DESARROLLO E IMPLEMENTACIÓN DE UN TELÉFONO DE TEXTO PARA LA COMUNICACIÓN ENTRE PERSONAS NO OYENTES

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ciencias y Humanidades
Departamento de Ingeniería Electrónica

DESARROLLO E IMPLEMENTACIÓN DE UN TELÉFONO DE TEXTO PARA LA COMUNICACIÓN ENTRE PERSONAS NO OYENTES

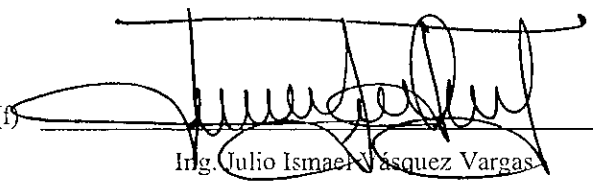
TRABAJO DE GRADUACIÓN PRESENTADO PARA OPTAR AL TÍTULO DE LICENCIADO EN INGENIERÍA ELECTRÓNICA

CARLOS MANUEL DE LEON VALLE

Guatemala

2003

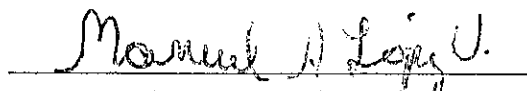
Vo. Bo.:

(f) 

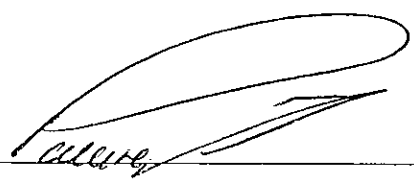
Ing. Julio Ismael Vásquez Vargas

Asesor

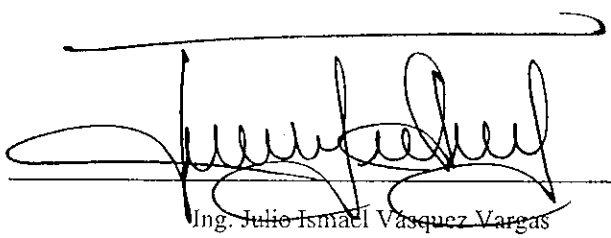
Tribunal:

(f) 

Dr. Ing. Manuel Antonio López

(f) 

Ing. Gonzalo Antonio Palarea Murga

(f) 

Ing. Julio Ismael Vásquez Vargas

Asesor

Fecha de aprobación: 16 de enero de 2004

Le dedico este trabajo de graduación a Dios, a mis papás, mis hermanos y a mi novia.

ÍNDICE

Página

LISTA DE TABLAS

Página

Tabla

LISTA DE FIGURAS

Página

Figura

RESUMEN

El objetivo principal de este trabajo de graduación es diseñar, construir e implementar un teléfono de texto para líneas «*POTS*» (*plain old telephone service*), por el cual las personas no oyentes podrán establecer una comunicación telefónica, sin necesitar ayuda de una persona oyente y a un precio menor que los ya existentes. Todo esto se logró, mediante la transmisión por la línea telefónica «*POTS*» de una señal serial asíncrona, la cual utiliza una modulación y demodulación de *FSK* por *software*. El teléfono de texto diseñado tiene un costo de $82.74 y una velocidad de transmisión de 285 *bits* por segundo. Con esto, el usuario logra comunicarse a una velocidad máxima de 342 palabras por minuto. Además, el teléfono de texto es capaz de detectar una llamada entrante por medio del encendido y apagado de un *led* y una lámpara que se le conecte a éste. También, tiene la capacidad de detectar si la línea a la cual se está llamando se encuentra ocupada o no, al igual de detectar el momento en que se establece conexión con el teléfono de texto receptor.

I. INTRODUCCIÓN

Desde la invención del teléfono por Graham Bell en 1876, millones de personas han visto mejorada su calidad de vida con un servicio que, sin embargo, ha permanecido inaccesible para el colectivo de personas no oyentes hasta que se desarrollaron los primeros teléfonos de texto. Estas terminales posibilitan la comunicación a través de un intercambio de caracteres en tiempo real, sustituyendo los dispositivos de habla y de escucha de una terminal telefónica convencional, por un teclado y una pantalla que permiten al usuario escribir la frase que desea enviar a su interlocutor, y leer lo que éste responde.

Estos teléfonos de texto, mejor conocidos como TTY (*teletypewriters*), han venido evolucionando con el paso del tiempo, logrando así, la comunicación telefónica de personas no oyentes en los países desarrollados. El problema con estos teléfonos de texto es que su costo es muy elevado y que la interfase con la persona no oyente depende del modelo de teléfono y del país en que se adquiera, pues en Estados Unidos ni siquiera se utiliza el mismo protocolo que en Europa. Asimismo, estos no detectan si está ocupada la otra línea o no, además de no detectar el momento en que se establece conexión. De igual forma, para un país subdesarrollado como Guatemala, estos teléfonos están prácticamente fuera del alcance de las personas no oyentes, pues sus precios son elevados y no se encuentran disponibles en el mercado.

El objetivo del proyecto en que se enmarca el presente trabajo de graduación es crear un teléfono de texto a un costo menor que los teléfonos de texto actuales, así como lograr la comunicación inasistida entre personas no oyentes. Esto se logrará al diseñar, construir e implementar un teléfono de texto, por el cual personas no oyentes podrán establecer una comunicación telefónica completa, sin necesitar de alguna ayuda de una persona oyente.

Este teléfono de texto tendrá como interfase con el usuario un teclado *PS2* y una pantalla *LCD* de 80 caracteres. Mediante el uso del teclado, el usuario podrá ingresar su conversación y por medio de la pantalla, el usuario podrá leer tanto lo que él escribe como lo que le escribe su interlocutor. Además, el TTY (se refiere al teléfono de texto) tendrá dos modos de implementación: de transmisión y de recepción. Al encontrarse en transmisión, el usuario sabrá si la línea está ocupada o no, además de poder reconocer el momento en que se establece conexión con otro TTY. Cuando el teléfono de texto se encuentre en recepción, éste prenderá y apagará una luz constantemente cuando se detecta una llamada entrante. Además, el usuario también podrá reconocer el momento en que se establece conexión con el TTY transmisor.

La forma de modulación y demodulación a utilizar será de *FSK (frequency shift keying)*, la cual utilizará dos señales de distintas frecuencias para la transmisión de datos por la línea de teléfono «*POTS*». La

modulación y la demodulación de los datos se realizarán por software, mediante la utilización de un microcontrolador PIC16F877.

II. ANTECEDENTES

A. Historia de TTY

El teléfono de texto, conocido en la comunidad no oyente como TTY, fue desarrollado por un físico no oyente, el Dr. Robert Weitbrecht, a mediados de 1960, basado en la tecnología de teletipo, «*teletype*» ya existente. Usada para la transmisión de telegramas y usos militares. La red de TTY creció dentro de la comunidad no oyente porque la tecnología, aunque antigua y lenta, es confiable y trabaja bien en un ambiente de voces. Desafortunadamente, el TTY utiliza un código, frecuencia y velocidad de transmisión de datos diferente de los utilizados por las computadoras.

Figura 1. Teléfono de texto

En Europa, el teléfono de texto se desarrolló después que en Estados Unidos. A diferencia de los Estados Unidos, la telefonía de texto tendió a desarrollarse por medio del diseño y decreto de las autoridades centrales de varios países europeos. Como resultado, existen varios tipos diferentes de protocolos usados para los teléfonos TTY, los que son incompatibles. Las personas que utilizan el TTY no tienen el alto nivel de conectividad que las personas que utilizan comunicación por voz: ellos no pueden llamar a otros teléfonos de texto en otros países.

Dick Brandt, un experto en comunicaciones, miembro de varios grupos de estándares, se interesó en este problema cuando empezó a consultar con el Programa de Gravamen de Tecnología y cómo mejorar los estándares de TTY americanos en una forma tal que se mejorara el trabajo entre TTY y computadores. Brandt reconoció inmediatamente la necesidad de una especificación internacional que llevara a la inclusión de los protocolos de TTY en los formas de datos convencionales.

Brandt desarrolló un esquema de interconexión que permite que un modem convencional se comunique con varios protocolos TTY, y este esquema, ahora estandarizado como V.18, fue el último

estudio aprobado por el Grupo de Estudio 14 de la Oficina Internacional de la Unión y Estandarización de Telecomunicaciones (CCITT).

El desarrollo del V.18 fue asistido con el diseño y especificación del Programa de Gravamen de Tecnología de la Universidad de Gallaudet, con inversiones de fabricantes americanos de teléfonos TTY y la asistencia de investigadores europeos. El grupo de Consumidores de Telecomunicaciones para los Sordos Inc. copatrocinó las reuniones americanas para discutir el desarrollo del V.18. El trabajo de Gallaudet fue patrocinado por el Departamento de Educación/NIDDR de los Estados Unidos al crear un Centro de Rehabilitación de Ingeniería e Investigación en Realce de Audición y Dispositivos de Asistencia.

La ITU recomienda la interconexión del V.18 con los siguientes protocolos de TTY:

- Baudot @ 45.45 baud (TTY de Estados Unidos)
- Baudot @ 50 baud (usado en Inglaterra, Australia, y y Sudáfrica, también conocido como Baudot internacional)
- V.21/Bell 103/versión de teléfono de texto (usado en Suecia, Noruega y Finlandia).
- DTMF (usado en Dinamarca, Holanda y otros países)
- EDT (Teléfono de Sordos Europeo, «*European Deaf Telephone*», usado en Alemania, Austria, Suiza y otros países)

Las llamadas con teléfonos de texto son llamadas silenciosas cuando son contestadas. Los usuarios tienen que presionar alguna tecla para llamadas salientes para que sepan que es una llamada TTY. Además, los TTY tienen dos modos de conexión. La mayormente usada es la conexión por acoplamiento acústico. El otro tipo es la conexión directa.

B. Diferentes protocolos para los teléfonos de texto

1. Protocolo Baudot. Este protocolo es el más usado y el más antiguo. Es un protocolo asíncrono *half duplex*, el cual utiliza dos frecuencias: 1400 Hz y 1800 Hz. No requiere un «*handshake*» para establecer conexión entre los dos TTY. No utiliza código ASCII. Usa un código que tiene dos sets de caracteres: letras y figuras. Soporta un total de 60 caracteres que incluyen del 0 al 9, sólo mayúsculas de la A a la Z y otros caracteres especiales. No posee corrección de errores. Además, utiliza 1 *bit* de inicio, «*start bit*», 5 *bits* de datos y 1 *bit* de fin, «*stop bit*».

Existen dos tipos de protocolo Baudot. Se encuentra el Baudot @ 45.45 baud, el cual tiene una velocidad de transmisión de 45 *bits* por segundo y es usado en Estados Unidos y Canadá, y el Baudot @ 50

baud, también conocido como Baudot internacional. Éste tiene una velocidad de transmisión de 50 *bits* por segundo, usado en Inglaterra, Australia y Sudáfrica. Estos dos protocolos tienen que tener una tolerancia para mandar datos del 1% y para recibir del 5%.

2. Otros protocolos. El protocolo que salió como mejora del Baudot es el Turbo Code. Éste manda 7 *bits* de datos en lugar de 5 *bits*, como era el caso del Baudot. El Turbo Code fue creado por Ultratec, compañía que hace teléfonos de texto. Éste le permite escribir a la persona a una velocidad alrededor de 100 palabras por minuto pues tiene una velocidad de transmisión de 110 baudios (*bits* por segundo). Éste utiliza en su protocolo un interrupto, con el que una persona puede interrumpir a otra cuando está escribiendo. Además, es un protocolo *half duplex* y solo los modelos nuevos de TTY de Ultratec lo tienen.

Otro protocolo que existe es el Bell 103, también llamado ASCII Code. Éste es similar al estándar internacional V.21 pero utiliza diferentes frecuencias. Éste puede ser *full duplex* o *half duplex*, por lo que utiliza dos canales y 4 frecuencias. Tiene una velocidad de transmisión de 110 *bits* por segundo o de 300 *bits* por segundo. Éste utiliza 8 o 7 *bits* de datos, además de 1 *bit* de control de paridad. No es muy popular entre los usuarios, debido a su dificultad de configuración y a que el costo de los TTY que traen este protocolo es más elevado. Antes de poder realizar una llamada usando Bell 103, hay que configurar si el TTY es origen o destino (se va a realizar llamada o a contestarla) y si se va a utilizar 110 bps o 300 bps como velocidad de transmisión. Además, hay que configurar si se va a utilizar 8 *bits* de datos o 7 *bits* de datos, así como si se utilizará paridad impar o par.

El protocolo *EDT* (*European Deaf Telephone*) usa la modulación de la norma V.21 pero con un sólo canal, lo que quiere decir que es *half duplex*. La velocidad de transmisión es de 110 *bits* por segundo. Se usa principalmente en Alemania, Suiza, Austria, Italia, España, Malta y algunos otros países.

En Francia y en Bélgica se utiliza el protocolo Minitel V.23. Éste es una versión especial, llamada Minitel Dialogue, que ofrece un terminal Minitel Videotex. Este protocolo utiliza un módem V.23 estándar. Además, en Finlandia, Suecia y Noruega se utiliza el protocolo V.21 Nórdico. En éste, los datos son enviados con una modulación de acuerdo con la norma V.21 y a 300 *bits* por segundo.

El protocolo de comunicación que se utiliza en Holanda y en Dinamarca es el DTMF. Éste está basado en las mismas combinaciones tonales que utilizan los teléfonos convencionales para marcación. Utiliza combinaciones de 1 hasta 4 caracteres tonales para poder representar alfabetos enteros.

Por la diversa cantidad de protocolos TTY, se estableció recientemente el estándar V.18 como se mencionó anteriormente. Éste soporta los protocolos DTMF, EDT, V.21 y V.23, usados en Europa, así

como el Baudot usado en Estados Unidos. Este estándar fue desarrollado con la ayuda de organizaciones para no oyentes, pero muy pocos usuarios no oyentes han tenido la oportunidad de probarlo en su ambiente.

Se observaron los diferentes modelos de teléfonos de texto que tienen a la venta una de las compañías más grande de TTY, Ultratec. Los precios van desde $250.00, siendo un TTY que solo acepta el código Baudot y el cual tiene una pantalla solamente de 24 caracteres; hasta los $1200.00, un TTY que se utiliza como teléfono público, el cual acepta el protocolo Baudot, el Turbocode y el ASCII. Los teléfonos que poseen una pantalla de 40x2 caracteres y aceptan estos tres protocolos oscilan alrededor de los $400.00. Solamente existe un teléfono de texto en Ultratec que acepta ciertos modelos de teléfonos celulares, la mayoría siendo TDMA, el cual es el Compact/C. Éste cuesta, en Estados Unidos, alrededor de $300.00 y trae una pantalla 40x2 caracteres y soporta los protocolos Baudot y Turbocode.

C. Modulación digital *BFSK* (*Binary Frequency Shift Keying*)

En BFSK, la onda de datos binaria d(t) genera una señal binaria

$$v_{BFSK} = \sqrt{2 * P_s} * \cos\left[\omega_0 * t + d(t) * \Omega * t\right],$$

en ésta, d(t) = +1 ó -1, lo que corresponde a los niveles lógicos de uno o cero de la señal de datos. La señal transmitida es de amplitud $\sqrt{2 * P_s}$ y es una de las dos que se muestran a continuación:

$$v_{BFSK} = \sqrt{2 * P_s} * \cos\left[(\omega_0 + \Omega) * t\right]$$

$$v_{BFSK} = \sqrt{2 * P_s} * \cos\left[(\omega_0 - \Omega) * t\right]$$

La señal transmitida tiene una frecuencia angular $\omega_0 + \Omega$ ó $\omega_0 - \Omega$, siendo Ω una constante que se le suma o resta a la frecuencia nominal portadora ω_0. Esta constante es la frecuencia angular de corte del filtro pasa baja por el que se pasa la señal original. Se llamará la frecuencia mayor $\omega_h = \omega_0 + \Omega$ y la frecuencia menor $\omega_L = \omega_0 - \Omega$. Para la modulación *BFSK* se utilizan dos moduladores. Uno con la portadora ω_h y el segundo con ω_L como portadora. Los valores de voltaje de $p_h(t)$ y de $p_L(t)$ son relacionados a los valores de voltaje de d(t) (*vid* tabla 1).

Tabla 1. Valores de voltaje

d(t)	$p_h(t)$	$p_L(t)$
+1V	+1V	0V
-1V	0V	+1V

Se puede observar que d(t) cambia de +1 a -1 cuando $p_h(t)$ cambia de de uno a cero y $p_L(t)$ cambia de cero a uno. En cualquier momento $p_h(t)$ o $p_L(t)$ pueden ser uno pero no lo pueden ser a la vez, para que así la frecuencia angular sea ω_h o ω_L.

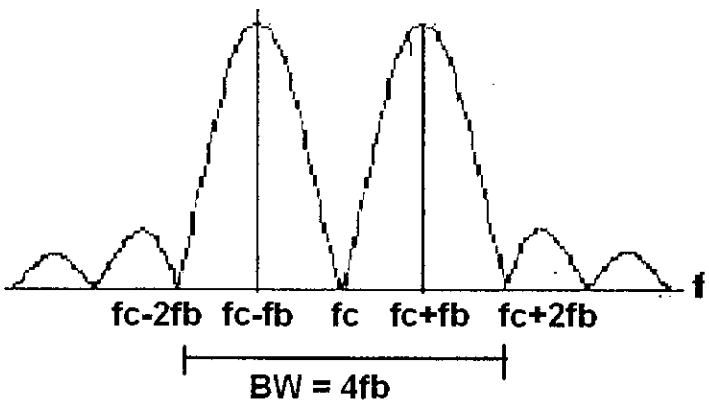
1. Espectro de *BFSK*. En términos de las variables $p_h(t)$ o $p_L(t)$ la señal *BFSK* es

$$v_{BFSK} = \sqrt{2*P_s} * p_H * \cos(2*\pi*f_H*t) + \sqrt{2*P_s} * p_H * \cos(2*\pi*f_L*t)$$

en donde se asume que cada una de las dos señales son independientes, con $\omega_h = 2*\pi*f_H$ y $\omega_L = 2*\pi*f_L$, en donde $f_H = f_C + f_B$ y $f_L = f_C - f_B$. Siendo f_C la frecuencia portadora y f_B la frecuencia de corte del filtro pasa baja o la velocidad de transmisión (V.T.) dividido dos. Al observar su espectro en la siguiente figura, se observa claramente que $f_H - f_L = 2*f_B$. Por lo que se concluye que el ancho de banda (*BW*) para *BFSK* está dado por

$$BW(BFSK) = 4*f_B = 4*\frac{V.T.}{2} = 2*V.T.$$

Figura 2. Espectro de la señal v_{BFSK}



2. Receptor para señales *BFSK*. Una señal *BFSK* es remodulada usualmente por el siguiente sistema. La señal pasa por dos filtros pasa banda, en donde uno tiene frecuencia central f_H y el otro f_L. Las frecuencias escogidas para los filtros no se traslapan y cada filtro es lo suficientemente ancho para contener el lóbulo principal en el espectro. Las salidas de los filtros son aplicadas cada una a un detector de envolvente y finalmente las salidas de los detectores de envolvente son comparadas por un circuito comparador. Este comparador genera una salida binaria, la cual está a un nivel o a otro dependiendo de cual señal de entrada es mayor. A la salida del comparador la señal d(t) será reproducida.

Cuando se encuentra ruido presente, la salida del comparador puede variar. Entonces, en sistemas prácticos se usa un sincronizador de *bits* y un integrador para así realizar un muestreo de la salida del comparador una vez al final de cada intervalo T_B.

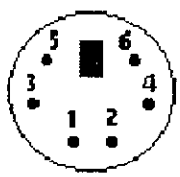
D. Teclado *PS2*

La interfase de comunicación PS/2 es un prototipo planteado por la IBM para comunicar dispositivos seriales en forma sincrónica, tanto teclado como mouse. En la actualidad la mayoría de los teclados corresponden a este tipo con las siguientes características:

- Gran número de teclas (101 a 104).
- Conector de 5 o 6 pines (incluyen adaptadores).
- Protocolo de comunicación serial bidireccional (PS/2).
- Garantizan sólo el conjunto 2 de los scan codes.
- Contestan todos los comandos enviados, sin embargo, no actúan en todos ellos.

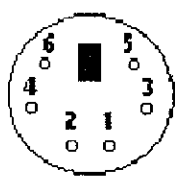
La interfase física que usualmente se utiliza se muestra a continuación:

Figura 3. Conector PS/2 (*Mini-Din*) macho



Plug

Figura 4. Conector PS/2 (*Mini-Din*) hembra



Socket

Tabla 2. Pines de Conector PS/2 (*Mini-Din*)

1	Datos
2	No Implementado
3	Tierra
4	Vcc (+5V)
5	Reloj
6	No Implementado

La alimentación del teclado/*mouse* corresponde a Vcc=+5V y una corriente máxima de 100 mA. Se recomienda no conectar el teclado a la tarjeta mientras se encuentre prendida. Las líneas de *Data* y *Clock* son ambas de colector abierto, con resistencias de «*pull-up*» para fijar ambas líneas en alto. El *mouse* y teclado PS/2 implementan un protocolo de comunicación serial bidireccional. El bus de comunicación se encuentra en estado «*idle*» cuando ambas líneas (*data* y *clock*) se encuentran en alto. Éste es el único estado en que al dispositivo (teclado o *mouse*) le está permitido enviar información al *host* (computadora o en este caso PIC16F877). El *host* tiene el control último sobre el *bus* y puede inhibir la comunicación en cualquier instante, colocando la línea de *clock* en nivel bajo. El dispositivo siempre genera la señal de *clock*. Si el *host* desea enviar datos, debe primero inhibir la comunicación desde el dispositivo, colocando la línea de *clock* en nivel bajo. Luego debe colocar en nivel bajo la línea de *data* y subir la línea de *clock*. Este estado es conocido como «*Request to Send*», con lo cual el *host* señaliza al dispositivo que comience a generar pulsos de reloj a través de la línea *clock* para enviar los datos. Por lo tanto el *bus* puede estar en uno de los tres estados que se muestran en la siguiente tabla.

Tabla 3. Estados del *bus*

Datos	Reloj	Estado
Alto	Alto	«*Idle*»
Alto	Bajo	«Comunicación Inhibida»
Bajo	Alto	«*Request to Send*»

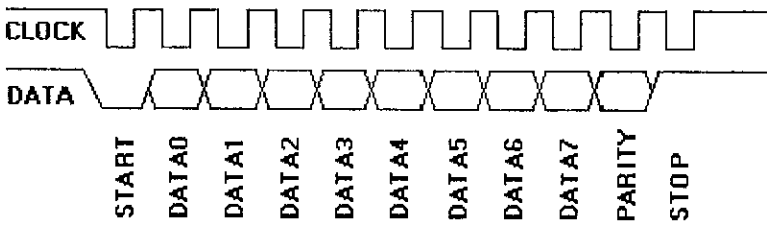
Todos los datos son mandados de un *byte* por vez y cada *byte* es enviado dentro de un frame o trama de 11 o 12 *bits*:

- 1 *bit* de inicio: siempre es 0.
- 8 *bits* de datos: se comienza por el menos significativo.
- 1 *bit* de paridad: se utiliza paridad impar.
- 1 *bit* de parada: siempre es 1.
- 1 *bit* de confirmación: sólo para comunicación desde el host al dispositivo.

El *bit* de paridad es colocado en alto si hay un número par de 1 en los *bits* de datos, y colocado en bajo si hay un número impar. La idea es que el número de 1 de los *bits* de datos más el *bit* de paridad siempre sean un número impar (paridad impar). Esto se utiliza para la detección de errores en la transmisión. Si el dispositivo detecta un error responde como que se le ha enviado un comando inválido.

En el caso de comunicación del dispositivo al *host* (caso que será utilizado) el siguiente diagrama ejemplifica la situación:

Figura 5. Líneas de reloj y de datos para comunicación de dispositivo a *host*



La frecuencia del reloj es de 10-16.7 kHz. Cuando el teclado o *mouse* desean mandar información deben asegurarse que la línea de reloj debe estar en alto por lo menos 50 µs antes de que el dispositivo pueda comenzar a enviar datos. El teclado/*mouse* escribe un *bit* en la línea de datos cuando el reloj está en alto. Además, es leído en el *host* cuando el reloj está bajo. Como se trata de comunicación desde el *host* hacia el dispositivo, no se envía un *bit* de confirmación.

En los teclados se distinguen distintos conjuntos de *scan codes* (códigos de teclado) para identificar la tecla o teclas que se presionan. Se utiliza por defecto el conjunto 2 (*set* 2). Cada tecla tiene asociado un *scan code* compuesto de dos códigos: un *make code*, que se emite cada vez que se presiona una tecla, y un *break code*, que se emite cuando se suelta la tecla. En general estos códigos suelen tener entre 2 y 1 *byte*, pero existen ciertos *scan codes* que son más largos debido a que corresponden a una combinación de teclas más elaborada.

III. OBJETIVOS

A. Generales

1. Diseñar y construir un teléfono de texto (TTY) a un costo menor que los teléfonos de texto actuales.

2. Lograr la comunicación telefónica inasistida entre dos personas no oyentes.

B. Específicos

1. Desarrollar un protocolo de transmisión, el cual transmita a una velocidad mayor de 100 *bits* por segundo.

2. Lograr una transmisión serial asíncrona *half duplex* por la línea de teléfono «*POTS*» mediante la utilización de *FSK*.

3. Al realizar una llamada, que la persona no oyente logre reconocer si el teléfono al que desea comunicarse se encuentra ocupado o no.

4. Lograr que la persona no oyente sea capaz de detectar una llamada entrante.

5. Lograr que tanto el receptor como el transmisor pueda reconocer el momento en el que se establece conexión con el otro teléfono de texto.

6. Lograr que la persona no oyente, se logre comunicar con una velocidad mayor a las 100 palabras por minuto.

IV. MÉTODO DE TRABAJO

El método de trabajo utilizado, consistió en diseñar e implementar los diferentes elementos que se necesiten para poder realizar dos estaciones de teléfonos de texto, TTY (*teletypewriter*), que se comuniquen por medio de la línea telefónica «*POTS*» (*plain old telephone service*).

A continuación se presentan los elementos que componen, logran y prueban el funcionamiento de cada una de las estaciones TTY.

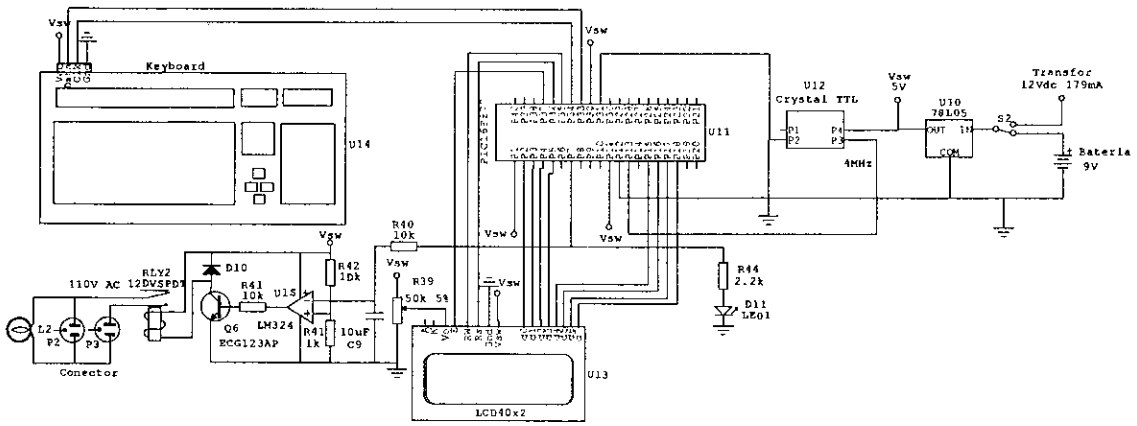
A. Interfase de transmisión y recepción entre persona no oyente y teléfono de texto

Se utilizó un teclado PS2 conectado a un microcontrolador PIC16F877 por los puertos rb0 y rb1 (*vid* figura 6). Éste utiliza una transmisión serial síncrona con una línea de reloj y otra de datos. La línea de reloj entra por el puerto rb1, mientras que la línea de datos utiliza el rb0. Los datos son transmitidos en código ASCII. Este teclado se utiliza para transmitir la información que la persona desee.

Además, se utilizó una pantalla *LCD* (*liquid crystal display*) 40x2 paralela conectada al microcontrolador PIC16F877 usando 8 pines de datos y 3 pines de control (*vid* figura 6). Los puertos de datos son del ra0 al ra3 y del rc0 al rc3. Siendo el ra0 el *bit* menos significativo y el rc3 el más significativo. Los tres pines de control son el rb2 utilizado como el «*read/write*» (lectura/escritura), el rb3 utilizado como el RS (selecciona datos o instrucciones) y el rb4, el cual es el enable (habilita señal). Esta pantalla sirve de interfase entre la persona que no oye y el teléfono para recepción y transmisión, pues tanto el teclado como el puerto serial de recepción le manda el caracter a desplegar.

Para la detección de una llamada entrante (*ring*), se utilizó una lámpara y un *led*, los cuales se encienden de manera intermitente hasta que se contesta el teléfono de texto (*vid* figura 6). Esta lámpara, al ser conectada a una espiga hembra que está conectada a un relé, es encendida y apagada al cerrar y abrir este relé que es abierto o cerrado mediante el puerto re2 del PIC16F877.

Figura 6. Circuito que sirve de interfase para pantalla *LCD*, lámpara y teclado PS2



B. Forma de transmisión del TTY

La forma de transmisión del teléfono de texto tanto de entrada, como de salida es de forma serial asíncrona *half duplex* a 285 bps (*bits* por segundo), usando 1 *stop bit*, 1 *start bit* y 8 *bits* de información en código ASCII.

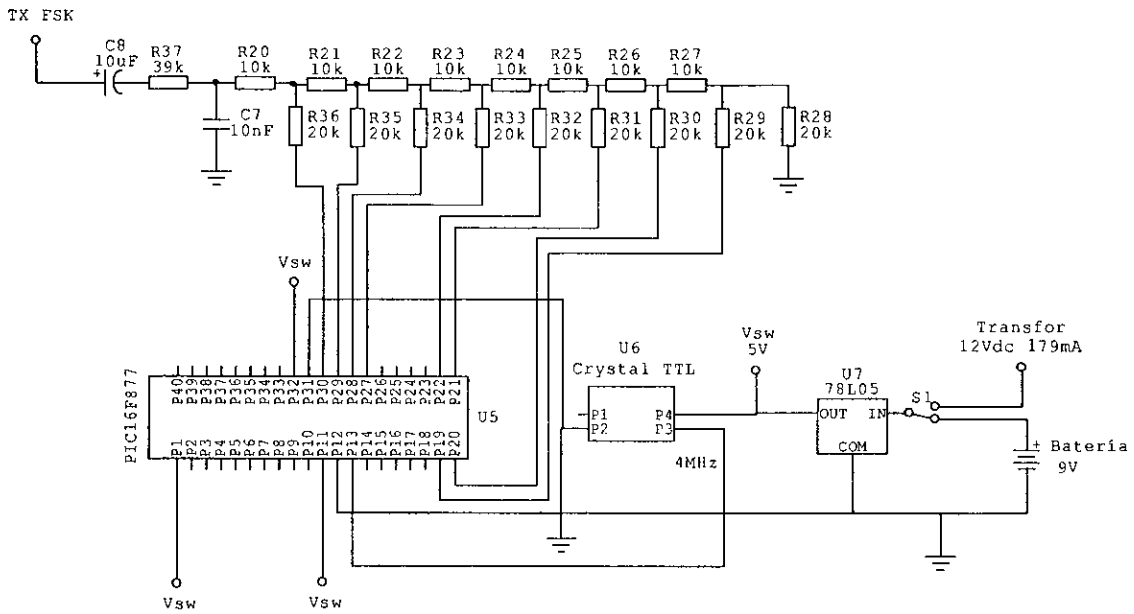
1. Modulación FSK. La señal a transmitir es modulada utilizando *FSK* (*frequency shift keying*) para mandarla por la línea de teléfono «*POTS*». Esta modulación se hace por *software* en el microprocesador PIC16F877 de la siguiente manera:

Se convierte la señal digital a una señal analógica por *software* adentro del PIC16F877, en donde un «0» digital equivale a una señal senoidal de frecuencia 2.63 kHz y un «1» digital en una señal senoidal a una frecuencia de 1.49 kHz. Los valores de estas señales senoidales salen por el puerto D (8 *bits*) del PIC16F877, siendo el rd0 el *bit* menos significativo y el rd7 el *bit* más significativo. Después pasa esta señal por un DAC (convertidor digital-analógico) de tipo escalera R-2R (*vid* figura 7), en donde se vuelve analógica la señal y está lista para ser enviada por la línea de teléfono «*POTS*» por medio del circuito transmisor que se presentará más adelante.

Tabla 4. Modulación *FSK*

Señal digital	Frecuencia de señal analógica
0	2.63 kHz
1	1.49 kHz

Figura 7. Circuito modulador de *FSK* con convertidor digital-analógico escalera R-2R



2. Transmisión de la señal a la línea telefónica «*POTS*». La transmisión por la línea de teléfono «*POTS*» de la señal modulada se realiza de la siguiente manera:

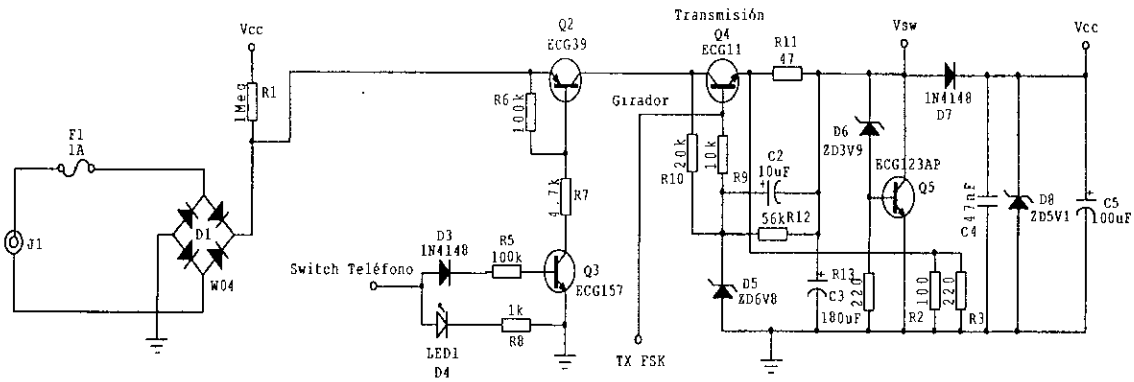
Para el acoplamiento de la línea telefónica «*POTS*» se utiliza un puente de diodos, W04, de 1A y 400V. Éste tiene dos propósitos, uno es que no importando la polaridad de la línea, siempre va a estar la polaridad correcta en el circuito transmisor. El segundo es que el puente ofrece una forma conveniente de mandar señales de audio a la línea telefónica «*POTS*».

Un lado de este puente se conecta al conector RJ11 que va a la línea telefónica «*POTS*», llevando entre el puente y el conector un fusible de 1A, mientras en el otro lado del puente se conecta una terminal a la tierra del circuito y la otra terminal al circuito que baja la línea telefónica «*POTS*».

El circuito que se utiliza para conectar la línea telefónica «*POTS*» consta de un transistor npn y un pnp. Estos transistores, Q3 ECG157 y Q2 ECG39, tienen que resistir el voltaje alto del *ring* por lo que

tienen un voltaje de colector a emisor de 300V. Lo que realiza este circuito a base de un transistor npn y un transistor pnp es un *switch*, el cual no está conectado cuando no hay voltaje en la terminal «*switch* de teléfono», lo que hace que el diodo D3 no conduzca y no se cierre el *switch* creado por Q3. Entonces la base de Q2 queda abierta y no deja conducir corriente por R16 (47Ω). Cuando se conecta la línea para así poder transmitir y recibir de la línea telefónica «*POTS*», se alimenta un voltaje de 5V en la terminal «*switch* de teléfono». Esto hace que el diodo D3 conduzca y se cierre el *switch* creado por Q3. Lo que produce que halla una corriente en la base de Q2 lo que hace que fluya una corriente en el colector por R16 y se cierre el *switch* de Q2 con lo que se baja la línea «*POTS*».

Figura 8. Circuito transmisor a línea telefónica «*POTS*»



El circuito girador es el que sirve como el transmisor de la señal a la línea telefónica «*POTS*». Este circuito tiene dos funciones. La primera es la de no dejar pasar la señal AC entrante de la línea telefónica «*POTS*», pues sirve como un simulador de inductancia al ofrecer una alta impedancia para las frecuencias deseadas. Al no dejar fluir la señal AC entrante de la línea, ésta no puede ser disipada en este circuito, lo que permite que ésta pase directo al circuito de recepción que se discutirá más adelante.

El girador es un amplificador de corriente de base de un transistor npn. Este transistor es Q4 ECG11 en la figura 8. La base es mantenida a un voltaje usando las resistencias R9 y R10 como un divisor de voltaje. El colector es la entrada del girador y la salida es el emisor. El capacitor C2 de 10μF que está conectado en paralelo con la resistencia debajo de la base simula una inductancia aproximada a 5H. Por lo que el girador usa al transistor para convertir la capacitancia en la base en una inductancia simulada en el colector.

Al ser la entrada del girador la señal DC de la línea telefónica «*POTS*», entonces la base es mantenida a un voltaje estable y el girador permanece en un estado estable de conducción. Cuando se encuentra una señal AC en el colector, una fracción de ésta trata de aparecer en la base, pero al pasar por la resistencia R10 y el capacitor C2, estos actúan como un filtro pasa baja y solo permite que fluya por la base señales de baja frecuencia. Entonces, al no haber corriente alterna fluyendo en la base, el transistor no

actúa como amplificador para las señales AC que entran de la línea telefónica «*POTS*» por lo que aparenta una alta impedancia para ellas.

La segunda función que tiene el circuito girador es la de amplificar la señal que proviene de la escalera R-2R y de mandarla por la línea telefónica «*POTS*». Como se mencionó anteriormente, el girador es fundamentalmente un amplificador de corriente. Al estar R9 conectada entre la base y C2 (*vid* figura 8), permite que una señal independiente de audio sea aplicada a la base por medio de C8, un capacitor de acoplamiento de 10μF. Esto permite que el girador no solamente bloquee la señal AC, sino que amplifique la señal generada en el PIC16F877 y la introduzca a la línea. Esto se obtiene gracias a que la señal AC proveniente de la escalera R-2R no es filtrada antes de llegar a la base por lo que es amplificada por Q4.

C. Forma de recepción del TTY

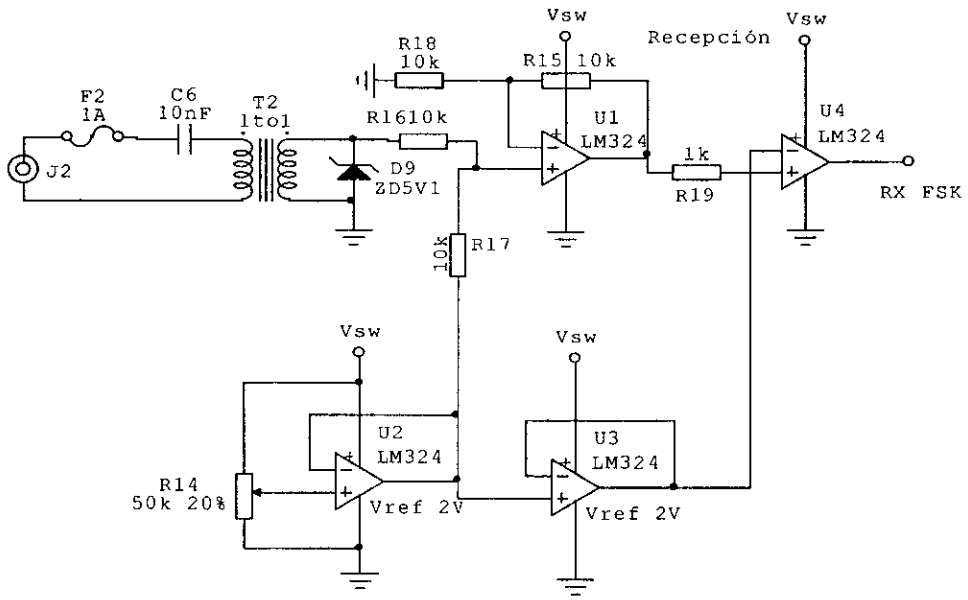
La recepción del teléfono de texto consta de dos partes. La primera es la recepción de la línea de teléfono «*POTS*», mientras que la segunda es acerca de la recepción y demodulación *FSK* que se hace por *software* utilizando el PIC16F877.

1. Recepción de la línea de teléfono «*POTS*». La recepción de la señal proveniente de la línea de teléfono «*POTS*» se realiza de la siguiente manera:

Para la recepción de la señal AC que viene en la línea telefónica «*POTS*» se utiliza un capacitor de acople C6 y un transformador de audio 1:1 con 600Ω de impedancia en ambos lados. Un lado del transformador se conecta a la línea telefónica «*POTS*» acoplada por el capacitor C6; mientras en el lado secundario del transformador, una de las terminales se une a la tierra del circuito del teléfono de texto y la otra terminal es la que lleva la señal AC que se va a demodular. Esta señal pasa por un zener D9 de 5.1V para no dejar pasar señales mayores a los 5.1V como sería el caso del *ring*.

La señal AC ya desacoplada de la línea telefónica «*POTS*» se suma con una señal DC de referencia de 2V por medio de un circuito sumador no inversor realizado con un amplificador operacional LM324. Esta señal que contiene a la señal AC sobre un voltaje de referencia se pasa por un comparador no inversor, el cual tiene como voltaje de referencia el mismo que fue sumado a la señal de entrada para poder detectar solamente la señal AC. De esta manera se logra que en la salida del comparador (*vid* figura 9) se encuentre una onda cuadrada con la misma frecuencia que la señal AC y con un ciclo de trabajo del 50%, el cual es indispensable para la demodulación de la señal.

Figura 9. Circuito de recepción



2. Demodulación *FSK*. La demodulación de la señal se realiza cuando la onda cuadrada que sale del comparador entra al puerto rb7 del PIC16F877. Ésta se lleva a cabo de la siguiente manera:

Este puerto utiliza el interrupto por cambio de flanco al igual que el módulo del *Timer*1 (cuenta las instrucciones) con un *pre-scaler* de 1:4. El rb7 es utilizado de esta forma para poder detectar la frecuencia de la onda entrante y así poder demodular la misma. Lo que hace el programa para demodular la onda es lo siguiente: al momento que hay un cambio de flanco en la señal, ocurre un interrupto; entonces, se guarda el valor del *Timer*1 en una variable y se borra el registro del *Timer*1. Después se revisa para ver si el valor que se guardó del *Timer*1 está en el rango de 2.3kHz y 2.8kHz o en el rango de 1.3kHz a 1.7kHz. Si está dentro del rango de 2.3kHz y 2.8kHz pone un «0» en el puerto rb5. Si se encuentra dentro del otro rango pone un «1» en el rb5. Si no está adentro de estos dos rangos, va a revisar si en el cambio de flanco anterior estuvo en alguno de los dos rangos. Si tampoco estuvo en algún rango, entonces se sale y deja en «1» el puerto rb5.

El puerto rb5 está conectado al puerto rc6 del PIC16F877. Este puerto es utilizado por el módulo *USART* (*universal synchronous asynchronous receiver transmitter*) del PIC16F877 como el puerto de recepción cuando este módulo está utilizado en modo asíncrono a 285 bps, con un *start bit*, 8 *bits* de datos, sin paridad y un *stop bit*. De ésta forma, la señal demodulada que sale por rb5 entra al receptor del puerto serial y al momento que tiene un *byte*, el puerto serial realiza un interrupto y manda a desplegar en la pantalla *LCD* el caracter enviado.

D. Modo de operación

El teléfono de texto funciona únicamente en líneas «*POTS*». A continuación se presenta el modo de operación del TTY para líneas telefónicas «*POTS*».

1. Recepción. El teléfono de texto se conecta a la línea de teléfono «*POTS*» mediante un conector RJ11. Éste, al encontrarse como receptor, detecta el *ring* de la siguiente forma: la señal del *ring* pasa por el circuito de recepción y entra al puerto rb7 del microcontrolador. En éste ocurre un interrupto por flanco y lo despierta, es aquí donde se va a revisar el tiempo muerto (tiempo en que se queda muda la línea telefónica «*POTS*»). Si es menor de 4.2 segundos, se está detectando un *ring*, mientras que si es mayor quiere decir que no se detectó el *ring* y se despliega un mensaje en la pantalla *LCD* que dice «ERROR EN CONEXIÓN».

Al detectar el *ring*, el puerto re2 genera una onda cuadrada, la cual activa un relé que prende y apaga una lámpara. Además, un *led* parpadea a la vez. Este *led* es utilizado, dado el caso en que el teléfono de texto estuviera alimentado por una batería. La lámpara y el *led* indican que está entrando una llamada. Se contesta el teléfono de texto al presionar la tecla *ENTER* en el teclado. En el momento que se presiona *ENTER*, el *ring* se apaga y por consiguiente la lámpara también. Se pone en alto el puerto rb6 el cual está conectado a la terminal «*switch* teléfono» en el circuito transmisor de la figura 8 con lo que se conecta la línea telefónica «*POTS*». Además, el TTY receptor empieza a establecer la conexión con el TTY transmisor. Los dos TTY hacen un «saludo» (*handshake*) y encienden un *led* al poner en alto el puerto re2; para así, avisar a los usuarios que ya están conectados y pueden empezar la comunicación.

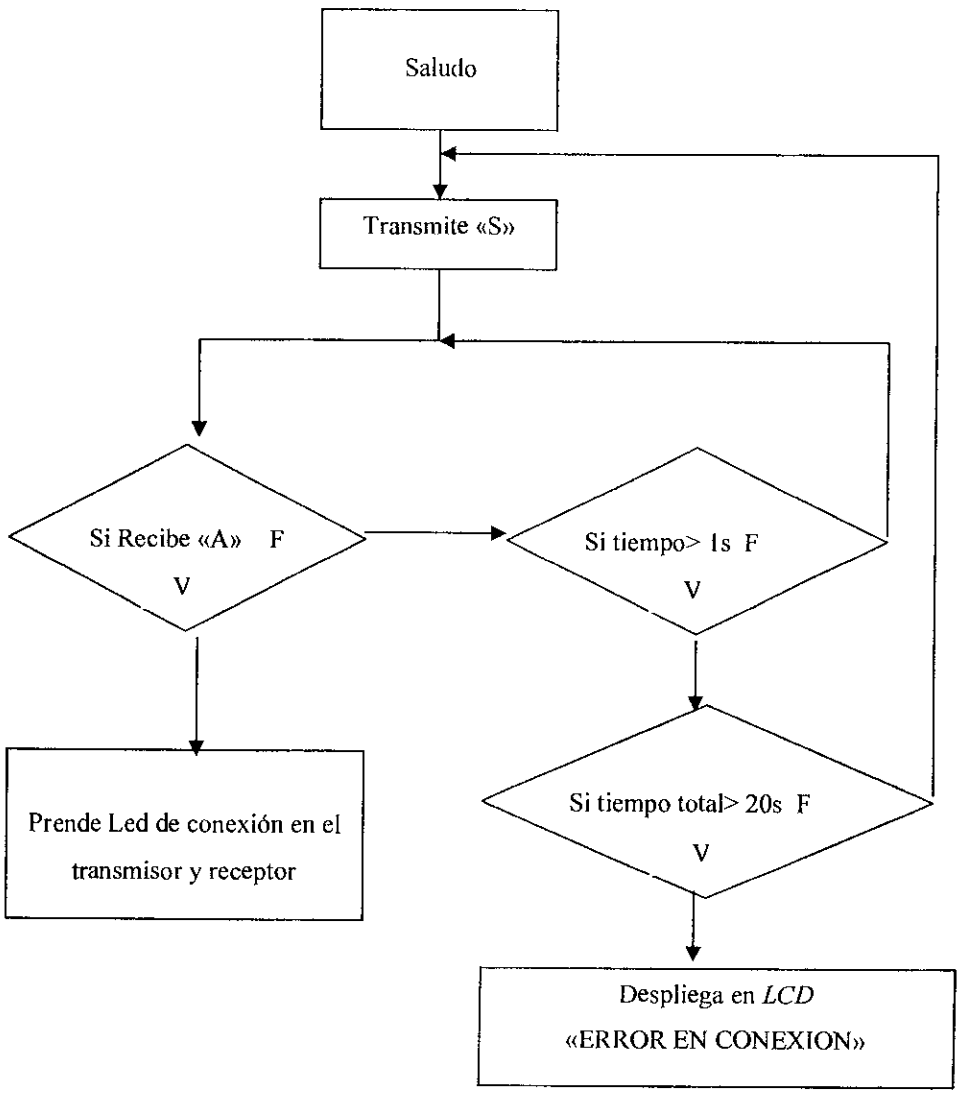
2. Transmisión. El teléfono de texto transmisor luego de haber marcado en un teléfono normal, procede a chequear si da línea o está ocupada la línea. Por medio del circuito de recepción, cualquiera de las dos señales entran por el puerto rb7 y hacen un interrupto por cambio de flanco y se va a revisar los tiempos muertos. Si el tiempo muerto es menor de 0.5 segundos, indica que está ocupado, y despliega «OCUPADO». Si el tiempo muerto es mayor de 0.5 segundos y menor que 4.2 segundos, significa que está dando línea, por lo que despliega en la pantalla «LINEA».

3. Saludo (*Handshake*). Cuando se contesta el TTY receptor, el TTY transmisor ya no detecta las frecuencias que indican si está dando línea o está ocupado. El transmisor espera 4 segundos para empezar el «saludo» entre los dos teléfonos de texto. Al momento de empezar el saludo el TTY transmisor despliega en la pantalla «CONECTANDO». Este saludo o *handshake* se basa en mandar del teléfono transmisor un *byte* con la letra «S» (de enviar en inglés) en código ASCII cada segundo para que se oiga un tono repetitivo en el teléfono receptor. Así el receptor sabrá que es una llamada TTY, si se da el caso que

levante alguien oyente el teléfono, para que proceda a presionar la tecla *ENTER* en el TTY receptor (*vid* figura 10). El TTY transmisor sigue mandando la letra «S» cada segundo hasta que el receptor se encuentre conectado y la reciba. En ese momento, el receptor manda un *byte* de reconocimiento (*acknowledge*) con la letra «A». En el momento de mandar la «A» el receptor pone en alto el puerto re2 del PIC16F877, el cual prende un led que indica que el TTY receptor se encuentra conectado. Cuando la «A» es recibida en el TTY transmisor, éste pone en alto el puerto re2 del PIC16F877, el cual prende un led indicando que el TTY transmisor se encuentra conectado. Es aquí donde se puede iniciar la conversación.

Si el TTY transmisor no ha recibido un reconocimiento del receptor (un *acknowledge*) después de 20 segundos, el transmisor despliega un mensaje en la pantalla *LCD* que dice «ERROR EN CONEXIÓN» y pone en cero el puerto rb6.

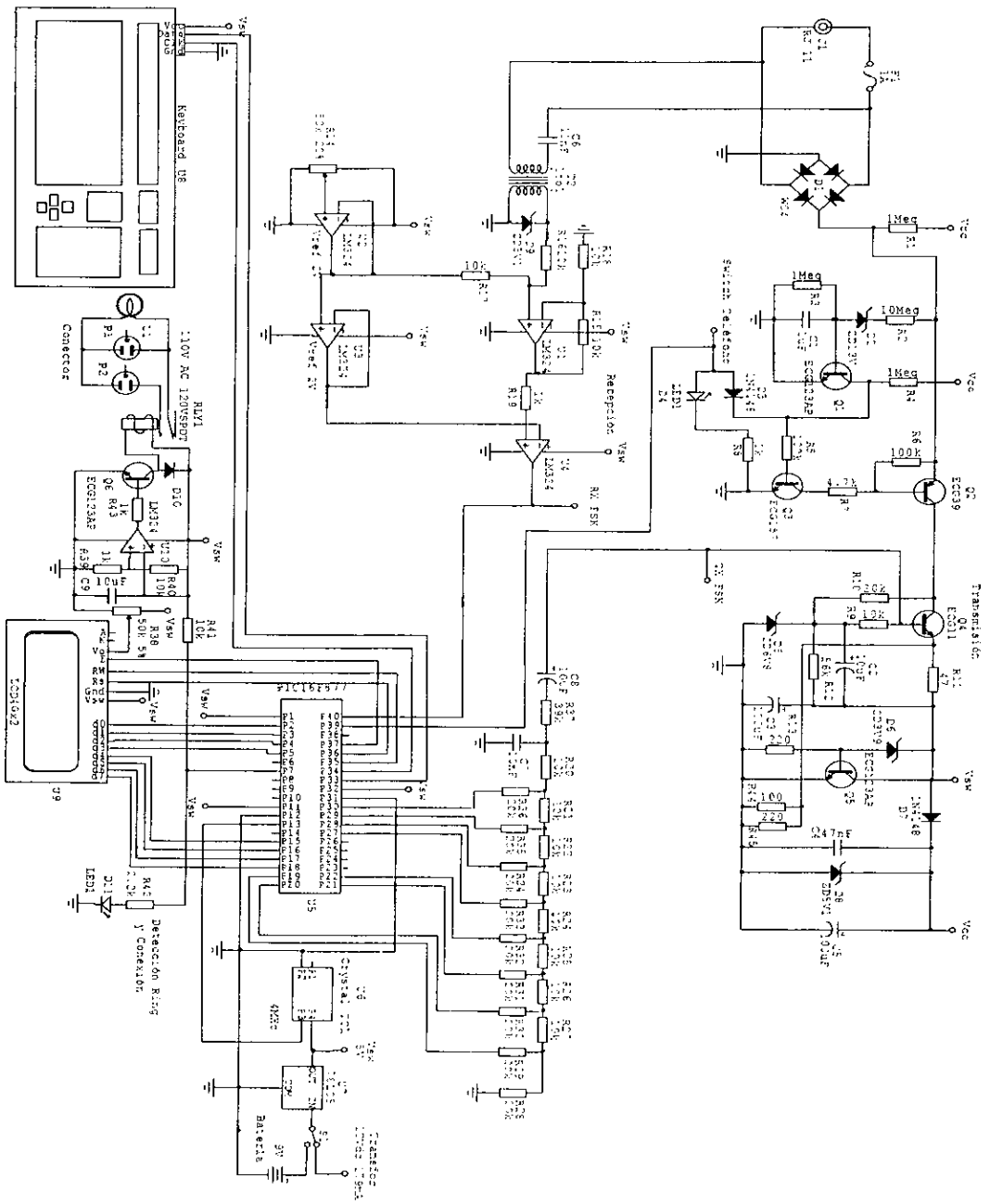
Figura 10. Diagrama de flujo del saludo o *handshake*



E. Circuito del teléfono de texto

El circuito completo del teléfono de texto se muestra a continuación. Este circuito incluye al circuito transmisor, al circuito receptor, al convertidor digital-analógico y la conexión del PIC16F877 con el resto de los circuitos.

Figura 11. Circuito del teléfono de texto



F. Lógica del programa

A continuación se muestran los diferentes bloques que conforman el programa realizado y compilado en MPLAB 6.3. Además, el código completo del programa se encuentra en el apéndice A.

Figura 12. Diagrama de flujo de inicialización del programa y de menú de entrada

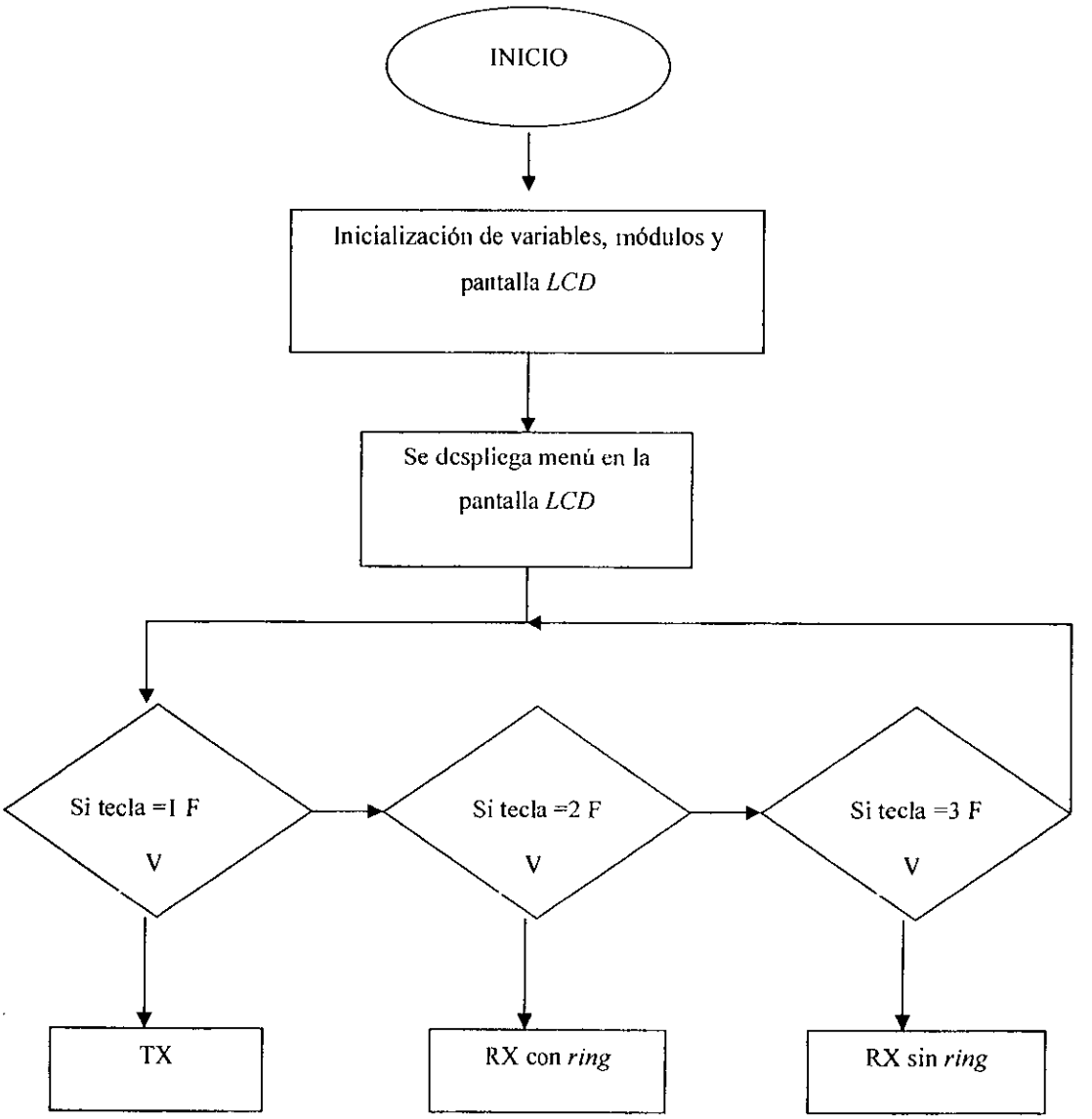


Figura 13. Diagrama de flujo de la transmisión

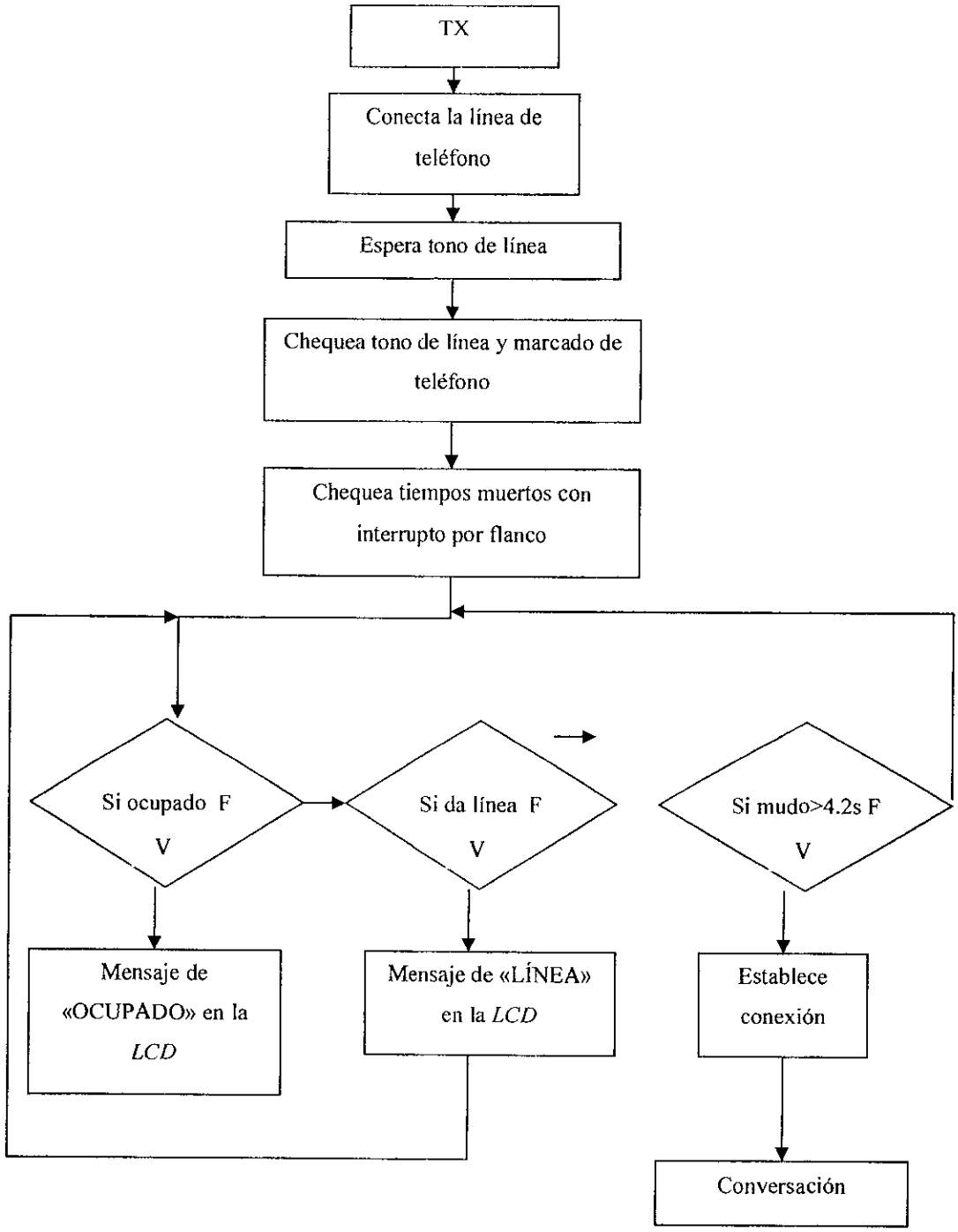


Figura 14. Diagrama de flujo de la recepción con detección de *ring*

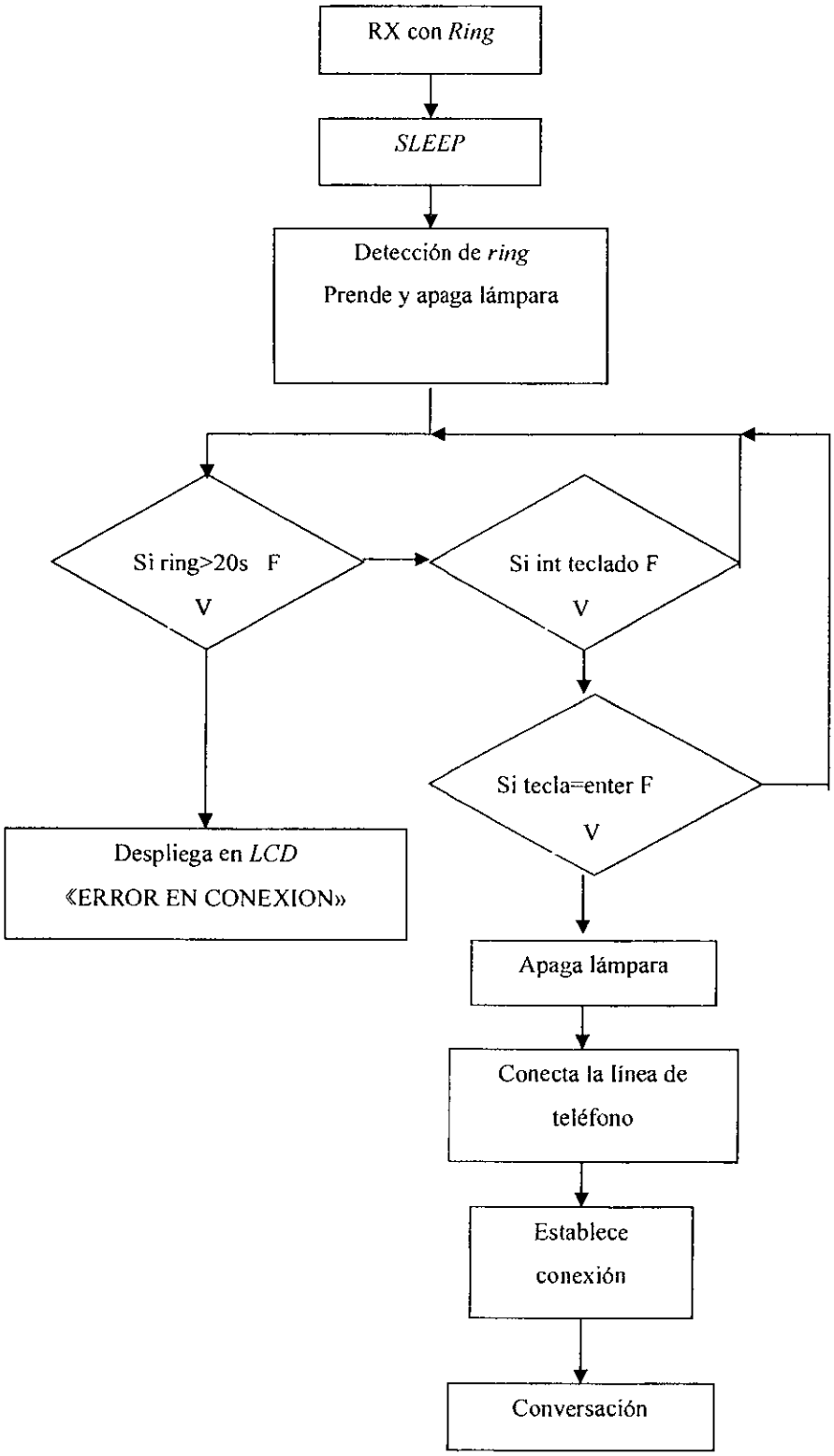


Figura 15. Diagrama de flujo de la recepción sin detección de *ring*

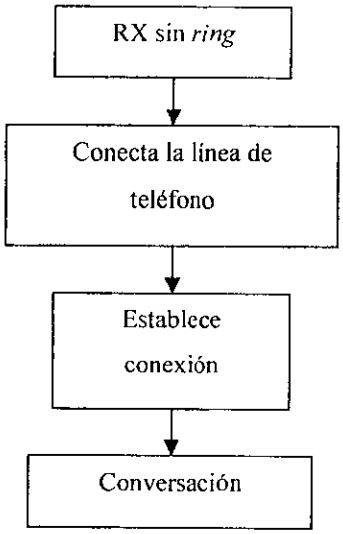


Figura 16. Diagrama de flujo de la conversación

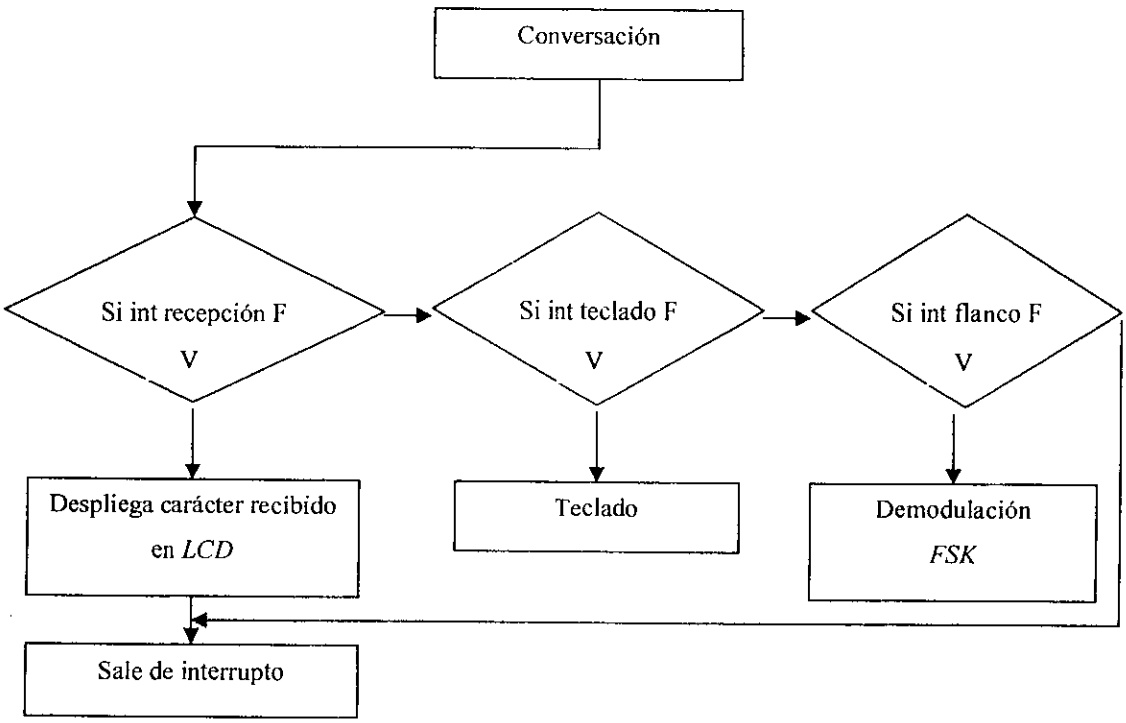


Figura 17. Diagrama de flujo del teclado

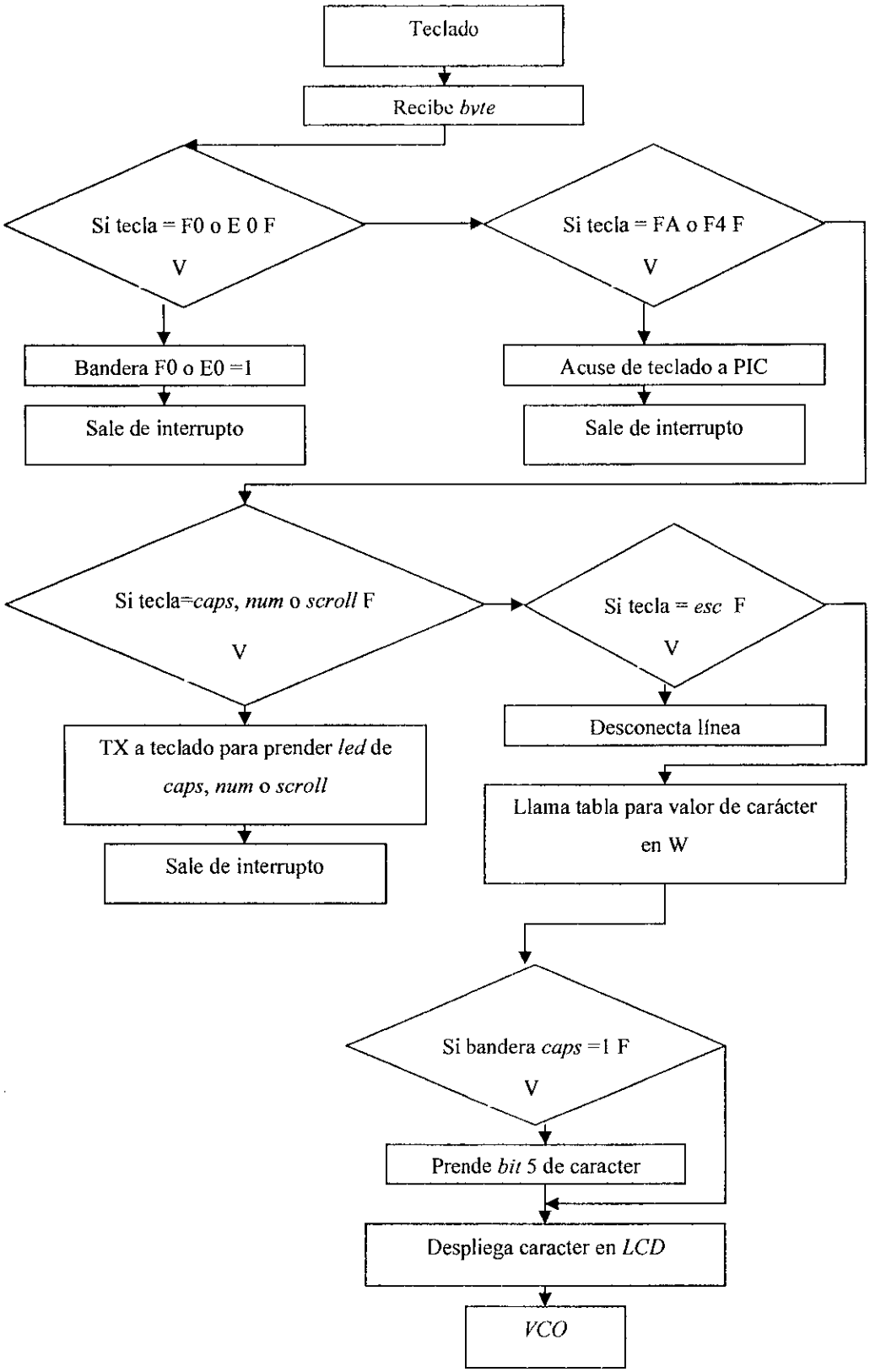


Figura 18. Diagrama de flujo del *VCO*

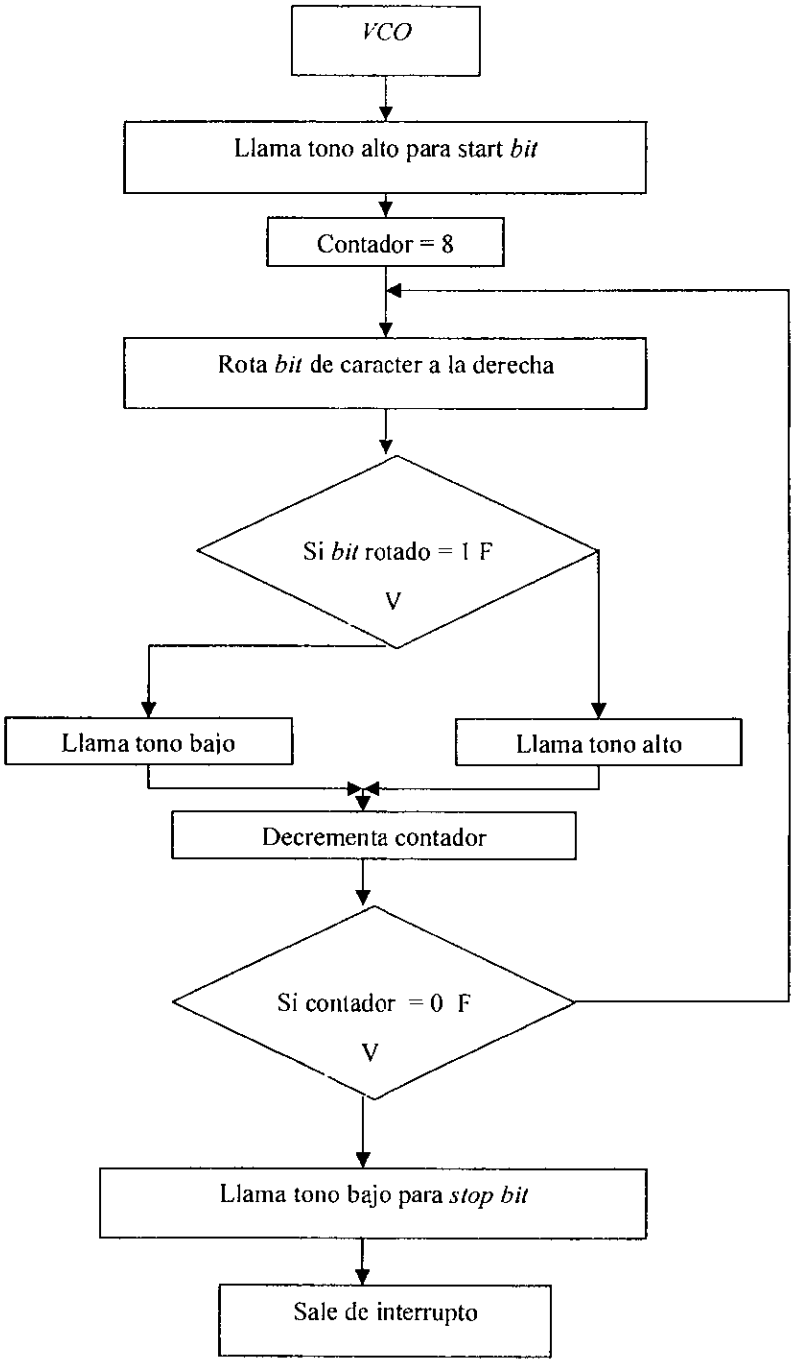


Figura 19. Diagrama de flujo de la onda senoidal

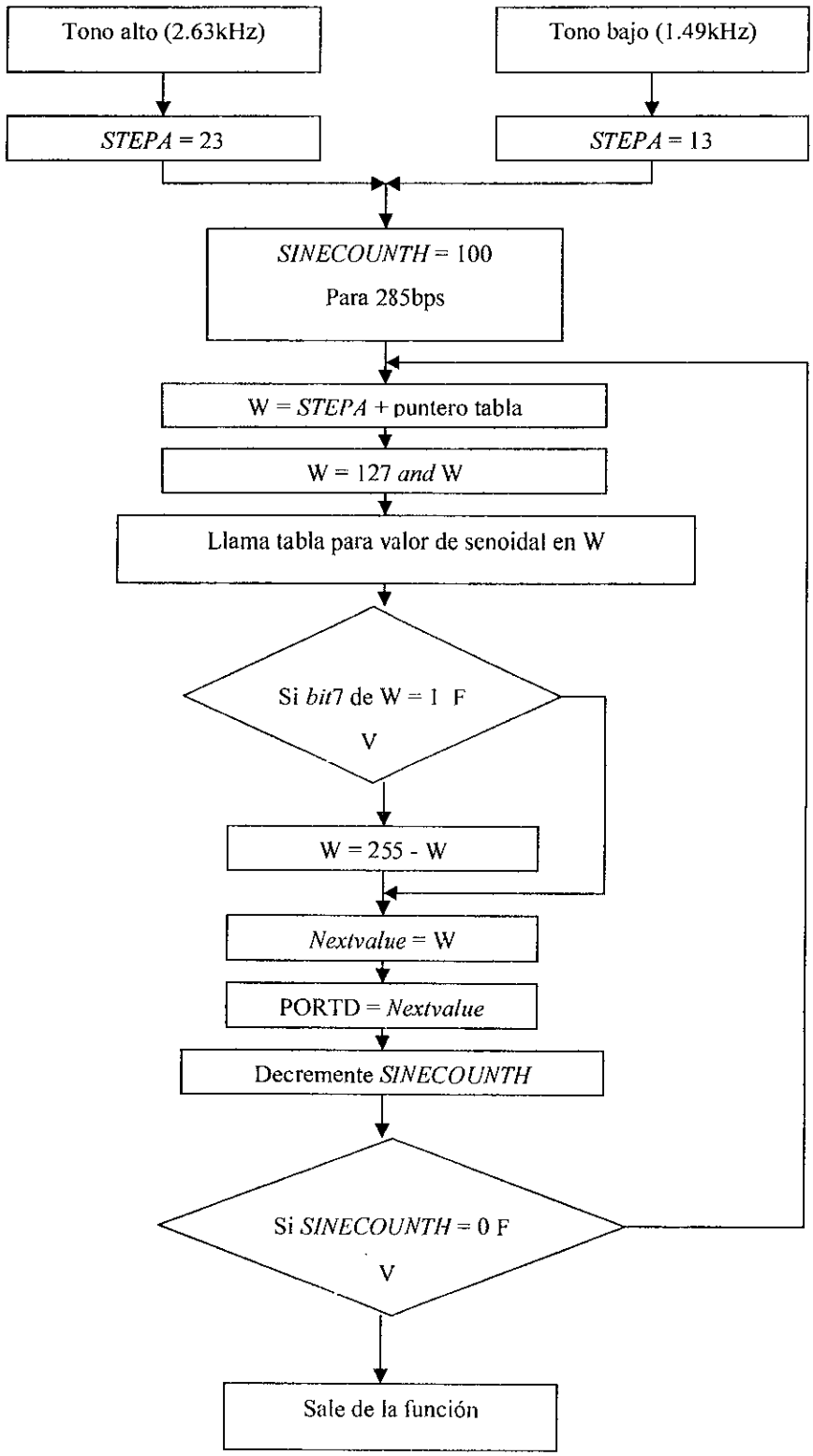
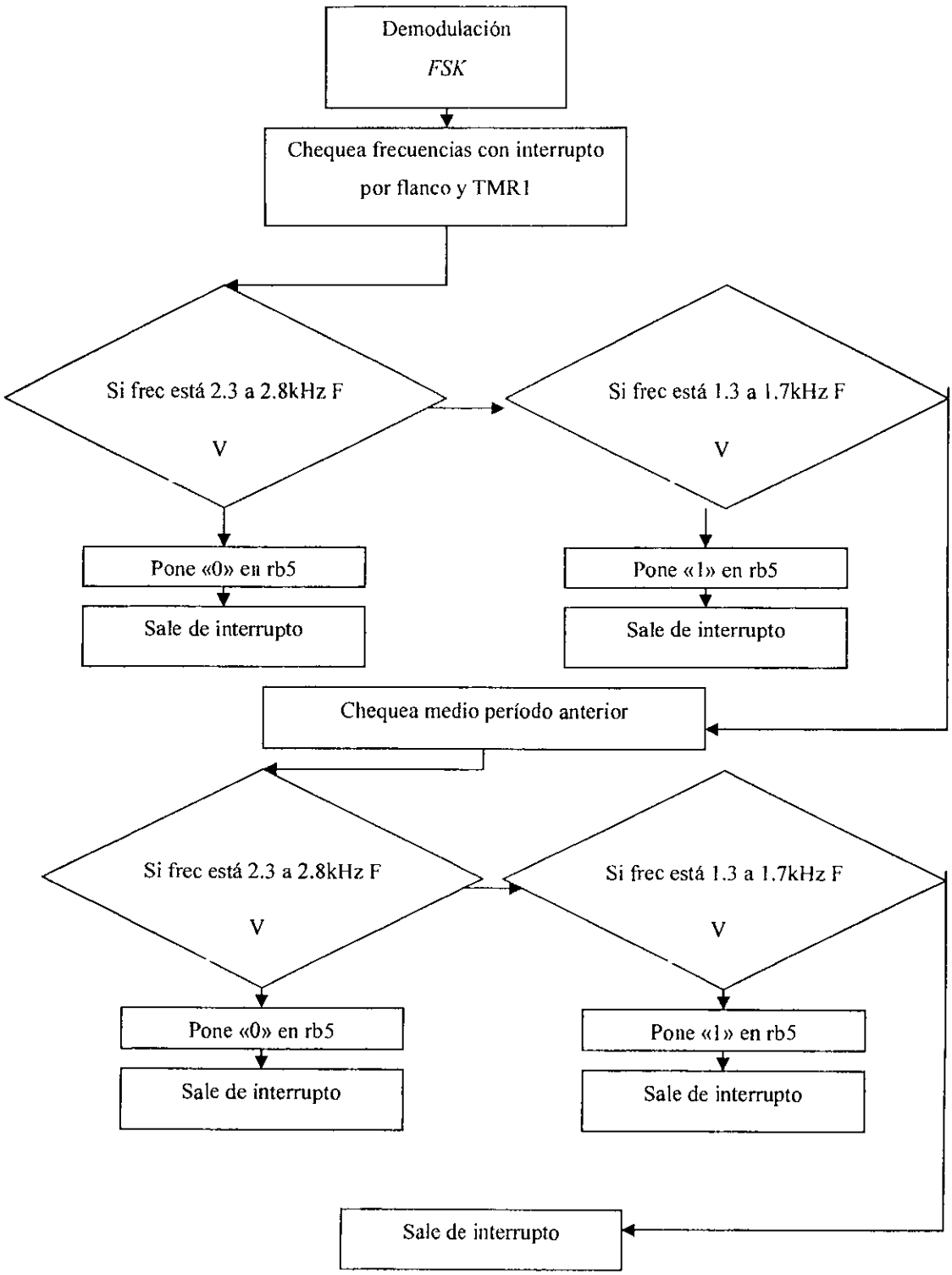


Figura 20. Diagrama de flujo de la demodulación *FSK*



G. Modo de implementación

Para la implementación del teléfono de texto, el usuario tendrá que escoger ciertas opciones antes de poder lograr la comunicación telefónica. Antes de encender el TTY, el usuario tendrá que escoger el tipo de alimentación del dispositivo.

1. Alimentación. Si desea utilizar una batería cuadrada de 9V, deberá mover el *switch* de encendido a la derecha. Si desea conectarlo a la red eléctrica (110V), deberá mover el *switch* de encendido a la izquierda y conectar el TTY al transformador de 110V/12V y así conectar el transformador al tomacorriente de 110V. Al haber realizado lo anterior, el teléfono de texto se encuentra encendido y se procede a escoger la opción deseada de transmisión o recepción.

2. Menú. Al momento de encender el TTY, aparecerá en la pantalla *LCD* un menú (*vid* figura 21). Este menú le indica que presione 1 si se va a realizar una transmisión (se va a realizar una llamada), que presione 2 si va a realizar una recepción con detección de *ring* o que presione 3 si se va a realizar una recepción sin *ring* (se va a contestar una llamada).

Figura 21. Menú que despliega la pantalla *LCD*

3. Transmisión. Si se desea realizar una transmisión, de primero se escoge en el menú de la pantalla del teléfono de texto la primera opción. En este momento se prende un led indicando que se conectó la línea telefónica «*POTS*». Después, se levanta el teléfono y se marca el número de teléfono. Al terminar de marcar se cuelga el teléfono. En este momento, el TTY procede a chequear si da línea o está ocupado. Si se da línea, entonces se despliega en la pantalla «LINEA» (*vid* figura 22), mientras si está ocupado, se despliega «OCUPADO» (*vid* figura 23).

Figura 22. Mensaje de «LÍNEA» en la pantalla *LCD*

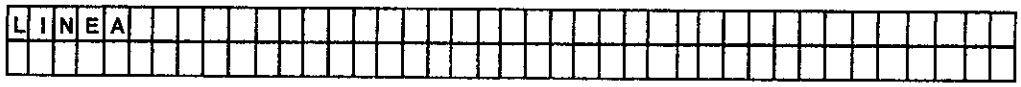
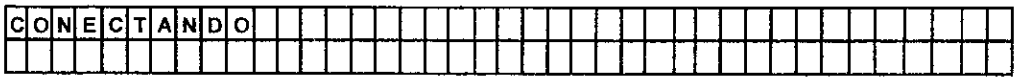


Figura 23. Mensaje de «OCUPADO» en la pantalla *LCD*

En el momento que se contesta la llamada, el TTY despliega un mensaje en la pantalla «CONECTANDO» (*vid* figura 24). Mientras aparezca este mensaje en la pantalla, el TTY transmisor está intentando establecer conexión con un TTY receptor.

Figura 24. Mensaje de «CONECTANDO» en la pantalla *LCD*



Si no se establece la conexión después de 20 segundos, el TTY transmisor despliega en la pantalla el mensaje «ERROR EN CONEXIÓN» (*vid* figura 25). En este momento se procede a cortar la llamada y a volver a intentar la transmisión si se deseara, al presionar el botón de *reset*, o el de *ESC* en el teclado. Después se procede a repetir la operación descrita anteriormente.

Figura 25. Mensaje de «ERROR EN CONEXIÓN» en la pantalla *LCD*

En el momento en que se establece conexión, se prende un led en el TTY transmisor y se pone la pantalla en blanco. En este momento, el usuario puede empezar la conversación.

Cuando se desee terminar la conversación y colgar el teléfono, se presiona en el teclado la tecla *ESC* y se cuelga la línea. Entonces se apaga el teléfono o si se desea hacer una transmisión o recepción, se presiona el botón de reset, o el de ESC nuevamente y se repite la operación descrita anteriormente.

4. Recepción. Existen dos formas de lograr una recepción en el TTY. Cuando éste se enciende, se muestra un menú en la pantalla (*vid* figura 21). En el menú se muestran dos opciones para realizar una recepción. Si se desea dejar el TTY en modo de recepción con detección de *ring*, se presiona 1. Si se

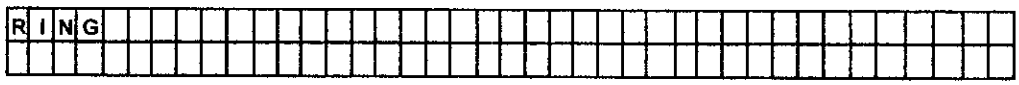
desea hacer una recepción sin detección de ring, se presiona 2. A continuación se describen más a fondo estas opciones.

a. Recepción con detección de *ring*. Antes de haber escogido esta opción, se conecta una lámpara para la detección del *ring* en la espiga del teléfono de texto si éste se encuentra conectado al tomacorriente. Al haber escogido esta opción, el teléfono de texto entra en *SLEEP*, estado en donde el consumo de potencia es mínimo. Además, se despliega en la pantalla *LCD* el mensaje «ESPERANDO LLAMADA» (*vid* figura 26).

Figura 26. Mensaje de «ESPERANDO LLAMADA» en la pantalla *LCD*

El TTY receptor detectará una llamada entrante al detectar el *ring*. En ese momento, la lámpara empezará a prenderse y apagarse, indicando que entra una llamada. Si estuviese alimentado por una batería, el TTY receptor detecta el *ring* por medio de un *led* que parpadea. Además, el TTY desplegará en la pantalla «*RING*» a la vez que la lámpara y el *led* parpadean (*vid* figura 27). Para contestar la llamada, se presiona la tecla de *ENTER* en el teclado durante el lapso que el *led* y la lámpara se encuentren apagados. En este momento se prende un *led* indicando que se conectó la línea telefónica «*POTS*». Además, empieza a establecer la conexión con el TTY transmisor y despliega «CONECTANDO» en la pantalla (*vid* figura 24). Cuando establece la conexión, se borra la pantalla, el receptor prende un led, el cual indica que ya se estableció la conexión y se puede empezar la comunicación.

Figura 27. Mensaje de «*RING*» en la pantalla *LCD*



Si por algún motivo ya no entra un *ring*, el TTY desplegará el mensaje en la *LCD* «ERROR EN CONEXIÓN» (ver figura #25). Si se desea volver a intentar la recepción, se presiona el botón de *reset*, o el de *ESC* en el teclado y se procede a repetir la operación descrita anteriormente.

b. Recepción sin detección de *ring*. Esta opción se escoge cuando se recibe una llamada TTY y el receptor no se encuentra conectado antes de recibirla. Es usada principalmente por usuarios que son oyentes. Cuando ya se ha contestado la llamada, el TTY receptor tiene 20 segundos para encenderse y ponerse en modo de recepción sin detección de *ring*. Ya escogida esta opción, el TTY receptor empieza a establecer la conexión con el TTY transmisor y despliega en la pantalla «CONECTANDO» (*vid* figura 24). A la vez, prende un led indicando que se conectó la línea telefónica «*POTS*». Cuando se logra la conexión, se borra la pantalla y el receptor prende un *led*, el cual indica que ya se estableció la conexión y se puede empezar la comunicación. Si el *led* no se enciende, la conexión no se logró establecer, desplegando en la pantalla «ERROR EN CONEXIÓN» (*vid* figura 25) y desconectando la línea de teléfono «*POTS*». Entonces hay que presionar el botón de *reset* si se desea volver a probar o se apaga el teléfono de texto.

En el momento que ya se ha establecido la conexión del TTY transmisor con el receptor, el primero que debe de empezar la comunicación es el TTY transmisor. Se debe procurar avisar cuando termina de escribir uno con alguna clave o frase. Se puede utilizar la tecla de ENTER que muestra «π» en la pantalla. Otra opción es la frase comúnmente utilizada en esta clase de teléfono, «*GA*» (*go ahead*), siga adelante. Una de estas dos opciones se utiliza al final de cada vez que alguno termine de escribir y espere una respuesta para indicar que ha terminado de escribir.

Además, si por alguna razón existiera ruido en la línea telefónica «*POTS*» y el TTY no funcionara correctamente, se procede a repetir la transmisión hasta que se logre la transmisión sin ningún error.

H. Implementación por personas no oyentes

Dos personas no oyentes probaron por el lapso de una semana el funcionamiento de las dos estaciones TTY. Estas personas conectaron cada estación TTY en la línea telefónica «*POTS*» de sus casas y se comunicaron sin la ayuda de personas oyentes por medio de los dos teléfonos de texto.

I. Costos

En la tabla 5 se presenta el costo en quetzales de cada parte que compone una estación TTY. También se muestra el costo total en quetzales y en dólares. No incluye lámpara.

Tabla 5. Costos de un teléfono de texto

Cantidad	Descripción	unidad (Q)	subtotal (Q)
1	PIC16F877	Q77.25	Q77.25
1	Pantalla LCD 40x2	Q244.12	Q244.12
1	Teclado Genius PS2 KB-06XE	Q56.03	Q56.03
1	Cristal TTL 4 MHz	Q16.20	Q16.20
1	Conector hembra RJ11	Q3.66	Q3.66
1	Conector hembra PS2 MiniDin 6 pines	Q6.10	Q6.10
1	Conector hembra para transformador	Q3.99	Q3.99
1	Batería Alkalina Cuadrada 9V	Q14.50	Q14.50
1	Transformador 110V/12V	Q37.36	Q37.36
1	Transformador de audio 1:1, 600Ω/600Ω	Q15.87	Q15.87
1	Regulador 5V 7805	Q2.36	Q2.36
1	OpAmp HA17324	Q4.50	Q4.50
1	OpAmp HA17358	Q3.50	Q3.50
1	Relé 120V	Q20.00	Q20.00
2	Tomacorrientes hembra	Q1.50	Q3.00
2	Espiga macho	Q1.50	Q3.00
1	Conector para batería 9V	Q6.43	Q6.43
2	Potenciómetro de 50k 1/4W	Q1.50	Q3.00
1	Resistencia 47Ω 1/2W	Q0.23	Q0.23
36	Resistencias variadas 1/4W	Q0.23	Q8.28
1	Capacitor cerámico 10nF 250V	Q3.36	Q3.36
2	Capacitores cerámicos varios 50V	Q0.56	Q1.12
4	Capacitores electrolíticos variados 25V	Q0.56	Q2.24
3	Transistor NPN ECG123AP	Q1.12	Q3.36
1	Transistor NPN ECG157 Vce=300V	Q4.48	Q4.48
1	Transistor PNP ECG39 Vce=300V	Q6.72	Q6.72
1	Transistor NPN ECG11 Ic=2A	Q2.24	Q2.24
1	Puente de diodos W04 300V 1A	Q5.60	Q5.60
2	Diodo Si 1N4148	Q0.84	Q1.68
4	Diodos Zener 1W variados	Q1.68	Q6.72
2	Led 1.4V	Q1.00	Q2.00
1	Fusible 1A	Q0.67	Q0.67
1	Portafusible	Q2.24	Q2.24
2	Metro de alambre calibre 24	Q1.25	Q2.50
1	Caja de aluminio	Q100.00	Q100.00
Total en quetzales			Q674.31
Total en dólares			$82.74

V. RESULTADOS

A. Generales

Se desarrolló e implementó un teléfono de texto para líneas «*POTS*». Con este teléfono de texto realizado se logró la comunicación telefónica inasistida de dos personas no oyentes durante el período de una semana (*vid* apéndice C).

Como se observa en la tabla 5, el costo del teléfono de texto creado es de $82.74. Éste es mucho menor a los ya existentes en el mercado estadounidense. Pues, el TTY más barato, cuesta $250.00 (de *ULTRATEC*) y su pantalla es de 24 caracteres con una velocidad de 45 bps. Además, el teléfono de texto diseñado en este trabajo de graduación, tiene muchas funciones adicionales que los teléfonos de textos existentes no poseen. Por lo que se demuestra que se logró un costo menor a los teléfonos de textos existentes.

B. Específicos

1. Se logró una velocidad mayor a los 100 *bits* por segundo. La velocidad de transmisión del teléfono de texto desarrollado en este trabajo de graduación es de 285 *bits* por segundo.

2. La comunicación entre personas no oyentes se realiza a 1710 caracteres por minuto. Con esto, se logra una comunicación con una velocidad de 342 palabras por minuto. Esto se puede observar en la tabla 6.

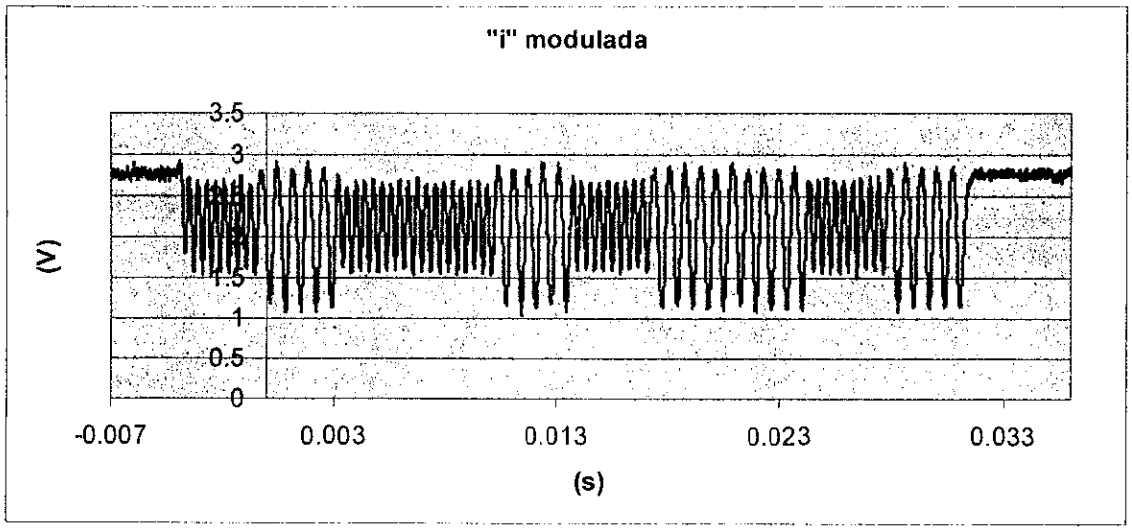
Tabla 6. Velocidades de transmisión

Bits por segundo	285
Caracteres por segundo	28.5
Caracteres por minuto	1710
Palabras por minuto	342

3. Se logró una transmisión serial asíncrona *half duplex* a 285 bps con 8 *bits* de datos en código ASCII, 1 *start bit* y 1 *stop bit*. Ésta se realizó por la línea de teléfono «*POTS*» mediante la utilización de *FSK* para la transmisión de datos.

La modulación de la señal se realizó por *software* en el PIC16F877. De éste sale la señal por un puerto de 8 *bits*, de donde pasa por un convertidor digital-analógico de tipo escalera R-2R, en donde ya se encuentra la señal modulada. Un «1» equivale a una frecuencia de 1.49kHz y un «0» equivale a una frecuencia de 2.63kHz. A continuación, se muestra la señal modulada de la letra «i» en código ASCII.

Figura 28. Señal modulada



La demodulación de la señal transmitida se realizó por *software* en el microcontrolador PIC16F877. Para lograr esta demodulación, de primero la señal transmitida se convierte en una onda cuadrada. Para esto se utilizó el circuito receptor que se muestra en la figura 9. En la figura 29 se muestra la señal de onda cuadrada que sale del circuito receptor, la cual equivale a la letra «i» en código ASCII. Esta señal es la que demodula el microcontrolador. Éste detecta las dos frecuencias y saca la señal demodulada por el puerto rb5. Esta señal se observa en la figura 30. La señal demodulada entra al receptor del puerto serial en donde se finaliza la demodulación *FSK*.

Figura 29. Onda cuadrada que sale del circuito de recepción

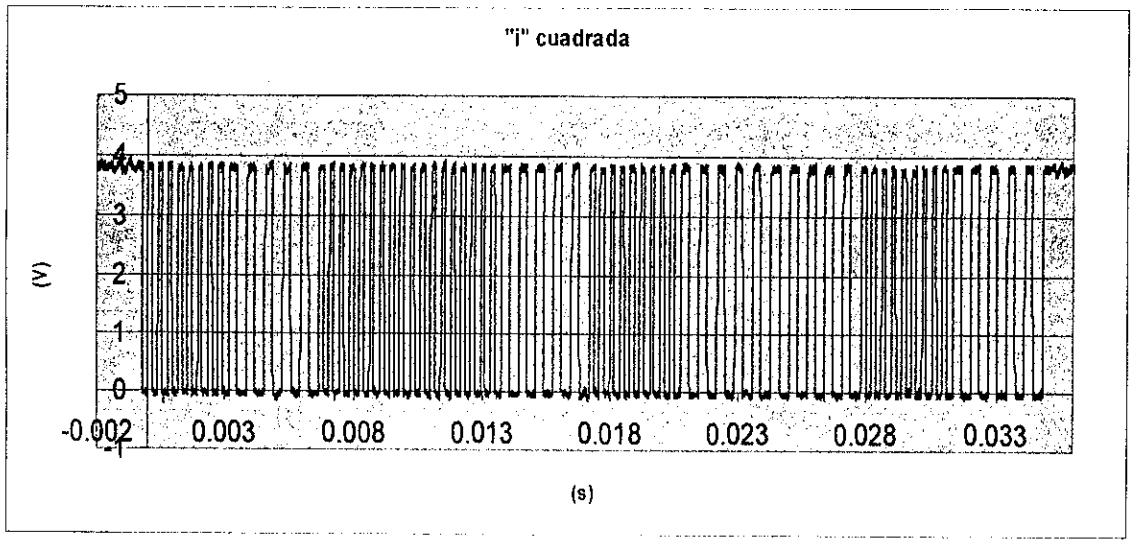
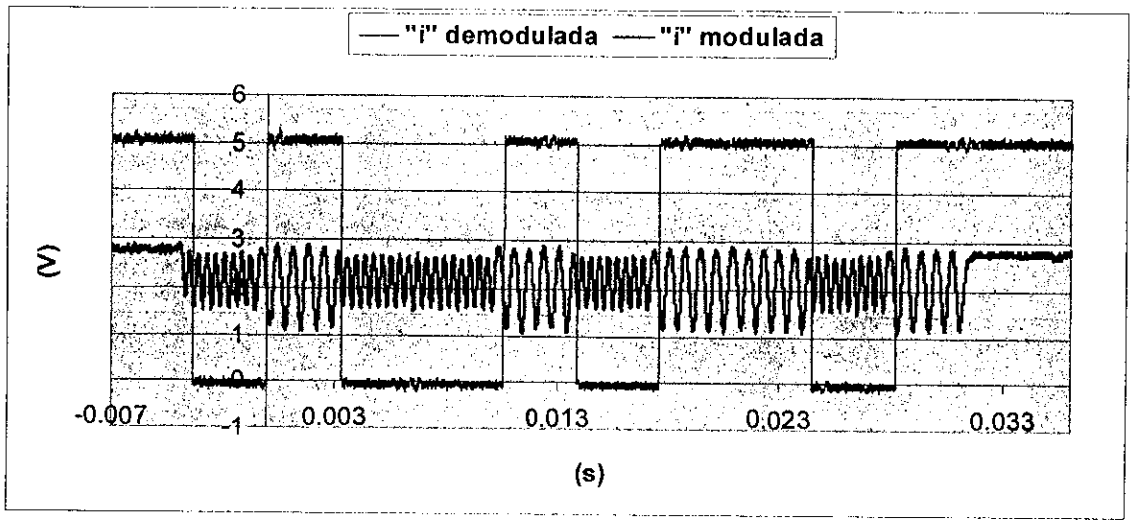


Figura 30. Señal modulada y señal demodulada que entra al puerto serial



4. La implementación del teléfono de texto en líneas de teléfono «*POTS*» se logró mediante la utilización del circuito de transmisión mostrado en la figura 8 y el circuito de recepción mostrado en la figura 9.

5. El teléfono de texto es capaz de reconocer si el teléfono al que desea comunicarse se encuentra ocupado o no, por medio de un mensaje que se despliega en la pantalla *LCD*. Además, logra reconocer el momento en el que se establece conexión con otro TTY mediante un *led* que se enciende en dicho momento.

6. El teléfono de texto es capaz de detectar una llamada entrante. Esto lo realiza al prender y apagar de forma intermitente un *led* y una lámpara hasta que se conteste la llamada en el teléfono de texto.

VI. OBSERVACIONES

Al inicio de este proyecto se había pensado utilizar tanto un modulador como un demodulador *FSK* por *hardware* (externo del PIC16F877).

La señal con los datos saldría del PIC16F877 en forma digital y habría un modulador *FSK*, en este caso un *VCO* (*voltage controller oscillator*) y ya se introduciría la señal en forma analógica a la línea telefónica «*POTS*». Como modulador *FSK*, se utilizó el «*Voltage-Controlled Oscillator*» XR2207 en la configuración mostrada en la página 18 figura 20 «*Multi-channel FSK Generation*» del manual «*XR-2207 Voltage-Controlled Oscillator*» y no se logró la modulación deseada. Este resultado se pudo deber al uso aproximado de valores de capacitancia y resistencia.

La demodulación *FSK* de la señal proveniente de la línea telefónica «*POTS*» la realizaría un demodulador *FSK* y de ahí este demodulador pasaría la señal digital al PIC16F877. Como demodulador *FSK*, se utilizó el «*FSK Demodulator*» XR2211 en la configuración mostrada en la figura 3 «*Generalized Circuit Connection for FSK and Tone Detection*» del manual «*XR-2211 FSK Demodulator/Tone Decoder*» y no se logró la demodulación debido a la inestabilidad que surgía de la señal demodulada debido al constante desajuste del potenciómetro usado en R0. Se trató de utilizar resistencias que daban el valor ajustado por el potenciómetro pero no se logró encontrar una combinación exitosa de resistencias en la cual el demodulador permaneciera estable por un período mayor de 2 horas. Además, se utilizó el demodulador «*Phase Locked Loop*» LM565 en la configuración mostrada en el primer circuito de la página 6 del manual «*LM565/LM565C Phase Locked Loop*» y no se logró la demodulación debido a que no se logró el correcto funcionamiento de dicho circuito utilizando las fórmulas indicadas en este manual. Este problema se pudo deber al uso aproximado de valores de capacitancia y de resistencia.

Debido a la problemática encontrada al utilizar estos circuitos integrados se procedió a realizar la modulación y demodulación *FSK* por *software* en el PIC16F877. Esta implementación disminuye tanto la complejidad de la circuitería utilizada, como los costos de la estación TTY.

VII. CRONOGRAMA

Actividad / Semana	1	2	3	4	5	6	7	8	9	10
Planificación y diseño del teléfono de texto										
Interfase de transmisión y recepción										
Transmisión y recepción de la señal										
Construcción final de las dos estaciones TTY										
Pruebas preliminares de las dos estaciones TTY										
Pruebas finales de las dos estaciones TTY por personas no oyentes										

FUENTES DE INFORMACIÓN

Chapweske, Adam. *PS/2 Mouse/Keyboard Protocol* [en línea]. 1999. <http://panda.cs.ndsu.nodak.edu/~achapwes/PICmicro/PS2/ps2.htm> [Consulta: 20-mayo-2003].

Harkins, Judy y N. Williams. *TTY Basics* [en línea]. *Gallaudet University,* 1998. . <http://tap.gallaudet.edu/TTY-Basics.htm> [Consulta: 17-junio-2003].

Holmes, Pamela. *Comments* [en línea]. Wisconsin: Ultratec, 1998. <http://www.fcc.gov/Bureaus/Common_Carrier/Comments/trs/211252-2.pdf> [Consulta: 9-junio-2003].

LM565/LM565C Phase Locked Loop. 1999. National Semiconductor Corporation. 11 págs.

Predko, Myke. 2002. *Programming and customizing PICmicro MCU Microcontrollers. New York*, McGraw-Hill. 1192 págs.

Taub, Herbert y D. Schilling. 1986. *Principles of communication systems.* 2ª ed. *New York*, McGraw-Hill Book Company. 758 págs.

XR-2207 Voltaje-Controlled Oscillator. 1997. EXAR Corporation. California. 24 págs.

XR-2211 FSK Demodulator/Tone Decoder. 1997. EXAR Corporation. California. 24 págs.

APÉNDICES

A. Código fuente del teléfono de texto

```
        list      p=16f877
        #include "p16f877.inc"


__CONFIG _HS_OSC & _CP_OFF & _WDT_OFF & _PWRTE_ON & _LVP_OFF & _BODEN_ON

errorlevel -302          ; elimina mensaje "register not in bank 0"

;*****************************************************************************
*******************
;               Oscilador HS  de 4 Mhz
;               Code Protection Off
;               Watchdog Timer          Off
;               Power up Timer Off
;               Low Voltaje Programming Off
;*****************************************************************************
*******************


;*****************************************************************************
*******************
;               VARIABLES BANCO 0
;*****************************************************************************
*******************

 cblock 0x20
        KEYBD                   ;guarda byte de teclado
        TEMP                ;temporal para escribir a LCD
```

```
        CONT1           ;variable de retrasos
        CONT2           ;variable de retrasos
        CONT3           ;variable de retrasos
        CONT4           ;variable de retrasos
        CONT5           ;variable de retrasos
        CONT6           ;variable de retrasos
        CONT_FA         ;contador para leds de teclado
        CONT_DIS        ;contador para LCD
        CONT_DIS2       ;contador temporal para LCD
        CONT_TEMP       ;contador para esperar tonos dtmf
        FLAGS           ;contiene banderas del teclado
        W_SAVE          ;temporal de W
        STAT_SAVE       ;temporal de STATUS
        PCLH_SAVE       ;temporal de PCLATH
        LEDS            ;guarda banderas de leds de teclado
        TEMP1           ;temporal para LCD
        CONTROL         ;banderas de control de línea de teléfono
        CONTROL2        ;banderas de control de línea y de llamada
        DELAY           ;variable de retrasos

    endc


;*******************************************************************************
*******************
;               VARIABLES COMPARTIDAS
;*******************************************************************************
*******************

    cblock  0x70
        CON1            ;contador para leer memoria de programa
        POINTERA        ;contador del valor de la onda senoidal
```

```
        SINECOUNTH      ;contador que da la duración de la onda
        STEPA           ;obtiene tamaño de salto para cierta frec.
        ALLONES         ;obtiene la parte alta o baja de la onda
        NEXTVALUE       ;temporal para valor de onda senoidal
        T0_count        ;mira medio periodo
        Tlast           ;mira medio periodo anterior
        RX              ;guarda byte recibido
        WORD            ;fin de puntero de memo. programa
        DATH            ;guarda 2 byte de memo. programa
        PA1             ;puntero para menús de pantalla

    endc

;*****************************************************************************************
*********************
;               Definición de las variables de bits (banderas)
;*****************************************************************************************
*********************

        #DEFINE     bien_r          FLAGS, 0
        #DEFINE     parity          FLAGS, 1
        #DEFINE     rx_err          FLAGS, 2
        #DEFINE     tx_err          FLAGS, 3
        #DEFINE     dat_err         FLAGS, 4
        #DEFINE     F0_flag         FLAGS, 5
        #DEFINE     E0_flag         FLAGS, 6


        #DEFINE     inicio          CONTROL, 0
        #DEFINE     linea           CONTROL, 1
        #DEFINE     da_lineaCONTROL, 2
        #DEFINE     ocupadoCONTROL, 3
```

```
#DEFINE     offhook     CONTROL, 4
#DEFINE     ocupado2    CONTROL, 5
#DEFINE     linetemp    CONTROL, 6
#DEFINE     ring        CONTROL, 7


#DEFINE     pico        CONTROL2, 0
#DEFINE     picosleep   CONTROL2, 1
#DEFINE     lineaerror  CONTROL2, 2
#DEFINE     lineauno    CONTROL2, 3
#DEFINE     rx_tty      CONTROL2, 4
#DEFINE     tx_tty      CONTROL2, 5
#DEFINE     tonoinicio  CONTROL2, 6
#DEFINE     fin         CONTROL2, 7

;*********************************************************************************
*******************

;*********************************************************************************
*******************

BANCO0 MACRO
        BCF STATUS, RP0
        BCF STATUS, RP1
         ENDM

BANCO1 MACRO
        BCF STATUS, RP1
        BSF STATUS, RP0
         ENDM

BANCO2 MACRO
```

```
        BSF STATUS, RP1
        BCF STATUS, RP0
        ENDM

BANCO3 MACRO
        BSF STATUS, RP1
        BSF STATUS, RP0
        ENDM

;*********************************************************************************
*******************

;*********************************************************************************
*******************
        org     0x00
        goto    INICIO
        org     0x04
        goto    INTERR
        org     0X06

;*********************************************************************************
*******************
;               Tabla de media onda senoidal. Son las 128 muestras positivas.
;*********************************************************************************
*******************

SINELOOKUP

    addwf       PCL,F           ;suma W a PC

sinstart
    retlw       .127
    retlw       .130
```

```
retlw   .133
retlw   .136
retlw   .140
retlw   .143
retlw   .146
retlw   .149
retlw   .152
retlw   .155
retlw   .158
retlw   .161
retlw   .164
retlw   .167
retlw   .170
retlw   .173
retlw   .176
retlw   .179
retlw   .182
retlw   .185
retlw   .187
retlw   .190
retlw   .193
retlw   .195
retlw   .198
retlw   .201
retlw   .203
retlw   .206
retlw   .208
retlw   .211
retlw   .213
retlw   .215
retlw   .218
retlw   .220
retlw   .222
```

```
retlw   .224
retlw   .226
retlw   .228
retlw   .230
retlw   .232
retlw   .233
retlw   .235
retlw   .237
retlw   .238
retlw   .240
retlw   .241
retlw   .243
retlw   .244
retlw   .245
retlw   .246
retlw   .248
retlw   .249
retlw   .249
retlw   .250
retlw   .251
retlw   .252
retlw   .253
retlw   .253
retlw   .254
retlw   .254
retlw   .254
retlw   .255
retlw   .255
retlw   .255
retlw   .255
retlw   .255
retlw   .255
retlw   .255
```

```
retlw   .254
retlw   .254
retlw   .254
retlw   .253
retlw   .253
retlw   .252
retlw   .251
retlw   .250
retlw   .249
retlw   .249
retlw   .248
retlw   .246
retlw   .245
retlw   .244
retlw   .243
retlw   .241
retlw   .240
retlw   .238
retlw   .237
retlw   .235
retlw   .233
retlw   .232
retlw   .230
retlw   .228
retlw   .226
retlw   .224
retlw   .222
retlw   .220
retlw   .218
retlw   .215
retlw   .213
retlw   .211
retlw   .208
```

```
        retlw   .206
        retlw   .203
        retlw   .201
        retlw   .198
        retlw   .195
        retlw   .193
        retlw   .190
        retlw   .187
        retlw   .185
        retlw   .182
        retlw   .179
        retlw   .176
        retlw   .173
        retlw   .170
        retlw   .167
        retlw   .164
        retlw   .161
        retlw   .158
        retlw   .155
        retlw   .152
        retlw   .149
        retlw   .146
        retlw   .143
        retlw   .140
        retlw   .136
        retlw   .133
        retlw   .130

;*********************************************************************************
*******************
;               Rutina de Inicialización del Programa
```

```
;*****************************************************************************
*******************

INICIO

        clrf    LEDS                    ;borra banderas de leds
        clrf    CONT_DIS                ;borra contador LCD
        movlw   .1
        movwf   CONT_FA                 ;pone 1 en contador de led mayusculas
        bsf     STATUS, RP0             ;***banco 1***
        bsf     INTCON, INTE            ;hab. interrupto externo en RB0
        bsf     INTCON, RBIE            ;habilita rb7 interrupto


        bcf     OPTION_REG, INTEDG      ;interr. en flanco de caida de RB0
        bcf     OPTION_REG, T0CS        ;inicializando Timer 0, modo Timer
        bcf     OPTION_REG, PSA         ;prescaler asignado al Timer 0
        bsf     OPTION_REG, 7           ;pullups off

        bcf     OPTION_REG, PS1
        bsf     OPTION_REG, PS2
        bcf     OPTION_REG, PS0         ;Timer 0 prescaler 1:32

        movlw   b'00000110'
        movwf   ADCON1                  ;pone en digital porta y porte

        clrf    TRISA                   ;PONE COMO SALIDAS PORTA PARA LCD
                                        ;LSBITS Y RA5 PARA LAMPARA
        movlw   b'011'
        movwf   TRISE                   ;RE0 ES CONTROL DE FSK, RE2 ES LED DE
                                        ;CONEXION
        movlw   b'11110000'             ;pone como entrada a rc7 para RX,
        movwf   TRISC                   ;y salida rc0-rc3 para msbits para lcd
```

```
        clrf    TRISD           ;El puerto D como salida PARA SENOIDAL
        movlw   B'10000111'     ;RB3, RB4, RB5, RB6 como salida
        movwf   TRISB           ;RB7 como demod FSK,
                                ;RB0,RB1 PARA KBOARD

        clrf    PIE1            ;deshabilita interruptos de periféricos

        BANCO0
        clrf    PORTD           ;PONE EN CERO EL PORTD PARA SENOIDAL
        clrf    PORTE           ;PONE EN CERO EL PORTE PARA CONECCION
        bsf     PORTB, 5        ;PONE EN ALTO RB5 QUE ES SALIDA DEMOD
                                ;FSK Y ENTRA A RC7
        bcf     PORTB, 6        ;APAGA LINEA DE TELEFONO
        nop
;*******************************************************************************
********************
;               CONFIGURACION TMR2
;*******************************************************************************
********************

        CLRF    T2CON           ;para Timer2, Prescaler = 1:1,
                                ;Postscaler = 1:1
        CLRF    TMR2            ;borra Timer2

        clrf    PIR1            ;borra flag int. de periféricos
        movlw   B'00000001'     ;TMR2 PRESCALER 1:4

        movwf   T2CON           ;Timer2 apagado
        bsf     T2CON, TMR2ON   ;Timer2 empieza a incrementar
```

```
;*******************************************************************************
*********************
;       CONFIGURACION DEL SERIAL
;*******************************************************************************
*********************

        BANCO1

        bcf     TXSTA, BRGH             ;BAUD RATE low SPEED
        movlw   .218
        movwf   SPBRG                   ;PONER BAUD RATE PARA 285 BPS

        BSF     PIE1,RCIE               ;prende interrupto de recepción

        bcf     STATUS, RP0             ;BANCO 0

        bsf     RCSTA, CREN             ;RECIBE CONTINUO
        bsf     RCSTA, SPEN  ;YA SE PUEDE RECIBIR EN ASINCRONO

        bcf     INTCON, 0               ;CERO LA FLAG DE RB7
        bsf     INTCON, 6               ;PERIPHERAL INT ON


;*******************************************************************************
*********************
;                       INICIALIZA VARIABLES, BANDERAS Y LCD
;*******************************************************************************
*********************

        clrf    POINTERA
        movlw   .255
        movwf   ALLONES
        clrf    NEXTVALUE               ;inicializa variables de sendtone
```

```
        call    DELAY2          ;llama DELAY2
        call    DELAY2
        call    RutinaLCD       ;inicializa LCD
        nop
        clrf    CONTROL
        clrf    CONTROL2        ;borra banderas de control
        movlw   .7
        movwf   CONT_TEMP       ;inicializa contador de dtmf
        call    UNO
        bsf     inicio          ;prende flag de inicio
        bsf     pico            ;prende flag de picos en la linea
                                ;simulada
        bcf     INTCON, RBIE    ;deshabilita rb7 interrupto
        call    DELAY2
        bsf     INTCON, GIE     ;habilitando los interruptos
        call    DELAY2
        call    DELAY2

;*******************************************************************************
*******************
;                       ESPERA OPCION DE MENU
;*******************************************************************************
*******************

MAININICIO

        btfsc   inicio
        goto    MAININICIO      ;si no se ha escogido opción se
                                ;queda aquí
        btfss   linea           ;salta si es transmisión
        goto    MAIN            ;se escogió recepción

        bcf     INTCON, INTE    ;deshab. interrupto externo en RB0
```

```
        bsf     INTCON, RBIE        ;habilita rb7 interrupto
        bsf     tonoinicio          ;flag de espera de tono de inicio
        sleep                       ;espera detección de tono en la línea
;*****************************************************************************
*******************
;                       DETECTA TONO INICIAL DE LINEA TELEFONICA
;*****************************************************************************
*******************

MAININICIOLINEA

        bsf     linetemp            ;prende bandera de tono inicial

SEGTEMP

        movlw   D'2'
        movwf   CONT3
        movlw   D'5'
        movwf   CONT2
        movlw   D'188'
        movwf   CONT1               ;inicializa retraso

SEG1TEMP

        decfsz  CONT1,F
        goto    SEG1TEMP
        decfsz  CONT2,F
        goto    SEG1TEMP
        decfsz  CONT3,F             ;si pasa más de 0.2seg salta
        goto    SEG1TEMP

        bcf     linetemp            ;apaga bandera de tono inicial
        bsf     lineauno            ;prende bandera de espera de dtmf
```

```
;*****************************************************************************
*******************
;                       ESPERA QUE MARQUE EL NUMERO DE TELEFONO
;*****************************************************************************
*******************

MARCADO

        nop
        sleep
        nop
        btfsc   lineauno            ;si ya terminó de marcar salta
        goto    MARCADO             ;regresa pues no ha terminado
                                    ;de marcar
        bsf     lineaerror          ;prende bandera de primer vez ocupado
        bsf     da_linea        ;prende bandera de dar línea
        goto    MAININICIOTEMPLINEA ;va a rutina de chequeo de
                                    ;dar línea

;*****************************************************************************
*******************
;                       RUTINA QUE CHEQUEA SI ESTA OCUPADO
;*****************************************************************************
*******************

MAININICIO1

        bsf     ocupado         ;prende bandera de ocupado

MEDIOSEG

        movlw   D'3'
```

```
        movwf  CONT3
        movlw  D'205'
        movwf  CONT2
        movlw  D'66'
        movwf  CONT1            ;inicializa espera de 0.5seg

MEDIOSEG1

        bcf    PORTE, 2         ;parpadea led de conexión si ocupado
        decfsz CONT1,F
        goto   MEDIOSEG1
        decfsz CONT2,F
        goto   MEDIOSEG1
        decfsz CONT3,F
        goto   MEDIOSEG1        ;si pasa 0.5seg se va a chequear si
                                ;da línea
        bcf    ocupado2         ;borra bandera de ocupado2
        bcf    ocupado      ;borra bandera de ocupado
        bsf    da_linea     ;prende bandera de dar línea


;*****************************************************************************
**********************
;                       RUTINA QUE CHEQUEA SI DA LINEA
;*****************************************************************************
**********************

MAININICIOTEMPLINEA

        movlw  D'22'
        movwf  CONT6

        movlw  D'100'
```

```
        movwf  CONT5
        movlw  D'101'
        movwf  CONT4                ;inicializa espera de 4.2seg

MAININICIOTEMP                      ;si pasan 4.2seg empieza conexión

        bcf    PORTE, 2             ;parpadea led de conexión si da línea

        btfsc  ocupado2             ;si cstá ocupado se va a MAININICIO1
        goto   MAININICIO1

        decfsz CONT4,F
        goto   MAININICIOTEMP
        decfsz CONT5,F
        goto   MAININICIOTEMP
        decfsz CONT6,F
        goto   MAININICIOTEMP

        btfss  da_linea         ;regresa a main inicio
        goto   MAININICIO1          ;si esta mudo pero no dio linea

        btfsc  ocupado2
        goto   MAININICIO1

        bcf    linea                ;borra bandera de inicio
        bcf    da_linea
        bsf    INTCON, RBIE         ;habilita rb7 interrupto
        bcf    INTCON, INTE         ;deshab. interrupto externo en RB0
        bcf    PORTE, 2             ;apaga led
        goto   MAIN1
```

```
;*****************************************************************************
*******************
;                          RUTINA QUE ESPERA RECEPCION DE RING
;*****************************************************************************
*******************

MAIN

        bcf     INTCON, INTE        ;deshab. interrupto externo en RB0
        bcf     PORTE, 2            ;APAGA LAMPARA TEMPORAL
        bsf     INTCON, RBIE        ;habilita rb7 interrupto
        nop
        btfsc   ring                ;si es RX SIN RING no se va a sleep
        sleep
        nop
        btfsc   picosleep           ;chequea si hay picos en la línea
        goto    MAIN                ;si hay al principio regresa a MAIN
        btfss   ring                ;si RX CON RING salta
        goto    MAIN1.0             ;si RX SIN RING va a MAIN1.0
        bsf     INTCON, INTE        ;hab. interrupto externo en RB0

;*****************************************************************************
*******************
;                              RUTINA PARA DETECTAR RING
;*****************************************************************************
*******************

RINGER:

        movlw   D'22'
        movwf   CONT6
        movlw   D'100'
        movwf   CONT5
```

```
        movlw   D'101'
        movwf   CONT4           ;inicializa espera de 4.2seg
        bcf     PORTE, 2        ;apaga led y lámpara

RINGER1

        btfsc   picosleep       ;si picos regresa a MAIN
        goto    MAIN
        bcf     PORTE, 2        ;apaga led y lámpara
        btfss   ring            ;si ya contestó se va a MAIN1.0
        goto    MAIN1.0
        decfsz  CONT4,F
        goto    RINGER1
        decfsz  CONT5,F
        goto    RINGER1
        decfsz  CONT6,F
        goto    RINGER1
        goto    RXTTY2          ;SI PASAN 4.2 SEG DA ERROR EN CONEXION


;*******************************************************************************
*******************
;               RUTINA PARA HABILITAR/DESHABILITAR INTERRUPTOS
;*******************************************************************************
*******************
MAIN1.0

        bsf     INTCON, RBIE    ;habilita rb7 interrupto
        bcf     INTCON, INTE    ;deshab. interrupto externo en RB0
;*******************************************************************************
*******************
;                       RUTINA PARA CONEXION DE RX O TX
```

```
;*****************************************************************************
*******************

MAIN1

        call    RutinaLCD           ;inicializa LCD
        clrf    CONT_DIS            ;borra contador LCD
        call    RETRASO
        call    CUATRO              ;escribe CONECTANDO

        btfsc   rx_tty              ;si RX se va a RXTTY
        goto    RXTTY
        btfsc   tx_tty              ;si TX se va a TXTTY
        goto    TXTTY

;*****************************************************************************
*******************
;                       RUTINA QUE INICIALIZA CONVERSACION
;*****************************************************************************
*******************

MAIN2

        call    RutinaLCD           ;inicializa LCD
        clrf    CONT_DIS            ;borra contador LCD
        call    RETRASO
        bsf     offhook             ;prende bandera de teléfono descolgado
        bsf     INTCON, INTE        ;hab. interrupto externo en RB0
        bsf     PORTE, 2            ;PRENDO LED DE CONECCION

;*****************************************************************************
*******************
;                       RUTINA DE CONVERSACION
```

```
;*****************************************************************************
*********************

MAIN3

        btfss   offhook             ;si presionó ESC cuelga
        goto    MAINEXIT            ;y se va a MAINEXIT
        nop
        nop
        nop
        nop
        nop
        nop
        goto    MAIN3

;*****************************************************************************
*********************
;                       RUTINA DE FIN DE CONVERSACION
;*****************************************************************************
*********************

MAINEXIT

        btfss   fin                 ;si presionó ESC se va a INICIO
        goto    INICIO              ;reinicia TTY
        nop
        nop
        goto    MAIN3

;*****************************************************************************
*********************
;               RECEPCION DEL HANDSHAKE
```

```
;*****************************************************************************
*******************

RXTTY

        movlw   .67
        movwf   CONT3
        movlw   .53
        movwf   CONT2
        movlw   .173
        movwf   CONT1           ;inicializa espera de 13seg

RXTTY1

        btfss   rx_tty          ;si hace conexión se va a MAIN2
        goto    MAIN2

        decfsz  CONT1,F
        goto    RXTTY1
        decfsz  CONT2,F
        goto    RXTTY1
        decfsz  CONT3,F
        goto    RXTTY1
        btfss   rx_tty          ;si más de 13seg da error en conexión
        goto    MAIN2           ;si hace conexión se va a MAIN2

RXTTY2

        call    RutinaLCD       ;inicializa LCD
        clrf    CONT_DIS        ;borra contador LCD
        call    RETRASO
        call    CINCO           ;escribe error en conexión
        bsf     fin             ;prende bandera de fin de conversación
```

```
        bcf     PORTB, 6            ;apago línea
        bcf     ring                ;apago bandera de ring
        bsf     INTCON, INTE        ;hab. interrupto externo en RB0

        goto    MAINEXIT            ;se va a MAINEXIT


;*****************************************************************************************
*******************
;               TRANSMISION DEL HANDSHAKE
;*****************************************************************************************
*******************

TXTTY

        movlw   D'20'
        movwf   CONT4

TXTTY20SEG

        movlw   D'6'
        movwf   CONT3
        movlw   D'24'
        movwf   CONT2
        movlw   D'169'
        movwf   CONT1               ;inicializa espera de 20seg

TXTTY1

        movlw   'S'
        movwf   TEMP                ;guarda S en TEMP
        btfss   tx_tty              ;si recibe A se va a MAIN2
        goto    MAIN2
```

```
decfsz  CONT1,F
goto    TXTTY1

decfsz  CONT2,F
goto    TXTTY1

decfsz  CONT3,F
goto    TXTTY1          ;si menor de 1seg regresa a TXTTY1

bcf     INTCON, RBIE    ;habilita rb7 interrupto

call    VCO
call    RETRASO         ;cada 1seg manda S

bsf     INTCON, RBIE    ;habilita rb7 interrupto

decfsz  CONT4,F
goto    TXTTY20SEG      ;se va a TXTTY20SEG

btfss   tx_tty          ;si recibe A se va a MAIN2
goto    MAIN2

call    RutinaLCD       ;inicializa LCD
clrf    CONT_DIS        ;borra contador LCD
call    RETRASO
call    CINCO           ;escribe error en conexión
bsf     fin             ;prende badera de fin de conexión

bcf     PORTB, 6        ;apago línea
bsf     INTCON, INTE    ;hab. interrupto externo en RB0
goto    MAINEXIT        ;se va a MAINEXIT
```

```
;*****************************************************************************
*********************
;               Rutina para escribir menus de inicio
;*****************************************************************************
*********************
UNO

        clrf    CON1
        movlw   .20                     ;mueve a word el valor donde finaliza
        movwf   WORD                    ;el puntero de programa
        goto    MENUS                   ;va a leer memoria de programa

;*****************************************************************************
*********************
;               Rutina para escribir linea
;*****************************************************************************
*********************
DOS

        movlw   .20
        movwf   CON1
        movlw   .23                     ;mueve a word el valor donde finaliza
        movwf   WORD                    ;el puntero de programa
        goto    MENUS                   ;va a leer memoria de programa

;*****************************************************************************
*********************
;               Rutina para escribir ocupado
;*****************************************************************************
*********************
TRES
```

```
        movlw  .23
        movwf  CON1
        movlw  .27                  ;mueve a word el valor donde finaliza
        movwf  WORD                 ;el puntero de programa
        goto   MENUS                ;va a leer memoria de programa
;*******************************************************************************
**********************
;               Rutina para escribir conectando
;*******************************************************************************
**********************
CUATRO

        movlw  .27
        movwf  CON1
        movlw  .32                  ;mueve a word el valor donde finaliza
        movwf  WORD                 ;el puntero de programa
        goto   MENUS                ;va a leer memoria de programa

;*******************************************************************************
**********************
;               Rutina para escribir error en conexión
;*******************************************************************************
**********************

CINCO

        movlw  .32
        movwf  CON1
        movlw  .41                  ;mueve a word el valor donde finaliza
        movwf  WORD                 ;el puntero de programa
        goto   MENUS                ;va a leer memoria de programa
```

```
;*******************************************************************************
*******************
;               Rutina para escribir RING
;*******************************************************************************
*******************

SEIS

        movlw   .9
        movwf   CON1
        movlw   .11             ;mueve a word el valor donde finaliza
        movwf   WORD            ;el puntero de programa
        goto    MENUS           ;va a leer memoria de programa


;*******************************************************************************
*******************
;               Rutina para escribir esperando llamada
;*******************************************************************************
*******************
SIETE

        movlw   .41
        movwf   CON1
        movlw   .50             ;mueve a word el valor donde finaliza
        movwf   WORD            ;el puntero de programa
        goto    MENUS           ;va a leer memoria de programa


;*******************************************************************************
*******************
;               Rutina para leer memoria de programa
```

```
;*******************************************************************************
*******************

MENUS:

        BSF     STATUS, RP1
        BCF     STATUS, RP0         ;BANCO 2
        MOVLW   B'00011000'
        MOVWF   EEADRH              ;MSBBYTE DEL PROGRAM ADDRESS
        MOVF    CON1, W             ;MANDA A W EL CON1
        MOVWF   EEADR               ;LSBBYTE DEL PROGRAM ADDRESS
        BSF     STATUS, RP0         ;BANCO 3
        BSF     EECON1, EEPGD       ;APUNTE A MEMORIA DE PROGRAMA
        BSF     EECON1, RD          ;LEE
        NOP
        NOP

        BANCO2
        BCF     STATUS, C           ;CARRY EN 0
        RLF     EEDATH, W           ;CORRO EL MSBYTE LEFT
        MOVWF   DATH                ;LO MANDO A DATH
        BCF     STATUS, C           ;CARRY EN 0
        RLF     EEDATA, W           ;SACO EL BIT DE MSBYTE AL CARRY
        BTFSC   STATUS, C           ;MIRA SI SE PRENDIO EL CARRY
        BSF     DATH, 0             ;PRENDE EL LSB DEL MSBYTE SI EL CARRY
                                    ;ERA UNO
        MOVF    DATH, W             ;EN W ESTA EL MSBYTE
        BCF     STATUS, RP1         ;BANCO 0
        call    RETRASO
        BANCO2
        MOVF    DATH, W             ;EN W ESTA EL MSBYTE
        BCF     STATUS, RP1         ;BANCO 0
        call    Mayus_otro          ;MANDA LA PRIMERA LETRA GUARDADA EN
```

```
                                ;LINEA DE PROGRAMA
        call    DELAY0
        call    DELAY0
        BANCO2
        BCF     EEDATA, 7       ;BORRO EL BIT QUE NO SIRVE
        MOVF    EEDATA, W       ;EN W ESTA EL LSBYTE
        BANCO0
        call    Mayus_otro      ;MANDA LA SEGUNDA LETRA GUARDADA EN
                                ;LINEA DE PROGRAMA
        INCF    CON1,F          ;AUMENTA CON1 PARA CAMBIAR DE LINEA
                                ;DE PROGRAMA
        MOVF    WORD, W         ;MANDA A W LO DE WORD
        XORWF   CON1, W
        BTFSS   STATUS, Z       ;SI W ES IGUAL A CON1 SALTE LINEA
        GOTO    MENUS
        return


;*******************************************************************************
*******************
;               Rutina para chequear que opción de inicio se presionó
;*******************************************************************************
*******************
CHECKINICIO

        MOVWF   KEYBD
        MOVLW   '1';A'1'
        XORWF   KEYBD, W        ;COMPARA SI ES 1 LA OPCION
        BTFSC   STATUS, Z       ;SI ERA DIFERENTE SE SALTEA
        GOTO    TX              ;SI ES 1 SE VA A TX
        MOVLW   '2'
        XORWF   KEYBD, W        ;COMPARA SI ES 2 LA OPCION
        BTFSC   STATUS, Z       ;SI ERA DIF SE SALTEA
```

```
        GOTO  RXCONRING         ;SI ES 2 SE VA A RX CON RING
        MOVLW  '3'
        XORWF  KEYBD, W         ;COMPARA SI ES 3 LA OPCION
        BTFSC   STATUS, Z       ;SI ERA DIF SE SALTEA
        GOTO  RXSINRING         ;SI ES 3 SE VA A RX SIN RING
        return

;*****************************************************************************
*******************
;               Rutina para chequear que opción de inicio se presionó
;*****************************************************************************
*******************
TX

        bsf     PORTB, 6        ;contesta la línea
        bcf     inicio          ;baja bandera de inicio
        bsf     linea           ;prende bandera de linea u ocupado
        bsf     tx_tty          ;sube bandera de transmisión del tty
        goto    RutinaLCD       ;borra LCD


;*****************************************************************************
*******************
;               Rutina para chequear que opción de inicio se presionó
;*****************************************************************************
*******************

RXCONRING

        bcf     inicio          ;baja bandera de inicio
        bsf     ring            ;sube bandera de ring
        bsf     rx_tty          ;sube bandera de recepción del tty
```

```
        call    RutinaLCD           ;inicializa LCD
        clrf    CONT_DIS            ;borra contador LCD
        call    RETRASO
        call    SIETE               ;escribe esperando llamada
        return
;*******************************************************************************
*********************
;               Rutina para chequear que opción de inicio se presionó
;*******************************************************************************
*********************

RXSINRING

        bsf     PORTB, 6            ;contesta la línea
        bcf     inicio              ;baja bandera de inicio
        bsf     rx_tty              ;sube bandera de recepción del tty
        goto    RutinaLCD           ;borra LCD


;*******************************************************************************
*********************
;                       Rutina de Inicialización de LCD
;*******************************************************************************
*********************

RutinaLCD

        bcf     STATUS, RP0         ;asegurando estar en
        bcf     STATUS, RP1         ;***banco 0***
        bcf     PORTB, 4            ;pin E = 0
        goto    InicLCD             ;borra LCD

InicLCD                             ;Rutina de inicialización del LCD
```

```
        bcf     PORTB, 3            ;pin RS = 0
        movlw   b'00111100'         ;modo de 2 líneas
        call    Enable              ;display on
        movlw   b'00001111'
                                    ;Display On, cursor On
        call    Enable              ;blink On
        movlw   b'00000001'
        call    Enable2     ;borra Display
        movlw   b'00000111'         ;modo de incrementar
        call    Enable              ;shift on

        return
;*****************************************************************************************
*********************
;                               Retraso de LCD
;*****************************************************************************************
*********************

Enable

        movwf   TEMP1
        andlw   b'00001111'
        movwf   PORTA               ;PONE LOS LSBITS EN RA0-RA4 PARA LCD
        swapf   TEMP1, W
        andlw   b'00001111'
        movwf   PORTC               ;PONE LOS MSBITS EN RC0-RC3 PARA LCD
        bsf     PORTB, 4            ;pin E = 1

DELAY0

        movlw   .75
        movwf   CONT1               ;inicializa retraso
```

```
DelayE

        decfsz  CONT1, f            ;si contador=0 salta
        goto    DelayE
        bcf     PORTB, 4            ;pin E = 0
        return


;*****************************************************************************
*******************
;                               Retraso 2 de LCD
;*****************************************************************************
*******************

Enable2

        movwf   TEMP1
        andlw   b'00001111'
        movwf   PORTA               ;PONE LOS LSBITS EN RA0-RA4 PARA LCD
        swapf   TEMP1, W
        andlw   b'00001111'
        movwf   PORTC               ;PONE LOS MSBITS EN RC0-RC3 PARA LCD
        bsf     PORTB, 4            ;pin E = 1

DELAY1
        clrf    TMR0                ;borra TMR0

DelayE2
        movlw   .35
        xorwf   TMR0, W
        BTFSS   STATUS, Z           ;si TMR0=35 salta
        goto    DelayE2
```

```
        bcf     PORTB, 4                ;pin E = 0
        return


;*****************************************************************************
*********************
;                               Retraso de TMR0 overflow
;*****************************************************************************
*********************
DELAY2
        clrf    TMR0                    ;borra TMR0

DelayE3
        btfss   INTCON, 2               ;si hay overflow, TOIF=1
        goto    DelayE3
        bcf     PORTB, 4                ;se sale de retraso y pin E = 0
        return

;*****************************************************************************
*********************
;                               Retraso de 1.5mseg
;*****************************************************************************
*********************

RETRASO
        movlw   D'2'
        movwf   CONT2
        movlw   D'241'
        movwf   CONT1                   ;inicializa retraso de 1.5mseg
LOOPRETRASO

        decfsz  CONT1, F
        goto    LOOPRETRASO
```

```
        decfsz  CONT2, F                ;si contadores = 0 salta
        goto    LOOPRETRASO
        return
;*****************************************************************************
*******************
;                               Retraso de 0.66seg
;*****************************************************************************
*******************
RETRASO1.0

        movlw   .6
        movwf   CONT3
        movlw   .24
        movwf   CONT2
        movlw   .168
        movwf   CONT1                   ;inicializa retraso de 0.66seg

RETRASO1.0S

        decfsz  CONT1,F
        goto    RETRASO1.0S
        decfsz  CONT2,F
        goto    RETRASO1.0S
        decfsz  CONT3,F                 ;si contadores = 0 salta
        goto    RETRASO1.0S
        return

;*****************************************************************************
*******************
;                               Retraso de 0.33seg
;*****************************************************************************
*******************
RETRASO0.33
```

```
        movlw  D'2'
        movwf  CONT3
        movlw  D'174'
        movwf  CONT2
        movlw  D'141'
        movwf  CONT1                ;inicializa retraso de 0.33seg

RETRASO0.33S
        decfsz  CONT1,F
        goto    RETRASO0.33S
        decfsz  CONT2,F
        goto    RETRASO0.33S
        decfsz  CONT3,F             ;si contadores = 0 salta
        goto    RETRASO0.33S
        return

;*******************************************************************************
*********************
;                               Retraso de 39uSeg
;*******************************************************************************
*********************
RETRASO39uSEG

        movlw  D'12'
        movwf  CONT1                ;inicializa retraso de 39useg

RETRASO39uSEG1
        decfsz  CONT1,F             ;si contador = 0 salta
        goto    RETRASO39uSEG1
        return

;*******************************************************************************
********************
```

```
;                              Rutina de Interrupto (inicio)
;*******************************************************************************
*********************

INTERR
        bcf     STATUS, RP0
        bcf     STATUS, RP1             ;***banco 0***
        movwf   W_SAVE                  ;guarda el cont. de W en W_SAVE
        movf    STATUS, w
        movwf   STAT_SAVE               ;guarda STATUS
        movf    PCLATH, w
        movwf   PCLH_SAVE               ;guarda PCLATH

;*******************************************************************************
*********************
;                          Rutina de Interrupto (determinación de fuente)
;*******************************************************************************
*********************

        btfsc   PIR1,RCIF               ;FUE EL RCREG LLENO
        goto    RECEPCION
        btfsc   INTCON,RBIF             ;FUE EL RB7?
        goto    CHANGEPIN               ;SE VA A RUTINA PARA DEMODULAR FREC

DET_INTERR
        btfsc   INTCON, INTF            ;fue el rb0/INT0?

;*******************************************************************************
*********************
;                                  Rutina de Teclado
;*******************************************************************************
*********************
TECLADO
```

```
        call    RECIBIR             ;si -> Recibir, no -> Final
        call    Chequeo
        goto    FIN_INTERR

;*****************************************************************************
*******************
;                           Rutina de Interrupto (final)
;*****************************************************************************
*******************

FIN_INTERR

        clrf    STATUS              ;***banco 0***
        movf    PCLH_SAVE, w
        movwf   PCLATH              ;restaura los registros
        movf    STAT_SAVE, w                ;STATUS, PCLATH y W
        movwf   STATUS
        swapf   W_SAVE, f           ;mueve W_SAVE a w sin afectar
        swapf   W_SAVE, w           ;el STATUS
        bcf     parity
        retfie

;*****************************************************************************
*******************
;                       Rutina de Demodulación de FSK
;*****************************************************************************
*******************

CHANGEPIN

        btfsc   linea               ;si bandera=1 se va a CHECKLINEA
        goto    CHECKLINEA
```

```
        btfsc   ring            ;si bandera=1 se va a RINGON
        goto    RINGON

        btfsc   PIR1, TMR2IF    ;si hay overflow en TMR2 se sale
        goto    clr_overflow

        movfw   T0_count
        movwf   Tlast           ;guarda en Tlast lo pasado de TMR2

        movfw   TMR2            ;lee Timer 2
        movwf   T0_count        ;lo guarda en T0_count
        clrf    TMR2            ;inicializa el timer2

FSKHIGH

        movlw   .43
        subwf   T0_count, W
        btfss   STATUS, C       ;SI >= 45 ENTONCES SALTA
        goto    FSKLOW

        movlw   .53
        subwf   T0_count, W
        btfsc   STATUS, C       ;SI < 55 ENTONCES ES 2.5KHZ
        goto    FSKLOW
        bcf     PORTB,5         ;pone en 0 rb5
        goto    clr_rbintf

FSKLOW

        movlw   .76
        subwf   T0_count, W
        btfss   STATUS, C       ;SI >= 78 ENTONCES SALTA
```

```
        goto    FSKANTES

        movlw   .90
        subwf   T0_count, W
        btfsc   STATUS, C           ;SI < 88 ENTONCES ES 1.5KHZ
        goto    FSKANTES
        bsf     PORTB,5             ;pone en 1 en rb5
        goto    clr_rbintf

FSKANTES

        movlw   .43;.45
        subwf   Tlast, W
        btfss   STATUS, C           ;SI >= 45 ENTONCES SALTA
        goto    FSKLOW2

        movlw   .51;.55
        subwf   Tlast, W
        btfsc   STATUS, C           ;SI < 55 ENTONCES ES 2.5KHZ
        goto    FSKLOW2
        bcf     PORTB,5             ;pone 0 en rb5
        goto    clr_rbintf

FSKLOW2

        movlw   .76
        subwf   Tlast, W
        btfss   STATUS, C           ;SI >= 78 ENTONCES SALTA
        goto    clr_rbintf1
        movlw   .90 ;.88
        subwf   Tlast, W
        btfsc   STATUS, C           ;SI < 88 ENTONCES ES 1.5KHZ
        goto    clr_rbintf1
```

```
        bsf     PORTB,5             ;pone 1 en rb5
        goto    clr_rbintf

clr_overflow
        bcf     PIR1, TMR2IF        ;borra flag de TMR2

clr_rbintf1
        nop


;*****************************************************************************************
*********************
;                       Rutina para salir de interrupto de flanco
;*****************************************************************************************
*********************

clr_rbintf

        movf    PORTB, F            ;lee portb para quitar el mismatch
        bcf     INTCON, RBIF        ;borra bandera de int por flanco
                                    ;del portb
        goto    FIN_INTERR

;*****************************************************************************************
*********************
;                               Rutina de RING
;*****************************************************************************************
*********************

RINGON

        BANCO0
        btfsc   pico                ;si es pico se va a PICOCHECK
```

```
        goto    PICOCHECK
        nop
        bcf     picosleep           ;borra bandera de picos

        call    RutinaLCD
        clrf    CONT_DIS
        bsf     PORTE, 2            ;PRENDE LAMPARA
        call    RETRASO

        call    SEIS                ;manda RING a LCD

        movlw   D'22'
        movwf   CONT6
        movlw   D'100'
        movwf   CONT5
        movlw   D'101'
        movwf   CONT4               ;inicializa espera
        goto    clr_rbintf

PICOCHECK

        bcf     pico                ;borra bandera de picos
        call    RETRASO
        call    RETRASO
        bsf     picosleep           ;borra la otra bandera de picos
        goto    clr_rbintf

;*******************************************************************************
*******************
;               Rutina de chequeo si ocupado o da linea o tono inicial
;*******************************************************************************
*******************
CHECKLINEA
```

```
        btfsc   tonoinicio              ;si esta prendida se va a TONOINICIAL
        goto    TONOINICIAL
        btfsc   lineauno                ;si esta prendida se va a ESPERAR
        goto    ESPERAR
        btfsc   ocupado         ;si esta prendida se va a OCUPADO
        goto    OCUPADO
        btfsc   da_linea        ;si esta prendida se va a LINEA
        goto    LINEA
        btfss   linetemp                ;si esta prendida se va a LINETEMP
        goto    clr_rbintf              ;si ninguna está prendida se sale
        goto    LINETEMP

;*******************************************************************************
*********************
;                       Rutina para empezar a detectar tono inicial
;*******************************************************************************
*********************
TONOINICIAL

        bcf     tonoinicio              ;baja bandera de tono inicial
        call    RETRASO
        goto    clr_rbintf

;*******************************************************************************
*********************
;                       Rutina de detección de tono inicial
;*******************************************************************************
*********************

LINETEMP

        call    RutinaLCD               ;borra LCD
```

```
        clrf    CONT_DIS            ;borra contador LCD
        call    RETRASO
        movlw   D'2'
        movwf   CONT3
        movlw   D'5'
        movwf   CONT2
        movlw   D'188'
        movwf   CONT1               ;reinicializa vars de espera
        goto    clr_rbintf


;*****************************************************************************
*******************
;       Rutina de chequeo si da linea
;*****************************************************************************
*******************

LINEA

        movfw   TMR2                ;lee Timer 2
        movwf   T0_count            ;lo guarda en T0_count para la próxima
        clrf    TMR2                ;inicializa el timer2

        movlw   .192
        subwf   T0_count, W
        btfss   STATUS, C           ;si < 192 -> es frec.mayor a 600Hz
                                    ;y se sale
        goto    LINEACONTINUA2

        bsf     PORTE, 2            ;parpadea led indicando dar línea

        call    RutinaLCD           ;borra LCD
```

```
        clrf    CONT_DIS
        call    RETRASO
        btfsc   ocupado2            ;si está ocupado se va a OCUPADOTEMP
        goto    OCUPADOTEMP
        movlw   .18
        subwf   CONT6, W
        btfss   STATUS, C           ;SI >= 18 ENTONCES SALTA
        goto    LINEATEMP           ;si no se va a LINEATEMP
        movlw   .22
        subwf   CONT6, W
        btfsS   STATUS, C           ;SI < 21 ENTONCES VA A OCUPADOTEMP
        goto    OCUPADOTEMP

LINEATEMP

        bsf     da_linea        ;prende bandera de dar línea
        call    DOS                     ;escribe LINEA en LCD

LINEACONTINUA

        movlw   D'22'
        movwf   CONT6
        movlw   D'100'
        movwf   CONT5
        movlw   D'101'
        movwf   CONT4               ;reinicializa vars. de espera
        goto    clr_rbintf

LINEACONTINUA2

        btfsc   PIR1, TMR2IF        ;hay overflow en TMR2
        goto    clr_overflow1       ;si no hay overflow se sale
```

```
        goto    LINEACONTINUA       ;si hay se va a LINEACONTINUA

OCUPADOTEMP

        bsf     PORTE, 2            ;prende led
        bsf     ocupado2            ;prende bandera ocupado2
        goto    LINEACONTINUA


;*******************************************************************************
*********************
;                       Rutina de esperar entre cada dtmf
;*******************************************************************************
*********************

ESPERAR

        call    RETRASO0.33         ;retraso de 0.33seg
        decfsz  CONT_TEMP,F         ;si acaba de marcar se sale
        goto    clr_rbintf

        bcf     lineauno            ;borra bandera de dtmf
        goto    clr_rbintf
clr_overflow1

        bcf     PIR1, TMR2IF        ;borra flag de TMR2
        goto    clr_rbintf


;*******************************************************************************
*********************
;                       Rutina de chequeo si ocupado
```

```
;*****************************************************************************************
*******************

OCUPADO

        movfw   TMR2                    ;lee Timer 2
        movwf   T0_count                ;lo guarda en T0_count para la próxima
        clrf    TMR2                    ;inicializa el timer2
        movlw   .192
        subwf   T0_count, W
        btfss   STATUS, C               ;SI < 192 -> FREC. MAYOR A 600HZ
                                        ;Y SE SALE
        goto    OCUPADOCONTINUA2

        btfsc   lineaerror              ;si primera vuelta se va a ESPERAR2
        goto    ESPERAR2

        bsf     PORTE, 2                ;parpadea led indicando estar ocupado
        call    RutinaLCD               ;borra LCD
        clrf    CONT_DIS                ;borra contador LCD
        call    RETRASO
        call    TRES                    ;escribe ocupado en LCD

OCUPADOCONTINUA

        movlw   D'3'
        movwf   CONT3
        movlw   D'205'
        movwf   CONT2
        movlw   D'66'
        movwf   CONT1                   ;reinicializa vars. de espera
        goto    clr_rbintf
```

```
OCUPADOCONTINUA2

        btfsc   PIR1, TMR2IF            ;si hay overflow en TMR2
        goto    clr_overflow2
        goto    OCUPADOCONTINUA ;se va a OCUPADOCONTINUA

ESPERAR2

        call    RETRASO1.0              ;llama retraso 1seg
        bsf     PORTE, 2                ;prende led
        bcf     lineaerror              ;borra bandera de lineaerror
        goto    OCUPADOCONTINUA

clr_overflow2

        bcf     PIR1, TMR2IF            ;borra flag de TMR2
        goto    clr_rbintf


;*******************************************************************************
********************
;                               RUTINA DE RECEPCION
;*******************************************************************************
********************


RECEPCION

        MOVF  RCREG, W
        MOVWF       RX              ;MANDO LO DE RCREG A RX
        movf    RX, w               ;chequea si presionó BACKSPACE para
                                    ;borrar letra
        xorlw   b'11010000'         ;0x66
```

```
        btfsc   STATUS, Z
        goto    RXERRASE            ;VA A BORRAR ULTIMA LETRA

        movf    RX, w               ;chequea si presionó ESCAPE para
        xorlw   b'11111100'         ;cortar llamada
        btfsc   STATUS, Z
        goto    ESCAPAR             ;VA A BORRAR ULTIMA LETRA

        btfsc   rx_tty              ;si está en RX se va a RECEPCIONTTYRX
        goto    RECEPCIONTTYRX
        btfsc   tx_tty              ;si está en TX se va a RECEPCIONTTYTX
        goto    RECEPCIONTTYTX


        bcf     PORTB, 4            ;pin E = 0
        bsf     PORTB, 3            ;pin RS = 1
        movf    RX, W
        call    Mayus_otro          ;escribe lo recibido en LCD
        call    RETRASO
        goto    FIN_INTERR          ;sale
;*********************************************************************************
*******************
;                       Rutina para contestar handshake del TTY RX
;*********************************************************************************
*******************
RECEPCIONTTYRX

        movf    RX, w               ;CHEQUEA SI SE RECIBIO UNA S PARA
        xorlw   'S'                 ;EL ACKNOWLEDGE
        btfss   STATUS, Z
        goto    FIN_INTERR
        bcf     rx_tty              ;borra bandera de conexión en RX
        call    RETRASO
```

```
        call    RETRASO
        movlw   'A'
        movwf   TEMP
        call    VCO                     ;manda A al TTY TX
        goto    FIN_INTERR

;*******************************************************************************
*******************
;                       Rutina de recepción de acknowledge de TTY TX
;*******************************************************************************
*******************
RECEPCIONTTYTX

        movf    RX, w                   ;CHEQUEA SI SE RECIBIO UNA A PARA
        xorlw   'A'                     ;EL ACKNOWLEDGE
        btfss   STATUS, Z
        goto    FIN_INTERR
        bcf     tx_tty                  ;borra bandera de conexión en TX
        goto    FIN_INTERR

;*******************************************************************************
*******************
;                       Rutina para terminar conversación
;*******************************************************************************
*******************

ESCAPAR

        bcf     offhook                 ;apaga bandera de linea abierta
        bcf     PORTB, 6                ;apaga linea de teléfono
        bcf     PORTE, 2                ;apaga led de conección
        bsf     fin                     ;prende bandera de final programa
        bcf     INTCON, RBIE            ;apaga interrupto de rb7
```

```
        goto    FIN_INTERR

;*****************************************************************************
*******************
;                       Rutina para borrar ultima letra en LCD
;*****************************************************************************
*******************

RXERRASE

        call    ERRASE          ;VA A BORRAR ULTIMA LETRA
        goto    FIN_INTERR

;*****************************************************************************
*******************
;                   Rutina de Recepción de la señal del teclado
;*****************************************************************************
*******************

RECIBIR                         ;estamos en ***banco 0***

        btfsc   PORTB, 0        ;revisa el 'start bit'
        goto    SalirR          ;si -> RecKeybd, no -> salir de rutina


RecKeybd

        btfsc   PORTB, 1        ;revisa el pin de clock, si está bajo
        goto    RecKeybd        ;->salte, no -> loop
        movlw   b'11110000'
        andwf   FLAGS, f        ;borra las cuatro banderas inferiores
        clrf    KEYBD                   ;borra el byte de dato del teclado
```

```
        movlw   0x08
        movwf   CONT1           ;inicializa el contador para
                                ;recibir 8 bits

RecLoop

        call    DelayCambioBit;llama a rutina de delay basado en
                                ;estado de clock
        btfsc   rx_err          ;error en recepción? si -> salir,
                                ;no -> continuar
        goto    SalirR
        bcf     STATUS, C       ;borra el carry
        btfsc   PORTB, 0        ;revisa el pin RB0: set -> C = 1,
                                ;no -> C = 0
        bsf     STATUS, C
        rrf     KEYBD, f        ;rote KEYBD a la derecha a través de C
        decfsz  CONT1, f        ;decremente el contador, salte
                                ;si da cero
        goto    RecLoop

RevParidad                      ;rutina para revisión de paridad

        call    DelayCambioBit
        btfsc   rx_err
        goto    SalirR

RecStopBit

        call    DelayCambioBit;espera cambio de bit
        btfsc   rx_err
        goto    SalirR
        btfsc   PORTB, 0        ;Stop Bit: 0 -> SalirR
        bsf     bien_r          ;1 -> enciende la bandera bien_r
```

```
        goto    SalirR

DelayCambioBit

        bcf     INTCON, T0IF        ;borra bandera del Timer 0
        clrf    TMR0                ;borra el Timer 0

DelayCambioBa

        btfsc   INTCON, T0IF        ;revisa si ha ocurrido un overflow
                                    ;del Timer 0
        goto    SalirDelay          ;si -> Salir, no -> skip
        btfss   PORTB, 1            ;revisa el clock, espera q' se ponga
        goto    DelayCambioBa ;en alto para continuar

DelayCambioBb

        btfsc   INTCON, T0IF
        goto    SalirDelay
        btfsc   PORTB, 1            ;revisa clock, espera q' se ponga
        goto    DelayCambioBb;en bajo para regresar a rutina
        return                                  ;de recepción


SalirDelay

        bsf     rx_err
        return

SalirR

        bcf     INTCON, INTF        ;borra la bandera de interrupt
        return                      ;***banco 0***
```

```
;*****************************************************************************************
*******************
;                    Rutina de Transmisión de código al teclado
;*****************************************************************************************
*******************

TRANSMITIR

        movwf   KEYBD                   ;mueve a KEYBD el byte a transmitir
        movlw   0x08
        movwf   CONT1           ;inicializa CONT1 a 8
        bsf     STATUS, RP0
        bcf     STATUS, RP1     ;***banco 1***
        bcf     TRISB, 0        ;coloca RB0 y RB1 como salidas
        bcf     TRISB, 1

        bcf     STATUS, RP0     ;***banco 0***

InicioTrans

        bcf     PORTB, 1        ;forzar pin de clock a nivel bajo
        call    DelayTrans
        call    DelayTrans
        call    DelayTrans

        bcf     PORTB, 0        ;forzar pin de data a nivel bajo
        bsf     STATUS, RP0     ;***banco 1***
        bsf     TRISB, 1        ;libera el pin de clock
        bcf     STATUS, RP0     ;***banco 0***
        bsf     parity          ;inicializa bit de paridad
        bcf     INTCON, T0IF    ;se utilizará el Timer 0
```

```
        clrf    TMR0            ;para verificación de rutina

LoopTrans

        btfsc   INTCON, T0IF    ;revisa T0IF: set -> ErrorTrans
        goto    ErrorTrans      ;no ->continúe
        btfsc   PORTB, 1        ;revisa el pin del clock y espera
        goto    LoopTrans       ;que sea bajo
        rrf     KEYBD, f        ;rote KEYBD a la derecha a través
        btfsc   STATUS, C       ;del carry.  Si C es '1' va a la
        goto    CheqParidad     ;rutina CheqParidad, si es '0'
        bcf     PORTB, 0        ;pone en bajo la línea de data

LoopTrans2

        btfsc   INTCON, T0IF    ;revisa por overflow del Timer 0
        goto    ErrorTrans
        btfss   PORTB, 1        ;chequea la línea de clock y espera
        goto    LoopTrans2      ;que se ponga en alto
        clrf    TMR0            ;borra el Timer 0
        decfsz  CONT1, f        ;decrementa en uno CONT1
        goto    LoopTrans

TransParidad                    ;Ahora el bit de paridad

        btfsc   INTCON, T0IF    ;revisa overflow del Timer 0
        goto    ErrorTrans
        btfsc   PORTB, 1        ;espera q' la línea de clock se ponga
        goto    TransParidad    ;baja para continuar
        btfsc   parity          ;revisa bandera parity
        goto    ActParidad      ;set -> ActParidad, no -> RB0 = 0
        bcf     PORTB, 0
```

```
TransParidad2

        btfsc   INTCON, T0IF        ;revisa Timer 0
        goto    ErrorTrans
        btfss   PORTB, 1            ;espera que la línea clock sea un nivel
        goto    TransParidad2       ;alto para continuar
        clrf    TMR0                ;borra el Timer 0

TransStop

        btfsc   INTCON, T0IF        ;revisa el Timer 0
        goto    ErrorTrans
        btfsc   PORTB, 1            ;espera que la línea clock sea un nivel
        goto    TransStop           ;bajo para transmitir el Stop bit
        bsf     PORTB, 0            ;Stop bit: RB0 = 1

TransStop2

        btfsc   INTCON, T0IF        ;revisa la bandera del Timer 0
        goto    ErrorTrans
        btfss   PORTB, 1            ;revisa el pin de clock, espera que sea
        goto    TransStop2          ;un nivel alto para continuar
        bsf     STATUS, RP0         ;cambio al
        bsf     TRISB, 0            ;regresa el pin de data RB0
                                    ;como entrada
        bcf     STATUS, RP0         ;***banco 0***
        clrf    TMR0                ;borra el Timer

RecAck

        btfsc   INTCON, T0IF        ;revisa bandera del Timer 0
        goto    ErrorTrans
```

```
        btfsc   PORTB, 1        ;revisa clock, espera que sea un nivel
        goto    RecAck          ;bajo para continuar
        btfsc   PORTB, 0        ;revisa el bit Acknowledge
        goto    ErrorTrans
        return

CheqParidad                     ;rutina para determinar el bit

        bsf     PORTB, 0        ;de paridad
        movlw   b'00000010'
        xorwf   FLAGS, f
        goto    LoopTrans2

ActParidad                      ;Activa el bit de paridad y

        bsf     PORTB, 0        ;pone alto el pin RB0
        goto    TransParidad2

DelayTrans                      ;delay de TX de PS2 de 119 us

        movlw   .40
        movwf   CONT2


TransLoop

        decfsz  CONT2, f
        goto    TransLoop
        return

ErrorTrans                      ;rutina de error

        bsf     STATUS, RP0     ;***banco 1***
```

```
        bcf     STATUS, RP1
        bsf     TRISB, 0            ;coloca de nuevo RB0 y RB1
        bsf     TRISB, 1            ;como entradas
        bcf     STATUS, RP0         ;***banco 0***
        bsf     tx_err              ;enciende la bandera de error en
        return                      ;transmisión

;*****************************************************************************
*******************
;                       Rutina de Chequeo del Byte recibido del teclado
;*****************************************************************************
*******************

Chequeo

        btfss   bien_r              ;revisa la bandera bien_r
        goto    SalirCh             ;set -> continúa, no set -> salir
        bcf     STATUS, Z           ;borra bandera Z

        btfsc   ring
        goto    ENTER

        btfsc   fin
        goto    FINAL               ;va a rutina para reiniciar TTY
                                    ;al presionar ESC
        movf    KEYBD, w            ;revisa si KEYBD es F0h, si -> enciende
        xorlw   0xF0                ;su respectiva bandera, no -> RevE1
        btfss   STATUS, Z
        goto    RevE0
        bsf     F0_flag
        goto    SalirCh             ;sale de la rutina
```

```
RevE0

        movf    KEYBD, w
        xorlw   0xE0            ;revisa si KEYBD es E0h, si -> enciende
        btfss   STATUS, Z       ;bandera, no -> RevF0
        goto    RevE1           ;RevF0
        bsf     E0_flag
        goto    SalirCh         ;sale de la rutina

RevE1

        movf    KEYBD, w        ;revisa si KEYBD es E1h, si -> rutina
        xorlw   0xE1            ;de botón Pause, no -> continúa
        btfsc   STATUS, Z
        goto    Pause_b     ;rutina para la tecla PAUSE

RevF4

        movf    KEYBD, w        ;revisa si el kboard da un enable
        xorlw   0xF4            ;al host
        btfsc   STATUS, Z
        goto    SalirCh

RevFA

        movf    KEYBD, w        ;revisa por el byte Acknowledge
        xorlw   0xFA            ;que manda el teclado después de
        btfss   STATUS, Z       ;recibir un byte
        goto    ByteNormal
        decfsz  CONT_FA, f
        goto    TransLEDs2
        movlw   .1
```

```
        movwf   CONT_FA
        goto    SalirCh

ByteNormal                      ;rutina para ver si ha llegado un
                                ;código MAKE
        btfss   E0_flag         ;o se ha completado un código BREAK
        goto    RevF0_flag
        btfss   F0_flag
        call    E0_make         ;llama a rutina especial para las
        bcf     E0_flag         ;teclas cuyo MAKE empieza con E0h
        bcf     F0_flag         ;borra las banderas
        goto    SalirCh

RevF0_flag

        btfss   F0_flag         ;si ninguna bandera está activada
        goto    RutinaLEDs      ;revise si hay cambio de LEDs
        bcf     F0_flag         ;si la bandera está encendida es un
        goto    SalirCh         ;código BREAK, regrese sin desplegar.

E0_make

        movlw   0x70
        addwf   KEYBD, f
        call    Despliegue
        call    VCO             ;llama a funcion que hace VCO
                                ;por software
        goto    SalirCh

Pause_b

        goto    SalirCh
```

```
RutinaLEDs                  ;rutina para revisar si la tecla
                            ;presionada cambia de estado los
                            ;leds del teclado
        bcf     STATUS, Z

        movf    KEYBD, w
        xorlw   0x58
        btfsc   STATUS, Z       ;si es 0x58 se va a CAPS
        goto    CAPS
        movf    KEYBD, w
        xorlw   0x77
        btfsc   STATUS, Z       ;si es 0x77 se va a NUM
        goto    NUM

        movf    KEYBD, w
        xorlw   0x7E
        btfsc   STATUS, Z
        goto    SCROLL          ;si es 0x7E se va a SCROLL

        btfsc   inicio          ;si bandera inicio se va a revisar
        goto    CHECKINI        ;opciones del menú

        movf    KEYBD, w        ;chequea si presionó ESCAPE
        xorlw   .118
        btfsc   STATUS, Z
        goto    ESCAPE
        movf    KEYBD, w        ;chequea si presionó BACKSPACE para
        xorlw   0x66            ;borrar ultima letra
        btfsc   STATUS, Z
        goto    CHKERRASE       ;LLAMA FUNCION PARA BORRAR ULTIMA LETRA

        call    Despliegue      ;llama función que escribe a LCD
        call    VCO             ;llama a función que hace VCO
```

```
                                    ;por software
        goto    SalirCh

ENTER

        movf    KEYBD, w            ;chequea si presionó enter
        xorlw   .90
        btfss   STATUS, Z
        goto    SalirCh
        bsf     PORTB, 6            ;ENCIENDE LINEA
        bcf     ring                ;apaga bandera de ring

FINAL

        movf    KEYBD, w            ;chequea si presionó ESCAPE
        xorlw   .118

        btfss   STATUS, Z
        goto    SalirCh
        bcf     fin                 ;apaga bandera para reiniciar TTY
        goto    SalirCh

ESCAPE

        bcf     offhook             ;apaga bandera de linea abierta
        movlw   b'11111100'
        movwf   TEMP

        call    VCO                 ;manda a apagar línea del otro TTY
        bcf     INTCON, RBIE        ;apaga interrupto de rb7
        bcf     PORTB, 6            ;apaga línea
        bcf     PORTE, 2            ;apaga conexión de línea
        bsf     fin                 ;prende bandera de final programa
```

```
        goto    SalirCh

CHKERRASE

        call    ERRASE          ;borra última letra en LCD
        movlw   b'11010000'     ;0x66
        movwf   TEMP
        call    VCO             ;manda a borrar última letra en el
        goto    SalirCh         ;otro TTY

CHECKINI

        call    Despliegue      ;se va a Despliegue
        goto    SalirCh

SCROLL

        movlw   b'00000001'
        goto    TransLEDs       ;se va a TransLEDS

NUM

        movlw   b'00000010'
        goto    TransLEDs       ;se va a TransLEDS

CAPS

        movlw   b'00000100'
        goto    TransLEDs       ;se va a TransLEDS
```

```
TransLEDs

        xorwf   LEDS, f         ;revisa si hay que prender o
        movlw   0xED            ;apagar leds
        call    TRANSMITIR      ;se va a transmitir
        btfsc   tx_err
        goto    SalirCh

TransLEDs2

        movf    LEDS, w         ;se va a prender o apagar leds
        call    TRANSMITIR
        goto    SalirCh

SalirCh

        bcf     bien_r
        return


;*****************************************************************************
*******************
;                       Rutina de Despliegue LCD
;*****************************************************************************
*******************

Despliegue

        call    RutinaTabla     ;llama a la rutina de lectura de Tabla
        btfsc   inicio          ;si es inicio va a revisar menú
        goto    CHECKINICIO

        btfsc   LEDS, 2         ;revisa si el bit CAPS está encendido
```

```
        goto    Mayus_otro          ;si -> continúe, no-> revise y cambie
        movwf   TEMP
        movlw   'A'                 ;b'01000001' ;si es A
        subwf   TEMP, W
        btfss   STATUS, C           ;SI >= QUE 45 ENTONCES SALTA
        goto    MayusTemp

        bcf     STATUS, C
        movlw   '['                 ;b'01011011' ;si es [
        subwf   TEMP, W
        btfsc   STATUS, C           ;SI ES MENOR QUE [
        goto    MayusTemp
        bsf     TEMP, 5

MayusTemp

        movf    TEMP, w

Mayus_otro

        bsf     PORTB, 3            ;prende RS

DISPLAYLCD

        movwf   TEMP
        andlw   b'00001111'
        movwf   PORTA               ;PONE LOS LSBITS EN RA0-RA4 PARA LCD
        swapf   TEMP, W
        andlw   b'00001111'
        movwf   PORTC               ;PONE LOS MSBITS EN RC0-RC3 PARA LCD

        bsf     PORTB, 4            ;prende E
        call    RETRASO39uSEG       ;espera 39useg
```

```
        bcf     PORTB, 4                ;apaga E

VCOAFTER

        bcf     STATUS, Z
        incf    CONT_DIS, f             ;increMente en uno el contador
        movlw   0x50
        xorwf   CONT_DIS, w             ;revisa si el contador es 80d
        btfss   STATUS, Z
        return
        clrf    CONT_DIS                ;si ya se llenó la LCD se borra
        goto    RutinaLCD               ;contador y LCD

;*******************************************************************************
*********************
;                       Rutina de VCO por software
;*******************************************************************************
*********************

VCO

;un I es frecuencia baja 1500Hz y un 0 es frecuencia alta 2500Hz

        movlw   .8
        movwf   CONT1

        call    tonehi                  ;MANDA LOW START BIT
        CLRF    POINTERA

        bcf     STATUS,C                ;BORRA EL CARRY
```

```
nextbit

        rrf   TEMP, F              ;SE MANDA EL LSB DE PRIMERO
        btfsc  STATUS,C
        goto    tonecheck
        call    tonehi             ;bit es 0 => manda frec. alta
        CLRF   POINTERA

nextbit2

        decfsz  CONT1, F
        goto   nextbit
        call    tonelo             ;MANDA EL HIGH STOP BIT
        CLRF   POINTERA
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        RETURN

tonecheck

        call    tonelo             ;bit es 1 => manda frec. baja
        CLRF   POINTERA
        goto    nextbit2

tonehi

        movlw  .23                 ;23 es 2.6kHz
        movwf  STEPA
```

```
        call    sendtone
        return

tonelo

        movlw  .13                  ;13 es 1488Hz
        movwf  STEPA
        call    sendtone
        return

;*****************************************************************************
*******************
;   RUTINA PARA HACER SENOIDAL PARA FSK QUE SALE POR EL PORTD
;*****************************************************************************
*******************

sendtone

        MOVLW  .100                 ;CON 100 DA 142.86HZ
        movwf   SINECOUNTH          ;empieza contador del ciclo en 100

v23loopsine2cyc

        goto    v23loopsine         ;pierde 2 ciclos para mantener un
                                    ;ciclo de 34 instrucciones
v23loopsine

        movlw  HIGH SINELOOKUP      ;mueve a w los 5 bits superiores
        movwf  PCLATH               ;de la dirección de la tabla y los
                                    ;pasa al PCLATH
        movfw  STEPA                ;obtiene tamaño de salto o step
        addwf   POINTERA,1          ;lo suma al puntero POINTERA
        movf    POINTERA,W          ;pone dirección de la onda en W
```

```
        andlw   .127                ;vuelve cero el bit 7 para tener
                                    ;índice de 7 bits pues la table es
        call    SINELOOKUP          ;de 128 datos va a buscar el valor
                                    ;de lasenoidal
        btfsc   POINTERA,7          ;si el puntero fue la 2a parte de
        subwf   ALLONES,0           ;la onda resta (255-W) y lo pone
                                    ;en W
        btfss   POINTERA,7
        nop                         ;si no lo es toma la muestra
                                    ;original
        movwf   NEXTVALUE           ;ponc el primer valor en NEXTVALUE
        nop
        nop
        nop
        movfw   NEXTVALUE

v23dacout

        movwf   PORTD               ;pone el valor de la onda en PORTD
        movlw   .120
        subwf   T0_count,0          ;pierde tiempo
        btfsc   STATUS,0
        nop
        btfss   STATUS,0
        nop
        nop
        nop
        nop
        nop
        decfsz  SINECOUNTH,F        ;Salta si ya se acabó
        goto    v23loopsine         ;regresa al ciclo
        retlw   0                   ;regresa de la rutina
```

```
;*****************************************************************************
********************
;                          RETROCEDE UNA DIRECCION EN LA LCD
;*****************************************************************************
********************

ERRASE:

        movlw   .0
        xorwf   CONT_DIS, W         ;revisa si el contador es 0
        btfsc   STATUS, Z
        call    ERRASE2             ;se va a ERRASE2
        movf    CONT_DIS, W
        movwf   CONT_DIS2           ;guarda contador de LCD en temporal
        movlw   .42
        subwf   CONT_DIS, W
        btfss   STATUS, C           ;si >= 42 ENTONCES SALTA
        goto    ERRASECONTINUE
        movlw   .24                 ;suma 24 para estar en 2a línea
        addwf   CONT_DIS2, F

ERRASECONTINUE

        bcf     PORTB, 3            ;pin RS = 0
        bcf     PORTB, 4            ;PIN E = 0
        DECF    CONT_DIS, F         ;PARA BORRAR ULTIMA LETRA
        DECF    CONT_DIS2, F        ;PARA BORRAR ULTIMA LETRA
        BSF     CONT_DIS2, 7        ;prende bit 7 para comando LCD
        MOVF    CONT_DIS2, W
        CALL    DISPLAYLCD          ;manda comando
        call    RETRASO
        BCF     CONT_DIS2, 7        ;apaga bit 7
        bsf     PORTB, 3            ;prende RS
```

```
        DECF   CONT_DIS, F          ;PARA BORRAR ULTIMA LETRA
        movlw  .32
        CALL   DISPLAYLCD           ;manda espacio blanco a LCD
        call   RETRASO
        bcf    PORTB, 3             ;pin RS = 0
        DECF   CONT_DIS, F          ;PARA BORRAR ULTIMA LETRA
        BSF    CONT_DIS2, 7
        MOVF   CONT_DIS2, W;regresa puntero de LCD
        CALL   DISPLAYLCD
        call   RETRASO
        BCF    CONT_DIS2, 7
        DECF   CONT_DIS, F          ;PARA BORRAR ULTIMA LETRA
        RETURN

ERRASE2

        movlw  .I
        movwf  CONT_DIS             ;inicializa contador LCD
        return
;*******************************************************************************
********************
;               RUTINA QUE BUSCA LA TECLA QUE SE PRESIONO
;*******************************************************************************
********************

RutinaTabla

        bcf    STATUS, RP0          ;asegurando estar en el
        bcf    STATUS, RP1          ;***banco 0***
        bcf    STATUS, C            ;borrando el carry
        movlw  HIGH Tabla           ;mueve a w los 5 bits superiores
        movwf  PCLATH               ;de la dirección de la tabla y los
        movlw  LOW Tabla            ;pasa al PCLATH
```

```
        addwf   KEYBD, w
        btfsc   STATUS, C           ;revisa carry, si 1 -> incremente
        incf    PCLATH, f           ;en uno el PCLATH, no -> lea la
        movwf   PCL                 ;tabla


;*********************************************************************************
*******************
;                       Tabla de Decodificación de Teclado
;*********************************************************************************
*******************

Tabla
        retlw   b'00000000'         ; 0
        retlw   ')'                 ; 1     )       F9
        retlw   b'00000000'         ; 2
        retlw   '%'                 ; 3     %       F5
        retlw   '#'                 ; 4     #       F3
        retlw   '!'                 ; 5     !       F1
        retlw   ""                  ; 6     "       F2
        retlw   b'01000000'         ; 7     @       F12
        retlw   b'00000000'         ; 8
        retlw   ';'                 ; 9             F10
        retlw   '('                 ; 10    (       F8
        retlw   '&'                 ; 11    &       F6
        retlw   '$'                 ; 12    $       F4
        retlw   b'01111110'         ; 13            TAB
        retlw   b'10110110'         ; 14     '
        retlw   b'00000000'         ; 15
        retlw   b'00000000'         ; 16
        retlw   b'11110101'         ; 17    dieresis L ALT
        retlw   '<'                 ; 18    <       L SHFT
        retlw   b'00000000'         ; 19
```

```
retlw   b'11110110'     ; 20    sumatoria L CTRL
retlw   'Q'             ; 21    Q
retlw   '1'             ; 22    1
retlw   b'00000000'     ; 23
retlw   b'00000000'     ; 24
retlw   b'00000000'     ; 25
retlw   'Z'             ; 26    Z
retlw   'S'             ; 27    S
retlw   'A'             ; 28    A
retlw   'W'             ; 29    W
retlw   '2'             ; 30    2
retlw   b'00000000'     ; 31
retlw   b'00000000'     ; 32
retlw   'C'             ; 33    C
retlw   'X'             ; 34    X
retlw   'D'             ; 35    D
retlw   'E'             ; 36    E
retlw   '4'             ; 37    4
retlw   '3'             ; 38    3
retlw   b'00000000'     ; 39
retlw   b'00000000'     ; 40
retlw   b'00100000'     ; 41    SPACE
retlw   'V'             ; 42    V
retlw   'F'             ; 43    F
retlw   'T'             ; 44    T
retlw   'R'             ; 45    R
retlw   '5'             ; 46    5
retlw   b'00000000'     ; 47
retlw   b'00000000'     ; 48
retlw   'N'             ; 49    N
retlw   'B'             ; 50    B
retlw   'H'             ; 51    H
retlw   'G'             ; 52    G
```

```
retlw   'Y'             ; 53    Y
retlw   '6'             ; 54    6
retlw   b'00000000'     ; 55
retlw   b'00000000'     ; 56
retlw   b'00000000'     ; 57
retlw   'M'             ; 58    M
retlw   'J'             ; 59    J
retlw   'U'             ; 60    U
retlw   '7'             ; 61    7
retlw   '8'             ; 62    8
retlw   b'11111000'     ; 63    sleep
retlw   b'00000000'     ; 64
retlw   ','             ; 65    ,
retlw   'K'             ; 66    K
retlw   'I'             ; 67    I
retlw   'O'             ; 68    O
retlw   '0'             ; 69    0
retlw   '9'             ; 70    9
retlw   b'00000000'     ; 71
retlw   b'00000000'     ; 72
retlw   '.'             ; 73    .
retlw   b'01011111'     ; 74    _
retlw   'L'             ; 75    L
retlw   b'11101110'     ; 76    ñ
retlw   'P'             ; 77    P
retlw   '?'             ; 78    ?
retlw   b'00000000'     ; 79
retlw   b'00000000'     ; 80
retlw   b'00000000'     ; 81
retlw   '''             ; 82    '
retlw   b'00000000'     ; 83
retlw   '['             ; 84    [
retlw   '='             ; 85    =
```

```
retlw   b'00000000'         ; 86
retlw   b'00000000'         ; 87
retlw   b'10110110'         ; 88    CAPS
retlw   b'10110110'         ; 89    R SHFT
retlw   b'10110110'         ; 90    ENTER
retlw   ']'                 ; 91    ]
retlw   b'00000000'         ; 92
retlw   b'01100000'         ; 93    \
retlw   b'00000000'         ; 94
retlw   b'00000000'         ; 95
retlw   b'00000000'         ; 96
retlw   b'00111110'         ; 97    >
retlw   b'00000000'         ; 98
retlw   b'00000000'         ; 99
retlw   b'00000000'         ; 100
retlw   b'00000000'         ; 101
retlw   b'00100000'         ; 102   BLANCO BKSP
retlw   b'00000000'         ; 103
retlw   b'00000000'         ; 104
retlw   '1'                 ; 105   KP 1
retlw   b'00000000'         ; 106
retlw   '4'                 ; 107   KP 4
retlw   '7'                 ; 108   KP 7
retlw   b'00000000'         ; 109
retlw   b'00000000'         ; 110
retlw   b'00000000'         ; 111
retlw   '0'                 ; 112   KP 0
retlw   '.'                 ; 113   KP .
retlw   '2'                 ; 114   KP 2
retlw   '5'                 ; 115   KP 5
retlw   '6'                 ; 116   KP 6
retlw   '8'                 ; 117   KP 8
retlw   b'11111100'         ; 118   ESC
```

```
retlw   b'10110110'     ; 119   NUM
retlw   ':'             ; 120   :       F11
retlw   '+'             ; 121   KP +
retlw   '3'             ; 122   KP 3
retlw   '-'             ; 123   KP -
retlw   '*'             ; 124   KP *
retlw   '9'             ; 125   KP 9
retlw   b'10110110'     ; 126   SCROLL
retlw   b'00000000'     ; 127
retlw   b'00000000'     ; 128
retlw   b'11110100'     ; 129   omega   R ALT
retlw   b'00000000'     ; 130
retlw   '^'             ; 131   ^       F7
retlw   'c'             ; 132   R       CTRL
retlw   b'00000000'     ; 133
retlw   b'00000000'     ; 134
retlw   b'00000000'     ; 135
retlw   b'00000000'     ; 136
retlw   b'00000000'     ; 137
retlw   b'00000000'     ; 138
retlw   b'00000000'     ; 139
retlw   b'00000000'     ; 140
retlw   b'00000000'     ; 141
retlw   b'00000000'     ; 142
retlw   'w'             ; 143   L WIN
retlw   b'00000000'     ; 144
retlw   b'00000000'     ; 145
retlw   b'00000000'     ; 146
retlw   b'00000000'     ; 147
retlw   b'00000000'     ; 148
retlw   b'00000000'     ; 149
retlw   b'00000000'     ; 150
retlw   'w'             ; 151   R WIN
```

```
retlw   b'00000000'         ; 152
retlw   b'00000000'         ; 153
retlw   b'00000000'         ; 154
retlw   b'00000000'         ; 155
retlw   b'00000000'         ; 156
retlw   b'00000000'         ; 157
retlw   b'00000000'         ; 158
retlw   'a'                 ; 159   APPS
retlw   b'00000000'         ; 160
retlw   b'00000000'         ; 161
retlw   b'00000000'         ; 162
retlw   b'00000000'         ; 163
retlw   b'00000000'         ; 164
retlw   b'00000000'         ; 165
retlw   b'00000000'         ; 166
retlw   b'00000000'         ; 167
retlw   b'00000000'         ; 168
retlw   b'00000000'         ; 169
retlw   b'00000000'         ; 170
retlw   b'00000000'         ; 171
retlw   b'00000000'         ; 172
retlw   b'00000000'         ; 173
retlw   b'00000000'         ; 174
retlw   b'00000000'         ; 175
retlw   b'00000000'         ; 176
retlw   b'00000000'         ; 177
retlw   b'00000000'         ; 178
retlw   b'00000000'         ; 179
retlw   b'00000000'         ; 180
retlw   b'00000000'         ; 181
retlw   b'00000000'         ; 182
retlw   b'00000000'         ; 183
retlw   b'00000000'         ; 184
```

```
retlw   b'00000000'         ; 185
retlw   '/'                 ; 186   KP /
retlw   b'00000000'         ; 187
retlw   b'00000000'         ; 188
retlw   b'00000000'         ; 189
retlw   b'00000000'         ; 190
retlw   b'00000000'         ; 191
retlw   b'00000000'         ; 192
retlw   b'00000000'         ; 193
retlw   b'00000000'         ; 194
retlw   b'00000000'         ; 195
retlw   b'00000000'         ; 196
retlw   b'00000000'         ; 197
retlw   b'00000000'         ; 198
retlw   b'00000000'         ; 199
retlw   b'00000000'         ; 200
retlw   b'00000000'         ; 201
retlw   b'01111111'         ; 202   KP EN
retlw   b'00000000'         ; 203
retlw   b'00000000'         ; 204
retlw   b'00000000'         ; 205
retlw   b'00000000'         ; 206
retlw   b'00000000'         ; 207
retlw   b'00000000'         ; 208
retlw   b'00000000'         ; 209
retlw   b'00000000'         ; 210
retlw   b'00000000'         ; 211
retlw   b'00000000'         ; 212
retlw   b'00000000'         ; 213
retlw   b'00000000'         ; 214
retlw   b'00000000'         ; 215
retlw   b'00000000'         ; 216
retlw   'F'                 ; 217   END
```

```
retlw   b'00000000'     ; 218
retlw   b'01111111'     ; 219   L ARROW
retlw   'I'             ; 220   HOME
retlw   b'00000000'     ; 221
retlw   b'00000000'     ; 222
retlw   b'00000000'     ; 223
retlw   'I'             ; 224   INSERT
retlw   'D'             ; 225   DELETE
retlw   b'01111100'     ; 226   D ARROW
retlw   b'00000000'     ; 227
retlw   b'01111110'     ; 228   R ARROW
retlw   b'01011110'     ; 229   U ARROW
retlw   b'00000000'     ; 230
retlw   b'00000000'     ; 231
retlw   b'00000000'     ; 232
retlw   b'00000000'     ; 233
retlw   'D'             ; 234   PG DOWN
retlw   b'00000000'     ; 235
retlw   b'00000000'     ; 236
retlw   'U'             ; 237   PG UP
retlw   b'00000000'     ; 238
retlw   b'00000000'     ; 239
retlw   b'00000000'     ; 240
retlw   b'00000000'     ; 241
retlw   b'00000000'     ; 242
retlw   b'00000000'     ; 243
retlw   b'00000000'     ; 244
retlw   b'00000000'     ; 245
retlw   b'00000000'     ; 246
retlw   b'00000000'     ; 247
retlw   b'00000000'     ; 248
retlw   b'00000000'     ; 249
retlw   b'00000000'     ; 250
```

```
        retlw   b'00000000'             ; 251
        retlw   b'00000000'             ; 252
        retlw   b'00000000'             ; 253
        retlw   b'00000000'             ; 254
        retlw   b'00000000'             ; 255
        retlw   b'00000000'             ; 239
;*****************************************************************************************
*******************
;*****************************************************************************************
*******************
        ORG 0x1800
        DA " 1.TX   2.RX CON RING   3.RX SIN RING "     ;UNO Y SEIS (RING)
        DA "LINEA "                                     ;DOS
        DA "OCUPADO "                                   ;TRES
        DA "CONECTANDO"                                 ;CUATRO
        DA "ERROR EN CONEXION "                         ;CINCO
        DA "ESPERANDO LLAMADA "                                 ;SIETE
        end
```

B. Código para teclado *PS2*, set 2

Tabla 7. Código (*Scan Code Set 2*) para teclado de 101, 102 y 104 teclas

KEY	*MAKE*	*BREAK*	-----	*KEY*	*MAKE*	*BREAK*	-----	*KEY*	*MAKE*	*BREAK*
A	1C	F0,1C		9	46	F0,46		[	54	FO,54
B	32	F0,32		`	0E	F0,0E		*INSERT*	E0,70	E0,F0,70
C	21	F0,21		-	4E	F0,4E		*HOME*	E0,6C	E0,F0,6C
D	23	F0,23		=	55	FO,55		*PG UP*	E0,7D	E0,F0,7D
E	24	F0,24		\	5D	F0,5D		*DELETE*	E0,71	E0,F0,71
F	2B	F0,2B		*BKSP*	66	F0,66		*END*	E0,69	E0,F0,69
G	34	F0,34		*SPACE*	29	F0,29		*PG DN*	E0,7A	E0,F0,7A
H	33	F0,33		*TAB*	0D	F0,0D		*U ARROW*	E0,75	E0,F0,75
I	43	F0,43		*CAPS*	58	F0,58		*L ARROW*	E0,6B	E0,F0,6B
J	3B	F0,3B		*L SHFT*	12	FO,12		*D ARROW*	E0,72	E0,F0,72
K	42	F0,42		*L CTRL*	14	FO,14		*R ARROW*	E0,74	E0,F0,74
L	4B	F0,4B		*L GUI*	E0,1F	E0,F0,1F		*NUM*	77	F0,77
M	3A	F0,3A		*L ALT*	11	F0,11		KP /	E0,4A	E0,F0,4A
N	31	F0,31		*R SHFT*	59	F0,59		KP *	7C	F0,7C
O	44	F0,44		*R CTRL*	E0,14	E0,F0,14		KP -	7B	F0,7B
P	4D	F0,4D		*R GUI*	E0,27	E0,F0,27		KP +	79	F0,79
Q	15	F0,15		*R ALT*	E0,11	E0,F0,11		KP EN	E0,5A	E0,F0,5A
R	2D	F0,2D		*APPS*	E0,2F	E0,F0,2F		KP .	71	F0,71
S	1B	F0,1B		*ENTER*	5A	F0,5A		KP 0	70	F0,70
T	2C	F0,2C		ESC	76	F0,76		KP 1	69	F0,69
U	3C	F0,3C		F1	05	F0,05		KP 2	72	F0,72
V	2A	F0,2A		F2	06	F0,06		KP 3	7A	F0,7A
W	1D	F0,1D		F3	04	F0,04		KP 4	6B	F0,6B

X	22	F0,22		F4	0C	F0,0C		KP 5	73	F0,73
Y	35	F0,35		F5	03	F0,03		KP 6	74	F0,74
Z	1A	F0,1A		F6	0B	F0,0B		KP 7	6C	F0,6C
0	45	F0,45		F7	83	F0,83		KP 8	75	F0,75
1	16	F0,16		F8	0A	F0,0A		KP 9	7D	F0,7D
2	1E	F0,1E		F9	01	F0,01		]	5B	F0,5B
3	26	F0,26		F10	09	F0,09		;	4C	F0,4C
4	25	F0,25		F11	78	F0,78		'	52	F0,52
5	2E	F0,2E		F12	07	F0,07		,	41	F0,41
6	36	F0,36		*PRNT SCRN*	E0,12, E0,7C	E0,F0, 7C,E0, F0,12		.	49	F0,49
7	3D	F0,3D		*SCROLL*	7E	F0,7E		/	4A	F0,4A
8	3E	F0,3E		*PAUSE*	E1,14,77, E1,F0,14, F0,77	*-NONE-*				

* Todos los valores están en hexadecimal

C. Encuesta a personas no oyentes que probaron los dos teléfonos de texto por una semana.

NOMBRE: Paula Deleón de Rodríguez

SEXO: Femenino EDAD: 25

No. DE CÉDULA: A-1 994814

DIRECCIÓN: 5ª Calle 35-13 "A" zona 11 Utatlán II.

No. DE TELÉFONO: 434-7169

ENCUESTA: FUNCIONAMIENTO DE UN TELEFONO DE TEXTO ENTRE DOS PERSONAS NO OYENTES

1. ¿Logra la detección de una llamada entrante (del ring) correctamente?
   siempre X casi siempre ___ pocas veces ___ nunca ___

2. ¿El teléfono de texto detecta si da línea o está ocupado?
   siempre X casi siempre ___ pocas veces ___ nunca ___

3. ¿La velocidad de escritura presentó alguna limitación en la transmisión?
   siempre ___ casi siempre ___ pocas veces ___ nunca X

4. ¿Reconoce el momento en que se establece la conexión entre los dos teléfonos de texto?
   siempre X casi siempre ___ pocas veces ___ nunca ___

5. ¿Presenta errores al escribir o recibir texto?
   siempre ___ casi siempre ___ pocas veces ___ nunca X

6. ¿Se logró una comunicación inasistida?
   siempre X casi siempre ___ pocas veces ___ nunca ___

7. ¿Es sencillo de utilizar?
   si X no___

NOMBRE: GERARDO MEZA FOIGAR

SEXO: MASCULINO EDAD: 24 AÑOS

No. DE CÉDULA: A-1119387

DIRECCIÓN: 1ra calle 2-26 zona 2 de Naco cal Sta Rita I

No. DE TELÉFONO: 4313242 / Cel 3148203

ENCUESTA: FUNCIONAMIENTO DE UN TELEFONO DE TEXTO ENTRE DOS PERSONAS NO OYENTES

1. ¿Logra la detección de una llamada entrante (del ring) correctamente?
   siempre X casi siempre ___ pocas veces ___ nunca ___

2. ¿El teléfono de texto detecta si da línea o está ocupado?
   siempre X casi siempre ___ pocas veces ___ nunca ___

3. ¿La velocidad de escritura presentó alguna limitación en la transmisión?
   siempre ___ casi siempre ___ pocas veces ___ nunca X

4. ¿Reconoce el momento en que se establece la conexión entre los dos teléfonos de texto?
   siempre X casi siempre ___ pocas veces ___ nunca ___

5. ¿Presenta errores al escribir o recibir texto?
   siempre ___ casi siempre ___ pocas veces ___ nunca X

6. ¿Se logró una comunicación inasistida?
   siempre X casi siempre ___ pocas veces ___ nunca ___

7. ¿Es sencillo de utilizar?
   si X no ___

Imprenta "GORA"
25 Av. 25-71, Zona 5
Telefax: 2335-5733 - 5218-7292