UNIVERSIDAD NACIONAL AUTÓNOMA DE MEXICO

FACULTAD DE INGENIERIA

SISTEMA PARA EL MANEJO DE UN BRAZO MECANICO CON KITS EDUCACIONALES

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A N:

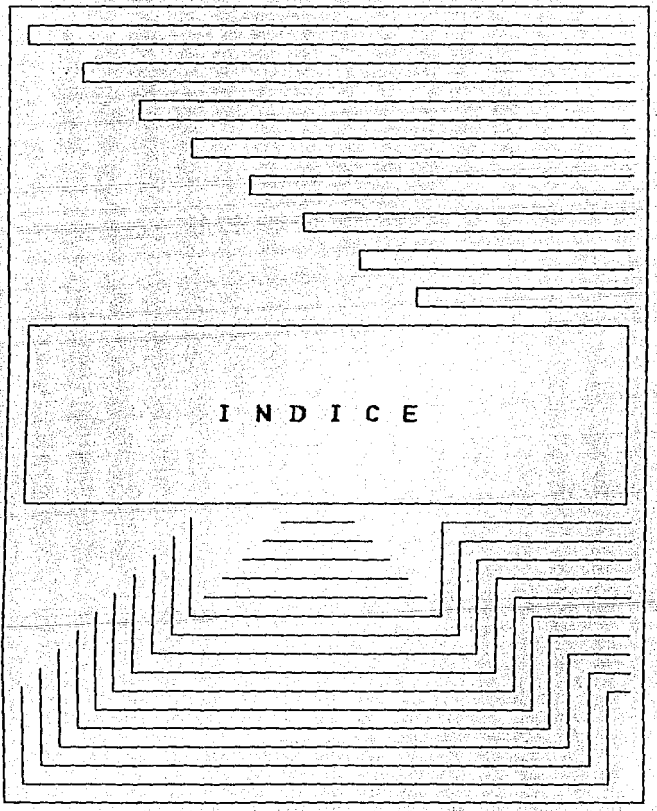
MANUEL ENRIQUE REINOSA SERVIN

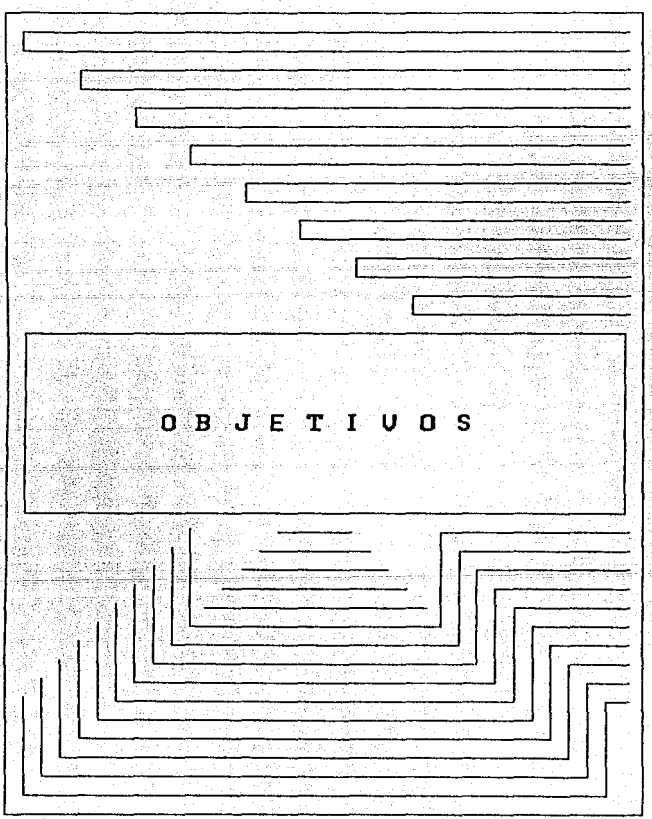
SANDRA LUZ LOPEZ CASTELLANOS

Director: Ing. Edmundo Rosales Valderrabano

México, D. F. 1992

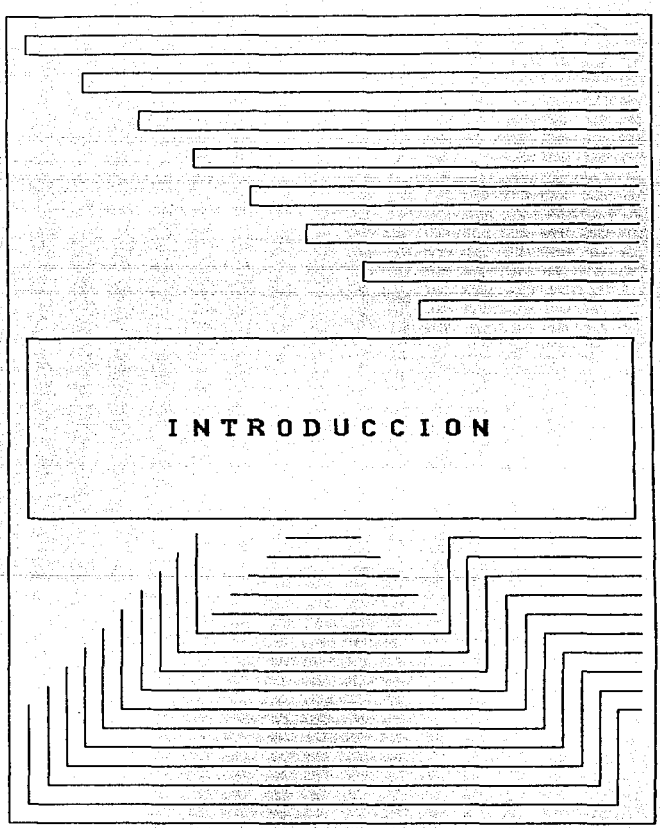
UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso


I N D I C E


OBJETIVOS

1) Establecer el enlace y desarrollar el software para lograr el manejo de un brazo mecánico a través de los kits educacionales MEX68KECB (con microprocesador MC68000) y SDK-Z80 (con microprocesador Z80).

2) Asentar las bases para el desarrollo de futuras prácticas de software y hardware para los laboratorios de las materias de Microcomputadoras y de Robótica.


INTRODUCCION

El hombre en su continua búsqueda de bienestar, ha conseguido que máquinas y equipos muy sofisticados efectúen todas aquellas labores rutinarias o peligrosas. Se tiende al uso de sistemas automáticos para obtener mayor seguridad, calidad, rapidez, precisión, ahorro, productividad, rendimiento y competitividad.

Este proyecto se planteó con el fin de presentar un panorama general del funcionamiento de un brazo mecánico, así como de la programación de circuitos periféricos de comunicación en paralelo.

El brazo mecánico empleado en este proyecto se halla conectado originalmente a un computador personal a través de un controlador (se describe en el capítulo 2). Al sustituir el computador por un kit educacional se logra:

a) Emplear el equipo con que cuenta el laboratorio de Microcomputadoras.

b) Reducir el espacio físico que requiere el trabajar con este equipo.

c) Facilitar el desarrollo de prácticas, ya que se utiliza algo conocido, como se ha expresado en el inciso a). Además se integrarían las nuevas prácticas con las ya existentes.

d) Que al controlar el brazo con equipos básicos, el alumno aprenda cuál es su funcionamiento y cuáles son los componentes que lo integran.

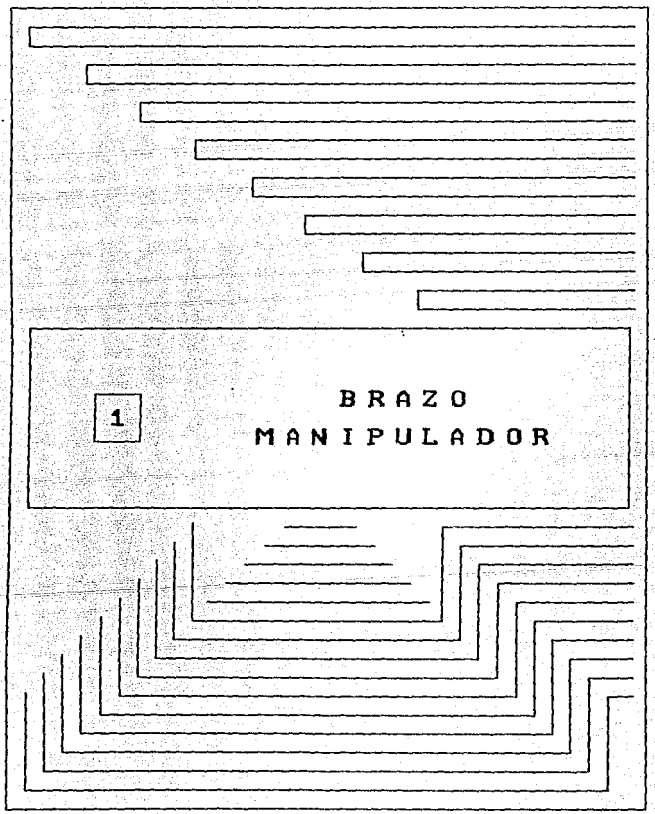
El capítulo 1 de este documento muestra al brazo mecánico en sus diferentes partes.

En el capítulo 2 se estudia al controlador, incluyéndose información sobre sensores que pueden ser utilizados con este equipo.

Los capítulos 3 y 4 describen la programación de los circuitos MC68230 y Z80-PIO (interfaces paralelo), que permiten la comunicación desde los procesadores MC68000 y Z80 respectivamente. Se refieren también a la conexión establecida para lograr la comunicación entre los kits educacionales y el controlador.

En el capítulo 5 se ha incluído lo referente a software (programación de circuitos, envío de datos al controlador, etc.), integrando un sistema que permite el manejo del brazo a través del teclado del controlador con un conjunto de funciones.

El capítulo 6 presenta una forma alterna para el manejo del brazo, desarrollándose un método para calcular posiciones.


1
BRAZO
MANIPULADOR

1.1 Introducción

Un brazo mecánico es un mecanismo capaz de realizar movimientos semejantes o equivalentes a los de un brazo humano.

Según su construcción puede pertenecer a alguno de los siguientes tipos:

a) **Cartesiano.**- éste puede desplazarse sobre los ejes de un sistema de coordenadas cartesianas, pero siempre en forma palalela a ellos, tal como se muestra en la siguiente figura.

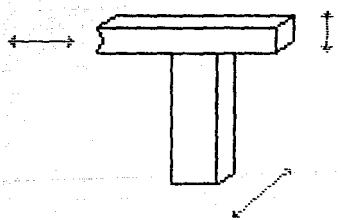


fig. 1.1 - Esquema general de un brazo de tipo cartesiano

b) Cilíndrico.- Se basa en un sistema de coordenadas cilíndricas. Existe un movimiento de giro sobre la base, además de desplazamientos lineales: uno vertical y otro horizontal (ver fig. 1.2).

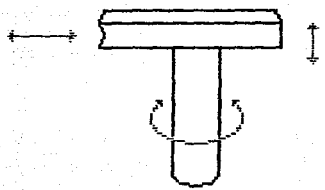


fig. 1.2 - Esquema general de un brazo de tipo cilíndrico

c) Polar.- Es similar al cilíndrico, pero el desplazamiento vertical se produce a través de un giro sobre el plano vertical (ver fig. 1.3).

d) Antropomórfico.- Como su nombre lo indica, es similar a un brazo humano en cuanto a su forma, ya que sus partes se hallan unidas mediante articulaciones. El brazo empleado en este proyecto pertenece a este tipo, y su configuración puede consultarse en la fig. 1.4.

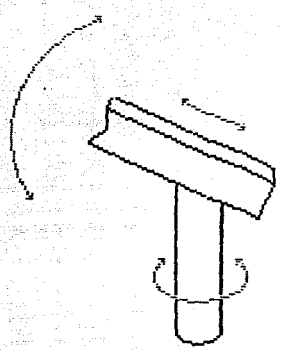


fig. 1.3 - Esquema general de un brazo de tipo polar

1.2 Descripción

El brazo mecánico que se empleará se encuentra compuesto de varias partes articuladas entre sí, distinguiéndose cinco ejes de rotación, más una pinza neumática para sujeción de objetos (ver fig. 1.4).

El movimiento en las articulaciones se produce mediante servomotores, cada uno de los cuales permite un giro de 120° de amplitud.

El brazo es capaz de girar sobre su base gracias al servo 0, el cual produce el movimiento de rotación (ROTATE).

El servo 1 mantiene su eje perpendicular al del servo 0, y permite el movimiento de hombro (SHOULDER).

Siguiendo sobre el brazo hacia arriba se encuentra el servo 2, que genera el movimiento de brazo (ARM).

En la parte extrema del brazo se halla el efector final, compuesto por el servo 4 (movimiento de mano (HAND)) con eje paralelo al del servo 2, por el servo 3 (movimiento de muñeca (WRIST), similar al manipular un destornillador) con eje perpendicular al del servo 4, y por la pinza (GRIPPER), la cual es de acción neumática y sólo de dos posiciones: abierto/cerrado.

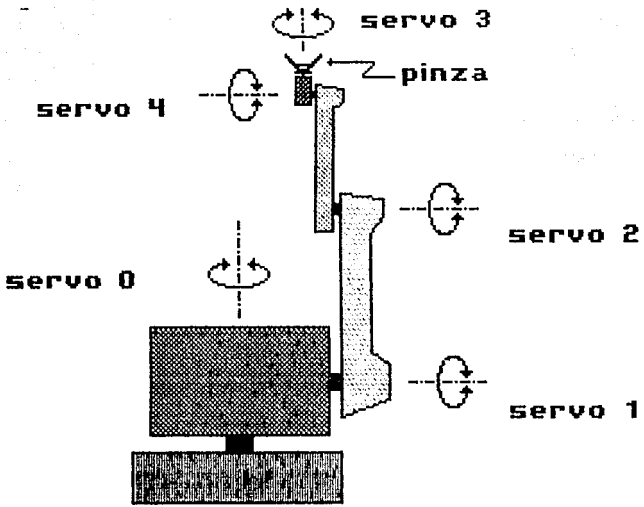


fig. 1.4 - Esquema del brazo mecánico, en el cual se indican servomotores y sus ejes de rotación

La extensión máxima del brazo es de 40 cm, medida desde el eje del servo 1 al extremo de la pinza. El máximo peso que puede soportar es de 200 g.

Bajo la base del brazo se encuentra una tarjeta de circuito impreso, en la cual se han montado las etapas de control y de potencia de los servomotores. En este mismo lugar se aloja la válvula de control para la pinza. A través de un conector de 34 líneas se reciben las señales provenientes del controlador, con la siguiente distribución:

CONECTOR (pin)	SEÑAL
1,2	GND
3,4,6,11-17,20	NC
5	servo 0
7	servo 1
8	servo 4
9	servo 2
10	servo 3
18	pinza
19	+5V
21-26,28	GND
27,29,30	+6V
31,32	+5V
33,34	+5V

tabla 1.1 - Distribución de señales en el conector del brazo

1.3 Motores

1.3.1 Definición de servomotor

Un servomotor forma parte de un servosistema, y por lo tanto se presenta primero un panorama general:

Para aumentar la precisión de un sistema de mando debe de establecerse un enlace entre las señales de entrada y de salida. Por medio de este enlace, denominado comúnmente **cadena de retorno**, la señal de salida se introduce en el sistema después de compararse con la señal de entrada (ver fig. 1.5). La señal resultante de la diferencia entre la señal de entrada y la señal de salida debe de actuar sobre el sistema a gobernar con el fin de corregir el error. Un sistema con una realimentación como la anteriormente descrita, se denomina **sistema en cadena cerrada o servosistema.**

Un servosistema reúne las dos propiedades siguientes:

- Amplifica la potencia; a partir de una señal de baja potencia, normalmente electrónica, es capaz de obtener un nivel de potencia mucho más elevado.

- Está dotado de una realimentación.

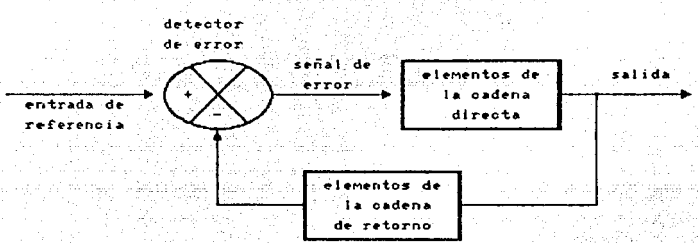


fig. 1.5 - Diagrama de bloques básico de un servosistema

La fig. 1.5 representa esquemáticamente los elementos más característicos de un servosistema. La cadena directa de un servosistema puede constar de elementos tales como detector de error, amplificador, motor y redes correctoras. La cadena de realimentación consta normalmente de transductores y redes correctoras. A menudo, para mejorar las características del sistema es necesario introducir redes correctoras en la cadena directa, en la de retorno o en ambas.

Los transductores pueden ser de muy diversa índole (termopares, células fotoeléctricas, termistores, potenciómetros, etc.), según la naturaleza del fenómeno a detectar. Las señales generadas por éstos son generalmente tensiones eléctricas con un nivel de potencia muy bajo (10 a

15 mW). Esto no produce inconveniente alguno, ya que estas señales pueden amplificarse suficientemente para el gobierno de los órganos de accionamiento (**servomotores**), sean éstos hidráulicos, neumáticos o eléctricos; por el contrario, ofrece la ventaja de una fácil medición y transmisión a distancia.

La fig. 1.6 muestra el diagrama de bloques de un sistema en cadena cerrada, cuya función consiste en posicionar una carga. En este caso se emplean un par de potenciómetros para detectar el error entre la posición real de la carga (salida) y la posición deseada (entrada de referencia). La tensión de error ϵ se amplifica y se aplica al motor para que éste gire en un sentido tal que tienda a eliminar la señal de error. Como ya se vió, a los motores empleados en estos sistemas se les denomina precisamente **servomotores**.

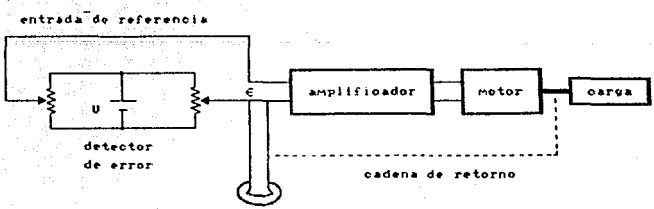


fig. 1.6 - Sistema en cadena cerrada para posicionar una carga

1.3.2 Características

Los motores que se encuentran incluidos en el brazo mecánico son de corriente directa, con polarización máxima de 2 V y consumo máximo de 450 mA.

Cada motor posee su eje mecánicamente acoplado al cursor de un potenciómetro, cuya señal proporciona retroalimentación de posición, como se verá en el punto 1.4.

Estos se convierten en servomotores por la adición de una etapa de control. Para los servos 0, 1 y 2 dicha etapa se basa en un circuito especial adicional, tal como se verá en el punto 1.4. Los servos 3 y 4 han sido armados en fábrica como una sola unidad, en la cual se incluye el motor y la etapa de control; su funcionamiento, en principio, es el mismo que para los otros servomotores.

1.4 Etapa de control

Cada uno de los servomotores es controlado mediante un circuito servo-amplificador NE544 de Signetics, cuyas características y conexión se incluyen en los siguientes párrafos.

El NE544 es un circuito integrado lineal con funciones de

servo-amplificador y demodulador de duracion de pulso. Su uso incluye aplicaciones de control de posición con lazo cerrado. A través de la conexión de diversos elementos a algunos de sus pines es posible adaptarlo a una gran variedad de servomotores.

En la fig. 1.7 se muestra el diagrama de bloques de este circuito.

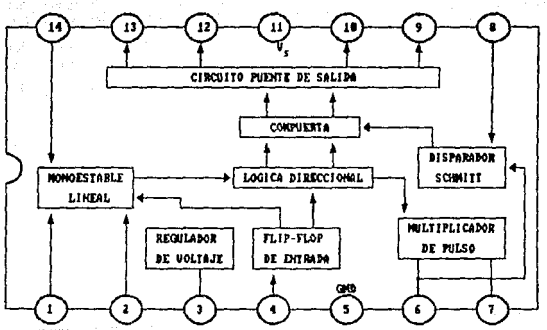


fig. 1.7 - Diagrama a bloques del circuito NE544 de Signetics

El regulador de voltaje proporciona una salida constante de 2.5 V, protegiendo contra variaciones de la fuente de alimentacion.

La duración de un pulso positivo aplicado como entrada (pin 4) determina la posición que ha de alcanzar el motor. Al aparecer éste habilita el flip-flop e inicia el período del monoestable. La lógica direccional compara la duración del pulso de entrada con la del pulso interno del monoestable, y almacena el resultado en un flip-flop de dirección. La diferencia exacta entre ambos pulsos es llamada pulso de error, y es alimentada a los circuitos disparador Schmitt y multiplicador de pulso. Estos circuitos determinan tres importantes parámetros:

a) **Mínimo pulso de salida.**- Es el menor pulso de salida que puede ser generado por el disparador Schmitt.

b) **Ganancia del multiplicador de pulso.**- Es la relación entre la duración del pulso de error y la del pulso de salida.

c) **Punto muerto.**- Es la mínima diferencia entre el pulso de entrada y el pulso interno del monoestable, necesaria para obtener una salida.

Se puede lograr el ajuste de estos parámetros con resistores y capacitores externos conectados a los pines 6, 7 y 8, como se verá más adelante.

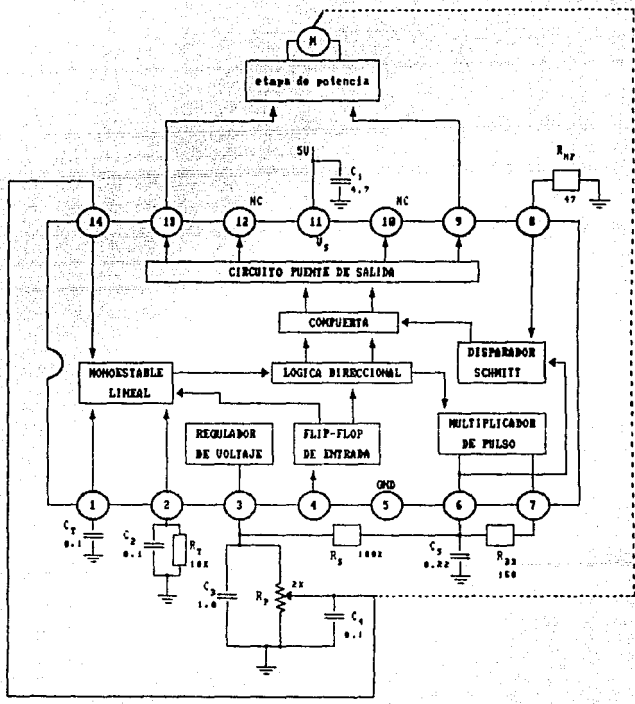


fig. 1.8 - Configuración de la etapa de control de un servomotor, basada en el NE544

El disparador Schmitt activa la compuerta por un periodo preciso de tiempo, proporcional a la duración del pulso de error, dando salida al motor a través del circuito puente.

El monoestable integrado es lineal, lo que permite diseñar servosistemas con muy alta precisión de posición y funciones de transferencia lineales <duración de pulso a posición>.

En la fig. 1.8 se muestra la conexión del NE544 en la tarjeta del brazo mecánico.

El eje del potenciómetro R_P se encuentra mecánicamente acoplado al del motor, proporcionando de este modo retroalimentación de posición.

Una vez que aparece el pulso de entrada, el flip-flop (tipo SET-RESET) toma un valor alto e inicia el periodo del monoestable.

MONOESTABLE LINEAL

Inicialmente el capacitor C_T en el pin 1 se halla descargado. La corriente que circula permanentemente por el resistor R_T, conectado al pin 2, es igual a la que comienza a fluir a

través de C_T. Cuando el voltaje en este último iguala al proporcionado por el potenciómetro R_P (ver fig. 1.9), concluye el período del monoestable. Así, la duración del pulso interno del monoestable está en razón directa a la posición del motor, obtenida a través del potenciómetro R_P, y viene dada por la ecuación

$$T = \frac{C_T \, V_{14}}{V_1 / R_T}$$

donde:

C_T es el valor del capacitor en el pin 1

V_{14} es el voltaje proporcionado por el potenciómetro R_P en el pin 14

V_1 es un voltaje interno que polariza al resistor R_T, y tiene un valor fijo de 1.8 V

R_T es el valor del resistor en el pin 2.

De este modo, para los valores mostrados en la fig. 1.8, se tiene

$$T = \frac{(0.1 \times 10^{-6}) \, V_{14}}{1.8 / (18 \times 10^{3})}$$

$$T = 10^{-3} \, V_{14} \quad .$$

Para dar una mejor idea de la magnitud de este periodo, se puede tener en cuenta lo siguiente: El voltaje proporcionado por el regulador es de 2.5 V, así que puede considerarse un valor cualquiera de 0 a 2.5 V en el cursor del potenciómetro R_P; si se toma arbitrariamente como 1.5 V, entonces T = 1.5 ms (nótese la linealidad).

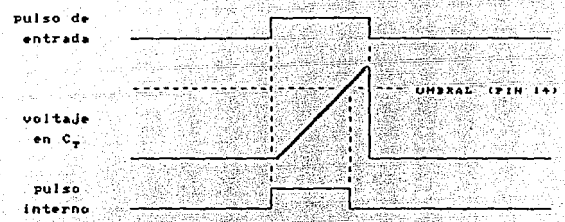


fig. 1.9 - Diagrama de tiempo que muestra la generación del pulso interno en el monoestable del NE544

Una vez que han finalizado el pulso del monoestable y el pulso de entrada, el capacitor C_T es descargado nuevamente a través de circuitería interna, y el flip-flop de entrada es reinicializado.

LOGICA DIRECCIONAL

Esta parte del NE544 genera el pulso de error a partir de la comparación de duraciones entre el pulso de entrada y el pulso interno, determinando a la vez la dirección de giro del motor.

En la fig. 1.10 puede verse la generación del pulso de error para cuando el pulso de entrada es mayor que el pulso interno.

fig. 1.10 - Generación del pulso de error para cuando el pulso de entrada poseé mayor duración que la del pulso interno

En la fig. 1.11 se muestra cómo aparece el pulso de error si el pulso de entrada es menor al pulso interno. En este caso el motor gira en sentido contrario al que tendría respecto a la relación mostrada en la fig. 1.10.

Nótese que en ambos casos el pulso de error inicia cuando uno de los pulsos de generación (entrada o interno) termina su duración, y finaliza cuando el pulso de generación contrario decae.

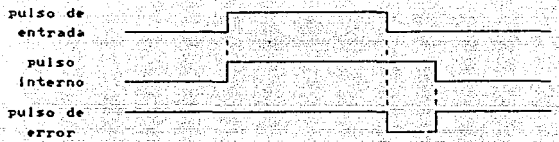


fig. 1.11 – Generación del pulso de error para cuando el pulso de entrada posee menor duración que la del pulso interno

MULTIPLICADOR DE PULSO Y DISPARADOR SCHMITT

Hasta antes de la aparición del pulso de error, el capacitor C_B (conectado al pin 6) se encuentra cargado casi al nivel del regulador de voltaje. En cuanto inicia el pulso de error, este capacitor comienza a descargarse a través del resistor R_{DB}, el cual está conectado en paralelo con un resistor interno R_I, hasta alcanzar el umbral inferior (V_L) del disparador Schmitt (véase fig. 1.12). El tiempo transcurrido desde que inicia el pulso de error hasta que se alcanza este último nivel es lo que se denomina como punto muerto (T_{DB}).

Una vez que C_B se encuentra en este nivel, aparece el pulso de salida y C_B sigue siendo descargado, pero esta vez a través de una fuente constante de corriente interna, hasta que el pulso de error desaparece.

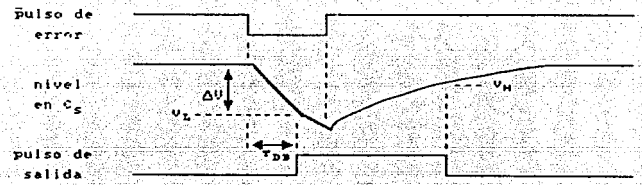


fig. 1.12 - Generación del pulso de salida a partir del pulso de error. Nótense los niveles V_L y V_H del disparador Schmitt en C_s

Después de que el pulso de error ha finalizado, C_s es cargado nuevamente a través del resistor R_s. El pulso de salida se mantiene hasta que el nivel en C_s iguala al umbral superior (V_H) del disparador Schmitt.

La duración del punto muerto puede ser calculada con la ecuación

$$T_{DS} \approx \frac{C_s \Delta V}{I_T} \quad .$$

donde:

C_s es el valor del capacitor en el pin 6

ΔV es la diferencia entre el voltaje inicial en C_s y el umbral inferior del disparador Schmitt

I_T es la corriente total de descarga de C_s.

Se tiene que

$$I_T \approx I_m + \frac{V_m\,(R_i + R_{DM})}{R_i\,R_{DM}} \quad .$$

donde:

I_m es la corriente de descarga producida por la fuente interna, cuyo valor es 3 mA

V_m es el voltaje inicial del capacitor C_m, el cual es de 2.2 V

R_i es el valor del resistor interno (150 Ω)

R_{DM} es el valor del resistor entre los pines 6 y 7.

De este modo

$$I_T \approx 3 \times 10^{-3} + \frac{2.2\,(150 + 150)}{(150)\,(150)}$$

$$I_T \approx 32.3 \text{ mA} \quad .$$

Por lo tanto

$$T_{DM} \approx \frac{0.22 \times 10^{-4}\,(0.7)}{32.3 \times 10^{-3}}$$

$$T_{DM} \approx 4.76\ \mu s \quad .$$

El valor del resistor R_{HH} conectado al pin 8 determina la histéresis del disparador Schmitt, y por lo tanto la duración del pulso de salida.

COMPUERTA

Ésta recibe el pulso de salida proporcionado por el disparador Schmitt, y permite el paso de las señales de dirección hacia el circuito puente de salida por el tiempo que permanezca activo.

CIRCUITO PUENTE DE SALIDA

Es una etapa de potencia para el manejo del motor. Recibe las señales de dirección provenientes de la lógica direccional. Una de estas señales es alta mientras la otra es baja, produciendo que el motor gire en cierto sentido. Cuando se invierten las señales, el motor gira en sentido contrario.

1.5 Etapa de potencia

Las señales que parten de la etapa de control a través de los pines 9 y 13 del NE544, entran a una etapa de potencia que proporciona la corriente necesaria para el manejo del motor.

El diagrama del circuito que compone a dicha etapa se muestra en la fig. 1.13.

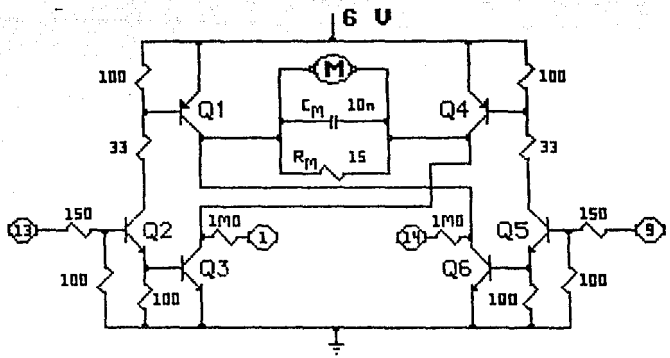


fig. 1.13 - Diagrama del circuito correspondiente a la etapa de potencia para un servomotor.
(Q1,Q4 = BD434, Q2,Q5 = ZTX300, Q3,Q6 = BD433)

Para efecto de explicar el modo de operación de este circuito, supóngase que la señal proveniente del pin 13 es alta y la del pin 9 es baja.

El nivel alto recibido en la base del transistor Q2 produce que éste encienda, al igual que Q3. La diferencia de potencial entre la base del transistor Q1 y su colector es baja, ya que este último posee un nivel alto referenciado por el pin 14 de la etapa de control, y por lo tanto se encuentra conduciendo.

El nivel bajo recibido en la base del transistor Q5 mantiene a éste y a los transistores Q4 y Q6 en estado de alta impedancia.

Así, la corriente circula a través de Q1, siguiendo por el paralelo motor-Cm-Rm, después del cual llega a tierra mediante Q3.

Si el giro del motor ha de efectuarse en sentido contrario, la señal del pin 9 será alta y la del pin 13 baja. En este caso los transistores Q4, Q5 y Q6 se hallan encendidos, mientras que Q1, Q2 y Q3 mantienen su alta impedancia.

Cuando las señales provenientes de los pines 9 y 13 son bajas, el motor es desenergizado.

1.6 Pinza neumática

La pinza con la que cuenta el brazo es de acción neumática, es decir funciona a través de aire comprimido. Esta puede adquirir solamente dos posiciones: abierto y cerrado.

El paso del aire es regulado por una válvula solenoide (ver fig. 1.14). Una válvula solenoide es una válvula cuyo vástago (extensión del elemento interruptor del flujo) se halla gobernado por un solenoide que se ha embobinado a su alrededor. Al hacer circular corriente por dicho solenoide se produce un campo magnético que desplaza al vástago en cierta dirección, ya que éste se encuentra hecho de un material ferromagnético.

Cuando es energizada, el vástago se mueve, permitiendo el paso del aire hacia la pinza. Si se desenergiza, el vástago regresa a su posición de reposo por acción de un resorte interno; el paso del aire es suprimido y la presión hacia la pinza es liberada.

Dentro del cuerpo de la pinza, el aire a presión impulsa un pistón hasta su extremo final, el cual está unido a un perno que por disposición mecánica cierra dos barras. Cuando la presión cede, el pistón regresa a su posición normal impulsado por un resorte interno, y las barras se separan.

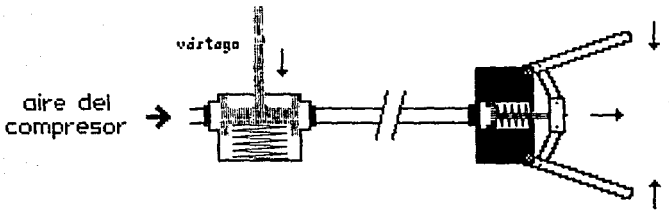


fig. 1.14 - Diagrama que muestra el funcionamiento del sistema neumático de la pinza

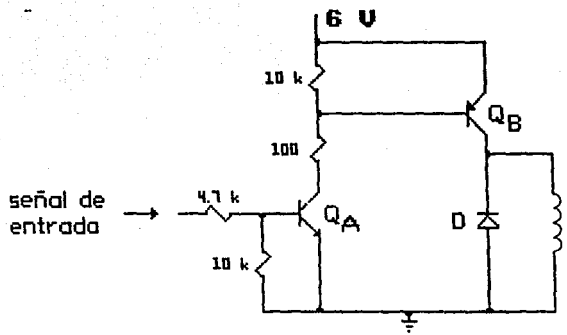
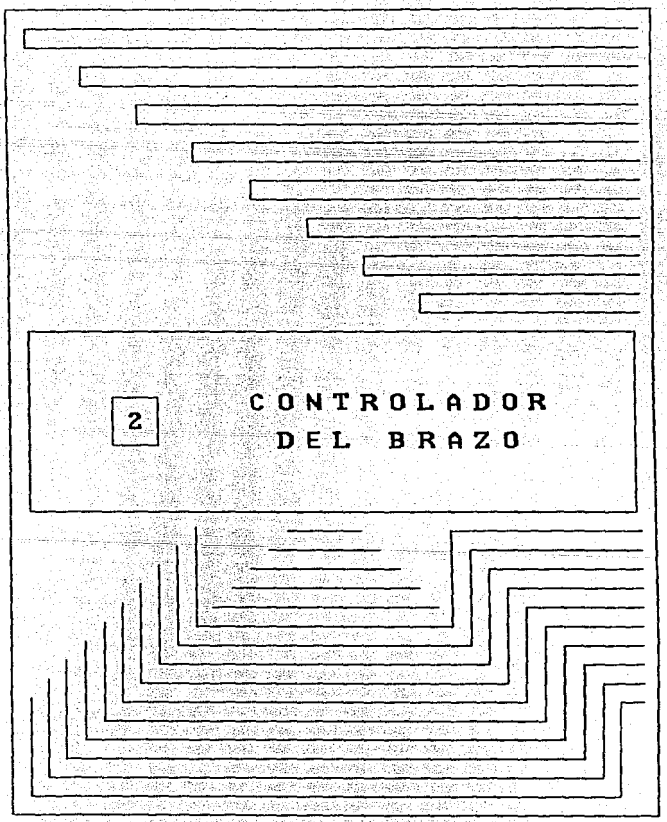


fig. 1.15 - Diagrama correspondiente al circuito amplificador para el manejo de la válvula solenoide.
(QA = BC184L, QB = BD434, D = IN4001)

En la tarjeta sobre la que se hallan montados los circuitos correspondientes a las etapas de control y de potencia del brazo, se encuentra también otro pequeño circuito que proporciona la corriente necesaria para energizar el solenoide de la válvula (ver fig. 1.15). Cuando la señal de entrada es alta, el transistor Q_A es encendido, provocando que Q_B alcance el mismo estado. Así, el solenoide es energizado y produce el movimiento del vástago.

Si la señal de entrada adquiere un nivel bajo, los transistores son apagados y se interrumpe el paso del aire por la desenergización del solenoide.


2
CONTROLADOR
DEL BRAZO

El controlador es una unidad que entrelaza al brazo mecánico y al equipo de cómputo, la cual realiza las siguientes funciones:

a) Convierte la información enviada por el computador en señales propias para el movimiento del brazo y para el control de los procesos de salida.

b) Retiene la información de los procesos de entrada (sensores y teclado) para que pueda ser leída por el computador.

c) Contiene la fuente de poder que alimenta al brazo.

2.1 Señales de entrada

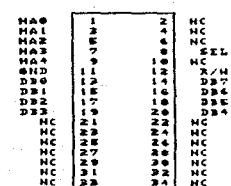


fig. 2.1 - Señales en el conector interno del controlador

Las señales recibidas por el controlador pueden ser clasificadas en tres tipos: señales de control, señales de dirección y señales de datos.

En la fig. 2.1 se muestra la distribución de señales en el conector de la tarjeta del controlador.

a) **Señales de control.**_ Son aquéllas que permiten una acción directa sobre el controlador, y son:

- $\overline{\text{SEL}}$ ---> Cuando se encuentra en un nivel lógico bajo (pin 8) permite realizar cualquier operación sobre el controlador, ya sea de lectura o escritura. Por así decirlo es la señal maestra que dirige al controlador.

- READ/$\overline{\text{WRITE}}$ ---> Indica la operación que se va a realizar: lectura (READ, habilitada con nivel alto) o escritura (WRITE, habilitada con nivel bajo), y siempre relativa al controlador.

b) **Señales de dirección.**_ Corresponden a los bits de dirección.

la cual permite seleccionar un servo o un puerto de entrada o salida en particular. Estas señales son:

MA0 (pin 1)
MA1 (pin 3)
MA2 (pin 5)
MA3 (pin 7)
MA4 (pin 9)

El bit más significativo viene dado por MA4. De este modo tenemos la siguiente relación de direcciones:

<-- DIRECCION --->				<-- PARTE CORRESPONDIENTE -->	
MA3	MA2	MA1	MA0		
0	0	0	0	Servo 0 (Rotate)	
0	0	0	1	Servo 1 (Shoulder)	
0	0	1	0	Servo 2 (Arm)	
0	0	1	1	Servo 3 (Wrist)	
0	1	0	0	Servo 4 (Hand)	
1	0	0	0	Teclado	
1	0	0	1	Salida al display	*
1	0	1	0	Entrada procesos 1 - 8	
1	0	1	1	Salida procesos 1 - 8	x
1	1	0	0	Entrada procesos 11 - 18	*
1	1	0	1	Salida procesos 11 - 18	*

1	1	1	0	Deshabilita controlador
1	1	1	1	Habilita controlador

* Sólo para diseños de tipo industrial
x Se incluyen sólo los 4 primeros como estándar

Nótese que la señal MA4 no se utiliza, debido a que está considerada en forma general, pero sólo se le incluye en diseños de tipo industrial. Por lo tanto se tomará siempre con un valor de 0.

Ya que la dirección 1001 (salida al display) no tiene uso en este tipo de controlador, al bit de datos menos significativo de ésta se le utiliza para el manejo de la pinza neumática. Sin embargo, se mencionará que un display externo de 7 segmentos podría ser utilizado, conectando las líneas de salida de esta dirección a cada uno de los segmentos. Otra opción de conexión sería pasar las líneas de salida por un par de convertidores a 7 segmentos, y de ahí a otro par de displays, con lo cual se lograría el desplegado de valores más altos que en el caso anterior.

La dirección 1010 (entrada procesos 1 - 8) es usada para obtener información del exterior (equipo adicional; p.ej. sensores). A cada línea de datos se le

asocia un proceso. En contraste, la dirección 1011 (salida procesos 1 - 8) se emplea para dar salida a información con un propósito general (puede enviarse a equipo externo: p.ej. relevadores, válvulas de control, motores de pasos).

Después de encenderse, el controlador se halla en un estado deshabilitado, por lo que debe de accesarse la dirección 1111 para dejarlo en posibilidad de ser utilizado (se efectúa con la señal SEL en bajo, sin importar el nivel de las líneas restantes). Hecho ésto, automáticamente los servomotores son energizados y colocados en una posición extrema: misma que la obtenida al dar un valor FF hex, según bits de datos (su significado se verá más adelante).

c) **Señales de datos.** Representan a los bits del bus de datos relativos al servo o puerto, indicado por los bits de dirección MA0 - MA4. El bit más significativo es representado por DB7. Puede verse entonces que el dato a manejar va de 00H a FFH, es decir puede tomar 256 posibles valores.

DB0 (pin 13)
DB1 (pin 15)
DB2 (pin 17)
DB3 (pin 19)
DB4 (pin 20)
DB5 (pin 18)
DB6 (pin 16)
DB7 (pin 14)

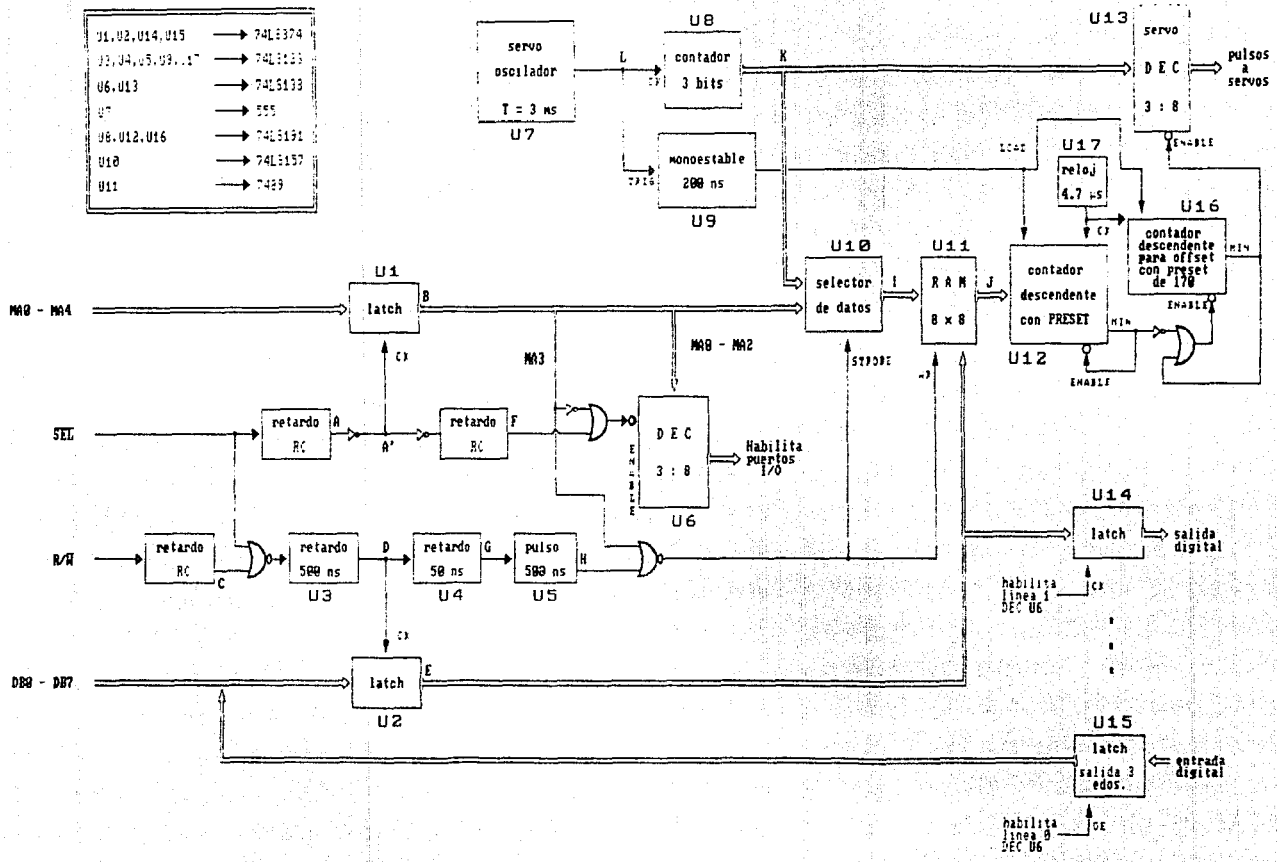


fig. 2.2 - Esquema general del controlador

2.2 Esquema general y funcionamiento

El esquema del controlador se muestra en la fig. 2.2, y habrá de irse consultando a la vez que se leen los siguientes párrafos.

Cuando la señal SEL adquiere un nivel bajo (ver fig. 2.3), el flanco de bajada genera la señal de reloj para que los bits de dirección MA0 - MA4 sean cargados en un latch (U1), de modo que se mantengan estables durante la operación.

Si la señal R/$\overline{W}$ es baja (operación de escritura), al combinarse con $\overline{SEL}$ (estado bajo) habilita la carga de datos (líneas DB0 - DB7) en un latch (U2).

Los retardos permiten que las señales se estabilicen para poder ser utilizadas.

La señal SEL junto con MA3 controlan las salidas del decodificador (U6) de puertos de entrada/salida (ver fig. 2.4). De este modo, cuando MA3 es baja, la operación se refiere a alguno de los servomotores, y el decodificador se halla deshabilitado. Si por el contrario esta señal es alta, el decodificador habilita la salida correspondiente de acuerdo con los bits indicados por MA0 - MA2 (siempre y cuando SEL se encuentre en un estado bajo).

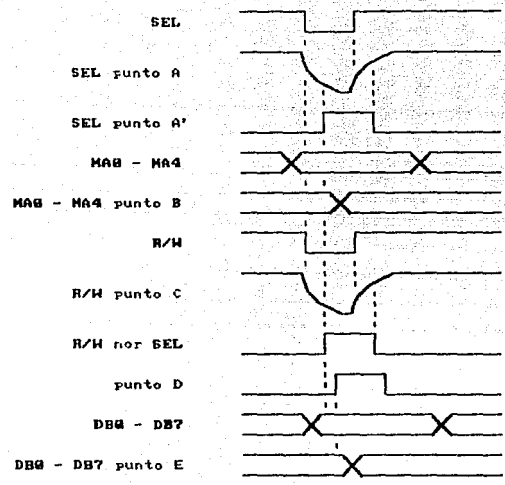


fig. 2.3 - Diagramas de tiempo correspondientes a las señales involucradas en la carga de dirección y dato en el controlador

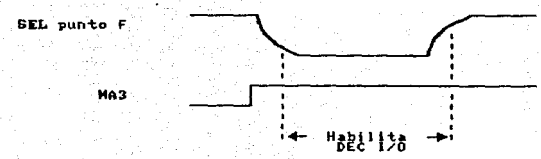


fig. 2.4 - Diagrama de tiempo correspondiente a la habilitación del decodificador de entrada/salida

Si la salida del decodificador que ha sido habilitada corresponde a un puerto de salida, se efectúa la carga de datos en el latch indicado (p.ej. U14). Tratandose de un puerto de entrada, la salida del decodificador ocasiona que la información registrada en el latch de dicho puerto (p.ej. U15) pueda accesar al bus de datos (lineas DB0 - DB7) en su etapa inicial, para así transferirla al computador.

2.3 Pulsos enviados al brazo, generados a partir de los datos de entrada

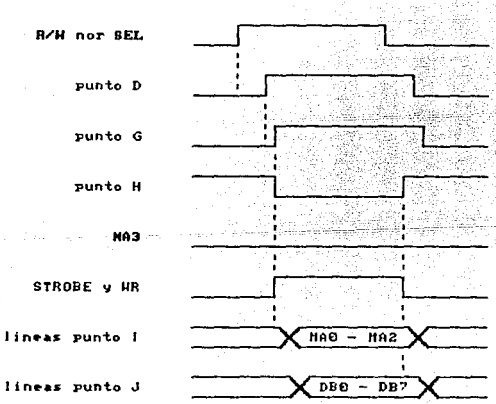


fig. 2.5 - Diagramas de tiempo correspondientes a la carga de datos de los servomotores

Una vez que SEL y R/$\overline{W}$ han sido combinadas (ver fig. 2.5), y siendo ambas bajas, se genera un pulso bajo de 500 ns de duración (U5), que junto con MA3 baja (que indica que la dirección se refiere a los servos) produce que el selector de datos (U10) permita que las líneas MA0 - MA2 direccionen la RAM, y que el dato que se halla en el latch U2 sea almacenado en dicha dirección por la aparición de la señal de habilitación de escritura.

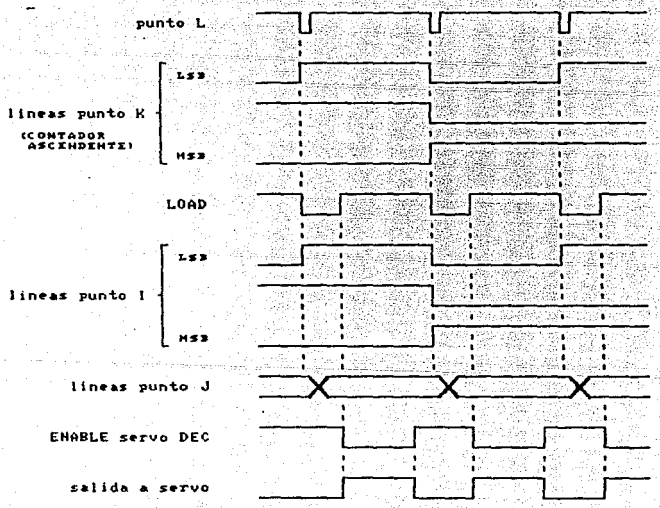


fig. 2.6 - Diagramas de tiempo correspondientes al multiplexado de salidas a los servomotores

El contador de 3 bits (U8) va proporcionando la dirección de cada uno de los servos (ver fig. 2.6), lo cual permite generar uno a uno los pulsos que parten hacia éstos. Normalmente la señal de selección de datos es baja, por lo que el selector de datos (U10) da salida a las líneas del contador, direccionando así a la RAM.

Un servo-oscilador (U7, el cual es un circuito 555) genera un pulso bajo cada 3 ms, incrementando de este modo la salida del contador U8 y permitiendo la carga inicial del contador descendente de 8 bits (U12) con el dato correspondiente que se halla en RAM (el monoestable de 200 ns (U9) da tiempo a que se estabilice dicho dato). A la vez que el dato es cargado en U8, el contador para offset (U16) es inicializado con un valor ya alambrado de 170.

La señal que genera el contador descendente al llegar a cero habilita a un contador auxiliar (U16) que proporciona un offset de 0.8 ms (170 * 4.7 µs)*. Así, la salida del decodificador de servos (U13) indicada por el contador de 3 bits, se mantiene activa por el tiempo que tarda el contador descendente desde que es cargado hasta que llega a cero, más el offset indicado.

* La razón de este valor para el offset se basa en especificaciones del NE544, ya que el pulso de entrada genera movimiento en el motor cuando su duración es mayor a 0.8 ms.

El pulso de salida tiene una duración de entre 0.8 ms (para un dato inicial de 00H) y 2 ms (para un dato inicial de FFH). De este modo, la señal se encuentra modulada por duración de pulso, y va directamente como entrada al servo-controlador NE544 (tratado en el capitulo 1). Estos dos pulsos determinan las posiciones extremas del servomotor correspondiente en el brazo. Dado que se tiene un valor máximo de 255 (FFH) y considerando que los servos poseén 120° de amplitud de giro, a cada unidad del dato se le asocia una diferencia de 8/17° (120/255).

2.4 Teclado

El teclado es una unidad auxiliar que se halla conectada al controlador, mediante la cual pueden ingresarse datos a éste.

El valor de alguna tecla presionada puede ser obtenido accesando la dirección 8, correspondiente al puerto de entrada del teclado.

En la fig. 2.7 se muestra la distribución de teclas y el valor de cada una de ellas.

Esta unidad se halla conectada a un codificador de teclado, el cual proporciona una parte de los valores mostrados en la figura anterior. La conexión puede consultarse en la fig. 2.8.

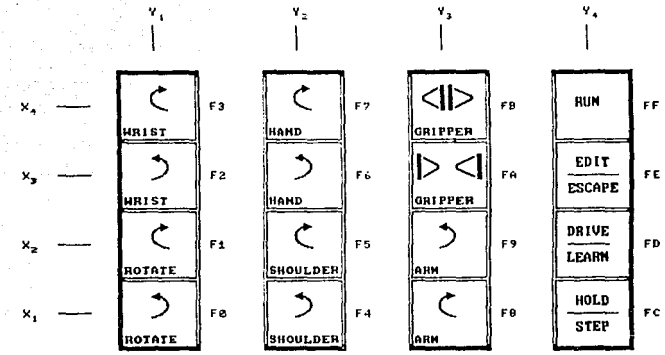


fig. 2.7 - Distribución y valores de teclas en la unidad conectada al controlador

El codificador identifica cada una de las teclas a través de pares únicos de valores (x,y), correspondientes a sus posiciones (ver fig. 2.7). De este modo, al presionar alguna tecla se obtiene como salida su código asociado, dado por los bits DCBA. Adicionalmente, la señal DA indica si alguna tecla se ha presionado, en cuyo caso adquiere un nivel lógico alto. Nótese que los tres bits más significativos de entrada al latch son permanentemente altos, de ahí que junto con la señal DA se produzca F como el dígito hexadecimal más significativo del código de las teclas.

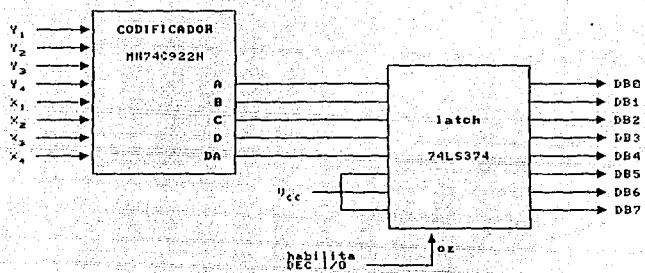


fig. 2.8 - Diagrama de conexión del circuito codificador de teclado y latch de entrada

Si ninguna tecla es presionada, el codificador mantiene los bits DCBA con el nivel de la última tecla registrada, y la señal DA en bajo. Esto origina que se obtenga un valor E* hexadecimal al accesar la dirección del teclado, donde * es el dígito menos significativo de la anterior tecla presionada.

2.5 Señal de control para la pinza neumática

Como se indicó en el punto 2.1, mediante la dirección 1001 (salida al display) se maneja la señal de control para la pinza

neumática. El efecto de dicha señal sobre la pinza se vió ya en el punto 1.6.

A dicha direccion se le asocia únicamente un latch de salida, tal y como se muestra en la fig. 2.9.

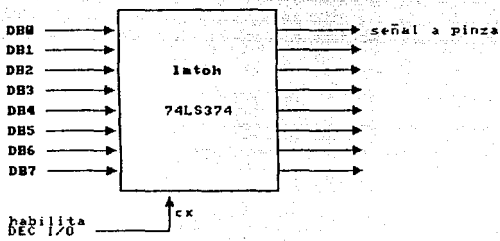


fig. 2.9 - Configuración de la direccion de salida al display, utilizada en este caso para el envio de la señal de control de la pinza neumática

2.6 Sensores

Las señales de entrada de procesos pueden ser obtenidas de sensores externos, por lo cual se incluyó información al respecto.

Los sensores son dispositivos de entrada que detectan cambios físicos en su ambiente inmediato y los transforman en variaciones de energía eléctrica.

Los sensores por su función pueden dividirse en dos categorías: sensores de estado interno y sensores de estado externo.

Los **sensores de estado interno** detectan la posición y se utilizan para el control.

Los **sensores de estado externo** detectan el alcance, la proximidad, el contacto y las características del medio ambiente. Se usan para la manipulación e identificación de los objetos.

a) **Sensor de alcance** --> Este sensor mide la distancia desde un punto de referencia hasta el objeto a detectar. Se utiliza para conocer las características de localización y forma general de objetos.

b) **Sensor de proximidad** --> Este sensor detecta la presencia de un objeto dentro de un intervalo de distancia específico.

c) **Sensor de contacto** --> Este sensor proporciona información

asociada con el contacto entre éste y los objetos. Es decir, la presencia o ausencia, la torsión y el deslizamiento de un objeto.

Existen dos tipos de sensores: **digitales** y **analógicos**.

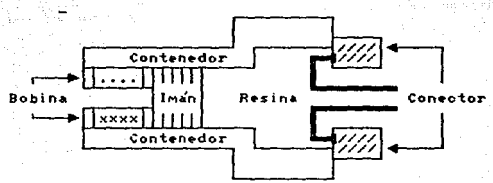
Sensores digitales.- Son los que detectan la presencia o ausencia de una condición. Cambian su estado de salida ante la presencia de un blanco adecuado dentro del radio de sensibilidad de la unidad, generando un bit cuyo valor corresponde a dicho estado. El cambio tiene lugar sin la necesidad de contacto físico. Entre estos sensores destacan:

1) **Sensores ópticos** --> Este tipo de sensores emite un haz infrarrojo sobre una superficie reflejante que regresa el haz nuevamente al sensor. Cuando un objeto aparece frente al sensor, éste obstruirá la trayectoria del haz y por lo tanto se presentará un cambio de estado.

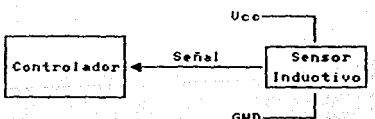
Un sensor óptico está constituido por un diodo emisor de luz de estado sólido (LED), que actúa como un transmisor de luz infrarrojo y un fotodiodo de estado sólido que actúa como el receptor.

Estos sensores tienen un radio de sensibilidad de hasta 8 metros.

2) Sensores inductivos --> Estos sensores producen un campo magnético que detecta la presencia de metales ferrosos o no ferrosos y otros materiales con características magnéticas.



a)



b)

fig. 2.10 - Sensor inductivo:
a) Esquema; b) Conexión

Un sensor inductivo consiste fundamentalmente en una bobina situada junto a un imán permanente

encapsulado en un receptáculo simple y robusto (ver fig. 2.10). El efecto de llevar el sensor a la proximidad de un material ferromagnético produce un cambio en la posición de las líneas de flujo del imán permanente. En condiciones estáticas no hay ningún movimiento de las líneas de flujo, y por consiguiente no se induce corriente en la bobina. Sin embargo, cuando un objeto ferromagnético penetra en el campo del imán o lo abandona, el cambio resultante en las líneas de flujo induce un impulso de corriente, cuya amplitud y forma son proporcionales a la velocidad de cambio en el flujo.

3) **Sensores capacitivos** --> Estos sensores detectan un cambio en la capacitancia, ocasionado por un objeto cercano al sensor. Son potencialmente capaces de detectar la presencia de cualquier material, con mayor sensibilidad a objetos con características dieléctricas.

Un sensor capacitivo está constituído por un electrodo sensible y un electrodo de referencia, separados por un material dieléctrico (ver fig. 2.11). Una cavidad de aire seco se suele colocar detrás del sensor para proporcionar aislamiento.

El resto del sensor está constituido por circuitos electrónicos que pueden incluirse como una parte integral de la unidad, en cuyo caso suelen estar cubiertos en una resina para proporcionar soporte mecánico y sellado.

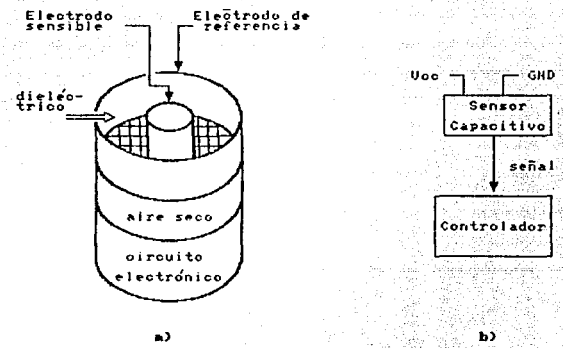


fig. 2.11 - Sensor capacitivo:
a) Esquema; b) Conexión

Sensores analógicos.- Un sensor analógico detecta las condiciones variables de un objeto. En consecuencia, el nivel de la señal de salida (voltaje o corriente) variará continuamente en proporción a la condición sensada.

La conexión de este tipo de sensores al controlador requiere de una etapa previa de conversión analógico-digital, ya que se pueden recibir únicamente señales digitales.

Entre los sensores analógicos destacan :

1) Potenciómetro --> Es una resistencia variable cuyo voltaje de salida varía en proporción directa con la rotación de su eje.

2) Sensor de altura --> Este sensor es básicamente un potenciómetro, el cual tiene una palanca; éste empujará la palanca hacia arriba haciendo rotar su eje a través de un ángulo finito. El nivel del voltaje proporcionado por el cursor del potenciómetro estará en razón directa con la altura del objeto.

3) Sensor opto-reflectivo --> Este sensor emite un rayo infrarrojo, mismo que vuelve al sensor cuando se coloca un objeto reflectivo frente a él (ver fig. 2.12). La intensidad del rayo reflejado, y en consecuencia el voltaje de salida del sensor, variarán dependiendo del color del objeto.

4) Termistor --> Es una resistencia cuyo valor óhmico varía

con respecto a la temperatura. Los termistores de coeficiente térmico positivo (CTP) incrementan su resistencia conforme aumenta la temperatura y los de coeficiente térmico negativo (CTN) disminuyen su valor resistivo a medida que aumenta la temperatura.

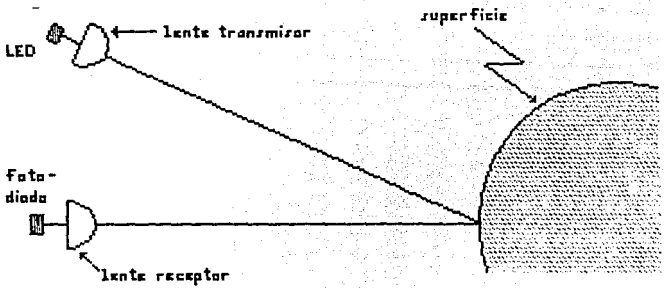


fig. 2.12 - Esquema que muestra el principio del funcionamiento de un sensor opto-reflectivo

2.7 Fuente de poder

La fuente de poder incluida en el controlador alimenta a

tres diferentes partes:

- Circuitos lógicos del controlador y del brazo (5V, 1A)
- Circuitos de la etapa de potencia del brazo (6V, 3A)
- Servos 3 y 4 (5V, 1A) .

El diagrama de dicha fuente se muestra en la fig. 2.13.

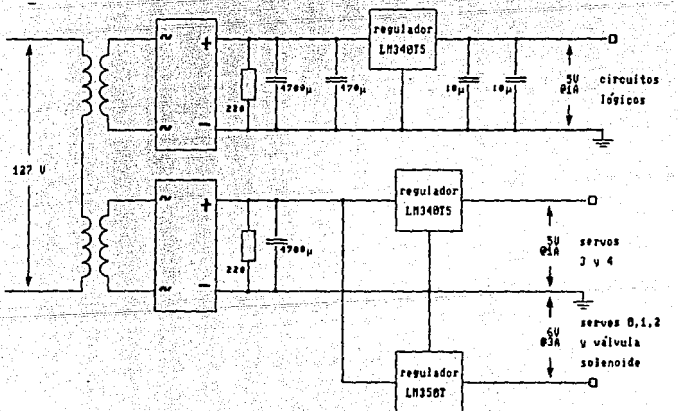
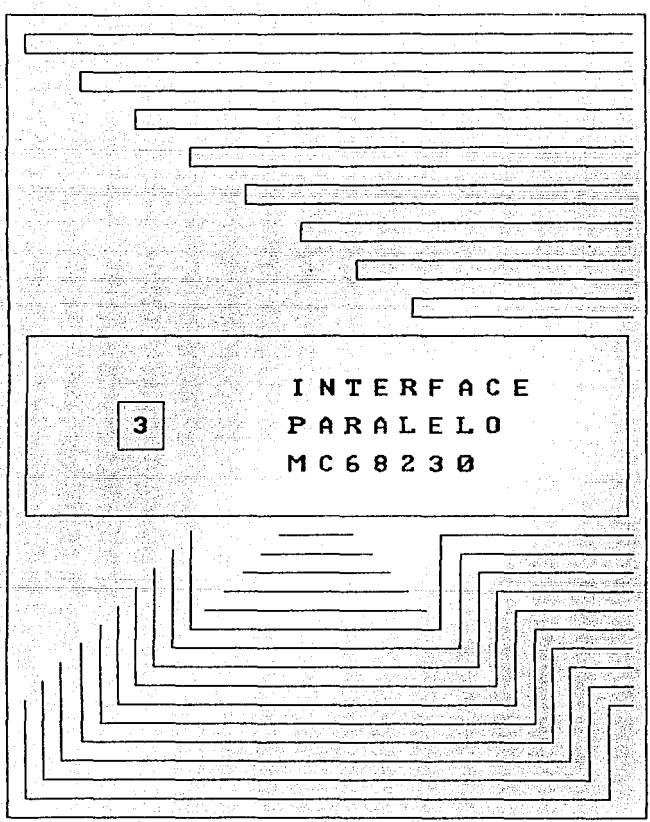


fig. 2.13 – Diagrama de la fuente de poder incluida en el controlador del brazo


3
INTERFACE
PARALELO
MC68230

El kit educacional MEX68KECB es un sistema mínimo que puede realizar funciones de una microcomputadora basada en los productos de la familia 68000.

Las características del kit pueden resumirse como:

- Frecuencia de operación de 4MHz.

- 32Kbytes de RAM dinámica.

- Dos puertos de comunicación serial (para host y terminal) compatibles con la interface RS-232C, y con capacidad de selección de velocidad de transmisión.

- Un puerto paralelo que puede ser usado para entrada/salida o como interface para impresora.

- Un puerto serial de entrada/salida para audio-cassette.

- Cuenta con firmware en ROM/EPROM, el cual consiste del sistema operativo TUTOR de 16Kbytes de tamaño. Éste cuenta con un depurador y funciones de ensamblado y desensamblado.

- Se puede enviar/recibir información a/de un host.

- Cuenta con switches para abortar programas (ABORT) y reinicializar el sistema (RESET).

El MC68230 (PI/T) es una interface para la transferencia de datos en paralelo entre los circuitos de la familia 68000 y los dispositivos periféricos.

Las principales caracteristicas del PI/T son :

- Consta de 3 puertos programables e independientes. Cada uno de ellos es de 8 bits, bidireccional y con lineas de protocolo para el control de transferencia de datos.

- Su función puede ser interface de comunicación en paralelo y/o contador/temporizador de 23 bits.

- Cuenta con 32 registros internos, de los que sólo se utilizan 23 para la programación de las caracteristicas de operación. Los primeros 14 registros se refieren a la comunicación en paralelo y los restantes al contador/temporizador.

- Cuatro modos de operación para comunicación, seleccionables por software.

3.1 Arquitectura interna del PI/T

Consiste de una interfase hacia el CPU, lógica de control de acceso directo a memoria (DMA), interrupciones y protocolo, tres puertos y un temporizador. En la fig. 3.1 se muestra el diagrama de bloques funcional del PI/T.

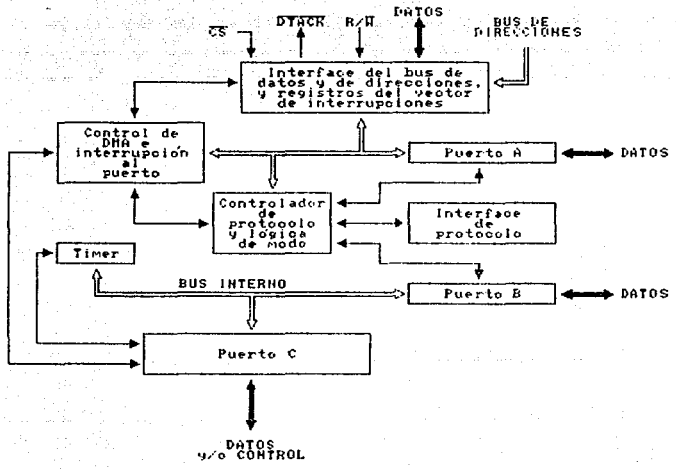


fig. 3.1 - Diagrama a bloques del circuito MC68230

3.1.1 Descripción de los registros del PI/T

En este trabajo, sólo se hará uso del PI/T como interface en paralelo.

A continuación se describe brevemente cada uno de los registros internos del PI/T:

Registro 00H. Control general de los puertos. Se selecciona el modo de operación, se habilita y definen las caracteristicas de las lineas de protocolo.

Registro 01H. Petición de atención a interrupción. Se definen las señales de control necesarias para la comunicación de datos entre el MC68000, el PI/T y los dispositivos periféricos.

Registro 02H. Dirección de lineas del puerto A. Se selecciona la dirección de cada una de las lineas del puerto. Cada bit corresponde a una linea del puerto. Si se le asigna un 0, la linea será de entrada; si se le asigna un 1, la linea se define como salida.

Registro 03H. Dirección de lineas del puerto B. Funciona igual que el registro 02H, pero para el puerto B.

Registro 04H. Dirección de lineas del puerto C. Funciona

igual que el registro 02H, pero para el puerto C.

Registro 05H. Vector de interrupciones. Almacena los 6 bits más significativos de los 8 que definen al vector de interrupciones. Los 2 bits menos significativos son proporcionados por la fuente de la interrupción, a través de las líneas de protocolo.

Registro 06H. Control del puerto A. Se selecciona el submodo del puerto A y las características de las líneas de protocolo. Este registro se debe de leer o escribir primero, debido a que la transferencia de información se hace a través de las 8 líneas.

Registro 07H. Control del puerto B. Se selecciona el submodo del puerto B y las características de las líneas de protocolo.

Registro 08H. Datos del puerto A. Se hace referencia a este registro cuando se desea leer/escribir datos en el puerto A.

Registro 09H. Datos del puerto B. Se hace referencia a este registro cuando se desea leer/escribir datos en el puerto B.

Registro 0AH. Alterno de datos del puerto A. Es un registro

alterno para la lectura de datos del puerto A. Proporciona información de los niveles lógicos en los pines correspondientes al puerto en el circuito.

Registro 0BH. Alterno de datos del puerto B. Funciona igual que el registro 0AH, pero para el puerto B.

Registro 0CH. Datos del puerto C. Funciona igual que el registro 08H, pero para el puerto C.

Registro 0DH. Estado de los puertos. Contiene el valor actual de las líneas de protocolo y del estado de los puertos. Este estado se refiere a si se puede enviar o recibir información.

3.1.2 Decodificación de los puertos

En el kit, cada registro interno de los puertos tiene asociada una dirección del mapa de memoria. Enseguida se listan las direcciones de acceso:

Registro		Dirección
00H	->	10001H
01H	->	10003H
02H	->	10005H

03H -> 10007H

04H -> 10009H

05H -> 1000BH

06H -> 1000DH

07H -> 1000FH

08H -> 10011H

09H -> 10013H

0AH -> 10015H

0BH -> 10017H

0CH -> 10019H

0DH -> 1001BH

3.2 Modos de programación

Las líneas de los puertos A y B se pueden programar como entrada, salida o líneas bidireccionales, ya sea como puertos independientes o trabajando como uno de 16 bits.

Las líneas del puerto C pueden programarse para entrada-salida o como líneas de control para el manejo de interrupciones, para el manejo del PI/T en su función de contador/temporizador o para el acceso directo a memoria.

Los modos bajo los cuales opera el PI/T son:

a) Modo 00 --> **Unidireccional de 8 bits.** Se debe seleccionar la dirección de cada una de las líneas del puerto.

b) Modo 01 --> **Unidireccional de 16 bits.** En este modo de operación el puerto A proporciona el byte de datos más significativo para entrada o salida, y el puerto B proporciona el byte de datos menos significativo. Se debe seleccionar la dirección de cada una de las líneas de los puertos.

c) Modo 10 --> **Bidireccional de 8 bits.** En este modo de operación, cada uno de los puertos tiene una diferente función:

En el puerto A se define la dirección de cada una de las líneas, la transferencia se hace sin protocolo, y el envío/recepción de datos se hace monitoreando las líneas del puerto.

En el puerto B se tiene un flujo de datos bidireccional. Las 8 líneas tienen la misma dirección.

d) Modo 11 --> **Bidireccional de 16 bits.** La comunicación en los puertos A y B es bidireccional. El puerto A proporciona el byte más significativo y el puerto B el menos significativo. Las líneas de protocolo controlan la entrada y la salida.

Los registros de dirección de las líneas de los puertos A y B no se usan en este modo, ya que la dirección se determina a través de las líneas de protocolo.

La programación del PI/T se hace a través de los registros internos.

3.3 Modo seleccionado de programación

Para la transferencia de datos entre el kit MEX68KECB y el controlador del brazo mecánico, no fue necesario hacer uso de la lógica de control de interrupciones del PI/T.

El PI/T se programó en modo de operación 01 (unidireccional de 16 bits).

Como el controlador recibe dos parámetros (dirección y dato), se seleccionó al puerto A para enviar la dirección y señales de control, y al puerto B para enviar/recibir el dato.

En el kit se encuentran alambradas las líneas del puerto A únicamente como salida, por lo que a través de él se envía la dirección a accesar y las señales R/W y SEL al controlador.

Las líneas del puerto B cambian su dirección de acuerdo a la tarea a realizar. Es decir, en modo salida cuando se envían datos al controlador, y en modo entrada cuando se lee un dato del controlador.

3.4 Registros utilizados y distribución de señales en los puertos

El dato y las instrucciones para la programación de cada uno de los registros utilizados se describe a continuación:

Registro 00H. Control general de los puertos. Se selecciona el modo de operación 01 y no se utilizan las líneas de protocolo. Su dirección de acceso es la 10001H.

bit:	7	6	5	4	3	2	1	0
valor:	0	1	0	0	0	0	0	0

Instrucciones:

```
MOVE.L #$10001,A0   ; Dir. de acceso
MOVE.B #$40,D0      ; Programa en modo 01
MOVEP.W D0,$0(A0)   ; Envía al registro
```

Registro 02H. Dirección de lineas del puerto A. Todas las lineas están alambradas como salida. Su dirección de acceso es la 10005H.

bit:	7	6	5	4	3	2	1	0
valor:	1	1	1	1	1	1	1	1

Instrucciones:

```
MOVE.L #$10005,A0  ; Dir. de acceso
MOVE.B #$FF,D0     ; Programa en modo salida
MOVEP.W D0,$0(A0)  ; Envia al registro
```

Registro 03H. Dirección de lineas del puerto B. La dirección de acceso es la 10007H.

Para el modo **entrada:**

bit:	7	6	5	4	3	2	1	0
valor:	0	0	0	0	0	0	0	0

Instrucciones:

```
MOVE.L #$10007,A0  ; Dir. de acceso
MOVE.B #$00.D0     ; Programa en modo entrada
MOVEP.W D0,$0(A0)  ; Envia al registro
```

Para el modo salida:

bit:	7	6	5	4	3	2	1	0
valor:	1	1	1	1	1	1	1	1

Instrucciones:

```
MOVE.L #$10007,A0  ; Dir. de acceso
MOVE.B #$FF,D0     ; Programa en modo salida
MOVEP.W D0,$0(A0)  ; Envia al registro
```

Registro 06H. Control del puerto A. Se deshabilitan las lineas de protocolo, las interrupciones y el acceso directo a memoria. Su dirección de acceso es la 1000DH.

bit:	7	6	5	4	3	2	1	0
valor:	0	0	0	0	0	0	0	0

Instrucciones:

```
MOVE.L #$1000D,A0  ; Dir. de acceso
MOVE.B #$00,D0     ; Control del puerto
MOVEP.W D0,$0(A0)  ; Envia al registro
```

Registro 07H. Control del puerto B. Se deshabilitan las lineas de protocolo, las interrupciones y el acceso directo a memoria. Su dirección de acceso es la 1000FH.

bit:	7	6	5	4	3	2	1	0
valor:	0	0	0	0	0	0	0	0

Instrucciones:

```
MOVE.L #$1000F,A0  ; Dir. de acceso
MOVE.B #$00,D0     ; Control del puerto
MOVEP.W D0,$0(A0)  ; Envia al registro
```

Registro 08H. Datos del puerto A. Contiene la dirección a utilizar del controlador y las señales de R/$\overline{W}$ y $\overline{SEL}$. El valor de este registro es variable. Su dirección de acceso es la 10011H.

bit:	7	6	5	4	3	2	1	0
valor:	R/$\overline{W}$	$\overline{SEL}$	*	MA4	MA3	MA2	MA1	MA0

Instrucciones:

```
MOVE.L #$10011,A0  ; Dir. de acceso
MOVEP.W D0,$0(A0)  ; Recibe en el puerto A
```

Registro 09H. Datos del puerto B. Contiene el dato de salida al controlador. El valor del dato es variable. Su dirección de acceso es la 10013H.

bit:	7	6	5	4	3	2	1	0
valor:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0

Instrucciones:

```
MOVE.L #$10013,A0      ; Dir. de acceso
MOVE.B #$<valor>,D0    ; Datos del puerto
MOVEP.W D0,$0(A0)      ; Envia al puerto B
```

3.5 Conexión MEX68KECB - controlador

En el capítulo 2 se explicó la distribución de señales como entradas en la tarjeta del controlador. Sin embargo el gabinete poseé un conector exterior, que es el que tiene relación física con el computador a través de un cable plano de 34 líneas, y cuya distribución de señales es diferente a la interna. El cruce de líneas se efectúa mediante un pequeño circuito impreso, en la forma siguiente:

Señal	Conector a PC (pin)	Conector en tarjeta del controlador (pin)
GND	1	11
R/$\overline{W}$	2	12
	3	11
	4	NC
	5	11
	6	NC
	7	11
	8	NC
	9	11
$\overline{SEL}$	10	8

	11	11
	12	NC
	13	11
	14	NC
	15	11
	16	NC
	17	11
DB0	18	13
DB1	19	15
DB2	20	17
DB3	21	19
DB4	22	20
DB5	23	18
DB6	24	16
DB7	25	14
	26	NC
MA0	27	1
MA1	28	3
MA2	29	5
MA3	30	7
MA4	31	9
	32	NC
	33	NC
	34	NC

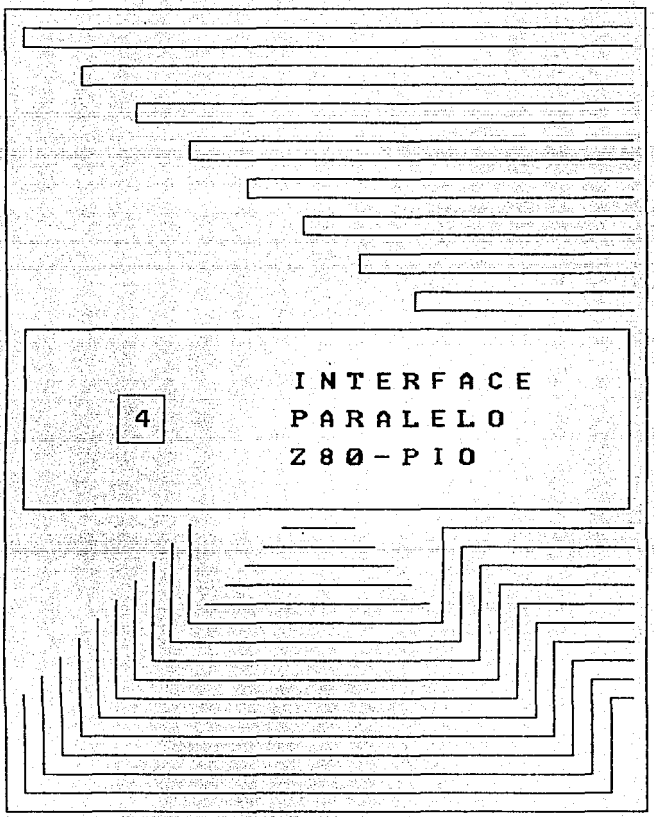
tabla 3.1 - Relación del cruce de líneas entre los conectores externo e interno del controlador

El kit MEX68KECB poseé una ceja para conexión al exterior en la tarjeta del mismo, en la que se distribuyen 50 líneas para comunicación en paralelo (provenientes del MC68230), encontrándose todas las de denominación par conectadas a tierra.

La conexión MEX68KECB - controlador se efectuó con un cable plano de 50 líneas, un conector de 50 líneas para tarjeta (para el kit), y un conector hembra tipo peine de 34 líneas (para el controlador). Se cruzaron las líneas como se indica a continuación:

Conector MEX68KECB		Conector controlador	
señal	pin	pin	señal
PB7	9	25	DB7
PB6	11	24	DB6
PB5	13	23	DB5
PB4	15	22	DB4
PB3	17	21	DB3
PB2	19	20	DB2
PB1	21	19	DB1
PB0	23	18	DB0
PA7	25	2	R/$\overline{W}$
PA6	27	10	$\overline{SEL}$
PA4	31	31	MA4
PA3	33	30	MA3
PA2	35	29	MA2
PA1	37	28	MA1
PA0	39	27	MA0
GND	40	1	GND
	el resto	NC	

tabla 3.2 - Relación de líneas entre el conector del MEX68KECB y el del controlador


4
INTERFACE
PARALELO
Z80-PIO

El kit Z80 es un sistema mínimo que permite adquirir experiencia en el manejo de microcomputadoras basadas en la familia Z80.

El kit tiene como características:

- Frecuencia de operación de 2MHz.
- 1 Kbyte de memoria RAM.
- Una interface de entrada/salida en paralelo que proveé de dos puertos.
- Un circuito contador/temporizador.
- Cuenta con teclado y displays de 7 segmentos.
- Una interface para salida a audio-cassette.
- Programador de memorias EPROM 2716/2758.
- Sistema operativo ZBUG de 2Kbytes de tamaño.
- Poseé un botón para reinicializar al sistema (RESET).

El Z80-PIO es una interface para la transferencia de datos en paralelo entre el Z80-CPU y los dispositivos periféricos.

El PIO puede conectarse con una gran variedad de circuitos periféricos que no requieren lógica adicional; entre los más típicos se encuentran teclados, lectoras de cinta de papel, impresoras, programadores de PROM, LEDs, switches, motores de pasos, y en este caso un controlador de brazo mecánico.

Las principales características del PIO son :

- Consta de dos puertos programables e independientes. Cada uno de ellos es de 8 bits, bidireccional y con líneas de protocolo para el control de transferencia de datos.

- Interrupción programada en las líneas de control para una rápida tarea de entrada-salida.

- Lógica de interrupción con prioridades tipo cadena "daisy chain". Esto proveé un mecanismo de interrupciones vectorizadas automático que no requiere de lógica externa.

- Cuatro modos de operación seleccionables por software.

4.1 Arquitectura interna del PIO

Consiste de una interface hacia el CPU, lógica de control interna, un bloque de control de interrupciones y dos puertos, cada uno de los cuales está compuesto de 6 registros con lógica de control incluida.

En la fig. 4.1 se muestra el diagrama de bloques funcional del PIO.

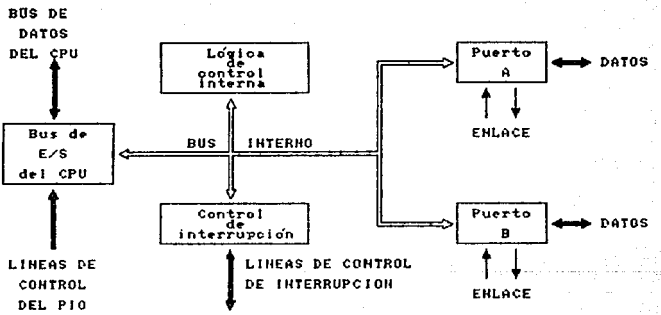


fig. 4.1 - Diagrama a bloques del circuito Z80-PIO

4.1.1 Descripción interna de los puertos

La fig. 4.2 muestra el diagrama de bloques funcional de un puerto del PIO.

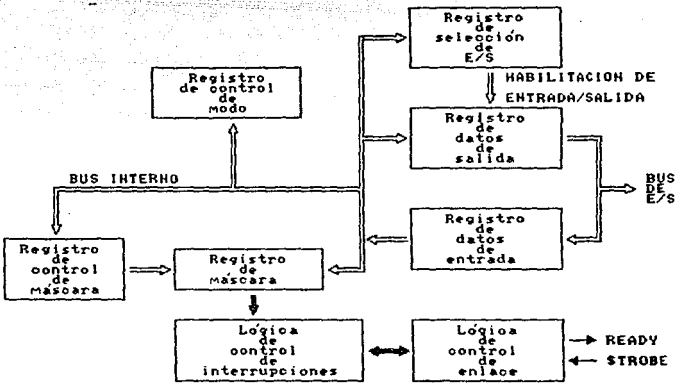


fig. 4.2 - Diagrama a bloques funcional de un puerto

A continuación se describe brevemente cada uno de los registros de un puerto:

- Registro de control de modo.- Palabra que carga el CPU para seleccionar el modo de operación.

- **Registro de datos de salida.**- Palabra de información que envía el CPU al dispositivo periférico.

- **Registro de datos de entrada.**- Palabra de información que envía el dispositivo periférico al CPU.

- **Registro de selección de entrada-salida.**- Sólo se utiliza en el modo 3 de operación. Palabra que carga el CPU para indicar la dirección (entrada o salida) de cada una de las líneas del puerto.

- **Registro de control de máscara.**- Sólo se utiliza en el modo 3 de operación. Palabra que carga el CPU para especificar el nivel activo (alto o bajo) de alguna de las líneas de un dispositivo periférico que se esté monitoreando, indica también si una interrupción puede ser generada cuando todas las líneas están activas (condición AND) o bien cuando alguna de ellas está activa (condición OR).

- **Registro de máscara.**- Sólo se utiliza en el modo 3 de operación. Palabra cargada por el CPU para seleccionar las líneas de un dispositivo periférico que se esté monitoreando para una condición de estado específica.

4.1.2 Decodificación de los puertos

En el kit, las direcciones de los puertos del PIO son:

80H -> Datos de entrada-salida del puerto A

81H -> Datos de entrada-salida del puerto B

82H -> Datos de control del puerto A

83H -> Datos de control del puerto B

4.2 Modos de programación

El PIO opera bajo los siguientes modos:

Modo 0.- Conocido como modo salida; se envía la información proveniente del CPU hacia los dispositivos periféricos a través de las 8 líneas del puerto.

Modo 1.- Conocido como modo entrada; se lee la información enviada por los dispositivos periféricos hacia el CPU a través de las 8 líneas del puerto.

Modo 2.- Es el modo bidireccional. sirve solamente para el puerto A, ya que utiliza las 4 señales de protocolo: las dos del puerto A y las dos del puerto B. Al seleccionar este modo de operación, el puerto B debe estar programado en modo bit.

Modo 3 .- Conocido como modo bit, se selecciona en forma independiente la dirección (entrada = 1 o salida = 0) de cada una de las 8 líneas del puerto.

La programación del PIO se hace a través de los registros internos de los puertos.

El PIO se diseñó para operar con el Z80-CPU usando el modo 2 de interrupción. Este modo requiere que un vector de interrupciones sea proporcionado por el dispositivo que interrumpe al CPU en el momento en que se reconoce la interrupción. Este vector es usado por el CPU para formar la dirección de la rutina de servicio de la interrupción.

El vector de interrupción se escribe en el registro de control del puerto seleccionado. El bit menos significativo debe tener como valor un 0, los demás indican la dirección.

Para seleccionar el modo de operación, el registro de control de modo debe tener el siguiente formato:

bit:	7	6	5	4	3	2	1	0
valor:	M1	M0	*	*	1	1	1	1

donde:

M1	M0		
0	0	-->	Modo 0 (Salida)
0	1	-->	Modo 1 (Entrada)
1	0	-->	Modo 2 (Bidireccional)
1	1	-->	Modo 3 (Bit)

Los bits 0, 1, 2 y 3 le indican al PIO que se está seleccionando el modo de operación.

En los modos 0, 1 y 2 las 8 lineas quedan programadas de igual manera. En cambio, en el modo 3 se debe indicar qué líneas serán de entrada y cuáles de salida. Para ésto se debe de escribir a continuación el registro de selección de entrada-salida, recordando que:

- Un 1 lógico significa que la linea es de entrada.
- Un 0 lógico significa que la linea es de salida.

La palabra de control de interrupción tiene el siguiente formato:

bit:	7	6	5	4	3	2	1	0
valor:	INT H/$\overline{D}$	AND/$\overline{OR}$	H/$\overline{L}$	MASCARA	0	1	1	1

El bit 7 habilita/deshabilita las interrupciones cuando tiene como valor 1/0, respectivamente.

El bit 6 solamente se utiliza en el modo 3 de operación y especifica el criterio a seguir para generar una interrupción. Un 1 lógico implica que todas las líneas deben ser activas para que se genere una interrupción, es decir se realiza una función AND. Un 0 lógico especifica que es suficiente la presencia de una sola línea para que se genere una interrupción, es decir se realiza una función OR.

El bit 5 solamente se utiliza en el modo 3 de operación y define el nivel lógico que se va a considerar activo (1/0).

Si el bit 4 está en 1 y se seleccionó el modo 3 de operación, entonces a continuación se deberá escribir el registro de máscara.

Aquellas líneas del puerto cuyos bits asociados de máscara estén a 1 lógico serán monitoreados con el propósito de generar interrupciones (realizando la función AND/OR).

Si no se seleccionó el modo 3 de operación y el bit 4 tiene valor de 1, se cancelarán las interrupciones pendientes.

4.3 Modo seleccionado de programación

Para realizar la comunicación entre el kit Z80 y el controlador del brazo mecánico, no fue necesario hacer uso de la lógica de control de interrupciones del PIO.

Al igual que para el MEX68KECB, se seleccionó al puerto A para enviar la dirección y señales de control, y al puerto B para enviar/recibir el dato.

El puerto A se programó como modo 0 (salida); por él se direcciona al servo y se envían las señales de R/W y de SEL al controlador.

Debido a que la entrada/salida de los datos no se hace de manera simultánea, fue necesario programar al puerto B en dos modos diferentes de operación, los cuales se conmutan de acuerdo a la tarea a realizar. Es decir, en modo 0 cuando se envían datos al controlador, y en modo 1 cuando se lee un dato del controlador.

4.4 Registros utilizados y distribución de señales en los puertos

A continuación se describe el formato de los registros de cada puerto del PIO, así como la forma de programarlos usando las instrucciones del lenguaje ensamblador del Z80-CPU:

Puerto A:

- Registro de control de modo --> Se seleccionó el modo 0 (salida) y se envió a través del puerto 82H el valor de 0FH.

bit:	7	6	5	4	3	2	1	0
valor:	0	0	0	0	1	1	1	1

Instrucciones:

```
LD A,0FH      ; Carga al acumulador el modo 0
OUT (82H),A   ; Programa el puerto A
```

- Registro de datos de salida --> Este valor es variable y se envía a través del puerto 80H.

bit:	7	6	5	4	3	2	1	0
valor:	R/$\overline{W}$	$\overline{SEL}$	0	MA4	MA3	MA2	MA1	MA0

donde:

MA4, MA3, MA2, MA1, MA0 indican la dirección a acesar en el controlador.

Instrucciones:

```
LD A,<valor>    ; Carga al acumulador el valor
OUT (80H),A     ; Da salida al dato
```

Puerto B:

- Registro de control de modo:

Para el modo 0 (salida) se envia a través del puerto 83H el valor de 0FH.

bit:	7	6	5	4	3	2	1	0
valor:	0	0	0	0	1	1	1	1

Instrucciones:

```
LD A,0FH        ; Carga al acumulador el modo 0
OUT (83H),A     ; Programa al puerto B
```

Para el modo 1 (entrada) se envia a través del puerto 83H el valor de 4FH.

bit:	7	6	5	4	3	2	1	0
valor:	0	1	0	0	1	1	1	1

Instrucciones:

```
LD A,4FH        ; Carga al acumulador el modo 1
OUT (83H),A     ; Programa al puerto B
```

- Registro de datos de Entrada/Salida --> El valor de este registro es variable. Se tiene acceso a él a través del puerto 81H.

bit:	7	6	5	4	3	2	1	0
valor:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0

Instrucciones:

En el caso de la lectura de datos:

```
IN A,(81H)      ; El periférico proporciona un dato
```

En el caso de la escritura de datos :

```
LD A,<valor>    ; Carga al acumulador el valor
OUT(81H),A      ; Envia el dato al periférico
```

4.5 Conexión kit Z80 - controlador

Se deben de tener en cuenta las consideraciones del punto 3.5, referentes al conector externo del controlador.

Para poder efectuar la conexión kit Z80 - controlador hubo de armarse un conector de peine de doble hilera, ya que el área de alambrado del kit no permitía ensamblar un conector comercial de este tipo, el cual había de corresponder al montado en el cable PC - controlador.

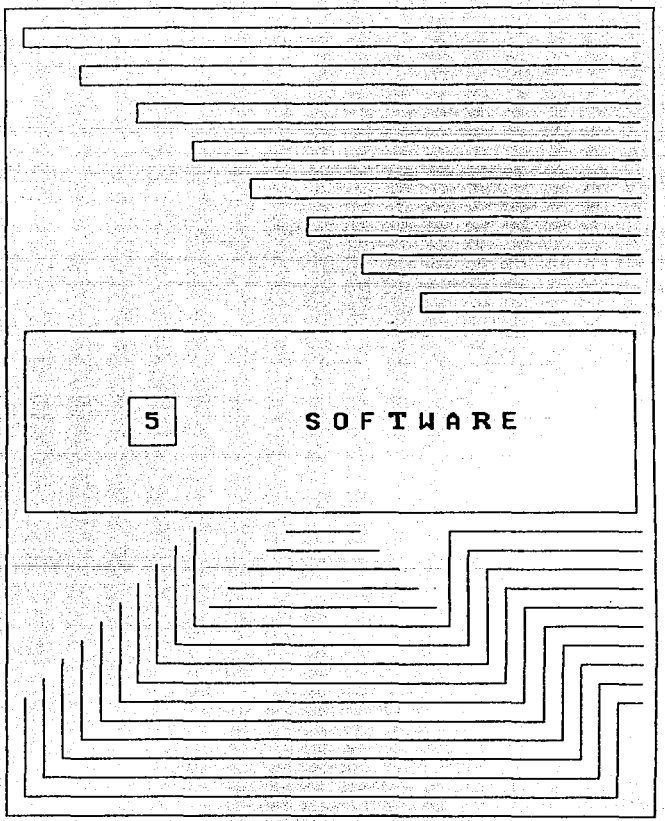
Así, en la conexión kit Z80 - controlador se empleó el cable que une al PC y al controlador, que consta de cable plano de 34 líneas y un par de conectores hembra tipo peine de 34 líneas. En éste la correspondencia de pines es uno a uno.

La distribución de líneas y su relación con el Z80-PIO se muestra en la siguiente tabla:

Conector kit Z80		Conector controlador	
señal	pin	pin	señal
GND	1	1	GND
PA7	2	2	R/$\overline{W}$
PA6	10	10	$\overline{SEL}$
PB0	18	18	DB0
PB1	19	19	DB1

PB2	20	20	DB2
PB3	21	21	DB3
PB4	22	22	DB4
PB5	23	23	DB5
PB6	24	24	DB6
PB7	25	25	DB7
PA0	27	27	MA0
PA1	28	28	MA1
PA2	29	29	MA2
PA3	30	30	MA3
PA4	31	31	MA4

tabla 4.1 - Relación de señales entre el conector del kit Z80 y el del controlador


5
SOFTWARE

Con el software de control se pretende manejar al brazo mecánico de un modo sencillo a través del teclado del controlador, accionando cada uno de los servomotores para alcanzar cierta posición. El sistema se diseñó en base a subrutinas, incluyendo las siguientes características:

a) Programa principal que monitorea el teclado del controlador, y de acuerdo con el valor obtenido ejecuta una función específica.

b) Rutinas que llevan a cabo acciones bien definidas:

- Inicialización del controlador.
- Salida de datos a la interface paralelo del kit.
- Lectura del teclado.
- Actualización de la posición del brazo de acuerdo con la tecla oprimida.
- Modo de aprendizaje.
- Salida secuencial y en ciclo de las posiciones previamente aprendidas.
- Salida sólo a la siguiente posición aprendida.
- Retardo variable de propósito general, utilizado por otras rutinas.

5.1 Diseño del software de control

Este diseño es general, y válido para los dos microprocesadores empleados (MC68000 y Z80).

El teclado del controlador puede ser dividido en dos partes:

1) **Teclas de movimiento del brazo** --> Son aquéllas que indican qué parte del brazo ha de moverse. Sus valores van de F0H a FBH inclusive (según fig. 2.7).

2) **Teclas de control** --> Son aquéllas que indican alguna función a realizar. Se localizan sobre la columna derecha extrema del teclado, cuyos valores van de FCH a FFH inclusive *.

Al momento de presionar alguna tecla es necesario obtener la siguiente información:

a) Conocer si la tecla en cuestión es de control o de movimiento. Esto puede realizarse con una simple comparación dentro del programa, ya que se tiene un valor límite entre los dos rangos de teclas.

* Las teclas marcadas como HOLD/STEP, DRIVE/LEARN y EDIT/ESCAPE (valores FCH a FEH) serán tratadas en este sistema de acuerdo con la segunda función que indican (STEP, LEARN y ESCAPE), dadas las características del mismo. En el caso de que llegara a desarrollarse un futuro sistema o una modificación sobre el presente, las funciones HOLD, DRIVE y EDIT podrían ser utilizadas, dependiendo de los alcances que se pretendieran.

b) Si la tecla resulta ser de control, se tienen sólo cuatro valores (FCH a FFH) a comparar para decidir la función a ejecutar.

c) Si la tecla resulta ser de movimiento, habrá de determinarse a qué parte del brazo se refiere y el sentido del giro a efectuar. Dado que se manejan 12 diferentes valores (F0H a FBH), sería poco práctico estar haciendo comparaciones de la tecla contra cada uno de esos valores. Además, para cada caso de igualdad habrían de asignarse los dos datos anteriormente mencionados.

Esto provocaría que el programa creciera mucho y fuera confuso su seguimiento. En su lugar se determinó el uso de una tabla de desplazamientos, la cual se explica más adelante.

El software maneja la información en base a dos tablas:

a) **Tabla de posiciones.-** Se maneja en memoria una tabla de posiciones del brazo, como se muestra en la fig. 5.1. Como puede verse, un bloque de datos (o paso) se compone de seis bytes, que indican la posición de cada uno de los servos y de la pinza, y siempre en orden ascendente conforme a su dirección en el controlador.

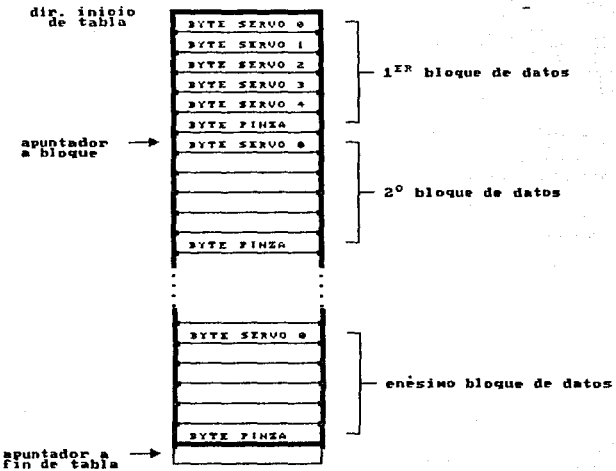


fig. 5.1 - Configuración de la tabla de posiciones

Al inicializar el controlador, el brazo adopta una cierta posición inicial conocida (como se vio en el punto 2.1 al hablar sobre las señales de dirección). En este momento se crea la tabla de posiciones, incluyéndose solamente un bloque de datos, cuyos valores representan esta posición. Cualquier tecla de movimiento que se presione afecta al byte correspondiente dentro de este bloque, en

tanto no sea seleccionada la función de copiado, la cual adiciona a la tabla un bloque idéntico al que se halla en uso.

El valor del apuntador a bloque indica la dirección de inicio del bloque en uso, la cual corresponde a la del byte del servo 0. Cuando apenas se ha inicializado el controlador, éste apunta al inicio de la tabla; posteriormente, cuando se tienen más de un bloque de datos, éste apunta al inicio del bloque en uso.

El valor del apuntador a fin de tabla es en todo momento la dirección siguiente a la del último byte dentro de la tabla.

b) **Tabla de desplazamientos.**- Una vez que se sabe que una tecla de movimiento del brazo ha sido presionada, es necesario identificar el servo al cual se refiere y el sentido del giro a realizar. Para ésto se utiliza una tabla con la configuración mostrada en la *fig. 5.2.*

La información contenida en esta tabla es permanente, y sólo es utilizada para consulta.

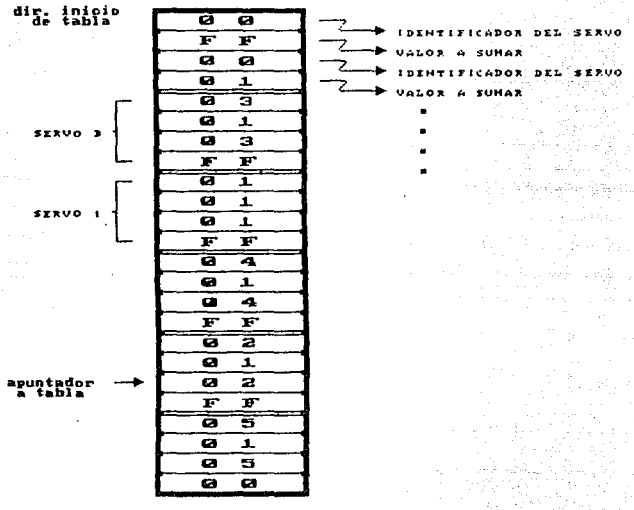


fig. 5.2 - Configuración de la tabla de desplazamientos

La nueva posición del servo en cuestión vendrá dada por la posición actual (indicada en el bloque correspondiente de la tabla de posiciones) más el valor a sumar. Nótese que los valores FF hexadecimal corresponden a -1 decimal.

Para la pinza (identificador 05), el valor que le sigue en la tabla proporciona directamente su nueva posición (0 o

1), ya que se manejan sólo dos posibles estados (abierto o cerrado).

El orden en que aparecen los identificadores de los servos está en razón directa a aquél que guardan los valores de las teclas (ver disposición en la fig. 2.7).

Cuando el apuntador a la tabla de desplazamientos es cargado con la dirección de inicio de ésta, más el dígito menos significativo del valor de la tecla multiplicado por 2 (dado que para cada servo se manejan 2 bytes), se estará accesando la dirección en la que se halla almacenado el identificador de la parte a mover del brazo. Incrementando en uno el valor de este apuntador se accesa la dirección que proporciona el valor a sumar á la posición actual.

Los programas que componen al sistema se explican enseguida, siendo apoyados por diagramas de flujo para su mejor comprensión, y presentando la programación de cada rutina en su parte correspondiente al MC68000:

PROGRAMA PRINCIPAL

Como es de esperarse, define la estructura principal del sistema:

a) La primera tarea consiste en inicializar al controlador, para poder efectuar cualquier operación posterior sobre éste; la lleva a cabo por medio de la rutina INICIALIZA.

b) Monitorea el teclado del controlador e identifica el dato obtenido. Si ninguna tecla ha sido utilizada (valor menor a F0H), regresa a repetir la operación de lectura.

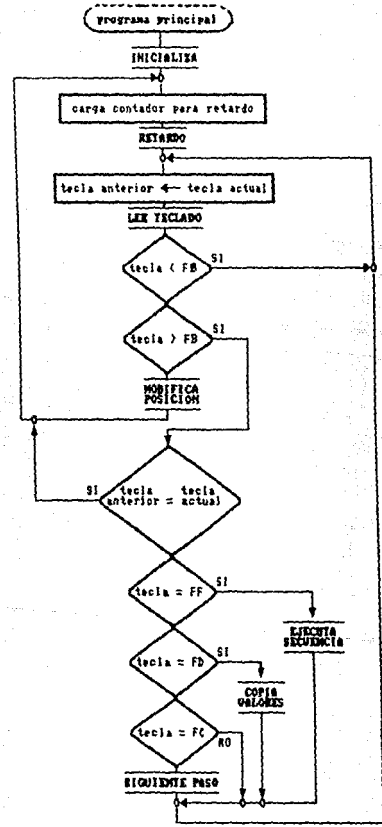
c) Si la tecla presionada es de control (valor mayor a FBH), invoca a la rutina asociada:

- EJECUTA SECUENCIA --> Tecla RUN (valor FFH)
- COPIA VALORES --> Tecla LEARN (valor FDH)
- SIGUIENTE PASO --> Tecla STEP (valor FCH) .

La tecla ESCAPE (valor FEH) no tiene efecto en esta parte del sistema, se le emplea solamente dentro de la rutina EJECUTA SECUENCIA.

Una vez finalizada la rutina correspondiente, regresa a monitorear el teclado. Si la tecla de control que se está identificando como presionada, resulta ser igual a la que provocó el llamado de una rutina apenas terminada, su función es inhibida en tanto no deje de presionarse. Esto evita que la rutina asociada sea ejecutada un sinnúmero de veces, como resultado de un múltiple monitoreo durante el tiempo que permanece activa la tecla.

d) Si la tecla presionada resulta ser para el movimiento del brazo (valor menor o igual a FBH), entonces realiza un llamado a la rutina MODIFICA POSICION. Se permite la ejecución repetida de esta rutina durante el tiempo que se halle presionada la tecla; se efectúa un retardo (rutina RETARDO) entre cada llamado, proporcionando así un tiempo razonable como para que no continúe modificándose la posición del brazo una vez que se suelta la tecla.


programa principal
INICIALIZA
carga contador para retardo
RETARDO
tecla anterior ← tecla actual
LEE TECLADO
tecla < FB
SI
tecla > FB
SI
MODIFICA POSICION
tecla = tecla anterior = actual
SI
tecla = FF
SI
EJECUTA SECUENCIA
tecla = FD
SI
COPIA VALORES
tecla = FC
NO
SIGUIENTE PASO

```
        JSR     INICIALIZA          ; rutina de inicialización
ET03:   MOVE.L  #4096,D1            ; dato para retardo
        JSR     RETARDO             ; rutina de retardo
ET01:   MOVE.B  D5,D4               ; valor tecla anterior
        JSR     LEE_TECLADO         ; rutina lectura teclado
        MOVE.B  D0,D5               ; valor tecla actual
        CMP.B   #$F0,D0             ; tecla no presionada?
        BLT     ET01
        CMP.B   #$FB,D0             ; tecla es de control?
        BGT     ET02
        JSR     POSICION            ; actualiza posición
        BRA     ET03
ET02:   CMP.B   D4,D5               ; tecla anterior = actual ?
        BEQ     ET03
        CMP.B   #$FF,D0             ; tecla = RUN ?
        BNE     ET04
        JSR     EJECUTA_SEC         ; rutina de ejecución
        BRA     ET01
ET04:   CMP.B   #$FD,D0             ; tecla = LEARN ?
        BNE     ET05
        JSR     COPIA_VAL           ; rutina de copiado
        BRA     ET01
ET05:   CMP.B   #$FC,D0             ; tecla = STEP ?
        BNE     ET01
        JSR     SIG_PASO            ; rutina siguiente posición
        BRA     ET01
```

Rutina INICIALIZA

Establece las condiciones iniciales para el sistema:

a) Programa a los puertos A y B para el envío de datos.

b) Carga dirección 1111 (0FH) para inicializar al controlador, y el dato FFH como posición inicial de los servomotores. Envía el par dirección-dato por medio de la rutina SALIDA.

c) Carga valores de las posiciones iniciales de los servomotores y de la pinza en la tabla de posiciones, así como de los apuntadores correspondientes (apuntador a bloque y apuntador a fin de tabla).

d) Indica que ninguna tecla ha sido presionada previamente.

Rutina SALIDA

Se emplea para el envío de la dirección y dato al controlador del brazo.

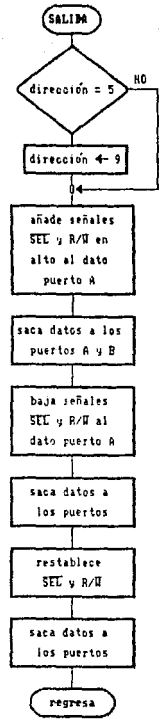
Por facilidad en el manejo de la información, en el software se le está asociando a la pinza un identificador de 5, el cual no corresponde a su dirección real en el controlador (dirección 9). Así, la rutina primeramente checa si la dirección a que se está haciendo referencia es la 5, en cuyo caso es sustituida por un valor de 9. A continuación se siguen tres etapas (cualquier dirección):

a) A los bits de dirección añade las señales de SEL y $R/\overline{W}$ en alto, y envía junto con los bits del dato hacia los puertos A y B con el fin de que las señales se encuentren estables al momento de ser cargadas al controlador.

b) Modifica los niveles de las señales SEL y $R/\overline{W}$, y vuelve a dar salida al par dirección-dato.

c) Restablece niveles para SEL y $R/\overline{W}$, y nuevamente da salida.


INICIALIZA
programa puertos A y B como salida
dato puerto A ← 0F
dato puerto B ← FF
SALIDA
carga apuntador a inicio de tabla de posiciones
almacena en tabla las posiciones iniciales
carga apuntador a fin de tabla
tecla ← 0
regresa


SALIDA
dirección = 5
NO
dirección ← 9
añade señales SEL y R/W en alto al dato puerto A
saca datos a los puertos A y B
baja señales SEL y R/W al dato puerto A
saca datos a los puertos
restablece SEL y R/W
saca datos a los puertos
regresa

```
INICIALIZA: MOVE.L   #$10001,A0        ; dir. base PIT
            MOVEP.L  $0000(A0),D0      ; valor original de regs.
            AND.L    #$00FF0000,D0
            OR.L     #$4000FFFF,D0     ; puertos A y B como salida
            MOVEP.L  D0,$0000(A0)      ; saca palabra de control
            MOVE.W   #0,D0             ; regs. control de A y B
            MOVEP.W  D0,$000C(A0)
            MOVE.W   #$0FFF,D0         ; dato de inicialización
            JSR      SALIDA            ; rutina de salida
            MOVE.L   #$2000,A2         ; dir. inicio tabla posición
            MOVE.L   #$FFFFFFFF,(A2)   ; posiciones iniciales
            MOVE.W   #$FF01,4(A2)      ; Idem
            MOVE.L   #$2006,D3         ; apuntador fin de tabla
            MOVE.B   #0,D5             ; valor inicial de tecla
            RTS


SALIDA:     MOVE.W   D0,D1             ; copia dato p/manejo int.
            CMP.W    #$0500,D1         ; dirección de pinza?
            BLT      ET06
            CMP.W    #$0501,D1         ; dirección de pinza?
            BGT      ET06
            AND.W    #$00FF,D1         ; sustituye direc. 05 ...
            OR.W     #$0900,D1         ; ... por 09
  ET06:     AND.W    #$0FFF,D1         ; se añaden señales ...
            OR.W     #$C000,D1         ; ... de control
            MOVEP.W  D1,$0010(A0)      ; salida a puertos A y B
            AND.W    #$0FFF,D1         ; SEL y R/W bajas
            MOVEP.W  D1,$0010(A0)
            OR.W     #$C000,D1         ; se restablecen señales
            MOVEP.W  D1,$0010(A0)
            RTS
```

Rutina LEE TECLADO

Obtiene el dato que se halla en el latch de entrada correspondiente al teclado en el controlador. Para ello sigue los pasos:

a) Modifica el registro de control del puerto B (puerto asociado al dato), de modo que permita el ingreso de información.

b) Carga un valor de 8 como dirección (valor asociado al latch del teclado), añade señales SEL y R/W en alto y envía al puerto A con fines de estabilización.

c) Modifica nivel de SEL y da salida nuevamente al puerto A.

d) Hace uso de un pequeño retardo para dar tiempo a que las señales provenientes del latch del teclado sean estables al momento de su lectura. Esta parte podría llegar a ser eliminada, sin embargo se le utiliza como una forma de seguridad.

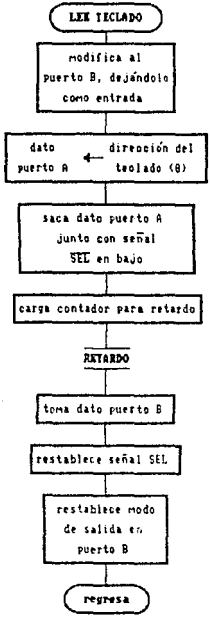
e) Toma el dato que ha ingresado al puerto B; éste representa el valor de la tecla presionada (o no presionada).

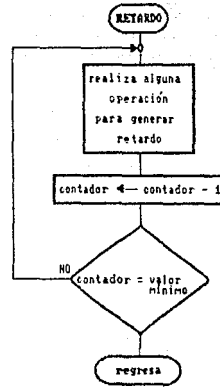
f) Restablece señal SEL y vuelve a enviar al puerto A.

g) Nuevamente modifica el registro del control del puerto B, volviendo al modo de salida, ya que se está considerando así por omisión.

Rutina RETARDO

Proporciona un tiempo de retraso, el cual varía de acuerdo con el valor especificado en el punto en que se efectuó su llamado. Consta únicamente de un ciclo repetitivo controlado por el valor mencionado, durante el cual lleva a cabo una operación cuyo resultado no importa, ya que el objetivo es obtener simplemente retardo.


LEE TECLADO
modifica al puerto B, dejándolo como entrada
dato puerto A ← dirección del teclado (8)
saca dato puerto A junto con señal SEL en bajo
carga contador para retardo
RETARDO
toma dato puerto B
restablece señal SEL
restablece modo de salida en puerto B
regresa


RETARDO
realiza alguna operación para generar retardo
contador ← contador - 1
NO
contador = valor mínimo
regresa

```
LEE_TECLADO:  MOVEP.L  $0000(A0),D0      ; regs. control PIT
              AND.L    #$FFFFFF00,D0     ; puerto B como entrada
              MOVEP.L  D0,$0000(A0)
              MOVE.W   #$C8AA,D0         ; B -> dir. de teclado
              MOVEP.W  D0,$0010(A0)      ; salida a puertos A y B
              AND.W    #$BFFF,D0         ; SEL baja
              MOVEP.W  D0,$0010(A0)
              MOVE.L   #1,D1             ; dato para retardo
              JSR      RETARDO           ; rutina de retardo
              MOVEP.L  $0010(A0),D0      ; dato de entrada
              OR.W     #$C000,D0         ; restablece SEL
              MOVEP.W  D0,$0010(A0)
              MOVEP.L  $0000(A0),D1      ; restablece modo de ...
              OR.L     #$000000FF,D1     ; ... salida en ...
              MOVEP.L  D1,$0000(A0)      ; ... puerto B
              RTS

RETARDO:      MOVE.L   #$FFFF,D2         ; valor cualquiera
              DIVS.W   D2,D2             ; cada instruc. DIVS ...
              DIVS.W   D2,D2             ; ... proporciona 150 ciclos
              DBEQ.L   D1,RETARDO
              RTS
```

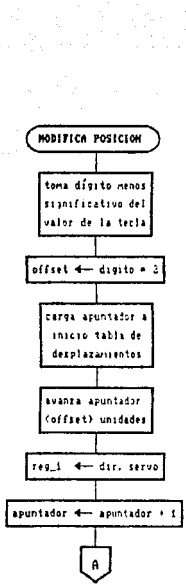
Rutina MODIFICA POSICION

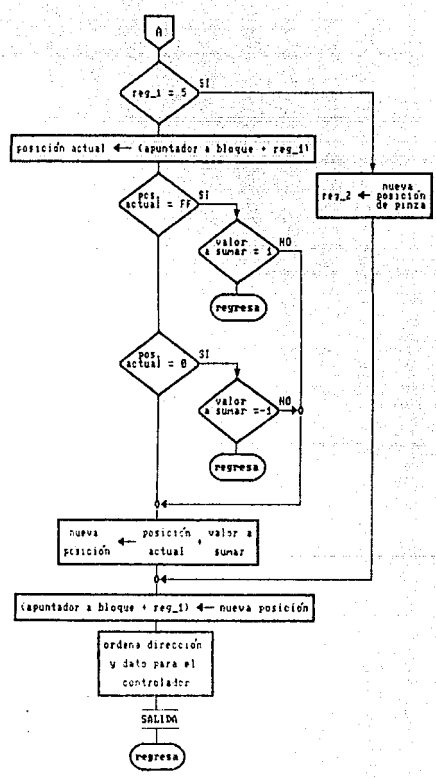
Modifica la posición del servo o pinza indicado por la tecla que se ha presionado, y actualiza su valor en la tabla de posiciones. Para lograrlo efectúa lo siguiente:

a) Toma el digito menos significativo del valor de la tecla presionada (único requerido para identificarla), el cual multiplicado por 2 y sumado a la dirección inicial de la tabla de desplazamientos, proporciona la dirección dentro de la misma tabla que contiene el identificador del servo o pinza correspondiente. En la siguiente dirección se encuentra el valor que ha de ser sumado a la posición actual para obtener la nueva posición, excepto para la pinza (como ya se explicó al definir la tabla de desplazamientos).

b) Si el identificador se refiere a la pinza (valor 5), entonces carga la nueva posición, actualiza el valor en el bloque correspondiente dentro de la tabla de posiciones y hace un llamado a la rutina SALIDA.

c) Si el movimiento es para alguno de los servos, primero obtiene la posición actual, que se encuentra en el bloque en uso dentro de la tabla de posiciones. Checa que la nueva posición no vaya a encontrarse fuera de los valores permitidos (es decir, que no exceda a FFH ni se halle por debajo de 0). Obtiene el valor de la nueva posición, actualiza en la tabla de posiciones y envia al controlador mediante un llamado a la rutina SALIDA.


MODIFICA POSICION
toma dígito menos significativo del valor de la tecla
offset ← digito * 2
carga apuntador a inicio tabla de desplazamientos
avanza apuntador (offset) unidades
reg_1 ← dir. servo
apuntador ← apuntador + 1
A


A
reg_1 = 5
SI
posición actual ← (apuntador a bloque + reg_1)
pos. actual = FF
SI
valor a sumar = 1
NO
regresa
reg_2 ← nueva posición de pinza
pos. actual = 0
SI
valor a sumar =-1
NO
regresa
nueva posición ← posición actual + valor a sumar
(apuntador a bloque + reg_1) ← nueva posición
ordena dirección y dato para el controlador
SALIDA
regresa

```
POSICION:   MOVE.L   #0,D1              ; limpia registro
            MOVE.B   D0,D1              ; valor tecla
            AND.B    #$0F,D1            ; no interesa digito izq.
            MULU.W   #2,D1              ; son 2 bytes por c/servo
            MOVE.L   #$3000,A3          ; dir. tabla desplazamientos
            ADD.W    D1,A3
            MOVE.W   #0,D0              ; limpia
            MOVE.B   (A3)+,D0           ; servo correspondiente
            CMP.B    #$05,D0            ; es pinza?
            BNE      ET07
            MOVE.B   (A3),D1            ; nueva posición
            BRA      ET08
ET07:       MOVE.B   0(A2,D0.W),D1      ; posición actual
            CMP.B    #$FF,D1            ; se halla al máximo?
            BNE      ET09
            CMP.B    #1,(A3)            ; valor a sumar es 1 ?
            BEQ      ET10
            BRA      ET11
ET09:       CMP.B    #0,D1              ; se halla al mínimo?
            BNE      ET11
            CMP.B    #1,(A3)            ; valor a sumar es -1 ?
            BNE      ET10
ET11:       ADD.B    (A3),D1            ; pos. actual + desplazam.
ET08:       MOVE.B   D1,0(A2,D0.W)      ; almacena nueva posición
            LSL.W    #8,D0              ; recorre bits de dirección
            OR.B     D1,D0              ; agrega posición
            JSR      SALIDA             ; rutina de salida
ET10:       RTS
```

Rutina EJECUTA SECUENCIA

Su finalidad es ir posicionando al brazo de acuerdo con la información contenida en la tabla de posiciones, y conforme a un ciclo repetitivo.

a) Posiciona al apuntador a bloque al inicio de la tabla.

b) Asigna un valor de 0 como primera dirección (primer servo).

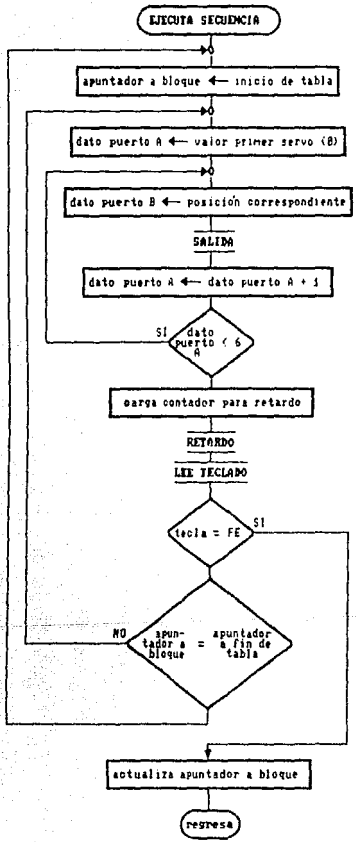
c) Carga el dato del servo (posición) y envía el par dirección-dato al controlador a través de la rutina SALIDA.

d) Incrementa en uno el valor de la dirección (siguiente servo) y verifica que no exceda de 5 (mayor identificador). Así mismo, avanza una posición el apuntador.

e) Cuando ha completado el envío de los 6 datos del bloque, da tiempo a que los motores finalicen su giro mediante un llamado a la rutina RETARDO.

f) Llama a la rutina LEE TECLADO y verifica si la tecla ESCAPE (valor FEH) ha sido presionada, en cuyo caso se abandona la ejecución, dejando previamente al apuntador a bloque al inicio del último bloque al que se dio salida.

g) Si la tecla ESCAPE no fue presionada, entonces checa si se ha alcanzado el fin de tabla. En caso afirmativo el siguiente bloque a sacar debe de ser el primero dentro de la tabla, para lo cual vuelve al inciso a). De lo contrario, utiliza el siguiente bloque en secuencia, yendo al inciso b).


EJECUTA SECUENCIA
apuntador a bloque ← inicio de tabla
dato puerto A ← valor primer servo (0)
dato puerto B ← posición correspondiente
SALIDA
dato puerto A ← dato puerto A + 1
SI
dato puerto A < 6
carga contador para retardo
RETARDO
LEE TECLADO
tecla = FE
SI
NO
apuntador a bloque = apuntador a fin de tabla
actualiza apuntador a bloque
regresa

```
EJECUTA_SEC:  MOVE.L   #$2000,A2        ; dir. inicio tabla pos.
       ET14:  MOVE.W   #0,D0            ; valor primer servo
       ET12:  MOVE.B   (A2)+,D0         ; posición correspondiente
              JSR      SALIDA           ; rutina de salida
              ADD.W    #$0100,D0        ; sig. servo
              CMP.W    #$0600,D0        ; existe ese servo?
              BLT      ET12
              MOVE.L   #65535,D1        ; dato de retardo
              JSR      RETARDO          ; rutina de retardo
              JSR      LEE_TECLADO      ; rutina de lectura
              CMP.B    #$FE,D0          ; tecla = ESCAPE ?
              BEQ      ET13
              CMP.L    A2,D3            ; fin de tabla?
              BNE      ET14
              BRA      EJECUTA_SEC
       ET13:  SUB.L    #6,A2            ; A2 debe apuntar al ...
              RTS                       ; ... inicio del bloque
```

Rutina COPIA VALORES

Efectúa una copia fiel del bloque dentro de la tabla de posiciones al que se encuentra señalando el apuntador a bloque. Los nuevos valores son almacenados como un bloque en posición inmediata a su original, y el apuntador es actualizado conforme a este nuevo. Si al haber elaborado la copia, resulta que se incrementó el tamaño de la tabla, entonces el apuntador a fin de tabla es desplazado 6 unidades (tamaño del bloque).

Rutina SIGUIENTE PASO

Permite posicionar al brazo conforme a los valores indicados por el siguiente bloque dentro de la tabla de posiciones. Presionando repetidamente la tecla STEP en el controlador, produce que se obtengan los mismos resultados que con la rutina EJECUTA SECUENCIA, excepto en el retardo.

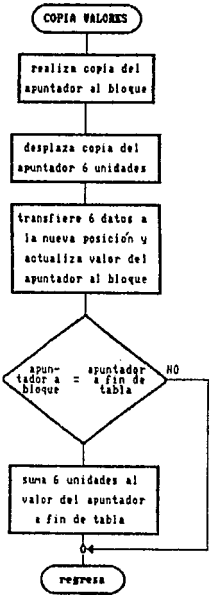
a) Posiciona el apuntador al inicio del siguiente bloque. Si resulta ser el fin de tabla, entonces el bloque a emplear viene a ser el primero de la tabla.

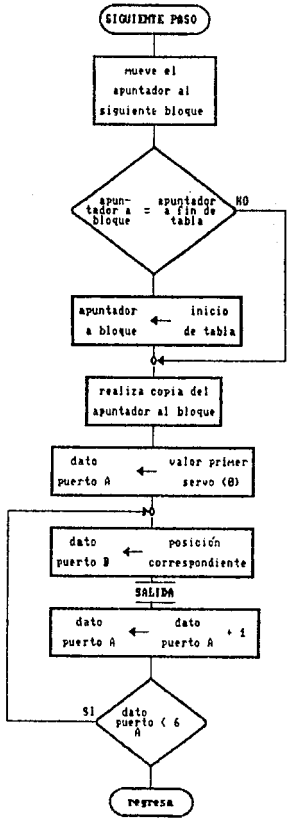
b) Realiza una copia del apuntador a bloque, con el fin de estar desplazando éste, y no mover el original.

c) Asigna un valor de 0 como primer identificador de servo.

d) Carga el dato correspondiente al servo y envía al controlador mediante la rutina SALIDA.

e) Avanza una posición la copia del apuntador a bloque, incrementa en uno el valor del identificador (siguiente servo) y verifica si ha finalizado con el bloque. En caso de no haberlo completado, continúa con la misma operación para el siguiente servo, yendo al inciso d).


COPIA VALORES
realiza copia del apuntador al bloque
desplaza copia del apuntador 6 unidades
transfiere 6 datos a la nueva posición y actualiza valor del apuntador al bloque
apuntador a bloque = apuntador a fin de tabla
NO
suma 6 unidades al valor del apuntador a fin de tabla
regresa


SIGUIENTE PASO
mueve el apuntador al siguiente bloque
apuntador a bloque = apuntador a fin de tabla
NO
apuntador a bloque ← inicio de tabla
realiza copia del apuntador al bloque
dato puerto A ← valor primer servo (0)
dato puerto B ← posición correspondiente
SALIDA
dato puerto A ← dato puerto A + 1
SI
dato puerto A < 6
regresa

```
COPIA_VAL:  MOVE.W   A2,A3              ; mismo bloque
            MOVE.L   (A2)+,4(A3)        ; copia 4 bytes
            MOVE.W   (A2)+,10(A3)       ; copia últimos 2 bytes
            CMP.L    A2,D3              ; fin de tabla?
            BNE      ET15
            ADD.L    #6,D3              ; aumentó tamaño tabla
      ET15: RTS
```

```
SIG_PASO:   ADD.L    #6,A2              ; siguiente bloque
            CMP.L    A2,D3              ; fin de tabla?
            BNE      ET16
            MOVE.L   #$2000,A2          ; inicio de tabla
      ET16: MOVE.L   A2,A3              ; mismo bloque
            MOVE.W   #0,D0              ; valor primer servo
      ET17: MOVE.B   (A3)+,D0           ; posición correspondiente
            JSR      SALIDA             ; rutina de salida
            ADD.W    #$0100,D0          ; sig. servo
            CMP.W    #$0600,D0          ; existe ese servo?
            BLT      ET17
            RTS
```

5.2 Software de control MC68000

A continuación se presenta un listado del programa principal y rutinas, obtenido directamente del kit educacional MEX68KECB. Solamente se ha editado para agregar comentarios y reducir el espacio entre columnas.

La dirección de inicio del programa principal es 1000.

```
000900  207C00010001    MOVE.L  #65537,A0       ; dir. base PIT
000906  01480000        MOVEP.L $0000(A0),D0    ; valor original de registros
00090A  028000FF0000    AND.L   #16711680,D0
000910  00B04000FFFF    OR.L    #1073807359,D0  ; puertos A y B como salida
000916  01C80000        MOVEP.L D0,$0000(A0)    ; saca palabra de control
00091A  303C000C        MOVE.W  #0.D0           ; regs. control de A y B
00091E  01880000        MOVEP.W D0,$000C(A0)
000922  303C0FFF        MOVE.W  #4095,D0        ; dato de inicialización
000926  4EB80950        JSR.S   $00000950       ; rutina de salida
00092A  247C00002000    MOVE.L  #8192,A2        ; dir. inicio tabla posición
000930  248CFFFFFFFF    MOVE.L  #-1,(A2)        ; posiciones iniciales
000936  357CFF010004    MOVE.W  #-255,4(A2)     ; Idem
00093C  263C00002006    MOVE.L  #8198,D3        ; apuntador fin de tabla
000942  1A3C0000        MOVE.B  #0.D5           ; valor inicial de tecla
000946  4E75            RTS
000948  4E71            NOP
00094A  4E71            NOP


000950  3200            MOVE.W  D0.D1           ; copia dato p/manejo interno
000952  0C410500        CMP.W   #1280,D1        ; dirección de pinza?
000956  6D0E            BLT     $966
000958  0C410501        CMP.W   #1281,D1        ; dirección de pinza?
00095C  6E08            BGT     $966
00095E  02410OFF        AND.W   #255,D1         ; sustituye direc. 05 ...
000962  00410900        OR.W    #2304,D1        ; ... por 09
000966  02400FFF        AND.W   #4095,D1        ; se añaden señales ...
00096A  0040C000        OR.W    #-16384,D1      ; ... de control
00096E  01880010        MOVEP.W D1,$0010(A0)    ; salida a puertos A y B
000972  02400FFF        AND.W   #4095,D1        ; SEL y R/W bajas
000976  01880010        MOVEP.W D1,$0010(A0)
00097A  0040C000        OR.W    #-16384,D1      ; se restablecen señales
00097E  01880010        MOVEP.W D1,$0010(A0)
000982  4E75            RTS
000984  4E71            NOP
000986  4E71            NOP
```

```
001000  4EB80900        JSR.S   $00000900       ; rutina de inicializacion
001004  223C00001000    MOVE.L  #4096,D1        ; dato para retardo
00100A  4EB81340        JSR.S   $00001340       ; rutina de retardo
00100E  1805            MOVE.B  D5,D4           ; valor tecla anterior
001010  4EB81100        JSR.S   $00001100       ; rutina lectura teclado
001014  1A00            MOVE.B  D0,D5           ; valor tecla actual
001016  0C0000F0        CMP.B   #240,D0         ; tecla no presionada?
00101A  6DF2            BLT.S   $00100E
00101C  0C0000FB        CMP.B   #251,D0         ; tecla es de control?
001020  6E08            BGT.S   $00102A
001022  4E71            NOP
001024  4EB81200        JSR.S   $00001200       ; actualiza posición
001028  60DA            BRA.S   $001004
00102A  BA04            CMP.B   D4,D5           ; tecla anterior = actual ?
00102C  67D6            BEQ.S   $001004
00102E  0C0000FF        CMP.B   #255,D0         ; tecla = RUN ?
001032  660B            BNE.S   $00103C
001034  4E71            NOP
001036  4EB81300        JSR.S   $00001300       ; rutina de ejecución
00103A  60D2            BRA.S   $00100E
00103C  0C0000FD        CMP.B   #253,D0         ; tecla = LEARN ?
001040  660B            BNE.S   $00104A
001042  4E71            NOP
001044  4EB81350        JSR.S   $00001350       ; rutina de copiado
00104B  60C4            BRA.S   $00100E
00104A  0C0000FC        CMP.B   #252,D0         ; tecla = STEP ?
00104E  66BE            BNE.S   $00100E
001050  4EB81400        JSR.S   $00001400       ; rutina siguiente posición
001054  60B8            BRA.S   $00100E
001056  4E71            NOP
001058  4E71            NOP


001100  01480000        MOVEP.L $0000(A0),D0    ; regs. control PIT
001104  0280FFFFFF00    AND.L   #-256,D0        ; puerto B como entrada
00110A  01C80000        MOVEP.L D0,$0000(A0)
00110E  303C00AA        MOVE.W  #-14166,D0      ; B -> dir. de teclado
001112  01880010        MOVEP.W D0,$0010(A0)    ; salida a puertos A y B
001116  02408FFF        AND.W   #-28673,D0      ; SEL baja
00111A  01880010        MOVEP.W D0,$0010(A0)
00111E  223C00000001    MOVE.L  #1,D1           ; dato para retardo
001124  4EB81340        JSR.S   $00001340       ; rutina de retardo
001128  01480010        MOVEP.L $0010(A0),D0    ; dato de entrada
00112C  0040C000        OR.W    #-16384,D0      ; restablece SEL
001130  01880010        MOVEP.W D0,$0010(A0)
001134  4E71            NOP
001136  03480000        MOVEP.L $0000(A0),D1    ; restablece modo de ...
00113A  00810000000F    OR.L    #255,D1         ; ... salida en ...
001140  03C80000        MOVEP.L D1,$0000(A0)    ; ... puerto B
001144  4E71            NOP
001146  4E75            RTS
001148  4E71            NOP
00114A  4E71            NOP
```

```
001200  223C00000000    MOVE.L  #0,D1          ; limpia registro
001206  1200            MOVE.B  D0,D1          ; valor tecla
001208  0201000F        AND.B   #15,D1         ; no interesa digito izq.
00120C  C2FC0002        MULU.W  #2,D1          ; son 2 bytes por c/servo
001210  267C00003000    MOVE.L  #12288,A3      ; dir. tabla desplazamientos
001216  D6C1            ADD.W   D1,A3
001218  303C0000        MOVE.W  #0,D0          ; limpia
00121C  101B            MOVE.B  (A3)+,D0       ; servo correspondiente
00121E  0C000005        CMP.B   #5,D0          ; es pinza?
001222  6604            BNE.S   $001228
001224  1213            MOVE.B  (A3),D1        ; nueva posición
001226  6026            BRA.S   $00124E
001228  12320000        MOVE.B  0(A2,D0.W),D1  ; posición actual
00122C  0C0100FF        CMP.B   #255,D1        ; se halla al maximo?
001230  660E            BNE.S   $001240
001232  4E71            NOP
001234  0C130001        CMP.B   #1,(A3)        ; valor a sumar es 1 ?
001238  6720            BEQ.S   $00125A
00123A  6010            BRA.S   $00124C
00123C  4E71            NOP
00123E  4E71            NOP
001240  0C010000        CMP.B   #0,D1          ; se halla al minimo?
001244  6606            BNE.S   $00124C
001246  0C130001        CMP.B   #1,(A3)        ; valor a sumar es -1 ?
00124A  660E            BNE.S   $00125A
00124C  D213            ADD.B   (A3),D1        ; pos. actual + desplazam.
00124E  15810000        MOVE.B  D1,0(A2,D0.W)  ; almacena nueva posición
001252  E148            LSL.W   #8,D0          ; recorre bits de dirección
001254  8001            OR.B    D1,D0          ; agrega posición
001256  4EB80950        JSR.S   $00000950      ; rutina de salida
00125A  4E75            RTS
00125C  4E71            NOP
00125E  4E71            NOP


001300  247C00002000    MOVE.L  #8192,A2       ; dir. inicio tabla posición
001306  303C0000        MOVE.W  #0,D0          ; valor primer servo
00130A  101A            MOVE.B  (A2)+,D0       ; posición correspondiente
00130C  4EB80950        JSR.S   $00000950      ; rutina de salida
001310  06400100        ADD.W   #256,D0        ; sig. servo
001314  0C400600        CMP.W   #1536,D0       ; existe ese servo?
001318  6DF0            BLT.S   $00130A
00131A  223C0000FFFF    MOVE.L  #65535,D1      ; dato de retardo
001320  4EB81340        JSR.S   $00001340      ; rutina de retardo
001324  4EB81100        JSR.S   $00001100      ; rutina de lectura
001328  0C0000FE        CMP.B   #254,D0        ; tecla = ESCAPE ?
00132C  6706            BEQ.S   $001334
00132E  B68A            CMP.L   A2,D3          ; fin de tabla?
001330  6ED4            BNE.S   $001306
001332  60CC            BRA.S   $001300
001334  95FC00000006    SUB.L   #6,A2          ; A2 debe apuntar al ...
00133A  4E75            RTS                    ; ... inicio del bloque
00133C  4E71            NOP
00133E  4E71            NOP
```

```
001340  243C0000FFFF    MOVE.L  #65535,D2        ; valor cualquiera
001346  85C2            DIVS.W  D2,D2           ; cada instruc. DIVS ...
001348  85C2            DIVS.W  D2,D2           ; ... proporciona 150 ciclos
00134A  57C9FFF4        DBEQ.L  D1,$001340
00134E  4E75            RTS


001350  3c4A            MOVE.W  A2,A3           ; mismo bloque
001352  275A0006        MOVE.L  (A2)+,6(A3)     ; copia 4 bytes
001356  375A000A        MOVE.W  (A2)+,10(A3)    ; copia últimos 2 bytes
00135A  B68A            CMP.L   A2,D3           ; fin de tabla?
00135C  6606            BNE.S   $001364
00135E  06B300000006    ADD.L   #6,D3           ; aumentá tamaño tabla
001364  4E75            RTS
001366  4E71            NOP
001368  4E71            NOP


001400  D5FC00000006    ADD.L   #6,A2           ; siguiente bloque
001406  B68A            CMP.L   A2,D3           ; fin de tabla?
001408  6606            BNE.S   $001410
00140A  247C00002000    MOVE.L  #8192,A2        ; inicio de tabla
001410  264A            MOVE.L  A2,A3           ; mismo bloque
001412  303C0000        MOVE.W  #0,D0           ; valor primer servo
001416  101B            MOVE.B  (A3)+,D0        ; posición correspondiente
00141B  4EB80950        JSR.S   $00000950       ; rutina de salida
00141C  06400100        ADD.W   #256,D0         ; sig. servo
001420  0C400600        CMP.W   #1536,D0        ; existe ese servo?
001424  6DF0            BLT.S   $001416
001426  4E75            RTS
001428  4E71            NOP
00142A  4E71            NOP


003000  00FF            00FF                    ; primer byte -> servo
003002  0001            0001                    ; segundo byte -> valor sum.
003004  0301            0301
003006  03FF            03FF
003008  0101            0101
00300A  01FF            01FF
00300C  0401            0401
00300E  04FF            04FF
003010  0201            0201
003012  02FF            02FF
003014  0501            0501
003016  0500            0500
```

5.3 Software de control Z80

Al igual que en el punto anterior, se muestra un listado del programa principal y rutinas, obtenido del macroensamblador Z80 (versión 3.08) de Cromemco. Este ha sido editado para reducir el espacio entre columnas, agregar comentarios, y eliminar encabezados de página y tablas de referencia.

```
;
(0082)      PCA   EQU 82H              ; dir. control puerto A
(0080)      PDA   EQU 80H              ; dir. datos puerto A
(0083)      PCB   EQU 83H              ; dir. control puerto B
(004F)      MI    EQU 4FH              ; dato prog. puerto entrada
(000F)      MO    EQU 0FH              ; dato prog. puerto salida
(0081)      PDB   EQU 81H              ; dir. datos puerto B

(1000)      ORG 1000H

            ; PROGRAMA PRINCIPAL
1000  CD4D10        INICIO:  CALL INICIA       ; rutina de inicialización
1003  010108           ETB:  LD BC,0801H       ; dato para retardo
1006  CD9F11                 CALL RETARDO      ; rutina de retardo
1009  DD7E01           ETA:  LD A,(IX+1)       ; valor tecla actual
100C  DD7702                 LD (IX+2),A       ; pasa a ser tecla anterior
100F  CD9010                 CALL LECTURA      ; rutina lectura de teclado
1012  FEF0                   CP 0F0H           ; tecla no presionada?
1014  FA0910                 JP M,ETA
1017  FEFC                   CP 0FCH           ; tecla es de control?
1019  F22210                 JP P,ETE
101C  CDBA10                 CALL MUEVE        ; actualiza posición
101F  C30310                 JP ETB
1022  DD4602           ETE:  LD B,(IX+2)       ; valor tecla anterior
1025  B8                     CP B              ; tecla anterior = actual ?
1026  CA0310                 JP Z,ETB
1029  FEFF                   CP 0FFH           ; tecla = RUN ?
102B  C23410                 JP NZ,ETC
102E  CD1911                 CALL EJECUTA      ; rutina de ejecución
1031  C30910                 JP ETA
1034  FEFD             ETC:  CP 0FDH           ; tecla = LEARN ?
1036  C23F10                 JP NZ,ETD
1039  CD5211                 CALL COPIA        ; rutina de copiado
103C  C30910                 JP ETA
103F  FEFC             ETD:  CP 0FCH           ; tecla = STEP ?
1041  C20910                 JP NZ,ETA
1044  CD7011                 CALL SIG_PASO     ; rutina sig. posición
1047  C30910                 JP ETA
```

```
104A  00                  NOP
104B  00                  NOP
104C  00                  NOP

                ; RUTINA DE INICIALIZACION
104D  3E0F        INICIA: LD A,M0           ; dato programación PIO
104F  D382                OUT (PCA),A     ; puertos A y B quedan ...
1051  D383                OUT (PCB),A     ; ... como salida
1053  06FF                LD B,0FFH       ; dato para inicialización
1055  DD210020            LD IX,2000H     ; dir. inicio memoria aux.
1059  CD7410              CALL SALIDA     ; rutina de salida
105C  210021              LD HL,2100H     ; dir. inicio tabla posición
105F  0605                LD B,05H        ; dato que controla ciclo
1061  36FF          ETO:  LD (HL),0FFH    ; posición inicial servos
1063  23                  INC HL          ; apunta a sig. localidad
1064  10FB                DJNZ,ETO        ; B <- B-1; brinca si B > 0
1066  3601                LD (HL),01H     ; posición inicial pinza
1068  210021              LD HL,2100H     ; dir. inicio tabla posición
106B  110621              LD DE,2106H     ; apunta a fin de tabla
106E  DD3680100           LD (IX+1),00H   ; valor inicial de tecla
1072  00                  NOP
1073  C9                  RET

                ; RUTINA DE SALIDA
1074  FE05        SALIDA: CP 05H          ; dato para pinza?
1076  C27B10              JP NZ,ETF
1079  3E09                LD A,09H
107B  E60F          ETF:  AND 0FH         ; se añaden senales ...
107D  F6C0                OR 0C0H         ; ... de control
107F  D380                OUT (PDA),A     ; salida al puerto A
1081  4F                  LD C,A          ; se salva temporalmente
1082  78                  LD A,B          ; valor del dato
1083  D381                OUT (PDB),A     ; salida al puerto B
1085  79                  LD A,C          ; se recupera dirección
1086  E60F                AND 0FH         ; SEL y R/W bajas
1088  D380                OUT (PDA),A
108A  F6C0                OR 0C0H         ; se restablecen señales
108C  D380                OUT (PDA),A
108E  00                  NOP
108F  C9                  RET

                ; RUTINA DE LECTURA
1090  3E4F        LECTURA: LD A,M1          ; se programa el puerto ...
1092  D383                OUT (PCB),A     ; ... B como entrada
1094  3EC8                LD A,0C8H       ; B -> dir. teclado
1096  D380                OUT (PDA),A
1098  E68F                AND 08FH        ; SEL baja
109A  D380                OUT (PDA),A
109C  DD7700              LD (IX+0),A     ; salva temporalmente
109F  010101              LD BC,0101H     ; dato para retardo
10A2  CD9F11              CALL RETARDO    ; rutina de retardo
```

```
10A5  DBB1                    IN A,(PDB)          ; dato de entrada
10A7  DD7701                  LD (IX+1),A         ; salva dato
10AA  DD7E00                  LD A,(IX+0)         ; recupera control
10AD  F6C0                    OR 0C0H             ; restablece SEL
10AF  D3B0                    OUT (PDA),A
10B1  3E0F                    LD A,M0             ; restablece modo de ...
10B3  D3B3                    OUT (PCB),A         ; ... salida en puerto B
10B5  DD7E01                  LD A,(IX+1)         ; recupera dato de entrada
10B8  00                      NOP
10B9  C9                      RET

                ; RUTINA DE MOVIMIENTO
10BA  E5              NUEVE:  PUSH HL             ; salva temporalmente
10BB  E60F                    AND 0FH             ; no interena digito izq.
10BD  CB27                    SLA A               ; multiplica por dos
10BF  FD2100                  LD IY,1200H         ; dir. tabla desplazamientos
10C3  FE00                    CP 0                ; si tecla fue F0 no ...
10C5  CACD10                  JP Z,ET21           ; ... avanza sobre la tabla
10C8  47                      LD B,A              ; No. bytes a avanzar
10C9  FD23            ET3:    INC IY              ; incrementa uno en uno
10CB  10FC                    DJNZ,ET3
10CD  FD7E00          ET21:   LD A,(IY)           ; servo correspondiente
10D0  FD23                    INC IY
10D2  DD7700                  LD (IX+0),A         ; salva
10D5  FE00                    CP 0                ; si servo es 0 no avanza ...
10D7  CAEB10                  JP Z,ET5            ; ... sobre tabla de posición
10DA  DD4600                  LD B,(IX+0)         ; No. bytes a avanzar
10DD  23              ET4:    INC HL              ; apunta a posición sig. servo
10DE  10FD                    DJNZ,ET4
10E0  FE05                    CP 05H              ; es pinza?
10E2  C2EB10                  JP NZ,ET5
10E5  FD4600                  LD B,(IY)           ; nueva posición de la pinza
10E8  C30F11                  JP ET6
10EB  4E              ET5:    LD C,(HL)           ; posición actual del servo
10EC  79                      LD A,C
10ED  FEFF                    CP 0FFH             ; se halla al máximo?
10EF  C2FD10                  JP NZ,ET7
10F2  FD7E00                  LD A,(IY)           ; valor a sumar ...
10F5  FE01                    CP 01H              ; ... es 1 ?
10F7  CA1611                  JP Z,ET8
10FA  C30A11                  JP ET9
10FD  FE00            ET7:    CP 00H              ; se halla al mínimo?
10FF  C20A11                  JP NZ,ET9
1102  FD7E00                  LD A,(IY)           ; valor a sumar ...
1105  FE01                    CP 01H              ; ... es -1 ?
1107  C21611                  JP NZ,ET8
110A  FD7E00          ET9:    LD A,(IY)           ; valor a sumar
110D  81                      ADD A,C             ; pos. actual + desplazam.
110E  47                      LD B,A              ; nueva posición del servo
110F  70              ET6:    LD (HL),B           ; actualiza en tabla pos.
1110  DD7E00                  LD A,(IX+0)         ; recupera dir.
1113  CD7410                  CALL SALIDA         ; rutina de salida
1116  E1              ET8:    POP HL              ; recupera su valor inicial
```

```
1117  00                    NOP
1118  C9                    RET


                ; RUTINA DE EJECUCION
1119  210021      EJECUTA:  LD HL,2100H      ; inicio tabla posiciones
111C  DD360000      ET12:   LD (IX+0),00H    ; valor primer servo
1120  46            ET10:   LD B,(HL)        ; posición correspondiente
1121  23                    INC HL
1122  DD7E00                LD A,(IX+0)      ; valor servo en turno
1125  CD7410                CALL SALIDA      ; rutina de salida
1128  DD7E00                LD A,(IX+0)      ; recupera valor servo
112B  C601                  ADD A,01H        ; sig. servo
112D  DD7700                LD (IX+0),A      ; salva
1130  FE06                  CP 06H           ; existe ese servo?
1132  FA2011                JP M,ET10
1135  0110E0                LD BC,0E010H     ; dato para retardo
1138  CD9F11                CALL RETARDO     ; rutina de retardo
113B  CD9010                CALL LECTURA     ; rutina lectura teclado
113E  FEFE                  CP 0FEH          ; tecla = ESCAPE ?
1140  CA4811                JP Z,ET11
1143  7D                    LD A,L           ; fin de ...
1144  BB                    CP E             ; ... tabla ?
1145  C21C11                JP NZ,ET12
1148  C31911                JP RUTINA        ; repite desde un inicio
114B  0606          ET11:   LD B,06H         ; controla ciclo
114D  2B            ET13:   DEC HL           ; HL debe apuntar al ...
114E  10FD                  DJNZ,ET13        ; ... inicio del bloque
1150  00                    NOP
1151  C9                    RET

                ; RUTINA DE COPIADO
1152  D5            COPIA:  PUSH DE          ; salva
1153  E5                    PUSH HL          ; HL y DE apuntan al ...
1154  D1                    POP DE           ; ... mismo bloque
1155  0606                  LD B,06H         ; controla ciclo para que ...
1157  13            ET20:   INC DE           ; ... DE apunte al sig. bloque
1158  10FD                  DJNZ,ET20
115A  0606                  LD B,06H         ; 6 datos a copiar
115C  7E            ET14:   LD A,(HL)        ; dato a copiar
115D  12                    LD (DE),A        ; dato copiado
115E  23                    INC HL           ; apunta a nuevo origen
115F  13                    INC DE           ; apunta a nuevo destino
1160  10FA                  DJNZ,ET14
1162  D1                    POP DE           ; apunta nuevamente fin tabla
1163  7D                    LD A,L           ; ha aumentado número de ...
1164  BB                    CP E             ; ... bloques en la tabla?
1165  C26D11                JP NZ,ET15
1168  0606                  LD B,06H         ; se incrementa DE en 6 ...
116A  13            ET16:   INC DE           ; ... para que apunte ...
116B  10FD                  DJNZ,ET16        ; ... al nuevo fin de tabla
116D  00            ET15:   NOP
116E  00                    NOP
116F  C9                    RET
```

```
                        ; RUTINA DE EJECUCION POR PASOS
1170  0606      SIG_PASO:   LD B,06H          ; HL apuntará al inicio ...
1172  23            ET19:   INC HL            ; ... del sig. bloque
1173  10FD                  DJNZ,ET19?
1175  7D                    LD A,L            ; nos encontramos al ...
1176  BB                    CP E              ; ... fin de la tabla?
1177  C27D11                JP NZ,ET17
117A  210021                LD HL,2100H       ; nuevo bloque es el inicial
117D  E5              ET17: PUSH HL           ; IY y HL apuntan al ...
117E  FDE1                  POP IY            ; ... mismo bloque
1180  DD360000              LD (IX+0),00H     ; valor primer servo
1184  FD7E00          ET18:  LD A,(IY)         ; posición correspondiente
1187  FD23                  INC IY
1189  47                    LD B,A
118A  DD7E00                LD A,(IX+0)       ; valor servo en turno
118D  CD7410                CALL SALIDA       ; rutina de salida
1190  DD7E00                LD A,(IX+0)       ; recupera valor servo
1193  C601                  ADD A,01H         ; sig. servo
1195  DD7700                LD (IX+0),A       ; salva
1198  FE06                  CP 06H            ; ya terminó el bloque?
119A  FA8411                JP M,ET18
119D  00                    NOP
119E  C9                    RET


                        ; RUTINA DE RETARDO
119F  E5              RETARDO:  PUSH HL       ; salva temporalmente
11A0  CD4F06             ET1:   CALL D20MS    ; retardo de 20 ms
11A3  0D                        DEC C
11A4  C2A011                    JP NZ,ET1
11A7  CD4F06             ET2:   CALL D20MS
11AA  10FB                      DJNZ,ET2
11AC  E1                        POP HL        ; recupera su valor
11AD  C9                        RET
```

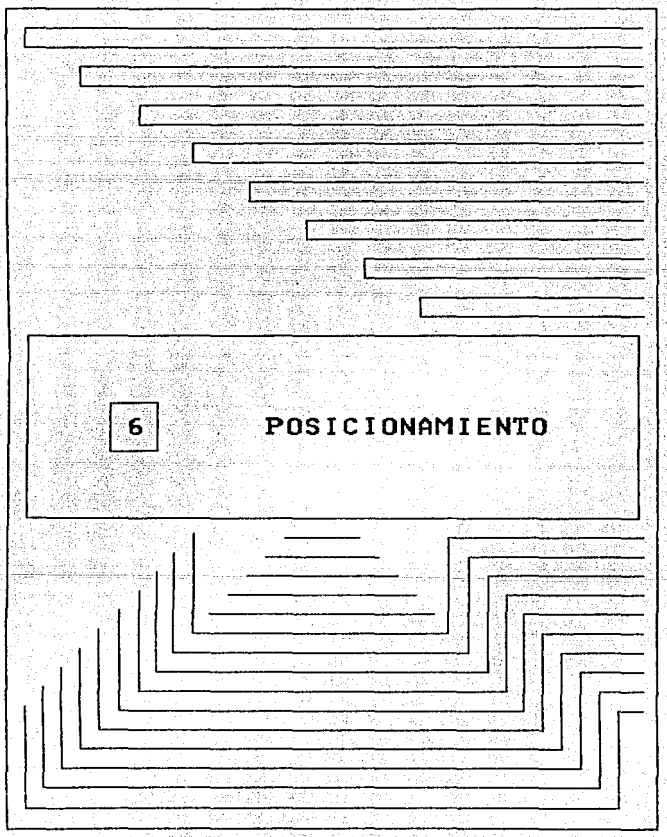
```
(1200)          ORG 1200H

                ; TABLA DE VALORES

1200  00                        DB    00H         ; dir. del servo
1201  FF                        DB    0FFH        ; valor a sumar
1202  00                        DB    00H
1203  01                        DB    01H
1204  03                        DB    03H
1205  01                        DB    01H
1206  03                        DB    03H
1207  FF                        DB    0FFH
1208  01                        DB    01H
1209  01                        DB    01H
120A  01                        DB    01H
120B  FF                        DB    0FFH
120C  04                        DB    04H
120D  01                        DB    01H
120E  04                        DB    04H
120F  FF                        DB    0FFH
1210  02                        DB    02H
1211  01                        DB    01H
1212  02                        DB    02H
1213  FF                        DB    0FFH
1214  05                        DB    05H
1215  01                        DB    01H
1216  05                        DB    05H
1217  00                        DB    00H
```

5.4 Observaciones

Las rutinas de ejecución y de ejecución por pasos, tal y como se encuentran, proporcionan al controlador el dato correspondiente a la posición final que debe de alcanzar un servomotor en un momento dado. Esto ocasiona que el giro se produzca con la máxima velocidad. Sin embargo, si se deseara que dicho giro fuera más lento, la salida habria de darse con incrementos respecto al paso anterior hasta alcanzar la posición final deseada. Entre menores sean estos incrementos menor será la velocidad de giro del servo en cuestión, es decir la menor velocidad se obtiene al proporcionar incrementos de uno.


6
POSICIONAMIENTO

En la industria, una gran cantidad de brazos mecánicos son dirigidos indicando las coordenadas que deben de accesar en cada paso. Con el fin de mostrar esta forma alterna de manejo del brazo empleado en este proyecto, se presenta a continuación el desarrollo del método para la obtención de los giros de los servomotores, de modo tal que se logre alcanzar el punto definido en el espacio.

El primer paso consiste en definir un sistema fijo de coordenadas. En la fig. 6.1 se muestra un sistema cartesiano obtenido a partir de cierta posición del brazo: El eje de giro del servo 1 se encuentra paralelo a los lados mayores de la base de asentamiento (base rectangular), y con la conexión proveniente del controlador al extremo opuesto en el que se halla dicho servo. Así, el eje horizontal Y coincide con el eje de giro del servo 1, y apuntando hacia ese lado. El eje horizontal X apunta hacia la derecha del eje Y, siendo visto éste de frente. El eje vertical Z es el mismo que el del servo 0, y apuntando hacia arriba.

Esta posición de los ejes cartesianos constituye el sistema fijo de referencia, y sobre él se efectuarán todos los cálculos.

Supóngase un punto $P(x,y,z)$ sobre el sistema fijo de referencia, el cual puede y debe ser accesado por la pinza del brazo. El problema puede ser dividido en 2 etapas:

a) Cálculo de los movimientos sobre el plano horizontal X-Y;

b) Cálculo de los movimientos sobre un plano vertical.

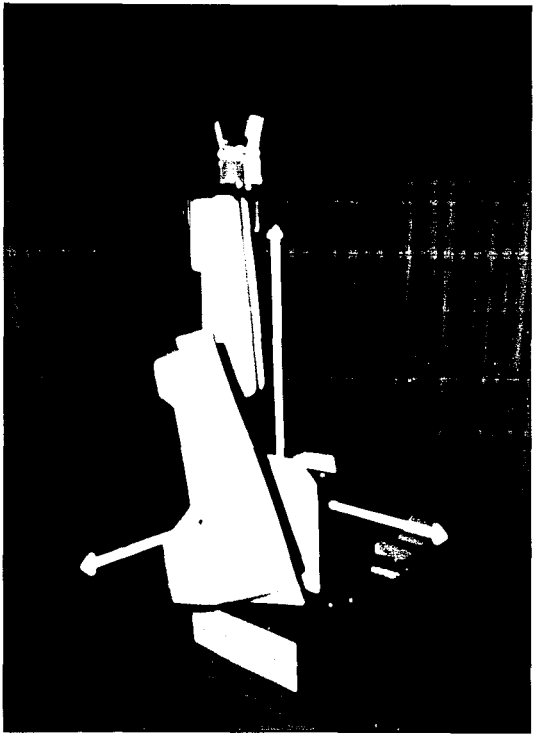


fig. 6.1 - Definición del sistema cartesiano de referencia. El eje X apunta hacia la derecha de la figura, y el eje Y apunta a la izquierda

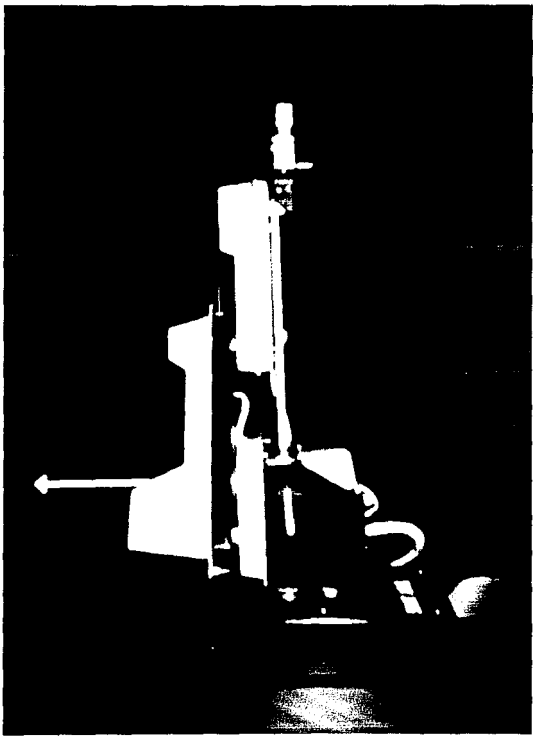


fig. 6.2 - Vista del brazo con el plano X-Z de canto. Nótese que la pinza no se encuentra contenida en dicho plano

6.1 Cálculo sobre el plano X-Y

a) Contexto

Encontrándose el brazo en la posición anteriormente descrita y apuntando en la misma dirección que el eje Z (brazo en forma vertical), y observando al plano X-Z de canto, puede verse que el eje de giro del servo 3 no pertenece a dicho plano (ver fig. 6.2). Expresado en otra forma, la pinza se encuentra a una pequeña distancia de este plano, en la región de los valores negativos de Y; denomínese l_0 a esta diferencia.

Al hacer girar el brazo sobre el servo 0, la pinza estará definiendo una circunferencia de radio l_0 sobre el plano X-Y (ver fig. 6.3a), cuya área no puede accesar. La región externa corresponde a los puntos que pueden llegar a alcanzarse.

Ahora bien, al girar los servos 1, 2 y 4 producen movimiento sobre un plano, el cual siempre es tangente a la circunferencia anterior, y por lo tanto el punto que desee alcanzarse ha de pertenecer a dicho plano. Los cálculos a efectuar en esta etapa conducirán a proporcionar un giro tal a la base que el punto P forme parte del plano tangente, y se harán sobre el plano X-Y.

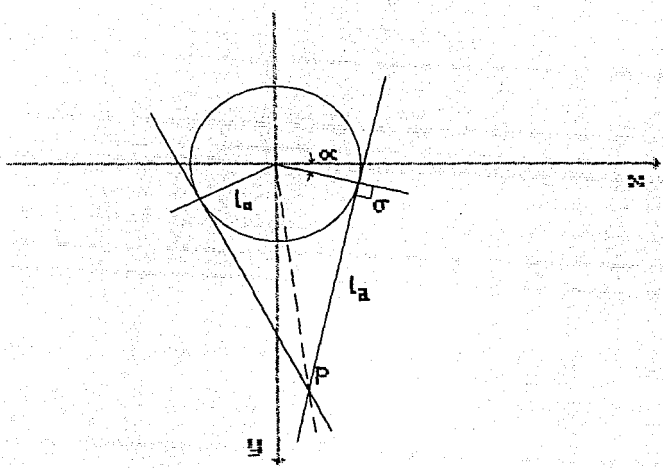


fig. 6.3a - Trazos referentes al movimiento del brazo, proyectado sobre el plano horizontal X-Y

b) Desarrollo

Nota: Dado que los planos tangentes son verticales, su proyección al plano horizontal X-Y se reduce a líneas rectas.

Del punto $P_{v}(x,y)$ han de obtenerse rectas tangentes a la circunferencia de radio l_o. La distancia del punto P_{v} al punto de tangencia se denota por l_a; además en el punto de tangencia se tiene una recta perpendicular a la tangente y que toca al origen (obviamente su longitud es l_o).

Por trigonometría se tiene que:

$$(x^2 + y^2)^{1/2} = (l_a^2 + l_o^2)^{1/2} \quad ,$$

de donde $$l_a^2 = x^2 + y^2 - l_o^2$$

Si se considera a α como el ángulo que forma la perpendicular con respecto al eje positivo X, y a σ como el ángulo de la tangente con respecto a su perpendicular, las coordenadas del punto P_{v} vienen a ser:

$$x = l_o \cos\alpha + l_a \cos(\alpha + \sigma)$$

$$y = l_o \operatorname{sen}\alpha + l_a \operatorname{sen}(\alpha + \sigma) \quad .$$

El término $l_o \cos\alpha$ proporciona la proyección de l_o sobre el eje X; a su vez $l_a \cos(\alpha + \sigma)$ representa la proyección de l_a sobre el mismo eje, y dependiendo del valor del argumento $(\alpha + \sigma)$ ésta será sumada o restada al término anterior. Para el caso de la coordenada en Y, se tienen las proyecciones de las mismas rectas, pero ahora sobre este eje.

Reagrupando y elevando al cuadrado las anteriores ecuaciones:

$$(x - l_o \cos\alpha)^2 = (l_a \cos(\alpha + \sigma))^2$$

$$(y - l_o \operatorname{sen}\alpha)^2 = (l_a \operatorname{sen}(\alpha + \sigma))^2 \quad .$$

Sumando y reduciendo por trigonometría:

$$(x - l_o\cos\alpha)^2 + (y - l_o\operatorname{sen}\alpha)^2 = l_a^2 \qquad \text{ec. 1}$$

Nótese que esta ecuación corresponde a una circunferencia con centro fuera del origen y con radio l_a. Sin embargo el centro cumple con la condición de hallarse sobre una circunferencia con centro en el origen y radio l_o. Por lo tanto deben de obtenerse 2 valores de α que satisfagan a la ec. 1; éstos pueden ser encontrados a través de aproximaciones sucesivas, y deberá de verificarse que

$$\alpha_{min} \leq \alpha \leq \alpha_{max} \quad .$$

α_{min} y α_{max} corresponden a los valores mínimo y máximo respectivamente, del ángulo que puede formarse con el giro del servo 0 en relación a la parte positiva del eje X, y en sentido del giro de las manecillas del reloj; ésto debido a que el movimiento en los servomotores es limitado.

En el capítulo 2 se determinó que a cada unidad del dato en el controlador se le asocia un giro de

120°/255: así

$$\text{dato}\ \frac{120^\circ}{255} = \text{Giro del servo desde su posición min.}$$

de donde

$$\text{dato}_\alpha = \frac{255}{120}\ (\alpha - \alpha_{min}) \quad .$$

Este valor es el que deberá de ser enviado al controlador con dirección 0 para posicionar a la base adecuadamente.

6.2 Cálculo sobre un plano vertical

a) Contexto

De la fig. 6.3a puede verse que si se hace coincidir uno de los puntos de tangencia con el eje Y, el plano vertical del movimiento del brazo quedará paralelo al plano X-Z. Así, la coordenada en Y se mantiene constante y sólo se trabaja con valores en X y Z, facilitando los cálculos.

Para no alterar las coordenadas originales, se efectúa una rotación sobre el eje Z, hasta lograr la

coincidencia deseada. Considérese que se va a emplear el plano $y = -l_a$ (ver fig. 6.3b); fácilmente puede observarse que la coordenada en X del punto $P_{u,v}$ trasladado queda indicada por la longitud l_a. Dada la simetría de las tangentes respecto del punto $P_{u,v}$, al efectuar la coincidencia del otro punto de tangencia con el eje Y, la coordenada en X del punto $P_{u,v}$ trasladado queda indicada por $-l_a$.

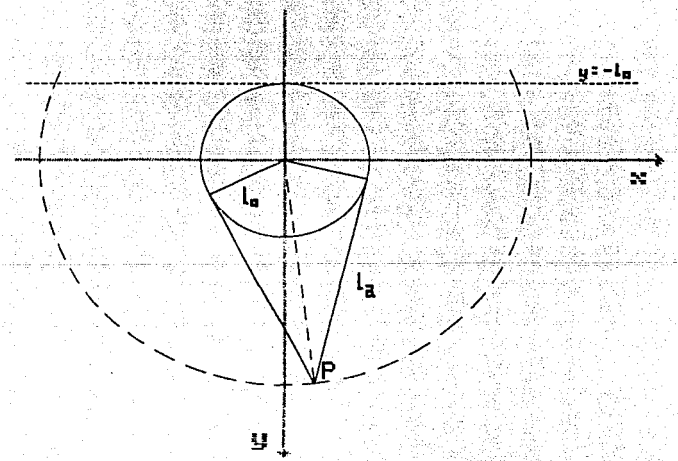


fig. 6.3b - Traslado del punto $P_{u,v}$ al plano $y=-l_a$, mediante la coincidencia de los puntos de tangencia con el eje Y

Llámese ahora $R(r_x, r_z)$ el punto P_w trasladado al plano $y = -l_0$, entonces:

$$r_x = l_1$$

$$r_z = z$$

Se harán primero los cálculos para el valor positivo de r_x (r_x).

<u>De la pinza</u>

En la fig. 6.4 se está mostrando la disposición de los componentes del brazo que intervienen en el movimiento vertical. El primer elemento (denominado l_1) se encuentra comprendido entre los ejes de giro de los servos 1 y 2, y el segundo elemento (l_2) entre los ejes de los servos 2 y 4, cuyas longitudes son 15 y 15.5 cm respectivamente.

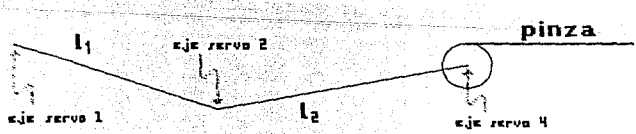


fig. 6.4 - Componentes del brazo que generan el movimiento en un plano vertical

Nótese en especial el detalle correspondiente a la pinza: ésta se encuentra a una cierta distancia del punto en que gira (eje del servo 4), por lo que al efectuar este movimiento, su extremo inicial describe una circunferencia con centro en el eje de giro del servo 4. Además, la pinza permanece siempre tangente a dicha circunferencia. Si se considera un triángulo rectángulo formado por la pinza, el radio de la circunferencia que genera (tomado en forma perpendicular a la pinza) y una recta l_3 que una al centro con el extremo final de la pinza (la cual viene a ser la hipotenusa), se podrán reducir los cálculos posteriores al emplear solamente la recta l_3; de este modo se hace referencia a la fig. 6.5, en donde:

$$l_3 = [(9.5)^2 + (0.9)^2]^{1/2} = 9.542536$$

$$\theta = \text{ang tan } \frac{0.9}{9.5} = 5.411869152^{\circ}$$

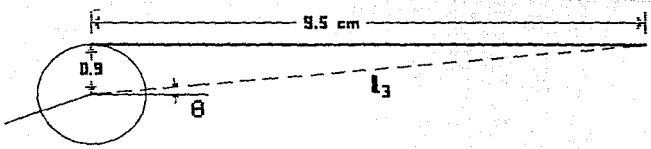


fig. 6.5 - Detalle del giro de la pinza respecto del servo 4, y simplificación de la misma

Para alcanzar el punto R, la pinza puede adoptar un gran número de posiciones, cuyos puntos iniciales forman en conjunto una circunferencia en torno a R, como puede consultarse en la fig. 6.6.

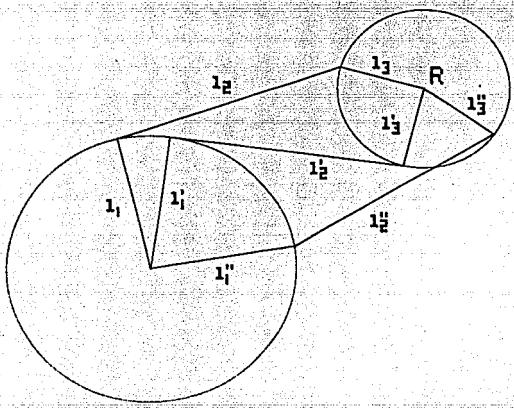


fig. 6.6 - Tres de las múltiples posibles alternativas para accesar un punto sobre el plano: l_1-l_2-l_3, l_1'-l_2'-l_3', l_1''-l_2''-l_3''

Cada uno de estos puntos sobre la circunferencia es accesado a su vez por dos posibles combinaciones de los elementos precedentes (l_1 y l_2). Ésto produce que el sistema sea indeterminado en sus posiciones. Para

resolver el problema que se plantea, basta con determinar el ángulo que ha de presentar la pinza con respecto a la horizontal, al cual se puede denominar δ.

En la fig. 6.7 se muestran las dos formas de llegar al punto R dado un valor seleccionado para δ. A partir de las coordenadas de R, del valor de δ y de la longitud de la pinza, puede obtenerse el punto que se ha indicado como Q y que pertenece a la circunferencia anteriormente descrita. Como ya se mencionó, para efectos de cálculos se sustituirá la pinza por la recta l_3 que se obtuvo en párrafos anteriores, cuyo ángulo con respecto a la horizontal es igual al de la pinza (δ) más la elevación inicial θ que poseé (repásese fig. 6.5).

De los componentes precedentes a la pinza

Nótese sobre la misma fig. 6.7 que ahora el objetivo para los dos primeros componentes (l_1 y l_2) es el punto Q. Esto viene a ser muy similar al planteamiento que se hizo para calcular los ángulos de giro de la base del brazo.

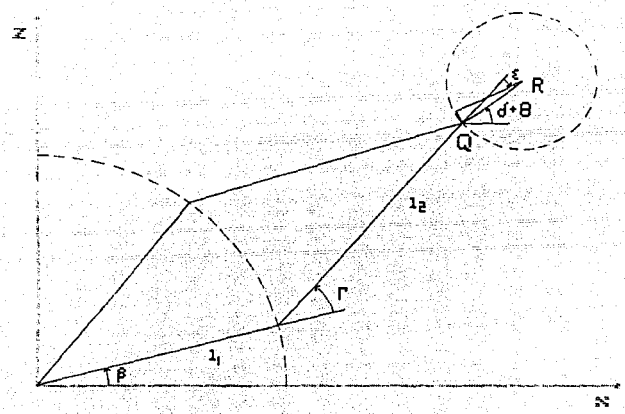


fig. 6.7 - Unicas dos formas de acceso al punto R(r,ϕ,z), dado el valor determinado de elevación de la pinza (δ)

El ángulo β indica la elevación del primer componente (l_1) respecto a la horizontal (plano X-Y). El ángulo Γ proporciona el giro del segundo componente (l_2) respecto al primero.

b) Desarrollo

Ya se indicó que el punto R tiene por coordenadas:

$$r_x = l_4 \quad , \quad r_y = z \quad .$$

Ahora deben de hallarse las correspondientes al punto $Q(q_x, q_y)$. Se ve que:

$$q_x = r_x + l_3 \cos(\delta + \theta + 180°)$$

$$q_y = r_y + l_3 \operatorname{sen}(\delta + \theta + 180°) \quad .$$

Se ha aumentado el ángulo en 180°, ya que considerando un subsistema de ejes con origen en R, $\delta+\theta$ se halla en el primer cuadrante mientras que l_3 se encuentra en el tercero; sin el ajuste, ésto produciría un resultado erróneo, por lo que deben de hacerse coincidir en el tercer cuadrante.

De la fig. 6.7:

$$q_x = l_1 \cos\beta + l_2 \cos(\beta + \Gamma)$$

$$q_y = l_1 \operatorname{sen}\beta + l_2 \operatorname{sen}(\beta + \Gamma) \qquad \text{ec. 2}$$

Agrupando y elevando al cuadrado:

$$(q_x - l_1 \cos\beta)^2 = (l_2 \cos(\beta + \Gamma))^2$$

$$(q_y - l_1 \operatorname{sen}\beta)^2 = (l_2 \operatorname{sen}(\beta + \Gamma))^2$$

Sumando y reduciendo:

$$(q_x - l_1\cos\beta)^2 + (q_y - l_1\operatorname{sen}\beta)^2 = l_2^2 \qquad \text{ec. 3}$$

Como en el caso de la ec. 1 para la base, se obtienen dos valores de β que satisfacen a la ecuación, los cuales pueden calcularse por aproximaciones sucesivas.

Si de las ec. 2 elevamos al cuadrado y sumamos, sin agrupar previamente:

$$q_x^2 + q_z^2 = l_1^2\cos^2\beta + 2l_1l_2\cos\beta\cos(\beta+\Gamma) + l_2^2\cos^2(\beta+\Gamma) + l_1^2\mathrm{sen}^2\beta + 2l_1l_2\mathrm{sen}\beta\,\mathrm{sen}(\beta+\Gamma) + l_2^2\mathrm{sen}^2(\beta+\Gamma)$$

$$= l_1^2 + l_2^2 + 2l_1l_2[\cos\beta(\cos\beta\cos\Gamma - \mathrm{sen}\beta\,\mathrm{sen}\Gamma) + \mathrm{sen}\beta(\mathrm{sen}\beta\cos\Gamma + \cos\beta\,\mathrm{sen}\Gamma)]$$

$$q_x^2 + q_z^2 = l_1^2 + l_2^2 + 2l_1l_2\cos\Gamma \quad ,$$

de donde

$$\Gamma = \pm \arccos\frac{q_x^2 + q_z^2 - l_1^2 - l_2^2}{2\,l_1\,l_2} \qquad \text{ec. 4}$$

Al menor valor de β se le asocia el valor positivo de Γ, mientras que al mayor β le corresponde Γ negativo. Nótese que en el caso de que la distancia del origen al punto Q sea igual a l_1+l_2, sólo existirá un valor de β que cumpla con la ec. 3 y Γ será nulo.

El ángulo ϵ indicado en la fig. 6.7 corresponde a la posición de la pinza con respecto al elemento l_2. Se

distingue fácilmente la relación:

$$\delta = \beta + \Gamma + \epsilon \quad ,$$

de donde

$$\epsilon = \delta - \beta - \Gamma \qquad \text{ec. 5}$$

Así, una vez calculados los valores de β, Γ y ϵ, deben de verificarse las condiciones:

$$\beta_{min} \leq \beta \leq \beta_{max}$$

$$\Gamma_{min} \leq \Gamma \leq \Gamma_{max}$$

$$\epsilon_{min} \leq \epsilon \leq \epsilon_{max} \quad ,$$

tal y como se hizo para α. Si se cumplen estas relaciones entonces se procede a calcular los correspondientes valores que han de enviarse al controlador:

$$dato_1 = \frac{255}{120} (\beta - \beta_{min})$$

$$dato_2 = 255 - \frac{255}{120} (\Gamma - \Gamma_{min}) \quad *$$

$$dato_4 = \frac{255}{120} (\epsilon - \epsilon_{min})$$

* El sentido del giro en que crecen los valores del dato es opuesto a aquél con que lo hace Γ, por lo que se efectúa la sustracción del valor mayor (255).

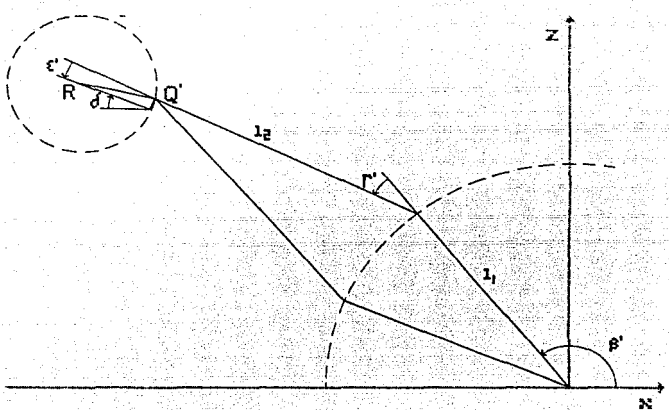


fig. 6.8 - Acceso al punto R(r_-,z), con una elevación determinada de la pinza (δ)

En base a la fig. 6.8, se efectuarán ahora los cálculos para cuando r_- adopta un valor negativo (r_-).

Todas las consideraciones hechas anteriormente son válidas para este punto, y varían solamente en que:

- El ángulo δ pasa a ser ahora δ', definido como:

$$\delta' = 180° - \delta$$

- Al obtener valores de q_x y q_z, δ debe de ser sustituido por δ', los cuales una vez reduciendo aparecen como:

$$q_x' = r_w - l_3 \cos(\theta - \delta)$$

$$q_z' = r_z + l_3 \operatorname{sen}(\theta - \delta) \quad .$$

- Los ángulos β, Γ y ϵ se traducen ahora a β', Γ' y ϵ' respectivamente.

- Con las coordenadas del punto Q' y aplicando la sustitución anterior, se calculan valores para estos ángulos mediante las ec. 3, 4 y 5, ésto es:

$$(q_x' - l_1 \cos\beta')^2 + (q_z' - l_1 \operatorname{sen}\beta')^2 = l_2^2$$

$$\Gamma' = \pm \operatorname{ang\,cos} \frac{q_x'^2 + q_z'^2 - l_1^2 - l_2^2}{2\, l_1\, l_2}$$

$$\epsilon' = 180 - (\delta + \beta' + \Gamma') \quad .$$

- Deben verificarse las relaciones:

$$\beta_{min} \leq \beta' \leq \beta_{max}$$

$$\Gamma_{min} \leq \Gamma' \leq \Gamma_{max}$$

$$\epsilon_{min} \leq \epsilon' \leq \epsilon_{max} \quad .$$

- Se obtienen los valores que han de ser enviados al controlador:

$$dato_1 = \frac{255}{120}(\beta' - \beta_{min})$$

$$dato_2 = 255 - \frac{255}{120}(\Gamma' - \Gamma_{min})$$

$$dato_4 = \frac{255}{120}(\epsilon' - \epsilon_{min})$$

6.3 Asociación de ángulos

Ahora debe de definirse la forma en que los ángulos primos (β',Γ',ϵ') y los no primos (β,Γ,ϵ) se relacionan con los ángulos de la base (α_{menor} y α_{mayor}).

Tomando en cuenta que el ángulo α se halla comprendido en el rango [0,360°), se procederá a analizar de qué forma pueden accesarse los diversos puntos. Para ello, se considerará a α en su mínima posible posición (α=0), en la que los valores críticos son x=0.6 y y=-0.9 (punto en que se encuentra la pinza al adoptar el brazo su posición vertical extrema, dada la excentricidad de aquélla). Así, puede generarse la siguiente tabla 6.1:

x	y	ángulos asociados a α_{menor}	ángulos asociados a α_{mayor}
> 0.6	< -0.9	primos	no primos
> 0.6	> -0.9	primos	no primos
< 0.6	< -0.9	no primos	primos
< 0.6	> -0.9	no primos	primos
0.6	≤ -0.9	primos	no primos
0.6	> -0.9	no primos	primos

tabla 6.1 - Análisis de asociación de ángulos por regiones

De aquí puede obtenerse la relación

if x > 0.6 or (x = 0.6 and y ≤ -0.9)

a α_{menor} se asocian ángulos primos

a α_{mayor} se asocian ángulos no primos

else

a α_{menor} se asocian ángulos no primos

a α_{mayor} se asocian ángulos primos

En la fig. 6.9 se indica la forma de asociar los ángulos calculados, de acuerdo con el resultado de la condición anterior.

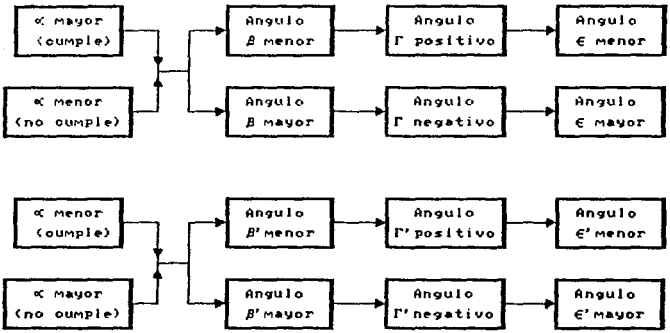
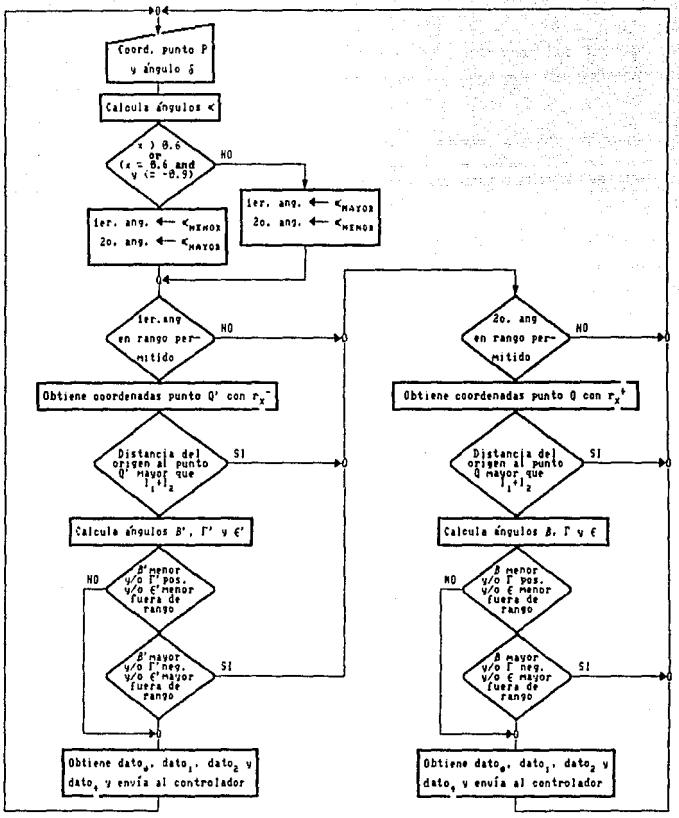


fig. 6.9 - Correspondencia entre los ángulos calculados

Una vez desarrollado el método que permite definir las posiciones del brazo, y mostrada la asociación de ángulos, se presenta el diagrama de flujo correspondiente al algoritmo:


Coord. punto P y ángulo δ
Calcula ángulos α
x) 0.6 or (x = 0.6 and y (= -0.9)
NO
1er. ang. ← α MENOR
2o. ang. ← α MAYOR
1er. ang. ← α MAYOR
2o. ang. ← α MENOR
1er. ang en rango permitido
NO
2o. ang en rango permitido
NO
Obtiene coordenadas punto Q' con r_x^-
Obtiene coordenadas punto Q con r_x^+
Distancia del origen al punto Q' mayor que l_1+l_2
SI
Distancia del origen al punto Q mayor que l_1+l_2
SI
Calcula ángulos β', Γ' y ε'
Calcula ángulos β, Γ y ε
β' menor y/o Γ' pos. y/o ε' menor fuera de rango
NO
β menor y/o Γ pos. y/o ε menor fuera de rango
NO
β' mayor y/o Γ' neg. y/o ε' mayor fuera de rango
SI
β mayor y/o Γ neg. y/o ε mayor fuera de rango
SI
Obtiene dato_0, dato_1, dato_2 y dato_4 y envía al controlador
Obtiene dato_0, dato_1, dato_2 y dato_4 y envía al controlador

6.4 Ejemplo

Se muestra a continuación un ejemplo del cálculo de posiciones, no sin antes repasar las longitudes de cada elemento del brazo y de indicar los valores máximos y mínimos obtenidos para los ángulos involucrados:

$l_o = 0.6$, $l_1 = 15$, $l_2 = 15.5$ (en cm)

$\alpha_{min} = 154.7826°$, $\alpha_{max} = 281.6124°$, amplitud = 126.8297°

$\beta_{min} = 52.7°$, $\beta_{max} = 180°$, amplitud = 127.3°

$\Gamma_{min} = -19.4°$, $\Gamma_{max} = 102.7°$, amplitud = 122.1°

$\epsilon_{min} = -50.55°$, $\epsilon_{max} = 82.8°$, amplitud = 133.35°

La amplitud del movimiento para un cierto ángulo corresponde a la diferencia entre el valor máximo y el valor mínimo (p.ej. $\alpha_{max} - \alpha_{min}$). Nótese que aunque el fabricante especifica una amplitud de giro de 120° para cada servomotor, las mediciones hechas proporcionan cierta diferencia con respecto a ese dato.

Ejemplo: Accesar el punto P(-23,7,-2), ubicando la pinza a -90° respecto a la horizontal.

a) Cálculo de ángulos α:

$$l_a^2 = x^2 + y^2 - l_o^2$$
$$= (-23)^2 + (7)^2 - (0.6)^2$$
$$= 577.64$$

$(x - l_0 \cos\alpha)^2 + (y - l_0 \operatorname{sen}\alpha)^2 = l_2^2$

$(-23 - 0.6 \cos\alpha)^2 + (7 - 0.6 \operatorname{sen}\alpha)^2 = 577.64$

de donde $\alpha_{menor} = 74.502°$

$\alpha_{mayor} = 251.642°$.

Las posiciones correspondientes a estos valores se muestran en la fig. 6.11.

b) Cálculo de ángulos β', Γ' y ϵ', asociados a α_{mayor}

$r_w^- = -l_0 = -24.0341$, $r_z = -2$

$q_w' = r_w^- + l_3 \cos(\theta - \delta) = -24.9341$

$q_z' = r_z + l_3 \operatorname{sen}(\theta - \delta) = 7.5$

La distancia del origen al punto Q' es

$$d = [(-24.9341)^2 + (7.5)^2]^{1/2} = 26.03 ,$$

la cual es menor que $l_1 + l_2$ (30.5 cm), y por lo tanto es posible accesar este punto.

$(q_w' - l_1 \cos\beta')^2 + (q_z' - l_1 \operatorname{sen}\beta')^2 = l_2^2$

$(-24.9341 - 15 \cos\beta')^2 + (7.5 - 15 \operatorname{sen}\beta')^2 = 240.25$

de donde $\beta'_{menor} = 131.297°$

$\beta'_{mayor} = 195.221°$.

$$\Gamma' = \pm \text{ang cos} \frac{q_w'^2 + q_z'^2 - l_1^2 - l_2^2}{2 l_1 l_2}$$

$$\Gamma' = \pm\ 62.7779^{\circ}$$

$$\epsilon'_{menor} = 180 - (\delta + \beta'_{menor} + \Gamma'_{pos})$$
$$= 180 - (-90 + 131.297 + 62.7779)$$
$$= 75.9251^{\circ}$$

$$\epsilon'_{mayor} = 180 - (\delta + \beta'_{mayor} + \Gamma'_{neg})$$
$$= 137.5569^{\circ}$$

En la fig. 6.12 se muestran las posiciones del brazo asociadas a los valores aquí obtenidos, así como para los que surgirán en el inciso d).

c) Obtención de datos para el controlador:

Dado que los valores de β'_{menor}, Γ'_{pos} y ϵ'_{menor} se encuentran dentro de los rangos permitidos, se les empleará para encontrar los datos que han de enviarse al controlador.

$$dato_0 = \frac{255}{126.8297}(251.642 - 154.7826) = 194.7 = \boxed{C3H}$$

$$dato_1 = \frac{255}{127.3}(131.297 - 52.7) = 157.4 = \boxed{9DH}$$

$$dato_2 = 255 - \frac{255}{122.1}(62.7779 + 19.4) = 83.3 = \boxed{53H}$$

$$dato_4 = \frac{255}{133.35}(75.9251 + 50.55) = 241.8 = \boxed{F2H}$$

d) Cálculo de ángulos β, Γ y ϵ, asociados a α_{menor}:

Aún cuando el menor valor de α se encuentra fuera de rango, se procederá a calcular los ángulos asociados a éste, con el objeto de mostrar la forma en que son obtenidos.

$r_w^+ = l_4 = 24.0341$, $r_z = -2$

$q_w = r_w^+ + l_3 \cos(\delta + \theta + 180) = 23.1341$

$q_z = r_z + l_3 \operatorname{sen}(\delta + \theta + 180) = 7.5$

La distancia del origen al punto Q es

$$d = [(23.1341)^2 + (7.5)^2]^{1/2} = 24.3194 ,$$

la cual es menor que $l_1 + l_2$ (30.5 cm).

$(q_w - l_1 \cos\beta)^2 + (q_z - l_1 \operatorname{sen}\beta)^2 = l_2^2$

$(23.1341 - 15\cos\beta)^2 + (7.5 - 15\operatorname{sen}\beta)^2 = 240.25$

de donde $\beta_{menor} = -19.876°$

$\beta_{mayor} = 55.801°$.

$$\Gamma = \pm \text{ang cos}\frac{q_w^2 + q_z^2 - l_1^2 - l_2^2}{2\, l_1\, l_2}$$

$\Gamma = \pm 74.2542^{\circ}$

$\theta_{menor} = \delta - \beta_{menor} - \Gamma_{pos}$

$= -144.3782^{\circ}$

$\theta_{mayor} = \delta - \beta_{mayor} - \Gamma_{neg}$

$= -71.5468^{\circ}$

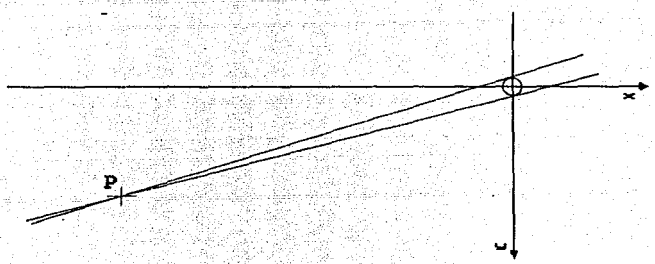


fig. 6.11 - Posiciones para accesar el punto P_{xy}, correspondientes a los valores menor y mayor de α

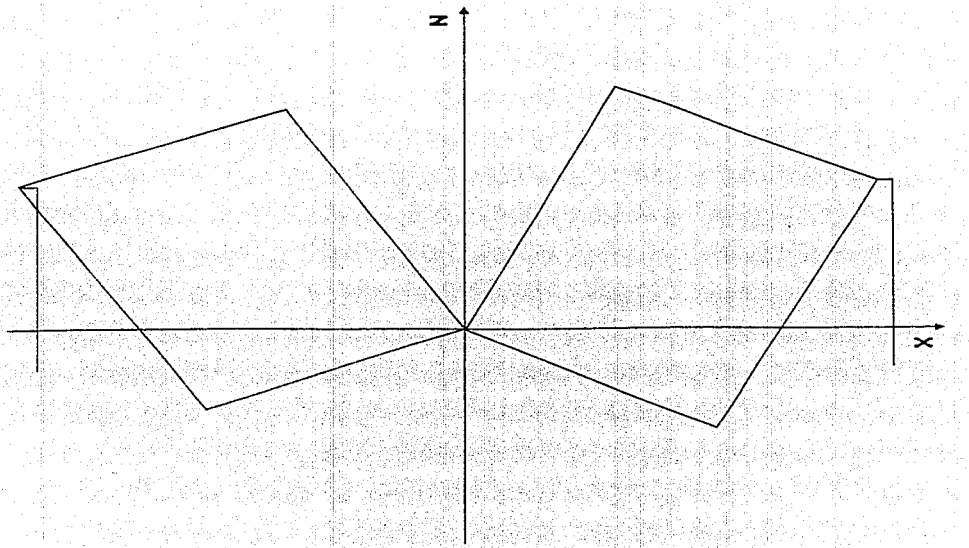
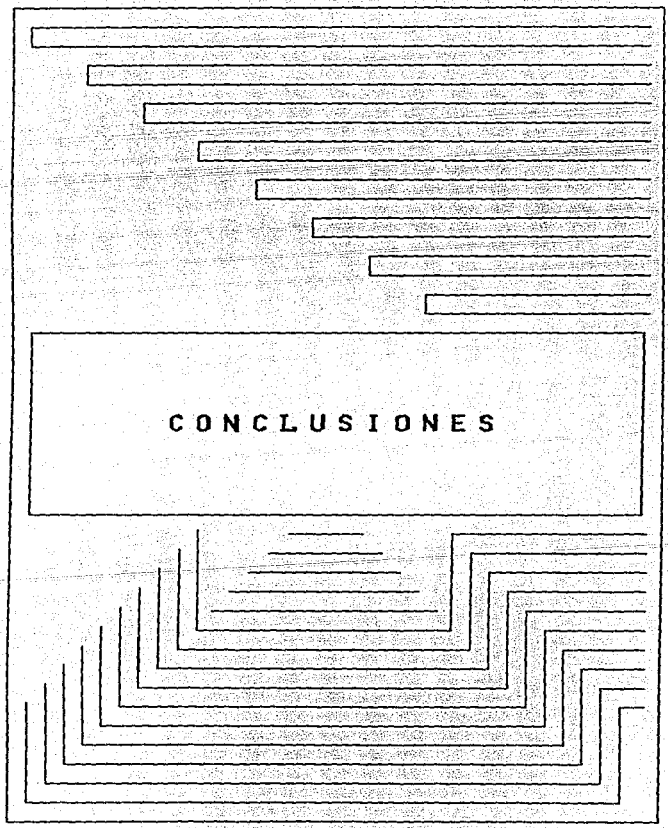


fig. 5.12 - Formas de acceso a los puntos R, dadas las condiciones del ejemplo


CONCLUSIONES

El desarrollo del presente proyecto implicó lo siguiente:

a) Estudio del manual del equipo y de los diagramas del controlador proporcionados por el distribuidor.

b) Investigación del cruce de líneas entre los conectores interno y externo del controlador.

c) Investigación de las señales involucradas en la carga de información en el controlador.

d) Estudio de las características y de la conexión de los circuitos de comunicación en paralelo MC68230 y Z80-PIO.

e) Conexión del controlador a los kits educacionales.

f) Investigación de los códigos asociados al teclado del controlador.

g) Investigación de las señales enviadas al brazo.

h) Desarrollo de los diversos diagramas de tiempo asociados a las funciones del controlador.

i) Definición de características, desarrollo y prueba del software de control.

j) Estudio profundo del funcionamiento del servo-amplificador NE544.

k) Estudio de la tarjeta de circuito impreso del brazo, con el fin de descubrir la distribución de señales en el conector de entrada.

l) Investigación acerca de las configuraciones de la pinza neumática, válvula solenoide y su circuito amplificador, así como del punto de partida de la correspondiente señal de control.

m) Desarrollo del método para el cálculo de giros de los servomotores, para lo cual se efectuó poca investigación en libros y se trabajó al 100% sobre el modelo real.

n) Mediciones de los ángulos generados por cada uno de los servomotores, y comprobación de resultados a través de una serie de ejemplos.

El constante problema que se enfrentó fue la carencia de información en el manual y diagramas del equipo proporcionados por el distribuidor, por lo que se hizo necesario incurrir en las acciones descritas en los incisos b,f,h,j,k y l. Más aún, cierta información incluída llegaba a ser errónea; tal fue el caso de la

dirección asociada a la señal de la pinza neumática en el controlador, para la cual proporcionan un valor, sin embargo el equipo posee una configuración completamente diferente a la especificada.

Al concluir este trabajo se ha logrado:

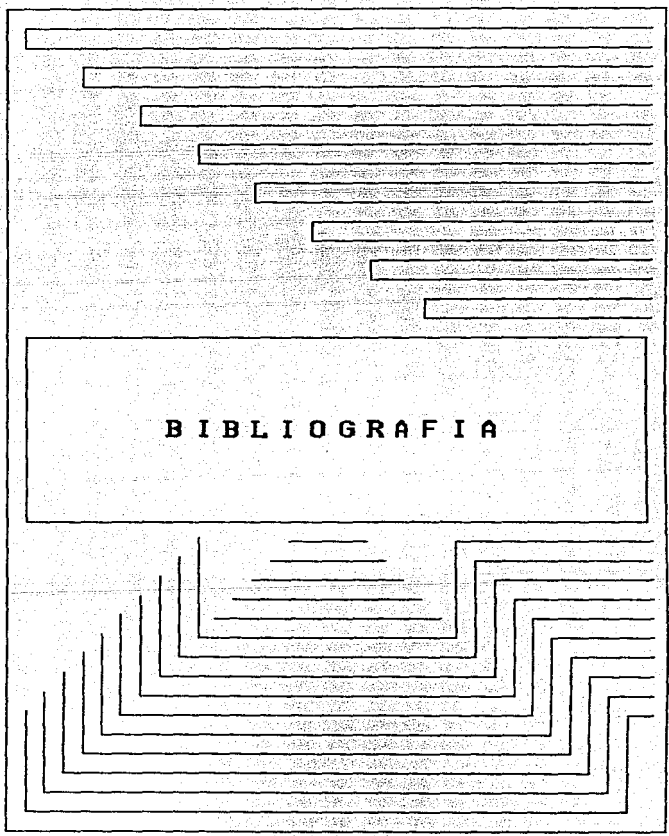
a) Presentar los componentes tanto del brazo mecánico como del controlador, proporcionando una clara explicación de su modo de operación.

b) Incluir un análisis del funcionamiento del servo-amplificador NE544, en relación al control de los servomotores. Esto permite familiarizarse con el circuito, propiciando a la vez su uso en el desarrollo de futuros proyectos.

c) Mostrar una aplicación de los kits educacionales y en especial de los circuitos de comunicación en paralelo.

d) Proporcionar un ejemplo del manejo de la información que utiliza el controlador, así como del aprovechamiento de su teclado.

e) Introducir una forma de análisis de los movimientos del brazo. En especial, en estos dos últimos puntos se encuentran las bases para el desarrollo de futuras prácticas de programación con los kits educacionales.


BIBLIOGRAFIA

1 - López Navarro, Tomás
Automatismo y Control
Ed. Gustavo Gili, S.A.
Barcelona, 1969

2 - Broida, V.
Automatización, Regulación Automática, Servomecanismos
Editorial Universitaria de Buenos Aires
Buenos Aires, 1965

3 - Fu, K.S.; González, R.C.; Lee, C.S.G.
Robótica: Control, Detección, Visión e Inteligencia
Ed. McGraw Hill
México, 1988

4 - Coiffet, P.; Chirouze, M.
Elementos de Robótica
Ed. Gustavo Gili, S.A.
Barcelona, 1986

5 - Avilés González, Rafael; Angulo Usategui, José María
Curso de Robótica
Ed. Paraninfo
Madrid, 1989

6 - Manual 'Smart-Arms (Industrial Robots) and
Smart-Cell (Flexible Manufacturing Units)'
Systems Control
Inglaterra

7 - Lara López, Ma. del Rocio
Manual 'Introducción al Equipo'
DRT México

8 - MC68000 Educational Computer Board User's Manual
Motorola, Inc.
Arizona, 1982

9 - Z80 Starter System Operations Manual
Micro Design Concepts
1978

10 - The Motorola MC68230
Motorola, Inc.
Austin, Texas; 1984

11 - NE544 Servo Amplifier Product Specification
Signetics Linear Products
Signetics
1987

12 - Applications Using the NE544 Servo Amplifier
Signetics Linear Products
Signetics
1987

13 - Rosales Valderrábano, Edmundo
Apuntes para el Manual de Prácticas del Laboratorio de
Microcomputadoras
México, 1991