Universidad Politécnica de Cartagena

industriales
etsii UPCT

SISTEMA DE CONTROL PARA UNA CERRADURA ELECTRÓNICA MEDIANTE MÓDULO GSM COMBINADO CON MICROCONTROLADOR PIC18F2520 Y TARJETAS SMARTCARD

Titulación: Ingeniería Técnica Industrial
Intensificación: Electrónica Industrial
Alumno/a: José Asensio Duarte
Director/a/s: Manuel Sánchez Alonso

Cartagena, 17 de Octubre de 2013

Índice

Índice de Figuras y Tablas

Menciones y agradecimientos

Quisiera agradecer en primer lugar el apoyo recibido por mi director de proyecto D. Manuel Sánchez Alonso, que me ha aconsejado y ofrecido su ayuda durante el desarrollo del proyecto. También al técnico de laboratorio David Henarejos, que meh a ofrecido el material necesario para la realización del proyecto.

Quisiera mencionar también el apoyo de mis padres José y Guadalupe, que han estado empujándome constantemente a terminar mis estudios en todo momento de mi carrera. También quiero agradecer a mi hermano Pablo su asistencia en la realización de la presentación de este proyecto.

Por último, y más importante, quisiera agradecer sobre todo a Marina Hernández Grau sus ideas, sugerencias y constante ayuda en el proyecto. Sin sus aportaciones y su apoyo durante la realización del proyecto, este no habría sido posible.

Muchas gracias.

I. Memoria

1. Introducción

1.1. Enunciado y objetivo del proyecto

En los últimos tiempos, la tecnología ha avanzado a pasos agigantados permitiendo un mayor y más fácil acceso a una gran variedad de servicios y aplicaciones. Muchas herramientas, como teléfonos móviles o ordenadores han evolucionado de tal modo que han pasado de ser un producto extravagante y que requería un conocimiento alto de su funcionamiento para aprovechar sus capacidades, a ser elementos cotidianos del día a día de los que ahora nos es imposible separarnos.

No obstante, son muchos esos productos que se han quedado estancados en su manera de funcionar y en las posibilidades que ofrecen para el usuario, hasta que alguien decide llevarlos un paso más allá. Podemos nombrar, por ejemplo, la reciente aparición y popularidad de los libros electrónicos, inimaginables hace poco tiempo ya que ¿que más se podía conseguir en un libro?.

Uno de estos avances es el acceso remoto a dispositivos cotidianos gracias a elementos como Internet y los mensajes cortos. Por poner un caso, actualmente se esta implementando de manera común el acceso remoto a explotaciones ganaderas, con el cual un granjero puede conocer con pulsar un par de teclas de su móvil el estado de su granja desde la cama. Las alarmas y monitorización de hogares también avanzan en este sentido, ofreciendo a los dueños información constante sobre que está sucediendo en su hogar.

Con esto en mente, este proyecto pretende demostrar lo simple que puede resultar ofrecer nuevas funcionalidades a elementos tan sencillos de la vida cotidiana.

Nuestro objetivo es el de implementar un sistema de comunicación remota entre el usuario y un elemento de seguridad de su hogar, como bien puede ser una caja fuerte o la entrada de nuestra casa. Para ello, haremos uso de la tecnología GSM de los dispositivos móviles, así como de un sistema de comunicación por SMS que nos permitirá mantenernos informados en todo momento.

Este proyecto posee además la capacidad de presentar un producto útil, sencillo y asequible al usuario. Nuestro dispositivo nos avisará automáticamente de cualquier actividad en nuestra ausencia, y simplemente deberemos poner nuestro número de teléfono en el móvil para conectar con nuestro dispositivo. Por último, el uso de un microcontrolador y un módulo GSM como elemento central para la comunicación entre el usuario y su elemento de seguridad hacen de este un producto de fácil acceso económico a cualquiera.

Este dispositivo, apodado *SMSWatch*, también se ve reforzado con una interacción física directa con el dispositivo, mediante pulsadores (teclado alfanumérico), interfaz (Pantalla LCD), identificadores (Smartcards), y los distintos sensores y actuadores que controlarán nuestro elemento de seguridad.

El dispositivo a mostrar estará constituidos por los siguientes componentes:

- Un microcontrolador 18F: el cual controla el estado del elemento de seguridad (en este caso, una caja fuerte), ejerce el control sobre él, almacena los datos del usuario (o usuarios), y maneja tanto el módulo GSM como las tarjetas Smartcard.

- Un módulo GSM con tarjeta: es el encargado de la comunicación con el usuario por telefonía móvil, ya sea transmitiendo mensajes cortos (SMS) con información y de aviso, o recibiendo órdenes del propio usuario mediante los mismos. Este elemento es controlado bilateralmente tanto por el microcontrolador como por el dispositivo móvil del usuario, haciendo de puente entre los dos.

- Un lector de tarjetas Smartcard: se trata del encargado de identificar a los usuarios, y permitir su acceso al elemento de seguridad. Hace de intermediario físico entre el microcontrolador 18F, que actúa y otorga los permisos al dispositivo, y el usuario, que se identifica mediante una tarjeta Smartcard con una memoria EEPROM que contiene la información del usuario.

- Indicadores e interruptores: formados por una pantalla LCD y un teclado alfanumérico, son los encargados de permitir la interacción física entre el usuario y el dispositivo. Permiten al usuario acceder al elemento de seguridad, así cómo permiten configurar diversos aspectos del SMSWatch como el teléfono al que llama el módulo GSM, la gestión de los usuarios y contraseñas, etc.

A continuación se enumeran los principales objetivos del proyecto:

- Diseño e implementación de un circuito impreso (PCB), para la creación de un prototipo que contenga las conexiones entre los distintos elementos anteriormente descritos. Este PCB se diseñará además, de tal modo que muestre al usuario las funciones del prototipo de manera sencilla y directa.

- Desarrollo de conexiones entre los distintos elementos, que permitan al usuario la configuración y manejo tanto del elemento de seguridad como del funcionamiento del dispositivo.

- Desarrollo del programa encargado del manejo del microcontrolador, y que implementará la gestión de los sensores y actuadores, la configuración del funcionamiento del SMSWatch, la comunicación entre los distintos elementos y el almacenamiento de información sobre los distintos usuarios y parámetros de configuración.

1.2. Resumen de la memoria

Para ofrecer una lectura mas clara y accesible al lector, se ha dividido esta memoria en distintos capítulos:

- Menciones y agradecimientos.
- Capítulo 1: Introducción
- Capítulo 2: Base teórica.
- Capítulo 3: Metodología de desarrollo.
- Capítulo 4: Evaluación de alternativas.
- Capítulo 5: Desarrollo.
- Capítulo 6: Implementación.
- Capítulo 7: Experimentación.
- Capítulo 8: Conclusiones y trabajos futuros.

2. Base teórica

2.1. Tecnología GSM

2.1.1. Comunicación móvil

La comunicación móvil se produce cuando los dos o más terminales en los que se produce la comunicación están en movimiento y no están físicamente conectados a la red.

La comunicación móvil se produce de dos formas básicas:

-**La comunicación inalámbrica**: se produce entre dos equipos situados en un radio de acción el uno con el otro. Ejemplos que podemos encontrar el la comunicación por BlueTooth y por infrarrojos. El principal problema de este sistema es tener un radio de acción limitado, por lo que ambos equipos donde se produzca la comunicación deben estar cercanos le uno con el otro.

-**La comunicación celular**: se produce entre dos equipos situados dentro de una misma célula, de varios kilómetros de amplitud. Posee protocolos de comunicación para administrar varias peticiones al mismo tiempo. Una comunicación más lejana se produce gracias a la comunicación entre distintas células, lo que es una ventaja respecto a la comunicación inalámbrica.

La comunicación GSM se produce mediante la comunicación celular.

2.1.2. Topología de un sistema célular.

Un sistema celular esta formado por 4 componentes básicos, que son los que se detallan a continuación.

-**La unidad móvil**: que será el transmisor o receptor desde el que se recibe o se envía la información deseada. El canal o frecuencia por el que se comunican es designado por el MSC.

-**Células:** son las zonas cubiertas por el sistema de comunicación.

-**La estación de transmisión-recepción base (BTS, o Base Transceiver Station)**: se sitúa en la célula, y ejerce de intermediario entre la unidad móvil y el centro de comunicación.

-**El centro de comunicación móvil (MSC, o Mobile Switching Centrer)**: el es el encargado de administrar las llamadas entre distintos dispositivos, conectar las líneas telefónicas, conmutar las llamadas entre las distintas células, controlar el tráfico para las tarifas, y administrar la red inalámbrica.

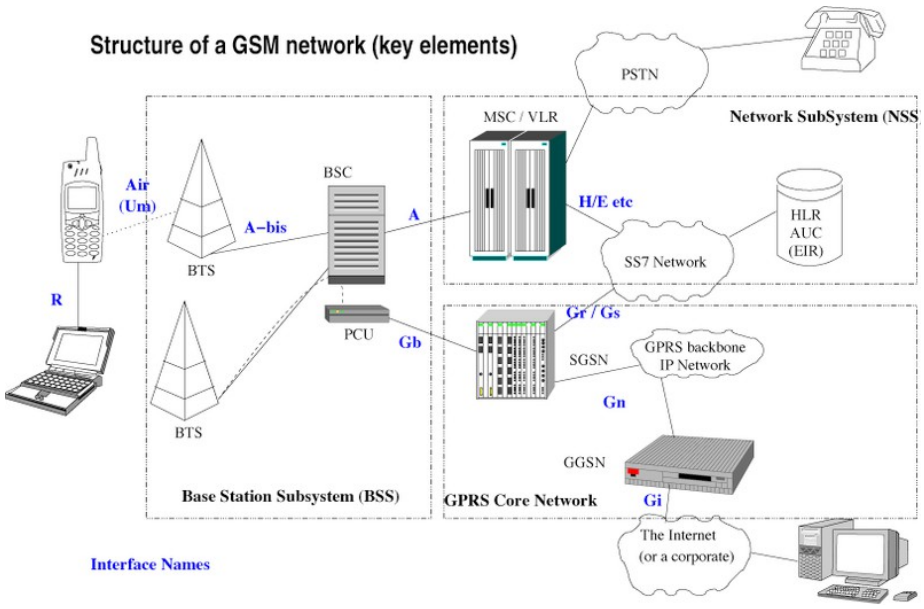

Figura 2.1. Arquitectura de GSM

No obstante, los sistemas celulares deben hacer frente a tres problemas: la cantidad de tráfico, las interferencias y la variedad de sistemas existentes.

En áreas muy pobladas, la cantidad de tráfico al mismo tiempo agota y satura los canales disponibles. Ejemplos de esto suceden en grandes urbes y en festividades como Año Nuevo.

Para solventar este problema, en un inicio se utilizan células más pequeñas y con menor potencia. Esto permite reutilizar las bandas disponibles en células no contiguas. Además, permite al proveedor cambiar o reducir el tamaño de estas células según varíe la densidad de población de la zona geográfica, lo que supone un ahorro a largo plazo.

En áreas urbanas muy pobladas, el volumen tan alto de tráfico local puede agotar los canales de radio disponibles. No obstante, es posible aumentar hasta cierto punto la capacidad del sistema reduciendo continuamente el tamaño de las células y la potencia transmitida de las estaciones base. La reducción en el radio de las células permite reutilizar las bandas disponibles en células no contiguas. La estrategia permite al proveedor de portadora celular reducir y aumentar el tamaño de las células para dar cabida al crecimiento o a la reducción de las poblaciones de esta base de suscriptores móviles.

Sin embargo, el principal problema al que se enfrentan las redes celulares es la aparición de interferencias.

Las interferencias se producen cuando se produce una intrusión en una banda de frecuencia no deseada, lo que puede producir varios efectos negativos: bloqueo de llamadas, errores, o las famosas *"cross-talks"* (en las que escuchamos inadvertidamente una conversación de otro canal). Esto es un problema principal en áreas urbanas, ya que la cantidad de ruido de otros dispositivos pueden formar cuellos de botella en las comunicaciones. Las interferencias producidas en los sistemas celulares son de dos tipos:

la co-canal y la de canales adyacentes.

La interferencia co-canal se produce cuando el transmisor o receptor está ubicado en dos células que utilizan el mismo rango de frecuencias al mismo tiempo. Esto produce una señal de ruido, ya que el sistema intenta propagar el mensaje por dos direcciones distintas. La solución más sencilla es separar estos sistemas co-canales, para reducir la interferencia entre ellos.

La interferencia de canales adyacente se produce cuando los filtros no bloquean el paso de una comunicación a otra de frecuencia cercana. Esto origina las "cross-talks", que son más frecuentes si los transmisores o receptores se encuentran cercanos físicamente. Para solventar esto, las estaciones base gestionan la potencia de las unidades móviles.

Por último, se tiene el problema de los distintos sistemas de comunicación según el área geográfica, lo que impedía la comunicación entre dispositivos que utilizasen sistemas distintos.

Con todo esto, se dio origen al sistema GSM como estándar mundial de comunicación.

2.1.3. GSM, historia e inicios.

El GSM se origino debido a los problemas de los teléfonos móviles de primera generación, los cuales hacían uso de redes analógicas. En 1982, se creo este estándar de comunicación gracias al *Groupe Spécial Mobile* (GSM), formado por 26 compañías europeas de telecomunicaciones, y que dio lugar a la digitalización de las comunicaciones.

Su implementación da lugar a la desaparición de la primera generación, y a las apariciones posteriores del 3G y el 4G (con acceso a internet y transmisión de datos, siendo el 4G la implementación más actual con ancho de banda). También pone fin a la incompatibilidad de sistemas de comunicación, y al envío y recepción de mensajes entre unidades portátiles.

2.1.4. Componentes del GSM

Los componentes principales del GSM son parecidos a los de las redes celulares, aunque incluyen otros componentes. Son los siguientes:

-**El centro de comunicación móvil (MSC):** es el encargado de establecer, gestionar y eliminar conexiones, así como de redireccionar las llamadas a la célula de destino correcta. Además, proporciona la interfaz con el sistema telefónico y determina la contabilidad del tráfico generado.

-**La célula**: el área geográfica, de unos 30 o 40 Km de alcance.

-**La unidad móvil (MS)**.

-**El controlador de estaciones base (BSC)**: incluido por primera vez en el sistema

GSM, se encarga del control de las llamadas realizadas, así del control de la señal de potencia en la transmisión entre la estación de transmisión y la unidad móvil.

-**La estación de transmisión-recepción base (BTS)**: ejerce de interfaz con la unidad móvil.

-**El HLR (Home Location register)**: es un registro que se encarga de recabar datos del usuario, y de los servicios que tiene contratados.

-**El VLR (Visitor Location Register)**: es un registro que recaba información sobre el estado de las estaciones y unidades móviles, así como la disponibilidad de los servicios complementarios.

-**El centro de validación (Authentication Center):** se encarga de proteger al usuario de accesos no autorizados o de invasiones al servicio contratado. Hace uso del HLR.

-**El registro de identidad del equipo (Equipment Identity Register)**: es un registro que recaba información del tipo de unidad móvil utilizado, y se usa para funciones como el bloqueo de equipos robados o no permitidos.

2.1.5. Funcionamiento de las llamadas.

En la siguiente figura, se puede observar cómo se produce una llamada GSM.

En los pasos 1 y 2, el usuario llama a la unidad móvil a través de la red telefónica, que llega al centro de comunicación (MSC). Luego, el MSC comprueba los registros del usuario en el HLR y VLR (pasos 3 y 4). Tanto el VLR como el HLR le devuelven la información al MSC y, si este comprueba que los registros son válidos, redirecciona la llamada al MSC receptor en el paso 6. El MSC receptor vuelve a comprobar la validez de los datos del usuario en el VLR (pasos 7 y 8). Una vez comprobados, envía en el paso 9 la notificación a la estación base pertinente, que a su vez se lo comunica a la estación móvil de destino. Por último, se envía una señal de confirmación a través del MSC hasta la red que envía la llamada para completarla (paso 10).

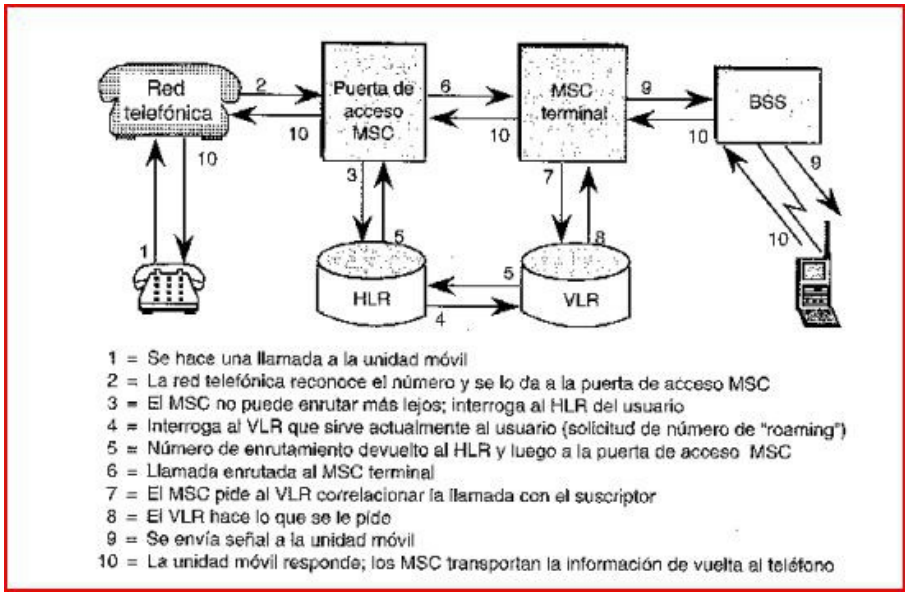

Figura 2.2. Comunicación GSM paso a paso

El sistema GSM también esta preparado en el caso de que se produzca una reubicación en una célula distinta a la anterior.

En este caso, la estación móvil envía su solicitud junto con su identificación a la estación base, que la redirige al MSC. El MSC comprueba en el registro VLR si esta el usuario (paso 1). En e caso de que el VLR no tenga información del usuario al ser nuevo, envía la información al HLR del usuario (paso 4), que elimina el VLR anterior y ubica el actual como él actualmente en uso. Una vez realizado esto, el VLR redirecciona la confirmación al usuario a través del MSC y de la estación base (paso 4).

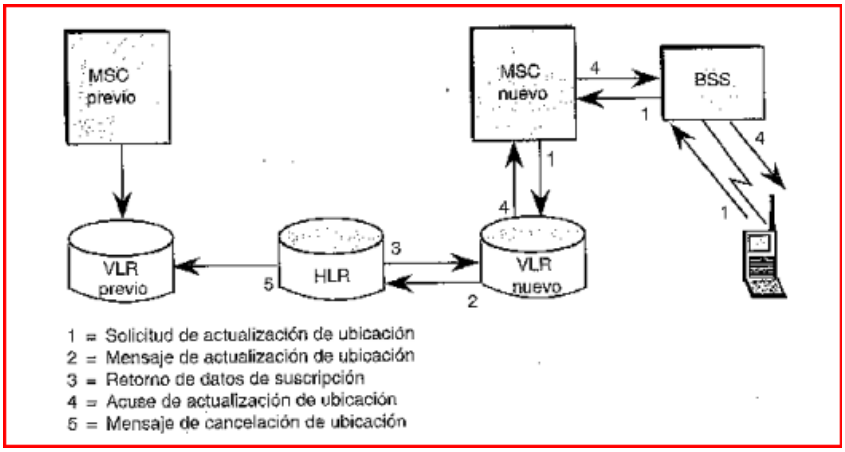

Figura 2.3. Funcionamiento en la reubicación

2.1.6. Interfaz GSM

GSM se diseñó de modo que permitiera la división en particiones funcionales. Dichas
particiones tienen sus fronteras en las diferentes interfaces que la componen. Estas son las siguientes:

- La interfaz A. Se encarga del MSC, HLR y VLR por un lado, y por el otro de BSC y de radio.

- La interfaz Abis. Defina la opercaión entre el BSC y la BTS.

- La interfaz de aplicación móvil, MAP (Mobile Application Part): se encarga de las opercaiones entre MSC y la red telefónica, así como el MSC, el HLR, el VLR y
el EIR. MAP se implementa encima de SS7.

– La interfaz de radio Um.

2.2. SMS.

2.2.1. Introducción

SMS es el término utilizado para el servicio de mensajes cortos (Short Message Service). Puede formarse a partir de palabras, números o combinaciones alfanuméricas, y su tamaño es de 160 caracteres para el alfabeto latino, y 70 caracteres para caracteres como el árabe o el chino.

Los SMS forman parte del estándar GSM y fueron introducidos en el año 1992.

Hoy en día, su uso es inmenso en todos los usuarios.

2.2.2. Características.

Según el estándar GSM, un mensaje corto tiene una capacidad de 160 caracteres. No obstante, también se pueden utilizar para enviar mensajes en otros formatos, como el binario y el hexadecimal. Un mensaje SMS se envía a través de un centro de SMS, en vez de directamente entre el transmisor y el receptor. El servicio de mensajes cortos posee además una confirmación de mensaje de salida. En cuanto el usuario envía el mensaje, recibe otro mensaje, que le indica si lo ha recibido el destinatario o no.

Otra característica de los mensajes cortos es que se pueden utilizar al mismo tiempo que la voz, el envío de datos y las llamadas de fax. Mientras que las llamadas, y la información de fax/datos viajan a través de un canal de radio dedicado durante la llamada, los SMS viajan a través de un canal dedicado independiente del normal.

En el caso de que se quieran enviar mas de 160 caracteres, existen dos métodos de envío distintos:

-**La concatenación**: que consiste en enviar varios mensajes cortos consecutivos.

-**La compresión**: que permite enviar un sólo mensaje de más de 160 caracteres.

Para utilizar los mensajes cortos se requiere de:

-Una suscripción a una red de telefonía móvil.

-Un teléfono móvil con capacidad de enviar SMS.

-Un destino válido para enviar el mensaje (fax, PC, e-mail u otro terminal móvil).

2.2.3. Arquitectura.

Al igual que los módulos GSM, los SMS están dotados de una red e infraestructura parecidas al estándar antes mencionado. Algunos de sus componentes son:

-**SME (Short Messaging Entity)**: se trata del dispositivo capaz de enviar y recibir mensajes cortos (teléfono móvil, PC, etc.)

-**SMSC (Short Message service Center)**: es el centro encargado del almacenamiento y transmisión de un mensaje corto entre dos dispositivos.

-**SMS-Gateway/Interworking MSC (SMS-GMSC)**: normalmente integrado en el SMSC, es un centro capaz de redirigir un mensaje corto, identificando al susuario mediante HLR y enviando el mismo al SMSC apropiado.

-**HLR (home Location Register):** es una base de datos que alamcena información sobre los usuarios y los servicios disponibles. Le indica al SMSC la ubicación del usuario destinatario del mensaje, o, si el receptor no esta disponible, avisa al centro cuando si lo está.

-**MSC (Mobile Switching Center)**: lleva a cabos funciones de conmutación y gestión del sistema.

-**VLR (Visitor Location Register):** da información al SMSC de los usuarios de paso, proporcionándoles permisos para el servicio de mensajes.

-**BSS (Base Station System)**: su principal uso es enviar y recibir tráfico entre terminales.

-**MS (Mobile Station)**: terminal móvil capaz de generar y recibir llamadas. El MS hace uso del MAP, que define los métodos de comunicación inalámbricos de la estación.

2.2.4. Envío de SMS

El MAP es la aplicación encargada de dar el soporte necesario al SMS. En el estándar internacional se le conoce por el nombre de GSM MAP.

Entre algunas de las operaciones del MAC se pueden hablar de:

-Solicitud de Información de Encaminamiento: en la que el SMSC extrae información del HLR para comprobar la ubicación del terminal antes del envío del SMS.

-Envío del Mensaje Punto a Punto: mecanismo que permite al SMSC enviar un mensaje corto al MSC destinatario. Se encarga también de transmitir la confirmación del envío al remitente.

-Indicación de Espera del Mensaje Corto: se activa cuando el envío falla por algún incidente. Esto hace que el SMSC solicite al HLR cuando el destinatario esta disponible.

-Alerta del Centro de servicio: hace que el HLR informe al SMSC que un destinatario previamente no disponible lo está ahora.

Hay que tener en cuenta también varios períodos: el de validación indica cuanto tiempo puede el SMSC almacenar el mensaje antes del envío, mientras la prioridad indica la posición del envío respecto a otros.

El procedimiento de envío de un SMS es como el de una llamada GSM. El mensaje se envía del SME al SMSC. Luego, el SMSC valida y localiza el destinatario con el HLR, y lo envía al MSC pertinente. A continuación, el MSC autentifica el destinatario con el VLR y envía el mensaje corto al MS. Por último, se envía el mensaje de confirmación de vuelta al SME.

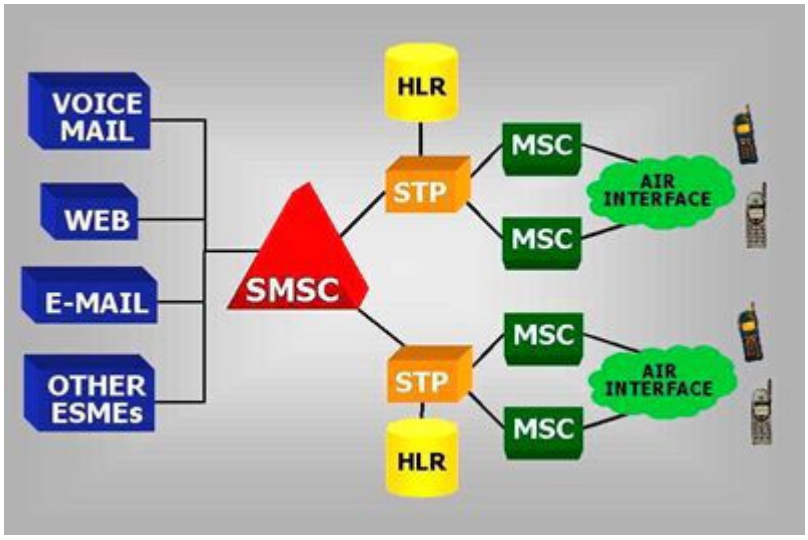

Figura 2.4. Envío de SMS

2.2.5. Aplicaciones

Entre las aplicaciones de los SMS se encuentran:

-Los mensajes simples entre personas, siendo de uso rutinario.

-Notificaciones de buzón de voz y fax.

-Alertas de correo electrónico.

-Servicios e información y descargas.

-Chat, emergiendo actualmente y sustituyendo al SMS convencional hoy en día.

-Otros usos: como la supervisión remota, almacenamiento de información, localización por GPS, etc.

Existen también distintas clases de SMSs, según su almacenamiento. La Clase 0 (no se almacena en memoria; Flash), la Clase 1 (se almacena en la memoria del teléfono), la Clase 2 (se almacena en la tarjeta SIM del teléfono), y la Clase 3 (se almacena en una aplicación externa del teléfono).

2.3. Smartcards

2.3.1. Introducción

Una smart card es un dispositivo con un chip de circuito integrado incorporado que puede ser tanto un microcontrolador, con o sin memoria, o un chip de memoria únicamente. La tarjeta se conecta con un lector ya sea mediante contacto físico o con una interfaz de radiofrecuencia a distancia. Con un microcontrolador integrado, una smartcard puede: almacenar datos, ejecutar sus propias funciones (encriptación, autentificación mutua) e interactuar con el lector de tarjetas.

Figura 2.5. Arquitectura de una Smartcard

La tecnología smart card esta realizada conforme a los siguientes estándares internacionales, ISO 7816 y 14443, y esta disponible en una amplia variedad de formas: tarjetas de plástico, SIMs de móviles, y piezas basadas en comunicación USB.

2.3.2. Tecnología

Hay dos categorías generales de smart cards: **con o sin** contacto.

Una smart card de contacto debe estar insertada en un lector de tarjetas con una conexión directa mediante placas conductivas en la superficie de la tarjeta (generalmente placas doradas). La transmisión de comandos, datos y estado de la tarjeta tienen lugar a través de estos puntos físicos de contacto.

Una tarjeta sin contacto requiere únicamente una proximidad cercana al lector. Tanto el lector como la tarjeta poseen una antena, y los dos se comunican usando radiofrecuencias (RF). Muchas tarjetas sin contacto suelen proporcionar energía al chip

interno gracias a esta señal electromagnética.

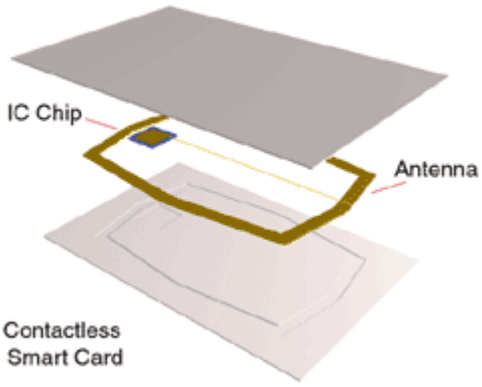

Figura 2.6. Arquitectura de Smartcard (2)

La distancia varía entre 1 y 7 cm para las tarjetas sin batería, y su uso es ideal para accesos a edificios y pagos que requieren una interacción rápida con la tarjeta.

Otras dos categorías adicionales de tarjetas son las tarjetas de interfaz dual y las tarjetas híbridas. Una tarjeta híbrida posee dos chips, uno con interfaz de contacto y el otro con interfaz sin contacto. Los dos chips no están interconectados. Una tarjeta de interfaz dual tiene un único chip que posee tanto interfaz por contacto como sin contacto. Con las tarjetas de interfaz dual, es posible acceder al mismo chip tanto con o sin contacto con un nivel muy alto de seguridad.

Los chips usados en todas estas tarjetas caen en dos categorías también: chips microcontroladores y chips de memoria.

Un chip de memoria es como un pequeño disco floppy con seguridad opcional. Los chips de memoria son mas baratos que los microcontroladores pero con un decremento añadido de seguridad en el manejo de datos. Las tarjetas que utilizan chips de memoria dependen de la seguridad del lector de tarjetas para procesar y son ideales para situaciones con seguridad media o baja.

Un chip microcontrolador puede añadir, borrar y manipular información de su memoria. Un microcontrolador es como un ordenador en miniatura, con un puerto de entrada/salida, un sistema operativo, y un disco duro. Las smart cards con un microcontrolador integrado tienen la habilidad única de guardar grandes cantidades de datos, llevar a cabo sus propias funciones (encriptación y firmas digitales) e interactuar de manera inteligente con el lector de tarjetas.

CONTACT DESCRIPTION

Pin#	Name	Function
C1	Vcc	Power Supply
C2	MCLR	Master Clear
C3	RB6/Osc1	Clock Input
C4	N/C	No Connect
C5	Vss	Ground
C6	N/C	No Connect
C7	RB7	Data I/O
C8	N/C	No Connect

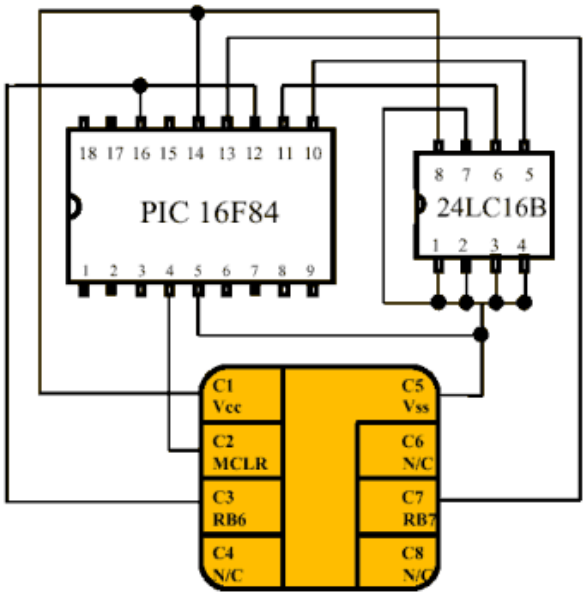

Figura 2.7. Conexiones de una Smartcard

3. Metodología y materiales elegidos.

3.1. Metodología

En el diseño de sistemas se suelen seguir una serie de pasos que van desde la especificación de requisitos hasta la comercialización del producto final. El presente proyecto tiene como objetivo final la comercialización de un prototipo, por lo que el ciclo de diseño no abarcará hasta la fase final de fabricación en serie del producto y posterior comercialización. Este ciclo de diseño se ha adaptado a las características del proyecto y constará de las fases de especificación de requisitos, establecimiento de arquitectura del sistema, desarrollo del hardware y software, test y debug, y , por último, diseño y fabricación. A continuación se detallan estas fases:

- **Especificación de requisitos:** En esta fase se establecen los requisitos que se quiere que cumpla el producto a desarrollar. Así pues, se tendrán que especificar parámetros tales como coste máximo, el rendimiento mínimo, las ampliaciones futuras, el consumo de energía, tamaño o peso.

- **Establecimiento de la arquitectura del sistema:** se tomará en cuenta el microcontrolador como elemento primordial del sistema. También habrá que tener en cuentas los puertos para las conexiones con el resto de elementos del sistema (pantalla, teclado, módulo GSM, etc.).

- **Desarrollo del hardware y software:** Una vez se tiene claro lo que se quiere hacer, es el momento de empezar a desarrollarlo y para ello, en primer lugar, se hace una selección de componentes a utilizar en la placa, realizando también el equema de la circuitería. Hay que tener en cuenta que en el ciclo de diseño hardware se pueden dar situaciones de diseño-prototipo-testeo-vuelta atrás, con el consiguiente encarecimiento del producto final. Por esta razón, se deja la fase de diseño y fabricación del PCB para el final, desarrollando antes una primera placa de prueba que servirá para verificar en la siguiente fase de testeo tanto el correcto diseño de la circuitería como el firmware del microcontrolador.

- **Test y debug:** Una vez comprobado el desarrollo del hardware y firmware se procede a comprobar su correcto funcionamiento. Para ello, se somete al sistema diversas pruebas para comprobar su respuesta ante ciertas acciones que provocaremos en él. Todo con tal de garantizar su funcionamiento.

- **Diseño y fabricación:** Por último, una vez que se han comprobado y aprobado los pasos anteriores, se procederá a diseñar la placa del circuito para su uso.

- **Testeo y experimentación:** Una vez montado el circuito, se testeará otra vez para comprobar que todo funciona en orden.

3.2. Software de desarrollo

Para la realización del proyecto se han utilizado varios programas de software, cada cual cumple un función determinada, desde el desarrollo del PCB hasta la programación del microcontrolador.

Los programas a utilizar son los siguientes: Orcad 9.2, CCS Compiler (ver. 4.1), Proteus 7 Professional, MPLAB IDE 8.5. Su funcionamiento y participación en el proyecto se detalla a continuación.

Orcad 9.2

El programa Orcad es un programa de desarrollo de circuitos electrónicos. Consta de dos programas principales. Orcad Capture es un programa que permite diseñar el esquemático del circuito, además de simularlo virtualmente. Orcad Layout permite el desarrollo de PCBs para la posterior fabricación de las mismas.

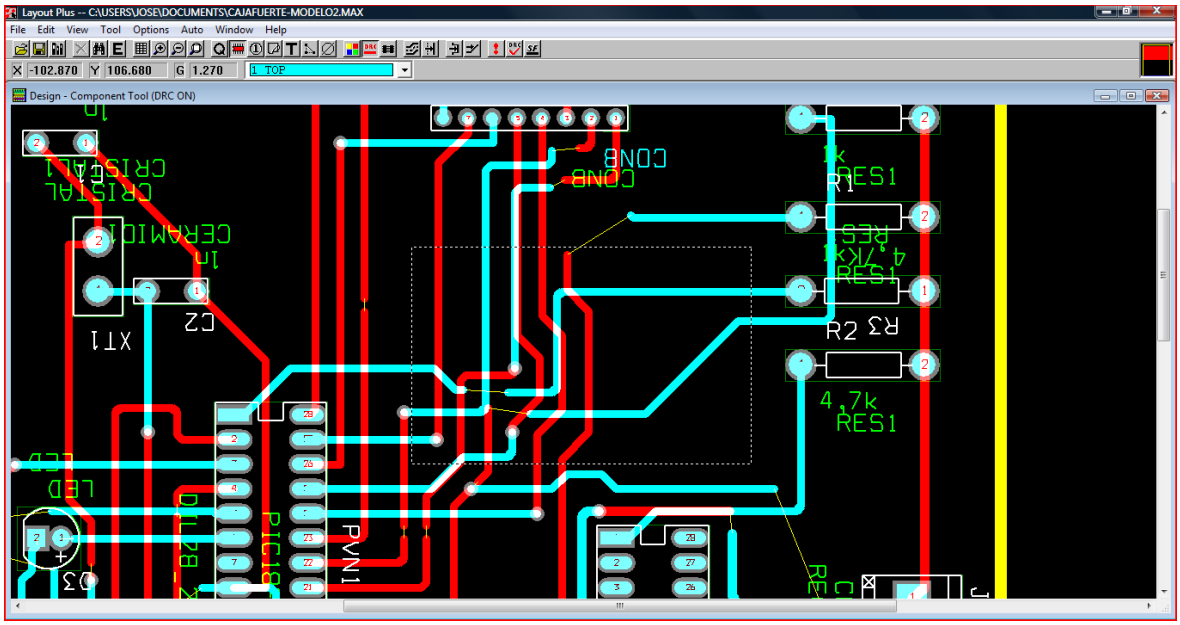

Figura 3.1. Ventana de Orcad

Este programa se usará principalmente para desarrollar el PCB del proyecto, ya que es un programa de gran difusión y cuyos diseños son compatibles en otros programas varios. Para ello, realizaremos el esquemático en Orcad Capture, y posteriormente lo pasaremos a Layout, donde seleccionaremos la ubicación física de los componentes y sus conexiones en la placa.

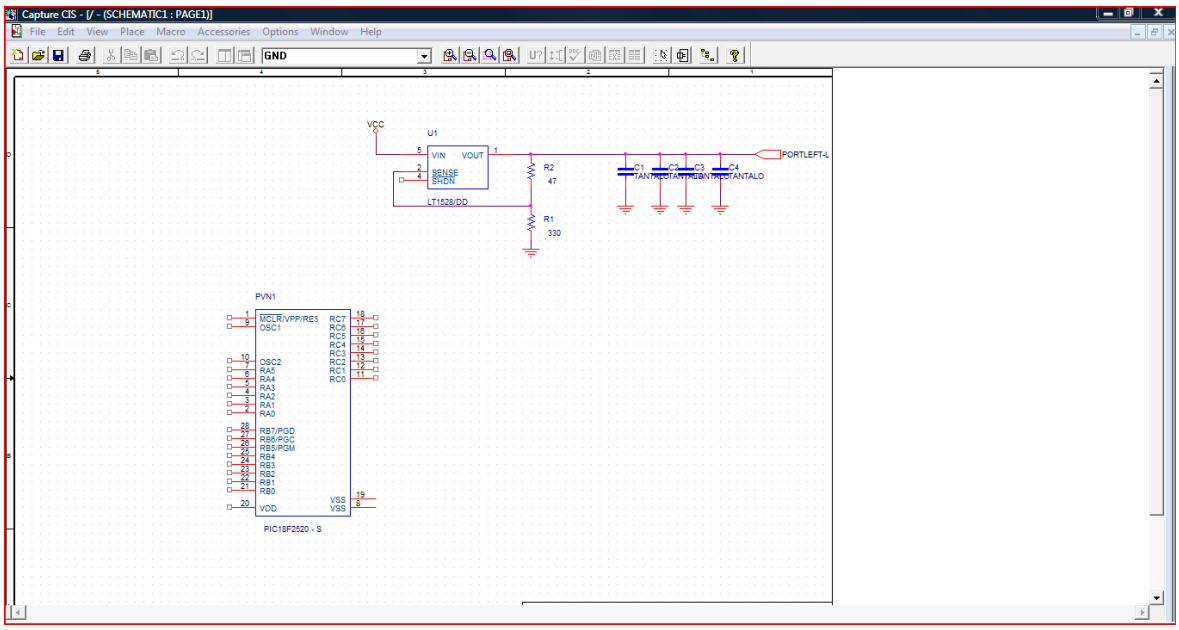

Figura 3.2. Ventana de esquemático en Orcad

CCS Compiler (ver. 4.1)

Nuestro principal programa para desarrollar el software del microcontrolador es el compilador de CCS. Este software nos permite hacer la programación del 18F2520 tanto en lenguaje ensamblador cómo en C. Además, tiene varias características que facilitan su manejo, cómo el uso de librerías tanto para microcontroladores así como los periféricos que vayamos a añadir a los mismos, así como notificación de errores y de uso de memoria en el programa del PIC.

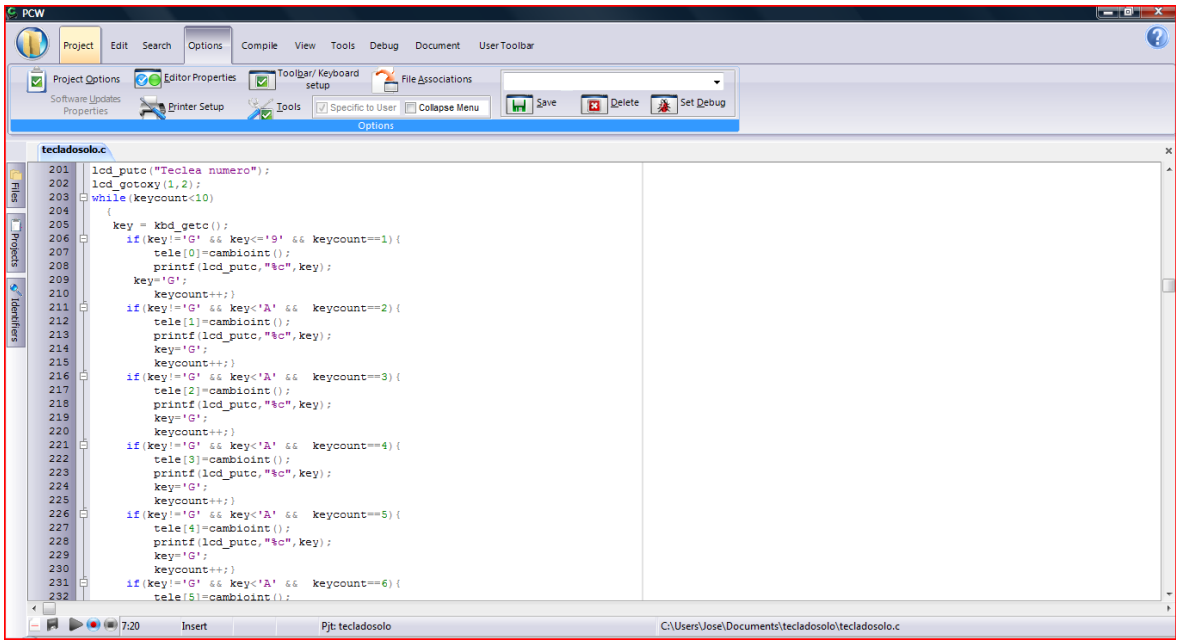

Figura 3.3. Ventana de CCS Compiler

Para este proyecto, realizaremos la programación en C al ser más sencilla. También haremos uso de librerías para el 18F2520 (que se ocupan de configurar el programa para el microcontrolador), así como para los periféricos del proyecto más complejos, como la pantalla LCD, la EEPROM de las tarjetas y el módulo GPS.

Proteus 7 Professional

Al igual que Orcad 9.2, Proteus 7 Professional es un programa de desarrollo y simulación de placas electrónicas. Posee dos programas principales, Isis y Ares. Isis es un programa de diseño de esquemas electrónicos y capaz de simular los mismos virtualmente. Por otro lado Ares es el encargado de desarrollar PCBs de los esquemáticos realizados en Isis, así como poder visualizarlos en 3D para comprobar la apariencia final que tendría la placa electrónica.

Nuestro uso de Proteus 7 de manera conjunta con Orcad se debe a que el primero es mucho más intuitivo, completo y sencillo en cuanto a la simulación de componentes electrónicos. Con Isis, podemos comprobar personalmente el funcionamiento del programa del microcontrolador antes de trasladarlo al entrenador o a la misma placa, lo que permite modificarlo inmediatamente si observamos un comportamiento no deseado en el transcurso de la simulación.

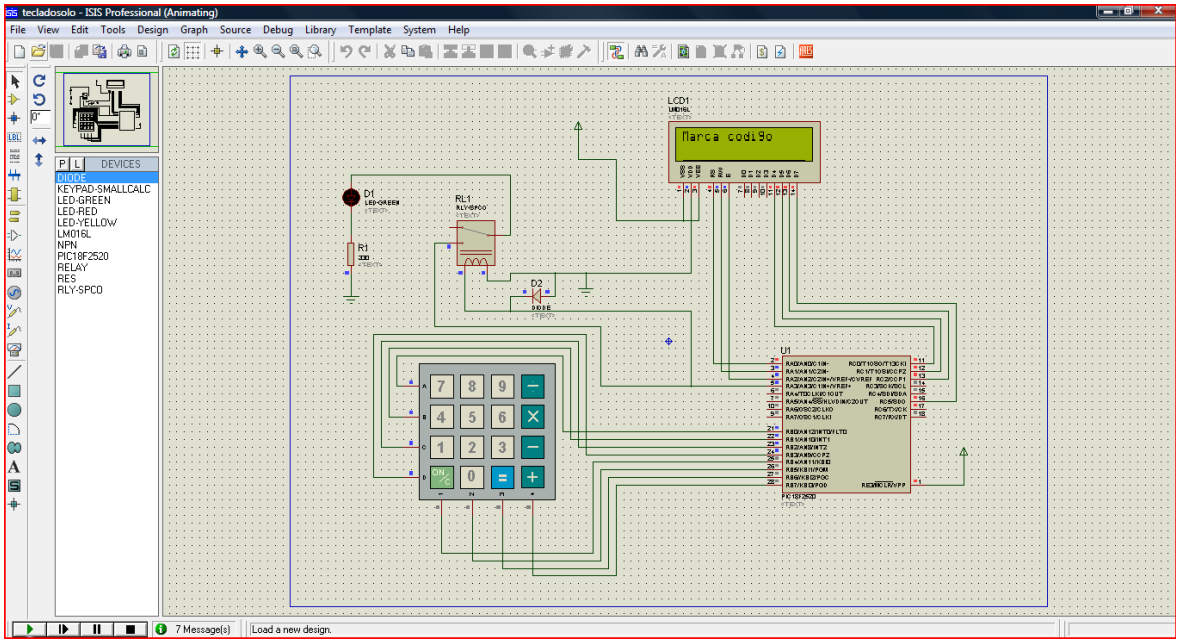

Figura 3.4. Ventana de Isis (Proteus)

No obstante, haremos uso de Orcad para el desarrollo del PCB, ya que Ares no es tan fiable en cuanto a compatibilidad y estándares.

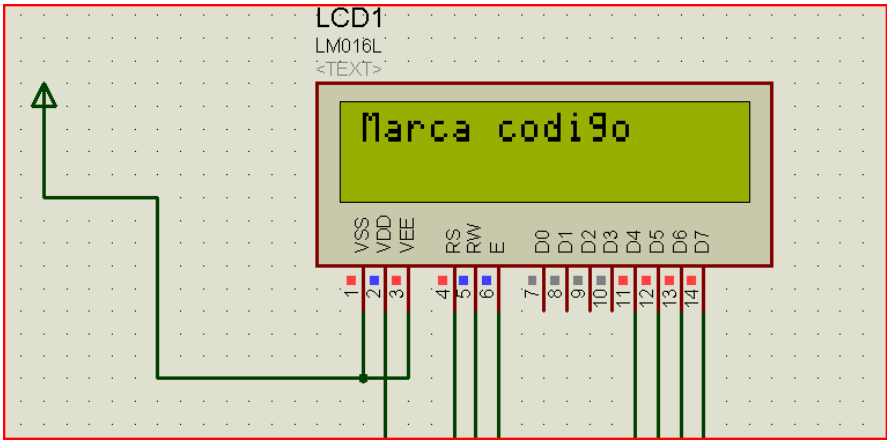

Figura 3.5. Simulación en Isis

MPLAB IDE 8.5

MPLAB IDE 8.5 es un programa destinado a la programación de dispositivos de Microchip, como microcontroladores (PIC y dsPIC).

Su uso principal en el proyecto es a la programación y depuración del código del microcontrolador 18F2520. Gracias a un add-on de CCS Compiler, podemos trasladar rápidamente el código de un programa a otro sin complicaciones. MPLAB IDE 8.5 esta diseñado para usar varios kits de depuración como el PicKit.

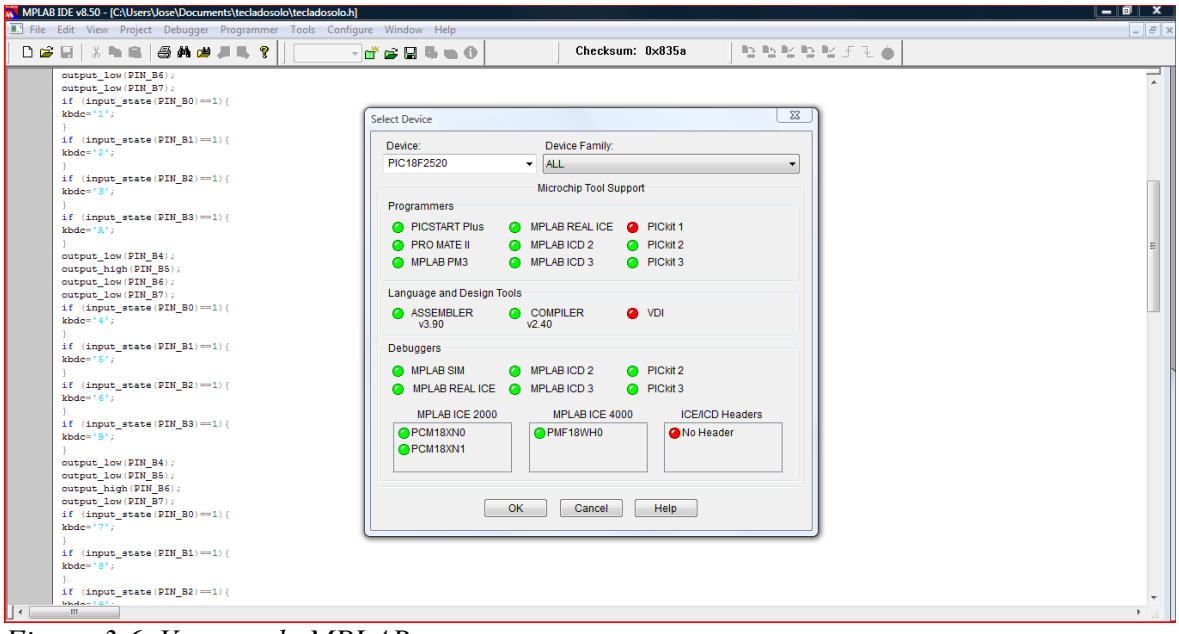

Figura 3.6. Ventana de MPLAB

Esto lo conseguimos con un debugger de hardware llamado ICD3, que permite programar nuestro microcontrolador da manera sencilla. El ICD3 posee la licencia de MPLAB para la programación del microcontrolador, y es totalmente compatible con el

programa. Además, incorpora puerto USB, lo que permite utilizarlo en cualquier ordenador de sobremesa o pórtatil sin recurrir a viejos puertos.

Figura 3.7. ICD3

3.3. Hardware de desarrollo

Para desarrollar el proyecto expuesto, se necesitarían de varios elementos. Por un lado, se requería de un ordenador con el software antes descrito para la programación y el debug del prototipo. Por otro, necesitábamos una placa de desarrollo en donde poder realizar las pruebas y el testeo del programa, así como para aplicar fácilmente los cambios requeridos. Por último, requeríamos de los componentes que pasarían a formar parte del proyecto en sí (microcontroladores, perífericos, etc).

Todo el firmware del proyecto se realizó en un portátil Intel Core 2 duo 2Ghz y 4GB de memoria RAM, con un sistema operativo Windows Vista. Las pruebas de compatibilidad se efectuaron satisfactoriamente en un Pentium 4 de sobremesa con Windows XP (el cual se utilizó en las primeras fases del proyecto antes de tener acceso al ICD3, ya que el portátil carecía de los puertos necesarios para conectarlo a la protoboard).

El montaje, programación y testeo del prototipo se realizó en una placa de desarrollo llamada PICSchool, preparada con elementos como pantalla LCD, teclado alfanumérico, conexión para varios tipos de microcontrolador y una protoboard para conexiones de cableado.

La programación del prototipo se realiza con un debugger de Microchip (el ICD3), que se conecta al ordenador mediante puerto USB, y al PicSchool mediante un cable conectado al puerto ICSP de la placa de prueba

4. Análisis de alternativas.

En este capítulo se muestran los requisitos necesarios para una realización adecuada del proyecto y su funcionamiento. Asímismo, se determinarán las alternativas disponibles tanto a la hora de determinar el tipo de hardware usado, como de la programación del dispositivo. Por último, se elegirán las opciones mas adecuadas para usarlas en el dispositivo y alcanzar los objetivos propuestos.

4.1. Requisitos.

Los requisitos ha cumplir en el proyecto para los objetivos propuestos se dividrián en dos grupos: requisitos de perífericos y requisitos generales del sistema.

Se requiere que el dispositivo sea del menor tamaño posible y de fácil manejo. Se necesita además que cumpla ciertos requisitos en las conexiones disponibles para él a la hora de interactuar con los perífericos. Por otro lado, el dispositivo debe de ser capaz de tener una alimentación independiente, ya que facilita a posteriori la instalación y adaptación del mismo.

Para lograrlo, el dispositivo debe de tener compatibilidad con los siguientes elementos:

- Una fuente de alimentación de 5 V, con un adaptador de tensión adecuado según la alimentación seleccionada.

- Una batería removible, preparada para proporcionar los 5 V requeridos en el dispositivo. La necesidad de esta batería nace de las posibles desconexiones del dispositivo de la red eléctrica.

El dispositivo también tiene que tener la capacidad de comunicarse mediante tecnología GSM. Esto es necesario para poder enviar los SMS que comunicarán el dispositivo con el usuario a distancia. También se requiere de un método adecuado para la identificación del usuario con el dispositivo.

El dispositivo debe de tener el número adecuado de sensores/actuadores, que nos permitan visualizar y manipular el funcionamiento del dispositivo. Debe tener también un uso intuitivo del mismo, ya que su público objetivo es cualquier usuario sin conocimientos previos de la electrónica.

Por otro lado, es necesario que el dispositivo este preparado para actualizaciones futuras de contenido, por lo que sus conexiones deben de ser lo más versátiles posibles. También debe de existir la posibilidad de que el dispositivo se pueda reprogramar de manera sencilla.

4.2. Evaluación y selección de alternativas.

Una vez determinadas las necesidades que requiere nuestro proyecto, se pasan a analizar las diferentes alternativas disponibles así cómo el porqué de nuestra decisión final.

4.2.1. Comunicación por SMS.

Siendo la comunicación a distancia con el usuario uno de los motivos más importantes para el desarrollo del proyecto, era necesario adoptar esta elección en primer lugar. De una elección u otra dependerá el diseño de todo el proyecto.

En el caso de un dispositivo capaz de comunicarse con SMS, hay una gran variedad de modelos de móviles disponibles en el mercado (Ericsson, Nokia, Motorola, etc.). No obstante, casi todos los modelos actuales presentan el problema de que requieren de aplicaciones móviles y arquitecturas complejas para poder enviar órdenes a los mismos. Por esa misma razón, la totalidad de los dispositivos móviles actuales fueron descartados, ya que, aún pasando por encima de la barrera tecnológica, quedaba la barrera financiera, ya que un terminal móvil actual tiene demasiado precio en este proyecto para la cantidad de funciones que ejercería.

Estaba claro que para poder elegir el dispositivo de comunicación, había que elegir primero un estándar de comandos para poder enviar al mismo. Esto nos permitiría por un lado tener un fácil uso del dispositivo a través del microcontrolador, y por el otro, tener la facilidad de intercambiar el dispositivo por otro compatible con el estándar de manera directa.

De entre varios tipos de instrucciones, se decidió usar los comandos Hayes, más comúnmente conocidos como comandos AT. El conjunto de comandos Hayes fue desarrollado por *Hayes Communications* para la transmisión de comandos a módems. Esto se consigue mediante la transmisión de cada comando por RS232, precedidos por el prefijo "AT".

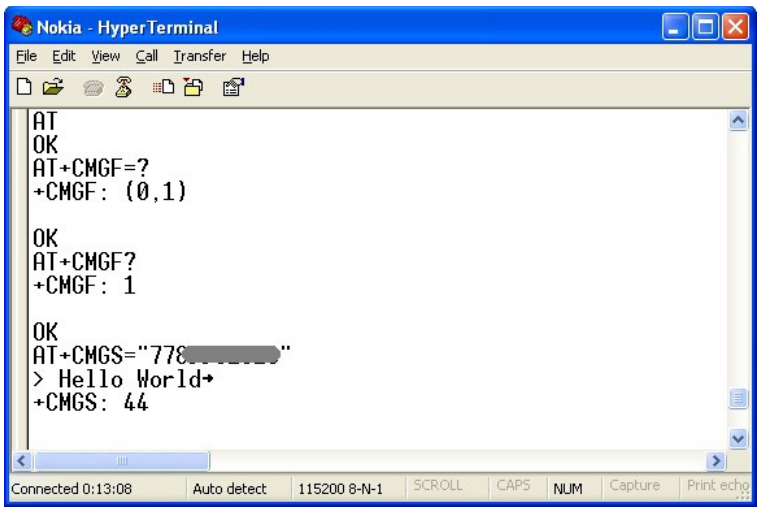

Figura 4.1. Comandos Hayes

Una vez determinado el lenguaje utilizado para la transmisión, había que localizar un terminal compatible con ese lenguaje. Desgraciadamente, muchos módems y móviles de hoy en día no son compatibles con los comandos AT (o al menos, no de forma directa), por lo que las opciones se limitaban a terminales móviles antiguos o a módulos GSM.

La opción del terminal móvil era sencilla, conectando a través de un cable Null Modem (cable cruzado) el móvil al dispositivo. El problema radicaba en que los terminales eran muy antiguos, y por lo tanto, estaban fuera de la lista posible. En el caso de un proyecto personal eran una opción adecuada, pero en este caso, y sabiendo que esta dirigido a un uso general no era adecuado hacerlo.

Figura 4.2. Móvil GSM

Esto nos dejaba una última opción: el uso de módulos GSM para la transmisión de mensajes SMS.

Un módulo GSM es un terminal pequeño para una tarjeta SIM (necesaria y que requiere de servicio de telefonía), que puede enviar y recibir tanto mensajes como llamadas. Los módulos GSM se rigen por el estándar GSM.

Dado la inmensa cantidad de módulos disponibles en el mercado, y la variedad de precios entre ellos, casi cualquier módulo compatible con los comandos Hayes era adecuado para utilizarlo en el proyecto.

Las compañías que producen estos módulos son varias: Citerion, Matrix, Telit, etc. Para el desarrollo de este prototipo en específico, se decidió recurrir a la marca Telit, específicamente el modelo GC864- PY.

Figura 4.3. GC864-PY

En la siguiente tabla las características de banda del mismo.

Mode	Freq. TX (MHz)	Freq. RX (MHz)	Channels (ARFC)	TX – RX offset
GSM-850	824.2÷848.8	869.2÷893.8	128 ÷ 251	45 MHz
E-GSM-900	890.0 – 914.8	935.0 – 959.8	0 – 124	45 MHz
	880.2 – 889.8	925.2 – 934.8	975 – 1023	45 MHz
DCS-1800	1710.2 – 1784.8	1805.2 – 1879.8	512 – 885	95 MHz
PCS-1900	1850.2 – 1909.8	1930.2 – 1989.8	512 – 810	80 MHz

Tabla 4.1. Modos de transmisión GC864-PY

4.2.2. Microcontrolador.

Para elegir el microcontrlador adecuado, es conveniente conocer los requisitos previos para el mismo. Entre esos requisitos conviene destacar los siguientes.

- Procesador: Por cuestiones de economía, se requiere que el dispositivo sea lo más sencillo posible en cuestiones de procesamiento de datos. Es importante por ello elegir un dispositivo lo bastante rápido para las funciones que ejercerá, pero con el menor número posible de bits. En este caso, nos fijamos un microcontrolador de 8 bits, o 16 si es absolutamente necesario.

- Entrada/Salida: Debido a la cantidad de perífericos que comunicarán con el microcontrolador, es necesario que éste tenga la cantidad de puertos necesaria para conectarse a todos los dispositivos sin necesidad de acudir a elementos tales como multiplexores y entradas analógicas. También debe de tener la capacidad obligatoria de permitir la transmisión I2C (para las tarjetas Smartcard) y la RS232 (para el módulo GSM).

- Consumo: Estando el dispositivo continuamente encendido a la espera de comunicarnos con él, es permisible que funcione a 3.3V o a 5V 8siendo este último

valor preferido debido a la alimentación del resto de periféricos).

- Memoria: Se recomienda que la memoria sea duradera y permita una gran cantidad de borrados y rescrituras. En este caso, se prefiere una memoria tipo Flash, a una memoria ROM 8menos recomendable cuando se necesita memoria no vólatil).

- Ancho de palabra: lo más recomendable en este caso es un ancho de palabra de 8 bits, por encima del valor de 4 y por debajo de los innecesarios 16 bits de ancho. También es el valor más utilizado en los microcontroladores.

Los fabricantes de microcontroladores son variados, siendo los dos más comunes Atmel y Microchip.

Se optará finalmente por Microchip sobre Atmel por varios motivos:

- Son fáciles de programar y usar. Lo que elimina carga al desarrollo del proyecto y a la programación de los diversos elementos interaccionando con el periférico.

- Posee una mayor variedad de dispositivos disponibles en el mercado (más de 150 distintos, más sus variantes según el encapsulado).

- La programación en ellos es familiar, dando ventaja sobre un microcontrolador del cual nunca se ha programado antes.

De entre varios modelos que encajaban con los requisitos antes descritos, se eligió al final el microcontrolador Flash PIC18F2520, con suficiente memoria para el tamaño del programa, y con la suficiente cantidad de puertos disponibles (más de 24) para conectar todos los periféricos necesarios al mismo.

En la imagen se adjunta un detalle de las conexiones del mismo.

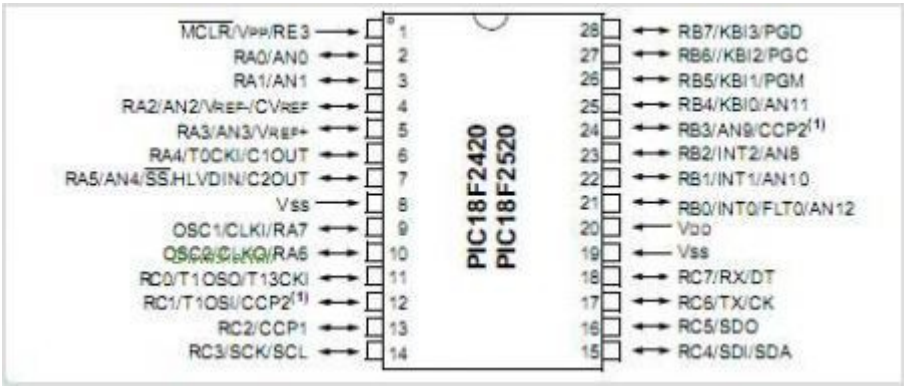

Figura 4.4. Conexiones del 18F2520

Una vez determinados los dos elementos principales, se procede al diseño e implantación del proyecto.

5. Diseño.

5.1. Introducción.

En las siguientes páginas se mostrará los pasos seguidos para el desarrollo del prototipo del SMSWatch. Desde su concepto inicial, pasando por los distintos añadidos y problemas presentados durante el desarrollo.

5.2. Diseño del hardware.

La primera decisión fue comenzar con el desarrollo del elemento con el que interactuará el módulo GSM. De entre varias opciones disponibles, se decidió por crear un elemento de seguridad con clave de acceso e identificación personal.

Para ello, hicimos uso del un microcontrolador como elemento central de control. En la actualidad existen muchos modelos disponibles tanto por Microchip como otras empresas como Atmel. Debido a la familiaridad de uso, se decidió por usar un modelo de microchip 18F, en concreto el PIC18F2520.

La elección del 18F2520 sobre otros modelos respondía a varias necesidades. En primer lugar, se necesitaba un microcontrolador con una variedad de puertos y opciones. En segundo lugar, había que elegir un modelo con suficiente memoria para el firmware a instalar. En tercer y último lugar, el modelo debía ser relativamente barato.

La solución me la presentó el tutor del proyecto desde un principio, ya que tenía varias unidades de un modelo que se adecuaba a los requisitos anteriores.

Figura 5.1. PIC18F2520

Entre otras cosas, el microcontrolador poseía:

-Una frecuencia de trabajo de hasta 40MHz, ideal para una relativa rapidez del firmware.

-3 puertos completos de 8 pines de conexión cada uno, la cantidad necesaria para el número de conexiones a realizar.

-Capacidad para comunicarse por I²C, ideal para enviar y recibir datos a y desde las

tarjetas SmartCard.

-Comunicación para RS232, independiente del I²C y que se usaría para comunicarse con el módulo GSM mediante los puertos Tx y Rx.

Una vez seleccionado el microcontrolador necesario, se procedía a la asignación de puertos a cada uno de los elementos con los que se comunica.

La totalidad de los puertos B0 a B7 fueron asignados al teclado alfanumérico, ya que se requerían de conexiones para las 4 filas y las cuatro columnas. En un principio se probó a utilizar una sola entrada analógica, combinando resistencias en los puertos del teclado, pero se descartó la idea al ser la señal susceptible de ruido.

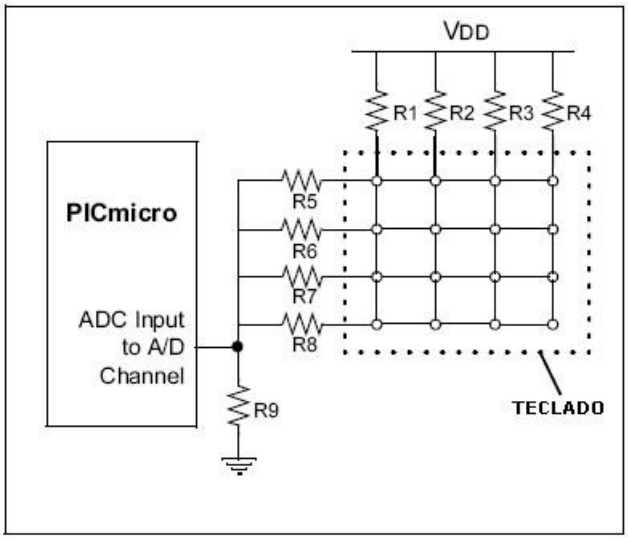

Figura 5.2. Teclado alfanumérico

Otro método que se probó fue el de usar un multiplexor para reducir el número de puertos de 8 a 6 (ya que aún habiendo 4 filas y 4 columnas, se requería también el estado de apagado, lo que llevaba a tres puertos para las filas y 3 para las columnas). La idea se descartó al poco al no ser necesaria una redistribución de los puertos y al no ser viable económicamente. Finalmente, se optó por la idea inicial de usar los ocho conectores del puerto B.

Figura 5.3. Teclado alfanumérico (2)

Otro problema que surgió con el teclado era que no funcionaba con una conexión directa de los puertos. En un primer intento, se intentó usar los pull-ups internos del microcontrolador pero no daban resultado ninguno. La solución tuvo que venir con añadir 4 resistencias como pull-downs para que el mismo puerto pudiese tanto enviar como recibir señales. Otro elemento utilizado era una pantalla LCD de 16x2 caracteres. Este elemento haría las veces de interfaz física con el usuario, serviría de notificación visual y facilitaría la configuración tanto del elemento de seguridad como la comunicación con el módulo GSM.

La pantalla LCD (cristal líquido) ofrece una matriz de caracteres que podemos transmitir, así como un microcontrolador interno encargado de la visualización en la pantalla. Las pantallas LCD ofrecen un consumo reducido, y pueden ser progamadas con 8 o 4 puertos de bits de datos, aparte de 3 bits de control. Los bits de control son el RS, el Enable y el Read/Write. Mediante el puerto R/W elegimos si queremos transmitir información a la pantalla o leerla. El Enable permite la transmisión de datos, y el RS actúa para seleccionar si estamos enviando un dato o un comando.

El módulo LCD posee una zona de memoria RAM llamada DDRAM (Data Display RAM) donde se almacenan los caracteres que se van a mostrar en la pantalla. Tiene una capacidad de 80 bytes, 40 por cada línea, de los cuales sólo 32 se pueden visualizar a la vez (16 bytes por línea). Además, posee una zona de memoria interna no volátil llamada CGROM. En esta zona de memoria, están almacenados los 192 caracteres que puede representar la pantalla, cada uno determinado por un número binario de 8 bits. Por otro lado, si lo deseamos podemos introducir 8 caracteres personalizados en la zona CGRAM.

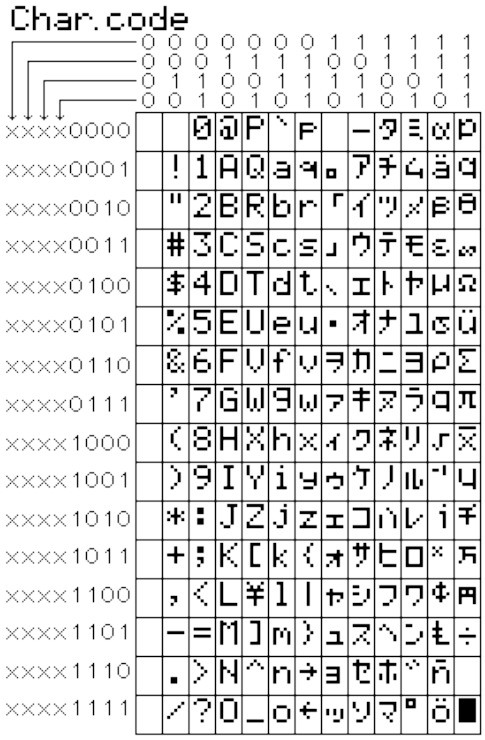

Tabla 5.1. caracteres ASCII

Para la comunicación con el lector de tarjetas SmartCard, necesitábamos una conexión I2C maestro-esclavo. En este caso, utilizaríamos al microcontrolador como maestro y a las tarjetas EEPROM conectadas como esclavas.

En un principio se temía que al conectar las tarjetas estas usasen los mismos puertos necesarios para la comunicación entre el módulo GSM y el microcontrolador (los puertos de transmisión serie RS232, Tx y Rx). No obstante, no fue el caso, ya que, aparte de la conexión a masa y a la alimentación, las tarjetas sólo necesitaban un puerto para la transmisión de datos, y otra de reloj, ambas controladas por el maestro.

Aprovechando las conexiones disponibles en el puerto A, se utilizó una para conectar a un relé electromagnético. Un relé electromagnético consiste en un inducido que puede conectarse a otros dos contactos magnéticos (uno a cada lado). Por defecto, se encuentra conectado en uno de los contactos (el que utilizaremos para indicar el cierre). Cuando entre los dos terminales de magnetización administramos una corriente, el contacto que se encuentra desconectado se magnetiza, atrayendo al inducido, que se desconecta del otro contacto y se junta con éste (el cual utilizaremos para indicar el estado de abierto). Una vez dejemos de aplicar corriente, el relé electromagnético vuelve a su estado por defecto (cierre).

Un relé electromagnético es el adecuado en estos casos, ya que gracias a su

aislamiento podemos trabajar con elementos de otros voltajes sin peligro para la integridad del dispositivo (12V por ejemplo). Debido a que los únicos modelos disponibles a mano eran un relé de 6V (no podíamos generar tensión suficiente) y uno de 3V, se optó por usar este último con ayuda de un divisor de tensión.

Para indicar los estados de cierre y apertura utilizaremos un led rojo y verde respectivamente.

Por último, había que conectar el módulo GSM al microcontrolador. De entre todos los dispositivos, el módulo GSM era el que más problemas ocasionaba.

Por un lado, teníamos una alimentación distinta, de 3.7V, que requería de un adaptador a partir de 5V. Las opciones eran varías: un divisor de tensión, una fuente de alimentación regulada lineal, etc. Para evitar problemas posteriores con el módulo (ya que era el elemento más caro del prototipo), se decidió olvidar cualquier adaptador que no tuviese las protecciones necesarias ante incidencias puntuales. Posibilidades tales como el divisor de tensión fueron inmediatamente descartadas.

Finalmente, para resolverlo se decidió por utilizar dos métodos: uno temporal y otro definitivo para el prototipo final. El método temporal se usaría durante el testeo del prototipo, y estaría formado por una batería de 3.7V de Ion-Litio.

El método definitivo se haría con un regulador de tensión. Para elegir el más adecuado, hicimos uso de una aplicación de Linear Technology llamada Ltspice IV. Esta aplicación era adecuada para comprobar el funcionamiento de los distintos reguladores de tensión, y para monitorizar la salida del regulador antes de su implantación en el equipo.

Tras probar varios reguladores lineales, se eligió finalmente el LT1528 (ya recomendado en el datasheet del fabricante del módulo GSM). Con una entrada de 5V, nos ofrecía una salida ajustable de 3 a 4V con una resistencia (se decidió usar la equivalente para los 3.7V de salida), y una corriente de salida de pico menor a 2 A (típicamente 750 mA).

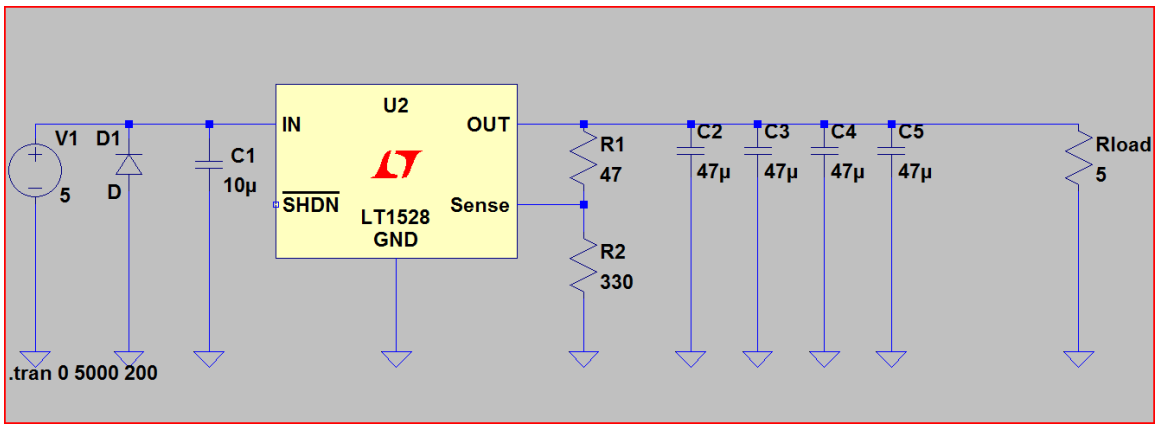

Figura 5.4. Regulador lineal para el GSM

Una vez resuelto el problema de la alimentación, estaba el problema de los puertos de comunicación.

Mientras que el módulo se alimentaba a 3.7 V, los puertos de transmisión y recepción utilizaban CMOS de 2.8 V. Esto suponía un quebradero de cabeza, ya que al utilizar tensiones distintas a la alimentación del módulo, había que implantar otro método de pasar de esas señales a los 5V del microcontrolador.

En primer lugar, para transmitir del puerto Tx del PIC18F2520 al módulo GSM, se optó por utilizar un divisor de tensión. Para sacar la tensión necesaria, se eligió un divisor de tensión de 470 ohm con 220 ohm, lo que originaba una tensión a la salida de 3.4V (un valor dentro de los tolerable).

Sin embargo, el verdadero problema surgía al pasar de los 2.8V a los 5V del microcontrolador. Realizando dos inversores en serie, y utilizando una alimentación de 5V en el colector de esos emisores (cómo se muestra en el dibujo), se consiguió arreglar el problema. Por último, añadiríamos switches y leds indicadores según el espacio disponible en el prototipo.

Para elegir la alimentación del dispositivo, optamos por una batería recargable de 5V de salida para el mismo, ya que el elemento de seguridad debía estar preparado para desconexiones momentáneas de la red eléctrica.

El oscilador, o reloj, requería que el microcontrolador fuese a una velocidad determinada (del entorno de los 4 Mhz).

Un 18F2520 tiene 10 modos de funcionamiento del reloj.

1.	LP	Low-Power Crystal
2.	XT	Crystal/Resonator
3.	HS	High-Speed Crystal/Resonator
4.	HSPLL	High-Speed Crystal/Resonator with PLL Enabled
5.	RC	External Resistor/Capacitor with FOSC/4 Output on RA6
6.	RCIO	External Resistor/Capacitor with I/O on RA6
7.	INTIO1	Internal Oscillator with FOSC/4 Output on RA6 and I/O on RA7
8.	INTIO2	Internal Oscillator with I/O on RA6 and RA7
9.	EC	External Clock with FOSC/4 Output
10.	ECIO	External Clock with I/O on RA6

Tabla 5.2. Modos de oscilación 18F2520

De entre todos los disponibles, se eligió el cristal (XT) ya que era el más adecuado para las características de frecuencia del reloj que queríamos.

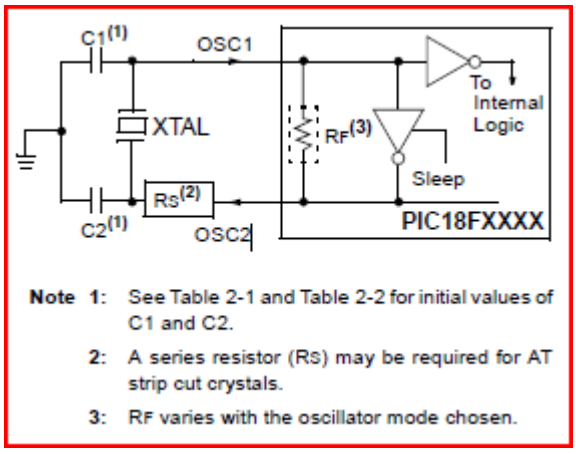

Figura 5.5. Esquemático del oscilador

Observando el datasheet y su tabla de valores, un oscilador de 4MHz y dos condensadores de 30pF eran el elemento necesario para su funcionamiento.

5.3. Diseño de firmware.

En primer lugar había que configurar el funcionamiento principal del micrcontrolador. Era necesario programar elementos como el Watchdog o los

temporizadores.

Entre los parámetros a configurar se eligió:

- Una frecuencia de reloj de 4MHz, generada a través de un cristal externo.
- Eliminar protecciones tales cómo el Watchdog, la protección de niveles,etc.
- No activar los pull-ups internos del puerto B, ya que se encontraban fuera conectados al teclado alfanumérico.
- Configurar la conexión de la Smartcard como una conexión maestro-esclavo (siendo el microcontrolador el maestro).
- Configurar la comunicación por RS232 entre el microcontrolador y el módulo GSM. Esto se hizo con los parámetros recomendados en el datasheet del módulo (9600 Baudios, 8 bits, sin paridad).
- Configurar los puertos de entrada y salida para los distintos elementos del proyecto.

Ente otras cosas, además el firmware debía controlar los distintos elementos del proyecto.

Por un lado debía transmitir información al módulo LCD según las acciones ejecutadas en el dispositivo de seguridad. También debía incorporar un modo de detectar cual tecla se había pulsado en el teclado alfanumérico. Debía ser capaz de escribir y leer en la memoria de las SmartCards mediante I2C.

Tenía que poseer un menú de configuración, al que sólo el administrador del dispositivo tendría acceso, con varias opciones de configuración. Debía ser capaz de administrar varios usuarios.

Tenía que tener la capacidad de mandar órdenes al módulo GSM, para poder enviar distintos mensajes de aviso al destinatario. Por último y no menos importante, tenía la obligación de bloquear o permitir el acceso a la cerradura electrónica según el caso expuesto.

6. Implementación

6.1. Implementación del hardware

Antes de montar el PCB había que realizar las pruebas y modificaciones pertinentes en una placa de pruebas para comprobar que el funcionamiento de todo era correcto.

En nuestro caso, utilizamos la placa de pruebas PicSchool de Microsystems Engineering. Esta placa posee ya implantados varios de los elementos que usaremos en el prototipo. Gracias a ella, no requeriremos de montaje del oscilador, de la pantalla LCD ni del teclado matricial en las pruebas.

También tiene en su disposición interruptores y leds para monitorizar el estado del dispositivo de seguridad, una protoboard para conectar los elementos restantes y acceso directo tanto a los puertos del microcontrolador como a las tomas de tierra y alimentación.

Figura 6.1. PicSchool toolkit

Para conectarlo al ordenador, se utiliza el debugger ICD3 conectado al puerto ICSP de la protoboard, ya mencionado anteriormente. Una vez se compruebe con las pruebas realizadas en el PicSchool son correctas y funcionan, se podrá empezar a diseñar y montar la placa de circuito impreso del prototipo.

6.2. Implementación del firmware.

Para escribir el programa que controlaría el firmware del microcontrolador, se utilizó el Code Composer Studio v.4.104. Aparte del "main.c", hicimos uso de las siguientes librerías y archivos:

- **18F2520.h**: librería con las funciones e instrucciones necesarias para manejar y configurar el PIC18F2520.

- **Lcd.c**: librería dedicada a facilitar el manejo de la pantalla LCD 16x2, con los protocolos de comunicación con la misma.

Para programar el firmware del microcontrolador, este posee dos fases principales.

En la fase inicial, el microcontrolador configura los puertos de comunicación, envía el saludo y comprueba en la EEPROM si hemos configurado ya las contraseñas mínimas necsarias (contraseña del administrador, contraseña del primer usuario y contraseña de pánico).

En el caso de que no lo hallamos configurado previamente, deberemos introducir esas contraseñas para ejecutar el programa principal. También deberemos introducir un número de intentos los cuales harán que se bloquee la caja si fallamos en esa cantidad de intentos.

Una vez transcurrida la fase inicial, llegamos a la fase principal. Nos pedirá que introduzcamos una contraseña de 6 dígitos, que pulsaremos con el teclado y nos los mostrará en pantalla con un símbolo "*". Una vez escrito el código, comprobará primero si se ha ejecutado la contraseña de pánico. Si la contraseña es la de pánico, abrirá la caja pero enviará un aviso por el módulo GSM si el aviso por móvil está activado.

En el caso de que no sea así, comprobará si se ha ejecutado el código de administrador, en ese caso se abrirá el menú de configuración y se quitará el bloqueo de la caja. En el menú de configuración podemos cambiar las contraseñas, el número de intentos, el número de teléfono, e, incluso, reiniciar toda la caja. Una vez hecho esto, el programa volverá a la comprobación de contraseña.

En el caso de que la contraseña tampoco sea la del administrador, leerá en la tarjeta SmartCard la identificación del usuario y comprobará la contraseña escrita con la guardada en la memoria del microcontrolador para ese usuario. Si acierta, se abrirá la caja. Si no se puede hacer esa comprobación o la contraseña es incorrecta, se gastará un intento y hay que volver a intentar poner la contraseña.

Por último, si se ha llegado al número de intentos máximo, la caja se bloqueará, impidiendo su apertura. La única manera de desbloquearla entonces es introduciendo la contraseña del administrador.

Si por algún motivo además tenemos activado el aviso por SMS, recibiremos un

mensaje cuando se abra la caja fuerte, se falle un intento, se bloquee la caja o salte la alarma de pánico.

Conviene mencionar varias características incorporadas al firmware.

Por un lado, kbd_getc() nos hace obtener un total de 17 valores posibles (del 0 al 16) y un total de 17 caracteres posibles asociados a ese numero (del “0” al “9”, el “*”,”#”, y las letras de la “A” a la “G”).

Tanto el valor “16” como la letra “G” es envían para indicar al microcontrolador que no se ha pulsado ninguna tecla. Los demás valores se utilizan para representarlos en la pantalla LCD (en el caso de los caracteres), como para guardarlos en memoria (en el caso de los números).

```
if (input_state(PIN_B4)==1 && teclado==1){
kbdc='7';
teclado=0;
}
else if (input_state(PIN_B5)==1 && teclado==1){
kbdc='8';
teclado=0;
}
else if (input_state(PIN_B6)==1 && teclado==1){
kbdc='9';
teclado=0;
}
else if (input_state(PIN_B7)==1 && teclado==1){
kbdc='C';
teclado=0;
}
```

Por otro lado, enviamos los comandos al módulo GSM mediante los comandos AT.

Los comandos AT (ATTENTION commands), forman parte del estándar GSM y se utilizan para la compatibilidad con diversos programas de software del mercado.

En este caso, hacemos uso de los siguientes comandos AT:

AT: sirve para comprobar si el módulo GSM es compatible con los comandos AT.

AT+CMGF=1: sirve para avisar al módulo GSM que deseamos enviar un SMS.

AT+CMGS=+34...: sirve para enviar el cuerpo del mensaje al móvil. Consta de dos instrucciones. En la primera elegimos el número a enviar el mensaje, y en la segunda instrucción enviamos el cuerpo del mensaje en sí.

```
if (gsmstatus==3){
```

```
printf("AT+CMGS=+34%c%c%c%c%c%c%c%c
%c\r",tele[0],tele[1],tele[2],tele[3],tele[4],tele[5],tele[6],tele[7],tele[8]);
delay_ms(500);
printf("Intento gastado");
delay_ms(500);
printf("%c",ctrlz);
delay_ms(500);
gsmstatus=0;
```

Por último, el menú se ha configurado para los 16 botones del teclado, usando el botón D para confirmar, el C para cancelar, y el B y el A para elegir opción adelante o atrás. La carcasa de estos botones se debe diseñar posteriormente para facilitar su uso más intuitivo.

Una vez realizado el programa, debemos programarlo al microcontrolador. Para ello conectamos el debugger entre el PC y el PicSchool y ejecutamos MPLAB IDE.

Nos metemos en la pestaña de Project → Set Language Tool localitations. Una vez se nos muestre la pantalla, comprobamos si tenemos CCS C Compiler for PIC12/14/16/18. Abrimos el proyecto ya compilado en CCS y dentro de Project seleccionamos CCS C Compiler como Toolsuite. Luego, seleccionamos el dispositivo en Configure → Select Device, el cual nos motrará la siguiente pantalla en la que seleccionamos el microcontrolador.

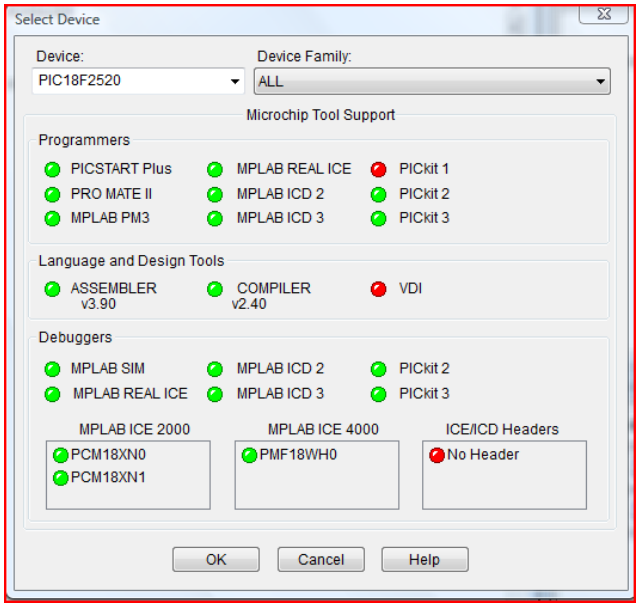

Figura 6.2. Selección de microcontrolador

Elegimos ICD 3 como programador en Programmer → Select Programmer → MPLAB ICD 3. Una vez hecho esto, le damos a Program, y luego a Verify (para comprobar que se ha programado correctamente). Nuestro programa ya esta programado en el microcontrolador.

7. Experimentación

Para comprobar que todo iba en orden, necesitábamos hacer algunas pruebas de campo y observar el funcionamiento del prototipo en la placa de pruebas.

Para ello, poníamos el programa en funcionamiento y averiguábamos si su funcionamiento era el adecuado en distintas situaciones. Si algo no funcionaba bien durante la experimentación, había que comprobar otra vez tanto la implementación del hardware como del software.

Entre varios fallos que se tuvieron que ajustar estaban el no funcionamiento del teclado alfanumérico (el cual se arreglo conectando los pull-downs), información de contraseñas que no se guardaban, un menú que había que reconfigurar, y diversos fallos menores en la transmisión de datos (como, por ejemplo, transmitir un dato dos veces seguidas).

También se probó empezando a manejar cada componente por separado, para luego terminar juntandolos. Esta parte no tenía muchos problemas, aparte de cambiar el orden de interacción con los periféricos y el microcontrolador.

Una vez hechos lo cambios pertinentes, se comprobó el funcionamiento final del prototipo realizando las siguientes pruebas:

- Se comprobó que la información de las contraseñas quedaba guardada tras el reset del dispositivo y que hacía bien las comprobaciones de contraseña.

- Se comprobó el correcto funcionamiento del menú de configuración para el administrador, así cómo que se guardasen las variables modificadas correctamente.

- Se monitorizó el estado de encendido y apagado del relé en todo momento, así como el bloqueo o desbloqueo de la caja.

- Se comprobó el intercambio de información con las tarjetas Smartcard, y que el usuario coincidiese con la contraseña introducida.

- Se comprobó que la comunicación GSM era correcta, y que los SMS se enviaban correctamente al destinatario.

- Se comprobó que la alarma funcionase correctamente con el código pánico.

Las contraseñas conseguían guardarse correctamente en la memoria, y se conseguía identificar al usuario. También la modificación de las mismas fue correcto. Las luces de encendido y apagado, así como el estado de alarma y bloqueo funcionaron de manera correcta.

El funcionamiento del menú era correcto, y lo bastante intuitivo de usar para el usuario. Las tarjetas Smartcard se leían y se programaban correctamente, permitiendo su

uso en el proyecto. Además, se comprobó que se pudiesen añadir o quitar usuarios.

Para finalizar, se demostró el funcionamiento del módulo GSM y el envío de SMSs al usuario destinatario.

8. Conclusiones y trabajo futuro.

8.1. Conclusiones.

Una vez finalizadas con éxito las pruebas finales, podemos hablar de las conclusiones tras haber realizado el proyecto. Tanto el microcontrolador como los diversos elementos conectados a él han terminado funcionando de manera satisfactoria y correcta.

Como producto final, hemos realizado una placa PCB que contiene un dispositivo de seguridad controlado por teclado alfanumérico, tarjetas Smartcard de identificación, y mensajes SMS. Tanto el control de la caja como el envío de mensajes SMS están en funcionamiento correctamente.

Nuestro dispositivo ha conseguido los siguientes objetivos:

- Ofrecer un dispositivo de comunicación remota entre un dispositivo y el usuario, en este caso mediante el envío de mensajes SMS, de muy bajo cose actualmente.

- Realizar un dispositivo fácil de utilizar y configurar. Gracias a un intuitivo menú para el administrador del dispositivo, este permite fácilmente cambiar los parámetros de funcionamiento con un simple teclado alfanumérico.

- Crear un dispositivo con varios elementos interactivos. Desde el envío de información a la pantalla, pasando por el intercambio con las tarjetas de memoria, hasta llegar al módulo GSM, hemos podido realizar un dispositivo completo con un único microcontrolador, aprovechando los recursos que ofrecía.

- Crear un dispositivo útil para varios usos. Puede controlar el acceso a nuestra caja fuerte, a una puerta de seguridad, conectar la alarma de casa... y todo ello con la posibilidad de hacer uso del número de usuarios que se requiera en cada situación.

La elección de un microcontrolador ha sido muy efectiva, ya que la relación coste/uso que le damos es alta. El simple hecho de que un microcontrolador llegue a manejar todos los elementos al mismo tiempo disminuye el coste del producto.

No obstante, este prototipo posee unas características de tamaño considerables, ya que hace uso de varios elementos de diversos tamaños y específicaciones. Usar encapsulado en SMD (montados en superficie) en un futuro puede ser una alternativa viable para solventar parcialmente esta cuestión.

De entre todos los elementos del prototipo, el más complejo y caro de implementar ha sido el módulo GSM, ya que requería distintos tipos de adaptación (lo que requería a su vez más espacio). Sin embargo, su versatilidad permite ampliar su uso de manera sencilla para otras funciones, lo que compensa el coste inicial. No obstante, no hay que olvidar que se requiere una SIMcard con servicio de telefonía para hacer uso del mismo (lo que lleva a costes adicionales si se paga de más por el servicio).

8.2. Líneas de trabajo futuras.

De entre las posibles ampliaciones del proyecto, caben destacar unas cuantas:

- Implementar el dispositivo en una cerradura real, teniendo en cuenta las limitaciones de voltaje e intensidad del relé electromagnético utilizado.

- Implementar una aplicación móvil (app), que permita una mayor interacción y monitorización del dispositivo, así cómo una interfaz más sencilla y menos tediosa que el uso de mensajes SMS.

- Implementar este dispositivo en un proyecto de domótica general, permitiendo no sólo el acceso y control de un único dispositivo sino de todo el hogar.

- Implementar un sistema alternativo para la identificación del usuario aparte del uso de Smartcards. Por ejemplo, con tarjetas de proximidad.

Y muchas opciones disponibles más.

II. Planos y especificaciones.

9.1. Esquema eléctrico.

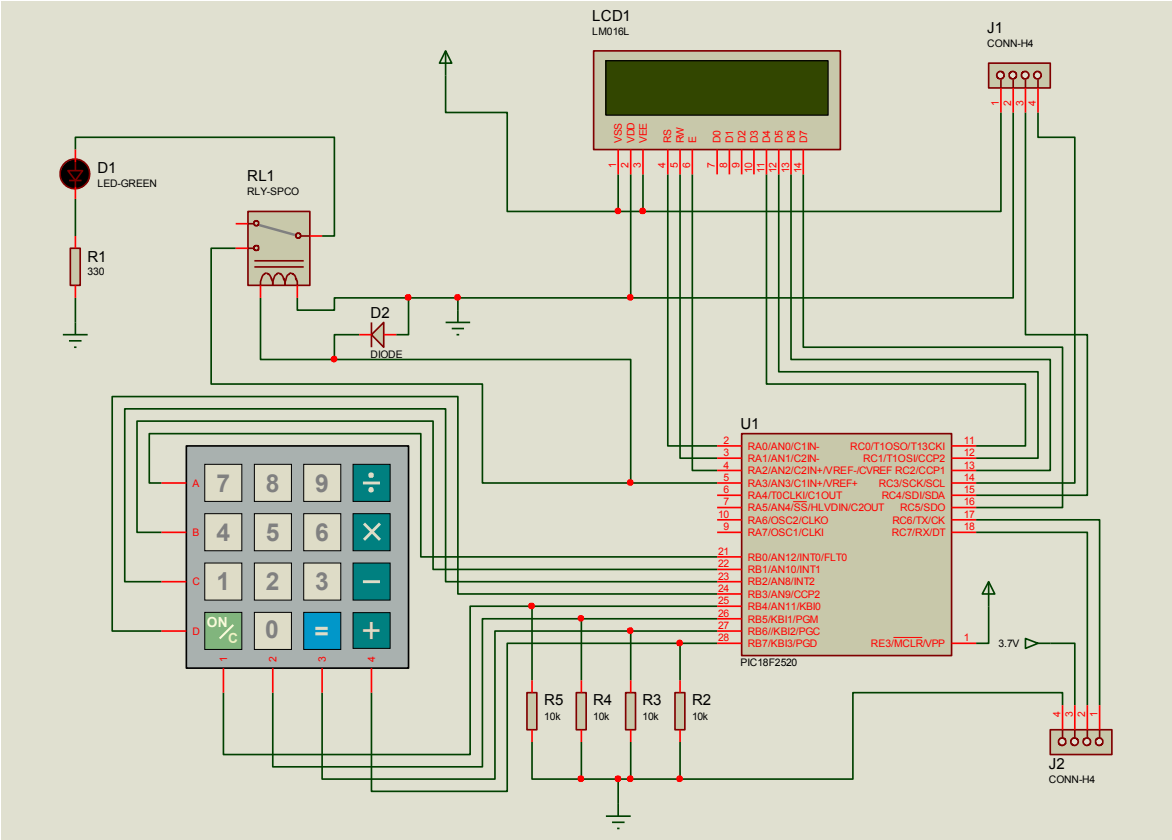

Figura 9.1. Esquemático del PCB

9.2. Listado de componentes.

En la tabla se muestra un listado de componentes utilizados en el circuito impreso.

NÚMERO	DESCRIPCIÓN
6	Resistencias de 1KΩ
2	Resistencias de 470Ω
1	Resistencia de 47Ω
2	Resistencias de 330Ω
1	Resistencias de 220Ω
2	Transistor BC547
4	Condensador electrolítico 47uF
2	Condensador cerámico 15pF
1	Cristal 4MHz
1	Led rojo
1	Led verde
1	Diodo 1N4007
1	Relé electromagnético 3V
1	Microcontrolador PIC 18F2520
1	Teclado matricial 4x4
1	Pantalla LCD 16x2
1	Conector Smartcard
1	Módulo GSM Telit GC864-PY

Tabla 9.1. Listado de componentes

9.3. Pinout

SEÑAL	PIN
MCLR/Vpp/RE3	**1**
RA0/AN0/C1IN-	**2**
RA1/AN1/C2IN-	**3**
RA2/AN2/C2IN+/VREF+/CVREF	**4**
RA3/AN3/C1IN+/VREF+	**5**
RA4/T0CLKI/C1OUP	**6**
RA5/AN4/SS/HLVDIN/C2OUT	**7**
Vss	**8**
RA7/**OSC1**/CLKI	**9**
RA6/**OSC2**/CLKO	**10**
RC0/T1OSO/T13CKI	**11**
RC1/T1OSI/CCP2	**12**
RC2/CCP1	**13**
RC3/SCK/**SCL**	**14**
RC4/SDI/**SDA**	**15**
RC5/SDO	**16**
RC6/**TX**/CK	**17**
RC7/**RX**/DT	**18**
Vss	**19**
Vdd	**20**
RB0/AN12/INT0/FLT0	**21**
RB1/AN10/INT1	**22**
RB2/AN87INT2	**23**
RB3/AN9/CCP2	**24**
RB4/AN11/KBI0	**25**
RB5/KBI1/PGC	**26**
RB6/KBI2/PGC	**27**
RB7/KBI3/PGD	**28**

Tabla 9.2. Listado de conexiones del PIC 18F

SEÑAL	**PIN**
Vcc	**1**
Master clear	**2**
SCL	**3**
Not connected	4
GND	**5**
Not connected	6
SDA	**7**
Not connected	8

Tabla 9.3. Listado de conexiones de la tarjeta Smartcard

III. Código del programa.

PROGRAMA PRINCIPAL

```
#include <18F2520.h>
#fuses XT,NOPUT,NOPROTECT,NOBROWNOUT,NOLVP,NOWDT,NOPBADEN

#use delay(clock=4000000)
#use i2c(MASTER, SDA=PIN_C4, SCL=PIN_C3)
#use RS232(BAUD=9600,BITS=8,PARITY=N,XMIT=PIN_C6,RCV=PIN_C7)
#use standard_io(b)
#use standard_io(c)

int num[6];
int menu=0;
char tele[9];
int teclado=1;
char ctrlz=26;
int bloq=0;
int gsmstatus=0;
int keycount=0;
int inicio;
int menuindex;
int intento=0;
int numintento;
int permiso=1;
char key;

#define LCD_ENABLE_PIN  PIN_A2
#define LCD_RS_PIN      PIN_A0
#define LCD_RW_PIN      PIN_A1
#define LCD_DATA4       PIN_C0
#define LCD_DATA5       PIN_C1
#define LCD_DATA6       PIN_C2
#define LCD_DATA7       PIN_C5

#include <lcd.c>

void kbd_init(){
set_tris_b(0xF0);
port_b_pullups(FALSE);}

int cambioint(){
int numer;
if (key=='0'){
numer=0;}
```

```
if (key=='1'){
numer=1;}

if (key=='2'){
numer=2;}

if (key=='3'){
numer=3;}

if (key=='4'){
numer=4;}

if (key=='5'){
numer=5;}

if (key=='6'){
numer=6;}

if (key=='7'){
numer=7;}

if (key=='8'){
numer=8;}

if (key=='9'){
numer=9;}

if (key=='A'){
numer=10;}

if (key=='B'){
numer=11;}

if (key=='C'){
numer=12;}

if (key=='D'){
numer=13;}

if (key=='*'){
numer=14;}

if (key=='#'){
numer=15;}

return numer;
}
```

```
char kbd_getc()
{
char kbdc;
kbdc='G';

output_high(PIN_B0);
output_low(PIN_B1);
output_low(PIN_B2);
output_low(PIN_B3);
if (input_state(PIN_B4)==1 && teclado==1){
kbdc='1';
teclado=0;
}

else if (input_state(PIN_B5)==1 && teclado==1){
kbdc='2';
teclado=0;
}

else if (input_state(PIN_B6)==1 && teclado==1){
kbdc='3';
teclado=0;
}

else if (input_state(PIN_B7)==1 && teclado==1){
kbdc='A';
teclado=0;
}

output_low(PIN_B0);
output_high(PIN_B1);
output_low(PIN_B2);
output_low(PIN_B3);

if (input_state(PIN_B4)==1 && teclado==1){
kbdc='4';
teclado=0;
}

else if (input_state(PIN_B5)==1 && teclado==1){
kbdc='5';
teclado=0;
}

else if (input_state(PIN_B6)==1 && teclado==1){
kbdc='6';
teclado=0;
}
```

```
else if (input_state(PIN_B7)==1 && teclado==1){
kbdc='B';
teclado=0;
}

output_low(PIN_B0);
output_low(PIN_B1);
output_high(PIN_B2);
output_low(PIN_B3);

if (input_state(PIN_B4)==1 && teclado==1){
kbdc='7';
teclado=0;
}

else if (input_state(PIN_B5)==1 && teclado==1){
kbdc='8';
teclado=0;
}

else if (input_state(PIN_B6)==1 && teclado==1){
kbdc='9';
teclado=0;
}

else if (input_state(PIN_B7)==1 && teclado==1){
kbdc='C';
teclado=0;
}

output_low(PIN_B0);
output_low(PIN_B1);
output_low(PIN_B2);
output_high(PIN_B3);

if (input_state(PIN_B4)==1 && teclado==1){
kbdc='*';
teclado=0;
}

else if (input_state(PIN_B5)==1 && teclado==1){
kbdc='0';
teclado=0;
}

else if (input_state(PIN_B6)==1 && teclado==1){
kbdc='#';
```

```
teclado=0;
}

else if (input_state(PIN_B7)==1 && teclado==1){
kbdc='D';
teclado=0;
}

output_low(PIN_B3);
output_high(PIN_B0);
output_high(PIN_B1);
output_high(PIN_B2);
output_high(PIN_B3);

if (input_state(PIN_B4)==0 && input_state(PIN_B5)==0 && input_state(PIN_B6)==0
&& input_state(PIN_B7)==0){
  teclado=1;
}

output_low(PIN_B0);
output_low(PIN_B1);
output_low(PIN_B2);
output_low(PIN_B3);

return kbdc;
}

void smstomobile(){
printf("AT+CMGF=1\r");
delay_ms(500);

if (gsmstatus==1){
printf("AT+CMGS=+34%c%c%c%c%c%c%c%c
%c",tele[0],tele[1],tele[2],tele[3],tele[4],tele[5],tele[6],tele[7],tele[8]);
delay_ms(500);
printf("Caja abierta por ususario");
delay_ms(500);
printf("%c",ctrlz);
delay_ms(500);
gsmstatus=0;
}

if (gsmstatus==2){
printf("AT+CMGS=+34%c%c%c%c%c%c%c%c
%c",tele[0],tele[1],tele[2],tele[3],tele[4],tele[5],tele[6],tele[7],tele[8]);
delay_ms(500);
printf("Caja bloqueada");
delay_ms(500);
```

```
printf("%c",ctrlz);
delay_ms(500);
gsmstatus=0;
}

if (gsmstatus==3){
printf("AT+CMGS=+34%c%c%c%c%c%c%c%c
%c",tele[0],tele[1],tele[2],tele[3],tele[4],tele[5],tele[6],tele[7],tele[8]);
delay_ms(500);
printf("Intento gastado");
delay_ms(500);
printf("%c",ctrlz);
delay_ms(500);
gsmstatus=0;
}

if (gsmstatus==4){
printf("AT+CMGS=+34%c%c%c%c%c%c%c%c
%c",tele[0],tele[1],tele[2],tele[3],tele[4],tele[5],tele[6],tele[7],tele[8]);
delay_ms(500);
printf("Alarma de panico");
delay_ms(500);
printf("%c",ctrlz);
delay_ms(500);
gsmstatus=0;
}

}

//============================I2C para número
void telefono(){
keycount=1;
lcd_putc("\f");
lcd_putc("Teclea numero");
lcd_gotoxy(1,2);
while(keycount<10)
  {
   key = kbd_getc();
     if(key!='G' && key<='9' && keycount==1){
        tele[0]=cambioint();
        printf(lcd_putc,"%c",key);
      key='G';
        keycount++;}
     if(key!='G' && key<'A' &&  keycount==2){
        tele[1]=cambioint();
        printf(lcd_putc,"%c",key);
        key='G';
        keycount++;}
```

```
    if(key!='G' && key<'A' &&  keycount==3){
       tele[2]=cambioint();
       printf(lcd_putc,"%c",key);
       key='G';
       keycount++;}
    if(key!='G' && key<'A' &&  keycount==4){
       tele[3]=cambioint();
       printf(lcd_putc,"%c",key);
       key='G';
       keycount++;}
    if(key!='G' && key<'A' &&  keycount==5){
       tele[4]=cambioint();
       printf(lcd_putc,"%c",key);
       key='G';
       keycount++;}
    if(key!='G' && key<'A' &&  keycount==6){
       tele[5]=cambioint();
       printf(lcd_putc,"%c",key);
       key='G';
       keycount++;}
    if(key!='G' && key<'A' &&  keycount==7){
       tele[6]=cambioint();
       printf(lcd_putc,"%c",key);
       key='G';
       keycount++;}
    if(key!='G' && key<'A' &&  keycount==8){
       tele[7]=cambioint();
       printf(lcd_putc,"%c",key);
       key='G';
       keycount++;}
    if(key!='G' && key<'A' &&  keycount==9){
       tele[8]=cambioint();
       printf(lcd_putc,"%c",key);
       key='\0';
       keycount++;
       lcd_putc("\f");
       lcd_putc("Number saved");}
    }

lcd_putc("\f Enviando datos");
delay_ms(2000);
keycount=11;
if (keycount==11){
       i2c_start ();
     i2c_write (0xA0);
     i2c_write (60);
     i2c_write(tele[0]);
     i2c_stop();
```

```
      delay_ms(50);
      keycount=12;
}

lcd_putc("\f 1 de 9");
delay_ms(2000);
if (keycount==12){
            i2c_start ();
      i2c_write (0xA0);
      i2c_write (61);
      i2c_write(tele[1]);
      i2c_stop();
      delay_ms(50);
      keycount=13;
}

lcd_putc("\f 2 de 9");
delay_ms(2000);
if (keycount==13){
      i2c_start ();
      i2c_write (0xA0);
      i2c_write (62);
      i2c_write(tele[2]);
      i2c_stop();
      delay_ms(50);
keycount=14;
}

lcd_putc("\f 3 de 9");
delay_ms(2000);
if (keycount==14){
      i2c_start ();
      i2c_write (0xA0);
      i2c_write (63);
      i2c_write(tele[3]);
      i2c_stop();
      delay_ms(50);
      keycount=15;
}

lcd_putc("\f 4 de 9");
delay_ms(2000);
if (keycount==15){
      i2c_start ();
      i2c_write (0xA0);
      i2c_write (64);
      i2c_write(tele[4]);
      i2c_stop();
```

```
      delay_ms(50);
      keycount=16;
}

lcd_putc("\f 5 de 9");
delay_ms(2000);
if (keycount==16){
      i2c_start ();
      i2c_write (0xA0);
      i2c_write (65);
      i2c_write(tele[5]);
      i2c_stop();
      delay_ms(50);
      keycount=17;
}

lcd_putc("\f 6 de 9");
delay_ms(2000);
if (keycount==17){
      i2c_start ();
      i2c_write (0xA0);
      i2c_write (66);
      i2c_write(tele[6]);
      i2c_stop();
      delay_ms(50);
      keycount=18;
}

lcd_putc("\f 7 de 9");
delay_ms(2000);
 if (keycount==18){
      i2c_start ();
      i2c_write (0xA0);
      i2c_write (67);
      i2c_write(tele[7]);
      i2c_stop();
      delay_ms(50);
      keycount=19;
 }

lcd_putc("\f 8 de 9");
delay_ms(2000);
if (keycount==19){
      i2c_start ();
      i2c_write (0xA0);
      i2c_write (68);
      i2c_write(tele[8]);
      i2c_stop();
```

```
      delay_ms(50);
      keycount=11;
}

 lcd_putc("\f Number sent");
delay_ms(2000);
permiso=2;
}

//============================MAESTRO
void clavemaestra()
{
lcd_putc("\f");
lcd_putc("Codigo maestro");
keycount=1;
while(keycount<7)
  {
   key = kbd_getc();
    if(key!='G' && key!='1' && key!='2' && key!='3' && key!='4' && key!='5' && key!
='6' && key!='7' && key!='8' && key!='9' && key!='0' && keycount==1){
        num[0]=cambioint();
        write_eeprom(1,num[0]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c",'*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==2){
        num[1]=cambioint();
        write_eeprom(2,num[1]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==3){
        num[2]=cambioint();
        write_eeprom(3,num[2]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==4){
        num[3]=cambioint();
        write_eeprom(4,num[3]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
```

```
        key='G';
        keycount++;
        }

    if(key!='G' && keycount==5){
        num[4]=cambioint();
        write_eeprom(5,num[4]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;
        }

    if(key!='G' && keycount==6){
        num[5]=cambioint();
        write_eeprom(6,num[5]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        delay_ms(500);
        key='G';
        keycount++;
        }

    }

  keycount=0;
permiso=2;
}

//============================PANICO
void clavepanico(){
lcd_putc("\f");
lcd_putc("Codigo panico");
keycount=1;
while(keycount<7)
  {
   key = kbd_getc();
    if(key!='G' && key!='1' && key!='2' && key!='3' && key!='4' && key!='5' && key!
='6' && key!='7' && key!='8' && key!='9' && key!='0' && keycount==1){
        num[0]=cambioint();
        write_eeprom(51,num[0]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==2){
        num[1]=cambioint();
```

```
        write_eeprom(52,num[1]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==3){
        num[2]=cambioint();
        write_eeprom(53,num[2]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==4){
        num[3]=cambioint();
        write_eeprom(54,num[3]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;
        }

    if(key!='G' && keycount==5){
        num[4]=cambioint();
        write_eeprom(55,num[4]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;
        }

    if(key!='G' && keycount==6){
        num[5]=cambioint();
        write_eeprom(56,num[5]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        delay_ms(500);
        key='G';
        keycount++;
        }

  }

  keycount=0;
permiso=2;
}
```

```
//==========================USUARIO
void claveusuario(){
lcd_putc("\f");
lcd_putc("Codigo usuario");
keycount=1;
while(keycount<7)
  {
   key = kbd_getc();
    if(key!='G' && key!='1' && key!='2' && key!='3' && key!='4' && key!='5' && key!
='6' && key!='7' && key!='8' && key!='9' && key!='0' && keycount==1){
        num[0]=cambioint();
        write_eeprom(11,num[0]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==2){
        num[1]=cambioint();
        write_eeprom(12,num[1]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==3){
        num[2]=cambioint();
        write_eeprom(13,num[2]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;}

    if(key!='G' && keycount==4){
        num[3]=cambioint();
        write_eeprom(14,num[3]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;
        }

   if(key!='G' && keycount==5){
        num[4]=cambioint();
        write_eeprom(15,num[4]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
```

```
        keycount++;
        }

    if(key!='G' && keycount==6){
        num[5]=cambioint();
        write_eeprom(16,num[5]);
        lcd_gotoxy(keycount,2);
        printf(lcd_putc,"%c", '*');
        key='G';
        keycount++;
        }

  }

  keycount=0;
permiso=2;
}

//============================INTENTOS DE METER CODIGO
void cambiaintentos(){
lcd_putc("\f");
lcd_putc("Numero intentos");
numintento=0;
while(numintento==0){
   key = kbd_getc();
   if(key=='1'){
   lcd_gotoxy(1,2);
   lcd_putc("\f 1 intento");
   numintento=1;
   intento=0;
   delay_ms(1000);}
   if(key=='2'){
   lcd_gotoxy(1,2);
   lcd_putc("\f 2 intentos");
   numintento=2;
   intento=0;
   delay_ms(1000);}
   if(key=='3'){
   lcd_gotoxy(1,2);
   lcd_putc("\f 3 intentos");
   numintento=3;
   intento=0;
   delay_ms(1000);}
   if (key!='G' && key>'3'){
   lcd_gotoxy(1,2);
   lcd_putc("Intentos de mas");
    }
```

```
  }

permiso=2;
}

void muestratelefono(){
lcd_putc("\f");
lcd_putc("Nº Telefono");
lcd_gotoxy(1,2);
printf(lcd_putc,"%u%u%u%u%u%u%u%u
%u",tele[0],tele[1],tele[2],tele[3],tele[4],tele[5],tele[6],tele[7],tele[8]);
while (key!='D'){
key=kbd_getc();
}

}

//============================MENU DE CONFIGURACION
void menumaestro(){
lcd_putc("\f");
lcd_gotoxy(1,1);
lcd_putc("Abriendo menu...");
delay_ms(5000);
menuindex=0;
permiso=2;
while(menuindex<10){
key=kbd_getc();
if(key=='1' || (menuindex==8 && menu==1) || (menuindex==1 && menu==1) ||
menuindex==0){
lcd_putc("\f");
lcd_putc("1.Codigo maestro");
menuindex=1;
menu=0;}

if(key=='2' || (menuindex==2 && menu==1)){
lcd_putc("\f");
lcd_putc("2.Codigo panico");
menuindex=2;
menu=0;}

if(key=='3' || (menuindex==3 && menu==1)){
lcd_putc("\f");
lcd_putc("3.Codigo usuario");
menuindex=3;
menu=0;}

if(key=='4' || (menuindex==4 && menu==1)){
lcd_putc("\f");
```

```
lcd_putc("4.Numerointentos");
menuindex=4;
menu=0;}

if(key=='5' || (menuindex==5 && menu==1)){
lcd_putc("\f");
lcd_putc("5.Reinicia todo");
menuindex=5;
menu=0;}

if(key=='6' || (menuindex==6 && menu==1)){
lcd_putc("\f");
lcd_putc("6.Marca telefono");
menuindex=6;
menu=0;}
if(key=='7' || (menuindex==7 && menu==1) || (menuindex==9 && menu==1)){
lcd_putc("\f");
lcd_putc("7.Muestra numero");
menuindex=7;
menu=0;}

if(key=='A'){
menuindex++;
menu=1;
}

if(key=='B'){
if(menuindex==1){menuindex=9;}
else {menuindex--;}
menu=1;
}

if(key=='D'){
  if (menuindex==1){key='G';delay_ms(200);clavemaestra();menuindex=10;}

  else if (menuindex==2){key='G';delay_ms(200);clavepanico();menuindex=10;}

  else if (menuindex==3){key='G';delay_ms(200);claveusuario();menuindex=10;}

  else if (menuindex==4){key='G';delay_ms(200);cambiaintentos();menuindex=10;}

  else if (menuindex==5){key='G';delay_ms(200);write_eeprom(0,13);menuindex=10;}

  else if (menuindex==6){key='G';delay_ms(200);telefono();menuindex=10;}

  else if (menuindex==7){key='G';delay_ms(200);muestratelefono();menuindex=10;}

}
```

```
if (key=='C'){key='G'; delay_ms(200);menuindex=10;}

}

}

//============================COMPROBAR CONTRASEÑA
void comprobar()
{
if (intento!=numintento){ permiso=1;}

else if (intento==numintento){permiso=0;}

if (num[0]==read_eeprom(11) && num[1]==read_eeprom(12) &&
num[2]==read_eeprom(13) && num[3]==read_eeprom(14) &&
num[4]==read_eeprom(15) && num[5]==read_eeprom(16) && intento<numintento &&
permiso==1){
intento=0;
lcd_putc("\f");
lcd_putc("Caja abierta");
gsmstatus=1;
smstomobile();
output_high(PIN_A3);
delay_ms(5000);
output_low(PIN_A3);
}//NUMERO CORRECTO, SE ABRE LA CAJA

else if (num[0]==read_eeprom(51) && num[1]==read_eeprom(52) &&
num[2]==read_eeprom(53) && num[3]==read_eeprom(54) &&
num[4]==read_eeprom(55) && num[5]==read_eeprom(56)){
permiso=1;
delay_ms(5000);
lcd_putc("\f");
lcd_putc("Abriendo caja");
lcd_gotoxy(1,2);
lcd_putc("*");
delay_ms(500);
lcd_gotoxy(2,2);
lcd_putc("*");
delay_ms(500);
lcd_gotoxy(3,2);
lcd_putc("*");
delay_ms(500);
lcd_gotoxy(4,2);
lcd_putc("*");
delay_ms(500);
lcd_gotoxy(5,2);
```

```
lcd_putc("*");
delay_ms(500);
lcd_gotoxy(6,2);
lcd_putc("*");
delay_ms(500);
lcd_gotoxy(7,2);
lcd_putc("*");
delay_ms(500);
lcd_gotoxy(8,2);
lcd_putc("*");
delay_ms(500);
lcd_putc("\f");
lcd_putc("Caja abierta");
gsmstatus=4;
smstomobile();
output_high(PIN_A3);
delay_ms(5000);
output_low(PIN_A3);
intento=numintento;}//NUMERO PANICO! LA CAJA SE ABRE PERO SE ACTIVA LA
ALARMA

else if (num[0]==read_eeprom(1) && num[1]==read_eeprom(2) &&
num[2]==read_eeprom(3) && num[3]==read_eeprom(4) && num[4]==read_eeprom(5)
&& num[5]==read_eeprom(6)){
intento=0;
permiso=1;
menumaestro();}//MAESTRO ENTRA, SE ACTIVA EL MENU

else {
lcd_putc("\f");
lcd_putc("Codigo falso");
gsmstatus=2;
smstomobile();
intento++;
}

if (intento==numintento){ permiso=0;}//SE HAN SUPERADO LOS INTENTOS SE
BLOQUEA LA CAJA

  lcd_putc("\f");
 keycount=0;
}

//============================PROGRAMA PRINCIPAL
void main() {

kbd_init();
lcd_init();
```

```
lcd_gotoxy(1,1);
lcd_putc("Preparando ...");
delay_ms(2000);
permiso=1;

while(TRUE)
  {

inicio=read_eeprom(0);
  if (inicio!=12){
clavemaestra();
clavepanico();
claveusuario();
cambiaintentos();
lcd_putc("\f");
write_eeprom(0,12);
}

if (permiso==2){permiso=1;}

   if(keycount==0 && permiso==1){
      lcd_putc("\f");
      lcd_putc("Marca codigo");
      keycount=1;}

else if (permiso==0 && bloq==0){
     bloq=1;
      lcd_putc("\f");
      lcd_putc("Caja bloqueada");
      keycount=1;}


key = kbd_getc();

      if(key!='G' && keycount==1){
         num[0]=cambioint();
         lcd_gotoxy(1,2);
         printf(lcd_putc,"%c", '*');
         key='G';
         keycount++;}

     if(key!='G' && keycount==2){
         num[1]=cambioint();
         lcd_gotoxy(2,2);
         printf(lcd_putc,"%c", '*');
         key='G';
         keycount++;}
```

```
    if(key!='G' && keycount==3){
       num[2]=cambioint();
       lcd_gotoxy(3,2);
       printf(lcd_putc,"%c", '*');
       key='G';
       keycount++;}

    if(key!='G' && keycount==4){
       num[3]=cambioint();
       lcd_gotoxy(4,2);
       printf(lcd_putc,"%c", '*');
       key='G';
       keycount++;}

    if(key!='G' && keycount==5){
       num[4]=cambioint();
       lcd_gotoxy(5,2);
       printf(lcd_putc,"%c", '*');
       key='G';
       keycount++;}

    if(key!='G' && keycount==6){
       num[5]=cambioint();
       lcd_gotoxy(6,2);
       printf(lcd_putc,"%c", '*');
       delay_ms(1000);
       key='G';
       keycount++;
     bloq=0;
       comprobar();
       keycount=0;
       }

  }

}
```

IV Bibliografía

Compilador C CSS y Simulador Proteus para microcontroladores PIC
Eduardo García Breijo
Editorial Alfaomega, 1ª edición 2008

Diseño y simulación de sistemas microcontrolados en lenguaje C
Juan Ricardo Clavijo Mendoza
1ª edición 2011

An Introduction to GSM (una introducción al GSM)
Redl, Weber y Oliphant
Artech House

Manuales del módulo GSM GC864-PY
disponibles en la página de Telit: www.telit.com

Manuales del microcontrolador PIC18F2520 de Microchip
disponibles en la página de Microchip: http://www.microchip.com/