# Instituto Tecnológico de Tuxtla Gutiérrez

**Proyecto:**
**Implementación de telealarmas para control y señalización por medio de un Modem GSM**

**Residencia Profesional**

**Presenta**
**Raúl Moreno Camacho**

**Asesor**
**Ing. Leonel Torres Miranda**

**Asesor Externo**
**Ing. Sergio Pérez Gordillo**

**Tuxtla Gutiérrez, Chiapas**
**Diciembre / 2008**

# ÍNDICE

## 1. NOMBRE DEL PROYECTO

Implementación de telealarmas para control y señalización por medio de un MODEM GSM

## 2. INTRODUCCIÓN

Los actuales sistemas de alarma no sólo cubren cada vez mayor tipo de incidencias, sino que permiten disfrutar de un sistema a la medida para cada necesidad de la oficina, industria o del hogar. Pueden ser aplicables a eventos como robo, incendio, etc.

Un sistema de alarma digital permite que se le integren diversos tipos de componentes como: detector de ruptura de cristal, detector de movimiento, sirena, luz estrobo, contactos magnéticos para puertas, ventanas, foto celdas (protección exterior).

Las telecomunicaciones aunadas a los grandes avances en la tecnología han logrado sistemas de alarmas cada vez más capaces de realizar acciones para la prevención y vigilancia de las distintas necesidades.

El uso de la telefonía celular en estos días es tan común que cada vez es mas frecuente ver distintos softwares desarrollados para localización, control y monitoreo.

## 3. JUSTIFICACIÓN

Hoy en día la sociedad cada vez mas envuelta en un mundo tecnológico se ve en la necesidad de mantener un control y monitoreo constante de los distintos sucesos que se pueden manifestar en sus áreas de trabajo u hogares, esto con la finalidad de evitar eventos no deseados.

Los beneficios que brinda la implementación del proyecto dependerán en gran medida al área en donde se encuentren instalados, entre los más destacados se encuentran: la capacidad de poder recibir información sobre el estado del lugar o dispositivo a monitorear, que puede ir desde el monitoreo de casas u oficinas hasta el de automóviles; así mismo otro beneficio importante consiste en el poder realizar acciones que impidan algún evento no deseado vía telefonía celular, estos beneficios dependerán en gran medida de la manera en que se implemente el proyecto presentado.

Desde un punto de vista más teórico se puede decir que el proyecto aporta un mayor entendimiento hacia el uso de redes celulares, la manera en que estas trabajan y el uso que se le puede dar con base a lo que se necesite monitorear.

En cuanto a los alcances de este proyecto se puede mencionar que apoyará mas el uso del monitoreo vía celular; con lo que muchas empresas posteriormente podrían optar por la implementación de telealarmas como un refuerzo para evitar cualquier tipo de acción no

deseada o simplemente para mantenerse informado sobre lo que sucede en distintas áreas o aparatos.

Por último, profesionalmente pondrá en manifiesto los conocimientos adquiridos durante la carrera y permitirá sentar las bases para otros estudios que surjan partiendo de la problemática aquí especificada.

# 4. OBJETIVO

## 4.1 GENERAL:

Implementar un sistema que permita controlar un dispositivo a distancia (Sony Ericsson GT48), el cuál será programado para enviar distintas señales de alerta a un dispositivo móvil (teléfono celular), con la posibilidad de recibir la respuesta y ejecutar una acción previamente programada.

## 4.2 ESPECÍFICO:

- Conocer los alcances de la telefonía celular
- Manipulación de la Terminal MODEM GSM GT48 para distintos propósitos
- Lograr el envío y recepción de información vía mensajes de texto de teléfonos celulares.
- Captar información del medio que rodea al MODEM GSM GT48, usando para ello sensores, switches o cualquier dispositivo digital simple que brinde información sobre su entorno.
- Activar/Desactivar relevadores vía mensaje de texto.

# 5. ÁREA DE PARTICIPACIÓN

Si bien el proyecto ha sido realizado en las instalaciones de la GERENCIA REGIONAL DE TRANSMISION SURESTE pertenecientes a la COMISION FEDERAL DE ELECTRICIDAD, el proyecto ha sido pensado para distintas áreas de utilización.

Dentro de esta área el proyecto puede ser de utilidad por permitir el monitoreo de variables importantes para distintos tipos de departamento, como por ejemplo el departamento de protección del área mencionada con anterioridad; aquí la implementación de la telealarma puede proporcionar un monitoreo constante sobre las distintos lugares de generación de energía eléctrica, al presentarse alguna anomalía esta puede ser informada directamente a los responsables vía celular.

# 6. ALCANCES Y LIMITACIONES

ALCANCES DEL PROYECTO:
El proyecto presenta las siguientes caracteristicas, asímismo se muestran las actividades a realizar con el fin de que el cumplimiento de los objetivos pueda ser logrado:

Se realizaran las pruebas necesarias usando distintos softwares de computación con la finalidad de lograr la recepcion de información digital y la activación de dispositivos a manipular, como son relevadores.

Se hará posible la recepción y envío de información vía mensajes de texto, usando para ello el software de computación proporcionado por la empresa fabricante del dispositivo y realizando un estudio previo de las funciones proporcionadas por el dispositivo y las permitidas por el software.

Se constituirá un algoritmo de programación para el funcionamiento del dispositivo en cualquier ubicación evitando la necesidad de manipular del dispositivo mediante un ordenador, es decir, lograr que las funciones antes vistas mediante el uso de la computadora puedan darse sin usar esta misma.

Se logrará la conexión del MODEM GSM GT48 con una interfaz de entrada y salida necesaria para la recepción de información por medio de sensores y la activación de los respectivos relevadores y evitar de esta manera daño alguno en el dispositivo principal (GT48); para ello se hará uso de la teoría electrónica adquirida en anteriormente (optoelectrónica).

Finalmente se logrará la unión de los puntos anteriores, para obtener como resultado un dispositivo que realizará la función de alarma y ejecutará las acciones que sean requeridas de acuerdo a lo ordenado por el usuario, todo esto vía telefonía celular.

LIMITACIONES DEL PROYECTO:
Se ha mencionado anteriormente algunos beneficios producidos por la posible implementación del proyecto planteado; es decir, si bien se recomienda el uso de la red de telefonía celular para aplicaciones de monitoreo y ejecución de acciones, también es necesario recordar que hoy en día las fallas en este tipo de redes son comunes y como consecuencia podría traducirse en una perdida de información, en caso de presentarse el caso mencionado.

Otro aspecto que vale la pena resaltar es la falta de información que existe sobre el dispositivo usado (Sony Ericsson GT48), lo cual hace difícil determinar la manera más óptima para usar todas las funciones requeridas del dispositivo.

# 7. MARCO TEÓRICO

## 7.1 DESCRIPCIÓN SONY ERICSSON GT48

El Sistema Global para las Comunicaciones Móviles (GSM) es un sistema estándar para comunicación utilizando teléfonos móviles que incorporan tecnología digital. Por ser digital cualquier cliente de GSM puede conectarse a través de su teléfono con su ordenador y puede hacer, enviar y recibir mensajes por e-mail, faxes, navegar por Internet, acceso seguro a la red informática de una compañía (LAN/Intranet), así como utilizar otras funciones digitales de transmisión de datos, incluyendo el Servicio de Mensajes Cortos (SMS).

El Sony Ericsson GT47/GT48 es una terminal de control GSM/GPRS inteligente que encapsula todo lo que se necesita para la comunicación wireless M2M con gran capacidad en una unidad compacta. En conjunción con el paquete de Sony Ericsson M2M, el GT47/GT48 puede hospedar y controlar aplicaciones inalámbricas, minimizando la necesidad de componentes extras. Alternativamente, puede ser usado como un poderoso modems independiente GPRS con su propio TCP/IP stack.

El GT47/48 contiene su propia terminal con su lector de tarjeta SIM y una interfaz estándar.

El GT47/GT48 puede ser usado para proveer una comunicación con una variedad de aplicaciones inalámbricas como son el monitoreo, de alarmas seguridad, mantenimiento electrónico y otras aplicaciones en telemetría.

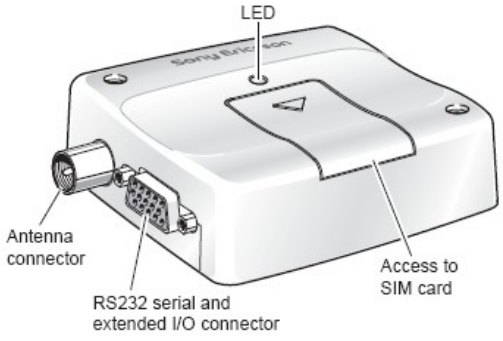



Figura 1- Vista del dispositivo por la cara izquierda

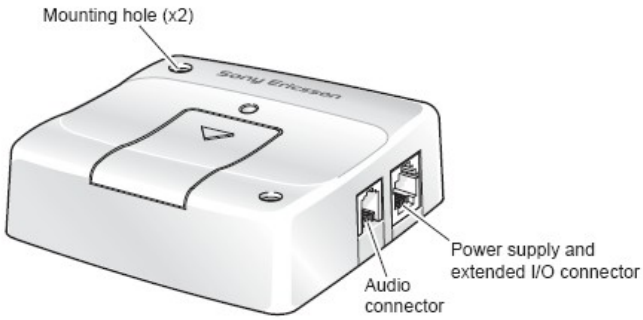



Figura 2- Vista del dispositivo de lado derecho.

El GT48 cuenta con 5 conectores, en los cuales se manejan distintas señales:

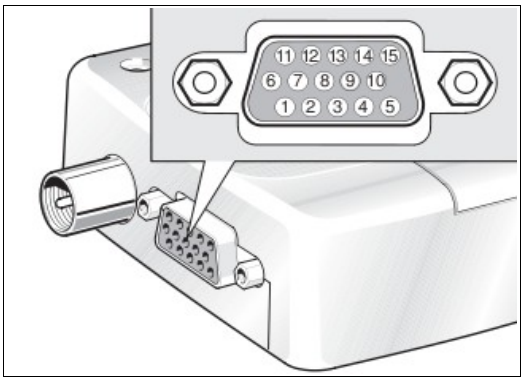
1 – conector DB15 (interfase RS232 y extensión de salidas y entradas)
2 – RJ12 (Fuente de energía y manejo de entradas y salidas)
3 – RJ9 (Conector de audio)
4 – Lector SIM
5 – Conector de antena

De los anteriores mencionados solamente el RJ9 no fue usado para la realización del proyecto, debido a que no es necesario el manejo de audio. Para los siguientes conectores se muestra a continuación una breve descripción técnica y se explicará el usó dado a cada uno de ellos.

**DB15 (Interfase Serial RS232 y extensión de entradas/salidas)**

La distribución de los pines dentro del conector db15 se presenta de la siguiente manera:

1 – DCD*
2 – RD*
3 – TD*
4 – IN 3
5 – OUT 3
6 – DSR*
7 – RTS*
8 – CTS*
9 – 4.8 V
10 - RI
11 – IN 2
12 – OUT 4
13 - DTR*
14 - GND*
15 - ANA_IN



Nota: los marcados (*) son usados para conformar la interfaz serial principal RS232.

Este puerto el RS232, existente en todos los ordenadores actualmente es el sistema mas común para la transmisión de datos entre ordenadores. Todos los ordenadores como mínimo poseen uno (módem, ratón,...).

El RS232 es un estándar de comunicaciones propuesto por la Asociación de Industrias Electrónicas (EIA) y es la última de varias versiones anteriores. Antiguamente se utilizaba para conectar terminales a un ordenador Host. Se envían datos de 7, 8 o 9 bits. La velocidad se mide en baudios (bits/segundo) y sólo son necesarios dos cables, uno de transmisión y otro de recepción.

Lo mas importante del estándar de comunicaciones es la funciones especifica de cada pin de entrada y salida de datos porque nos encontramos básicamente con dos tipos de conectores los de 25 pines y los de 9 pines, es probable que se encuentre mas la versión de 9 pines aunque la versión de 25 permite muchas mas información en la transferencia de datos.

Las señales con la que actúa el puerto son digitales (0 - 1) y la tensión a la que trabaja es de 12 Voltios, resumiendo:

12 Volts = Lógica “0”
-12 Volts = Lógica “1”

Las características de los pines y su nombre típico son:

| | | |
|---|---|---|
| TD | Transmitir Datos | Señal de salida |
| RD | Recibir Datos | Señal de entrada |
| RTS | Solicitud de envió | Señal de salida |
| DTR | Terminal de datos listo | Señal de salida |
| CTS | Libre para envió | Señal de entrada |
| DSR | Equipo de datos listo | Señal de entrada |
| DCD | Detección de portadora | Señal de entrada |
| GND | Tierra | Referencia para señales |
| RI | Indicador de llamada | Señal de entrada |

Los pines que portan los datos son RD y TD los demás se encargan de otros trabajos, el DTR indica que el ordenador esta encendido, DSR que el dispositivo conectado al puerto esta encendido, RTS que el ordenador al no estar ocupado puede recibir datos, al revés de CTS que lo que informa es que es el dispositivo el que puede recibir datos, DCD detecta que existen presencia de datos, etc.

Antes de iniciar cualquier comunicación con el puerto RS232 se debe de determinar el protocolo a seguir dado que el estándar del protocolo no permite indicar en que modo se

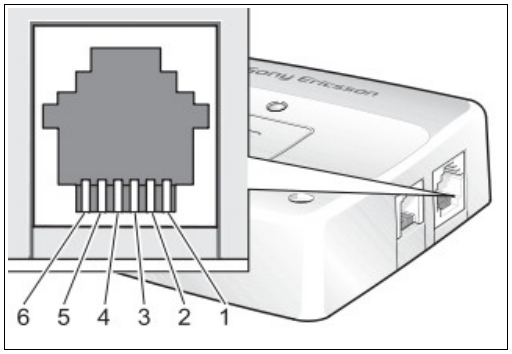
esta trabajando, es la persona que utiliza el protocolo el que debe decidir y configurar ambas partes antes de iniciar la transmisión de datos.
Siendo los parámetros a configurar los siguientes:

- Protocolo serie (numero bits-paridad-bits stop)
- Velocidad de puerto
- Protocolo de control de flujo (RTS/CTS o XON/XOFF).

Como se menciona antes en este conector se dispone de 2 salidas digitales (OUT 3 y OUT 4) y 2 entradas del mismo tipo (IN2 e IN3), se puede accesar y manipularlas usando los comandos AT correspondientes o las funciones intrínsecas proporcionadas para la programación de scripts mediante el software M2mpower IDE.

**RJ12 (Fuente de Energía)**

1 – $V_{IN}$
2 – OUT 2
3 – IN 1
4 – TO_IN
5 – OUT 1
6 - GND

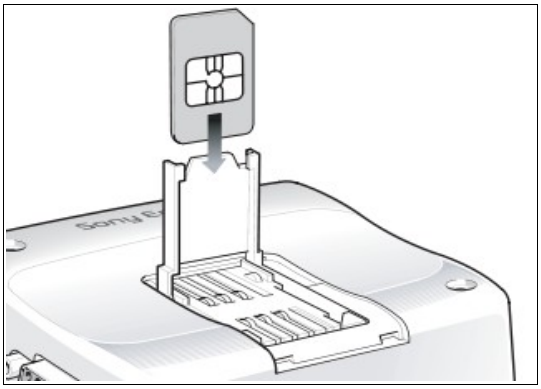



Para los fines del proyecto se excluye el uso de los pines 2, 3, 4 y 5; solamente se hace uso de este conector para proporcionar la energía necesaria para su funcionamiento mediante los pines 1 y 6 (Voltaje y Tierra respectivamente).

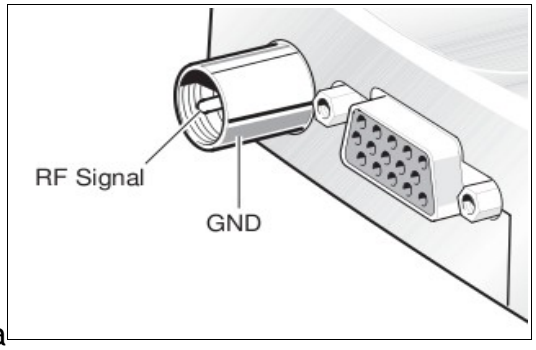
**Lector de tarjetas SIM**

El modulo cuenta con un lector de tarjetas SIM, con el fin de poder manejar la terminal como un teléfono celular, esto elimina la necesidad de contratar la renta de un número fijo.

La terminal al igual que los teléfonos celulares cuenta con un detector de presencia de tarjetas SIM para asegurar el óptimo funcionamiento del dispositivo; cabe mencionar que muchas de las funciones del GT48 no podrán ser utilizadas si se opera la terminal sin tarjeta SIM.



**Conector de Antena**




Se cuenta con un conector de antena tipo hembra el cual permite al dispositivo estar al alcance de la red GSM y disponer de servicios como mensajes de texto, llamadas de datos, etc.

Para más información los datos técnicos de cada uno de los anteriores conectores son detallados en la hoja de datos técnicos del GT47/48.

## 7.2 SOFTWARE

***M2mpower IDE***

El C es un lenguaje de propósito general lenguaje de programación conocido por su eficiencia, economía, y la portabilidad.

Si bien esas características hacen que una buena opción para casi cualquier tipo de programación, C ha demostrado ser especialmente útil en los sistemas de programación, ya que facilita la escritura rápida, compacta programas que sean fácilmente adaptables a otros sistemas. Programas en C son a menudo tan rápido como el montaje de programas de lengua, y son normalmente más fácil para los programadores a leer y mantener.

El intérprete M2mpower hace uso exclusivo del lenguaje C como lenguaje de programación. El intérprete C M2mpower utiliza un conjunto de funciones generales y específicas incorporadas en bibliotecas intrínsecas. El intérprete M2mpower usa el conjunto ANSI C para su gramática.

***Hyperterminal***

Es un programa que se puede utilizar para conectar con otros equipos, sitios Telnet, sistemas de boletines electrónicos (BBS), servicios en línea y equipos host, mediante un módem, un cable de módem nulo o Ethernet.

Aunque utilizar HyperTerminal con un servicio de boletín electrónico para tener acceso a información de equipos remotos es una práctica que está dejando de ser habitual gracias al World Wide Web, HyperTerminal sigue siendo un medio útil para configurar y probar el módem o examinar la conexión con otros sitios.

HyperTerminal graba los mensajes enviados o recibidos por servicios o equipos situados al otro extremo de la conexión. Por esta razón, puede actuar como una valiosa herramienta para solucionar problemas de configuración y uso del módem. Para confirmar que el módem está bien conectado o ver su configuración, puede enviar comandos a través de HyperTerminal y ver los resultados.

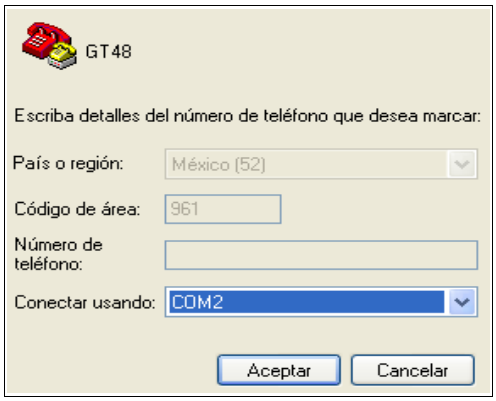
# 8. DESARROLLO DE ACTIVIDADES

## 8.1 PRUEBAS PRELIMINARES

Como primer paso para poder entender la manera en que funciona el dispositivo, se realizaron pruebas con comandos AT para comprobar algunas características que serán usadas posteriormente.
Para dichas pruebas se hizo uso de la aplicación de Microsoft Windows llamada ”HyperTerminal” la cual es usada con el fin de conectar el ordenador a otros sistemas remotos.

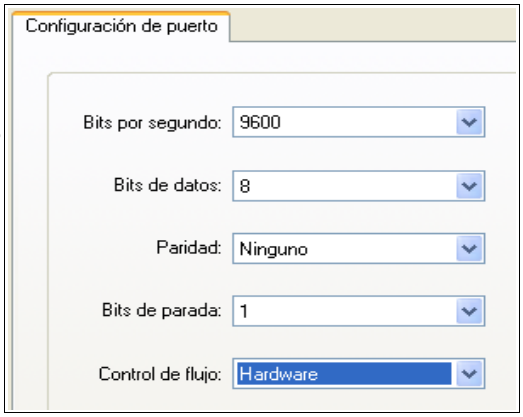
Para poder acceder a la terminal GT48 se deben tener en cuenta algunas especificaciones proporcionadas en los datos técnicos del dispositivo, así como también se debe realizar la configuración de la aplicación, los pasos para lograr esa configuración se muestran a continuación:

1 – Se requiere inicialmente de la identificación que se le dará a dicha conexión entre el ordenador y el GT48, así como un icono que denote el tipo de conexión, todo esto sirve únicamente para que podamos identificar la conexión; no influye de ningún modo con la transmisión de datos.


2 – Se escoge el puerto por el cual se está accesando a la terminal, en este caso se ha realizado usando el puerto com.

3 – Por último se accesa a un menú donde se configura los bits por segundo, bits de datos, paridad, bits de parada y el control de flujo; dichos datos son proporcionados en la hoja de datos del GT48, se presentan a continuación:




bits/seg = 9600
Bits de datos = 8
Paridad = Ninguna
Bits de parada = 1
Control de flujo = Hardware

Con lo anterior se logra exitosamente la comunicación entre el GT48 y el ordenador, accesando de esa manera a la aplicación principal, un indicador de que la comunicación se realizó con éxito, es el hecho de poder comenzar a escribir un comando AT dentro de la aplicación.

El GT48 internamente consta de un dispositivo de ingeniería en el cual se tienen todas las aplicaciones GSM/GPRS, este es denominado como GR48, para acceder a distintas funciones del GR48 se usa una extensa lista de comandos AT's; a continuación solo se presentan los usados en este proyecto, y el motivo de su uso.

| | |
|---|---|
| AT | Comando de atención |
| AT+CPIN | Usado para introducir el código PIN del SIM en caso de ser necesitado para el desbloqueo del mismo.<br>AT+CPIN=”*CODIGO*” |
| ATD | Usado para realizar llamadas de datos desde la terminal GT48 hacia cualquier número.<br>ATD*número_telefónico* |
| AT*EMAR | Es el master reset del ericsson con el cual todos lo datos del usuario son descartados y reiniciados con los datos por default.<br>AT*EMAR=”código de desbloqueo” |
| AT*E2RESET | Reinicia la terminal; no son necesarios parámetros adicionales. |
| AT+CFUN | Comando que permite apagar o poner en completo funcionamiento a la terminal.<br>AT+CFUN=*CODIGO (CODIGO = 0 para apagar, 1 para completa funcionalidad).* |

| | |
|---|---|
| AT+CGSMS | Selecciona el servicio para mensajes SMS.<br>AT+CGSMS=*CODIGO (código = 2 GPRS, 3 Circuit switched)* |
| AT*E2IO | Mediante este comando se tiene acceso a las entradas y salidas de la terminal; dependiendo de los parámetros usados se tiene el control de las entradas/salidas; en apartados posteriores se especificará detalladamente los códigos usados para el control de estos pines. |
| ATZ | Es un reset que permite regresar a la configuración por default de la terminal |
| AT+CPMS | Comando usado para definir los espacios de memoria usados en donde se leerá, escribirá, etc., se cuentan con tres espacios que el usuario puede redefinir en cualquier momento <mem1>,<mem2>,<mem3>, donde cada uno de estos espacios es destinado para diferentes acciones:<br>*Mem1:* espacio de memoria donde los mensajes son leídos y borrados.<br>*Mem2:* espacio en el cual se escriben y se hacen operaciones<br>*Mem3:* espacio donde se reciben SMS's.<br>Para cada espacio hay dos opciones: ME (almacenamiento de la terminal) y SM (almacenamiento del SIM<br><br>AT+CPMS = ”mem1(ME/SM)”,”mem2(ME/SM)”, ”mem3(ME/SM)”. |
| AT+CMGF | Define el formato de mensajes; ya sea en modo PDU(0) o modo de texto(1):<br>AT+CMGF=0/1 |
| AT+CMGS | Comando usado para el envío de SMS's:<br>AT+CMGS=”*Número de celular*”<br>> *Texto a enviar* **CTRL+Z** (finaliza el texto y envía el mensaje) |
| AT+CNMI | Mediante este comando se puede configurar la manera en que se indica un nuevo mensaje a la terminal;5 parámetros deberán ser definidos; para este proyecto solamente fue necesario la modificación de los primeros 2. para mayor información checar el manual de comandos AT's del GT48. |

| | |
|---|---|
| AT+CMGR | Comando usado para leer un mensaje dentro de la memoria, para eso se debe especificar el numero entero que indica en que espacio esta guardado el mensaje.<br>AT+CMGR=espacio (1,2,3...40 para ME) |
| AT+CMGL | Permite desplegar una lista de los mensajes para leer guardados en mem1; las posibles listas a desplegar pueden ser las siguientes:<br>Mensajes sin leer – REC UNREAD<br>Mensajes leídos – REC READ<br>Mensajes no enviados – STO UNSENT<br>Mensajes enviados – STO SENT<br>Todos los mensajes – ALL<br><br>AT+CMGL=”lista a desplegar” |
| AT+CMGD | Borra el mensaje del espacio indicado por un número entero:<br>AT+CMGD=espacio (1,2,3...40) |
| AT+CSDH | Se muestran si se cumplen los parámetros del modo de texto |
| AT+CSMP | Coloca los parámetros del modo de texto |

Para más información de cada uno de los comandos mostrados en la tabla anterior hacer uso del manual de comandos At's proporcionados con el dispositivo.

Una vez conocidos los comandos usados en las pruebas, se procederá a mostrar paso a paso de que manera fueron usados y que se pretendía conseguir con cada una de las pruebas realizadas usando la Hyperterminal.

**Prueba No. 1 – Realización de llamadas telefónicas**

ATD0449611240080 *Se realiza una llamada de datos hacia el numero 0449611240080*

Se logró comprobar que el dispositivo no necesita de configuración extra para realizar llamadas telefónicas.

**Prueba No. 2 – Envío de Mensaje SMS**

AT+CSMP=17,167 Configuración de los parámetros del modo de texto

| | |
|---|---|
| OK | Respuesta del GT48 |
| AT+CMGF=1 | Configuración de mensajes en modo de texto |
| OK | Respuesta del GT48 |
| AT+CMGS=”9611240080” | Envío de un mensaje al número indicado |
| >Texto prueba no. 2 **ctrl+z** | texto a enviar, seguido de la combinación de teclas |
| ctrl+z para | indicar el final del texto y el envío del |
| mensaje SMS. | |
| OK | Respuesta del GT48 |
| | |
| +CMGS: 64 | Indicación de que el mensaje ha sido enviado. |

Se logró exitosamente el envío de mensajes SMS; el manejo del comando no es complejo, aunque se debe tener en cuenta que lo parámetros con los que fueron configurados los primeros comandos se obtuvieron al analizar la hoja de especificaciones para comandos AT's del GT48.

**Prueba No. 3 – Lectura de Mensajes SMS**

AT+CNMI=3,1,0,0,0
OK
AT+CMGF=1
OK
AT+CMGR=2
+CMGR: “REC UNREAD”,“+447747008670”,“Matt L”,“02/11/19,09:57:28+00”,145,36,0,0,
“+447785016005”,145,8
Prueba No. 3
OK

Al recibir un mensaje, en la aplicación de hyperterminal se recibirá un aviso el cual dirá que un mensaje ha sido recibido y proporcionará el numero entero de espacio donde se encuentra almacenado el mensaje para poder ser leído, ese número será usado con el comando CMGR para localizarlo y desplegarlo en pantalla; es importante recordar que para desplegar mensajes en modo de texto se debe configurar CMGF=1.

AT+CMGF=1
OK
AT+CMGL=”ALL”
+CMGL: 1,“REC READ”,“+447747008670”,“Matt L”,“02/10/21,10:07:23+04”,145,4
Test
+CMGL: 2,“REC READ”,“+447747008670”,“Matt L”,“02/11/19,09:57:28+00”,145,8
Test sms
+CMGL: 3,“REC UNREAD”,“+447747008670”,“Matt L”,“02/11/19,09:58:06+00”,145,8
Test sms

OK

En el anterior ejemplo se muestra como realizar la lectura de todos los mensajes almacenados en memoria.

**Prueba No. 4 – Control de Entradas y Salidas**

Para la manipulación de las entradas y salidas solo es necesario el comando AT*E2IO, este tiene la característica de manejar múltiples opciones para conocer el estado de la entrada/ salida.

Para manejar el comando se deben conocer dos parámetros que son: operación a realizar y señal a manipular.

Dentro de las operaciones realizables por el comando encontramos definidas por números enteros las siguientes:
0 – Lectura de la señal
1 – Colocación de un valor en la salida de un pin, valores de 0 ó 1 lógicos para salidas digitales y de 0 a 255 para analógicas.
2 – Configura el pin deseado, ya sea como entrada o salida.
3 – Muestra la configuración actual del pin especificado.
4 – Permite el aviso en pantalla cuando se realice un cambio del valor de la señal en una entrada
5 – Checa si el pin se encuentra configurado para mostrar cuando cambie su valor.
6 – Algunas líneas se encuentran multiplexadas y necesitan ser “switched in” para su uso.
7 – Muestra el estado del pin que se desee manipular, los valores de respuesta del GT48 se explican a continuación:
0 – Pin no avalado, es necesario usar el parámetro 6 para hacer uso de dicha línea
1 – Pin en uso por aplicaciones internas o no permitido para manipular.
2 – pin en uso por la interfase serial RS232, puede ser manejada usando otro comando
3 – Pin avalado para usar.

Ejemplos:

Lectura:
AT*E2IO=0,”IO3”
*E2IO=0,”IO3”,0 el ultimo valor muestra el valor de la señal en el pin IO3, los primeros dos parámetros muestran lo enviado hacia la terminal.

Pasos para la escritura en una salida:

AT*E2IO=7,"IO3" 1 – Verificar si es posible usar el pin
*E2IO=7,"IO3",3 el ultimo numero indica que es posible el uso del IO3
OK
AT*E2IO= 3,"IO3" 2 – Checar la configuración del pin (entrada o salida)
*E2IO=3,"IO3",0 para este ejemplo el pin se encuentra como entrada (0)
OK
AT*E2IO=2,"IO3",1 3 – Se cambia la configuración para manejarlo como salida.
OK
AT*E2IO=1,"IO3",1 4 – Se envía el valor lógico de la salida
OK

Si bien el ultimo comando es el que realiza la escritura es sugerible que se sigan los anteriores pasos para conocer el estado del pin y no accidentalmente tratar de manipular lineas que pueden causar daño alguno el cambio en su configuración.

Nota: el paso 3 puede ser omitido en caso de obtener como respuesta del paso 2 un estado deseado.

Es importante recordar que para poder hacer uso de una línea se debe conocer el nombre de dicha señal dentro del GR47, es decir no basta con conocer el nombre de la señal en el GT48, sino también hay que realizar una relación entre las señales del GT48 y GR48; para este caso se hará uso únicamente de 4 señales colocadas en el conector DB15; dichas señales son:

| Pin del GT48 | Señal en el GR48 |
|---|---|
| IN 2 | IO4 |
| IN 3 | IO7 |
| OUT 3 | IO8 |
| OUT 4 | IO3 |

Los nombres de las señales, estas serán usadas para acceder a dichas líneas y poder manipularlas.

Una vez conocidas las pruebas se pudo comprobar que la terminal realizaba satisfactoriamente todas las funciones necesitadas para el desarrollo del proyecto, por lo que se procedió a conocer el software proporcionado por Sony Ericsson para la programación del script final.

## 8.2 PROGRAMACIÓN USANDO M2MPOWER IDE

Este software esta basado en lenguaje de programación C, de tal modo que el entendimiento de la programación es bastante sencilla; es permitido el uso de recursos básicos que se tendrían en cualquier tipo de software desarrollado para programar en este

lenguaje; es decir, se puede hacer uso de:

- bucles (for, while, do while)
- operaciones aritméticas simples
- operaciones lógicas
- condicionantes (if, else, return)

Hasta este punto la programación no es distinta a otros softwares; pero una característica esencial de este paquete es que provee de funciones propias de la terminal, que permiten hacer uso de ciertas características que también podrían ser activadas usando comandos AT's. El software cuenta con aproximadamente 100 funciones que permiten activas diversas características del GT48 sin la necesidad de usar directamente los comandos AT's.

De todas las funciones proporcionadas, se hizo uso de las siguientes:

| Función | Descripción |
|---|---|
| atcrt | Crea un canal de comandos AT |
| atdst | Destruye un canal de comandos AT |
| atoi | Convierte cadena a entero |
| atsnd | Enviar comandos AT obtener respuesta |

| | |
|---|---|
| io | Maneja los módulos pines IO, lo que permite la configuración, la fijación de los niveles, lectura y valores. |

| | |
|---|---|
| slen | Obtener la longitud de una cadena |
| smsd | Borrar mensaje corto (SMS) |
| smsi | Inicializa el servicio de mensajes cortos (SMS) |
| smsrm | Leer mensaje de datos (SMS) |
| smsrs | Buscar el primer mensaje corto en memoria |
| smss | Enviar mensaje corto (SMS) |

A continuación se explica como usar cada función y la importancia que tendrá dentro del programa principal.

***Atcrt***

Esta función crea un canal para poder enviar comandos AT desde el microcontrolador interno del GT48 hacia el GR48; en muchos casos es indispensable el uso de esta función debido a que la mayoría de las funciones necesitan enviar comandos AT al GR48 por lo que es necesario un medio por el cual transmitirlo, similar a la aplicación de hyperterminal usada con anterioridad pero con la diferencia de que esto es realizado internamente en la terminal.

Se usa como cualquier función dentro del lenguaje c:

canal = atcrt();

se le asigna a una variable de tipo entero, el cual obtendrá el resultado de dicha función:
0 si se ha creado el canal correctamente
> 0 si se ha fallado en la creación del canal.

### *Atdst*

Caso contrario a la función de creación se tiene esta función para destruir el canal creado y cerrar la comunicación con el dispositivo. Se declara de la misma manera:

canal = atdst();
para un valor de respuesta de:
0 canal destruido exitosamente
>2 fallo de la destrucción del canal

### *Atsnd*

Creado el respectivo canal para la comunicación, es posible enviar comandos AT's si estos no se encuentran disponibles en forma de funciones propias proporcionadas por el software de Sony Ericsson.

Presenta algunos parámetros a llenar:
canal = atsnd( char *sendCmd, char *resCmd, int sendLength, int resCmdLength, int *resCmdSize );

sendCmd da lugar al comando que se quiere enviar (tipo char).
ResCmd es una variable char destinada a recibir la respuesta de la terminal.
SendLength es un valor entero dado para que se mencione la longitud del comando a enviar.
ResCmdLenght es un valor entero que esta destinado a la longitud de la respuesta.
ResCmdSize guarda el tamaño actual de la respuesta.

Ejemplo:

```
int tamaño;
char respuesta[160];

canal = atsnd("ATD0449611240080",respuesta, 16, 100, &tamaño);
```

se omite la creación y destrucción del canal porque previamente fue mencionada.

***Atoi***

Función que permite convertir una cadena a entero; esto servirá dentro del proyecto para poder convertir las cadenas recibidas por medio de mensajes SMS's

int numero;
char cadena[4] = “1234”;
numero = atoi (cadena);

como resultado tendremos el numero 1 234 para poder realizar cualquier operación aritmética o como identificador.

***io***

Función que permite el uso de entradas y salidas, la configuración y monitoreo de las lineas de i/o.
Maneja parámetros muy similares a los vistos en el comando AT*E2IO.

Configuración = io (codigo, num_io, valor);

código de operación
- 0 = IO_Read
- 1 = IO_Write
- 2 = IO_Config (output or input if allowed)
- 3 = IO_Trigger (if allowed)
- 4 = IO_PinSwitch (switches to pin specified)
- 5 = IO_PinStatus (current op status returned)

num_io define el pin a usar
- 0 = IO1 (Pin 21 default) muxed with I2
- 1 = IO2 (Pin 22 default) muxed with ADC5
- 2 = IO3 (Pin 23 default) muxed with I3
- 3 = IO4 (Pin 24 default) muxed with I4
- 4 = IO5 (Pin 13 default) muxed with ADC4
- 5 = IO6 (Pin 33) LED as default
- 6 = IO7 (Pin 43 default) muxed with UART3 Tx (changes to Tx3 when UART3 is opened - IO7 is then disabled)
- 7 = IO8 (Pin 44 default) muxed with UART3 Rx (changes to Rx3 when UART3 is opened - IO8 is then disabled)
- 8 = IO9 (Pin 45) RTS as default
- 9 = I1 (Pin 37) DTR as default
- 10 = I2 (Pin 21) muxed with IO1
- 11 = I3 (Pin 23) muxed with IO3
- 12 = I4 (Pin 24) muxed with IO4

13 = O1 (Pin 38) DCD as default
14 = O2 (Pin 36) RI as default
15 = O3 (Pin 32) DSR as default
16 = O4 (Pin 40) CTS as default
17 = ADC1 (Pin 26)
18 = ADC2 (Pin 27)
19 = ADC3 (Pin 28)
20 = ADC4 (Pin 13) muxed with IO5
21 = ADC5 (Pin 22) muxed with IO2
22 = DAC1 (Pin 20)

valor: depende del código de operación
if codigo = Config/Write/Trigger
value (1) for Output/High/Set Trigger On
value (0) for Input/Low/Set Trigger Off
if codigo = Read/PinSwitch/PinStatus
no se usa el parámetro (completar con 0)

ejemplo:

```
char IO3 = 2;
char ok;
    ok = io(2,IO3,1);
```

***slen***

Función que permite medir la longitud de una cadena.

```
char palabra[10]=”procesando”;
int longitud;

longitud = slen (palabra);
```

devuelve el valor entero que determina la longitud de la cadena en la variable tipo entero.

***Smsd***

Función con la que se permite borrar un mensaje SMS a la vez. Elimina el mensaje en el espacio de memoria indicado. Esta función tiene el inconveniente de que cada ocasión que es usada dentro del programa este reinicializa el dispositivo.

```
Int borrar;
int espacio_numero = 1;
```

```
borrar = smsd(espacio_numero);
```

la variable *espacio_numero* será el indicador del espacio en memoria donde se encuentra el mensaje a ser borrado, indicado por un numero entero; en la variable *borrar* se obtiene la respuesta de la función; esta función internamente hace uso de los comandos siguientes: AT+CPMS y AT+CMGD.

### *Smsi*

Es una función que permite inicializar el servicio de mensajes cortos; puede configurarse para borrar todos los mensajes guardados en el ME y en el SIM si se esta colocado en 1; o caso contrario no removerlos si esta colocado en 0.

utiliza tres de los comandos antes vistos:
AT+CPMS, AT+CPIN, AT+CMGF

```
int inicializar;
inicializar = smsi(1,1);
```

### *smsrm*

Esta función permite leer el mensaje en un SMS y guardar este en una variable de tipo char, así también usa dos variables más como parámetros para poder realizar esta acción; uno de ellos es una variable definida por el usuario de tipo entero y en esta se determina el número de bytes máximo que será ocupado por el mensaje, mientras que en la otra variable se indica el espacio en memoria de donde se leerá el mensaje.

```
Char mensaje[160];
int espacio_numero;

Leer = smsrm(mensaje, 160, espacio_numero);
```

la función devuelve un valor que puede ser interpretado de dos maneras:
si es 0 el mensaje se ha leído con éxito.
Si es mayor a 0 se ha fallado en la lectura en el espacio de memoria indicado.

### *Smsrs*

función que localiza el primer mensaje no leído de un SMS o de un mensaje guardado en la memoria de la terminal. La función devuelve un valor de 0 si se logró encontrar un mensaje

no leído, y un valor mayor a 0 en caso de no encontrar ningún nuevo mensaje no leído.

```
Int buscar;
        buscar = smsrs();
```

esta función hace uso de los comandos AT+CMGS, AT+CMGF.

***Smss***

para realizar el envío de mensajes SMS a otros dispositivos se hace uso de esta función; esta necesita 5 parámetros para poder enviar el mensaje:

smss(char *número*, char *mensaje*, int *dirección_tipo*, int *long_dirección*, int *long_mensaje*);
donde:

| | |
|---|---|
| *número* | pertenece al número al cual se desea enviar el mensaje. |
| *Mensaje* | en este apartado se escribe el mensaje a enviar |
| *dirección_tipo* | hace referencia a un parámetro requerido para poder enviar mensajes en la región, que para nuestro casi siempre tendrá el valor de 129. |
| *long_dirección* | referido a la longitud del número telefónico colocado en *número* |
| *long_mensaje* | referido a la longitud de mensaje a enviar. |

Ejemplo:

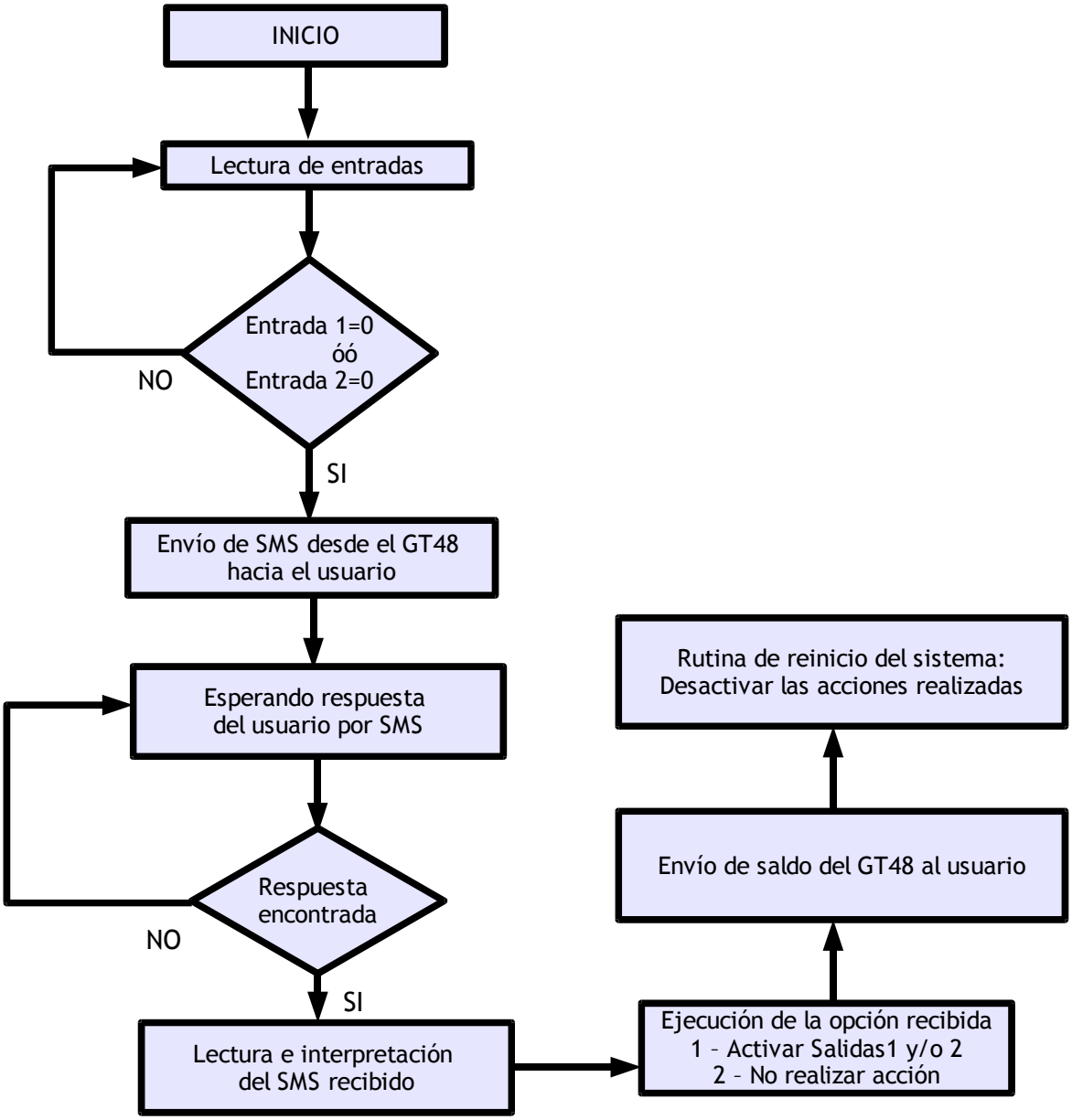
```
int enviar;
enviar = smss("9611240080","Test SMS", 129, 10, 8);
```

si el mensaje ha sido enviado con éxito la función devuelve un valor de 0 que se almacena en la variable enviar y que puede ser usado para otros fines.

## 8.3 PROGRAMA PRINCIPAL

Con lo previamente explicado sobre las funciones propias del software se logra estructurar un programa que permita realizar todas las funciones deseadas. Para ello antes de crear el programa se debe analizar la manera en que el sistema funcionará.

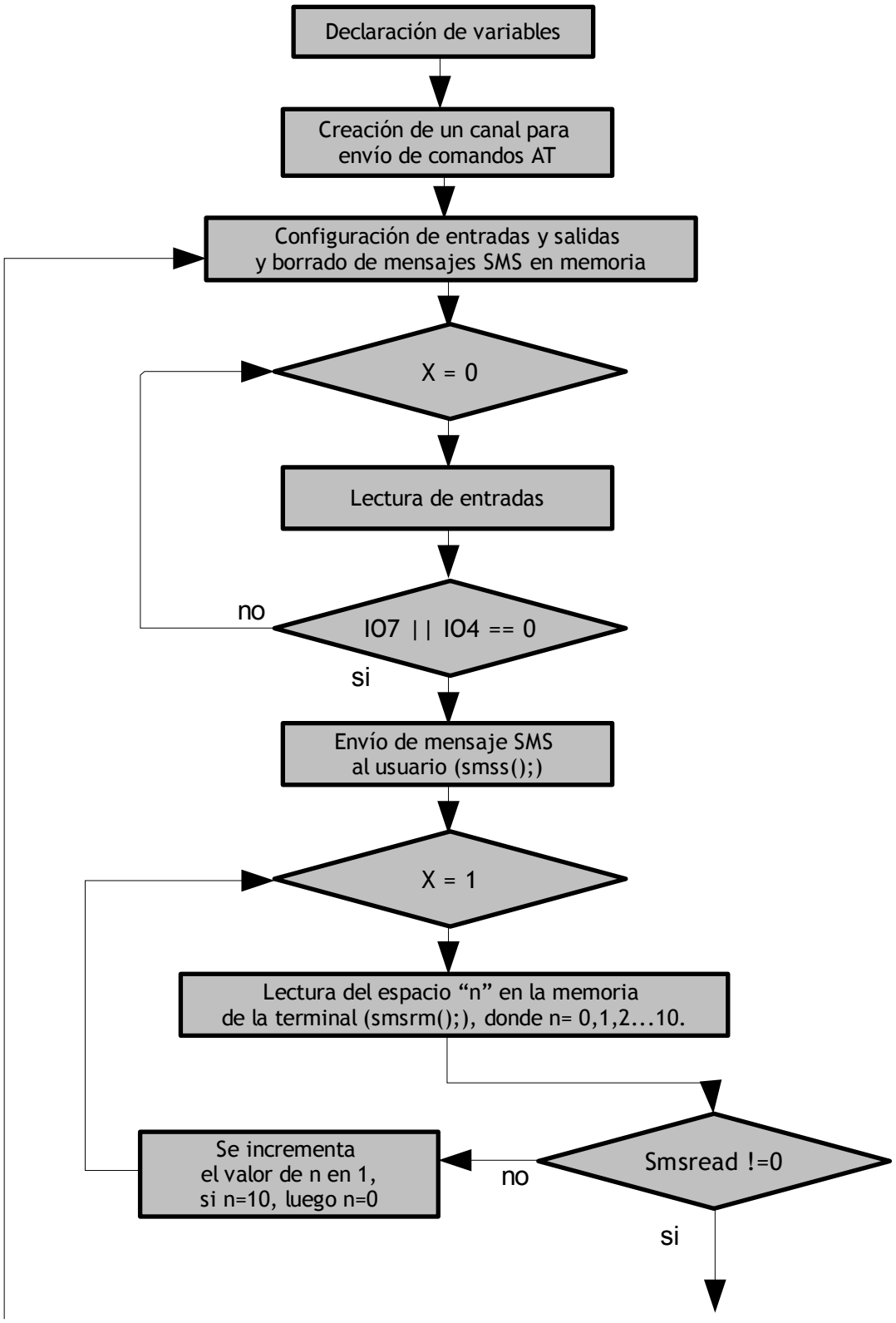
Dicho funcionamiento se explica en el siguiente diagrama:

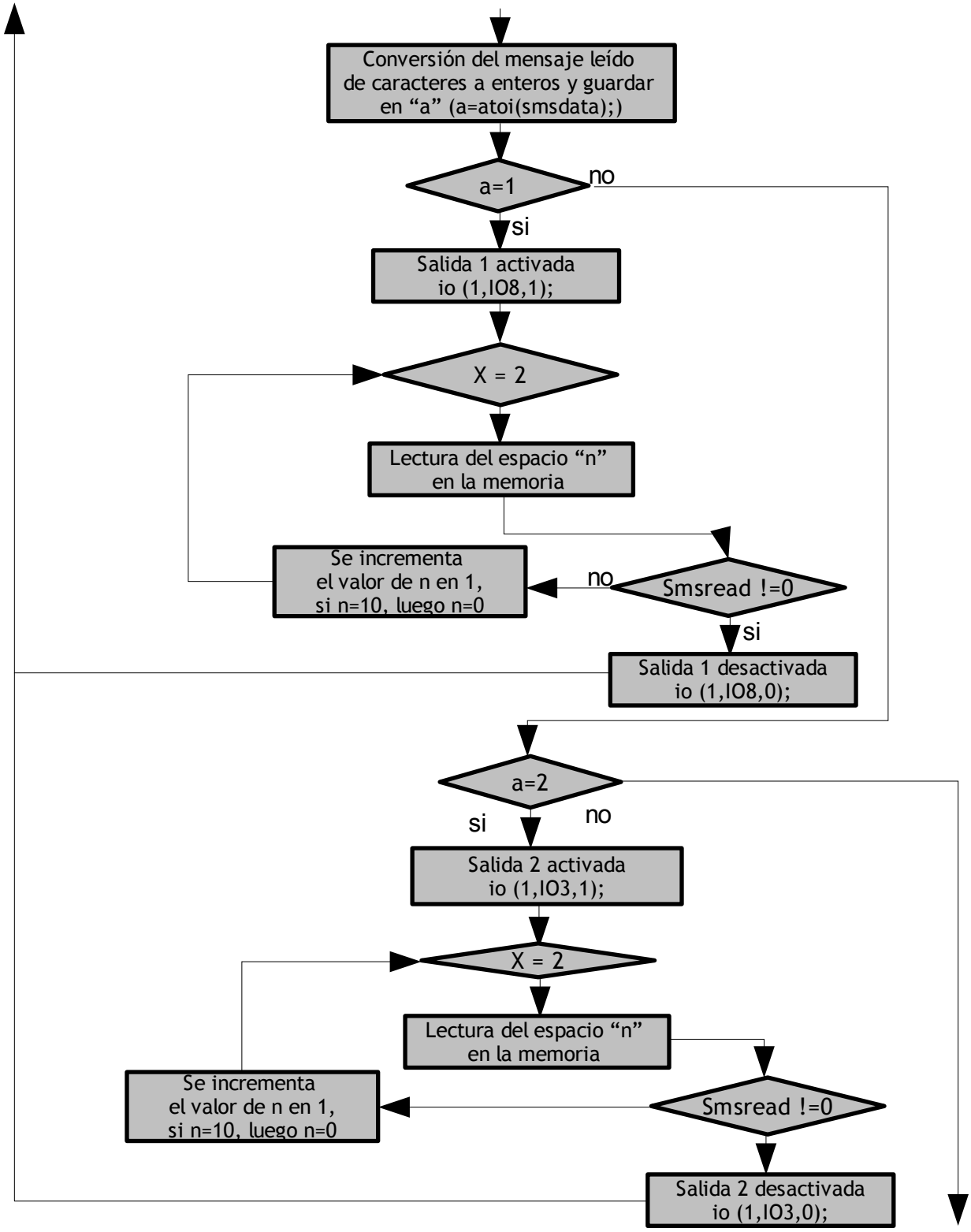


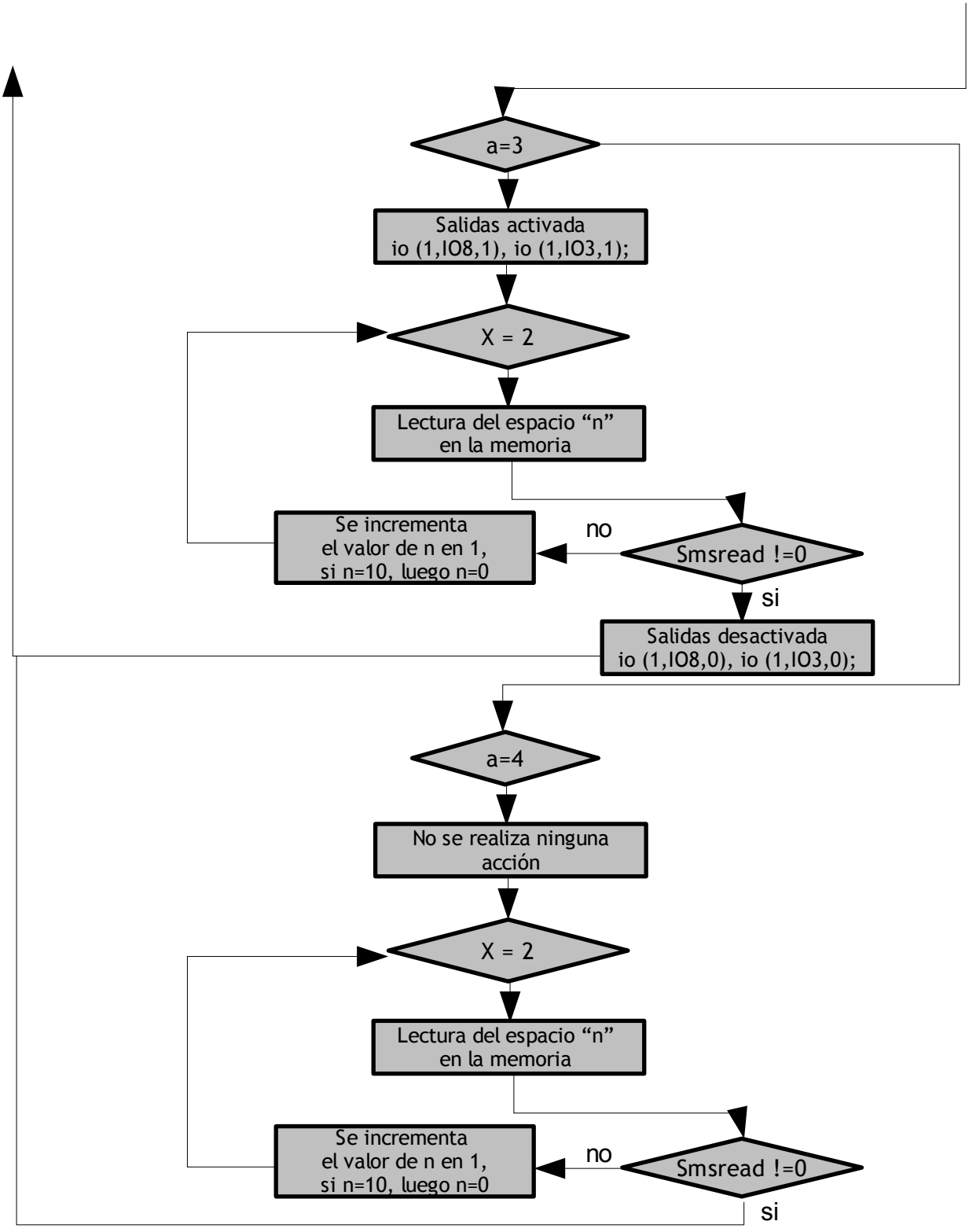

El diagrama presentado muestra claramente el funcionamiento básico de la terminal GT48 cuando este se encuentra en uso y en condiciones adecuadas para su funcionamiento.

Con base en el diagrama mostrado es posible estructurar un programa usando todas las funciones antes mencionadas; a continuación se presenta un nuevo diagrama a bloques que muestra específicamente como se realizarán las acciones.

Diagrama de flujo




Conversión del mensaje leído de caracteres a enteros y guardar en “a” (a=atoi(smsdata);)
a=1
no
si
Salida 1 activada io (1,IO8,1);
X = 2
Lectura del espacio “n” en la memoria
Se incrementa el valor de n en 1, si n=10, luego n=0
no
Smsread !=0
si
Salida 1 desactivada io (1,IO8,0);
a=2
si
no
Salida 2 activada io (1,IO3,1);
X = 2
Lectura del espacio “n” en la memoria
Se incrementa el valor de n en 1, si n=10, luego n=0
Smsread !=0
Salida 2 desactivada io (1,IO3,0);


a=3
Salidas activada
io (1,IO8,1), io (1,IO3,1);
X = 2
Lectura del espacio “n”
en la memoria
Se incrementa
el valor de n en 1,
si n=10, luego n=0
no
Smsread !=0
si
Salidas desactivada
io (1,IO8,0), io (1,IO3,0);
a=4
No se realiza ninguna
acción
X = 2
Lectura del espacio “n”
en la memoria
Se incrementa
el valor de n en 1,
si n=10, luego n=0
no
Smsread !=0
si

## *Script basado en lenguaje de Programación C.*

Declaración de variables que seriviran para distinguir las entradas y salidas:

```
/*entradas*/
char IO4=3;
char IO7=6;

/*salidas*/
char IO8=7;
char IO3=2;
```

```
/* main */
main ()
{
```

Las siguientes variables son declaradas para ser usadas en funciones, comparaciones o para asignación de valores:

```
int resCmdSize;
char resCmd[100];
int canal;
int smsread;
int smsslot=0;
int smsenvio;
char smsdata[160];
char config;
int long;
int a;
int leer0;
int leer1;
int x=0;
int ok;
int smsborrar;
int smsslot1=0;
int incorrecto=1;
```

```
canal = atcrt();
```

Creación de un canal por el cual se recibiran comandos AT.

```
canal = atsnd ("AT+CNMI=3,1,0,0,0",  resCmd, 17, 100, &resCmdSize);
```

Configuración de parámetros

Configuración de entradas y salidas:

```
config = io(2,IO8,1);
```
Salida 1

```
config = io(2,IO3,1);
```
Salida 2

```
config = io(2,IO7,0);
```
Entrada IO7

```
config = io(2,IO4,0);
```
Entrada IO4

```
while(1)
{
config = smsi(1,1);
```
Borrado de mensajes en memoria

```
while(x==0)
{
```
Inicio de bucle para verificación de entradas

Lectura de entradas:

```
leer0 = io(0,IO7,0);
leer1 = io(0,IO4,0);
```

Comparación de las lecturas obtenidas con la establecida en la condición, entrada IO7=1 e IO4=0;
```
if((leer0==0)&&(leer1==1))
{
```
Envío de mensaje SMS de alerta por alteración en la entrada IO7:
```
smsenvio = smss("9611240080", "entrada IO7 abierta, accion: 1-salida 1, 2-salida 2, 3-activar todo, 4-ninguno", 129, 10, 78);
x=1;
```
Cambio de valor de “x” para propiciar la salida del bucle inicial.
```
}
```

Comparación de las lecturas obtenidas con la establecida en la condición, entrada IO7=0 e IO4=1;
```
if((leer0==1)&&(leer1==0))
{
```
Envío de mensaje SMS de alerta por alteración en la entrada IO4:
```
smsenvio = smss("9611240080", "entrada IO4 abierta, accion: 1-salida 1, 2-salida 2, 3-activar todo, 4-ninguno", 129, 10, 78);
x=1;
```
Cambio de valor de “x” para propiciar la salida del bucle inicial.
```
}
```

Comparación de las lecturas obtenidas con la establecida en la condición, entrada IO7=0 e IO4=0;
```
if((leer0==0)&&(leer1==0))
{
```
Envío de mensaje SMS de alerta por alteración en las entradas IO7 e IO4:
```
smsenvio = smss("9611240080", "Las entradas estan abiertas, accion: 1-salida 1, 2-salida 2, 3-activar todo, 4-ninguno", 129, 10, 86);
x=1;
```
Cambio de valor de “x” para propiciar la salida del bucle inicial.
```
}

}
```
**Fin del bucle inicial**

```
/*esperando respuesta por sms del usuario*/
```

**INICIO DEL SEGUNDO BUCLE**
```
while(x==1)
{
```

Lectura de mensaje SMS en el lote de memoria indicado por la variable smsslot y el texto es guardado en la variable smsdata
```
smsread=smsrm(smsdata,160,smsslot);

if(smsread!=0)
```
Mensaje encontrado
```
{
```
Conversión del mensaje de texto, de variables tipo carácter a enteros y almacenamiento de ese valor en la variable a.
```
a=atoi(smsdata);
```

**RESPUESTA 1**
```
/*activando salida 1*/
if(a==1)
```
comparación del valor recibido del mensaje SMS, a=1
```
{
ok = io(1,IO8,1);
```
Activación de la salida 1
```
if(ok==1)
```
Verificación de activación de la salida
```
{
```
Envío de mensaje al usuario confirmando la activación de la salida 1
```
smsenvio = smss("9611240080", "Salida 1 activada", 129, 10, 17);
}
x=2;
```
Cambio de valor a “x” para entrar en un nuevo bucle

**Inicio de bucle interno**

```
while(x==2)
{
```

Comparación de variables asignadas a lotes de memoria para lectura de mensajes, para evitar la lectura del primer mensaje recibido al inicio del proceso.

```
if(smsslot1==smsslot)
{
smsslot1++;        incremento de la variable en 1
}
smsread=smsrm(smsdata,160,smsslot1);      Lectura de mensaje SMS en el lote asignado por smsslot1
if(smsread!=0)                            Verificación de que se ha logrado la lectura de un nuevo mensaje
{
ok = io(1,IO8,0);                         Desactivación de la salida 1 (Antes activada)
x=0;                                      Cambio de valor de x para forzar la salida del bucle interno y el segundo bucle y
                                          permitir el regreso al primer bucle de lectura de entradas
}
smsslot1++;                               Incremento del valor del lote en memoria a verificar
```

máximo valor de smsslot1=10, al alcanzar ese valor el conteo se reinicia para proceder nuevamente a la lectura de los espacios en memoria.

```
if(smsslot1==10)
{
smsslot1=0;
}
}
```

**Fin del bucle interno**

```
}
```

**RESPUESTA 2**

```
/*activando salida 2*/
if(a==2)                  comparación del valor recibido del mensaje SMS, a=2
{
ok = io(1,IO3,1);         Activación de la salida 2
if(ok==1)                 Verificación de activación de la salida
{
```

Envío de mensaje al usuario confirmando la activación de la salida 2

```
smsenvio = smss("9611240080", "Salida 2 activada", 129, 10, 17);
}
x=2;                      Cambio de valor a “x” para entrar en un nuevo bucle
```

**Inicio de bucle interno**

```
while(x==2)
{
```

Comparación de variables asignadas a lotes de memoria para lectura de mensajes, para evitar la lectura del primer mensaje recibido al inicio del proceso.

```
if(smsslot1==smsslot)
{
smsslot1++;               incremento de la variable en 1
}
smsread=smsrm(smsdata,160,smsslot1);      Lectura de mensaje SMS en el lote asignado por smsslot1
if(smsread!=0)                            Verificación de que se ha logrado la lectura de un nuevo mensaje
{
ok = io(1,IO3,0);                         Desactivación de la salida 2 (Antes activada)
x=0;                                      Cambio de valor de x para forzar la salida del bucle interno y el segundo bucle
                                          y permitir el regreso al primer bucle de lectura de entradas
}
smsslot1++;                               Incremento del valor del lote en memoria a verificar
```

máximo valor de smsslot1=10, al alcanzar ese valor el conteo se reinicia para proceder nuevamente a la lectura de los

espacios en memoria.
```
if(smsslot1==10)
{
smsslot1=0;
}
}
```
**Fin del bucle interno**
```
}
```

**RESPUESTA 3**
```
/*activando salida 1 y 2*/
if(a==3)                    Comparación del valor recibido del mensaje SMS, a=3
{
ok = io(1,IO8,1);           Activación de las salidas 1 y 2
ok = io(1,IO3,1);
if(ok==1)                   Verificación de activación de la salida
{
```
Envío de mensaje al usuario confirmando la activación de las salidas
```
smsenvio = smss("9611240080", "Salidas activadas", 129, 10, 17);
}
x=2;                        Cambio de valor a “x” para entrar en un nuevo bucle
```

**Inicio de bucle interno**
```
while(x==2)
{
```
Comparación de variables asignadas a lotes de memoria para lectura de mensajes, para evitar la lectura del primer mensaje recibido al inicio del proceso.
```
if(smsslot1==smsslot)
{
smsslot1++;                 incremento de la variable en 1
}
smsread=smsrm(smsdata,160,smsslot1);    Lectura de mensaje SMS en el lote asignado por smsslot1
if(smsread!=0)                          Verificación de que se ha logrado la lectura de un nuevo mensaje
{
ok = io(1,IO8,0);                       Desactivación de las salidas (Antes activadas)
ok = io(1,IO3,0);
x=0;                                    Cambio de valor de x para forzar la salida del bucle interno y el segundo bucle
                                        y permitir el regreso al primer bucle de lectura de entradas
}
smsslot1++;                             Incremento del valor del lote en memoria a verificar
```

máximo valor de smsslot1=10, al alcanzar ese valor el conteo se reinicia para proceder nuevamente a la lectura de los espacios en memoria.
```
if(smsslot1==10)
{
smsslot1=0;
}
}
```
**Fin del bucle interno**
```
}
```

**RESPUESTA 4**
```
if(a==4)                    comparación del valor recibido del mensaje SMS, a=4
{
```
Envío de mensaje al usuario confirmando que no se activó ninguno salida
```
smsenvio = smss("9611240080", "No se realizo accion alguna", 129, 10, 27);
```

```
x=2;                    Cambio de valor a “x” para entrar en un nuevo bucle
```

**Inicio de bucle interno**

```
while(x==2)
{
```

Comparación de variables asignadas a lotes de memoria para lectura de mensajes, para evitar la lectura del primer mensaje recibido al inicio del proceso.

```
if(smsslot1==smsslot)
{
smsslot1++;             incremento de la variable en 1
}
smsread=smsrm(smsdata,160,smsslot1);    Lectura de mensaje SMS en el lote asignado por smsslot1
if(smsread!=0)                          Verificación de que se ha logrado la lectura de un nuevo mensaje
{
x=0;                                    Cambio de valor de x para forzar la salida del bucle interno y el segundo bucle
                                        y permitir el regreso al primer bucle de lectura de entradas
}
smsslot1++;                             Incremento del valor del lote en memoria a verificar
```

maximo valor de smsslot1=10, al alcanzar ese valor el conteo se reinicia para proceder nuevamente a la lectura de los espacios en memoria.

```
if(smsslot1==10)
{
smsslot1=0;
}
}
```

**Fin del bucle interno**

```
}
```

**RESPUESTA 5**

```
if((a!=1) && (a!=2) && (a!=3) && (a!=4))    Comparación del valor recibido del mensaje SMS, donde “a” es diferente a
                                            cualquier valor entero desde 1 hasta 4
{
smsenvio = smss("9611240080", "codigo incorrecto", 129, 10, 17);        Envío de mensaje al usuario confirmando
                                                                        que no se introdujo un valor válido
x=1;                        valor de “x” para regresar al bucle en espera de un nuevo valor.
config = smsi(1,1);         Borrado de mensajes guardados en memoria
}
}


smsslot++;                  Incremento de variable asignada para verificación de mensajes sms
if(smsslot==10)             Valor máximo de incremento de la variable smsslot
{
smsslot=0;
}

}
```

**FIN DEL SEGUNDO BUCLE**

```
}
canal = atdst ();           Destrucción del canal de transmisión de comandos
}
```

**FIN DE PROGRAMA**

## 8.4 SALDO DEL DISPOSITIVO

El GT48 al ser un dispositivo capaz de usar la red GSM es necesario que se cuente de un número y consecuentemente de tiempo aire para poder realizar las operaciones de envío de mensajes SMS, por lo que el dispositivo cuenta con lector de tarjetas SIM, permitiendo la posibilidad de usar cualquier número que sea adquirido.

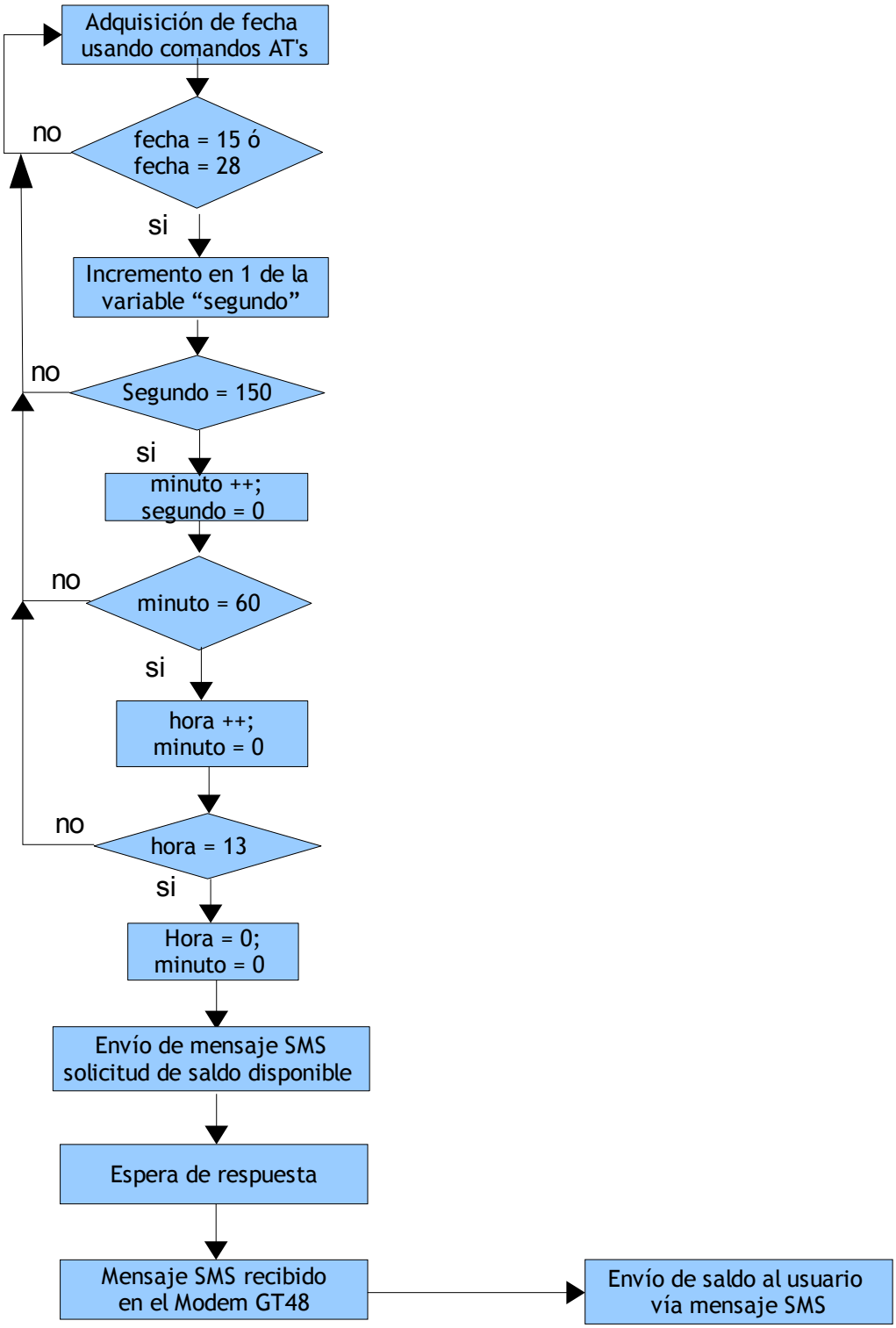
La inserción de tiempo aire se puede realizar de la siguiente manera:

- Extrayendo la tarjeta SIM e introducir mediante celular al 333 el número de ficha adquirida.
- Usando un número en plan de renta (de cualquier tipo).
- Traspasando crédito de un número celular a otro.
- Realizar recargas al número desde cualquier centro de recarga de la compañía elegida.

Como se pudo observar, la obtención de tiempo aire es posible realizarla de la misma manera que cualquier teléfono móvil.

En el caso del uso de recargas de tiempo aire, es indispensable conocer el crédito disponible para que las funciones del dispositivo puedan ser llevadas a cabo. El sistema esta diseñado para que cada día 15 y 28 de cada mes sea enviado un mensaje SMS a un número previamente programado elegido por el usuario, a continuación se muestra un diagrama a bloques y parte del script en el cual se realiza la obtención y envío del crédito disponible para el sistema de alarma:

Diagrama a bloques de la adquisición de saldo del dispositivo



## *Script basado en lenguaje de programación C para la consulta de saldo del dispositivo*

```
while(1)
{
canal = atsnd ("AT+CCLK?",  resCmd, 8, 18, &resCmdSize);
          sncpy(resCmd,"              ",16);

          ok=atoi(resCmd);
          if(y==0)
          {
          dia =ok+1;
          y=1;
          }
          if((ok == 15)||(ok == 28))
          {
                                              segundo++;
                                              if(segundo==150)
                                              {
                                              minuto++;
                                              segundo=0;
                                              }
                                                        if(minuto==60)
                                                        {
                                                        hora++;
                                                        minuto=0;
                                                        }

                    if(hora==13)
                    {
                    segundo=0;
                    minuto=0;
                    hora=0;
                    smsenvio = smss("333", "saldo", 129, 3, 5);
                              while(x==0)
                              {
                              smserr=smsrm(smsdata,160,smsslot);
                              smsslot++;
                                        if(smsslot==10)
                                        {
                                        smsslot=0;
                                        }
                                        if(smserr!=0)
                                        {
                                        x=2;
                                        }
                                        if(smserr!=0)
                                        {
                                        ok=atoi(smsdata);
                                        itoa(ok, smsdata,100);
                                        scpy (saldo,"El saldo del modem GSM es de: ");
                                        scat(saldo,smsdata);
                                        smsenvio = smss("9611240080", saldo, 129, 10, 32);
                                        }
                              }
                    smserr=0;
                    smsslot=0;
                    y=0;
                    }
```

```
        }
        else
        {
        segundo=0;
        minuto=0;
        hora=0;
        }
}
```

De manera básica se explica lo anterior en los siguientes pasos:

1. Obtención de la fecha mediante el comando AT correspondiente.
2. Comparación de la fecha obtenida con la almacenada (días: 15 – 28)
3. Al cumplir la condición se inicia un conteo de aproximadamente 13 horas.
4. Al llegar el conteo a su límite se procede a un envío de mensaje SMS dirigido al centro de servicio de telefonía celular requiriendo el tiempo aire disponible.
5. El sistema se mantiene en espera de la respuesta por parte del centro de servicio.
6. Al recibir la información vía mensaje SMS, se procede a reenviar la información al número del usuario.
7. Se reinicia el contador y el proceso de obtención de saldo.

## 8.5 HARDWARE DE ENTRADA/SALIDA

Las entradas y salidas de la terminal GT48 se encuentran diseñadas para operar y ser manipuladas directamente desde el conector DB15; pero para asegurar el correcto funcionamiento y evitar cualquier tipo de daño hacia el dispositivo, es prescindible diseñar e implementar una interfaz tanto de entrada como de salida.

### Entradas

La terminal GT48 como se estudió con anterioridad cuenta con 3 entradas digitales, de las cuales 2 se encuentran localizadas en el conector DB15 y otra en el conector RJ12 del dispositivo.

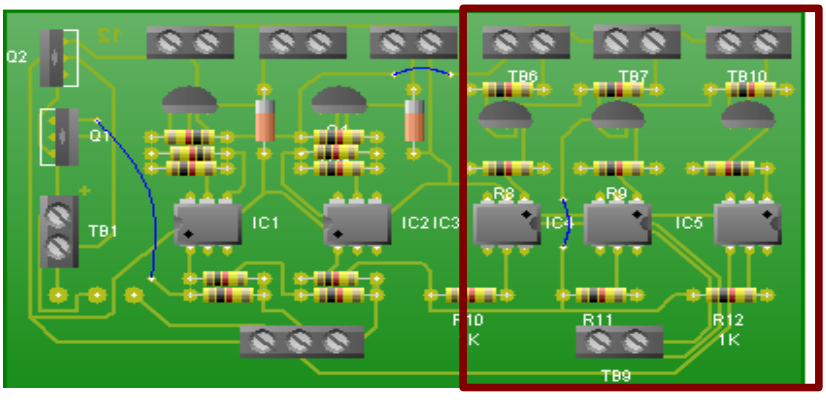



Fig. 3.- Placa construida para entradas y salidas.

Cada línea que constituye la interfaz de entrada (3 en total), esta diseñada para ser polarizada, la señal de entrada esta conectada directamente a la base del transistor BC548, lo que hace que pase de los estados de corte y saturación, logrando de esta forma el cambio de aprox. 0 a 5 volts; dicho nivel de voltaje se introduce al opto acoplador, de tal manera se logra una separación entre el dispositivo conectado a la línea de entrada y el Modem GSM GT48.

La conexión de cada línea se presenta a continuación:

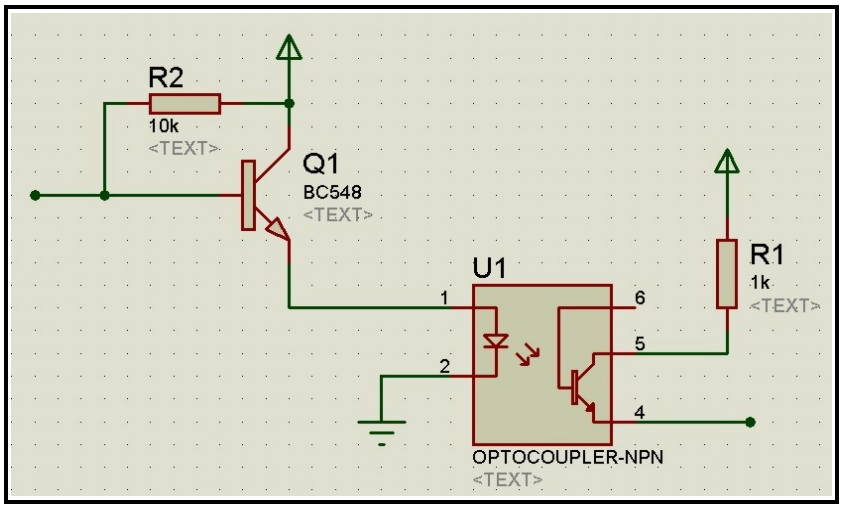



Fig. 4.- Diagrama esquemático de las entradas del dispositivo

**Salidas**

Las dos líneas de salidas con las que cuenta la interfaz de salida se encuentran estructuradas de forma similar a las líneas de entrada; La descripción técnica del GT48 indica que las salidas IO8 e IO3 tienen una salida del tipo Low Side Switch, esto indica que para poder manipular la salida es necesario polarizarla, por lo cual se coloca una resistencia directamente a la terminal positiva de una fuente de energía de 5 volts; hecho esto la salida puede ser enviada hacia el opto acoplador.

Conectado al emisor del fototransistor se encuentra un transistor BC548 para realizar el cambio de 0 a 12 volts a su salida; la cual puede ser conectado a un relevador que funcione con este valor de voltaje.

La conexión se muestra a continuación:

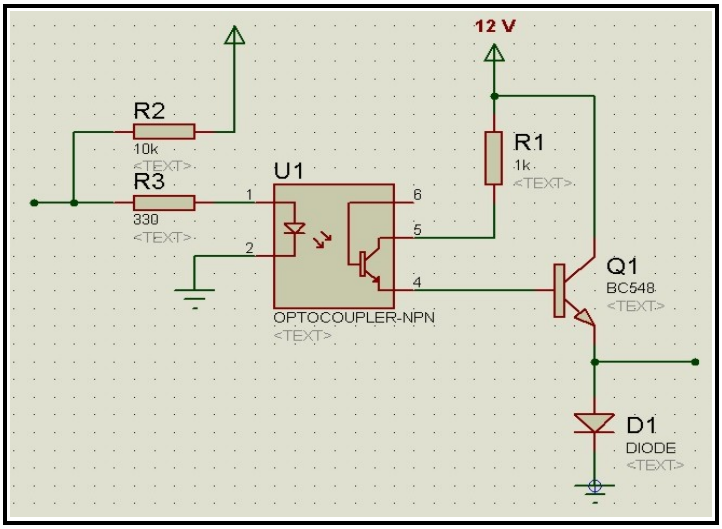



Fig. 5.- Diagrama esquemático de las salidas del dispositivo

## 9. RESULTADOS

A continuación se presenta el dispositivo físico construido después de lo antes visto:


Fig. 6.- Vista desde arriba

Fig. 7.- Vista lateral izquierda

Fig. 8.- Vista lateral derecha


Fig. 9.- Vista frontal


Fig. 10.- Vista desde arriba, enfocando el led indicador de señal

Fig. 11.- Vista Posterior

**PROGRAMA PRINCIPAL**

```
/*entradas*/
char IO4=3;
char IO7=6;
char IO5=4;

/*salidas*/
char IO8=7;
char IO3=2;

/* main */
main ()
{
int resCmdSize;
char resCmd[100];

int canal;
int smsread;
int smsslot=0;
int smsenvio;
char smsdata[160];
char config;
int long;
int a;
int leer0;
int leer1;
int leer2=1;
int x=0;
int ok;
int smsborrar;
int smsslot1=0;
int incorrecto=1;
int segundo=0;
int minuto=0;
int hora=0;
int smserr=0;
char saldo[160];
int y=0;
int dia=0;

canal = atcrt();
canal = atsnd ("AT+CNMI=3,1,0,0,0",  resCmd, 17, 100, &resCmdSize);
```

```
config = io(2,IO8,1);
config = io(2,IO3,1);
config = io(2,IO7,0);
config = io(2,IO4,0);
config = io(2,IO5,0);

while(1)
{
x=0;
config = smsi(1,1);

while(leer2==1)
{
leer2 = io(0,IO5,0);
canal = atsnd ("AT+CCLK?",  resCmd, 8, 18, &resCmdSize);
                        sncpy(resCmd,"               ",16);

                        ok=atoi(resCmd);
                        if(y==0)
                        {
                        dia =ok+1;
                        y=1;
                        }
                        if((ok == 15)||(ok == 28))
                        {

                                        segundo++;
                                        if(segundo==150)
                                        {
                                        minuto++;
                                        segundo=0;
                                        }
                                                if(minuto==60)
                                                {
                                                hora++;
                                                minuto=0;
                                                }

                         if(hora==13)
                         {
                         segundo=0;
                         minuto=0;
                         hora=0;
                         ok = io(1,IO3,0);
```

```
                        smsenvio = smss("333", "saldo", 129, 3, 5);
                                while(x==0)
                                {
                                smserr=smsrm(smsdata,160,smsslot);
                                smsslot++;
                                        if(smsslot==10)
                                        {
                                        smsslot=0;
                                        }
                                        if(smserr!=0)
                                        {
                                        x=2;
                                        }
                                        if(smserr!=0)
                                        {
                                        ok=atoi(smsdata);
                                        itoa(ok, smsdata,100);

                                        scpy (saldo,"El saldo del modem GSM es de: ");
                                        scat(saldo,smsdata);

                                        smsenvio = smss("9611240080", saldo, 129, 10, 32);
                                        }
                                }
                         smserr=0;
                         smsslot=0;
                         y=0;
                         }
                        }
                        else
                        {
                        segundo=0;
                        minuto=0;
                        hora=0;
                        }

}
while(x==0)
                         {
                         leer0 = io(0,IO7,0);
                         leer1 = io(0,IO4,0);
                         leer2 = io(0,IO5,0);

                                        if((leer0==0)&&(leer1==1))
```

```
                                        {
                                        dlys(5);
                                        leer2 = io(0,IO5,0);

                                                if(leer2==1)
                                                {
                                                smsenvio = smss("9611240080", "entrada IO7
abierta, accion: 1-salida 1, 2-salida 2, 3-activar todo, 4-ninguno", 129, 10, 78);
                                                x=1;
                                                }
                                        }
                                        if((leer0==1)&&(leer1==0))
                                        {
                                        dlys(5);
                                        leer2 = io(0,IO5,0);

                                                if(leer2==1)
                                                {
                                                smsenvio = smss("9611240080", "entrada IO4
abierta, accion: 1-salida 1, 2-salida 2, 3-activar todo, 4-ninguno", 129, 10, 78);
                                                x=1;
                                                }
                                        }

                                        if((leer0==0)&&(leer1==0))
                                        {
                                        dlys(5);
                                        leer2 = io(0,IO5,0);

                                                if(leer2==1)
                                                {
                                                smsenvio = smss("9611240080", "Las entradas
estan abiertas, accion: 1-salida 1, 2-salida 2, 3-activar todo, 4-ninguno", 129, 10, 86);
                                                x=1;
                                                }
                                        }

canal = atsnd ("AT+CCLK?",  resCmd, 8, 18, &resCmdSize);
                                sncpy(resCmd,"               ",16);

                                ok=atoi(resCmd);
                                if(y==0)
                                {
                                dia =ok+1;
```

```
y=1;
}
if((ok == 15)||(ok == 28))
{

                        segundo++;
                        if(segundo==150)
                        {
                        minuto++;
                        segundo=0;
                        }
                                if(minuto==60)
                                {
                                hora++;
                                minuto=0;
                                }

 if(hora==13)
 {
 segundo=0;
 minuto=0;
 hora=0;
 ok = io(1,IO3,0);
 smsenvio = smss("333", "saldo", 129, 3, 5);
        while(x==0)
        {
        smserr=smsrm(smsdata,160,smsslot);
        smsslot++;
                if(smsslot==10)
                {
                smsslot=0;
                }
                if(smserr!=0)
                {
                x=2;
                }
                if(smserr!=0)
                {
                ok=atoi(smsdata);
                itoa(ok, smsdata,100);

                scpy (saldo,"El saldo del modem GSM es de: ");
                scat(saldo,smsdata);
```

```
                                        smsenvio = smss("9611240080", saldo, 129, 10, 32);
                                        }
                                }
                        smserr=0;
                        smsslot=0;
                        y=0;
                        }
                        }
                        else
                        {
                        segundo=0;
                        minuto=0;
                        hora=0;
                        }


}
/*esperando respuesta por sms del usuario*/
while(x==1)
{
                        smsread=smsrm(smsdata,160,smsslot);
                        if(smsread!=0)
                        {
                        a=atoi(smsdata);
                                        if(a==1)
                                        {
                                                ok = io(1,IO8,1);
                                                        if(ok==1)
                                                        {
                                                        smsenvio = smss("9611240080", "Salida
1 activada", 129, 10, 17);
                                                        }
                                                        x=2;
                                                                while(x==2)
                                                                        {
                                                                        if(smsslot1==smsslot)
                                                                                {
                                                                                smsslot1++;
                                                                                }

                        smsread=smsrm(smsdata,160,smsslot1);
                                                                        if(smsread!=0)
                                                                                {
                                                                                ok = io(1,IO8,0);
```

```
                                                x=0;
                                                }
                                        smsslot1++;
                                        if(smsslot1==10)
                                        {
                                        smsslot1=0;
                                        }
                                        }
                                }

                                /*activando salida 2*/
                                if(a==2)
                                {
                                        ok = io(1,IO3,1);
                                                if(ok==1)
                                                {
                                                smsenvio = smss("9611240080", "Salida
2 activada", 129, 10, 17);
                                                }
                                                x=2;
                                                        while(x==2)
                                                        {
                                                        if(smsslot1==smsslot)
                                                                {
                                                                smsslot1++;
                                                                }

                        smsread=smsrm(smsdata,160,smsslot1);
                                                        if(smsread!=0)
                                                                {
                                                                ok = io(1,IO3,0);
                                                                x=0;
                                                                }
                                                        smsslot1++;
                                                        if(smsslot1==10)
                                                        {
                                                        smsslot1=0;
                                                        }
                                                        }
                                }

                                if(a==3)
                                {
                                        ok = io(1,IO8,1);
```

```
                                        ok = io(1,IO3,1);
                                                if(ok==1)
                                                {
                                                smsenvio = smss("9611240080", "Salidas
activadas", 129, 10, 17);
                                                }
                                                x=2;
                                                        while(x==2)
                                                                {
                                                                if(smsslot1==smsslot)
                                                                        {
                                                                        smsslot1++;
                                                                        }

                                smsread=smsrm(smsdata,160,smsslot1);
                                                                if(smsread!=0)
                                                                        {
                                                                        ok = io(1,IO8,0);
                                                                        ok = io(1,IO3,0);
                                                                        x=0;
                                                                        }
                                                                smsslot1++;
                                                                if(smsslot1==10)
                                                                {
                                                                smsslot1=0;
                                                                }
                                                                }
                                        }

                                        if(a==4)
                                        {
                                        smsenvio = smss("9611240080", "No se realizo accion
alguna", 129, 10, 27);
                                        x=2;
                                                        while(x==2)
                                                                {
                                                                if(smsslot1==smsslot)
                                                                        {
                                                                        smsslot1++;
                                                                        }

                                smsread=smsrm(smsdata,160,smsslot1);
                                                                if(smsread!=0)
                                                                        {
```

```
                                                                x=0;
                                                                }
                                                        smsslot1++;
                                                        if(smsslot1==10)
                                                        {
                                                        smsslot1=0;
                                                        }
                                                        }
                                        }

                                        if((a!=1) && (a!=2) && (a!=3) && (a!=4))
                                        {
                                                smsenvio = smss("9611240080", "codigo
incorrecto", 129, 10, 17);
                                                x=1;
                                                config = smsi(1,1);
                                        }
                                }


                                smsslot++;
                                 if(smsslot==10)
                                 {
                                 smsslot=0;
                                 }

}

}
canal = atdst ();
}
```

### *Datos técnicos*

$V_{entrada}$ = 12.0 Volts
Gnd = 0.0 Volts

Salidas
$V_{out}$ = 12.0 Volts

Entradas
$V_{in}$ = 5.0 Volts
$V_{out}$ = 12.0 Volts

# 10. CONCLUSIÓN

Usando un dispositivo con funciones propias de la telefonía celular, se hizo posible el desarrollo de un sistema de alarma vía celular, todo lo anterior llevado a cabo mediante un software especializado para la creación de scripts basados en lenguaje de programación C.

Se realizó una interfaz para facilitar el control de las señales de entradas y salidas basadas básicamente en opto acopladores, buscando de esta manera la protección del dispositivo principal (GT48) y permitiendo de esa manera el uso de diferentes tensiones en el caso de las señales de entrada.

A lo largo de la realización del proyecto se cumplieron cada uno de los objetivos específicos. Lo anterior permite el cumplimiento del objetivo principal antes mencionado, que es la creación de un sistema que permita la manipulación de una terminal Modem GT48 de Sony Ericsson para fines de vigilancia, haciendo uso adicionalmente de dispositivos eléctricos y/o electrónicos como son sensores de impacto, presencia, movimiento, relevadores, etc., para que en conjunto el sistema sea capaz de alertar al usuario en caso de que existan alteraciones en los dispositivos de entradas; el sistema desarrollado también cuenta con la posibilidad de activar hasta dos salidas en caso de ser requerido.

Por último cabe mencionar que el sistema es capaz de ser colocado en distintos lugares como son casas, locales comerciales, automóviles, etc. El sistema es capaz de ser usado en cualquier lugar en donde se cuente con cobertura de la telefonía celular (GSM), únicamente es requerido la energía eléctrica con el voltaje y corriente suficientes para la conexión tanto de la alarma como de los actuadores y sensores usados.

# 11. BIBLIOGRAFÍA


- Sony Ericsson Mobile Communications International; ***GT47/GT48 Technical Description***; Sony Ericsson; 1a. ed., 2003.

- Sony Ericsson Mobile Communications International; ***GT47/GT48 AT Commands Manual***; Sony Ericsson; 6a. Ed., 2004

- Castellano Díaz, Manuel; Barrientos González, Humberto; ***Sistema de seguridad con emisión de mensaje SMS***; Práctica No. 1., 2004

# A N E X O S