
L		H	H
L		L	L
L	L	X	Q_n

X = Nivel bajo o alto

Q_n = Estado de almacenamiento

Si los datos procedentes del sistema de bus de datos están disponibles en el componente 74378, su aceptación por este último depende tan sólo de las entradas de sincronización y de «no disponibilidad». La entrada de sincronización obtiene del transistor T1 un nivel adecuado. Este transistor está controlado por un elemento diferenciador que obtiene su señal de la salida WRP del microprocesador 2650.

El transistor T1 trabaja en conexión de emisor común y trabaja, por tanto, como puerta NO.

La señal de salida WRP del 2650 es un impulso especial de escritura que se genera durante la fase de escritura del microprocesador. Esta señal se utiliza para excitar nuestra sección de audio. Ante un flanco negativo WRP (salto de nivel alto a bajo) el transistor T1 pasa a nivel bajo gracias al elemento diferenciador. El transistor T1 se bloquea y la salida se comutará a nivel alto la entrada de sincronización del circuito integrado 74378, con lo cual se aceptarán los datos. Los datos se almacenarán cuando la salida WRP pase de nuevo a nivel alto, puesto que, entonces, el transistor conduce de nue-

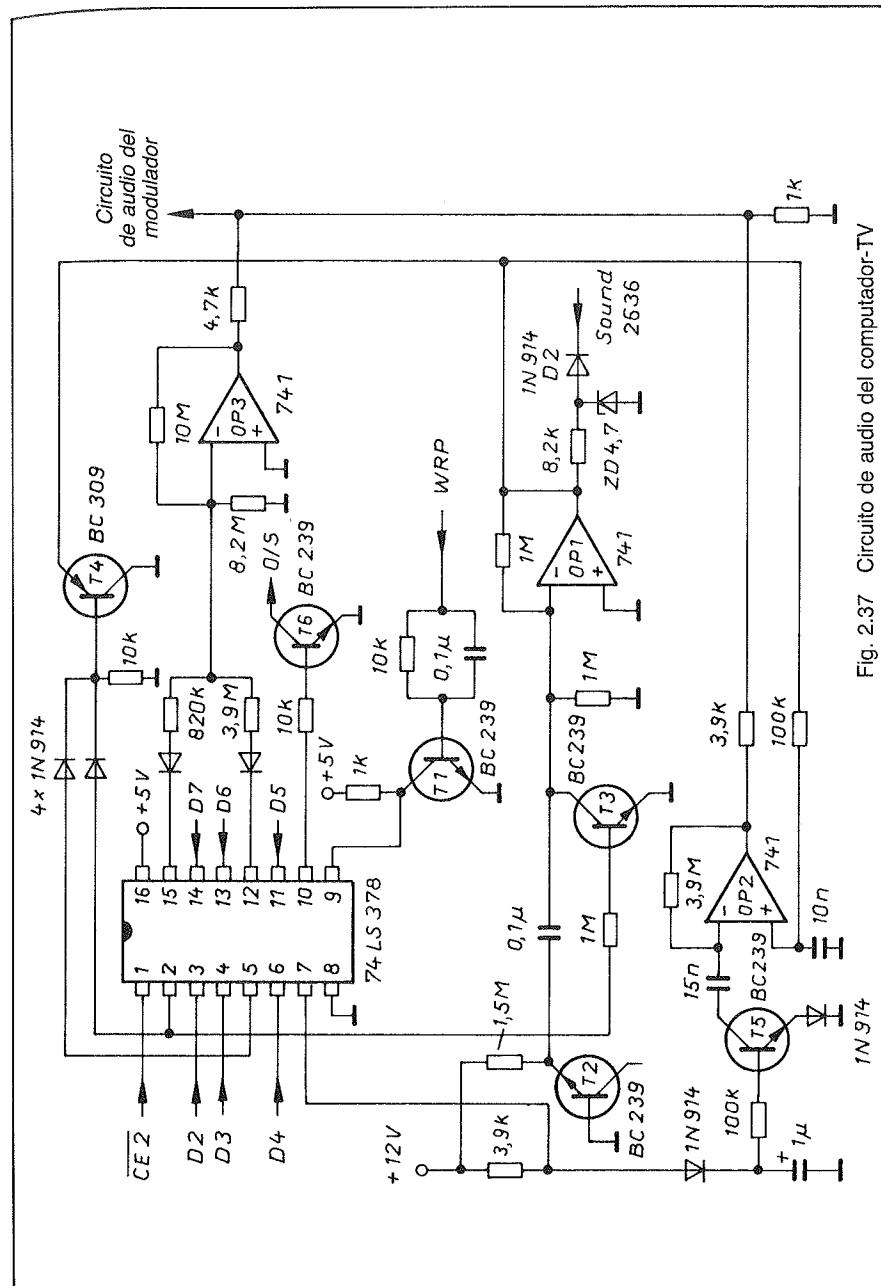


Fig. 2.37 Circuito de audio del computador TV

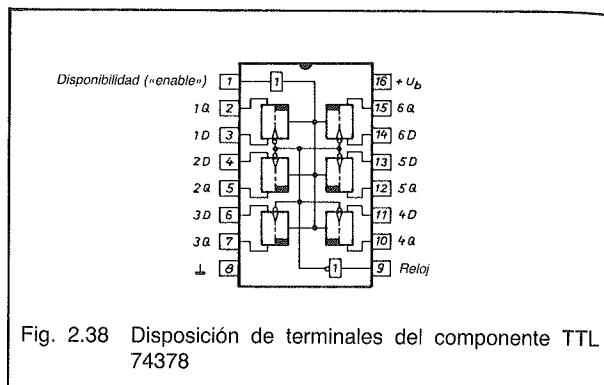


Fig. 2.38 Disposición de terminales del componente TTL 74378

vo y la entrada de sincronización queda a masa (nivel bajo). Conseguimos una aceleración en la conmutación mediante el condensador en la base.

El circuito integrado 74378 puede trabajar sólo cuando la entrada CE2 del esquema tenga un nivel bajo. Ese nivel lo recibe el componente de la interfaz de video programable. El terminal 19 (salida CE2) de la interfaz tiene nivel bajo cuando el bus de direcciones se encuentra en la zona de direcciones entre E80 y EFF. En caso contrario, tenemos un nivel alto.

La admisión de datos puede tener lugar sólo si nos mantenemos dentro de esa gama de direcciones y si tenemos una instrucción de escritura. Si no es así, el componente almacenará sus informaciones.

Con las partes restantes del esquema de la figura 2.37 y en conexión con la entrada «sound» generamos las distintas secuencias de sonidos. Mediante los seis datos podemos depositar en la memoria de datos hasta 2^5 posibilidades. Mediante el bit de datos D5 controlamos el transistor T2 que está unido a la interfaz de video programable y al sumador de video.

La frecuencia del impulso rectangular f_r para nuestra sección de audio la obtenemos del terminal 22 de la interfaz de video 2636. Esta frecuencia

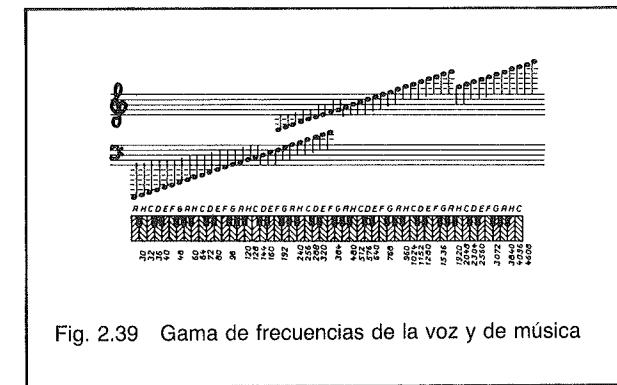


Fig. 2.39 Gama de frecuencias de la voz y de música

queda disponible en la entrada «sound» de la placa. La frecuencia se calcula mediante:

$$f_r = \frac{7874}{n+1}$$

El valor n depende de lo que hayamos almacenado en hexadecimal en la dirección 1FC7 de la interfaz de video. El valor hexadecimal allí almacenado puede variar entre 00 y FF. Con ello tenemos 256 posibilidades para secuencias de sonidos. Con valor 00 no obtenemos ningún sonido. Con el valor 01 obtenemos la frecuencia más alta $f_r = 3,937$ KHz y con FF obtenemos $F_r = 30$ Hz.

La figura 2.39 muestra la zona de frecuencias completa que corresponde a la música y a la voz. Partiendo de este espectro de frecuencias podemos programar los distintos sonidos. Las secuencias las programaremos en las EPROM de la tarjeta de memoria de instrucciones. Todo el problema puede resolverse también mediante el software.

Para nuestra sección de audio necesitamos tres amplificadores del tipo 741. Este amplificador es seguro frente a cortocircuitos y normalmente precisa de dos tensiones de alimentación de, por ejemplo, $\pm 12V$ y un potenciómetro para la compensación de la desviación. La figura 2.40 nos muestra el esque-

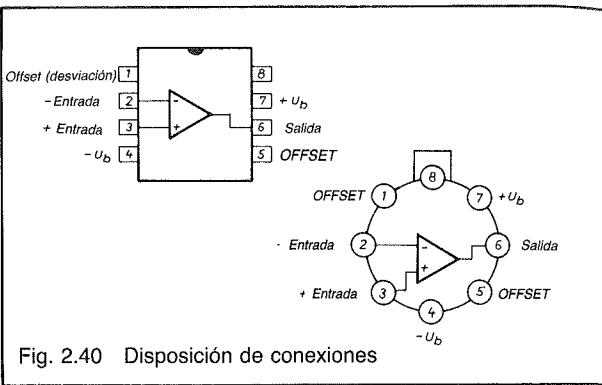


Fig. 2.40 Disposición de conexiones

ma de conexiones para el circuito integrado 741 o TBA 221 (denominación Siemens).

En lugar de emplear dos tensiones de alimentación, disponemos el amplificador 741 entre +12V y masa. Esa tensión es suficiente para nuestro caso de aplicación. Entre los terminales 1 y el 5 tenemos la compensación de la desviación («offset»). Pero como no existe tensión negativa ambas conexiones permanecerán abiertas. Sin embargo, se produce un pequeño defecto de «offset», pero evitamos que se produzcan errores de conmutación al incorporarlo en las conexiones analógicas. La conexión U_b de la tensión de alimentación se pone a masa. Con ello la tensión de salida del amplificador 741 varía entre 0V y +12V, aunque estos sólo son valores teóricos. El intervalo efectivo está entre $U_a = +1V \dots +11V$. El bit de datos D3 está en el terminal 3 del componente TTL 74378. La salida para ese bit de datos es el terminal 2. Este terminal está unido al transistor T3 a través de una alta impedancia de $1 M\Omega$. El transistor T3 invierte el nivel de la señal del bit de datos D2 y controla así la entrada del amplificador OP1. Simultáneamente, a través de un diodo de desacoplo, la señal del bit de datos D2 está también en el transistor T4. Este transistor PNP es conductor cuando en la base hay nivel bajo. Cuando este transistor está en conducción, la salida del amplificador OP1 está a

$U_a \approx 0,7 V$ y los impulsos de sonido entrantes quedan sin efecto.

El transistor T2 está conectado como generador de ruido. La base está unida a masa y el emisor está unido a la tensión de alimentación positiva de +12V a través de una resistencia de $1,5 M\Omega$. El colector del transistor no está conectado y está «al aire». Con un componente de tensión continua de $U_g = 6V$ obtenemos un ruido $U_{ss} = 0,1V$. El condensador conectado detrás, con una capacidad de $0,1 \mu F$ acopla la corriente continua, con lo que el ruido del transistor pasará al amplificador. Este amplifica el ruido si el T3 no ha conducido.

El bit de datos D3 está en el terminal 4 del 74378 y sale por el terminal 5. Este bit de datos está unido al transistor T4 a través de un diodo desacoplador. Si en el registro de memoria tenemos nivel-bajo, el transistor T4 conduce. Si el nivel almacenado es alto, el transistor se bloquea. Los dos diodos para el transistor T4 están conectados a masa como diodos en montaje de puertas OR, con una resistencia suplementaria. De esa forma, con el transistor T4 conectado detrás obtenemos, a pesar de todo, una conexión OR. No necesitamos el transistor T4 como función lógica sino para desacoplar la tensión de salida del amplificador OP1 y el nivel TTL del sistema microprocesador.

El bit de datos D4 está asociado al terminal 6 y en el terminal obtenemos la señal de salida almacenada. Si hay un nivel alto almacenado puede cargarse el condensador, con $C = 1 \mu F$, a través de los diodos. Cuando la tensión en la base del transistor T5 alcanza el valor $U_{BE} = 1,4 V$, el transistor conduce. La tensión $U_{BE} = 1,4 V$ se logra a partir de la U_{BE} del transistor T5 y la tensión de saturación U_d de los diodos. De esta forma conseguimos controlar el amplificador OP2. Si el bit de datos D4 tiene nivel bajo permanece la carga del condensador, ya que los diodos impiden la descarga. Aunque, sin embargo, se produce descarga a través del transistor T5.

El amplificador OP2 está conectado como comparador con características funcionales de integración y diferenciación. La salida de este amplificador está conectada al modulador del aparato de televisión a través de una resistencia de 3.9 k Ω .

El bit de datos D5 sirve para controlar el sumador de video a través del transistor T6, según acabamos de ver.

Los bits de datos D6 y D7, a través de dos diodos de desacoplo y dos resistencias, controlan el amplificador OP3. Mediante las dos resistencias de $820\text{ k}\Omega$ y $3,9\text{ M}\Omega$ conseguimos un sumador. Como, además, la resistencia de realimentación tiene un valor de $10\text{ M}\Omega$, se consigue una amplificación suplementaria. La salida del amplificador está unida a la del modulador a través de una resistencia de $4,7\text{ k}\Omega$.

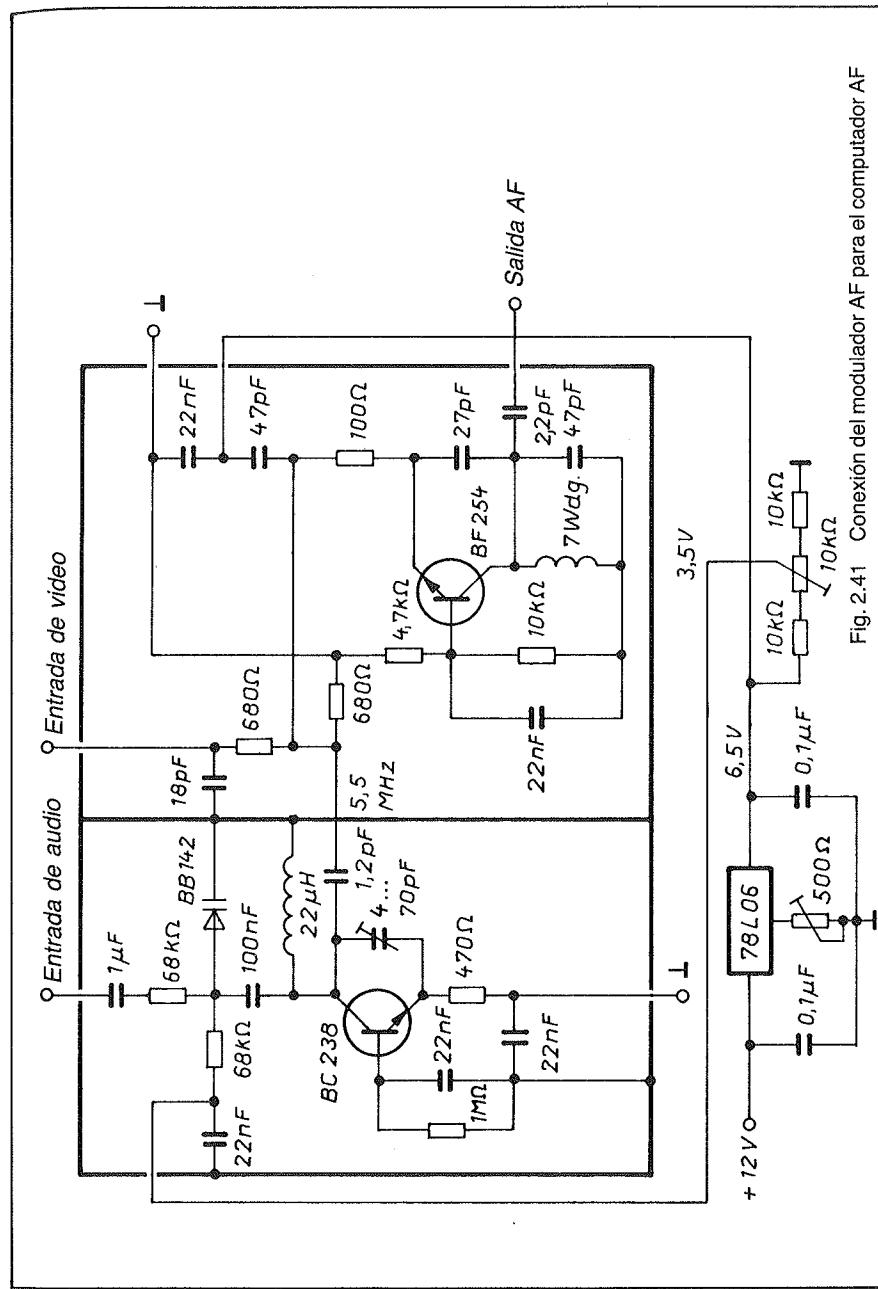
4 MODULADOR F

La figura 2.41 muestra el esquema del modulador AF para nuestro computador TV. El esquema está subdividido en dos bloques. A la izquierda tenemos el modulador para la portadora de sonido y a la derecha el modulador propiamente dicho. La tensión de alimentación para los dos bloques se genera mediante un regulador de tensión integrado.

Antes de ocuparnos en detalle de este esquema, hemos de prestar atención a la modulación de emisión. Como frecuencia portadora empleamos el canal 2,3 o 4 de TV. Tenemos para ellos las frecuencias portadoras siguientes:

Zona	Canal	Imagen	Sonido
VHF	2	48,25 MHz	53,75 MHz
	3	55,25 MHz	60,75 MHz
	4	62,25 MHz	67,75 MHz

No sintonizaremos los canales exactamente. Desp  s de conectar el computador TV se sintonizar   el aparato de televisi  n a la frecuencia portadora.



Efig. 2.41 Conexión del modulador AE para el comunicador AF

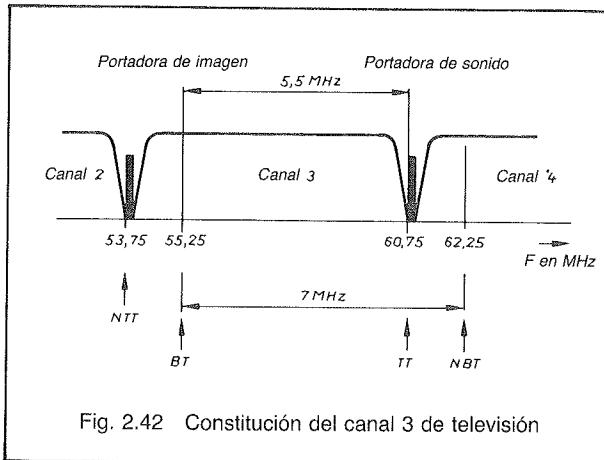


Fig. 2.42 Constitución del canal 3 de televisión

Esto no constituye problema alguno pues en la mayoría de los televisores la elección de programa es muy sencilla. La frecuencia portadora suele estar comprendida entre 55 MHz y 65 MHz.

La figura 2.42 nos muestra la constitución del canal 3 de televisión en la zona VHF-I (Very High frequencies - muy altas frecuencias). El canal 3 comienza en la frecuencia de 55,25 MHz y termina en 62,25 MHz. Con ello disponemos de un ancho de canal de 7 MHz.

Para la portadora de imagen tenemos un ancho de banda de 5,5 MHz. Esta frecuencia la encontramos también en el esquema de la figura 2.41. Se encuentra en la conexión entre el bloque izquierdo y el derecho. El canal de sonido es ± 50 kHz y está alrededor de los 60,75 MHz.

Para la modulación de portadora para la salida de AF utilizamos dos procedimientos. El sonido lo modulamos en una frecuencia de 5,5 MHz y con ello conseguimos una modulación de frecuencia. Mediante la entrada video modulamos la amplitud de la tensión AF de unos 60 MHz y conseguimos una modulación de amplitud.

De los dos tipos de modulación se obtiene la tensión de AF. La estructura de la portadora de imagen

se designa como BT y el ancho de banda de 7 MHz alcanza hasta la portadora NBT de la imagen contigua. La portadora de sonido TT tiene también 7 MHz hasta su portadora adyacente. Por estas circunstancias, no se plantean grandes problemas durante la transmisión de los datos desde el sistema microprocesador a un aparato de televisión.

El oscilador en la cámara izquierda del modulador de AF oscila con una frecuencia de unos 5,5 MHz. Podemos ajustar esa frecuencia con un condensador variable. En la entrada tenemos la tensión de la sección de audio. Mediante un condensador eliminamos la componente continua de modo que en el diodo capacitivo BB142 sólo tengamos la tensión alterna. La capacidad de este diodo depende de la tensión de bloqueo que se aplique. De este modo, puede regularse la frecuencia de resonancia del circuito oscilante mediante la tensión en la entrada de audio. Así pueden construirse osciladores de frecuencias regulables y que son adecuados para emisoras moduladas en frecuencia.

El transistor BC238 trabaja como oscilador. Pueden aparecer problemas con la inductancia, ya que el bobinado no es una cosa simple para el aficionado a la electrónica cuando aquélla tiene que equilibrarse. Ha de utilizarse una inductancia fija de $22 \mu\text{H}$, de las que se encuentran en los establecimientos especializados o en los distribuidores. La frecuencia de 5,5 MHz puede equilibrarse mediante el condensador variable. Si se dispone de un osciloscopio, esta tarea no plantea problemas. En caso contrario, exigirá más trabajo y, sobre todo, paciencia.

La entrada de video está en la segunda cámara del modulador de AF. En ella se modula la señal BAS del sumador de video en la señal portadora de sonido modulada en frecuencia según el procedimiento de modulación anódica. De esta manera, se obtiene en la salida la señal de AF.

El transistor BF254, junto con el condensador y la bobina, constituye un oscilador de unos 60 MHz de frecuencia de oscilación. Sobre esta frecuencia

se modulan los 5,5 MHz. Aquí la bobina supone también un pequeño problema. Esta debe tener 7 espiras de unos 4 mm de diámetro. Como hilo utilizaremos hilo de cobre de 0,1 mm de diámetro que arrollaremos en un destornillador. La bobina deberá tener una longitud de unos 10 mm.

El circuito de la figura 2.41 debe colocarse dentro de una caja de cobre o aluminio para que la tensión de AF no pueda influir sobre los demás circuitos. Además, impedimos las posibles influencias externas sobre el modulador de AF.

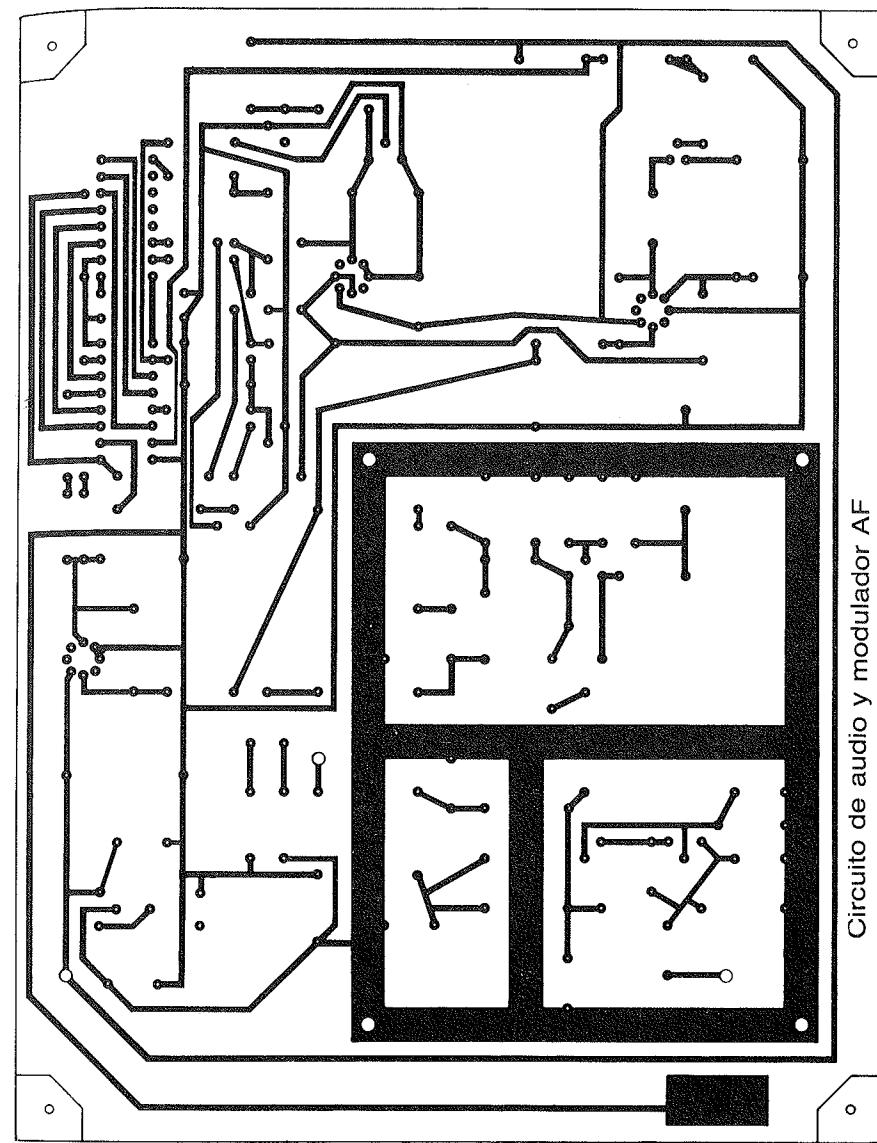
Si conectamos el computador TV a un aparato de televisión en color, éste no deberá tener ningún filtro de sonido de F.I. cerámico. Este requisito lo cumplen hoy la mayor parte de los aparatos de televisión.

Después de conectar el computador TV, ha de ajustarse la tensión de salida del regulador de tensión integrado 78L06 a $U_a = 6,5$ V. Con esta tensión alimentamos al oscilador de AF.

Tras el regulador de tensión hay un divisor de tensión que genera la tensión de alimentación para la sección de audio. Esta tensión ha de ajustarse a $U_a = 3,5$ V. Si, más tarde, surgen dificultades de sonido al ponerle en servicio, se reducirá la tensión a $U_a = 3,2$ V.

A continuación se introducirá un programa de prueba que se describe en el apartado 3.2. Ese programa puede introducirse en una RAM o bien estar programado en EPROM de prueba. Habrá que pulsar la tecla «programm» para que el microprocesador se reponga a la dirección ADRO. A continuación se pulsará la tecla «start».

Si en la pantalla de su televisor en color aparece una imagen, se definirá esta última de modo óptimo mediante el sintonizador del aparato. Deberá oírse un sonido limpio. El condensador variable de la sección de audio del modulador de AF se necesita sólo para la compensación del ruido. Moveremos el condensador hasta que obtengamos un mínimo de ruido. El ruido lo escucharemos en el altavoz del aparato.



Circuito de audio y modulador AF

Fig. 2.43 Diseño de placa

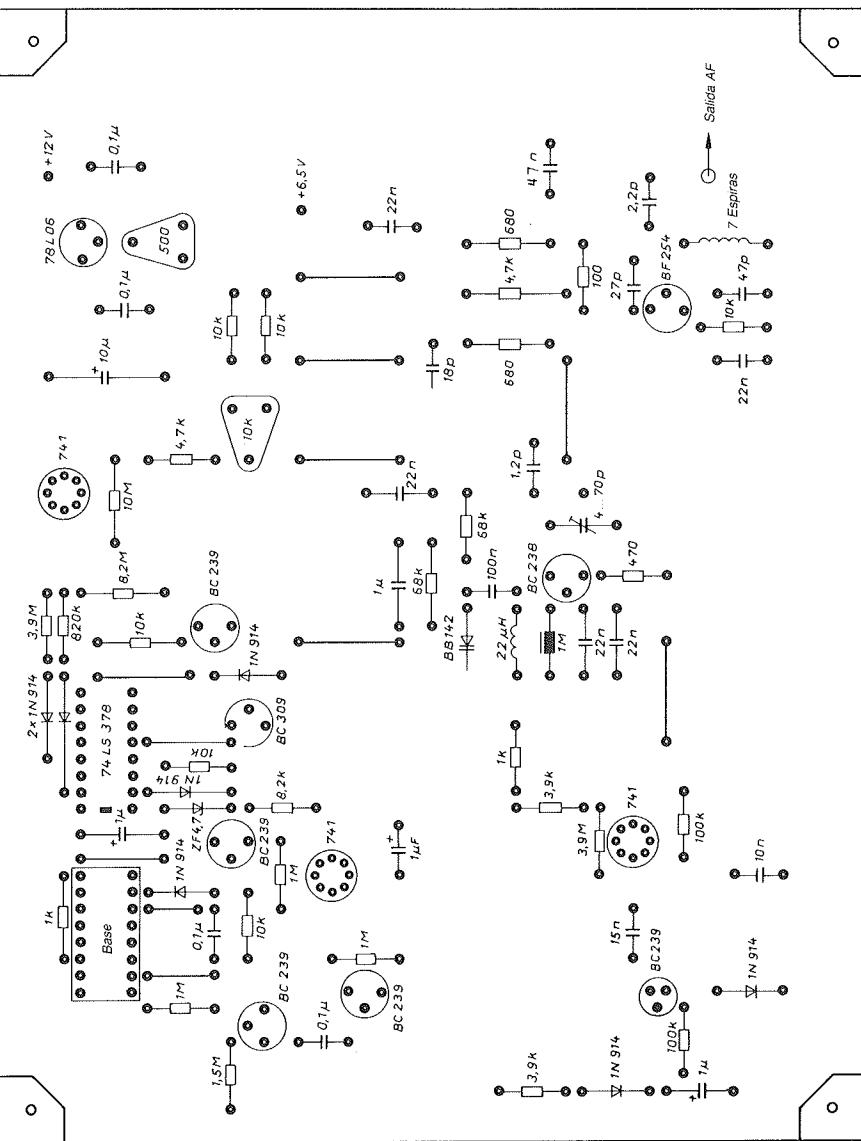


Fig. 2.44 Situación de componentes para el diseño de placa

to de televisión. Pulsar la tecla «start» y una de las del teclado. Al hacerlo ha de escucharse un sonido (señal de impacto).

La sintonización puede llegar a ser un problema para quienes no sean técnicos en televisión cuando aparece un sonido en imagen o una imagen en sonido. Para ello tenemos el potenciómetro para la tensión de alimentación de la sección de audio. Esta tensión ha de regularse de modo que no puedan aparecer las posibilidades antes mencionadas. Con el potenciómetro de la placa del sumador de video podemos obtener una imagen sin defectos en el televisor en color. Ha de procurarse que no se produzcan efectos Moires ni sobremodulación.

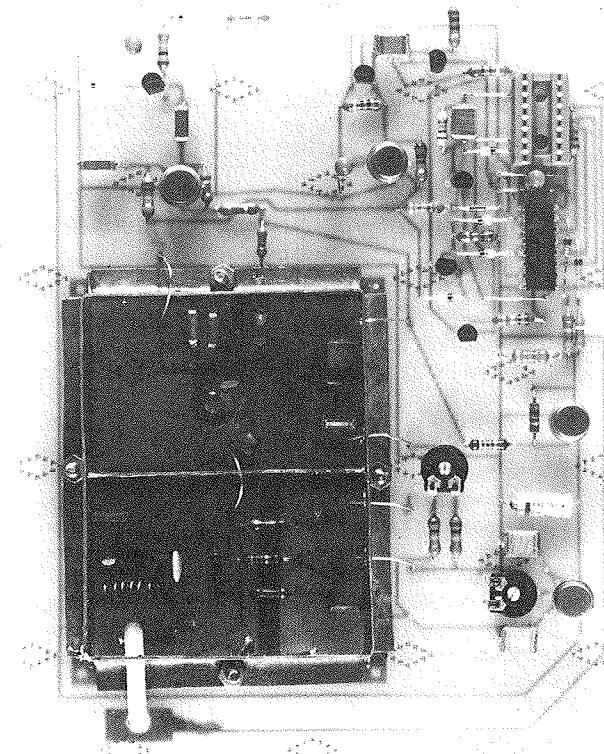


Fig. 2.45 Fotografía de placa

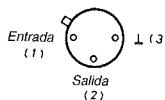


Fig. 2.46 Esquema de conexión del regulador de tensión 78106

Advertencia: El ajuste de este computador TV no suele dar lugar a ningún problema si se llevan a cabo ordenadamente las etapas del ajuste de prueba y se repiten éstas adecuadamente.

La figura 2.43 muestra la placa para la sección de audio y el modulador AF. Este último queda protegido contra los efectos secundarios no deseables, gracias a un borde suficientemente ancho. Sobre

ese borde debe acoplarse necesariamente una caja blindada de chapa de cobre o aluminio.

En la carcasa deberán hacerse orificios para que a través de ellos podamos unir los componentes interiores con los puntos externos de conexión. Para el condensador variable hemos de prever también un orificio de ajuste.

La figura 2.44 muestra el esquema de disposición de componentes para la sección de audio y el modulador AF. Ha de prestarse atención a la disposición de conexiones del amplificador 741 y de los transistores.

La figura 2.45 muestra una fotografía de esta placa, y la figura 2.46 el esquema de conexión del regulador de tensión 78L06.

Lista de componentes para la selección de audio y modulador

- | | |
|--------------------------------------|-----------------------|
| 1 Registro D 74LS378 | 1 Condensador 18 pF |
| 3 Amplificador 741 | 1 Condensador 2,2 pF |
| 1 Regulador de tensión 78L06 | 1 Regulador 10 kΩ |
| 1 Transistor BC238 | 1 Regulador 500 kΩ |
| 5 Transistores BC239 | 1 Resistencia 100 Ω |
| 1 Transistor BC309 | 1 Resistencia 470 Ω |
| 1 Transistor BF254 | 2 Resistencias 680 Ω |
| 7 Diodos 1N914 | 2 Resistencias 1 kΩ |
| 1 Diodo Zener ZD4,7 | 2 Resistencias 3,9 kΩ |
| 1 Diodo capacitivo BB142 | 1 Resistencia 4,7 kΩ |
| 1 Inductancia fija 22 μH | 1 Resistencia 8,2 kΩ |
| 1 Condensador de ajuste 4 pF...70 pF | 6 Resistencias 10 kΩ |
| 1 Condensador 10 μF/16V | 2 Resistencias 68 kΩ |
| 4 Condensadores 1 μF/16V | 2 Resistencias 100 kΩ |
| 5 Condensadores 0,1 μF bzw 100 nF | 1 Resistencia 820 kΩ |
| 5 Condensador 22 nF | 4 Resistencias 1 MΩ |
| 1 Condensador 15 nF | 1 Resistencia 1,5 MΩ |
| 1 Condensador 10 nF | 2 Resistencias 3,9 MΩ |
| 2 Condensadores 47 pF | 1 Resistencia 8,2 MΩ |
| 1 Condensador 27 pF | 1 Resistencia 10 MΩ |
| 1 Condensador 22 pF | |

2.5 FUENTE DE ALIMENTACION

Con la placa para la fuente de alimentación generamos la tensión de alimentación para todo el sistema. Aquí hemos de decidir ahora qué EPROM vamos a montar en la tarjeta de memoria de instrucciones. En el esquema de la figura 2.47 la EPROM 2708 con tres tensiones de alimentación. Por el contrario, con la EPROM 2716 necesitamos sólo dos tensiones de alimentación.

La figura 2.47 nos muestra el circuito para un aparato de red. Como transformador empleamos un núcleo de placas SM55 con dos devanados separados de 12 V/0,9 A, que puede adquirirse en cualquier establecimiento suministrador de componentes electrónicos.

Los dos devanados de 12 V están conectados a dos puentes rectificadores con lo que obtenemos respectivamente una tensión continua de unos 16 V. Ambas tensiones son aliadas mediante los condensadores y se aplican a los reguladores de tensión.

El regulador de tensión 78M12 produce una tensión de salida de +12 V con una corriente de 500 mA. Se suministra en una cápsula TD-220. La figura 2.48 muestra el esquema de conexión. Para nuestra apli-

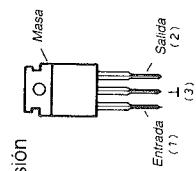
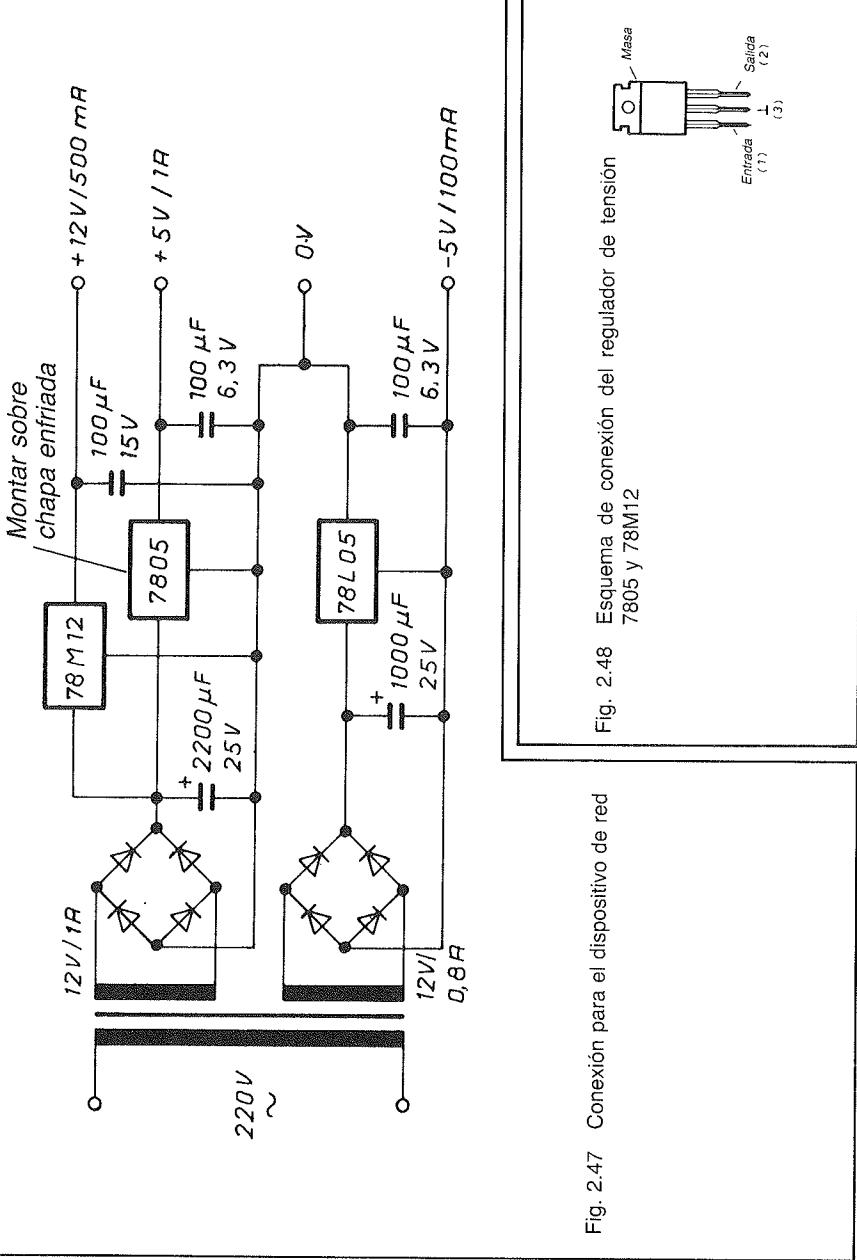


Fig. 2.48 Esquema de conexión del regulador de tensión 7805 y 78M12

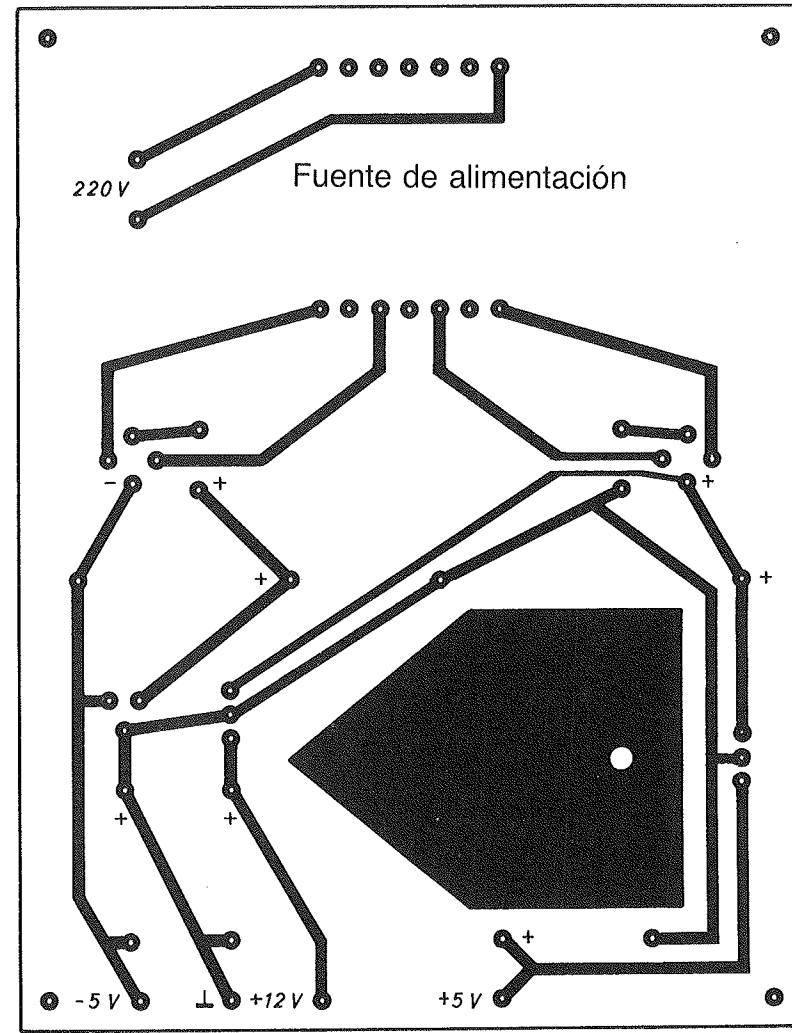


Fig. 2.49 Placa para la fuente de alimentación

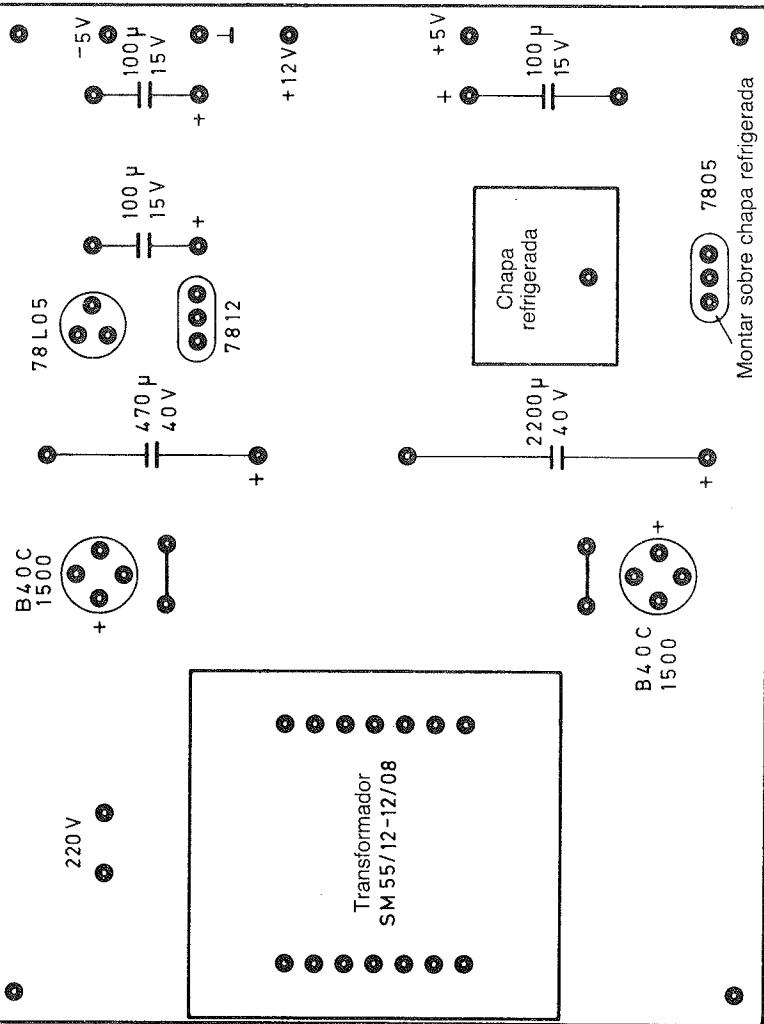


Fig. 2.50 Situación de componentes para la placa

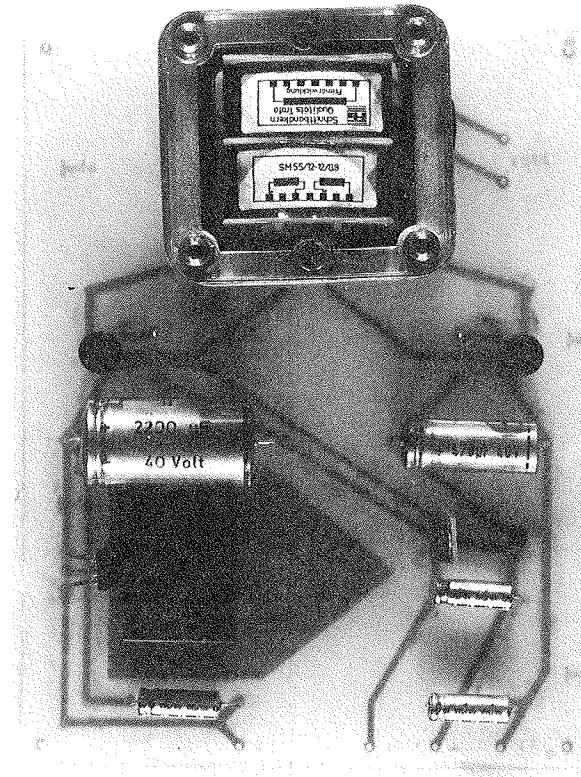


Fig. 2.51 Fotografía del dispositivo de red

Lista de componentes para la fuente de alimentación

- 1 Transformador de devanado con cinta metálica SM 55/12-12/08
- 2 Rectificadores B40C1500
- 1 Condensador 2200 μ F/40V
- 1 Condensador 470 μ F/40V
- 3 Condensadores 100 μ F/16V
- 1 Regulador de tensión 7805
- 1 Regulador de tensión 78L05
- 1 Regulador de tensión 78M12

6 INTERCONEXION DEL COMPUTADOR TV

Tras haber montado cada una de las 4 placas para el computador TV (sumador de video, microprocesador, circuito de audio y modulador AF, fuente de alimentación) comenzamos con el montaje conjunto y conexiones entre ellas.

Antes del ensamblaje conjunto ha de comprobarse, otra vez, todas las conexiones entre los componentes con objeto de que no queden restos de soldadura en las placas.

Atornillar las placas entre sí como indican las figuras 2.53 y 2.54. Como elementos de unión se utilizaron varillas roscadas.

Conéctese eléctricamente las diferentes placas. Ha de prestarse atención a la polaridad de los enchufes de conexión.

cación, el regulador no necesita enfriamiento suplementario y se dispone perpendicularmente a la placa.

El regulador de tensión 7805 genera una tensión de salida de +5 V para una corriente de 1 A. Este regulador hay que enfriarlo convenientemente. En la placa de la figura 2.49 hemos dispuesto una superficie de refrigeración de 20 cm². El regulador se suministra en una cápsula T0-220 y lo fijamos a la placa mediante un tornillo M3.

El regulador 78L05 genera una tensión de salida de +5 V para una corriente de 100 mA. Debido a la conexión realizada, no obtenemos tensión positiva, sino negativa, de -5 V.

En la figura 2.49 se muestra la placa, en la 2.50 el esquema de disposición de componentes y en la 2.51 una fotografía.

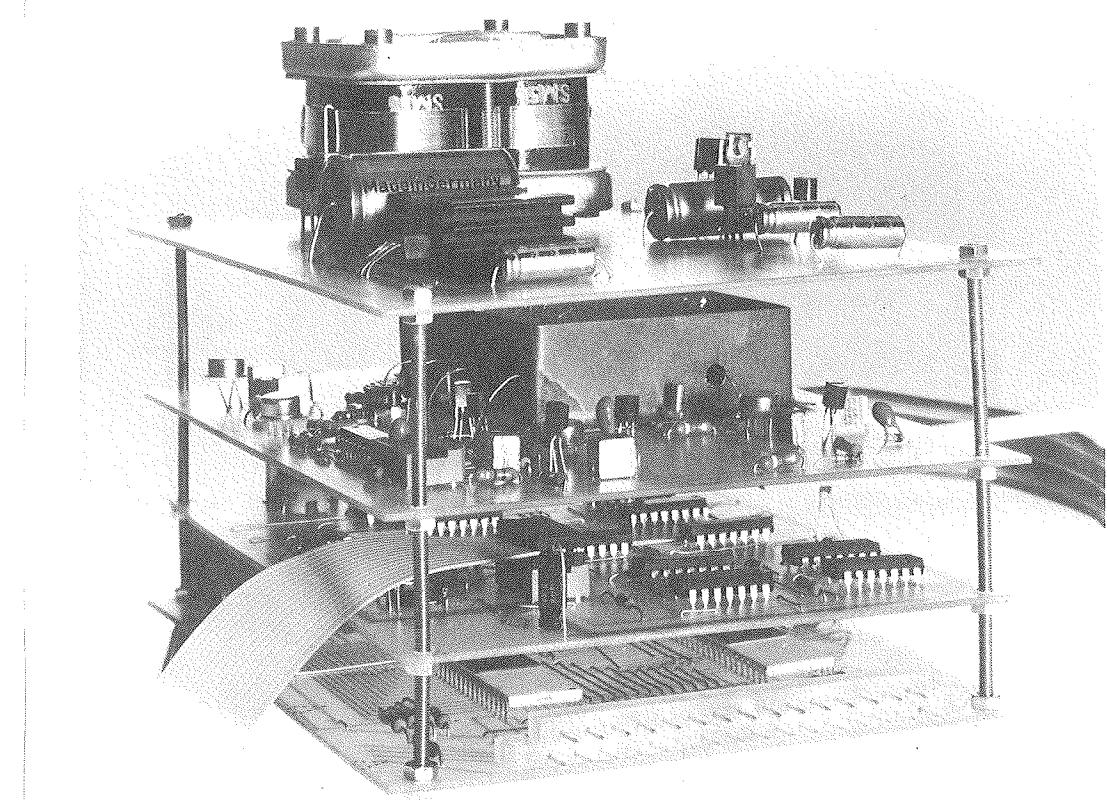


Fig. 2.52 Interconexión completa y conexión eléctrica

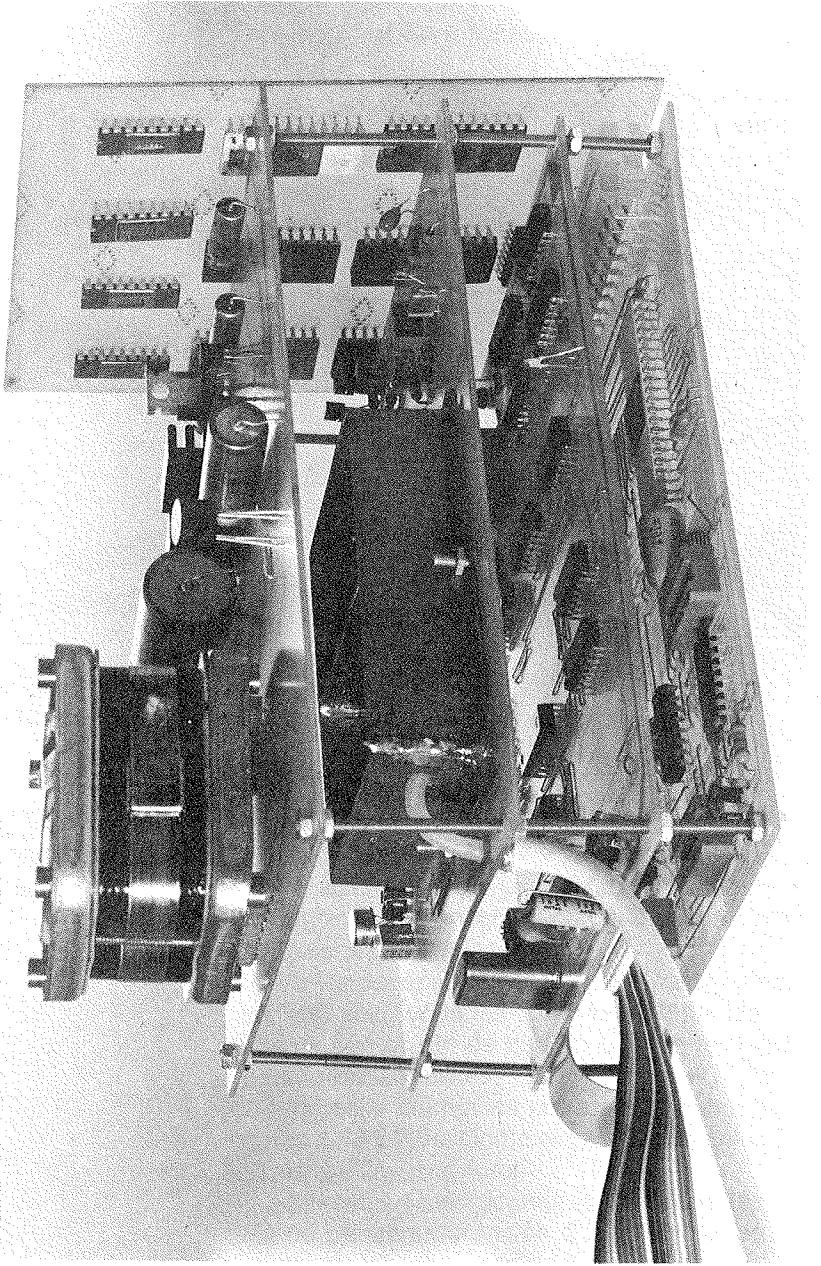


Fig. 2.53 Sistema completo

3. Programación del computador TV

Si disponemos de las instalaciones adecuadas podremos programar el computador TV con un lenguaje de programación como el BASIC, PASCAL o LOGO, si se domina bien alguno de ellos. Como accesorios necesitaremos un teclado usual en el código ASCII, un traductor del lenguaje de programación a lenguaje máquina y «mucha paciencia».

Mucho más sencillo es programarlo en lenguaje máquina de microordenador y tener «algo más de paciencia». El emulador, que se describe a continuación, puede facilitar la programación. Si el programa es «correcto» en el emulador, puede, entonces, documentarse e introducirse en el programador.

3.1 EMULADOR PARA EL DESARROLLO DE PROGRAMAS

Con el esquema de la figura 3.1 podemos desarrollar un programa para nuestro computador TV, sin tener que programar una EPROM. Mediante el emulador podemos borrar, modificar e introducir, de nuevo, nuestro programa tantas veces como deseemos. En el esquema hemos renunciado «conscientemente» a la electrónica usual en calculadoras con objeto de que el circuito sea transparente para los no especialistas y pueda construirse con facilidad.

El núcleo del emulador son las cuatro memorias de lectura-escritura del tipo 2114. Esas memorias se conectaron de modo que dispongamos de un emulador completamente direccionable de 2048 direcciones de 8 bits cada una. Con ello obtenemos un sistema de desarrollo para simulación de programas. Puesto que tal sistema sirve para desarrollar el software y para la simulación de la ejecución del programa, lo denominamos «emulador».

En el emulador diferenciamos dos fases de trabajo. En la primera de ellas introducimos, poco a poco, los datos de programa en el emulador. Entre tanto, el computador TV no aceptará datos del emulador. En la segunda fase, el computador TV recoge su programa del emulador y no podemos introducir

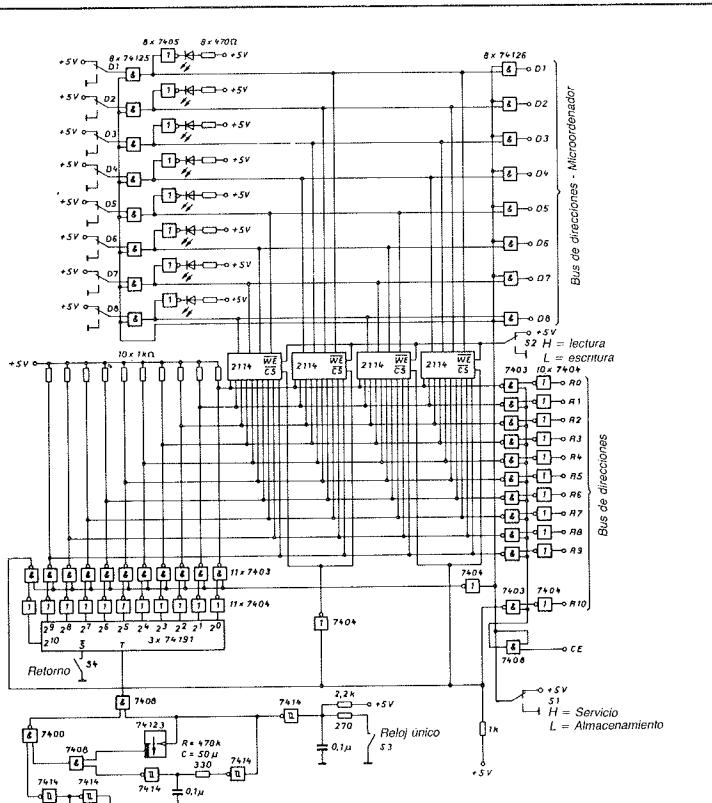


Fig. 3.1 Esquema de conexiones del emulador para el desarrollo de programas

datos. Tenemos, por tanto, un estado de «trabajo» y otro de «almacenamiento». Durante la fase de trabajo, el emulador simula los datos de la memoria de instrucciones y en la de almacenamiento podemos introducir, modificar o borrar los datos para la memoria de instrucciones. Con ello logramos una posibilidad de desarrollo de software.

En la fase de almacenamiento, el conmutador S1 ha de tener un nivel bajo. Ahora podrán trabajar las salidas de los componentes TTL 74125 y las salidas de los 74126 adoptan un estado Z (flotante). Gracias

a esto podremos registrar datos y la salida de éstos queda bloqueada.

Los datos se introducen mediante los 8 conmutadores de datos del emulador. Los datos introducidos se encuentran en las puertas NO, en las memorias de lectura-escritura y en los 8 componentes excitadores de BUS 74126. A través de las puertas NO, observamos una indicación inmediata del flujo de datos en el sistema. Si damos un nivel alto en una línea del sistema, al transistor de salida de una puerta NO conduce y comuta, con lo que los LED se encienden, es decir, obtenemos una indicación del nivel directa y con «buffer». Sin embargo, los LED no pueden conectarse directamente en el sistema de datos. Cuando se dispone de los datos correctos gracias a los 8 conmutadores de entrada, pulsamos el conmutador S2 y se almacenarán las informaciones en las RAM. Después, comprobaremos, mediante los indicadores, si la formación se ha almacenado.

Sin embargo, durante la entrada de datos pueden surgir problemas en este esquema. Si en las memorias de lectura-escritura ya se han almacenado datos, los LED pueden indicar informaciones erróneas puesto que aquellos se encuentran predominantemente en el sistema interno. Pero si pulsamos el conmutador S2 para activar el modo de escritura de la memoria de lectura-escritura obtendremos siempre los indicadores correctos. Podemos eliminar ese inconveniente del esquema tomando medidas de carácter electrónico suplementarias. La medida más sencilla es situar 8 LED con resistencia adicional en serie en los conmutadores de datos. Con ello obtenemos una información inmediata.

Si pasamos el conmutador S1 de la posición de «almacenar» a la posición de «trabajo», las puertas de entrada tendrán un nivel bajo y las salidas adoptan un nivel Z. Las puertas de salida están en el nivel en el que dan paso y así, el sistema interno de datos se encontrará conectado al sistema de datos del computador TV. Ahora queda disponible para la si-

mulación el programa almacenado en la memoria de lectura-escritura.

Cuando se lean datos a partir del emulador, no podrá pulsarse el conmutador S2 con el fin de que no aparezcan informaciones erróneas. Ello puede evitarse mediante una puerta Y o una puerta NAND.

El sistema de datos del emulador no es susceptible a las averías, puesto que tenemos un sistema BUS puro.

Sin embargo, en el sistema de direcciones pueden surgir dificultades ya que los componentes RAM 2114 sólo tienen una entrada CS.

Para el sistema de direcciones del emulador necesitamos las direcciones del computador TV y las direcciones para la entrada de datos. Las primeras se encuentran a través de las entradas A0 y A9. Mediante las puertas NO recibimos un valor de dirección negado que se convertirá en el valor correcto a través de la puerta NAND situada a continuación. Con la dirección A10 determinamos el par de memorias RAM que han de trabajar. La salida de la puerta NAND, a través de una conexión de «Y cableada», se encuentra unida directamente a las entradas CS de un par de memorias RAM por una parte y por otra parte, esa salida es negada mediante una puerta NO. De ese modo, conseguimos un control entre las direcciones 0 a 1023 y 1024 a 2047.

Si el conmutador S1 se encuentra en la posición de «trabajo», la puerta con la entrada CE desde el computador TV quedará a un nivel alto. Con ello esa puerta Y queda preparada cuando el microprocesador 2650 envía para el trabajo de lectura un nivel alto a través de su bus de control. Si surgieran dificultades, tendremos que negar la entrada CE del emulador mediante una puerta NO previa. Ha de tenerse presente este proceso.

Si conmutamos S1 a la posición de «almacenar», el computador TV interrumpirá el direccionamiento externo. Las salidas de las puertas NAND quedan bloqueadas y tendremos en el sistema inter-

no de direcciones un estado que sólo podrá modificarse por ese mismo sistema.

Para el direccionamiento interno de la memoria de lectura-escritura disponemos de un contador de 0 a 2047. Con él podemos llegar a cada una de las direcciones del sistema de RAM del emulador. Las direcciones A0 a A9 se obtienen directamente en las RAM a través de las conexiones de «Y cableada». Las salidas del contador son negadas y después introducidas en el sistema interno de direccionamiento de RAM a través de la puerta NAND. La conmutación entre los dos pares de RAM tiene lugar mediante un control indirecto (puerta NO) y mediante un control directo de las entradas CS. De esta manera, conseguimos nuevamente una dirección subdividida desde 0 a 1023 y desde 1024 a 2047.

Con el conmutador S3 introducimos el ritmo de sincronización para el oscilador del circuito. Si el conmutador permanece pulsado más de 0,5 segundos, el contador avanza automáticamente a un ritmo aproximado de 10 impulsos por segundo. Conseguimos una comodidad de servicio importante para la entrada y la búsqueda de errores. Si liberamos el conmutador S3 (abierto), el circuito queda inmediatamente fuera de servicio y podremos avanzar hacia la correspondiente dirección mediante una leve pulsación de tecla. Con cada pulsación incrementamos el estado del contador en una posición.

Con el conmutador S4 efectuamos la reposición del contador a 0 y podremos introducir nuevamente un programa en el emulador a partir de dirección 0. Como empleamos el contador del tipo 74191, podemos ajustar un valor previo. Ejemplo: Necesitamos la dirección 4FF y el contador pasa inmediatamente a esta dirección. Mediante un conmutador adicional podremos contar también en sentido creciente o decreciente.

El esquema de la figura 3.1 puede ampliar sus posibilidades mediante numerosas variantes de circuitos. Así es posible introducir datos en forma hexa-

decimal y también obtener la reproducción adecuada de los datos mediante indicadores.

En este pautado se muestran los posibles juegos de pelota para el computador TV. Modificando adecuadamente los programas es posible obtener numerosas variantes de dichos juegos.

Las figuras 3.2 a 3.13 reproducen los juegos de pelota. Cada uno de ellos se descompone a su vez en 5 subjuegos. De esta manera conseguimos 60 juegos de pelota distintos.

Las variaciones para los subjuegos son:

- Para los principiantes: La pelota tiene siempre una trayectoria lenta.
- Para los entrenados: La pelota tiene inicialmente una trayectoria lenta. Después de 2 segundos aproximadamente la pelota se acelera y tiene una trayectoria rápida.
- Para «profesionales»: La pelota tiene desde el primer momento en que se pulsa la tecla «start» una trayectoria rápida.
- Truco de ventaja I: La pelota tiene una trayectoria lenta. Pulsando la tecla I de un adaptador manual, la pelota se acelera inmediatamente. Al soltar la tecla la pelota ralentiza inmediatamente su trayectoria. Pulsando la tecla II, la pelota modifica su trayectoria oblicuamente. Al soltar esa tecla volvemos a tener una trayectoria rectilínea.
- Truco de ventaja II: La pelota adquiere una trayectoria rápida inmediatamente después de pulsar la tecla «start». Al pulsar la tecla II se altera la trayectoria. La tecla I queda bloqueada en este caso.

3.2.1 Tenis individual

En el apartado 3.3 se expone el programa para los 60 juegos de pelota. Con las direcciones que descri-

bimos ahora podremos modificar las funciones del juego.

En la dirección 678 comienzan los valores para delimitar el campo de juego. Esa dirección establece la parte izquierda de la primera línea de delimitación fina. La dirección siguiente 679 define la parte de recha.

Las siguientes direcciones definen consecutivamente las líneas de demarcación gruesas y finas según se explicó al hablar de la interface de video programable. La última dirección, a este respecto, es la 69B. Así tenemos 36 direcciones con magnitudes distintas:

Dirección 678: izquierda arriba, línea fina

Dirección 678: derecha arriba, fina

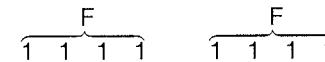
Dirección 67A: izquierda, gruesa

Dirección 67B: derecha, gruesa

Dirección 67C: izquierda, fina

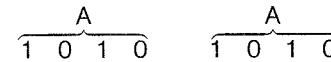
De este modo, se obtienen 18 marcas gruesas y 18 finas.

Al analizar en el apartado 3.3 la dirección 678 observamos el contenido «FF». Este valor hemos de descomponerlo del modo siguiente:



Así, obtendremos para la parte izquierda de esa línea de video fina una sucesión de puntos que ampliaremos convenientemente mediante la dirección 69C. Como en esta dirección está igualmente un valor «FF» obtendremos un segmento de 8 impulsos de reloj (debido a la interface de video) y en la pantalla aparecerá una raya.

La dirección 67A tiene un valor «AA». Este valor se descompone así:



De esta forma, obtenemos en la parte izquierda de la línea video ancha una relación raya-espacio:

1 0 1 0 1 0 1 0
| | | | | | | |

En la dirección 67B hay el valor «55» que descompondremos así:

5 5
— — — — —
0 1 0 1 0 1 0 1

Con ello las direcciones 67A y 67B nos proporcionan en la pantalla la imagen siguiente:

1 0 1 0 1 0 1 0 0 1 0 1 0 1 0
| | | | | | | | | | | | | |

Mediante el valor de la dirección 69C podemos estirar nuevamente las rayas y obtenemos:

1 0 1 0 1 0 1 0 0 1 0 1 0 1 0
■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■

Ha de prestarse atención a la distancia entre las rayas.

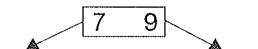
Después de la dirección de raya ancha 67A y 67B tenemos, de nuevo, una dirección de raya fina 67C y 67D.

Mediante las 36 direcciones en el programa y las 5 direcciones consecutivas para alargar puntos y rayas obtenemos la imagen de la fotografía en la figura 3.2.

Con la dirección 648 definimos los colores para la demarcación del campo y el fondo. En la dirección mencionada hay un valor «79». Obtenemos, así, una demarcación en color blanco y un color de fondo azul. El color de la demarcación establece a la vez el de los marcadores. En el programa se incluye una instrucción especial que niega el valor del color de demarcación. Así, obtenemos el color complementa-

rio del blanco; es decir, color negro. El marcador para el juego y la puntuación aparece en negro.

Los colores para la demarcación y el fondo se establecen del modo siguiente:



Demarcación Fondo
Marcador (neg.)

Examinando la composición de colores siguiente pueden alterarse adecuadamente los colores. Ha de tenerse en cuenta siempre los colores complementarios.

Valor	Color
0	Blanco
1	Amarillo
2	Violeta
3	Rojo
4	Azul verdoso
5	Verde
6	Azul
7	Negro

Téngase en cuenta los valores de los distintos colores. Esta composición no ha sido establecida por el autor, sino por el fabricante de la interface de video programable. No obstante, esta composición es sencilla.

Demarcación		Fondo	
Valor	Color	Valor	Color
0	Negro	8	Negro
1	Azul	9	Azul
2	Verde	A	Verde
3	Azul verdoso	B	Azul verdoso
4	Rojo	C	Rojo
5	Violeta	D	Violeta
6	Amarillo	E	Amarillo
7	Blanco	F	Blanco

Mediante la dirección 654 establecemos el color de las raquetas para este juego. De nuevo hemos de aludir a las características de la interface de video programable. Los colores se establecen mediante la tabla de la página anterior.

En la programación de la dirección 654 tenemos el valor «09». Este valor descompuesto según los condicionantes de la interface de video programable nos da:

$$09 \triangleq \overbrace{0\ 0\ 0}^0 \overbrace{0\ 1}^9$$

1 1

Ambas raquetas tienen color amarillo. Si deseáramos otro color diferente, hemos de atenernos a lo siguiente:

Ejemplo 1: raqueta izquierda: negra $\triangleq 7 \triangleq 1\ 1\ 1$
raqueta derecha: azul $\triangleq 6 \triangleq 1\ 1\ 0$

programación: $\overbrace{0\ 0\ 1}^3 \overbrace{1\ 1\ 0}^E \triangleq 3E$

izq. dcha.

Ejemplo 2: raqueta izquierda: azul ver. $\triangleq 4 \triangleq 1\ 0\ 0$
raqueta derecha: verde $\triangleq 5 \triangleq 1\ 0\ 1$

programación: $\overbrace{0\ 0\ 1}^2 \overbrace{0\ 1\ 0}^5 \triangleq 25$

izq. dcha.

La conversión de cantidades hexadecimales en binarias o viceversa puede hacerse siempre directamente.

Con la dirección 655 establecemos el color de la pelota. Para ello nos atenemos a la misma tabla de las raquetas. En esa dirección tenemos el valor «05», es decir:

05 \triangleq Verde

Mediante la dirección 7C6 podemos modificar la forma de la raqueta. Pero, en este caso, se altera simultáneamente la forma de ambas raquetas. Podemos también modificar simplemente el ancho de las mismas.

Para las cifras de los marcadores tenemos dos representaciones diferentes. Las hemos establecido mediante el valor «00» en la dirección 7F0.

Al comienzo del juego, la bola sale o bien hacia la derecha o hacia la izquierda. Esto puede programarse también. Cuando la pelota sale del campo de juego se marca un punto y se indica en el marcador correspondiente. Mediante sonidos acústicos diferentes apreciamos las «faltas en el juego».

Cuando la pelota encuentra la demarcación de campo superior o inferior, la pelota rebota de modo que el ángulo de incidencia sea igual al de reflexión. Esta programación del ángulo es la misma para ambos jugadores.

En el apartado 3.3 se indican las instrucciones correspondientes a las direcciones individuales que permiten modificar el curso del juego, los valores acústicos y los datos.

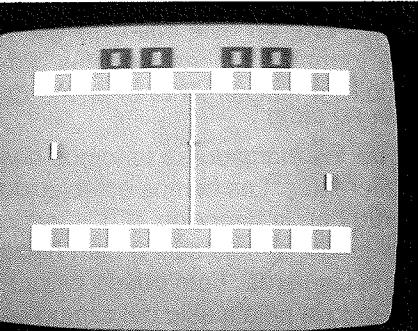
3.2.2 Tenis de dobles

En este juego empleamos la misma demarcación de campo que en el tenis individual (apartado 3.2.1). Pero ahora podemos cambiar los colores de nuevo como deseemos. La distancia entre las dos raquetas las establecemos mediante las direcciones 7E7 a 7EF.

La figura 3.3 muestra un encuentro de dobles.

3.2.3 Fútbol

La figura 3.4 muestra un campo de juego de fútbol con un portero que puede ser a la vez defensa, medio o delantero.



3.2 Imagen para tenis individual

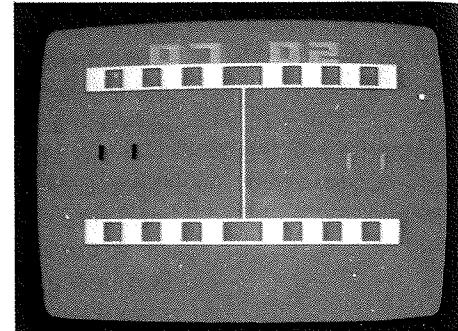


Fig. 3.3 Imagen para tenis de dobles

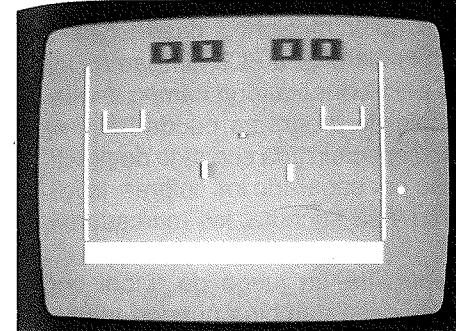


Fig. 3.8 Imagen para baloncesto con dos jugadores horizontales

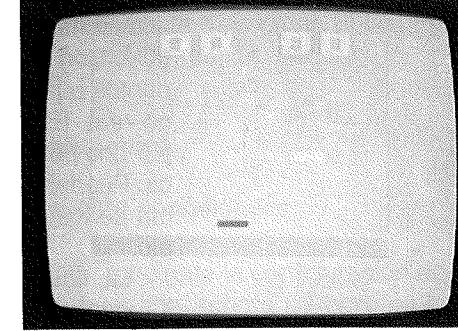
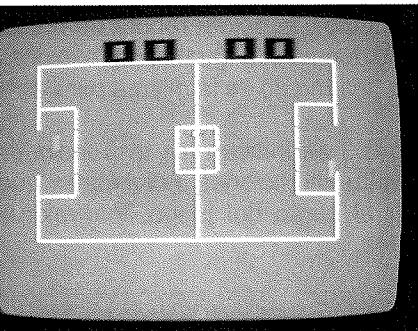


Fig. 3.9 Imagen para baloncesto con dos jugadores verticales



3.4 Imagen para fútbol sencillo

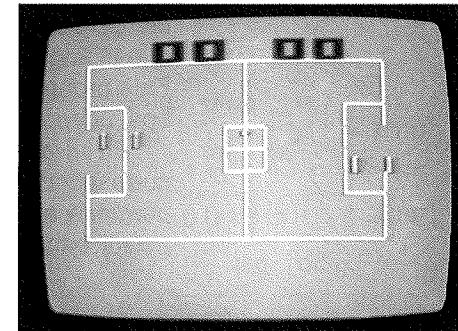


Fig. 3.5 Imagen para fútbol con cuatro jugadores

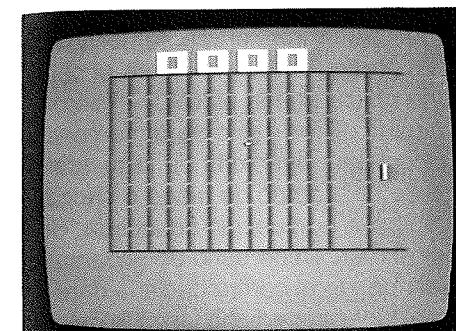


Fig. 3.10 Imagen para juego de obstáculos 1

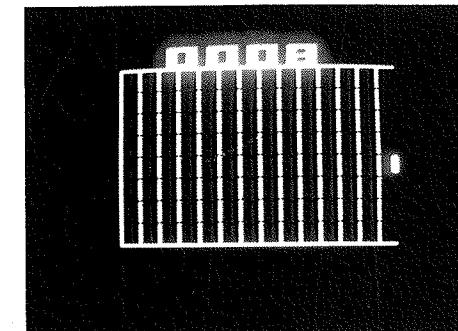
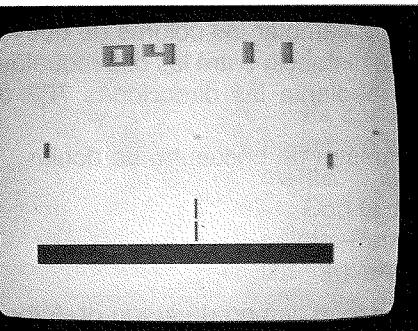


Fig. 3.11 Imagen para juego de obstáculos 2



3.6 Imagen para tenis de mesa

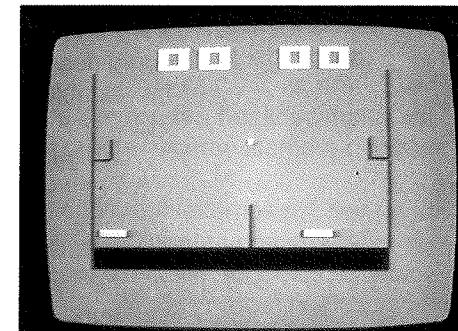


Fig. 3.7 Imagen para balón-volea

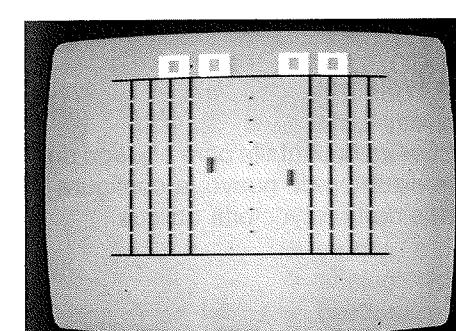


Fig. 3.12 Imagen para juego de obstáculos 3

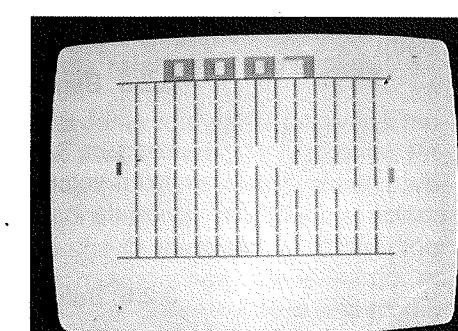


Fig. 3.13 Imagen para juego de obstáculos 4

Los valores para el campo de juego los tenemos almacenados entre las direcciones desde 6A1 a 6C4. Con las direcciones 6C5 a 6C9 alargamos los puntos o las rayas para conseguir un campo de juego real. Este ejemplo de programa puede hacerse por una persona no especializada con algo de práctica. Pueden definirse por separado los colores, la forma de los jugadores y su anchura, los marcadores, etc.

La figura 3.5 muestra un juego de fútbol con 4 jugadores. En este caso, también pueden modificarse todos los datos para el juego.

3.2.4 Tenis de mesa

La figura 3.6 nos muestra la imagen de pantalla de un tenis de mesa. La particularidad reside en el cálculo de la trayectoria oblicua de la pelota. El microprocesador 2650, a partir de los datos de la memoria de instrucciones, calcula la trayectoria de la pelota.

Si la pelota se sale del campo de juego o choca contra la red, se efectúa la puntuación adecuada. Si la pelota se sale por el margen derecho de la pantalla, ésta vuelve a salir desde el centro hacia la derecha. Por ejemplo, si el jugador derecho lanza la pelota contra la red, el izquierdo recibirá la pelota después de haberse producido esa indicación.

Mediante el programa pueden alterarse todos los datos como color, tamaño, etc.

3.2.5 Balón volea (voleibol)

La figura 3.7 nos muestra la pantalla para el juego de balón volea. Aquí utilizamos de nuevo la configuración de instrucciones de la trayectoria oblicua.

3.2.6 Baloncesto

En el juego de baloncesto disponemos de dos jugadores horizontales (figura 3.8) o dos jugadores verti-

cales (figura 3.9). Empleamos nuevamente la estructura de instrucciones de la trayectoria oblicua y podemos modificar todos los demás datos.

3.2.7 Juego de obstáculos

En este juego hay cuatro posibilidades:

Figura 3.10: La pelota, mediante la configuración de las instrucciones de memoria, rebota siempre en la raqueta. Por cada obstáculo derribado se incrementa el marcador. Si el jugador no da a la pelota comienza un juego de nuevo.

Figura 3.11: La configuración de instrucciones en memoria hace que la pelota rebote en la pared frontal. Por cada obstáculo derribado el marcador se incrementa. Si el jugador no da a la pelota comienza un nuevo juego.

Figura 3.12: Tenemos en la mitad de la imagen 2 jugadores. La pelota rebota en las raquetas. Si aquélla derriba un obstáculo éste desaparece y la pelota se refleja hacia el contrario. Si la pelota llega hasta detrás de los obstáculos comienza un juego nuevo.

Figura 3.13: Tenemos 2 jugadores y los obstáculos entre ellos. Ambos jugadores han de mantener la pelota en juego. Si la pelota pasa detrás de los jugadores, el juego finaliza.

3.2.8 Observaciones relativas a los juegos

Mediante los adaptadores manuales podemos mover los jugadores en sentido vertical y horizontal. Pero si los desplazamos muy rápidamente, el microprocesador 2650 no podrá calcular el rebote o el acierto ya que, a pesar de que la velocidad del convertidor analógico/digital es muy elevada, sin embargo, no basta para este proceso. Sobrepasando una modificación de velocidad de jugador de 1 m/s no podemos contar con que el cálculo tenga algún efecto.

3.3 PROGRAMA PARA LOS JUEGOS DE PELOTA

Dirección	Contenido	Comentario
000	20 EORZ	O-exclusiva entre R0 y R0 (se borra R0)
001	1B BCTR	instrucción para direccionamiento relativo
002	01 valor	valor relativo de la dirección
003	17 RETC	retorno condicional desde subprograma
004	76 PPSU	activar la palabra superior de estado de programa
005	00 libre	
006	76 CPSL	borrar la palabra inferior de estado de programa
007	00 libre	
008	77 PPSL	activar la palabra inferior de estado de programa
009	00 libre	
00A	74 CPSU	borrar la palabra superior de estado de programa
00B	40 HALT	detener microprocesador 2650 y reponer sólo con RESET
00C	C1 STRZ	almacenar R0 en R1
00D	CD STRA	almacenar valor de dirección absoluta en R1
00E	7E ADR	valor más alto de la dirección
00F	FF ADR	valor más bajo de la dirección
010	F9 BDRR	Calcular dirección de salto relativa en R1
011	7B valor	Valor para calcular dirección relativa de salto
012	3F BSTA	instrucción de salto absoluta (depende del bit de estado)
013	05 ADR	valor superior de la dirección
014	7A ADR	valor inferior de la dirección
015	3F BSTA	instrucción de salto absoluta (depende del bit de estado)
016	04 ADR	valor superior de la dirección
017	66 ADR	valor inferior de la dirección
018	0C LODA	carga absoluta de R0 con la dirección siguiente
019	1F ADR	valor superior de la dirección
01A	50 ADR	valor inferior de la dirección
01B	18 BCTR	instrucción para direccionamiento relativo
01C	03 valor	valor de dirección relativa
01D	3F BSTA	instrucción de salto absoluto (depende del bit de estado)
01E	00 ADR	valor superior de dirección
01F	A0 ADR	valor inferior de la dirección
020	3F BSTA	instrucción de salto absoluto (depende del bit de estado)
021	04 ADR	valor superior de la dirección
022	51 ADR	valor inferior de la dirección
023	1B BCTR	instrucción para direccionamiento relativo
024	70 valor	valor relativo de dirección
025	0F LODA	carga absoluta de R3 con la dirección siguiente
026	1F ADR	valor superior de la dirección
027	4E ADR	valor inferior de la dirección
028	0F LODA	carga absoluta de R3 con la dirección siguiente
029	66 ADR	valor superior de la dirección
02A	3B ADR	valor inferior de la dirección
02B	C2 STRZ	contenido de R0 a R2
02C	0B LODR	direccionamiento relativo mediante R3
02D	1F valor	valor para direccionamiento relativo
02E	59 BRNR	direccionamiento relativo mediante R1
02F	0E valor	valor para direccionamiento relativo
030	66 IORI	direccionamiento inmediato mediante R2
031	0C valor	valor para direccionamiento inmediato
032	E1 COMZ	comparación entre R0 y R1
033	9A BCFR	salto relativo desde R2
034	07 valor	valor para salto relativo
035	0E LODA	instrucción de carga absoluta para R2
036	66 ADR	valor superior de la dirección
037	0D ADR	valor inferior para la dirección
038	E1 COMZ	Comparar R0 y R1
039	99 BCFR	salto relativo mediante R1
040	01 valor	valor para salto relativo
041	01 LODZ	Registro R1 a R0
042	CC STRA	direccionamiento absoluto mediante R0
043	1F ADR	valor superior de la dirección
044	0C ADR	valor inferior de la dirección
045	E1 COMZ	Comparar R0 y R1
046	9A BCFR	salto relativo mediante R2

Dirección	Contenido	Comentario
047	07 valor	valor para salto relativo
048	0E LODA	instrucción de carga absoluta mediante R2
049	66 ADR	valor superior de dirección
04A	0D ADR	valor inferior de dirección
04B	E1 COMZ	Comparar R0 y R1
04C	99 BCFR	salto relativo mediante R1
04D	01 valor	valor para salto relativo
04E	01 LODZ	contenido de R1 a R0
04F	CC STRA	valor de R0 pasa a dirección absoluta
050	1F ADR	valor superior de dirección
051	1C ADR	valor inferior de dirección
052	0D LODA	instrucción de carga absoluta mediante R1
053	1F ADR	valor superior de dirección
054	57 ADR	valor inferior de dirección
055	0E LODA	instrucción de carga absoluta mediante R2
056	66 ADR	valor superior de dirección
057	10 ADR	valor inferior de dirección
058	E7 COMI	comparación inmediata entre R3 y valor
059	07 valor	valor para la comparación inmediata
06A	9A BCFR	salto calculado relativo mediante R2
06B	06 valor	valor para salto relativo
06C	E5 COMI	comparación inmediata entre R2 y valor
06D	60 valor	valor para comparación inmediata
06E	1A BCTR	salto calculado relativo mediante R2
06F	0C valor	valor para salto relativo
06G	AB SUBI	resta inmediata de R1 y valor
06H	60 valor	valor para resta inmediata
06I	E1 COMZ	comparar R0 y R1
06J	9A BCFR	salto calculado relativo mediante R2
06K	07 valor	valor para salto relativo
06L	0E LODA	instrucción de carga absoluta mediante R2
06M	66 ADR	valor superior de dirección
06N	0E ADR	valor inferior de dirección
06O	06 LODA	instrucción de carga absoluta, valor en R1
07O	1F ADR	valor superior de dirección
071	58 ADR	valor inferior de dirección
072	0E LODA	instrucción de carga absoluta, valor a R2
073	66 ADR	valor superior de dirección
074	0F ADR	valor inferior de dirección
075	E7 COMI	comparación inmediata entre R3 y valor
076	07 valor	valor para comparación inmediata
077	9A BCFR	salto relativo calculado mediante R2
078	06 valor	valor para salto relativo
079	E5 COMI	comparación inmediata entre R1 y valor
07A	08 valor	valor para comparación inmediata
07B	19 BCTR	salto calculado relativo mediante R1
07C	0C valor	valor para salto relativo
07D	85 ADDI	conexión Y entre R1 y valor
07E	37 valor	valor para operación Y inmediata
07F	E1 COMZ	comparar R0 y R1
080	99 BCFR	salto calculado relativo mediante R1
081	07 valor	valor para salto relativo
082	0E LODA	instrucción de carga absoluta mediante R2
083	66 ADR	valor superior de dirección
084	11 ADR	valor inferior de dirección
085	E1 COMZ	comparar R0 y R1
086	9A BCFR	salto relativo calculado mediante R2
087	01 valor	valor para salto relativo
088	01 LODZ	cargar contenido de R1 en R0
089	CC STRA	direcciónamiento absoluto, valor de R0 pasa a dirección
08A	1F ADR	valor superior de dirección
08B	1A ADR	valor inferior de dirección
08C	17 RETC	retorno condicional desde subprograma
08D	06 LODI	carga inmediata de R2 con valor

Dirección	Contenido	Comentario	Dirección	Contenido	Comentario
08E	01 valor	valor para carga inmediata	OD6	1F ADR	valor superior de dirección
08F	04 LODI	carga inmediata de R0 con valor	OD7	CB ADR	valor inferior de dirección
090	04 valor	valor para carga inmediata	OD8	CD STRA	almacenar R1 en dirección absoluta
091	4C ANDA	direcciónamiento absoluto y operación Y de R0	OD9	1F ADR	valor superior de dirección
092	1F ADR	valor superior de dirección	ODA	54 ADR	valor inferior de dirección
093	68 ADR	valor inferior de dirección	ODB	68 IORR	O-inclusiva relativa entre R0 y valor
094	62 IORZ	operación Y entre R0 y R2	ODC	F1 valor	valor para operación relativa
095	C8 STRR	almacenamiento relativo mediante R0	ODD	CC STRA	almacenar contenido de R0 en dirección absoluta
096	F6 valor	valor para almacenamiento relativo	ODE	1F ADR	valor superior de dirección
097	17 RETC	retorno condicional desde subprograma	ODF	53 ADR	valor inferior de dirección
098	06 LODI	carga inmediata de R2 con valor	OEG	OF LODA	dirección absoluta a R3
099	02 valor	valor para carga inmediata	OE1	1F ADR	valor superior de dirección
09A	1B BCTR	salto relativo incondicional	OE2	4E ADR	valor inferior de dirección
09B	73 valor	valor para salto relativo	OE3	08 LODR	instrucción relativa de carga entre R0 y valor
09C	06 LODI	carga inmediata de R2 con valor	OE4	F9 valor	valor para dirección relativa
09D	03 valor	valor para carga inmediata	OW6	44 ANDI	operación Y inmediata entre R0 y valor
09E	6F valor	valor para salto relativo	OE6	10 valor	valor para operación Y
0A0	20 EORZ	O-exclusiva entre R0 y R0 (R0 se borra)	OE7	18 BCTR	dirección de memoria relativa calculada entre R0 y valor
0A1	05 LODI	carga inmediata de R1 con valor	OE8	05 valor	valor para dirección relativa
0A2	03 valor	valor para carga inmediata	OE9	3F BSTA	dirección absoluta de salto
0A3	CD STRA	valor de posición de memoria absoluta a R1	OEA	02 ADR	valor superior de dirección
0A4	7F ADR	valor superior de dirección	OEB	77 ADR	valor inferior de dirección
0A5	62 ADR	valor inferior de dirección	OEC	1B BCTR	dirección de memoria relativa calculada
0A6	F9 BDRE	salto calculado relativo mediante R1	OED	08 valor	valor para dirección relativa
0A7	7B valor	valor para salto relativo	OEE	0E LODA	dirección absoluta a R2
0A8	OD LODA	cargar R1 en posición de memoria absoluta	OEF	1F ADR	valor superior de dirección
0A9	1F ADR	valor superior de dirección	OFO	54 ADR	valor inferior de dirección
0AA	OF ADR	valor inferior de dirección	OF1	46 ANDI	operación Y inmediata entre R2 y valor
0AB	18 BCTR	dirección de memoria calculada relativa mediante R0	OF2	0A valor	valor para operación Y
0AC	02 valor	valor del direcciónamiento relativo	OF3	BC BSFA	salto absoluto
0AD	F9 BDRR	dirección de memoria calculada relativa entre R1 y valor	OF4	01 ADR	valor superior de dirección
0AE	03 valor	valor de la dirección de memoria relativa	OF5	C5 ADR	valor inferior de dirección
0AF	CC STRA	almacenar contenido de R0 en dirección absoluta	OF6	0E LODA	dirección absoluta a R2
0B0	1F ADR	valor superior de dirección	OF7	1F ADR	valor superior de dirección
0B1	C7 ADR	valor inferior de dirección	OF8	4F ADR	valor inferior de dirección
0B2	C9 STRR	contenido de R1 a dirección relativa calculada	OF9	E6 COMI	comparación inmediata entre R2 y valor
0B3	F5 valor	valor para dirección relativa	OF0A	01 valor	valor para comparación
0B4	OC LODA	valor de dirección absoluta a R0	OFB	98 BCPR	dirección relativa calculada entre R0 y valor
0B5	1F ADR	valor superior de dirección	OFC	04 valor	valor para el cálculo
0B6	62 ADR	valor inferior de dirección	OFD	5B BSTR	dirección relativa calculada
0B7	18 BCTR	dirección relativa calculada mediante R0 y valor	OFE	90 valor	valor para cálculo relativo
0B8	OD valor	valor para dirección relativa	OFF	1B BCTR	salto calculado relativo
0B9	84 ADDI	suma inmediata de R0 y valor	100	87 valor	valor para cálculo
0BA	FF valor	valor para suma inmediata	101	E6 COMI	comparación inmediata entre R2 y valor
0BB	C8 STRR	contenido de R0 a dirección relativa calculada	102	03 valor	valor para comparación
0BC	F8 valor	valor para dirección relativa	103	98 BCPR	salto relativo calculado
0BD	44 ANDI	operación Y inmediata entre R0 y valor	104	05 valor	dirección para cálculo
0BE	03 valor	valor para operación Y inmediata	105	3B BSTR	dirección relativa calculada
0BF	98 BCPR	dirección de memoria relativa calculada entre R0 y valor	106	98 valor	valor para cálculo
0CO	04 valor	valor para dirección relativa de memoria	107	1F BCTA	direcciónamiento absoluto
0C1	04 LODI	carga inmediata de valores R0	108	01 ADR	valor superior de dirección
0C2	28 valor	valor para carga inmediata	109	63 ADR	valor inferior de dirección
0C3	C8 STRR	contenido de R0 a dirección relativa calculada	10A	E6 COMI	comparación inmediata entre R2 y valor
0C4	EB valor	valor para dirección relativa	10B	02 valor	valor para comparación
0C5	17 RETC	bifurcación condicional desde subprograma	10C	98 BCPR	dirección relativa calculada
0C6	OD LODA	valor de dirección absoluta a R1	10D	13 valor	valor para cálculo
0C7	1F ADR	valor superior de dirección	10E	3F BSTA	direcciónamiento absoluto
0C8	4C ADR	valor inferior de dirección	10F	00 ADR	valor superior de dirección
0C9	E5 COMI	comparación inmediata entre R1 y valor	110	8D ADR	valor inferior de dirección
0CA	D2 valor	valor para comparación inmediata	111	0C LODA	dirección absoluta a R0
0CB	19 BCTR	salto calculado relativo por R1 y valor	112	1F ADR	valor superior de dirección
0CC	13 valor	valor para salto relativo	113	61 ADR	valor inferior de dirección
0CD	6C IORA	O inclusiva entre dirección absoluta y R0	114	E4 COMI	comparación inmediata entre R0 y valor
0CE	1F ADR	valor superior de dirección	115	07 valor	valor para comparación
0CF	CA ADR	valor inferior de dirección	116	3D BSTA	direcciónamiento absoluto
ODO	C2 STRZ	almacenar R2 en R0	117	00 ADR	valor superior de dirección
OD1	46 ANDI	operación Y inmediata entre R2 y valor	118	98 ADR	valor inferior de dirección
OD2	01 valor	valor para operación Y inmediata	119	0C LODA	dirección absoluta a R0
OD3	18 BCTR	dirección de memoria relativa calculada entre R0 y valor	11A	1F ADR	valor superior de dirección
OD4	78 valor	valor para dirección relativa	11B	61 ADR	valor inferior de dirección
OD5	OD LODA	valor absoluto direccionado a R1	11C	E4 COMI	comparación inmediata entre R0 y valor

Dirección	Contenido	Comentario	Dirección	Contenido	Comentario
11D	0F valor	valor para comparación	192	8D valor	valor para cálculo
11E	3D BSTA	direcciónamiento absoluto	193	84 ADDI	valor para cálculo
11F	00 ADR	valor superior de dirección	193	84 ADDI	suma inmediata entre R0 y valor
120	9C ADR	valor inferior de dirección	194	01 valor	valor para la suma
121	0C LODA	dirección absoluta a R0	195	FF COMA	comparar con R3 posición de memoria absoluta
122	1F ADR	valor superior de dirección	196	61 ADR	valor superior de dirección
123	63 ADR	valor inferior de dirección	197	BC ADR	valor inferior de dirección
124	98 BCFR	salto calculado relativo	198	1A BCTR	dirección de memoria calculada relativa
125	E2 valor	valor para cálculo	199	05 valor	valor para cálculo
126	E6 COMI	comparación inmediata entre R2 y valor	19A	E4 COMI	Comparación inmediata entre R0 y valor
127	04 valor	valor para comparación	19B	0E valor	valor para comparación
128	98 BCFR	salto relativo calculado	19C	19 BCTR	dirección relativa de memoria calculada
129	04 valor	valor para cálculo	19D	01 valor	valor para cálculo
12A	3B BSTR	salto relativo calculado	19E	17 RETC	retorno desde subprograma
12B	3E valor	valor para cálculo	19F	CC STRA	cargar R0 en posición de memoria absoluta
12C	1B BCTR	salto relativo calculado	1AO	1F ADR	valor superior de dirección
12D	06 valor	valor para cálculo	1A1	5E ADR	valor inferior de dirección
12E	E6 COMI	comparación inmediata entre R2 y valor	1A2	17 RETC	retorno desde subprograma
12F	05 valor	valor para comparación	1A3	0D LODA	direcciónamiento absoluto, contenido a R1
130	98 BCFR	salto relativo calculado	1A4	17 ADR	valor superior de dirección
131	31 valor	valor para cálculo	1A5	66 ADR	valor inferior de dirección
132	3B BSTR	salto relativo calculado	1A6	36 RETE	retorno desde subprograma
133	E3 valor	valor para cálculo	1A7	09 LODR	dirección calculada relativa
134	0C LODA	direcciónamiento absoluto, contenido a R0	1A8	08 valor	valor para cálculo
135	1F ADR	valor superior de dirección	1A9	84 ADDI	suma inmediata entre R0 y valor
136	65 ADR	valor inferior de dirección	1AA	8D valor	valor para suma
137	44 ANDI	operación Y inmediata de R0 con valor	1AB	62 IORZ	operación O-inclusiva entre R0 y R2
138	04 valor	valor para operación Y	1AC	32 REDC	PORT C a R2
139	18 BCTR	dirección de memoria relativa calculada	1AD	CC STRA	cargar R0 en posición absoluta de memoria
13A	06 LODI	cargar valor en R2	1AE	1F ADR	valor superior de dirección
13C	00 valor	valor para R2	1AF	4 C ADR	valor inferior de dirección
13D	1B BCTR	dirección de memoria calculada relativa	1B0	17 RETC	retorno desde subprograma
13E	02 valor	valor para cálculo	1B1	0C LODA	direcciónamiento absoluto, contenido a R0
13F	06 LODI	cargar valor en R2	1B2	1F ADR	valor superior de dirección
140	04 valor	valor para R2	1B3	5C ADR	valor inferior de dirección
141	0E LODA	direcciónamiento absoluto, contenido a R2	1B4	44 ANDI	operación Y inmediata entre R0 y valor
142	7E ADR	valor superior de dirección	1B5	01 valor	valor para operación Y
143	88 ADR	valor inferior de dirección	1B6	D1 RRL	desplazar R0 un lugar a la izquierda
144	6E IORZ	direcciónamiento absoluto de memoria con O inclusiva	1B7	61 IORZ	O inclusiva entre R0 y R1
145	7E ADR	valor superior de dirección	1B8	17 RETC	retorno desde subprograma
146	8A ADR	valor inferior de dirección	1B9	02 valor	Valor de cálculo para trayectoria de la pelota
147	44 ANDI	operación Y inmediata de R0 con valor	1BA	03 valor	Valor de cálculo para trayectoria de la pelota
148	80 valor	valor para operación Y	1BB	00 valor	Valor de cálculo para trayectoria de la pelota
149	CC STRA	cargar R0 en posición de memoria absoluta	1BC	01 valor	Valor de cálculo para trayectoria de la pelota
14A	1F ADR	valor superior de dirección	1BD	06 valor	Valor de cálculo para trayectoria de la pelota
14B	69 ADR	valor inferior de dirección	1BE	07 valor	Valor de cálculo para trayectoria de la pelota
14C	0E LODA	posición de memoria absoluta a R2	1BF	04 valor	Valor de cálculo para trayectoria de la pelota
14D	7E ADR	valor superior de dirección	1CO	05 valor	Valor de cálculo para trayectoria de la pelota
14E	89 ADR	valor inferior de dirección	1C1	06 valor	Valor de cálculo para trayectoria de la pelota
14F	3A BSTR	salto calculado relativo	1C2	06 valor	Valor de cálculo para trayectoria de la pelota
150	CE valor	valor para cálculo	1C3	07 valor	Valor de cálculo para trayectoria de la pelota
151	08 LODR	posición de memoria calculada relativa a R0	1C4	08 valor	Valor de cálculo para trayectoria de la pelota
152	F7 valor	valor para cálculo	1C5	07 valor	Valor de cálculo para trayectoria de la pelota
153	EC COMA	direcciónamiento absoluto y comparación con R0	1C6	3F BSTA	dirección absoluta en CC0 o CC1
154	1F ADR	valor superior de dirección	1C6	02 ADR	valor superior de dirección
155	6A ADR	valor inferior de dirección	1C7	63 ADR	valor inferior de dirección
156	18 BCTR	direcciónamiento relativo calculado con R0	1C8	46 ANDI	operación Y entre R2 y valor
157	0B valor	valor para cálculo	1C9	02 valor	valor valor para operación
158	C8 STRR	dirección de memoria calculada relativa	1C6	A8 ADR	valor inferior de dirección
159	FA valor	valor superior de dirección	1C6	0D LODA	direcciónamiento absoluto, contenido a R1
15A	0D LODA	posición de memoria direccionalizada absoluta a R1	1C7	1F ADR	valor superior de dirección
15B	1F ADR	valor superior de dirección	1C8	65 ADR	valor inferior de dirección
15C	66 ADR	valor inferior de dirección	1C9	3F BSTA	salto absoluto de memoria
15D	0D LODA	direcciónamiento absoluto, contenido a R1	1C6	01 ADR	valor superior de dirección
15E	61 ADR	valor superior de dirección	1C6	B1 ADR	valor inferior de dirección
15F	B9 ADR	valor inferior de dirección	1C6	OC STRA	R0 a la posición de memoria absoluta
160	CC STRA	R0 posición absoluta de memoria	1C7	1F ADR	valor superior de dirección
161	1F ADR	valor superior de dirección	1C8	52 ADR	valor inferior de dirección
162	66 ADR	valor inferior de dirección	1C9	8D ADDA	sumar con R1 la posición de memoria absoluta direccio- nada
163	3F BSTA	salto absoluto de memoria	1C6	62 ADR	valor superior de dirección
164	02 ADR	valor superior de dirección	1C7	53 ADR	valor inferior de dirección

Dirección	Contenido	Comentario
172	C8 STRR	R0 a dirección calculada relativa
173	F9 valor	valor para cálculo
174	0F LODA	direcccionamiento absoluto, contenido a R3
175	1F ADR	valor superior de dirección
176	4E ADR	valor inferior de dirección
177	0F LODA	direcccionamiento absoluto, contenido a R3
178	67 ADR	valor superior de dirección
179	EF ADR	valor inferior de dirección
17A	44 ANDI	comparación inmediata entre R0 y valor
17B	3C valor	valor para comparación
17C	18 BCTR	dirección de memoria calculada relativa
17D	25 valor	valor para cálculo
17E	08 LODR	dirección de memoria calculada relativa
17F	A0 valor	valor para cálculo
180	88 ADDR	direcccionamiento calculado relativo con suma de R0
181	AC valor	valor para cálculo
182	E4 COMI	comparación inmediata entre R0 y valor
183	18 valor	valor para comparación
184	19 BCTR	salto de memoria calculado relativo
185	02 valor	valor para cálculo
186	04 LODI	carga inmediata de R0 con un valor
187	19 valor	valor para la carga
188	C8 STRR	R0 a dirección de memoria calculada relativa
189	A4 valor	valor para cálculo
18A	04 LODI	carga inmediata del valor en R0
18B	07 valor	valor para carga
18C	4C ANDA	operación Y entre dirección absoluta y R0
18D	1F ADR	valor superior de dirección
18E	5C ADR	valor inferior de dirección
18F	98 BCPR	dirección relativa calculada
190	11 valor	valor para cálculo
191	08 LODR	carga relativa de la dirección de memoria calculada desde R0
1CA	18 BCTR	direccion relativa obtenida en CCO a CC1
1CB	02 valor	valor para cálculo
1CC	06 LODI	carga inmediata de valor en R2
1CD	10 valor	valor para carga
1CE	0F LODA	valor de dirección absoluta a R3
1CF	67 ADR	valor superior de dirección
1D0	C5 ADR	valor inferior de dirección
1D1	E4 COMI	comparación inmediata entre R0 y valor
1D2	14 valor	valor para comparación
1D3	18 BCTR	dirección relativa evaluada en CCO o CC1
1D4	2D valor	valor para cálculo
1D5	0E LODA	valor de dirección absoluta a R2
1D6	7F ADR	valor superior de dirección
1D7	0C ADR	valor inferior de dirección
1D8	A8 SUBR	restar de R0 dirección relativa calculada
1D9	D4 valor	valor para cálculo
1DA	0C LODA	valor para dirección absoluta a R2
1DB	62 ADR	valor superior de dirección
1DC	42 ADR	valor inferior de dirección
1DD	C8 STRR	almacenar R0 en dirección relativa calculada
1DE	88 valor	valor para cálculo
1DF	0F LODA	valor de dirección absoluta a R3
1E0	67 IORI	O inclusiva inmediato entre R3 y valor
1E1	EF valor	valor para O inclusiva
1E2	44 ANDI	operación Y inmediata entre R0 y valor
1E3	3C valor	valor para operación Y
1E4	18 BCTR	dirección calculada relativa en CCO y CC1
1E5	0C valor	valor para cálculo
1E6	0D LODA	valor de dirección absoluta a R1
1E7	1F ADR	valor superior de dirección
1E8	66 ADR	valor inferior de dirección
1E9	D1 RRL	desplazar Bl un lugar a la izquierda
1EA	08 LODR	R0 a la posición relativa calculada
1EB	84 valor	valor de cálculo
1EC	8D ADDA	sumar R3 con la posición de memoria absoluta
1ED	62 ADR	valor superior de dirección
1EE	32 ADR	valor inferior de dirección
1EF	CC STRA	cargar R0 con posición absoluta de memoria

Dirección	Contenido	Comentario
1FO	1F ADR	valor superior de dirección
1F1	5E ADR	valor inferior de dirección
1F2	0D LODA	valor de dirección absoluta a R1
1F3	1F ADR	valor superior de dirección
1F4	65 ADR	valor inferior de dirección
1F5	02 LODZ	valor de R2 a R0
1F6	18 BCTR	posición relativa calculada mediante R0
1F7	06 valor	valor para el cálculo
1F8	45 ANDI	operación Y entre R1 y dirección calculada
1F9	04 valor	valor para cálculo
1FA	98 BCPR	dirección relativa calculada por R0
1FB	2E valor	valor para cálculo
1FC	17 RETC	retorno desde subprograma
1FD	45 ANDI	operación Y entre R1 y dirección calculada
1FE	04 valor	valor para cálculo
1FF	18 BCTR	posición de memoria calculada relativa por R0
200	29 valor	valor para cálculo
201	17 RETC	retorno desde subprograma
202	0E LODA	valor de dirección absoluta a R2
203	7F ADR	valor superior de dirección
204	0A ADR	valor inferior de dirección
205	AC SUBA	restar de la dirección absoluta el valor de R0
206	1F ADR	valor superior de dirección
207	4A ADR	valor inferior de dirección
208	84 ADDI	sumar R0 con valor inmediato
209	01 valor	valor para la suma
20A	44 ANDI	operación Y entre R0 y valor inmediato
20B	0F valor	valor de operación
20C	0C LODA	dirección absoluta a R0
20D	62 ADR	valor superior de dirección
20E	42 ADR	valor inferior de dirección
20F	CC STRA	R0 a posición absoluta de memoria
210	1F ADR	valor superior de dirección
211	66 ADR	valor inferior de dirección
212	08 LODR	dirección relativa calculada
213	D3 valor	valor para cálculo
214	24 EORI	O exclusiva inmediata entre R0 y valor
215	07 valor	valor para operación O
216	C8 STRR	R0 a dirección relativa calculada
217	CF valor	valor para cálculo
218	0C LODA	dirección absoluta a R0
219	1F ADR	valor superior de dirección
21A	4E ADR	valor inferior de dirección
21B	0F LODA	dirección absoluta a R3
21C	67 ADR	valor superior de dirección
21D	EF ADR	valor inferior de dirección
21E	44 ANDI	operación Y entre R0 y valor inmediato
21F	3C valor	valor para operación Y
220	14 RETC	retorno desde subprograma
221	08 LODR	dirección calculada relativa
222	CD valor	valor para cálculo
223	24 EORI	O exclusiva entre R0 y valor inmediato
224	FF valor	valor para comparación
225	84 ADDI	sumar R0 a valor inmediato
226	01 valor	valor para suma
227	C8 STRR	R0 a dirección relativa calculada
228	C7 valor	valor para cálculo
229	17 RETC	retorno desde subprograma
230	0C LODA	dirección absoluta a R0
231	1F ADR	valor superior de dirección
232	65 ADR	valor inferior de dirección
233	24 EORI	O exclusiva entre R0 y valor inmediato
234	04 valor	valor para conexión
235	C8 STRR	R0 a dirección calculada relativa
236	FA valor	valor para cálculo
237	17 RETC	retorno desde subprograma
238	0C LODA	dirección absoluta a R0
239	1F ADR	valor superior de dirección
240	65 ADR	valor inferior de dirección
241	24 EORI	O exclusiva entre R0 y valor inmediato
242	FF valor	valor para cálculo de colisiones
243	FE valor	valor para cálculo de colisiones
244	FF valor	valor para cálculo de colisiones
245	FE valor	valor para cálculo de colisiones
246	FE valor	valor para cálculo de colisiones

Dirección	Contenido	Comentario	Dirección	Contenido	Comentario
237	FE valor	valor para cálculo de colisiones	27E	1A BCTR	dirección relativa calculada
238	00 valor	valor para cálculo de colisiones	27F	04 valor	valor para cálculo
239	00 valor	valor para cálculo de colisiones	280	E5 COMI	comparar R1 y valor inmediato
23A	00 valor	valor para cálculo de colisiones	281	B3 valor	valor para comparación
23B	00 valor	valor para cálculo de colisiones	282	99 BCFR	dirección relativa calculada
23C	01 valor	valor para cálculo de colisiones	283	10 valor	valor para cálculo
23D	01 valor	valor para cálculo de colisiones	284	CD STRA	almacenar R1 en dirección absoluta
23E	02 valor	valor para cálculo de colisiones	285	1F ADR	valor superior de dirección
23F	01 valor	valor para cálculo de colisiones	286	63 ADR	valor inferior de dirección
240	02 valor	valor para cálculo de colisiones	287	CD STRA	almacenar R1 en dirección absoluta
241	02 valor	valor para cálculo de colisiones	288	1F ADR	valor superior de dirección
242	07 valor	valor para cálculo de colisiones	289	64 ADR	valor inferior de dirección
243	07 valor	valor para cálculo de colisiones	28A	OF LODA	dirección absoluta a R3
244	07 valor	valor para cálculo de colisiones	28B	65 ADR	valor superior de dirección
245	06 valor	valor para cálculo de colisiones	28C	FF ADR	valor inferior de dirección
246	06 valor	valor para cálculo de colisiones	28D	44 ANDI	operación Y entre R0 y valor inmediato
247	06 valor	valor para cálculo de colisiones	28E	80 valor	valor para operación Y
248	05 valor	valor para cálculo de colisiones	28F	1C BCTA	saltar a dirección absoluta si se satisfacen CCO y CCI
249	05 valor	valor para cálculo de colisiones	290	02 ADR	valor superior de dirección
24A	04 valor	valor para cálculo de colisiones	291	12 ADR	valor inferior de dirección
24B	03 valor	valor para cálculo de colisiones	292	3B BSTR	dirección relativa calculada
24C	02 valor	valor para cálculo de colisiones	293	FC valor	valor para cálculo
24D	02 valor	valor para cálculo de colisiones	294	OD LODA	posición absoluta a R1
24E	02 valor	valor para cálculo de colisiones	295	1F ADR	valor superior de dirección
24F	01 valor	valor para cálculo de colisiones	296	4A ADR	valor inferior de dirección
250	01 valor	valor para cálculo de colisiones	297	E5 COMI	comparar R1 y valor inmediato
251	00 valor	valor para cálculo de colisiones	298	1B valor	valor para comparación
252	00 valor	valor para cálculo de colisiones	299	1A BCTR	dirección relativa calculada
253	00 valor	valor para cálculo de colisiones	29A	08 valor	valor para cálculo
254	00 valor	valor para cálculo de colisiones	29B	E5 COMI	comparar R1 y valor inmediato
255	FF valor	valor para cálculo de colisiones	29C	87 valor	valor para comparación
256	FF valor	valor para cálculo de colisiones	29D	19 BCTR	dirección calculada relativa
257	FE valor	valor para cálculo de colisiones	29E	01 valor	valor para cálculo
258	FF valor	valor para cálculo de colisiones	29F	17 RETC	retorno desde subprograma
259	FE valor	valor para cálculo de colisiones	2A0	E7 COMI	comparar R3 y valor inmediato
25A	FE valor	valor para cálculo de colisiones	2A1	09 valor	valor para comparación
25B	00 valor	valor para cálculo de colisiones	2A2	14 RETC	retorno desde subprograma
25C	CO valor	valor para cálculo de colisiones	2A3	C9 STRR	almacenar R1 en dirección relativa calculada
25D	01 valor	valor para cálculo de colisiones	2A4	EO valor	valor para cálculo
25E	01 valor	valor para cálculo de colisiones	2A5	1F ECTA	salir a dirección absoluta si se satisfacen CCO y CCI
25F	02 valor	valor para cálculo de colisiones	2A6	02 ADR	valor superior de dirección
260	01 valor	valor para cálculo de colisiones	2A7	2A ADR	valor inferior de dirección
261	02 valor	valor para cálculo de colisiones	2A8	OF LODA	cargar posición absoluta en R3
262	02 valor	valor para cálculo de colisiones	2A9	65 ADR	valor superior de dirección
263	OC LODA	posición absoluta a R0	2AA	FF ADR	valor inferior de dirección
264	1F ADR	valor superior de dirección	2AB	C1 STRZ	almacenar R0 en R1
265	61 ADR	valor inferior de dirección	2AC	44 ANDI	operación Y entre R0 y valor inmediato
266	84 ADDI	suma de R0 y valor inmediato	2AD	01 valor	valor para operación
267	01 valor	valor para suma	2AE	B8 BSFR	dirección calculada relativa
268	C8 STRR	contenido de R0 a posición relativa calculada	2AF	1F valor	valor para cálculo
269	FA valor	valor para cálculo	2B0	OC LODA	posición absoluta a R0
26A	44 ANDI	operación Y entre R0 y valor inmediato	2B1	1F ADR	valor superior de dirección
26B	OF valor	valor para operación	2B2	53 ADR	valor inferior de dirección
26C	84 ADDI	suma de R0 y valor inmediato	2B3	44 ANDI	operación Y entre R0 y valor inmediato
26D	01 valor	valor para suma	2B4	10 valor	valor para operación
26E	CC STRA	R0 a posición absoluta	2B5	18 BCTR	dirección relativa calculada
26F	1F ADR	valor superior de dirección	2B6	OF valor	valor para cálculo
270	C7 ADR	valor inferior de dirección	2B7	OF LODA	posición absoluta a R3
271	04 LODI	cargar R0 con valor inmediato	2B8	65 ADR	valor superior de dirección
272	05 valor	valor para R0	2B9	FF ADR	valor inferior de dirección
273	CC STRA	R0 a posición absoluta	2BA	44 ANDI	operación Y entre R0 y valor inmediato
274	1F ADR	valor superior de dirección	2BB	02 valor	valor para operación
275	OF ADR	valor inferior de dirección	2BC	B8 BSFR	dirección relativa calculada si se satisfacen CCO y CCI
276	17 RETC	retorno desde subprograma	2BD	1F valor	valor para cálculo
277	3B BSTR	dirección relativa calculada	2BE	OF LODA	cargar dirección absoluta en R3
278	6A valor	valor para cálculo	2BF	65 ADR	valor superior de dirección
279	OD LODA	posición absoluta a R1	2C0	FF ADR	valor inferior de dirección
27A	1F ADR	valor superior de dirección	2C1	44 ANDI	operación Y entre R0 y valor inmediato
27B	4C ADR	valor inferior de dirección	2C2	04 valor	valor para operación Y
27C	E5 COMI	comparar R1 y valor inmediato	2C3	BC BSFA	saltar a dirección absoluta si se satisfacen CCO y CCI
27D	19 valor	valor para comparación	2C4	03 ADR	valor superior de dirección

Dirección	Contenido	Comentario
2C5	78 ADR	valor inferior de dirección
2C6	0F LODA	cargar posición de direccionamiento absoluta en R3
2C7	65 ADR	valor superior de dirección
2C8	FF ADR	valor inferior de dirección
2C9	44 ANDI	operación Y entre R0 y valor inmediato
2CA	10 valor	valor para operación
2CB	BC BSFA	saltar a dirección absoluta si se satisfacen CCO y CC1
2CC	03 ADR	valor superior de dirección
2CD	E1 ADR	valor inferior de dirección
2CE	17 RETC	retorno desde subprograma
2CF	OC LODA	cargar posición absoluta en R0
2D0	1F ADR	valor superior de dirección
2D1	4A ADR	valor inferior de dirección
2D2	E4 COMI	comparar R0 con valor inmediato
2D3	10 valor	valor para comparación
2D4	1E BCTA	saltar a posición absoluta si se satisfacen CCO y CC1
2D5	04 ADR	valor superior de dirección
2D6	08 ADR	valor inferior de dirección
2D7	E4 COMI	comparar R0 y valor inmediato
2D8	9A valor	valor para comparación
2D9	1D BCTA	saltar a posición absoluta si se satisfacen CCO y CC1
2DA	04 ADR	valor superior de dirección
2DB	05 ADR	valor inferior de dirección
2DC	17 RETC	retorno desde subprograma
2DD	08 LODR	salto de memoria relativo calculado
2DE	F1 valor	valor para cálculo
2DF	OE LODA	posición absoluta a R2
2E0	1F ADR	valor superior de dirección
2E1	63 ADR	valor inferior de dirección
2E2	18 BCTR	dirección relativa calculada
2E3	0F valor	valor para cálculo
2E4	E7 COMI	comparar R0 y valor inmediato
2E5	06 valor	valor para comparación
2E6	18 BCTR	dirección relativa calculada
2E7	05 valor	valor para cálculo
2E8	E7 COMI	comparar R3 y valor inmediato
2E9	06 valor	valor para comparación
2EA	18 BCTR	salto relativo calculado
2EB	29 valor	valor para cálculo
2EC	17 RETC	salto desde subprograma
2ED	OE LODA	carga posición absoluta en R2
2EE	1F ADR	valor superior de dirección
2EF	64 ADR	valor inferior de dirección
2FO	98 BCFR	salto relativo calculado
2F1	11 valor	valor para cálculo
2F2	17 RETC	salto desde subprograma
2F3	E7 COMI	comparar R3 y valor inmediato
2F4	07 valor	valor para comparación
2F5	18 BCTR	salto relativo calculado
2F6	04 valor	valor para cálculo
2F7	E7 COMI	comparar R3 y valor inmediato
2F8	08 valor	valor para comparación
2F9	98 BCFR	salto relativo calculado
2FA	10 valor	valor para cálculo
2FB	OE LODA	cargar posición absoluta en R2
2FC	1F ADR	valor superior de dirección
2FD	5E ADR	valor inferior de dirección
2FE	14 RETC	retorno desde subprograma
2FF	E6 COMI	comparar R2 y valor inmediato
300	F3 valor	valor para comparación
301	19 BCTR	salto relativo calculado
302	1D valor	valor para cálculo
303	E4 COMI	comparar R0 y valor inmediato
304	51 valor	valor para comparación
305	1D BCTA	saltar a dirección absoluta si se satisfacen CCO y CC1
306	04 ADR	valor superior de dirección
307	05 ADR	valor inferior de dirección
308	1F BCTA	saltar a dirección absoluta si se satisfacen CCO y CC1
309	04 ADR	valor superior de dirección
30A	09 ADR	valor inferior de dirección
30B	OC LODA	cargar posición absoluta a R0

Dirección	Contenido	Comentario
30C	1F ADR	valor superior de dirección
30D	65 ADR	valor inferior de dirección
30E	44 ANDI	operación Y entre R0 y valor inmediato
30F	04 valor	valor para operación Y
310	1C BCTA	saltar a dirección absoluta si se satisfacen CCO y CC1
311	04 ADR	valor superior de dirección
312	06 ADR	valor inferior de dirección
313	1B BCTR	dirección relativa calculada
314	73 valor	valor para cálculo
315	E6 COMI	comparar R2 y valor inmediato
316	B3 valor	valor de comparación
317	99 BCFR	dirección relativa de memoria calculada si se satisfacen CCO y CC1
318	06 valor	valor para cálculo
319	44 ANDI	operación Y entre R0 y valor inmediato
31A	FC valor	valor para operación
31B	E4 COMI	comparar R0 y valor inmediato
31C	50 valor	valor para comparación
31D	18 BCTR	posición relativa calculada si se satisfacen CCO y CC1
31E	6C valor	valor para cálculo
31F	17 RETC	retorno desde subprograma
320	04 LODI	cargar R0 con valor inmediato
321	41 valor	valor para carga
322	CC STRA	valor absoluto direccionado a R0
323	1F ADR	valor superior de dirección
324	4C ADR	valor inferior de dirección
325	1F BCTA	saltar a posición absoluta si se satisfacen CCO y CC1
326	02 ADR	valor superior de dirección
327	12 ADR	valor inferior de dirección
328	16 RETC	retorno desde subprograma
329	2B EORR	O exclusiva entre R3 y dirección calculada
32A	3F valor	valor para cálculo
32B	53 RRR	desplazar R3 un lugar a la derecha
32C	67 IORI	O exclusiva entre R3 y valor inmediato
32D	7B valor	valor para operación
32E	8F valor	Borrado de obstáculos y rebotes en juego de obstáculos
32F	A3 valor	Borrado de obstáculos y rebotes en juego de obstáculos
330	OC valor	Borrado de obstáculos y rebotes en juego de obstáculos
331	14 valor	Borrado de obstáculos y rebotes en juego de obstáculos
332	1C valor	Borrado de obstáculos y rebotes en juego de obstáculos
333	24 valor	Borrado de obstáculos y rebotes en juego de obstáculos
334	2C valor	Borrado de obstáculos y rebotes en juego de obstáculos
335	34 valor	Borrado de obstáculos y rebotes en juego de obstáculos
336	3C valor	Borrado de obstáculos y rebotes en juego de obstáculos
337	44 valor	Borrado de obstáculos y rebotes en juego de obstáculos
338	4C valor	Borrado de obstáculos y rebotes en juego de obstáculos
339	54 valor	Borrado de obstáculos y rebotes en juego de obstáculos
33A	5C valor	Borrado de obstáculos y rebotes en juego de obstáculos
33B	64 valor	Borrado de obstáculos y rebotes en juego de obstáculos
33C	6C valor	Borrado de obstáculos y rebotes en juego de obstáculos
33D	74 valor	Borrado de obstáculos y rebotes en juego de obstáculos
33E	7C valor	Borrado de obstáculos y rebotes en juego de obstáculos
33F	84 valor	Borrado de obstáculos y rebotes en juego de obstáculos
340	06 valor	Borrado de obstáculos y rebotes en juego de obstáculos
341	07 valor	Borrado de obstáculos y rebotes en juego de obstáculos
342	0A valor	Borrado de obstáculos y rebotes en juego de obstáculos
343	0B valor	Borrado de obstáculos y rebotes en juego de obstáculos
344	0E valor	Borrado de obstáculos y rebotes en juego de obstáculos
345	0F valor	Borrado de obstáculos y rebotes en juego de obstáculos
346	12 valor	Borrado de obstáculos y rebotes en juego de obstáculos
347	13 valor	Borrado de obstáculos y rebotes en juego de obstáculos
348	16 valor	Borrado de obstáculos y rebotes en juego de obstáculos
349	17 valor	Borrado de obstáculos y rebotes en juego de obstáculos
34A	1A valor	Borrado de obstáculos y rebotes en juego de obstáculos
34B	1B valor	Borrado de obstáculos y rebotes en juego de obstáculos
34C	1E valor	Borrado de obstáculos y rebotes en juego de obstáculos
34D	1F valor	Borrado de obstáculos y rebotes en juego de obstáculos
34E	22 valor	Borrado de obstáculos y rebotes en juego de obstáculos
34F	23 valor	Borrado de obstáculos y rebotes en juego de obstáculos
350	7F valor	Valor para rebotes en las demarcaciones del juego de obstáculos
351	BF valor	Valor para rebotes en las demarcaciones del juego de obstáculos

Dirección	Contenido	Comentario	Dirección	Contenido	Comentario
352	DF valor	Valor para rebotes en las demarcaciones del juego de obstáculos	399	03 LODZ	cargar R3 en R0
353	EF valor	Valor para rebotes en las demarcaciones del juego de obstáculos	39A	50 RRR	desplazar R0 un lugar a la derecha
354	F7 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	39B	50 RRR	desplazar R0 un lugar a la derecha
355	FB valor	Valor para rebotes en las demarcaciones del juego de obstáculos	39C	50 RRR	desplazar R0 un lugar a la derecha
356	FD valor	Valor para rebotes en las demarcaciones del juego de obstáculos	39D	44 ANDI	operación Y entre R0 y valor inmediato
357	FE valor	Valor para rebotes en las demarcaciones del juego de obstáculos	39E	01 valor	valor para operación
358	OF valor	Valor para rebotes en las demarcaciones del juego de obstáculos	39F	D2 RRL	desplazar R2 un lugar a la izquierda
359	1F valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3AO	62 IORZ	O inclusiva entre R0 y R2
360	4A valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A1	C2 STRZ	almacenar R0 en R2
361	05 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A2	47 ANDI	operación Y entre R3 y valor inmediato
362	10 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A3	07 valor	valor para operación
363	0D valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A4	E7 COMI	comparar R3 con valor inmediato
35E	63 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A5	02 valor	valor para comparación
35F	2F valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A6	1A BCTR	posición relativa calculada
360	84 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A7	32 valor	valor para cálculo
361	FE valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A8	0E LODA	cargar R2 con el valor de dirección absoluta
362	E3 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3A9	63 ADR	valor superior de dirección
363	18 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3AA	40 ADR	valor inferior de dirección
364	18 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3AB	C2 STRZ	R0 a R2
365	84 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3AC	0F LODA	cargar R3 con valor de posición absoluta
366	01 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3AD	63 ADR	valor de superior dirección
367	E3 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3AE	80 ADR	valor inferior de dirección
368	18 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3AF	CC STRA	valor de dirección absoluta a R0
369	13 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B0	1F ADR	valor superior de dirección
36A	84 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B1	68 ADR	valor inferior de dirección
36B	FE valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B2	C3 STRZ	R0 a R3
36C	E3 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B3	0E LODA	valor de dirección absoluta a R2
36D	18 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B4	7F ADR	valor superior de dirección
36E	OE valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B5	80 ADR	valor inferior de dirección
36F	E9 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B6	27 EORI	O exclusiva entre R5 y valor inmediato
370	6C valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B7	FT valor	valor para operación
371	17 valor	Valor para rebotes en las demarcaciones del juego de obstáculos	3B8	43 ANDZ	operación Y entre R0 y R3
372	OC LODA	dirección absoluta a R0	3B9	18 BCTR	dirección relativa calculada
373	1F ADR	valor superior de dirección	3BA	1C valor	valor para cálculo
374	63 ADR	valor inferior de dirección	3BB	07 LODI	cargar R3 con valor inmediato
375	18 BCTR	dirección relativa calculada	3BC	01 valor	valor para carga
376	06 valor	valor para cálculo	3BD	4F ANDA	operación Y entre R3 y dirección absoluta
377	06 LODA	posición absoluta a R0	3BE	1F ADR	valor superior de dirección
378	1F ADR	valor superior de dirección	3BF	4E ADR	valor inferior de dirección
379	64 ADR	valor inferior de dirección	3C0	BC BSFA	saltar a dirección absoluta si se satisfacen CCO y CC1
37A	98 BCFR	dirección relativa calculada si se satisfacen CCO y CC1	3C1	02 ADR	valor superior de dirección
37B	5C valor	valor para cálculo	3C2	2A ADR	valor inferior de dirección
37C	17 RETC	retorno desde subprograma	3C3	08 LODR	cargar R0 con dirección relativa calculada
37D	OC LODA	posición absoluta a R0	3C4	EB valor	valor para cálculo
37E	17 ADR	valor superior de dirección	3C5	4E ANDA	operación Y entre R2 y dirección absoluta
37F	4C ADR	valor inferior de dirección	3C6	7F ADR	valor superior de dirección
380	06 LODI	cargar R2 con valor inmediato	3C7	80 ADR	valor inferior de dirección
381	08 valor	valor para carga	3C8	EE COMA	comparar R2 y dirección absoluta
382	EE COMA	comparar R2 y dirección absoluta	3C9	7F ADR	valor superior de dirección
383	63 valor	valor superior de dirección	3CA	80 ADR	valor inferior de dirección
384	27 valor	valor inferior de dirección	3CB	18 BCTR	dirección relativa calculada
385	19 BCTR	posición relativa calculada	3CC	0A valor	valor para cálculo
386	02 valor	valor para cálculo	3CD	CE STRA	contenido de dirección absoluta a R2
387	FA BDRR	decrementar el contenido de R2 con dirección calculada	3CE	7F ADR	valor superior de dirección
388	79 valor	valor para cálculo	3CF	80 ADR	valor inferior de dirección
389	86 ADDI	sumar R2 y valor inmediato	3D0	OB LODR	cargar dirección relativa calculada en R3
38A	FF valor	valor para suma	3D1	EC valor	valor para cálculo
38B	OC LODA	posición absoluta a R0	3D2	E7 COMI	comparar R5 y valor inmediato
38C	1F ADR	valor superior de dirección	3D3	OB valor	valor para comparación
38D	4A ADR	valor inferior de dirección	3D4	9C BCFA	saltar a dirección absoluta si se satisfacen CCO y CC1
38E	07 LODI	cargar R3 con valor inmediato	3D5	04 ADR	valor superior de dirección
38F	10 valor	valor para carga	3D6	22 ADR	valor inferior de dirección
38G	EF COMA	comparar R3 con posición absoluta	3D7	OB LODR	cargar dirección relativa calculada en R3
391	63 ADR	valor superior de dirección	3D8	E5 valor	valor para cálculo
392	27 ADR	valor inferior de dirección	3D9	17 RETC	retorno desde subprograma
393	19 LODR	cargar dirección relativa calculada en R1	3DA	02 LODZ	R2 a R0
394	02 valor	valor para cálculo	3DB	44 ANDI	operación Y entre R0 y valor inmediato
395	FB BDRR	decrementar contenido de R3 con dirección calculada	3DC	01 valor	valor para operación
396	79 valor	valor para cálculo	3DD	98 BCFR	saltar a dirección relativa calculada
397	87 ADDI	sumar valor inmediato a R3	3DE	49 valor	valor para cálculo
398	FF valor	valor para suma	3DF	1B BCTR	saltar a dirección relativa calculada

Dirección	Contenido	Comentario	Dirección	Contenido	Comentario
3E0	76	Valor para cálculo	427	84	ADDI sumar RO y valor inmediato
3E1	OC LODA	valor de dirección absoluta a RO	428	67	valor para suma
3E2	1F ADR	valor superior de dirección	429	94	DAR corregir contenido de RO y convertir en decimal
3E3	4A ADR	valor inferior de dirección	43A	CD STRA cargar R1 en posición de memoria absoluta	
3E4	E4 COMI	comparar RO y valor inmediato	42B	7F ADR	valor superior de dirección
3E5	10 valor	valor para comparación	42C	5E ADR	valor inferior de dirección
3E6	1A BCTR	salto de dirección relativa calculada	42D	98 BCFR	salir a posición relativa calculada
3E7	05 valor	valor para cálculo	42E	02 valor	valor de la dirección de memoria
3E8	E4 COMI	comparación entre RO y valor inmediato	42F	F9 BDRR	disminuir R1 y saltar al valor de dirección calculado
3E9	9A valor	valor para comparación	430	73 valor	valor para cálculo
3EA	19 BCTR	salto a posición relativa calculada	431	05 LODI	cargar R1 con valor inmediato
3EB	01 valor	valor para cálculo	432	03 valor	valor para carga
3EC	17 RETC	retorno desde subprograma	433	C9 STRR	contenido de R1 a dirección relativa calculada
3ED	OC LODA	valor de dirección absoluta a RO	434	D7 valor	valor para cálculo
3EE	1F ADR	valor superior de dirección	435	17 RETC	retorno desde subprograma
3EF	SD ADR	valor inferior de dirección	436	CC STRA	almacenar RO en dirección absoluta
3FO	84 ADDI	sumar RO y valor inmediato	437	1F ADR	valor superior de dirección
3F1	01 valor	valor para suma	438	67 ADR	valor inferior de dirección
3F2	E4 COMI	comparar RO y valor inmediato	439	09 LODR	cargar dirección calculada en R1
3F3	06 valor	valor para comparación	43A	D1 valor	valor para cálculo
3F4	18 BCTR	salto a posición relativa calculada	43B	4D ANDA	operación Y entre R1 y valor de dirección absoluta
3F5	25 valor	valor para cálculo	43C	64 ADR	valor superior de dirección
3F6	C8 STRR	contenido de RO a dirección relativa calculada	43D	40 ADR	valor inferior de dirección
3F7	F6 valor	valor para cálculo	43E	CC STRA	almacenar RO en dirección absoluta
3F8	04 LODI	cargar RO con valor inmediato	43F	1F ADR	valor superior de dirección
3F9	03 valor	valor para carga	440	C7 ADR	valor inferior de dirección
3FA	C8 STRR	contenido de RO a dirección relativa calculada	441	36 RETE	salir a última dirección en «stack-pointer»
3FB	90 valor	valor para cálculo	442	13 SPSL	cargar RO con el registro inferior de estado de programa PSL
3FC	04 LODI	cargar RO con valor inmediato	443	3B BSTR	salir a dirección calculada si se satisfacen CCO y CC1
3FD	32 valor	valor para carga	444	0C valor	valor para cálculo
3FE	3B BSTR	salto a posición relativa calculada	445	0C LODA	cargar RO con valor de dirección absoluta
3FF	36 valor	valor para cálculo	446	1F ADR	valor superior de dirección
400	3F BSTA	salto a dirección absoluta si se satisfacen CCO y CC1	447	50 ADR	valor inferior de dirección
401	05 ADR	valor superior de dirección	448	14 RETC	retorno desde subprograma
402	65 ADR	valor inferior de dirección	449	08 LODR	cargar dirección calculada en RO
403	1B BCTR	salto a posición relativa calculada	44A	EC valor	valor para cálculo
404	19 valor	valor para cálculo	44B	F8 BDRR	reducir RO y saltar a la dirección calculada
405	05 LODI	cargar R1 con valor inmediato	44C	69 valor	valor para cálculo
406	01 valor	valor para carga	44D	17 RETC	retorno desde el subprograma
407	1B BCTR	salto a posición relativa calculada	44E	33 valor	valor para secuencias de sonido en caso de pelota fuera de campo
408	02 valor	valor para cálculo	44F	33 valor	valor para secuencias de sonido en caso de pelota fuera de campo
409	05 LODI	cargar R1 con valor inmediato	450	33 valor	valor para secuencias de sonido en caso de pelota fuera de campo
40A	02 valor	valor para carga	451	B4 TPSU	comprobar palabra superior de estado de programa
40B	CD STRA	almacenar R1 en dirección absoluta	452	80 ADDZ	sumar RO a RO, multiplicar por 2 el contenido de RO
40C	1F ADR	valor superior de dirección	453	18 BCTR	salir a posición calculada si se satisfacen CCO y CC1
40D	51 ADR	valor inferior de dirección	454	7C valor	valor para cálculo
40E	OD LODA	cargar valor absoluto de dirección en R1	455	17 RETC	retorno desde subprograma
40F	7F ADR	valor superior de dirección	456	B4 TPSU	comprobar palabra superior de estado de programa
410	5E ADR	valor inferior de dirección	457	80 ADDZ	sumar RO a RO, multiplicar por 2 el contenido de RO
411	84 ADDI	sumar RO con valor inmediato	458	1A BCTR	salir a posición calculada si se satisfacen CCO y CC1
412	67 valor	valor para suma	459	7C BSNA	salir a valor absoluto de dirección
413	94 DAR	corregir contenido de RO y convertir en decimal	460	05 ADR	valor superior de dirección
414	CD STRA	almacenar RO en dirección absoluta	461	C9 ADR	valor inferior de dirección
415	7F ADR	valor superior de dirección	462	CD STRA	almacenar R1 en dirección absoluta
416	5E ADR	valor inferior de dirección	463	1E ADR	valor superior de dirección
417	4E COMI	comparar RO con valor inmediato	464	80 ADR	valor inferior de dirección
418	15 valor	valor para comparación	465	OD LODA	cargar R1 con valor absoluto de dirección
419	98 BCFS	salto a posición calculada	466	7F ADR	valor superior de dirección
41A	61 valor	valor para cálculo	467	04 ADR	valor inferior de dirección
41B	3F BSTA	salto a dirección absoluta si se satisfacen CCO y CC1	468	CD STRA	almacenar R1 en dirección absoluta
41C	05 ADR	valor superior de dirección	469	52 ADR	valor superior de dirección
41D	21 ADR	valor inferior de dirección	470	F9 BDRR	reducir R1 y saltar a dirección calculada
41E	OF LODA	cargar R3 con contenido de posición de memoria absoluta	471	78 valor	valor para cálculo
41F	1F ADR	valor superior de dirección	472	06 LODI	cargar R2 con valor inmediato
420	4E ADR	valor inferior de dirección	473	02 valor	valor de carga
421	17 RETC	retorno desde subprograma	474	OC LODA	cargar RO con dirección absoluta
422	05 LODI	cargar R1 con valor inmediato	475	1F ADR	valor superior de dirección
423	02 valor	valor de carga			
424	OD LODA	cargar R1 con contenido de posición absoluta de memoria			
425	7F ADR	valor superior de dirección			
426	5E ADR	valor inferior de dirección			

Dirección	Contenido	Comentario	Dirección	Contenido	Comentario
46B	5C ADR	valor inferior de dirección	4B2	1F ADR	valor superior de dirección
46C	44 ANDI	operación Y entre R0 y valor inmediato	4B3	C1 ADR	valor inferior de dirección
46D	01 valor	valor para operación	4B4	2D BORA	O exclusiva entre R0 y dirección absoluta
46E	18 BCTR	saltar a dirección calculada	4B5	1F ADR	valor superior de dirección
46F	OC valor	valor para cálculo	4B6	6D ADR	valor inferior de dirección
470	OE LODA	cargar R2 con dirección absoluta	4B7	CD STRA	almacenar R1 en dirección absoluta
471	7F ADR	valor superior de dirección	4B8	1F ADR	valor superior de dirección
472	S4 ADR	valor inferior de dirección	4B9	C2 ADR	valor inferior de dirección
473	CE STRA	almacenar R2 en dirección absoluta	4BA	05 LODI	cargar R1 con valor inmediato
474	7F ADR	valor superior de dirección	4BB	32 valor	valor para carga
475	56 ADR	valor inferior de dirección	4BC	C9 STRR	almacenar R1 en dirección calculada
476	FA BDRR	reducir R2 y saltar al valor calculado	4BD	83 valor	valor para cálculo
477	78 valor	valor para cálculo	4BE	1B BCTR	salir a dirección calculada si se satisfacen CCO y CC1
478	76 PFSU	activar palabra superior de estado de programa a 1	4BF	13 valor	valor para cálculo
479	40 HALT	el 2650 se detiene y se repone a RESET/INTREQ	4C0	00 LODA	cargar R0 con dirección absoluta
47A	1B BCTR	salir a posición de memoria calculada	4C1	1F ADR	valor superior de dirección
47B	OA valor	valor para cálculo	4C2	5C ADR	valor inferior de dirección
47C	OC LODA	cargar R2 con valor absoluto de dirección	4C3	A4 SUBI	restar de R0 valor inmediato
47D	7E ADR	valor superior de dirección	4C4	01 valor	valor para resta
47E	54 ADR	valor inferior de dirección	4C5	C8 STRR	almacenar R0 en dirección calculada
47F	CE STRA	almacenar R2 en dirección absoluta	4C6	FA valor	valor para cálculo
480	7F ADR	valor superior de dirección	4C7	17 RETC	retorno desde subprograma
481	58 ADR	valor inferior de dirección	4C8	OA LODR	almacenar R2 en dirección calculada
482	FA BDRR	reducir R2 y saltar al valor calculado	4C9	87 valor	valor para cálculo
483	78 valor	valor para cálculo	4CA	3B BSTR	salir a dirección calculada si se satisfacen CCO y CC1
484	74 GPSU	borrar palabra superior de estado de programa si hay equivalencia	4CB	74 valor	valor para cálculo
485	40 HALT	el 2650 se detiene y se repone a RESET/INTREQ	4CC	98 BCFR	salir a dirección calculada si se satisfacen CCO y CC1
486	06 LODI	cargar R2 con valor inmediato	4CD	05 valor	valor para cálculo
487	02 valor	valor para carga	4CE	A6 SUBI	restar de R2 valor inmediato
488	OE LODA	cargar R2 con valor absoluto de dirección	4CF	01 valor	valor para resta
489	7F ADR	valor superior de dirección	4D0	CE STRA	almacenar R2 en dirección absoluta
49A	5E ADR	valor inferior de dirección	4D1	1F ADR	valor superior de dirección
49B	CE STRA	almacenar R2 en dirección absoluta	4D2	5B ADR	valor inferior de dirección
49C	7F ADR	valor superior de dirección	4D3	3F BSTA	salir a dirección absoluta si se satisfacen CCO y CC1
49D	C7 ADR	valor inferior de dirección	4D4	00 ADR	valor superior de dirección
49E	FA BDRR	reducir R2 y saltar al valor calculado	4D5	25 ADR	valor inferior de dirección
49F	78 valor	valor para cálculo	4D6	OC LODA	cargar R0 con dirección absoluta
49G	OE LODA	cargar R2 con valor de dirección absoluta	4D7	1F ADR	valor superior de dirección
49H	1F ADR	valor superior de dirección	4D8	52 ADR	valor inferior de dirección
49I	50 ADR	valor inferior de dirección	4D9	CC STRA	almacenar R0 en dirección absoluta
49J	98 BCFR	saltar a la dirección calculada	4DA	1F ADR	valor superior de dirección
49K	33 valor	valor para cálculo	4DB	4A ADR	valor inferior de dirección
49L	OA LODR	cargar R2 con valor inmediato	4DC	OC LODA	cargar R0 con dirección absoluta
49M	BA valor	valor para carga	4DD	1E ADR	valor superior de dirección
49N	98 BCFR	saltar a dirección calculada	4DE	8B ADR	valor inferior de dirección
49O	31 valor	valor para cálculo	4DF	3A BSTR	salir a dirección calculada si se satisfacen CCO y CC1
49P	3B BSTA	saltar a dirección calculada si se satisfacen CCO y CC1	4EO	1C valor	valor para cálculo
49Q	25 valor	valor para cálculo	4E1	OC LODA	cargar R0 con dirección absoluta
49R	98 BCFR	saltar a posición de memoria calculada	4E2	1E valor	valor para cálculo
49S	36 valor	valor para cálculo	4E3	8B ADDR	sumar R3 con valor inmediato
49T	OC LODA	cargar R0 con dirección absoluta	4E4	44 valor	valor para suma
49U	1F ADR	valor superior de dirección	4E5	40 HALT	el microprocesador se detiene y espera a RESET o INTREQ
49V	1E ADR	valor inferior de dirección	4E6	9C BCFA	salir a dirección absoluta si R0 es distinto de 1
4AO	24 EORI	O exclusiva entre R0 y valor inmediato	4E7	05 ADR	valor superior de dirección
4A1	FF valor	valor para operación	4E8	4A ADR	valor inferior de dirección
4A2	C8 STRR	almacenar R0 en dirección calculada	4E9	17 RETC	retorno desde subprograma
4A3	FA valor	valor para operación	4EA	20 BORZ	O exclusiva entre R0 y R0, R0 se borra
4A4	C1 STRZ	almacenar R0 en R1	4EB	09 LODR	cargar R1 en dirección calculada
4A5	2C EORA	O exclusiva entre R0 y dirección absoluta	4EC	8C valor	valor para cálculo
4A6	1F ADR	valor superior de dirección	4ED	85 ADDI	suma de R1 y valor inmediato
4A7	1F ADR	valor inferior de dirección	4EE	01 valor	valor para suma
4A8	64 IORI	O inclusiva entre R0 y valor inmediato	4EF	E5 COMI	comparación entre R1 y valor inmediato
4A9	08 valor	valor para comparación	4FO	0D valor	valor para comparación
4AB	CC STRA	almacenar R0 en dirección absoluta	4F1	98 BCFR	saltar a posición calculada
4AC	1F ADR	valor superior de dirección	4F2	05 valor	valor para cálculo
4AD	C6 ADR	valor inferior de dirección	4F3	20 EORZ	O exclusiva entre R0 y R0, R0 se borra
4AE	01 LODZ	cargar R1 en R0	4F4	08 LODI	cargar R1 con valor inmediato
4AF	2C EORA	O exclusiva entre R0 y dirección absoluta	4F5	01 valor	valor para carga
4B0	1F ADR	valor superior de dirección	4F6	C8 STRR	almacenar R0 en dirección calculada
4B1	6C ADR	valor inferior de dirección	4F7	91 valor	valor para cálculo
	CC STRA	almacenar R0 en dirección absoluta	4F8	CD STRA	almacenar R1 en dirección absoluta

Dirección	Contenido	Comentario
4F9	1F ADR	valor superior de dirección
4FA	4E ADR	valor inferior de dirección
4FB	1B BCTR	salir a dirección calculada si se satisfacen CCO y CC1
4FC	03 valor	valor para cálculo
4FD	OC LODA	cargar RO con valor absoluto de dirección
4FE	1F ADR	valor superior de dirección
4FF	4F ADR	valor inferior de dirección
500	84 ADDI	sumar RO y valor inmediato
501	01 valor	valor para suma
502	E4 COMI	comparar RO y valor inmediato
503	06 valor	valor para comparación
504	18 LODR	almacenar RO en posición de memoria calculada
505	64 valor	valor para cálculo
506	C8 STRR	almacenar RO en dirección calculada
507	F6 valor	valor para cálculo
508	OD LODA	cargar R1 con dirección absoluta
509	1F ADR	valor superior de dirección
50A	AD ADR	valor inferior de dirección
50B	85 ADDI	suma de R1 con valor inmediato
50C	67 valor	valor para suma
50D	95 DAR	corregir contenido de R1 y convertir a decimal
50E	C9 STRR	almacenar R1 en dirección calculada
50F	F9 valor	valor para cálculo
510	CD STRA	almacenar R1 en dirección absoluta
511	1F ADR	valor superior de dirección
512	60 ADR	valor inferior de dirección
513	CD STRA	almacenar R1 en dirección absoluta
514	1F ADR	valor superior de dirección
515	C9 ADR	valor inferior de dirección
516	20 EORZ	O exclusiva entre RO y RO, RO se borra
517	CC STRA	almacenar RO en dirección absoluta
518	1F ADR	valor superior de dirección
519	C5 ADR	valor inferior de dirección
51A	CC STRA	almacenar RO en dirección absoluta
51B	00 ADR	valor superior de dirección
51C	60 ADR	valor inferior de dirección
51D	A5 SUBI	restar de R1 un valor inmediato
51E	62 valor	valor para resta
51F	25 EORI	O exclusiva entre R1 y valor inmediato
520	FF valor	valor para operación
521	CD STRA	almacenar R1 en dirección absoluta
522	1F ADR	valor superior de dirección
523	OE ADR	valor inferior de dirección
524	04 LODI	cargar RO con valor inmediato
525	03 valor	valor de carga
526	CC STRA	almacenar RO en dirección absoluta
527	1F ADR	valor superior de dirección
528	5B ADR	valor inferior de dirección
529	3F BSTA	salir a dirección absoluta si se satisfacen CCO y CC1
52A	05 ADR	valor superior de dirección
52B	7D ADR	valor inferior de dirección
52C	04 LODI	cargar RO con valor inmediato
52D	05 valor	valor de carga
52E	3B BSTR	salir a dirección calculada si se satisfacen CCO y CC1
52F	0F valor	valor para cálculo
530	08 LODR	cargar RO con valor de memoria calculado
531	F0 valor	valor para cálculo
532	CC STRA	almacenar RO en dirección absoluta
533	1F ADR	valor superior de dirección
534	C7 ADR	valor inferior de dirección
535	04 LODI	cargar RO con valor inmediato
536	05 valor	valor para carga
537	36 RETE	salir a la dirección de memoria del «stack-pointer».
538	06 LODI	cargar R2 con valor inmediato
539	C8 valor	valor para carga
53A	9B ZBRR	salto incondicional a dirección relativa
53B	C8 valor	valor de la dirección relativa
53C	F6 TMI	comprobar si el valor inmediato coincide con R2
53D	04 valor	valor para comprobación
53E	01 LODZ	cargar R1 en RO
53F	B4 TPSU	comprobar la palabra superior de estado de programa

Dirección	Contenido	Comentario
540	80 ADDZ	sumar RO con RO, multiplicar por 2 el contenido
541	18 BCTR	salir a dirección calculada
542	7C valor	valor para cálculo
543	B4 TPSU	comprobar palabra superior de estado de programa
544	80 ADDZ	sumar RO con RO, contenido multiplicado por dos
545	1A BCTR	salir a dirección calculada
546	7C valor	valor para cálculo
547	F8 BDRR	reducir contenido de RO y saltar a dirección calculada
548	76 valor	valor para cálculo
549	17 RETC	retorno desde subprograma
54A	0F LODA	cargar R3 con dirección absoluta
54B	1F ADR	valor superior de dirección
54C	4E ADR	valor inferior de dirección
54D	OF LODA	cargar R3 con dirección absoluta
54E	67 ADR	valor superior de dirección
54F	EF ADR	valor inferior de dirección
550	C8 STRR	almacenar R1 en dirección calculada
551	C6 valor	valor para cálculo
552	20 EORZ	O exclusiva entre RO y RO, borrar RO
553	06 LODI	cargar R2 con valor inmediato
554	08 valor	valor de carga
555	CE STRA	almacenar R2 en dirección absoluta
556	1F ADR	valor superior de dirección
557	50 ADR	valor inferior de dirección
558	0E STRA	almacenar R2 en dirección absoluta
559	7F ADR	valor superior de dirección
55A	5C ADR	valor inferior de dirección
55B	FA BDRR	reducir R2 y saltar a dirección calculada
55C	7B valor	valor para cálculo
55D	OC LODA	cargar RO con dirección absoluta
55E	1F ADR	valor superior de dirección
55F	5C ADR	valor inferior de dirección
560	44 ANDI	operación Y entre RO y valor inmediato
561	07 valor	valor para operación
562	CC STRA	almacenar RO en dirección absoluta
563	1F ADR	valor superior de dirección
564	66 ADR	valor inferior de dirección
565	3F BSTA	saltar a dirección absoluta si se satisfacen CCO y CC1
566	00 ADR	valor superior de dirección
567	8D ADR	valor inferior de dirección
568	20 EORZ	O exclusiva entre RO y RO, RO se borra
569	CC STRA	almacenar RO en dirección absoluta
56A	1F ADR	valor superior de dirección
56B	5E ADR	valor inferior de dirección
56C	CC STRA	almacenar RO en dirección absoluta
56D	1F ADR	valor superior de dirección
56E	61 ADR	valor inferior de dirección
56F	3B BSTR	saltar a dirección absoluta si se satisfacen CCO y CC1
570	0C valor	valor para cálculo
571	08 LODR	cargar RO con dirección calculada
572	63 valor	valor para cálculo
573	14 RETC	retorno desde subprograma
574	04 LODI	cargar RO con valor inmediato
575	32 valor	valor para carga
576	CC STRA	almacenar RO en dirección absoluta
577	1F ADR	valor superior de dirección
578	62 ADR	valor inferior de dirección
579	17 RETC	retorno desde subprograma
57A	3F BSTA	saltar a dirección absoluta si se satisfacen CCO y CC1
57B	04 ADR	valor superior de dirección
57C	3F ADR	valor inferior de dirección
57D	OD LODA	cargar R1 con dirección absoluta
57E	1F ADR	valor superior de dirección
57F	4E ADR	valor inferior de dirección
580	OD LODA	cargar R1 con dirección absoluta
581	66 ADR	valor superior de dirección
582	47 ADR	valor inferior de dirección
583	CC STRA	almacenar RO en dirección absoluta
584	1F ADR	valor superior de dirección
585	1F ADR	valor inferior de dirección
586	CC STRA	almacenar RO en dirección absoluta

Dirección	Contenido	Comentario	Dirección	Contenido	Comentario
587	1F ADR	valor superior de dirección	5CE	BF ADR	valor inferior de dirección
588	C6 ADR	valor inferior de dirección	5CF	CF STRA	almacenar R3 en dirección absoluta
589	OD LODA	cargar dirección absoluta en R1	5D0	7F ADR	valor superior de dirección
58A	66 ADR	valor superior de dirección	5D1	47 ADR	valor inferior de dirección
58B	6E ADR	valor inferior de dirección	5D2	FB BDRR	reducir contenido de R3 y saltar a dirección calculada
58C	C2 STRZ	almacenar R0 en R2	5D3	78 ADR	valor para el cálculo
58D	07 LODI	cargar R3 con valor inmediato	5D4	E5 COMI	comparar R1 y valor inmediato
58E	29 valor	valor para carga	5D5	08 valor	valor para comparación
58F	E5 COMI	comparar R1 con valor inmediato	5D6	99 BDFR	saltar a dirección calculada si se satisfacen CCO y CCl
590	08 valor	valor para comparación	5D7	04 valor	valor para cálculo
591	19 BCTR	saltar a dirección calculada si se satisfacen CCO y CCl	5D8	20 EORZ	O exclusiva entre R0 y R0, R0 se borra
592	1E valor	valor para cálculo	5D9	CC STRA	almacenar R0 en dirección absoluta
593	OE LODA	cargar dirección absoluta en R2	5DA	1F ADR	valor superior de dirección
594	46 ADR	valor superior de dirección	5DB	49 ADR	valor inferior de dirección
595	78 ADR	valor inferior de dirección	5DC	04 LODI	cargar R0 con valor inmediato
596	CF STRA	almacenar R3 en dirección absoluta	5DD	FA valor	valor para carga
597	7F ADR	valor superior de dirección	5DE	CC STRA	almacenar R0 en dirección absoluta
598	83 ADR	valor inferior de dirección	5DF	1F ADR	valor superior de dirección
599	FB BDRR	reducir contenido de R3 y saltar a dirección calculada	5E0	OD ADR	valor inferior de dirección
59A	78 valor	valor para cálculo	5E1	CC STRA	almacenar R0 en dirección absoluta
59B	E5 COMI	comparar R1 y valor inmediato	5E2	1F ADR	valor superior de dirección
59C	05 valor	valor para comparación	5E3	1D ADR	valor inferior de dirección
59D	98 BCFF	saltar a dirección calculada si se satisfacen CCO y CCl	5E4	CC STRA	almacenar R0 en dirección absoluta
59E	1A valor	valor para cálculo	5E5	1F ADR	valor superior de dirección
59F	20 EORZ	O exclusiva entre R0 y R0, R0 se borra	5E6	2C ADR	valor inferior de dirección
5A0	07 LODI	cargar R3 con valor inmediato	5E7	04 LODI	cargar R0 con valor inmediato
5A1	26 valor	valor para carga	5E8	50 valor	valor para carga
5A2	CF STRA	almacenar R3 en dirección absoluta	5E9	CC STRA	almacenar R0 en dirección absoluta
5A3	7F ADR	valor superior de dirección	5EA	1F ADR	valor superior de dirección
5A4	7F ADR	valor inferior de dirección	5EB	52 ADR	valor inferior de dirección
5A5	FB BDRR	reducir contenido de R3 y saltar a dirección calculada	5EC	04 LODI	cargar R0 con valor inmediato
5A6	78 valor	valor para cálculo	5ED	65 valor	valor para carga
5A7	04 LODI	cargar R0 con valor inmediato	5EE	CC STRA	almacenar R0 en dirección absoluta
5A8	80 valor	valor de carga	5EF	1F ADR	valor superior de dirección
5A9	CC STRA	almacenar R0 en dirección absoluta	5FO	CO ADR	valor inferior de dirección
5AA	1F ADR	valor superior de dirección	5FL	06 LODI	cargar R2 con valor inmediato
5AB	A3 ADR	valor inferior de dirección	5F2	02 valor	valor para carga
5AC	CC STRA	almacenar R0 en dirección absoluta	5F3	D1 RRL	desplazar R1 en lugar a la izquierda
5AD	1F ADR	valor superior de dirección	5F4	OD LODA	cargar dirección absoluta en R1
5AE	9F ADR	valor inferior de dirección	5F5	46 ADR	valor superior de dirección
5AF	1B BCTR	saltar a dirección calculada si se satisfacen CCO y CCl	5F6	54 ADR	valor inferior de dirección
5B0	08 valor	valor para cálculo	5F7	CE STRA	cargar R2 en dirección absoluta
5B1	OE LODA	cargar dirección absoluta en R2	5F8	5F ADR	valor superior de dirección
5B2	47 ADR	valor superior de dirección	5F9	C1 ADR	valor inferior de dirección
5B3	1C ADR	valor inferior de dirección	5FA	CE STRA	cargar R2 en dirección absoluta
5B4	CF STRA	almacenar R3 en dirección absoluta	5FB	7F ADR	valor superior de dirección
5B5	7F ADR	valor superior de dirección	5FC	6C ADR	valor inferior de dirección
5B6	83 ADR	valor inferior de dirección	5FD	SA BRNR	saltar a dirección calculada
5B7	FB BDRR	reducir contenido de R3 y saltar a dirección calculada	5FE	76 valor	valor para cálculo
5B8	78 valor	valor para cálculo	5FF	17 RETC	retorno desde subprograma
5B9	8D ADDA	sumar R1 y dirección absoluta	600	01 valor	valor para determinar demarcación de campo en juego 1
5BA	87 ADR	valor superior de dirección	601	01 valor	valor para determinar demarcación de campo en juego 2
5BB	C5 ADR	valor inferior de dirección	602	81 valor	valor para determinar demarcación de campo en juego 3
5BC	C2 ATRZ	almacenar R0 en R2	603	81 valor	valor para determinar demarcación de campo en juego 4
5BD	07 LODI	cargar R3 con valor inmediato	604	03 valor	valor para determinar demarcación de campo en juego 5
5BE	0A valor	valor para carga	605	82 valor	valor para determinar demarcación de campo en juego 6
5BF	OE LODA	cargar dirección absoluta en R2	606	82 valor	valor para determinar demarcación de campo en juego 7
5C0	47 ADR	valor superior de dirección	607	82 valor	valor para determinar demarcación de campo en juego 8
5C1	D2 ADR	valor inferior de dirección	608	9C valor	valor para determinar demarcación de campo en juego 9
5C2	CF STRA	almacenar R3 en dirección absoluta	609	9C valor	valor para determinar demarcación de campo en juego 10
5C3	7E ADR	valor superior de dirección	60A	05 valor	valor para determinar demarcación de campo en juego 11
5C4	FF ADR	valor inferior de dirección	60B	1C valor	valor para determinar demarcación de campo en juego 12
5C5	CF STRA	almacenar R3 en dirección absoluta	60C	18 valor	valor para determinar posición base de jugador en juegos 1 y 3
5C6	7F ADR	valor superior de dirección	60D	06 valor	valor para determinar posición base de jugador en juegos 1 y 3
5C7	OF ADR	valor inferior de dirección	60E	53 valor	valor para determinar posición base de jugador en juegos 1 y 3
5C8	FB BDRR	reducir contenido de R3 y saltar a dirección calculada	60F	8F valor	valor para determinar posición base de jugador en juegos 1 y 3
5C9	75 valor	valor para cálculo	610	21 valor	valor para determinar posición base de jugador en juegos 1 y 3
5CA	07 LODI	cargar R3 con valor inmediato	611	5E valor	valor para determinar posición base de jugador en juegos 1 y 3
5CB	06 Valor	valor para carga	612	18 valor	valor para determinar posición base de jugador en juegos 2 y 4
5CC	OF LODA	cargar dirección absoluta en R3	613	06 valor	valor para determinar posición base de jugador en juegos 2 y 4
5CD	67 ADR	valor superior de dirección	614	4E valor	valor para determinar posición base de jugador en juegos 2 y 4

Dirección	Contenido	Comentario	Dirección	Contenido	Comentario
615	8A valor	valor para determinar posición base de jugador en juegos 2 y 4	65C	12 valor	jugador izdo.: violeta, jug. dcho.: violeta 5
616	25 valor	valor para determinar posición base de jugador en juegos 2 y 4	65D	07 valor	pelota: negro 8
617	61 valor	valor para determinar posición base de jugador en juegos 2 y 4	65E	09 valor	jugador izdo.: amarillo, jug. dcho. amarillo 6
618	80 valor	valor para determinar posición base de jugador en juegos 6	65F	00 valor	pelota: blanco
619	80 valor	valor para determinar posición base de jugador en juegos 6	660	08 valor	jugador izdo.: amarillo, jug. dcho.: blanco 7
61A	50 valor	valor para determinar posición base de jugador en juegos 6	661	05 valor	pelota: verde 7
61B	90 valor	valor para determinar posición base de jugador en juegos 6	662	38 valor	jugador izdo.: negro, jug. dcho.: blanco 8
61C	20 valor	valor para determinar posición base de jugador en juegos 6	663	01 valor	pelota: amarillo 8
61D	5F valor	valor para determinar posición base de jugador en juegos 6	664	01 valor	no hay jugador izdo., jug. dcho.: amarillo 9
61E	30 valor	valor para determinar posición base de jugador en juegos 5	665	01 valor	pelota: amarillo 9
61F	C0 valor	valor para determinar posición base de jugador en juegos 5	666	00 valor	no hay jugador izdo., jug. dcho.: negro 10
620	50 valor	valor para determinar posición base de jugador en juegos 5	667	03 valor	pelota: rojo 10
621	94 valor	valor para determinar posición base de jugador en juegos 5	668	12 valor	jugador izdo.: violeta, jug. dcho.: violeta 11
622	1C valor	valor para determinar posición base de jugador en juegos 5	669	00 valor	pelota: blanco 11
623	60 valor	valor para determinar posición base de jugador en juegos 5	670	1D valor	jugador izdo.: rojo, jug. dcho.: verde 12
624	48 valor	valor para determinar posición base de jugador en juegos 7 y 8	66B	05 valor	pelota: verde 12
625	B4 valor	valor para determinar posición base de jugador en juegos 7 y 8	66C	29 valor	valor de corrección para campo de juego y colisiones juego 1
626	8F valor	valor para determinar posición base de jugador en juegos 7 y 8	66D	29 valor	valor de corrección para campo de juego y colisiones juego 2
627	8F valor	valor para determinar posición base de jugador en juegos 7 y 8	66E	52 valor	valor de corrección para campo de juego y colisiones juego 3
628	22 valor	valor para determinar posición base de jugador en juegos 7 y 8	66F	52 valor	valor de corrección para campo de juego y colisiones juego 4
629	22 valor	valor para determinar posición base de jugador en juegos 7 y 8	670	7B valor	valor de corrección para campo de juego y colisiones juego 5
62A	21 valor	valor para determinar posición base de jugador en juegos 9 y 10	671	7B valor	valor de corrección para campo de juego y colisiones juego 6
62B	BE valor	valor para determinar posición base de jugador en juegos 9 y 10	672	A4 valor	valor de corrección para campo de juego y colisiones juego 7
62C	E2 valor	valor para determinar posición base de jugador en juegos 9 y 10	673	A4 valor	valor de corrección para campo de juego y colisiones juego 8
62D	8F valor	valor para determinar posición base de jugador en juegos 9 y 10	674	29 valor	valor de corrección para campo de juego y colisiones juego 9
62E	E2 valor	valor para determinar posición base de jugador en juegos 9 y 10	675	52 valor	valor de corrección para campo de juego y colisiones juego 10
62F	8F valor	valor para determinar posición base de jugador en juegos 9 y 10	676	7B valor	valor de corrección para campo de juego y colisiones juego 11
630	18 valor	valor para determinar posición base de jugador en juegos 11	677	A4 valor	valor de corrección para campo de juego y colisiones juego 12
631	C4 valor	valor para determinar posición base de jugador en juegos 11	678	FF valor	valor para campo de juego 1 y 2 (izda. fina)
632	48 valor	valor para determinar posición base de jugador en juegos 11	679	FF valor	valor para campo de juego 1 y 2 (dcha. fina)
633	68 valor	valor para determinar posición base de jugador en juegos 11	67A	AA valor	valor para campo de juego 1 y 2 (izda. ancha)
634	48 valor	valor para determinar posición base de jugador en juegos 11	67B	55 valor	valor para campo de juego 1 y 2 (dcha. ancha)
635	68 valor	valor para determinar posición base de jugador en juegos 11	67C	FF valor	valor para campo de juego 1 y 2 (izda. fina)
636	18 valor	valor para determinar posición base de jugador en juegos 12	67D	FF valor	valor para campo de juego 1 y 2 (dcha. fina)
637	C4 valor	valor para determinar posición base de jugador en juegos 12	67E	00 valor	valor para campo de juego 1 y 2 (izda. ancha)
638	21 valor	valor para determinar posición base de jugador en juegos 12	67F	80 valor	valor para campo de juego 1 y 2 (dcha. ancha)
639	8F valor	valor para determinar posición base de jugador en juegos 12	680	00 valor	valor para campo de juego 1 y 2 (izda. fina)
63A	21 valor	valor para determinar posición base de jugador en juegos 12	681	80 valor	valor para campo de juego 1 y 2 (dcha. fina)
63B	8F valor	valor para determinar posición base de jugador en juegos 12	682	00 valor	valor para campo de juego 1 y 2 (izda. ancha)
63C	00 valor	valor para coordenadas de corrección en juego 1	683	80 valor	valor para campo de juego 1 y 2 (dcha. ancha)
63D	06 valor	valor para coordenadas de corrección en juego 2	684	00 valor	valor para campo de juego 1 y 2 (izda. fina)
63E	00 valor	valor para coordenadas de corrección en juego 3	685	80 valor	valor para campo de juego 1 y 2 (dcha. fina)
63F	06 valor	valor para coordenadas de corrección en juego 4	686	00 valor	valor para campo de juego 1 y 2 (izda. ancha)
640	12 valor	valor para coordenadas de corrección en juego 5	687	80 valor	valor para campo de juego 1 y 2 (dcha. ancha)
641	OC valor	valor para coordenadas de corrección en juego 6	688	00 valor	valor para campo de juego 1 y 2 (izda. fina)
642	18 valor	valor para coordenadas de corrección en juego 7	689	80 valor	valor para campo de juego 1 y 2 (dcha. fina)
643	18 valor	valor para coordenadas de corrección en juego 8	68A	00 valor	valor para campo de juego 1 y 2 (izda. ancha)
644	1E valor	valor para coordenadas de corrección en juego 9	68B	80 valor	valor para campo de juego 1 y 2 (dcha. ancha)
645	1E valor	valor para coordenadas de corrección en juego 10	68C	00 valor	valor para campo de juego 1 y 2 (izda. fina)
646	24 valor	valor para coordenadas de corrección en juego 11	68D	80 valor	valor para campo de juego 1 y 2 (dcha. ancha)
647	2A valor	valor para coordenadas de corrección en juego 12	68E	00 valor	valor para campo de juego 1 y 2 (izda. fina)
648	79 valor	Color demarcación blanco color de fondo azul juego 1	68F	80 valor	valor para campo de juego 1 y 2 (dcha. ancha)
649	6C valor	Color demarcación amarillo color de fondo rojo juego 2	690	00 valor	valor para campo de juego 1 y 2 (izda. fina)
64A	7C valor	Color demarcación blanco color de fondo rojo juego 3	691	80 valor	valor para campo de juego 1 y 2 (dcha. fina)
64B	7D valor	color demarcación blanco color de fondo violeta juego 4	692	00 valor	valor para campo de juego 1 y 2 (izda. fina)
64C	4F valor	color demarcación rojo color de fondo blanco juego 5	693	80 valor	valor para campo de juego 1 y 2 (dcha. ancha)
64D	OD valor	color demarcación negro color de fondo violeta juego 6	694	FF valor	valor para campo de juego 1 y 2 (izda. fina)
64E	79 valor	color demarcación blanco color de fondo azul juego 7	695	FF valor	valor para campo de juego 1 y 2 (dcha. fina)
64F	49 valor	color demarcación rojo color de fondo azul juego 8	696	AA valor	valor para campo de juego 1 y 2 (izda. fina)
650	OD valor	color demarcación negro color de fondo violeta juego 9	697	55 valor	valor para campo de juego 1 y 2 (dcha. ancha)
651	68 valor	color demarcación amarillo color de fondo negro juego 10	698	FF valor	valor para campo de juego 1 y 2 (izda. fina)
652	OA valor	color demarcación negro color de fondo verde juego 11	699	FF valor	valor para campo de juego 1 y 2 (dcha. fina)
653	5F valor	color demarcación violeta color de fondo blanco juego 12	69A	00 valor	valor para campo de juego 1 y 2 (izda. ancha)
654	09 valor	jugador izdo.: amarillo, jug. dcho.: amarillo 1	69B	00 valor	valor para campo de juego 1 y 2 (dcha. ancha)
655	05 valor	pelota: verde 1	69C	FF valor	valor para la ampliación de los puntos de demarcación de campo
656	3E valor	jugador izdo.: negro, jug. dcho.: azul 2	69D	01 valor	valor para la ampliación de los puntos de demarcación de campo
657	00 valor	pelota: blanco 2	69E	00 valor	valor para la ampliación de los puntos de demarcación de campo
658	36 valor	jugador izdo.: azul, jug. dcho.: azul 3	69F	00 valor	valor para la ampliación de los puntos de demarcación de campo
659	01 valor	pelota: amarillo 3	6A0	3F valor	valor para la ampliación de los puntos de demarcación de campo
65A	29 valor	jugador izdo.: verde, jug. dcho.: amarillo 4			
65B	01 valor	pelota: amarillo 4			

7BF	08	valor	valor para ampliación de puntos de demarcación
7C0	01	valor	valor para generación de puntos
7C1	01	valor	valor para generación de puntos
7C2	50	valor	valor para tipo de punto
7C3	F0	valor	valor para tipo de punto
7C4	58	valor	valor para tipo de punto
7C5	F0	valor	valor para tipo de punto
7C6	0A	valor	forma de jugador para juego 1
7C7	1E	valor	forma de jugador para juego 2
7C8	0A	valor	forma de jugador para juego 3
7C9	1E	valor	forma de jugador para juego 4
7CA	0A	valor	forma de jugador para juego 5
7CB	14	valor	forma de jugador para juego 6
7CC	0A	valor	forma de jugador para juego 7
7CD	14	valor	forma de jugador para juego 8
7CE	0A	valor	forma de jugador para juego 9
7CF	0A	valor	forma de jugador para juego 10
7D0	0A	valor	forma de jugador para juego 11
7D1	0A	valor	forma de jugador para juego 12
7D2	00	valor	anchura de jugador
7D3	10	valor	anchura de jugador
7D4	10	valor	anchura de jugador
7D5	10	valor	anchura de jugador
7D6	10	valor	anchura de jugador
7D7	10	valor	anchura de jugador
7D8	10	valor	anchura de jugador
7D9	10	valor	anchura de jugador
7DA	00	valor	anchura de jugador
7DB	00	valor	anchura de jugador
7DC	00	valor	anchura de jugador
7DD	00	valor	anchura de jugador
7DE	00	valor	anchura de jugador
7DF	00	valor	anchura de jugador
7E0	7E	valor	forma de jugador para juego 6
7E1	7E	valor	forma de jugador para juego 6
7E2	7E	valor	forma de jugador para juego 6
7E3	00	valor	forma de jugador para juego 6
7E4	00	valor	forma de jugador para juego 6
7E5	00	valor	forma de jugador para juego 6
7E6	00	valor	forma de jugador para juego 6
7E7	81	valor	distancia de jugadores para juegos 2 y 4
7E8	81	valor	distancia de jugadores para juegos 2 y 4
7E9	81	valor	distancia de jugadores para juegos 2 y 4
7EA	81	valor	distancia de jugadores para juegos 2 y 4
7EB	81	valor	distancia de jugadores para juegos 2 y 4
7EC	81	valor	distancia de jugadores para juegos 2 y 4
7ED	81	valor	distancia de jugadores para juegos 2 y 4
7EE	00	valor	distancia de jugadores para juegos 2 y 4
7EF	00	valor	distancia de jugadores para juegos 2 y 4
7F0	00	valor	forma de representación de las cifras de marcador en juego 1
7F1	00	valor	forma de representación de las cifras de marcador en juego 2
7F2	00	valor	forma de representación de las cifras de marcador en juego 3
7F3	00	valor	forma de representación de las cifras de marcador en juego 4
7F4	04	valor	forma de representación de las cifras de marcador en juego 5
7F5	0C	valor	forma de representación de las cifras de marcador en juego 6
7F6	04	valor	forma de representación de las cifras de marcador en juego 7
7F7	0C	valor	forma de representación de las cifras de marcador en juego 8
7F8	82	valor	forma de representación de las cifras de marcador en juego 9
7F9	82	valor	forma de representación de las cifras de marcador en juego 10
7FA	00	valor	forma de representación de las cifras de marcador en juego 11
7FB	82	valor	forma de representación de las cifras de marcador en juego 12
7FC	00	libre	
7FD	00	libre	
7FE	00	libre	
7FF	00	libre	

Anexo

Lista de componentes completa para el juego televisivo

- 1 Microprocesador 2650A (sólo versión rápida)
- 1 interface de video programable 2636
- 1 generador PAL 2621
- 2 EPROM 2708 o
- 1 EPROM 2716
- 1 74LS00
- 2 74LS04
- 1 74LS08
- 1 74LS109
- 3 74LS113
- 2 74LS136
- 1 74LS156
- 2 74LS251
- 1 74LS258
- 1 74LS378
- 1 4053
- 3 amplificadores operacionales 741
- 1 regulador de tensión 7805
- 1 regulador de tensión 78L05
- 1 regulador de tensión 78L06
- 1 regulador de tensión 78M12
- 1 transformador de placas SM55/12-12/0,8
- 2 rectificador B40C1500
- 1 cristal de cuarzo de 8,867238 MHz
- 1 condensador 2200 μ F/40 V
- 1 condensador 470 μ F/40 V
- 3 condensadores 100 μ F/16 V
- 2 condensadores 22 μ F/16 V
- 1 condensador 10 μ F/16 V
- 4 condensadores 1 μ F/16 V
- 6 condensadores 0,1 μ F bzw. 100 nF
- 5 condensadores 22 nF
- 1 condensador 15 nF
- 1 condensador 10 nF
- 1 condensador 220 pF
- 7 condensador 74 pF
- 1 condensador 27 pF
- 1 condensador 22 pF
- 1 condensador 18 pF
- 1 condensador 2,2 pF

1 condensador de ajuste variable 1,8 pF...35 pF
 1 condensador de ajuste variable 4 pF...70 pF
 1 inductancia fija 22 μ H
 1 potenciómetro de ajuste 500 Ω
 1 regulador 1 k Ω
 2 potenciómetros de ajuste 5 k Ω
 1 potenciómetro de ajuste 10 k Ω
 1 resistencia 100 Ω
 1 resistencia 220 Ω
 3 resistencias 470 Ω
 2 resistencias 680 Ω
 1 resistencia 820 Ω
 18 resistencias 1 k Ω
 1 resistencia 1,5 k Ω
 3 resistencias 2,2 k Ω
 1 resistencia 3,3 k Ω
 2 resistencias 3,9 k Ω
 15 resistencias 4,7 k Ω
 1 resistencia 8,2 k Ω
 7 resistencias 10 k Ω
 2 resistencias 15 k Ω
 1 resistencia 33 k Ω
 2 resistencias 68 k Ω
 3 resistencias 100 k Ω
 1 resistencia 820 k Ω
 4 resistencias 1 M Ω
 1 resistencia 1,5 M Ω
 2 resistencias 3,9 M Ω
 1 resistencia 8,2 M Ω
 1 resistencia 10 M Ω
 1 transistor BC238
 5 transistores BC239
 1 transistor BC309
 1 transistor BF254
 7 diodos 1N914 o 1N4148
 1 diodo-ZZD4,7
 1 diodo capacitivo BB142

NOTA

La tabla de símbolos adjunta compara las antiguas normas DIN con las nuevas normas Din Nr. 407000.

(La tabla contiene únicamente los símbolos incluidos en este libro)

		Nuevo	Antiguo
Puertas	INV.		
	AND		
	NAND		
	OR		
	NOR		
	EXOR		
Flip-Flops	RS-		
	D-		
	JK-		

CIRCUITOS DE ELECTRÓNICA

Títulos publicados

- 55 circuitos especiales de baja frecuencia
- 71 circuitos con transistores
- Triacs y tiristores
- Transistores MOS
- Circuitos impresos
- Juegos electrónicos en TV
- Fuentes de alimentación electrónica
- Proyectos de seguridad
- Circuitos electrónicos por control remoto

CIRCUITOS PRACTICOS DE ELECTRONICA

Todo buen aficionado a la electrónica tiene la posibilidad de organizar sus propios juegos comprando el televisor a un microprocesador, ya que éste permite programar libremente su control; de esta manera resulta sencillo desarrollar toda una serie de juegos personales televisivos.

En este libro se describen los distintos componentes y circuitos de un ordenador de programación libre que podrá conectarse a un televisor, obteniendo los resultados apetecidos.

Aquí la obra se inicia con la descripción del microprocesador, del interfase del video programable, del generador de sincronismos, del comutador analógico CMOS con multiplexor y de la memoria fija borrable y programable. Siguiéndose con la estructura y conexión del ordenador en sus distintas etapas: sumador de video, microprocesador, generador PAL y memoria de instrucciones, circuito de audio, modulador AF, fuente de alimentación e interconexión del ordenador. Finalmente, se describe el emulador para el desarrollo de programas y se dan las instrucciones para la realización de diversos juegos: tenis individual, tenis de dobles, fútbol, tenis de mesa, voleibol, baloncesto, juegos de obstáculos, etc.



9 788432 968068