Universidad Nacional Autónoma de México

FACULTAD DE INGENIERIA

DISEÑO Y CONSTRUCCION DE LA UNIDAD DE CONTROL DE UNA MAQUINA EMPAPELADORA DE MOSAICOS VENECIANOS

TESIS PROFESIONAL

Que para obtener el título de:
INGENIERO MECANICO ELECTRICISTA
P R E S E N T A N

Víctor Javier González Villela
Ricardo Alfonso Martínez Garza Fernández

Director Ing. Sócrates Alberto Muñis Zafra

Ciudad Universitaria, D. F. Mayo 1987

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
FACULTAD DE INGENIERIA

DISEÑO Y CONSTRUCCION DE LA UNIDAD DE CONTROL DE UNA MAQUINA EMPAPELADORA DE MOSAICOS VENECIANOS

TESIS PROFESIONAL QUE PARA OBTENER EL TITULO DE

INGENIERO MECANICO ELECTRICISTA

PRESENTAN

VICTOR J. GONZALEZ VILLELA

RICARDO A. MARTINEZ GARZA FERNANDEZ

DIRECTOR

ING. SOCRATES ALBERTO MUÑIZ ZAFRA

CIUDAD UNIVERSITARIA, D.F., MAYO 1987

Contenido

Introducción

El control de procesos tiene una historia muy antigua. Algunos principios de control ya eran utilizados en el siglo tercero antes de cristo. Posteriormente, en los siglos XVII y XVIII se inventaron el termostato y algunos aparatos que hacían que los molinos mantuvieran las aspas contra el viento aun cuando éste cambiara de dirección. Sin embargo, se puede decir que los sistemas de control iniciaron su desarrollo intenso durante la revolución industrial. En esa época, James Watt inventó el primer regulador automático de velocidad y lo adaptó a la máquina de vapor. A mediados de siglo XIX, James C. Maxwell analizó por primera vez diversos tipos de gobernadores de velocidad. Alrededor de 1930, Nyquist y Bode desarrollaron técnicas de análisis para sistemas retroalimentados utilizando conceptos de respuesta en frecuencia. Todo ésto evolucionó y se convirtió en la Teoría del Control Automático, donde analógicamente se analizan las características de salida de un sistema, se procesan y, finalmente, se producen cambios en las características de entrada para lograr que las salidas tengan siempre un valor constante.

Sin embargo, no todos los procesos son analógicos, los hay también discretos. Desde los últimos años de la década de los 50, gracias al advenimiento de las computadoras digitales, se han redefinido los conceptos de la teoría de control, aplicando el término "control" a sistemas que clásicamente no podrían definirse como tales. Originalmente la computadora se utilizó en sistemas de control exclusivamente para simular digitalmente lo que antes se hacía en forma analógica. Posteriormente se desarrolló toda una teoría alrededor de las computadoras digitales que se conoce como la Teoría de Control Digital. Hoy en día, podemos aplicar el concepto de control digital a un secuenciador "inteligente" que "controla" el funcionamiento de una máquina.

Existen muchas maneras de realizar digitalmente un secuenciador inteligente, desde lógica secuencial hasta microprocesadores. El bajo costo y la gran versatilidad de los circuitos han hecho que los métodos discretos sean descartados durante el desarrollo de sistemas. En la actualidad, es el microprocesador el que se aplica en la mayoría de los casos. El microprocesador es un circuito que ha traído consigo un cambio en la manera en que los sistemas electrónicos son diseñados. Los microprocesadores se utilizan en aplicaciones tan diversas como calculadoras de bolsillo, instrumentos de laboratorio, sistemas de control de vuelo y hasta en tarjetas de crédito inteligentes.

Introducción

En el presente trabajo se estudian dos métodos generales para el desarrollo de controles secuenciadores inteligentes: Los controles microprogramados y los controles con microprocesador.

Definición y orígenes del problema

Capítulo uno

La empresa Mosaicos Venecianos de México, a través del programa de riesgo compartido del Consejo Nacional de Ciencia y Tecnología (CONACYT), hizo posible que el Centro de Diseño Mecánico e Inovación Tecnológica (CDMIT) de la Facultad de Ingeniería de la UNAM desarrollara el diseño de una máquina empapeladora de mosaicos venecianos. En este proyecto se nos presentó la oportunidad de desarrollar nuestro trabajo de tesis, no como un proyecto puramente académico, sino como un proyecto en el cual, además de aplicar los conocimientos adquiridos durante la carrera, se tuvieron que estudiar otros conceptos relacionados con el funcionamiento de la industria. Dentro de estos conceptos se encuentran: Estándares, disponibilidad de materiales, costos de fabricación y funcionamiento, etc.

Para llegar a definir bien el problema es imprescindible dar a conocer un panorama general de éste. La planta de Mosaicos Venecianos de México, que se ubica en Cuernavaca Morelos, produce mosaicos venecianos, mosaicos para pisos y mosaicos para acabados artísticos. Cuantitativamente hablando, su principal producto son los mosaicos venecianos. Estos mosaicos son los que se utilizan generalmente en baños y albercas. Sus pequeñas dimensiones (2x2 cm.) hacen imposible el colocarlos uno por uno, por lo tanto, para facilitar su colocación, estos mosaicos se forman en arreglos de 15x31 y se les pega una hoja de papel por el anverso para darle rigidez al arreglo. A estos arreglos se les da el nombre de "tapetes". Para colocarlos, se toma todo un tapete y se pega a la pared. Cuando el cemento ya ha secado, se empapa la hoja de papel y se desprende. De esta manera, los mosaicos son más fáciles de colocar, además de permitir que queden mejor alineados.

La producción de tapetes resulta insuficiente para satisfacer las demandas de los mercados nacional y extranjero. Por lo tanto, se desea producir de 2000 a 3000 tapetes diariamente para satisfacer dichas demandas, es decir, un tapete cada 15 segundos. Actualmente el proceso de formación y empapelado de los tapetes se lleva a cabo manualmente y el trabajo lo realiza un grupo de aproximadamente 50 personas. Con la automatización de la línea de producción, además de incrementar la producción, se puede reducir a estas personas para la elaboración de mosaicos artísticos, para el empaquetado de los tapetes, para la distribución y venta del producto, etc., actividades que por naturaleza deben realizarse manualmente.

Definición y orígenes del problema

El proyecto completo de Mosaicos Venecianos de México, CONACYT y CDMT consta de dos partes:

(a) Diseño de una máquina que forme los mosaicos en arreglos de 15x31.

(b) Diseño de una máquina que empapele los arreglos para formar los tapetes.

Para el diseño y construcción de estas máquinas se requiere de la participación de ingenieros mecánicos electricistas en sus tres áreas: Mecánica, electrónica e industrial. El caso que nos ocupa, para el presente trabajo, es el diseño y construcción del control de la máquina empapeladora.

Los requerimientos de funcionamiento del control son los siguientes:

(a) Operación automática.
(b) 250 tapetes por hora (mínimo).
(c) Utilizar componentes de fabricación nacional (o importados de de fácil adquisición en México).
(d) Utilizar materiales altamente resistentes a la corrosión, abrasión y altas temperaturas ya que estarán en contacto con polvo de vidrio, agua, harina y sosa.

El control se diseñó posteriormente al diseño y construcción del prototipo de la máquina empapeladora. La operación de empapelado de los tapetes se lleva a cabo haciendo pasar una hoja de papel Kraft de dimensiones adecuadas por un rodillo engomador. Una vez engomada una cara del papel, éste se deposita y se pega sobre los mosaicos que se encuentran formados sobre una charola de aluminio. El prototipo consta de los siguentes conjuntos de partes:

Depósito de hojas: Es un depósito de base rectangular con capacidad para 1000 hojas. Cuenta con guías y brazos que permiten la dosificación de una sola hoja bajo la tolva de vacío.

Rodillo engomador: Es un rodillo de cerdas que gira a velocidad constante y que absorbe pegamento (90% agua, 8% harina de trigo, 2% sosa) de un recipiente al cual está sujeto un "peine" dosificador de la goma. El rodillo distribuye el pegamento sobre toda la hoja que va siendo transportada por la tolva de vacío.

Transportador de charolas: Aquí se efectúa el posicionamiento del papel sobre la charola con mosaicos. El papel baja, ya engomado, verticalmente, sujeto por la tolva de vacío. La charola se encuentra esperando sobre el transportador que la alinea para recibir la hoja. Cuando el papel y la charola hacen contacto, la tolva de vacío ejerce cierta presión sobre los mosaicos con objeto de lograr un acoplamiento uniforme entre los mosaicos y el papel. El papel se adhiere a los mosaicos y la tolva sube, retrayéndose, para iniciar un nuevo ciclo. La charola es entonces retirada del transportador por medio de un pistón neumático.

Definición y orígenes del problema

Tolva de vacío: Es el elemento que se encarga de succionar una hoja de papel del depósito, transportarla por el rodillo engomador y acoplarla sobre la charola de mosaicos. La tolva es accionada por medio de dos pistones neumáticos, uno vertical y otro horizontal. Ambos pistones se controlan por medio de válvulas solenoide e interruptores límite magnéticos en los extremos de los pistones. La succión se logra por medio de una bomba de vacío de 1.5 HP y 30 m^3/h.

Bastidor: El bastidor se diseñó de perfil tubular en el cual se apoyan directamente el motorreductor que acciona el engomador, la bomba de vacío, los pistones y la unidad de control.

A continuación se presenta un diagrama esquemático de las partes que constituyen el prototipo de la máquina empapeladora:

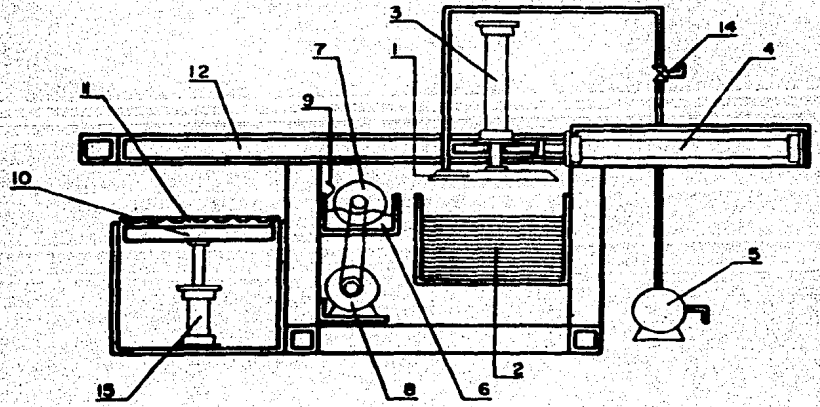


FIGURA 1.1

(1) Tolva de vacío
(2) Depósito de hojas
(3) Pistón neumático vertical
(4) Pistón neumático horizontal
(5) Bomba de vacío
(6) Recipiente para el pegamento
(7) Rodillo engomador
(8) Motorreductor
(9) Palma depositadora
(10) Transportador de charolas
(11) Charola
(12) Soatidor
(13) Unidad de control
(14) Válvula de control de vacío
(15) Pistón neumático de expulsión de charolas

Este prototipo es la alternativa que mejor cubre las necesidades planteadas. La selección se efectuó en base a los resultados obtenidos en pruebas realizadas a diferentes modelos fabricados en el taller del CDMIT.

Nuestro trabajo consistió, pues, en diseñar y construir la unidad de control de la máquina empapeladora. Durante el estudio inicial para elegir la mejor alternativa de diseño se analizaron las siguientes opciones:

(a) Microsecuenciador neumático: Es un producto de la marca FESTO que consiste de un émbolo que se va desplazando, abriendo y cerrando válvulas según se tiene "programado". Actúa por medio de señales de entrada neumáticas o microswitches. Se desechó esta opción por su costo y dificultad de mantenimiento. Además, su tiempo de respuesta es demasiado alto para los requerimientos de un tapete cada 15 segundos.

(b) Relevadores y microswitches: Se encuentra sujeto a muchos errores debidos a fallas en los dispositivos y ocupa un volumen muy grande.

(c) Electrónicamente: Es pequeño, puede tener cualquier número de entradas y de salidas. Su costo es reducido y es fácil de mantener (casi no requiere mantenimiento). Es interfaceable con casi cualquier dispositivo eléctrico, neumático, etc. Consume muy poca energía. Es completamente programable y se pueden incorporar sistemas de protección para detener el funcionamiento en casos de emergencia o falla de la máquina. Su tiempo de respuesta es casi inmediato (ns.)

Del análisis anterior se tomó la decisión de diseñar el control electrónicamente.

Criterios de diseño

Capítulo dos

Para dar solución a un problema es necesario tomar en cuenta todos los factores que intervienen en él. Es la única forma de llegar a una solución completa. Sin embargo, es difícil llevar a la práctica este concepto ya que, muchas veces, intervienen factores externos o inclusive internos que se pasan por alto inadvertidamente dado que, en el momento del análisis, no se les concede la importancia que realmente tienen.

Por lo tanto, para diseñar el control de la máquina empapeladora, fue necesario analizar todas las alternativas con el fin de satisfacer los objetivos presentes y futuros del cliente. Los factores más importantes que se tomaron en cuenta fueron:

- Funcionamiento
- Costo
- Mantenimiento
- Requisitos de instalación
- Consumo de potencia
- Versatilidad
- Limitaciones de espacio
- Características ambientales
- Seguridad
- Facilidad de manejo

Funcionamiento: El control de la máquina debe llevar a cabo todas las funciones automáticamente sin la intervención del operador. De nada sirve un control que requiera de la asistencia de un persona para su correcto funcionamiento.

Costo: Existen muchas alternativas (neumáticas, eléctricas, mecánicas y electrónicas) que satisfacen las necesidades de funcionamiento. Sin embargo, se debe escoger aquella cuyo costo sea menor sin sacrificio de la calidad.

Mantenimiento: Aquí debe incluirse tanto el costo como la frecuencia y facilidad de mantenimiento. Un equipo es mejor en tanto requiera del menor mantenimiento posible y que, cuando llegue el momento de efectuarlo, sea fácil, rápido y económico.

Requisitos de instalación: Mientras menor sea el número de requisitos para la instalación del equipo, menor será su costo de funcionamiento y mayor la facilidad de la instalación.

Criterios de diseño

Consumo de potencia: El costo de un equipo incluye, además de su costo de fabricación, su costo de operación. Aun cuando un equipo pueda tener el menor costo de adquisición, quizá no sea el más económico operativamente si requiere de un gran suministro de energía.

Versatilidad: Dado un equipo, sus necesidades futuras de funcionamiento pueden llegar a diferir de las actuales. Por lo tanto, es necesario realizar un diseño que permita el cambio de funciones con el menor número de modificaciones.

Limitaciones de espacio: De entre todas las alternativas se debe escoger aquella que cumpla con todas las características anteriores y que además sea la más compacta. De esta manera, será mucho más fácil su colocación dentro de la planta.

Características ambientales: El equipo debe resistir condiciones extremas de temperatura y atmósferas corrosivas, características que son comunes de la industria.

Seguridad: El equipo no debe representar un peligro para las personas que lo operen.

Facilidad de manejo: Una parte del costo de funcionamiento de un equipo es la capacitación del operador. Si un equipo es fácil de manejar, el tiempo y costo de capacitación se reducen.

Además de todas las características anteriores, es necesario analizar si los lineamientos de diseño deben darse en forma general o particular. Si el equipo que se va a diseñar es único en su género, no conviene darle una forma general tratando de prever equipos con características semejantes que puedan llegar a aprovechar el diseño del equipo actual. Por lo tanto, si el equipo es único, conviene hacer el diseño particularmente. Sin embargo, la máquina empapeladora es una máquina que lleva a cabo un número finito de operaciones, cíclica y casi indefinidamente. Sabemos que existen muchas máquinas que desarrollan sus funciones de la misma manera. De aquí que, en este caso se debe diseñar el control de la máquina empapeladora de tal manera que pueda ser utilizado en otros equipos cuyo funcionamiento sea similar.

El criterio de diseño fue, pues, dividir el control en dos módulos: Tarjeta principal (o de control) e interfaces. El primer módulo se pensó de la manera más general, es decir, un control con varias líneas de entrada y varias de salida. Su aplicación directa a una máquina en particular depende de las interfaces de entrada/salida que se conecten a la tarjeta de control. De esta manera, la próxima vez que se requiera diseñar un control para una máquina cíclica, será necesario únicamente desarrollar las interfaces ya que el módulo de control es el mismo.

Criterios de diseño

Electrónicamente hablando, existen sólo dos alternativas que cumplen con todas las características mencionadas anteriormente, a saber: Controles microprogramados y controles con microprocesador. Se desarrollaron ambas alternativas y posteriormente se compararon para escoger la mejor, utilizando como criterios de comparación las características de funcionamiento, costo, mantenimiento, etc.

En los próximos dos capítulos se hará un desarrollo de ambas alternativas y en el subsiguiente se hará la comparación entre las dos para la elección de la mejor.

Controles microprogramados

Capítulo tres

3.1 Generalidades

Un sistema microprogramado es una simulación simplificada y decreta de un microprocesador. Sin embargo, representa una alternativa viable para el desarrollo de algunos sistemas sencillos, en donde un microprocesador queda muy sobrado.

Un sistema microprogramado se forma básicamente de una memoria ROM, un reloj (secuenciador) y uno o más registros.

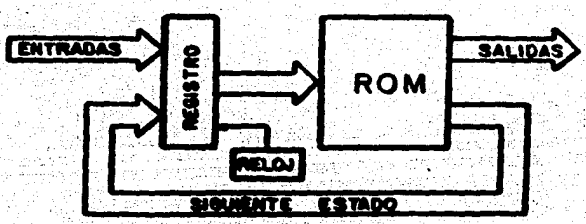


FIGURA 3.1

Cada una de las localidades de la ROM almacena una microinstrucción. En general, cada microinstrucción está dividida en dos partes. La primera es un conjunto de bits que, junto con las entradas al sistema, sirven para generar la dirección de la siguiente microinstrucción (siguiente estado). La segunda es otro conjunto de bits que constituyen las salidas del sistema.

La secuenciación de las microinstrucciones se lleva a cabo de la siguiente manera: Al aparecer el frente de onda positivo del reloj (tren de pulsos), se almacenan en las salidas del registro el estado de las entradas en ese momento y las salidas de la ROM. La unión de estos conjuntos de bits representa una dirección para la ROM. Una vez estable esta dirección, se presentan en las salidas de la ROM los bits que servirán para generar el siguiente estado y los bits de salida, es

decir, la microinstrucción. La microinstrucción permanece estable en las salidas de la ROM durante todo un periodo del reloj hasta que se presenta el siguiente frente de onda positivo y de esta manera dar inicio a un nuevo ciclo.

Como se puede observar, estos sistemas son por naturaleza secuenciales y la función no depende del alambrado de los circuitos, sino del microprograma. Es decir, una vez alambrado el circuito, se puede cambiar completamente su funcionamiento simplemente cambiando el microprograma. De aquí que, se puede decir que un sistema microprogramado es una emulación discreta de un microprocesador.

Existen varias maneras de implementar un circuito microprogramado dependiendo de las necesidades específicas del sistema. En el caso de la máquina empapeladora, las entradas nunca se presentan en forma simultánea. Por lo que toca a las salidas, a veces se requiere que varias estén presentes al mismo tiempo. Dadas las características de este sistema, se desarrolló una implementación de direccionamiento implícito, que es el que permite multiplexar las entradas y a la vez obtener salidas simultáneas. El esquema general de direccionamiento implícito es el siguiente:

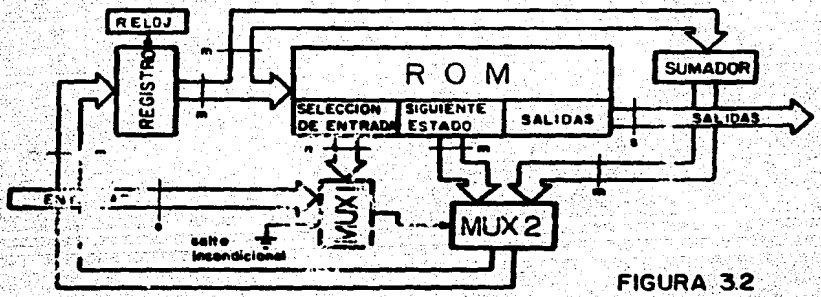


FIGURA 3.2

El formato de la microinstrucción es el siguiente:

- **n** Es el número de bits de selección de entradas. Dependiendo del código de estos n bits, es la entrada que se multiplexa durante ese periodo de reloj.
- **m** Es el número de bits que se utilizan para determinar el siguiente estado (siguiente dirección en la ROM).
- **s** Es el número de bits de salida.

El circuito funciona de la siguiente manera. Supongamos un sistema donde m=3 y cuyo estado inicial sea 000. A la salida del sumador obtendremos el código 001 el cual se alimenta a la mitad de las entradas del multiplexor 2. El multiplexor debe ser del tipo m x m/2 con una sola entrada de selección. A la otra mitad de este multiplexor se alimentan los m bits de siguiente estado de la microinstrucción (salidas de la ROM). Los n bits de selección de entrada de la microinstrucción se conectan a las entradas de selección del multiplexor de entradas (multiplexor 1). Este multiplexor debe ser del tipo e x 1 tal que se cumpla que e+1 = 2^n. Dependiendo de la entrada multiplexada en mux1 se seleccionará en el mux2, ya sea el siguiente estado secuencial o, la dirección de salto que proviene de la ROM. La salida del mux2 se alimenta a las entradas del registro de tal manera que al llegar el siguiente pulso de reloj se almacene en él la dirección de la siguiente microinstrucción. Durante cada estado existen s bits de salida. Es importante notar que una de las entradas del mux1 se denomina "de salto incondicional"; esta entrada se conecta permanentemente a 0 ó 1 lógico y se utiliza para realizar cambios de estado incondicionales cuando se desea romper la secuencia.

Por ejemplo, supongamos un sistema microprogramado con n=2 bits de selección de entrada (3 entradas y 1 salto incondicional), m=3 bits de direccionamiento y s=4 bits de salida. Supongamos, también, que cuando la entrada es 0 lógico se sigue en secuencia y cuando es 1 se lleva a cabo un salto. Hagamos el microprograma de la carta ASM de la figura 3.3

Supongamos que estamos en el estado 000 (renglón 1), el bit de salida I_1 está encendido ya que esa es la salida que necesitamos según la carta ASM. En este estado sólo nos interesa el estado de la entrada Q_2 (código 01 de selección). Si dicha entrada es falsa (0 lógico), el mux2 deja pasar las salidas del sumador (001) por lo que pasamos al estado 001 (renglón 2) con el siguiente pulso de reloj. Preguntamos por Q_1 (código 00 de selección), si dicha entrada es verdadera (1 lógico), el mux 2 deja pasar las m bits de siguiente estado que provienen de la ROM. Las salidas son I_2 e I_4. Al llegar el siguiente pulso de reloj, y por ser verdadera la entrada Q_1, pasamos al estado 101 (renglón 6). La salida es I_3. Volvemos a preguntar por el estado de Q_1. Si Q_1 es falso, seguimos en secuencia al estado 110 (renglón 7). La salida de este estado es I_3. En este caso es necesario romper la secuencia para regresar incondicionalmente al estado 000 (renglón 1) por lo que preguntamos por el estado de Q_0 (código 11 de selección). Como esta entrada siempre es verdadera, se hace un salto incondicional al estado 000 para iniciar un nuevo ciclo.

De esta misma forma es posible analizar cualquiera de las ramas de la carta ASM para seguir la secuencia en función del estado de las entradas.

Como se puede observar, la implementación de direccionamiento implícito cumple con el esquema general mostrado al inicio del capítulo. La única diferencia es que las entradas, en vez de presentarse simultáneamente, se multiplexan para determinar el siguiente estado.

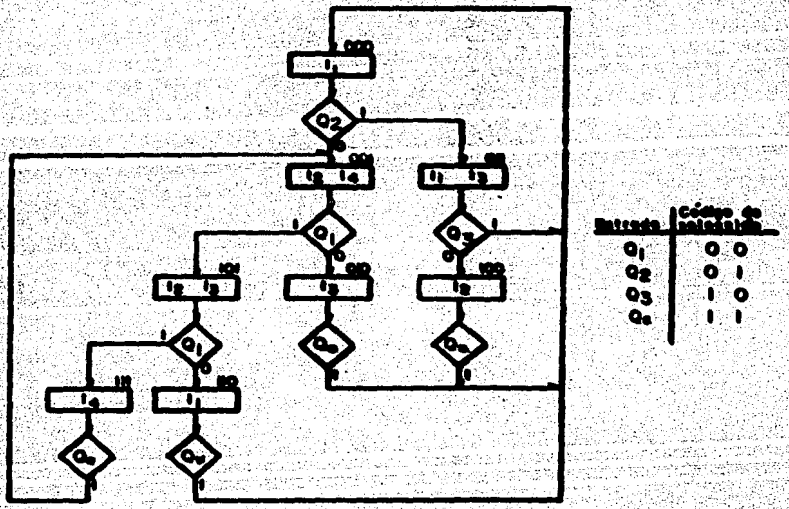


FIGURA 3.3

Dirección (Registro)	Selección de entrada	Siguiente estado	Salidas
000	01	011	0001
001	00	101	1010
010	11	000	0100
011	10	000	0101
100	11	000	0010
101	00	111	0110
110	11	000	0001
111	11	001	1000

MICROINSTRUCCION

La gran desventaja de los sistemas microprogramados es que la longitud de la microinstrucción depende directamente del número de entradas, del número de salidas y del número de estados. En un sistema con 7 entradas, 5 salidas y 256 estados, la longitud de la microinstrucción asciende a 16 bits (n=3, s=5, m=8). Para implementar un sistema de este tipo, lo que aparece en el esquema como "ROM", realmente son dos circuitos ROM ya que comercialmente estos circuitos tienen 8 bits de salida. Además, para implementar el mux2 se requieren dos multiplexores 8 x 4 y para el sumador, dos sumadores o contadores de 4 bits. De aquí que el costo y complejidad de implementación dependen de la longitud de la microinstrucción.

A continuación se presentan los esquemas implementados para el control de la máquina empapeladora utilizando el concepto de direccionamiento implícito. Uno con 2 EPROM y el otro con 3 EPROM.

3.2 Implementación con 2 EPROM

256 estados dentro de un microprograma podrían parecer excesivos; sin embargo, para la realización de un secuenciador inteligente, en donde se toman muchas decisiones dependiendo del estado de las entradas, es apenas un número adecuado de estados. Por lo tanto, se requieren 8 bits para el direccionamiento de los estados, de aquí que se requiera una EPROM exclusivamente para efectuar el direccionamiento. Optimizando el número de entradas y salidas, se requiere por lo menos de otra EPROM.

Es posible, utilizando algunos trucos, implementar el control de la empapeladora empleando únicamente 2 EPROM. La asignación de las entradas y las salidas se realizó de la siguiente forma:

Entradas:

Q_1: Pistón 1 contraído
Q_2: Pistón 1 extendido
Q_3: Pistón 2 contraído
Q_4: Pistón 2 extendido
Q_5: Pistón 3 contraído
Q_6: Falta papel en el depósito
Q_7: Esperando charola (charola fuera)
Q_8: Salto incondicional

Salidas actuadoras: (decodificación horizontal)

P1: Cambio de dirección de movimiento del pistón 1
P2: Cambio de dirección de movimiento del pistón 2
P3: Cambio de dirección de movimiento del pistón 3

Salidas indicadoras: (decodificación vertical)

NORMAL:	Operación normal de la máquina
CHAROLA:	Esperando la llegada de una charola
PAPEL:	Se acabó el papel del depósito
FALLA:	La máquina está en estado de falla

Como se puede observar de la descripción de las salidas, el truco consiste en aumentar dos de los cinco bits de salida a un decodificador 2x4 y de esta manera aumentar el número de salidas de 5 a 7. Esto es posible ya que las salidas indicadoras son mutuamente exclusivas. Por lo tanto, tenemos 3 salidas decodificadas horizontalmente (directamente del registro, pueden presentarse simultáneamente) y 4 salidas decodificadas verticalmente (a través del decodificador). Además, existen dos entradas asíncronas que son:

RST:	Para restablecer el sistema
PSE:	De tipo "push-on/push-off" para detener momentáneamente el funcionamiento de la máquina

A continuación se presentan los diagramas de bloques y electrónico de la implementación de la tarjeta de control de la máquina empapeladora empleando 2 EPROM:

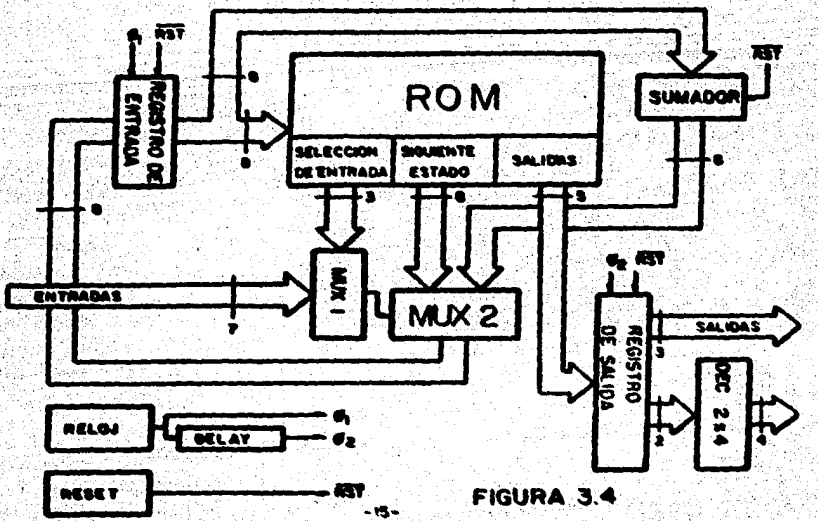


FIGURA 3.4

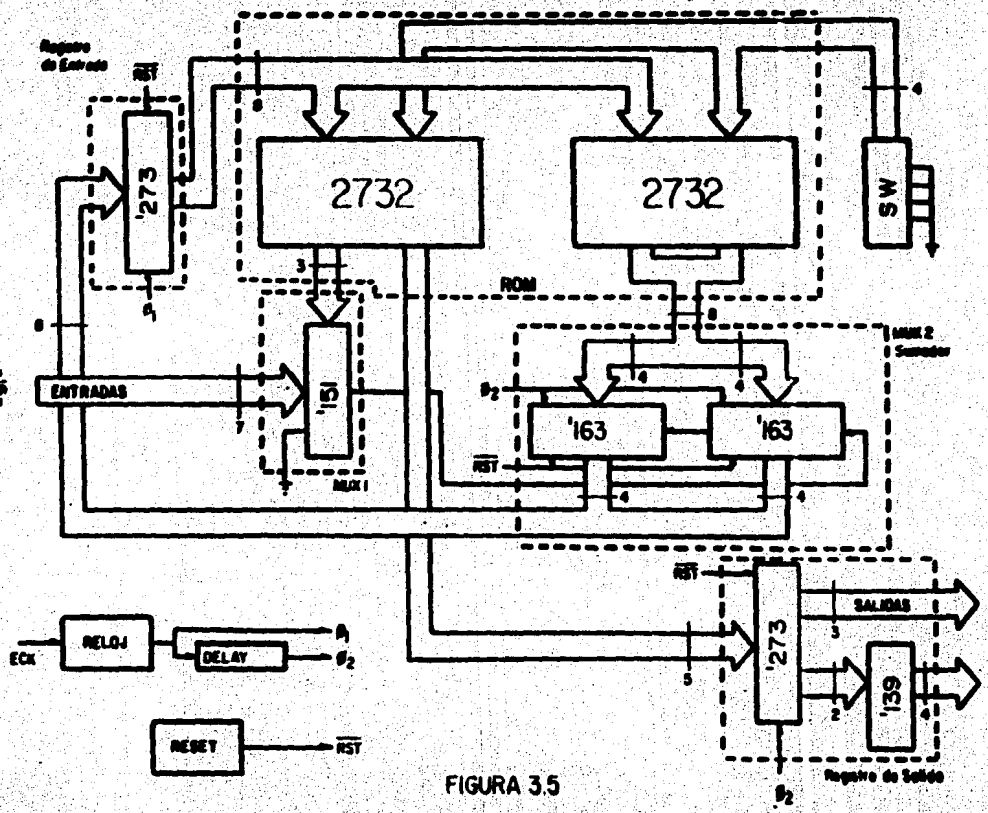

Registro de Entrada
RST
'273
2732
2732
SW
ROM
MUX 2 Sumador
ENTRADAS
'151
MUX 1
'163
'163
RST
RELOJ
DELAY
RESET
SALIDAS
'273
'139
Registro de Salida

FIGURA 3.5

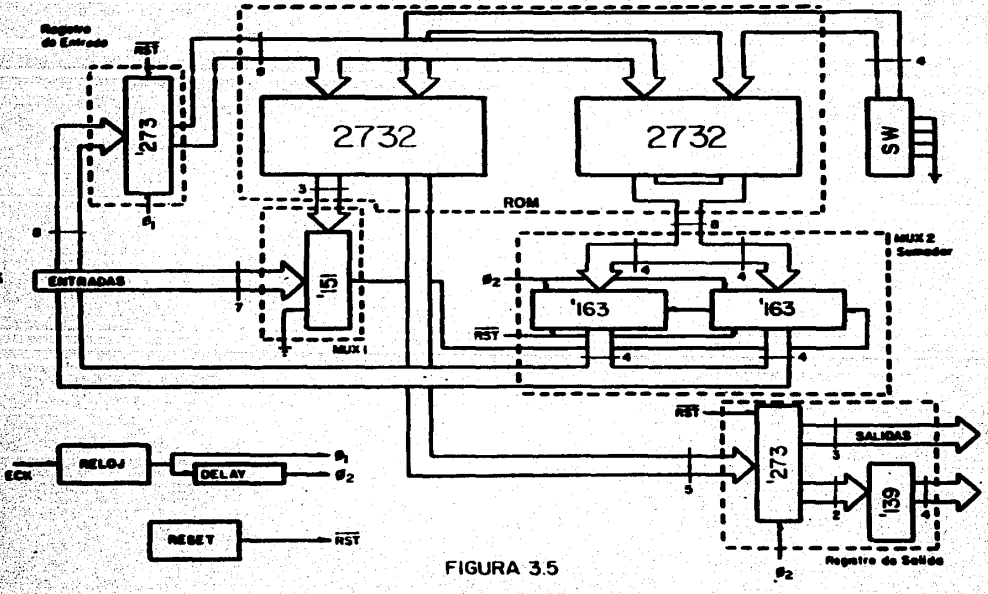

Registro de Entrada
RST
'273
2732
2732
SW
ROM
MUX 2 Sumador
ENTRADAS
'151
MUX 1
ϕ_2
'163
'163
RST
RST
'273
SALIDAS
'139
Registro de Salida
ϕ_2
ECK
RELOJ
DELAY
ϕ_1
ϕ_2
RESET
RST

FIGURA 3.5

BLOQUE ROM:

El bloque ROM consta de 2 UVEPROM 2732 de 4Kx8. Debido a que definimos m=8 bits para direccionamiento de los estados y a que las EPROM tienen 12 bits de direccionamiento, se conectan los 4 bits más significativos a cuatro "pull-up's" (5.6k) y a cuatro switches (SPST) conectados a tierra. De esta manera, se puede dividir la capacidad de la memoria en 16 páginas de 256 microinstrucciones abriendo o cerrando los switches. Cuando el switch está abierto, esa línea representa un 1 lógico a través del "pull-up". Cuando el switch se cierra, esa línea se conecta a tierra proporcionando un 0 lógico. Dependiendo de la posición de los cuatro switches, es la página de memoria que se está direccionando. Con esta implementación se pueden tener simultáneamente hasta 16 microprogramas diferentes o de respaldo.

En la primera EPROM (a la izquierda) se almacenan los 3 bits de selección de entrada y 5 de los 8 bits de direccionamiento del siguiente estado. En la segunda se almacenan los otros 3 bits de direccionamiento y los 5 bits de salida. Las 16 salidas en conjunto forman la microinstrucción.

BLOQUE MUX 1:

Este bloque lo constituye un circuito multiplexor 8x1 (74LS151) con tres líneas de selección. A las entradas de selección se conectan las salidas de selección de entrada que provienen del bloque ROM. De la ocho líneas de entrada al multiplexor, siete se utilizan para las entradas del control y una se conecta permanentemente a tierra (salto incondicional). La salida de este circuito se conecta al bloque MUX 2.

BLOQUE SUMADOR/MUX-2:

Para realizar el bloque SUMADOR se pueden utilizar dos sumadores de 4 bits (CD4008) encadenados, en donde por un lado se alimentan las salidas del registro de entrada y por el otro se alambra la palabra 01 (hexadecimal). La suma se lleva a cabo asíncronamente (25 ns). Esto es, a la salida de los sumadores se encuentra "siempre" presente la dirección secuencial siguiente a la dirección actual.

Para el bloque MUX 2 se pueden utilizar dos multiplexores 8x4 (74LS157). A las entradas de cada multiplexor se conectan, por un lado las salidas de uno de los sumadores y, por el otro la mitad de las salidas de direccionamiento de siguiente estado del bloque ROM. Las salidas de ambos multiplexores se conectan a las entradas del registro de entrada. La selección (SUMADOR o ROM) se lleva a cabo con la salida de MUX 1, es decir, dependiendo del estado de la entrada multiplexada.

Sin embargo, estos dos bloques se pueden implementar utilizando dos contadores síncronos con carga síncrona (74LS163). Estos circuitos están constituidos por un registro contador de cuatro bits que por medio de la entrada LD modifican su comportamiento. Si la entrada LD esta en 1 lógico, al llegar el siguiente pulso de reloj (retrasado por

el bloque DELAY) simplemente se suma uno a la cuenta anterior. Si la entrada está en 0 lógico, al llegar el siguiente pulso se carga en el registro el valor que en ese momento se encuentra en las entradas

Aprovechando las características de estos circuitos, se pueden incorporar en un mismo bloque las funciones de SUMADOR y MUX 2. A las entradas de los contadores se conectan las líneas de direccionamiento del siguiente estado que provienen del bloque ROM. A la entrada LD se conecta la salida del bloque MUX 1. Cuando la entrada multiplexada (MUX 1) está en 1 lógico, al llegar el pulso de reloj, simplemente se suma uno al contador y la dirección del siguiente estado será la secuencial siguiente. Cuando la entrada multiplexada está en 0 lógico, al llegar el pulso de reloj se carga en el contador la salida de direccionamiento del bloque ROM, es decir, la dirección de salto de la microinstrucción. En resumen, si la entrada multiplexada está en 1 lógico, no hay salto. Si la entrada multiplexada está en 0 lógico, sí lo hay.

BLOQUE REGISTRO DE ENTRADA:

Este bloque está constituido en su totalidad por un registro de 8 bits (74LS273). A sus entradas se conectan las salidas del bloque MUX 2. Sus salidas se conectan a las entradas de direccionamiento (8 bits menos significativos) del bloque ROM. Finalmente, la entrada CK se conecta a la salida del bloque RELOJ.

BLOQUE RELOJ:

El reloj está formado por un circuito oscilador RC-TTL con la mitad de un circuito NAND Schmitt trigger (74LS132). La frecuencia de oscilación se calcula como:

$$f_O = \frac{1}{1.4\ RC}$$

si R=1K y C=47nf, entonces f_O = 15KHz. De aquí que se ejecutan, aproximadamente, 15000 microinstrucciones por segundo. Esto hace que el tiempo de respuesta de un control microprogramado sea muy pequeño.

Con la entrada ECK se puede detener la oscilación del reloj. Cuando esta entrada tiene un 0 lógico, el oscilador se detiene. Cuando tiene un 1 lógico funciona normalmente. Esta entrada se utiliza para la entrada asíncrona PSE. Si se desea detener la operación de la máquina, simplemente se oprime el botón de pausa. Debido a que el botón es del tipo "push-on/push-off", con solo oprimirlo nuevamente la máquina continúa a partir de donde se quedó.

La salida de este bloque se conecta a la entrada CK del registro de entrada y a la entrada del bloque DELAY.

BLOQUE DELAY:

Ya que las EPROM requieren de un cierto tiempo para que las salidas se establicen (450ns tiempo de acceso), el reloj de los bloques MUX 2 y REGISTRO DE SALIDA debe estar retrasado con respecto al reloj del registro de entrada para dar tiempo a que las salidas de la EPROM se establicen. El tiempo de retraso debe ser tal que

$$450ns < t_{delay} < \frac{1}{f_o}$$

Para lograr este efecto se utiliza un circuito monoestable no redisparable (74LS121). El reloj del sistema se alimenta a la entrada B del monoestable. Cuando el frente de onda positivo se presenta, el monoestable se dispara (salidas: Q=1, Q^n=0). Un tiempo t_{delay} después, el monoestable regresa a su estado original (salidas: Q=0, Q^n=1). Como la entrada CK de todos los bloques es de frente de onda positivo, se utiliza la salida Q_n del monoestable para generar el reloj retrasado y alimentarlo a los bloques MUX 2 y REGISTRO DE SALIDA. El tiempo de disparo del circuito se calcula como:

$$t_{delay} = 0.45\ RC \text{ nanosegundos}$$

R está en kohm
C está en pf

si R=1.5K y C=22nf, entonces t_{delay} = 15 microseg, lo cual cumple con las restricciones de retraso enunciadas anteriormente.

BLOQUE REGISTRO DE SALIDA:

Este bloque está constituido por otro registro de 8 bits (74LS873). A sus entradas se conectan los 5 bits de salida de la microinstrucción y sus salidas se conectan, 3 directamente a las salidas de la tarjeta de control y, 2 a un decodificador 2x4 (74LS139). Las cuatro salidas del decodificador también se conectan directamente a las salidas de la tarjeta.

Este bloque es necesario para mantener estable el estado de las salidas durante todo el periodo; si no existiera, el estado de las salidas sería incierto durante el tiempo de acceso del bloque ROM.

BLOQUE RESET:

Para proporcionar un circuito de reset que funcione también durante el encendido, se implementó un circuito RC de tal forma que mientras se carga el capacitor el sistema está en estado de reset. Cuando el capacitor se carga, libera al sistema para que funcione normalmente. Con ésto se garantiza que el sistema arrancará siempre a partir de la dirección 00H y que no habrá salidas (todas en estado 0 lógico). Además, limpia también el bloque MUX 2.

Para este circuito se aprovechó la otra mitad del circuito NAND Schmitt trigger. Ya que la entrada CL de todos los circuitos es negada, es necesario invertir doblemente el estado del arreglo RC.

Además, con el "push button" conectado en paralelo con el capacitor se puede dar un reset al sistema en cualquier momento. Su efecto será el mismo que el de haber apagado y vuelto a encender la máquina.

Las salidas de este bloque se conecta a las entrada CL de los bloques MUX 2 y REGISTROS DE ENTRADA y SALIDA.

3.3 Implementación con 3 EPROM

La implementación con 2 EPROM es suficiente para realizar el control de la máquina empapeladora. Sin embargo, si se quisiera ampliar el número de funciones de la máquina, no podría hacerse ya que todas las entradas y salidas del control ya han sido utilizadas. Por lo tanto, conviene modificar el esquema de 2 a 3 EPROM para incrementar el número de entradas y salidas con el fin de permitir ampliar las funciones de la máquina en versiones futuras. Además, como se indicó en el capítulo anterior, uno de los objetivos de diseño del control es que pueda ser utilizado en otros equipos. Es decir, que el control sea general.

Con la ampliación de 2 a 3 EPROM se logran las siguientes ventajas:

(a) El número de entradas se incrementa de 7 a 15, más del doble (n=4).

(b) El número de salidas se incrementa de 5 a 8 (s=8).

(c) El número de estados por microprograma se incrementa de 256 a 4096 (m=12).

La gran ventaja de los controles microprogramados es su modularidad. Cada uno de los diferentes bloques permanece funcionalmente igual. Las modificaciones de los bloques son las siguientes:

BLOQUE ROM:

Se conectan 3 EPROM en vez de 2. La microinstrucción se forma de la siguiente manera: En la primera EPROM se almacenan los cuatro bits de selección de entrada y los cuatro bits más significativos de direccionamiento del siguiente estado. En la segunda EPROM se almacenan los ocho bits menos significativos de direccionamiento del siguiente estado. Finalmente, en la tercera EPROM se almacenan los ocho bits de salida. Por lo tanto, el bloque tiene ahora 12 bits de entrada (se pueden direccionar las 4K microinstrucciones) y 24 bits de salida (n=4, m=12, s=8)

BLOQUE REGISTRO DE ENTRADA:

En el esquema con 2 EPROM este bloque estaba totalmente constituido por un registro de 8 bits (74LS273). Ahora, como existen 12 bits de direccionamiento del bloque ROM, dentro de este bloque se conecta, en cascada, otro registro de 4 bits (74LS75). De esta manera el bloque tiene ahora 12 bits de entrada y 12 de salida.

BLOQUE SUMADOR/MUX-2:

Debido a que los contadores 74LS163 son de 4 bits, simplemente se conecta otro en cascada para tener un total de 12 bits.

BLOQUE MUX-1:

Dado que con este nuevo esquema existen 4 bits para selección de entrada, se cambia el multiplexor 8x1 (74LS151) por un multiplexor 16x1 (74LS150).

BLOQUE REGISTRO DE SALIDA:

Este bloque no se modifica. Simplemente se aprovechan los 8 bits del registro 74LS273, a comparación de los 5 que se utilizaban en el esquema anterior.

Los bloques RST, DELAY y RELOJ no se modifican en absoluto. Funcionalmente el esquema no se modifica, lo que es más, si se desea trabajar con el esquema con 2 EPROM habiendo alambrado el esquema con 3 EPROM, sólo es necesario quitar los circuitos de expansión de sus bases y el esquema funciona perfectamente.

Controles con microprocesador

Capítulo cuatro

4.1 Generalidades

Los controles basados en un microprocesador se incluyen en el grupo de las microcomputadoras de propósito específico. Esto quiere decir que un control con microprocesador es, simplemente, una microcomputadora.

Las microcomputadoras de propósito general contienen muchos elementos que pueden llegar a utilizarse dependiendo del uso que se le vaya a dar a la microcomputadora. En el caso de las de propósito específico, se emplean únicamente los elementos necesarios para que ésta realice su función. Sin embargo, el esquema general de ambas es el mismo:

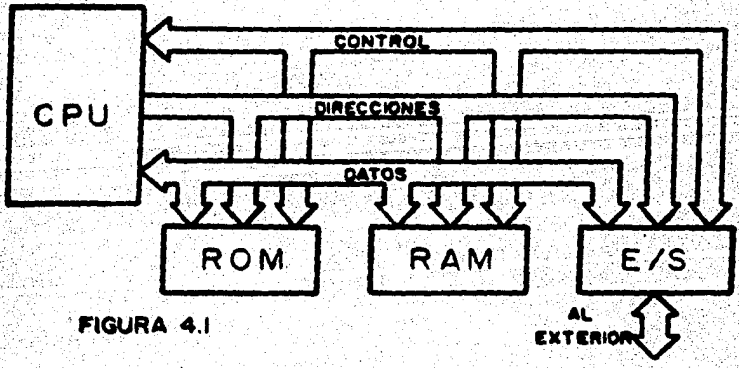


FIGURA 4.1

BLOQUE CPU:

El principal elemento de este bloque es, precisamente, el microprocesador. La elección del microprocesador adecuado depende de muchos factores, todos ellos relacionados con la aplicación que se le vaya a dar.

(a) Longitud de palabra: Existen microprocesadores con longitudes de palabra que varían desde 4 hasta 32 bits. El criterio para determinar cual es la longitud adecuada al caso depende, principalmente, de la precisión que se requiera en los cálculos y de los requerimientos de entrada/salida.

(b) Registros: Los procesadores que contienen registros de propósito general tienen, inherentemente, mayor versatilidad que los que contienen registros de propósito específico.

(c) Stack: Es importante que el microprocesador contenga el manejo de stacks. Estos son indispensables para el anidamiento de subrutinas, el manejo de interrupciones y el almacenamiento temporal de datos de programas residentes en ROM.

(d) Set de instrucciones: Si la aplicación requiere del manejo de grandes volúmenes de información, se debe buscar el microprocesador que tenga el set de instrucciones más versátil. Algunos microprocesadores contienen instrucciones que permiten la manipulación de bits independientes dentro de sus registros, e inclusive dentro de cualquier localidad de memoria. Esta característica es muy importante en aplicaciones de control.

(e) Modos de direccionamiento: Los diferentes modos de direccionamiento proven la capacidad y flexibilidad de manejar la información en la manera que mejor convenga al programador. Entre los modos más importantes que debe manejar un microprocesador están: Directo, indirecto, inmediato e indexado.

(f) Requerimientos de voltaje: Algunos microprocesadores requieren de dos o más fuentes de voltaje. La mejor opción es, siempre, aquellos que funcionen con una sola fuente.

(g) Velocidad: No siempre el microprocesador más rápido es el mejor. En el caso de aplicaciones de control, generalmente la velocidad no es un factor muy importante. Por lo tanto debe escogerse aquel cuya velocidad sea la adecuada.. Si se elige un microprocesador más rápido de los necesario, no se obtendrá ganancia alguna y, en cambio, el costo del microprocesador depende directamente de su velocidad.

(h) Compatibilidad: Es de suma importancia que el microprocesador sea directamente compatible con otras familias lógicas (principalmente TTL y CMOS). Si el microprocesador requiere de drivers o interfaces, el costo del diseño completo se verá afectado por el costo de estos componentes.

(i) Software: Este concepto, muchas veces pasado por alto, es quizá uno de los más importantes. Debe escogerse aquel microprocesador para el cual se pueda conseguir algún tipo de software que facilite la programación. Un compilador que genere código objeto para el microprocesador es la mejor opción ya que permite la programación en lenguaje de alto nivel. Si no se cuenta con un compilador, debe escogerse un microprocesador para el cual se tenga un

ensamblador. Debe evitarse, a toda costa, el programar directamente en lenguaje de máquina ya que el proceso es muy lento y está sujeto a infinidad de errores.

(J) Costo: Sus implicaciones son obvias.

(K) Asequibilidad: Debe escogerse un microprocesador que sea fácil de conseguir.

Dentro de este bloque, además del microprocesador, está contenida toda la lógica de control e interfaz con los otros tres bloques: Reloj, RESET, WAIT, DMA, HALT, WRITE, READ, interrupciones, etc.

BLOQUE ROM:

Este bloque está constituido por memorias de tipos ROM, PROM o EPROM. Es aquí donde se almacena, permanentemente, el programa de aplicación o el BIOS. Su tamaño en KBytes depende de la magnitud de dichos programas.

Dependiendo del volúmen de producción se elige entre los tres tipos de memorias. Si el volúmen es grande, se pueden utilizar memorias de tipo ROM. Sin embargo, aun en volúmenes grandes, generalmente se utilizan memorias de tipo EPROM ya que son fácilmente programables en el campo, además de permitir borrarlas y volverlas a programar.

BLOQUE RAM:

La magnitud de RAM depende del volúmen de información que se vaya a manejar. Por lo general, en los sistemas de propósito general, la magnitud de la RAM es grande. Para sistemas de control, la RAM se utiliza principalmente para manejo del stack, por lo que con 256 ó 512 Bytes es suficiente.

BLOQUE E/S:

Este bloque es el que permite la comunicación de la microcomputadora con el exterior. Entre los dispositivos más comunes se encuentran: Controladores de TRC, manejadores de teclado, controladores de disco y cinta, puertos de comunicación serie y paralelo, etc.

4.2 Controles con microprocesador:

El control de la máquina empapeladora se realizó utilizando un microprocesador Z-80 funcionando a 2 MHz. La elección del dispositivo se hizo en base a todos los criterios ya mencionados. El criterio

principal de selección fue que contamos con un sistema de desarrollo, diseñado por nosotros, para este microprocesador. La descripción de este sistema de desarrollo se presenta en el capítulo 6 de este trabajo. El diagrama de la tarjeta de control se presenta en la figura 4.2.

BLOQUE CPU:

En la tarjeta de control, el bloque CPU lo constituyen: El microprocesador y los circuitos de Reloj, RESET, vigilancia (watch-dog) y decodificación de memoria y puertos.

El circuito de reloj emplea un cristal de 4 MHz conectado a dos inversores 74LS14 cada uno en paralelo con dos resistencias de 1K. La señal de salida se introduce a un flip-flop "D" 74LS74 en donde es dividida entre dos para generar el reloj del sistema de 2 MHz.

El circuito de RESET es igual al de los controles microprogramados y su salida se conecta al microprocesador, al puerto paralelo 8255 y al circuito de vigilancia.

El circuito de vigilancia está constituido por 1/2 monoestable redisparable 74LS123 y la otra mitad del flip-flop 74LS74. El flip-flop está alambrado como un flip-flop tipo "T". La entrada CK del flip-flop se conecta a la salida Q^0 del monoestable. La salida Q^0 del flip-flop se conecta a la entrada INT del microprocesador. El monoestable es disparado por una de las líneas de decodificación de puerto (dir. 04H-07H), esto hace que la salida Q^0 del monoestable cambie de estado de 0 a 1 lógicos. Si después de 5 segundos el monoestable no ha sido redisparado, su salida Q^0 regresa a su estado inicial (0 lógico) con lo que se dispara el flip-flop y su salida Q^0=0 indica al Z-80 que desea interrumpirlo. Cuando el microprocesador atiende la interrupción, habilita simultáneamente las salidas M1 e IORQ que sumadas (AND) se alimentan a la entrada CL del flip-flop, con lo que el flip-flop regresa a su estado inicial (Q^0=1) y libera la interrupción. El funcionamiento del circuito de vigilancia se explica ya que podemos colocar instrucciones, dentro del programa, que direccionen el puerto del circuito con intervalos menores a 5 segundos.

Este circuito se utiliza en el control para detectar fallas en el equipo. Por ejemplo, cuando se va a realizar el movimiento horizontal de la tolva, podemos disparar el circuito de vigilancia y después monitorear el estado del microswitch de límite horizontal. Si después de 5 segundos, el microswitch no ha cambiado de estado, quiere decir que la tolva se encuentra atascada en algún lado o que el motor de movimiento horizontal no funciona. Por lo tanto, el microprocesador es interrumpido por el circuito de vigilancia y la rutina de interrupción detiene el funcionamiento de la máquina, suena la alarma y enciede el indicador de falla. Si la máquina funciona normalmente, se deshabilitan las interrupciones cuando se detecta el cambio de estado del microswitch y se continúa con el proceso normalmente.

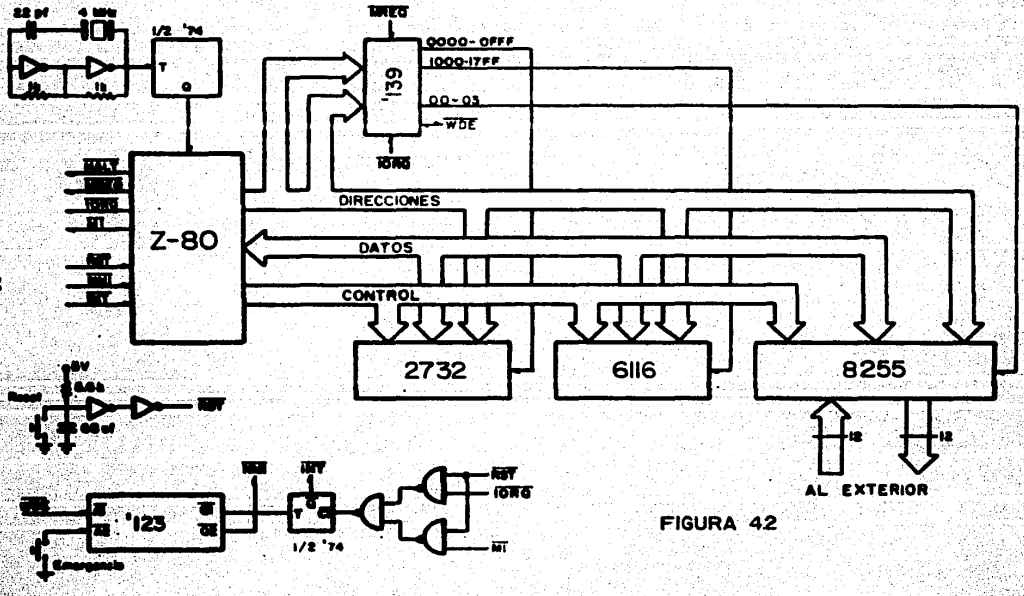

1/2 '74
'139
0000- 0FFF
1000-17FF
00-03
WDE
IORQ
Z-80
DIRECCIONES
DATOS
CONTROL
2732
6116
8255
12
12
AL EXTERIOR
'123
1/2 '74
RST
IORQ
MI

FIGURA 42

Otra parte del circuito de vigilancia es el boton de EMERGENCIA. Este botón, que se encuentra en el panel frontal, puede ser accionado en cualquier momento por el operador. Debido a que es una entrada asincrona que debe ser atendida de inmediato, esta entrada es conectada a la entrada de interrupción no enmascarable NMI del microprocesador a través de la otra mitad del monoestable 74LS123. Ya que esta entrada funciona con frente de onda y no con nivel, es necesario garantizar el tiempo de bajada de la señal para que el microprocesador la reconozca. Por lo tanto, el botón de EMERGENCIA se conecta a la entrada de disparo del monoestable; cuando el botón es oprimido, el monoestable se dispara interrumpiendo al microprocesador. No importa que el botón "rebote" ya que el monoestable, al ser redisparable, simplemente prolonga la duración del pulso, pero como ya se indicó, la entrada NMI funciona con el frente de onda de bajada y no importa cuanto tiempo permanezca la señal activada.

La salida del bloque RESET se conecta a las entradas CL de los monoestables y del flip-flop para garantizar el estado inicial de los dispositivos.

La decodificación de memoria y puertos se hace a través de un decodificador dual 2x4 74LS139. Para la decodificación de memoria se conectan las líneas A_{12} y A_{13} de direcciones del microprocesador a las entradas A y B del primer decodificador. La señal de MREQ se conecta a la entrada G del decodificador. Cuando esta señal está activa, el decodificador se habilita dividiendo la memoria en 4 bloques de 4 kBytes; dependiendo el estado de A_{12} y A_{13} se decodifican los bloques de ROM o RAM. Para la decodificación de puertos se conectan las líneas A_2 y A_3 de direcciones a las entradas A y B del segundo decodificador. La señal de IORQ se conecta a la entrada G. Cuando esta señal está activa, dependiendo del estado de A_2 y A_3, se decodifica el puerto paralelo 8255 o el circuito de vigilancia.

BLOQUE ROM:

Este bloque está constituido por una memoria UVEPROM 2732 de 4Kx8. Aquí se almacena el programa de la máquina. Aunque éste no ocupe los 4 kBytes, el costo de una 2732 y de una 2716 es el mismo. Utilizando la 2732 se tiene la posibilidad de que el programa crezca hasta 4 kBytes para ampliar el número de funciones de la máquina. Además, como el diseño del control se hizo en forma general, es preferible tener mayor espacio de programa en ROM.

BLOQUE RAM:

La cantidad de memoria contenida en este bloque es de 2 kBytes. Aunque se mencionó en la sección anterior que con 512 Bytes de RAM es suficiente, la asequibilidad de memorias RAM de esta capacidad es pequeña; por lo tanto, se utilizó una memoria SRAM 6116 de 2Kx8. En este bloque se almacenan temporalmente el stack y las direcciones de retorno de las subrutinas.

BLOQUE E/S:

Este bloque está constituido por un puerto paralelo 8255 que tiene 24 líneas de entrada/salida. Estas líneas pueden se configuradas de varias maneras distintas: Las 24 de salida, las 24 de entrada, 12 de salida y 12 de entrada, 8 de salida y 16 de entrada, 16 de salida y 8 de entrada, 4 de salida y 20 de entrada o 20 de salida y 4 de entrada. En el caso de la máquina empapeladora, las líneas están configuradas como 12 de salida y 12 de entrada.

Estas líneas se conectan a la tarjeta de interfaz que es la que comunica los elementos sensores, indicadores y actuadores de la máquina con la tarjeta principal. La tarjeta de interfaz se describe en el capítulo 7.

Comparación de alternativas

Capítulo cinco

En los capítulos anteriores se hizo una descripción general, así como un desarrollo particular de las alternativas viables para este proyecto. Sólo queda elegir la más conveniente. Los parámetros de comparación utilizados son los enunciados en el capítulo referente a los criterios de diseño.

Los criterios de Funcionamiento, Mantenimiento, Requisitos de instalación, Consumo de potencia, Limitaciones de espacio, Características ambientales, Seguridad y Facilidad de manejo se cumplen o satisfacen de igual manera con ambas alternativas. Por lo tanto, los criterios decisivos fueron:

C o s t o :

El análisis de costos se hizo en base a los costos de los componentes de ambas alternativas ya que los costos de programación, elaboración del circuito impreso y fabricación son esencialmente los mismos. Debido a las diferencias en precios entre los diferentes distribuidores del país, el análisis del costo de los componentes se hizo en base a su precio de distribución en dolares:

(a) Sistema con 2 EPROM:

2	2732	4.98
2	74LS163	0.98
2	74LS273	1.58
1	74LS151	0.39
1	74LS132	0.39
		$ 8.81

(b) Sistema con 3 EPROM:

3	2732	7.47
3	74LS163	1.47
2	74LS273	1.58
1	74LS175	0.39
1	74LS150	1.29
1	74LS132	0.39
		$ 12.59

(c) Sistema con microprocesador:

1	Z-80	1.75
1	2732	2.49
1	6116	1.69
1	8255	1.95
1	74LS74	0.25
1	74LS14	0.39
1	74LS139	0.39
1	74LS123	0.49
1	74LS00	0.25
1	XTAL 4M	1.19

		$ 10.84

Versatilidad:

No existe punto de comparación entre la versatilidad de un sistema microprogramado y uno con microprocesador. Basta repetir que un sistema microprogramado es una simulación simplificada y discreta de un microprocesador. Si de versatilidad se trata, no hay como un microprocesador.

C O N C L U S I O N :

Se eligió el sistema con microprocesador ya que su costo no es significativamente mayor que el de uno con 2 EPROM y su versatilidad es incomparable.

Sistema de desarrollo

Capítulo seis

6.1 Generalidades

A continuación se presenta una descripción del sistema de desarrollo SDD/RMG que se utilizó para el desarrollo del proyecto. Sus diferentes funciones le permiten ensamblar programas en código Z-80, cargarlos en cualquier página de memoria RAM, correrlos, depurarlos, simular sistemas, emular y programar EPROMs y otras utilerías.

El sistema de desarrollo está basado en un microprocesador Z-80. Junto con los otros componentes, se convierte en una microcomputadora de propósito general en una sola tarjeta. Este sistema, para su funcionamiento, hace uso de hasta 8 kBytes de EPROM (2 kBytes mínimo) y hasta 4 kBytes de RAM en la misma tarjeta. Contiene un puerto paralelo 8255 con 24 líneas de entrada/salida y circuitos de RESET, WAIT, Interrupción, Vigilancia, Decodificación de memoria y puertos, y Generación de reloj (4 MHz). Además cuenta con un puerto serie (serializador) 8251, generación de "BAUD rates" y drivers para la conversión TTL/RS-232C.

Esta tarjeta puede comunicarse con el exterior por medio del puerto paralelo (dispositivos de control y manejadores) y por medio del puerto serie (terminales y microcomputadoras).

El diagrama de bloques del sistema se muestra en la figura 6.1.

Además de la tarjeta principal, el sistema de desarrollo cuenta actualmente con dos tarjetas auxiliares. La primera es una expansión de memoria RAM que funciona, a la vez, como un emulador de EPROMs para aplicaciones externas al sistema. La segunda es un programador de EPROMs que permite la programación de memorias de tipo 2716, 2732, 2732A y 2764. Para establecer la comunicación entre todas las tarjetas se creó un bus de tipo "backplane" que, además de intercomunicarlas, permite la expansión del sistema con la adición de otras tarjetas que contengan funciones que quieran ser añadidas.

El sistema de desarrollo es autosuficiente, sin embargo, si se cuenta con la ayuda de una microcomputadora PC, se pueden editar y ensamblar los programas en ella y de ahí ser transmitidos hacia el sistema de desarrollo.

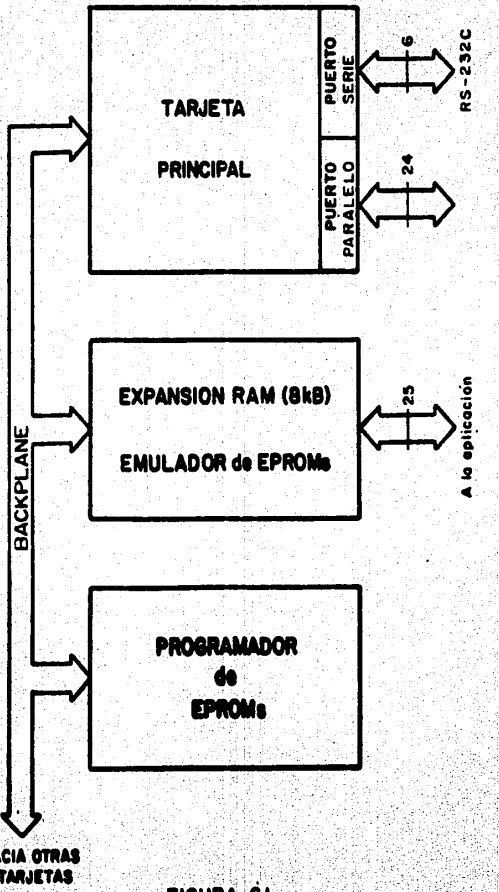

TARJETA
PRINCIPAL
PUERTO SERIE
PUERTO PARALELO
6
RS-232C
24
EXPANSION RAM (8kB)
EMULADOR de EPROMs
25
A la aplicación
BACKPLANE
PROGRAMADOR
de
EPROMs
HACIA OTRAS TARJETAS

FIGURA 6.1

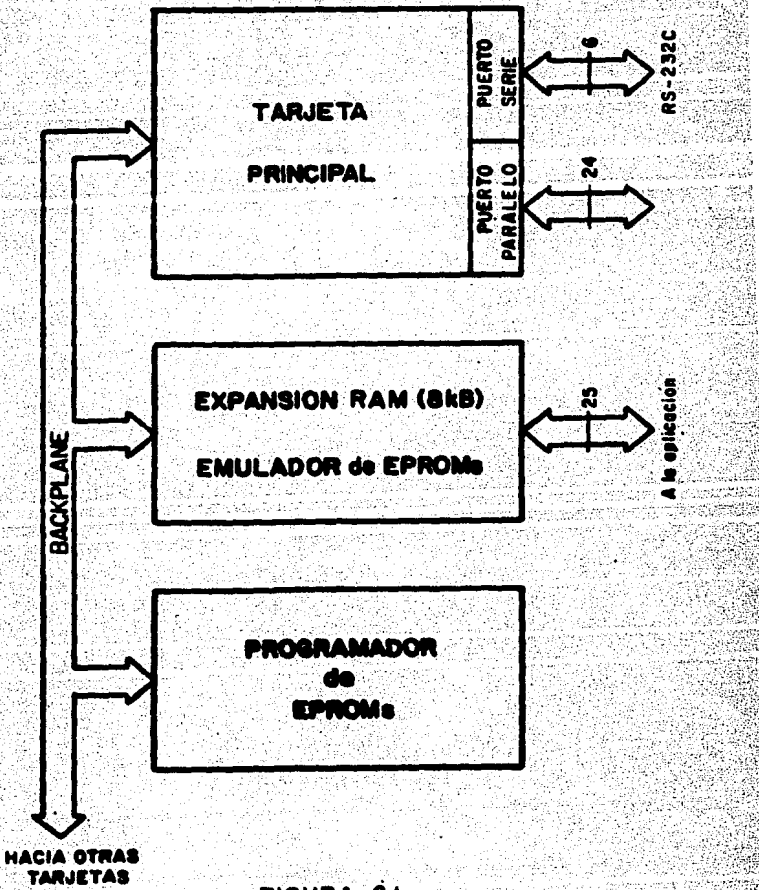

TARJETA
PRINCIPAL
PUERTO SERIE
PUERTO PARALELO
6
RS-232C
24
EXPANSION RAM (8kB)
EMULADOR de EPROMs
25
A la aplicación
BACKPLANE
PROGRAMADOR
de
EPROMs
HACIA OTRAS
TARJETAS

FIGURA 6.1

6.2 Tarjeta principal

Como ya se mencionó, el corazón del sistema reside en el microprocesador Z-80 que opera a 4 MHz. Contiene 8 KBytes de ROM residentes en dos UVEPROM 2732 y 4 KBytes de RAM alojados en dos SRAM 6116. Cuenta además con 24 líneas de entrada/salida programables y un serializador para la comunicación con una terminal o una microcomputadora. Se incluye también toda la lógica de control y decodificación de memoria y puertos y un circuito de vigilancia.

El firmware lo constituye un programa supervisor basado en el del sistema PAT del Instituto de Ingeniería de la UNAM. Tiene capacidad para el manejo de la memoria, registros y puertos; ejecución de programas e inserción de "breakpoints" para la depuración de los programas. Si se desea, además se puede utilizar un intérprete de lenguaje BASIC orientado al control de procesos.

A continuación se describe la operación de la tarjeta principal dividida en bloques funcionales. El diagrama de la tarjeta principal se muestra en la figura 6.2.

Reloj:

El oscilador central se forma con dos inversores TTL (1/3 74LS04) conectados a un cristal de 8 MHz. La señal de salida se divide entre dos por medio de un flip-flop "D" (1/2 74LS74) obteniéndose una señal de reloj de 4 MHz que se alimenta directamente al procesador y que, además, sirve de base para la generación de las diferentes velocidades de transmisión.

WAIT:

La señal M1* que se genera en el procesador durante los ciclos de "fetch", se introduce a la otra mitad del 74LS74 para sincronizar la señal de WAIT* de manera que sólo se introduzcan ciclos de espera durante los accesos a memoria en búsqueda del código de operación. Esto permite el acoplamiento de memorias más lentas (450 ns t_{access}) durante estos ciclos ya que el tiempo que permance activa la señal de MREQ* es aproximadamente 400 ns.

Decodificación de memoria:

Como se puede observar en el diagrama, el circuito de decodificación de memoria se compone de un decodificador 3x8 (74LS138) un inversor (1/6 74LS04) y dos compuertas AND (1/2 74LS08). Con este esquema se organizó la memoria en dos bloques. El primero consta de 16 KBytes dividido en páginas de 2 KBytes. El segundo consta de 48 KBytes que no se decodifican internamente en esta tarjeta.

Dado que las UVEPROM 2732 son de 4 KBytes, para la selección de cada uno de estos dispositivos se suman, a través de las AND, las dos primeras y las dos segundas líneas de salida del decodificador.

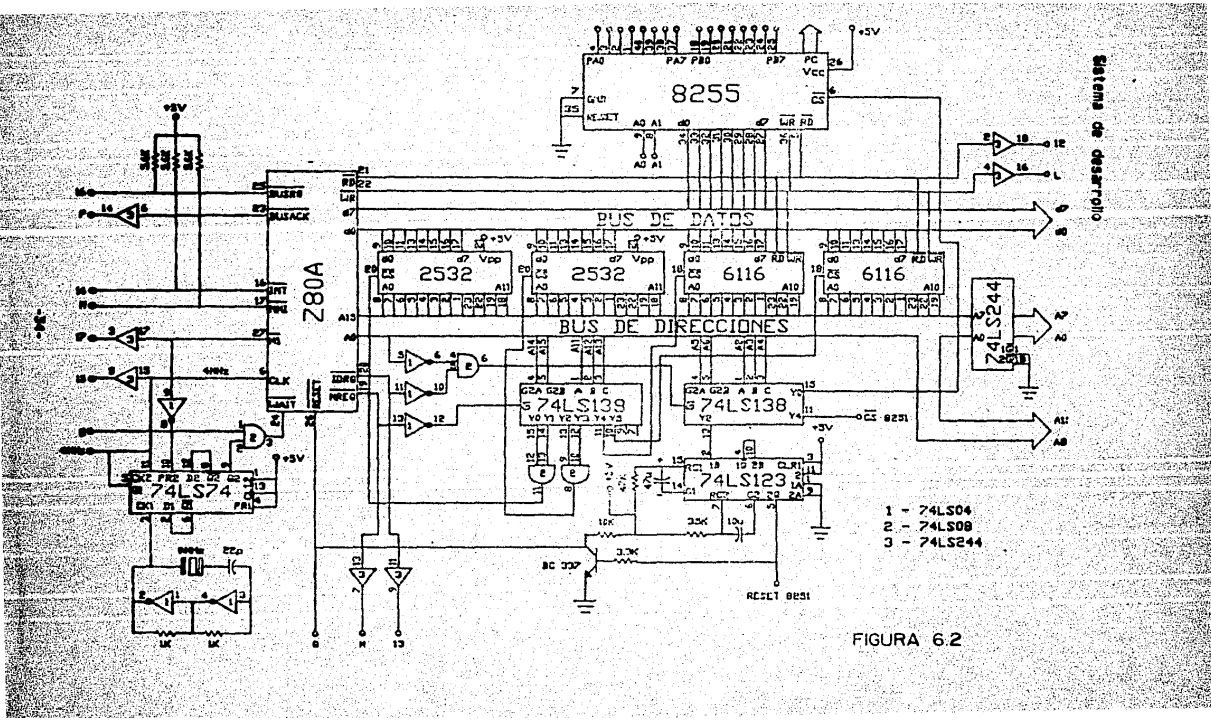

8255
PA0
PA7 PB0
PB7
PC
Vcc
RESET
A0 A1
d0
d7
WR RD
+5V
BUS DE DATOS
BUS DE DIRECCIONES
2532
2532
6116
6116
Vpp
A11
A10
Z80A
74LS244
74LS138
74LS123
74LS74
G2A G2B A B C
Y0 Y1 Y2 Y3 Y4 Y5 Y6 Y7
CLR
BUSREQ
BUSACK
WAIT
CLK
RESET
4MHz
BC 337
10K
33K
3.3K
RESET BBSI
1 - 74LS04
2 - 74LS08
3 - 74LS244

FIGURA 6.2

Decodificación de puertos:

La decodificación de puertos también se lleva a cabo por medio de otro decodificador 3x8 (74LS138), dos inversores (1/3 74LS04) y una compuerta AND (1/4 74LS08). Las tres líneas que se decodifican son A_2, A_3 y A_4, mientras que A_5, A_6 y A_7 en conjunto con la señal IORQ* habilitan el decodificador. Con ésto se forma el primer bloque de 32 Bytes, divido en 8 páginas de 4 Bytes cada una. Esta subdivisión se hace porque la mayoría de los dispositivos periféricos hacen la decodificación restante a partir de A_0 y A_1.

Circuito de vigilancia:

Este circuito está formado por dos monoestables redisparables (74LS123), arreglos RC para generación de tiempos y un transistor NPN 2A2222. La señal de selección del "watchdog" o circuito de vigilancia que proviene del decodificador de puertos dispara el primer monoestable (Q^*=0). Este cambio de nivel no dispara el segundo monoestable ya que se requere de un frente de onda ascendente para hacerlo.

Como los monoestables son redisparables y dados los valores de RC asociados a ellos, si la frecuencia de la señal de selección es mayor a 1 Hz, la señal Q^* permanecerá en 0 lógico. Sin embargo, si en algún momento deja de activarse el primer monoestable, éste regresará a su estado estable (Q^*=1) disparando el segundo monoestable. El disparo del segundo monoestable genera una señal de RESET al microprocesador a través del transistor. (El transistor es necesario para implementar, en el bus, que la señal de RESET* sea de tipo colector abierto.)

La función de vigilancia se explica ya que podemos colocar instrucciones que direccionen el puerto del watchdog con intervalos menores a 1 segundo. En el momento en que el procesador pierda el control sobre el programa, se dejará de direccionar ese puerto y se generará entonces un RESET que restablecerá el control.

Además, se puede generar un RESET en forma manual por medio de un "push button" conectado entre el colector del transistor y tierra.

Puerto paralelo 8255:

El 8255 de Intel es un dispositivo de entrada/salida programable de propósito general diseñado para ser utilizado con microprocesadores. Tiene 24 líneas de entrada/salida que pueden ser individualmente programadas en dos grupos de 12 y en tres modos de operación. En el modo 0, cada grupo de 12 líneas puede ser programado en conjuntos de 4 para ser de entrada o de salida. En el modo 1, Cada grupo puede ser programado para tener 8 líneas de entrada o salida y 4 de "handshake". En modo 2, el puerto A se programa como un puerto bidireccional con 5 líneas de "handshake". El puerto B y las 3 líneas restantes se pueden programar de entrada o salida.

Este dispositivo tiene cuatro puertos internos, los cuales son decodificados internamente a partir de A_0 y A_1. La selección del puerto se hace directamente con la primera línea de selección del decodificador de puertos (dir. 00H - 03H).

Puerto serie 8251:

Como ya se mencionó, la tarjeta principal aloja como posibilidad de comunicación un puerto serie conforme a la norma RS-232C. Este puerto incluye un USART 8251 de Intel, un generador de velocidades de transmisión y circuitos para interfaz TTL/RS-232C.

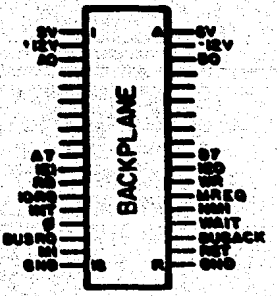
El generador de las velocidades de transmisión se forma dividiendo el reloj del sistema (4 MHz) por medio de un contador de de 4 bits (74LS93) para obtener una frecuencia de 250 KHz. Esta señal se alimenta a un divisor programable (XR2240). Como las salidas del divisor son de tipo colector abierto, por medio de switches se puede seleccionar la velocidad de transmisión desde 300 hasta 19200 BAUD.

La señal de salida del generador de velocidades se conecta a las entradas de reloj de transmisión y recepción del 8251. El dispositivo se selecciona con la cuarta línea de selección del decodificador de puertos (dir. 10H - 13H).

Las señales de transmisión (XMIT) y recepción (RCV), así como las de control para modem (RTS, DTR, CTS y DSR) están conectadas a través de los drivers MC1488 y MC1489 a un conector tipo DB-25S montado en la parte trasera del rack del sistema de desarrollo.

Bus principal (backplane):

Este bus está diseñado alrededor de un conector 18/36 e incluye las líneas de polarización de +5, ± 12 y tierra, las líneas de control del procesador, la parte baja de las líneas de direcciones (A_0 a A_7) y las líneas de datos. La figura 6.3 muestra la asignación de los pines de este bus.



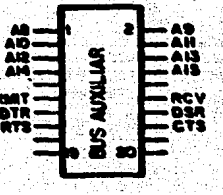


FIGURA 6.3

Bus auxiliar frontal:

En la figura 6.3 se muestra también la asignación de pines de este bus que contiene 20 líneas. Por este bus se comunican la parte alta del bus de direcciones (A_8 a A_{15}) y las señales de comunicación serie (de aquí parten hacia el conector DB-25S).

6.3 Tarjeta de programación de EPROMs

Con esta tarjeta se pueden programar los siguientes tipos de EPROM:

(a) 2716 (2K x 8)
(b) 2732 (4K x 8)
(c) 2732A (4K x 8)
(d) 2764 (8K x 8)

los cuales forman parte de la familia compatible Intel de EPROMs borrables con luz ultravioleta (UV). La única diferencia entre la 2732 y la 2732A es que la primera se programa con 25V y la segunda con 21V.

Funcionalmente, el programador consta principalmente de tres circuitos 74LS374, dos circuitos 74LS121 y un circuito 74LS244. El resto de la circuitería realiza operaciones de decodificación y conmutación. Se pueden efectuar tres funciones diferentes con el programador:

Programación:

Para llevar a cabo la programación del dispositivo, se requiere que tanto los datos, como las direcciones, se encuentren perfectamente estables antes y durante la aplicación del pulso de programación. Para lograr dicha estabilidad se hace uso de los tres registros de 8 bits (74LS374) y de los dos monoestables no redisparables (74LS121). Ver figura 6.4.

La información se latchea de la siguiente forma:

(a) Datos: (puerto 51H, 74LS374) Se almacenan en este registro los datos a programar en la localidad de la EPROM.

(b) Parte alta de la dirección a programar (A_8 a A_{12}), tipo de EPROM y bit de programación: (puerto 52H, 74LS374)

P/V	M1	M2	A12	A11	A10	A9	A8	
			▪	▪	▪	▪	▪	– Dirección (parte alta)
	0	0	----------					2716
	0	1	----------					2732
	1	0	----------					2732A
	1	1	----------					2764
0	----------							Modo lectura
1	----------							Modo programación

(c) Parte baja de la dirección a programar: (puerto 53H, 74LS374)

La secuencia de programación debe realizarse en ese orden. Primero, se latchean los datos en el puerto 51H. Segundo, se latchean la parte alta de la dirección a programar, el tipo de EPROM y el bit de programación (P/V = 1) en el puerto 52H. Finalmente, se latchea la parte baja de la dirección a programar en el puerto 53H.

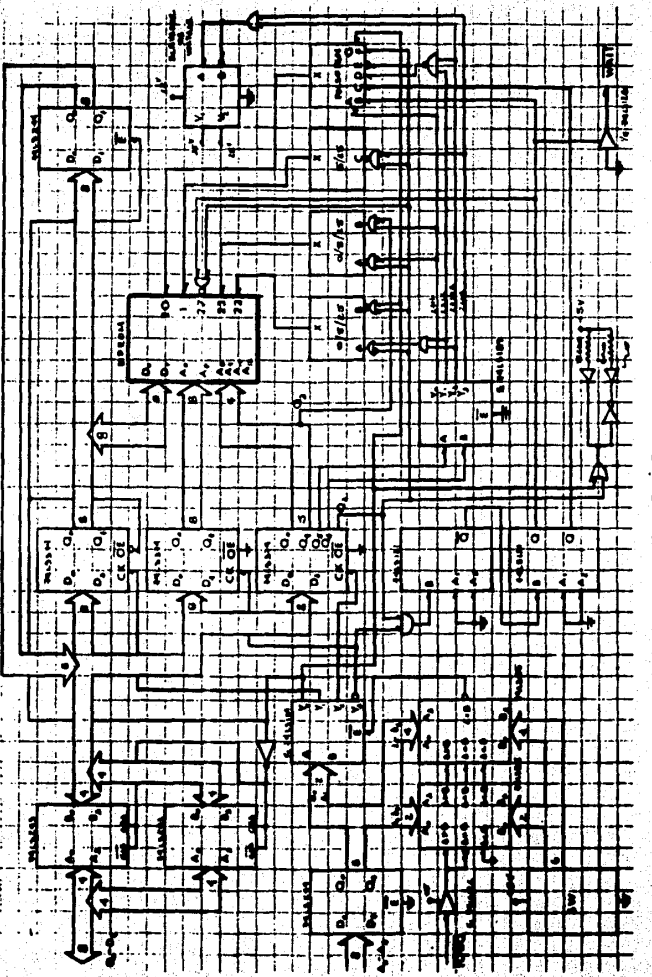


FIGURA 6.4

Al darse el pulso de reloj del puerto 53H (si P/V = 1) se dispara el primer monoestable para dar un retraso de 80 ns que permite que los datos se latcheen y estabilicen en el registro (los puertos 51H y 52H ya están estables). El frente de onda de subida de la salida negada del primer monoestable dispara al segundo que es el que proporciona el pulso de programación de 50 ms.

Así pues, hay que repetir esta secuencia por cada dirección a programar.

Lectura:

En este caso se utilizan dos de los registros (74LS374) y el buffer (74LS244). La secuencia de lectura es la siguiente. Primero, se latchean la parte alta de la dirección a leer, el tipo de EPROM y el bit de programación (P/V = 0) en el puerto 52H. Segundo, se latchea la parte baja de la dirección a leer en el puerto 53H. Finalmente, se lee el puerto 50H (74LS244) y se almacena el valor leído en RAM.

Como el bit de programación P/V = 0, no se disparan los monoestables y no se genera el retraso de 50 ms. Además, las patas de programación permanecen en 5V.

Verificación:

Esta operación es muy similar a la de lectura, con la diferencia de que no se carga en RAM lo que se lee de la EPROM. Además, se despliega en pantalla toda aquella localidad cuyo contenido es diferente de FFH.

El programa de la tarjeta de programación de EPROMs se encuentra almacenado permanentemente en la segunda EPROM de la tarjeta principal a partir de la dirección 1000H. En el apéndice A se proporciona una copia de dicho programa.

6.4 Tarjeta de emulación de EPROMs

Esta tarjeta permite la emulación de memorias de aplicaciones externas al sistema de desarrollo. Además funciona como una expansión de 8 kBytes de memoria RAM para el sistema de desarrollo.

El funcionamiento es sencillo. Se conecta una SRAM 6264 de 8Kx8 y todas sus líneas se multiplexan para que el control de dicha memoria lo pueda tener, ya sea el sistema de desarrollo o la aplicación. El diagrama de esta tarjeta se muestra en el figura 6.5.

La decodificación de memoria se realiza dentro de la trajeta por medio del decodificador 3x8 (74LS138) de la misma manera en que se decodifica cualquier otro bloque de RAM del sistema de desarrollo. La decodificación se hace en páginas de 8 kBytes. Como los primeros 16

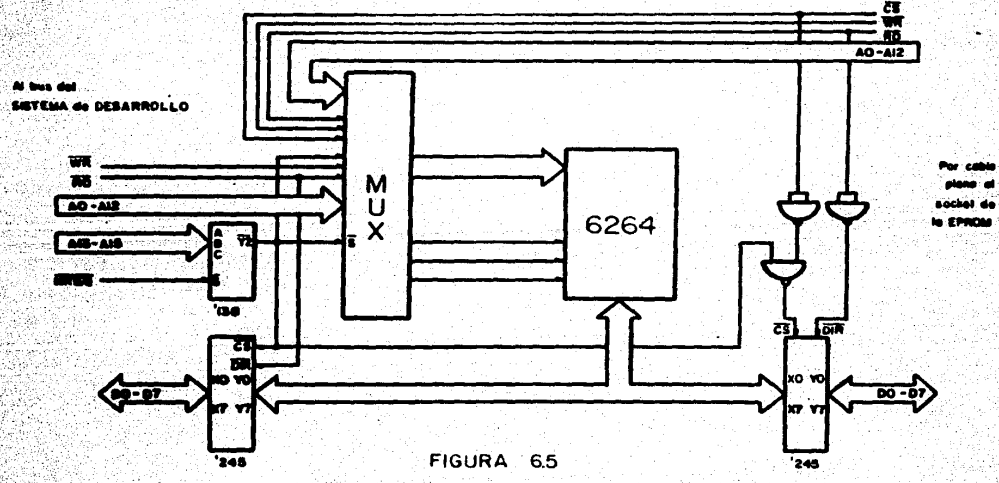

Al bus del
SISTEMA de DESARROLLO
WR
RD
A0 - A12
A13 - A15
MREQ
MUX
6264
'138
'245
'245
CE
WR
RD
A0 - A12
CS
DIR
D0 - D7
D0 - D7
Por cable plano al socket de la EPROM

FIGURA 6.5

KBytes de memoria ya han sido decodificados, se utiliza la tercera línea de salida del decodificador (Y_2: dir. 4000H - 5FFFH) para seleccionar la memoria.

Dentro del diagrama, el bloque marcado MUX es realmente un conjunto de cuatro multiplexores 8x4 (74LS157) con una sola línea de selección. A las entradas de estos multiplexores se conectan las líneas de direcciones A_0 a A_{12} y las líneas de control RD^*, WR^* y CS^* que provienen de los buses del sistema de desarrollo y del cable plano que conecta la tarjeta con la aplicación. Las salidas de este bloque se conectan directamente a las entradas correspondientes de la 6264.

El bus de datos, por ser bidireccional, se multiplexa de otra forma. El bus de datos del sistema de desarrollo se conecta a las líneas X_0 a X_7 del "transceiver" de 8 bits (74LS245) de la izquierda. El bus de datos de la aplicación se conecta a las líneas Y_0 a Y_7 del transceiver de la derecha. La dirección de comunicación se da por medio de la señal RD^* de cada bus. En el caso del bus del sistema de desarrollo, esta señal se conecta directamente a la entrada DIR de su 74LS245, ya que cuando DIR=0 la comunicación se hace desde Y hacia X; cuando DIR=1 la comunicación se hace desde X hacia Y. Esto es congruente con las funciones de lectura o escritura según el diagrama. En el caso del bus de la aplicación, es necesario invertir su señal de RD^* ya que éste está conectado a las líneas Y en vez de a las X.

Debido a que se puede presentar el conflicto de que tanto el sistema de desarrollo como la aplicación quieran accesar la memoria al mismo tiempo, es necesario establecer algún sistema de prioridad. En este caso se le dio prioridad al sistema de desarrollo. El conflicto se resuelve de la siguiente manera. La salida Y_2 (CS^* del sistema de desarrollo) se conecta a la entrada de selección del bloque MUX y al 74LS245 del sistema de desarrollo. Cuando esta señal está activa (CS^*_{SDD}=0), el multiplexor deja pasar las señales que provienen del bus del sistema de desarrollo. Cuando no está activa (CS^*_{SDD}=1), los multiplexores dejan pasar las señales que provienen del bus de la aplicación; además, el 74LS245 del sistema de desarrollo se encuentra en tercer estado en ambos sentidos. La habilitación de las salidas del 74LS245 de la aplicación depende no solo del estado de la señal CS^* de la aplicación, sino también del estado de la señal CS^* del sistema de desarrollo. Esto se hace por medio de dos compuertas NAND (74LS00). La primera sirve para invertir el estado de la señal CS^* de la aplicación, la segunda sirve para sumar la señal CS^* del sistema de desarrollo y la señal CS de la aplicación de manera que se cumpla la siguiente tabla:

CS^*_{SDD}	CS_{APL}	$G^*_{245(APL)}$
0	x	1
1	0	0
1	1	1

De esta manera, el sistema de desarrollo tiene prioridad de control sobre la tarjeta de emulación de EPROMs.

Como se puede observar, el funcionamiento de la tarjeta es completamente transparente tanto para el sistema de desarrollo como para la aplicación. Esto permite que la tarjeta pueda ser conectada directamente a los buses del sistema de desarrollo y a la aplicación (vía cable plano, conectando éste al socket donde normalmente se colocaría la EPROM).

Dado que dentro del cable plano también se maneja la señal de VR^{a}, este emulador también puede ser utilizado para memorias RAM. Sin embargo, su principal utilidad se presenta en la emulación de EPROMs.

6.5 Programa ensamblador para código 8085/Z-80

El programa XASM85 de Avocet Systems Inc. (V1.04, (c)1983) es un ensamblador cruzado (cross-assembler) para programas escritos en código ensamblador de 8085/Z-80. Este programa puede correrse en sistemas basados en CPM/80, CPM/86, MS-DOS, PC-DOS o TURBO-DOS. En nuestro caso lo utilizamos en una microcomputadora PC Televideo 2605 basada en el sistema operativo Tele-DOS (MS-DOS). Bajo este sistema operativo, el programa ocupa aproximadamente 20 kBytes de memoria.

XASM85 acepta lenguaje ensamblador proveniente de un archivo "fuente" en disco y genera dos archivos de salida (también en disco). El archivo "objeto" que contiene el código de máquina producido por el ensamblador y el archivo "listado" que contiene una copia del archivo fuente junto con el código generado por el ensamblador. En este segundo listado aparecen los errores de ensamblado en caso de que hayan existido. Además contiene la tabla de símbios definidos en el archivo fuente y los valores (hexadecmales) asignados a ellos.

El archivo fuente debe contener los mnemónicos estandard de Intel para su microprocesador 8085. Además pueden ser utilizados los mnemónicos TDL Intel para las instrucciones especiales del Z-80 de Zilog.

El archivo de listado está paginado. Cada página está numerada y contiene un título y un subtítulo. El objetivo de este listado es, obviamente, su impresión.

El archivo objeto se encuentra en formato Intel HEX. Este formato representa los códigos de operación y los datos como números hexadecimales en código ASCII. Este formato minimiza el tamaño del archivo objeto además de incluir códigos de CHECKSUM para cada registro.

El formato del archivo objeto es el siguiente:

```
:NnAaaa00DdDdDdDdDd ... DdDdDdCc
                .
                .
                .
:NnEeee01Cc
```

donde:

:	Indica el inicio de un nuevo registro
Nn	Indica el número de bytes de datos contenidos en el registro
Aaaa	Indica la dirección de organización del primer byte de datos
00	Indica que se trata de un registro de datos
Dd	Es un byte de datos
Cc	Es el checksum de ese registro
Eeee	Es la dirección de carga indicada en la pseudo-operación END
01	Indica que es el registro final del del archivo

El uso de este programa permite la programación en lenguaje ensamblador, lo que facilita tanto la programación, como la depuración y modificación de los programas.

6.6 Programa cargador

Este es un programa desarrollado por nosotros para ser utilizado en el sistema de desarrollo en conjunción con el ensamblador XASM65 o cualquier otro ensamblador cuyo archivo objeto tenga el mismo formato. El objetivo de este programa es cargar y relocalizar el código generado por el ensamblador en la memoria RAM del sistema de desarrollo.

El programa funciona de la siguiente forma. Por medio del programa de comunicación VTERM (que se describe en la siguiente sección), el programa cargador/relocalizador recibe el archivo objeto generado por el ensamblador. Durante la recepción, verifica que el número de bytes recibidos por registro concuerde con el número indicado en él, verifica que el registro tenga el formato adecuado, verifica que los registros de datos sean del tipo "00" y que el registro final sea del tipo "01", genera el checksum del registro y lo compara con el recibido. Cuando alguna de las verificaciones es negativa, detiene la recepción y manda un mensaje de error a la pantalla indicando de que tipo fue. Esto permite detectar errores en la transmisión de los archivos.

El programa cargador almacena temporalmente el archivo recibido a partir de la dirección 2800H. Esto permite que los registros del archivo objeto puedan estar revueltos (en lo que a dirección de organización se refiere) y posteriormente, dependiendo de la dirección de carga especificada en la pseudo-operación END, lo relocaliza. Durante los procesos de almacenamiento temporal y de relocalización, el programa verifica el estado de la memoria. En caso de encontrar alguna localidad dañada o inexistente, detiene el proceso y manda un mensaje de error a la pantalla indicando la localidad con falla.

Para la transmisión del archivo objeto, es necesario que el programa de comunicación termine cada registro con <CR><LF>, en ese orden. Además, es recomendable que el archivo objeto termine con un <CTRL-Z>.

Si no existió error en ninguno de los dos procesos, al finalizar el proceso de relocalización, se indica en la pantalla el número de bytes de datos recibidos, las direcciones inicial y final de organización y las direcciones inicial y final de relocalización.

Este programa se encuentra almacenado permanentemente en la segunda EPROM de la tarjeta principal del sistema de desarrollo a partir de la dirección 1380H (junto con el programa del programador de EPROMs). Una copia del listado de este programa se presenta en el apéndice B del presente trabajo.

6.7 Programa de comunicación VTERM

Este es un programa elaborado por Coefficient Labs. ((c)1984) que permite emular terminales VT52 y VT100 de Digital Equipment Corp.. Además permite la transmisión de archivos almacenados en disco desde y hacia la computadora con que la PC esté conectada, en este caso, con el sistema de desarrollo.

Este programa es sumamente útil para nuestro sistema de desarrollo ya que provee el medio de comunicación entre el usuario y el sistema, además de ser el complemento perfecto para los programas ensamblador y cargador.

alta del movimiento vertical de la tolva. La detección se hace por medio de un fotodetector colocado en la columna del transportador de la tolva.

Límite de engomado: Este es la posición de la tolva para que, en su trayecto hacia la charola con mosaicos, pase por el rodillo engomador. Este límite se sensa con otro fotodetector que también se encuentra colocado en la columna del transportador.

Límite inferior: Este límite se detecta por medio de un microswitch que hace contacto con el papel (dentro del depósito) y con la charola con mosaicos cuando la tolva realiza un movimiento descendente. El sensor se encuentra colocado en la parte inferior de la tolva.

Papel: La ausencia de papel dentro del depósito se detecta por medio de un microswitch que se encuentra colocado, también, en la parte inferior de la tolva.

Charola: La llegada y la salida de la charola con mosaicos se sensa con un microswitch colocado sobre el transportador de charolas.

Expulsor: Con un microswitch se detecta la posición del expulsor de charolas con el fin de asegurar su correcto funcionamiento.

Arranque: Cuando se enciende la máquina, el control realiza una rutina de inicialización. Al terminar la rutina, el control espera que el operador oprima el botón de arranque que se encuentra en el pánel frontal.

Paro: Es otro botón del pánel frontal que permite al operador detener el funcionamiento de la máquina en cualquier momento. El funcionamiento continúa oprimiendo el botón de arranque.

Restablecer: Es un botón que está conectado directamente al circuito de RESET de la tarjeta principal, por lo tanto, no se controla en la tarjeta de interfaz. Este botón se encuentra colocado en el pánel frontal.

Emergencia: Este botón está conectado directamente al circuito de vigilancia de la tarjeta principal, tampoco es controlado por la tarjeta de interfaz.

Los botones y switches son de tipo "normalmente cerrados", de manera que, cuando están accionados, esa entrada recibe un 1 lógico. Es decir, se utilizó lógica positiva. El circuito de interfaz para los botones y microswitches es el siguiente:

Tarjeta de interfaz

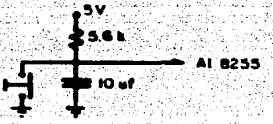


FIGURA 7.1

El circuito de interfaz para los fotodetectores es el siguente:

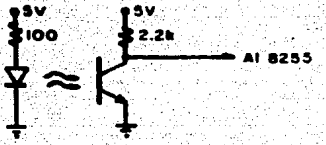


FIGURA 7.2

Los fotodetectores también fueron alambrados para lógica positiva. Cuando se interrumpe la llegada del haz infrarrojo a la base del fototransistor, la entrada recibe un 1 lógico.

Indicadores:

Los elementos indicadores sirven al operador para conocer el estado de operación de la máquina. Existen 8 elementos indicadores. Los primeros siete de la lista son "leds" y el octavo es un "buzzer" (Sonalert)

NORMAL: Indica que la máquina está funcionando normalmente.

CHAROLA: Indica que la máquina está esperando la llegada de una charola.

PAPEL: Indica que se agotó la reserva de papel en el depósito.

FALLA: Indica que el control detectó alguna situación de falla en el funcionamiento de la máquina. Cuando este led se enciende, se detiene la operación del equipo.

PARO: Indica que se ha oprimido el botón de paro y que por eso se encuentra detenida la máquina.

EMERGENCIA: Indica que se ha oprimido el botón de emergencia. Este led está conectado a la salida HALT del procesador, por lo tanto, no es controlado por la tarjeta de interfaz.

ENERGIA: Se enciende al encender la máquina. Indica la presencia de energía eléctrica. No es controlado por la tarjeta de interfaz.

BEEP: Se utiliza para generar alarmas audibles en caso de falta de papel, falla o emergencia.

El circuito para encender los leds es el siguente:

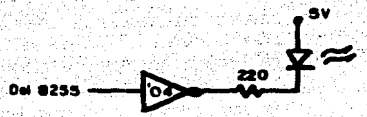


FIGURA 7.3

El circuito para encender la bocina se muestra a continuación:

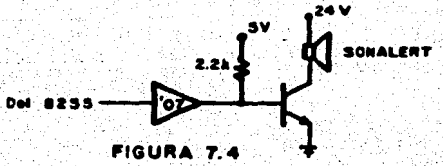


FIGURA 7.4

Actuadores:

Existen dos tipos de elementos actuadores: Motores eléctricos y un solenoide. Los circuitos para cada uno de los actuadores son diferentes por lo que se describirán separadamente.

Motores de movimiento vertical y horizontal: Dado que estos motores deben girar en ambos sentidos, se requieren tres bits del puerto 8255 para controlarlos. El circuito de interfaz se muestra en la figura 7.5.

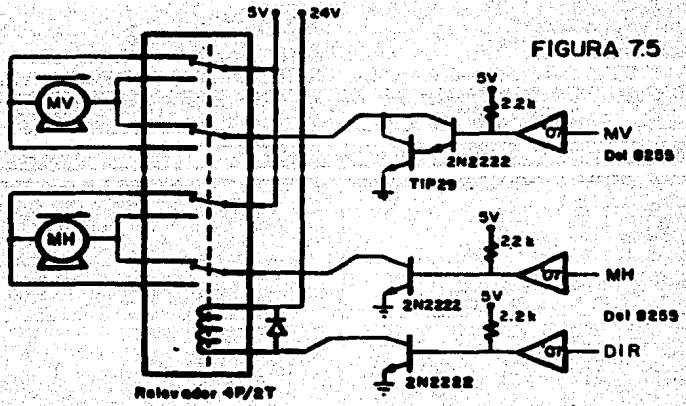
El motor vertical se enciende con la señal MV que es amplificada en corriente por medio de un buffer de colector abierto (74LS07). La salida del buffer se conecta a un par darlington ya que este motor llega a consumir hasta 1 ampere bajo carga.

El motor horizontal se enciende con la señal MH que es amplificada de la misma forma. En este caso no es necesario un par darlington ya que el motor consume 70 mA máximo.

El control del sentido de giro se hace por medio de un relevador 4P/2T. Cuando la señal DIR = 0, el transistor que acciona la bobina del relevador se encuentra en corte por lo que el relevador no se

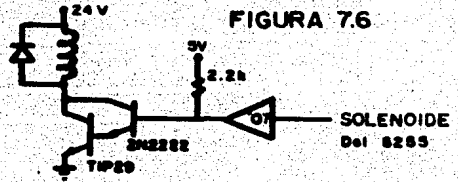
acciona. En este estado, si alguna de las señales MV o MH está presente, el sentido de la corriente a través de los motores se realiza como se indica en las flechas de la figura 7.5. Si la señal DIR = 1, el relevador se acciona invirtiendo el sentido de la corriente y, por lo tanto, el sentido de giro de los motores.

FIGURA 7.5



Solenoide del expulsor de charolas: El circuito para accionar el solenoide se muestra en la figura 7.6. Se requiere de un par darlington ya que el solenoide consume hasta 500 mA.

FIGURA 7.6



Motor del ventilador de vacío: Con este ventilador, acoplado a la tolva, se simula la bomba de vacío de la máquina. Con el fin de poder controlar la presión de succión se implementó el circuito de la figura 7.7. La señal VENT controla el encendido del motor del ventilador. La velocidad del motor y, por lo tanto, la presión de vacío se controla por medio del potenciómetro de 5K que se encuentra colocado en el panel frontal.

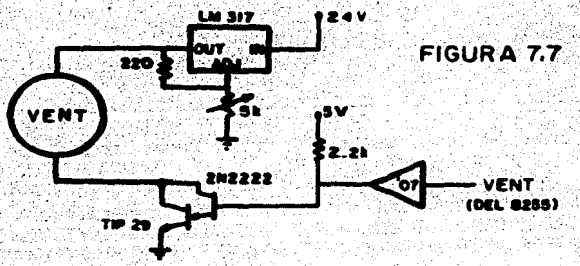


FIGURA 7.7

7.2 Asignación de puertos del 8255:

Como se mencionó en el capítulo cuatro, el 8255 se configuró de manera que se tuvieran 12 líneas de entrada y 12 de salida. El puerto A y la parte alta del puerto C se definieron de entrada y el puerto B y la parte baja del puerto C, de salida. Los elementos sensores se encuentran conectados a las entradas y los indicadores y actuadores, a las salidas. La distribución se hizo de la siguiente forma:

Puerto A: (entrada)

PA_0 <= Límite horizontal
PA_1 <= Límite superior
PA_2 <= Límite de engomado
PA_3 <= Límite inferior
PA_4 <= Papel
PA_5 <= Charola
PA_6 <= Expulsor

Puerto C parte alta: (entrada)

PC_4 <= Arranque
PC_5 <= Paro

Puerto B: (salida)

PB_0 => MH
PB_1 => MV
PB_2 => DIR
PB_3 => VENT
PB_4 => BEEP
PB_5 => SOLENOIDE

Puerto C parte baja: (salida)

PC_0 => PAPEL
PC_1 => CHAROLA
PC_2 => PARO
PC_3 => FALLA/NORMAL

Tarjeta de interfaz

En el apéndice C se presenta una copia del programa de control de la máquina empapeladora.

Conclusiones

A lo largo de este proyecto surgieron muchas conclusiones de diversos tipos y referentes a varios aspectos. Las conclusiones obtenidas sobre el desarrollo de circuitos se encuentran presentes, tácita o explícitamente, a lo largo del texto. Por lo tanto, expresaremos aquellas sacadas del desarrollo del proyecto. Las primeras, conclusiones. Las segundas, inquietudes.

La técnica de "desarrollo por prototipos" es quizá la más adecuada para la realización de proyectos nuevos. Un nuevo diseño no surge de la noche a la mañana y a la primera. Es necesario analizar todas las alternativas e inclusive desarrollar aquellas que se presenten viables. Del análisis y el desarrollo surgen valiosos criterios que sirven, no solo para el problema actual, sino como experiencia para problemas futuros. Fue de la práctica de donde obtuvimos los elementos necesarios para poder producir, finalmente, una solución adecuada a las necesidades. Es cierto que todo proyecto se inicia en el papel, pero la decisión sobre qué camino tomar no debe hacerse sobre el papel. Es necesario llevar la teoría a la práctica para realmente saber cual es el camino correcto. En nuestro caso, en el papel se quedaron las opciones más rudimentarias (lógica combinacional, lógica secuencial, etc.), pero tuvieron que llevarse a la práctica las dos alternativas que, por varias razones, en el papel se presentaban funcionalmente iguales. Fue la práctica la que reveló la gran desigualdad entre ambas.

Comprobamos que es posible que la Universidad desarrolle proyectos para la industria. Sin embargo, creemos que no se les da la difusión necesaria. Los industriales mexicanos no confían en la Universidad, por lo que prefieren importar la tecnología en vez de desarrollarla, a cualquier costo. Si la Universidad destinara recursos para el desarrollo de proyectos industriales y, además, les diera más difusión, poco a poco lograríamos establecer una relación Universidad-industria similar a la que se da en otros países industrializados. Además, la participación de estudiantes en estos proyectos crearía ingenieros completos y con una visión más amplia de la realidad industrial.

Los trabajos de tesis son, por lo general, proyectos puramente académicos. Una vez pasado el examen profesional, los proyectos se quedan almacenados en las bibliotecas de familiares y amigos. Realmente es un desperdicio de tiempo y de recursos. Así como el servicio

social, el trabajo de tesis debería tener fines más trascendentes que el de aprobar el examen profesional. Creemos que la Universidad, especialmente la Facultad de Ingeniería, podría proponer proyectos de tesis enfocados a resolver problemas reales de la industria. Con el tiempo, todos los participantes en estos proyectos saldrían ganando: La Universidad obtendría recursos y elevaría el nivel de sus profesionistas, El Estudiante complementaría sus conocimientos y ampliaría su criterio hacia la solución de los problemas, La industria empezaría a producir tecnología de fabricación nacional.

En lo que se refiere al texto del trabajo de tesis, concluimos que debe ser una presentación breve del proyecto y no un libro de texto. En el trabajo se deben expresar los descubrimientos, los diseños ingeniosos, los criterios, las experiencias; es decir, lo que no aparece en los libros de texto y que solo se aprende con la experiencia.

Por ejemplo, casi todos los textos de tesis de ingeniería electrónica dedican todo un capítulo al diseño de la fuente de poder. La teoría sobre el diseño de fuentes se puede encontrar en libros y manuales. En cambio, se olvidan de presentar los criterios de selección de componentes y técnicas de desarrollo.

En resumen, el texto del trabajo de tesis debería dedicar menos espacio a la teoría y más espacio a la experiencia.

Apéndice A

Firmware Programador de EPROMs

```
                CD F7 01                CALL    DESPA
                4F                      LD      C,A
                C5                      PUSH    BC      ;Carga IX con BC
                DD E1                   POP     IX
1074            FD 21 9C 16     PANT4   LD      IY,169C ;Carga dir 4a. pantalla
                CD 88 11                CALL    DESPL
                CD 60 04                CALL    CAPTH
                CD F7 01                CALL    DESPA
                67                      LD      H,A
                CD 60 04                CALL    CAPTH
                CD F7 01                CALL    DESPA
                6F                      LD      L,A
                FD 21 7A 16             LD      IY,167A ;Carga dir 3a. pantalla
                CD 88 11                CALL    DESPL
                CD 60 04                CALL    CAPTH
                CD F7 01                CALL    DESPA
                47                      LD      B,A
                CD 60 04                CALL    CAPTH
                CD F7 01                CALL    DESPA
                4F                      LD      C,A
109E            FD 21 59 16     CORRECT LD      IY,1659 ;Carga dir 6a. pantalla
                CD 88 11                CALL    DESPL
                CD 16 04                CALL    CAPTAS
                CD 1E 04                CALL    DESPAS
                FE 53                   CP      53      ;Valida información
                C2 2E 10                JPNZ    MENU    ;Salta si <> S
                7B                      LD      A,E     ;Recupera opción
                FE 56                   CP      56      ;Salta a opción
                CA C3 10                JPZ     VERIFI  ;correspondiente
                FE 4C                   CP      4C
                CA 26 11                JPZ     LECTURA
                FE 50                   CP      50
                CA 4D 11                JPZ     ESCRIT
                C3 2E 10                JP      MENU
10C3            D9              VERIFI  EXX             ;*VERIFICAR*
                06 17                   LD      B,17    ;Inicializa contador
                D9                      EXX             ;de paginación
                7C                      LD      A,H     ;Agrega código tipo EPROM
                B2                      OR      D       ;a parte alta dirección
                67                      LD      H,A     ;inicial a leer
                78                      LD      A,B     ;Agrega código tipo EPROM
                B2                      OR      D       ;a parte alta dirección
                47                      LD      B,A     ;final a leer
                18 01                   JR      INTVFY  ;No incrementa la. vez
10CF            23              LOOPVFY INC     HL      ;Incrementa cont.dir.
10D0            7C              INTVFY  LD      A,H     ;Carga latches
                D3 52                   OUT     52,A
                7D                      LD      A,L
                D3 53                   OUT     53,A
10D6            DB 50           RELEE1  IN      A,50    ;Lee EPROM
                5F                      LD      E,A     ;Salva temporalmente A
                DB 50                   IN      A,50    ;Vuelve a leer EPROM
                BB                      CP      E       ;Compara lecturas
                20 F8                   JRNZ    RELEE1  ;Sigue leyendo hasta iguales
                FE FF                   CP      FF      ;Valida dato
```

```
                20 15                  JRNZ    ERROR    ;Salta a ERROR si no FFH
1082            79          CONT       LD      A,C      ;Compara dirección inicial
                95                     SUB     L        ;con dirección final
                20 E9                  JRNZ    LOOPVFY
                78                     LD      A,B
                94                     SUB     H
                20 E5                  JRNZ    LOOPVFY
                FD 21 2F 16            LD      IY,162F  ;Carga dir 7a. pantalla
                CD 88 11               CALL    DESPL
                CD 16 04               CALL    CAPTAS
                C3 2E 10               JP      MENU     ;Salta MENU si dir final
10F7            3E 0D       ERROR      LD      A,0D     ;Despliega dirección y
                CD 1E 04               CALL    DESPAS   ;contenido
                3E 0A                  LD      A,0A
                CD 1E 04               CALL    DESPAS
                CD EF 01               CALL    DESPHL
                3E 3A                  LD      A,3A
                CD 1E 04               CALL    DESPAS
                3E 20                  LD      A,20
                CD 1E 04               CALL    DESPAS
                78                     LD      A,E
                CD F7 01               CALL    DESPA
                D9                     EXX              ;Checa paginación
                10 0E                  DJNZ    REGRESA  ;Página incompleta
1115            CD 16 04    INVALI     CALL    CAPTAS   ;Página completa
                FE 03                  CP      03       ;Regresa a Menu si Ctrl-C
                CA 2E 10               JPZ     MENU
                FE 20                  CP      20       ;Detiene programa hasta
                20 F4                  JRNZ    INVALI   ;leer barra esp.
                06 17                  LD      B,17     ;Reinicializa contador
1123            D9          REGRESA    EXX
                18 BC                  JR      CONT     ;Continua
1126            7C          LECTURA    LD      A,H      ;*LECTURA*
                B2                     OR      D
                67                     LD      H,A
                78                     LD      A,B
                B2                     OR      D
                47                     LD      B,A
                18 03                  JR      INTLECT
112E            DD 23       LOOPLEC    INC     IX
                23                     INC     HL
1131            7C          INTLECT    LD      A,H
                D3 52                  OUT     52,A
                7D                     LD      A,L
                D3 53                  OUT     53,A
1137            DB 50       RELEE2     IN      A,50
                5F                     LD      E,A
                DB 50                  IN      A,50
                BB                     CP      E
                20 F8                  JRNZ    RELEE2
                DD 77 00               LD      (IX),A   ;Carga dato en RAM
                79                     LD      A,C
                95                     SUB     L
                20 E8                  JRNZ    LOOPLEC
                78                     LD      A,B
```

```
                94                      SUB     H
                20 E4                   JRNZ    LOOPLEC
                C3 2E 10                JP      MENU
  114D          7C            ESCRIT    LD      A,H       ;*ESCRITURA*
                F6 80                   OR      80        ;Enciende PB7
                B2                      OR      D
                67                      LD      H,A
                78                      LD      A,B
                F6 80                   OR      80
                B2                      OR      D
                47                      LD      B,A
                18 03                   JR      INTESC
  1159          DD 23         LOOPESC   INC     IX
                23                      INC     HL
  115C          DD 7E 00      INTESC    LD      A,(IX)    ;Lee dato de RAM
                FE FF                   CP      FF        ;Si FFH no graba dato
                20 13                   JRNZ    FINESC
                D3 51                   OUT     51,A      ;Carga dato en Pto.
                7C                      LD      A,H
                D3 52                   OUT     52,A
                7D                      LD      A,L
                D3 53                   OUT     53,A
  116B          D9            RETARDO   EXX               ;Retardo 50 ms
                11 E0 2F                LD      DE,2FE0
  116F          1D            INTRET    DEC     E
                20 FD                   JRNZ    INTRET
                15                      DEC     D
                20 FA                   JRNZ    INTRET
                D9                      EXX
  1176          79            FINESC    LD      A,C       ;Compara direcciones
                95                      SUB     L
                20 DF                   JRNZ    LOOPESC
                78                      LD      A,B
                94                      SUB     H
                20 DB                   JRNZ    LOOPESC
  117E          CB BC         RESETB    RES     (7),H     ;Apaga PB7
                CB B8                   RES     (7),B
                7C                      LD      A,H
                D3 52                   OUT     52,A
                C3 2E 10                JP      MENU
  1188          FD 7E 00      DESPL     LD      A,(IY)    ;*DESPLIEGA*
                FE 03                   CP      03        ;Compara <EOT>
                C8                      RETZ
                CD 1E 04                CALL    DESPAS    ;Despliega caracter
                FD 2B                   DEC     IY        ;ASCII si no <EOT>
  1193          18 F3                   JR      DESPL
```

"Hexadecimal Dump" del sistema de pantallas:

```
1600  FF FF FF FF   FF FF FF 03   72 61 75 6E   69 74 6E 6F   ........reunitno
1610  63 20 61 72   61 70 20 61   6C 63 65 74   20 72 65 69   c arap alcet rei
1620  78 71 6C 61   78 63 20 61   6D 69 72 70   4F 0A 0A 0D   uqlauc amirpO...
1630  03 20 3F 20   29 4E 2F 53   20 20 73 6F   74 61 64 20   . ? )N/S( sotad
1640  73 6F 6C 20   73 6F 74 63   65 72 72 6F   63 20 6E 61   sol sotcerroc na
1650  74 73 45 20   20 20 0A 0A   0A 0D 03 20   20 20 4D 4F   tsE .... MO
1660  02 00 45 20   6C 61 6E 69   66 20 6E 6F   69 63 63 65   RPE lanif noicce
1670  72 69 44 20   20 20 20 20   20 0A 0D 03   20 4D 4F 52   riD ... MOR
1680  50 45 20 6C   61 69 63 69   6E 69 20 6E   6F 69 63 63   PE laicini noicc
1690  65 72 69 44   20 20 20 20   20 20 0A 0D   0A 03 20 20   eriD ....
16A0  20 4D 41 52   20 6C 61 69   63 69 6E 69   20 6E 6F 69   MAR laicini noi
16B0  63 63 65 72   69 44 20 20   20 20 20 20   0A 0A 0D 3A   cceriD ...:
16C0  73 65 6E 6F   69 63 63 65   72 69 44 20   20 20 0A 0A   senoicceriD ..
16D0  0A 0D 03 20   3A 6E 6F 69   63 63 65 6C   65 53 20 20   ... :noicceleS
16E0  20 0A 0A 0D   72 61 63 69   66 69 72 65   56 20 2D 20   ...racifireV -
16F0  56 20 20 20   20 20 20 0A   0D 72 61 6D   61 72 67 6F   V ..ramargo
1700  72 50 20 2D   20 50 20 20   20 20 20 20   0A 0D 72 6F   rP - P ..ro
1710  74 69 6E 4D   4D 20 2D 20   4D 20 20 20   20 20 20 0A   tinoM - M .
1720  0D 72 65 65   4C 20 2D 20   4C 20 20 20   20 20 20 0A   .reeL - L .
1730  0D 4D 4F 52   50 45 20 6F   70 69 74 20   72 61 69 62   .MORPE opit raib
1740  6D 61 43 20   2D 20 43 20   20 20 20 20   20 0A 0A 0D   maC - C ...
1750  3A 73 65 6E   6F 69 63 6E   75 46 20 20   20 0A 0A 0A   :senoicnuF ...
1760  0D 03 20 3A   65 6F 69 63   63 65 6C 65   53 20 20 20   .. :noicceleS
1770  0A 0D 0A 34   36 37 32 20   2D 20 34 20   20 20 20 20   ...4672 - 4
1780  20 0A 0D 41   32 33 37 32   20 2D 20 33   20 20 20 20   ..A2372 - 3
1790  20 20 0A 0D   32 33 37 32   20 2D 20 32   20 20 20 20   ..2372 - 2
17A0  20 20 0A 0D   36 31 37 32   20 2D 20 31   20 20 20 20   ..6172 - 1
17B0  20 20 0A 0A   0D 3A 73 6F   70 69 54 20   20 20 0A 0A   ...:sopiT ..
17C0  0A 0D 2A 2A   20 47 4D 52   2F 46 53 47   20 73 27 4D   ..** GMR/FSG s'M
17D0  4F 52 50 45   20 72 6F 64   61 6D 61 72   67 6F 72 50   ORPE rodamargorP
17E0  20 2A 2A 0A   0D 0A 0A 0A   0A 0A 0A 0A   0A 0A 0A 0A    **............
17F0  0A 0A 0A 0A   0A 0A 0A 0A   0A 0A 0A 0A   0A 0A 0A 0A   ................
```

Apéndice B

Programa cargador/reubicador

```
MVALLET SYSTEMS 8085/Z80 ASSEMBLER -  VERSION 1.05M  SERIAL #00410
SOURCE FILE NAME: LOADER.ASM                                    PAGE  1

0000                    ;
0000                    ; PROGRAMA CARGADOR/RELOCALIZADOR
0000                    ;
0000                    ; ARCHIVOS ENSAMBLADOS EN FORMATO INTEL "HEX"
0000                    ;
0000                    ; SDD/RMG V87.01                (ABRIL '87)
0000                    ;
0000                    ;
0000                    ; RUTINAS DEL MONITOR
0000                    ;
0460            ASCHEX  EQU   0460H             ;Lee 2 ASCII convierte a HE
                X
0416            CAPTAS  EQU   0416H             ;Lee 1 ASCII
0486            CRLF    EQU   0486H             ;Envia <CR-LF> a terminal
0490            CRLFLF  EQU   0490H             ;Envia <CR-LF-LF> a termina
                l
041E            DESPAS  EQU   041EH             ;Despliega cont. ASCII en A
01EF            DESPHL  EQU   01EFH             ;Despliega cont. HEX en HL
053B            DESPL   EQU   053BH             ;Despliega texto
0000                    ;
0000                    ;
0000                    ; DIRECCION DE CARGA TEMPORAL (SCRATCH RAM)
0000                    ;
2800            LDADDR  EQU   2800H
0000                    ;
0000                    ;
1197                    ORG 1197H
                        .Z80
1197                    ;
1197                    ;
1197                    ; DIRECCIONES DE TABLAS
1197                    ;
2000            CHKSUM  EQU   2000H             ;Almacen temporal CHECKSUM
2001            BYTECNTR EQU  CHKSUM+1          ;Contador de bytes de datos
2003            LSADDR  EQU   CHKSUM+3          ;Direccion menos signif.
2005            MSADDR  EQU   CHKSUM+5          ;Direccion mas signif.
2007            NXADDR  EQU   CHKSUM+7          ;Dir. siguiente registro
2009            DIFADDR EQU   CHKSUM+9          ;Dif. de dir. entre registr
                os
200B            ABSLSADD EQU  CHKSUM+11         ;Dir. menos signif. absolut
                a
200D            ABSMSADD EQU  CHKSUM+13         ;Dir. mas signif. absoluta
1197                    ;
1197                    ;
1197                    ; TEXTOS
1197                    ;
1197 030A0A0A   STRTXT: DB    3,10,10,10,10,13
1198 3E542054           DB    '>T TLA< amirp0  :YDAER'
119B 20202020           DB    '                '
119E 20202020           DB    '                '
11A1 0A0A0A0D           DB    10,10,10,13
```

```
11D3 29373827           DB    ')78' RBA  10.78V GMR/DD8("
11EC 20202020           DB    '
11F6 20202020           DB    '
1207 0A0A0D             DB    10,10,13
120A 3D3D3D3C           DB    '===< "XEH" letnI otamroF ne sovihcrA'
122E 20656420           DB    ' ed rodazilacoleR/rodagraC >==='
1253 0A0A0A0A           DB    10,10,10,10
1257 0D         PGMHDR: DB    13
1258 030A0A0D           DB    3,10,10,13
125C 2D2D2D3C           DB    '---< NOICUCEJE ED NIF >---'
1276 20202020           DB    '                          '
128F 20         ENDPGHDR: DB  ' '
1290 03                 DB    3
1291 28202020           DB    '(   :sodibiceR setyB #'
12A7 20202020           DB    '                      '
12B9 20         ENDPNUR1: DB  ' '
12BA 03                 DB    3
12BB 28202020           DB    '(   :noicazinagrO .riD'
12D1 20202020           DB    '                      '
12E4 0A0D29             DB    10,13,')'
12E7 48         ENDPNUR2: DB  'H'
12E8 03                 DB    3
12E9 28203A6E           DB    '( :noicazilacoleR .riD'
12FF 20202020           DB    '                      '
1312 0A0D29             DB    10,13,')'
1315 48         ENDPNDR3: DB  'H'
1316 03                 DB    3
1317 28202D20           DB    '( - )'
131C 48         ENDPNDR4: DB  'H'
131D 0329               DB    3,')'
131F 48         ENDPNDR5: DB  'H'
1320 03                 DB    3
1321 464F4520           DB    'FOE detcepxen'
132E 55         EOFTXT: DB    'U'
132F 03                 DB    3
1330 79726F6D           DB    'yromeM daB ro -fo tuO'
1345 20         MEMTXT: DB    ' '
1346 03                 DB    3
1347 4D555348           DB    'MUSKCEH'
134E 43         CHSUMTXT: DB  'C'
134F 03                 DB    3
1350 74616D72           DB    'tamroF eliF llegel'
1362 49         FORMTXT: DB   'I'
1363 03                 DB    3
1364 203A726F           DB    ' :rorrE'
1368 20202020           DB    '
137F 20         ERRTXT: DB    ' '
1380                    ;
1380                    ;
1380                    ; PROGRAMA PRINCIPAL
1380                    ;
```

```
1380 FB215712  INICIO:  LXI   Y,PUPMUK
1384 CD3B0B             CALL  DELPL
1387 DD210020           LXI   X,CHKSUM
138B 110000             LXI   D,0000H
138E ED5B3012           XUAD  BYTECNTR
1392 210020             LXI   H,LOADDR
1395 220920             SHLD  ABSLOADD
1398 E5                 PUSH  H
1399 CD1604    BUFERCH: CALL  CAPTAB
139C FE3A               CPI   ':'
139E 20F9               JRNZ  BUFERCH
13A0 CD4004             CALL  ASCHEX
13A3 DD7700             MUV   O(X),A
13A6 4F                 MOV   C,A
13A7 0600               MVI   B,00H
13A9 09                 DAD   B
13AA 2B                 DCX   H
13AB 220D20             SHLD  ABSMSADD
13AE C5                 PUSH  B
13AF CD4004             CALL  ASCHEX
13B2 67                 MOV   H,A
13B3 DD8600             ADD   O(X)
13B6 DD7700             MOV   O(X),A
13B9 CD4004             CALL  ASCHEX
13BC C1                 POP   B
13BD 6F                 MOV   L,A
13BE DD8600             AUD   O(X)
13C1 DD7700             MUV   O(X),A
13C4 220D20             SHLD  LSADDR
13C7 E5                 PUSH  H
13C8 09                 DAD   B
13C9 2B                 DCX   H
13CA 220B20             SHLD  MSADDR
13CD E1                 POP   H
13CE C5        ENTERCH: PUSH  B
13CF CD4004             CALL  ASCHEX
13D2 C1                 POP   B
13D3 FE01               CPI   01H
13D5 CAAB15             JZ    EOFERR
13D8 FE00               CPI   00H
13DA C2BE15             JNZ   FORMERR
13DD E5        LOOP:    PUSH  H
13DE 2A0120             LHLD  BYTECNTR
13E1 09                 DAD   B
13E2 220120             SHLD  BYTECNTR
13E5 E1                 POP   H
13E6 09                 DAD   B
13E7 220720             SHLD  NXADDR
13EA 41                 MUV   B,C
13EB E1                 POP   H
13EC C5        LOOATA:  PUSH  B
```

```
136D CD6004             CALL    ASCHEX
13F0 C1                 POP     B
13F1 77                 MOV     M,A
13F2 4F                 MOV     C,A
13F3 7E                 MOV     A,M
13F4 B9                 CMP     C
13F5 C28B15             JNZ     MEMERR
13F8 DD8600             ADD     O(X)
13FB DD7700             MOV     O(X),A
13FE 23                 INX     H
13FF 10EB               DJNZ    LDDATA
1401 DD7E00             MOV     A,O(X)
1404 ED44               NEG
1406 47                 MOV     B,A
1407 C5                 PUSH    B
1408 CD600A             CALL    ASCHEX
140B C1                 POP     B
140C B8                 CMP     B
140D C23315     ERRSRCH: JNZ    CHSUMERR
1410 CD1604             CALL    CAPTAS
1413 FE0D               CPI     ODH
1415 C2BE15             JNZ     FORMERR
1418 CD1604             CALL    CAPTAS
141B FE0A               CPI     OAH
141D C2BE15             JNZ     FORMERR
1420 CD1604             CALL    CAPTAS
1423 FE3A               CPI     ':'
1425 C2BE15             JNZ     FORMERR
1428 E5         NEXTREG: PUSH   H
1429 C5                 PUSH    B
142A CD6004             CALL    ASCHEX
142D C1                 POP     B
142E DD7700             MOV     O(X),A
1431 4F                 MOV     C,A
1432 0600               MVI     B,00H
1434 C5                 PUSH    B
1435 CD6004             CALL    ASCHEX
1438 C1                 POP     B
1439 67                 MOV     H,A
143A DD8600             ADD     O(X)
143D DD7700             MOV     O(X),A
1440 C5                 PUSH    B
1441 CD6004             CALL    ASCHEX
1444 C1                 POP     B
1445 6F                 MOV     L,A
1446 DD8600             ADD     O(X)
1449 DD7700             MOV     O(X),A
144C C5         EOFSRCH: PUSH   B
144D CD6004             CALL    ASCHEX
1450 C1                 POP     B
1451 FE01               CPI     01H
```

```
1483 285E                  JNZ   LOF
1485 FE00                  CPI   00H
1487 C29E15                JNZ   FURMORE
148A E5          CHLSADDR: PUSH  H
148B ED5B0320              LDED  LSADDR
148F ED52                  DSBC  D
1491 E1                    POP   H
1492 3015                  JRNC  CHMSADDR
1494 E5                    PUSH  H
1495 D5                    PUSH  D
1496 EB                    XCHG
1497 ED52                  DSBC  D
1499 EB                    XCHG
149A 220920                LHLD  ABSLSADD
149D ED52                  DSBC  D
149F 220920                SHLD  ABSLSADD
14A2 D1                    POP   D
14A3 E1                    POP   H
14A4 220320                SHLD  LSADDR
14A7 181F                  JMPR  CHNXADD1
14A9 E5          CHMSADDR: PUSH  H
14AA ED5B0520              LDED  MSADDR
14AE 09                    DAD   B
14AF 28                    DCX   H
14B0 E5                    PUSH  H
14B1 ED52                  DSBC  D
14B3 E1                    POP   H
14B4 3811                  JRC   CHNXADDR
14B6 E5                    PUSH  H
14B7 D5                    PUSH  D
14B8 ED52                  DSBC  D
14BA EB                    XCHG
14BB 2A0D20                LHLD  ABSMSADD
14BE 19                    DAD   D
14BF 220D20                SHLD  ABSMSADD
14C2 D1                    POP   D
14C3 E1                    POP   H
14C4 220520                SHLD  MSADDR
14C7 E1          CHNXADDR: POP   H
14C8 E5          CHNXADD1: PUSH  H
14C9 ED5B0720              LDED  NXADDR
14CD ED52                  DSBC  D
14CF 220920                SHLD  DIFADDR
14D2 E1                    POP   H
14D3 CAD013                JZ    LUOP
14D6 D1                    POP   D
14D7 E5                    PUSH  H
14D8 2A0920                LHLD  DIFADDR
14DB EB                    XCHG
14DC 19                    DAD   D
14DD D1                    POP   D
```

```
14AE E5                 PUSH  H
14AF EB                 XCHG
14B0 C3DD13             JMP   LOOP
14B3 DD7E00     EOF:    MOV   A,0(X)
14B6 3C                 INR   A
14B7 ED44               NEG
14B9 47                 MOV   B,A
14BA C5                 PUSH  B
14BB CD6004             CALL  ASCHEX
14BE C1                 POP   B
14BF B8                 CMP   B
14C0 C28315             JNZ   CHSUMERR
14C3 EB                 XCHG
14C4 ED4B0320           LBCD  LSADDR
14C8 2A0520             LHLD  MSADDR
14CB ED42               DSBC  B
14CD 23                 INX   H
14CE E5                 PUSH  H
14CF C1                 POP   B
14D0 2A0B20             LHLD  ABSLSADD
14D3 E5                 PUSH  H
14D4 ED52               DSBC  D
14D6 E1                 POP   H
14D7 2B3Y               JRZ   ENDPGM
14D9 3020               JRNC  MOVUP
14DB EB         MOVDN:  XCHG
14DC 2B                 DCX   H
14DD 09                 DAD   B
14DE EB                 XCHG
14DF 2A0D20             LHLD  ABSMSADD
14E2 ED530D20           SDED  ABSMSADD
14E6 CD6A15     LDECR:  CALL  ERMEMSCH
14E9 1B                 DCX   D
14EA 2B                 DCX   H
14EB 0B                 DCX   B
14EC 3E00               MVI   A,00H
14EE 81                 ADD   C
14EF 20F5               JRNZ  LDECR
14F1 80                 ADD   B
14F2 20F2               JRNZ  LDECR
14F4 13                 INX   D
14F5 ED530B20           SDED  ABSLSADD
14F9 1817               JMPR  ENDPGM
14FB ED530B20   MOVUP:  SDED  ABSLSADD
14FF CD6A15     LINCR:  CALL  ERMEMSCH
1502 13                 INX   D
1503 23                 INX   H
1504 0B                 DCX   B
1505 3E00               MVI   A,00H
1507 81                 ADD   C
1508 20F5               JRNZ  LINCR
```

```
150A 80                    ADD   M
150B 20F2                  JRNZ  LINCR
150D 1B                    DCX   D
150E ED830D20              SDED  ABSMSADD
1512 FD21BF12  ENDPGM:     LXI   Y,ENDPGHUK
1516 CD3B05                CALL  DESPL
1519 FD21BY12              LXI   Y,ENDPNOR1
151D CD3B05                CALL  DESPL
1520 2A0120                LHLD  BYTECNTR
1523 CDEF01                CALL  DESPHL
1526 FD21E712              LXI   Y,ENDPNOR2
152A CD3B05                CALL  DESPL
152D 2A0320                LHLD  LSADDR
1530 CDEF01                CALL  DESPHL
1533 FD211C13              LXI   Y,ENDPNOR4
1537 CD3B05                CALL  DESPL
153A 2A0520                LHLD  MSADDR
153D CDEF01                CALL  DESPHL
1540 FD211513              LXI   Y,ENDPNOR3
1544 CD3B05                CALL  DESPL
1547 2A0820                LHLD  ABSLSADD
154A CDEF01                CALL  DESPHL
154D FD211C13              LXI   Y,ENDPNOR4
1551 CD3B05                CALL  DESPL
1554 2A0020                LHLD  ABSMSADD
1557 CDEF01                CALL  DESPHL
155A FD211F13              LXI   Y,ENDPNOR5
155E CD3B05                CALL  DESPL
1561 CDD515                CALL  CTRL_Z
1564 CD8A04                CALL  CRLF
1567 C39004                JMP   CRLFLF
156A 7E        ERMEMSCH:   MOV   A,M
156B 12                    STAX  D
156C C5                    PUSH  B
156D 4F                    MOV   C,A
156E 1A                    LDAX  D
156F B9                    CMP   C
1570 EB                    XCHG
1571 C28B15                JNZ   MEMERR
1574 EB                    XCHG
1575 C1                    POP   B
1576 C9                    RET
1577 3E03      ENDPGERR:   MVI   A,03H
1579 CD1E04                CALL  DEBPAS
157C FD21BF12              LXI   Y,ENDPGHUK
1580 CD3B05                CALL  DESPL
1583 FD217F13              LXI   Y,ERRTXT
1587 CD3B05                CALL  DESPL
158A C9                    RET
158B CD7715    MEMERR:     CALL  ENDPGERR
158E 3E28                  MVI   A,'('
```

```
1590 CD1604                CALL   DESPAS
1593 CDEF01                CALL   DESPHL
1596 3E48                  MVI    A,'H'
1598 CD1604                CALL   DESPAS
159B 3E29                  MVI    A,')'
159D CD1604                CALL   DESPAS
15A0 FD214513              LXI    Y,MEMTXT
15A4 FDE5                  PUSH   Y
15A6 181F                  JMPR   ERRDISPL
15A8 CD7715    EOFERR:     CALL   ENDPGERR
15AB FD212E13              LXI    Y,EOFTXT
15AF FDE5                  PUSH   Y
15B1 1814                  JMPR   ERRDISPL
15B3 CD7715    CHSUMERR:   CALL   ENDPGERR
15B6 FD214E13              LXI    Y,CHNUMTXT
15BA FDE5                  PUSH   Y
15BC 1809                  JMPR   ERRDISPL
15BE CD7715    FORMERR:    CALL   ENDPGERR
15C1 FD216213              LXI    Y,FORMTXT
15C5 FDE5                  PUSH   Y
15C7 FDE1      ERRDISPL:   POP    Y
15C9 CD3605                CALL   DESPL
15CC CDD515                CALL   CTRL_Z
15CF CD8604                CALL   CRLF
15D2 C39004                JMP    CRLFLF
15D5 06FF      CTRL_Z:     MVI    B,0FFH
15D7 48                    MOV    C,B
15D8 1620                  MVI    D,20H
15DA 05        CTRL_Z_1:   DCR    B
15DB 20FD                  JRNZ   CTRL_Z_1
15DD 0D                    DCR    C
15DE 20FA                  JRNZ   CTRL_Z_1
15E0 15                    DCR    D
15E1 20F7                  JRNZ   CTRL_Z_1
15E3 C9                    RET
15E4                       ;
15E4                       ;
1197                       END    1197H
```

--- SYMBOL TABLE ---

ABLSADD	200B	DEBPL	0536	FORMERR	15BE
ABSMSADD	200D	DIFADDR	2009	FORMTXT	1362
ABCHEX	0460	EFERSRCH	13C6	INICIO	1360
BDFSRCH	1399	ENDPGERR	1577	LDADDR	2800
BYTECNTR	2001	ENDPGHDR	126F	LDDATA	13EC
CAPTAB	0416	ENDPGM	1512	LDECR	14E6
CHKSUM	2000	ENDPNUM1	1239	LINCR	14FF
CHLSADDR	145A	ENDPNUM2	1267	LOOP	13DD
CHMSADDR	1479	ENDPNUM3	1315	LSADDR	2003
CHNXADD1	1498	ENDPNUM4	131C	MEMERR	15BB
CHNXADDR	1477	ENDPNUM5	131F	MEMTXT	1345
CHSUMERR	15B3	EOF	1483	MOVDN	14DB
CHSUMTXT	134E	EOFERR	15A8	MOVUP	14F9
CRLF	0486	EOFSRCH	144C	MSADDR	2005
CRLFLF	0490	EOFTXT	1326	NEXTREG	1428
CTRL_Z	15D5	ERMSRCH	1566	NXADDR	2007
CTRL_Z_1	15DA	ERRD1SPL	1507	PGMHDR	1257
DESPAR	041E	ERRSRCH	140D	STRTXT	1197
DESPHL	01EF	ERRTXT	137F		

***** NO ERRORS DETECTED *****

Apéndice C

Programa de control

```
AVOCET SYSTEMS 8085/Z80 ASSEMBLER - VERSION 1.05M  SERIAL #00410
SOURCE FILE NAME: BITPROU.ASM                                  PAGE  1

0000                    ;PROGRAMA DE CONTROL DE LA MAQUINA EMPAPELADORA
0000                    ;
0000                    ;ABRIL, 1987                                  RMG/VGV
0000                    ;
0000                    ;
0000                    ;ZONA DE DECLARACIONES
0000                    ;
0090    INITB255  EQU   90H             ;MODO DE OPERACION 8255
0000    PORTA     EQU   00H             ;SENSORES
0001    PORTB     EQU   PORTA+1         ;ACTUADORES
0002    PORTC     EQU   PORTA+2         ;SENSORES/INDICADORES
0003    CONTROL   EQU   PORTA+3
0004    WD        EQU   04H             ;WATCHDOG
0000                    ;
0000                    ;
0000                    ;INICIALIZACION DE FUNCIONAMIENTO
0000                    ;
0000                    ORG   0000H
                        .Z80
0000                    ;
0000 CDE502 INICIO:     CALL  DISEG           ;DELAY PARA ESTABILIZACION
0003 3E90               MVI   A,INITB255      ;INICIALIZACION DE PUERTOS
0005 D303               OUT   CONTROL
0007 ED56               IM1                   ;MODO 1 DE INTERRUPCIONES
0009 C30001             JMP   ORGTOLVA
000C                    ;
000C                    ;
000C                    ;RUTINA DE INTERRUPCION DEL WATCHDOG
000C                    ;(FALLAS EN EL EQUIPO)
000C                    ;
0038                    ORG   0038H
                        .Z80
0038                    ;
0038 3E00       FALLA:  MVI   A,00H           ;APAGA TODOS LOS MOTORES
003A CDDD02             CALL  OPORTB
003D CBC7               SET   0,A             ;ENCIENDE INDICADOR DE FALL
                A
003F CDE002             CALL  OPORTC
0042 3E20       BEEP:   MVI   A,20H           ;SUENA ALARMA
0044 CDDD02     BEEPLOOP: CALL  OPORTB
0047 CDF102             CALL  D20MS
004A CBAF               RES   5,A
004C CDDD02             CALL  OPORTB
004F CBEF               SET   5,A
0051 CDE902             CALL  D500MS
0054 CDDD02             CALL  OPORTB
0057 CDF102             CALL  D20MS
005A CBAF               RES   5,A
005C CDDD02             CALL  OPORTB
005F CBEF               SET   5,A
0061 CDE502             CALL  DISEG
```

```
0064 18DE                  JMPR  BEEPLOOP
0066                       ;
0066                       ;
0066                       ;RUTINA DE INTERRUPCION DE EMERGENCIA
0066                       ;
0066                       ORG   0066H
                           .Z80
0066                       ;
0066 3E20        EMERG:    MVI   A,20H          ;APAGA MOTORES
0068 CDDD02                CALL  OPORTB         ;SUENA ALARMA
006B 76                    HLT                  ;ENCIENDE INDICADOR EMERG.
006C                       ;
006C                       ;
006C                       ;PROGRAMA PRINCIPAL
006C                       ;
0100                       ORG   0100H
                           .Z80
0100                       ;
0100 3E00        ORGTOLVA: MVI   A,00H          ;ENCIENDE INDICADOR "NORMAL"

0102 CDE002                CALL  OPORTC
0105 CBEF                  SET   5,A            ;BEEP PRINCIPIO DE INICIALI

0107 CDDD02                CALL  OPORTB
010A CDE502                CALL  DISEG
010D CBAF                  RES   5,A
010F CDDD02                CALL  OPORTB
0112 CDB502                CALL  INPORTA
0115 CB4F                  BIT   1,A            ;LIMITE SUPERIOR ?
0117 CC2602                CZ    SUBTOLVA       ;NO, SUBE TOLVA
011A CDB502                CALL  INPORTA
011D CB47                  BIT   0,A            ;LIMITE HORIZONTAL ?
011F 2832                  JRZ   ORGCARRO       ;NO, REGRESA CARRO
0121 3E02                  MVI   A,02H          ;SI, LOCALIZA CARRO
0123 CDDD02                CALL  OPORTB
0126 CDE502                CALL  DISEG
0129 CDE902                CALL  D500MS
012C CDB502                CALL  INPORTA
012F CB47                  BIT   0,A            ;CARRO SOBRE CHAROLA
0131 2820                  JRZ   ORGCARRO       ;REGRESA CARRO
0133 3E06                  MVI   A,06H
0135 CDDD02                CALL  OPORTB
0138 CDE502                CALL  DISEG
013B CDE902                CALL  D500MS
013E CDB502                CALL  INPORTA
0141 CB47                  BIT   0,A            ;CARRO EN EL ORIGEN
0143 C23800                JNZ   FALLA          ;CARRO TRABADO => FALLA
0146 3E20                  MVI   A,20H
0148 CDDD02                CALL  OPORTB         ;BEEP FIN DE INICIALIZACION
014B CDE502                CALL  DISEG
014E CBAF                  RES   5,A
```

```
0150 CDDD02             CALL  OPORTB
0153 CD4B02  ORGCARRO:  CALL  RETCARRO
0156 CD0703             CALL  ARRANQUE          ;ESPERA BOTON DE ARRANQUE
0159 F3                 DI
015A CDB502  WAITCHAR:  CALL  INPORTA
015D CB6F               BIT   5,A               ;CHAROLA ?
015F 2008               JRNZ  NOWAIT            ;SI, INICIA EMPAPELADO
0161 3E02               MVI   A,02H             ;NO, ENCIENDE INDICADOR
0163 CDE002             CALL  OPORTC            ;  ESPERANDO CHAROLA
0166 CD0003             CALL  PARO
0169 F3                 DI                      ;MONITOREA BOTON DE PARO
016A 18EE               JMPR  WAITCHAR
016C 3E00    NOWAIT:    MVI   A,00H             ;APAGA INDICADOR CHAROLA
016E CDE002             CALL  OPORTC
0171 CD7002             CALL  BAJTOLVA
0174 CD0003             CALL  PARO
0177 F3                 DI
0178 CDB502             CALL  INPORTA
017B CB67               BIT   4,A               ;HAY PAPEL ?
017D CA1203             JZ    PAPEROUT          ;NO, REGRESA Y AVISA
0180 FB                 EI                      ;HABILITA INTERRUPCIONES
0181 D304               OUT   WD                ;DISPARA WATCHDOG
0183 3E09               MVI   A,09H
0185 CDDD02             CALL  OPORTB
0188 CDB502             CALL  INPORTA
018B CB5F               BIT   3,A               ;LIMITE INFERIOR
018D 280A               JRZ   LOOP2             ;NO, MUEVE TOLVA
018F CD0003  LOOP1:     CALL  PARO              ;SI, COMPRUEBA QUE SALGA
0192 CDB502             CALL  INPORTA
0195 CB5F               BIT   3,A
0197 20F6               JRNZ  LOOP1
0199 CD0003  LOOP2:     CALL  PARO
019C CDB502             CALL  INPORTA
019F CB57               BIT   2,A               ;LIMITE ENGOMADO ?
01A1 28F6               JRZ   LOOP2
01A3 F3                 DI                      ;SI, DESHABILITA INTERRUPS.
01A4 CD9002             CALL  MOVCARRO
01A7 CD7002             CALL  BAJTOLVA
01AA CDE502             CALL  DISEG             ;PRESIONA SOBRE LOS MUAICO
                        S
01AD 3E00               MVI   A,00H             ;  DURANTE 1 SEGUNDO
01AF CDDD02             CALL  OPORTB
01B2 CD0003             CALL  PARO
01B5 CD2602             CALL  SUBTOLVA
01B8 FB                 EI                      ;REGRESA CARRO, EXPULSA
01B9 D304               OUT   WD                ;  CHAROLA Y REGRESA A
01BB 3E02               MVI   A,02H             ;  ESPERAR CHAROLA
01BD CDDD02             CALL  OPORTB
01C0 08                 EXAF
01C1 CD0003  LOOP3:     CALL  PARO
01C4 CDB502             CALL  INPORTA
```

```
01C7 CB47                BIT   0,A
01C9 20F6                JRNZ  LOOP3
01CB 08                  EXAF
01CC CBE7                SET   4,A
01CE CDD002              CALL  OPORTB
01D1 08                  EXAF
01D2 CD0003    LOOP4:    CALL  PARO
01D5 CD8502              CALL  INPURTA
01D8 CB77                BIT   6,A
01DA 2809                JRZ   LOOP5
01DC CB47                BIT   0,A
01DE 28F2                JRZ   LOOP4
01E0 CD1E02              CALL  RESIA
01E3 18ED                JMPR  LOOP4
01E5 CD0003    LOOP3:    CALL  PARO
01E8 CD8502              CALL  INPURTA
01EB CB47                BIT   0,A
01ED 2803                JRZ   LOOP6
01EF CD1E02              CALL  RESIA
01F2 CD0003    LOOP6:    CALL  PARO
01F5 CD8502              CALL  INPORTA
01F8 CB6F                BIT   5,A
01FA 20E9                JRNZ  LOOP5
01FC 08                  EXAF
01FD CBA7                RES   4,A
01FF CDD002              CALL  OPORTB
0202 08                  EXAF
0203 CD0003    LOOP7:    CALL  PARO
0206 CD8502              CALL  INPORTA
0209 CB47                BIT   0,A
020B 28F6                JRZ   LOOP7
020D CD1E02              CALL  RESIA
0210 CD0003    LOOP8:    CALL  PARO
0213 CD8502              CALL  INPURTA
0216 CB77                BIT   6,A
0218 28F6                JRZ   LOOP8
021A F3                  DI
021B C35A01              JMP   WAITCHAR
021E 08        RESIA:    EXAF
021F CBBF                RES   1,A
0221 CDD002              CALL  OPORTB
0224 08                  EXAF
0225 C9                  RET
0226                     ;
0226                     ;
0226                     ;RUTINAS DE MOVIMIENTO DE LA TOLVA
0226                     ;
0226 FB        SUBTOLVA: EI                    ;SUBE TOLVA SIN VENTILADOR
0227 D304                OUT   WD
0229 3E01                MVI   A,01H
022B CDD002              CALL  OPORTB
```

```
022E CD0003     SUBLOOP1: CALL  PARO
0231 CDB502               CALL  INPORTA
0234 CB5F                 BIT   3,A
0236 2802                 JRZ   SUBLOOP2
0238 18F4                 JMPR  SUBLOOP1
023A CD0003     SUBLOOP2: CALL  PARO
023D CDB502               CALL  INPORTA
0240 CB4F                 BIT   1,A
0242 28F6                 JRZ   SUBLOOP2
0244 F3                   DI
0245 3E00                 MVI   A,00H
0247 CDDD02               CALL  OPORTB
024A C9                   RET
024B FB         RETCARRO: EI                ;REGRESA CARRO A ORIGEN
024C D304                 OUT   WD
024E 3E02                 MVI   A,02H
0250 CDDD02               CALL  OPORTB
0253 CD0003     RETLOOP1: CALL  PARO
0256 CDB502               CALL  INPORTA
0259 CB47                 BIT   0,A
025B 2802                 JRZ   RETLOOP2
025D 18F4                 JMPR  RETLOOP1
025F CD0003     RETLOOP2: CALL  PARO
0262 CDB502               CALL  INPORTA
0265 CB47                 BIT   0,A
0267 28F6                 JRZ   RETLOOP2
0269 F3                   DI
026A 3E00                 MVI   A,00H
026C CDDD02               CALL  OPORTB
026F C9                   RET
0270 FB         BAJTOLVA: EI                ;BAJA TOLVA CON VENTILADOR
0271 D304                 OUT   WD
0273 3E0D                 MVI   A,0DH
0275 CDDD02               CALL  OPORTB
0278 CD0003     BAJLOOP1: CALL  PARO
027B CDB502               CALL  INPORTA
027E CB4F                 BIT   1,A
0280 2802                 JRZ   BAJLOOP2
0282 18F4                 JMPR  BAJLOOP1
0284 CD0003     BAJLOOP2: CALL  PARO
0287 CDB502               CALL  INPORTA
028A CB5F                 BIT   3,A
028C 28F6                 JRZ   BAJLOOP2
028E F3                   DI
028F C9                   RET
0290 FB         MOVCARRO: EI                ;MUEVE CARRO CON VENTILADOR
0291 D304                 OUT   WD
0293 3E0E                 MVI   A,0EH
0295 CDDD02               CALL  OPORTB
0298 CD0003     MOVLOOP1: CALL  PARO
029B CDB502               CALL  INPORTA
```

```
029E CB47                BIT   0,A
02A0 2802                JRZ   MOVLOOP2
02A2 18F4                JMPR  MOVLOOP1
02A4 CD0003    MOVLOOP2: CALL  PARD
02A7 CDB502              CALL  INPORTA
02AA CB47                BIT   0,A
02AC 28F6                JRZ   MOVLOOP2
02AE F3                  DI
02AF 3E00                MVI   A,00H
02B1 CDDD02              CALL  OPORTB
02B4 C9                  RET
02B5                     ;
02B5                     ;
02B5                     ;RUTINAS DE ENTRADA/SALIDA
02B5                     ;
02B5 DB00      INPORTA:  IN    PORTA
02B7 E67F                ANI   7FH
02B9 5F                  MOV   E,A
02BA CDF102              CALL  D20MS             ;DEBOUNCE DE MICROSW
02BD DB00                IN    PORTA
02BF E67F                ANI   7FH
02C1 BB                  CMP   E
02C2 20F1                JRNZ  INPORTA
02C4 C9                  RET
02C5 DB02      INPORTC:  IN    PORTC
02C7 E630                ANI   30H
02C9 5F                  MOV   E,A
02CA CDF102              CALL  D20MS
02CD DB02                IN    PORTC
02CF E630                ANI   30H
02D1 BB                  CMP   E
02D2 20F1                JRNZ  INPORTC
02D4 CB3F                SRLR  A
02D6 CB3F                SRLR  A
02D8 CB3F                SRLR  A
02DA CB3F                SRLR  A
02DC C9                  RET
02DD D301      OPORTB:   OUT   PORTB
02DF C9                  RET
02E0 E60F      OPORTC:   ANI   0FH
02E2 D302                OUT   PORTC
02E4 C9                  RET
02E5                     ;
02E5                     ;
02E5                     ;RUTINAS DE TIEMPO
02E5                     ;
02E5 1632      D15EG:    MVI   D,32H             ;50 x 20 MS
02E7 180A                JMPR  DELAY
02E9 1619      D500MS:   MVI   D,19H             ;25 x 20 ms
02EB 1806                JMPR  DELAY
02ED 1605      D100MS:   MVI   D,05H             ;5 x 20 ms
```

```
02EF 1802              JMPR  DELAY
02F1 1601    D20MS:    MVI   D,01H
02F3 01FF0B  DELAY:    LXI   B,0BFFH
02F6 0D      DLOOP:    DCR   C
02F7 20FD              JRNZ  DLOOP
02F9 05                DCR   B
02FA 20FA              JRNZ  DLOOP
02FC 15                DCR   D
02FD 20F7              JRNZ  DLOOP
02FF C9                RET
0300                   ;
0300                   ;
0300                   ;RUTINAS DE ARRANQUE/PARO
0300                   ;
0300 CDC502  PARO:     CALL  INPORTC
0303 CB4F              BIT   1,A              ;PARO ?
0305 C8                RZ                     ;NO, REGRESA
0306 F3                DI
0307 CDC502  ARRANQUE: CALL  INPORTC
030A CB47              BIT   0,A              ;ARRANQUE ?
030C 28F9              JRZ   ARRANQUE         ;NO, ESPERA
030E FB                EI
030F D304              OUT   WD
0311 C9                RET
0312                   ;
0312                   ;
0312                   ;RUTINA DE FIN DE PAPEL
0312                   ;
0312 CD2602  PAPEROUT: CALL  SUBTOLVA
0315 CD4E02            CALL  RETCAHHU
0318 3E04              MVI   A,04H             ;ENCIENDE INDICADOR PAPEL
031A CDE002            CALL  OPORTC
031D 3E20    PAPELOOP: MVI   A,20H            ;SUENA ALARMA
031F CDDD02            CALL  OPORTB
0322 CDED02            CALL  D100MS
0325 CBAF              RES   5,A
0327 CDE902            CALL  D500MS
032A 18F1              JMPR  PAPELOOP
032C                   ;
0000                   END
```

```
AVOCET SYSTEMS 8085/280 ASSEMBLER -  VERSION 1.05M  SERIAL #00410

SOURCE FILE NAME: BITFROG.ASM                                    PAGE  H
---- SYMBOL TABLE ----

AKRANOUE 0307        INPORTA  02B5        ORGTOLVA 0100
BAJLOOP1 0278        INPORTC  02C5        PAPELOOP 031D
BAJLOOP2 0284        LOOP1    01BF        PAPEROUT 0312
BAJTOLVA 0270        LOOP2    0199        PARO     0300
BEEP     0042        LOOP3    01C1        PORTA    0000
BEEPLOUP 0044        LOOP4    01D2        PORTB    0001
CONTADL  0003        LOOP5    01E5        PORTC    0002
D100MS   02ED        LOOP6    01F2        RESTA    021E
D15ED    02E5        LOOP7    0203        RETCARRO 0248
D20MS    02F1        LOOP8    0210        RETLOOP1 0253
D500MS   02E9        MOVCARRO 0290        RETLOOP2 025F
DELAY    02F3        MOVLOOP1 029B        SUBLOOP1 022E
DLOOP    02F6        MOVLOOP2 02B4        SUBLOOP2 023A
EMERG    0046        NOWAIT   016C        SUBTOLVA 0226
FALLA    0038        OPORTB   02DD        WAITCHAR 015A
INICIO   0000        OPORTC   02E0        WD       0004
INIT8255 009B        ORGCARRO 0153

***** NO ERRORS DETECTED *****
```

Bibliografía

Avocet Systems Inc. 8085/Z-80 Assembler User's Manual
Delaware, 1984

Deboo,Gordon J., Burrous, Clifford N. Integrated Circuits and Semiconductor Devices: Theory and Application
McGraw-Hill, Tokio, 1979

Digital Equipment Corporation Data Communication Fundamentals
Massachusetts, 1981

Electronic Industries Association EIA Standard RS-232C (Revision of RS-232B) Washington, 1981

Garland,Harry Introduction to Microprocessor System Design
McGraw-Hill, Tokio, 1981

Intel Corp. Memory Components Handbook California, 1985

Intel Corp. Microsystem Components Handbook California, 1985

Leventhal,Lance Z-80 Assembly Language Programming
Osborne/McGraw-Hill, California, 1979

Libes,Sol, Garetz,Mark Interfacing to S-100/IEEE 696 Microcomputers
Osborne/McGraw-Hill, California, 1981

Martínez,Juan B. Microcontrolador PAT 85 Lab. Automatización, Instituto de Ingeniería, UNAM, Mexico, 1985

Miller,Alan R. The 8080/Z-80 Assembly Language, Techniques for Improved Programming John Wiley & Sons Inc., New York, 1981

Mostek Corp. Z-80 Technical Manual Texas, 1978

National Semiconductor Corp. CMOS Databook California, 1981

National Semiconductor Corp. Interface Databook California, 1980

National Semiconductor Corp. Linear Databook California, 1982

Ramírez,Edward V., Weiss,Melvyn Microprocessors Fundamentals
McGraw-Hill, Tokio, 1981

Televideo Systems Inc. TS 2908 User's Manual California, 1984

Televideo Systems Inc. Tele-DOS User's Manual California, 1984

Texas Instruments Inc. Manual de semiconductores de silicio
Texas, 1984/85

Texas Instruments Inc. The TTL Databook for Design Engineers
Texas, 1984/85

Waite,Mitchel, Pardee,Michael Microcomputer Primer Second ed.
Howard W. Sams & Co. Inc., Indiana, 1980

Wiatrowski,Claude A. Logic Circuits and Microcomputer Systems
McGraw-Hill, Tokio, 1981