

17
24



UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

DISEÑO DE UN DISPOSITIVO PARA TRANSMITIR SEÑALES DIGITALES ENTRE DOS APARATOS TRANSCPTORES A TRAVES DEL TELEFONO EN FORMA ACUSTICA

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO MECANICO ELECTRICISTA

P R E S E N T A N :

ROBERTO AXALCO MONTES

OLGA MARIA IGLESIAS RAMIREZ

ENRIQUE MARTINEZ FLORES

MARIA DEL ROSARIO VAZQUEZ FUENTES

Director de Tesis: ING. GERARDO CASTRO PARRA



MEXICO, D. F.

TESIS CON FALLA DE ORIGEN

1989



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.

INDICE

INTRODUCCION	1
CAPITULO 1 CONDICIONES DE DISEÑO.	
1.1 Antecedentes	4
1.2 Requerimientos de diseño	5
CAPITULO 2 SOLUCION DEL PROBLEMA.	
2.1 Metodología de diseño	8
2.1.1 Planteamiento	10
2.1.2 Diseño preliminar o arquitectura	11
2.1.3 Diseño detallado	12
2.1.4 Construcción del prototipo	13
2.1.5 Depuración	13
2.1.6 Documentación final	14
2.2 Planteamiento general de solución	14
2.3 Estructura y funcionamiento del circuito ...	16
2.3.1 Estructura del circuito	16
2.3.2 Funcionamiento del circuito	20
CAPITULO 3 DESCRIPCION DEL HARDWARE UTILIZADO EN EL DISEÑO DEL CIRCUITO.	
3.1 Control del sistema	23
3.1.1 CPU Z-80	23
3.1.2 Sistema de reloj y sistema de reset ..	32
3.2 Memoria	34
3.2.1 Tipos de memoria	34
3.2.2 Memoria utilizada en el circuito	35
3.2.2.1 Memoria EPROM 2716	35
3.2.2.2 Memoria RAM 6116	37
3.3 Circuitos de entrada y salida digital	39
3.4 SIO Z-80	42
3.4.1 Descripción general	42
3.4.2 Generador de baudaje MC14411	47
3.5 Direccionamiento de dispositivos	48
3.6 Condiciones estáticas y dinámicas del circuito	49
3.6.1 Análisis dinámico	49
3.6.2 Análisis estático	52
3.7 Modulación y demodulación	55
3.7.1 Proceso de modulación y demodulación ..	55
3.7.2 Tipos de modulación	56
3.7.3 Elección del tipo modulación	62
3.7.4 Modos de comunicación	62
3.7.5 Elección del modo de comunicación	63

3.7.6	Requisitos de las normas para la transmisión de datos por la red telefónica	64
3.7.7	Clasificación de las velocidades de transmisión	64
3.7.8	Diseño del modulador	65
3.7.9	Diseño del demodulador	70
3.8	Acooplamiento del circuito a la línea telefónica	75
3.8.1	Acooplamiento del transmisor	75
3.8.2	Acooplamiento del receptor	79
3.8.3	Pruebas realizadas	80

CAPITULO 4 DESCRIPCION DEL SOFTWARE UTILIZADO EN EL DISEÑO DEL CIRCUITO.

4.1	Programación del CPU Z-80	82
4.1.1	Instrucciones del CPU Z-80	83
4.1.2	Modos de direccionamiento	85
4.1.3	Modos de interrupción	87
4.2	Programación del SIO Z-80	88
4.2.1	Modo asincrónico	90
4.2.2	Modo síncrono	90
4.2.3	Técnicas de I/O de datos del SIO	91
4.2.4	Registros de lectura y escritura	94
4.3	Programación del circuito	97
4.3.1	Rutina de inicialización del SIO Z-80 y del sistema	97
4.3.2	Rutina de validación y transmisión de dato	103
4.3.3	Rutina de recepción y validación de dato	110
4.4	Prueba del software	114

COMENTARIOS Y CONCLUSIONES

APENDICES

BIBLIOGRAFIA

INTRODUCCION.

El gran desarrollo en el campo de la electrónica y las múltiples aplicaciones en diversas áreas que esto permite, hace posible encontrar soluciones que hasta hace poco tiempo resultaban costosas o poco prácticas. El trabajo desarrollado en esta tesis es un ejemplo de los avances alcanzados en la ingeniería electrónica.

Con el objeto de proporcionar una mayor posibilidad de integración social y productiva de individuos con deficiencia auditiva, surge la idea de diseñar este proyecto, ya que la sordera, tanto parcial como total es una gran limitante en el desarrollo normal de las personas afectadas.

Dichas deficiencias se presentan en diferentes grados y por diversos motivos, que a continuación se mencionan con el fin de enfatizar los beneficios que aporta este diseño.

Existe la sordera adquirida, que en la mayoría de los casos es provocada por enfermedades de etiología viral, sufridas durante la infancia (sarampión, rubéola, meningitis, traumatismos, etc.) y la sordera hereditaria, que es de etiología genética, en tal caso, los niños sufren la sordera en forma progresiva, llegando a ser total antes de salir de la niñez.

Un hipoacúsico es una persona con deficiencia auditiva y un sordo profundo es una persona con sordera total o anacusico.

En ambos casos, estas personas tienen atrofiada la función del habla, ya que aparte del aspecto psicológico provocado por no oír, existe un daño irreparable afectando el desarrollo sensorial y cognoscitivo del lenguaje, por no tener una realimentación acústica de su medio.

Sin embargo, existen para el sordomudo algunas posibilidades de educación especial, tales como aprender la lectura de los labios como medio de comunicación, la mímica y además poder utilizar un aparato auditivo auxiliar, con características de amplificación y cambios de ganancia auditiva según la ubicación de la fuente sonora, frecuencia, eliminación de ruido indeseable, etc., así mismo puede aprender el dominio de los monosílabos. Para el sordo profundo, las técnicas anteriores no le proporcionan resultados satisfactorios, por considerarlos distorsionantes y de poco beneficio práctico.

Para la rehabilitación del hipoacúsico y del sordo profundo se desarrolló un sistema especial, que se base en un

lenguaje llamado: Lenguaje-Tonal-Vibratorio-Simbolico-Non-Verbal (LETOVISIMNOVER), cuyo diseño lleva este nombre. El objetivo principal es comunicar a personas sordas profundas entre sí y a éstas con personas de audición normal. Así se tendrá un medio de comunicación universal, en donde no importará la nacionalidad, idioma ni dialecto. Este sistema de comunicación tiene como base la generación de ciertos patrones vibratorios o tonos, correspondientes a estímulos acústicos con diferentes frecuencias, que se asocian a ideas de un idioma por medio de imágenes e ideogramas.

Los ideogramas o pictogramas representan conceptos o ideas, que con ayuda de sistemas complementarios al LETOVISIMNOVER, se pueden reproducir gráficamente en una pantalla de rayos catódicos, donde se observan los espectrogramas de las palabras y sonidos deseados.

El sistema LETOVISIMNOVER constará de dos partes fundamentales:

1. La primera es la encargada de generar los patrones tonales o vibratorios, que corresponden a conceptos, los cuales sustituyen las frases o sílabas de un idioma por pictogramas. Estos patrones pueden ser reconocidos tanto por personas normales como sordomudas, ayudando a las últimas en cierta medida a salir de su medio, generalmente aislado o reducido al ambiente familiar y facilitando así su desarrollo en la sociedad actual.

Este aparato está construido con circuitos electrónicos capaces de reproducir fielmente en base a un tono grave fundamental, nueve múltiplos del mismo, más uno independiente de control.

Los siete primeros tonos están asociados a un grupo de colores que van desde el morado hasta el rojo, pasando por el añil, azul, verde, amarillo y naranja al mismo tiempo con los siete primeros números, del 1 al 7. Los siguientes tres tonos están asociados a los colores blanco, negro y gris y al mismo tiempo con los números 8, 9 y 0. El tono independiente sirve para delimitar los patrones vibratorios de acuerdo al mensaje transmitido.

Para utilizar el LETOVISIMNOVER se necesitan dos fases de entrenamiento.

En una de ellas se deberá familiarizar al usuario con los sonidos producidos, en forma de sea tonal y vibratoria, asociándolos al color y dígito correspondiente.

La segunda fase consiste en reconocer las ideas o conceptos que se asocian a grupos de tres tonos, del 100 al 999 y finalmente grupos de cuatro tonos, del 1000 al 9999.

Las posibilidades de la simbología empleada pueden ser infinitas, pero bastará con 2000 símbolos, que representan los pictogramas fácilmente reconocibles, los cuales asocian las ideas relacionadas con un tronco común. Por ejemplo, con el agua: lluvia, nieve, lago, vapor, nube, etc; de posición: debajo, encima, al lado, arriba, etc; de tiempo: hoy, mañana, luego, ayer, etc; de ideas abstractas: alegría, emoción, amor, tristeza, dolor, etc.

También se manejan tiempos verbales como el presente, pasado, futuro, modo imperativo y tiempos compuestos, en base a un sólo símbolo. Cada símbolo se podrá combinar con otros para formar nuevos conceptos más complejos.

Se está realizando ya un programa de entrenamiento dirigido a hipoacúsicos para medir su capacidad de pensamiento abstracto ante estímulos vibratorios, percepción vibratoria y a éstos relacionándolos con operaciones matemáticas de memoria. Los conjuntos de estímulos son separados por medio del tono de control.

2. La otra parte del LETOVISIMNOVER debe ser capaz de comunicar a las personas por medio del teléfono y desde cualquier lugar. Para ello se diseñará un sistema auxiliar con las características deseadas, con el fin de complementar la función de todo el sistema. Esta parte es la correspondiente al proyecto que se desarrolla en la presente tesis.

Las ventajas del LETOVISIMNOVER son: comunicar por vía telefónica a hipoacúsicos e hipoacúsicos profundos entre sí y a éstas con personas de audición normal, o en donde los medios de comunicación presentan mucha interferencia, como la que existe a veces en el ambiente de los pilotos de las naves aéreas y su torre de control.

El beneficio socioeconómico del aparato estriba principalmente en incorporar a la sociedad activa a la mayoría de las personas con problemas de sordera, ofreciendo una forma de comunicación universal, que les brindará alternativas más dignas y humanas en su futuro.

CAPITULO 1 CONDICIONES DE DISEÑO.

1.1 Antecedentes.

Uno de los objetivos del LETOVISIMNOVER es poder ofrecer la oportunidad a personas sordomudas de comunicarse por teléfono, tanto con personas normales como con personas sordomudas, que cuenten con otro aparato igual. La comunicación no se realiza por medio de un lenguaje verbal ni tampoco en algún idioma conocido hasta ahora.

La idea del aparato es fundamentalmente asociar una palabra o un concepto básico, con el correspondiente pictograma o ideograma, éstos últimos a su vez se asocian cada uno a un patrón tonal o vibratorio que es reconocido por la persona sordomuda, quien con ayuda del aparato pueda generar un conjunto de ellos para formar un mensaje capaz de ser transmitido de teléfono a teléfono. Es decir, si cada símbolo representa una palabra o concepto completo y único, pueden combinarse entre sí, de tal manera que formen conceptos o ideas más complejas.

Por otro lado, aparte de generar pictogramas, el LETOVISIMNOVER debe contar con una etapa adicional de circuitos, que permita captar y amplificar la señal que se recibe o se envía por el teléfono, tanto en el audifono como en el micrófono del auricular. Lo anterior significa que el aparato LETOVISIMNOVER consta de dos conjuntos de circuitos, uno de ellos es el diseño a realizar en el presente trabajo, es decir, el que se encargará de permitir la comunicación via telefónica.

Otra etapa adicional ligada con la función del presente diseño y por lo tanto con el LETOVISIMNOVER, es adobar un circuito detector de llamada telefónica, que con una etapa de potencia genere una señal capaz de llamar la atención de la persona sordomuda, cuando el teléfono suene. Esto sería por medio de una serie de lámparas colocadas en lugares estratégicos en su casa, activándose en forma intermitente e indicando que el teléfono está sonando. En el momento en que la persona descuelgue la bocina, las lámparas se desactivarán definitivamente. Para establecer la comunicación con la persona que hizo la llamada, bastará con colocar el auricular sobre el aparato.

Cuando termina la comunicación por teléfono, cada uno de las personas retira el auricular del aparato y cuelga.

Ahora bien, cuando se necesite efectuar una llamada, se tienen dos posibilidades: la primera es que debe estar una persona normal, la cual se encargue de obtener el tono de marcar e indicárselo a la persona sordomuda; la otra posibilidad es conectar un circuito detector de tono de marcar, que le indique de alguna manera a la persona sordomuda que el tono está presente, si está ocupado, etc. En caso de que exista el tono, se marca el número telefónico, se coloca el auricular sobre el acoplador del aparato, se espera respuesta y finalmente, si la hay, se establece la comunicación.

1.2 Requerimientos de diseño.

Para el desarrollo del presente trabajo de tesis, así como para la comprensión del mismo y de sus objetivos implícitos, es indispensable enumerar los requerimientos o condiciones planteadas que debe cumplir el diseño para poder acoplarse al sistema del que forma parte, es decir, del LETOVISIMNOVER.

Cabe hacer notar que algunas de estas condiciones fueron señaladas desde el principio, otras se dedujeron en el momento de enfrentarse a los problemas que se presentaron al estar diseñando cada etapa.

A continuación se enumeran todas estas condiciones :

1. El diseño debe ser capaz de transmitir doce señales digitales, que son las salidas del teclado del LETOVISIMNOVER, de un aparato a otro a través de la línea telefónica.
2. Cada señal o salida del teclado se encuentra en un nivel de +5 V cuando no se oprime la tecla, estas salidas del teclado, son a su vez las entradas correspondientes al circuito a diseñar.
3. Cuando se oprime una tecla para enviar un mensaje, el nivel de voltaje de salida de la tecla cambia de +5 V a 0 V y se mantiene en este último valor mientras esté oprimida.
4. De las doce teclas, el aparato sólo debe permitir oprimir una a la vez. En caso de oprimir dos o más, el circuito debe ser capaz de ignorar ese mensaje, es decir, no debe transmitirlo ya que es un error o

mensaje irreconocible por los aparatos LETOVISIMOVER (ver tabla de códigos permitidos, cap 4).

5. En el momento en que se oprime una tecla, se obtiene un código que consta de doce señales en paralelo, una de ellas en nivel bajo de voltaje (0 V) y las restantes en un nivel alto de voltaje (+5 V) Este código debe ser transmitido y la otra persona debe recibir exactamente el mismo mensaje y al mismo tiempo que la persona que lo envió, con el consabido retardo electrónico.

6. El tiempo que se mantenga oprimida una tecla en un aparato transmisor, debe ser el mismo que permanezca el código en el receptor.

7. La comunicación debe ser de tipo half-duplex, es decir, cuando un aparato transmite, el otro únicamente puede recibir.

8. La conexión del aparato a la línea telefónica cuando se realiza una llamada, debe ser en forma acústica, es decir, por acoplamiento del auricular del aparato telefónico al LETOVISIMOVER con el fin de no abrir la línea. Esto significa que una forma de entrada y salida de información al circuito a diseñarse por el auricular del teléfono, por esto, debe ser capaz de captar y reconocer la señal que recibe del audífono y también de transformar las señales digitales para enviarlas por su salida acoplada a la bocina del auricular.

9. El diseño debe permitir realizar llamadas desde cualquier teléfono, de casa o público, para que el usuario tenga mayor posibilidad de comunicación, por lo que el tamaño del aparato debe ser adecuadamente pequeño, como un radio o grabadora portátil. Lo anterior implica que el tamaño del aparato es sumamente importante, por lo que el número de circuitos debe ser el menor posible, sin descuidar la funcionalidad del circuito.

10. Como el aparato será portátil, debe contar con dos posibilidades de alimentación al mundo: una, en lugares donde sea posible, conectarlo a la línea de corriente alterna directamente y la otra, cuando la persona esté en la calle, poder utilizar pilas comerciales.

11. Como se pretende lograr la fabricación en serie de todo el aparato del que formara parte el circuito a diseñar, debe tomarse en cuenta el aspecto económico, es decir, el costo de los componentes, facilidad de adquisición en el país y que sean lo más comercialmente posible.

12. Desde el punto de vista de mantenimiento es importante considerar la tecnología con que cuenta nuestro país, de ahí que el diseño debe ser lo más sencillo posible en cuanto a hardware para que en caso de que el aparato requiera un mantenimiento correctivo, la falla sea fácilmente localizable y si existieran componentes dañados, éstos se encuentren fácilmente en establecimientos del ramo de la electrónica en el país.

CAPITULO 2 SOLUCION DEL PROBLEMA.

2.1 Metodología de diseño.

Un aspecto fundamental que debe tomarse en cuenta antes de desarrollar cualquier tipo de trabajo o investigación, es la metodología a seguir, de no ser así, el tiempo empleado y el esfuerzo aumenta considerablemente, así como la dificultad en realizarlo. Esto origina un incremento en el costo del trabajo.

Es sumamente importante contar con una metodología, en este caso de diseño, misma que sirve para organizar el trabajo a realizar entre otras cosas, ya que existe con ello una planeación definida de tiempo asignado a cada parte del trabajo. En ingeniería esta metodología es necesaria y fundamental para optimizar todos los recursos disponibles.

Con lo anterior se llega a definir perfectamente cuáles son los requerimientos que debe cumplir el diseño a realizar; de otra manera se corre el riesgo de desviarse de los objetivos planteados.

Una vez que se tiene definido el comportamiento específico y las condiciones de diseño o restricciones de cualquier tipo, se trata de dividir todo el sistema en grandes bloques básicos, cada uno de los cuales efectúa una tarea específica y hasta cierto punto independiente por ser especializada.

Cada bloque se distingue de otro, por el tipo de componentes, tarea que realiza y nivel de complejidad. Sin embargo, debe tenerse en cuenta la interacción de cada uno en todo el sistema del que forma parte.

En este caso se realiza un diseño del tipo Top-Down (de arriba hacia abajo o descendente). Esto significa que se parte de un objetivo principal, el cual se descompone en grandes bloques y éstos a su vez se dividen en otros bloques más específicos cada vez, conforme se avanza de nivel. Se manejan tantos niveles como sea necesario, para cubrir cada uno lo mejor posible.

Este tipo de diseño permite tener una visión bastante clara de lo que se quiere hacer, de las partes de que consta el trabajo, la jerarquía entre ellas y como se relacionan entre sí.

En general puede decirse que el diseño se realiza en base a lo siguiente:

1. Diseñar A utilizando bloques de nivel B1.

2. Diseñar cada bloque B_1 utilizando bloques de nivel B_2 , si es posible.
3. Diseñar cada bloque B_2 , utilizando bloques de nivel B_3 , si es posible.
- .
- N. Diseñar cada bloque $B(N-1)$, utilizando bloques de nivel B_N , si es posible.

Gráficamente se puede representar como se muestra en la figura 2.1.a.

De acuerdo a lo anterior, se obtiene el diagrama de diseño top-down de la figura 2.1.b, que corresponde al diseño a realizar.

Este tipo de diseño es de gran utilidad en la presente tesis, ya que para el caso de descompostura del circuito, el hecho de desglosarlo en bloques, permite localizar la falla fácilmente.

Antes de seguir adelante, es importante definir algunos conceptos básicos.

Un sistema es un conjunto de elementos o partes, relacionados o interconectados entre sí para formar un todo unificado o cubrir un objetivo común.

Existe una gran cantidad de ejemplos de sistemas, tales como : el sistema nervioso, el sistema solar, un sistema computacional, etc.

Cuando las interconexiones son difíciles de entender, se dice que el sistema es complejo.

Por otra parte, cuando el sistema está hecho por el hombre, se le llama artificial y se considera sistema natural cuando el hombre no intervino en su formación.

El tipo de sistema que interesa, en este caso, es un sistema digital, el cual se define de la siguiente manera:

Es un sistema artificial que consta de un conjunto de componentes y/o circuitos interconectados, que realizan una función específica por medio del manejo de señales de tipo digital. Los componentes son circuitos integrados, en su mayoría componentes discretos y los enlaces son cables o pistas.

Ahora bien, la diferencia entre una variable analógica y una digital es, que la primera puede tomar un valor entre un número infinito de valores dentro de un rango de variación, aun cuando éste sea muy pequeño y la segunda puede tomar un solo valor de un conjunto finito de valores permitidos. Es por ello, que al graficar una variable de tipo analógico en

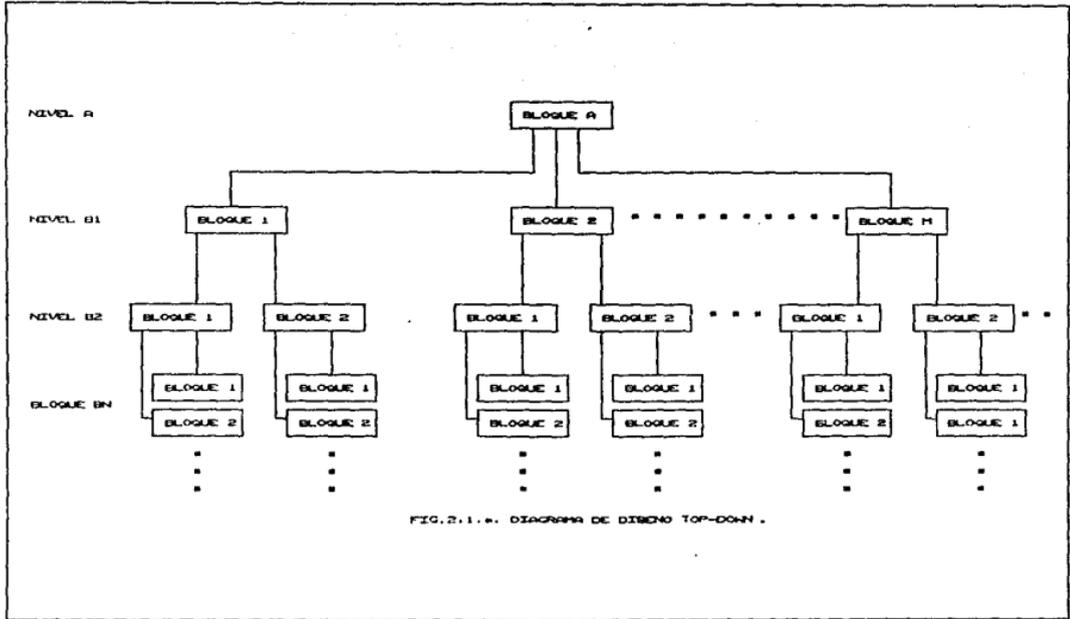
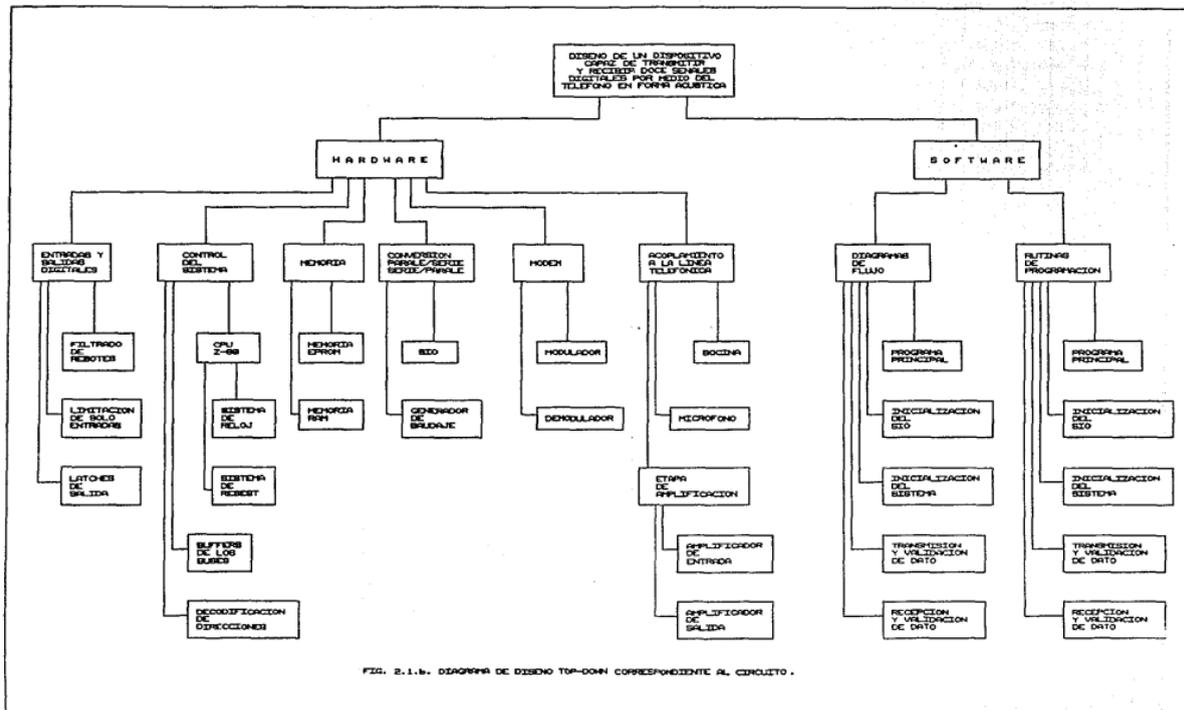


FIG. 2.1.a. DIAGRAMA DE DISEÑO TOP-DOWN.



función de otra variable, se obtiene una línea continua.

La estructura de un sistema indica cómo está construido y de qué componentes consta. En cambio el comportamiento del sistema indica qué hace o cómo funciona.

Para analizar mejor un sistema es conveniente dividirlo en sistemas más pequeños y fáciles de entender, a éstos se les llama subsistemas y deben existir tantos como el nivel de complejidad del sistema lo requiera.

En el diseño, los pasos a seguir pueden diferir de un sistema a otro, al igual que el tiempo empleado en cada etapa, sin embargo, los pasos principales son los siguientes:

1. Planteamiento.
2. Diseño preliminar o arquitectura.
3. Diseño detallado.
4. Construcción del prototipo.
5. Depuración.
6. Documentación final.

A partir de este momento, al hablar de diseño y sus etapas, se hace un enfoque al diseño de un circuito digital.

2.1.1 Planteamiento.

En él se obtienen las especificaciones sobre el comportamiento que debe tener el sistema. Se hace una planificación del trabajo y una estimación probable de costos en general, si es posible.

En esta etapa se definen los objetivos que debe cumplir el diseño, por lo tanto, el punto de partida para resolver el problema es saber exactamente lo que se quiere resolver. En este caso, el objetivo es el siguiente:

Diseñar un dispositivo capaz de transmitir y recibir 12 señales digitales por medio de la línea telefónica con un acoplamiento acústico.

En este punto se enumeran todas las condiciones posibles de diseño, limitantes o restricciones y es muy importante tener mucha comunicación con la persona que pidió el diseño.

Debe quedar bien definido qué es lo que desea obtener el usuario o las personas interesadas en el diseño a realizar; se debe platicar con ellos para que exista un intercambio de ideas en donde se puedan plantear las dudas, sugerencias y problemas con los que no se contaba, de tal manera que se llegue a un acuerdo entre ambas partes.

Aquí se definen requerimientos, no sólo de funcionalidad, si no también de costos y tiempo, factores que muchas veces son más decisivos en la elección de un diseño. En muchas ocasiones, se cuenta con dispositivos funcionales, más poderosos y eficientes los cuales son desplazados por otro más barato y no necesariamente igual o más eficiente.

En este proyecto, la funcionalidad, el costo y el tiempo de diseño son factores primordiales que se deben tener en cuenta durante todo el diseño.

Debe existir también, una etapa de investigación, la cual incluye temas relacionados con el diseño a realizar, así como conceptos básicos que se requieren.

Se hace una investigación sobre lo que ya existe en el área, por ejemplo, si ya existen otros dispositivos similares, qué tan parecido será a ellos, las ventajas y desventajas de cada uno, es decir, hacer una comparación en general.

Todo esto es con el fin de llegar a una solución lógica y que el diseño esté de acuerdo a lo que hay en el mercado en el país, además de asegurar que se resuelve un problema, hasta cierto punto de una manera diferente, en otras palabras, no resolver lo que ya está resuelto y de la misma forma.

2.1.2 Diseño preliminar o arquitectura.

En este punto se hace un diseño básico o preliminar del circuito en sí, por ello se llama arquitectura del sistema.

Aquí se entra de lleno al proyecto, es decir, partiendo del objetivo principal y en base a los requerimientos y limitantes, surge primeramente un esbozo del circuito, para dar paso a un diagrama de bloques de todo el sistema, en el cual se indican los bloques o partes necesarios que debe tener. Básicamente se hace la estructura del circuito para posteriormente, en este paso también, hacer un diseño usando ya los componentes principales.

En subtemas siguientes se muestra el diagrama de bloques correspondiente al presente proyecto.

Para el software se hace algo similar, ya que primero se elabora un algoritmo de lo que será el programa y a partir de éste se elabora el diagrama de flujo correspondiente.

2.1.3 Diseño detallado.

En este punto se hace un diseño a detalle de cada bloque o subsistema, a nivel de circuitos. Se puede decir que aquí empieza la depuración del diseño, debido a que por ser más específico y detallado, indica claramente los errores de la etapa anterior. Muchas veces el hecho de ver los problemas de una forma general hace que se pasen por alto los detalles importantes que se presentan en este punto. Esto puede incluso llegar a alterar o modificar gran parte del diseño previo y en casos críticos cambiar el enfoque de la solución.

Se realizan consultas frecuentes y a fondo de manuales, tanto para software como para hardware. Se consultan además libros y personas del área. Se investigan costos, existencia de componentes en el mercado, entre otras cosas, ya que son factores importantes en la depuración.

Aquí se invierte un tiempo considerablemente mayor por todo el trabajo que debe realizarse a detalle. La finalidad de este punto es obtener un circuito diseñado con el menor número de errores ya que el paso siguiente es la construcción del mismo. A menos que el circuito sea muy simple, generalmente presenta errores debido a que no se ha probado físicamente.

En el caso del software se hace algo similar, ya que a partir del algoritmo, se elabora el diagrama de flujo correspondiente, al cual se le hacen las pruebas de escritorio necesarias para comprobar que el funcionamiento del programa sea correcto. Se trata de depurar el diagrama de flujo a fin de que la codificación resulte sencilla y clara. Un requisito importante del programa es que debe ser estructurado para que en caso de corrección no se tenga que cambiar toda la estructura, debe ser muy claro y utilizar el menor número de instrucciones de programación. El diagrama de flujo que se obtiene, se codifica.

El paso siguiente en la programación, es probar que el software no tenga errores, para lo cual se utiliza la computadora y la ayuda de algún software adicional. La depuración consiste en obtener un programa libre de errores de sintaxis y lógica. En este caso las pruebas realizadas al programa elaborado para el circuito, se hicieron por medio de un simulador del sistema del microprocesador utilizado.

Como se observa, existen basicamente dos etapas de diseño en sí: el diseño preliminar o arquitectura y el diseño detallado; las otras etapas aunque no corresponden al diseño propiamente dicho, son necesarias para la comprobación del mismo, entre otras cosas.

2.1.4 Construcción del prototipo.

Se implementa un modelo de acuerdo al diseño elaborado. Esta es ya la etapa de construcción, donde se obtiene un prototipo o modelo en base al diseño previo en la cual se conectan y colocan los componentes del sistema.

Este armado, también llamado alambrado del circuito, se debe hacer con el mayor cuidado posible con el fin de tener el menor número de errores en el momento de realizar las pruebas de funcionamiento del mismo.

2.1.5 Depuración.

Consiste en realizar una serie de pruebas sobre el funcionamiento del modelo para ver si cumple con los requerimientos. De no ser así, se realizan modificaciones hasta obtener los resultados deseados, ya que en base a la experiencia, el armado de un circuito generalmente presenta errores, a menos que sea muy simple.

Las pruebas se realizan con el hardware y con el software trabajando juntos.

Cuando existen errores, se deben reemplazar o corregir las etapas o elementos involucrados y realizar otras pruebas. Este paso se repite tantas veces como sea necesario.

Es importante hacer notar que si las etapas de diseño anteriores no se llevaron a cabo correctamente o completamente, esta etapa puede ocupar mucho tiempo o resultar muy costosa por el cambio constante de componentes.

En cuanto al software puede resultar caro, desde el punto de vista de tiempo empleado al estar modificando cierto conjunto de instrucciones o la estructura del programa, en el peor de los casos.

La depuración que se hizo, en este caso, con ayuda del simulador, asegura que el programa es correcto, funcional, claro, pero la prueba final es en el momento de utilizarlo en el circuito cuando este último se encuentre ya funcionando.

2.1.6 Documentación final.

La etapa final, como en todo buen diseño debe ser la documentación final anexa al diseño y construcción del sistema.

En esta etapa se completa la documentación necesaria del sistema. Por ejemplo, manuales de uso y mantenimiento.

Se debe hacer una especie de manual con el fin de que el dispositivo sea fácilmente operable, que se aproveche al máximo e incluso se pueda modificar a futuro. El manual debe servir también para que se use correctamente y disminuya así la posibilidad de que se presenten fallas por mala operación.

Respecto a las fallas, el manual debe dar la facilidad de detectarlas y corregirlas fácilmente. Debe contar con bibliografía de otras fuentes que amplíen la información.

Para el presente trabajo, los porcentajes de tiempo empleados en cada etapa de diseño se toman en base al tiempo y esfuerzo invertido. A continuación se muestran dichos porcentajes:

Etapa	Porcentaje
Planteamiento	15 %
Diseño preliminar o arquitectura	25 %
Diseño detallado y depuración	45 %
Documentación final	15 %
<hr/>	
Total	100 %

En este subcapítulo se describieron las etapas de diseño, las cuales se desarrollan implícitamente en capítulos posteriores.

2.2. Planteamiento general de solución.

De acuerdo a las condiciones de diseño mencionadas anteriormente, se presenta a continuación, a grandes rasgos, la forma de atacar el problema y la mejor solución encontrada con su justificación.

Considerando las entradas al circuito, que son señales digitales y por ser éstas las salidas del teclado, se llega a la conclusión inmediata de utilizar lógica digital TTL en su mayoría, por las condiciones de trabajo.

Inicialmente se optó por una solución, que parecía la más simple, a base de lógica alambrada, utilizando un arreglo de circuitos tales como: latches, para retener el dato proveniente del teclado; decodificadores, para obtener un número de señales adecuadas al bus del sistema, que es determinado por las entradas del circuito serializador; un arreglo de compuertas AND para detectar cuándo se oprimió una tecla; un conjunto de XOR para detección de paridad; un contador para hacer el control de activar y desactivar circuitos y un reloj como base de tiempo de todo el sistema.

Pronto se vió que esta posible solución no fué la más adecuada, ya que se presentaron muchos problemas de control del circuito. Otra desventaja es, que para posibles modificaciones se debía cambiar toda una serie de circuitos. Sin quedar totalmente terminado este diseño, se vió que el número de elementos empleados sería tal, que no cumpliría la condición de tamaño deseado y muy pronto quedaría obsoleto.

La solución más lógica fué entonces, enfocar el diseño hacia el uso de un microprocesador, debido a que éste se ha convertido en una herramienta sumamente poderosa en el diseño de sistemas de control, donde dicho microprocesador es necesario y común debido a la flexibilidad y versatilidad que ofrece al diseñador, por el hecho de que no es necesario cambiar un conjunto de elementos (cuya función pueda sustituirse por una rutina de programación), cada vez que se quiera modificar el funcionamiento del diseño. Esto hace que el sistema tenga grandes ventajas en mercadotecnia.

El hecho de cambiar el conjunto de componentes electrónicos por otro a base del microprocesador, da como resultado un tiempo de diseño mucho menor.

Las ventajas en cuanto a costos se refiere son también considerables, ya que disminuye el número de circuitos integrados, cuyas funciones las realiza ahora el microprocesador.

Además el control sobre el sistema resulta menos complicado, lo cual da mayor confiabilidad y mejores resultados.

La selección del microprocesador no fué difícil, tomando en cuenta que la función de éste dentro del diseño, es tan sólo de comunicar, es decir, reconocer un dato a la entrada de él, verificarlo y mandarlo hacia el puerto correspondiente de un serializador/deserializador. Así mismo, debe esperar a que el puerto correspondiente de un serializador/deserializador tenga un dato disponible; verificarlo y mandarlo al puerto de salida.

Comparando costos de microprocesadores, existencia en el mercado, facilidad de conexión y funcionalidad, para este

diseño en particular, se decidió usar el microprocesador Z-80 que es de los más baratos, fácil de conectar y es comercial; la programación es menos complicada que en otros microprocesadores y sobre todo cubre perfectamente las funciones que se requieren en el diseño del circuito.

2.3. Estructura y funcionamiento del circuito.

Siguiendo con las etapas de diseño, una vez que se logra plantear la solución del problema, el siguiente paso es construir un diagrama de bloques que sirva de base para el desarrollo posterior a detalle del hardware y software.

En la figura 2.3.a se muestra el diagrama de bloques correspondiente.

2.3.1. Estructura del circuito.

Como se mencionó anteriormente, la estructura del sistema muestra cómo está construido.

1. Filtro de rebotes de teclado.

En base al diagrama de bloques, y partiendo de las entradas digitales, el primer problema a resolver es el de rebotes, que como se explica en otro capítulo merece especial atención, debido a que se debe asegurar que las señales de entrada al circuito estén perfectamente definidas y limpias en todo momento.

Se utilizan para ello flip-flops S-C, los cuales son muy utilizados para este tipo de problemas. Se usan tres circuitos 74LS279, ya que se tienen doce entradas al sistema y cada uno de los flip-flops permiten sólo cuatro entradas.

2. Circuito limitador de sólo entradas.

Otro problema a resolver, es que se tienen doce señales de entrada al sistema: éstas provienen de un teclado y forman un bus de datos que se debe acollar al bus de datos del sistema basado en un microprocesador, dicho bus es de tipo bidireccional y además consta solamente de 8 bits.

Como las salidas del teclado no pueden ser bidireccionales, se conectan las doce entradas que previamente pasan por los circuitos que filtran el rebote, a

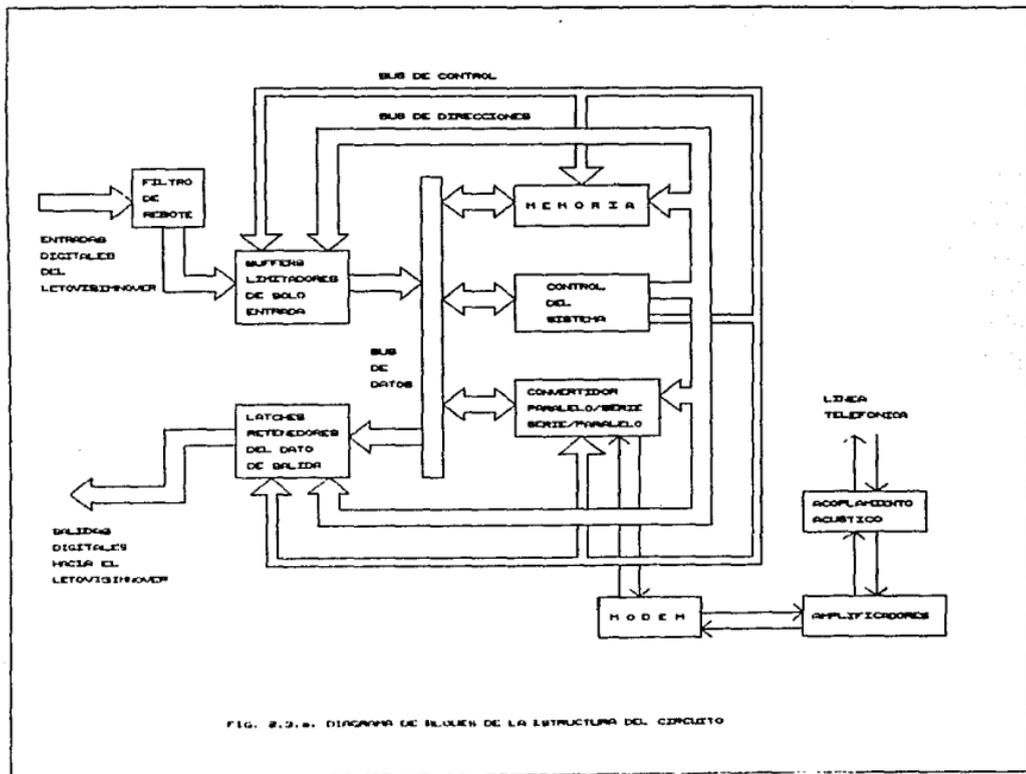


FIG. 2.3.8. DIAGRAMA DE BLOQUES DE LA ESTRUCTURA DEL CIRCUITO

circuitos buffers tres estados unidireccionales. Esto se hace, por un lado, con el fin de que cuando se efectúe la transmisión, la recepción quede deshabilitada y por otro lado, con el fin de acoplar el bus de entrada y salida de doce señales con el bus de ocho bits del sistema.

Se emplean dos circuitos buffers 74LS244 unidireccionales, los cuales colocan sus salidas en alta impedancia cuando no están habilitados. Se usan dos debido a que cada uno de ellos permite solo ocho entradas y se tienen doce, las cuatro que no se utilizan se conectan a +5 V para manejarlas en nivel 1 lógico y de esta manera completar dos conjuntos de entradas de 8 bits cada uno, ya que por medio del bus de datos se efectúa la lectura de los puertos de 8 en 8 bits.

3. Circuito que mantiene el dato de salida.

Uno de los requerimientos del diseño es por un lado, que el dato recibido se mantenga mientras dure oprimida la tecla en el otro aparato y por otro lado, cuando no se recibe un dato, se mantenga un nivel de salida constante de +5 V. Es por ello que se coloca este par de circuitos. En el capítulo 4 se explica como funcionan las subrutinas en combinación con estos puertos de salida (PS01 y PS02).

Estos se encuentran siempre habilitados mediante la terminal OUTPUT CONTROL, ya que no se desea tener salidas en alta impedancia. Por medio de la terminal de control G, se coloca el dato recibido y se mantiene este último en los puertos de salida mientras se esté recibiendo el mismo dato, es decir, mientras se mantenga la misma tecla oprimida en el aparato que está transmitiendo.

Como se manejan también doce señales de salida y el bus de datos es de ocho, se conectan en paralelo dos circuitos 74LS173 entre el bus del sistema y el bus de salidas hacia el LETOVISIMNOVER.

4. Control del sistema.

Todo el control, como se ha dicho, lo hace el microprocesador Z-80.

El bloque de control del sistema y memoria, es un sistema básico de Z-80, que cuenta con el microprocesador, buffers de los buses, su sistema de reloj, reset, un decodificador de direccionamiento de dispositivos, compuertas e inversores. Este sistema básico, es fácilmente modificable, puede tener diversas aplicaciones y hace el control del sistema.

Con el fin de asegurar una corriente adecuada en los buses del microprocesador y por protección del mismo, se utilizan buffers. En el bus de datos se coloca un circuito 74LS245, que es un buffer bidireccional. En el bus de direcciones, por ser salidas del microprocesador se colocan buffers unidireccionales, que son los 74LS244, ya que éstos permiten ocho entradas y ocho salidas, cada uno. Se conectan dos circuitos de este tipo, debido a que el sistema utiliza más de ocho líneas de direcciones.

Para la habilitación de los diferentes dispositivos que actúan como puertos de I/O, se coloca un decodificador de tres líneas a ocho, el 74LS138, cuyas líneas de salida direccionan a:

- a) 1 buffer de entrada que es el puerto PE01
- b) 1 buffer de entrada que es el puerto PE02
- c) 1 latch de salida que es el puerto PS01
- d) 1 latch de salida que es el puerto PS02

Nota: Al hablar de señales de entrada y de salida, se toma como referencia al circuito a diseñar. En el caso de los puertos anteriores, tanto las entradas como las salidas son de tipo digital y en paralelo.

5. Memoria.

Se utilizan dos tipos de memoria, la EPR0M y la RAM.

La memoria EPR0M es indispensable ya que contiene el programa o software que completa la función del sistema.

Se utiliza una memoria 2716 con capacidad de 2K bytes, que como se verá más adelante, es más que suficiente para alojar a todas las rutinas, tablas y programa principal, que juntos ocupan tan solo la sexta parte de memoria EPR0M, aproximadamente.

Se usa esta memoria por ser muy comercial y fácil de conseguir en el mercado. Por otra parte, se tiene mayor acceso a programadores de EPR0M 2716 y resulta sencilla su programación.

Con la memoria que no se utiliza por el momento, se tiene disponible a futuro, si en un momento dado se desea hacer modificaciones al sistema, éste cuenta con la capacidad de almacenar programas más grandes, sin necesidad de colocar otra memoria adicional.

La memoria RAM es una 6116, también de 2K de memoria, la cual se eligió por su popularidad y existencia en el mercado.

Además resulta sencillo conectarla y se coloca en el sistema viendo a futuro, ya que en caso de que se quiera

realizar una comunicación más elaborada, se requerira mas memoria RAM de apoyo.

6. Etapa de serializacion y deserialización.

Aprovechando que se usa el microprocesador Z-80, se emplea como dispositivo serializador/deserializador un SIO (Serial input-output), que es un dispositivo de entrada y salida de la familia del Z-80. La conexión con el sistema es directa y muy simple; además es comercial y cubre las necesidades del mismo.

Anexo al SIO, se utiliza como generador de baudaje un circuito MC14411, el cual es muy utilizado para conectarlo a dispositivos que transmiten a diferentes velocidades, en bauds.

En este caso se usa la salida que proporciona 4800 bauds.

El uso de este circuito evita el utilizar un conjunto de circuitos que genere la velocidad de transmisión deseada.

7. Modem.

En este bloque, la idea inicial era colocar un solo circuito que realizara ambas funciones, es decir, modular y demodular.

Se encuentran varios tipos de modems, sin embargo solo en los manuales correspondientes. Por ejemplo el modem de National Semiconductor el MM74HC942, que es un circuito integrado de 20 terminales, especial para conectarse con la línea telefónica directamente o bien por acoplamiento acústico al auricular del aparato telefónico. Modula en FSK, se conecta fácilmente a la línea de 600 ohms, es compatible con lógica TTL, permite comunicación half-duplex y full-duplex. Tiene sus filtros incluidos. Una de sus típicas aplicaciones es modular a bajas velocidades, como 300 bauds. La conexión de este chip es muy sencilla; desafortunadamente el inconveniente es que no existe comercialmente en el país, por lo que se descartó la posibilidad de usarlo en el proyecto.

Otro modem en circuito integrado, es el TMS99502 de Texas Instruments, el cual es un modem en un solo circuito que trabaja a 300 bauds y el cual se conecta a la línea telefónica en forma directa o acústica.

Sin embargo, al igual que el modem anterior, éste tampoco se encuentra comercialmente en el país y se descartó del proyecto.

Se decidió entonces, partir de circuitos existentes en el mercado dentro del país y de bajo costo.

Los circuitos que cumplen con estos requisitos son: el XR-2206, que se encarga de la modulación y el XR-2211, que se encarga de la demodulación.

Ambos circuitos requieren de componentes adicionales, tales como resistencias y capacitores, los que se calculan en función de los parámetros que se requieren manejar en el diseño, tales como: frecuencias asociadas a la marca (1 lógico) y al espacio (0 lógico), niveles de voltaje de entrada y salida, etc.

Una limitante sumamente importante a considerar en esta etapa y las siguientes, es la que impone la línea telefónica, por sus características fijas.

8. Acoplamiento acústico.

Como los niveles y frecuencias de las señales que viajan por la línea, tienen un rango de valores determinado, se deben ajustar los señales que entrega el circuito a dichos parámetros.

De acuerdo a cálculos y pruebas realizadas, se concluye que deben colocarse amplificadores, tanto a la salida como a la entrada del modem.

A la salida del modulador se coloca una configuración de inversor, utilizando un amplificador operacional que es el TL072 dual. Uno se usa para la recepción y el otro para la transmisión.

En los capítulos siguientes se hace una descripción más detallada y completa de cada uno de los circuitos y componentes empleados en el diseño.

2.3.2. Funcionamiento del circuito.

Como se mencionó anteriormente, el funcionamiento del sistema se refiere a su comportamiento.

A partir del diagrama de la figura 2.3.a se describe el funcionamiento general del circuito como sigue:

En el momento en que se alimenta al circuito, se provoca un reset general del sistema.

A partir de este momento el control del sistema queda a cargo del microprocesador, el cual direcciona la localidad 00H de la memoria EPROM, en donde se coloca la rutina de inicialización del SIO (ésta se ejecuta una sola vez, hasta que exista un reset nuevamente), la cual lo programa para que actúe como receptor o como transmisor de datos (dato es un conjunto de bits que en este caso son 16, partiendo de los

puertos de entrada o salida y son 8 bits, partiendo de las líneas en paralelo de entrada y salida del SIO). En seguida de dicha rutina, está el programa principal y posteriormente la rutina de transmisión y la rutina de recepción. Todas ellas se explican en el capítulo 4.

1. Recepción de un dato.

Como el circuito funciona en modo asincrónico, permite al sistema recibir y transmitir en cualquier momento. Los circuitos de la etapa del módem y acoplamiento acústico, así como los flip-flops de entrada, están siempre activos, mientras reciban alimentación sin importar lo que esté haciendo el microprocesador.

Cuando se recibe un dato por la bocina del teléfono, el micrófono del sistema lo detecta inmediatamente y hace que la etapa de amplificación reciba una señal de tipo FSK, misma que se filtra de ruido y se amplifica. Esta señal pasa a la etapa de demodulación, la cual proporciona a la salida una señal digital y en serie que a su vez entra al SIO por su línea de recepción.

El microprocesador mientras tanto, se encuentra ejecutando el programa principal. Cuando el SIO tiene un dato disponible, interrumpe al microprocesador; éste atiende la interrupción almacenando en RAM la dirección en la que se quedó al ser interrumpido y ejecuta la rutina de recepción que se encuentra en memoria EPROM.

La primera prueba del dato la realiza el SIO, el cual indica errores de paridad, encuadre o desbordamiento; debido a que se maneja una velocidad muy baja, la probabilidad de que aparezcan es casi nula. Cuando se presenta alguno de estos errores, el dato es rechazado por el sistema.

En la rutina de recepción se verifica el dato nuevamente, ya que pueden existir errores diferentes a los anteriores. Si es así, se ignora el dato, de lo contrario se decodifica por medio del programa, de 8 a 16 bits.

Finalmente el dato correcto se envía al puerto correspondiente de salida (PS01 o PE02).

Todo el proceso se repite mientras existan datos disponibles en el receptor del SIO.

2. Transmisión de un dato.

Cuando la recepción termina o simplemente cuando el SIO está en condiciones de enviar datos fuera del sistema, interrumpe por su transmisor vacío al microprocesador, el cual atiende la interrupción y ejecuta la rutina de

transmisión, misma que a su vez permite monitorear el teclado.

En el momento en que se oprime una tecla, el conjunto de señales generado pasa por los filtros de rebote, que siempre se encuentran habilitados. Posteriormente las 12 señales en paralelo pasan a los buffers de entrada (PE01 y PE02) y al ejecutarse la rutina de transmisión monitorea estos puertos, es decir, se leen las 8 líneas de salida de cada puerto. Debido a que el bus de datos del microprocesador es de 8 bits, se leen primero los 8 bits menos significativos (PE01) y en una segunda lectura los 8 bits más significativos (PE02). El dato de 16 bits se compara con una tabla de posibles valores, en caso de que no se encuentre en ella significa que hay error y se ignora. En caso de ser correcto, se busca su equivalente de 8 bits en otra tabla, para realizar la codificación correspondiente.

El dato de 8 bits se manda al SIO, el cual se encarga de convertirlo en dato serie a su salida, además de que le asigna el bit de inicio, de paridad y de paro.

Posteriormente el SIO envía el dato a 300 bauds, por su línea de transmisión, que está conectada a la entrada del modulador. El modulador recibe la señal digital en serie y la transforma en una señal FSK que pasa a la etapa de amplificación, en donde adquiere las características adecuadas para llegar a la bocina del sistema. Dicha bocina se acopla con el micrófono del teléfono que capta la señal generada y manda el mensaje al otro extremo.

Este proceso se repite muchas veces mientras se teclean los datos.

CAPITULO 3 DESCRIPCION DEL HARDWARE UTILIZADO EN EL DISEÑO DEL CIRCUITO.

En este capítulo se hace una descripción más detallada del hardware con el fin de complementar la explicación de la estructura y el funcionamiento del sistema.

El hardware se refiere al conjunto de circuitos o dispositivos de cualquier tipo (mecánicos, magnéticos, electrónicos y eléctricos), cableado, conectores, chasis, etc. los cuales constituyen la parte física del diseño.

El software se refiere al conjunto de programas, rutinas, diagramas de flujo, etc. que se encargan de complementar la función del diseño haciéndolo más versátil.

Una computadora está formada por los siguientes elementos: unidad de control, unidad aritmética y lógica, unidad de memoria o almacenamiento y unidad de entrada salida.

El presente diseño se considera una microcomputadora ya que se compone de las siguientes partes:

1. Microprocesador o unidad de procesamiento central.
2. Memoria o unidad de almacenamiento.
3. Dispositivos de entrada y salida.

En la figura 3.a se muestra el diagrama de bloques de una microcomputadora.

3.1 Control del sistema.

El control del sistema está a cargo del microprocesador; éste requiere de un sistema de reloj y un sistema de reset. A continuación se describe cada uno de ellos.

3.1.1 CPU Z-80.

Un microprocesador es un circuito integrado que está hecho a base de: registros, contadores, codificadores y sumadores. Estos a su vez, están formados de arreglos de compuertas lógicas y flip-flops, los cuales se construyen a partir de arreglos de transistores.

El diagrama de la estructura interna del CPU Z-80 se muestra en la figura 3.1.1.a.

El microprocesador es una parte fundamental de una microcomputadora, ya que se encarga de ejecutar todas las operaciones de control, aritméticas y lógicas. Las partes de que consta se describen a continuación:

1. Unidad lógica y aritmética (ALU).

Se encarga de realizar las operaciones lógicas y aritméticas, para ello el microprocesador utiliza generalmente dos operandos, uno se encuentra en el acumulador y el otro en el registro de instrucciones, ambos son de ocho bits. Esta unidad se comunica con los registros del Z-80 internamente y la operación que debe realizar se la indica la unidad de control.

Las operaciones que realiza la ALU son: suma, resta, desplazamiento (izquierdo o derecho), incremento, decremento, AND, OR, OR exclusiva, comparación, poner un bit a 1, poner un bit a 0 y prueba de bits.

2. Unidad de control.

Se encarga de ejecutar una a una, las instrucciones indicadas por el registro correspondiente. Es la unidad maestra, ya que se encarga de coordinar la función de las demás unidades, generando las señales necesarias para transmitir los resultados a la unidad correspondiente.

La unidad de control cuenta con un decodificador de instrucciones que se encarga de decodificar la instrucción que previamente se leyó de memoria y se guarda en el registro de instrucciones. Este registro almacena el contenido direccionado por el PC (program counter) en la memoria y ese contenido se carga durante el ciclo de búsqueda de cada instrucción (ciclo fetch) en el registro de instrucciones. Otra de sus funciones consiste en almacenar uno de los operandos que la ALU necesita.

Esta unidad sabe qué instrucción se va a ejecutar, porque analiza los ocho bits que forman la instrucción, esta información se manda al controlador de secuencias que es parte de la unidad de control, el cual manda las señales necesarias al lugar adecuado por medio del bus de control, datos y direcciones, así se termina de ejecutar la instrucción.

3. Registros del CPU.

Los registros en general permiten almacenar información, con ello es posible guardar resultados o transferirlos hacia

otro dispositivo, memoria o puerto de I/O.

El Z-80 cuenta con 18 registros cada uno de 8 bits y 4 registros de 16 bits. Todos ellos se describen en la tabla siguiente:

Nombre	Notación	No. de bits	Uso	Tipo
Acumulador	A	8	Especial	Principal
Acumulador	A'	8	Especial	Alternativo
Indicador de Edo.	F	8	Especial	Principal
Indicador de Edo.	F'	8	Especial	Alternativo
Registro B	B	8	General	Principal
Registro B'	B'	8	General	Alternativo
Registro C	C	8	General	Principal
Registro C'	C'	8	General	Alternativo
Registro D	D	8	General	Principal
Registro D'	D'	8	General	Alternativo
Registro E	E	8	General	Principal
Registro E'	E'	8	General	Alternativo
Registro H	H	8	General	Principal
Registro H'	H'	8	General	Alternativo
Registro L	L	8	General	Principal
Registro L'	L'	8	General	Alternativo
Reg. de Interrup.	I	8	Especial	Unico
Refresco memoria	R	8	Especial	Unico
Reg. Índice X	IX	16	Especial	Unico
Reg. Índice Y	IY	16	Especial	Unico
Apuntador de pila	SP	16	Especial	Unico
Contador de prog.	PC	16	Especial	Unico

3a. Acumuladores.

Se encargan de almacenar un operando antes de realizarse la operación y al final guardan el resultado. Son de gran ayuda para almacenar datos de o hacia memoria.

3b. Registros indicadores de estado.

Como su nombre lo indica, es un registro que indica la condición específica del acumulador después de efectuarse ciertas operaciones.

- D0 Bandera de carry (C)
- D1 Bandera Add/Subtract (N)
- D2 Bandera Parity/Overflow (P/V)
- D3 Bandera half carry (H)
- D4 Bandera no utilizada
- D5 Bandera no utilizada

D6 Bandera Zero (Z)
D7 Bandera de Sign (S)

3c. Registros de uso general.

Sirven para almacenamiento de datos en general, por medio de ellos es posible realizar diversas operaciones tales como: almacenamiento, transferencia, etc. Pueden usarse por separado, es decir, de 8 bits o usarse en pares, en operaciones de 16 bits de la siguiente manera: BC, DE, HL o BC', DE', HL'.

3d. Registros de uso especial.

Contador de programa: Contiene la dirección (en 16 bits) de memoria en donde se encuentra la instrucción que se está ejecutando. Es un contador porque se incrementa con cada instrucción ejecutada o se modifica si se presenta un salto a otra dirección.

Apuntador de pila: Permite almacenar el valor del PC cuando se presenta un salto a otra dirección y se mantiene apuntando a la dirección más reciente que entre a la pila.

Registros índices: Guardan una dirección base a partir de la cual se hace un desplazamiento en memoria, con el fin de facilitar el manejo de tablas.

Registro para interrupción: En este registro se guardan los 8 bits más significativos de una dirección correspondiente a una rutina que se ejecuta al ocurrir una interrupción al CPU.

Registro de refresco de memoria: Sirve para almacenar 7 bits que junto con una señal de refresco (RFSH), se manda por el bus de direcciones para refrescar información en memorias dinámicas, mientras el CPU decodifica y ejecuta la instrucción.

4. Diagrama de conexiones de las terminales del CPU Z-80.

El microprocesador cuenta con 40 terminales, que se muestran en la figura 3.1.1.b.

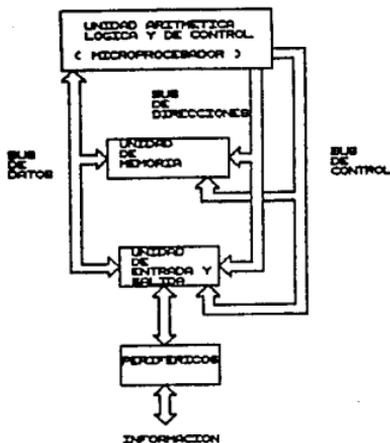


Fig. 3.a Diagrama de bloques de una microcomputadora.

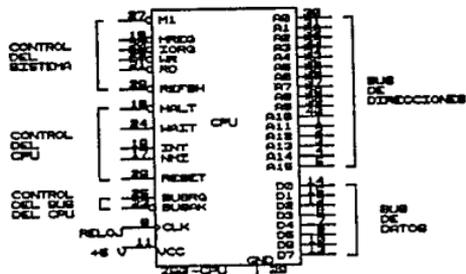


Fig.3.1.1.b Terminales del CPU Z-80.

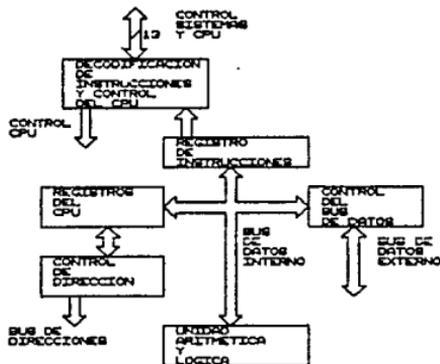


Fig.3.1.1.a Diagrama de bloques de la estructura interna del CPU Z-80.

La función de cada una de las terminales se describe a continuación:

Control del sistema.

No.	Terminal	Descripción
27	M1	Indica que el ciclo de máquina actual es el ciclo de búsqueda del código de operación en la ejecución de una instrucción.
19	MREQ	Requerimiento de memoria. Indica que el bus de direcciones mantiene una dirección válida para efectuar una operación de lectura o escritura en memoria.
20	IORQ	Requerimiento de entrada o salida. Indica que los 8 bits menos significativos del bus de direcciones mantienen una dirección válida de I/O, para efectuar una operación de lectura o escritura en dispositivos de I/O. Esta señal se genera junto con una señal M1 cuando se está reconociendo una interrupción para indicar que el vector de respuesta de interrupción puede ser colocado en el bus de datos.
21	RD	Lectura. Indica que el CPU lee datos de memoria o un dispositivo de I/O direccionado en el momento en que aparece esta señal.
22	WR	Escritura. Indica que el bus de datos de la CPU mantiene un dato válido para ser almacenado en la memoria o dispositivo de I/O direccionado.
28	RFSH	Refresco de memoria. Indica que los 7 bits menos significativos del bus de direcciones contienen una dirección de refresco para las memorias dinámicas y que la señal actual de MREQ debe ser utilizada para efectuar una lectura de refresco para todas las memorias dinámicas habilitadas.

Control del CPU.

No.	Terminal	Descripción
18	HALT	Estado de paro. Indica que el CPU ha ejecutado una instrucción de software que indica paro (HALT) y espera una interrupción no enmascarable o enmascarable antes de que pueda continuar la operación.
24	WAIT	Espera. Su función consiste en sincronizar la operación de dispositivos de I/O o memoria con el CPU. Indica que el dispositivo direccionado no se encuentra preparado para la transferencia de datos.
6	INT	Requerimiento de interrupción. Es una señal generada por los dispositivos de I/O. Se atiende la interrupción al final de la instrucción que se está ejecutando, siempre y cuando, se habiliten previamente las interrupciones por software y la señal BUSRQ no esté activa.
17	NMI	Interrupción no enmascarable. Tiene prioridad sobre la anterior y siempre se reconoce al final de la instrucción corriente. Hace que el CPU salte a la localidad 0066H y el contenido del PC se guarda en el stack externo para que al terminar de atender la interrupción, el programa continúe donde fue interrumpido.
26	RESET	Reinicialización. Durante el tiempo de RESET, el bus de direcciones y el de datos quedan en estado de alta impedancia y las salidas del bus de control quedan inactivas. No se efectúa ciclo de refresco. Forza al PC a iniciar en 0000H y la inicialización del CPU consiste en: 1. Deshabilitar el flip-flop de habilitación de interrupción. 2. Colocar el registro I en 00H. 3. Habilitar el modo cero de interrupción.

Control del bus del CPU.

No.	Terminal	Descripción
25	BUSRQ	Peticion de bus. Indica que el bus de direcciones, el bus de datos, bus de control y salidas de tipo tres estados, se mantienen en ese estado, para que otros dispositivos puedan controlar estos buses.
23	BUSAK	Reconocimiento de peticion de bus. Se utiliza para indicar al dispositivo que su peticion es reconocida. El bus de datos, de direcciones y las señales de control de tres estados se han colocado en alta impedancia y el dispositivo, puede hacer uso de ellos o controlar dichas señales.

Bus de direcciones.

No.	Terminal	Descripción
30	A0	Bus de direcciones. Todas son salidas activas en nivel alto de tipo tres estados.
31	A1	
32	A2	
33	A3	
34	A4	
35	A5	
36	A6	
37	A7	
38	A8	
39	A9	
40	A10	
1	A11	
2	A12	
3	A13	
4	A14	
5	A15	

Bus de datos.

No.	Terminal	Descripción
14	D0	Bus de datos. Bidireccional se activan en nivel alto. Es de tipo tres estados.
15	D1	
12	D2	
8	D3	
7	D4	
9	D5	
10	D6	
13	D7	

Otros.

No.	Terminal	Descripción
6	Ø	Señal de entrada de reloj al CPU.
11	+Vcc	Señal de alimentación del CPU de +5 volts.
29	GND	Señal de entrada de referencia a tierra.

De las terminales anteriores, no todas se utilizaron en el circuito.

Debido a que la memoria RAM usada es estática, no se requiere conectar la terminal de salida RFSH del CPU.

Las señales de HALT y WAIT tampoco se usan, la primera debido a que esta instrucción no se utiliza en el programa y la segunda no se utiliza debido a que los componentes del sistema tienen un tiempo de respuesta y acceso suficiente para no requerir ciclos de espera del microprocesador.

El dispositivo que pueda interrumpir es el SID, por eso se conecta su terminal de INT con la del CPU directamente, y la terminal de NMI no se utiliza.

Como ningún dispositivo hace petición de los buses para tener el control sobre ellos, no se usan las terminales de BUSRQ y BUSAK.

Las líneas de dirección utilizadas hasta el momento van desde A0 hasta A11, suficientes para direccionar a 4K de memoria.

3.1.2 Sistema de reloj y sistema de reset.

1. Reloj.

Existen varias versiones de Z-80 como se muestra a continuación:

	Z-80	Z-80A	Z-80B	Z-80L	Z-80L-1	Z-80L-2	Z-80H
f _{máx} (MHz)	2.5	4	6	1	1.5	2.5	8
P _d (mW)	750	1000	1000	150	175	225	1000

El CPU del sistema es un Z-80, que requiere un reloj de menos de 2.5 MHz. Se utiliza una señal de 2.5 MHz que cambia de 0 a +5 V; ésta se genera por medio de un circuito oscilador construido con 2 inversores (74LS04), los cuales se realimentan y reciben la señal del cristal de 2.5 MHz.

Para una compuerta TTL, el valor típico de nivel de voltaje alto es 3.3 V y el valor mínimo de nivel alto que requiere el reloj del Z-80 es de 4.4 V, por ello se requiere colocar una resistencia de activación (pull-up) de 300 Ω.

El diagrama del sistema de reloj se muestra a continuación en la fig 3.1.2.a.

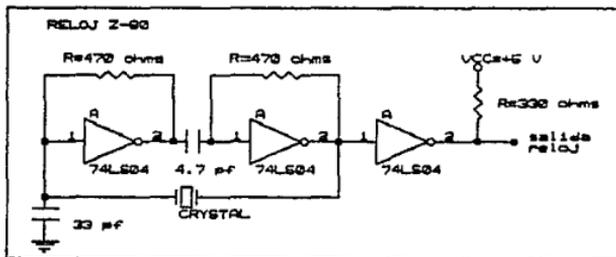


Fig.3.1.2.a. Sistema de reloj.

2. Reset.

El reset es muy importante en cualquier sistema basado en microprocesador; se requiere una reinicialización del mismo debido a que los registros del CPU tienen valores

aleatorios en el momento de alimentar al circuito y el software no puede ejecutarse correctamente. Para ello se implementa un sistema de reset, que puede ser automático, manual o una combinación de ambos, de acuerdo a las necesidades de cada circuito.

Con el reset se fuerza al PC a iniciar en cero, es decir, el contador de programa apunta a la dirección más baja de memoria, donde el usuario coloca a partir de esa localidad el programa que desea para control del sistema y cada vez que exista una reinicialización se asegura que la ejecución comienza desde 0000H.

El circuito de reset se muestra a continuación en la figura 3.1.2.b.

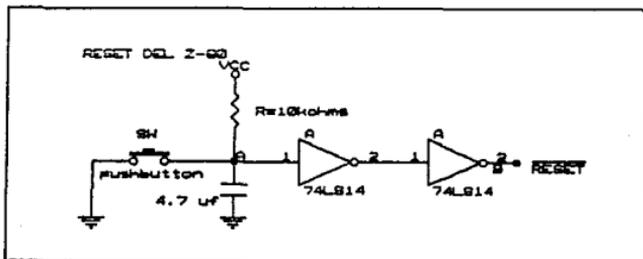


Fig.3.1.2.b. Sistema de reset.

En este caso se utiliza un reset automático, ya que ocurre después de que se alimenta el sistema y se justifica este tipo de reset porque los requerimientos de diseño así lo muestran. Sin embargo, es conveniente tener una combinación de ambos tipos, lo cual puede hacerse añadiendo un botón de reset como se muestra en la figura 3.1.2.b.

En el momento en que se aplica la potencia al circuito, la configuración RC a la entrada ocasiona que el capacitor mantenga un nivel bajo por varios milisegundos. Esta señal origina una entrada de 0 V a la primera compuerta, la segunda tiene una entrada de +5 V lo cual hace que entregue una señal de 0 V que entra a la terminal de RESET del CPU y de los dispositivos que requieren un reset, como el SIO y el generador de baudaje.

Para generar una señal limpia y de forma cuadrada, se utilizan dos compuertas 74LS14 de tipo schmitt trigger.

Si se mantiene la alimentación al circuito, el capacitor genera una salida de nivel alto lo cual hace que la señal de reset cambie a nivel alto y termina su efecto. Si se utiliza el botón de reset manual, se reinicia el sistema cada vez que se quiera sin necesidad de quitar la alimentación.

3.2 Memoria.

En el sistema, parte de la memoria la constituyen los registros del CPU, los cuales almacenan la información temporalmente. Para almacenar el programa es necesario tener un dispositivo auxiliar que es sumamente importante y se le llama memoria externa al CPU. Puede ser de dos tipos: la que sólo permite leer y la que permite leer y escribir.

3.2.1 Tipos de memoria.

1. ROM. (Read Only Memory). Es una memoria que sólo permite lectura, de manera que los datos se graban y no es posible alterarlos. La programación la realiza el fabricante por pedido del cliente.

2. PROM. (Programmable Read Only Memory). Este tipo de memoria al igual que la anterior sólo permite escribir información en ella una vez y posteriormente sólo se puede leer, pero a diferencia de la ROM, ésta puede ser programada por el usuario según sus necesidades.

3. EPROM. (Erasable Programmable Read Only Memory). Es una memoria de sólo lectura, programable y borrable, se comporta como una ROM en cuanto a la forma de acceder a la información, tiene la característica de la PROM en cuanto a que puede ser programada por el usuario. Inicialmente, la EPROM se encuentra sin programar, el usuario graba en ella lo que desea y tiene la facilidad de borrar los datos y grabar nuevamente. La más común es la UVEFROM (Ultra Violet Erasable Programmable Read Only Memory), cuyas iniciales indican que su contenido se borra con luz ultravioleta para permitir ser grabada nuevamente. La E²FROM se diferencia básicamente de la UVFROM en que se borra eléctricamente.

4. RAM. (Random Access Memory). Memoria de acceso aleatorio. Permite que los datos sean leídos o escritos en ella cuantas

veces lo requiera el sistema, pero el almacenamiento es temporal y dura mientras el circuito esté alimentado.

Existe la siguiente clasificación de memorias RAM:

1. Memoria Estática. Guarda la información de tal manera que mientras tenga alimentación, los datos se mantienen intactos.
2. Memoria Dinámica. Por el tipo de almacenamiento, va perdiendo la información a pesar de que exista alimentación, por lo que es necesario regenerar los datos o refrescarlos, lo cual se hace con la terminal RFSH del CPU.

3.2.2 Memoria utilizada en el circuito.

El circuito cuenta con capacidad de 4 Kb de memoria en total, 2 Kb son de memoria RAM y 2 Kb son de memoria EPROM.

Como se mencionó anteriormente, para almacenar las rutinas de programación en forma permanente, se requiere una memoria tipo ROM. El programa utiliza en total menos de una sexta parte de la EPROM, el resto es para alojar programas posteriores.

Para operaciones de escritura y lectura, se requiere una RAM que sirva como auxiliar del CPU. Este cuenta con un número de registros considerables que son suficientes para poder ejecutar todo el programa, pero para posibles modificaciones, la programación quedaría limitada y por tanto, el funcionamiento del circuito. La RAM se usa, en este caso, para guardar direcciones de rutinas cuando existen interrupciones, así como para almacenar datos de inicialización y comparación.

Para direccionar a 2 Kb de cada memoria se utilizan las líneas de dirección del CPU de A0-A10 y para diferenciar el tipo de memoria, se utiliza la terminal A11, de tal manera que con A0-A11 se direccionan los 4 Kb.

Los dos tipos de memoria utilizados en el circuito se describen a continuación.

3.2.2.1 Memoria EPROM 2716 (2K por 8).

La EPROM 2716 es una memoria de sólo lectura, borrable, reprogramable y estática. Está hecha con tecnología M05; consta de 2048 localidades de 8 bits cada una.

Trabaja con una alimentación de +5 V al igual que la mayoría de los circuitos utilizados.

La disipación de potencia es de 525 mW como máximo cuando está activa; en standby disipa 132 mW como máximo.

El diagrama de terminales de la memoria 2716 se muestra en la figura 3.2.2.1.a.

La programación es muy simple y rápida, se puede hacer por posicionamiento único de cada palabra, en forma secuencial o aleatoria. Los pulsos de programación son de una duración de 50 ms y las señales de control usadas son de niveles TTL, durante la programación y durante la lectura.

Después de borrar la memoria todos los bits son 1 lógico; cuando se programa ciertos bits cambian de nivel, de 1 a 0 lógico.

Los pasos de programación son los siguientes:

1. Terminal de power down y programación (PD/FGM) en nivel bajo inicialmente.
2. Terminal de Vcc conectada a +5 V y terminal de voltaje de programación (Vpp) conectada a +25 V.
3. La terminal de habilitación de chip (CS) en nivel alto.
4. Colocarse en la localidad o dirección deseada para grabar una palabra.
5. Se coloca un 0 lógico en aquellas salidas que deben tenerlo.
6. Se aplica un pulso de programación de 50 ms en la entrada (PD/FGM) de nivel alto.

Para el borrado de la 2716 se requiere exponerla a luz ultravioleta. Los pasos para borrar son los siguientes:

1. Usar luz ultravioleta de una longitud de onda de 25 a 37 Angstroms.
2. Colocar la lámpara muy cerca de la ventana de la memoria (2.54 cm).
3. Si la lámpara tiene un filtro en el tubo, retirarlo.
4. Exponer la memoria a la luz UV durante 15 o 20 minutos con una dosis de 15 W-s/cm².

Terminado lo anterior, todos los bits de la memoria son 1 lógico.

Durante la lectura, la terminal Vop debe estar en +5 V. El dato se presenta a la salida del bus cuando PD/FGM y CS están ambos en nivel bajo.

Modos de operación	PD/PGM	CS	Vpp (V)	Vcc (V)	D0-D7
Modo programación:					
Escritura		H	+25	+5	dato de entrada
Deshabilitado	L	H	+25	+5	alta impedancia
Verificación	L	L	+25	+5	dato de salida
Modo de lectura :					
Lectura	L	L	+5	+5	dato de salida
Power down	H	X	+5	+5	alta impedancia
Deshabilitado	X	H	+5	+5	alta impedancia

3.2.2.2 Memoria RAM 6116 (2K por 8).

La RAM 6116 es una memoria de lectura-escritura, hecha con tecnología CMOS, está formada de 2048 palabras de 8 bits cada una.

Trabaja con una alimentación de +5 V únicamente.

La disipación de potencia es de 180 mW en operación y en standby (no seleccionado) es de 100 mW típico.

Esta memoria es totalmente estática y no necesita señal de reloj o strobe.

Los tiempos de acceso son :120 ns para la MCM6116-12, 150 ns para la MCM6116-15 y 200 ns para la MCM6116-20.

Modos de operación de la RAM 6116:

Modo de operación	Terminales				Ciclo
	CS	OE	WE	DQ0-DQ7	
No seleccionado	H	X	X	alta impedancia	ninguno
Lectura	L	L	H	dato de salida	lectura
Escritura	L	H	L	dato de entrada	escritura
Escritura	L	L	L	dato de entrada	escritura

Tiene 2 terminales cuyas señales de entrada y salida son compatibles con TTL, sus salidas también son compatibles con la memoria EPROM de 2K por 8.

En la figura 3.2.2.a se muestra el diagrama de terminales de la memoria RAM 6116.

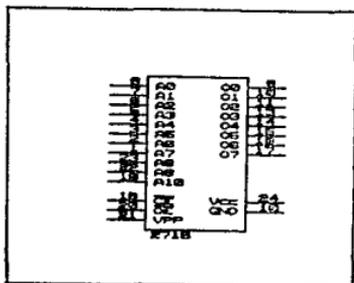


Fig. 3.2.2.1.a Memoria EPROM 2716.

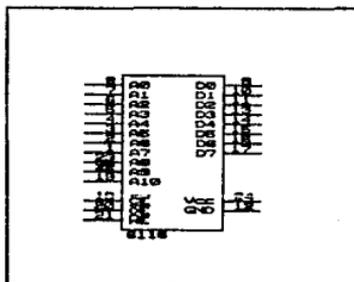


Fig. 3.2.2.2.a Memoria RAM 6116.

3.3 Circuitos de entrada y salida digital.

El circuito diseñado tiene la característica de contar con cuatro posibilidades de entrada o salida de información que constituyen las formas de comunicación con el exterior y son las siguientes:

1. Recibe información en forma digital desde el teclado del LETOVISIMNOVER, por un bus de 12 líneas en paralelo.
2. Transmite una señal FSK que es la salida de la bocina y que se acopla en forma acústica al auricular del teléfono.
3. Recibe una señal acústica por el micrófono de tipo FSK.
4. Transmite la información en forma digital por el bus de salida de 12 líneas en paralelo, que se dirigen a las entradas del LETOVISIMNOVER.

Los circuitos involucrados en los puntos 1 y 4 se explican a continuación. Los circuitos que forman parte de los puntos 2 y 3 se explican en los subcapítulos del modem y acoplamiento acústico.

1. Circuitos de eliminación de rebote y buffers limitadores de sólo salidas.

El circuito, como la mayoría de los sistemas digitales requiere de un teclado, es decir, de un conjunto de interruptores mecánicos para generar las señales de entrada. Dichos interruptores requieren especial atención debido a sus características de piezas mecánicas, en particular el fenómeno de rebote (contact-bounce) que se encuentra siempre en la mayoría de los interruptores de este tipo. Este fenómeno se explica a continuación:

Cuando se oprime o se suelta una tecla y debido a las irregularidades de las partes que hacen contacto, se presenta una serie de rebotes aleatorios en amplitud y duración; los contactos oscilan entre abiertos y cerrados antes de quedar finalmente estables. La duración del rebote varía con el tipo de interruptor.

Este fenómeno representa un gran problema en circuitos digitales porque cada rebote se toma como una señal generada tantas veces como rebotes existan. En el caso del presente diseño basado en microprocesador Z-80, el cual es capaz de ejecutar hasta 200,000 instrucciones por segundo

aproximadamente en el momento de leer este los datos del teclado, reconoce cientos de cambios de nivel, como si se oprimiera la tecla muchas veces en lugar de una sola vez.

No es posible eliminar el fenómeno, pero sí es posible bloquear su efecto hacia el circuito o bien que el circuito lo ignore. Existen dos formas de hacerlo: por hardware o por software.

a) Bloqueo de rebote por hardware.

Consiste en colocar un circuito entre las salidas del teclado y las entradas del circuito digital. Existen varias formas de lograrlo; una de las más comunes es utilizar un flip-flop de compuertas NAND para cada línea de entrada. Como se tienen 12 líneas de entrada, se utilizan 3 circuitos 74LS279 ya que cada uno tiene 4 flip-flops de tipo set-clear.

A continuación se muestra el diagrama correspondiente:

Tabla de verdad

S	C	salida
1	1	sin cambio
0	1	Q=1
1	0	Q=0
0	0	no se usa

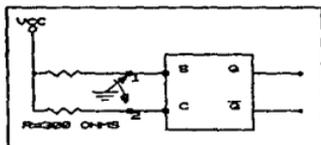


Fig. 3.3.a. Circuito utilizado para bloquear efecto de rebote.

b) Bloqueo de rebote por software.

Cuando el teclado es de buena calidad, puede utilizarse un bloqueo de rebote por medio de software. Esto se logra haciendo una rutina de retardo (generalmente cientos de ns de duración), con el fin de dar tiempo suficiente a la tecla que se oprime, de estabilizarse en un nivel. Después ya se pueden efectuar lecturas sin temor a que sean erróneas. En este caso no se utiliza este método debido a que se parte de la suposición de que el teclado será muy sencillo para disminuir costos; es por esto que se presentan graves problemas de rebote.

Como ya se mencionó anteriormente en el capítulo 2, los circuitos limitadores de sólo entradas son dos circuitos buffers unidireccionales de tipo tres estados, cada uno de los cuales tiene 8 entradas y 8 salidas en paralelo. Su diagrama de terminales se muestra en la figura 3.3.b.

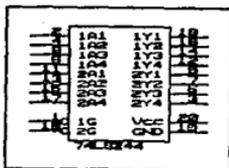


Fig. 3.3.b. Diagrama de terminales del 74LS244.

2. Circuitos que mantienen el dato de salida.

Para retener los datos a la salida se colocan dos latches de tipo tres estados, los 74LS373. Su tabla de verdad se muestra a continuación:

OUTPUT CONTROL	G	D	salida
0	H	H	H
0	H	L	L
0	L	X	Q ₀
1	X	X	Z

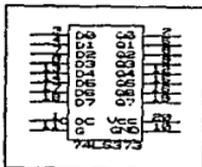


Fig. 3.3.c. Diagrama de terminales del 74LS373.

3.4 SIO Z-80.

3.4.1 Descripción general.

El SIO es un puerto serie de entrada-salida, programable, diseñado para la comunicación de datos a alta velocidad, el cual funciona como un convertidor/controlador de datos serie-paralelo y paralelo-serie. Tiene dos canales que permiten trabajar diferentes formatos para la comunicación en forma serie. Maneja la comunicación asincrónica y síncrona. Esta última es capaz de manejar los protocolos IBM Síncrono, HDLC (High Level Data Link Control), SDLC (Síncrono Data Link Control) y CRC (Cyclic Redundance Code) en cualquier modo síncrono. Es programado por el CPU para realizar las funciones.

1. Características.

- Dos canales independientes tipo full-duplex.
- Velocidad de comunicación desde 0 a 550 Kbits/seg.
- Registros de datos de recepción cuatro veces buffereados y de transmisión dos veces buffereados.
- En operación asincrónica:
 - 5, 6, 7 u 8 bits por caracter.
 - 1, 1½ o 2 bits de paridad.
 - Paridad par, impar o no paridad.
 - Operaciones de reloj x1, x16, x32 y x64.
 - Generación y detección de break.
 - Detección de errores de paridad, overrun (desbordamiento) y framing (encuadre).
- Ocho líneas de control de entrada y salida para modem.
- Lógica de interrupción de prioridades de tipo daisy chain para proveer un vector de interrupción automático sin requerir lógica externa.
- Totalmente compatible con TTL.

2. Estructura interna del SIO.

La estructura interna del SIO esta formada por: la interfase con el CPU, la lógica de control de interrupciones y dos canales full-duplex. Cada canal contiene su propio set de control, de registros de estado (escritura y lectura), lógica de control e indicadores de estado que proveen la interfase para modem u otros dispositivos externos.

Los registros para cada canal son WR0-WR7 (registros de escritura) y RR0-RR2 (registros de lectura), cada uno formado por ocho bits, cinco registros de 8 bits son de control, dos de carácter de sincronía y dos indicadores de estado, dependiendo del tipo de programación que se desee.

El vector de interrupción WR2 sólo existe para el canal B y únicamente puede ser leído por RR2 que también existe solo en canal B. Cada uno de los bits está configurado para simplificar y organizar la programación, como se observa en la figura 3.4.1.a.

3. Diagrama de bloques por canal.

Cada canal está formado por:

- Cinco registros de control de ocho bits cada uno.
- Dos registros de estado de ocho bits cada uno.
- El receptor tiene tres registros de ocho bits con arreglo FIFO, además del registro de corrimiento de entrada de ocho bits.
- El transmisor tiene un registro de ocho bits, además del registro de corrimiento de salida de ocho bits.
- Vector de interrupción solo en canal B.
- El generador/cheCADOR CRC es un registro de corrimiento de 16 bits con realimentación interna apropiada, programable para dos diferentes códigos CRC. (figura 3.4.2.b.).

4. Descripción externa del SIO.

La función de cada una de sus terminales se describe a continuación.

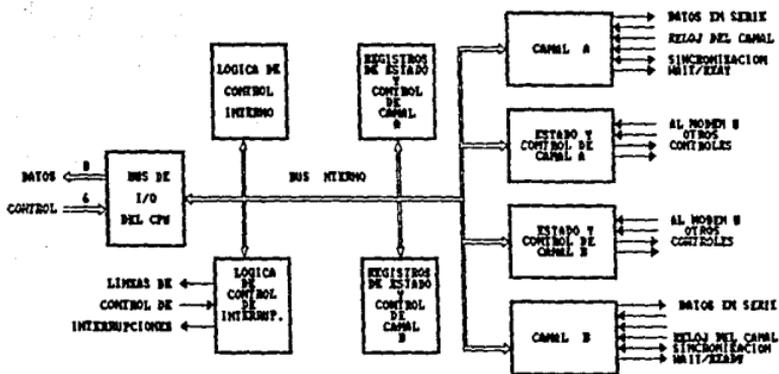


FIG. 3.4.1.a ESTRUCTURA INTERNA DEL SIO.

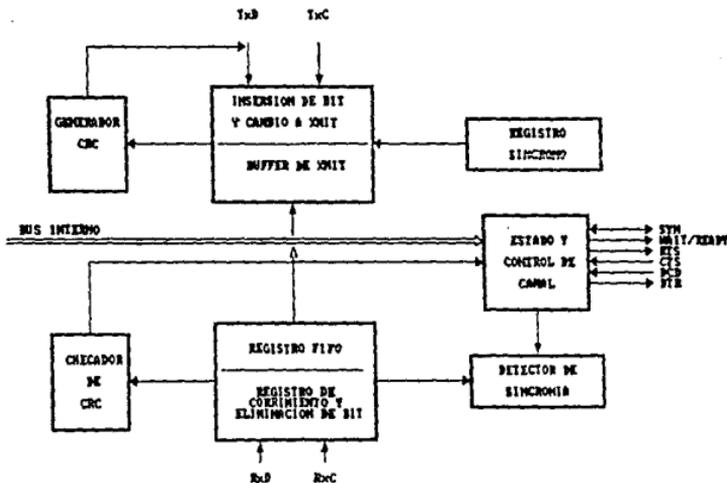


FIG. 3.4.2.b DIAGRAMA DE BLOQUES POR CANAL.

Control desde el CPU.

No.	Terminal	Descripción
35	CE	Habilitación del chip, entrada baja.
21	RESET	Deshabilita las operaciones de transmisión y recepción, los controles del modem son forzados a colocarse en estado alto, los registros de control deben ser reescritos antes de que algún dato se transmita o se reciba. Todas las interrupciones son deshabilitadas. Entrada baja.
8	M1	Inicio del ciclo de instrucción. Entrada baja.
36	IORQ	Petición de I/O. Entrada baja.
32	RD	Ciclo de lectura. Entrada baja.

Control de interrupciones.

No.	Terminal	Descripción
5	INT	Petición de interrupción. Salida baja.
6	IEI	Habilitación de interrupción de entrada. Entrada alta.
7	IED	Habilitación de interrupción de salida. IEI e IED forman la conexión daisy chain para la prioridad en el control de interrupción. Salida baja.

Bus de datos.

No.	Terminal	Descripción
40	D0	Sistema de bus de datos, bidireccional triestado. Transfiere los datos y los comandos entre el CPU y el SIO.
1	D1	
39	D2	
2	D3	
38	D4	
3	D5	
37	D6	
4	D7	

Canales A y B.

No.	Terminal	Descripción
12,28	RxDA,B	Recepción de datos en serie. Entrada baja.
13,27	RxCA,B	Reloj de recepción (1, 16, 32, 64) X la relación de datos en modo asincrónico. Entrada baja.
15,26	TxDA,B	Transmisión de datos en serie. Salida alta.
14	TxCA	Reloj de transmisión. Entrada baja.
11,29	SYNCA,B	Para una sincronización externa, el ensamble de caracteres inicia en el flanco ascendente de RxC y si es sincronizado internamente las señales son activadas durante la parte de los ciclos de reloj, en donde el caracter de sincronización es reconocido. I/O baja.
10,30	W/RA,B	El ready controla al dispositivo de Acceso Directo a Memoria (DMA) y el Wait sincroniza a los dispositivos a la velocidad del SIO. Salida, baja.

Control de modem.

No.	Terminal	Descripcion
17,24	RTSA,B	Peticion para mandar, si esta en estado alto, entonces pasara al estado bajo. Si esta en estado bajo en el modo asincrono pasará al estado alto, pero solo despues de que el transmisor está vacío. Salida baja.
18,23	CTSA,B	Borrar para mandar, si se encuentran programadas en autohabilitacion, habilitan al transmisor, de no ser así, estas pueden ser programadas como entradas de proposito general. Entrada baja.
16,25	DTRA,B	Terminal de datos lista, sigue al estado programado por el bit DTR. Entrada baja.
19,22	DCDA,B	Detector de portadora de datos, funciona igual que CTS e inhiben a la recepcion. Salida baja

Otras.

No.	terminal	descripcion
33	C/D	Selección: datos (0 lógico) o control (1 lógico). Entrada baja para datos.
34	B/A	Selección: canal A (0 lógico) o canal B (1 lógico). Entrada baja para canal A.
20	Ø	Reloj del sistema. Entrada.
9	+5 V	Alimentación de CD. Entrada.
31	GND	Referencia de niveles de DC (tierra). Entrada

3.4.2 Generador de baudaje MC14411.

Como ya se mencionó anteriormente, para generar la frecuencia de transmisión y recepción del SIO, se utiliza un generador de baudaje también llamado generador de bit-rate.

El MC14411 está hecho con tecnología CMOS LSI. Contiene una red divisora de frecuencia para generar sus diferentes salidas.

Se tiene la posibilidad de programar las salidas multiplicadas por un factor (x1, x8, x16 y x64), de la siguiente manera:

Terminal		factor
RSB	RSA	
0	0	x1
0	1	x8
1	0	x16
1	1	x64

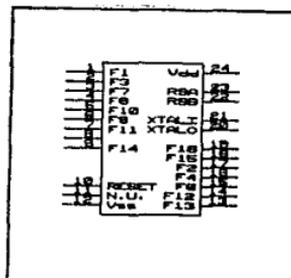


Fig. 3.4.2.a. Terminales del MC14411.

Puede generar 16 diferentes frecuencias para cada factor seleccionado.

Trabaja con una alimentación de +5 V. Sus salidas cuentan con buffers y son compatibles con TTL LS.

La terminal de reset se conecta a una resistencia de 100 Kilohms y Vcc, para mantener un nivel alto mientras trabaja; pero se conecta directamente al circuito de reset del sistema para que se reinicialice al mismo tiempo que el resto de los circuitos.

Como se trabaja a una velocidad de 300 bauds y debido a que el SIO se programa por un factor de 16, el factor que se requiere para el generador de baudaje es x1; se requiere conectar la terminal 2 correspondiente a F2 como entrada de reloj de transmisión y recepción del SIO. La terminal RSA y RSB se conectan a tierra.

3.5 Direccionamiento de dispositivos.

En la tabla 3.5.a se muestra la forma de direccionar los diferentes dispositivos que controla el CPU.

La tabla consta de cuatro secciones donde se ilustra cómo el CPU selecciona un dispositivo determinado o bien a un elemento dentro de dicho dispositivo, como es el caso del direccionamiento de las localidades de memoria o de canales del SIO.

La primera sección, que es el direccionamiento de los puertos de ENTRADA/SALIDA funciona como sigue: La línea de control IORQ se activa con un "0", esta línea a su vez activa al decodificador 74LS138. Siempre que IORQ está habilitada, la línea MREQ se encuentra deshabilitada, es decir se solicita un puerto de ENTRADA/SALIDA o se solicita una localidad de memoria. el direccionamiento se realiza con las líneas A0, A1, y A6 del bus de direcciones, para formar las direcciones correspondientes a cada puerto. Es conveniente mencionar que para los FE01 y FE02 (Dirección 40H Y 41H respectivamente) solamente se efectúan operaciones de lectura y para los puertos PS01 y PS02 (Dirección 42H y 43H) sólo se efectúan operaciones de escritura.

La segunda sección corresponde al SIO Z-80, la salida de control IORQ del CPU Z-80 se conecta a la entrada de una compuerta OR 74LS32, la otra entrada de esta compuerta se conecta a la línea negada A7 del bus de direcciones; la salida de esta compuerta se conecta a la terminal de habilitación del SIO, la conexión del resto de las terminales entre el CPU y el SIO Z-80 se muestran a continuación:

Z-80 CPU	Z-80 SIO
RD	RD
IDRQ	IDRQ
A0	C/D
A1	B/A

De manera que las direcciones del SIO Z-80 son como sigue:

DIRECCION	PUERTO DEL Z-80 SIO
80H	A DATO LECTURA/ESCRITURA
81H	A CONTROL LECTURA/ESCRITURA
82H	B DATO LECTURA/ESCRITURA
83H	B CONTROL LECTURA/ESCRITURA

Se debe recordar que el puerto B no se utiliza.

La tercera sección corresponde a la memoria EPROM 2716 la cual se selecciona con MREQ, RD y de A0 a A10 para

LINEAS DE CONTROL DEL Z-80 CPU		LINEAS DEL BUS DE DIRECCIONES										DECODIF. DIRECCION 74LS 138	FUERTO DIREC.											
IORQ	MREQ	RD	WR	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	7	6	5	4	
0	1	0	1	X	X	X	X	X	X	X	X	0	1	0	0	0	0	0	0	1	1	1	0	PE01
0	1	0	1	X	X	X	X	X	X	X	X	0	1	0	0	0	0	0	0	1	1	0	1	PE02
0	1	1	0	X	X	X	X	X	X	X	X	0	1	0	0	0	0	0	1	0	1	1		PS01
0	1	1	0	X	X	X	X	X	X	X	X	0	1	0	0	0	0	0	1	0	1	1		PS02
SIO Z-80																								
0	1	0	1	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	0	OPERACION PUERTO
0	1	1	0	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	0	LECTURA A DATO
0	1	0	1	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	1	ESCRITURA A DATO
0	1	1	0	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	1	LECTURA A CTRL
0	1	1	0	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	1	ESCRITURA A CTRL
MEMORIA EPROM 2716																								
1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LECT. LOC.0000H
1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	LECT. LOC.0001H
1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	LECT. LOC.0002H
.
.
.
1	0	0	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	LECT. LOC.07FFH
MEMORIA SRAM 6116																								
1	0	X	X	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LEC/ESC LOC. 0800H
1	0	X	X	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	LEC/ESC LOC. 0801H
1	0	X	X	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	LEC/ESC LOC. 0802H
.
.
.
1	0	X	X	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	LEC/ESC LOC. 0FFFH

Tabla 3.5.a Tabla de direccionamiento de dispositivos.

direccionar 2Kbits de memoria. Para habilitar esta memoria se utiliza MREQ y A11 que pasan por una compuerta OR (74LS32), cuya salida se conecta a la terminal de habilitacion de la memoria.

La última sección corresponde a la memoria SRAM 6116 que se habilita con la salida de una compuerta OR (74LS32), cuyas entradas son A11 negada y MREQ. Dependiendo de la operación que se quiera realizar, se habilita el modo de lectura o en su defecto el modo de escritura por medio de la terminal WR.

3.6 Condiciones estáticas y dinámicas del circuito.

Para asegurar el funcionamiento óptimo del circuito, es necesario hacer dos tipos de análisis: estático y dinámico. Dichos análisis se realizan por separado; el primero se hace considerando las condiciones de carga, principalmente de circuitos de tipo LSI, que toleran menos cargas; el segundo se hace considerando al circuito funcionando para diferentes tiempos.

3.6.1 Análisis dinámico.

Tiene como objetivo asegurar que los circuitos conectados entre sí trabajen o respondan justo en el momento requerido y durante el tiempo necesario.

Como su nombre lo indica, las consideraciones dinámicas del circuito se hacen en base a los tiempos de respuesta y/o acceso de sus elementos.

Es claro que para realizar todo el análisis dinámico, debe tratarse cada circuito interactuando con otro como un bloque aparte, para ello se consultan sus diagramas de tiempo durante los ciclos en que su funcionamiento coincide.

En este caso, el microprocesador controla pocos elementos, por lo que el análisis se simplifica bastante y consta de las siguientes partes:

1. Acoplamiento dinámico del CPU Z-80 y la memoria EPROM 2716.
2. Acoplamiento dinámico del CPU y la memoria RAM 6116.
3. Acoplamiento dinámico del CPU Z-80 y circuitos TTL.
4. Acoplamiento dinámico de los circuitos restantes.

1. Acoplamiento dinámico del CPU Z-80 y la memoria EPROM 2716.

Antes de conectar la memoria EPROM al CPU, es importante considerar que esta debe tener presentes los datos en el instante que lo requiera el microprocesador; en caso de que la memoria sea más lenta que este, se deben añadir ciclos de espera (WAIT) al CPU para sincronizarlo con dicha memoria. La memoria EPROM debe mantener los datos hasta que el CPU los ha leído.

En base a los diagramas de tiempo se hace el siguiente análisis (fig. 3.6.1.a):

Si el CPU trabaja a 2.5 MHz. su periodo de reloj es de 400 ns.

Sólo importa comparar el ciclo de lectura del CPU y de la memoria. Cuando existe un requerimiento de memoria, se coloca una dirección en el bus de direcciones. La señal MREQ y en este caso, la señal RD se ponen en nivel bajo, esto da como resultado que la terminal de CS que habilita a la memoria se ponga en nivel bajo. En estas condiciones, la memoria presenta los datos válidos en el bus después de 450 ns máximo, que es su tiempo de acceso máximo. El CPU durante su ciclo de lectura, hace el muestreo del bus de datos 1000 ns después de que el bus de direcciones tiene una localidad de memoria válida.

La memoria tiene un tiempo de respuesta más rápido al muestreo del microprocesador, por lo tanto, si cumple las condiciones dinámicas y es correcto conectarla al CPU sin necesidad de añadir ciclos de espera.

2. Acoplamiento dinámico del CPU Z-80 y la memoria RAM 6116.

Se requiere hacer un análisis tanto para el ciclo de lectura como para el de escritura.

En base a los diagramas de tiempo se puede hacer un análisis similar al anterior:

a) Ciclo de lectura.

Se comparan los ciclos de lectura del CPU y la memoria RAM, como se muestra en la figura 3.6.1.b.

El análisis es el siguiente:

Quando existe una instrucción de lectura de memoria, el microprocesador coloca la dirección correspondiente en su bus, poco después de la mitad del primer ciclo de reloj la terminal MREQ y RD se colocan en nivel bajo.

La memoria puede trabajar con tres ciclos de lectura

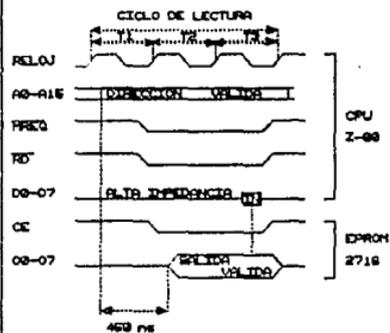


FIG. 3.8.1.a. ACOPLAMIENTO DINAMICO ENTRE LA MEMORIA EPROM 2716 DURANTE EL CICLO DE LECTURA.

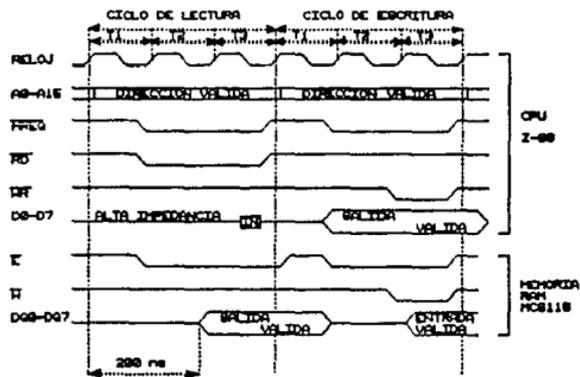


FIG. 3.8.1.b. ACOPLAMIENTO DINAMICO ENTRE EL CPU Y LA MEMORIA RAM DURANTE EL CICLO DE LECTURA.

FIG. 3.8.1.c. ACOPLAMIENTO DINAMICO ENTRE EL CPU Y LA MEMORIA RAM DURANTE EL CICLO DE ESCRITURA.

diferentes, en este caso se utiliza el ciclo 3. El tiempo de acceso máximo es de 200 ns y el tiempo que tarda ésta en colocar en su bus el dato válido desde que se habilita, es de 200 ns. Debido a que el microprocesador controla la habilitación y lectura de la memoria por medio de las terminales MRQ y WR respectivamente, realiza el muestreo del bus de datos sin ningún problema, ya que esto ocurre después de 1000 ns. Por lo anterior, se concluye que el CPU no requiere ciclos de espera y se dice que la memoria no presenta problemas dinámicos durante el ciclo de lectura a ella.

b) Ciclo de escritura.

Para el ciclo de escritura, se comparan los ciclos de escritura del CPU y de la memoria RAM, los diagramas se muestran en la figura 3.6.1.c.

El análisis es el siguiente:

Cuando existe una instrucción de escritura a memoria, el microprocesador coloca la dirección correspondiente en su bus, poco después de la mitad del primer ciclo de reloj, la terminal MREQ se pone en nivel bajo y las terminales del bus de datos del microprocesador mantienen el dato válido de salida. Poco después de ciclo y medio de reloj, la terminal de WR se pone en nivel bajo. Con la línea de dirección A11 y MRQ se habilita la memoria y con la señal de WR se habilita el modo de escritura en ella. La memoria puede trabajar con dos ciclos de escritura diferentes, en este caso se utiliza el ciclo 2; debido a que el tiempo máximo de acceso es de 120 ns y presenta el dato de salida casi inmediatamente después de que ha sido habilitada, se concluye que el microprocesador tiene tiempo para escribir el dato en la memoria, ya que además el control de habilitación y escritura lo realiza este. Por lo anterior no se requiere colocar ciclos de espera en el microprocesador.

3. Acoplamiento dinámico del CPU Z-80 con los puertos de I/O circuitos TTL.

Debido a que el SIO es un puerto de I/O de la familia del Z-80, no existe problema alguno en el acoplamiento dinámico. Sus señales son totalmente compatibles y el SIO responde tan rápido como el CPU lo requiere, ya que la señal de reloj es común a ambos. A pesar de que el SIO trabaja a una velocidad de transmisión de 300 bauds, se hace una rutina de programación con el fin de asegurar que el CPU se sincronice con el manejo de los datos que realiza el SIO durante la recepción (ver cap. 4).

Los puertos de entrada y salida son los circuitos buffers 74LS244 y los latches 74LS373, cuyo tiempo de acceso es suficiente para responder al CPU y no presentan problemas de acoplamiento dinámico.

Se tienen los siguientes circuitos conectados directamente al CPU, cuyo tiempo de acceso se muestra a continuación con el fin de verificar que no presentan problemas de acoplamiento dinámico:

Circuito		Tiempo de prop. max (ns)
Decodificador	74LS138	41
Buffers unidirecc.	74LS244	18
Buffers bidirecc.	74LS245	12

4. Acoplamiento dinámico del resto de los circuitos.

Debido a que se utilizan circuitos TTL LS, no se tienen problemas de acoplamiento dinámico entre ellos.

Los circuitos utilizados en la etapa de modulación son compatibles con TTL. El XR-2211 y el XR-2206 tienen un rango de operación bastante amplio y tiempos de respuesta apropiados para el SID y los circuitos amplificadores.

Los circuitos empleados en la etapa de amplificación tienen un rango de operación apropiado, ya que se utilizan especialmente para bajo ruido y para el manejo de señales por vía telefónica.

3.6.2 Análisis estático.

En este análisis se examinan las condiciones de carga de todas las salidas, en cada circuito integrado, tanto en los ciclos de lectura como en los de escritura, lo anterior se realiza tanto para el nivel alto como para bajo respectivamente.

El trabajo se simplifica probando solo las salidas más susceptibles de presentar problemas, porque las salidas de los circuitos de gran escala de integración (LSI), regularmente toleran hasta 4 cargas LS (baja escala de integración).

Por convención, se toman como positivas las corrientes que entran al circuito y como negativas las que salen.

tipos de cargas:

- 1 - Entrada MOS : Según especificaciones, representa una carga de +10 μ A.

- 2 - Salida Tres Estados :Representan una carga de +20 μ A en estado alto y de -20 μ A en estado bajo.
- 3 -Entrada TTL : Representa una carga de 20 μ A en estado alto y -0.4 mA en estado bajo.

El valor de los parámetros antes descritos pueden verificarse en los manuales de operación de los circuitos correspondientes. Conocidas las cargas que intervienen, se utiliza la ley de Suma de corrientes de Kirchoff. Si la corriente consumida por los circuitos es menor que la suministrada por el circuito fuente, no existirán problemas con la carga, de lo contrario existiran problemas y para solucionarlos se tendrá que colocar un buffer, el cual es un dispositivo que provee una corriente mayor a la suministrada por una salida LSI.

En el diagrama electrónico se observa la existencia de posibles problemas en el bus de datos del sistema. Es por ello que en esta área se realiza el análisis estático y de una forma simplificada se muestra en la figura 3.6.a.

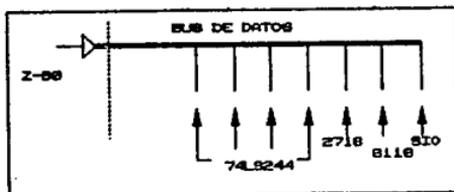


Fig. 3.6.a Diagrama de carga del bus de datos.

Análisis en estado alto:

Para realizar el análisis en estado alto, las condiciones de carga se muestran en la figura 3.6.b.

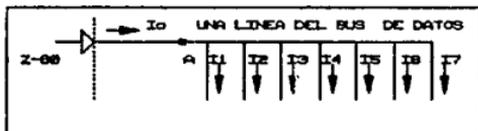


Fig. 3.6.b Diagrama de carga en estado alto.

Aplicando la primera Ley de Kirchoff en el punto A.

$$I_o - I_1 - I_2 - I_3 - I_4 - I_5 - I_6 - I_7 = 0$$

$$I_o = I_1 + I_2 + I_3 + I_4 + I_5 + I_6 + I_7$$

$$I_1 = I_2 = I_3 = I_4 = -20 \mu A$$

$$I_5 = I_6 = I_7 = -10 \mu A$$

$$I_o = 4(-20) + 3(-10) = -110 \mu A$$

Como la corriente es de signo negativo y de acuerdo a la convención antes mencionada, la corriente sale del circuito.

Del manual técnico de datos del Z-80 se sabe que la corriente máxima que suministra éste en su bus de datos, en estado alto es de $-250 \mu A$ (IOH). Como la corriente que requieren las cargas es menor a la proporcionada por el Z-80, se concluye que es necesario colocar buffer en este estado.

Analisis en estado bajo:

Para realizar el analisis en estado bajo, las condiciones de carga se muestran en la figura 3.6.c.

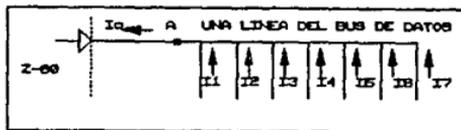


Fig. 3.6.c Diagrama de carga en estado bajo.

Aplicando la primera Ley de Kirchoff en el punto A.

$$- I_o + I_1 + I_2 + I_3 + I_4 + I_5 + I_6 + I_7 = 0$$

$$I_o = I_1 + I_2 + I_3 + I_4 + I_5 + I_6 + I_7$$

$$I_1 = I_2 = I_3 = I_4 = 20 \mu A$$

$$I_5 = I_6 = I_7 = 10 \mu A$$

$$I_o = 4(20) + 3(10) = 110 \mu A$$

Del manual técnico de datos del Z-80 se sabe que la corriente máxima que suministra éste en su bus de datos en estado bajo es de $1.8 mA$ (IOL). Como la corriente que

requieren las cargas es menor a la proporcionada por el Z-80. se concluye que no es necesario colocar buffer en este estado.

El resultado de este análisis es que el sistema no necesita de los buffers, pero se colocarán porque el sistema tiene la posibilidad de sufrir una expansión en el futuro, además cuando se esté en la etapa de pruebas es necesario proteger los circuitos mas valiosos de posibles errores en el armado del sistema, porque se puede causar algun corto circuito y dañarse alguno de ellos.

3.7 Modulación y Demodulación.

3.7.1 Proceso de Modulación y Demodulación.

Debido a que la línea telefónica fué diseñada para la transmisión de la voz, es decir, señales analógicas, no es posible transmitir información digital. Para poder transmitir datos a través de los circuitos telefónicos, es necesario convertir los datos digitales a una forma que les permita ser transmitidos más facilmente por los circuitos y al final de ésto regresarlos a su forma original para su uso.

En un extremo de la línea telefónica se lleva a cabo un proceso conocido como MODULACION, el cual coloca la información digital en la portadora. En el otro extremo de la línea, otro proceso conocido como DEMODULACION toma la información de la portadora (fig. 3.7.1.a).

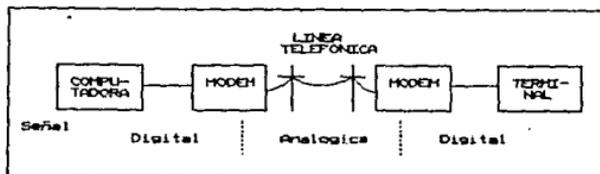


Fig. 3.7.1.a Proceso de modulación y demodulación.

El proceso de modulación y demodulación de datos digitales es realizado por un modem, la palabra MODEM es la contracción de las palabras MODulador-DEModulador.

La línea telefónica tiene aproximadamente un ancho de banda de 3000 Hz (300 Hz - 3400 Hz), por lo tanto, las señales moduladas deben estar contenidas dentro de este rango.

3.7.2 Tipos de modulación.

Para que una señal sea trasladada en frecuencia a otra parte del espectro, es necesario que la señal varíe una de las características de una onda (generalmente una onda senoidal) conocida con el nombre de PORTADORA, cuya frecuencia ocupa esa parte del espectro. El proceso por el que una de las características de la onda portadora queda modificada de acuerdo con las características de la señal a trasladar (MODULADORA) recibe el nombre de MODULACION (fig. 3.7.2.a).

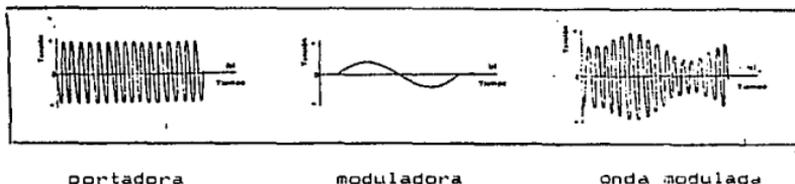


Fig. 3.7.2.a Proceso de modulación.

La señal utilizada como portadora generalmente es del tipo senoidal o digital, la primera se utiliza para enviar solo un mensaje por un canal, la segunda cuando se desea transmitir más de una conversación por un solo canal utilizando el método de multiplexaje por división de tiempo.

La señal moduladora puede ser del tipo analógico o digital, es decir, continua o discontinua en el tiempo, la combinación de estas características genera la mayoría de los tipos de modulación utilizados, en este caso solo se explicarán brevemente los tipos de modulación con portadora analógica los cuales se muestran en la tabla 3.7.2.a, los demás tipos de modulación no se explicarán por que se utilizan para transmitir más de una conversación por canal y esto queda fuera del objetivo de esta tesis.

PORTADORA	MODULADORA	PARAMETRO	NOMBRE
Sencoidal	Analógica	Amplitud	Amplitud Modulada
		Frecuencia	Frecuencia Modulada
		Fase	Fase Modulada
	Digital	Amplitud	Encendido y apagado
		Frecuencia	Corrimiento de frecuencia (FSK)
		Fase	Corrimiento de Fase

Tabla 3.7.2.a Tipos de modulación.

La expresión general para una onda portadora sencoidal

$$v(t) = A \text{ sen}(wt + \theta)$$

Donde :

- $v(t)$ = Voltaje instantaneo de la onda.
- A = Amplitud máxima de la onda.
- w = Velocidad angular de la onda en rad/seg.
- θ = Fase de la onda en el instante cero.

Para lograr un proceso de modulación se pueden modificar tres variables:

- Amplitud.
- Frecuencia.
- Fase.

1. Modulación con moduladora Analógica.

En esta parte se explicaran brevemente las modulaciones que utilizan la portadora sencoidal y moduladora analógica.

a. Modulación en Amplitud (A.M.).

En este caso la frecuencia de la onda portadora permanece constante, pero su amplitud crece o decrece dependiendo de la forma de la señal moduladora (fig 3.7.2.b).

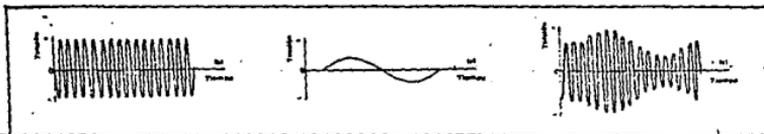


Fig 3.7.2.b Modulación en Amplitud.

Se emplea en sistemas de telefonía, telegrafía, radiodifusión en bandas media y corta, en señales de imágenes de sistemas de televisión etc.

b. Modulación en frecuencia (F.M.).

En este caso la amplitud permanece constante y la frecuencia varía dependiendo de la forma de la señal a modular (fig. 3.7.2.c).

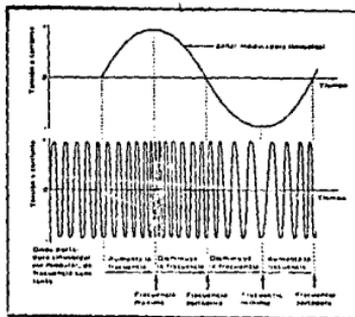


Fig. 3.7.2.c Modulación en Frecuencia.

Se utiliza en radio difusión en V.H.F. (Very High Frequency), en el sonido de la televisión y para algunos sistemas radio-telefónicos.

La modulación en frecuencia tiene la ventaja sobre la modulación en amplitud, de que es menos inmune a la interferencia del ruido, porque es más difícil de modificar la frecuencia.

C. Modulación en fase (P.M.).

En este caso se mantiene constante la amplitud y frecuencia, sólo varía la fase de la señal dependiendo de los cambios que sufra la señal moduladora. El inconveniente de este sistema es el grado de complejidad del circuito, debido a esto, es poco utilizado.

2. Modulación con moduladora digital.

En esta parte se explican brevemente los tipos de modulación que utilizan como portadora una onda senoidal y como moduladora una señal digital.

a. Modulación por corrimiento de amplitud. (ASK Amplitud Shift Keying).

La forma de onda de los datos puede utilizarse para conmutar una onda portadora de frecuencia apropiada para formar la onda mostrada en la figura 3.7.2.d.

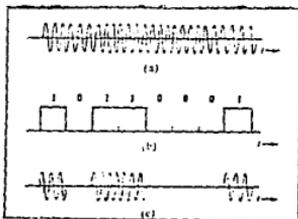


Fig. 3.7.2.d Modulación ASK.

El proceso de conmutación de unos a ceros lógicos genera un gran número de frecuencias laterales, para limitar el ancho de banda ocupado, la señal debe pasar a través de un filtro. La limitación del número de frecuencias laterales transmitidas aumenta tanto tiempo como le tome a la envolvente alcanzar su máximo valor. Cuando mayor sea la velocidad de transferencia de datos, más redondeada será la envolvente (fig. 3.7.2.e).

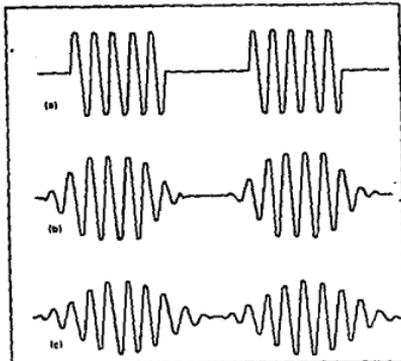


Fig. 3.7.2.e Distorsión que sufre ASK.

- a. Baja velocidad.
- b. Velocidad media.
- c. Alta velocidad.

Los inconvenientes del método, residen en la exactitud de la señal recibida, la cual depende mucho de la relación señal-ruido del enlace, y que cualquier interrupción momentánea en la vía de transmisión se interpreta como un cero y entonces aparecen errores en los datos recibidos.

b. Modulación por corrimiento de frecuencia.
(FSK Frequency Shift Keying).

La modulación por corrimiento de frecuencia de una portadora es una versión de la modulación de frecuencia (FM), solamente que aquí la frecuencia de la portadora puede tener dos valores, uno de los cuales se utiliza para representar al símbolo lógico cero y otro para el uno lógico.

Cuanto más rápida sea la transferencia necesaria de datos, más separadas deben estar las frecuencias (fig. 3.7.2.f).

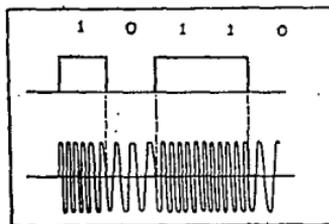


Fig. 3.7.2.f Modulación FSK.

c. Modulación en fase.
(PSK Phase Shift Keying).

En estos casos se mantiene constante la amplitud y frecuencia, sólo se varía la fase de la señal dependiendo de los cambios que sufra la señal moduladora. En sistemas digitales es posible representar los datos binarios (ceros y unos), cambiando la fase de una onda portadora de frecuencia constante entre dos valores específicos (fig. 3.7.2.g). Este método de modulación exige que el extremo receptor del sistema tenga una onda de referencia de exactamente la misma frecuencia y fase constante, pero esto implica algunas dificultades prácticas considerables, por lo tanto, se utilizan raramente (sólo en sistemas de alta velocidad).

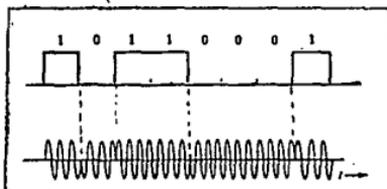


Fig 3.7.2.g Modulación PSK.

3.7.3 Elección del tipo de modulación.

Se puede utilizar tanto modulación en amplitud como en frecuencia, pero a menudo el mejor camino para modular es FSK (Frequency Shift Keying, desplazamiento en frecuencia), debido a las siguientes ventajas que ofrece:

- La modulación FSK es sencilla y de bajo costo.
- Es básicamente una técnica de modulación de frecuencia, por lo que la señal transmitida puede ser amplificada por el receptor sin control automático de ganancia.
- Presenta una mayor inmunidad al ruido con respecto a la modulación en amplitud.

Debido a estas ventajas, la modulación FSK tiene mayor uso en la transmisión de datos por línea telefónica a velocidades medias y bajas. Por lo tanto, este tipo de modulación será utilizado en el diseño.

3.7.4 Modos de comunicación.

Un modem puede transmitir y recibir información en uno de los tres modos siguientes:

- Simplex (Unidireccional).
- Half Duplex (Bidireccional no simultáneo).
- Full Duplex (Bidireccional simultáneo).

a. Simplex.

En este modo de operación, los datos son enviados en una sola dirección, lo que limita el dispositivo a transmitir o recibir. La radio y la televisión son algunos ejemplos de este tipo de transmisión (fig. 3.7.4.a).



FIG. 3.7.4.a Modo Simplex.

b. Half Duplex.

En este modo los datos son enviados en dos direcciones pero NO simultáneamente. Un ejemplo típico es la radio Banda

Civil (CB), en donde el operador puede transmitir y recibir, pero no puede hacerlo simultáneamente por un mismo canal; al final de su transmisión, es necesario que se avise al otro operador que se terminó de transmitir y que está listo para recibir (fig. 3.7.4.b).

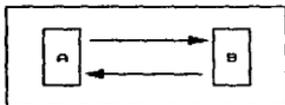


Fig. 3.7.4.b Modo Half Duplex.

c. Full Duplex.

En este modo, la transmisión se realiza en ambas direcciones simultáneamente. Para que la radio CB funcione en este modo se necesitan dos canales, uno para transmitir y otro para recibir (fig 3.7.4.c).

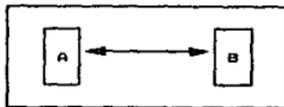


Fig. 3.7.4.c Modo Full Duplex.

3.7.5 Elección del modo de comunicación.

El objetivo del presente trabajo es diseñar un dispositivo que permita reincorporar a la sociedad a personas con problemas auditivos, es decir, estas personas podrán recobrar la estructura de la comunicación. Serán capaces de transmitir y recibir mensajes (hablar y oír respectivamente). Pero como toda persona normal para entender un mensaje o transmitir una idea, se debe escuchar, después hablar o hablar y después escuchar, pero nunca realizar las dos acciones al mismo tiempo, porque esto trae como consecuencia en la mayoría de los casos la deformación de la información emitida.

Esta característica de la comunicación, define que se debe realizar una comunicación en modo HALF DUPLEX, es decir, en ambas direcciones pero no en forma simultánea (fig. 3.7.5.a).

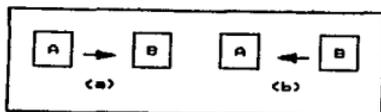


Fig. 3.7.5.a Half Duplex de una comunicación.
a.- A habla, B escucha.
b.- B habla, A escucha.

3.7.6 Requisitos de las normas para la transmisión de datos por la red telefónica.

Los equipos de comunicación de datos (modems) son de gran utilidad y debido a esto existe una gran diversidad de diseños de estos dispositivos.

Para que cualquiera de éstos pueda acoplarse indistintamente en cualquier lugar, es necesario que todos cumplan con ciertas normas que son emitidas por el Comité Consultivo Internacional de Telegrafía y Telefonía (CCITT), el cual es una parte de la Unión Internacional de Telecomunicaciones en Ginebra.

El objetivo de esta tesis es diseñar un dispositivo que será utilizado en el área de rehabilitación de personas con problemas auditivos, no en la transmisión de información entre computadoras, donde la velocidad de la información es grande y la transmisión de datos complicada.

Esto trae como resultado un diseño diferente a los sistemas comerciales utilizados hasta el momento, debido a esto la mayoría de los parámetros se designan en base a los requerimientos y necesidades del sistema, utilizando las normas del CCITT como apoyo.

3.7.7 Clasificación de las velocidades de transmisión.

El Baudio es la unidad de medida de la velocidad de transmisión de datos y es específicamente Bits Por Segundo

(BPS). Describe el número de datos binarios que pueden ser transmitidos por segundo.

La clasificación utilizada es la siguiente;

Baja velocidad : 0 - 1200 BPS
Media velocidad: 2400 - 9600 BPS
Alta velocidad : más de 9600 BPS

Como el dispositivo diseñado está encaminado a la comunicación de personas con problemas auditivos no es necesaria una gran velocidad en la transmisión de datos, se escogió la velocidad de 300 BPS por ser fácil de obtenerse a partir de los circuitos utilizados.

3.7.8 Diseño del modulador.

El dispositivo usado como modulador, es el circuito integrado XR-2206. Es un generador de funciones capaz de producir formas de onda senoidal, cuadrada, triangular, rampas y pulsos de alta calidad. Las formas de onda de las salidas, pueden ser moduladas en frecuencia o en amplitud. Las frecuencias de operación, se seleccionan externamente sobre un rango de 0.01 Hz a 1 MHz. Este circuito fue diseñado especialmente para comunicaciones, instrumentación y otras aplicaciones que requieren generar modulación en amplitud o en fase.

Como se muestra en la figura 3.7.8.a el circuito está compuesto por cuatro partes principales:

- 1 - Oscilador Controlado por Voltaje (VCO).
- 2 - Formador de Onda Senoidal.
- 3 - Buffer de ganancia unitaria.
- 4 - Interruptores de corriente.

Los interruptores de corriente, transfieren la corriente del oscilador a una de las dos resistencias externas, las cuales se utilizan para producir las frecuencias seleccionadas por el nivel lógico en la terminal de entrada.

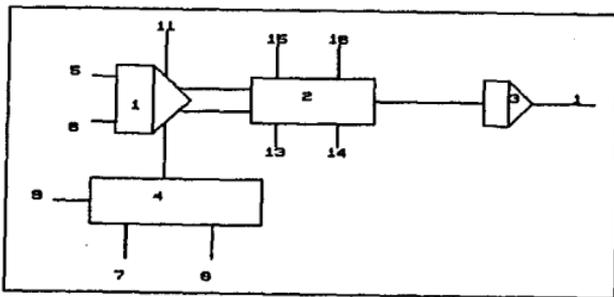


Fig. 3.7.B.a Diagrama de bloques del XR-2206.

1 Descripción de los elementos de control (fig.3.7.8.b).

a. Frecuencia de Operación.

La frecuencia de oscilación (f_0), es fijada por un capacitor externo (C), a través de las terminales 5 y 6 y por una resistencia (R), conectada a las terminales 7 u 8. La frecuencia es controlada por la siguiente relación.

$$f_0 = 1/(RC)$$

Puede ser ajustada variando la resistencia R o el capacitor C. Para obtener una óptima estabilidad contra la temperatura se recomienda que R se encuentre en el rango de 4 a 200 K Ω . Los valores recomendados para C son:

de 100 pf a 100 uf.

b. Amplitud de Salida.

La máxima amplitud de salida es directamente proporcional a la resistencia externa (R_3) conectada en la terminal 3, para la salida senoidal. la amplitud es aproximadamente de 60 mV por K Ω de R_3 . Para la onda triangular la amplitud de pico es aproximadamente de 10 mV de pico del valor de R_3 .

c. Desplazamiento de Frecuencias (FSK).

El XR-2206 puede operar con dos resistencias diferentes R1 y R2, conectadas a las terminales 7 y 8 respectivamente como lo muestra la figura 3.7.8.c.

Dependiendo de la polaridad de la señal lógica en la terminal 9, se activará una de las dos resistencias (R1 o R2). Si el voltaje en la terminal 9 se deja sin conectar o es conectada a un voltaje mayor o igual a 2 volts, sólo se activará R1. Similarmente, si el nivel de voltaje en la terminal 9 es menor o igual a 1 volt, sólo se activará R2. De esta manera, las frecuencias de salida pueden ser manipuladas entre los dos niveles f1 y f2. Las cuales se definen como:

$$f_1 = 1/(R_1 \cdot C)$$

$$f_2 = 1/(R_2 \cdot C)$$

d. Ajuste Externo.

El contenido de armónicos de la salida senoidal puede reducirse a un 0.4%, por medio de ajustes adicionales como se muestra en la figura 3.7.8.c. El potenciómetro RA ajusta la forma de la onda senoidal y RB provee un ajuste fino para la simetría de onda. El procedimiento de ajuste es el siguiente:

- 1.- Fijar RB en el punto medio y ajustar RA para mínima distorsión.
- 2.- Con RA fija, como se especifica en el punto 1, ajustar RB para una mayor reducción de la distorsión.

RA se conecta entre las terminales 13 y 14 y RB entre las terminales 15 y 16.

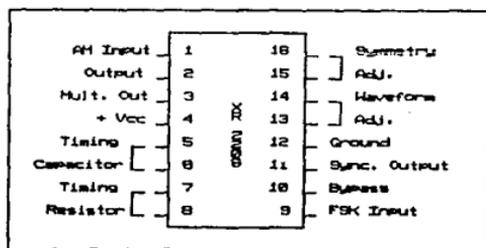


Fig. 3.7.8.b Diagrama de terminales de XR-2206.

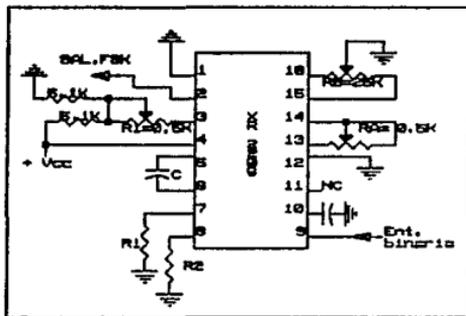


Fig. 3.7.8.c Configuración como modulador FSK del XR-2206.

2. Aplicación del XR-2206 como modulador.

Valores de frecuencias utilizados en este diseño.

$$f_1 = 2225 \text{ Hz Marca.}$$

$$f_2 = 2025 \text{ Hz Espacio.}$$

$$f_0 = 2125 \text{ Hz Portadora.}$$

$$f_0 = (f_1 + f_2) / 2$$

- 1.- Suponiendo un valor FSK de 1 Vpp a la salida del modulador. El circuito XR-2206 proporciona 60 mV por cada k Ω de R3.

realizando el planteamiento:

$$\begin{array}{l} 1 \text{ k}\Omega \text{ es a } 60 \text{ mVpp} \\ \text{como } x \text{ es a } 1 \text{ Vpp} \end{array}$$

resolviendo

$$x = 1000 / (60 \times 10^{-3}) \quad -3 \quad x = 16.6666 \text{ k}\Omega$$

$$R3 = 16.6666 \text{ K}\Omega \approx \underline{18 \text{ K}\Omega}$$

2.- El valor del capacitor C debe estar en el rango:

1000 pf a 100 μ f

Para poder determinar los valores de R (R_1, R_2) y C tales que permitan obtener la frecuencia más cercana a la deseada, se calculan los valores de R_1 y R_2 partiendo de los valores comerciales de C y se obtiene la tabla 3.7.8.a de valores. Para el cálculo de éstos se utilizaron las siguientes expresiones.

$$R_1 = 1/(f_1 \cdot C) \qquad f_1 = 2225 \text{ Hz}$$
$$R_2 = 1/(f_2 \cdot C) \qquad f_2 = 2025 \text{ Hz}$$

valor de C (μ f)	valor de R_1 (K Ω)	valor de R_2 (K Ω)
0.068	6.609	7.262
0.033	13.619	14.964
0.022	20.429	22.447
0.018	24.969	27.435
0.015	29.943	32.922

Tabla 3.7.8.a Tabla de valores.

El valor de las resistencias más apropiado para los valores de R_1 y R_2 son los que se obtienen con un capacitor de 0.022 μ f.

por lo tanto: $R_1 = R_2 = \underline{22 \text{ K}\Omega}$

3.7.9 Diseño del demodulador.

La función de demodulación la realiza el circuito XR-2211, el cual está diseñado especialmente para comunicaciones de datos. Este circuito puede operar en un rango de voltaje de 4.5 V a 20 V y un rango de frecuencias de 0.01 Hz a 300 KHz. Puede detectar señales entre 2 mV y 3 V de pico y es compatible con las familias lógicas TTL, DTL, ECL.

El circuito consiste básicamente de un PLL (Phase-Locked Loop), el cual sigue a la señal de entrada dentro de una determinada banda; un detector de cuadratura de fase que realiza la detección de portadora y un comparador de voltaje que proporciona la demodulación (fig. 3.7.9.a).

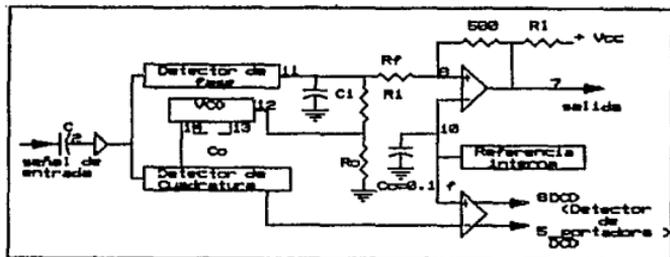


Fig. 3.7.9.a Diagrama de bloques del XR-2211.

1. Descripción de los elementos de control (fig 3.7.9.b)

a. Entrada de la señal (terminal 2).

Tiene una impedancia interna de 20 K Ω . Se recomienda un nivel de señal en el rango de 10 mVrms a 3 Vrms.

- b. Salida del detector de cuadratura de fase (terminal 3).

Es una salida de alta impedancia y esta conectada internamente a la entrada del comparador y detector de portadora. En aplicaciones donde se requiera detección de portadora, la terminal 3 se conecta a tierra a través de una combinación en paralelo de un capacitor y una resistencia (Rd y Cd), para eliminar las variaciones en la detección de portadora. Como en este diseño no se requiere la detección de portadora, la terminal 3 se deja sin conectar.

- c. Salida de detección de portadora (terminal 5).

Esta salida se encuentra en un nivel alto cuando la señal de entrada esta fuera de los límites del PLL y en un nivel bajo cuando la señal de entrada está dentro de estos límites. Es una salida de colector abierto, la cual requiere de una resistencia conectada a +Vcc para poder operar. En un nivel bajo puede manejar hasta 5 mA de corriente de carga. La terminal 6 es el complemento de esta salida.

- d. Salida de datos (terminal 7).

Es una salida de colector abierto y puede manejar 5 mA de corriente de carga. Esta salida se encuentra en un nivel alto para bajas frecuencias y en nivel bajo para las altas. Si no existe señal de entrada, el estado lógico de esta terminal es indeterminado.

- e. Entrada al comparador FSK (terminal 8).

Es una entrada de alta impedancia, la cual se conecta normalmente a través de un filtro de datos a la salida del detector de fase del PLL (terminal 11). Este filtro de datos está formado por Rf y Cf. El voltaje de umbral del comparador es determinado por la referencia interna en la terminal 10.

- f. Voltaje de referencia Vr (terminal 10).

Esta terminal está internamente polarizada a un voltaje de $V_r = (+V/2) = 650$ mV. El nivel de voltaje de DC en esta terminal es una referencia interna para

los niveles de voltaje de las terminales 5, 8, 11 y 12. La terminal 10 debe ser conectada a tierra a través de un capacitor de 0.1 μ f para un buen funcionamiento del circuito. Este valor es recomendación del fabricante.

g. Salida del detector de fase (terminal 11).

Es una salida de alta impedancia. El filtro del PLL está formado por una resistencia y un capacitor (R_i y C_i) conectados a esta terminal. Cuando no hay señal de entrada, o cuando no existe error de fase en el PLL, el nivel de DC, en la terminal 11 es casi igual a V_r . El voltaje de pico máximo a la salida del detector de fase es igual a V_r .

h. Control del VCO (terminal 12).

La frecuencia de oscilación libre del VCO (Voltage Controller Oscillator) es determinada por la resistencia externa R_o , conectada de esta terminal a tierra, esta frecuencia está dada por la relación:

$$f_o = 1/(C_o \cdot R_o)$$

Donde:

C_o - Es el capacitor entre las terminales 13 y 14.

Para una óptima estabilidad con la temperatura R_o debe de estar en el rango de 10 a 100 $K\Omega$. Este valor es proporcionado por el fabricante.

i. Capacitor del VCO (terminales 13 y 14).

La frecuencia del VCO es inversamente proporcional al capacitor C_o conectado a través de estas terminales. C_o no debe ser electrolítico y debe encontrarse en el rango de 200 pf a 10 μ f (por recomendación del fabricante).

j. Frecuencia de Oscilación libre.

El circuito XR-2211 no tiene una terminal de salida del VCO. Sin embargo, para propósitos de ajuste, esta frecuencia puede ser medida en la terminal 3 (con C_d desconectada), sin señal de entrada y con la terminal 2 conectada a la terminal 10.

Aplicación del XR-2211 como demodulador de FSK.

La figura 3.7.9.c muestra los elementos básicos que nos permiten la demodulación FSK. Con referencia a las figuras 3.7.9.a y 3.7.9.b las funciones de los componentes externos se definen como:

- Ro y Co .- Fijan la frecuencia central del PLL.
- R1 .- Fija el ancho de banda del sistema.
- C1 .- Fija la constante de tiempo del filtro y su factor de amortiguamiento.
- Cf y Rf .- Forman un filtro de un polo para la salida de datos FSK.
- Rb .- Es una resistencia de retroalimentación positiva a través del comparador FSK para facilitar una rápida transición entre los diferentes estados lógicos de salida.

Al unir la terminal 6 con la 7 la salida de datos estará inhabilitada en un estado bajo, hasta que no se presente una portadora dentro del rango de detección del PLL, y entonces la terminal 6 pasará a un nivel alto permitiendo la salida de datos.

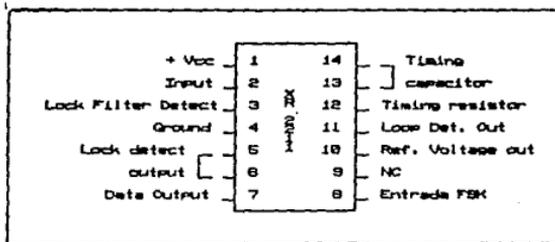


Fig. 3.7.9.b. Diagrama de terminales del demodulador XR-2211.

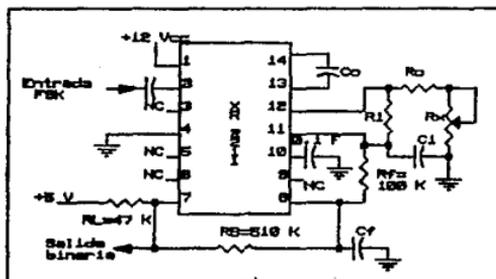


Fig. 3.7.9.c. Configuración como demodulador FSK del XR-2211.

Para que el circuito XR-2211 funcione como demodulador se tienen que calcular 5 componentes:

1. Calcular la frecuencia central del PLL.

$$f_0 = (f_1 + f_2) / 2$$

2. Escoger un valor arbitrario de R_0 en el siguiente rango:

$$10 \text{ K}\Omega \leq R_0 \leq 100 \text{ K}\Omega$$

3. Calcular C_0 .

$$C_0 = 1 / (R_0 \cdot f_0)$$

4. Calcular R_1 con f ($f =$ diferencia entre frecuencias f_1 y f_2).

$$R_1 = (R_0 \cdot f_0) / f = (R_0 \cdot f_0) / (f_1 - f_2)$$

5. Calcular C_1 para fijar la constante de amortiguamiento.

$$\xi = \frac{1}{2} \sqrt{C_0/C_1}$$

por recomendación del fabricante $\xi = 1/2$

despejando.

$$C_1 = C_0/4$$

Cálculo de valores para un demodulador de 300 Bauds.

- $f_1 = 2225 \text{ Hz}$
 $f_2 = 2025 \text{ Hz}$
 $f_0 = (f_1+f_2)/2 = 2125 \text{ Hz}$
- Escogiendo $R_0 = 18 \text{ K}\Omega$
- $C_0 = 1/(R_0-f_0) = 1/(18 \text{ K}\Omega \cdot 2125) = 0.026 \mu\text{f}$
 $C_0 = 0.022 \mu\text{f}$
- $R_1 = (R_0-f_0)/(f_1-f_2) = (18 \text{ K}\Omega \cdot 2125)/(2225-2025)$
 $R_1 = 191.250 \text{ K}\Omega \approx 200 \text{ K}\Omega$
 $R_1 = 200 \text{ K}\Omega$
- $C_1 = C_0/4 = 0.022 \mu\text{f}/4 = 0.0055 \mu\text{f}$
 $C_1 = 0.0047 \mu\text{f}$

3.8 Acoplamiento del circuito a la línea telefónica.

En esta etapa se realiza el acoplamiento acústico del sistema, el cual es otro de los requisitos de diseño, éste le dará la dinámica de una conexión sencilla a la línea telefónica

3.8.1 Acoplamiento del transmisor.

El circuito XR-2206 utilizado como modulador, no es capaz de proporcionar los niveles de voltaje y corriente necesarios para realizar una buena conversión electro-acústica, es decir, las señales generadas por el modulador

(cto. XR2206) no son lo suficientemente adecuadas para que estas produzcan un nivel de sonido ideal para el acoplamiento con la línea. El circuito modulador entrega una señal de salida con un nivel de 1 Vpp y a la entrada de la bocina de 8 Ω , se necesita un mínimo de 10 Vpp para lograr una buena comunicación por la línea telefónica. En caso de que esta sea de muy mala calidad este voltaje sufre un incremento de 5 vpp, es decir, el nivel de voltaje debe de ser de 15 Vpp, estos valores se obtuvieron en forma experimental y serán explicados más adelante.

Para esta etapa de acoplamiento se utilizaron amplificadores operacionales, debido a que poseen las siguiente características.

- Alta impedancia de entrada.
- Bajo nivel de ruido.
- Produce baja distorsión.
- Baja impedancia de salida.
- Inmunidad al ruido, del medio ambiente y a las variaciones de la fuente.

El amplificador operacional utilizado es el TL072CN, el cual es un amplificador doble, es decir, contiene dos amplificadores en un solo encapsulado.

La configuración utilizada es la tipo inversor (fig. 3.8.1.a) debido a que la corriente de entrada i es la misma que I y está determinada por la relación $i = V_i/R_1$, es decir, la corriente es suministrada por la fuente de alimentación en este caso la proporciona el modulador.

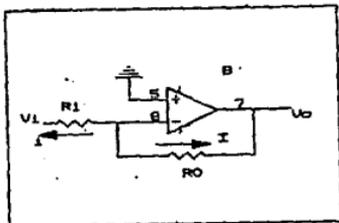


Fig. 3.8.1.a Configuración inversora.

El amplificador operacional debe de ser capaz de soportar esta corriente y amplificar el voltaje V_i "n" veces, esto tiene como resultado un incremento de "n" veces en la potencia de salida con respecto a la de entrada.

$$\begin{aligned} P_o &= V_o \cdot I && \text{Potencia de salida} \\ P_i &= V_i \cdot i && \text{potencia de entrada} \end{aligned}$$

$$V_o = n \cdot V_i$$

$$P_o = n \cdot V_i \cdot I \quad \text{pero} \quad V_i \cdot i = P_i$$

por lo tanto:

$$P_o = n \cdot P_i$$

La ganancia de este amplificador está determinada por la siguiente expresión :

$$V_o = - (R_o/R_i) \cdot V_i$$

Donde:

- V_o .- Voltaje de salida.
- R_i .- Resistencia de entrada.
- R_o .- Resistencia de Realimentación.
- V_i .- Voltaje de entrada.

Cálculo de valores.

La ganancia máxima del amplificador que requiere es de 15 y la mínima de 10. Calculando el valor de las resistencias R_o y R_i para el primer caso.

Se fija un valor de R_i a 1 $K\Omega$, V_i a 1 V_{pp} y V_o a 15 V_{pp} , se tiene sólo una incognita que es R_o , despejando de la expresión anterior, se obtiene que el valor de R_o es igual a 15 $K\Omega$.

Calculando el valor de R_o para el segundo caso, se mantienen fijos los valores de R_i y V_i , se sustituye el valor de V_o a 10 V_{pp} y se obtiene un valor de 10 $K\Omega$ para el segundo caso.

Para que el sistema tenga una mayor dinámica se coloca el siguiente arreglo (fig. 3.8.1.b) para poder ajustar la ganancia dependiendo de la calidad de la línea.

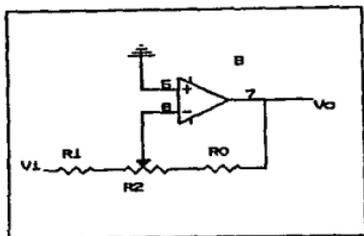


Fig. 3.8.1.b Modificación a la configuración inversora.

Esta nueva configuración garantiza una ganancia mínima de 2, es decir, cuando el potenciómetro (R2) se encuentre totalmente a la izquierda (fig 3.8.1.c) y una ganancia máxima de 17 cuando en el potenciómetro se encuentra totalmente a la derecha (fig 3.8.1.d).

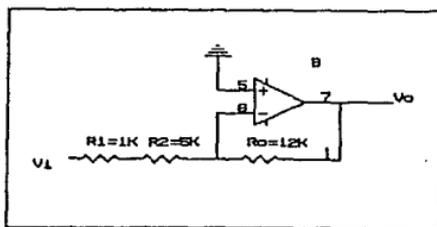
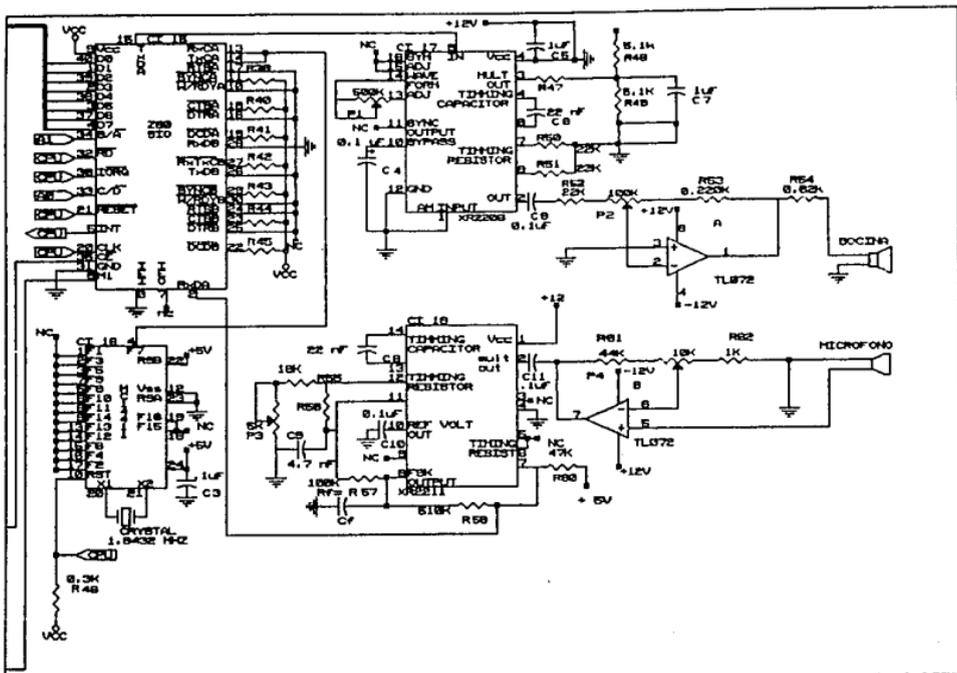


Fig. 3.8.1.c Mínima ganancia.



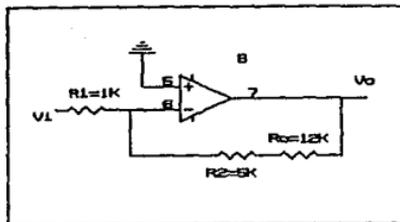


Fig. 3.8.1.d Máxima ganancia.

3.8.2 Acoplamiento del receptor.

Los voltajes proporcionados por el micrófono son del orden de $30 \text{ mVpp} \approx 10.6 \text{ Vrms}$, y el circuito utilizado como demodulador XR-2211 tiene un rango de detección de 10 mVrms a 3 Vrms , como se sabe la señal recuperada por el micrófono difícilmente puede ser reconocida por el demodulador. Para asegurar que siempre será reconocida, se amplificará 100 veces, es decir, se garantizará que la señal entregada al demodulador siempre esté dentro de su rango de detección.

En este amplificador se utiliza la configuración NO INVERSORA (fig 3.8.2.a).

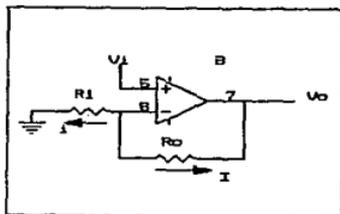


Fig. 3.8.2.a Configuración NO INVERSORA.

Esta etapa tiene la característica de que la corriente i no es proporcionada por el voltaje V_i , sino por el amplificador y está determinada por la siguiente relación $i = V_m/R_1$ la cual no es de importancia por que el demodulador detecta voltaje no corriente y bastará con asegurar que el voltaje suministrado sea correcto.

Cálculo de valores.

La ganancia de este amplificador está determinada por la siguiente expresión.

$$V_o = (1 + R_o/R_1) \cdot V_i$$

Donde:

- V_o .- Voltaje de salida.
- V_i .- Voltaje de entrada.
- R_o .- Resistencia de realimentación.
- R_1 .- Resistencia de entrada.

Fijando el valor de R_1 a un valor de $1 \text{ k}\Omega$, y suponiendo un voltaje promedio de entrada de $30 \text{ Vpp} \approx 15 \text{ Vp} \approx 10.6 \text{ Vrms}$, y una ganancia de 100, despejando el valor de R_o se obtiene un valor de $R_o = 100 \text{ k}\Omega$.

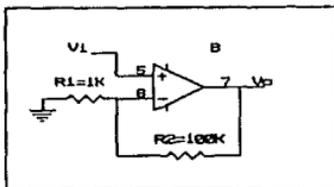


Fig. 3.8.2.b Configuración del amplificador utilizado.

3.8.3 Pruebas realizadas.

Debido a los problemas típicos de la línea telefónica de la ciudad de México (ruido, atenuación, etc.), no se tuvo la

confianza suficiente en las normas que se proporcionan, aunado a la falta de información acerca de las características de la bocina y del micrófono utilizados en el sistema telefónico y cuando se encontró información alterna, es decir, manuales, folletos etc. no se encontraron estos artículos en el mercado Mexicano, por estas razones se tuvo la necesidad de experimentar y utilizar componentes que sí existieran en México.

Las pruebas se realizaron de la siguiente manera:

La elección del tipo de calidad de la línea telefónica se realizó tomando como referencia la claridad con la cual se pueden comunicar personas normales, es decir, hablar sin gritar y escuchar con claridad (sin distorsión), la prueba se tenía que realizar en las peores condiciones (máximo y mínimo nivel de atenuación, ruido, distorsión, etc.) para poder encontrar estos extremos, nos comunicamos con el mayor número de personas posible, entre los que se encontraban familiares, amigos y aun los propios compañeros de tesis. Dependiendo de los factores antes mencionados se eligieron cuales líneas eran las apropiadas.

Una vez determinados los casos extremos se procedió a realizar la prueba. En el extremo que simularía la parte transmisora (fig 3.8.3.a) se utilizó un generador de frecuencias, el cual se utilizó para generar las frecuencias de 2225 Hz, la cual para nuestro caso representara un 0 lógico y la de 2025 Hz un 1 lógico, ambas con un voltaje de salida de 1 Vpp (el que entrega el circuito modulador XR-2206), se acoplo una de estas frecuencias al micrófono del teléfono con ayuda de una bocina de 8 Ohms y una potencia de 0.025 watts (fig 3.8.3.a).

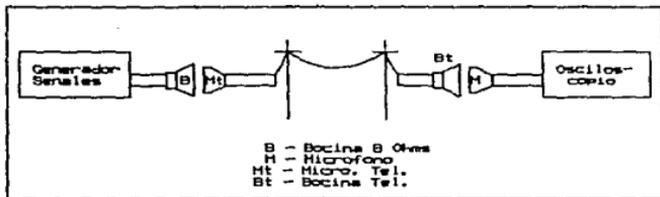


Fig 3.8.3.a Implementación de las pruebas realizadas.

En la parte que simularía al receptor se acopló un micrófono a la bocina del teléfono y este a su vez a un osciloscopio, en el cual se observó la forma de onda de la señal recuperada y en él se realizan las mediciones de amplitud y distorsión.

Tomando como referencia la frecuencia de 2225 Hz, se procedió a incrementar la amplitud del voltaje de salida del generador y en el receptor se observó el comportamiento de la señal recibida, anotando en cual de estas variaciones la señal fuera lo más clara y con la menor distorsión posible, un aspecto importante que se tomó en cuenta es que el acoplamiento acústico tiene que ser lo más silencioso posible porque al generar una sola frecuencia, y esta es escuchada por personas normales (sin problemas auditivos), llega a ser molesta cuando se escucha por un largo tiempo, lo cual es una gran limitante para el diseño. Para resolver este problema se recomienda realizar un acoplamiento a presión, es decir, cada vez que se coloque la bocina en el LETOVISIMNOVER, éste tiene que sellar a la bocina por medio de un hule especial como lo muestra la figura siguiente (fig. 3.8.3.b).

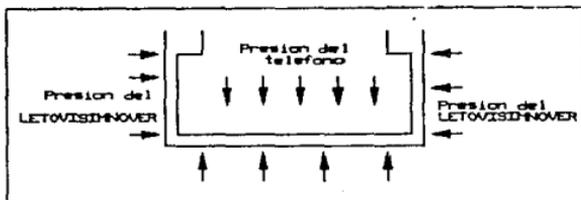


Fig. 3.8.3.b Al presionar el teléfono contra el LETOVISIMNOVER éste presiona al aislante contra el teléfono.

Aunado a esto los niveles de voltaje se escogieron tomando en cuenta la posible molestia que podrían causar, los valores elegidos fueron los siguientes :

Línea normal ---- 10 Vpp
Línea con ruido -- 15 Vpp

CAPITULO 4 DESCRIPCION DEL SOFTWARE UTILIZADO EN EL DISEÑO DEL CIRCUITO.

4.1 Programación del CPU Z-80.

En este capítulo, se presenta una pequeña introducción al conjunto de instrucciones del CPU Z-80. Es importante mencionar que no se describen las instrucciones a detalle ya que no es la finalidad de esta tesis.

4.1.1 Instrucciones del CPU Z-80.

El CPU Z-80 puede ejecutar 158 tipos diferentes de instrucciones, que pueden ser agrupadas como sigue:

1. Carga e intercambio.

En las instrucciones de carga, se mueven datos entre los registros internos del CPU o entre éstos y la memoria. Cada una de estas instrucciones deberá especificar una localidad origen, de la cual el dato es movido a una localidad destino. La localidad origen no es alterada por la instrucción de carga. El segundo tipo de instrucciones sirve para intercambiar el contenido entre registros.

2. Transferencia y búsqueda de bloques.

El CPU Z-80 cuenta con un conjunto de instrucciones para la transferencia de bloques. Con una sola instrucción un bloque de memoria de cualquier tamaño puede moverse a cualquier otra parte de la misma.

También cuenta con un conjunto de instrucciones de búsqueda de bloques, con una simple instrucción, en un bloque de memoria de determinada longitud es posible buscar un carácter de 8 bits.

Tanto las instrucciones de búsqueda como las de movimiento de bloques, pueden ser interrumpidas durante su ejecución, para que no mantengan ocupado al procesador por largos periodos de tiempo.

3. Aritméticas y lógicas.

Todas estas instrucciones de 8 bits, operan sobre un dato que está en el acumulador y sobre otro que se encuentre en cualquiera de los registros del CPU o en alguna localidad de memoria. También en este grupo se encuentran las operaciones de 16 bits.

4. Rotación y corrimiento.

Estas instrucciones permiten que cualquier registro o localidad de memoria, sea rotada tanto a la derecha como a la izquierda, con o sin acarreo, ya sea aritmético o lógico.

5. Manipulación de bits (Set, Reset y Test).

Las instrucciones para el manejo de bits permiten que cualquier bit en el acumulador, de cualquier registro de propósito general o de una localidad externa de memoria, sea puesto a 1 lógico, 0 lógico o evaluado en una simple operación.

6. Salto, llamada y retorno.

Realizan la transferencia del control del programa a otra dirección. Este grupo usa varias técnicas para obtener la nueva dirección del PC (Contador de Programa). Los saltos en los programas pueden llevarse a cabo por medio de una carga de los registros HL, IX o IY directamente al PC.

7. Entrada/Salida.

Este grupo de instrucciones permite la transferencia de información, entre las localidades de memoria o los registros de propósito general y los dispositivos de Entrada/Salida. Para esta instrucción, la dirección del puerto es colocada en los 8 bits menos significativos del bus de direcciones, durante la operación de Entrada/Salida. La instrucción de entrada automáticamente habilita el registro de banderas. El CPU incluye las instrucciones que pueden mover datos hacia un puerto o desde éste directamente a cualquier localidad de memoria, que en unión con un conjunto

de registros de propósito general, permiten una transferencia de datos bastante apropiada.

B. Control para el CPU.

Permiten varios modos de operación, este grupo incluye instrucciones para habilitar o deshabilitar el flip-flop de interrupciones. También sirven para habilitar los modos de interrupción y para que el CPU consuma ciclos de reloj sin ejecutar operación alguna o pare su proceso.

4.1.2 Modos de direccionamiento.

Por modo de direccionamiento se entiende, la manera de acceder el dato de la instrucción a ejecutar. Se tienen varios modos de direccionamiento que se explican brevemente con un ejemplo:

Direccionamiento inmediato.

- LD C,40H

Con esta instrucción se carga el registro C con el número hexadecimal que se tenga inmediatamente.

Direccionamiento inmediato extendido.

- LD BC,0457H

Se carga el registro B con el número 04 en hexadecimal y el registro C con el número 57 en hexadecimal.

Carga entre registros.

- LD C,A

Se carga lo que se tenga en el segundo registro, al primero.

Direccionamiento indirecto.

- LD A,(HL)

Se carga lo que se tenga en la dirección señalada por HL en la memoria, al acumulador.

Direccionamiento extendido.

- LD (1203H),A

El dato que se tenga en el registro acumulador, es almacenado en la localidad de memoria señalada por el número 1203 hexadecimal.

Direccionamiento indexado.

- LD A, (I+d)

Se carga lo que se tenga en la dirección de memoria señalada por Ix+d, al registro señalado.

Direccionamiento modificado página cero.

- RST 10H

Estas instrucciones provocan que el control del programa sea transferido a una dirección de memoria señalada por el número hexadecimal inmediato. En esta dirección se tiene una pequeña subrutina y con esta instrucción sólo se permiten subrutinas que comiencen en las siguientes direcciones:

RST 00H	dirección	0000H
RST 08H	"	0008H
RST 10H	"	0010H
RST 18H	"	0018H
RST 20H	"	0020H
RST 28H	"	0028H
RST 30H	"	0030H
RST 38H	"	0038H

Direccionamiento implícito.

-ADD A,B

Es el caso de instrucciones que se aplican automáticamente a un registro determinado, en el ejemplo anterior este se aplica al registro acumulador.

Direccionamiento de bit.

-SET J,B

Este conjunto de instrucciones dan la facilidad de colocar un uno, un cero o examinar el bit que se indica del registro señalado.

Direccionamiento relativo.

-JR 09H

Es un modo de direccionamiento muy especializado que se aplica a las instrucciones de saltos, llamados saltos relativos (JR). Tal como sucede con el direccionamiento indexado, el primer byte después del código de operación es un número en complemento a dos que representa un desplazamiento desde alguna dirección.

4.1.3 Modos de interrupción.

Las interrupciones sirven para realizar actividades de más alta prioridad. Existen 2 tipos de interrupciones:

1) Interrupción no mascarable (NMI).

No puede ser inhibida por el programador, siempre será aceptada por el microprocesador al final de la instrucción actual y causa lo siguiente:

- El contenido del registro PC (Contador de Programa) es almacenado en la dirección de memoria apuntada por el registro SP (Apuntador de Pila).
- Salta a la localidad de memoria (0066H).
- Cuando termina de ejecutar la instrucción o rutina de dicha localidad y encuentra una instrucción RETN (Regreso de Interrupción No Mascarable), lo que tenga en la dirección apuntada por el registro SP será regresado al registro PC.

2) Interrupción mascarable (INT).

Este tipo de interrupción es mascarable por software y hay tres modos:

- Modo 0: Se habilita por medio de la instrucción IM0. El CPU espera que el dispositivo que lo interrumpió coloque la instrucción a seguir, típicamente un RESTART.
- Modo 1: Se habilita por medio de la instrucción IM1. Cuando esta interrupción es aceptada, el registro PC almacena la dirección 0038H y el CPU ejecuta lo que este indicado en esta localidad.
- Modo 2: Se habilita por medio de la instrucción IM2. Este modo de interrupción es muy poderoso, ya que permite tener varios dispositivos conectados y listos para interrumpir. Permite jerarquizar interrupciones.

Funcionamiento: Cuando el CPU detecta su señal de entrada INT en estado bajo, espera que un dispositivo le proporcione 8 bits de una dirección (parte baja). La parte alta de esta dirección es el contenido del registro I de interrupciones. Con esto el CPU forma un vector de interrupciones de 16 bits, lo que permite a 128 dispositivos diferentes (La combinación de los 7

bits menos significativos) interrumpir al CPU. Cuando el CPU ya formo su vector de interrupción, salta a la dirección especificada y el contenido de esa dirección proporciona la dirección de inicio de la rutina de interrupción.

Es conveniente mencionar que algunos autores consideran a la habilitación de Bus Request como una interrupción más y realiza lo siguiente:

Permite tener un acceso directo a memoria. El controlador de DMA controla el bus de datos, direcciones y control del CPU.

4.2 Programación del SIO Z-80.

Los canales del SIO pueden ser programados en una amplia variedad de protocolos de comunicación de datos, existiendo varios formatos para cada modo de operación.

Debido a que se utiliza el modo asincrónico en la tesis, las formas de operación síncronas solo se mencionan brevemente. Para ampliar información consultar referencias del Manual Técnico del Z-80.

4.2.1 Modo asincrónico.

En este modo, la recepción y la transmisión se realiza independientemente en cada canal con 5,6,7 u 8 bits por caracter, más uno opcional de paridad par o impar. La terminal SYNC puede ser programada como una entrada que puede ser usada como un indicador de monitoreo.

a. Recepción

Los puertos de recepción A y B tienen cuatro etapas de almacenamiento temporal, esto es, hay tres registros de almacenamiento en adición al registro de corrimiento de entrada, proporcionando con ello tiempo suficiente para que el CPU dé servicio de una interrupción para que se inicie la transferencia de un bloque de datos de alta velocidad.

El receptor está protegido de picos de voltaje por un mecanismo de rechazo de transitorios que checan la señal en 1/2 tiempo de bit cuando un nivel bajo es detectado sobre la entrada de datos (RxD A o RxD B).

Los errores son detectados y almacenados en buffers junto con el carácter donde ocurrieron. Creandose un proceso de chequeo y reset (borrado) que evita interpretar un error como un nuevo bit de inicio (tal es el caso de un error de framing, que resulta de la adición de $1\frac{1}{2}$ tiempo de bit, para que la búsqueda del próximo bit de inicio comience).

La recepción se iniciara cuando el bit de recepción habilitada esté en 1 lógico (SET), entonces, la terminal detector portadora de datos (DCD) debe estar en estado bajo. El estado bajo de la terminal recepción de dato (RxD), indica un bit de paro y si éste permanece para $\frac{1}{2}$ tiempo de bit, el bit de paro será valido y el dato de entrada es probado para ese $\frac{1}{2}$ tiempo de bit hasta que el carácter completo es ensamblado. En este método, un bit de paro permite rechazar errores cuando existen picos de ruido sobre la línea en estado alto, es decir, no se estan enviando bits.

Los errores de overrun y paridad no son borrados a menos que un comando de reset de error (comando 6) sea utilizado.

b. Transmisión.

Para la transmisión, hay una secuencia de bits en donde no se transmiten caracteres (estado alto), hasta que el registro de control es programado, este estado permanecerá hasta que un comando SEND BREAK haya sido utilizado o se resetee el chip como se muestra en la figura 4.2 a.

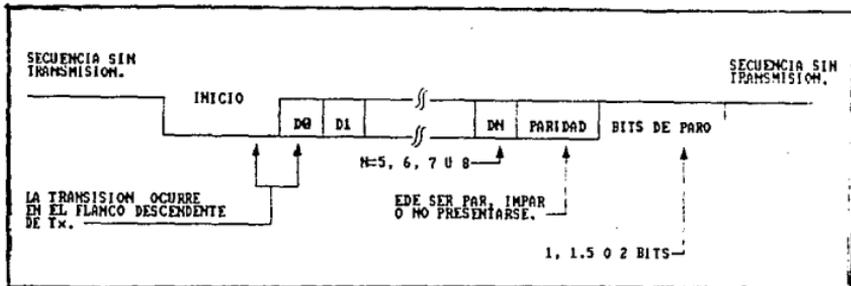


Fig. 4.2.a. Formato asincrónico.

La transmisión no puede empezar a menos que el bit de habilitación del transmisor esté en 1 lógico (SET), si la autohabilitación se selecciona, entonces la terminal borrar para mandar (CTS), debe estar en estado bajo.

4.2.2 Modo síncrono.

En el modo síncrono se requiere un factor de reloj $x1$ para la transmisión y recepción, el dato es muestreado sobre el flanco ascendente de recepción de reloj (RxC) y es transmitido sobre el flanco descendente de transmisión de reloj (TxC). Puede manejar varios modos que permiten la sincronización de carácter, clasificandose de la siguiente manera:

a. Modo monosíncrono (8 bits).

Apareando un simple carácter síncrono, programado dentro del registro WR7, implica sincronización con habilitación de transferencia de dato.

b. Modo bisíncrono (16 bits).

Acoplando dos caracteres síncronos adyacentes en los registros WR6 y WR7 se logra una sincronización, la terminal de sincronización (SYNC) se activara en estado bajo en cualquier momento donde el carácter de sincronización es detectado y así permanecerá durante el ciclo de reloj en donde se detectó.

c. Modo síncrono externo.

El carácter a ensamblar comienza sobre el flanco ascendente de RxC, después de que la terminal SYNC regresa al estado bajo, así debe mantenerse hasta que se completan los últimos tres ciclos de reloj.

d. Modo SDLC.

El modo SDLC provoca el envío automático de banderas. El receptor automáticamente se sincroniza con la primera bandera de un cuadro (frame) y provee una señal de

sincronización en la terminal SYNC; una interrupción puede ser también programada.

La diferencia en la operación en todos los modos síncronos, es la manera en como se logra la inicialización. El modo de operación debe ser seleccionado antes de que el carácter síncrono sea cargado.

4.2.3 Técnicas de I/O de datos del SIO.

El SIO Z-80 tiene la capacidad de realizar la transferencia de datos, el control de información y de estado, manejados desde el CPU a través de las funciones de poleo del dato, transferencia de bloques e interrupciones (vectorizadas o no vectorizadas).

a. Poleo.

Cuando el CPU opera en el modo poleo, éste lee uno de los registros de estado del SIO periódicamente, para determinar si un carácter de dato ha sido recibido o si está listo para su transmisión. Cuando el SIO está listo, el CPU maneja la transferencia con su programa principal. Este registro de estado indica si se tiene algún dato o si lo está recibiendo.

Dos bits en el registro indican si es necesaria la transferencia del dato, las condiciones de error son indicadas en el mismo registro.

El segundo registro de estado (condición especial de recepción) no es leído en una secuencia de poleo, hasta que un carácter no haya sido recibido. Todos los modos de interrupción son deshabilitados cuando el dispositivo opera en modo poleo.

b. Transferencia de bloque CPU/DMA.

En este modo, el SIO es usado junto con el DMA (Acceso Directo a Memoria) o con otros dispositivos, para una mayor velocidad en la transferencia de datos. El SIO interrumpe al CPU o al DMA solo cuando el primer carácter del mensaje está disponible y después, el SIO usa solo sus terminales de salida WAIT/READY para indicar su disposición para la transferencia de caracteres subsecuentes.

Debido a la alta velocidad de transferencia que se logra con este modo, es sólo utilizado en el modo sincrónico.

La terminal READY del SIO indica que éste está listo para la transferencia de datos, para o desde la memoria. Para el CPU, la terminal WAIT indica que el SIO no está listo para la transferencia de datos.

c. Interrupciones.

El servicio de interrupciones en el SIO, se realiza en un tiempo real de aplicación, los registros de control y de estado en el canal B contienen al vector de interrupción. El SIO informa al CPU de la necesidad de interrumpir su secuencia en el programa, por medio de su terminal de requerimiento de interrupción (INT), indicando que se requiere de la transferencia de un carácter. Para completar el proceso de interrupción, El CPU debe habilitarse para chequear la presencia de interrupción a la señal INT al final de el ciclo de instrucción M1. Cuando el CPU detecta una interrupción la ramifica para una rutina de interrupción la cual maneja la transferencia de un sólo carácter.

1. Recepción.

Existen dos modos de interrupción que pueden ser programados:

a. Interrupciones sólo en el primer carácter.

Se utiliza al iniciar un loop de poleo o una transferencia de un bloque de instrucción, usando la salida WAIT/READY para sincronizar al CPU de acuerdo con la relación de datos. El SIO interrumpirá en el primer carácter y después, sólo si hay errores detectados. Este modo se selecciona con el comando 4.

b. Interrupción en cada carácter.

Siempre que el buffer del receptor tiene un carácter, se genera una interrupción y si se selecciona el modo ESTADO AFECTA AL VECTOR, un vector especial se genera para condiciones especiales de error, (los borra).

Cuando la recepción es habilitada, el CPU puede ser interrumpido por el buffer de recepción lleno del SIO.

2. Transmisión.

Las principales interrupciones son: en la recepción de datos y sobre el estado externo. Cada interrupción es habilitada por el control del programa, con el canal A, teniendo una mayor prioridad el canal B.

Cuando la transmisión es habilitada, el CPU puede ser interrumpido por el buffer de transmisión vacío del SIO.

3. Manejo de interrupciones.

En la programación del vector de interrupciones, el SIO puede modificarlo dependiendo del tipo de interrupción que se presente. Con el bit D2 del registro de escritura WR1 se elige si se desea un solo vector de interrupción (que sería programado previamente) o que dependa de la condición de el ESTADO AFECTA AL VECTOR. Este bit sólo funciona en el canal B.

La figura 4.2.3.a muestra los 8 casos de interrupciones que se pueden presentar en el SIO. Los bits afectados en el vector de interrupciones son el D1, D2 y D3. El bit D0 no se afecta para que en la tabla de vectores resulten 8 apuntadores a las rutinas de atención de interrupciones.

Canal	D3	D2	D1	Motivo de la interrupción
A	0	0	0	Buffer de transmisor vacío.
	0	0	1	Cambio de estado externo.
	0	1	0	Carácter disponible en el receptor.
	0	1	1	Condición de recepción especial.
B	1	0	0	Buffer de transmisión vacío.
	1	0	1	Cambio de estado externo.
	1	1	0	Carácter disponible en el receptor.
	1	1	1	Condición de recepción especial.

Fig. 4.2.3.a

El reconocimiento de interrupción en modo 0 (D3) y en modo 1 (D4), juntos especifican la condición de varios caracteres disponibles como se muestra en la figura 4.2.3.b.

Modo	D4	D3	Interrupciones
0	0	0	De recepción deshabilitada.
1	0	1	En el receptor solo en el primer carácter.
2	1	0	Sobre todos los caracteres la paridad afecta.
3	1	1	Sobre todos los caracteres recibidos, el error de paridad no afecta al vector.

Fig. 4.2.3.b

4.2.4 Registros de lectura y escritura.

La programación del SID depende del contenido del registro de control y se debe programar antes de que se realice cualquier operación, se pueden cambiar algunos comandos y modos durante su operación y también leer los registros en cualquier momento. Para programar al SID, se debe de iniciar con el modo básico de operación deseado por medio de los comandos de registro que controlan las direcciones, tal como las entradas de control o datos (C/D) y selección de canal B o A (B/A) que son manejados por el bus de direcciones del CPU Z-80 (fig. 4.2.4.a).

C/D	B/A	FUNCION
0	0	Canal A dato
0	1	Canal B dato
1	0	Canal A control
1	1	Canal B control

Fig. 4.2.4.a

a. Registros de escritura. .

El SID contiene ocho registros de escritura, el WRO inicializa al SID pudiendo acceder todos los controles (CRC0, CRC1) y los comandos básicos (CMD0-CMD2) (fig. 4.2.4.b).

DO	D6	
Reset CRC código 1	Reset CRC código 0	
0	0	Código nulo (no afecta).
0	1	Reset en el receptor chegador de CRC.
1	0	Reset en transmisor generador de CRC.
1	1	Reset en envío de latch CRC/SYNC.

Fig. 4.2.4.b

Los siguientes son los comandos: (D3-D5) (fig. 4.2.4.c).

CMD	CMD2	CMD1	CMD0	
0	0	0	0	Comando nulo (no afecta).
1	0	0	1	Envío abortado (modo SDLC).
2	0	1	0	Reset ext/edo. de interrupciones.
3	0	1	1	Reset del canal.
4	1	0	0	Reset inter. en el primer carácter.
5	1	0	1	Reset inter. del Tx pendiente.
6	1	1	0	Reset del error (latches).
7	1	1	1	Retorno de una inter. (canal A).

Fig. 4.2.4.c

Todos los registros, excepto el WRO requieren de dos bytes para ser programados. el primer byte que es el WRO contiene los tres bits (D0-D2) que selecciona al segundo byte, que será la palabra de control que se escriba en la programación del SID. La siguiente figura (4.2.4.d) muestra la tabla de los registros de escritura.

Funciones de los registros de escritura	
WR0	Registro puntero, inicializa al CRC y a los comandos para los varios modos.
WR1	Controla las interrupciones en transmisión/recepción y la transferencia de datos en el modo definido.
WR2	Controla al vector de interrupción.
WR3	Controla la recepción en modo sincrónico o asíncrono.
WR4	Contiene a los bits que afectan a la transmisión/recepción, (es de carácter general).
WR5	Controla la transmisión en modo asíncrono/sincrónico.
WR6	Contiene el BLS del carácter de sincronía interno o el campo de direcciones SDLC.
WR7	Contiene el BMS del carácter de sincronía interno o la bandera de SDLC.

Fig. 4.2.4.d.

b. Registros de lectura.

Son registros de estado y sirven para leer el estado, los errores o la información relacionada con la comunicación. Existen tres registros de estado los cuales se direccionan de la misma forma que los registros de comando, indicando sobre el registro WR0 qué registro se desea leer. Si no se procede de la forma indicada se leera el registro de lectura 0 (RR0). Los registros de lectura se muestran en la figura 4.2.4.e.

Funciones de los registros de lectura	
RR0	Estado del buffer en transmisión/recepción, de interrupciones y estado externo.
RR1	Estado de condición especial en la recepción.
RR2	Vector de interrupción modificado (sólo canal B).

Fig. 4.2.4.e.

4.3 Programación del circuito.

La programación del circuito, constituye el Software y el usuario sólo podrá acceder datos por medio del teclado para ser procesados. El usuario NO podrá acceder nuevas características a la programación.

La programación consta de las siguientes rutinas principales:

- 1) Inicialización del SIO y del sistema.
- 2) Rutina de validación y transmisión de dato.
- 3) Rutina de recepción y validación de dato.

La programación se empieza a ejecutar una vez que el CPU recibe un RESET. Está estructurada como se ilustra en el diagrama de flujo de la figura 4.3.a y funciona como se describe a continuación.

4.3.1 Rutina de inicialización del SIO Z-80 y del sistema.

Al activar el circuito (on), este quedará reseteado y listo para ser programado. Se utilizan cinco de sus registros para darle una personalidad definida. Estos son programados en una secuencia específica, como se muestra en la tabla 4.3.1.a.

Los diagramas de flujo de inicialización del SIO y de inicialización del circuito se muestran en las figuras 4.3.1.b y 4.3.1.c respectivamente.

PROGRAMA PRINCIPAL

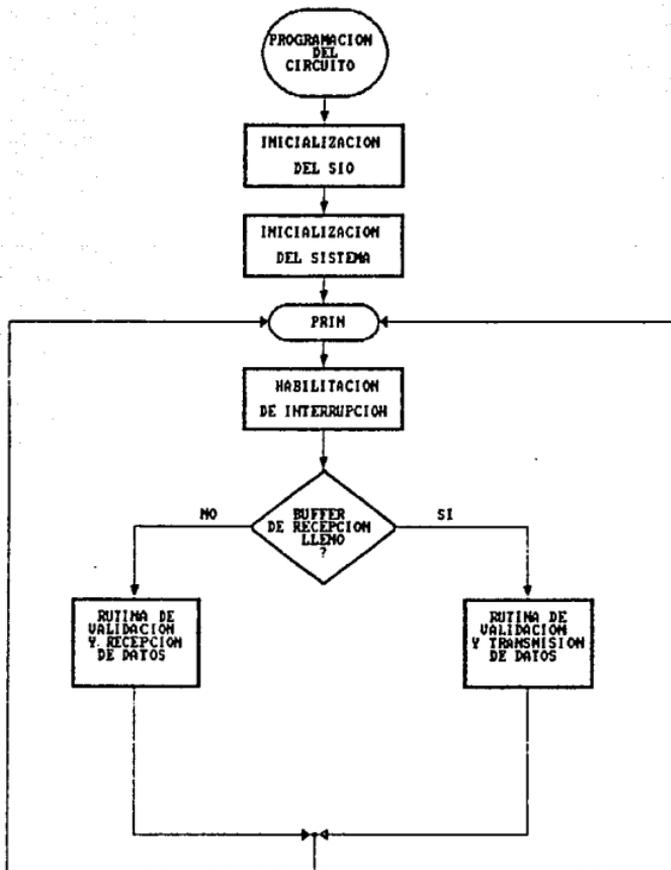


FIG. 4.3.a DIAGRAMA DE FLUJO DE LA PROGRAMACION DEL CIRCUITO

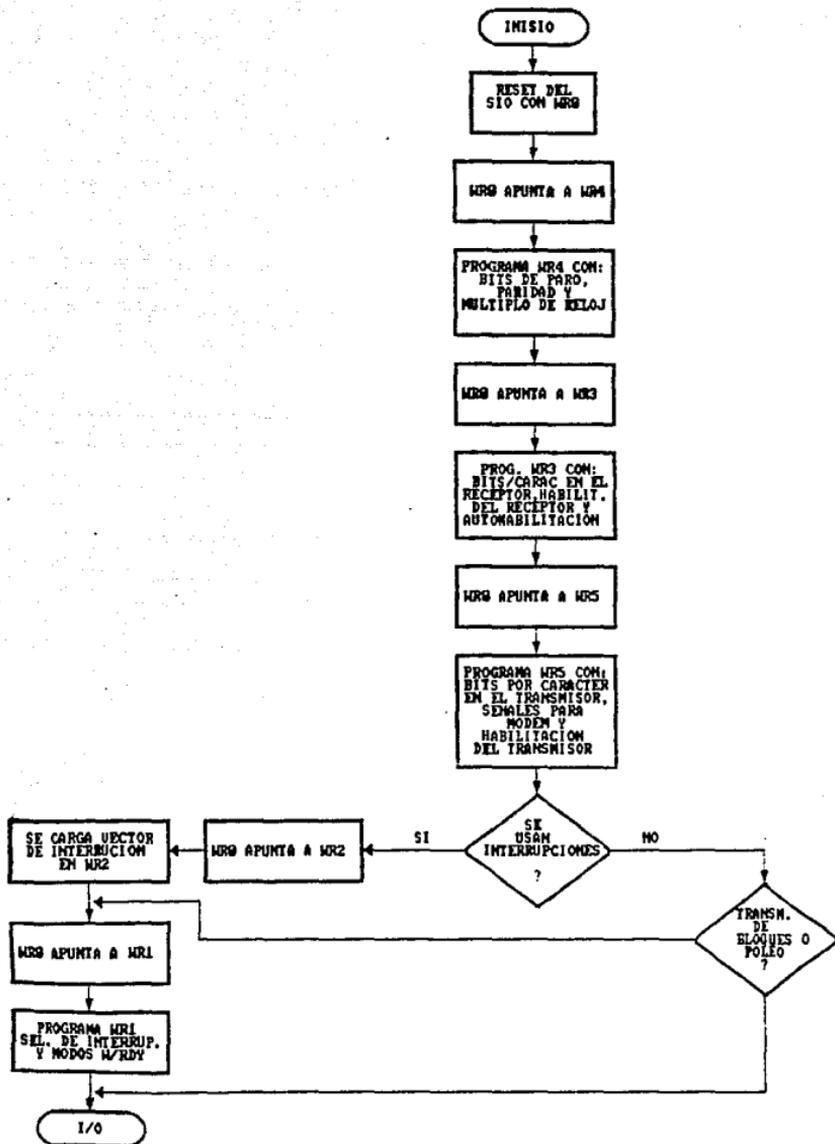


FIG. 4.3.1.a DIAGRAMA DE FLUJO GENERAL DE LA INICIALIZACION DEL SIO.

registro	código	binario	función
WRO	18H	00011000	Resetear el canal A control, con comando 3.
WRO	04H	00000100	WRO apunta para escribir en WR4.
WR4	47H	01000111	Vel de Tx X16, sin 8 bits, 1 bit de paro, paridad impar habilit.
WRO	05H	00000101	WRO apunta para escribir en WRS.
WRS	6AH	01101010	8 bits/carácter en transmisión, transmisión habilitada y RTS.
WRO	03H	00000011	WRO apunta para escribir en WR3.
WR3	E1H	11100001	8 bits/carácter en recepción, autohabilitación, Rx habilitada.

Tabla 4.3.1.a

Para asignar los valores mostrados a los registros correspondientes, se usó la instrucción de alto nivel OTIR, donde el contenido del par de registros HL se coloca en el bus de direcciones para seleccionar una posición de memoria, cuyo contenido es almacenado temporalmente en el procesador central con el fin de ser colocado posteriormente en el bus de datos y ser enviado al dispositivo de destino. B es un contador de bytes que se va decrementando en cada transferencia. El contenido de C se coloca en la parte baja del bus de direcciones y sirve para seleccionar el dispositivo de entrada-salida de destino; la instrucción incrementa HL y decrementa B, así como al PC, transfiriendo en cada paso un dato y terminando cuando B=0.

OTIR: (C) - (HL) B - B-1 HL - L+1

Repetir hasta que B = 0.

El sistema utiliza el manejo de interrupciones y el CPU es programado para funcionar en el modo 2 de interrupción (IM2). El dispositivo periférico interruptor (SIO) es el medio para obtener la dirección inicial de la subrutina de servicio de interrupción.

El CPU proporciona el byte más significativo (BMS) de la dirección donde buscar la subrutina y lo hace leyendo su propio registro I que debe ser cargado con 0BH.

Registro I.

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

Estos registros WR2 e I forman la dirección de la memoria, donde el CPU Z-80 va a buscar la dirección inicial de la subrutina que le dá servicio a la interrupción, cuya parte baja está dada por la dirección mencionada y la parte alta está en la dirección contigua superior que es cargado con 0BH.

Una vez programado el SID para interrumpir al CPU, se inicializa al resto del sistema de la siguiente manera:

- Se inicializa el apuntador de pila (SP):
 - LD SP,0B20H
- Se habilita el modo 2 de interrupciones:
 - IM2

Se pueden presentar los siguientes casos de interrupción.

Interrupción: Buffer de transmisión vacío en el canal B.

Reg. I(BMS)=0BH Reg. WR2 (BLS)=00H Dirección formada

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

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

0B00H

Dirección formada : 0B00H = parte baja con contenido 0BH

Dirección siguiente: 0B01H = parte alta con contenido 00H

Dirección inicial de la subrutina: 00B3H que contiene un RETI
(canal B no se usa).

Interrupción: Cambio de estado externo en canal B.

Reg. I (BMS)=0BH Reg. WR2(BLS)=02H Dirección formada

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

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

0B02H

Dirección formada: 0B02H = parte baja con contenido 03H
Dirección siguiente: 0B03H = parte alta con contenido 00H
Dirección inicial de la subrutina: 0007H que contiene un RETI
(canal B no se usa).

Interrupción: Recepción de carácter disponible en canal B.

Reg. I (BMS)=0BH Reg. WR2(BLS)=04H Dirección formada

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

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

0B04H

Dirección formada: 0B04H = parte baja con contenido 03H
Dirección siguiente: 0B05H = parte alta con contenido 00H
Dirección inicial de la subrutina: 0008H que contiene un RETI
(canal B no se usa).

Interrupción: Recepción de condición especial en canal B.

Reg. I (BMS)=0BH Reg. WR2(BLS)=06H Dirección formada

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

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

0B06H

Dirección formada: 0B06H = parte baja con contenido 03H
Dirección siguiente: 0B07H = parte alta con contenido 00H
Dirección inicial de la subrutina: 0009H que contiene un RETI
(canal B no se usa).

Interrupción: Buffer de transmisión vacío en canal A.

Reg. I (BMS)=0BH Reg. WR2(BLS)=0BH Dirección formada

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

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

0B0BH

Dirección formada: 0B0BH = parte baja con contenido 08H
Dirección siguiente: 0B09H = parte alta con contenido 00H
Dirección inicial de la subrutina: 006BH (que contiene la subrutina de detección de tecla oprimida, validación de tecla y envío de código de tecla al SIO).

Interrupción: Cambio de estado externo en canal A.

Reg. I(BMS)=08H

Reg. WR2(BLS)=0AH

Dirección formada

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

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

080AH

Dirección formada: 080AH = parte baja con contenido 83H

Dirección siguiente: 080BH = parte alta con contenido 00H

Dirección inicial de la subrutina: 0000H que contiene un RETI
(no existen cambios externos)

Interrupción: Recepción de carácter disponible en canal A.

Reg. I(BMS)=08H

Reg. WR2(BLS)=0CH

Dirección formada

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

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

080CH

Dirección formada: 080CH = parte baja con contenido B5H

Dirección siguiente: 080DH = parte alta con contenido 00H

Dirección inicial de la subrutina: 00B5H (que contiene la subrutina de recepción de dato por el SID para enviarlo al CPU Z-80).

Interrupción: Recepción de condición especial en canal A.

Reg. I(BMS)=08H

Reg. WR2(BLS)=0EH

Dirección formada

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

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

080EH

Dirección formada: 080EH = parte baja con contenido B3H

Dirección siguiente: 080FH = parte alta con contenido 00H que

Dirección inicial de la subrutina: 00B3H que contiene un RETI
(ya que no se presenta).

A continuación se carga el registro acumulador con el número hexadecimal FF y se guarda este dato en la localidad (0822H) de memoria. Este mismo dato es sacado por los puertos de salida 1 y 2.

- LD A,FFH
- LD (0822H),A
- OUT(42H),A
- OUT(43H),A

Esto se hace para que el dato almacenado en 0822H sirva como punto de comparación en las rutinas de transmisión y recepción. Al inicializar el sistema ninguna tecla se encuentra oprimida, así que los puertos de salida 1 y 2 deben mantenerse en 1 lógico. A continuación se habilita el flip-flop de interrupciones, en espera de que ocurra alguna, ejecutando no operaciones.

- PRIN EI
- NDP
- JP PRIN

Si el buffer de recepción del Puerto A del SIO está lleno, la interrupción que genera hace que se ejecute la Rutina de Recepción y Validación de Dato.

En caso de no ocurrir lo anterior se tendrá una interrupción por buffer de transmisión vacío del puerto A del SIO. Esta interrupción hace que se ejecute la rutina de validación y transmisión de dato.

El CPU siempre hace caso a la interrupción y si ésta es generada por el buffer de recepción del SIO tiene mas alta prioridad que la generada por el buffer de transmisión.

Una vez que el CPU termina de ejecutar la rutina por la que fué interrumpido, regresa a habilitar la interrupción y espera a que ocurra alguna de ellas.

4.3.2 Rutina de validación y transmisión de dato.

Esta rutina verifica que el dato que es teclado sea válido, en caso de ocurrir esto el dato es codificado en 8 bits. En la tabla 4.3.2.1 se muestran los posibles datos válidos, la tecla que sea activada permanecerá en 0 lógico.

Es conveniente recordar que en el circuito se completan los dos bytes, habilitando con 1 lógico los bits de D12 a D15 por lo tanto estos bits nunca serán 0 lógico.

Puerto de Entrada O2				PE02				Puerto de Entrada 01 PE01							
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1

Tabla 4.3.2.1 Posibles datos válidos a enviar.

La tabla 4.3.2.1 tiene su equivalente que es la tabla 4.3.2.2 en donde se muestran los posibles datos válidos en código hexadecimal.

Tecla	O1 oprimida	PE02	PE01
"	01	FF	FE
"	02	FF	FD
"	03	FF	FB
"	04	FF	F7
"	05	FF	EF
"	06	FF	DF
"	07	FF	BF
"	08	FF	7F
"	09	FE	FF
"	10	FD	FF
"	11	FB	FF
"	12	F7	FF

Tabla 4.3.2.2 Código hexadecimal de datos válidos.

El diagrama de flujo de esta rutina se muestra en la figura 4.3.2.a y realiza lo siguiente:

Una vez dentro de la rutina, se deshabilita la interrupción para no tener una interrupción dentro de la ejecución de una rutina.

- DI

A continuación para mantener una señal constante en los puertos de salida PS01 y PS02 se hace una comparación del dato (almacenado en el registro H) con el número hexadecimal FF almacenado en A.

- LD H,A
- LD A,FFH
- CP H

Si la comparación es diferente de cero, la rutina tiene que cambiar el contenido de (0822H) a un valor FFH y este mismo valor es enviado por los puertos de salida 1 y 2.

- JP Z,CONT
- LD (0822H),A
- OUT(42H),A
- OUT(43H),A

Si la comparación es cero, va a la dirección del programa señalada por la etiqueta CONT o si la comparación es diferente de cero y se terminó de ejecutar el anterior bloque de instrucciones, también se va a la dirección señalada por la etiqueta CONT. A partir de aquí la rutina verifica si el dato teclado está entre D0 y D7, si existe un dato teclado en PE01 se debe verificar que no se presiona al mismo tiempo una tecla en el byte más significativo PE02. Del mismo modo, si existe un dato válido teclado en PE02, no debe existir un dato teclado en PE01.

El dato que se tenga en el byte más significativo (PE02) es colocado en el registro D del CPU:

- CONT IN A,(41H)
- LD D,A

El dato que se tenga en el byte menos significativo (PE01) es colocado en el registro E del CPU:

- IN A,(40H)
- LD E,A

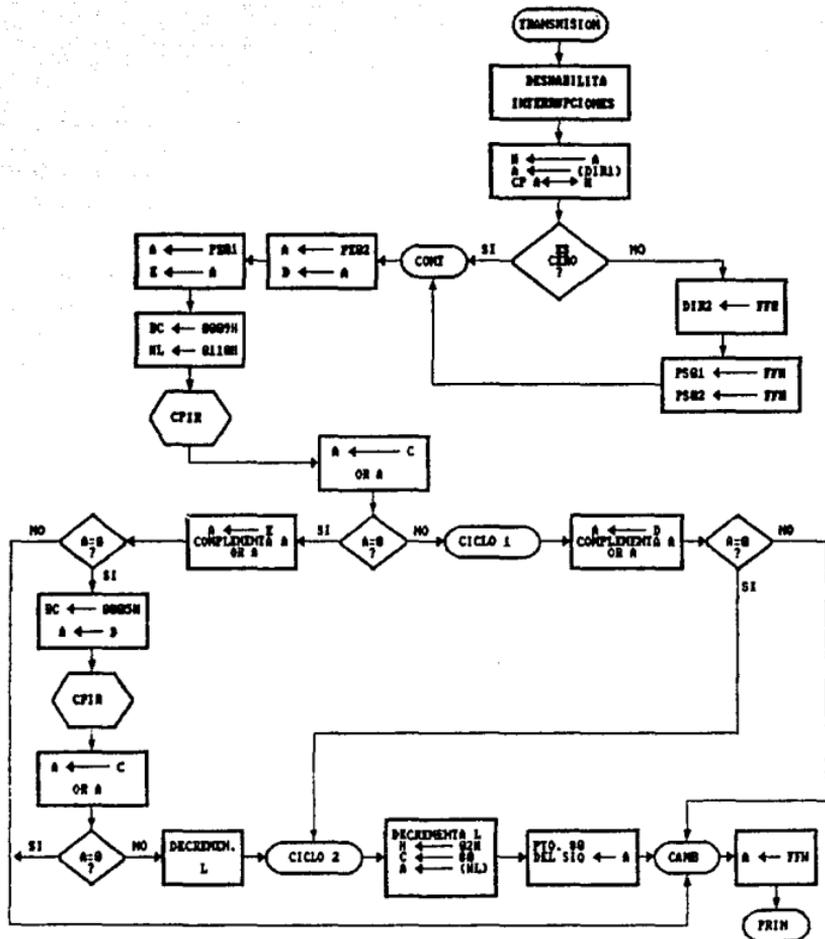


FIG. 4.3.2.a DIAGRAMA DE FLUJO DE LA RUTINA DE VALIDACION Y TRANSMISION DE DATO.

A continuación se prepara a detectar si el dato es válido para transmitir, esto se hace con la instrucción CPIR (Compara Incrementa y Repite). Esta instrucción compara el dato que se tenga en el registro A del CPU con un conjunto de datos señalados por una dirección inicial de la memoria EPROM, que es dada por los registros HL del CPU y se repite la comparación el número de veces señalada por un contador, cuyo trabajo será efectuado por los registros BC del CPU:

```
- LD BC,0009H
- LD HL,0110H
- CPIR
```

Dir. de Mem.	Contenido	Dir. de Mem.	Contenido
0110	FE	0210	01
0111	FD	0211	02
0112	FB	0212	03
0113	F7	0213	04
0114	EF	0214	05
0115	DF	0215	06
0116	BF	0216	07
0117	7F	0217	08
0118	7F	0218	09
0119	FE	0219	0A
011A	FD	021A	0B
011B	FB	021B	0C
011C	F7	021C	0C
011D	F7	021D	0C
(a)		(b)	

Tabla 4.3.2.3 Datos de comparación, para validar y enviar dato válido.

La tabla 4.3.2.3 muestra, en el inciso a) el posible dato a validar para este caso y en el inciso b) el dato a enviar. La instrucción CPIR compara el dato señalado por HL con lo que se tenga en el registro A, en la primera comparación HL=0110H y BC=0009H, a continuación incrementa HL en 1 y decrementa BC en 1 con lo que HL=0111H y BC=0008H. Este proceso de comparación, incremento y decremento se repite y solo puede terminar por dos motivos:

1. En las comparaciones no se encontró un dato igual al del registro A, por lo que BC=0000H.

2. En las comparaciones se encontró un dato igual al del registro A y BC≠0000H. Si éste es el caso, indica que en esta primera instrucción CPIR existe un dato válido en el byte menos significativo (PE01).

Quando se termina de ejecutar la instrucción CPIR, se procede a preguntar si el contador no es igual a 0000H:

```
- LD A,C
- OR A
- JP NZ,CICLO1
```

Si el contador no es igual con cero, el programa se posiciona en la dirección correspondiente a CICLO1. Esto indica que existe un dato válido en el byte menos significativo, por lo que se tiene que verificar que el byte más significativo no tenga presionada ninguna tecla. En CICLO1. Esto se logra complementando el byte más significativo, si el complemento es cero, todas las teclas son lógicas. Se tiene lo siguiente:

```
- CICLO1 LD A,D
- CPL A
- OR A
- JP NZ,CAMB
```

Dentro de CICLO1, si se da el caso de que una tecla del byte más significativo sea presionada, indicará al programa que se posicione en CAMB, ya que se tiene un dato no válido. Si ninguna tecla del byte más significativo ha sido presionada, entonces se tiene un dato válido, por lo que el programa se posiciona en la dirección correspondiente a CICLO2.

Si el programa se posiciona en CICLO2, entonces se tiene un dato válido y ocurrirá lo siguiente:

```
- CICLO2 DEC L
- LD H,02H
- LD C,80H
- LD A,(HL)
- OUT (C),A
```

Ya que la instrucción CPIR termina en la siguiente localidad al encontrar un dato válido, se tiene que decrementar el registro L. A continuación se toma el dato de la tabla 4.3.2.3.b que es el equivalente en 8 bits al de la tabla 4.3.2.3.a, o es también el equivalente respectivo a

alguno de los primeros 8 renglones de la tabla 4.3.2.2. Esto se hace al cargar al registro H con 02H y después cargar el registro A con lo que se tenga en la localidad HL, a continuación se le dá salida al dato por el buffer de transmisión del SIO que tiene la dirección 80H (Que previamente se programó, en la rutina de inicialización del SIO).

La anterior explicación corresponde a la rama derecha después de la primera instrucción CPIR del diagrama de flujo de esta rutina, que corresponde a la figura 4.3.2.3.a. Ahora se procede a explicar la rama izquierda del diagrama señalado.

Se debe recordar que al salir de la primera instrucción CPIR se preguntó por el estado del contador en los siguientes términos:

- JP NZ,CICLO1

En caso de cumplirse la condición de que el contador sea igual con cero se toma la rama izquierda del diagrama de flujo, esto indica que ninguna o más de una tecla de las primeras 8 ha sido presionada. Por lo tanto se debe verificar que ninguna tecla de estas primeras 8 haya sido presionada, ya que de otra forma se tiene una condición de error, para ello se realizan los siguientes pasos:

El byte menos significativo es cargado en el registro A; se obtiene el complemento del registro A y se realiza la operación lógica OR de este registro. A continuación se está en condiciones de preguntar si el byte menos significativo tiene todos sus bits en 1 lógico.

- LD A,E
- CPL A
- OR A
- JP NZ,CAMB

Si el registro de banderas indica que el registro A no es cero, el programa se posiciona en la localidad señalada por la etiqueta CAMB, indicando una condición de error, en esta localidad la rutina carga el registro A con FFH; a continuación existe una instrucción RETI por lo que el control del programa ejecuta un salto al programa principal en espera de una nueva interrupción. Si el registro de banderas indica que el registro A es cero, entonces ninguna tecla del byte menos significativo ha sido presionada y se debe verificar el estado del byte más significativo. Esto se hace con una segunda instrucción CPIR, con HL=0119H en cuyo

valor concluyó la primera instrucción CPIR. En la tabla 4.3.2.3.a, además se debe proporcionar el número de comparaciones que se deben realizar y cargar el byte más significativo en el registro A:

```
- LD BC,0005H
- LD A,D
- CPIR
```

Una vez que se salió de este segundo CPIR se debe preguntar si BC=0000H, en caso de ocurrir, no existe dato válido en el byte más significativo, por lo que el programa se posiciona en la dirección correspondiente a la instrucción CAMB:

```
- LD A,C
- OR A
- JP Z,CAMB
```

Si el contador es diferente de cero, se tiene un dato válido y el apuntador HL se quedó en la siguiente dirección al dato válido, por lo que se debe decrementar el registro L en uno y posicionar a la rutina en la instrucción correspondiente a la localidad CICLO2:

```
- DEC L
- JP CICLO2
```

Ya se explicó que en CICLO2 se envía el dato a través del buffer de transmisión del Puerto A del SIO y es aquí donde la rama izquierda y derecha del diagrama de flujo se unen.

Al enviar el dato por el SIO u obtener una condición de error el programa se posiciona en la dirección de programa señalada por la etiqueta CAMB, en esta dirección se tiene la instrucción de cargar el registro A con el número hexadecimal FF y a continuación se tiene una instrucción RETI.

```
- LD A,FFH
- RETI
```

Con esto se da por terminada la rutina de validación y transmisión de dato y se regresa al programa principal en espera de una nueva interrupción.

4.3.3 Rutina de recepción y validación de dato.

Una vez que el buffer de recepción del SIO está lleno se genera una interrupción y manda al programa a la Rutina de Recepción de Dato, en la que el CPU debe verificar que el dato existente en el buffer de recepción sea válido.

Esto se logra comparando el dato existente en el buffer de recepción del SIO con los datos de las localidades de memoria 0210H al 021CH (Tabla 4.3.2.3.b). Si se dá el caso de que el dato del buffer sea igual a alguno señalado en estas localidades de memoria, entonces se tiene un dato válido, del cual se tiene que conocer si pertenece al byte menos significativo o más significativo. Una vez verificadas estas dos condiciones se puede pasar de la Tabla 4.3.2.3.b a la Tabla 4.3.2.3.a, el dato correcto de esta última tabla se compara con el dato de la localidad de memoria (0822H), que tendrá el mismo dato válido si la anterior interrupción fue por estar presionada la misma tecla, si este es el caso el control pasa a la localidad señalada por la etiqueta RETARDO, en donde el control del microprocesador ejecuta instrucciones de no operación un tiempo que depende de una constante.

Esto es debido a que el modem trabaja a 300 bauds; es muy lento comparado con la velocidad de la Rutina de Recepción y Validación de Dato por lo que esta rutina debe mantener el dato válido un cierto tiempo de retardo en espera de que exista el mismo dato en el buffer de recepción del SIO y este pueda mostrarse de manera continua en los terminales de salida PS01 y PS02. Si una vez terminada la Rutina de Recepción de Dato no existe interrupción por buffer de recepción del SIO lleno, se tiene que del otro lado de la línea telefónica no se oprimió alguna tecla o ya se terminó de oprimir la tecla.

En caso de que el dato en el buffer de recepción del SIO no sea igual a alguno de los señalados en la tabla 4.3.2.3.b el control pasa a la localidad señalada por la etiqueta RETARDO.

El diagrama de esta rutina se muestra en la figura 4.3.3.a y su funcionamiento es el siguiente:

En la rutina de recepción, se deshabilita el flip-flop de interrupciones y se almacena el dato del buffer de recepción del puerto A del SIO en el registro E del CPU, por claridad y como se maneja la rutina de validación y transmisión de dato se carga el registro D con 00H.

- DI
- IN A, (80H)
- LD E, A
- LD D, 00H

A continuación se completa la preparación de los registros internos del CPU para utilizar la instrucción CPIR aplicada a la tabla 4.3.2.3.b.

- LD HL, 0210H
- LD BC, 000DH

Se utiliza la instrucción CPIR

- CFIR

Se prepara a preguntar porqué salió de la instrucción CPIR, recuerdese que existen 2 casos: cuando el contador es igual a 0 (registros BC=0000H) o cuando se encuentra un dato válido, con lo que el contador es diferente de cero, BC≠0000H.

- LD A, C
- OR A

Se pregunta si el contador es igual a cero, en cuyo caso se va a la localidad del programa señalado por la etiqueta RETARDO.

- JP Z, RETARDO

Si el contador es distinto de cero, se tiene un dato válido por lo que se debe preguntar si el dato está en el byte menos significativo o más significativo. Esto se logra con las siguientes instrucciones:

- SUB 04H
- JP Z, CICLO4
- JP M, CICLO4

Las preguntas anteriores indican la rama izquierda del diagrama de flujo de esta rutina, que corresponde al byte más significativo o la rama derecha que señala la etiqueta CICLO4, que corresponde al byte menos significativo.

Si el dato está en el byte más significativo ocurre lo siguiente:

- DEC L
- LD H,01H

Con las anteriores instrucciones se está señalando un dato correcto del byte más significativo en la tabla 4.3.2.3.a.

A continuación se guarda este dato en el registro H y se compara con el dato almacenado en la localidad de memoria 0822H. El dato de la memoria será el número FFH si la anterior rutina ejecutada fué de validación y transmisión de dato; o si la anterior rutina ejecutada fué de recepción y validación de dato, el dato a compararse será el último transmitido:

- LD A, (HL)
- LD H,A
- LD A, (0822H)
- CF H

Si el dato válido actual es igual al dato de la localidad 0822H, se tiene la condición de que el usuario que está transmitiendo no ha dejado de presionar la tecla, por lo que el contador de programa pasa directamente a la localidad señalada por la etiqueta RETARDO sin modificar las condiciones de los puertos de salida del usuario que está recibiendo:

- JP Z RETARDO

Si los datos no son iguales, se tiene el caso de estar recibiendo un dato nuevo con respecto a la anterior recepción y se debe actualizar con este último dato la localidad de memoria 0822H, también se debe modificar el puerto 02 de salida que corresponde al byte más significativo de un dato válido que se está simulando en 12 bits y a continuación el control pasa a la dirección señalada por RETARDO:

- LD A,H
- LD (0822H),A
- OUT(47H),A
- JP RETARDO

En CICLO4 se tiene lo mismo que lo que ya se explico, a partir de las preguntas:

- JP Z,CICLO4
- JP M,CICLO4

Pero ahora para el byte menos significativo:

- CICLO4 LD H,01H
- LD A,(HL)
- LD H,A
- LD A,(0822H)
- CP H

Con las instrucciones anteriores se señala un dato válido y se compara con el dato de la localidad 0822H, con lo que se verifica si es igual al anterior dato recibido; en caso de ocurrir esto último el contador de programa se posiciona en la localidad señalada por la etiqueta RETARDO:

- JP Z,RETARDO

Si el dato anterior y el actual válidos no son iguales, se actualiza el contenido de la localidad 0822H y el mismo dato válido actual es sacado por el puerto 42H lo que permite tener un dato válido, simulando tenerlo codificado en 12 bits:

- LD A,H
- LD (0822H),A
- OUT(46H),A

A continuación se pasa a la localidad señalada por la etiqueta RETARDO, donde se tienen instrucciones de no operación con la finalidad de consumir tiempo en espera de que llegue el mismo dato al buffer de recepción del SIU, debido a que el microprocesador trabaja mas rápido que el MODEM.

- RETARDO LD B,CONSTANTE DE RETARDO
- REPITE NOP
- DJNZ REPITE

A continuación se tiene una instrucción RETI con lo que el control regresa al programa principal en espera de una nueva interrupción. Con esto se da por terminada la rutina de recepción y validación de dato.

- RETI

```

1. 0000:
2. 0000:
3. 0000:
4. 0000:
5. 0000:
6. 0000:
7. 0000:
8. 0000:
9. 0000:
10. 0000:
11. 0000:
12. 0000:
13. 0000:
14. 0000:
15. 0000:
16. 0000:
17. 0000:
18. 0000:
19. 0000:
20. 0000:
21. 0000:
22. 0000:
23. 0000:
24. 0000:
25. 0000: OE B1
26. 0001:
27. 0002: 06 07
28. 0004: 21 00 01
29. 0007:
30. 0007: ED D3
31. 0009:
32. 0009:
33. 0009:
34. 0009: 3E 02
35. 000B: D3 B1
36. 000D: 3E 00
37. 000F: D3 B3
38. 0011: 3E 01
39. 0013: D3 B1
40. 0015: 3E 16
41. 0017:
42. 0017:
43. 0017: D3 B3
44. 0019:
45. 0019:
46. 0019:
47. 0019:
48. 0019:
49. 0019:
50. 0019:
51. 0019:
52. 0019:
53. 0019: 31 20 0B
54. 001C: ED 5E
55. 001E:

```

```

*****
!* U.N.A.M. FACULTAD DE INGENIERIA *
!*
!* PROGRAMACION DEL CIRCUITO *
!*
!* AUTORES: AXALCO MONTES ROBERTO *
!* IGLESIAS RAMIREZ OLGA MARIA *
!* MARTINEZ FLORES ENRIQUE *
!* VAZQUEZ FUENTES MA. DEL ROSARIO *
!*
*****

.ORG 00H ;Indica el origen del prog.

*****
!*
!* ;Inicializacion del SID *
!*
*****

LD C,B1H ;Puerto control A del SID
;ten C.
LD B,07H ;Inicia contador B con 7
LD HL,0100H ;Guarda en HL direccion.
;de tabla de inicialización.
DTIR ;Inicializa el SID.
;B=B-1
;HL=HL+1
;Repite hasta que B=0.
;WR0 apunta a WR2.
OUT (B1H),A ;Se programa WR2 en canal A.
LD A,00H ;Cargar vec. de interrup.
OUT (B3H),A ;Escribirlo en pto. B3 B.
LD A,01H ;WR0 apunta a WR1 de int.
OUT (B1H),A ;Escribirlo en pto. A
LD A,16H ;WR1: int. c/canac.,pariti-
;dad afecta al vec., edo
;afec. vec., trans. hab.
OUT (B3H),A ;WR1 a pto. B3 B CONTROL

*****
!*
!* ;(inicializacion del sistema *
!*
*****

LD SP,0B20H ;Apuntador pila a 0B20H.
IM 2 ;Modo 2 de interrupcion
;enmascarable habilitada.

```

```

56. 001E: 3E 0B          LD A,0BH          !0B en A, vect. de int.
57. 0020: ED 47          LD I,A           !es el BMS del vector en I.
58. 0022: 3E B3          LD A,-77         !ir a RETI en caso de
59. 0024:                 !interruccion en canal B:
60. 0024: 32 00 0B      LD (0B00H),A     ! El vector de interr.
61. 0027: 32 02 0B      LD (0B02H),A     ! forma estas direcciones
62. 002A: 32 04 0B      LD (0B04H),A     ! cuyo contenido es un
63. 002D: 32 06 0B      LD (0B06H),A     ! RETI, porque no importan.
64. 0030: 32 0A 0B      LD (0B0AH),A     ! Interr.ext. no interesan.
65. 0033: 32 0E 0B      LD (0B0EH),A     !Cond. esp. A no importa.
66. 0036: 3E 68          LD A,68H         !Direcc. de la rutina de
67. 0038:                 !transmision se almacena
68. 0038: 32 0B 0B      LD (0B0BH),A     !en 0B0BH.
69. 003B: 3E B5          LD A,-75         !Direcc. de la rutina de
70. 003D:                 !recepcion se almacena
71. 003D: 32 0C 0B      LD (0B0CH),A     !en 0B0CH.
72. 0040: 3E 00          LD A,00          !BMS de todos los vec.
73. 0042:                 !de interrupcion.
74. 0042: 32 01 0B      LD (0B01H),A     !cargar en esa direcc.
75. 0045: 32 03 0B      LD (0B03H),A     ! " " " "
76. 0048: 32 05 0B      LD (0B05H),A     ! " " " "
77. 004B: 32 07 0B      LD (0B07H),A     ! " " " "
78. 004E: 32 09 0B      LD (0B09H),A     ! " " " "
79. 0051: 32 0B 0B      LD (0B0BH),A     ! " " " "
80. 0054: 32 0D 0B      LD (0B0DH),A     ! " " " "
81. 0057: 32 0F 0B      LD (0B0FH),A     ! " " " "
82. 005A: 3E FF          LD A,-1          !Cargar A con FFH y
83. 005C: 32 22 0B      LD (0B22H),A     !colocarlo en 0B22H
84. 005F: D3 42          OUT (42H),A      !para sacarlo por los
85. 0061: D3 43          OUT (43H),A      !puertos de salida y
86. 0063:                 !mantener nivel alto.
87. 0063:
88. 0063:
89. 0063:
90. 0063:
91. 0063:
92. 0063:
93. 0063:
94. 0063: FB          PRIN          EI          !Habilitar interrupciones
95. 0064: 00          NOP          !Ejecutar no operaciones
96. 0065: C3 63 00      JP PRIN       !en espera de una interrup.
97. 006B:
98. 006B:
99. 006B:
100. 006B:
101. 006B:
102. 006B:
103. 006B:
104. 006B:
105. 006B:
106. 006B: F3          TRANS        DI          !Deshabilitar interrupciones.
107. 0069: 57          LD H,H         !Cargar el dato de A en H.
108. 006A: 3E FF      LD A,-1H      !Cargar el registro H con FFH.
109. 006C: BC          CP H          !Comparar el dato con FFH.
110. 006D: CA 77 00    JP Z,CNT      !Brinca a CNT si es igual 00H.

```

111.	0070:	32 23 08		LD (0822H),A	¡Almacenar el dato en memoria.
112.	0073:	05 42		OUT (42H),A	¡Sacar el dato por PS01.
113.	0075:	D3 43		OUT (43H),A	¡Sacar el dato por PS02.
114.	0077:	0E 01	CONT	LD C,70H	¡Simular FE02.
115.	0079:	ED 78		IN A,(C)	¡Leer byte más significativo
116.	007B:				¡(BMS) de la tecla oprimida.
117.	007B:	57		LD D,A	¡Cargar el BMS en D.
118.	007C:	0E 01		LD C,41H	¡Simular FE01.
119.	007E:	ED 78		IN A,(C)	¡Leer byte menos significativo
120.	0080:				¡(BLS) de la tecla oprimida.
121.	0080:	5F		LD E,A	¡Cargar el BLS en E.
122.	0081:	01 09 00		LD BC,0009H	¡Inicia contador BC con 0009H.
123.	0084:	21 10 01		LD HL,0110H	¡Direccion de inicio
124.	0087:				¡de la tabla 4.3.2.3.a
125.	0087:	ED B1		CPIR	¡Compara BLS con la
126.	0089:				¡primera parte de la tabla.
127.	0089:	79		LD A,C	¡Almacena contador en el
128.	008A:				¡registro A.
129.	008A:	57		OR A	¡Hacer la función logica OR.
130.	008B:	C2 A3 00		JP NZ,CICLO1	¡Preguntar el estado
131.	008E:				¡del contador.
132.	008E:	7B		LD A,E	¡Recuperar el BLS
133.	008F:				¡de la tecla presionada.
134.	008F:	2F		CPL	¡Complementario.
135.	0090:	57		OR A	¡Obtener la función
136.	0091:				¡logica OR.
137.	0091:	C2 B1 00		JP NZ,CAMB	¡Si el BLS no es cero se
138.	0094:				¡tiene dato invalido.
139.	0094:	01 05 00		LD BC,0005H	¡Iniciar contador con 0005H
140.	0097:				¡para comparacion con segunda
141.	0097:				¡parte de tabla 4.3.2.3.a
142.	0097:	7A		LD A,D	¡Recuperar el BMS.
143.	009B:	ED B1		CPIR	¡Compara BMS con la
144.	009A:				¡segunda parte de la tabla.
145.	009A:	79		LD A,C	¡Almacenar contador en el
146.	009B:				¡registro A.
147.	009B:	B7		OR A	¡Hacer la función logica OR.
148.	009C:	CA B1 00		JP Z,CAMB	¡Si el estado del contador
149.	009F:				¡es cero se tiene dato
150.	009F:				¡no valido.
151.	009F:	2D		DEC L	¡Decrementa registro L.
152.	00A0:	C3 A9 00		JP C,CICLO2	¡Brinca a CICLO2.
153.	00A2:	7A	CICLO1	LD A,D	¡Recuperar BMS
154.	00A4:	2F		CPL	¡Complementario.
155.	00A5:	B7		OR A	¡Obtener la función
156.	00A6:				¡logica OR.
157.	00A6:	C2 B1 00		JP NZ,CAMB	¡Si el BMS no es cero
158.	00A9:				¡se tiene condicion de error.
159.	00A9:	2D	CICLO2	DEC L	¡Decrementar registro L
160.	00AA:				¡para preparar cambio a
161.	00AA:				¡la tabla 4.3.2.3.b.
162.	00AA:	26 02		LD H,02H	¡Cargar el registro H con
163.	00AC:				¡02H para completar el cambio
164.	00AC:				¡de tabla.

165.	00AC:	0E 80	LD C,80H	!Señalar la direccion del
166.	00AE:			!puerto de salida.
167.	00AE:	7E	LD A, (HL)	!Almacena el dato valido
168.	00AF:			!en el acumulador.
169.	00AF:	ED 79	OUT (C),A	!Sacar el dato valido
170.	00B1:			!por el puerto señalado
171.	00B1:			!en este caso el SID.
172.	00B1:	3E FF	CAMB LD A,-1	!Almacena FFH en A.
173.	00B3:	ED 4D	RETI	!Regresar a rutinas de
174.	00B5:			!programacion.
175.	00B5:			
176.	00B5:			
177.	00B5:			
178.	00B5:			
179.	00B5:			
180.	00B5:			
181.	00B5:			
182.	00B5:			
183.	00B5:			
184.	00B5:	F3	RECEP DI	!Deshabilitar interrupciones.
185.	00B6:	DB 80	IN A, (80H)	!Leer dato del buffer
186.	00B8:			!de recepcion del SID.
187.	00B8:	5F	LD E,A	!Almacena el dato en E.
188.	00B9:	16 00	LD D,00H	!Almacena 00H en D.
189.	00B8:	21 10 02	LD HL,0210H	!Direccion de inicio
190.	00BE:			!de la tabla 4.3.2.3.b.
191.	00BE:	01 0D 00	LD BC,000DH	!Inicia contador BC con 000DH.
192.	00C1:	ED B1	CPIR	!Comparar el dato con
193.	00C3:			!los datos de validacion.
194.	00C3:	79	LD A,C	!Almacena contador en el
195.	00C4:			!registro A.
196.	00C4:	B7	OR A	!Hacer la función lógica OR.
197.	00C5:	CA F6 00	JP Z,RETARDO	!Preguntar el estado
198.	00CB:			!del contador, si es
199.	00CB:			!cero se tiene dato valido.
200.	00CB:	D6 04	SUB 04H	!Decrementar el contador
201.	00CA:			!en 04H.
202.	00CA:	CA E5 00	JP Z,CICLO4	!Si es cero
203.	00CD:	FA E5 00	JP M,CICLO4	!o es negativo salta a
204.	00D0:			!CICLO4 para obtener su
205.	00D0:			!equivalente ELS.
206.	00D0:	2D	DEC L	!Decrementa el registro L
207.	00D1:			!para obtener su equivalente
208.	00D1:			!en el BMS.
209.	00D1:	26 01	LD H,01H	!Completar el cambio de
210.	00D3:			!la tabla 4.3.2.3.b a
211.	00D3:			!la tabla 4.3.2.3.a.
212.	00D3:	7E	LD A, (HL)	!Almacena el BMS validado
213.	00D4:			!en el acumulador.
214.	00D4:	67	LD H,A	!Almacena el dato validado
215.	00D5:			!en H.
216.	00D5:	3A 22 0B	LD A, (0B22H)	!Almacena dato de comparacion
217.	00D8:			!de la loc. 0B22H en A.
218.	00D8:	8C	CP H	!Comparar dato validado con
219.	00D9:			!dato de comparacion.
220.	00D9:	CA F6 00	JP Z,RETARDO	!Si es cero saltar a rutina
221.	00DC:			!de retardo.
222.	00DC:	7C	LD A,H	!Almacena el dato valido
223.	00DD:			!en A.

```

*****
!*                               *
!*           Rutina de recepcion           *
!*                               *
*****

```

224.	00DD:	32 22 08		LD (0822H),A	!Almacenar el dato valido
225.	00E0:				!en la loc. 0822H.
226.	00E0:	D3 42		OUT (42H),A	!Sacar el dato por PS01.
227.	00E2:	C3 F6 00		JP RETARDO	!Brinca a RETARDO.
228.	00E5:	26 01	CICLO4	LD H,01H	!Completa cambio de la
229.	00E7:				!tabla 4.3.2.3.a a la
230.	00E7:				!tabla 4.3.2.3.a.
231.	00E7:	7E		LD A,(HL)	!Almacenar el BLS valido
232.	00E8:				!en el acumulador.
233.	00E8:	67		LD H,A	!Almacenar el dato valido
234.	00E9:				!en H.
235.	00E9:	3A 22 08		LD A,(0822H)	!Almacenar dato de comparacion
236.	00EC:				!de la loc. 0822H en A.
237.	00EC:	BC		CP H	!Comparar dato valido con
238.	00ED:				!dato de comparacion.
239.	00ED:	CA F6 00		JP Z.RETARDO	!Si es cero salta a
240.	00F0:				!rutina de RETARDO.
241.	00F0:	7C		LD A,H	!Almacenar el dato valido
242.	00F1:				!en A.
243.	00F1:	32 22 08		LD (0822H),A	!Almacenar el dato valido
244.	00F4:				!en la loc. 0822H
245.	00F4:	D3 43		OUT (43H),A	!saca el dato por PS02.
246.	00F6:	06 03	RETARDO	LD B,03H	!Carga 03 en B.
247.	00F8:	00	REPITE	NOF	!Ejecuta no operaciones.
248.	00F9:	10 FD		DJNZ REPITE	!Si ni es cero salta a
249.	00FB:				!REPITE.
250.	00FB:	ED 4D		RETI	!Retorno de interrupcion.
251.	0000:			.END	!Fin de programa.

Linea	Identificador	Valor
172	CAMB	00B1
183	CICLO1	00A3
189	CICLO2	00A9
228	CICLO4	00E5
114	CONT	0077
94	PRIN	0063
184	RECEP	00B5
247	REPITE	00F8
246	RETARDO	00F6
106	TRANS	0068

4.4 Prueba del Software.

El software involucrado en el comportamiento dinámico del diseño ya fue presentado en incisos anteriores como un producto terminado, sin embarco, para su total desarrollo fue necesario depurarlo en sus etapas finales, lo cual se logró con la ayuda de un paquete de programas diseñados para computadoras compatibles.

Dicho paquete consiste de 2 programas que forman una unidad de simulación del microprocesador CPU Z-80.

El primer programa, llamado ASMZ80, hace uso de un archivo fuente, generado por algún procesador de texto (se usó el ambiente de TURBOFASCAL), el cual consiste del programa a ser simulado, escrito en lenguaje ensamblador, incluyendo los comentarios pertinentes y las ordenes de principio y fin de archivo.

Una vez que ha terminado su tarea entrega dos archivos como respuesta.

El primero de ellos contiene el programa original en lenguaje ensamblador, así como los comentarios; pero además entrega, en hexadecimal, los códigos correspondientes a las instrucciones y los operandos, la numeración progresiva de los mismos, así como la correcta conversión a direcciones absolutas de las transferencias condicionales e incondicionales indicadas por las etiquetas usadas. Aunado a lo anterior este archivo contiene un reporte, si es el caso, de los errores detectados en la labor de ensamblado incluyendo comentarios pertinentes. El archivo es modificable con algún procesador de textos y permite tener un listado en limpio del archivo fuente, los códigos de operación, las direcciones absolutas, los comentarios, y un listado de las etiquetas usadas.

El segundo archivo consiste exclusivamente en un grupo de códigos hexadecimales que contienen toda la información del programa ya ensamblado y que será usado por el segundo programa del paquete.

Este segundo programa al que acabamos de referirnos, que se llama SIMZ80, logra que una computadora compatible simule el comportamiento de un sistema basado en el microprocesador Z80, incluyendo la simulación de puertos de entrada y salida, así como las interrupciones vectorizadas que el propio microprocesador es capaz de manejar. Al invocarlo presenta un menú de opciones con el cual se puede elegir la realización de diversas tareas involucradas en el proceso de la simulación, como son:

- 1) Introducción a la memoria de datos en hexadecimal desde el disco o desde teclado (pueden ser programas ensamblados o datos).
- 2) Envío desde memoria principal hacia el disco de datos en hexadecimal (programas o datos).
- 3) Definición de puertos de entrada y salida.
- 4) Definición de condiciones iniciales.
- 5) Edición y visualización de cualquier zona de la memoria.
- 6) Ejecución de la simulación.
- 7) Salir del programa.

En el punto 1) es posible introducir en la memoria, desde un disco, un programa previamente ensamblado por el programa ASMZ80; o bien, datos en hexadecimal que necesite usar la simulación.

La opción 2) permite hacer exactamente la operación inversa del punto anterior.

En el punto 3) se tiene la oportunidad de definir ocho puertos, que pueden ser: de entrada únicamente, de salida únicamente, o bien de entrada y salida. Es posible usar la consola (teclado y monitor) como noveno puerto, y tener con ello una opción interactiva con el programa.

El punto 4) permite establecer un estado inicial para el sistema. Esto es: dirección de arranque del programa, valor inicial para cada uno de los registros del microprocesador (principales y alternos) y valor inicial al contador de pila.

Se puede modificar un código de operación en el programa o bien un determinado dato previamente establecido, y esto se logra con el uso de la opción 5).

El punto 6) resulta ser el núcleo de la simulación ya que permite correr el programa introducido, con las condiciones iniciales especificadas, y con la posibilidad de hacer uso de los puertos definidos. Se pueden simular las interrupciones no enmascarables al oprimir la tecla F9. Se puede simular el rearranque al oprimir la tecla F10. Si el programa establece el modo 2 de interrupciones enmascarables y si habilita a estas últimas, las teclas F1-F8 proporcionan la posibilidad de lograr dichas interrupciones simulando el

periférico y devolviendo el vector de subrutina de servicio correspondiente. Durante la simulación en sí, es posible correr el programa paso a paso o en forma trazador, teniendo la oportunidad de estar observando sobre la marcha los valores contenidos en todos los registros, en la pila, en el contador de programa, en cada una de las banderas de estado, en los flip-flop de habilitación de interrupción, así como, estar observando las transferencias de información con los puertos, y, algo muy importante, el programa es desensamblado y el código fuente es mostrado simultáneamente a la corrida.

Conclusiones y Comentarios.

México es un país en el que es necesario hacer una fuerte vinculación entre sus necesidades, sus problemas específicos y el planteamiento de una solución adecuada a ellos. El objetivo de la ingeniería es encontrar dicha solución, mediante la optimización de los recursos disponibles.

Uno de éstos problemas es que gran parte de la población nacional sufre problemas auditivos. La tecnología debe ser encausada para solucionar y ayudar en la medida de lo que sea posible a solventar estas deficiencias.

En el caso de nuestro país la solución a este problema puede ser comprar tecnología o desarrollar una propia; esto último es posible siempre y cuando no se lleve mucho tiempo y recursos económicos.

La inquietud de nosotros es colaborar a un desarrollo integral de las personas sordomudas, ya que éstas como todos los minusválidos no cuentan con una solución apropiada a su problema que les permita vivir normalmente en la sociedad. Esto es lo que nos motiva a contribuir en el desarrollo del LETOVISIMNOVER. Es conveniente recordar que este trabajo forma parte de todo un sistema de educación para sordomudos, el cual está en desarrollo paralelamente, por lo mismo es factible de modificar.

La meta básica en el diseño de un sistema de este tipo es que funcione como se había previsto, que sea confiable, de fácil mantenimiento y que tenga bajo costo. La simplicidad en el diseño es la clave para alcanzar esta meta. En este trabajo se trata de satisfacer las necesidades antes mencionadas, tomando en cuenta como factor más importante de diseño el que todos los componentes utilizados fueran de bajo costo y fáciles de encontrar.

Existen múltiples soluciones a dicho problema, sin embargo consideramos que ésta es la más adecuada debido a las condiciones económicas y tecnológicas del país. Como este cuenta con muy poco o nada de diseños como el que se presenta en esta tesis, puede llegar a ser muy comercial y de gran aceptación, ya que el objetivo es que esté al alcance de la mayoría de la gente con pocos recursos.

La importancia del presente diseño radica en que la mayoría de los trabajos desarrollados para ayudar a las personas con deficiencias auditivas, no incluyen a personas sordas profundas y este diseño se realizó tomando en cuenta a personas con cualquier problema auditivo, por grave que sea.

Este trabajo es factible de posteriores modificaciones las cuales se harán en base a las necesidades y sugerencias del grupo de personas al que va dirigido, de acuerdo a la experiencia obtenida por estos al utilizarlo.

En el desarrollo de la presente tesis, están involucrados una gran diversidad de temas de la carrera, desde comunicaciones, sistemas digitales, sistemas analógicos, aspectos fisiológicos, etc. Dichos temas se exponen en algunos capítulos de una forma elemental con el fin de que el lector tenga al alcance la mayor información posible.

Por lo anterior se concluye que los objetivos se cumplen bajo los requerimientos enumerados al principio de la tesis.

APENDICE A.

FUENTE DE ALIMENTACION.

Debido a que el sistema debe ser portátil, éste debe poder trabajar con baterías o con voltaje alterno. Para satisfacer este requisito, se diseñaron ambas fuentes de alimentación, las cuales no son las definitivas debido a que el sistema presentado es sólo una parte del proyecto LETOVISIMNOVER.

A.1 Diseño de la fuente de alimentación de corriente alterna.

Esta fuente de alimentación fué diseñada para satisfacer las necesidades del sistema, las cuales son: proporcionar al sistema los valores de voltaje y corriente indispensables para que cada Circuito Integrado (CI) funcione de la forma más eficiente.

Para obtener las características de los CI fué consultado el manual de especificaciones técnicas del fabricante, del cual se obtuvieron el voltaje y la corriente de polarización máxima de cada circuito. Por ejemplo, el CI 74LS04 contiene 6 inversores, tiene un voltaje de polarización máximo de 5.5 V y una corriente máxima de 6.4 mA, estos valores fueron obtenidos para cada uno de los CI utilizados en el sistema, dando como resultado la siguiente tabla de valores (tabla A.1.1).

En esta tabla se observa que se tiene la necesidad de diseñar una fuente de alimentación con tres valores diferentes de voltaje, 5 y 12 Volts positivos y -12 Volts negativos. La corriente máxima total del sistema es la suma de la corriente máxima de polarización de cada uno de los CI utilizados en el sistema (el peor caso de operación de los circuitos). De la tabla se obtuvo un valor máximo de 970 mA, para asegurar un óptimo funcionamiento, se tomó como parámetro de diseño 1 A de corriente, el cual satisface perfectamente las necesidades máximas del sistema.

Un parámetro importante en el diseño de la fuente es el factor de regulación de voltaje, mismo que tiene que ver con la etapa reguladora.

- Regulación de Voltaje.

El valor del voltaje a la salida de la fuente de alimentación es diferente cuando se encuentra en vacío (sin carga) o cuando se le drena corriente (con carga) (fig. A.1.a).

CANTIDAD	CIRCUITO	VOLTAJE DE OPERACION [U]	CONSUMO DE CORRIENTE [mA]			SUBTOTAL
			MAXIMA	TIPICA	MAXIMA	
1	74LS04	+ 5	12		24	24
1	74LS14	+ 5	39		60	60
1	74LS32	+ 5	5		10	10
1	74LS130	+ 5	6,3		10	10
4	74LS244	+ 5	32		54	216
1	74LS245	+ 5	64		95	95
3	74LS279	+ 5	3,8		7	21
2	74LS373	+ 5	40		40	80
1	CPU 2-80	+ 5		150		150
1	SIO 2-80	+ 5		100		100
1	2716	+ 5		100		100
1	6116	+ 5		35		35
1	MC14411	+ 5		35		35
1	XR-2206	+ 12		20		20
1	XR-2211	+ 5		9		9
1	TL072	+/- 12		5		5
22 CIRCUITOS			TOTAL = 970 mA			

Tabla A.1.1 Corriente maxima consumida por el sistema.

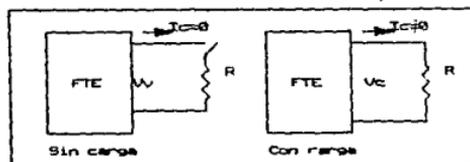


Fig. A.1.a Variacion de voltaje.

Esta variacion de voltaje recibe el nombre de factor de regulacion de voltaje, que expresado en forma matematica es:

$$Vr = [(Vv - Vc) / Vc] * 100$$

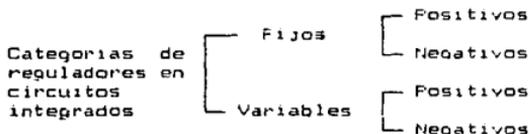
Donde :

- Vr - Factor de regulación.
- Vv - Voltaje en vacío.
- Vc - Voltaje a plena carga.

El regulador es un dispositivo diseñado para disminuir al máximo las variaciones de voltaje, es decir, trata de mantener en cero el factor de regulación de voltaje. En el mejor de los casos debería ser igual a cero, aunque esto en la realidad es imposible. Estos dispositivos proporcionan un buen margen de confiabilidad, mientras no se les demande mas corriente de la que pueden manejar.

Existen diferentes tipos de reguladores de voltaje; en forma discreta, por ejemplo el diodo Zener, reguladores con transistores, etc.; existen también en circuitos integrados, los cuales nos brindan un buen margen de confiabilidad.

Las diferentes categorías de reguladores en CI se muestran en el siguiente diagrama.



Estos reguladores pueden seleccionarse para manejar corrientes del orden de los miliamperes, hasta varias decenas de amperes y regular diferentes valores de voltaje.

Los reguladores utilizados son de la serie 78XX para voltajes positivos y de la serie 79XX para voltajes negativos. Estos se escogieron porque son muy comerciales y soportan una demanda máxima de 1.5 Amperes, el regulador de +5 v es el que más corriente demanda (tabla A.1.1) y su consumo es mucho menor a 1 Amper. Las citras xx proporcionan el voltaje de regulación del CI, por ejemplo, el CI 7805 es un regulador de 5 Volts positivo. En este caso se requiere de 3 valores de voltaje para el funcionamiento del sistema, éstos son: +5, +12, y -12 V, por lo tanto, los circuitos a utilizar son:

Para 5 volts positivos el 7805

Para 12 volts positivos el 7812

Para 12 volts negativos el 7912

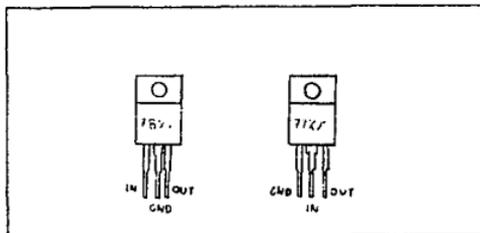
Estos CI emplean un voltaje mínimo de entrada, el cual es diferente para cada uno de ellos, esto se muestra en la tabla A.1.2.

CI	Voltaje regulado (V)	Voltaje mínimo de entrada (V)	Corriente máxima que soporta (A)
7805	+5	+ 7.3	1.5
7812	+12	+ 14.6	1.5
7912	-12	- 14.6	1.5

Tabla A.1.2 Voltajes mínimos de entrada.

Los valores de voltaje mínimo de entrada son muy importantes para la correcta elección del transformador de voltaje, si el voltaje de entrada es menor que el estipulado por el fabricante este no funcionará adecuadamente.

Estos circuitos integrados contienen físicamente 3 terminales, la figura A.1.b muestra cual es la posición de cada una de ellas.



IN - Voltaje de entrada GND - Voltaje de referencia
 OUT - Voltaje de salida

Fig A.1.b Diagrama de terminales de las series 78XX y 79XX.

Etapas de la fuente de alimentación.

La fuente de alimentación está formada por los siguientes bloques básicos.

1. Transformador de Voltaje.
2. Etapa Rectificadora.
3. Etapa de Filtrado.
4. Etapa de regulación de Voltaje.

En la figura A.1.c se muestran los bloques básicos de la fuente y las modificaciones que sufre la señal al pasar por cada etapa.

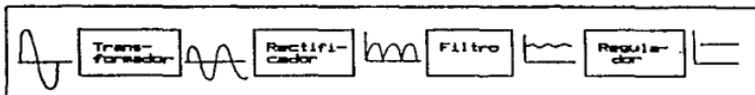


Fig. A.1.c Variaciones en la señal senoidal de entrada.

1. Transformador de voltaje

Este dispositivo eléctrico incrementa o disminuye el voltaje de entrada. Aquí se utiliza un transformador reductor.

El voltaje mínimo necesario para que los reguladores trabajen en forma eficiente es de ± 14.6 Vrms y para asegurar que el transformador proporciona un voltaje mayor que el requerido se utiliza un transformador de 127 a 32 Vrms con tap central (el tap se usa como referencia para poder utilizar voltajes positivos y negativos), es decir, 16 volts serán positivos y los restantes negativos, con esto se asegura que los voltajes suministrados son los correctos. La corriente que maneje el transformador es mayor a la utilizada por el sistema y tendrá un valor de 1 Amper.

2. Etapa rectificadora.

Esta etapa es la encargada de hacer la conversión de voltajes alternos a voltajes directos, es decir, elimina o invierte los lóbulos negativos o positivos, dependiendo del rectificador utilizado. La mayoría de estos circuitos basan su funcionamiento en diodos, los cuales son dispositivos semiconductores que tienen la característica de permitir el paso de la corriente en una sola dirección (fig.A.1.d).

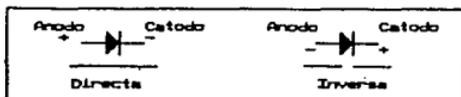


Fig. A.1.d Comportamiento del diodo.

El rectificador de onda completa utilizado se muestra en la figura A.1.e y la forma de onda de la señal rectificada se muestra en la figura A.1.f.

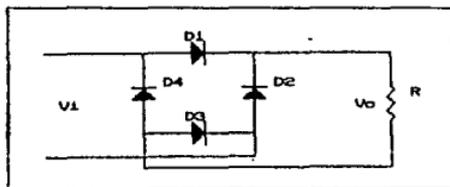


Fig. A.1.e Rectificador de onda completa.

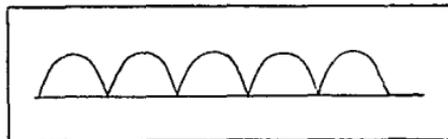


Fig. A.1.f Señal rectificada.

3. Filtrado.

En esta etapa se trata de eliminar al máximo el rizo de la señal rectificadora (fig.A.1.f), para lograr este efecto se utilizó un capacitor de 1000 μf paralelo a la carga, el cual durante la subida del lóbulo se carga y durante la parte decreciente del mismo se descarga en una forma más lenta, después el ciclo vuelve a repetirse. Este proceso da como resultado la señal mostrada en la figura A.1.g.

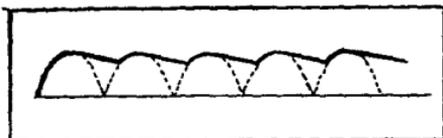


Fig. A.1.g Señal filtrada.

4. Regulación de voltaje.

Esta etapa es la encargada de eliminar los rizados no suprimidos por la etapa anterior, esto se logra colocando un regulador de voltaje entre la etapa de filtrado y la carga como lo muestra la figura A.1.h.

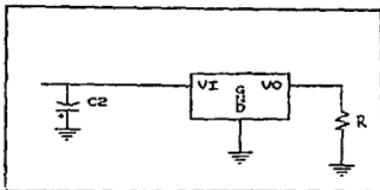


Fig. A.1.h Etapa reguladora.

La conexión de las etapas anteriores, genera la fuente de alimentación que podrá ser utilizada en lugares donde sea posible utilizar voltajes alternos, está se muestra en la figura A.1.i.

A.2 Fuente de alimentación con baterías.

Del análisis de consumo de corriente del sistema se sabe que requiere de 970 mA y tres valores de voltaje para funcionar. Este problema se soluciona generando valores de voltaje superiores a los valores mínimos requeridos por los reguladores (Tabla A.1.2) estos valores son: +14.6 V y -14.6 V; para generar éstos se utiliza el arreglo mostrado en la figura A.2.a.

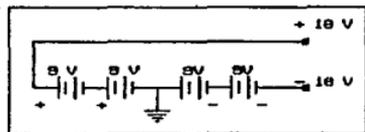


Fig. A.2.a Arreglo de baterías.

El sistema puede tener problemas de alimentación cuando las baterías no son capaces de suministrar los valores mínimos requeridos de voltaje, este caso se presenta cuando las baterías disminuyen 3.4 volts (para los voltajes positivos y negativos), es decir, 1.7 volts por cada una de ellas.

Para evitar que las variaciones de voltaje de las baterías perturben el funcionamiento del sistema éstas se conectarán antes de los reguladores de voltaje.

La elección del tipo de alimentación se llevará a cabo por medio de un switch de 1 tiro 2 polos el cual desconectará el uso del transformador cuando se utilicen baterías y lo conectará al circuito cuando sea necesario. La fuente de alimentación completa se muestra en la figura A.2.b.

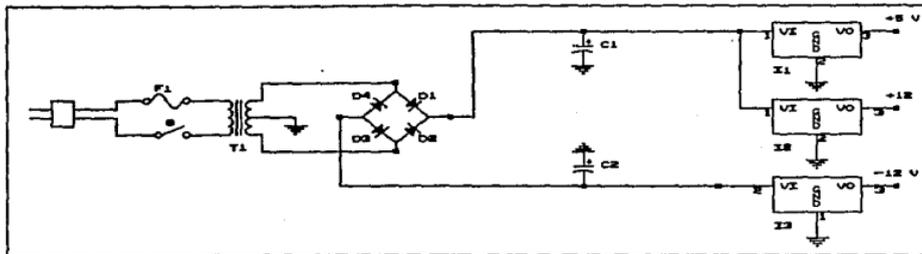


Fig. A.1.1 Fuente de alimentación de A.C.

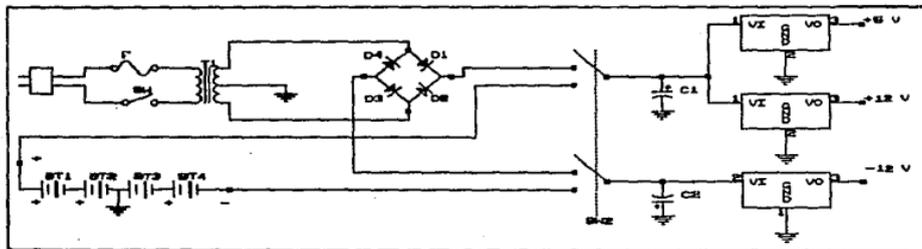


Fig. A.2.b. Fuente de alimentación completa.

BIBLIOGRAFIA

LIBRO	CAPITULO
- Bosco Martínez Juan Apuntes de microprocesadores y sistemas digitales. UNAM Fac. de Ingeniería.	2.3
- Boviestad Nashelsky. Electronica Teoría de Circuitos. Prentice Hall 1983.	Apendice A
- Ciancia Steve. Construya una microcomputadora basada en Z-80. Byte-McGraw-Hill 1984.	3.4
- Cypser. R.J. Communications Architecture for Distributed Systems. Addison-Wesley 1978.	3
- Haydes John P. Diseño de sistemas digitales con microprocesador. Mc-Graw Hill.	2
- Jules H. Gilder. Telephone accessories you can build. Hyden Edit.	3
- Lathi B. P. Sistemas de Comunicación. Interamericana 1986.	3
- Mc Namara John E. Technical Aspects of data Communication. Digital Equipment Corp. 1978.	3
- Morris Mano M. Lógica digital y diseño de computadores Prentice-Hall Hispanoamericana, S.A. 1984.	2,3

- Peatman Jhon B. 3
Microcomputer based design.
International student edition.
- Schwartz Misha. 3
Information, transmission, modulation
and noise.
International student edition.
- Strembler Ferrel. 3
Sistemas de Comunicacion.
Fondo Educativo Interamericano 1985.
- Taub Hebert. 3
Digital Circuits and Microprocessors.
Mc Graw-Hill Book Company 1982.
- Uffenbeck Jhon E. 2,3,4
Microcomputers and Microprocesors:
The 8080, 8085 and Z-80 programming,
interfacing and troubleshooting.
Englewood Cliffs, New Jersey.
Prentice-Hall 1985.
- Ziemer R.E. y Tranter W.H. 3
Principios de comunicaciones
sistemas, modulacion y ruido
Edit. Trillas, Mexico 1987

MANUALES.

- Exar Co. 3
Data Book.
1987.
- Motorola Semiconductor Product. 3
The Semiconductor Data Library.
- Motorola Inc. 3
Motorola memory data.
U.S.A. 1978.
- Motorola Inc. 3
Motorola CMOS data.
U.S.A. 1984.

- Mostek. 2.4
Z-80 Microcomputer Device Technical Manual
- National Semiconductor Corporation. 3
MOS/LSI Data Book.
- Texas Instrument Incorporated. 3
Microprocessors/Microcomputer/
Systems Design.
- Zilog Inc. 4
Z-80 Family Data Book.
Enero 1989

REVISTAS

- Electronic Engineering, London Inglaterra. 3
Vol. 54 No. 667 pag. 33-42 1982.
Single-Chip FSK Modem Cost and Design.
- Galera Garcia C. 1
Se estudia un sistema tactil vibratorio
que substituya al oido atrofiado.
Gaceta UNAM. Organo Informativo de la
Universidad Nacional Autónoma de
México. Octubre 9, 1980.
- Galera Garcia C. 1
Vibratory stimulation to the teeth as a
communication aid for profoundly deaf
persons.
International Journal of Neuroscience.
Vol. 21, pag. 293-294.
- Galera Garcia C. 1
Sistema vibratorio visual en la terapia
del lenguaje y la audicion.
Boletín de la sociedad Mexicana de
Ciencias Fisiológicas.
Vol. 8, Núm. 2, Julio.