



Universidad Nacional Autónoma de México

FACULTAD DE INGENIERIA

CENTRAL TELEFONICA ELECTRONICA CONTROLADA
POR UN MICROPROCESADOR.

TESIS PROFESIONAL
Que para obtener el Título de
INGENIERO MECANICO-ELECTRICISTA
p r e s e n t a
MANUEL FACUNDO A. HUERTA CHAVARRIA



Director de Tesis:

ING. MARIO A. IBARRA PEREYRA

MEXICO, D. F.

1 9 9 1

FALLA DE ORIGEN



Universidad Nacional
Autónoma de México



UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

DERECHOS RESERVADOS © PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

C O N T E N I D O

	Página
1. Introducción.....	2
2. Conceptos de Conmutación.....	4
2.1 Centrales Telefónicas.....	4
2.2 Jerarquía de las Centrales Telefónicas.....	7
2.3 Sistemas de Control de Conmutación y Tecnologías.....	10
3. Requerimientos de la Central Telefónica Electrónica.....	11
3.1 Definición de la Central Electrónica.....	11
3.2 Simbología.....	11
3.3 Requerimientos del Cliente.....	15
3.4 Diagrama a bloques del sistema.....	16
4. Funcionamiento de la Central Telefónica Electrónica.....	17
4.1 Funcionamiento de la Parte de Conmutación.....	17
4.2 Funcionamiento de la Parte Interface.....	25
4.3 Funcionamiento de la Parte de Control.....	30
4.4 Secuencia de Conmutación en la Central Telefónica.....	39
5. Software del Sistema.....	42
5.1 Lenguajes de Programación.....	42
5.2 Diagramas de Flujo.....	64
5.3 Programación.....	67
5.4 Pruebas.....	92
6. Conclusiones.....	93
Bibliografía.....	95

1. INTRODUCCION

Las comunicaciones han tenido una importancia vital en la humanidad ya que han sido determinantes en el desarrollo de los pueblos.

Los primeros medios de comunicación que la humanidad utilizó fueron los mensajeros, que transmitían en forma verbal o escrita el mensaje.

Años más tarde, se valieron de estafetas humanas, relevos que llevaban el mensaje a grandes distancias, después se emplearon animales rápidos como caballos y palomas mensajeras.

También se utilizaron otros medios de comunicación en forma de señales ópticas y acústicas, como son: Las hogueras, banderas, tambores y espejos.

Algunos de estos sistemas se encuentran en uso aún hoy en día, especialmente en el ejército y en la marina.

Es hasta después del año 1800 cuando el físico italiano, Alejandro Volta da a conocer la pila eléctrica por lo que es posible hacer experimentos de comunicación empleando corriente eléctrica.

Poco después del descubrimiento de la pila eléctrica se desarrollaron experimentos, hasta lograr la transmisión de la voz humana por medio del teléfono inventado por Alejandro Graham Bell en 1876.

Este teléfono, tenía algunas deficiencias que se fueron eliminando con otros experimentos, como el micrófono de carbón, inventado por Tomas A. Edison en 1877.

Para el año de 1877 habían instalado 1300 aparatos telefónicos en diferentes casas particulares de los E. U., poniendo de manifiesto la necesidad de conectar los aparatos a una central, para establecer las comunicaciones de los usuarios.

Lógicamente este tipo de central que consistía en un conmutador, en donde las funciones inteligentes las efectuaba una operadora, fué el principio de la conmutación, que se desarrollaría rápidamente desde las centrales manuales, las mecánicas, las electromecánicas, las semielectrónicas, y últimamente las electrónicas, controladas con microprocesadores.

De aquí el surgimiento de esta tesis en la cual se describe la evolución de las comunicaciones telefónicas desde las más sencillas como las controladas por una operadora, hasta las más modernas utilizadas en la planta telefónica, para lo cual se analiza el funcionamiento general de una central electrónica controlada por un microprocesador desarrollándolo de la siguiente manera :

El tema 1 es una introducción, en donde se describe la evolución de las comunicaciones.

El tema 2 se refiere a los conceptos telefónicos utilizados en Teléfonos de México, haciendo énfasis en las jerarquías de las centrales telefónicas.

El tema 3, trata sobre la simbología que se utilizará, los diagramas a bloques y las condiciones de uso para poder cumplir con los requerimientos del cliente, que en este caso es representado por el director de tesis.

El tema 4 habla sobre el funcionamiento de las partes que intervienen en nuestra central telefónica, profundizando con más detalle en los conceptos electrónicos.

El tema 5, trata sobre el software que se desarrolla para este sistema, haciendo hincapié en el lenguajes que se utilizará en la programación del microprocesador, para lo cual, se da el programa que será utilizado en este proyecto, así como las pruebas que se desarrollaron.

El tema 6 se refiere a las conclusiones; cabe aclarar que en esta tesis no se está diseñando una central telefónica diferente a las ya existentes en Teléfonos de México, sino que se realizaron adaptaciones al conmutador electrónico SX-10 para satisfacer los requerimientos del director de tesis.

Esto pone de manifiesto la gran flexibilidad de los equipos telefónicos electrónicos, en los que modificando el programa almacenado en su memoria, es posible hacer que ejecute una gran diversidad de funciones.

2. CONCEPTOS DE CONMUTACION

2.1 CENTRALES TELEFONICAS

2.2 JERARQUIA DE LAS CENTRALES TELEFONICAS

2.3 SISTEMAS DE CONTROL DE CONMUTACION Y TECNOLOGIAS

2.1 CENTRALES TELEFONICAS

Las centrales telefónicas son sistemas que representan unidades de conmutación y que contienen en su interior una gran cantidad de dispositivos de conexión, donde se aplican varias técnicas de conmutación y control y están organizadas por jerarquías que se enlazan de acuerdo a las necesidades de conexión.

Existen 2 tipos de Centrales Telefónicas:

- a) centrales telefónicas Privadas.
- b) centrales telefónicas Públicas.

Las Centrales Telefónicas Privadas son aquellas que tienen un uso particular o privado y se instalan en empresas y fábricas donde los usuarios son el propio personal de dicha empresa y que requieren comunicarse entre sí como podemos ver en la figura 2.1.1

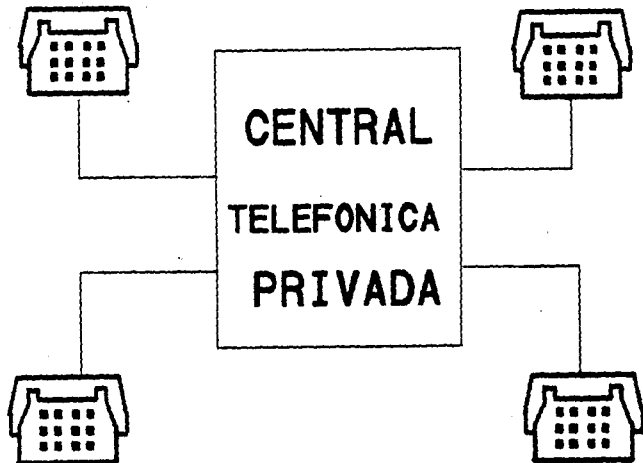


FIG 2.1.1 CENTRAL TELEFONICA PRIVADA

Este tipo de central privada, fue el principio de las centrales telefónicas, en el cual el control lo realiza una operadora quien controla a los dispositivos de conexión; esto lo podemos observar en la figura 2.1.2, cabe hacer notar que este tipo de central telefónica privada con control a través de operadora, todavía existe en servicio en algunos establecimientos.

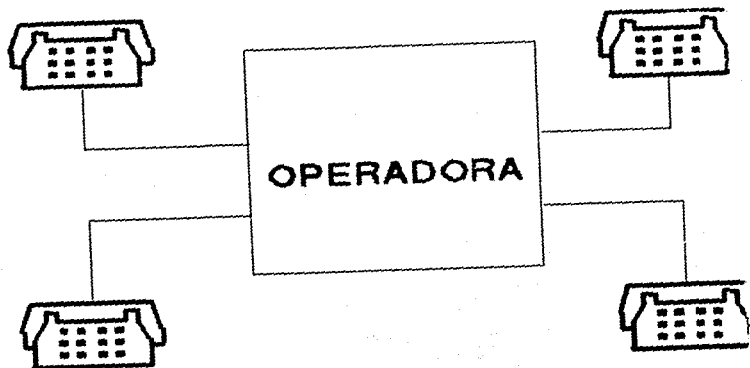


FIG. 2.1.2 CENTRAL TELEFONICA PRIVADA CONTROLADA POR OPERADORA

Para que alguno de los usuarios o abonados de esta Central Telefónica Privada, pueda comunicarse con el exterior, es necesario que dicha Central Telefónica esté enlazada con una Central Telefónica Pública a través de líneas troncales que permitan una comunicación con el sistema público como lo podemos observar en la figura 2.1.3

Lógicamente la evolución en las tecnologías se ha reflejado en los dispositivos que conforman a las centrales telefónicas privadas y públicas, desde elementos electromecánicos, semielectrónicos y electrónicos.

Actualmente, este tipo de Centrales Telefónicas se fabrican con tecnología electrónica en su parte de control, esto quiere decir que son controladas por microprocesadores que es el tema de esta tesis.

Las Centrales Telefónicas Públicas, son aquellas en donde el usuario es el público en general.

Las Centrales Telefónicas Públicas están agrupadas para atender cada una a 10,000 abonados por lo que si la población requiere mas de 10,000 líneas, es necesario que todas las centrales estén unidas a través de troncales o cables troncales que pueden ser cables físicos, cable coaxial, fibra óptica o radio.

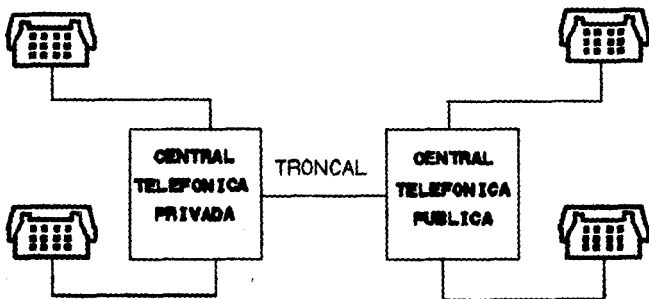


FIG. 2.1.3 CENTRAL TELEFONICA PRIVADA CONECTADA AL EXTERIOR

2.2 JERARQUIA DE LAS CENTRALES TELEFONICAS.

Debido a lo complejo de una red telefónica, como la de Teléfonos de México, es necesario agrupar a las Centrales Telefónicas en una jerarquía de acuerdo a su aplicación, función y tráfico que manejan, como se describe a continuación.

a) CENTRAL RURAL.- Tiene la jerarquía de oficina terminal manual, proporciona servicio telefónico a pequeños poblados variando su capacidad de acuerdo al interés de tráfico que exista. En jerarquía, la Central Rural es la más pequeña y el número de abonados es limitado.

b) CENTRAL LOCAL O URBANA.- Tiene la jerarquía de oficina terminal (OT) que proporciona el servicio telefónico automático en una población, por lo que también se le conoce como Central Local.

De acuerdo con la cantidad y distribución de los usuarios, su perficie y forma geográfica, costos de inversión y mantenimiento en una población, etc., se puede instalar una o más Centrales Locales.

De necesitarse sólo una central, la OT se conoce como Oficina Terminal Aislada (OTA).

Cuando la población es atendida por dos o más centrales, a éstas se les conoce como Oficina Terminal Urbana (OTU).

Para poblaciones muy grandes como la ciudad de México, Guadalajara y Monterrey entre otras, es necesario utilizar Centrales Telefónicas Intermedias llamadas TANDEM (CT) que manejan tráfico de tránsito originado o terminado en Centrales Locales y se utiliza como un elemento de optimización en el manejo del tráfico en la Red Troncal. Esto lo podemos observar en la figura 2.2.1

El centro TANDEM realiza principalmente las siguientes funciones:

Maneja el tráfico de tránsito entre centrales que no justifique vía directa.

- Maneja el tráfico de desborde entre 2 centrales que tienen vía directa de alto uso, pero que en ésta, todas las troncales están ocupadas.

Enruta el tráfico de centrales electromecánicas en las que el número de vías es limitado.

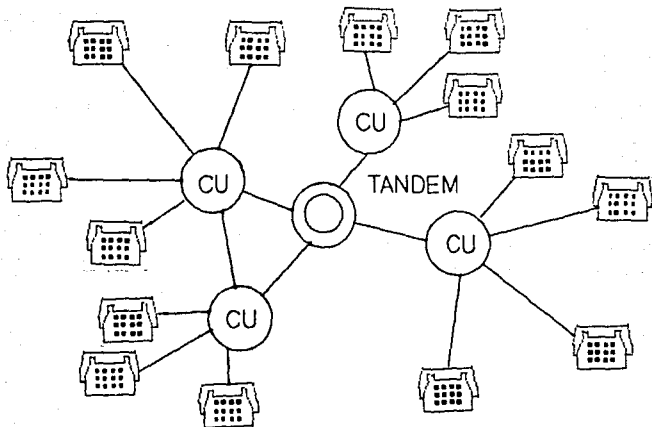


FIG. 2.2.1 RED URBANA UTILIZANDO CENTRAL TANDEM

c) Central Automática de larga distancia (CALD). En este tipo de central telefónica, no hay ningún abonado que se conecte a ella, sino que solamente tiene conectadas centrales locales adyacentes, cursa tráfico interurbano o de larga distancia originado o terminado en centrales subordinadas a ella, las cuales pueden ser centrales locales u otras Cald's.

d) CENTRAL INTERNACIONAL (CI).- También es una Central Automática de larga distancia, pero tiene en particularidad de que está comunicada a la red nacional con redes telefónicas de otros países. Esta central puede ser exclusivamente para tráfico internacional o manejar simultáneamente tráfico nacional, e internacional, como se muestra en la figura 2.2.2

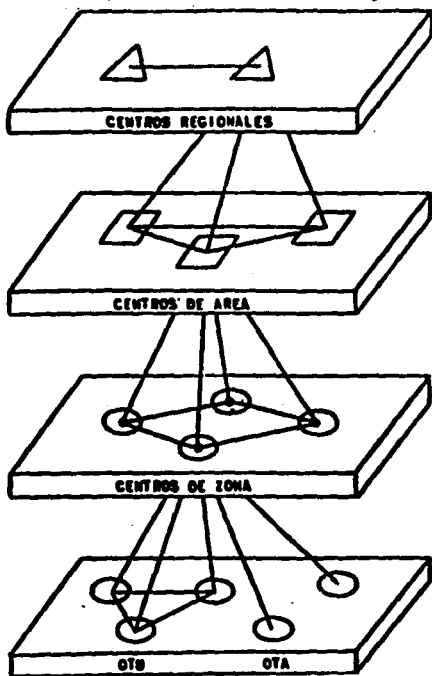


FIG.2.2.2 JERARQUIA DE LAS CENTRALES TELEFONICAS

2.3 SISTEMAS DE CONTROL DE CONMUTACION Y TECNOLOGIAS

Una Central Telefónica se compone básicamente de una red de conexión y una unidad de control.

La red de conexión efectúa la conmutación entre las líneas a las que sirve la central, llevando así las corrientes de conversación.

La función de la unidad de control es ejercer una influencia determinada en la red de conexión, de forma que, de acuerdo con el servicio solicitado, se establezcan y desaparezcan las oportunas conexiones de cada caso.

Esta parte de control también realiza ciertas tareas administrativas, como son: las estadísticas de tráfico, la información de averías y la tarificación.

De acuerdo con la filosofía utilizada en esos órganos, los sistemas telefónicos pueden clasificarse en:

a) Atendiendo a la red de conexión.

- Sistemas de Conmutación Espacial (Rotatorio y Coordenadas).
- Sistemas de Conmutación Temporal (PCM Y AXE).
- Sistemas por división de frecuencia (Multiplex).

b) Atendiendo a la Unidad de Control.

- Sistemas de Control por lógica cableado.
- Sistema de Control por programa cableado.
- Sistemas de Control por programa almacenado.

c) Atendiendo a la Transmisión de Señales de voz.

- Sistemas Analógicos.
- Sistemas Digitales.

3. REQUERIMIENTOS DE LA CENTRAL TELEFONICA

3.1 DEFINICION DE LA CENTRAL ELECTRONICA

3.2 SIMBOLOGIA

3.3 REQUERIMIENTOS DEL CLIENTE

3.4 DIAGRAMA A BLOQUES DEL SISTEMA

3.1 DEFINICION DE LA CENTRAL ELECTRONICA

De acuerdo con las definiciones dadas en el capítulo 2, nuestra Central Telefónica Electrónica será del tipo privada, con 4 abonados, y de acuerdo a la red de conexión, será del tipo de Sistema de Conmutación Espacial, y en lo que se refiere a la Unidad de Control, ésta será del Sistema de Control por programa almacenado, y en lo que se refiere a la transmisión de señal de voz, será del tipo analógico.

3.2 SIMBOLOGIA

Se utilizará la siguiente simbología para una mejor comprensión de las funciones de la Central Telefónica Electrónica.



Abonado telefónico o usuario

LI Interfase de línea

SS Etapa de abondo

ABJ Circuito de enlace AB o juntor de abonado (Circuito de cordón).

KR Receptor de señales de teclado

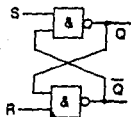
BS Transmisor de tono de ocupado

ABONADO A: Es el que origina la llamada.

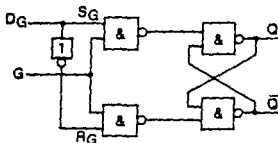
ABONADO B: Es el que recibe la llamada.

Aquí se muestran las compuertas más usuales con tres diferentes tipos de representación.

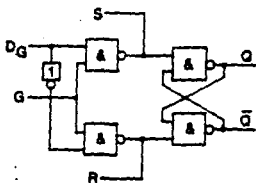
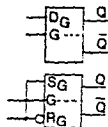
AND	a		<table border="1"> <thead> <tr> <th colspan="2">Entr.</th> <th>Sal.</th> </tr> <tr> <th>A</th> <th>B</th> <th>X</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	Entr.		Sal.	A	B	X	0	0	0	0	1	0	1	0	0	1	1	1
	Entr.		Sal.																		
	A	B	X																		
	0	0	0																		
0	1	0																			
1	0	0																			
1	1	1																			
b																					
c																					
NAND	a		<table border="1"> <thead> <tr> <th colspan="2">Entr.</th> <th>Sal.</th> </tr> <tr> <th>A</th> <th>B</th> <th>X</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	Entr.		Sal.	A	B	X	0	0	1	0	1	1	1	0	1	1	1	0
Entr.		Sal.																			
A	B	X																			
0	0	1																			
0	1	1																			
1	0	1																			
1	1	0																			
b																					
c																					
OR	a		<table border="1"> <thead> <tr> <th colspan="2">Entr.</th> <th>Sal.</th> </tr> <tr> <th>A</th> <th>B</th> <th>X</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	Entr.		Sal.	A	B	X	0	0	0	0	1	1	1	0	1	1	1	1
	Entr.		Sal.																		
	A	B	X																		
	0	0	0																		
0	1	1																			
1	0	1																			
1	1	1																			
b																					
c																					
NOR	a		<table border="1"> <thead> <tr> <th colspan="2">Entr.</th> <th>Sal.</th> </tr> <tr> <th>A</th> <th>B</th> <th>X</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	Entr.		Sal.	A	B	X	0	0	1	0	1	0	1	0	0	1	1	0
Entr.		Sal.																			
A	B	X																			
0	0	1																			
0	1	0																			
1	0	0																			
1	1	0																			
b																					
c																					



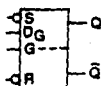
D flip-flop



Símbolo
(dos alternativas)



Símbolo



Entradas S R	Estado anterior Q Q̄	La posición es	La posición va inmediata- mente hacia:
0 0	0 0	inestable	0011
0 0	0 1	inestable	0011
0 0	1 0	inestable	0011
0 0	1 1	Estable	
0 1	0 0	inestable	0110
0 1	0 1	inestable	0110
0 1	1 0	Estable	
0 1	1 1	inestable	0110
1 0	0 0	inestable	1001
1 0	0 1	Estable	
1 0	1 0	inestable	1001
1 0	1 1	inestable	1001
1 1	0 0	inestable	1101 o 1110 (al azar)
1 1	0 1	Estable	
1 1	1 0	Estable	
1 1	1 1	inestable	1101 o 1110 (al azar)

El flip-flop está desestableado.
El flip-flop está fijado. } Condiciones
destables

CIRCUITOS DE FUNCIONES DE MEMORIA

3.3 REQUERIMIENTOS DEL CLIENTE.

En este subtema, se realiza la simulación de que un cliente personificado por el director de tesis, llega a la empresa Teléfonos de México y solicita el servicio telefónico a través de un conmutador que realice las siguientes funciones:

- a) Se requiere un aparato para conmutar 4 extensiones.
- b) Que los abonados que pertenecen a dicho conmutador, se puedan comunicar entre ellos internamente.
- c) La marcación debe ser por teclado de "alta velocidad".
- d) La Central Telefónica, aceptará la solicitud de un solo abonado al tiempo. Esto quiere decir que si otro abonado trata de iniciar una llamada al mismo tiempo, recibirá tono de ocupado.
- e) Debe realizar la función de consulta. Esta función se refiere a que ya establecida una comunicación entre dos abonados, uno de ellos puede retener la llamada primero, para después consultar a un tercer abonado y posteriormente reanudar la conversación con el abonado anterior.
- f) Por último, se requiere tener comunicación con la red de Teléfonos de México. Para realizar esta función es necesario contar con una troncal que comunique a nuestro conmutador con la Central de Teléfonos de México.

En estos requerimientos hay algunas funciones que por ser obvias, el cliente no las solicita explícitamente; estas son:

- Envío de tonos de marcar, de ocupado, de llamada.
- Envío de corriente de llamada.
- Distinción entre las frecuencias de los tonos de la red pública y los del conmutador.
- Enviar tono de ocupado al abonado que no cuelga al terminar una conversación.
- Enviar tono de ocupado si no hay línea disponible.
- El teclado de "alta velocidad" implica que los dígitos marcados se transmitan en código MFC (2 frecuencias por cada dígito) ya que el codificador de pulsos emite los dígitos a la misma velocidad que un disco normal.

3.4 DIAGRAMA A BLOQUES DEL SISTEMA.

El conmutador seleccionado para satisfacer los requerimientos del cliente es el SX-10 de Mitel que se analizará a continuación.

El primer paso para estructurar el comportamiento de la central, es obtener un diagrama a bloques de las diferentes funciones de la Central Telefónica Electrónica.

En la figura 3.4.1 podemos observar el diagrama a bloques de la Central Telefónica que está formada por la parte de conmutación y la parte de control.

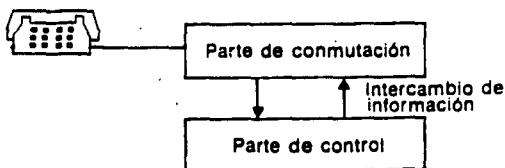


FIG 3.4.1 DIAGRAMA A BLOQUES DE LA CENTRAL TELEFONICA.

Como se puede observar, los abonados están conectados a la parte de conmutación. Esta parte contiene conductores y contactos sobre los cuales tiene lugar la conexión de habla y la transmisión de señales, conocida como red de conmutación. La parte de conmutación también contiene circuitos para funciones simples de telefonía, tales como generadores de tono de marcar y tono de llamada, circuitos para recepción de señales de descuelgue y señalización de aparatos telefónicos de teclado y para traducción de estas señales, a una forma adecuada para la parte de control.

La parte de control, contiene circuitos y programas que atienden las funciones "inteligentes" de la central, tales como la identificación e interpretación de los cambios de estado en la parte de conmutación, y la operación de circuitos en la parte de conmutación de acuerdo a los programas basados en los requerimientos del comportamiento de la central en las diferentes situaciones.

En otras palabras, se puede resumir que la parte de control determina "qué y dónde se debe hacer" basándose en los cambios de estado en la parte de conmutación, la cual luego ejecuta las decisiones.

**4. FUNCIONAMIENTO DE LA CENTRAL
TELEFONICA ELECTRONICA**

**4.1 FUNCIONAMIENTO DE LA PARTE
DE CONMUTACION**

**4.2 FUNCIONAMIENTO DE LA PARTE
INTERFACE**

**4.3 FUNCIONAMIENTO DE LA PARTE
DE CONTROL**

**4.4 SECUENCIA DE CONMUTACION
EN LA CENTRAL TELEFONICA**

4.1 FUNCIONAMIENTO DE LA PARTE DE CONMUTACION.

Los aparatos telefónicos están conectados a la red de conmutación, como vimos en la FIG. 3.4.1

Desde el punto de vista funcional, el aparato telefónico está dividido en cuatro partes principales: el circuito de habla, los contactos de horquilla, el timbre de campana y el tablero de teclado con generadores de tonos, como vemos en la figura 4.1.1

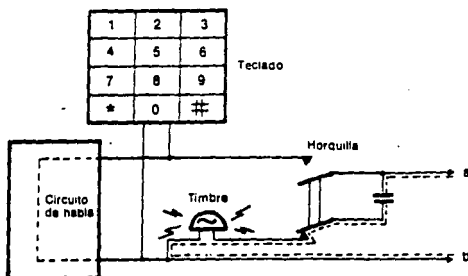


FIG. 4.1.1 APARATO TELEFONICO DE TECLADO.

El circuito de habla comprende al micrófono de carbón, el audifono o receptor y el transformador de habla, los dos primeros están incorporados en el microteléfono.

El contacto de horquilla se conmuta cuando se levanta o reposa el microteléfono, por lo que se dice que su función es la señalización de la llamada, de la desconexión y de respuesta.

El timbre o campana es del tipo AC (Corriente Alterna) la que se conecta a los hilos "a" y "b" de la línea del abonado via el condensador y el contacto de horquilla.

El tablero de teclado es utilizado para el envío de los dígitos de 0 a 9, como también las señales especiales, asterisco (*) y gato (#) que sirven para dar acceso a servicios especiales en redes más avanzadas.

Cuando se oprime una tecla se generan dos tonos de frecuencia de acuerdo con la tabla de la figura 4.1.2

Hz →	1209	1336	1477
697	1	2	3
770	4	5	6
852	7	8	9
941	*	0	#

FIG. 4.1.2 TABLA DE FRECUENCIAS DEL APARATO TELEFONICO.

Los cuatro abonados de nuestra central están conectados a la red de contactos, como ya se ha dicho anteriormente, en la que los dispositivos de contactos son relés, como se muestra en la figura 4.1.3 en donde se observa que hay un relé en cada punto de cruce de una línea horizontal y una vertical, por lo que tendremos 16 relés en total, y cada relé será controlado por un FLIP-FLOP.

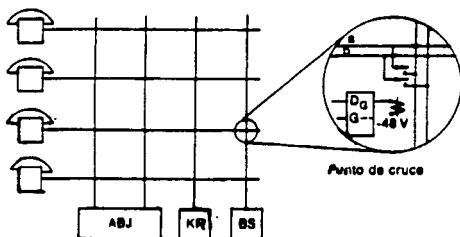


FIG. 4.1.3 RED DE CONMUTACION

Además de la red de contactos, la parte de conmutación está formada por un número de dispositivos con diferentes funciones, como son las siguientes:

- a)- ABJ (JUNTOR AB). Este dispositivo sirve para conectar a los abonados vía los relés de conmutación para una comunicación.
- b)- KR (KEY SET RECEIVER) Receptor de teclado, sirve para recibir las señales de los abonados.
- c)- BS (BUSY SENDER) Transmisor de tono de ocupado, se utiliza cuando la conexión no puede realizarse.
- d) LI (LINE INTERFACE) Interface para línea, se requiere para enviar señales de bucle de los abonados.

Estos dispositivos y los selectores de relés son controlados por el microprocesador, vía el bus de datos, como podemos ver en la figura 4.1.4

En dicha figura podemos ver como el abonado número cero está conectado al abonado número 2 vía el dispositivo ABJ, mientras que el abonado número 3, está conectado al dispositivo KR.

El dispositivo ABJ ó JUNTOR ABJ, en algunos sistemas convencionales se le conoce como "SNR" (juego de relés del circuito de cordón). Su labor es juntar las corrientes de voz, pero separadas de sus respectivas componentes de corriente directa, en esta forma, el estado del contacto de horquilla de uno de los abonados, no afectará la lectura del estado del contacto del otro abonado.

El juntor ABJ, atiende también la alimentación microfónica de ambos abonados (abonado "A" y abonado "B") y aún más, ABJ contiene circuitos para el envío de señales de llamada para el abonado "B" y de tono de control de llamada para el abonado "A", y circuitos para desconexión de todas estas señales, una vez que el abonado "B" conteste, por lo tanto, el control de la señal de timbre, no se lleva a cabo por programa de la parte de control con ayuda de exploración de los puntos de prueba y el cambio de estado en los puntos de operación, sino que esto se hace por Hardware (circuitaría).

Para comprender esto, consideremos una llamada cuando el procesador ha dado la orden de conexión de dos abonados en sus respectivas terminales de ABJ, en este momento, el microprocesador inicia el envío de la señal de timbre mediante la fijación del respectivo FLIP-FLOP en ABJ. La señal de timbre va hacia afuera vía el circuito RT (RING TRIPPING).

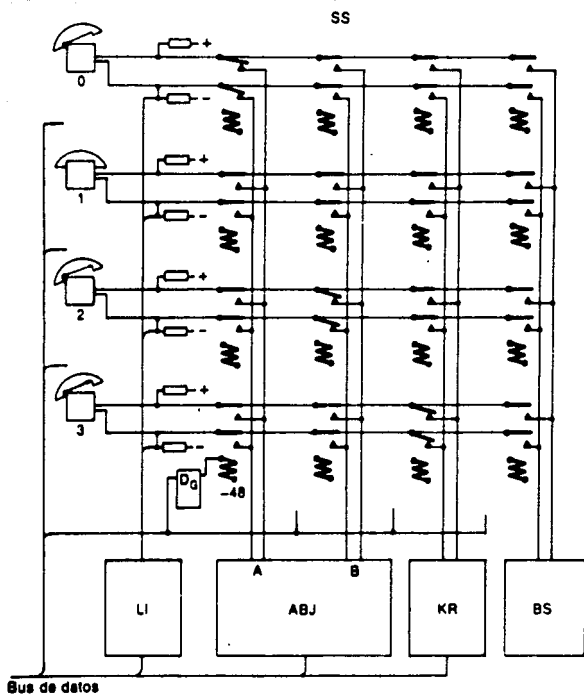


FIG. 4.1.4 EJEMPLOS DE CONEXION EN LA RED DE CONMUTACION

Cuando el abonado "B" descuelga su microteléfono (contesta abonado "B"), el circuito RT detecta la corriente continua del bucle formado con el abonado "B", y desactiva al FLIP-FLOP, por lo que el relé suelta y establece la conexión de habla entre el abonado "A" y el abonado "B" como se observa en la figura 4.1.5

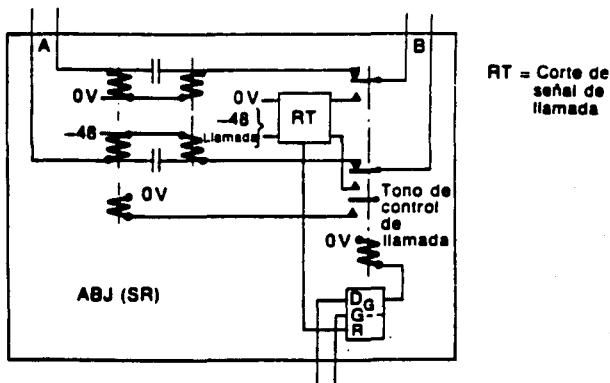


FIG. 4.1.5 DISPOSITIVO ABJ

Con respecto al dispositivo KR (KEY SET RECEIVER), que es el receptor de teclado, este contiene filtros para detectar cualquier cifra marcada por el abonado "A". Como se observa en la figura 4.1.2 cada dígito es codificado con una combinación de dos frecuencias, pudiendo realizarse hasta 12 diferentes combinaciones.

Además de estos filtros, KR contiene un generador de frecuencia de tono de marcar, llamado también "TONO DE INVITACION A MARCAR" (Su2).

Cuando el abonado "A" quiere comunicarse con otro abonado, pulsa la tecla correspondiente a dicho abonado, llamado abonado "B" y lo que sucede es que, al pulsar una tecla se envían dos frecuencias por los hilos "a" y "b" hacia los 7 filtros del KR, y solo responderán 2 de ellos de acuerdo a los valores a que corresponden dichas frecuencias y darán un "1" lógico.

Las salidas de los filtros se conectan a compuertas NAND cuyas salidas darán el número recibido en forma de un "0" lógico en uno de los 8 hilos conectados al bus de datos, de tal forma que nuestra pequeña central telefónica electrónica podrá recibir como máximo 8 números diferentes de abonado como se observa en la figura 4.1.6 que nos muestra al KR ejemplificando lo que sucede al pulsar la tecla correspondiente al abonado cero como abonado "B".

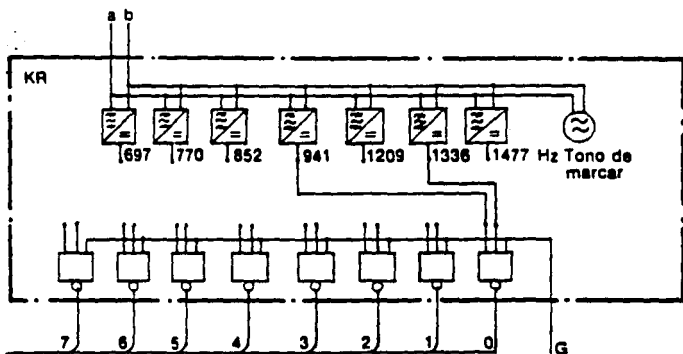


FIG. 4.1.6 DISPOSITIVO KR

De esta forma el receptor de teclado KR decodifica los dígitos con lógica cableada, esto se hace para lograr un entendimiento sencillo de los dígitos por parte del procesador. En diseños reales, se usa a menudo la técnica de datos para la decodificación, conectándose el bus del procesador directamente a la salida de los filtros, vía una compuerta por filtro.

Cuando el abonado "A", recibe el tono de invitación a marcar (Su2), y este marca el número del abonado "B" en su teclado, el tono de invitación marcar tendría que suspenderse para liberar al o los receptores de teclado, pero debido a que en nuestra central solo podemos tener una conversación a la vez no es necesario que se suspenda el tono de invitación a marcar, puesto que esto es tan rápido que el abonado "A" lo experimenta como si el tono de invitación a marcar hubiese cesado inmediatamente después que se ha marcado el número del abonado "B", pero lo que realmente sucede es que el abonado "A" será conectado al abonado "B" a través del ABJ y el KR se liberará. En centrales mas grandes, en las que el número del abonado "B" consta de varios dígitos, el tono de invitación a marcar se desconecta despues de marcar el primer dígito, por lo que se requiere de un punto de operación en cada uno de los KR dados para este propósito, por lo cual, los circuitos se complican en su funcionamiento.

El dispositivo LI (INTERFACE DE LINEA), contiene COMPUERTAS mediante las cuales el procesador se informará acerca del estado de los contactos de horquilla de los abonados.

La verificación tiene lugar con órdenes enviadas desde el microprocesador hasta la entrada de control "G" de las compuertas que son de tipo especial en las cuales los estados lógicos de las entradas conectadas a las líneas de abonado, están caracterizadas por voltajes diferentes a los normales, es decir 0 volts y +5 volts. Esto es debido a que la alimentación de los abonados requieren voltajes mayores que los utilizados en los circuitos lógicos y que normalmente son de 24 y 48 volts de corriente directa y que se utiliza para alimentar al micrófono del aparato telefónico que es de gránulos de carbón, como se muestra en la figura 4.1.7

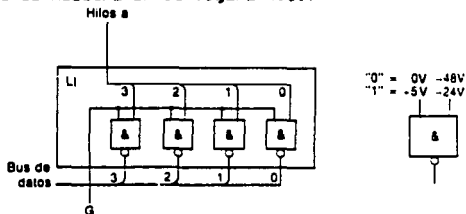


FIG. 4.1.7 DISPOSITIVO LI.

En el dispositivo LI mostrado en la figura anterior, se observa que los hilos "a" de los abonados, están conectados a las 4 compuertas del LI, y este hilo "a" está conectado a una diferencia de potencial que pasa a través del abonado descolgado como se observa en la figura 4.1.B

En esta figura, también se observa al dispositivo BS, y que contiene un generador de tono (Su1), para ser acoplado al abonado "A", cuando su solicitud de conexión no pueda ser satisfecha porque el abonado "B" esté ya ocupado o porque los dispositivos de conexión (ABJ o KR) se encuentren ocupados.

Asumiremos que el dispositivo BS, es de baja resistencia óhmica, así que los 4 abonados pueden estar conectados hacia él simultáneamente.

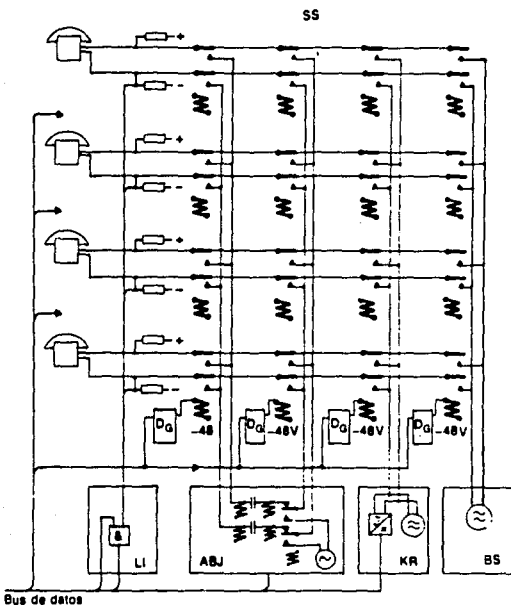


FIG 4.1.B DISPOSITIVOS LI,ABJ KR y BS

4.2 FUNCIONAMIENTO DE LA INTERFACE.

Las técnicas aplicadas en las diferentes partes de nuestra Central Telefónica Electrónica, son las mismas que se aplican hoy en los Sistemas de Control por Programa Almacenado, esto es, la parte de Conmutación utiliza elementos electromecánicos y también circuitos lógicos electrónicos y la parte de control utiliza técnica electrónica de procesamiento de datos con programa y datos almacenados.

Las diferencia en las técnicas usadas en las dos partes es el resultado de muchos factores, como por ejemplo el aspecto tradicional de la técnica electromecánica usada en el campo de la conmutación.

Esta diferencia de técnicas utilizada en nuestra Central Telefónica, significa que debemos tener una parte interface, a fin de lograr la comunicación en la parte de conmutación con la parte de control, por un lado los elementos electromecánicos generalmente requieren para su operación voltajes superiores que los necesarios para los circuitos electrónicos, y por otro lado la velocidad de operación de los circuitos es diferente, ya que mientras los cambios de estado de los elementos electromecánicos requieren entre 5 y 10 milisegundos, los cambios de estados de los circuitos electrónicos son del orden de los microsegundos.

En otras palabras los circuitos electrónicos son mil veces más rápidos que los circuitos electromecánicos; la diferencia de velocidades de operación, de las partes de conmutación, interface y control la podemos observar en la figura 4.2.1

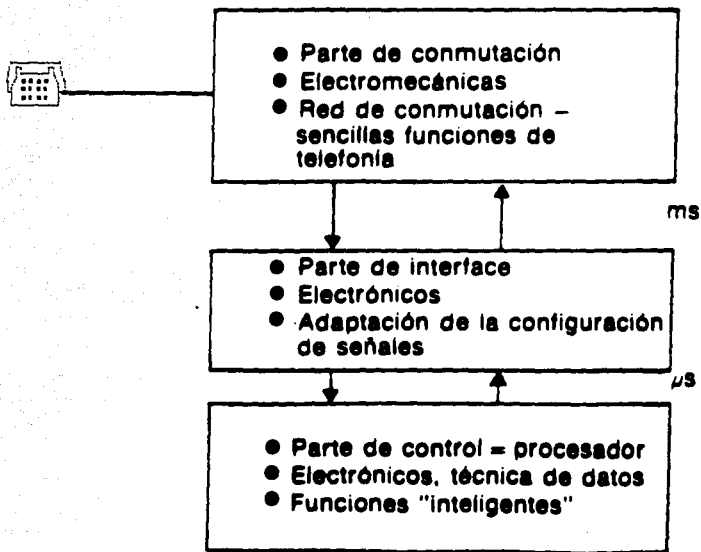


FIG. 4.2.1 FUNCIONES DE LAS PARTES DE CONMUTACION, INTERFACE Y CONTROL.

Para poder compensar estas diferencias de voltajes y velocidades, es necesario contar con la parte interface, la cual consta de circuitos electrónicos bastante rápidos para reaccionar a nivel de los microsegundos como se requiere en la parte de control.

Pero también es necesario contar con un elemento de memoria en la parte interface, que retenga la orden recibida de la parte de control y la transfiera a la parte de conmutación con una duración suficiente para lograr su aceptación por parte de los elementos que forman la parte de conmutación, este elemento puede ser un FLIP-FLOP como lo podemos ver en la figura 4.2.2

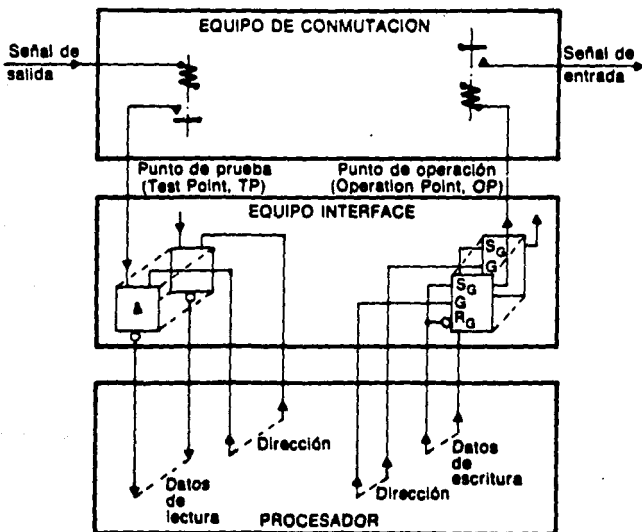


FIG.4.2.2. PARTE INTERFACE EN RELACION CON LAS OTRAS PARTES

Como se puede observar en la figura 4.2.2 la parte interface tiene dos divisiones, la parte de lectura y la parte de escritura.

La parte de lectura es para indicar a la parte de control, lo que sucede en la parte de conmutación, usando señales de dirección que vienen del microprocesador para abrir una compuerta y enviar el estado de un relé por ejemplo.

Esto quiere decir que la señal de lectura se detecta hasta que la parte de control lo decida, y como respuesta a esta lectura, la parte de control envía una señal dando una dirección a la parte de escritura de la interface y con esto hacer operar al relé deseado, es por esto que se requieren de FLIP-FLOPS, en la parte interface para que exista diálogo entre la parte de conmutación y la parte de control.

Después de la orden de operación, el FLIP-FLOP retiene su estado y opera su mecanismo.

Con frecuencia se leen o escriben varias posiciones de bit en paralelo, las que constituyen luego una "palabra".

Una palabra que consta de un número de puntos de prueba se llama "palabra de prueba" y la que consta de varios puntos de operación se denomina "palabra de operación."

La señal requerida para la lectura o escritura de información consta de bits de direcciones y de órdenes. La dirección de palabra indica la palabra a ser leída o escrita y la orden de lectura, RED, o de escritura, WRD, dan la función a realizar, es decir, si se va a leer o a escribir. Cada dirección indica una palabra de una determinada cantidad de bits, por ejemplo B, la que será escrita o leída.

En nuestra pequeña central podemos, por ejemplo usar la palabra de dirección "0" para probar el estado de las líneas de abonado (un bit por abonado) y para operar la vía de conmutación para un abonado que actúe como abonado A. (Un bit por abonado).

La palabra en la dirección 1 puede contener puntos de prueba para recepción de aparatos de teclado, y puntos de operación para la vía de conmutación del abonado B etc. Esto se dará en detalle posteriormente.

Estos procesos se pueden analizar en la figura 4.2.3 y se implementarán en el tema 5 de esta tesis.

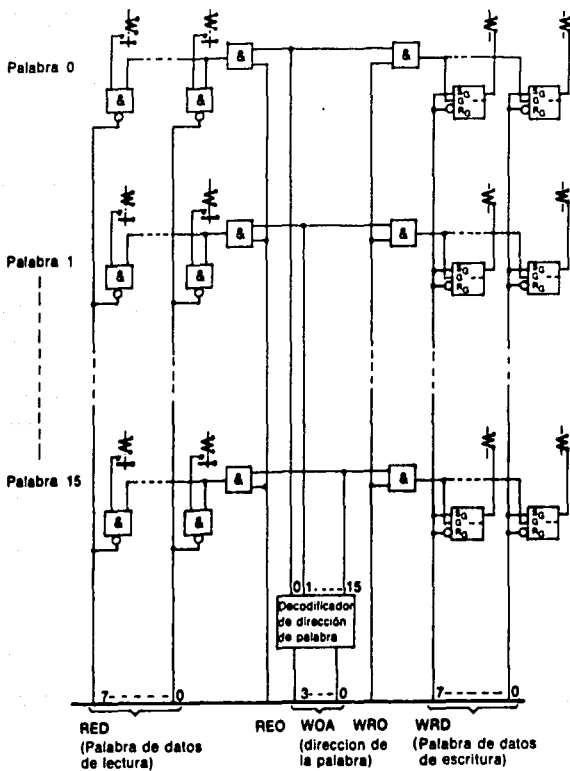


FIG. 4.2.3 EJEMPLO DE COMO SE ENVIAN LAS INSTRUCCIONES DE LA PARTE DE CONTROL A LA PARTE DE CONMUTACION.

4.3 FUNCIONAMIENTO DE LA PARTE DE CONTROL.

La parte de control, está formada principalmente por el microprocesador con sus respectivas memorias de programas y datos que realiza las funciones inteligentes, como son la identificación e interpretación de los cambios de estado en la parte de conmutación, así como también ordena la operación de los relés en la misma parte de conmutación, todo esto a través de la parte de interface.

El microprocesador debe conocer lo que está sucediendo en la parte de conmutación todo el tiempo, como por ejemplo, si un abonado descuelga o repone su microteléfono y también otras funciones como son las de activar la parte de conmutación para realizar la conexión de un abonado con otro por lo que debe de existir un intercambio de información en ambas direcciones.

Esto quiere decir que debe existir información de la parte de conmutación hacia la parte de control y viceversa a través de la parte de interface como se observó en la figura 4.2.1.

El microprocesador verifica vía la parte de interface lo que sucede en la parte de conmutación, usando señales de dirección para abrir una compuerta, a través de la cual obtendremos la información del estado de un relé de la parte de conmutación, a esta prueba se le llama lectura (READ) y la respuesta conseguida de la lectura, se llama dato de lectura (READ DATA).

Este principio de prueba, muestra que una señal de la parte de conmutación no afecta automáticamente al microprocesador ya que como se dijo anteriormente la señal no se detecta hasta que el microprocesador decida leerlo con el fin de conocer si realmente existe.

Para que una señal sea detectada en la parte de control, el microprocesador debe leer la señal en los puntos de prueba a intervalos regulares, puesto que una señal corta necesita un intervalo corto entre las lecturas, de otra forma el microprocesador podría perder una señal.

Cuando un relé de la parte de conmutación, va a ser operado, el microprocesador da una orden de operación, indicando una dirección al FLIP FLOP particular que controla al relé deseado, dicho FLIP FLOP será fijado o despejado con el dato de escritura (WRITE DATA) en él, en forma de un "1" lógico o un "0" lógico, después de la orden de operación el FLIP FLOP retiene su estado y opera o suelta a su relé.

La parte de control está formada por el microprocesador 8085 y una página de memoria, la función del microprocesador 8085 es ejecutar programas. Durante su ejecución, los programas se almacenan total o parcialmente en una memoria principal M, exterior a la propia CPU, la figura 4.3.1 nos muestra como está organizado una CPU básica utilizando el microprocesador 8085.

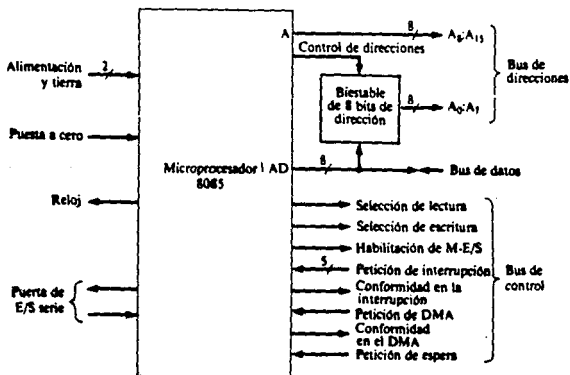


FIG 4.3.1 CIRCUITO CPU UTILIZANDO EL MICROPROCESADOR 8085

El microprocesador 8085 procesa palabras de 8 bits tanto en la CPU como en la memoria principal M, por lo que podemos direccionar hasta 256 bytes de memoria.

El microprocesador 8085 también puede direccionar palabras de 16 bits para lo cual se necesita utilizar las líneas A y AD.

Pero para nuestro uso que es muy pequeño no se necesitaría direccionar dichas palabras de 16 bits puesto que solo tenemos 4 teléfonos y no necesitamos mayor rapidez, pero en nuestros programas utilizaremos 12 bits debido a que las instrucciones están dadas por una parte paramétrica de 8 bits y por otra parte de operación de 4 bits, por lo tanto es necesario utilizar las líneas "A", que son de A₈ a A₁₅ y las salidas AD que representan a las salidas A₀ a A₇.

El 8085 multiplexa, es decir comparte temporalmente determinadas patas para multiplexar señales de datos y direcciones.

Otra característica del 8085, es que tiene un generador de reloj, en la propia pastilla y todas las líneas de control del bus del sistema están conectadas directamente al 8085 así como el bus de datos bidireccional de 8 bits.

Sin embargo, solo las 8 líneas denominadas A8 a A15 de orden mayor de direcciones están unidas directamente al 8085, los restantes 8 bits de dirección están multiplexados con las palabras de datos a través del bus AD de 8 bits de direcciones/datos, por lo tanto, para obtener 16 líneas paralelo de dirección, hay que utilizar un biestable externo de 8 bits como se muestra en la figura 4.3.1 vista anteriormente en las entradas ADO a AD7.

El chip del microprocesador 8085 en donde se ve como están distribuidas las patas de la pastilla, lo podemos observar en la FIG 4.3.2

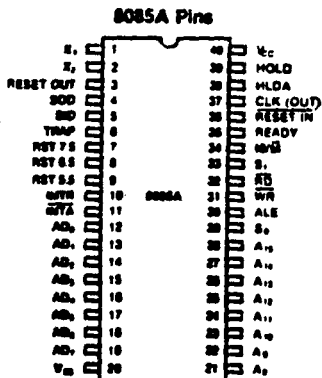


FIG. 4.3.2 CHIP DEL MICROPROCESADOR 8085

El 8085 está formado básicamente por una ALU (Unidad Aritmética y Lógica) de 8 bits, un acumulador A, de 8 bits, un Registro Indicador o de Estado de 5 bits (SR) y 6 Registros de Trabajo de propósito general denominados B,C,D,E,H y L.

Existe también en el 8085 dos registros de 16 bits dedicados a direcciones, el contador del programa PC y el Puntero de pila SP.

PC contiene la dirección de la palabra de instrucción siguiente y SP contiene la dirección del byte situado en la cabecera de la pila definida por el usuario en la memoria principal M. PC se incrementa automáticamente en uno, cada vez que se capta de M un byte de instrucción.

Cada vez que se extrae o inserta un byte en la pila, SP se incrementa o decrementa en uno. La estructura del microprocesador 8085, la podemos ver en la figura 4.3.3

En el 8085 se obtienen registros adicionales de direcciones emparejando los 6 registros de trabajo, para formar 3 registros de 16 bits, denominados BC, DE y HL. La pareja de registros HL tiene un significado especial dado que se utiliza como un registro de direcciones de memoria implícita en muchas instrucciones, tales como por ejemplo ADD A,M que significa la operación $A_i = A + M$ (HL).

En el 8085 se utilizan puertos de entrada y salida (E/S) aisladas o sin direccionamiento a memoria. El tamaño de la dirección de E/S está limitado a 8 bits de tal manera que se pueden asignar hasta 256 direcciones o dispositivos de E/S.

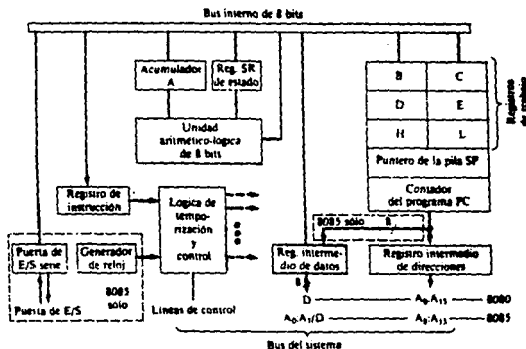


FIG. 4.3.3 ESTRUCTURA DEL MICROPROCESADOR 8085

Estas direcciones se utilizan únicamente en las instrucciones de E/S y por lo tanto son independientes de las 64 K direcciones que se pueden asignar a M.

El 8085 dispone de más de 70 tipos de instrucciones diferentes y se alimenta con +5 volts.

Debido a que no vamos a manejar un volumen grande de información de memoria principal utilizaremos el chip 2111 de INTEL que es una memoria RAM estática de 256×4 bits como se muestra en la figura 4.3.4

Esta memoria es de 1 Kbits, además tiene un bus de datos bidireccional, un bus de direcciones de 8 bits, y dos entradas de selección de pastilla, CE1 y CE2.

Estas dos líneas de selección de pastilla pasan por una compuerta AND, de manera que ambas deben estar habilitadas, es decir $CE1 = CE2 = 0$ para que se habilite la pastilla, esto quiere decir que cuando $CE1 = CE2 = 1$, el bus de datos del 2111 entra en el estado de alta impedancia.

Esta pastilla de memoria RAM, también tiene una línea de entrada de control extra denominada OD (Output-disable o salida inhabilitada) que deshabilita los circuitos de salida del 2111, independientemente de los estados de habilitación de escritura y de selección de pastilla, en otras palabras, la línea OD se activa durante los ciclos de escritura para evitar que se lean datos internos espúreos desde el 2111 al bus común de datos, mientras este último se está utilizando como bus de entrada para transferir a la RAM datos suministrados externamente.

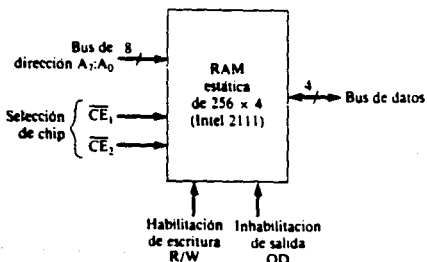


FIG. 4.3.4 CI RAM nMOS ESTADICO DE 1 Kbit, EL INTEL 2111

La línea extra de selección de pastilla de la RAM 2111, esta diseñada para simplificar la ampliación de la capacidad de almacenamiento como por ejemplo, si se quiere diseñar una memoria RAM DE 1024 por 8 bits es decir 1 K byte de memoria RAM se necesitan 8 CIs 2111 y que pueden configurarse esencialmente con la misma organización 4 x 2 que aparece en la figura 4.3.5

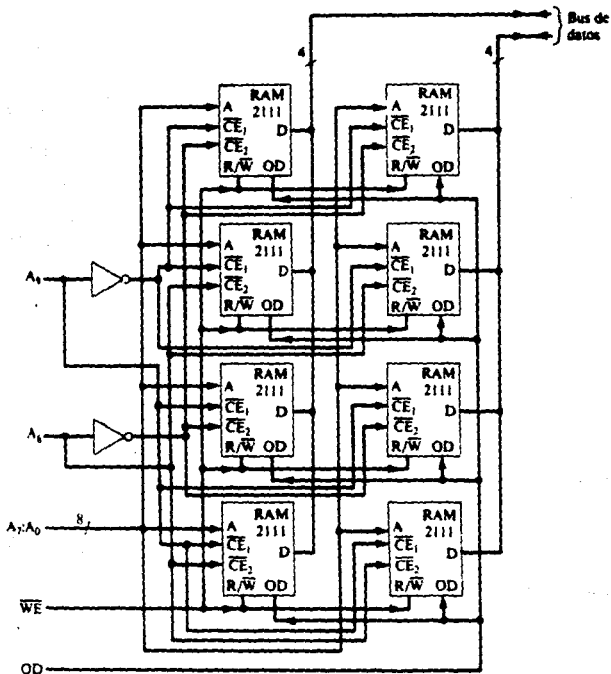


FIG. 4.3.5 MEMORIA RAM DE 1 Kbyte

Para la memoria de programas se utilizarán memorias PROM, debido a que este tipo de memorias pueden borrarse y reprogramarse utilizando un equipo sencillo, las PROM'S son particularmente recomendables cuando se trata de diseñar prototipos como es el caso de nuestra Central Telefónica Electrónica.

La memoria PROM tiene las conexiones externas como aparecen en la figura 4.3.6 que incluye un bus de direcciones A de p bits, que en nuestro caso será de 8 bits, un bus de datos D de m bits y que en nuestro caso será de 8 bits, también tiene una línea de control CS que es de selección de pastilla.

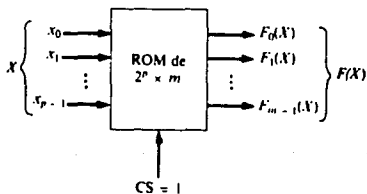
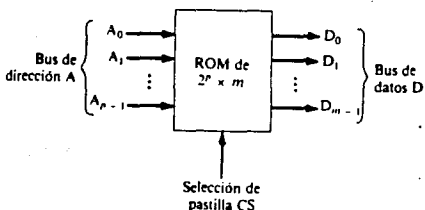


FIG.4.3.6 CHIP DE UNA MEMORIA EPROM $2^p \times m$ bits.

Cuando la PROM esta habilitada (CS = 1) y se aplica una dirección de 8 bits en A, en D se lee la palabra de 8 bits cuya dirección esta en A, por lo que se puede decir que la señal D que aparece en cada línea de datos, está determinada por la dirección A.

En nuestra Central Telefónica podemos utilizar la PROM 2708 que es de 1 Kbit para almacenar programas y se puede borrar exponiéndola a la luz ultravioleta y a continuación poder reprogramarla eléctricamente.

La parte de control, también incluye los puertos de entrada y salida (E/S) que consisten en biestables que en este caso serán de 8 bits direccionales, que puedan utilizarse como puertos de entrada y salida para lo cual se utilizará el chip 8212 que consta de 8 biestables tipo D con reloj, cuya salidas se conectan a puertas adaptadoras triestado.

Con fines de direccionamientos se pueden utilizar dos líneas de selección del dispositivo DS1 y DS2 como el de la figura 4.3.7 en donde vemos también como se conecta la PROM 2708.

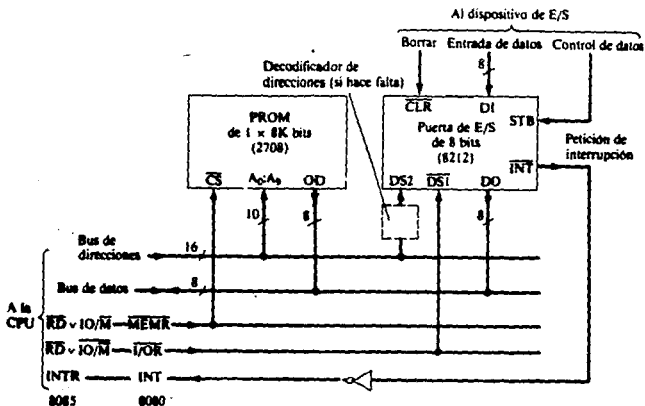


FIG. 4.3.7 CHIP DE LA MEMORIA PROM

El puerto de entrada 8212, se habilita por el microprocesador para lectura cuando DS1 y DS2 = 1 pero los datos se pueden escribir en el 8212, independientemente de los valores de DS1 y DS2 activando la línea de control STB. Por lo tanto el dispositivo de entrada puede cargar asincrónicamente datos en el 8212 utilizando STB.

Para que el microprocesador 8085 pueda leer los datos del 8212, este debe situar una dirección adecuada en el bus de direcciones del sistema y activar la línea de habilitación de lectura RDVIO/M que llega a DS1.

Al 8212 se le puede asignar cualquier dirección de 8 bits, pero en sistemas pequeños como es nuestro caso es posible realizar esta asignación de manera que no se requiera una decodificación explícita de la dirección.

El 8212 también puede generar señales de petición de interrupción hacia el microprocesador 8085 a través de su línea INT, ya que contiene un FLIP-FLOP para servicio de la petición que se cambia junto con INT al estado activo mediante una señal en la línea de control STB, y después de ser atendida por el microprocesador, el FLIP-FLOP e INT son desactivados.

En resumen la parte de control va a estar formado por el microprocesador, la memoria de datos RAM, la memoria de programas PROM y los puertos de entrada y salida como se observa en la figura 4.3.8

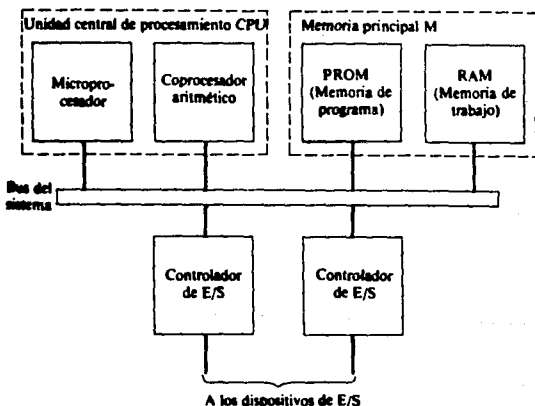


FIG. 4.3.8 ESTRUCTURA DE LA PARTE DE CONTROL.

4.4 SECUENCIA DE CONMUTACION EN LA CENTRAL TELEFONICA.

Cuando un abonado hace una llamada (descuelga su microteléfono) es detectado por el microprocesador, el cual a intervalos regulares por ejemplo cada 10 milisegundos lee el estado de los abonados vía LI.

El microprocesador interpreta el cambio de un bucle abierto a un bucle cerrado como una llamada. Este debe por lo tanto recordar el resultado de la prueba anterior.

Cuando el microprocesador ha detectado una nueva llamada debe conectar el abonado a un KR. Si KR se encuentra ocupado atendiendo otro abonado, el abonado que llama será conectado a BS, el que le dará tono de ocupado.

Para hacer la conexión el microprocesador opera el correspondiente relé en el punto de contacto.

Cuando se conecta KR, envía al abonado tono de marcar, entonces marcaría un dígito en su teclado, el correspondiente al abonado con el que desea su comunicación. El dígito se recibe en KR y se identifica con ayuda de los filtros, luego el dígito será leído por el microprocesador.

El microprocesador verifica también a KR en intervalos regulares, cuando el dígito es detectado por el microprocesador este conoce cuál abonado B debe ser conectado.

Si el abonado seleccionado está ocupado, la conexión no puede realizarse. Lo mismo diremos si ABJ está ocupado. En estos casos el abonado A será conectado al BS para que obtenga tono de ocupado.

Si tanto el abonado B como el juntor ABJ están libres, el microprocesador conectará el abonado A, al lado A del juntor ABJ y el abonado B al lado B. Luego, el microprocesador iniciará la señal de timbre por la operación del relé en ABJ. La señal continúa hasta que el abonado B responda. La conexión de la vía de habla se establece sin ayuda del procesador.

Cuando la conversación termina, los abonados reponen sus microteléfonos y el microprocesador libera la conexión. En nuestro caso la liberación de la llamada se inicia cuando cualquiera de los abonados cuelga. El circuito ABJ suelta y el otro abonado se conecta a BS hasta que reponga su microteléfono.

El microprocesador detecta si hay algún trabajo a realizar leyendo en LI y KR. Vía LI el microprocesador puede, detectar llamadas, obtener respuesta del microteléfono y respuesta de B. Vía KR el microprocesador puede detectar los dígitos que llegan a la central.

Las medidas que toma el microprocesador en las distintas situaciones, se pueden describir con más claridad con ayuda de un diagrama de flujo. Para este propósito usaremos los siguientes símbolos de la figura 4.3.9.






SÍMBOLO	NOMBRE	DESCRIPCIÓN
	Señal de entrada	señal de entrada que inicia una secuencia
	Acción	Trabajo que es ejecutado por el microprocesador
	Decisión	selección entre un número de caminos en el diagrama de flujo.
	Línea de flujo	Unión de las diferentes actividades en la secuencia correcta.
	Señal de salida	Señal de salida para activar otra secuencia.

FIG. 4.3.9 SÍMBOLOS UTILIZADOS EN UN DIAGRAMA DE FLUJO.

Con los símbolos anteriores podremos escribir lo que sucede cuando se hace una llamada en una forma clara y concisa como se observa en la figura 4.3.10

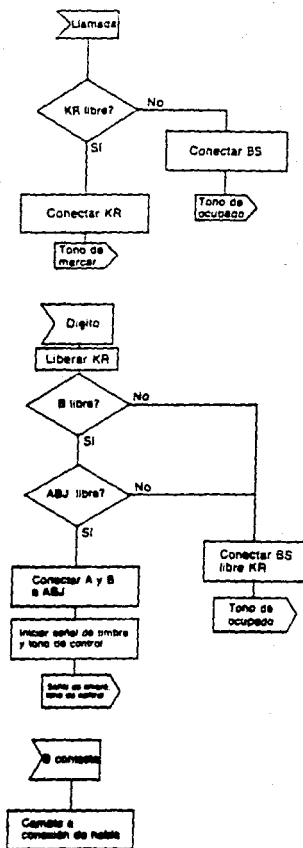


FIG. 4.3.10 DIAGRAMA DE FLUJO REPRESENTANDO FUNCIONES TELEFONICAS

5. SOFTWARE DEL SISTEMA

5.1 LENGUAJES DE PROGRAMACION

5.2 DIAGRAMAS DE FLUJO

5.3 PROGRAMACION

5.4 PRUEBAS

5.1- LENGUAJES DE PROGRAMACION

Para la escritura de programas en los microprocesadores, a la fecha se han desarrollado una gran cantidad de lenguajes de los cuales nos interesan basicamente 2 tipos:

a)-Una clase de lenguajes de propósito general y orientados a los problemas, y que son conocidos como "LENGUAJES DE ALTO NIVEL" tales como el COBOL, el FORTRAN, el BASIC y el PASCAL entre otros, que están diseñados para asemejarse a los lenguajes más frecuentemente utilizados para especificar técnicas o algoritmos para la solución de problemas, tales como la notación matemática o el lenguaje coloquial.

Idealmente hay una correspondencia uno a uno entre las sentencias, es decir las instrucciones escritas en el lenguaje de alto nivel y los pasos del algoritmo a programar, por ejemplo:

"Calcular la raíz cuadrada positiva de los números X,Y; sumarmas y llamar ALFA al resultado".

En lenguaje de alto nivel FORTRAN sería de la forma :

ALFA=SQRT(X) + SQRT(Y)

b)- Otra clase de lenguajes orientados a las máquinas conocidos como lenguajes "ENSAMBLADORES", representados por los lenguajes ensambladores de los microprocesadores 6800, y los de la familia 8080/8085, y como posteriormente se observará, el lenguaje ensamblador refleja la estructura de una familia en particular de microprocesadores y su uso se restringe a esta familia.

Por lo tanto existen las dos alternativas para programar a nuestro procesador, pero en la alternativa No. 1 que es lenguaje de alto nivel, el proceso sería el siguiente:

Primero tenemos un programa fuente o código base que es el programa en lenguaje de alto nivel, después se pasa a través de un computador comercial con programas "compiladores", que es como un programa traductor para la máquina, y posteriormente esta información, que ya está en "CODIGO MAQUINA", se graba esta información en cassette, para que después con esta información se pueda cargar la memoria de programas en nuestra memoria M.

En la alternativa dos, que es un código ensamblador, primero se pone en "código base" o "programa fuente", que en este caso es un programa de lenguaje ensamblador, propio del microprocesador que se está utilizando y posteriormente se pasa a través de un computador comercial con programas "ensambladores", que es como un programa traductor, que en este caso es un ensamblador para que después ya lo entrega como un programa objeto, que en este caso ya es un programa en lenguaje máquina.

La figura 5.1.1, nos muestra los pasos que se realizaron para la programación de una central telefónica electrónica llamada "AXE" tanto en lenguaje de alto nivel llamado "PLEX", así como en lenguaje ensamblador.

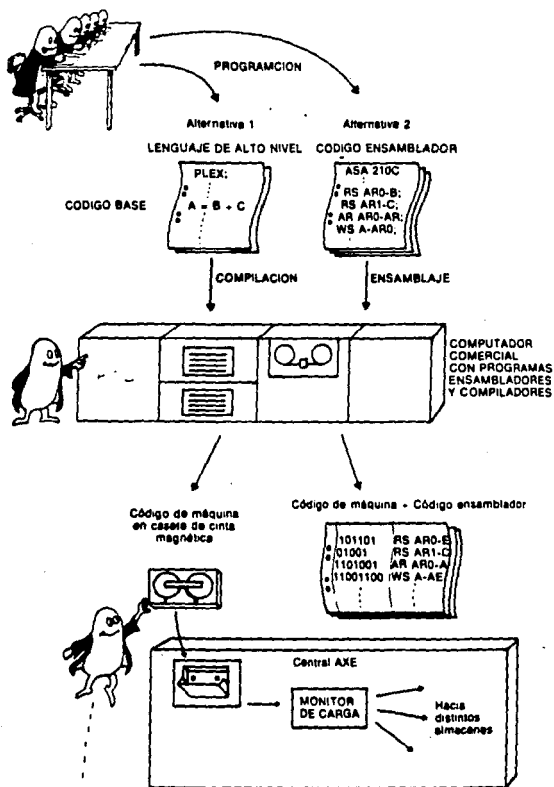


FIG. 5.1.1 PROCESO DE PROGRAMACION EN LENGUAJES DE ALTO NIVEL Y ENSAMBLADOR.

Normalmente los lenguajes de programación de alto nivel se diseñan para una amplia gama de tareas de cálculo, por lo que puede decirse que son de propósito general.

Así tenemos que el FORTRAN está dirigido principalmente a las aplicaciones de programación numérica en ciencia e ingeniería; el COBOL se diseñó para el procesamiento de datos comerciales, en donde el manejo eficiente de archivos alfanuméricos grandes es más importante que la eficiencia en cálculos numéricos; el PASCAL, es un lenguaje de alto nivel más reciente, cuya estructura simplifica el diseño y la depuración de programas. Otro lenguaje de alto nivel, es el PL/M, el cual está dirigido específicamente para la programación de los microprocesadores y que fue diseñado por INTEL; dicho lenguaje tiene muchas características del lenguaje PASCAL, pero además tiene características dependientes de la máquina y que interesan a los diseñadores de sistemas basados en microprocesadores, por lo tanto se puede considerar como intermedios entre los lenguajes de alto nivel convencionales y los lenguajes en ensamblador.

Un ejemplo de programación en lenguaje de alto nivel en telefonía, es el lenguaje "PLEX", utilizado en la Central Telefónica AXE de Ericsson, que en realidad es una versión del lenguaje de alto nivel PLM, esto se puede observar en la figura 5.1.1

Los programas en lenguaje ensamblador, son los lenguajes de programación más ampliamente utilizados en el diseño de sistemas basados en microprocesadores, ya que permiten que las instrucciones se codificaran con símbolos alfabéticos y dígitos decimales que resultó más cómodo para los programadores.

Los programas en lenguaje ensamblador son, en una primera instancia, versiones simbólicas y más legibles de los programas en lenguaje máquina. Si el programa es muy grande, es necesario dividir dicho programa en subprogramas.

Esta posibilidad de dividir el programa en módulos permite que cada uno de estos sea programado por separado para posteriormente poderlos combinar o ensamblar de diferentes formas de tal manera que se pueda construir un programa único para su ejecución. Este proceso de ensamblaje es el que da origen al término "lenguaje ensamblador".

Cuando se realice un diseño de un prototipo, se debe analizar que tan complejo es el programa, que funciones realizará y en base a esto elegir el tipo de lenguaje a utilizar.

El lenguaje utilizado en el programa del equipo SX-10 de Mitel, es el lenguaje ensamblador del microprocesador 8085 o el lenguaje ensamblador del microprocesador 6800 ya que cada microprocesador cuenta con su propio lenguaje ensamblador.

Estos programas se escriben en la memoria en forma de programas que dialogan con el microprocesador a través de un bus de 8 bits como se vio en el tema 4.

Para comprender como se relaciona el microprocesador con las memorias de programas y datos, a la memoria de programas se le llamará PRS (Program Store) que indica "ALMACEN DE PROGRAMAS".

A la memoria de datos se le llamará DAS (Data Store), que indica "ALMACEN DE DATOS".

A la unidad procesadora se nombrará como CPU, que sera "LA UNIDAD CENTRAL DE PROCESAMIENTO", como lo vemos en la figura 5.1.2

CPU es la unidad activa, la cual lleva trabajos tales como Tranferencia de datos y toma de decisiones.

PRS y DAS son almacenos, y por tanto, parte pasiva del microprocesador.

El almacén de programas PRS contiene programas que indican a CPU lo que debe de hacer, y el almacén de datos DAS, contiene los datos con los cuales trabajará el microprocesador.

Es por supuesto posible ubicar los programas y los datos en un mismo almacén que sería representado por la memoria M, sin embargo es más práctico utilizar almacenos separados, pues de esta forma CPU podrá leer en el almacén de programas y al mismo tiempo leer o escribir en el almacén de datos, aumentando así la rapidez del microprocesador.

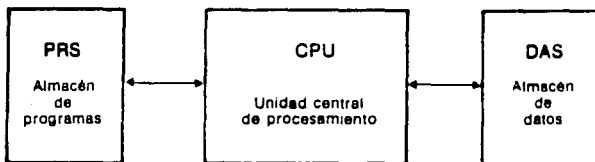


FIG. 5.1.2 RELACION ENTRE LA UNIDAD PROCESADORA Y LAS MEMORIAS DE PROGRAMAS Y DATOS.

El CPU deberá también interconectarse con la parte de conmutación como ya se ha descrito anteriormente.

Como ejemplo de interconexión entre las tres partes del microprocesador, PRS-CPU-DAS, veremos en la figura 5.1.3 que muestra a la CPU que contiene un registro llamado Instruction Address Register (Registro de direcciones de instrucciones) "IAR", cuyo contenido, es la dirección para una palabra en el almacén de programas.

- Cuando esta dirección se envía a PRS, el microprocesador obtiene como respuesta el contenido de la dirección indicada, es decir, recibe una instrucción.

En este caso en la dirección "m", encontraremos la instrucción "leer la palabra 2".

La instrucción no está en texto claro por supuesto, sino que estará en código binario.

La instrucción leída será situada en el registro de instrucciones (IR).

El microprocesador interpreta el código binario, toma en cuenta que se debe realizar una lectura en la dirección 2 en el almacén de datos. La dirección será enviada al almacén de datos.

Como respuesta obtendremos el contenido de esta dirección en el registro de datos quedando así completa la ejecución de la instrucción.

La palabra que ha sido leída puede ser procesada por la siguiente instrucción.

En general, las instrucciones son ejecutadas en el mismo orden en que se encuentran situadas en el almacén de programas.

De esta manera CPU necesita solamente adelantar un paso el contenido de su registro de direcciones de instrucciones (IAR), para alcanzar la siguiente instrucción.

Lo anterior es realizado fácilmente si el registro de direcciones de instrucciones (IAR), se construye como un contador binario el cual adelantará un paso por medio de un pulso.

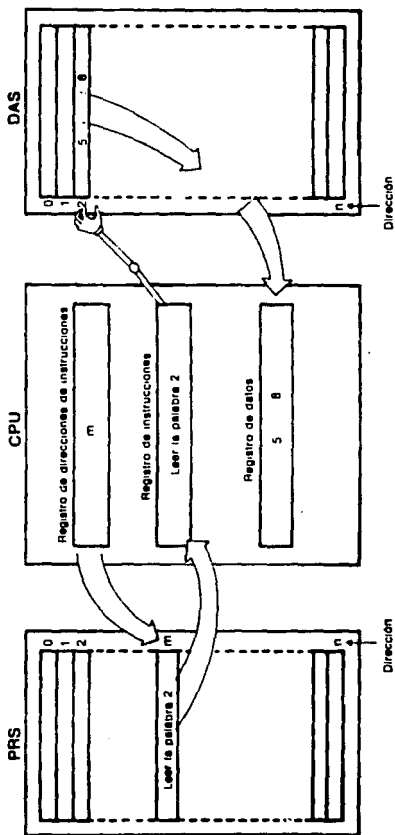


FIG. 5.1.3 EJEMPLO DE LECTURA DE UNA INSTRUCCION EN CPU-PRS-DAS

Con la ayuda de tales contadores, CPU puede manejar las instrucciones en el orden de direcciones de "Ejecución Normal" como lo podemos observar en la figura 5.1.4 en la que el registro IAR indica una dirección en PRS y la instrucción leída en esta dirección es almacenada en IR.

Mientras esta instrucción está siendo llevada a cabo, el contador IAR avanza un paso, así que la nueva instrucción puede ser iniciada tan pronto como la anterior haya terminado.

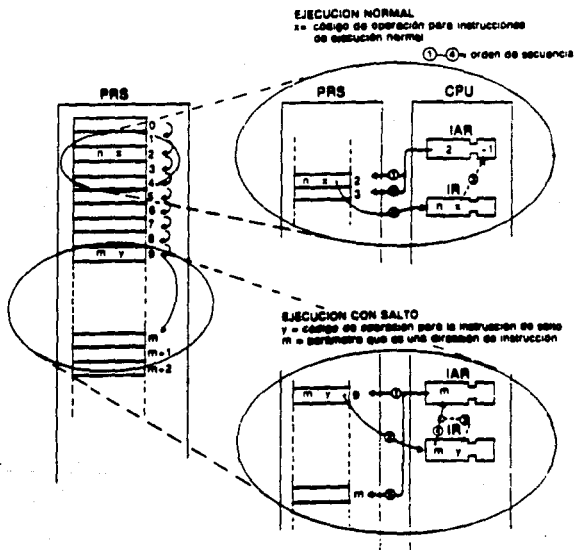


FIG. 5.1.4. EL REGISTRO IAR.

A veces es necesario hacer saltos (JUMP) en el programa, en tal caso la dirección de saltos esta almacenada en IAR, la siguiente instrucción será entonces tomada de aquella dirección. La dirección de salto puede, por ejemplo estar incluida en el código binario de la instrucción que ordena el salto.

Las instrucciones en PRS, estan dadas en forma de palabras en código binario. Por ejemplo la instrucción "READ WORD 2" (Leer la palabra 2) puede dividirse en dos partes: "El código de operación" y "la parte paramétrica", como se puede observar en la figura 5.1.5

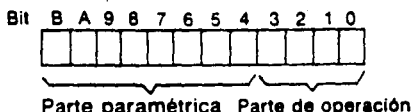


FIG. 5.1.5 EJEMPLO DE UNA INSTRUCCION EN PRS.

El código de operación indica "que va ha ser hecho" (tipo de operación) y para nuestro ejemplo sería "Leer", mientras que la parte paramétrica indica " donde se va a realizar la operación", por ejemplo: "se va a leer la palabra 2".

Si en lugar de la anterior instrucción, hubiésemos tenido la instrucción " READ WORD 5", que es del mismo tipo de la primera, la parte de operación sería igual, solo la parte del parámetro tendrá diferente valor.

Cuando el programador escribe un programa con un gran número de instrucciones, se prefiere un método corto y conciso de escritura.

Por esta razón, a cada instrucción se le da una designación corta de letras, es decir una sigla.

Por ejemplo a la instrucción READ WORD FROM STORE (Leer una palabra del almacén) se le puede asignar la sigla RWS.

El parámetro puede ser dado como un número, por ejemplo podríamos escribir RWS,X, en donde X es el parámetro, este método de escritura se le llama "SOURCE CODE" (Código Base ó Código Ensamblador).

Antes de escribir esta instrucción en un programa, debe ser convertida a código binario. A esta conversión se le llama "ASSEMBLY" (Ensamblar).

La instrucción RWS,2 puede, por ejemplo ser representada por 0000 0010 0000, a este código binario se le llama "OBJECT CODE ó MACHINE CODE (Código de Máquina).

El proceso de ensamblar es realizado por computadora, usando programas ensambladores.

Para que el computador pueda entender el código base, hay reglas más estrictas acerca de como debemos escribir este código, por ejemplo, como usaremos los signos de puntuación, en que orden debemos escribir los parámetros si es que hay más de uno, y así sucesivamente.

El número de bits en el código de máquina, está seleccionado de tal manera que es suficiente para expresar todo lo que se requiere, sin demandar gran capacidad de almacenamiento.

Para estar más familiarizados con los conceptos de función, sigla y código ensamblador, a continuación daremos estas definiciones:

Función: Es una pequeña descripción verbal de lo que logra la instrucción que el microprocesador realice.

Sigla: Es el nombre de la instrucción pero considerablemente reducido de la descripción en Inglés.

Código Ensamblador: Se obtiene de la sigla y este código es una abreviación nemotécnica.

El código ensamblador contiene tanto caracteres que siempre deben ser escritos, como los que son reemplazados por valores de dígitos, u otros símbolos llamados variables.

Las variables estan incluidas normalmente en la parte paramétrica.

En nuestro ejemplo, RWS debe escribirse siempre, mientras que el parámetro X será reemplazado por la dirección en la cual vamos a hacer la operación a un determinado tiempo.

Código Máquina: Es el estado que tienen los elementos de memoria cuando la instrucción ha sido situada en el computador o microprocesador.

Todas estas definiciones las podemos observar en la figura 5.1.6

Función En DAS la palabra en dirección X será escrita en el registro de datos en CPU

Sigla Read Word from Store

Código ensamblador

— como modelo



— como se escribe en el programa, en el cual deseamos leer una palabra con dirección 2

RWS, 2

Código de máquina (código binario)

— Asumimos que la instrucción RWS tiene el código de operación No. 0

0000 0010	0000
└──────────┘ └──────────┘	
Código paramétrico	Código de operación
= 2	= 0

FIG 5.1.6 RELACION DE LAS DIFERENTES DEFINICIONES.

La fig. 5.1.7 muestra la interconexión de CPU con el almacén de programas PRS.

Asumimos que el registro de direcciones de instrucciones IAR contiene una cierta dirección de instrucción por ejemplo 29.

Esta dirección será transferida al almacén de programas y la instrucción direccionada será leída de PRS.

El código de operación, el cual dice el tipo de función de la instrucción, es decodificada en CPU, después de lo cual la instrucción se llevará a cabo, es decir será ejecutada.

Para su logro, CPU envía señales de control las que controlan los datos transferidos y las señales de tal manera que el trabajo ordenado por la instrucción se realiza.

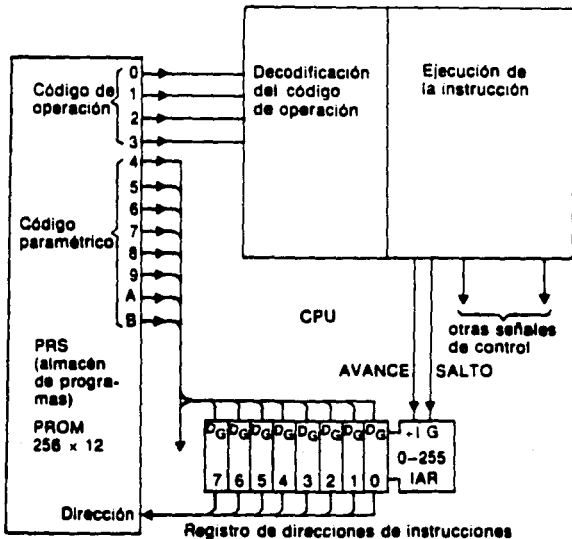


FIG. 5.1.7 INTERCONEXION DEL CPU CON PRS.

Junto con la ejecución de una instrucción, la siguiente debe también ser preparada.

Por lo tanto, el trabajo de una instrucción incluye el envío de una señal de AVANCE, lo cual avanza un paso al contador IAR con la posición de bit "+1".

La nueva dirección obtenida en esta forma va hacia PRS, leyéndose la siguiente instrucción a la anterior.

En ciertos casos es necesario realizar saltos dentro del programa, a instrucciones diferentes de la siguiente.

Estos se inician mediante una instrucción especial de salto (JUMP).

La parte paramétrica de esta instrucción incluye la dirección de la instrucción siguiente.

Cuando se decodifica el código de operación para la instrucción de salto, IAR recibe un pulso de control SALTO en su entrada "G" con lo que la parte paramétrica de la instrucción será introducida en el registro IAR, después de lo cual se ejecutará la instrucción localizada en la nueva dirección.

El trabajo realizado por una instrucción, a menudo consta de varias partes que deben realizarse con cierto orden cronológico.

A lo anterior se acostumbra llamarlo PRINCIPIO DE MICRO-PROGRAMA.

Para conseguir esta secuencia en tiempo, debemos utilizar un contador que avance en pasos un número de direcciones en una memoria ROM que contiene el microprograma para la respectiva instrucción.

Para cada palabra direccionada en ROM, enviará señales de control, las que aseguran que la instrucción será ejecutada

La fig. 5.1.8 muestra una parte de esta memoria y el registro MAR (Micro instrucción address register) Registro de Direcciones de Microinstrucciones.

Este registro es avanzado por un pulso de un generador el que da un pulso cada 250 nanosegundos ($=0.25 \mu s$).

Cuando se inicia una micro-instrucción, el código de operación se introduce a los bits de posición 2 a 5 en MAR. Al mismo tiempo, las posiciones de bits 0 y 1 son puestas en cero. La razón de esto se explica más tarde.

Esta dirección de 6 bits va de MAR hasta el almacén de micro-programas, donde obtendremos una palabra almacenada de 13 bits. Cada palabra (cada fila horizontal) es una micro-instrucción y cada "1" de ésta (cada diodo) constituye una señal de control.

Estas señales son enviadas durante 250 nanosegundos, después el contador MAR avanza un paso y se indicará la siguiente micro-instrucción la que enviará sus respectivas señales de control.

El generador de pulsos genera períodos fijos de 250 ns, durante los cuales las diferentes señales de control se enviarán a los diferentes dispositivos que intervienen en la ejecución de la micro-instrucción.

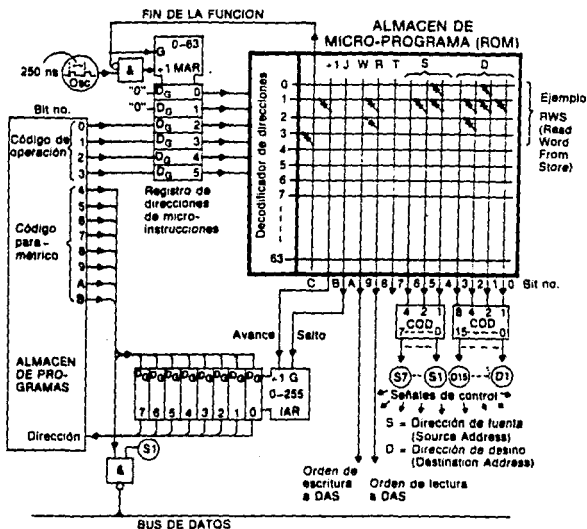


FIG. 5.1.8 RELACION DEL MICROPROGRAMA Y MAR.

Un número de tales periodos, forman una instrucción. En nuestro microprocesador hemos estandarizado el número de pasos para cada instrucción igual a 4.

Durante estos 4 pasos se ejecuta completamente la instrucción. De esta manera la ejecución de cada instrucción tarda un microsegundo. (Vemos ahora porque fijamos a cero las posiciones de bits 0 y 1 en MAR antes de iniciar la ejecución de cada instrucción, cada cuarta dirección MAR tiene, por supuesto, los últimos bits menos significantes iguales a cero).

El micro-programa debe incluir la preparación para el arranque de la siguiente instrucción. En la FIG. 5.1.8 vemos el micro-programa para la instrucción RWS en las direcciones 0 a 3. El código de operación para esta instrucción es 0. De esta forma en el momento de iniciar la ejecución, MAR contendrá 00000.

MAR avanzará a 1, 2 y 3 cuando se ejecute la instrucción. En la posición 1 el micro-programa dá una señal de control en la posición de bit "B", la que avanza un paso a 1AR, indicándose así la dirección para la siguiente instrucción.

Cuando el micro-programa ha avanzado hasta la posición 3, se envía una señal de control sobre el hilo C indicando FIN DE LA FUNCION.

El código de operación para la siguiente instrucción será colocado desde PRS en MAR. Al mismo tiempo se bloquea el generador de pulsos. Cuando se ha indicado una nueva dirección en MAR la señal FIN DE FUNCION desaparece y el conteo de MAR puede iniciarse nuevamente. La siguiente instrucción será ejecutada.

La ejecución de una instrucción se lleva a cabo con ayuda de los bits 0 a A de las micro-instrucciones.

Gran parte del trabajo de las instrucciones se hace en el transporte de datos dentro de CPU. Esta transferencia tiene lugar via el "BUS DE DATOS" al cual están conectadas las diferentes partes de CPU.

El control de estas transferencias se hace como a continuación se explica. Los bits 4 a 6 dan la FUENTE de transferencia. Las direcciones de FUENTE y DESTINO están dadas por la micro-instrucción en código binario.

Con una dirección de 3 bits podemos llegar hasta siete diferentes fuentes y con cuatro bits de dirección, hasta 15 destinos (una posición se usa para "ausencia de dirección").

Las direcciones son decodificadas en los circuitos COD y sus señales de salida abren las compuertas que controlan la transferencia de información via el bus de datos.

En la fig. 5.1.8 podemos ver lo anterior (SI = source 1 = fuente 1) la que controla el paso de la parte paramétrica de la instrucción hasta el bus de datos.

Notar que los datos en el bus son transportados en forma invertida, los "unos" se invierten a "ceros" y viceversa. Esto se hace debido a que hemos supuesto que un cero es más "fuerte" que un uno.

Al bus de datos están conectadas un cierto número de compuertas en paralelo. Las compuertas cerradas deben enviar "unos" al bus, de otra forma no sería posible cambiar el estado del bus cuando transportemos un dato desde cualquiera de las compuertas.

Para la interconexión con el almacén de datos, el bit 8 da la orden de lectura y el bit 9 la orden de escritura. El bit A da la señal de SALTO, la que habilita al registro IAR para recibir la nueva dirección a la cual debemos saltar. Regresaremos luego para explicar el bit 7.

En la figura 5.1.9 se muestra al microprocesador completo. En este diagrama el almacén de datos DAS está en la parte inferior a la derecha.

Para la interconexión con el almacén de datos, CPU tiene un registro de direcciones ADR (Address Register) y un registro de datos DAR (Data Register).

La instrucción RWS, X, donde X es la dirección de la palabra a leer en DAS, se ejecuta de la forma siguiente:

- Fase 0 (Dirección 0 en el micro-programa): el parámetro X, dirección de palabra, se transfiere via el bus de datos hasta el registro de direcciones ADR. Esto se hace con ayuda de las señales S1 (fuente 1) y D2 (destino 2).
- Fase 1 La transferencia de S3 hasta D7 significa transferencia de DAR hasta DRB. Esto es una preservación del contenido "viejo" de DAR. Explicaremos más adelante porque se hace esto. En la misma fase avanzamos también la orden de lectura a DAS y avanzamos un paso al registro IAR por medio de la salida +1.
- Fase 2 Como resultado de la dirección y de la orden de lectura a DAS, podemos obtener de DAS el dato leído y con la ayuda de la señal D4 almacenarlo en DAR. La orden de lectura aún continúa.
- Fase 3 Esta es la fase final de la instrucción. El código de operación de la siguiente instrucción es llevado hasta MAR.

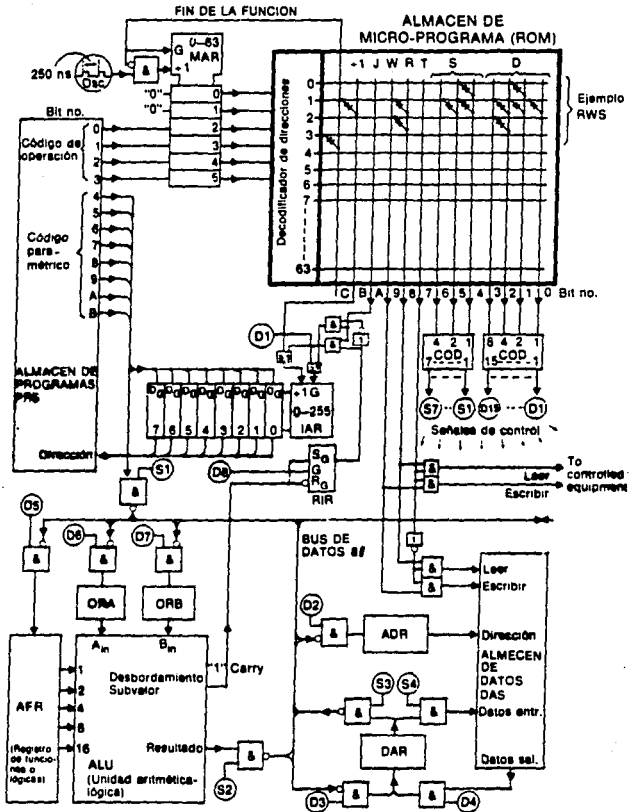


FIG. 5.1.9 ESTRUCTURA DEL MICROPROCESADOR CON SUS FUNCIONES.

El resultado de la instrucción RWS, X será entonces:

La palabra situada en la dirección X de DAS está ahora almacenada en DAR. Además, el contenido anterior de DAR ha sido preservado en el registro DRB.

En la figura 5.1.9 se puede ver que el MICROPROCESADOR contiene algunos circuitos que aún no hemos discutido. Antes de hacerlo, vamos a dar las seis principales instrucciones sobre las cuales se basa el trabajo del MICROPROCESADOR con datos del almacén de datos.

La tabla de la fig. 5.1.10 muestra el código ensamblador, la sigla y la función en el lado izquierdo y la disposición de bit en código de máquina en el lado derecho.

LISTA DE INSTRUCCIONES PARA EL MINI-PRO

Instrucción ensamblador Código base. Sigla. Función	Código de máquina											
	Código paramétrico					Código de operación						
Bit no.	B	A	9	8	7	6	5	4	3	2	1	0
RWS,X (Read Word from Store) La palabra en dirección X de DAS se transfiere a DAR. El contenido previo de DAR se almacena en ORB.	X	X	X	X	X	X	X	X	0	0	0	0
WWS,X (Write Word in Store) La palabra en DAR se escribe en dirección X de DAS	X	X	X	X	X	X	X	X	0	0	0	1
WWC,C (Write Word Constant) La constante C se escribe en DAR	C	C	C	C	C	C	C	C	0	0	1	0
DAO,M (Do Arithmetic or Logic Operation) Operación aritmética o lógica, según parámetro M, con los contenidos del DAR y ORB. El resultado se envía a DAR, y el residuo a RIR.	M	M	M	M	M	M	M	M	0	0	1	1
JUN,L (Jump Normal) Salto a la instrucción en dirección L.	L	L	L	L	L	L	L	L	0	1	0	0
JCZ,L (Jump Conditionally on Zero) Si RIR = 0, salto a la instrucción en dirección L. Si RIR = 1, avance normal de un paso.	L	L	L	L	L	L	L	L	0	1	0	1

FIG. 5.1.10 LISTA DE INSTRUCCIONES PARA EL MICROPROCESADOR.

La lista comprende las siguientes instrucciones:

- RWS, X La palabra en dirección X de DAS se transfiere a DAR. El contenido previo de DAR se conserva en DRB.
- WWS, X Escritura de una palabra en DAS. La palabra que está en DAR se escribe en dirección X de DAS.
- WVC, C Escritura de una constante en DAR. La constante, C, está en la parte paramétrica de la instrucción.
- DAD, M Realizar una operación aritmética o lógica. La operación requerida está dada por el parámetro M.
- JUN, L Llevar a cabo un salto en el programa a la dirección dada por el parámetro L.
- JCZ, L Esta instrucción permite el paso a la siguiente instrucción, o bien permite saltar a otra dirección dada por el parámetro L de acuerdo con el contenido de RIR. El contenido de RIR, es decir la selección de estas alternativas depende del resultado de una operación aritmética o lógica que se ejecuta antes de la instrucción JCZ.

La siguiente tabla, FIG. 5.1.11 muestra los MICROPROGRAMAS para las anteriores instrucciones. Con ayuda de los micro-programas es fácil ver como el procesador ejecuta las instrucciones.

Para la instrucción DAD, es necesario llegar hasta la unidad de operaciones aritméticas y lógicas en CPU.

Este es el código ensamblador que usamos para hacer posible los programas de trabajo del MICROPROCESADOR.

EL CONTENIDO DEL ALMACEN DE MICRO-PROGRAMAS (MPS) EN EL MINI-PRO

Instrucción ensamblador	Código de op	Dirección en MPS	Bit numero															
			C	B	A	9	8	7	6	5	4	3	2	1	0			
RWS, X Read Word from Store	0	0 1 2 3		1			1				1	1		1	1			
WWS, X Write Word in Store	1	4 5 6 7		1		1			1			1		1				
WWC, C Write Word Constant	2	8 9 A B		1							1				1	1		
DAO, M Do Arithmetic or Logic Operation	3	C D E F		1						1	1		1	1		1		
JUN, L Jump Normal	4	10 11 12 13		1												1		
JCZ, L Jump Conditionally on Zero	5	14 15 16 17			1													
SEÑALES DE CONTROL			Peso	1	1	1	1	1	1	4	2	1	8	4	2	1		
			Nombre	E	+1	J	W	R	T	S	D							

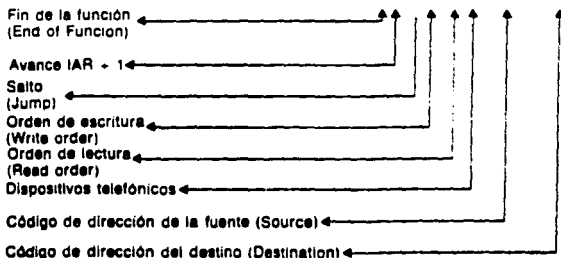


FIG. 5.1.11 CONTENIDO DEL ALMACEN DE MICRO-PROGRAMAS (MPS)

FUNCIONES ARITMETICAS Y LOGICAS.

Puesto que el sistema debe realizar cierto tipo de trabajo inteligente, éste debe contener funciones para hacer operaciones de tipo aritmética, comparativas u otras operaciones del proceso de datos.

La unidad que lleva a cabo la mayoría de estas funciones es llamada UNIDAD ARITMETICA-LOGICA. Tal unidad está construida de un número de diferentes circuitos integrados para lograr las funciones requeridas.

Hoy en día la mayoría de estas unidades están construidas en circuitos integrados que contienen toda la unidad y las cuales pueden realizar 64 tipos de operación estandard. La fig. 5.1.12 muestra un ejemplo de tales circuitos, "Arithmetic Logic Unit" (ALU).

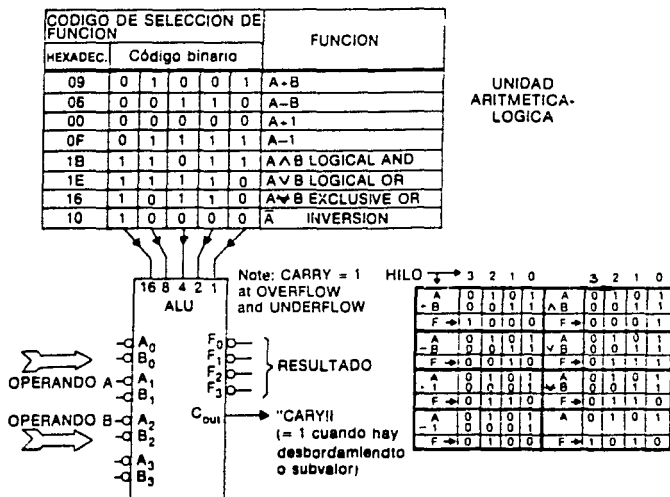


FIG. 5.1.12 CIRCUITO "ALU".

De la figura se puede ver que el circuito ALU trabaja únicamente en una longitud de palabra de cuatro bits y por lo tanto hay que interconectar varios circuitos ALU si se necesitan longitudes de palabra mayores que éstas.

La figura 5.1.12 muestra en forma de tabla algunas de las operaciones aritméticas o lógicas que puede realizar el circuito ALU. La tabla a la derecha muestra algunos ejemplos de tales operaciones.

Para seleccionar el tipo de operación, la cápsula usa las entradas SELECCION DE FUNCION marcadas como 1,2,4,8 y 16, cuyas combinaciones dan las diferentes funciones del circuito. La tabla de FUNCION contiene los símbolos usados para representar las diferentes operaciones. Para operaciones con dos números, uno de estos (OPERANDO A) se introduce vía las entradas Ao a A3; mientras que el otro (OPERANDO B) se introduce vía las entradas Bo a B3.

El resultado se obtiene vía las salidas Fo a F3 (RESULTADO). Para el caso de operaciones aritméticas se puede obtener también CARRY = 1 para el caso de sobre flujo (overflow) o de flujo mínimo (underflow). En operaciones lógicas obtenemos carry = 1 si el resultado (Fo a F3) es diferente de 0.

Si volvemos al dibujo del MICROPROCESADOR en la FIG. 5.1.9 vemos la unidad aritmética-lógica en la parte inferior a la izquierda. Los operandos A y B se introducen vía los registros ORA y ORB respectivamente.

La función deseada se obtiene por un código el que se coloca en el registro AFR (Arithmetic or Logic Function Register).

El resultado de la operación se coloca en el registro DAR, cuando se realiza una operación de suma o resta podemos obtener "overflow" o "underflow", los cuales hacen la salida carry igual a "1"; este valor se transporta hasta un flip-flop llamado RIR (Result Indicator Register). En el caso de operaciones lógicas obtenemos RIR = 0 si el resultado de la operación es cero, de otra manera RIR será igual a 1.

La suma de dos números que se encuentran en el almacén de datos, DAS, se realiza de la manera siguiente: Primero leemos uno de los números de DAS con la instrucción RWS, para así conseguir almacenarlo en DAR. Cuando leamos el segundo número con una nueva instrucción RWS, el primer número es trasladado al registro ORB y el segundo se transfiere al registro DAR.

La siguiente instrucción es DAO, M (DD Arithmetic Operation with the parameter M). Esta instrucción transfiere el segundo número de DAR a ORA. El parámetro M, que indica el tipo de operación a realizarse, se transfiere al registro AFR. El resultado de la operación será conducido de ALU a DAR y el bit CARRY será colocado en el flip-flop RIR.

De esta manera la suma se realiza mediante una secuencia de instrucciones (programa) que comprende las siguientes instrucciones:

RWS,X
RWS,X
DAO,M

INSTRUCCION DE SALTO

La instrucción JUN, L implica que se realice un salto hasta otra instrucción cuya dirección está dada por el parámetro L.

La instrucción se ejecuta en forma muy simple colocando el parámetro L en el registro IAR. Ver el MICROPROGRAMA para esta instrucción. Puesto que el parámetro contiene 8 bits, el salto puede realizarse a cualquiera de las 256 diferentes direcciones, es decir, a cualquier sitio dentro de todo el almacén de programas.

La última instrucción en la lista, JCZ,L, es importante porque es una instrucción de toma de decisión. JCZ,L significa SALTO CONDICIONADO A CERO EN EL INDICADOR DE RESULTADO es decir, que se debe realizar un salto solo en el caso de que el resultado en el flip-flop RIR sea igual a 0; en el caso de que sea igual a 1, se debe ejecutar la siguiente instrucción avanzando un paso (+1) al registro de direcciones de instrucciones.

Así entonces con esta instrucción tendremos la posibilidad de escoger entre dos vías alternativas dentro del programa. La instrucción JCZ corresponde al símbolo de decisión en el diagrama de flujo:

Ejemplo: La supervisión de tiempo de una fase de conmutación, por ejemplo marcación de dígitos del abonado A. Tan pronto como nos lleque dígito, el procesador cuenta regresivamente el contenido de una palabra por cada intervalo primario. Esto lo hace la unidad ALU con la instrucción DAO. Esta instrucción es seguida por una JCZ. Si RIR = 0 el tiempo no ha sido excedido y se ejecutará la siguiente instrucción. Si, por otro lado, RIR = 1 se hará la desconexión hecha hasta el momento y se conectará tono de ocupado al abonado.

(El tiempo se determina al fijar la palabra procesada a un determinado valor n al iniciarse la espera del dígito. El tiempo de supervisión será n veces la longitud del intervalo primario).

5.2 DIAGRAMAS DE FLUJO

Con el fin de describir el trabajo del microprocesador, debemos elaborar los DIAGRAMAS DE FLUJO DETALLADOS correspondientes a cada función, como se observa en los diagramas siguientes.

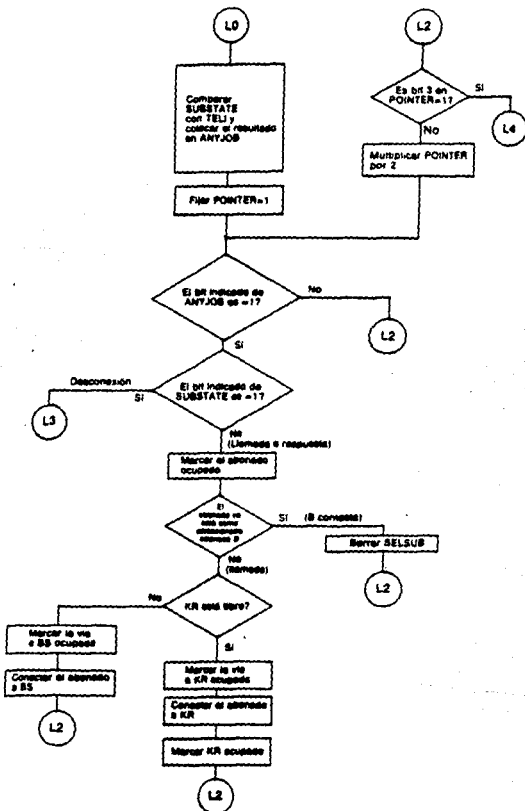


FIG. 5.2.1 DIAGRAMA DE FLUJO DE UNA LLAMADA NORMAL.

El siguiente diagrama de flujo, nos representa la condición de una llamada, cuando se efectúa la desconexión.

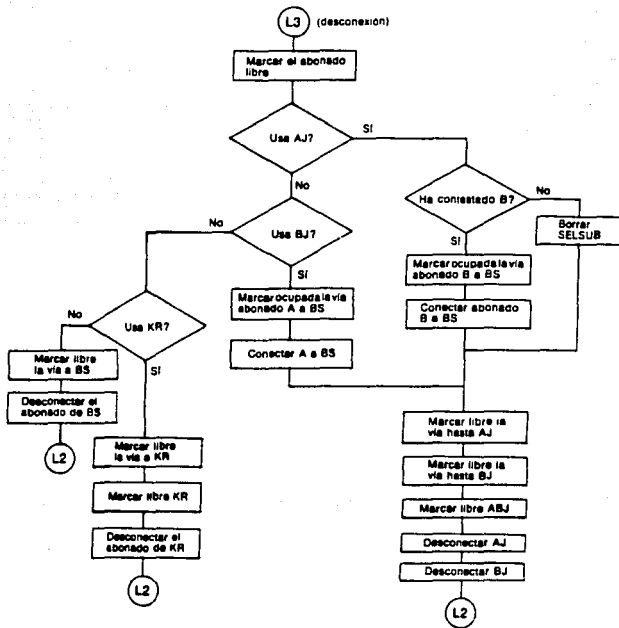


FIG. 5.2.2 DIAGRAMA DE FLUJO PARA REPRESENTAR LA DESCONEXIÓN.

Este diagrama de flujo, nos representa el análisis que realiza el microprocesador, cuando un abonado "A", ha marcado alguna cifra.

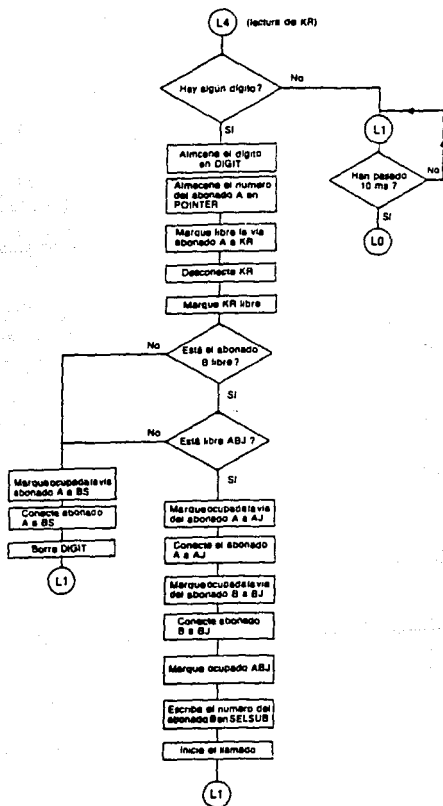


FIG. 5.2.3 ANALISIS DE CIFRAS QUE REALIZA EL MICROPROCESADOR

5.3 PROGRAMACION

Cuando se va a escribir un programa, este debe estar basado en un diagrama de flujo, el cual describe en forma general que hace el programa.

Los diagramas de flujo pueden variar desde poco detallados como en el caso de la fig 4.3.10 hasta extremadamente detallados, casi instrucción por instrucción. El programador escribe su programa el cual lleva a cabo lo que el diagrama de flujo define.

Durante el trabajo de programación el programador puede hallar conveniente cambiar el orden de ciertas secuencias, insertando ciertas funciones que el diagrama de flujo no muestra. Para simplificar este trabajo de planificación la forma de escritura del programa debe ser breve, pero clara y concisa. Debe ser sencillo visualizar el objetivo del programa.

Para nuestro MICROPROCESADOR, programaremos un código ensamblador con los nombres de las instrucciones, tales como RWS, WWS y DAO. La parte paramétrica de estas instrucciones es la más problemática.

Esta contiene direcciones en DAS, direcciones en PRS, constantes y códigos, las cuales son difíciles de determinar, direcciones de salto por ejemplo; más aún no son tan informativos para alguien que desea leer el programa. Consecuentemente escribiremos el programa en forma simbólica, es decir con nombres en lugar de valores.

En las instrucciones de transferencia RWS y WWS se da el nombre del dato en lugar de la dirección actual en DAS. Escribiremos el nombre de la instrucción el parámetro y después un comentario que describe lo realizado por la instrucción. Por ejemplo:

	Comentario
RWS, COUNTER	DAR:=COUNTER

El contenido de DAR se hace igual al contenido de la palabra COUNTER en DAS. (El símbolo:= significa "está hecho igual a").

WWS, COUNTER	COUNTER:=DAR
--------------	--------------

La palabra COUNTER en DAS se hace igual al de DAR.

En la instrucción DAO se da la función en lugar del parámetro M. Eso se hace según la tabla en la fig. 5.1.12 en código hexadecimal.

Los siguientes símbolos también son utilizados para efectuar operaciones como las siguientes:

DAD,09	DAR:=DAR+ORB
DAD,06	DAR:=DAR-ORB
DAD,00,	DAR:=DAR+1
DAD,0F	DAR:=DAR-1
DAD,1B	DAR:=DAR^ORB
DAD,1E	DAR:=DAR∨ORB
DAD,16	DAR:=DAR∧ORB
DAD,10	DAR:=DAR

Durante el trabajo de programación, será suficiente escribir el parámetro como una M. El valor no se ilustra particularmente, ya que puede insertarse luego.

Todos los valores de números en las instrucciones están dados en forma hexadecimal.

En las instrucciones JUN y JCZ la dirección del salto se da simbólicamente por L seguida de un dígito, por ejemplo L5. la instrucción hasta donde se realiza el salto se marca en el programa con la misma designación L5.

Antes que un programa sea escrito debemos conocer los datos que hay en DAS y donde se encuentran. Debemos hacer una estructura de datos. En nuestra pequeña central debemos tener, por ejemplo, un bit para cada abonado, para indicar si un abonado está libre u ocupado. Si el bit es igual a "0" el abonado respectivo está libre y si es igual al "1" el abonado está ocupado.

Para cuatro abonados necesitamos 4 de tales bits. Situaremos a estos en una misma palabra en los bits de posiciones de 0 a 3, los que darán una palabra que llamaremos SUBSTATE (Estado del Abonado). Esto se ilustra en la figura 5.3.1

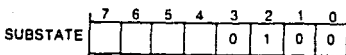


FIG. 5.3.1 ASIGNACION DE LOS ABONADOS EN SUBSTATE.

Así tendremos que en el ejemplo el abonado 2 está ocupado y el resto se encuentra libre. Si deseamos leer esta palabra, debemos escribir:

RWS,SUBSTATE

DAR:=SUBSTATE

Cuando se ha escrito completamente el programa, se debe traducir a código de máquina (código binario) dándose la dirección real del dato. Si la palabra SUBSTATE está en la dirección 0B de DAS, la instrucción anterior será:

RWS,0B

DAR:=SUBSTATE

Trasladada a código binario será:

0000	1000	0000
0	B	RWS

La siguiente tabla que se muestra en la figura 5.3.2 representa la secuencia de instrucciones que ocurren frecuentemente en las tareas que desarrolla el microprocesador.

TABLA DE REFERENCIA DE LAS SECUENCIAS DE INSTRUCCIONES QUE OCURREN FRECUENTEMENTE.

<i>Operaciones con bits</i> Función	Instrucción	Comentarios	Resultado en DAR
Fijar el bit 2 a uno en la palabra X	WWC.04 RWS,X DAO.1E	DA:=04 DAR:=X (ORB:=04) DAR:=DAR VORB	0000 0100 xxxx xxxx xxxx x1xx
Fijar el bit 3 a cero en la palabra X	WWC.F7 RWS,X DAO.1B	DAR:=F7 DAR:=X DAR:=DAR ^ORB	1111 0111 xxxx xxxx xxxx 0xxx
Fijar a 1 el bit en la palabra X que sea dado por un uno en la palabra POINTER	RWS.POINTER RWS,X DAO.1E	DAR:=POINTER DAR:=X DAR:=DAR VORB	0000 0010 xxxx xxxx xxxx xx1x
Fijar a cero el bit en la palabra X que sea dado por un uno en la palabra POINTER	RWS.POINTER DAO.10 RWS,X DAO.1B	DA:=POINTER DAR:=DAR DAR:=X DAR:=DAR ^ORB	0000 0010 1111 1101 xxxx xxxx xxxx xx0x
<i>Comparaciones</i> Función	Instrucción	Comentarios	Resultado en DAR
¿Qué valor tiene el bit 3 en la palabra X?	WWC.08 RWS,X DAO.1B JCZ.Ln	DAR:=8 DAR:=X DAR:=DAR ^ORB Si bit 3=0 ir a Ln	0000 1000 xxxx xxxx 0000 x000
¿La palabra COUNTER contiene el valor 2D?	WWC.2B RWS, COUNTER DAO.16 JCZ.Ln	DAR:=2B DAR=COUNTER DAR:=DAR VORB Si igual, ir a Ln	0010 1011 0010 1010 0000 0001

FIG. 5.3.2 TABLA DE LAS SECUENCIAS DE INSTRUCCIONES QUE OCURREN FRECUENTEMENTE.

INTERCONEXION DEL MICROPROCESADOR Y LA PARTE DE CONMUTACION

Para que el MICROPROCESADOR pueda interconectarse con la parte de conmutación, este requiere de circuitos para el direccionamiento de las diferentes unidades telefónicas. Además se requiere de otros dos tipos de instrucción. El control se realiza vía el bus de datos y los hilos de lectura y escritura.

La FIG. 5.3.3 muestra la parte de conmutación y la parte de interface entre la primera y el microprocesador.

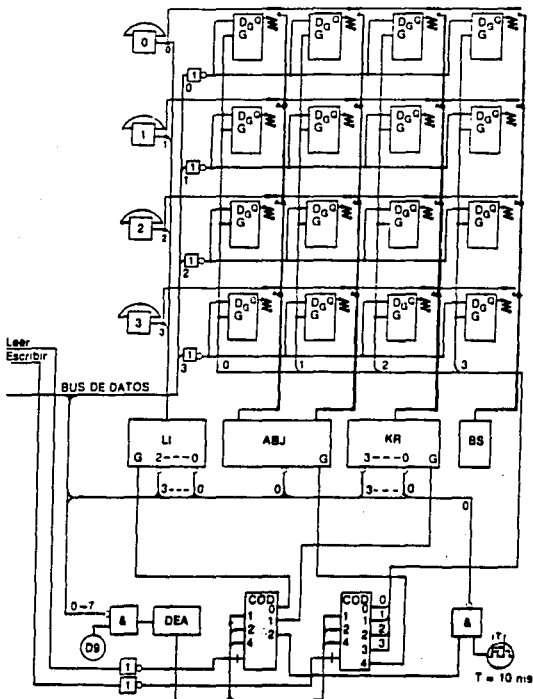


FIG. 5.3.3 LA PARTE DE CONTROL CONECTADA A LA PERIFERIA.

La parte de INTERFACE consta de los FLIP-FLOPS en cada punto de cruce, un registro de direcciones DEA, dos decodificadores de direcciones COD y varios circuitos de compuertas.

Algunos dispositivos telefónicos contienen también compuertas que, desde el punto de vista de funcionamiento, actualmente pertenecen a la parte de interface, pero que por razones prácticas, se incluye en dispositivos que pertenecen a la parte de conmutación. (Esto ocurre generalmente aún en sistemas grandes).

En la fig. 5.3.3 podemos ver como se conecta el bus de datos con los hilos de lectura y escritura desde el procesador hasta el equipo interface.

El direccionamiento lo inicia el procesador enviando la dirección de órgano hasta el registro DEA, Device Address Register (Registro de direcciones de dispositivos). Esta dirección se decodifica en uno de los dos circuitos COD.

Uno de ellos se activa con la orden de lectura y el otro con la de escritura.

Las salidas de éstos activan las compuertas de entrada de los diferentes dispositivos telefónicos, por medio de las cuales, los datos se llevan dentro o fuera de los dispositivos vía el bus de datos.

Para esta interconexión con los dispositivos telefónicos necesitamos las instrucciones RWD,D (Read Word From Device = leer una palabra del dispositivo) y WWD,D (Write Word in Device = escribir una palabra en el dispositivo).

El parámetro D indica la dirección del dispositivo. No se debe confundir esto con las señales de control, D1 - D15, en el micro-programa.

La instrucción RWD,D trabaja de la forma siguiente: Se transfiere primero el parámetro D hasta el registro DEA.

Luego se envía la orden de lectura y con ayuda de la señal T (bit 7) en el micro-programa, se guía ésta hacia el equipo telefónico.

Esta abre el circuito COD, cuya salida abre las compuertas de datos del dispositivo telefónico hacia el bus de datos. Los datos de respuesta son conducidos por el bus hasta el registro DAR en el procesador.

La instrucción WWD,D escribe en los dispositivos telefónicos, es decir hace operar un relé.

La instrucción transfiere el dato almacenado en DAR hasta el dispositivo telefónico cuya dirección está dada por el parámetro D.

El micro-programa para esta instrucción se muestra en la figura 5.3.4

El valor del parámetro D, que indica la dirección de los dispositivos telefónicos, se da en la fig. 5.3.3

A continuación se da una tabla con más detalles al respecto.

<u>D PALABRA DE PRUEBA</u>	<u>NOMBRE</u>
0 Interface de línea	TELI
1 Receptor de código KR	TEKR
2 Intervalo primario (=10 ms)TEPI	

<u>D PALABRA DE OPERACION</u>	<u>NOMBRE</u>
0 Operación de contacto AJ	SOAJ
1 Operación de contacto BJ	SOBJ
2 Operación de contacto KR	SOKR
3 Operación de contacto BS	SDBS
4 Inicio de señal de llamada	STRI

Instrucción ensamblador	Cód.	Dirección en MPS	Bit número																
			C	B	A	9	8	7	6	5	4	3	2	1	0				
RWD, D Read Word from Device	6	1B											1	1			1		
		1A	1										1	1		1	1		
		1B	1														1	1	
		1C												1	1			1	
WWD, D Write Word in Device	7	1D		1	1		1	1	1	1									
		1E				1	1	1	1	1									
		1F	1																
		Peso			1	1	1	1	1	1	4	2	1	8	4	2	1		
SEÑALES DE CONTROL			Nombre			E	+1	J	W	R	T	S				D			

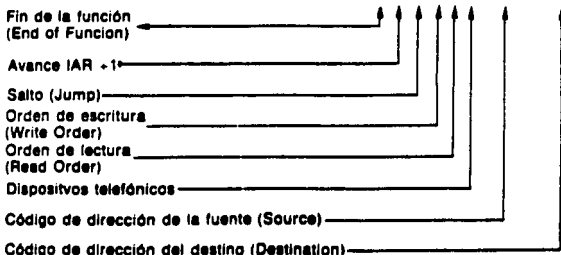


FIG. 5.3.4 EL MICRO-PROGRAMA DE OPERACION.

EL PRINCIPIO DEL TRABAJO DEL MICROPROCESADOR

Tenemos ahora todos los circuitos (hardware) de los dispositivos que se requieren para que nuestro pequeño sistema trabaje.

Solamente nos resta determinar la estructura de los datos en DAS y del programa mismo.

Debemos primero decidir el principio del funcionamiento del procesador.

El procesador investigará en forma cíclica los puntos de prueba en la parte de conmutación con el objeto de detectar señales externas, es decir, cambios de estado en los abonados.

De esta manera el procesador encuentra si hay algún trabajo a realizar.

Via la unidad LI el procesador detecta cuando un abonado llama, descuelga su microteléfono o contesta, y via KR detecta si ha recibido o no dígitos.

Cuando el procesador ha ido hasta todos los puntos de prueba y ha realizado todo el trabajo que fué detectado, podrá empezar nuevamente desde el principio.

Decidimos que la exploración debe ser iniciada cada 10 milisegundos.

El procesador trabaja rápidamente y por lo tanto dentro de este intervalo realiza todo el trabajo que sea necesario, aún si los cuatro abonados han cambiado el estado que tenían en el intervalo inmediatamente anterior.

Por otro lado este periodo es tan corto que no existe la posibilidad de que el procesador pierda alguna señal proveniente de los abonados.

En el equipo de conmutación tenemos un generador de pulsos que da un pulso corto cada 10 milisegundos. Mediante prueba repetitiva el punto de prueba correspondiente, el procesador puede decidir cuando debe empezar la exploración nuevamente.

Este trabajo es realizado por el procesador cada 10 milisegundos.

El intervalo de exploración se llama INTERVALO PRIMARIO. Debido a que éste es fijo, el procesador puede llevar a cabo medidas de tiempo para medir la duración de las señales o hacer supervisión de tiempo etc.

En la fig. 5.3.5 se muestra el diagrama de flujo general que describe el trabajo del microprocesador.

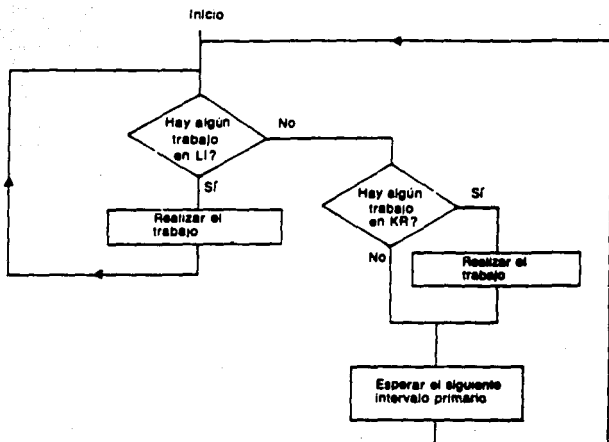


FIG. 5.3.5 DIAGRAMA DE FLUJO QUE REPRESENTA EL TRABAJO DEL MICROPROCESADOR

El microprocesador primero investiga los LI mediante lectura, uno por uno, para ver si detecta llamadas, señal de finalización o señal de respuesta del abonado B.

Al encontrar una de estas señales el procesador realiza el trabajo exigido.

Después, el procesador sigue al próximo LI. Cuando los cuatro abonados han sido investigados, el procesador verifica si KR ha recibido un dígito. Cuando se ha realizado este trabajo sólo le resta esperar hasta que se inicie el siguiente intervalo primario.

Cuando el procesador lee las señales de descuelgue del abonado, vía LI, debe interpretar un cambio de "cero" a "uno" como una llamada (o B contesta) y un cambio de "uno" a "cero" como una señal de desconexión (el abonado repone su microteléfono).

Si el estado del abonado no cambia no será necesaria ninguna acción.

Con objeto de que el procesador pueda conocer si un estado ha cambiado, el resultado de la prueba de LI debe ser comparado con el resultado de la prueba anterior. Por consiguiente el procesador debe recordar siempre el resultado de la prueba anterior.

El resultado de la prueba anterior es almacenado en el almacén de datos en una palabra que ya hemos encontrado antes, llamada SUBSTATE (Estado del Abonado). Comparando SUBSTATE con la nueva palabra de prueba de LI, el procesador puede determinar si ha ocurrido cambio o no, e interpretar el significado del cambio.

En esta forma el procesador puede mantener una verificación de lo que se debe hacer. La comparación puede describirse con la tabla siguiente.

Resultado de una nueva prueba de LI

		1	0	
Estado del abonado anterior de acuerdo con la palabra SUBSTATE en DAS	Colgado=0	Y	X	X=Ningún cambio Y=Nueva llamada o B contesta Z=Desconexión
	Descolgado=1	X	Z	

El procesador compara las dos palabras bit por bit. Cuando son iguales, no hay ningún trabajo para realizar, cuando existe diferencia se debe hacer algún tipo de trabajo.

La acción requerida puede determinarse del valor de SUBSTATE

La comparación se hace muy fácilmente por medio de la instrucción DAD, la cual realiza la operación "OR EXCLUSIVO".

Esto puede hacerse simultáneamente para los cuatro abonados.

El resultado de abajo muestra solamente los bits 0 a 3. El resto de los bits no son de interés por existir en nuestra central sólo 4 abonados.

Ejemplo:	Resultado de la prueba de LI	0101
	<u>SUBSTATE</u>	1100
	Cambios	1001

El resultado de la ejecución de DAD muestra que se han producido cambios en los estados de los abonados 0 y 3.

El procesador debe ahora tratar con los bits individuales en el resultado de comparación. Con objeto de encontrar el bit que debe ser tratado (en otras palabras que abonado) usamos una palabra la cual hemos llamado POINTER (puntero). En esta palabra marcaremos con "uno" el bit que debe ser procesado. Moviendo el bit en pasos hacia la izquierda podemos recorrer las posiciones de bit 0 a 3, las que corresponden a los abonados 0 a 3:

Número de abonado	POINTER
0	0001
1	0010
2	0100
3	1000

Este movimiento lateral de uno, puede hacerse empezando con la constante 1, y luego doblando ésta para cada paso, mediante la suma del mismo número.

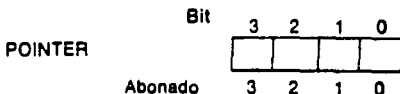
Así por medio de POINTER podemos indicar, en diferentes palabras de datos, el bit que pertenece al abonado que va a ser tratado en el momento.

La estructura de datos

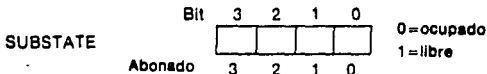
Podemos ahora preparar la estructura de datos para nuestra pequeña central. Hasta donde sea posible haremos corresponder a los abonados 0 a 3 a los bits 0 a 3.

Por esta razón usaremos solamente los bits 0 a 3 de todas las palabras.

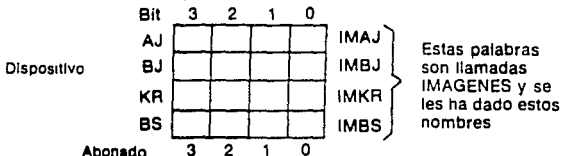
Como mencionamos anteriormente, necesitamos una palabra llamada POINTER para indicar el abonado que debe ser tratado en un determinado momento. Para cuatro abonados necesitaremos cuatro bits.



Para recordar el estado anterior del contacto de horquilla de cada abonado se requiere un bit por abonado. Microteléfono colgado=0 y descolgado=1. Esta palabra se llama SUBSTATE.



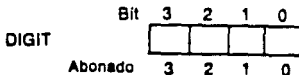
Cada punto de conmutación en la etapa de selección requiere de un bit para indicar si el punto está libre (=0) u ocupado (=1). Así los 16 puntos de conmutación requieren de 16 bits. Arreglaremos estos 16 bits en cuatro palabras con cuatro bits en cada una de manera que las posiciones de bit correspondan a los números de abonados.



Es necesario un bit por cada dispositivo telefonico con objeto de indicar si está libre (=0) u ocupado (=1). Para BS no se requiere de bits, ya que varios abonados pueden usarlo en forma paralela. A estas palabras las llamaremos DEVSTATE (Device State = estado de dispositivo).



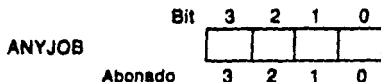
Para almacenar el dígito recibido en KR se requiere una palabra de 4 bits, a la que llamamos DIGIT y haremos que sus posiciones 0 a 3 por lo tanto un "1" en una de las posiciones de bit dé el abonado seleccionado.



El procesador debe tomar ciertas medidas en caso de que el estado de un abonado cambie. El cambio de libre a ocupado significa llamada (o B contesta), mientras que un cambio de ocupado a libre significa "DESCONEXION".

Los cambios se detectan por medio de una comparación entre la respuesta de prueba de LI (=estado presente) y el contenido de la palabra SUBSTATE (estado anterior).

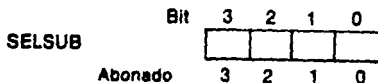
Esta comparación puede hacerse simultáneamente a los cuatro abonados. El resultado de esta comparación se situará en la palabra ANYJOB (algún trabajo), en la cual un "1" indicará alguna labor (un cambio). Los bits de posición de la palabra ANYJOB corresponden a los abonados.



Para un abonado un cambio de libre a ocupado significa que el abonado ha levantado su microteléfono para hacer una llamada (Abonado A) o para recibir una llamada (Abonado B).

El procesador debe distinguir entre estos dos casos, y para este propósito necesitamos una palabra SELSUB (Abonado seleccionado) que indique el abonado que ha sido seleccionado (Abonado B) y que se ha conectado a BJ, y recibe la señal de timbre.

Cuando el abonado descuelga, el procesador controlará si el bit correspondiente al abonado en SELSUB está = 1. Si es así, el procesador interpretará que el abonado contesta una llamada en lugar de originar una nueva.



Las palabras de los datos mencionados anteriormente están situados en el almacén de datos en las direcciones 0 a 9. Ver la figura 5.3.6

Función	Nombre	Bit				Dirección
		3	2	1	0	
0=libre 1=ocupado	Imagen de operación. Palabra 0					0
	Imagen de operación. Palabra 1					1
	Imagen de operación. Palabra 2					2
	Imagen de operación. Palabra 3					3
	Estado de abonado					4
						5
	Abonado seleccionado					6
						7
						8
(0=libre 1=ocupado)				KR	ABJ	9

A

FIG. 5.3.6 ESTRUCTURA DE DATOS EN DAS.

Los bits 0 a 3 en la palabra corresponden a los abonados 0 a 3. Esta palabra da una imagen del estado de estos relés y por eso se le denomina IMAGEN de la palabra de operación. La palabra 1 es la imagen de la conexión al lado B de ABJ, la palabra 2 será la imagen de la conexión al KR y la 3 da la imagen del estado de los relés para conexión a BS. La palabra 4, POINTER, se usa para señalar el abonado que debe ser tratado en el momento.

La palabra 5, SUBSTATE, se utiliza para almacenamiento del estado previo de los abonados.

La palabra 6, ANYJOB, es el resultado de la comparación de SUBSTATE y la palabra de prueba de LI.

La palabra 7, DIGIT, indica el dígito que ha sido recibido en KR.

La palabra 8, SELSUB, indica el abonado que ha sido seleccionado y por lo tanto conectado a la parte B de ABJ. Antes de contestar el abonado está marcado como libre en SUBSTATE, cuando contesta, no debe interpretarse como una nueva llamada sino como respuesta a una llamada.

La palabra 9 DEVSTATE, indica si ABJ o KR están ocupados BS no se indicará ya que a él pueden estar conectados varios abonados a la vez.

En las palabras 0-8, los bits corresponden a los cuatro abonados.

EL PROGRAMA

Con ayuda de estas palabras podemos ahora escribir un programa que controle la central para que los cuatro abonados puedan llamar.

La fig. 5.2.1 muestra la primera parte. El programa se inicia cada diez milisegundos (cada intervalo primario) en la posición marcada L0.

Inicialmente el estado de los abonados es comparado con su estado previo. El resultado de esta comparación (llamado ANYJOB) debe ser investigado bit por bit con ayuda de la palabra POINTER.

Se inicia con POINTER=1 por medio del cual se investiga el bit indicado por POINTER, el cual para este valor será el bit de posición 0 en la palabra ANYJOB. Si este bit es igual a 0, el estado del abonado no ha sido cambiado.

Como resultado de lo anterior, continuaremos a la derecha del diagrama del flujo hasta el círculo que contiene una referencia para la posición L2, el que se encuentra en la misma figura.

De esta manera hemos hecho un salto en el diagrama hasta el punto de entrada llamado L2, el cual se encuentra en la parte superior a la derecha de la figura.

El programa continúa a partir de aquí con una verificación para conocer si el bit de POINTER ha alcanzado la posición 3, en otras palabras, si todos los abonados han sido investigados. (La ramificación L2 es en efecto común para todos los abonados menos para el primero).

El 1 en POINTER es avanzado por adición de la misma palabra (es lo mismo que duplicarlo cada vez). La anterior secuencia se repite para investigar a todos los abonados.

Si encontramos un 1 en ANYJOB investigamos luego a SUBSTATE con objeto de encontrar el tipo de trabajo que debe realizarse.

En esta forma podemos leer del diagrama de flujo como el procesador maneja las diferentes situaciones de cambios.

Escribiremos ahora una parte del programa y empezaremos con la parte inicial del diagrama de flujo en la fig. 5.2.1. El primer rectángulo "comparar SUBSTATE con TELI y colocar el resultado en ANYJOB" puede escribirse de la manera siguiente:

L0	RWD,0	DAR:=TELI
	RWS,5	DAR:=SUBSTATE
	DA0,16	DAR:=DAR V ORB
	WWS=6	ANYJOB:=DAR

El próximo rectángulo "Fijar POINTER = 1" se escribe así:

WVC,1	DAR:=1
WWS,4	POINTER:=DAR

Después, debemos investigar el bit en el resultado de la comparación ANYJOB el cual está indicado por POINTER. Si el bit=0 no le ha sucedido nada desde la última verificación, y podremos continuar con el bit siguiente.

Si en cambio, el bit = 1 hay un trabajo que hacer. De esta forma tendremos dos ramificaciones en el programa. El programa que corresponde al símbolo de decisión "El bit indicado de ANYJOB es = 1 ?" se puede escribir como:

L5	RWS,4	DAR:=POINTER
	RWS,6	DAR:=ANYJOB
	DAD,1B	DAR:=DAR^ORB
	JCZ,L2	Si ANYJOB=0 ir a L2

Entonces, si no hay trabajo que realizar, saltaremos a la dirección del programa marcada como L2. Aún no conocemos la dirección real de ésta. Debemos por tanto, usar este tipo de símbolo. A tales direcciones se las denomina L (label=etiqueta) seguidas de un número. (La razón de por qué a la primera dirección se le ha dado la designación L5 la tendremos luego).

Si, de otra manera hay un trabajo, éste será realizado. El programa entonces continúa a la siguiente dirección luego de la instrucción JCZ. De acuerdo al diagrama de flujo, el bit de abonado en SUBSTATE se investigará ahí.

Pero por el momento seguiremos el salto a la ramificación L2, que empieza en la parte superior a la derecha del diagrama con la pregunta "El bit 3 de POINTER es igual a 1?" Si es así hemos investigado ya todos los abonados.

El diagrama de flujo se ejecuta de tal forma que empezemos con L0 para el primer abonado y luego pasemos a L2 para cada nuevo abonado. Cada vez que pasemos L2 el 1 en POINTER es movido un paso a la izquierda, terminando en el bit de posición 3 luego de investigar los cuatro abonados. Cuando empezemos una vez más la ramificación L2, el procesador podrá guiar el programa hasta la ramificación L4.

Esta parte del programa puede escribirse como sigue:

L2	WVC,8	DAR:=0000 1000
	RWS,4	DAR:=POINTER
	DAO,06	DAR:=DAR-ORB
	JCZ,L4	si POINTER 7 IR a L4

Si todos los abonados aún no han sido tratados, POINTER, debe tener uno de los valores 1, 2, o 4. De esta forma el resultado de la resta tendrá "underflow", RIR=1 y continuaremos a la siguiente instrucción en lugar de realizar el salto a L4. Aquí debemos correr el 1 de POINTER un paso a la izquierda. Esto se hace multiplicando POINTER por 2.

RWS,4	DAR:=POINTER
RWS,4	DAR:=POINTER
DA0,09	DAR:=DAR+ORB
WWS,4	POINTER:=DAR

Lo que hemos hecho es sumar a POINTER el mismo que es equivalente a multiplicarlo por 2.

De acuerdo al diagrama de flujo de la fig. 5.2.1 debemos ahora saltar de nuevo a "El bit indicado de ANYJOB es =1?" el que ya hemos pasado una vez y el cual ya hemos programado.

Dimos a la primera instrucción en la secuencia la designación L5. De esta forma podremos ahora salir ahí con:

JUN,L5	Ir a L5
--------	---------

En esta forma describiremos el programa completo, sección por sección. Luego, pondremos todas las secciones en secuencia.

De acuerdo con esto podemos también determinar las direcciones reales para las instrucciones. Entonces a la dirección simbólica de salto, L5, se da la dirección real, 6.

Antes de situar el programa en el almacén de programas, éste debe convertirse a código binario.

Las primeras nueve instrucciones en el programa se dan a continuación en la forma como aparecen en código binario.

Dirección	Código de máquina
00	0000 0000 0110
01	0000 0101 0000
02	0001 0110 0711
03	0000 0110 0001
04	0000 0001 0010
05	0000 0100 0001
06	0000 0100 0000
07	0000 0110 0000
08	0001 1011 0011

Para más claridad, en la figura 5.3.6 mostramos la parte completa del programa que hemos preparado hasta aquí.

Para mostrar un ejemplo del manejo del programa hemos marcado con números lo que sucede en un intervalo primario cuando el abonado dos ha llamado, mientras el estado de los otros abonados ha permanecido sin cambios desde el intervalo inmediatamente anterior.

1 - 4 Escritura del trabajo en ANYJOB
5 - 6 El abonado relacionado=abonado 0 (POINTER=0001)
7 - 10 Existe algún trabajo para este abonado?
11- 14 No. Han sido investigados los abonados?
15- 18 No. POINTER será movido al siguiente abonado
(Abonado 1).
19 Salto de regreso
20- 23 Existe algún trabajo para el abonado relacionado?
24- 27 No. Todos los abonados han sido investigados?
28- 31 No. POINTER será movido al siguiente abonado.
(Abonado 2).
32 Salto de regreso
33- 36 Hay algún trabajo para el abonado relacionado?
37- n Sí. El trabajo se lleva a cabo
n+1-n+4 Todos los abonados han sido investigados?
n+5-n+8 No. POINTER será movido al siguiente abonado.
(Abonado 3).
n+9 Salto de regreso.
n+10-n+13 Hay algún trabajo para este abonado?
n+14-n+17 No. Todos los abonados han sido investigados?
n+18-m Sí. El programa se ocupa ahora de otras cosas hasta
que los 10 milisegundos hayan pasado, después de lo
cual se inicia el nuevo ciclo desde la dirección.

L0	RWD,0	DAR:=TELI
	RWS,5	DAR:=SUBSTATE
	DAO,16	DAR:=DAR \vee ORB
	WWS,6	ANYJOB:=DAR
	WWC,1	DAR:=1
	WWS,4	POINTER:=DAR
L5	RWS,4	DAR:=POINTER
	RWS,6	DAR:=ANYJOB
	DAO,1B	DAR:=DAR \wedge ORB
	JCZ,L2	IF ANYJOB=0 go to L2
		Si hay trabajo a rea-
		lizar, este se reali-
		zará aqui. No hemos
		codificado aun esta
		parte del flujograma.
L2	WWC,8	DAR:=0000 1000
	RWS,4	DAR:=POINTER
	DAO,06	DAR:=DAR - ORB
	JCZ,L4	IF POINTER > 7 go to L4
	RWS,4	DAR:=POINTER
	RWS,4	DAR:=POINTER
	DAO,09	DAR:=DAR + ORB
	WWS,4	POINTER:=DAR
	JUN,L5	Go to L5
L4		La secuencia que va
		a investigar si ha
		llegado un dígito a
		KR o no.

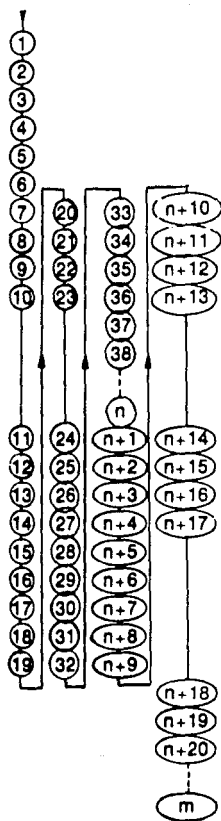


FIG. 5.3.6 PROGRAMA EN LENGUAJE ENSAMBLADOR.

RUTINA DE INICIO

Cuando iniciemos por primera vez la pequeña central, podemos encontrar problemas si el almacén de datos no está borrado y los flip-flops despejados.

Si, por ejemplo, un órgano o un relé de conexión aparece marcado como ocupado, éste no podría ser seleccionado ni será posible marcarlo como libre.

Con objeto de obtener una posición de arranque definida, introducimos una rutina de inicio la que empieza con una llave manual. La llave despeja el registro IAR, el que dará la dirección de instrucción No. 0 para el almacén de programas. Esto se puede observar en la figura 5.3.7

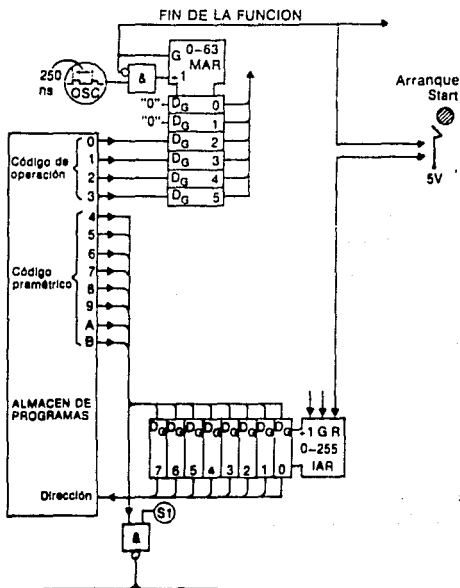


FIG. 5.3.7 DIAGRAMA PARA REPRESENTAR LA RUTINA DE INICIO.

La misma llave también abre las entradas del registro MAR para recibir la primera instrucción. Cuando se desopere la llave, el procesador empezará a ejecutar el programa desde la dirección 0, donde inicia el programa de arranque del sistema. En éste se "limpian" todas las palabras del almacén de datos y se despejan todos los flip-flops de la parte de conmutación.

Cuando se lleva a cabo lo anterior, hay un salto al programa de telefonía y la central será puesta en operación (ver figura 5.3.8).

La rutina de inicio quizá sea necesaria en caso de ocurrir alguna falla en el sistema para lograr su restablecimiento.

La rutina de inicio borra los datos y despeja los flip-flops después de lo cual el manejo del tráfico empieza desde la posición 0. Esto significa que las conexiones en proceso de establecimiento y las que ya se han establecido todas serán interrumpidas.

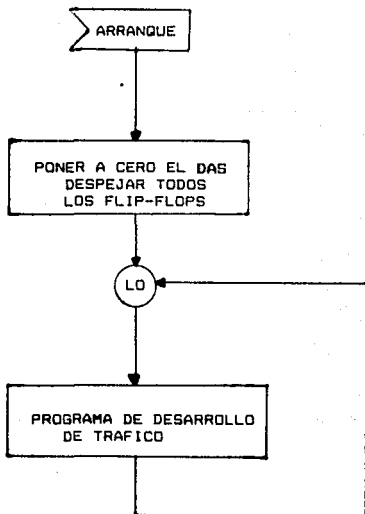


FIG. 5.3.8 DIAGRAMA PARA REPRESENTAR LA RUTINA DE INICIO.

Texto del diagrama de flujo	Dirección	Instrucción	Comentario
	Abso- luta	simbó- lica	
	00	WVC,0	DAR:=0
	01	WWS,0	IMAJ:=DAR
	02	WWS,1	IMBJ:=DAR
	03	WWS,2	IMKR:=DAR
	04	WWS,3	IMBS:=DAR
	05	WWS,4	POINTER:=DAR
	06	WWS,5	SUBSTATE:=DAR
Rutina de arranque, borra todas las palabras	07	WWS,6	ANYJOB:=DAR
	08	WWS,7	DIGIT:=DAR
	09	WWS,8	SELSUB:=DAR
	0A	WWS,9	DEVSTATE:=DAR
	0B	WWD,0	SOAJ:=DAR
	0C	WWD,1	SOBJ:=DAR
	0D	WWD,2	SOKR:=DAR
	0E	WWD,3	SOBS:=DAR
	0F	WWD,4	STRI:=DAR
Comparación SUBSTATE con TELI	10	L0 RWD,0	DAR:=TELI
	11	RWS,5	DAR:=SUBSTATE
	12	DAO,16	DAR:=DAR ^w ORB
	13	WWS,6	ANYJOB:=DAR
Fijar POINTER=1	14	WVC,1	DAR:=1
	15	WWS,4	POINTER:=DAR
El bit indicado en ANYJOB es=1?	16	L5 RWS,4	DAR:=POINTER
	17	RWS,6	DAR:=ANYJOB
	18	DAO,1B	DAR:=DAR ^w ORB
	19	JCZ,L2	Si ANYJOB=0 ir a L2
	1A	DAO,1A	DAR:=POINTER
El bit indicado en SUBSTATE es=1?	1B	RWS,5	DAR:=SUBSTATE
	1C	DAO,1B	DAR:=DAR ^w ORB
	1D	JCZ,L6	Si SUBSTATE=0 ir a L6
	1E	L3 RWS,4	DAR:=POINTER
	1F	DAO,10	DAR:=DAR
Marcar abonado libre	20	RWS,5	DAR:=SUBSTATE
	21	DAO,1B	DAR:=DAR ^w ORB
	22	WWS,5	SUBSTATE:=DAR
Utilizó AJ?	23	RWS,4	DAR:=POINTER
	24	RWS,0	DAR:=IMAJ
	25	DAO,16	DAR:=DAR ^w ORB
	26	JCZ,L7	Si AJ fué usado ir a L7
	27	RWS,4	DAR:=POINTER
Utilizó BJ?	28	RWS,1	DAR:=IMBJ
	29	DAO,16	DAR:=DAR ^w ORB
	2A	JCZ,L8	Si BJ fué usado ir a L8
	2B	RWS,4	DAR:=POINTER
Utilizó KR?	2C	RWS,2	DAR:=IMKR
	2D	DAO,16	DAR:=DAR ^w ORB
	2E	JCZ,L9	Si KR fué usado ir a L9
	2F	RWS,4	DAR:=POINTER
Marcar libre la vía a BS	30	DAO,10	DAR:=DAR
	31	RWS,3	DAR:=IMBS
	32	DAO,1B	DAR:=DAR ^w ORB
	33	WWS,3	IMBS:=DAR
Desconexión del abo- nado desde BS	34	WWD,3	SOBS:=DAR

El biten 3 POINTER es=1?	35	L2	WWC,8	DAR:=8 (0000 1000)
	36		RWS,4	DAR:=POINTER
	37		DAO,06	DAR:=DAR-ORB
	38		JCZ,L4	Si POINTER > 7 ir a L4
Multiplicar POINTER por 2	39	RWS,4	DAR:=POINTER	
	5A	RWS,4	DAR:=POINTER	
	5B	DAO,09	DAR:=DAR+ORB	
	3C	WWS,4	POINTER:=DAR	
Marcar libre la via a KR	3D	J 'N,L5	Ir a L5	
	3E	L9	WWC,0	DAR:=0
	5F		WWS,2	IMKR:=DAR
	40		WWD,2	SOKR:=DAR
41	WWC,FD		DAR:=FD (1111 1101)	
Desconectar KR	42	RWS,9	DAR:=DEVSTATE	
	43	DAO,1B	DAR:=DAR-ORB	
	44	WWS,9	DEVSTATE:=	
	45	JUN,L2	Ir a L2	
Marcar ocupada la via de abonado A hasta BS	46	L8	RWS,4	DAR:=IMAJ
	47		RWS,5	DAR:=IMBS
	48		DAO,1E	DAR:=DAR-ORB
	49		WWS,3	IMBS:=DAR
Conectar el abonado A a BS.	4A	WWD,3	SOBS:=DAR	
	4B	L10	WWC,0	DAR:=0
			WWS,0	IMAJ:=DAR
	4C		WWS,1	IMBJ:=DAR
4D	WWC,FE		DAR:=FE(1111 1110)	
Marcar libre la via a BJ	4E	RWS,9	DAR:=DEVSTATE	
	4F	DAO,1B	DAR:=DAR-ORB	
	50	WWS,9	DEVSTATE:=DAR	
	51	WWC,0	DAR:=0	
Desconectar AJ y BJ	52	WWD,0	SOAJ:=DAR	
	53	WWD,1	SOBJ:=DAR	
	54	JUN,L2	Ir a L2	
	55	RWS,1	DAR:=IMBJ	
Ha contestado el abonado B?	56	L7	RWS,5	DAR:=SUBSTATE
	57		DAO,1B	DAR:=DAR-ORB
	58		JCZ,L11	Si SUBSTATE=0 ir a L11
	59		RWS,1	DAR:=IMBJ
Marcar ocupada la via BS al abonado B	5A	RWS,3	DAR:=IMBS	
	5B	DAO,1E	DAR:=DAR-ORB	
	5C	WWS,3	IMBS:=DAR	
	5D	WWD,3	SOBS:=DAR	
Conectar abonado B a BS	5E	JUN,L10	Ir a L10	
	5F	WWC,0	DAR:=0	
	60	WWS,8	SELSUB:=DAR	
	61	JUN,L10	Ir a L10	
Borrar SELSUB	62	L6	RWS,4	DAR:=POINTER
	63		RWS,5	DAR:=SUBSTATE
	64		DAO,1E	DAR:=DAR-ORB
	65		WWS,5	SUBSTATE:=DAR
Marcar ocupado al abonado	66	RWS,4	DAR:=POINTER	
	67	RWS,8	DAR:=SELSUB	
	68	DAO,16	DAR:=DAR-ORB	
	69	JCZ,L13	Si está seleccionado, ir a L13	
Está el abonado selec- cionado como abonado B?	6A			

Está libre KR?	6B	WVC,02	DAR:=02(0000 0010)
		RWS,9	DAR:=DEVSTATE
		DAO,1B	DAR:=DAR^ORB
		JCZ,L12	Si KR está libre, ir a L12
Marque ocupada la vía a BS	6E	RWS,4	DAR:=POINTER
		RWS,3	DAR:=IMBS
		DAO,1E	DAR:=DAR^ORB
		WWS,3	IMBS:=DAR
Conecte el abonado a BS	70	WWD,3	SOBS:=DAR
		71	
Marque ocupada la vía a KR	72	JUN,L2	Ir a L12
		RWS,4	DAR:=POINTER
Conecte el abonado a KR	75	WWS,2	IMKR:=DAR
		76	SOKR:=DAR
Marque KR ocupado	77	WWD,2	
		WVC,02	DAR:=02(0000 0010)
		RWS,9	DAR:=DEVSTATE
		DAO,1E	DAR:=DAR^ORB
Borre SELSUB	7B	WWS,9	DEVSTATE:=DAR
		JUN,L2	Ir a L12
		WVC,0	DAR:=0
		WWS,8	SELSUB:=DAR
Hay algún dígito?	7C	JUN,L2	Ir a L12
		WVC,0F	DAR:=0000 1111
		RWD,1	DAR:=TEKR
		DAO,1B	DAR:=DAR^ORB
Almacénelo	7D	JCZ,L1	Si TEKR=0, ir a L1
		WWS,7	DIGIT:=DAR
Almacene el número del abonado A en POINTER	7E	RWS,2	DAR:=IMKR
		WWS,4	POINTER:=DAR
Marque libre la vía del abonado a KR	7F	WVC,0	DAR:=0
		WWS,2	IMKR:=DAR
Desconecte KR	80	WWD,2	SOKR:=DAR
Marque KR libre	81	WVC,FD	DAR:=FD(1111 1101)
		RWS,9	DAR:=DEVSTATE
		DAO,1B	DAR:=DAR^ORB
		WWS,9	DEVSTATE:=DAR
El abonado B está libre?	82	RWS,7	DAR:=DIGIT
		RWS,5	DAR:=SUBSTATE
		DAO,1B	DAR:=DAR^ORB
Marque ocupada la vía del abonado A a BS	83	JCZ,L1	Si el abonado está libre, ir a L14
		RWS,4	DAR:=POINTER
		RWS,3	DAR:=IMBS
		DAO,1E	DAR:=DAR^ORB
Conecte abonado A a BS	84	WWS,3	IMBS:=DAR
		WWD,3	SOBS:=DAR
Borre DIGIT	85	WVC,0	DAR:=0
		WWS,7	DIGIT:=DAR
		JUN,L1	Ir a L1
ABJ está libre?	86	WVC,01	DAR:=01(0000 0001)
		RWS,9	DAR:=DEVSTATE
		DAO,1B	DAR:=DAR^ORB
		JCZ,L16	Si ABJ está libre, ir a L16

	9E		JUN,L15	Ir a L15
Marque ocupada la vía	{9F	L16	RWS,4	DAR:=POINTER
abonado A a A}	A0		WWS,0	IMAJ:=DAR
Conecte el abonado A	A1		WWD,0	SOAJ:=DAR
a A}				
Marque ocupada la vía	A2		RWS,7	DAR:=DIGIT
B a B}	A3		WWS,1	IMBJ:=DAR
Conecte el abonado B	A4		WWD,1	SOBJ:=DAR
a B}				
Marque ABJ ocupado	{A5		WWC,01	DAR:=01(0000 0001)
	A6		RWS,9	DAR:=DEVSTATE
	A7		DAO,1E	DAR:=DAR∨ORB
	A8		WWS,9	DEVSTATE:=DAR
Escribir el número del	{A9		RWS,7	DAR:=DIGIT
abonado B en SELSUB	AA		WWS,8	SELSUB:=DAR
Borrar DIGIT	AB		WWC,0	DAR:=0
	AC		WWS,7	DIGIT:=DAR
Inicio de señal de	{AD		WWC,01	DAR:=01(0000 0001)
llamada	AE		WWD,4	STRI:=DAR
	AF	L1	WWC,01	DAR:=01
Han pasado 10 ms?	{B0		RWD,2	DAR:=TEPI
	B1		DAO,16	DAR:=DAR∨ORB
	B2		JCZ,L0	Si han pasado 10 ms, ir a L0
	B3		JUN,L1	Ir a L1

5.4 PRUEBAS DE FUNCIONAMIENTO

Una vez integrado todo el sistema con sus partes de conmutación y de control se realizaron las siguientes pruebas para comprobar el funcionamiento de dichas partes:

Primero se hicieron las pruebas de comunicación de un abonado con otro, por ejemplo el abonado "0" llama al abonado "1", el "1" llama al "2", el "2" llama al "3", el "3" llama al "0" y así todas sus combinaciones posibles, detectándose algunas fallas por lo que se tuvo que reprogramar el Chip, ya que solo se había logrado que el abonado "0" pudiera llamar al abonado "1".

Posteriormente se realizó la prueba de que un abonado "A" llame al abonado "B" que en ese instante está ocupado con lo cual el abonado "A" debe de recibir tono de "ocupado". Esta prueba se realizó con éxito para la combinación entre los 4 abonados.

Otra prueba consistió en ponerle una troncal a nuestra central electrónica, esto se desarrolló tomando una extensión del conmutador de la U.N.A.M. (550-52-15), que sirvió en ese momento como una central urbana.

En esta prueba que fué un éxito, se logró una comunicación de un abonado de nuestra central que en este caso fue el abonado "0", con un abonado del conmutador de la U.N.A.M. ya que se tomo línea marcando un "9" y posteriormente se marcaron las extensiones 3756 y 3763 de dicho conmutador dándose así la comunicación sin ningún problema.

Más aún sobre esta prueba, se realizó una comunicación con un abonado de la planta telefónica, para lo cual se utilizó la misma troncal del conmutador de la U.N.A.M. y se tomo línea marcando un "9" para tener acceso al conmutador de la U.N.A.M., posteriormente se marco un "8" que es la cifra para tomar línea hacia la planta telefónica de Teléfonos de México y se marco el 650-66-88 realizándose la comunicación sin ningún problema.

Una prueba más fue la llamada de consulta y que consistió en tener a 2 abonados comunicados (el "0" y el "2"), se llamo al abonado "3" a través del botón del teléfono para retener la llamada y obtener tono y poder marcar al abonado "3" teniendo también éxito.

6. CONCLUSIONES

6- CONCLUSIONES

Los objetivos de esta tesis puede decirse que se lograron en su totalidad, ya que se habia propuesto que el conmutador SX-10, realizara funciones diferentes a las que tenia programadas de fábrica.

Lo anterior se logró gracias a que el conmutador SX-10 es del tipo de control por programa almacenado, por lo tanto, cuando se quiere modificar sus funciones, no es necesario tocar la circuiteria, sino que basta con realizar un nuevo programa y almacenarlo en las memorias ROM.

Este trabajo de programación, puede considerarse que es la aportación del autor de la tesis, misma que fué realizada gracias a los conocimientos adquiridos al laborar en Teléfonos de México.

Esto no quiere decir que lo aprendido en la facultad de Ingeniería no haya sido utilizado, ya que en este proyecto se requirieron conocimientos de electrónica, comunicaciones, programación y microprocesadores, y un aspecto muy importante que muchas veces pasa desapercibido: Después de haber cursado aproximadamente 50 materias en la carrera, el alumno ha visto 50 formas diferentes de plantear y resolver problemas, y esto va proporcionándole al alumno una metodología que aplica en su vida profesional.

Con este proyecto el personal técnico de Teléfonos de México podrá apoyarse en este material para realizar prácticas simulando los Sistemas AXE, S-1240 y E-10, que son los que se están utilizando en la modernización de la Planta Telefónica, siendo estos de gran utilidad dentro de la capacitación y para el usuario, una gama de futuros servicios telefónicos.

De lo anterior podríamos deducir que este proyecto es exclusivo para el personal de Teléfonos de México, pero también se puede utilizar para entrenar a cualquier alumno de la especialidad de Ingeniería en Comunicaciones.

El trabajo que realizarían los alumnos sería del mismo tipo que el realizado en esta tesis; esto es el profesor indicaría las funciones que debe realizar el conmutador, y los alumnos modificarían el software y grabarían las EPROM con el fin de ejecutar las nuevas funciones.

El equipo SX-10 demuestra la gran versatilidad de los equipos electrónicos digitales, ya que utilizando su hardware y modificando su software, pudimos cambiar sus funciones.

La ventaja de los equipos telefónicos electrónicos, es que cuentan con mayor capacidad de conmutación y ofrecen nuevos servicios con mayor rapidez de conexión que los equipos que tienen su unidad de control con lógica cableada o de relevadores.

Otra característica del equipo que se usó en esta tesis, es su compatibilidad, ya que puede conectarse a cualquier central electromecánica o electrónica.

Finalmente, se mencionará que en la escuela se aprenden las aplicaciones de los microprocesadores como dispositivos de cómputo y como controladores de procesos industriales; aquí se ha presentado su aplicación en el control de una central telefónica, cosa que no se ve todos los días, y que puede ser usado en los laboratorios de la facultad de Ingeniería.

BIBLIOGRAFIA

- 1) MANUAL DE TELEFONIA ELEMENTAL, Teléfonos de México
- 2) MANUAL DE CONMUTACION, Teléfonos de México
- 3) MANUAL DE TECNICAS SPC, Ericson
- 4) DISEÑO DE SISTEMAS DIGITALES Y MICROPROCESADORES, John F. Hayes Ed. Mc. Graw Hill
- 5) COMMUNICATIONS TELEPHONICS A GRANDE DISTANCE, Comité Consultatif International
- 6) LENGUAJES DE PROGRAMACION, Allen B. Tucker Ed. Mc Graw Hill