UNIVERSIDAD DON BOSCO
FACULTAD DE INGENIERÍA
ESCUELA DE ELECTRÓNICA

DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE IDENTIFICACIÓN AUTOMÁTICA, CON APLICACIONES DE HARDWARE Y SOFTWARE, HACIENDO USO DE LA TECNOLOGÍA RFID

TRABAJO DE GRADUACIÓN

PRESENTADO POR
Luis Mario Mercado Cortez
Renato Raúl Lara Orellana

PARA OPTAR AL GRADO DE
Ingeniero en Automatización

Asesor:
Ing. Héctor Rubén Carías Juárez

Diciembre de 2007
Soyapango – El Salvador – Centro América

UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERÍA

AUTORIDADES:

ING. FEDERICO HUGUET RIVERA
RECTOR

PBRO. VÍCTOR BERMÚDEZ, sdb.
VICERRECTOR ACADÉMICO

LIC. MARIO RAFAEL OLMOS
SECRETARIO GENERAL

ING. ERNESTO GODOFREDO GIRÓN
DECANO FACULTAD DE INGENIERÍA

UNIVERSIDAD DON BOSCO

TRIBUNAL EXAMINADOR

_____

Ing. Héctor Ruben Carías Juárez

ASESOR DE TESIS

_____

Ing. Erick Alexander Blanco Guillen

JURADO EVALUADOR

_____

Ing. Marcos Tulio Guevara

JURADO EVALUADOR

_____

Ing. Jorle Alonso López Salazar

JURADO EVALUADOR

AGRADECIMIENTOS

Agradezco a Dios Todopoderoso por haberme dado la sabiduría, la paciencia y la fuerza necesaria para poder culminar mis estudios.

A mis padres por el apoyo incondicional, el AMOR, comprensión y paciencia que siempre me brindaron en este caminar.

A mi madre por su apoyo, sus atenciones, sus ánimos que siempre me ayudaron en esta etapa y que aun me ayudan para seguir adelante.

A mi padre por sus siempre acertados consejos, que me han ayudado a enfrentar y salir adelante ante las adversidades que se me han presentado.

A mis hermanos Raúl, Keny, Danilo y Carolina, por su inmensa solidaridad y ayuda que siempre me demostraron.

A mi esposa y compañera Mireya quien siempre me brindo su apoyo, su ayuda, su atención y sus ideas.

A los ingenieros Jorge López y Erick Rivas por su especial apoyo.

A mis amigos Evert Rivas, Gerson Joksan, Luis Enrique Martínez por su apoyo para conmigo en esta última etapa.

Muchas Gracias a todas las personas, profesores y amigos que de una u otra forma me ayudaron y con los cuales he compartido muy gratos momentos.

Este triunfo se lo dedico especialmente a mis padres, que han sido y seguirán siendo uno de los pilares en mi vida.

También es su triunfo.

Muchas Gracias.

Luis Mario Mercado Cortez.

CONTENIDO

1. INTRODUCCIÓN

El presente documento contiene la información concerniente al trabajo de graduación "DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE IDENTIFICACIÓN AUTOMÁTICA, CON APLICACIONES DE HARDWARE Y SOFTWARE, HACIENDO USO DE LA TECNOLOGÍA RFID[1]". En el marco teórico se dan a conocer los fundamentos y principios que sirven como base para la implementación del sistema RFID desarrollado, donde para dicho sistema se utilizaron: microcontroladores PIC, comunicación USB con PIC, programación en lenguaje C tanto para PC como para microcontroladores PIC, Microsoft SQL Server 2005 y modificaciones al registro de Windows XP para el inicio automático de sesión.

Posteriormente se dedica una parte a una explicación del diseño especifico del Sistema RFID desarrollado, en este se detalla el diseño de las etapas: lectura RFID, procesamiento de datos, regulación de voltaje, reset, entradas y salidas de potencia, modo de operación, circuito impreso, y el diseño de la antena, en este último se detallan los cálculos y mediciones necesarias para tener un funcionamiento óptimo de la antena.

El funcionamiento del sistema es explicado detalladamente en un capítulo dedicado solamente a esto, en el se detalla el funcionamiento del sistema: modos de trabajo, modo administrador, modo control de acceso, modo control de acceso y arranque de automóvil, modo inicio de sesión de Windows XP y el modo de marcación de horas de trabajo; la implementación de las aplicaciones es desarrollada en otro capítulo donde se explica detalladamente que se debe hacer para poner en marcha las aplicaciones desarrolladas. Hay que mencionar que el código fuente de todos los programas, y los formularios de Visual Basic, por ser muy extensos y para mejorar la estructura del presente documento se colocaron en los Anexos.

[1] **RFID,** identificación por Radiofrecuencia (Radio Frecuency Identification).

2. MARCO TEÓRICO

2.1 SISTEMAS RFID

"RFID es el acrónimo de **Radio Frequency Identification,** que significa Identificación por Radio Frecuencia. Es un método de identificación automática sin contacto, que es la tecnología más nueva y de más rápido crecimiento en el segmento de identificación automática"[2].

RFID permite identificación automática, localización y monitoreo de personas, objetos y animales en una gran cantidad de aplicaciones que van desde simple inventario hasta sistemas complejos de casetas de cobro en carreteras. Por ejemplo, en sistemas de inventario, la idea es que cada producto lleve una etiqueta que tendrá un número de identificación único (a diferencia del código de barras que es el mismo para todos los productos iguales), pudiendo asociarlo perfectamente al comprador.

En la siguiente figura se puede observar todos los componentes que conforman un Sistema de identificación automática RFID.

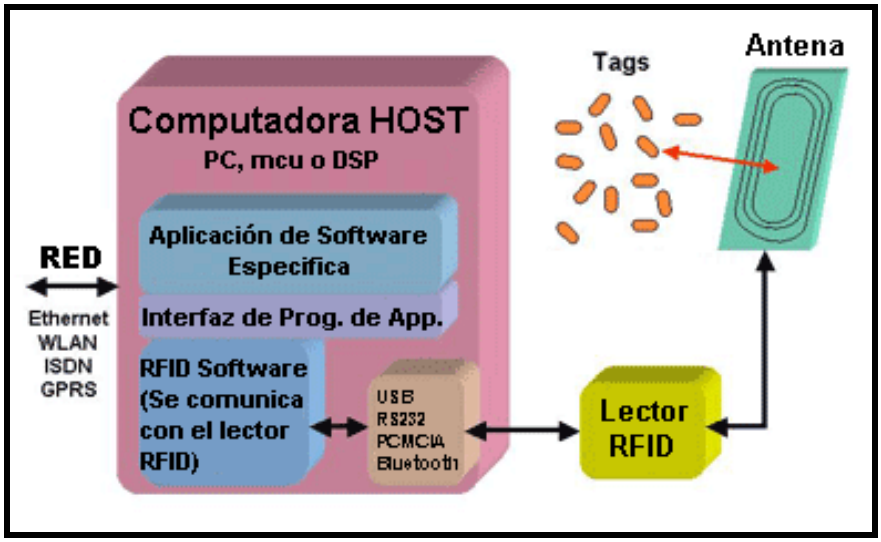


Figura 2.1.1 Diagrama de un Sistema RFID

[2] Adaptado del articulo "Trasteando" (http://nkieto.f2o.org/blog/archivos/000143.php Accesado en diciembre de 2006.).

La etiqueta o Tag de RFID: está compuesta de un chip y una antena muy pequeña. Una etiqueta RFID es un pequeño chip adherido o incorporado a un producto o persona y a través del cual es posible realizar algún tipo de identificación, autenticación o mantener un rastreo de su localización. Las etiquetas o *tags* RFID contienen antenas para permitirles recibir y responder a peticiones por radiofrecuencia desde un emisor-receptor RFID. Existen dos tipos de etiquetas:

- Etiquetas pasivas, que no necesitan alimentación eléctrica interna.
- Etiquetas activas, que sí necesitan alimentación eléctrica interna.

Circuito Lector RFID: Todo sistema RFID posee un circuito lector o sistema de base que lee datos en los tags o etiquetas RFID, la manera en que lee depende del estándar con el que este trabaje. De manera general el circuito lector RFID lee el número de identificación de las etiquetas o tags en la siguiente forma:

- El interrogador o lector genera un campo de radiofrecuencia, normalmente conmutando una bobina a alta frecuencia. Las frecuencias usuales van desde 125 kHz hasta la banda de frecuencias de 2.4 GHz.
- El campo de radiofrecuencia genera una corriente eléctrica sobre la bobina de recepción del tag. Esta señal provee la sincronía de reloj y es rectificada para alimentar el circuito (si es un tag pasivo).
- Cuando la alimentación llega a ser suficiente, el tag transmite su número único de identificación al circuito lector.
- El circuito lector detecta los datos transmitidos por la etiqueta RFID y los decodifica según sea el protocolo de modulación, codificación y transmisión.

El rango de lectura del circuito lector depende de la frecuencia en la que se trabaje, la potencia del circuito y el tipo de etiqueta RFID que lee (pasiva o activa). El rango está entre los 2 cm. hasta 30 m. de distancia entre el lector y la etiqueta RFID, y depende de los factores mencionados anteriormente.

Los lectores se pueden encontrar en dos tipos:

- Sistemas con bobina simple**,** en donde la misma bobina sirve para transmitir la energía y los datos. Entre sus características se pueden mencionar que no son sistemas complejos; sin embargo poseen menos alcance.
- Sistemas interrogadores con dos bobinas, en donde una bobina sirve para transmitir energía y la otra para transmitir datos. Entre sus características se puede mencionar que a diferencia de los sistemas con bobinas simples, estos son más complejos, pero poseen mas alcance.

La computadora HOST: Esta parte del sistema RFID es la encargada del procesamiento de los datos que el circuito lector RFID lee y decodifica, así como también de la transmisión remota de los datos si es necesario (Ethernet, WLAN, ISDN, GPRS, etc.) el HOST puede tener las siguientes partes:

- Aplicación de software especifica, la cual será desarrollada para el uso que se halla determinado del sistema RFID, estos usos pueden ser: determinar accesos, guarda registros o rastrear actividad.
- Interfaz de programa de aplicación, esta también debe ser desarrollada específicamente para la aplicación que tenga el sistema RFID.
- Software de comunicación, este es el encargado del control, la transmisión y recepción de datos con el circuito lector RFID, esta comunicación puede ser llevada a cabo mediante varios estándares (USB, Bluetooth, PCMCIA, RS232, etc.).

2.2 MICROCONTROLADORES PIC

Los microcontroladores PIC[3] son una familia de microcontroladores tipo RISC[4] fabricados por Microchip Technology Inc.

El PIC usa un juego de instrucciones tipo RISC, cuyo número varia, desde 35 para PICs de gama baja a 70 para los de gama alta. Las instrucciones son de los siguientes tipos:

- Las que realizan operaciones entre el acumulador y una constante.
- Las que lo hacen entre el acumulador y una posición de memoria
- Instrucciones de condicionamiento y de salto/retorno.
- Implementación de interrupciones.
- Implementación del modo de bajo consumo de energía.

La compañía fabricante proporciona software de desarrollo para PC gratuito, cuyo nombre es MPLAB, el cual incluye un simulador software y un ensamblador. Otras empresas desarrollan compiladores C y BASIC, entre ellas están Mikroelektronica, CCS, SDCC y otras.

Todos los PICs manejan datos en trozos de 8 bits, todos menos los dsPIC[5], por lo que se deberían llamar microcontroladores de 8 bits. Pero a diferencia de la mayoría de CPUs, la arquitectura del PIC permite que el tamaño de las instrucciones pueda ser distinto del de la palabra de datos.

Las familias de los microcontroladores PIC se pueden ver en el siguiente diagrama de Funcionalidad y Desempeño.

[3] **PIC,** controlador de interfaz de periféricos.
[4] **RISC,** computadora con set de instrucciones reducido.
[5] **dsPIC**, PIC con capacidad de procesamiento digital de señales.

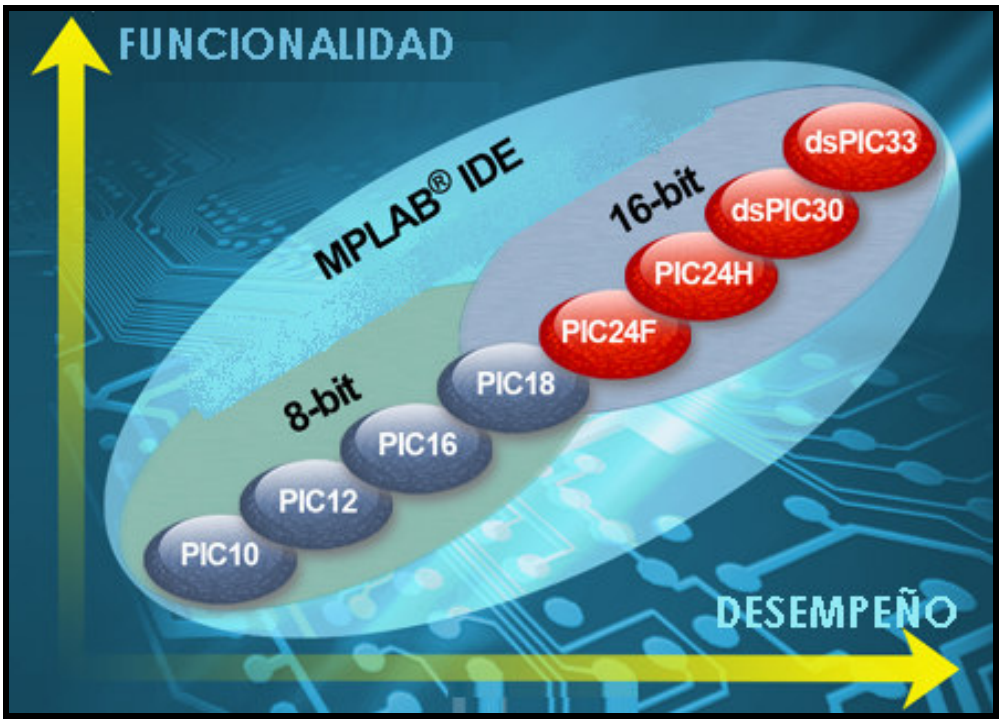


Figura 2.2.1 Familias de microcontroladores PIC, funcionalidad y desempeño.

Las características más comunes de los PICs son las siguientes:

- Puertos de E/S (típicamente 0 a 5,5 voltios)
- Memoria Flash y ROM disponible desde 256 bytes a 256 kilobytes
- Núcleos de CPU de 8/16 bits.
- Temporizadores de 8/16 bits
- Tecnología para modos de ahorro de energía
- Periféricos serie síncronos y asíncronos: USART (Transmisor/Receptor Síncrono/Asíncrono Universal), EUSART (Transmisor/Receptor Síncrono/Asíncrono Universal Mejorado)
- Conversores analógico/digital de 10-12 bits
- Comparadores de tensión
- Módulos de captura y comparación PWM (Modulación por ancho de pulso)
- Controladores LCD (Display de cristal liquido)

- Periférico MSSP para comunicaciones I²C (bus de comunicación serie de circuitos Inter-Integrados), SPI (Interfaz serial de Periféricos), y I²S (Inter-IC sound, es una interfaz de bus usada para conectar dispositivos de audio digital)
- Memoria EEPROM interna con duración de hasta un millón de ciclos de lectura/escritura
- Periféricos de control de motores
- Soporte de interfaz USB
- Soporte de controlador Ethernet
- Soporte de controlador Irda

2.3 BUS SERIE UNIVERSAL (USB)

Antes de definir el control OCX utilizado para el control del puerto USB, se dará una pequeña introducción de lo que es el Puerto USB y la clase que se utilizará, que es el estándar HID.

2.3.1 Descripción del USB

El USB o Universal Serial Bus es una interfaz para la transmisión serie de datos y distribución de energía desarrollado por empresas líderes del sector de las telecomunicaciones y de los ordenadores, y que ha sido introducida en el mercado de las computadoras personales y periféricos para mejorar las lentas interfaces serie rs-232 y paralelo. Provee una mayor velocidad de transferencia (de hasta 100 veces más rápido) comparado con el puerto paralelo de 25 pines y el Serial DB-9, DB-25, rs-232 que son los puertos que se encuentran en la mayoría de los computadores. Tenía en principio como objetivo el conectar periféricos relativamente lentos (ratones, impresoras, cámaras digitales, unidades zip, etc.) de una forma realmente sencilla, rápida y basada en comunicaciones serie, aunque por sus características también podía conectarse hasta discos duros.

Esta interfaz de 4 hilos distribuye 5V para la alimentación y puede transmitir datos a una velocidad de hasta 480 Mbps en su versión 2.0. Es un bus serie que hace posible la conexión de hasta 127 periféricos a una única puerta de un PC, con detección y configuración automáticas, siendo esto posible con el PC encendido, sin tener que instalar software adicional, y sin tener que reiniciar el ordenador (plug and play), algo que con los puertos convencionales serie y paralelo no sucedía. Tampoco hay que preocuparse por conflictos de IRQ's[6] o instalar tarjetas de adaptador para

[6] Pedido de Interrupción **IRQ** (Interrupt Request). También conocida como interrupción hardware, es una señal recibida por el procesador de un ordenador, indicando que debe "interrumpir" el curso de ejecución actual y pasar a ejecutar código específico para tratar esta situación.

cada periférico, estos periféricos pueden ser: Ratones, teclados, impresoras, escáneres, grabadoras, discos duros, módems, cámaras digitales, etc.

2.3.2 Características generales del USB

La especificación del USB proporciona una serie de características que pueden ser distribuidas en categorías. Estas características son comunes para todas las versiones (desde la 1.0 hasta la 2.0):

- Fácil uso para los usuarios
- Modelo simple para el cableado y los conectores
- Detalles eléctricos aislados del usuario (terminaciones del bus)
- Periféricos auto-identificativos
- Periféricos acoplados y reconfigurados dinámicamente (hot Swappable[7])
- Flexibilidad
- Amplio rango de tamaños de paquetes, permitiendo variedad de opciones de buffering[8] de dispositivos.
- Gran variedad de tasas de datos de dispositivos acomodando el tamaño de buffer para los paquetes y las latencias.
- Control de flujo para el manejo del buffer construido en el protocolo
- Ancho de banda isócrono[9]
- Se garantiza un ancho de banda y bajas latencias apropiadas para telefonía, audio, etc.
- Cantidad de trabajo isócrono que puede usar el ancho de banda completo del bus.
- Amplia gama de aplicaciones y cargas de trabajo
- Adecuando el ancho de banda desde unos pocos kbs hasta varios Mbs

[7] El término "**hot swap" o "hot swappable"** hace referencia a la capacidad de algunos componentes de hardware para sufrir su instalación o sustitución sin necesidad de detener o alterar la operación normal de la PC donde se alojan.
[8] Almacenamiento temporal; una ubicación de la memoria en una computadora o en un instrumento digital reservada para el almacenamiento temporal de información digital, mientras que está esperando ser procesada.
[9] Sinónimo de sincrónico. Que tiene un intervalo de tiempo constante entre cada evento.

- Soporta tanto el tipo de transferencia isócrono como el asíncrono[10] sobre el mismo conjunto de cables
- Conexiones múltiples, soportando operaciones concurrentes de varios dispositivos
- Soporta hasta 127 dispositivos físicos
- Soporta la transferencia de múltiples datos y flujos de mensajes entre el host y los dispositivos
- Robustez
- Manejo de errores y mecanismos de recuperación ente fallos implementados en el protocolo
- Inserción dinámica de dispositivos
- Soporte para la identificación de dispositivos defectuosos
- Implementación de bajo coste
- Sub canal de bajo coste a 1.5Mbs
- Conectores y cables de bajo coste
- Adecuado para el desarrollo de periféricos de bajo coste.

2.3.3 Arquitectura del bus USB

Para lograr un mínimo grado de comprensión del presente trabajo, es necesario analizar los elementos del bus USB desde el punto de vista de los sistemas de comunicaciones.

2.3.4 Elementos del USB

El USB es un bus ideado para intercambio de datos entre un computador anfitrión (Host), y dispositivos conectados a él (Esclavos). Los periféricos conectados

[10] Que no tiene un intervalo de tiempo constante entre cada evento.

al USB comparten el ancho de banda del bus mediante un protocolo basado en mensajes (tokens[11]).

Un sistema USB consta de 3 partes

- Anfitrión USB (USB Host o Host).
- Dispositivos USB (USB devices).
- Interconexión USB (USB interconnect).

Existe sólo un Host en un sistema USB. Los dispositivos USB proveen servicios o funciones al Host. La interconexión USB es la que soporta el tráfico entre el Host y los dispositivos USB, es el canal de comunicación.

2.3.5 Topología

La topología física del USB es de tipo estrella jerarquizada. Con un máximo de 7 niveles de jerarquía.

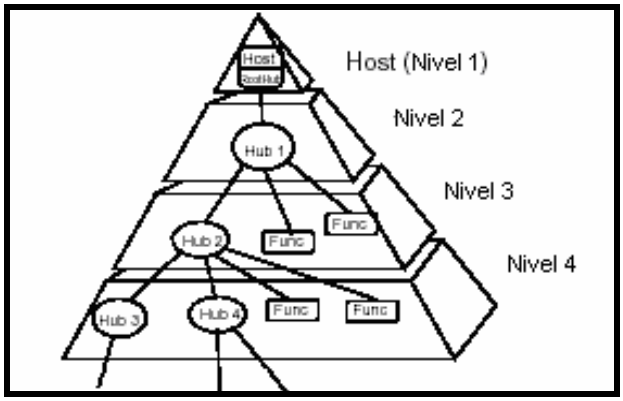


Figura 2.3.5.1 Capas del USB

La topología lógica del USB es de tipo estrella. Lo que implica que en un dispositivo físico puede haber implementado más de un dispositivo lógico (por ejemplo: un teclado con un mouse incluido).

[11] Es un bloque de texto categorizado. Por ejemplo una marca de puntuación, un operador, un identificador, un número, etc.

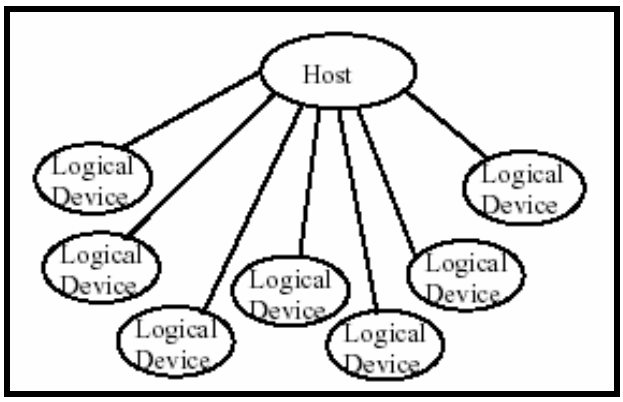


Figura 2.3.5.2

2.3.6 Transmisión y codificación

Los datos son transmitidos en forma serie, en 2 líneas de datos complementarias denominadas D+ y D-. Además se proveen 2 líneas de alimentación y de masa respectivamente, las cuales pueden servir para que el dispositivo tome alimentación del Host (5 V, 500 mA máx.). Para transmitir los datos en forma serie se utiliza la codificación Non-Return-To-Zero-Inverted o NRZI. En este tipo de codificación, un 0 (cero) se representa sin un cambio de nivel en la tensión, y un 1 (uno) se representa con un cambio de nivel en la tensión. Conjuntamente, se utiliza el bit stuffing[12], técnica que consiste en insertar un 0 (cero) cada 6 (seis) 1s (unos) consecutivos en el flujo de bits. Además, del bit stuffing y de la codificación NRZI, se utilizan CRCs[13]. Los CRCs se generan después del bit stuffing.

[12] Técnica de inserción de bit o de relleno de bit
[13] **CRC** o control de redundancia cíclica, es un mecanismo de detección de errores en sistemas digitales.

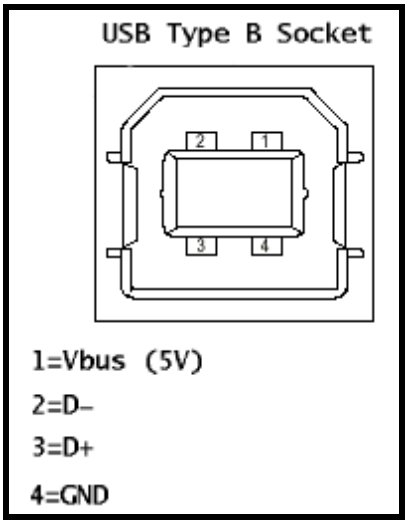


Figura 2.3.6 Conector USB de tipo B

2.3.7 Componentes y método

Acorde al objetivo propuesto, se comenzó por el estudio de la norma y se consultó bibliografía y artículos específicos sobre el tema. Como resultado de las investigaciones preliminares se decidió centrar el trabajo en los dispositivos de la clase Human Interface Device (HID). Estos poseen las características necesarias para que el usuario interactúe con una PC. Además, los drivers para los mismos están ya incluidos en el sistema operativo.

La ventaja que tiene el HID es que el sistema operativo Windows ya trae los controladores para dispositivos clase HID, eso ayuda que para hacer un dispositivo se requiriera un mínimo de firmware, lo que vuelve más simple la utilización del USB. Algunas desventajas que tiene el HID es que solo se puede trasmitir datos a 64KB/s eso es mucho menor de la capacidad del USB en full-speed que son de 12Mbits/s, pero aun así es más rápido que los puertos serial (RS232) y paralelo.

Se seleccionó un circuito integrado controlador USB para construir el hardware necesario para la realización del trabajo. La función del mismo es, por un lado proveer los niveles eléctricos de las señales que intervienen en la comunicación, y por otro llevar el control de la misma acorde al protocolo, facilitando así la conexión con

un microcontrolador de uso habitual. Hecho esto se pasó a diseñar las rutinas básicas del software del microcontrolador para lograr la correcta interconexión con el controlador USB. Esta es la primera etapa para lograr una comunicación con la PC. Cuando un dispositivo USB se conecta a una PC comienza un dialogo entre ellos conocido como *enumeración*, en el cual la PC interroga al dispositivo para identificarlo y conocer sus características, por ejemplo: tipo de dispositivo, forma en que envía la información, etc. Se amplió el software para que lleve a cabo la enumeración de forma tal que el dispositivo fuera reconocido por la PC.

Una vez logrado esto quedó sentada la base para agregar funcionalidad al dispositivo, ya que a grandes rasgos se puede decir que esta parte es común para una amplia gama de los mismos.

2.3.8 Clase HID

El nombre HID es la abreviatura de "Human Interface Devices". Esta Clase, cuya versión actual es el estándar HID 1.11 fue ideada con el propósito de englobar a dispositivos que permitan la interacción del usuario (ser humano) con el Host. Por lo tanto, los requerimientos de ancho de banda son mínimos, y la transferencia de datos debe ser confiable.

Los datos que los dispositivos HID envían al Host son interpretados por el "HID Class Driver" del sistema operativo, para luego poder ser utilizados por la aplicación que los requiera (Client Software). Los requisitos para la implementación de un dispositivo HID son:

- Control Endpoint (Endpoint0): obligatorio
- Interrupt IN Endpoint: obligatorio
- Interrupt OUT Endpoint: opcional

2.3.9 Facilidades para el desarrollo de dispositivos USB

Microchip, unos de los más importantes fabricantes de microcontroladores tiene entre su gama alta a la familia 18F2455/2550/4455/4550 dispositivos que entre sus múltiples periféricos cuentan con una interfaz USB 2.0 con compatibilidad para LS y FS que permite una complejidad moderada a la hora de realizar una interfaz utilizando este puerto.

Estos dispositivos cuentan con el hardware necesario para realizar una conexión USB a baja velocidad (Low speed o LS) o velocidad completa (full speed o FS), las operaciones de protocolo deben ser programadas y consumen importantes recursos del sistema como lo son algunas interrupciones y muchos ciclos de máquina de tal forma que sería necesario incluir en el programa un complejo set de rutinas para controlar el puerto USB, esta situación incrementa la complejidad al desarrollar aplicaciones donde se requiere alto rendimiento y es crítico un control exacto del tiempo, por ejemplo en un sistema de adquisición temporizado internamente, no obstante no se debe descartar este dispositivo a la hora de desarrollar pues se deben tener en cuenta la enormes ventajas que ofrece este, entre las cuales encontramos un bajo costo, altísima calidad, flexibilidad y el invaluable respaldo y soporte técnico de su fabricante Microchip.

2.4 CONTROL DE PUERTOS USB PARA DISPOSITIVOS CON MICROCONTROLADORES PIC[14]

2.4.1 Control OCX[15] HIDComm para Visual Basic 6.0

El control de ActiveX[16] HIDComm esta diseñado para facilitar el proceso de comunicación de la PC a los dispositivos HID[17] USB. Sin el control de ActiveX HIDComm, lo usuarios deberán utilizar formas mas complicadas para comunicarse con el dispositivo.

2.4.2 Instalación

El Control de ActiveX HIDComm viene en un archivo fácil de correr "SETUP.EXE" para instalar solo hay que correr este ejecutable, el programa de instalación creara unos cuantos accesos directos en el botón de inicio, para usar el control ActiveX simplemente se debe iniciar Visual Basic 6.

Cuando se aparezca la ventana de nuevo proyecto, se debe elegir "Standard.EXE". Visual Basic creará el proyecto, en la parte izquierda de la pantalla se encuentra una ventana donde están todos los controles ActiveX instalados, como se muestra a continuación.

[14] Adaptado y traducido de la ayuda del control OCX HidComm para Visual Basic. http://www.microchip.com descargado en junio del 2004, ya no está disponible.
[15] **OCX,** módulo de programa independiente que puede ser accesado por otros programas en un ambiente de Windows.
[16] **ActiveX,** es un conjunto de tecnologías desarrolladas por Microsoft que permiten a los componentes de software interactuar entre sí en un ambiente conectado de red, como Internet, independientemente del lenguaje de desarrollo en que fueron creados. Los elementos fundamentales de la tecnología ActiveX son el Modelo de objetos componentes (COM) y el Modelo de objetos componentes distribuido (DCOM).
[17] **HID,** dispositivo de interfaz de humano.

Figura 2.4.2.1 Controles ActiveX instalados en VB 6.0

Se debe hacer click derecho en un punto vacío y elegir "Components". Una ventana aparecerá, hay un ítem con el nombre "HIDComm ActiveX Control" se debe poner un check en la cajita y hacer click en OK para cerrar la ventana. Ahora el control ActiveX debe aparecer en la ventana de componentes, para usarlo simplemente hay que arrastrarlo a un formulario que este siendo diseñado.

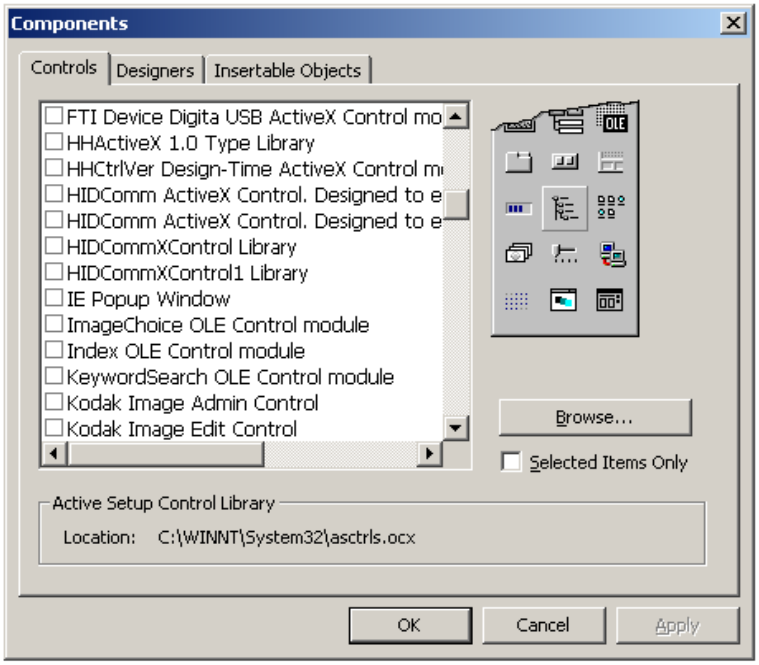


Figura 2.4.2.2 Ventana de componentes disponibles en VB 6.0

2.4.3 Preparando el dispositivo USB

El control de ActiveX HIDComm esta diseñado para facilitar la comunicación entre la PC y un dispositivo USB HID usando microcontroladores PIC, para mas detalles de cómo programar estos dispositivos se puede ver la sección de este trabajo dedicado a los microcontroladores PIC y referirse a la pagina web del fabricante Microchip.

2.4.4 Ajustando los Criterios de Concordancia

La caja de criterios de concordancia, como se muestra a continuación, habilita al usuario para que éste pueda ajustar de manera rápida los criterios de búsqueda para encontrar el dispositivo HID. El usuario puede llenar cada campo con sus criterios de búsqueda, y habilitar dicho criterio poniendo un cheque en la checkbox de al lado. El usuario puede también buscar a través de los dispositivos HID conectados a la PC haciendo clic en el botón "browse", y escogiendo el dispositivo deseado.

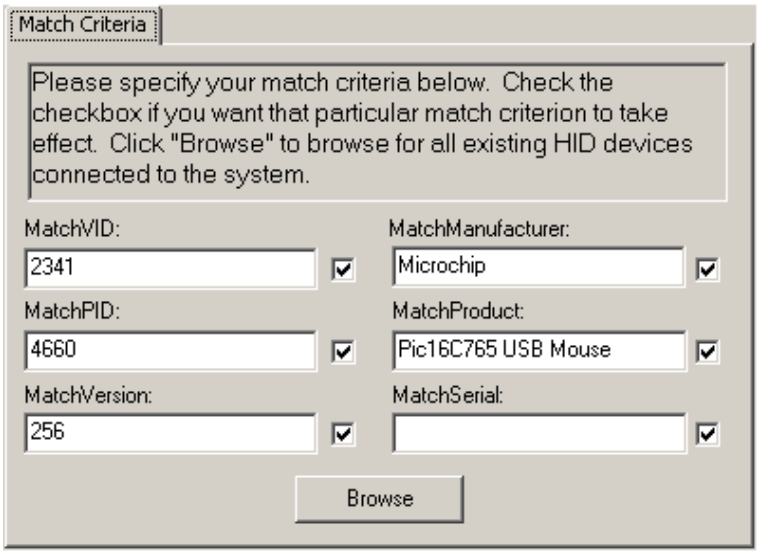


Figura 2.4.4 Criterios de concordancia

El significado de cada campo en la caja de criterios es el siguiente:

MatchPID: es uno de los criterios que se pueden especificar para localizar un dispositivo HID, por ejemplo si se quiere conectar a un dispositivo cuyo Product ID es 1234, se debe especificar 1234 en este campo.

MatchVID: es otro de los criterios que se pueden especificar para localizar un dispositivo HID, por ejemplo si se quiere conectar a un dispositivo cuyo Vendor ID es 1234, se debe especificar 1234 en este campo.

Match Version: es otro de los criterios que se pueden especificar para localizar un dispositivo HID, por ejemplo si se quiere conectar a un dispositivo cuya versión es 1.00, se debe especificar 256 en este campo porque la versión esta expresada en un numero BDC.

MatchManufacturer: es otro de los criterios que se pueden especificar para localizar un dispositivo HID, por ejemplo si se quiere conectar a un dispositivo cuyo fabricante es "Microchip", se debe especificar "Microchip" en este campo.

MatchProduct: es otro de los criterios que se pueden especificar para localizar un dispositivo HID, por ejemplo si se quiere conectar a un dispositivo que tiene por nombre de Producto "Mouse", se debe especificar "Mouse" en este campo.
MatchSerial: es otro de los criterios que se pueden especificar para localizar un dispositivo HID, por ejemplo si se quiere conectar a un dispositivo cuyo serial es 1234ABCD, se debe especificar 1234ABCD en este campo.

2.4.5 Ejemplo de uso de Entradas y Salidas

Después de colocar el control de ActiveX HIDComm en un formulario vacío se debe revisar la ventana de propiedades.

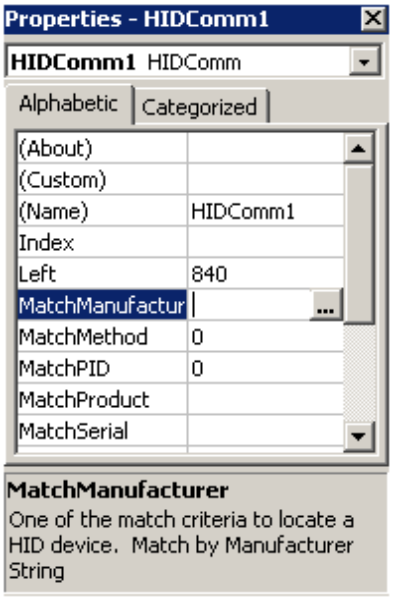


Figura 2.4.5.1 Ventana de Propiedades

Se puede observar que todas las propiedades de tiempo de diseño se muestran aquí, abajo se puede observar una breve descripción de la propiedad siendo editada actualmente. En este caso el MatchManufacturer.

Se debe hacer click en el set de propiedades de MatchManufacturer, después hacer click en el botón de 3 puntos; una interfaz de usuario se mostrara en donde se puede ajustar los criterios de match.

Posteriormente se debe conectar el PIC programado a la PC, cuando el dispositivo USB es conectado por primera vez, la PC desplegara el aviso de "Nuevo Hardware Encontrado" y se buscaran automáticamente los drivers necesarios, no hay necesidad de un driver especial ya que el que trae Windows funciona.

Se debe hacer click en el botón "Browse" en la ventana de criterios de Match, el dispositivo debe ser presentado. Se debe elegir el dispositivo y hacer click en "OK", todos los criterios de Match deberían ser llenados automáticamente. Se debe

tener en cuenta que un control ActiveX solo puede comunicarse con un dispositivo a la vez.

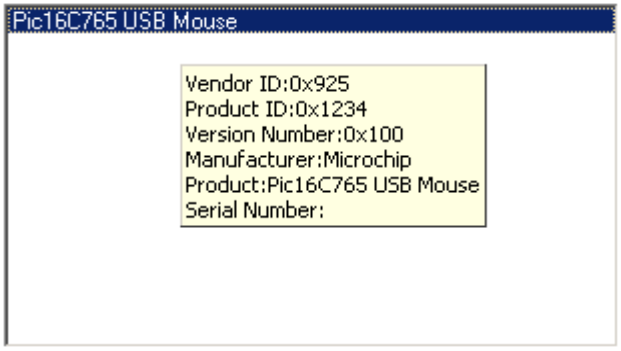


Figura 2.4.5.2 Selección de dispositivo

Los comandos más importantes en el código fuente del formulario son:

"HIDComm1.Connect": le dice a HIDComm que se conecte al dispositivo HID. HIDComm usará los criterios de match para localizar el dispositivo.

"HIDComm1.WriteTo Buffer BufferSize": envía el buffer al dispositivo USB. El BufferSize especifica el número de bytes a ser enviados, este se debe declarar como variable Long.

"HIDComm1.ReadFrom(BufferSize)": lee de el dispositivo USB. El parámetro BufferSize especifica el máximo número de bytes que se puede recibir. Los bytes recibidos son guardados en el buffer, y el número de bytes leídos son guardados en BufferSize.

"HIDComm1.Uninit": es crítico agregar esta llamada a subrutina en el evento de "Unload" del formulario para desinicializar el control ActiveX

2.5 PROGRAMACION EN C PARA PICS USANDO EL COMPILADOR CCS[18]

2.5.1 Descripción de los compiladores PCB, PCM, y PCH

PCB esta hecho para los opcodes[19] de 12 bits, PCM esta hecho para los opcodes de 14 bits, y PCH esta hecho para los opcodes de 16 bits, todos son para opcodes de microcontroladores PIC. Debido a que son muy similares, es la misma información para los tres, estos compiladores están específicamente diseñados para cumplir con las necesidades únicas de los microcontroladores PIC, esto permite a los desarrolladores diseñar rápidamente aplicaciones de software de una manera mas fácil, en lenguaje de alto nivel.

Cuando se compara con compiladores C más tradicionales, PCB, PCM y PCH tienen algunas limitantes. Como ejemplo de estas limitaciones tenemos, que la recursión de funciones no esta permitida. Esto debido al hecho de que los PICs no tienen pilas para guardar los valores de las variables actuales. Los compiladores pueden implementar de manera eficiente construcciones, operaciones de entrada/salida, y operaciones de manejo de bits. Todos los tipos de datos de C son soportados así como también punteros, arreglos de constantes, decimales de punto fijo, y arreglos de bits.

2.5.2 Estructura Principal

Un programa esta formado de los siguientes cuatro elementos en un archivo:

- Comentario
- Directiva pre-procesador
- Definición de datos
- Definición de función

[18] Adaptado y traducido de la ayuda del compilador CCS para C. http://www.ccsinfo.com descargado diciembre del 2006.

[19] **Opcodes,** código de operación, es la porción de una palabra de instrucción que especifica que tipo de instrucción es.

Todo programa en C debe contener una función "main", que es el punto de inicio del programa en ejecución. El programa puede ser partido en múltiple funciones de acuerdo con su propósito y las funciones pueden ser llamadas desde la función "main" o desde otras subfunciones. En grandes funciones de proyecto las subfunciones pueden ser colocadas en diferentes archivos de C o encabezado de archivos, y estos pueden ser incluidos en el archivo "main" de C. CCS C también requiere la inclusión del archivo de dispositivo apropiado, usando la directiva #include para incluir las funcionalidades del dispositivo a usar. También hay unas directivas de procesador como #fuses para especificar los fusibles del chip, y #use para especificar la velocidad del clock. Las funciones contienen las declaraciones de datos, definiciones, declaraciones, y expresiones. El compiladote también provee un gran número de librerías Standard de C, así como también drivers para otros dispositivos que pueden ser incluidos y usados en los programas. CCS también provee un gran número de funciones incorporadas para acceder a varios periféricos incluidos en los microcontroladores PIC.

2.5.3 Declaraciones

Las principales declaraciones del compilador CCS para C, se muestran en la siguiente tabla:

Declaracion	Ejemplo

if (expr) stmt; [else stmt;]	`if (x==25)`
	`  x=1;`
	`else`
	`  x=x+1;`

while (expr) stmt;	`while (get_rtcc()!=0)` `  putc('n');`

do stmt while (expr);	`do {`
	`  putc(c=getc());`
	`} while (c!=0);`

for (expr1;expr2;expr3) stmt;	`for (i=1;i<=10;++i)` `  printf("%u\r\n",i);`

switch (expr) {
expr de caso: stmt; //uno o mas casos [default:stmt]
... }
	`switch (cmd) {`
	`  case 0:  printf("cmd 0");`
	`  break;`
	`  case 1: printf("cmd 1");`
	`  break;`
	`  default: printf("bad cmd");`
	`  break; }`

return [expr];	`return (5);`
goto label;	`goto loop;`
label: stmt;	`loop: I++;`
break;	`break;`
continue;	`continue;`
expr;	`i=1;`
;	`;`
{[stmt]}	`{a=1;` `b=1;}`
Cero o mas	

Tabla 2.5.3.1 Declaraciones del compilador CCS

2.5.4 Operadores

Los operadores principales del compilador CCS para C, se muestran en la siguiente tabla:

Operador	Definición
+	Operador suma
+=	Operación de suma y resultado, x+=y
&=	Operador Bitwise y resultado
&	Operador de dirección
&	Operador Bitwise
^=	Bitwise exclusive or assignment operator, x^=y
^	Bitwise exclusive or operator
l=	Bitwise inclusive or assignment operator, xl=y
l	Bitwise inclusive or operator
?:	Operador de expresión condicional
- -	Decremento
/=	Operador División y resultado, x/=y
/	Operador División
==	Igualdad
>	Operador Mayor que
>=	Operador Mayor o igual que
++	Incremento
*	Operador de Indirección
!=	Inigualdad
<<=	Operador de rotación a la izquierda y resultado
<	Operador Menor que
<<	Operador de rotación a la izquierda
<=	Operador Menor o igual que
&&	Operador lógico AND
!	Operador Lógico de Negación
ll	Operador Lógico OR
%=	Operador de asignación de módulos x%=y
%	Operador de Módulos
*=	Operador de Multiplicación y resultado
*	Operador de Multiplicación
~	Operador de Complemento a Uno

>>=	Operador de rotación a la derecha y resultado
>>	Operador de rotación a la derecha
->	Operación de puntero de Estructura
-=	Operador de resta y resultado
-	Operador de Resta
sizeof	Determina el tamaño en bytes del operador

Tabla 2.5.4.1 Operadores del compilador CCS

2.5.5 Definiciones de Datos

Los Principales tipos de datos del compilador CCS para C, se muestran en la siguiente tabla:

Dato	Definición
int1	Define un numero de 1 bit
int8	Define un numero de 8 bits
int16	Define un numero de 16 bits
int32	Define un numero de 32 bits
Char	Define un caracter de 8 bits
Flota	Define un numero de punto flotante de 32 bits
Short	Por predeterminación es igual que int1
Int	Por predeterminación es igual que int8
Long	Por predeterminación es igual que int16
Void	Indica un tipo no especifico

Tabla 2.5.5.1 Tipos de datos del compilador CCS

2.5.6 Declaraciones

Una declaración especifica un tipo de calificador y un tipo de especificador, y es seguido por una lista de uno o más variables de ese tipo, por ejemplo:

int a,b,c,d;

mybit e,f;

mybyte g[3][2];

*char *h;*

colors j;

struct data_record data[10];

static int i;

extern long j;

2.5.7 Uso de la memoria de programa para datos

El compilador C de CCS permite usar la memoria de datos de varias maneras. Estas se discuten a continuación:

Datos Constantes:

Un calificador constante coloca las variables en la memoria de programa. La sintaxis es "const type especif id [cexpr] = {value}" si la palabra clave CONST es usada antes de identificador, el identificador es tratado como constante. Las constantes deben ser inicializadas y no deben ser cambiadas en la ejecución.

Por ejemplo:

const char cstaring[6]={"hello"}	Para colocar una cadena en la memoria ROM
*const char *cptr;*	Para crear punteros de constantes
cptr = string;	
#ORG 0x1C00, 0x1C0F	El comando #org puede ser usado para
CONST CHAR ID[10]={"123456789"};	colocar una constante en bloques de direcciones especificados, para el ejemplo, esta ID estará en la dirección 1C00

La function "label_address" puede usarse para obtener la dirección de una constante. La variable constante puede ser acezada en el código. Esta es una buena manera de guardar datos constantes en programas grandes. Cadenas constantes de tamaño variado pueden ser guardadas en la memoria de programa.

Para PIC18 el compilador permite de una manera no standard en C, la implementacion de un arreglo constante de cadenas de tamaño variable. La sintaxis es la siguiente:

const char id[n] [*] = { "strint", "string" ...};

Donde n es opcional, y id es el identificador de la tabla. Como por ejemplo:

const char colors[] [*] = { "Red", "Green", "Blue"};

Directiva #ROM:

Otro método es usar #rom para asignar datos a la memoria de programa, el uso es #rom address={data,data,…,data}. Por ejemplo:

#rom 0x1000={1,2,3,4,5} //pondra 1,2,3,4,5 en la memoria rom comenzando en 0x1000

#rom address={"hello"} // pondra hello en la memoria rom

Funciones Incorporadas:

El compilador tambien provee funciones incorporadas para colocar datos en la memoria de programa, las funciones son las siguientes:

- write_program_eeprom(address,data); escribe datos de 16 bit en la memoria de programa.
- write_program_memory(address, dataptr, count); escribe count bytes de datos desde dataptr a address en la memoria de programa. Estas funciones pueden ser usadas solo en chips que permitan escribir en la memoria de programas. El compilador usa rutinas de escritura y borrado de la memoria flash para esta funcionalidad.

Los datos colocados en la memoria de programa usando los tres métodos anteriores pueden ser leídos desde el código de usuario con:

- read_program_eeprom(address); lee datos de 16 bits de address en la memoria de programa.
- read_program_memory((address, dataptr, count); Lee count bytes de la memoria de programa en address a la RAM en dataptr, esta function puede ser usada solo en chips que permitan la lectura de la memoria de programa. El compilador usa rutinas de lectura de la flash para implementar esta funcionalidad.

2.5.8 Funciones Incorporadas del compilador CCS

El compilador CCS tiene funciones incorporadas que sirven para utilizar de manera rapida los subsistemas que esten disponibles en el modelo de PIC que se este usando, los subsistemas y las funciones incorporadas para el uso de estaos, son loas siguientes:

RS232 I/O	ASSERT( ) FGETC( ) FGETS( ) FPRINTF( ) FPUTC( ) FPUTS( )	GETCH( ) GETCHAR( ) GETS( ) KBHIT( ) PERROR( ) PRINTF( )	PUTC( ) PUTCHAR( ) PUTS( ) SET_UART_SPEED( ) SETUP_UART( )	
SPI DOS CABLES I/O	SETUP_SPI( ) SPI_XFER( )	SPI_DATA_IS_IN( )	SPI_READ( )	SPI_WRITE( )
I/O DISCRETAS	GET_TRISx( ) INPUT( ) INPUT_A( ) INPUT_B( ) INPUT_C( )	INPUT_K( ) INPUT_STATE( ) INPUT_x( ) OUTPUT_A( ) OUTPUT_B( )	OUTPUT_FLOAT( ) OUTPUT_G( ) OUTPUT_H( ) OUTPUT_HIGH( ) OUTPUT_J( )	SET_TRIS_B( ) SET_TRIS_C( ) SET_TRIS_D( ) SET_TRIS_E( ) SET_TRIS_F( )
ESCLAVO PARALELO I/O	PSP_INPUT_FULL( ) PSP_OUTPUT_FULL( )		PSP_OVERFLOW( ) SETUP_PSP( )	
I²C I/O	I²C_ISR_STATE() I²C_POLL( ) I²C_READ( )	I²C_SlaveAddr( ) I²C_START( ) I²C_STOP( )	I²C_WRITE( )	
CONTROLES DE PROC.	CLEAR_INTERRUPT( ) DISABLE_INTERRUPTS( ) ENABLE_INTERRUPTS( ) EXT_INT_EDGE( ) GETENV( )	GOTO_ADDRESS( ) INTERRUPT_ACTIVE( ) JUMP_TO_ISR LABEL_ADDRESS( ) READ_BANK( )	RESET_CPU( ) RESTART_CAUSE( ) SETUP_OSCILLATOR( ) SLEEP( ) WRITE_BANK( )	
MANIPULACION BIT/BYTE	BIT_CLEAR( ) BIT_SET( ) BIT_TEST( )	MAKE8( ) MAKE16( ) MAKE32( )	_MUL( ) ROTATE_LEFT( ) ROTATE_RIGHT( )	SHIFT_LEFT( ) SHIFT_RIGHT( ) SWAP( )
MATEMATICA STANDARD C	ABS( ) ACOS( ) ASIN( )	COSH( ) DIV( ) EXP( )	LABS( ) LDEXP( ) LDIV( )	SIN( ) SINH( ) SQRT( )
VOLTAJE DE REF	SETUP_LOW_VOLT_DETECT( )		SETUP_VREF( )	
CONVERSION A/D	SET_ADC_CHANNEL( ) SETUP_ADC( )		SETUP_ADC_PORTS( ) READ_ADC( )	
CHAR STANDARD C	ATOF( ) ATOI( ) ATOI32( )	ISLOWER(char) ISPRINT(x) ISPUNCT(x)	STRCMP( ) STRCOLL( ) STRCPY( )	STRRCHR( ) STRSPN( ) STRSTR( )
TIMERS	GET_TIMER0( ) GET_TIMER1( ) GET_TIMER2( ) GET_TIMERx( ) RESTART_WDT( )	SET_RTCC( ) SET_TIMER0( ) SET_TIMER1( ) SET_TIMER5( ) SETUP_COUNTERS( )	SETUP_TIMER_0 ( ) SETUP_TIMER_1 ( ) SETUP_TIMER_2 ( ) SETUP_WDT ( )	
MEMORIA STANDARD C	CALLOC( ) FREE( ) LONGJMP( ) MALLOC( ) MEMCHR( )	MEMCMP( ) MEMCPY( ) MEMMOVE( ) MEMSET( ) OFFSETOF( )	OFFSETOFBIT( ) REALLOC( ) SETJMP( )	
CAPTURA/COMPARA/PWM	SET_POWER_PWM_OVERRIDE( ) SET_POWER_PWMX_DUTY( ) SET_PWM1_DUTY( ) SET_PWM2_DUTY( )		SETUP_CCP2( ) SETUP_CCP3( ) SETUP_CCP4( ) SETUP_CCP5( )	
EEPROM INTERNA	ERASE_PROGRAM_EEPROM( ) READ_CALIBRATION( ) READ_EEPROM( ) READ_EXTERNAL_MEMORY( )		SETUP_EXTERNAL_MEMORY( ) WRITE_CONFIGURATION_MEMORY( ) WRITE_EEPROM( ) WRITE_EXTERNAL_MEMORY( )	

	READ_PROGRAM_EEPROM( ) READ_PROGRAM_MEMORY( )	WRITE_PROGRAM_EEPROM( ) WRITE_PROGRAM_MEMORY( )	
RTOS	RTOS_AWAIT( ) RTOS_DISABLE( ) RTOS_ENABLE( ) RTOS_MSG_POLL( ) RTOS_MSG_READ( )	RTOS_MSG_SEND( ) RTOS_OVERRUN( ) RTOS_RUN( ) RTOS_SIGNAL( ) RTOS_STATS( )	RTOS_TERMINATE( ) RTOS_WAIT( ) RTOS_YIELD( )
LCD	LCD_LOAD( )	LCD_SYMBOL( )	SETUP_LCD( )

Tabla 2.5.8.1 Funciones incorporadas en el compilador CCS

2.6 INICIO DE SESION Y AUTENTICACION EN WINDOWS XP[20]

2.6.1 Proceso de inicio en Windows XP

Uno de los procesos más ocultos y críticos que existe en un sistema operativo grafico como Windows XP, es el de inicio del sistema o Booting, dado que en él se define el comportamiento posterior del sistema.

Inicializar el sistema cuando se enciende un equipo es un proceso directo para la mayoría de los usuarios, pero cuando se habla de administración se debe conocer éste en detalle para poder resolver muchos de los problemas relacionados con el arranque y configuración de dispositivos.

El primer paso en todo el proceso de encendido de un sistema operativo es el denominado POST (Power-on Self Test) el cual abarca una comprobación eléctrica del sistema que es realizado por la configuración del BIOS[21], y abarca los siguientes pasos:

- PROM Checksum: Esta verificación se hace para validar que la información almacenada en la memoria ROM del sistema sea igual a la información de los dispositivos instalados. Es una suma de comprobación.
- Segment Map Address: En este punto se prepara la memoria para recibir la información; se envía una serie de códigos que luego son leídos para comprobar que la memoria es completamente funcional.
- Page Map Address: Divide la memoria en segmentos de paginación, y se realiza la comprobación de paginación.

[20] Seccion tomada de: http://www.creangel.com/drupal/?q=node/116
[21] **BIOS (Basic Input/Output System)** Programa que se almacena generalmente en una EPROM y que es utilizado por la CPU para ejecutar los procedimientos de arranque cuando se enciende la computadora.

- Se lee la información de la memoria C-MOS[22] para ser utilizada en el resto del proceso.

Una ves el sistema ha realizado las comprobaciones eléctricas y físicas (POST) se procede a buscar el sistema operativo que se desea cargar, en muchos sistemas se puede definir el orden de la búsqueda, este orden corresponde al orden de arranque en la información almacenada en la BIOS. Depende del orden definido, la secuencia tomará caminos diferentes, en general se dice que la mayoría de los sistemas se pueden programar para que hagan la búsqueda en los siguientes dispositivos:

- Floppy disk
- CD-ROM
- Disco duro
- Adaptadores de red (si el adaptador soporta este sistema)
- Almacenamientos externos.

Cuando se intenta iniciar el sistema por la unidad de disco duro se busca el MBR (Master Boot Record) que son los primeros 512 bytes de todo disco duro, en donde se encuentra la información que le dice al sistema como proceder para lograr el inicio del sistema; adicionalmente contiene los datos de la tabla de partición.

Desde este punto se procede a la inicialización del sistema operativo como tal, este proceso conlleva la carga en un orden específico de una serie de archivos; y el cambio de modo de operación del procesador es el primer paso. Se asumirán los procesadores X86 como base de estudio dado que son actualmente la mayoría. Al Inicio, después de encendido el equipo o reiniciado el procesador arranca en modo "Real" esto implica que el procesador es puesto en modo de compatibilidad, lo que quiere decir, que le brinda soporte a aplicaciones desarrolladas para 8 Bits y 16 bits,

[22] **C-MOS (Complementary Metal Oxide Semiconductor)** pequeña porción de memoria en donde esta almacenadas todas las configuraciones de la computadora.

así se ofrece compatibilidad hacia atrás con aplicaciones antiguas y DOS. Para poder aprovechar todo el rendimiento del procesador se debe cambiar de modo "Real" a modo "Extendido" o Enhanced; este cambio en los procesadores de familia X86 se hace por medio de software y se puede realizar sin necesidad de reiniciar el equipo, esto fue posible desde la serie 486 de Intel. El cambio es realizado por el programa NTLDR[23]. Al realizarlo, se puede además tener acceso a porciones superiores de memoria puesto que al inicio, el sistema solo tiene acceso a los primero 640 Kb de memoria.

Carga del sistema de archivos

La carga del sistema de archivos se realiza después que el sistema ha cambiado de modo y tiene un procesador de 32bits disponible; en este punto es que en realidad se puede acceder a la estructura de árbol de la partición y se puede proceder a la lectura de toda la configuración.

Carga de manejador de BOOT.INI

Este archivo permite seleccionar que sistema operativo se desea iniciar dependiendo de cuantos sistemas se tienen instalados en el computador. Si el equipo solo posee un sistema operativo, se procede a la fase de detección de hardware que permite a la parte lógica reconocer la parte física en el equipo.

Detección de hardware

Este proceso se lleva acabo por medio del archivo NTDETECT.COM el cual da la información necesaria al KERNEL de Windows para iniciar de manera apropiada. Entre otros datos este sistema busca los perfiles de hardware que se han podido crear y brinda la posibilidad de selección de los mismos para cargar la configuración

[23] **NTLDR** (NT Loader) es el Boot Loader de Windows NT, incluyendo algunas versiones posteriores Windows 2000/XP/Server 2003.

mas apropiada. Adicionalmente se hace la detección de los sistemas de ahorro de energía tipo ACPI[24] incorporados en las torres ATX[25] y en los equipos portátiles.

Carga de KERNEL

La carga del Kernel[26] de sistema operativo inicia por ejecutar el sistema de abstracción denominado HAL (Hardware Abstraction Layer o capa de abstracción de hardware) y esta ubicado en el archivo HAL.DLL.

Carga de dispositivos

El proceso de carga de dispositivos, comprende la información de activación y la activación en sí de éstos. Toda la información para dicho procesos se encuentra almacenada en un conjunto de entradas dentro del registro, bajo la llave HKEY_LOCAL_MACHINESYSTEM, entrada a la cual se le hace una copia o clonación que permite ir validando y modificando el estado de cada dispositivo, todo esto para asegurar el correcto funcionamiento del sistema y de los dispositivos individuales.

Manejador de sesión

Una vez se han cargado los dispositivos, se inicia el manejador de sesión, que es el encargado de pasar de modo texto a modo grafico, y posee otras funcionalidades, como lo son:

- Creación de las variables de ambiente

[24] **ACPI,** (Advanced Configuration and Power Interface**)** es un estándar resultado de actualización a nivel de hardware, que controla el funcionamiento del BIOS y proporciona mecanismos avanzados para la gestión y ahorro de la energía.
[25] **ATX,** (Advanced Technology Extended) estandar creado por Intel en 1995, en el cual la motherboard fue rotada noventa grados y cada slot puede almacenar una tarjeta de expansion de tamaño completo.
[26] **KERNEL**, Es el núcleo o la parte fundamental de un sistema operativo, es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma más básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema.

- Iniciar el modo del Kernel (ubicado en systemrootSystem32Win32k.sys) que se encarga de cambiar de modo texto a modo grafico, para que las aplicaciones basadas en Windows puedan ser ejecutadas.
- Inicia el modo de usuario que permitirá la interacción del usuario con el sistema, esto implementado en *systemroot*System32Csrss.exe.
- Inicia al manejador de inicio de sesión (winlogon), ubicado en *systemroot*System32Winlogon.exe.
- Crea memoria virtual adicional, para cargar más información del sistema en operación.
- Ejecuta acciones tardías para operaciones del registro, como la de reiniciar el sistema por cambios en el mismo (instalaciones de nuevos dispositivos encontrados u otros que requieran reiniciar el sistema después de su configuración).

El administrador de sesión se encarga adicionalmente de buscar en el registro varias llaves de configuración e inicio para procesarlas y así terminar el arranque del sistema. Las llaves son las siguientes:

HKEY_LOCAL_MACHINESYSTEMCurrentControlSetControlSession Manager: Procesa las entradas previas al inicio de servicios, así como la herramienta de verificación de discos Autochk.exe, para garantizar la integridad del disco y las particiones.

HKEY_LOCAL_MACHINESYSTEMCurrentControlSetControlSession ManagerSubsystems: Inicia los subsistemas de modo de usuario Csrss.exe.

HKEY_LOCAL_MACHINESYSTEMCurrentControlSetServices*serviceman:* Es la sección encargada de iniciar los servicios de auto carga dentro del sistema.

Sistemas de Seguridad

Cuando se inician los procesos, cada uno de ellos, tiene una entrada de control de acceso ACL,[27] que identifica los permisos que tiene el mismo en el sistema, pero esto depende si el sistema es parte de un dominio o no. Para lograr que el sistema tome las credenciales, se debe cargar un componente específico que permita la carga de los descriptores de los archivos, este componente se llama SRM (Security referente monitor, o monitor de referencia de seguridad), el cual se inicia en el modo de KERNEL y se comunica con el modo de usuario.

2.6.2 WINLOGON

Winlogon[28] maneja las funciones del interfaz de Windows que son independientes de la política de autentificación. Crea los escritorios para la estación de windows, operaciones del descanso de los instrumentos y en versiones de Windows previas a Windows Vista proporciona un grupo de funciones de soporte para el GINA[29] y es responsable de la configuración la política de grupo de la máquina y de usuario. Winlogon también comprueba si la copia de Windows es una licencia legítima en Windows XP y versiones más posteriores.

Winlogon tiene las responsabilidades siguientes:

- Protección del escritorio y de la estación Windows
- Reconocimiento de Standard SAS[30]
- Envío de rutinas SAS
- Carga del perfil de usuario

[27] **ACL** (Access Control List), La lista de control de acceso, es un concepto de seguridad informática usado para fomentar la separacion de privilegios.
[28] **Winlogon,** es el componente de los sistemas operativos de Windows Microsoft responsable de manejar la llave de atención segura o secure attention key, encargada de cargar el perfil de usuario en la conexión. La obtención y la verificación reales de las credenciales del usuario se deja a otros componentes.
[29] **GINA** (graphical identification and authentication library), es un componente de algunos sistemas operatives de Microsoft Windows que prove autenticacion segura e interactive con los servicios de logon. GINA es una Dinamically Linked Library o DLL que es cargada en el proceso del Winlogon cuando la computadora se inicia.
[30] **SAS (Serial Attached SCSI),** es una interfaz de transferencia de datos, sucesora del SCSI paralelo. Aumenta la velocidad y permite la conexión y desconexión en caliente.

- Asignación de seguridad al usuario de shell
- Control del protector de pantalla
- Soporte de proveedor de red múltiple

2.6.3 Configurando la computadora para la autenticación RFID.

Muchas compañías venden los dispositivos de autenticación y los paquetes de software alternativos para las computadoras personales y corporativas, pero son demasiado costosos. Los exploradores biométricos de huella digital, toman a veces varios intentos para funcionar y el sensor es propenso a los rasguños o a otro tipo de daño. Las Smart cards funcionan bien, pero son complicadas de implementar, y los contactos electrónicos en los lectores y en las tarjetas se desgastan.

Con RFID, que es impermeable al polvo y a la suciedad y siendo totalmente sin contacto, es un candidato perfecto a la autenticación de escritorio. Con estas características, se construyó un sistema RFID capaz de realizar el inicio de sesión y autenticación en windows. Se utilizara software de código abierto para permitir el proceso de la conexión en la computadora con Windows XP.

GINA es el acrónimo para Grafical Identification and Authentication (Identificación y autenticación grafica). Gina es básicamente lo que se ve cuando se desea entrar a windows; la pantalla que pregunta por el nombre de usuario y contraseña es esencialmente el GINA. Pregunta en una forma gráfica (una ventana) cuales son las credenciales de su cuenta (nombre de usuario y contraseña) luego autentica esas credenciales contra cualquier método de autenticación que este disponible para hacerlo. Por ejemplo si la PC es parte de un dominio de windows, o una estructura de directorio activo, comprobará la información con el controlador del dominio.

Existe una configuración en el registro que le dice a windows cual DLL[31] utilizar para las funciones de GINA. Se procederá a reemplazar el GINA por defecto por una nueva DLL llamada PollGINA que permita todas las funciones GINA estándar, pero le agrega la capacidad para que aplicaciones externas especifiquen credenciales de cuentas.

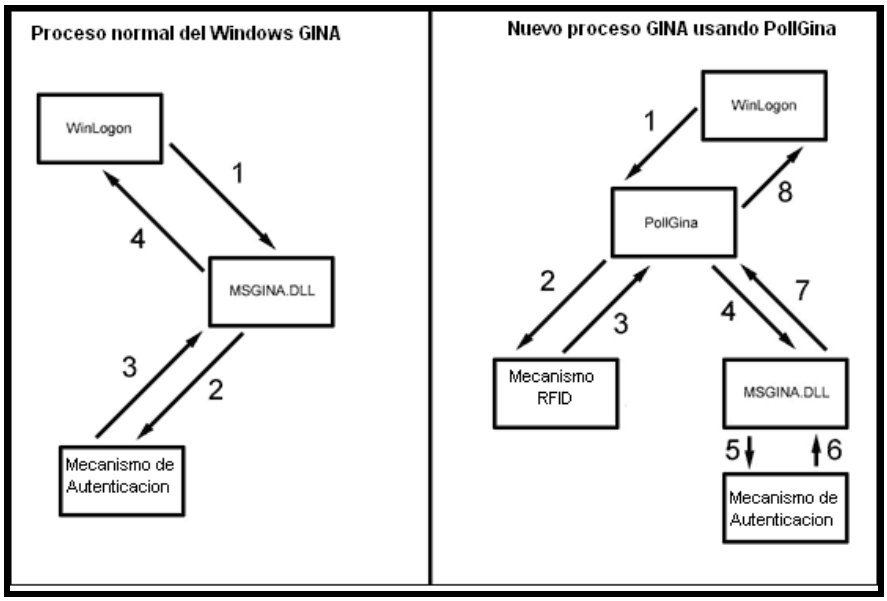


Figura 2.6.3.1 Proceso nomal de autenticación con GINA, y proceso modificado usando PollGina.

Como se muestra en la figura arriba, windows procesa normalmente acontecimientos de la conexión de winlogon al modulo de MSGINA, luego MSGINA consigue la información de la cuenta y la transfiere a varios mecanismos de autenticación, la naturaleza de los cuales depende de la versión del sistema operativo o si su PC es parte de un dominio de RED. Una vez que estos mecanismos realicen la autenticación, los datos pasan de nuevo a MSGINA y de nuevo a Winlogon, lo cual permite abrir una sesión de usuario.

[31] **DLL** (Dinamically Linked Library) Es una librería conectada dinámicamente, que almacena funciones ejecutables o datos que pueden ser usados por un aplicación Windows.

Al decirle a windows que use Pgina como el GINA oficial, winlogon invoca pollgina y espera por una autenticación exitosa. Pollgina por si solo no hace el trabajo de MSGINA pero por el contrario actúa como un intermediario. Debido a que la posición de pollgina es de intermediario, este puede ligar información de la cuenta así como lo es el nombre de usuario y la contraseña a etiquetas de RFID. Si una etiqueta de RFID se desliza sobre el lector, y la información de la cuenta ha sido atribuida a la etiqueta, pollgina pasara la información de esa cuenta a MSGINA para que la procese como si el usuario hubiera tecleado por si mismo.

Una vez MSGINA ha realizado su trabajo, pollgina le regresa el control a winlogon y se le permite iniciar sesión al usuario.

NOTA: El software de pollgina es usado en este proyecto cambiara la forma en que la PC procese la autenticación. Deshabilita el cambio rápido de usuario y si el sistema normalmente requiere que se presionen las teclas CTRL + ALT + DEL antes de iniciar sesión, pollgina deshabilita ese requerimiento.

2.7 MICROSOFT SQL SERVER 2005[32]

2.7.1 ¿Qué es SQL Server 2005?

SQL[33] Server 2005 es una plataforma global de base de datos que ofrece administración de datos con herramientas integradas de inteligencia empresarial. El motor de la base de datos SQL Server 2005 ofrece almacenamiento seguro tanto para datos relacionales como estructurados.

El motor de datos SQL Server 2005 constituye el núcleo de esta solución de administración de datos empresariales. Asimismo, SQL Server 2005 combina análisis, información, integración y notificación. Esto permite que se cree y despliegue soluciones rentables que ayuden a incorporar datos en todos los lugares donde se use, a través de tableros de comando, escritorios digitales, servicios Web y dispositivos móviles.

SQL Server 2005 tiene integración directa con Microsoft Visual Studio, el Microsoft Office System y un conjunto de herramientas de desarrollo, como Development Studio. El siguiente diagrama ilustra los componentes básicos en SQL Server 2005, muestra cómo SQL Server 2005 es una parte importante de Windows Server System y se integra con la plataforma Microsoft Windows, incluidos Microsoft Office System y Visual Studio.

---

[32] Información adaptada de la pagina de información de producto de Microsoft SQL Server 2005, http://www.microsoft.com/spain/sql/productinfo/default.mspx accesado en octubre del 2007.

[33] **SQL** (Structured Query Lenguaje) Lenguaje de consulta estructurado, es usado para recuperar registros o partes de un registro en una base de datos y ejecutar cálculos antes de desplegar los resultados.

Figura 2.6.1.1 Componentes básicos de Microsoft SQL Server 2005

2.7.2 Características de SQL Server 2005

- **Facilidad de uso:**

SQL Server 2005 simplifica el empleo y la administración y optimización de las aplicaciones empresariales para análisis de datos. Esta plataforma para la administración empresarial de datos cuenta con una única consola de administración que permite a los administradores de datos controlar, administrar y ajustar todas las bases de datos y servicios relacionados desde cualquier lugar de la empresa. Cuenta con una amplia infraestructura de administración que se puede programar fácilmente utilizando el manejo de objetos de SQL, permitiendo a los usuarios personalizar y extender su entorno de administración y a los proveedores de software independientes diseñar herramientas y funcionalidades adicionales para desarrollar las capacidades predeterminadas.

- **SQL Management Objects (SMO):**

Es un conjunto nuevo de objetos de programación que expone por completo la funcionalidad de la administración de las bases de datos del SQL Server. SMO puede ser utilizado para automatizar tareas de administración comunes del SQL Server:

recuperar programáticamente valores de configuración, crear nuevas bases de datos, crear tareas para el agente del SQL Server y programar copias de seguridad. El modelo de objetos SMO es un sustituto más seguro, confiable y escalable de Distributed Management Objects (DMO), que se incluyó en versiones anteriores de SQL Server.

- **Reflejo de base de datos:**

Permite transmitir de forma continua el registro de transacciones desde un servidor de origen a un único servidor de destino. En caso de registrarse un error en el sistema primario, se pueden volver a conectar las aplicaciones de inmediato a la base de datos en el servidor secundario. La segunda instancia detecta el error en el servidor primario en cuestión de segundos y acepta las conexiones de bases de datos de inmediato. El reflejo de bases de datos trabaja con hardware para servidores estándar y no necesita almacenamiento ni controladores especiales.

- **Instantáneas de bases de datos:**

SQL Server 2005 introduce la posibilidad para los administradores de bases de datos de crear visualizaciones instantáneas y de solo lectura de las bases de datos. La instantánea de base de datos permite obtener una visualización estable sin la sobrecarga de tiempo y almacenamiento de crear una copia completa de la base de datos. Como resultado, la instantánea se puede utilizar para recuperarse rápidamente de un cambio accidental en la base de datos simplemente volviendo a aplicar las páginas originales de la instantánea a la base de datos primaria.

- **Escalabilidad:**

Los avances en escalabilidad, tales como la partición de tablas, el aislamiento de instantáneas y la admisión de 64 bits le permitirán crear e implementar aplicaciones más exigentes utilizando el SQL Server 2005. La partición de tablas e

índices de gran tamaño mejora significativamente los resultados de las consultas en bases de datos de gran tamaño.

- **Supervisor de réplicas:**

El Supervisor de réplicas es una herramienta que establece una nueva forma de uso que facilita la administración de operaciones de réplica de información complejas gracias a su interfaz de usuario fácil de usar y una gran variedad de indicadores para el manejo de la información.

Característica	Enterprise	Comentarios
Escalabilidad y rendimiento		
Número de CPU	Ilimitado	Es compatible con procesadores multinúcleo
RAM	OS Max	Memoria limitada a un máximo compatible con el sistema operativo
Admite 64 bits	✓	
Tamaño de la base de datos	Ilimitado	
Partición	✓	Compatibilidad para bases de datos a gran escala
Operaciones de índice paralelo	✓	Procesamiento paralelo de operaciones de indexación
Vistas indizadas	✓	Se admite la creación de vista indizada en todas las ediciones. La correspondencia de vista indexada por el procesador de consulta sólo se admite en la Enterprise Edition.
Disponibilidad		
Reflejo de base de datos[1]	✓	Solución avanzada de alta disponibilidad que incluye conmutación rápida por error y redireccionamiento automático del cliente
Organización en clústeres de conmutación por error	✓	
Transmisión de registros de seguridad	✓	Solución de copia de seguridad y recuperación de datos
Cambios del sistema en línea	✓	Incluye Memoria Hot Add, conexión administrativa dedicada y otras operaciones en línea
Indización en línea	✓	
Restauración en línea	✓	
Recuperación rápida	✓	Base de datos disponible cuando comienzan las operaciones para deshacer
Seguridad		
Auditoría, autenticación y autorización avanzadas	✓	
Cifrado de datos y administración de claves	✓	Cifrado de datos incorporado para lograr seguridad avanzada de datos
Analizador de prácticas más adecuadas	✓	Explora su sistema para garantizar que se sigan las prácticas más adecuadas recomendadas
Integración con Microsoft Baseline Security Analyzer	✓	Explora su sistema para verificar vulnerabilidades comunes de seguridad
Integración con Microsoft Update	✓	

Tabla 2.7.2.1 Caracteristicas Principales de Microsoft SQL Server 2005

3. DESCRIPCION GENERAL DEL PROYECTO

El nombre del proyecto es: "DISEÑO Y CONSTRUCCION DE UN SISTEMA DE IDENTIFICACIÓN AUTOMÁTICA, CON APLICACIONES DE HARDWARE Y SOFTWARE, HACIENDO USO DE LA TECNOLOGÍA RFID"**,** Consiste en construir una tarjeta lectora de etiquetas pasivas RFID de 125 Khz con modulación Manchester a 2KBd, dicha configuración es la estándar de las etiquetas del tipo EM4102 fabricadas por EM Microelectronics, la aplicaciones de hardware y software a desarrollar son las siguientes:

- Control de acceso e ignición de un automotor.
- Control de acceso con cerradura electrónica.
- Marcación y registro de horas de trabajo.
- Inicio de sesión y autenticación en Windows XP.

La tarjeta lectora tendrá la capacidad de ser autónoma para las aplicaciones de: control de acceso con cerradura electrónica, y control de acceso e ignición de un automotor, así como también de poseer una interfaz de comunicación USB para poder utilizarla en conjunto con un computador, para las aplicaciones de autenticación e inicio de sesión en Windows XP y Marcación de horas de trabajo.

La tarjeta lectora, estará compuesta de una antena para poder enviar y recibir la señal de las etiquetas, un chip con capacidades de lectura de etiquetas RFID y de microcontroladores PIC; siendo estos los elementos mas importantes.

En los modos autónomos, el microcontrolador será el encargado de tomar las decisiones según programación previamente establecida y según el caso de aplicación; en el modo de comunicación, el microcontrolador servirá de interfaz de comunicación para la computadora.

Vale aclarar que en nuestro país según el Cuadro Nacional de Atribución de Frecuencias de la SIGET, los 125Khz están libres para ser usados por dispositivos de baja potencia (no mayor a 100mW), tales como los identificadores de radio frecuencia.

4. OBJETIVOS

4.1 OBJETIVO GENERAL

- Desarrollar un sistema de identificación automática por RFID con aplicaciones de hardware y software como lo son: apertura e ignición de un automotor, control de acceso, marcación de entrada y salida en una oficina e identificación y autenticación de usuario en Windows XP.

4.2 OBJETIVOS ESPECIFICOS

- Dar a conocer la tecnología RFID.
- Desarrollar un sistema de identificación automática con RFID, que pueda ser usado para realizar múltiples aplicaciones de acceso e identificación, según sean las necesidades del usuario.
- Aplicar el sistema desarrollado para la identificación y control de acceso de personal e ignición de un automotor.
- Aplicar el sistema desarrollado para la identificación y control de horas de trabajo de personal.
- Aplicar el sistema desarrollado para la identificación e inicio de sesión para una computadora con Windows XP.

5. ALCANCES

- La tarjeta lectora podrá configurarse en dos modalidades: autónoma y para comunicación. La modalidad está relacionada con la posición de un jumper.
- Para el sistema desarrollado podrá configurarse una lista de 10 números de identificación autorizados, a través de un programa desarrollado en Visual Basic 6.0 o de forma manual ingresando cada una de las etiquetas.
- La aplicación del sistema para un automotor controlará el acceso e ignición del mismo y podrá ser usado para cualquier automotor con seguros electrónicos y arranque eléctrico.
- En la aplicación del sistema para el inicio de sesión en Microsoft Windows XP; se podrá configurar la lista de números de identificación autorizados, a través de un programa desarrollado.
- La aplicación de registro de horas de trabajo guardará las fechas, horas de entrada y salida de las personas con respecto a su etiqueta de identificación, y presentará un registro el total de las horas trabajadas en horario normal, horas extra y las horas trabajadas en días feriados; este registro se podrá llevar de manera mensual, quincenal o semanal.
- Se incluirá en el documento del trabajo de graduación, una sección que abarcará los criterios de selección de los dispositivos RFID a utilizar.
- El sistema tendrá salidas de potencia con la capacidad de manejar elementos relevadores (relays), para poder utilizar dispositivos que consuman hasta 1-Amperio de corriente continua.

6. LIMITACIONES

- El programa para la configuración de los números de identificación autorizados estará diseñado para trabajar con Microsoft Windows XP y no se garantiza que trabaje con otras versiones de Microsoft Windows.
- La aplicación del sistema para un automotor solo podrá ser usado en automotores con seguros electrónicos y arranque eléctrico.
- La aplicación del sistema para el inicio de sesión en Microsoft Windows será desarrollado para Microsoft Windows XP y no se garantiza que trabaje en otras versiones de Microsoft Windows.
- Cuando el sistema funcione en modo autónomo podrá permitir acceso a un número determinado de etiquetas diferentes. El número de etiquetas que se podrán almacenar dependerá de la cantidad de memoria del microcontrolador.
- No se garantizará la lectura de las etiquetas a través de materiales metálicos o cualquier material con un grosor mayor a 5 milímetros; para determinar los parámetros de operación del sistema se realizaran pruebas con diferentes tipos de materiales y grosores.
- La tarjeta lectora solo podrá leer etiquetas pasivas de 125 Khz del tipo EM4102 o etiquetas que sean compatibles a este estándar.
- La lista de números de identificación autorizados, será almacenada en la memoria EEPROM del microcontrolador, y la cantidad de números almacenados, dependerá del tamaño de la memoria, del algoritmo de almacenamiento y de la aplicación para la cual se esté utilizando el sistema.
- La selección de la base de datos no dependerá exclusivamente de la tabla 4, sino que se evaluara conforme a la marcha del proyecto y se dejará claro el proceso de selección de una u otra base de datos en el trabajo de graduación.

7. VALIDACIÓN DE RESULTADOS

La validación de resultados se llevará a cabo realizando demostraciones de las aplicaciones propuestas, comprobando así las capacidades y versatilidad del sistema.

Las validaciones serán presentadas según la aplicación respectiva, así:

- Se validará la capacidad de identificación automática y el control de acceso del sistema instalándolo en una puerta, tanto de automóvil como de casa u oficina, presentando así el control de acceso. También se instalará el sistema en un automotor, para demostrar la ignición del vehículo sin la utilización de llaves.
- El sistema se presentará en una computadora personal, y se demostrará su capacidad de interactuar con el lector de RFID y leer las etiquetas, para así poder llevar un control de identificación y de horas trabajadas por el personal de una empresa.
- Se demostrará el inicio de sesión automática en Windows XP, configurando el inicio de sesión, modificando el sistema de identificación y autenticación gráfica, logrando así que no se necesite introducir la contraseña, sino más bien deslizar la etiqueta de RFID y automáticamente iniciar sesión.

8. DESARROLLO DEL SISTEMA LECTOR RFID

8.1 CRITERIOS DE SELECCIÓN DE ELEMENTOS RFID

Los elementos RFID a utilizar sistema, Tags e IC lector, fueron seleccionados de entre varias alternativas, tomando en cuenta varios aspectos, entre ellos: si eran tags activos o pasivos, si tenían antena incorporada, costo, cantidad mínima a la venta, y costo del IC lector; la combinación de todos los aspectos anteriores nos llevaron a la mejor alternativa para el proyecto.

Tag	**EM4102**	**Q5**	**MCRF200**
Fabricante	EM Microelectronics	Sokymat	Microchip
Tag Pasivo o Activo	Pasivo	Pasivo	Pasivo
R/W	Solo lectura	Si	Escritura una sola vez
Memoria	64 bits	264bits	128 bits
Capacitor de resonancia interno	Si	Si	No
Capacitor de Buffer de fuente de alimentación interno	Si	Si	No
Limitador de voltaje interno	Si	Si	No
Antena Incorporada	Si	Si	No
Frecuencia de operación	100-150 Khz	125 Khz	100-400 Khz
Modulación	Manchester, Biphase y PSK	FSK (2 diferentes), Manchester, Biphase, PSK (3 diferentes)	ASK, PSK, FSK
Presentaciones	Glass tag, World tag, clear disc, epoxy disc, sticker disc, iso cards, clamshell cards, wristband.	Glass tag, World tag, clear disc, epoxy disc, sticker disc, iso cards, clamshell cards, wristband.	Die, wafer, PDIP y SOIC
Orden mínima de Clamshell Cards	1 unidad	2000 unidades	1000 unidades
Precio unitario	$0.84	$2.68	$1.15
Pagina Web del distribuidor	http://www.trossenrobotics.com/store/p/4753-Thick-Cards-Clamshell-.aspx accesado en junio del 2007	http://rfidusa.com/superstore/product_info.php?products_id=346 accesado en junio del 2007	http://www.dolphin-electronics.com accesado en junio del 2007
IC lector para el tag correspondiente	**EM4095**	**IC no disponible, solo lectores desarrollados.**	**IC no disponible, Nota de App. De circuito lector disponible.**
Precio unitario	$3.48	$212.17	$80
Pagina Web del Distribuidor	http://derikon.com/electronical-components/search/go/?query=em4095	http://www.rfidshop.com/index.asp?function=DISPLAYPRODUCT&productid=1055	http://ww1.microchip.com/downloads/en/DeviceDoc/51115F.pdf
SISTEMA ELEGIDO	Se eligió Tag EM4102 para el proyecto, porque como se puede ver en la presente tabla, es un tag de bajo costo, y fácil acceso, es decir puede ser comprado por unidades, a diferencia de las otras dos alternativas que deben comprarse con órdenes mínimas de 2000 y 1000 unidades. El EM4102 viene en muchas presentaciones lo cual lo hace más versátil que las otras alternativas, así como también que viene con la antena ya incorporada en el tag. El IC para leer el EM4102 es el EM4095, este también supera a los otros lectores puesto que con este pequeño IC de bajo costo se puede desarrollar todo el sistema de lectura con pocos recursos, a diferencia de los otros dos que son lectores ya desarrollados con aplicaciones específicas.		

Tabla 8.1.1 Criterios de selección de elementos RFID

8.2 DISEÑO DEL CIRCUITO DE LA TARJETA LECTORA RFID

Para el diseño del sistema lector, se hace uso del circuito integrado EM4095, de dos microcontroladores, el primero PIC16F877A y el segundo PIC18F4455, tres circuitos integrados optoacopladores 4N35; así como también de elementos varios como lo son resistencias, capacitores, transistores y dos relays.

A continuación, se presenta el diagrama de bloques de la tarjeta lectora RFID.

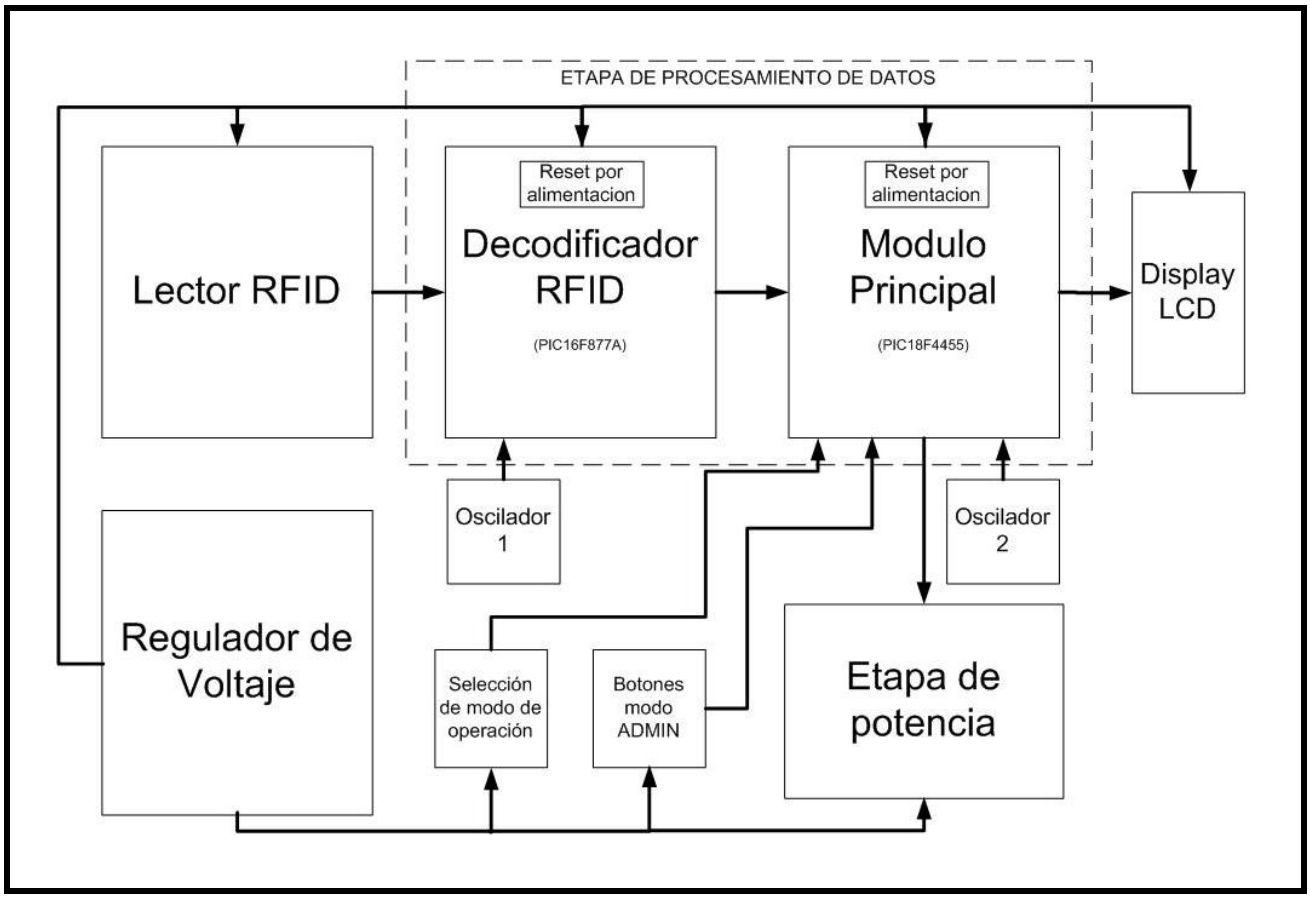


Figura 8.2.1.1 Diagrama de bloques de la tarjeta lectora RFID

En las siguientes secciones se explica detalladamente el diseño y los elementos utilizados en las diferentes etapas del circuito de la Tarjeta Lectora RFID.

8.2.1 Diseño de la etapa de lectura RFID

La etapa lectora RFID de la tarjeta tiene por elemento principal el circuito integrado EM4095, que a continuación se describe con más profundidad.

EM4095[34]: Es el circuito integrado encargado de enviar y recibir las señales de las etiquetas de RFID, éste a su vez envía la señal de reloj de sincronía por el pin 2 (RDY/CLK), y por el pin 13 (DEMOD_OUT) de manera serial la trama de datos del número que leyó de la etiqueta RFID al microcontrolador PIC16F877A, para que éste lo procese y decodifique, ya que el circuito integrado EM4095 lee y envía el número de la etiqueta RFID en codificación manchester.

El circuito de aplicación que se debe basar el diseño, es el que el fabricante recomienda en sus notas de aplicación, y es el siguiente:

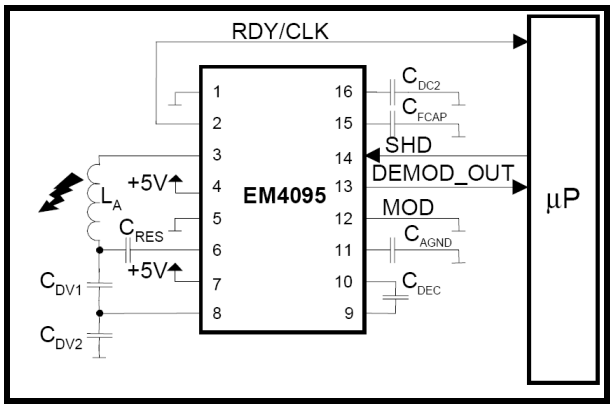


Figura 8.2.1 Circuito de aplicación del EM4095 recomendado por el fabricante.

[34] El circuito de aplicación para este IC, se obtuvo de la hoja de datos del fabricante, http://www.emmicroelectronic.com/webfiles/Product/RFID/DS/EM4095_DS.pdf (Accesado en febrero de 2007) y de la nota de aplicación 404 http://www.emmicroelectronic.com/webfiles/product/rfid/an/an404.pdf (Accesado en febrero de 2007).

Basándonos en el circuito que el fabricante recomienda y tomando en cuenta los valores recomendados para todos los elementos, el circuito de la etapa de lectura RFID es el siguiente:

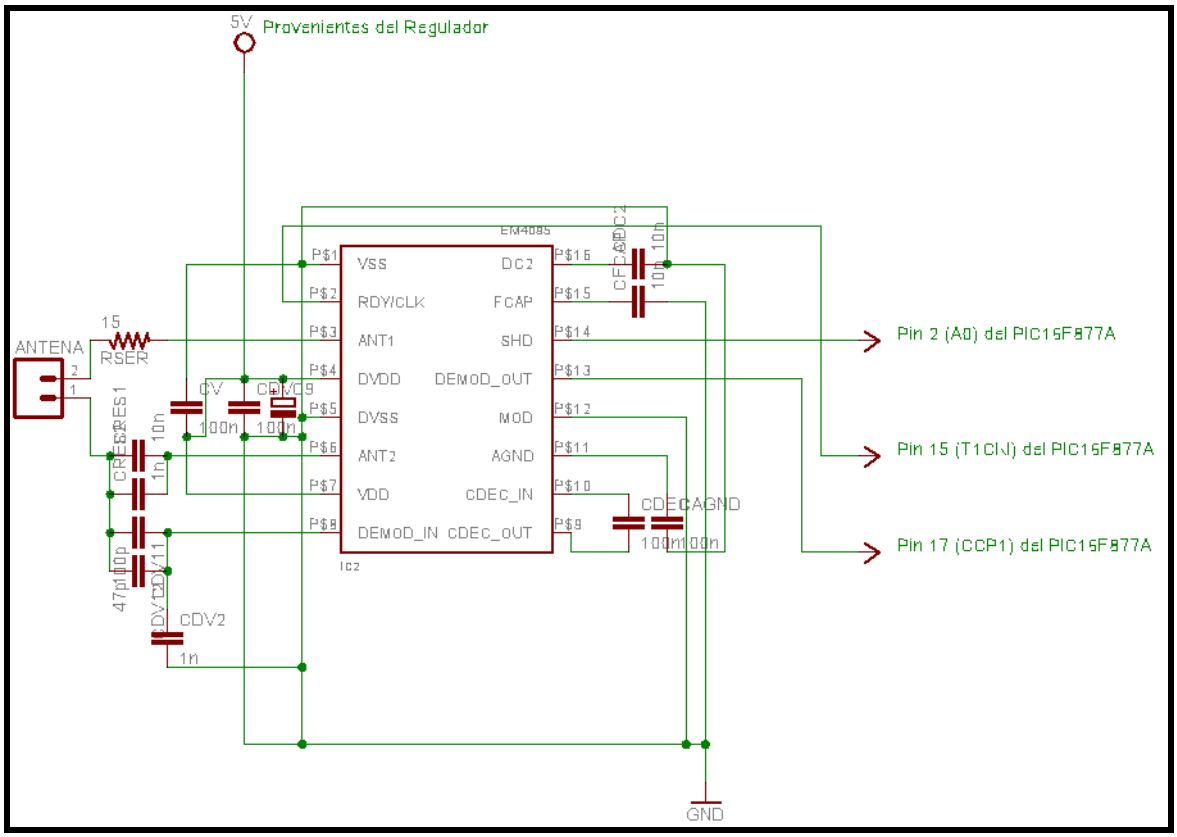


Figura 8.2.2 Circuito de aplicación del EM4095 implementado.

8.2.2 Diseño de la etapa de procesamiento de datos

Para la etapa de procesamiento de datos se tomó la decisión de utilizar dos microcontroladores, debido a que el compilador PCWH, todavía está siendo depurado, es decir, presenta problemas para ciertos dispositivos, y entre los dispositivos que presentan problemas para ciertas rutinas preestablecidas e interrupciones son los microcontroladores PIC de la serie 18 y 24, ya que son relativamente una serie nueva, por tal razón, al generar el código para el PIC18F4455, y usar las interrupciones para poder controlar el IC EM4095, éste presenta problemas;

específicamente el problema que presenta es que el PIC18F4455 no decodificaba los datos provenientes del chip de RFID, por lo tanto se optó por utilizar un PIC de la serie 16 que es mas antiguo y para el cual el código generado fue satisfactorio, a continuación se explica de manera mas detallada la función de cada microcontrolador.

PIC16F877A: Este circuito integrado microcontrolador, está conectado directamente al IC EM4095 y también al microcontrolador PIC18F4455, la función de éste es decodificar los datos que recibe del IC EM4095 y enviarlos por comunicación en paralelo microcontrolador PIC18F4455.

PIC18F4455: Este circuito integrado microcontrolador, está conectado directamente al microcontrolador PIC16F877A, y es el encargado de manejar los datos (números de la etiqueta) que el PIC16F877A le envía. El manejo de los datos que hace este microcontrolador consiste en: guardarlos en memoria, presentarlos en pantalla (LCD), comparar los números con otros que se encuentren en memoria, enviarlos por USB, activar o desactivar salidas.

Osciladores de Cristal: Los osciladores de cristal se caracterizan por su estabilidad de frecuencia y pureza de fase, dada por el resonador.

Todo microprocesador o microcontrolador requiere de un circuito que le indique a que velocidad debe trabajar. Este circuito es conocido como un oscilador de frecuencia; en el caso de los microcontroladores estos osciladores se utilizan para introducir la frecuencia de reloj. Existen microcontroladores que poseen su oscilador internamente, pero este no es el caso para ninguno de los microcontroladores PIC que se han utilizado para el sistema lector RFID.

Los microcontroladores utilizados para el sistema necesitan de osciladores externos para generar su frecuencia de reloj, por esta razón, se escogieron osciladores de cristal para la generación del reloj. Según las características de los microcontroladores a usar en el sistema, así son los valores de cristal que se deben usar para la generación del reloj; para el caso del microcontrolador PIC16F877A, se debe usar el valor de cristal mas alto que soporta, cuyo valor es de 20MHz, es decir, que éste microcontrolador funcionará a una frecuencia de reloj interna de 20Mhz, que es proporcionada por el cristal; para el microcontrolador PIC18F4455, se debe usar un cristal de 4MHz, sin embargo, este microcontrolador posee una frecuencia de reloj interna de 48Mhz, y eso se logra mediante subsistemas y divisores de frecuencia internos que posee el microcontrolador para poder trabajar con esta frecuencia.

Osc Type	Crystal Freq.	Cap. Range C1	Cap. Range C2
LP	32 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	200 kHz	47-68 pF	47-68 pF
	1 MHz	15 pF	15 pF
	4 MHz	15 pF	15 pF
HS	4 MHz	15 pF	15 pF
	8 MHz	15-33 pF	15-33 pF
	20 MHz	15-33 pF	15-33 pF

Tabla 8.2.2.1 Tipos de Osciladores y velocidades para el PIC 16F877A.

Para la programación de los microcontroladores se hace uso del programa Winpic800 versión 3.62 en dicho programa se configuran ciertas características de los microcontroladores como lo es el reloj interno de los microcontroladores.

Se seleccionaron las frecuencias más altas para un óptimo funcionamiento de los microcontroladores, y por lo tanto, para el sistema, de esta manera se esta garantizando un mejor manejo de datos.

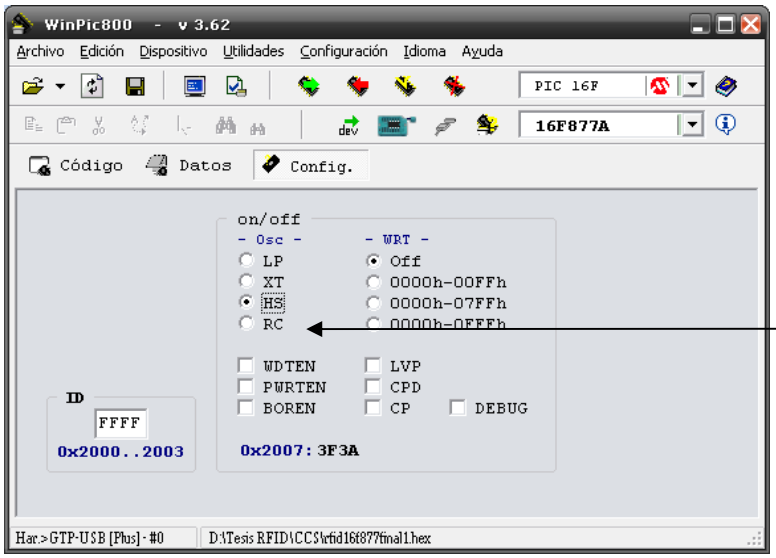


Figura 8.2.2.2 Configuración del Oscilador para el PIC 16F877A, en el programa WinPic800.

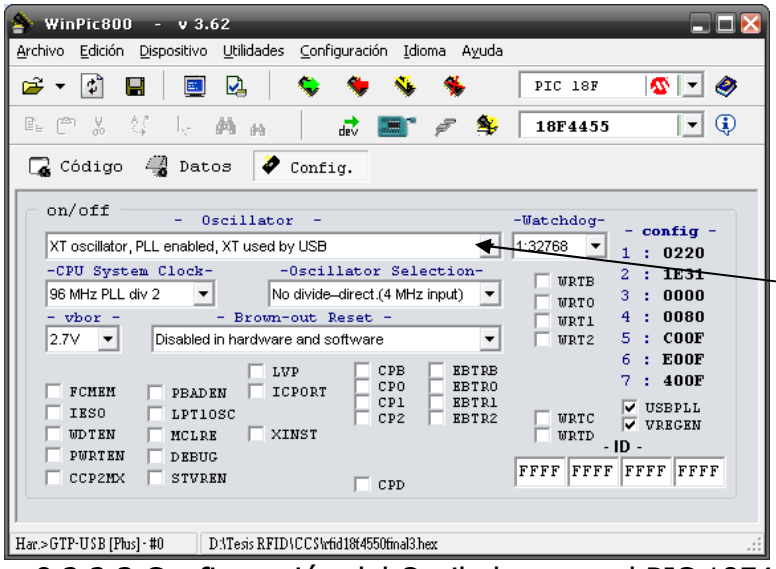


Figura 8.2.2.3 Configuración del Oscilador para el PIC 18F4455, en el programa WinPic800.

El circuito de la etapa de procesamiento de datos es el siguiente:

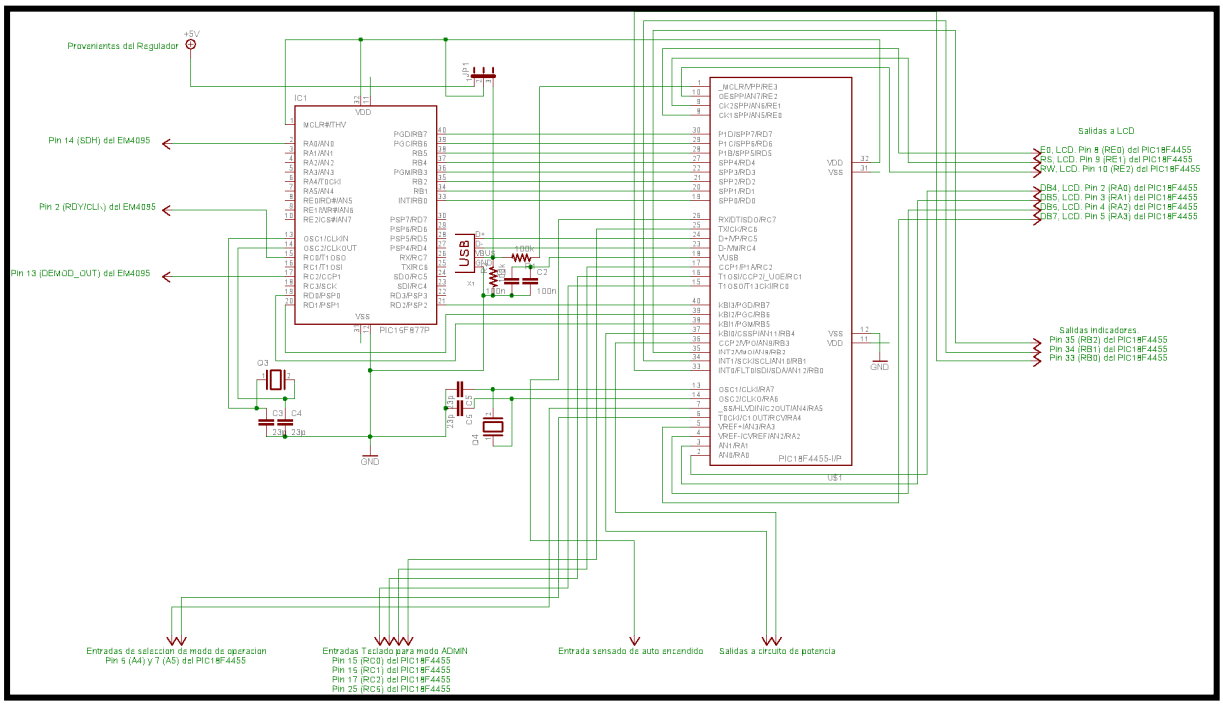


Figura 8.2.2.4 Circuito de la etapa de procesamiento de datos y configuración de microcontroladores.

A continuación se presenta una tabla en la que se especifica la conexión de todos los pines de los microcontroladores utilizados en esta etapa de procesamiento de datos.

PIC 16F877A			
Pin	**Puerto / Modulo**	**Entrada / Salida**	**Descripción**
1	MCLR	Entrada	Pin de Reset
2	RA0	Salida	Salida al Pin 14 (SHD) del IC EM4095
13	OSC1	Entrada	Pin de Oscilador
14	OSC2	Salida	Pin de Oscilador
15	T1CKI	Entrada	Entrada del Pin 13 (DEMOD_OUT) del IC EM4095
17	CCP1	Entrada	Entrada del Pin 2 (RDY/CLK) del IC EM4095
19 - 21	RD0 – RD2	Salida	Puntero de dato. Salida a pin 38 - 40 (RB5 – RB7) del PIC18F4455
11, 32	VDD	Entrada	Entrada de Voltaje (5V).
12, 31	VSS	Entrada	Entrada de Voltaje (GND).
33 - 40	RB0 – RB7	Salidas	Datos (numero de etiqueta). Salidas a pines 19-30 (RD0-RD7) del PIC18F4455.

PIC18F4455			
Pin	**Puerto / Modulo**	**Entrada / Salida**	**Descripción**
1	RE0	Entrada	Entrada de censado de conexión USB
2 - 5	RA0 – RA3	Salida	Salida a pin DB4 – DB7 de LCD.
6 – 7	RA4 – RA5	Entrada	Entrada de selección de modo de operación
8	RE0	Salida	Salida a pin E de LCD.
9	RE1	Salida	Salida a pin RS de LCD.
10	RE2	Salida	Salida a pin RW de LCD.
11, 32	VDD	Entrada	Entrada de Voltaje (5V).
12, 31	VSS	Entrada	Entrada de Voltaje (GND).
13	OSC1	Entrada	Pin de Oscilador
14	OSC2	Salida	Pin de Oscilador
15 - 17	RC0 – RC2	Entrada	Entrada de teclado para modo de ADMIN
25	RC6	Entrada	Entrada de teclado para modo de ADMIN
18	VUSB	Entrada	Entrada al regulador de voltaje interno del USB
19 - 22	RD0 – RD3	Entrada	Datos (numero de etiqueta). Entradas de pines 33 - 36 (RB0-RB7) del PIC18F877A
27 – 30	RD4 – RD7	Entrada	Datos (numero de etiqueta). Entradas de pines 37 - 40 (RB0-RB7) del PIC18F877A
23	D-	Entrada /Salida	Línea de USB de datos negativa
24	D+	Entrada /Salida	Línea de USB de datos negativa
26	RC7	Entrada	Entrada de censado del encendido del automóvil
33 - 35	RB0 – RB2	Salida	Salida a indicadores (LED).
36 - 37	RB3 - RB4	Salida	Salida a circuito de potencia.
38 - 40	RB5 – RB7	Entrada	Puntero de dato. Entrada de pin 19 - 21 (RD0 – RD2) del PIC16F877A

Tabla 8.2.2.5 Configuración de pines de los microcontroladores de la etapa de procesamiento de datos.

8.2.3 Diseño de la etapa de Regulación de voltaje.

Los elementos que componen el sistema lector de RFID, en su mayoría trabajan con 5V, por lo tanto, se procedió a realizar un circuito regulador de voltaje, cuya finalidad es mantener una tensión constante de 5V a su salida, siempre y cuando su entrada sea mayor que este valor de voltaje.

El circuito integrado que se utilizó para realizar esta función es el 7805, cuyas característica principales son las siguientes:

- Voltaje de entrada de 5 a 35 voltios máximo.
- Voltaje de salida de 5V (siempre que en la entrada se mantenga el rango de 5V - 35V).
- Corriente de salida de hasta 1A.

- Protección por sobrecarga térmica.
- Protección por corto circuito.
- Circuito integrado de tres pines. 1. Entrada, 2. Tierra o GND, 3. Salida.

A continuación se presentan las figuras del encapsulado del circuito integrado, así como también el diagrama de bloques y el diagrama de aplicación usado para el sistema lector RFID.

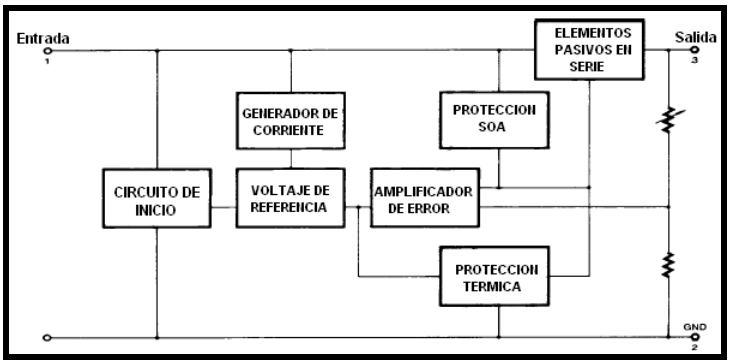


Figura 8.2.3.1 Diagrama de bloques interno del regulador de voltaje 7805.

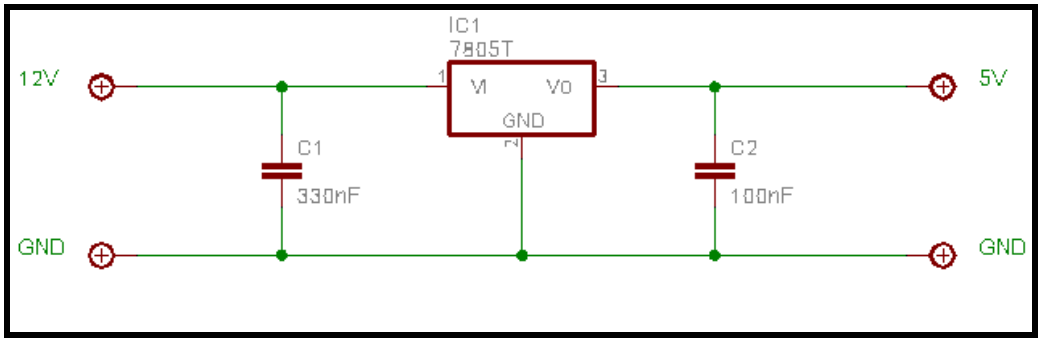


Figura 8.2.3.2 Diagrama esquemático del circuito regulador de voltaje utilizado para alimentar el sistema RFID.

8.2.4 Diseño de la etapa de Reset.

Los microcontroladores PIC, poseen diversas fuentes de RESET como lo son:

- Una conexión a la alimentación del circuito.
- Una acción sobre el pin MLCR mientras que el circuito está en modo normal.

- Una acción sobre el pin MLCR mientras que el circuito está en modo SLEEP.
- Un desbordamiento del watchdog, mientras el circuito está en modo normal.
- Un desbordamiento del watchdog, mientras el circuito está en modo SLEEP.

En el sistema lector de RFID, se está utilizando el primer método de RESET, ya que por las mismas características de los microcontroladores, éstos permiten que esta acción sea controlada por la falta de alimentación al mismo circuito, sin embargo, para el microcontrolador PIC16F877A, se realizó un arreglo para que funcione de esta manera, ya que en su hoja de especificaciones nos dice que para la acción de RESET se usa el primero tanto como el segundo método, que es una acción sobre el pin MCLR, así como se muestra en la figura abajo.

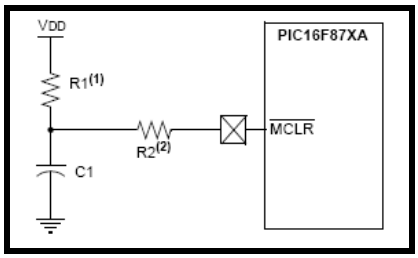


Figura 8.2.4.1 Diagrama del circuito de Reset
para el microcontrolador PIC 16F877A.

Para el microcontrolador PIC16F877A, se conectó el pin MCLR a la entrada de voltaje VCC, por lo tanto estamos permitiendo una sola fuente de RESET, la cual se presenta al haber una falta de alimentación al circuito, es decir, se modificó el circuito que aparece en la figura arriba por el que se muestra en la figura abajo. Por otra parte el microcontrolador PIC18F4455, es mas versátil en este aspecto, ya que posee una característica especial que consiste en que el pin MCLR y el RESET que va asociado a este pin sea controlado por la alimentación del circuito y poder utilizar el pin MCLR como un pin de entrada de datos; esta configuración se da al momento de programar el microcontrolador.

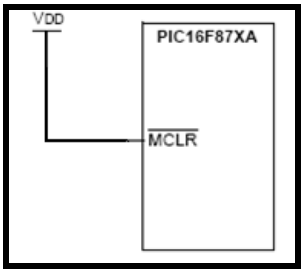


Figura 8.2.4.2 Diagrama del circuito de Reset
para el microcontrolador PIC 16F877A

8.2.5 Diseño de la etapa de entrada y salida de potencia.

El sistema lector de RFID posee salidas de potencia para poder manejar más corriente a su salida, o bien para utilizar algún otro subsistema o actuador; el sistema lector de RFID está aislado eléctricamente con la finalidad de protección de sus propios elementos y del propio sistema en sí. Para el aislamiento eléctrico se hace uso de circuitos integrados 4N35 que son dispositivos optoacopladores.

Optoacopladores 4N35: Este circuito integrado se utiliza para acoplar las señales del sistema lector con el exterior, para evitar sobretensiones y posibles daños al sistema lector.

Las salidas de potencia del sistema lector de RFID están compuestas por transistores 2N2222 y relevadores de 12V, como se puede observar en la figura abajo; lo cual le permite al sistema un control de mayor potencia a su salida.

Con respecto a la entrada de potencia, ésta es utilizada para detectar cuando el automóvil está encendido, al igual que las salidas de potencia, esta entrada se encuentra aislada eléctricamente con los dispositivos 4N35.

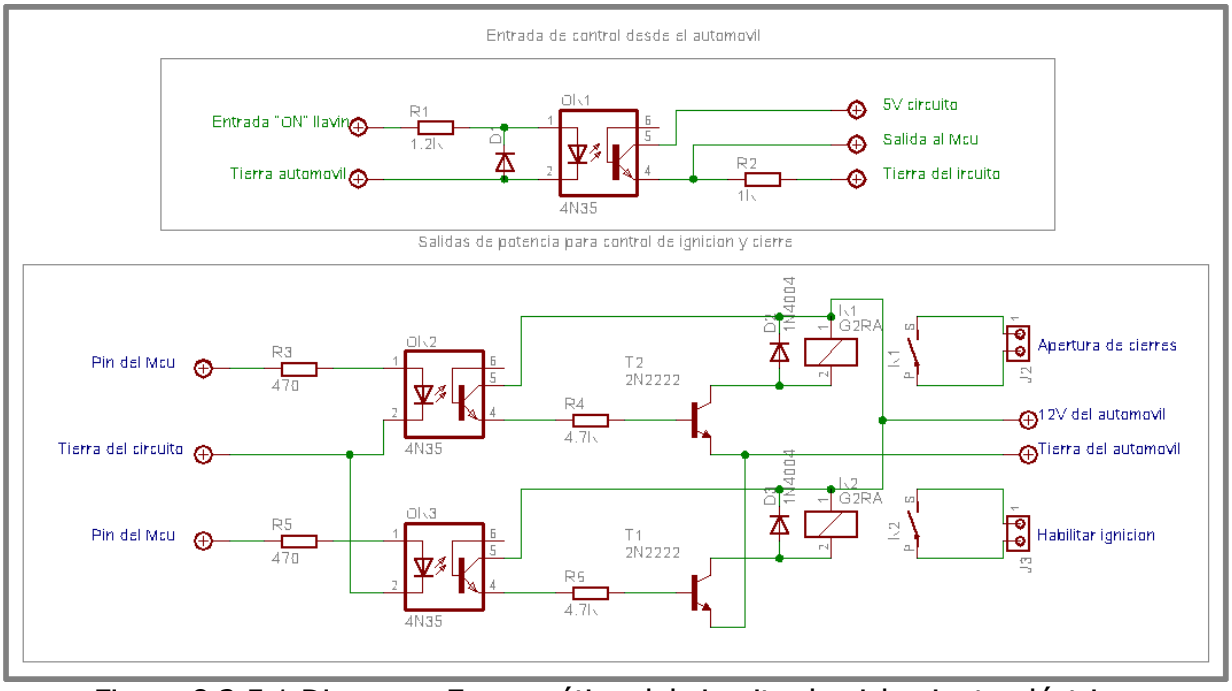


Figura 8.2.5.1 Diagrama Esquemático del circuito de aislamiento eléctrico, entrada de potencia y salida de potencia.

8.2.6 Diseño de la etapa de Selección de modo operación.

El sistema lector de RFID posee la característica de ser un dispositivo que puede cambiar de modo de funcionamiento con solo modificar la combinación en el selector minidip y así ser utilizado para la tarea que se requiera, como se explica más profundamente en la sección 10.2 del presente documento.

El selector minidip se encuentra conectado al microcontrolador PIC18F4455 específicamente a los pines 6 (RA4) y 7 (RA5), estos pines del microcontrolador se han configurado como entrada de datos. La configuración del minidip presenta un cero lógico a sus salidas cuando está en posición de "OFF" es decir, que en las entradas del microcontrolador habrá cero voltios; por otra parte, cuando el selector minidip esté en posición "ON", a su salida presenta un uno lógico, lo cual conlleva a tener en las entradas del microcontrolador cinco voltios. Gracias a esto y a la programación previamente establecida, el microcontrolador detectará estos cambios

de estado en el selector y adecuará su programación al modo de funcionamiento que se seleccione en el minidip.

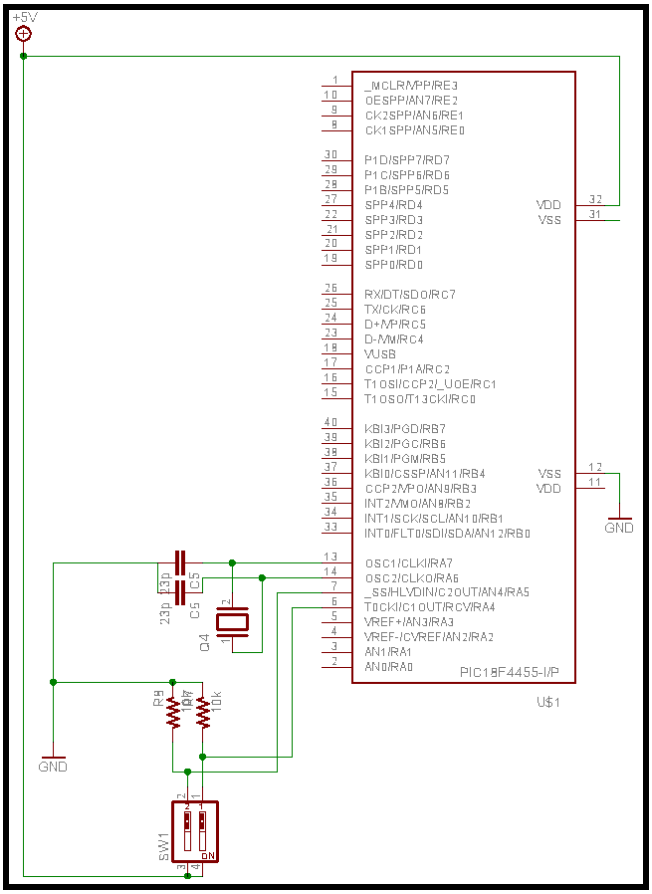


Figura 8.2.6.1 Diagrama del circuito selector de modo de operación.

8.2.7 Diseño de las pistas del circuito impreso

Para el diseño del circuito impreso se utilizó software electrónico con las capacidades de realizar circuitos impresos a partir del esquema, el software utilizado es Cadsoft Eagle en su versión 4.11, que tiene una interfaz para diseño de circuitos esquemáticos y partiendo de ese diseño, automáticamente genera el diseño de impreso.

Con respecto al tamaño de pistas y puntos de conexión, el software tiene en sus librerías los dispositivos y los tamaños de puntos de conexión de una gran variedad de dispositivos electrónicos, pero hubo necesidad de diseñar el EM4095 pues este no se encontraba en las librerías, también se diseñó un base para poder montar y desmontar fácilmente el EM4095.

El Software posee la característica de poder variar el tamaño y forma de las pistas, para lo cual se procedió a especificar un ancho de pista de 0.024 de pulgada y un estilo que no generará pistas de forma cuadrada o rectangular, pero todas las pistas fueron ruteadas de forma manual pues el ruteado automático daba como resultado un circuito con demasiadas bias (cambio de una pista de un lado al otro en de la tarjeta).

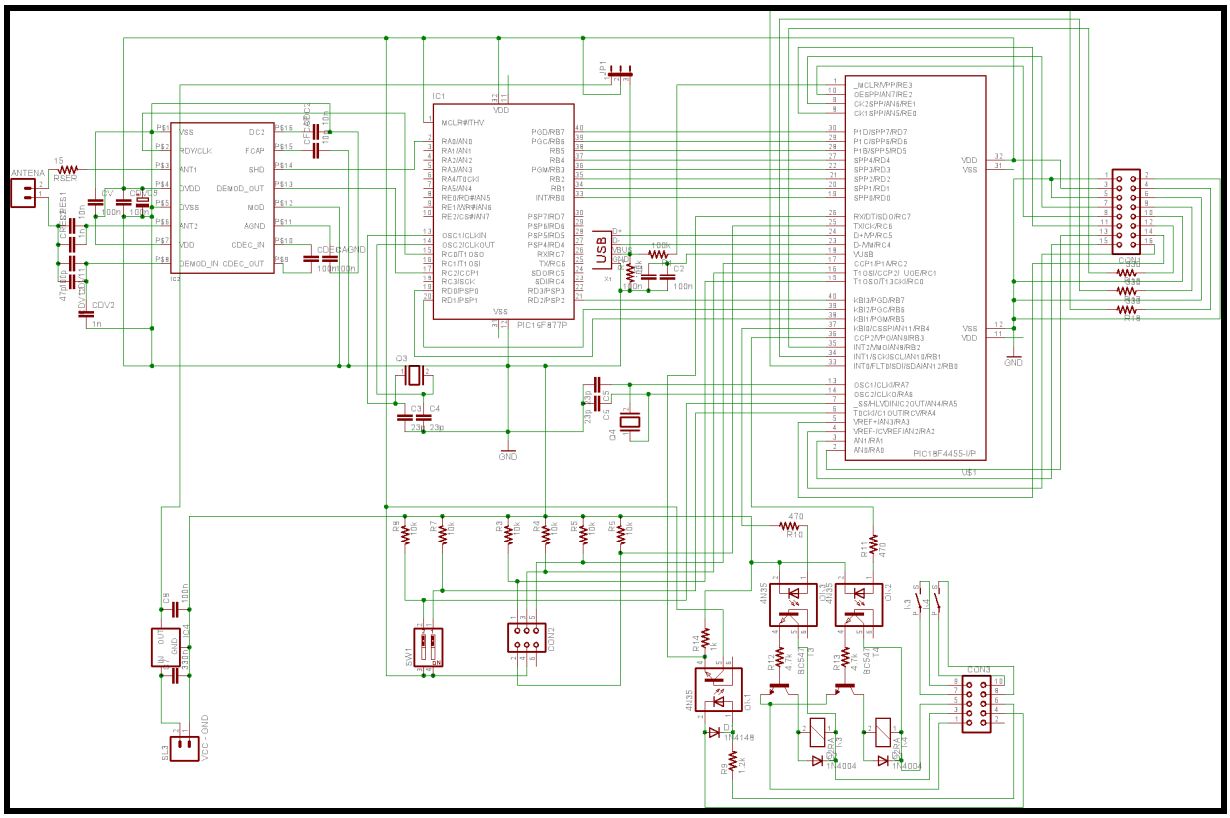


Figura 8.2.7.1 Esquemático del circuito de la Tarjeta lectora RFID

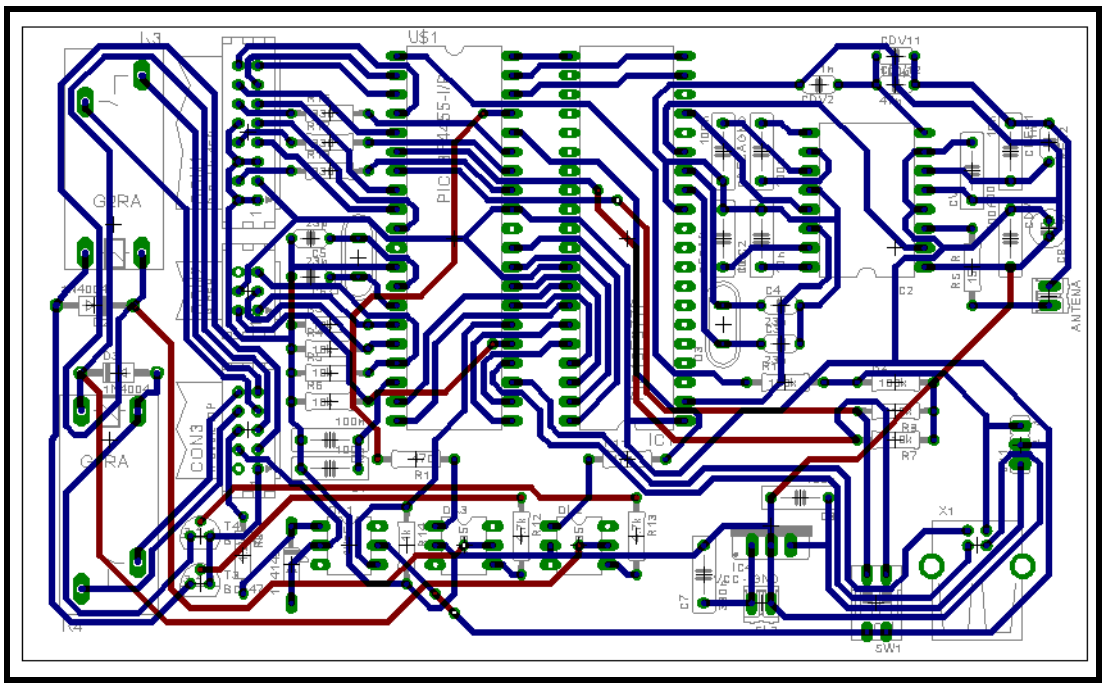


Figura 8.2.7.2 Pistas del circuito impreso de la Tarjeta lectora RFID

8.2.8 Diseño de la etapa de protección del circuito.

Para la selección de un buen actuador en caso de sobrecarga eléctrica, se procedió a realizar mediciones de corriente en el sistema lector, y gracias a las pruebas realizadas, se llego a la conclusión de utilizar un fusible, el cual es un dispositivo de seguridad utilizado para proteger un circuito eléctrico de un exceso de corriente. Su componente esencial es, habitualmente, un hilo o una banda de metal que se derrite a una determinada temperatura. Si la corriente del circuito excede un valor predeterminado, el metal fusible se derrite y se rompe o abre el circuito.

Para el sistema se opto por utilizar n fusible cilíndrico, el cual está formado por una banda de metal fusible encerrada en un cilindro de cerámica o de fibra. Unos bornes de metal ajustados a los extremos del fusible hacen contacto con la banda de metal. Este tipo de fusible se coloca a la entrada de potencia del sistema lector de RFID de modo que la corriente fluya a través de la banda metálica para que el circuito se complete. Si se da un exceso de corriente en el circuito, la conexión de

metal se calienta hasta su punto de fusión y se rompe. Esto abre el circuito, detiene el paso de la corriente y, de ese modo, protege al circuito.

De acuerdo con las mediciones realizadas, se opto por utilizar un fusible de 12V por 0.5A.

Figura 8.2.8.1 Tipo de fusible a utilizar
para el sistema lector de RFID.

8.2.9 Selección de mueble para montar el sistema.

El sistema lector de RFID, se procedió a montarlo en una caja plástica color negro, dicha caja plástica, fue modificada en gran parte, para la adecuada presentación e instalación del circuito en ella, así como también para la instalación en el automotor y para las demás aplicaciones.

Las dimensiones de la caja modificada son las siguientes: 12.5 cm. de ancho, 17.5 cm. de largo y 4 cm. de alto, así como también una serie de agujeros para conectores, pantalla de cristal liquido (LCD) y un pequeño teclado.

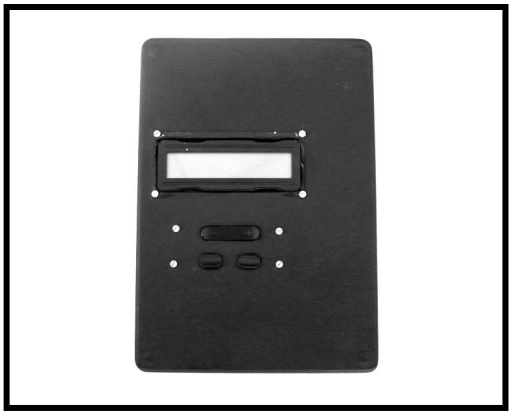


Figura 8.2.9.1 Caja plástica que guarda el Sistema RFID en su interior.

8.2.10 Diseño y cálculos para la construcción de la Antena[35]

La inductancia de la antena debe ser elegida dentro del rango típico que recomienda el fabricante (100uH-750uH), para el caso elegimos 150uH para que la resistencia de la antena sea la menor posible. El conductor de cobre esmaltado elegido para la antena es calibre 32 que tiene un diámetro de 0.2mm. La antena tiene un núcleo de láminas de ferrita para transformador. Para calcular el número de vueltas necesarias para lograr la inductancia deseada debemos usar siguiente ecuación:

$$L = \frac{\mu_0 \mu_r N^2 A}{l}$$

Donde:

L es la inductancia de la bobina

μ0 es la permitividad magnética del vacío

μr es la permitividad magnética relativa del núcleo

N es el número de vueltas

A es el área transversal de la bobina

l es la longitud de una vuelta

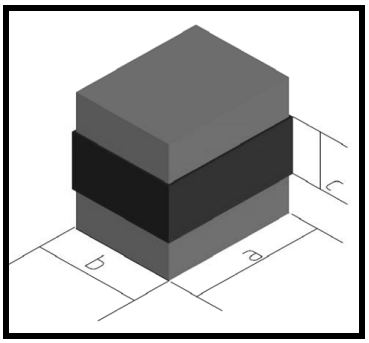


Figura 8.2.10.1 Diagrama de la Antena con núcleo de ferrita (a=16cm, b=38cm, c=1cm).

[35] Todas las formulas usadas en esta seccion fueron tomadas de las siguientes Fuentes:

- Nota de aplicación 404 de EM microelectronic: http://www.emmicroelectronic.com/webfiles/product/rfid/an/an404.pdf (Accesado en febrero de 2007).
- JOHNK, CARL T. A. "Teoria Electromagnetica, Campos y Ondas / Carl T. A. Johnk" New York: John Wiley & Sons, 1988 ([2nd Ed.])

Para encontrar el número de vueltas tenemos que:

$$N = \sqrt[2]{\frac{Ll}{\mu_0 \mu_r A}}$$

L = 150 µH

µ0 = 4*pi*10^-7

µr laminas de ferrita para transf. = 5000

A= 0.01m x 0.0002m (altura de la bobina multiplicada por el ancho del conductor)

l = (0.016m + 0.038m)*2

$$N = \sqrt[2]{\frac{(150\mu H)(0.016m + 0.038m) * 2}{(4\pi * 10^{-7})(5000)(1 * 10^{-2}m)(2 * 10^{-4}m)}}$$

$$\underline{N = 36\ vueltas}$$

Sabiendo el número de vueltas necesarias para lograr la inductancia deseada se fabrica la antena, posteriormente se procede a comprobar el valor de la inductancia montando el siguiente circuito:

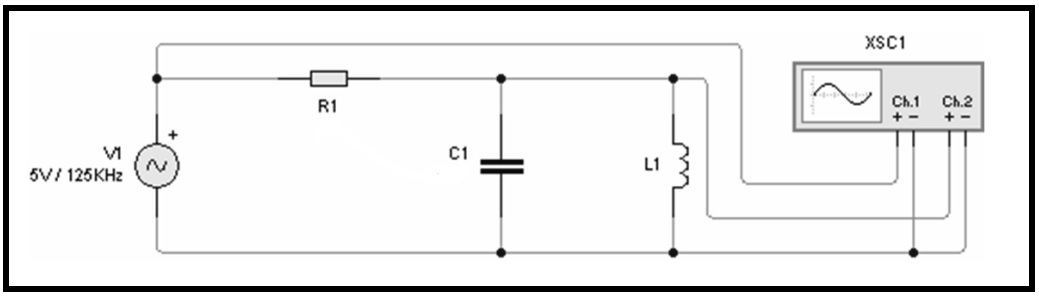


Figura 8.2.10.2 Diagrama esquematico del circuito para determinación experimental de parametros.

Con este circuito se puede determinar la inductancia L a partir de la siguiente ecuación:

$$L = \frac{1}{C * (2\pi * f_0)^2}$$

Donde, f_0 es la frecuencia de resonancia, a la cual el voltaje en la antena es máximo y está en fase con respecto al voltaje de la fuente. Usando un arreglo capacitivo de 20nF, observamos que la frecuencia de resonancia aproximada fue de 90Khz, por lo tanto la inductancia de la antena es:

$$L = \frac{1}{20 * 10^{-9} * (2\pi * 90 * 10^3)^2}$$

$$\underline{L = 156\mu H}$$

El capacitor de resonancia para lograr los 125KHz a los que debe trabajar la antena se calcula con la siguiente ecuación:

$$Cres = \frac{1}{L * (2\pi * f_0)^2}$$

$$Cres = \frac{1}{(156 * 10^{-6}) * (2\pi * (125 * 10^3))^2}$$

$$\underline{Cres = 10.39nF}$$

Debido a que en el mercado no hay capacitores con este valor se debe armar un arreglo con dos capacitores para lograr 11nF.

Posteriormente se procede a armar el mismo circuito con el que se determinó la inductancia de la antena, pero esta vez con el arreglo capacitivo de 11nF para que la Fo sea de 125KHz, y así comprobar de manera experimental la frecuencia de resonancia de la antena, la cual debe ser de 125KHz para tener máxima eficiencia.

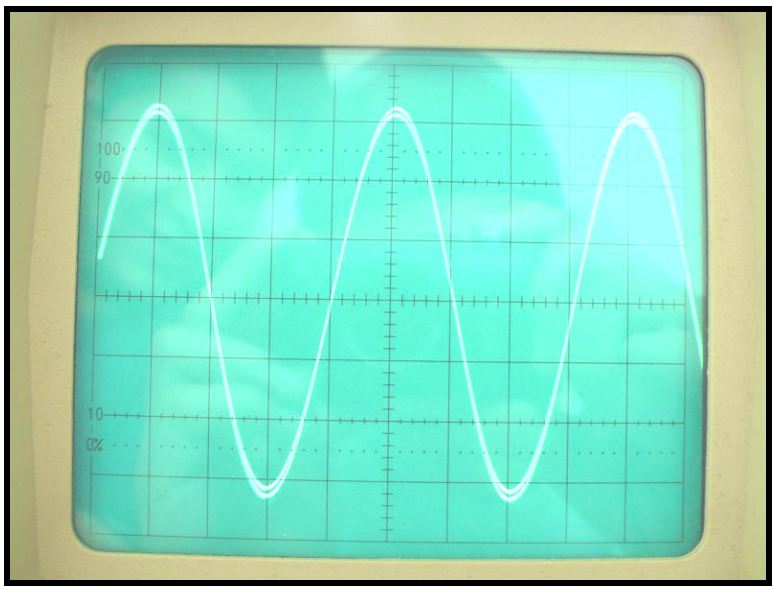


Figura 8.2.10.3 Oscilograma de la señal de entrada y la señal en la antena (Osciloscopio en 2uS/div y 5V/div, por lo tanto el periodo es 8uS y la frecuencia es 125Khz).

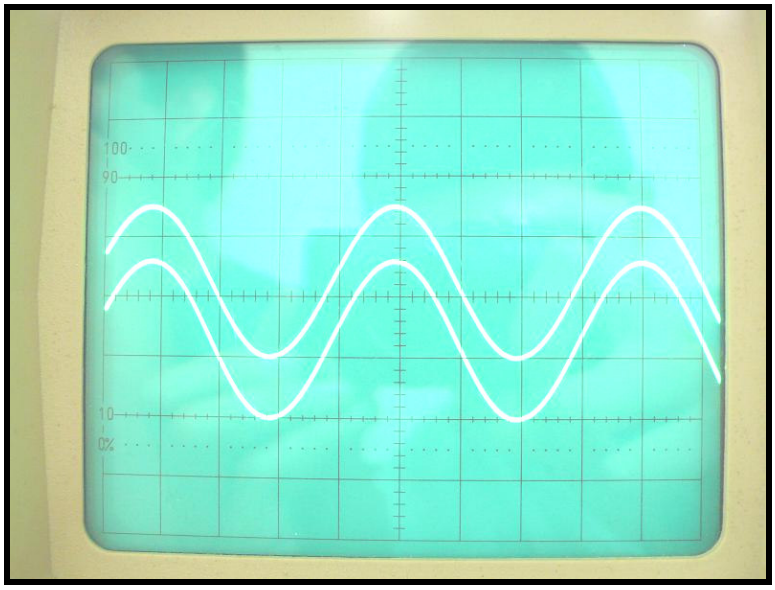


Figura 8.2.10.4 Oscilograma de la señal de entrada y la señal en la antena (Osciloscopio en 2uS/div y 1V/div, por lo tanto el periodo es 8uS y la frecuencia es 125Khz).

Se puede observar en los Oscilogramas que la frecuencia de resonancia del arreglo es 125Khz, por lo tanto el arreglo capacitivo para la antena es el adecuado.

La resistencia óhmica de la antena debe medirse con un multímetro, para el caso, es de 2.5 Ohms. Por lo tanto, el factor de calidad de la antena trabajando a 125KHz será:

$$Q = \frac{2\pi * f_0 * L}{R}$$

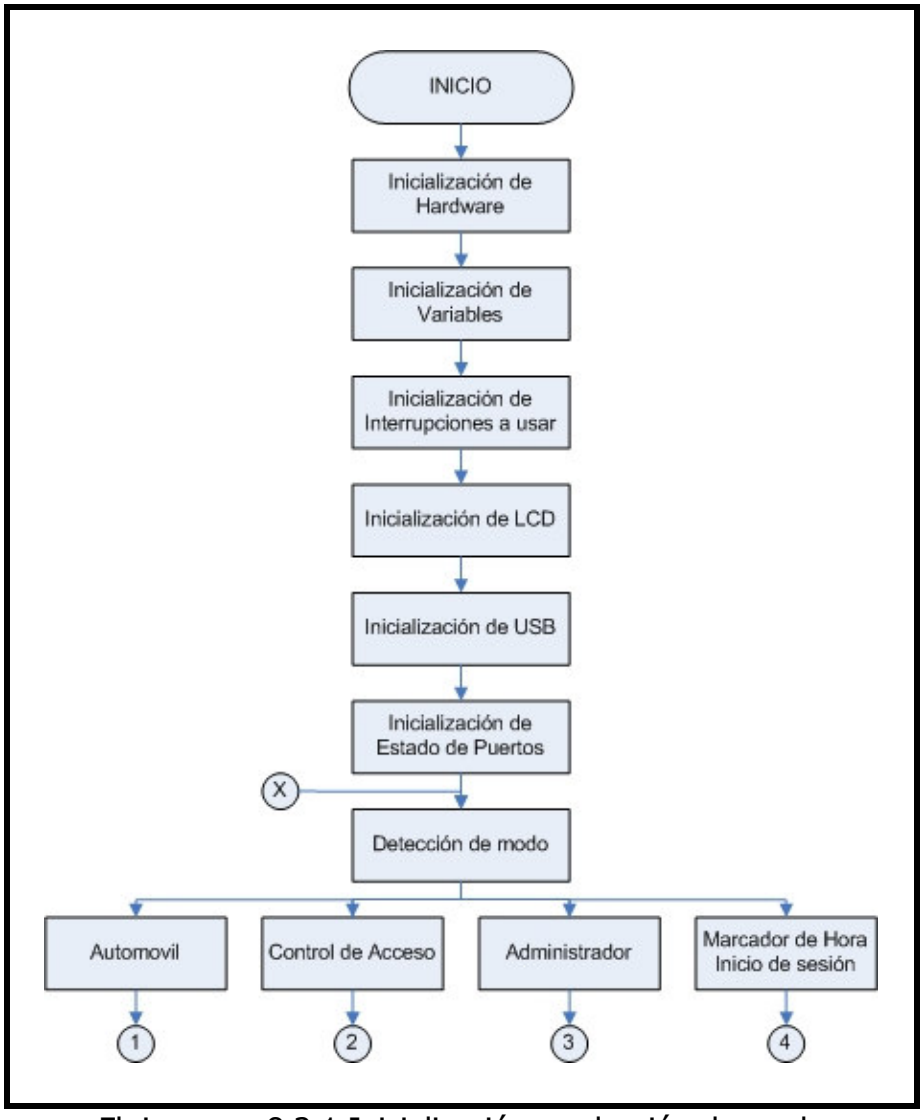
$$Q = \frac{2\pi * (125 * 10^3) * (156 * 10^{-6})}{2.5}$$

$$\underline{Q = 49}$$

8.3 FLUJOGRAMAS PARA EL DESARROLLO DEL PROGRAMA DE LA TARJETA LECTORA RFID[36]

Para desarrollar el programa de la tarjeta lectora RFID se usaron los siguientes flujogramas antes de proceder a realizar el código.

El siguiente flujograma muestra como se debe llevar a cabo la inicialización y la selección de modo de operación de la tarjeta.



Flujograma 8.3.1 Inicialización y selección de modo

[36] El código de programa asociado a los flujogramas se encuentra en la sección de anexos, página 121.

A continuación se detallara cada etapa del fluograma mostrado en la figura 8.3.1.

Inicializacion de Hardware: Dentro de esta etapa están constituidas las siguientes rutinas:

- Inicio del Microcontrolador (Registros internos, oscilador, palabras de configuración, subsistemas a utilizar).
- Inicio de puertos en modo rápido.
- Configuracion de protocolo USB.
- Inserción de archivos de cabecera a utilizar.
- Configuración de interrupciones a utilizar con su código asociado.

Inicialización de variables: En esta etapa se definen todas las variables a utilizar en el programa.

Inicialización de interrupciones a utilizar: En esta etapa del programa se habilitan las interrupciones que se configuraron anteriormente y que están disponibles en cualquier momento de ejecución del programa.

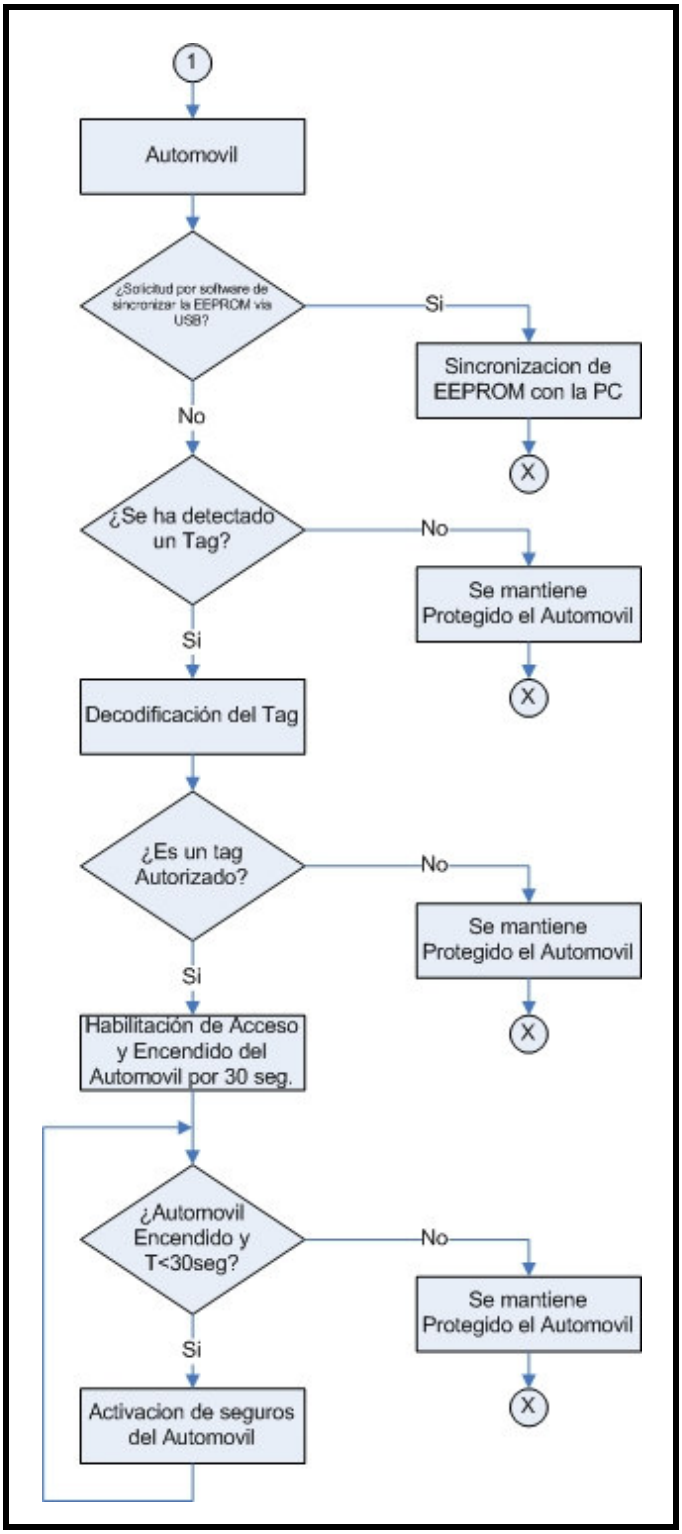
Inicialización de LCD: Se inicia el modulo para el control del display LCD contenido en uno de los archivos de cabecera previamente incluido. En dicho modulo se especifican los pines del microcontrolador se desean utilizar para el control de dicho display.

Inicialización de USB: Se inicia el modulo USB del microcontrolador contenido en uno de los archivos de cabecera previamente incluido; para lo cual el microcontrolador queda en espera de conexión al bus USB

Inicializaciòn de estado de puertos: Se configuran las salidas de los pines designados a puertos de I/O, con un valor de estado alto o bajo, según sea el caso necesario.

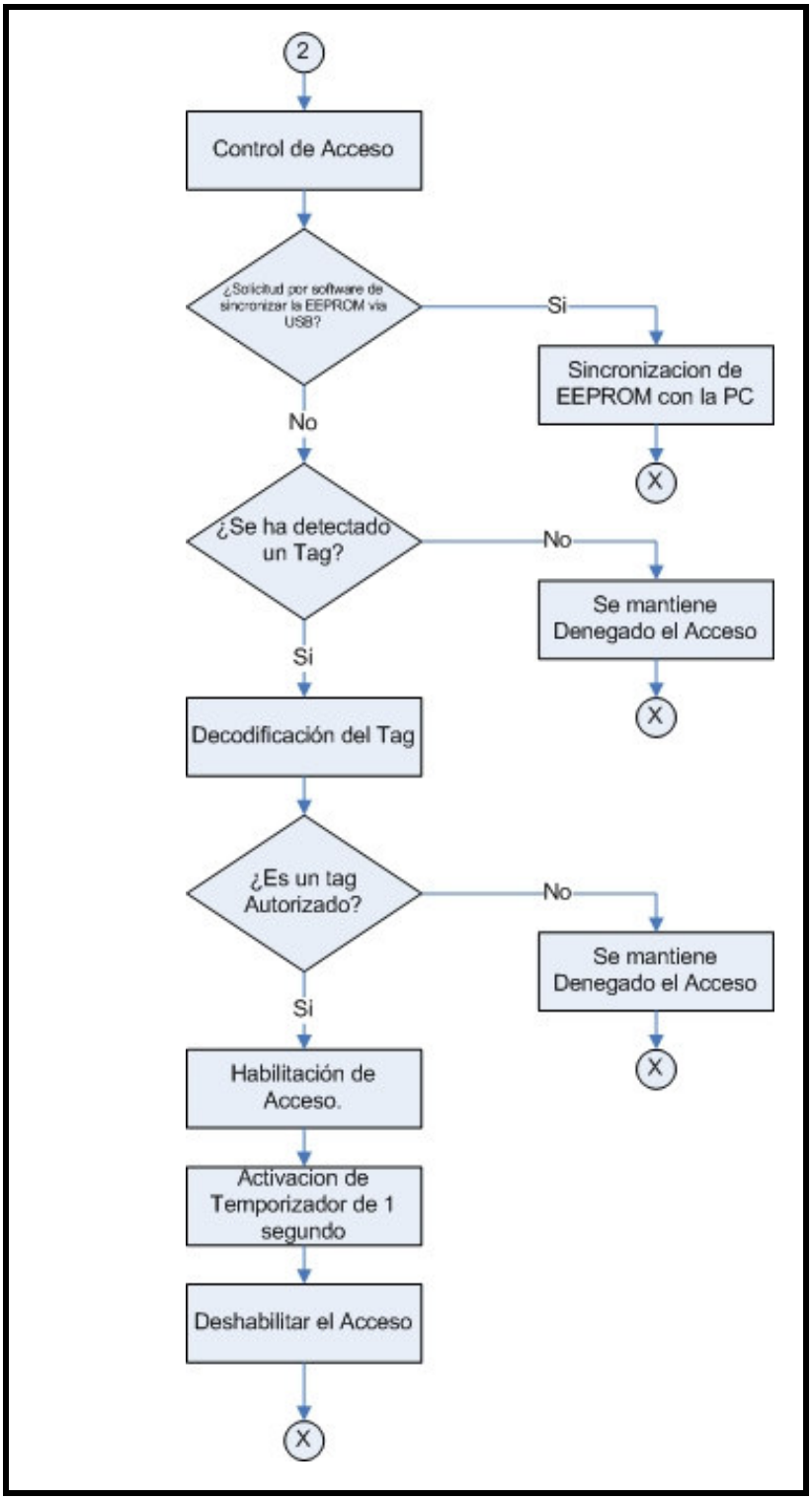
Detección de Modo: El microcontrolador detecta el modo de operación dependiendo de la configuración del selector minidip (del estado de los pines de I/O designados al selector).

El siguiente flujograma muestra como debe ser el proceso del modo: control de acceso e ignición de un automotor.



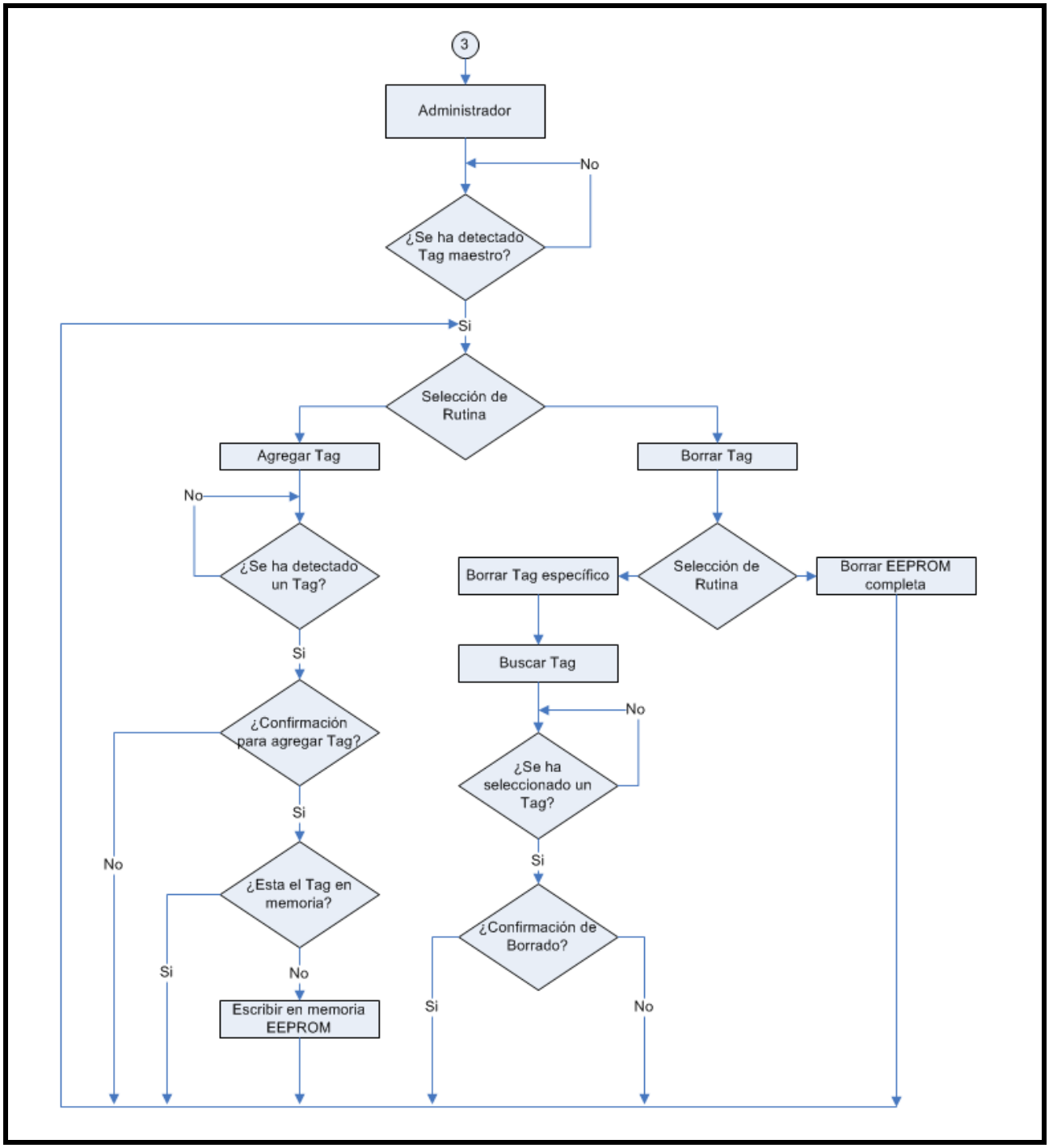
Flujograma 8.3.2 Control de acceso e ignición de un automotor

El siguiente flujograma muestra como debe ser el proceso del modo: control de acceso o cerradura electrónica.



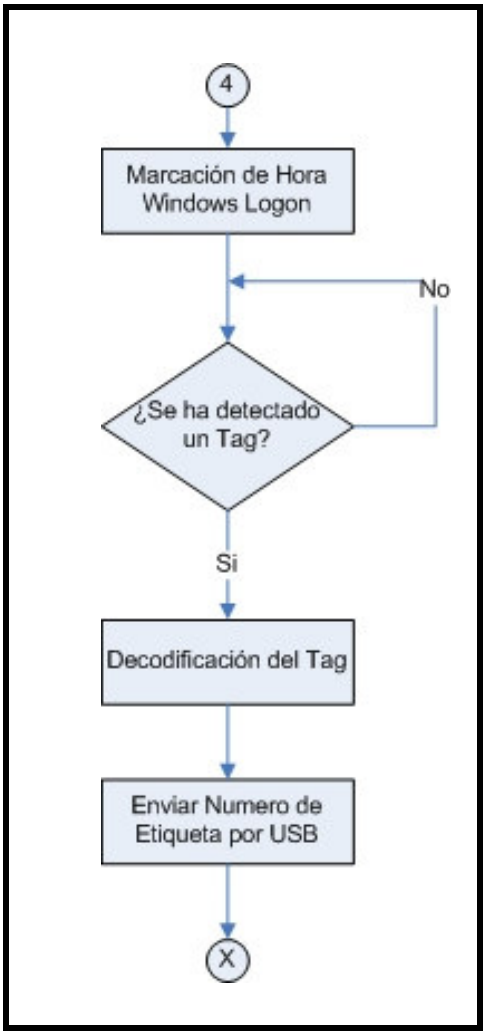
Flujograma 8.3.3 Control de acceso o cerradura electrónica.

El siguiente flujograma muestra como debe ser el proceso del modo Administrador.



Flujograma 8.3.4 Modo Administrador

El siguiente flujograma muestra como debe ser el proceso del modo de Marcación de horas de trabajo e inicio de sesión en Windows XP.



Flujograma 8.3.5 Modo de Marcacion de horas de trabajo e inicio de sesión en Windows XP.

9. FUNCIONAMIENTO DEL SISTEMA LECTOR RFID

9.1 FUNCIONAMIENTO DEL SISTEMA LECTOR RFID

El sistema lector de RFID basa su funcionamiento a partir de una concordancia entre el número de la etiqueta del usuario y el número previamente guardado en la memoria del sistema. A partir de esta concordancia, el sistema tomara decisiones, según sea el modo de trabajo y la programación establecida.

Debido a que las etiquetas de usuario son modelo EM4102, se tienen los siguientes 64 bits en el arreglo de memoria[37]:

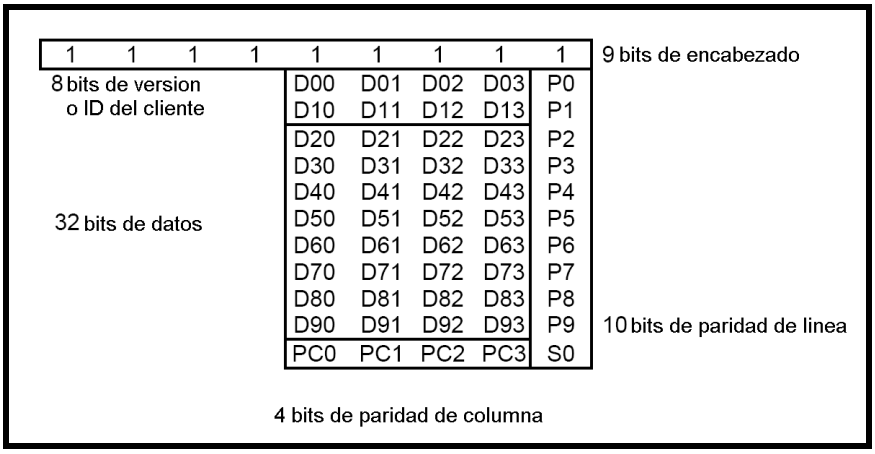


Diagrama 9.1.1 Arreglo de memoria de las etiquetas EM4102.

En la trama de datos que transmite una etiqueta EM4102 van primero 9 bits de encabezado que son todos 1, luego van 10 grupos de 4 bits de datos (8 bits son de la versión o ID del fabricante y 32 bits son del numero único de la etiqueta), cada grupo de 4bits de datos va seguido de un bit de paridad de la fila; el ultimo grupo consta de 4 bits de paridad de columna y 1 bit de paro que esta grabado con 0. La trama de datos es transmitida a 2KHz, sobre una portadora de 125Khz por amplitud modulada, con codificación Manchester.

[37] Información obtenida de la hoja de especificaciones técnicas de los EM4102 http://www.emmicroelectronic.com/webfiles/Product/RFID/DS/EM4102_DS.pdf

Los datos que se observan al demodular el stream Manchester son los siguientes:

111111111 D00D01D02D03P0 D10D11D12D13P1 D20D21D22D23P2 D30D31D32D33P3 [...] PC0PC2PC3 SO

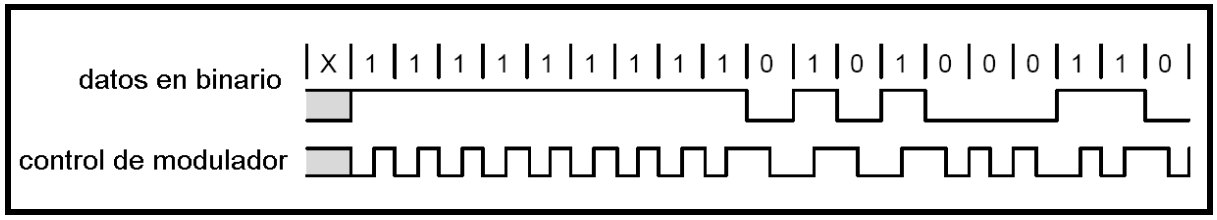


Diagrama 9.1.2 Datos en binario de la memoria del EM4102 y Datos con el control de modulación Manchester.

9.2 MODOS DE TRABAJO

El sistema lector RFID posee cuatro modos de operación:

- Modo de Administración.
- Modo para el acceso de personal (cerradura electrónica).
- Modo para el acceso e ignición de un automotor.
- Modo para el control de empleados (marcador de horas de trabajo).

La tarjeta lectora de RFID posee en un costado un selector minidip, que es por medio del cual se seleccionan los modos de operación de la tarjeta así:

SELECTOR	MODO
00	Modo para el acceso e ignición de un automotor.
01	Modo para el acceso de personal (cerradura electrónica).
10	Modo para el control de empleados (marcador de horas de trabajo).
11	Modo de Administración.

Tabla 9.2.1 Modos de la Tarjeta lectora RFID

Primeramente se comenzara a detallar el funcionamiento del modo de administración, ya que es el modo principal del sistema lector de RFID y por medio del cual logran un funcionamiento optimo los demás modos del sistema.

9.2.1 Modo de administración

Este modo, es por medio del cual el usuario puede grabar y borrar etiquetas de la memoria del sistema; las etiquetas grabadas, serán las que el sistema aceptará como validas.

El sistema posee la capacidad de guardar 30 números en su memoria, esto es equivalente a 30 usuarios para los cuales se les permite el acceso. Además, en la última localidad de memoria (localidad 31) se tiene la etiqueta de seguridad, la cual nos permite entrar al modo de administración para poder agregar y borrar etiquetas.

Nota: La etiqueta de seguridad solo puede ser agregada desde la interfaz visual en la PC.

A este modo de trabajo se accede poniendo en el selector minidip la combinación "11", o bien poniendo ambos selectores en su estado de "ON". Al haber realizado lo anterior, en la pantalla de cristal liquido (LCD) aparecerá la leyenda que cita lo siguiente "Deslizar TAG de Administración", pidiendo la etiqueta de seguridad para poder entrar a este modo de administración, así como se muestra en la figura 9.2.1.1.

Figura 9.2.1.1

Una vez se ha deslizado la etiqueta de administración, la leyenda en la pantalla cambia y cita: "Agregar: up / Borrar: down", como se muestra en la figura 9.2.1.2.

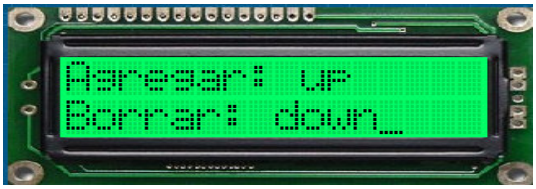


Figura 9.2.1.2

A partir de acá, el usuario decidirá lo que desea hacer, ya sea agregar etiquetas al sistema o borrar tanto etiquetas como toda la memoria.

Agregar una etiqueta: Para agregar una etiqueta a la memoria del sistema se hace lo siguiente:

Según la leyenda citada en la pantalla de la figura 9.2.1.2, se muestra que, para agregar una etiqueta hay que presionar la tecla up; al presionar la tecla, cambiara la leyenda en la pantalla a la siguiente "Deslizar TAG", como se muestra en la figura 9.2.1.3.

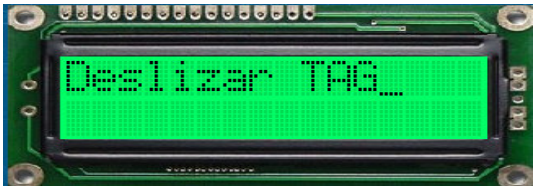


Figura 9.2.1.3

A partir de acá, se esperará a que una etiqueta sea deslizada cerca del sistema para reconocerla, decodificarla y presentar el numero en la pantalla. Habiendo deslizado la etiqueta cerca del sistema lector, la pantalla cambiara nuevamente y citará lo siguiente "# de etiqueta / Agregar: OK/ESC", como se muestra en la figura 9.2.1.4.

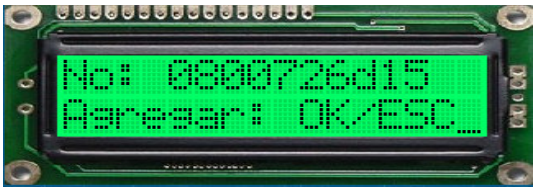


Figura 9.2.1.4

Habiendo realizado lo anterior, se pide una confirmación si se desea guardar la etiqueta en memoria o no. Si se desea agregar la etiqueta, se presiona la tecla "OK" y la etiqueta queda guardada en la memoria del sistema; si no se desea guardar la etiqueta, se presiona la tecla "ESC" y volvemos a la pantalla de la figura 9.2.1.2.

Cabe mencionar una consideración. Cuando se desea guardar una etiqueta que ya has sido almacenada en la memoria del sistema, éste muestra en la pantalla una confirmación que la etiqueta ya esta previamente guardada en la memoria, así como se muestra en la figura 9.2.1.5.

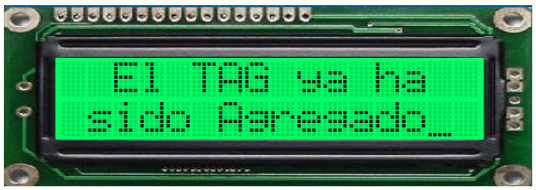


Figura 9.2.1.5

Borrar una etiqueta o memoria completa: Para borrar una etiqueta o la memoria completa del sistema se hace lo siguiente:

Según la leyenda citada en la pantalla de figura 9.2.1.2, muestra que para borrar una etiqueta o la memoria hay que presionar la tecla "down"; al presionar la tecla, cambiara la leyenda en la pantalla a la siguiente "Dirección: OK / Completa: ESC", como se muestra en la figura 9.2.1.6.

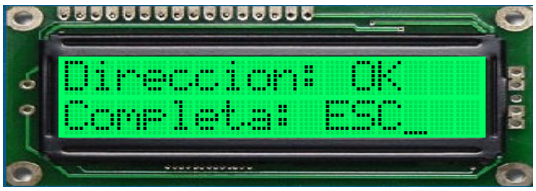


Figura 9.2.1.6

A partir de acá el sistema esperará a que el usuario seleccione la opción que desea, ya sea borrar una etiqueta específica o la memoria completa del sistema

Para borrar una etiqueta especifica se presiona le tecla "OK", y al hacer esto, la pantalla cambia su leyenda, mostrando el numero de etiqueta y pidiendo confirmación si se desea o no borrar, así como se muestra en la figura 9.2.1.7.

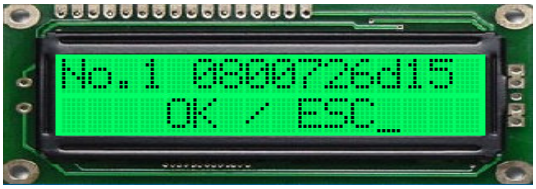


Figura 9.2.1.7

Cabe mencionar, que podemos desplazarnos por la memoria del sistema con las teclas "up y down" y así seleccionar la etiqueta que se desea borrar.

Una vez seleccionada la etiqueta a borrar, se presiona la tecla "OK" y se muestra en pantalla la confirmación de que se ha borrado, así como se muestra en la figura 9.2.1.8.

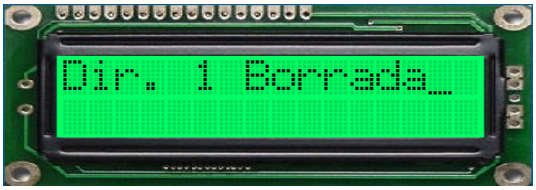


Figura 9.2.1.8

Cuando se ha borrado una etiqueta del sistema, este vuelve a mostrar la pantalla de figura 9.2.1.2.

Para borrar la memoria completa del sistema se presiona le tecla "ESC", y al hacer esto, la pantalla cambia su leyenda y cita "EEPROM BORRADA…", lo cual nos indica que la memoria ha sido borrada en su totalidad, así como se muestra en la figura 9.2.1.9.

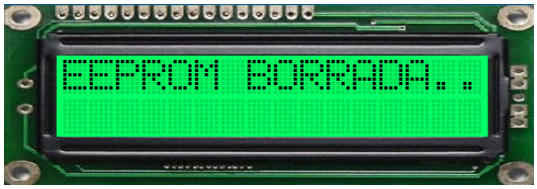


Figura 9.2.1.9

Cuando se ha borrado la memoria en su totalidad, el sistema vuelve a mostrar la pantalla de la figura 9.2.1.2.

Nota: La localidad de memoria en la cual se encuentra la etiqueta de seguridad no se puede borrar mediante los métodos anteriormente mencionados. La única manera de borrarlo o cambiarlo es mediante la interfaz grafica realizada para la PC.

9.2.2 Programa en Visual Basic 6 para el Control de Etiquetas en la memoria EEPROM

El programa para sincronizar la memoria del sistema, al igual que el modo de administración posee seguridad para no permitir que cualquier usuario pueda modificar la información de la memoria. Al iniciar la interfaz visual del sistema de sincronización, observamos la pantalla de la figura 10.2.2.1.

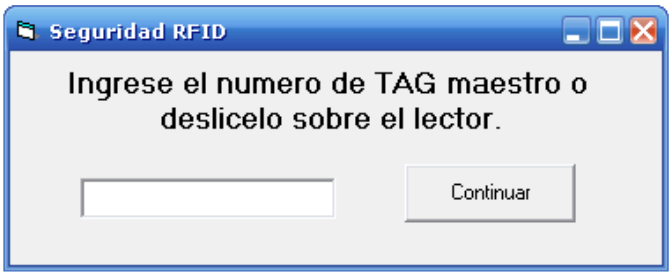


Figura 9.2.2.1

La pantalla de la figura 10.2.2.1, nos permite introducir nuestra contraseña (que es el No. de la etiqueta de seguridad) manualmente por el teclado, o bien, deslizando la etiqueta de seguridad. Si el numero introducido o la etiqueta deslizada es la correcta, el programa habilita la pantalla de sincronización con el sistema lector, que es el modulo principal del programa.

Por otro lado, si la contraseña es incorrecta, el programa nos muestra un mensaje de error que notifica que la contraseña ingresada o bien que la etiqueta deslizada no es la correcta, como se muestra en la figura 9.2.2.2.

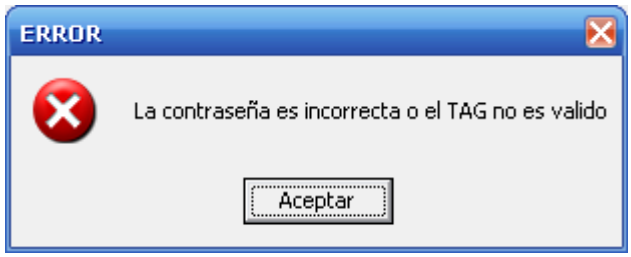


Figura 9.2.2.2

Por otro lado, si no se tiene ninguna etiqueta almacenada en memoria, que corresponda a la etiqueta de seguridad, el programa detecta esto y al ingresar una contraseña o deslizar una etiqueta, este nos informa que no hay una etiqueta de seguridad almacenada como se muestra en la figura 10.2.2.3, por lo tanto, la primera operación a realizar es designar una etiqueta de seguridad y moverla hasta la última localidad de memoria, que corresponde a la localidad numero 31.

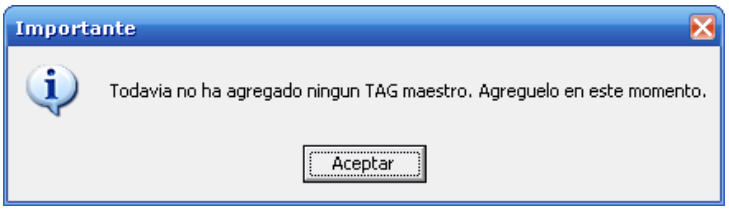


Figura 9.2.2.3

El modulo principal del programa que se utiliza para sincronizar los números tiene la apariencia de la figura 9.2.2.4. Posee una tabla en la cual se sincronizan las etiquetas con la memoria del microcontrolador y otra en la que se editan los nombres de los usuarios a los cuales corresponde cada etiqueta.

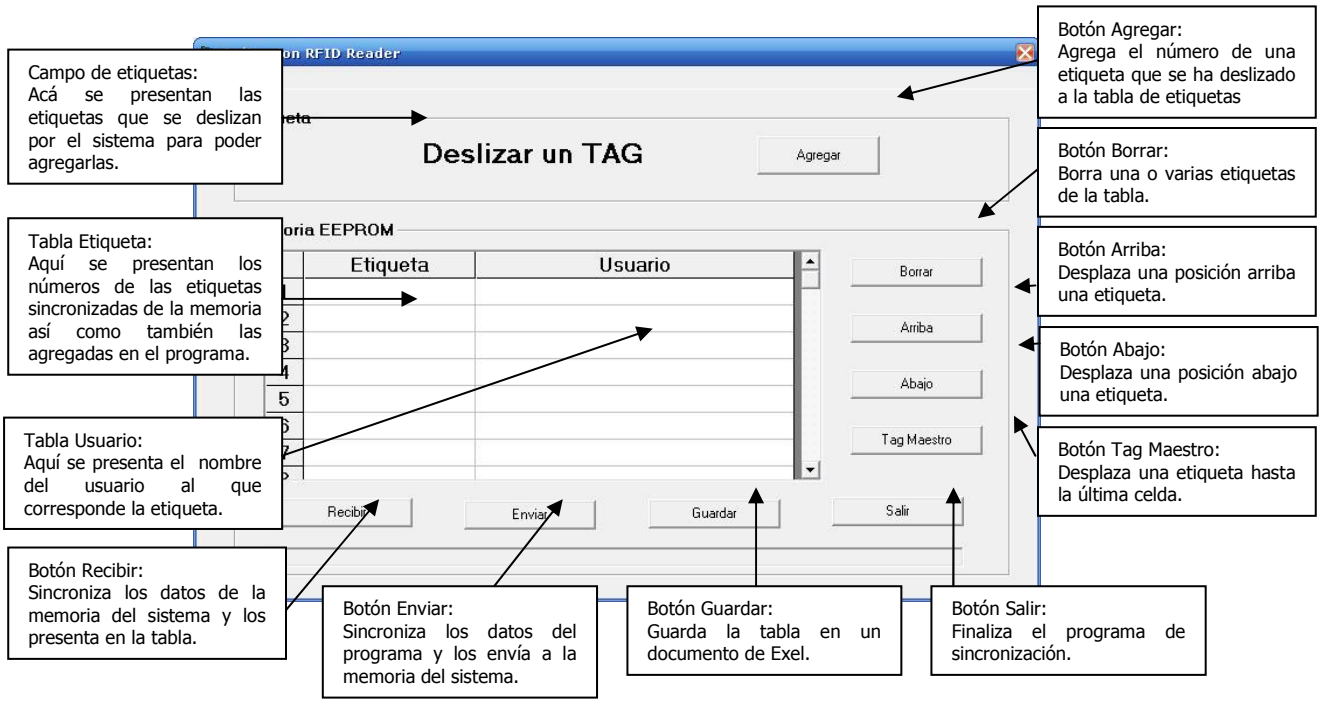


Figura 9.2.2.4

Cuando la tarjeta es desconectada del puerto USB sin salirse del programa, éste detecta que la tarjeta se ha desconectado y muestra el mensaje de la figura 9.2.2.5.

Figura 9.2.2.5

Al conectar de nuevo la tarjeta al puerto USB, el mensaje desaparece y vuelve a salir en pantalla la ventana pidiendo la contraseña de administrador, figura 9.2.2.1.

Campo de Etiquetas: En este lugar, se mostrara el numero de la etiqueta que se deslice por el sistema lector, con la finalidad de poder agregarla a la tabla de etiquetas y si se desea sincronizar el sistema para guardar el numero de etiqueta en la memoria del sistema.

Tabla de Etiquetas: En este lugar, se muestran todas las etiquetas que están grabadas en la memoria del sistema después de haber hecho una sincronización de recibir, o bien, las etiquetas que han sido agregadas a la tabla para posteriormente hacer una sincronización de enviar y así grabarlas en la memoria del sistema.

Tabla de Usuarios: En este lugar, se agregan (editan) los usuarios a los cuales corresponde cada etiqueta que se encuentra a su izquierda; la forma de editar cada una de estas celdas, es: haciendo doble clic sobre una de ella o bien colocarse encima y escribir el nombre del usuario.

Botón Recibir: Al hacer clic en este botón, se sincronizan los datos de todos las etiquetas que se encuentran guardados en la memoria del sistema, es decir, los números de las etiquetas que se encuentran guardadas en la memoria pasan a la tabla de etiquetas para posteriormente poder administrarlas.

Botón Enviar: Al contrario del botón de recibir, el botón enviar, sincroniza los datos de las etiquetas que se encuentran en la tabla de etiquetas para guardarlos en la

memoria del sistema, es decir, las etiquetas que han sido guardadas, o bien borradas de la tabla de etiquetas para posteriormente guardarlas en la memoria del sistema.

Botón Guardar: Al hacer clic en este botón, automáticamente se genera un documento de Excel, el cual contiene la información de la tabla de etiquetas y usuarios. Este documento es guardado en el directorio raíz del ordenador "C:\".

Botón Agregar: Al hacer clic en este botón, se agrega a la tabla de etiquetas la etiqueta que se muestra en el campo de etiquetas para su posterior administración y sincronización con el sistema lector.

Botón Borrar: Este botón sirve para borrar una o varias etiquetas de la tabla de etiquetas. Para borrar una etiqueta se selecciona y se hace clic en el botón; para borrar varias etiquetas se seleccionan y se hace clic en el botón, para borrar toda la tabla de etiquetas, se hace clic en la celda de titulo de las etiquetas y se selecciona automáticamente todos los campos luego al hacer clic en el botón borrar, se borra todo el contenido, para su posterior administración o sincronización con el sistema lector.

Botón Arriba: Este botón sirve para la administración de las etiquetas. Se selecciona una etiqueta y al hacer clic, la etiqueta sube una posición.

Botón Abajo: Este botón sirve para la administración de las etiquetas. Se selecciona una etiqueta y al hacer clic, la etiqueta baja una posición.

Nota: Las funciones del botón arriba y abajo son meramente para administración de la memoria, es decir, para mantener ordenadas las etiquetas que se encuentran o serán grabadas en la memoria del sistema; también para mover hacia la última localidad la etiqueta de administración. Cabe mencionar que la utilización de estos botones es para una celda a la vez; no se pueden seleccionar varias celdas para moverlas.

Botón Tag Maestro: Al hacer clic en este botón, se traslada la etiqueta seleccionada a la última posición de la columna de etiquetas, con la finalidad de que funcione como etiqueta de administración.

Botón Salir: Al hacer clic en este botón, se finaliza el programa de sincronización con el sistema lector y a su vez, se detiene la comunicación USB.

NOTA: El archivo generado en Excel a partir de los datos de la tabla, queda guardado en la ruta "C:\" por defecto, sin embargo, no tienen ningún tipo de seguridad, por lo tanto se recomienda al administrador del sistema, encriptar el archivo o dotarlo de algún tipo de seguridad como lo es una contraseña para que así ninguna persona ajena pueda tener acceso a la información contenida en el.

9.2.3 Modo para el acceso de personal (cerradura electrónica)

A este modo de trabajo se accede poniendo en el selector minidip la combinación "01", esto quiere decir, poniendo un selector en posición "OFF" y el otro en "ON", al haber realizado lo anterior, en la pantalla de cristal liquido (LCD) aparecerá la leyenda que cita lo siguiente "Cerradura Electrónica", como se muestra en la figura 9.2.3.1. Esto indica que el sistema está listo para iniciar su funcionamiento en el modo para el acceso o bien cerradura electrónica.

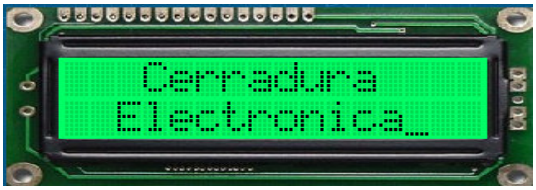


Figura 9.2.3.1

Cuando se desliza una etiqueta que esta previamente guardada en la memoria del sistema, este dará acceso, activara una salida de potencia del sistema, que abrirá la cerradura electrónica; al suceder esto, en la pantalla del sistema aparecerá una

leyenda que cita lo siguiente "TAG Aceptado / No: ". Todo lo contrario sucede cuando la etiqueta que se desliza no está guardada en la memoria del sistema, este no habilita el acceso y por lo contrario cita "TAG Denegado / No: ". Esto se muestra en las figuras abajo.

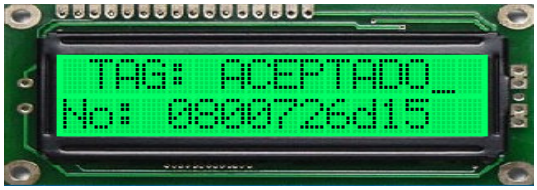


Figura 9.2.3.2

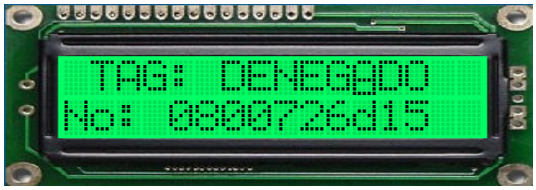


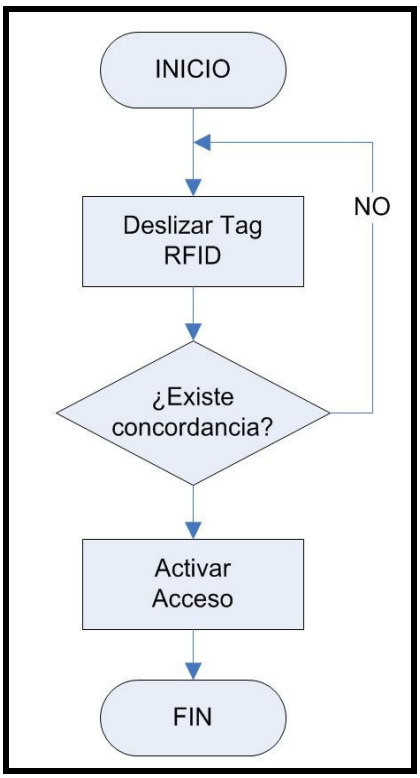
Figura 9.2.3.3

El modo para el acceso o cerradura electrónica, posee una característica particular. En este modo, el sistema tiene la posibilidad de interactuar con una computadora personal, es decir, el sistema se puede conectar a una PC, vía USB para sincronizar los números guardados en la memoria del sistema.

El diagrama de bloques del sistema cuando opera en este modo se muestra a continuación.

Diagrama 9.2.3.4 Diagrama de bloques
para el control de acceso de personal

El flujograma básico de operación del usuario para el modo de control de acceso o cerradura electrónica se muestra a continuación.



Flujograma 9.2.3.5 Flujograma para el control de acceso de personal

9.2.4 Modo para el acceso e ignición de un automotor

A este modo de trabajo se accede poniendo en el selector minidip la combinación "00" o bien poniendo ambos selectores en su estado de apagado "OFF", al haber realizado lo anterior, en la pantalla de cristal liquido (LCD) aparecerá la leyenda que cita lo siguiente "Sistema de seguridad-Auto", como se muestra en la figura 9.2.4.1. Esto indica que el sistema está listo para iniciar su funcionamiento en el modo para el acceso e ignición de un automotor.

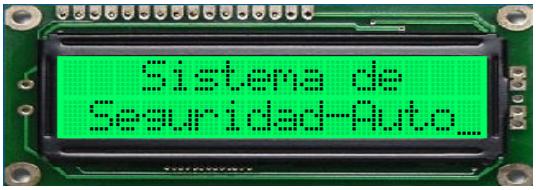


Figura 9.2.4.1

Al estar listo el sistema para su funcionamiento, estará esperando la presencia de una etiqueta para proporcionar o denegar el acceso al automóvil, así como también habilitar o deshabilitar la ignición del automotor.

Cuando se desliza una etiqueta que esta previamente guardada en la memoria del sistema, este dará acceso al automóvil, tanto en quitar los seguros de las puertas así como también habilitara la ignición del motor; al suceder esto, en la pantalla del sistema aparecerá una leyenda que cita lo siguiente "TAG Aceptado / No: ". Todo lo contrario sucede cuando la etiqueta que se desliza no está guardada en la memoria del sistema, este no habilita el acceso ni la ignición y por lo contrario cita "TAG Denegado / No: ". Esto se muestra en las figuras abajo.

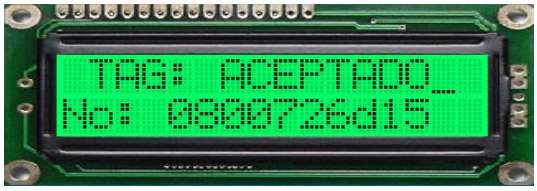


Figura 9.2.4.2

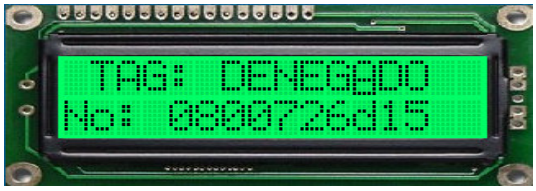


Figura 9.2.4.3

El modo para el acceso e ignición de un automotor, posee una característica particular. En este modo, el sistema tiene la posibilidad de interactuar con una computadora personal, es decir, el sistema se puede conectar a una PC, vía USB para sincronizar los números guardados en la memoria del sistema.

Cabe mencionar que el automóvil, poseerá los dos sistemas para la apertura de las puertas, ya sea por llave o por RFID.

Al validarse el sistema de RFID, éste quita los seguros de las puertas y habilita al sistema tradicional de encendido del automóvil, es decir, que primero se tendrá que validar el sistema de RFID y luego encender el automóvil, ya que de otra manera no funcionará.

A continuación se muestran los diagramas de bloques y flujogramas del funcionamiento del sistema.

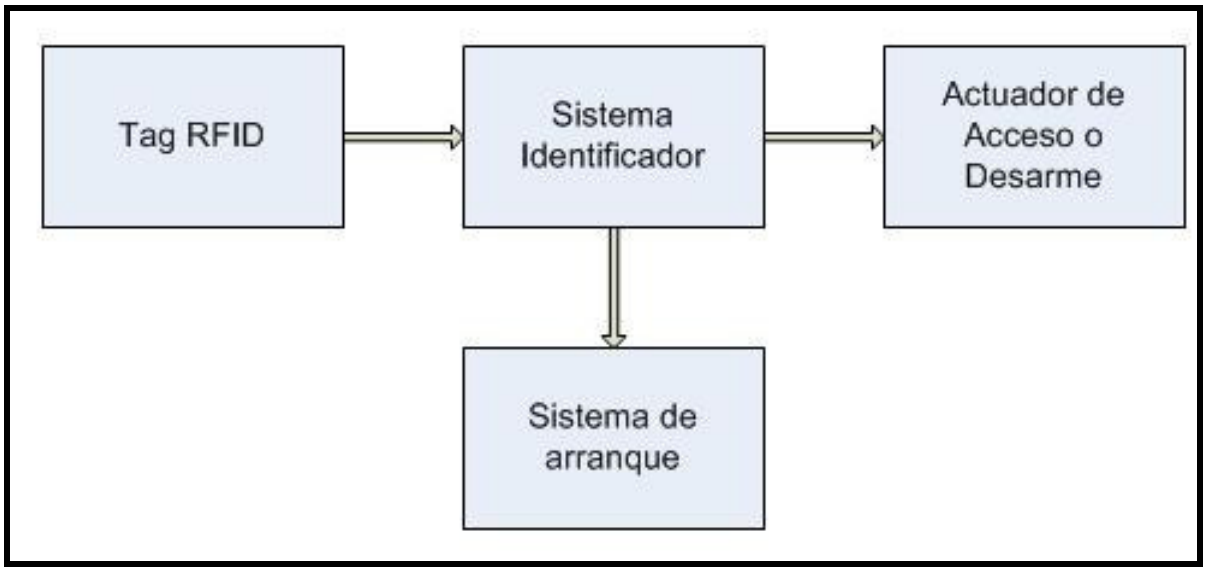
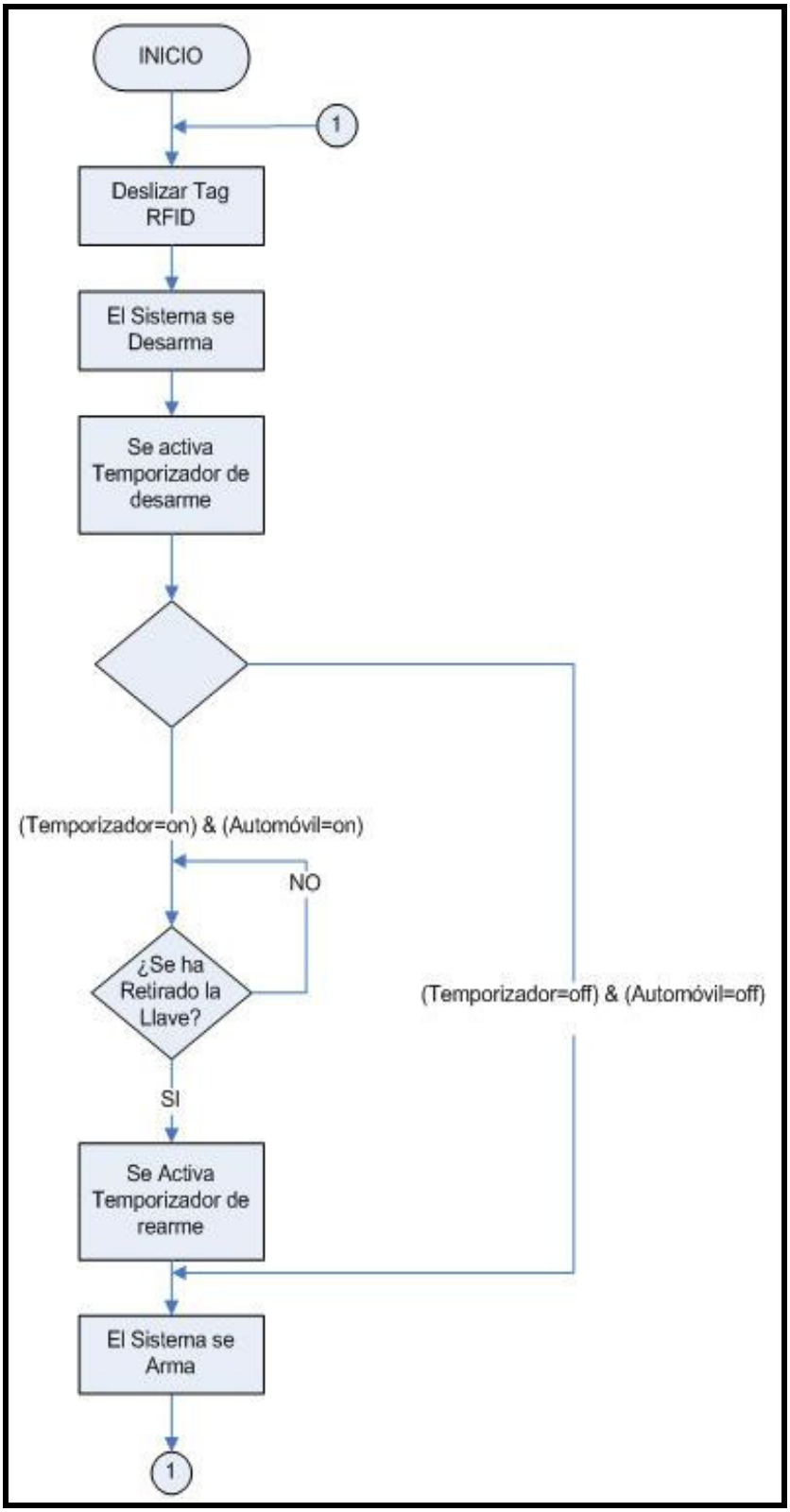


Diagrama 9.2.4.4 Diagrama de Bloques para el control de acceso e ignición de un automotor



Flujograma 9.2.4.5 Flujograma para el control de acceso e ignición de un automotor

9.2.5 Modo para Marcación de horas de trabajo

A este modo de trabajo se accede poniendo en el selector minidip la combinación "11", esto quiere decir, poniendo los selectores en posición "ON", al haber realizado lo anterior, en la pantalla de cristal liquido (LCD) aparecerá la leyenda que cita lo siguiente "Marcador Horas / Inicio Windows", como se muestra en la figura 9.2.5.1. Esto indica que el sistema está listo para iniciar su funcionamiento en el modo para el sistema marcador de horas de trabajo.

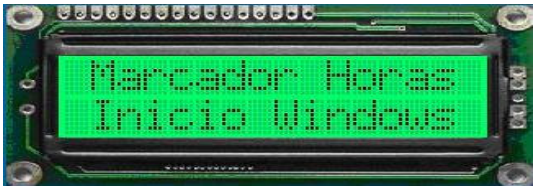


Figura 9.2.5.1

Este modo de trabajo del sistema lector se caracteriza por ser un modo de comunicación, es decir, que el sistema lector funciona en conjunto con una computadora personal. Cuando se desliza una etiqueta sobre el sistema lector, éste envía el numero de etiqueta por la interfaz USB a la computadora personal, para lo cual en la PC, se tendrá el software que funciona en conjunto con esta aplicación.

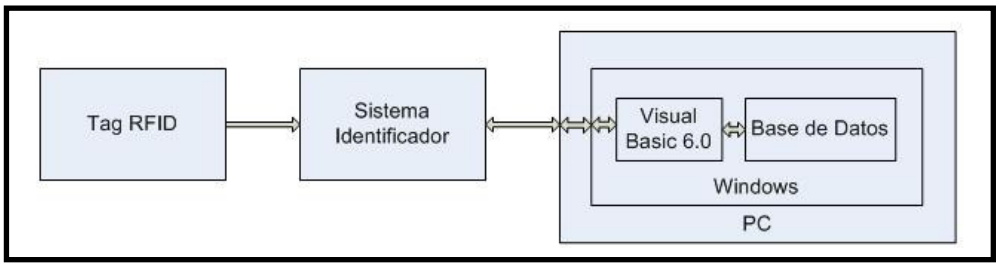


Figura 9.2.5.2 Diagrama de bloques para el modo de marcación de horas de trabajo

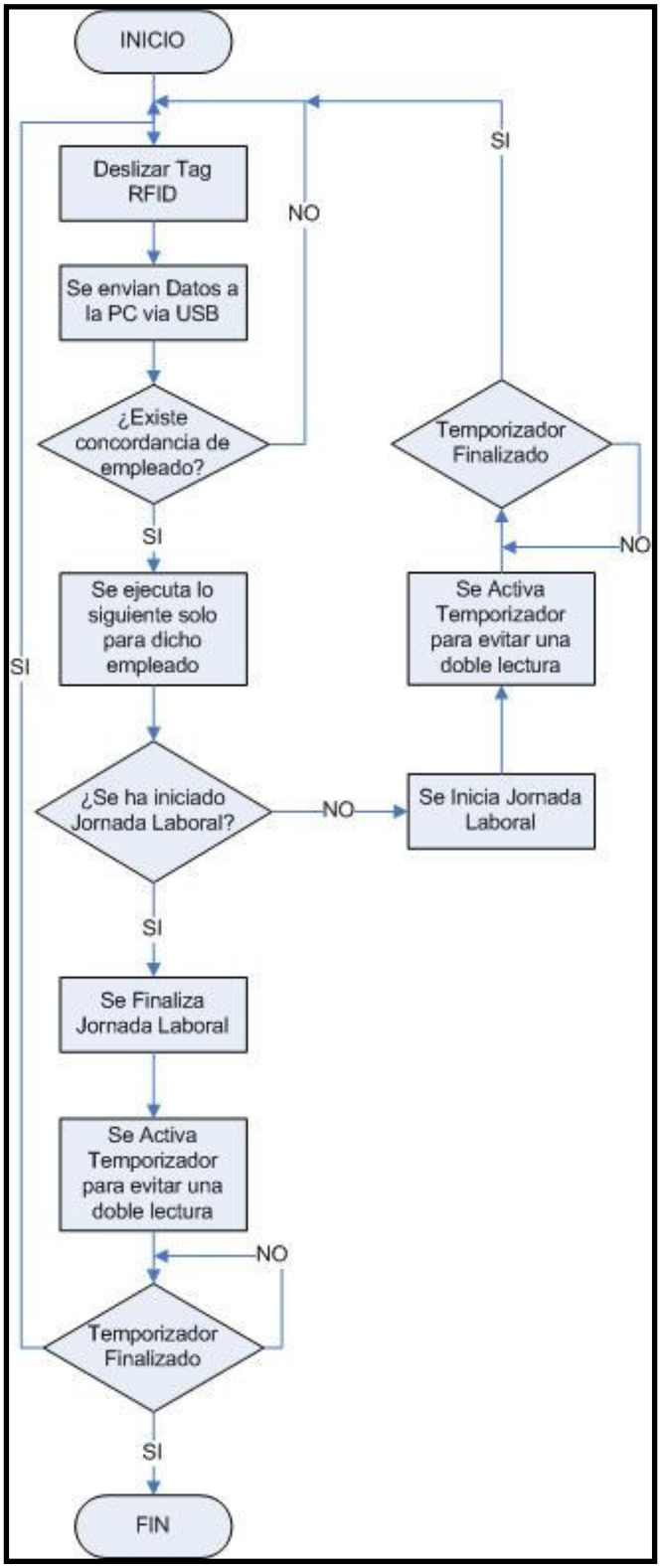


Figura 9.2.5.3 Diagrama de flujo para el modo de marcación de horas de trabajo.

9.2.6 Programa en Visual Basic 6 para la marcación de horas de trabajo.

El programa para la marcación de horas de trabajo de empleados, tiene como finalidad guardar registros de la hora de entrada así como también la hora de salida de empleados. Al iniciar el programa de marcación de horas de trabajo, observamos la pantalla de la figura abajo, que es la pantalla principal del sistema.

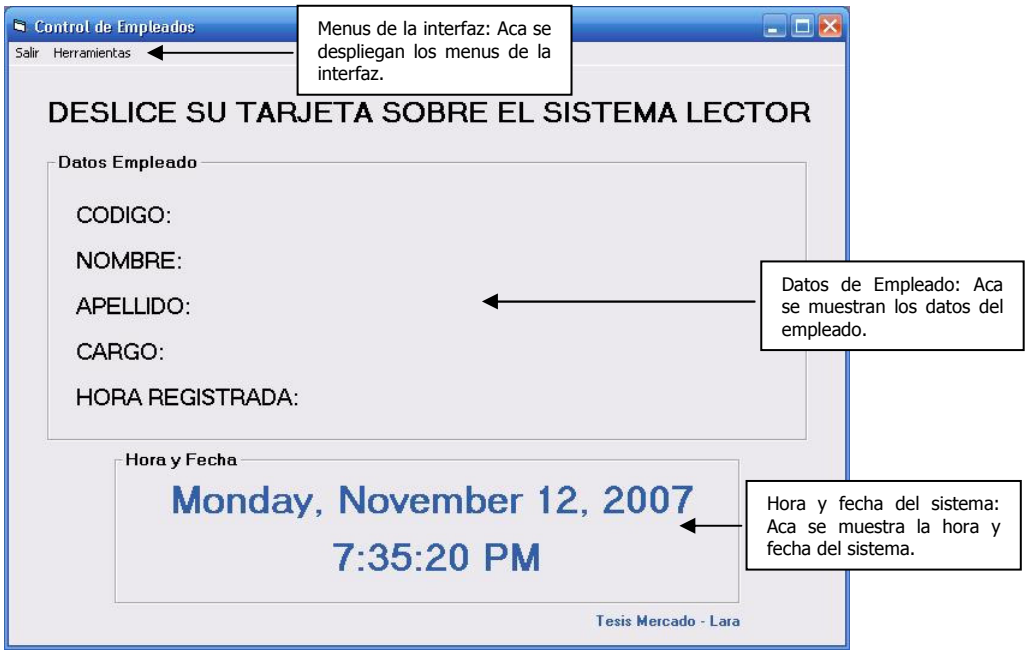


Figura 9.2.6.1 Formulario principal del sistema

Datos de empleado: En esta sección de la interfaz, se presentan algunos datos del empleado. Cuando el empleado desliza su etiqueta por el sistema lector, automáticamente, se presentan en pantalla sus datos y registra la fecha y hora de entrada.

Hora y fecha del sistema: En esta sección de la interfaz, se presenta la fecha y la hora del sistema.

Menús del sistema: La interfaz posee menús desplegables, que al hacer clic sobre uno de ellos, realiza una acción previamente programada.

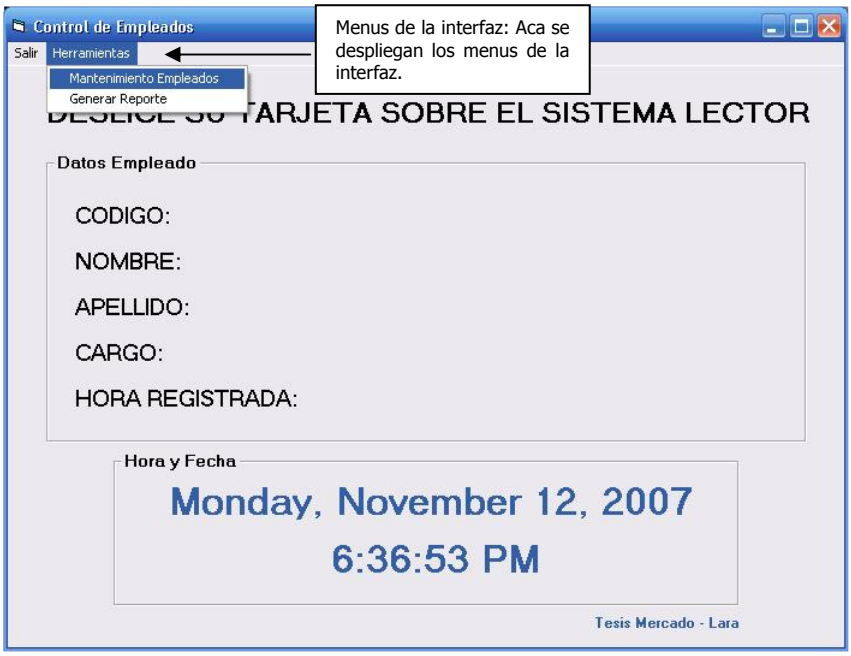


Figura 9.2.6.2 Menús del formulario principal del sistema.

Menú Salir: Al hacer clic sobre este menú, el programa finaliza su ejecución.

Menú Mantenimiento de empleados: Al hacer clic sobre este menú, se despliega otra ventana, que contiene información mas detallada del empleado, en dicha ventana, se puede ingresar, modificar o borrar información de un empleado, así como también borrar un empleado o toda la nomina de empleados.

Menú generar reporte: Al hacer clic en este menú, se despliega otra ventana, que contiene un registro de las horas trabajadas por los empleados.

Al deslizar una etiqueta por el sistema lector; automáticamente, el sistema registra la hora y fecha del empleado asociado a la etiqueta deslizada, así como se muestra en la figura abajo.

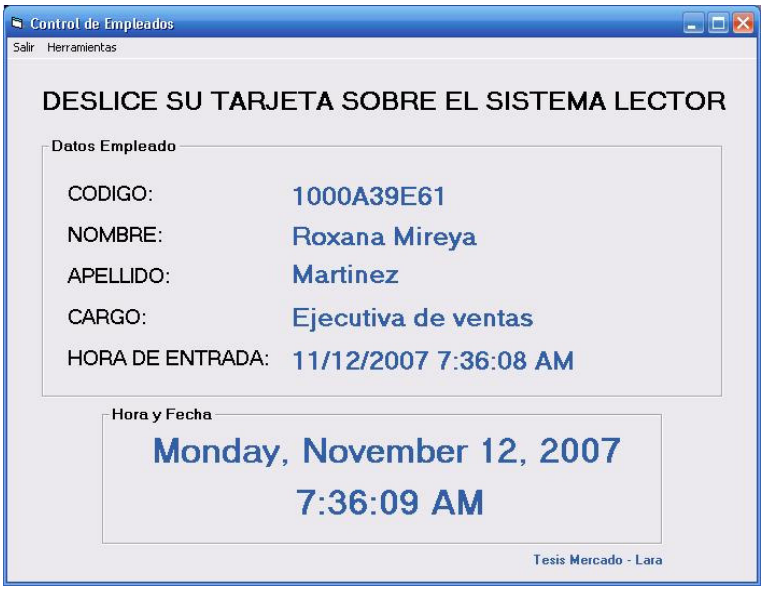


Figura 9.2.6.3 Sistema registrando hora de entrada.

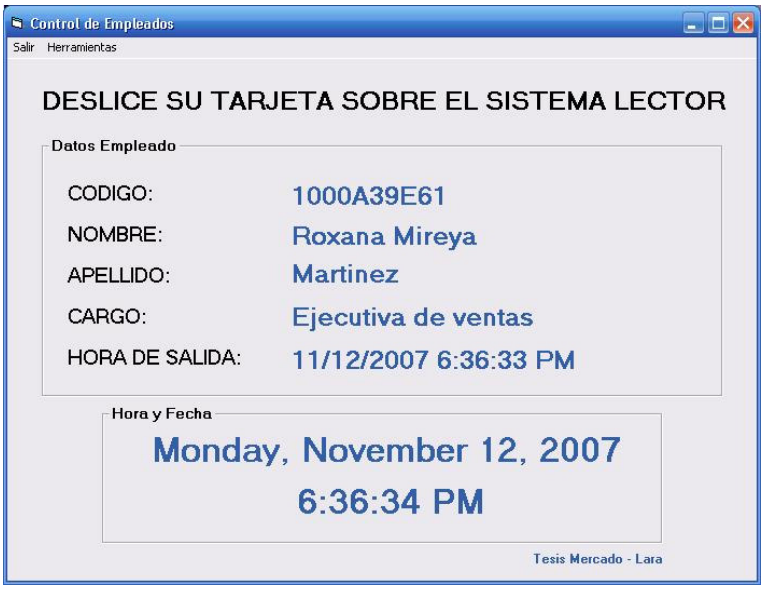


Figura 9.2.6.4 Sistema Registrando hora de salida.

Al hacer clic sobre el menú mantenimiento de empleados que se encuentra en el formulario principal, se presenta la figura mostrada abajo, la cual nos permite ingresar nuevos empleados, modificar, borrar empleados que están contenidos en la tabla o bien eliminar por completo toda la información de los empleados contenida.

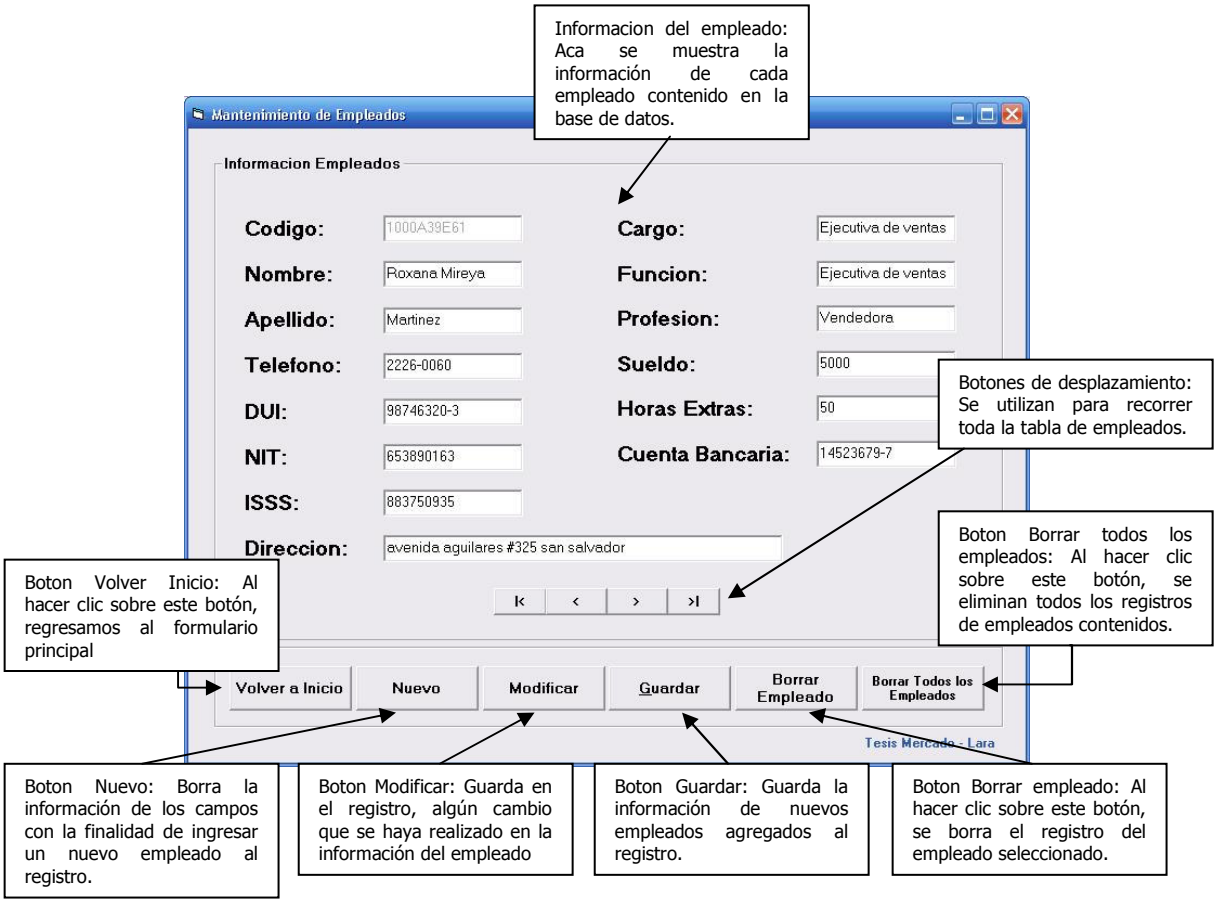


Figura 9.2.6.5 Formulario de Mantenimiento de Empleados

Botón Volver a Inicio: Al hacer clic sobre este botón, se cierra el formulario mantenimiento de empleados y se vuelve al formulario principal del sistema.

Botón Nuevo: Al hacer clic sobre este botón, automáticamente se borra la información contenida en los campos que muestran la información del empleado, esto con la finalidad de poder ingresar los datos de un nuevo empleado.

Botón Modificar: Al hacer clic sobre este botón, se actualiza el registro de datos de empleado. Este botón tiene utilidad siempre y cuando se haya hecho una modificación a la información del empleado contenida en los campos de datos de empleado.

Botón Guardar: Este botón es el complemento del botón nuevo, ya que se utiliza para grabar la información de un nuevo empleado, al hacer clic sobre el botón nuevo, se borran todos los campos de información del empleado, para poder ingresar uno nuevo; seguido de esto, se introduce la información completa del nuevo empleado para luego poder almacenarla en el registro haciendo clic sobre el botón Guardar.

Botón Borrar Empleado: Este botón trabaja en conjunto con los botones de desplazamiento; se selecciona el empleado con dichos botones, una vez seleccionado, se hace clic en el botón borrar empleado y automáticamente, el empleado seleccionado queda eliminado del registro.

Botón Borrar todos los Empleados: Al hacer clic sobre este botón, se eliminan todos los registros de los empleados.

Botones de desplazamiento: Al hacer clic sobre cualquiera de estos botones, se desplaza por los empleados contenidos en el registro, mostrando la información concerniente a cada uno de ellos.

Al hacer clic sobre el menú Generar Reporte, que se encuentra en el formulario principal, se presenta la figura mostrada abajo, la cual nos permite crear reportes de las horas trabajadas por los empleados.

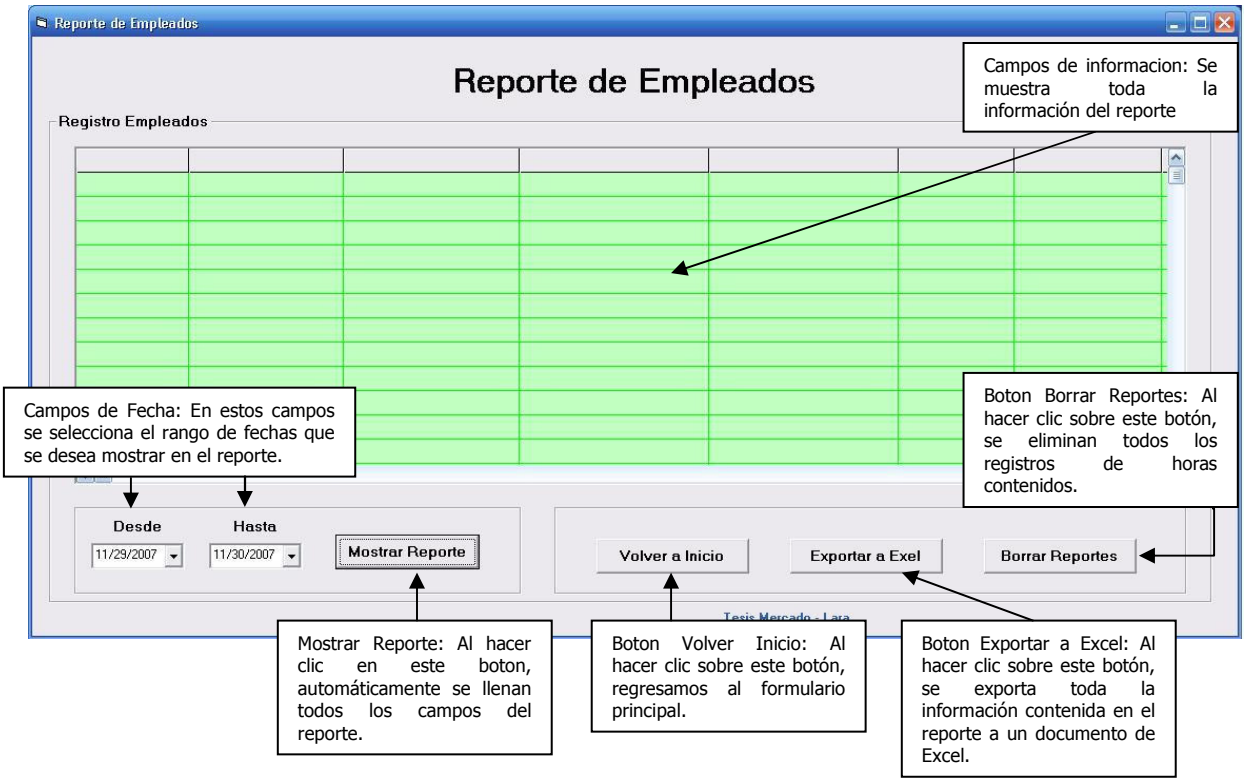


Figura 9.2.6.5 Formulario Reporte de horas trabajadas.

Campos de Fecha: En estos campos se introduce en rango de fechas que se desea mostrar en el reporte. Funciona en conjunto con el boton Mostrar Reporte.

Botón Mostrar Reporte: Al hacer clic sobre este botón, se toma el rango de fechas contenidos en los campos "desde y hasta" y se muestra en el formulario la información de los empleados refrente al rango de fechas introducido.

Botón Volver a Inicio: Al hacer clic sobre este botón, se cierra el formulario de reportes de empleados y se vuelve al formulario principal del sistema.

Botón Borrar Reportes: Al hacer clic sobre este botón, se eliminan todos los registros de horas trabajadas por los empleados contenidos en el sistema.

Figura 9.2.6.6 Informacion presentada en el formulario de Reporte

9.2.7 Modo para inicio de sesión y autenticación en Windows XP

Este modo de trabajo esta compartido con el de "Marcación de horas de trabajo" y se accede poniendo en el selector minidip la combinación "11", esto quiere decir, poniendo los selectores en posición "ON", al haber realizado lo anterior, en la pantalla de cristal liquido (LCD) aparecerá la leyenda que cita lo siguiente "Marcador Horas / Inicio Windows", como se muestra en la figura 10.2.6.1. Esto indica que el sistema está listo para iniciar su funcionamiento en el modo para el sistema marcador de horas de trabajo.

Figura 9.2.7.1

Este modo de trabajo del sistema lector se caracteriza por ser un modo de comunicación, es decir, que el sistema lector funciona en conjunto con una computadora personal. Cuando se desliza una etiqueta sobre el sistema lector, éste envía el numero de etiqueta por la interfaz USB a la computadora personal, para lo cual en la PC, se tendrá el software que funciona en conjunto con esta aplicación.

Cuando se presente la pantalla de inicio de sesión de Windows XP en la cual se pide el nombre de usuario y contraseña, solo bastará con deslizar la etiqueta sobre la tarjeta lectora y automáticamente los campos de usuario y contraseña se llenarán con la información adecuada.

A continuación se presenta para la aplicación del inicio de sesión y autenticación para Windows XP el diagrama de bloques y el diagrama de flujo para su uso.

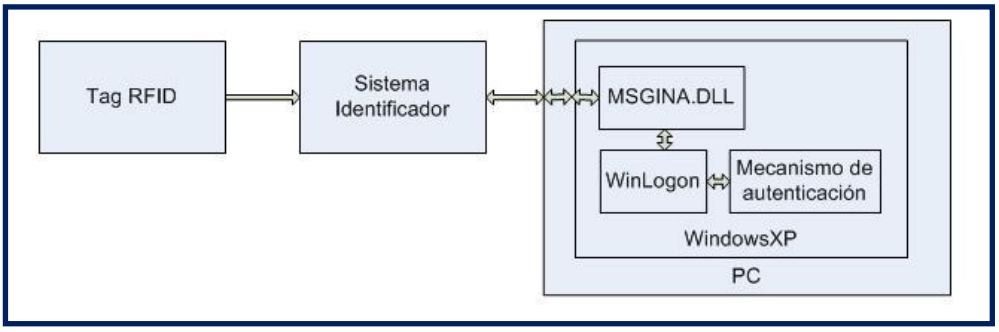


Figura 9.2.7.2 Diagrama de bloques para el modo de inicio de sesión y autenticación de WinXP.

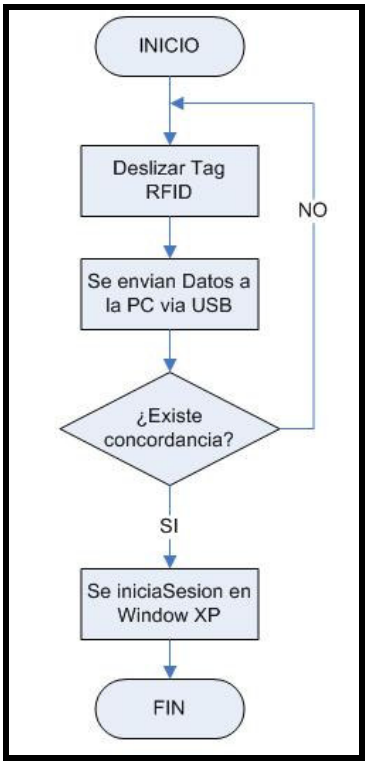


Figura 9.2.7.2 Diagrama de flujo para el modo de inicio de sesión y autenticación de WinXP.

9.2.7.1 Programa en Visual Basic 6 para Configurar las etiquetas RFID que pueden Iniciar Sesión en Windows XP

Los números de las etiquetas RFID y las credenciales de cuentas asociadas a ellas estarán presentes en el registro de Windows. Para crear y modificar esta información de una forma sencilla, se utilizara un software desarrollado en Visual Basic 6.0, para el óptimo funcionamiento con nuestro sistema lector RFID.

Para asociar etiquetas RFID a cuentas de usuario se procede a hacer lo siguiente:

- Asegurarse que la tarjeta lectora este en modo "marcación de Tarjeta"
- Conectar la tarjeta lectora al computador por la interfaz USB.
- Ejecutar el archivo RFIDAuth.exe, se desplegara una ventana como la de la figura abajo.

Figura 9.2.7.1.1 Programa para configurar los tags que pueden iniciar sesión en WinXP.

- Para asociar una etiqueta a una cuenta de usuario se debe hacer clic en el botón Agregar Tag; aparecerá una nueva ventana en la cual se hace la asignación del tag a la cuenta de usuario. Deslizar la etiqueta por el sistema lector y automáticamente se llena el campo de tag ID luego llenar los demás campos con la información pertinente y hacer clic en OK.

Figura 9.2.6.1.2 Ventana de Asignación de Tag a cuenta de usuario.

La información se guarda automáticamente en el registro de Windows, asociando la etiqueta deslizada por el lector a la cuenta de usuario ingresada.

Las etiquetas RFID que han sido ingresadas al registro y asociadas a los usuarios, se presentan en la ventana principal del programa RFIDAuth.exe.

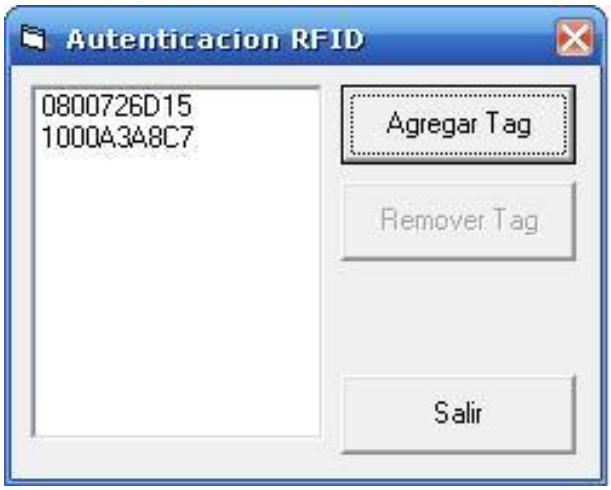


Figura 9.2.6.1.3 Ventana con tags asignados a cuentas de usuario.

Para remover una asociación de etiqueta RFID a una cuenta de usuario, se debe seleccionar la etiqueta que se desea eliminar y se hace clic en el botón remover Tag, para lo cual el programa solicitara confirmación del borrado de la etiqueta, como se muestra en la figura abajo.

Figura 9.2.6.1.3 Confirmación de eliminación de etiqueta.

10. IMPLEMENTACION DE LAS APLICACIONES DESARROLLADAS PARA EL SISTEMA RFID

10.1 IMPLEMENTACIÓ DEL SISTEMA PARA EL CONTROL DE ACCESO E IGNICIÓN DE UN AUTOMOTOR

El sistema se instalará en un Hyundai Accent del aaño 2002, a continuación se presentan los diagramas donde se pueden observar las líneas que deben ser conectadas para controlar el acceso y la ignición del automóvil, así como también los puntos físico donde se debe tomar las dichas líneas.

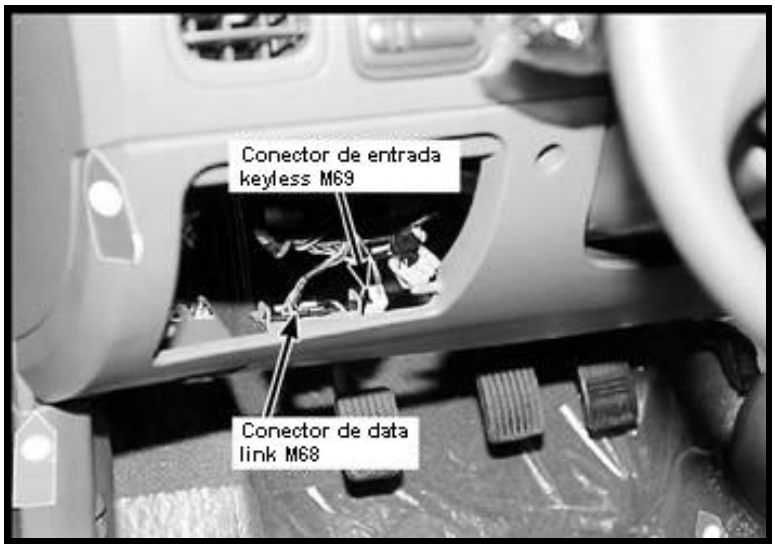


Figura 10.1.1 Compartimiento de conexiones de lado izquierdo del tablero de un Hyundai Accent 2002

En la figura 10.1.1 se muestra el compartimiento donde se encuentra la terminal de conexión OBDII del automóvil Hyundai, modelo Accent, año 2002; en dicho compartimiento, se debe instalar el sistema lector, en este se harán las conexiones necesarias hacia los subsistemas del automotor para controlar el acceso y la ignición del automóvil.

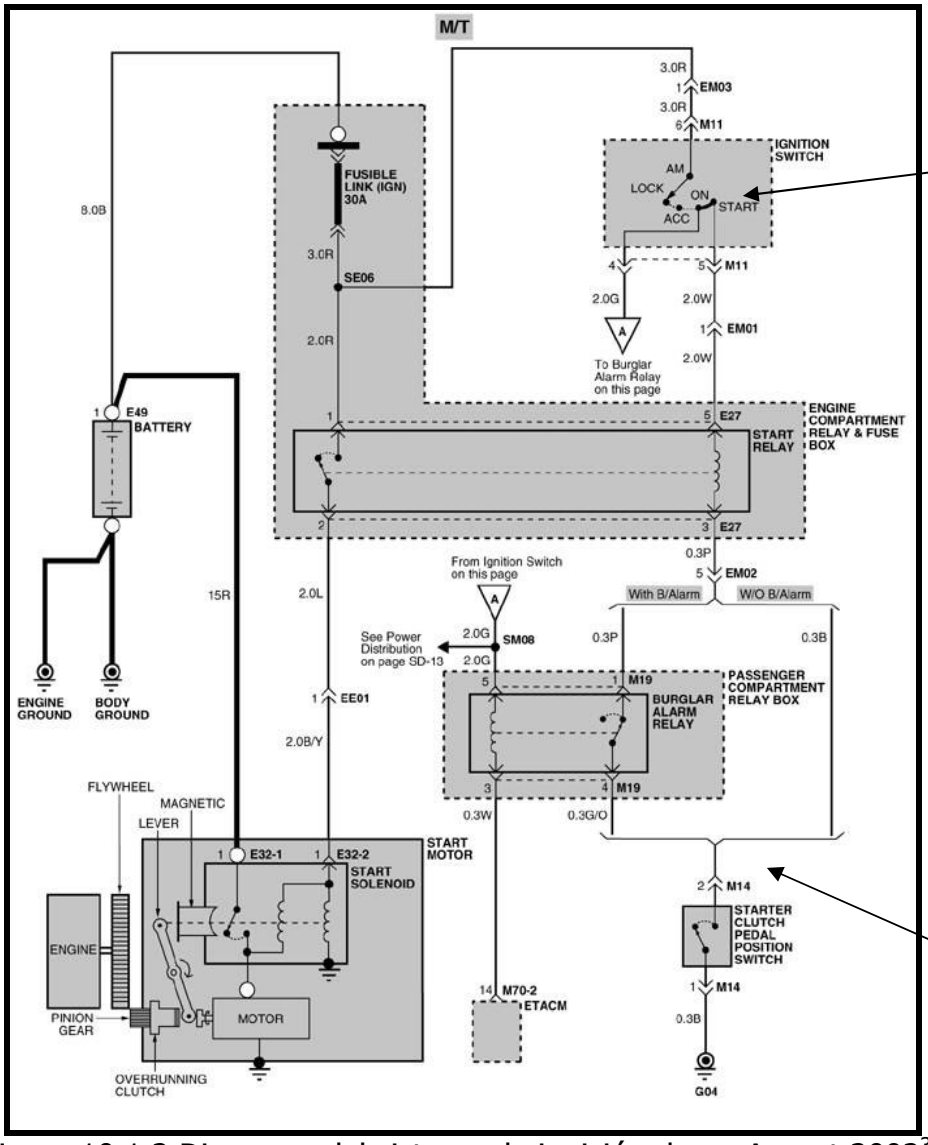


Figura 10.1.2 Diagrama del sistema de ignición de un Accent 2002[38]

La figura 10.1.2 muestra el diagrama de ignición del automotor Hyundai, modelo Accent, año 2002; en dicho diagrama se puede observar el cableado y los elementos que actúan para la ignición del automóvil.

El sistema lector, necesita una línea para detectar que el automóvil esta encendido, dicha línea se tomara del interruptor de ignición del automóvil,

[38] Diagrama tomado del "Ignition System Schematic ETM" (electrical troubleshooting manual) para Hyundai Accent 2002, http://www.hmaservice.com (accesado como miembro, septiembre de 2007)

específicamente de la línea de "ON"; a su vez, el sistema lector controlara el encendido; esto se hará colocando en serie el sistema RFID después del interruptor del embrague. Al estar conectado de esta manera, para que el motor de arranque pueda trabajar, el interruptor del clutch debe estar cerrado y el sistema RFID debe estar activado y haber detectado un tag autorizado.

10.2 IMPLEMENTACION DEL SISTEMA PARA EL INICIO DE SESION Y AUTENTICACION DE WINDOWS XP.

El software de PollGina[39] usado en esta aplicación, cambia la forma en que la computadora personal procesa la autenticación. Deshabilita el cambio rápido de usuario, y si el sistema normalmente requería que se presionara la combinación de teclas CRTL + ALT + DEL antes de iniciar alguna sesión, PollGina deshabilita ese requerimiento.

Los archivos[40] a utilizar para realizar esta aplicación son los siguientes:

- PollGina.dll: El archivo de reemplazo para GINA.
- RFIDAuth.exe: Usado para asociar etiquetas RFID a cuentas de usuario
- RFIDGina.reg: Archivo de registro con las configuraciones de RFIDGina.
- rfidPoll.dll: Librería RFID para PollGina.
- DllHidac.dll: Librería para control de dispositivos HID.

No existe ningún instalador para PollGina, mas bien, se tienen que copiar los archivos necesarios a las carpetas del sistema operativo correspondientes. Para instalar PollGina, mover o copiar los archivos **PollGina.dll, rfidPoll.dll y Hidac.dll** a la carpeta "C:\WINDOWS\system32".

[39] PollGina es una solución de código abierto que actua como interfaz entre el usuario y el GINA normal de Windows. El propósito es proveer métodos alternativos de autenticación. PollGina es un software de código abierto escrito por Nathan Yocom de XPA Systems.
[40] Archivos descargados de: http://www.rfidtoys.net/forum/forum_posts.asp?TID=4&PN=1 (Accesado en enero de 2006).

ACTUALIZANDO EL REGISTRO DE WINDOWS.

Actualización automática por archivo REG

Se necesitara modificar algunas entradas de registro de Windows y agregar algunas otras para que esta aplicación funcione con RFIDGina. Después de haber copiado los archivos mencionados anteriormente, solo es cuestión de ejecutar el archivo RFIDGina.reg para actualizar la configuración del registro. Hacer doble clic al archivo RFIDGina.reg para importar la correcta configuración en el registro local de la computadora. Como se muestra en la figura abajo, el cuadro de dialogo preguntara si se desea importar la información para actualizar el registro; al hacer clic, el sistema devolverá un mensaje confirmando que los cambios se han realizado con éxito.

Nota: Los cambios al registro se deben de realizar habiendo iniciado sesión con una cuenta de administrador.

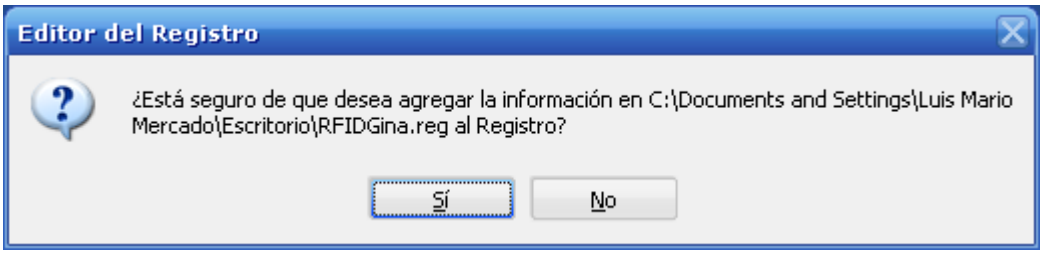


Figura 10.2.1 Confirmación de Cambio del Registro de Windows XP

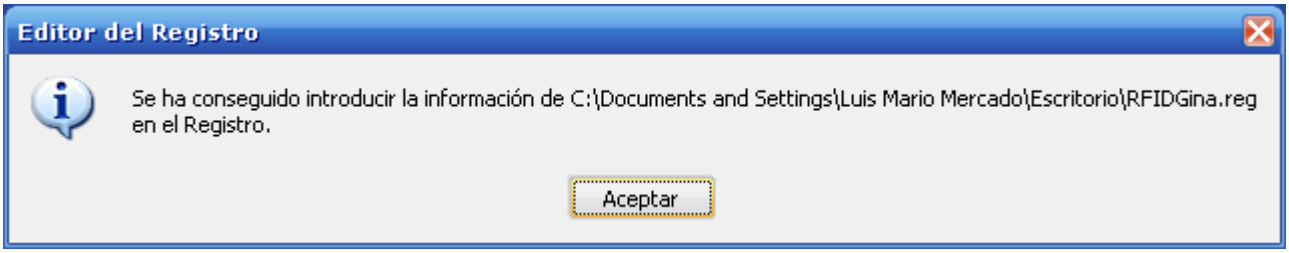


Figura 10.2.2 Mensaje de Cambio realizado al Registro de Windows XP

Actualización Manual

A continuación se detallarán los pasos para la correcta modificación del registro de Windows:

1. Abrir el editor de registro de Windows, haciendo clic en el botón inicio, luego seleccionar ejecutar y en la ventana de ejecutar digitar "regedit" (Inicio – Ejecutar – regedit); presionar enter o hacer clic en el botón aceptar.
2. Lo primero que se debe hacer es cambiar el Gina estándar que utiliza Windows por el PollGina de reemplazo. Para hacer esto, se debe agregar o modificar el valor de cadena de GinaDll (REG_SZ). Navegar por el árbol de registro hasta llegar a la siguiente clave de registro:

   ```
   HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
   NT\CurrentVersion\WinLogon
   ```

3. Buscar la cadena llamada GinaDLL. Si ya existe una cadena con ese nombre, se necesitará modificar su valor. Si no existiera, se deberá agregar una nueva cadena haciendo clic en el menú Edición, Nuevo, Valor alfanumérico como se muestra en la figura abajo. Nombrar el nuevo valor alfanumérico como GinaDLL. Tener en cuenta que el nombre es sensitivo a mayúsculas y minúsculas. El valor de GinaDLL deberá ser la ruta completa del archivo PollGina.dll que se copio anteriormente, esto se muestra en la figura abajo.

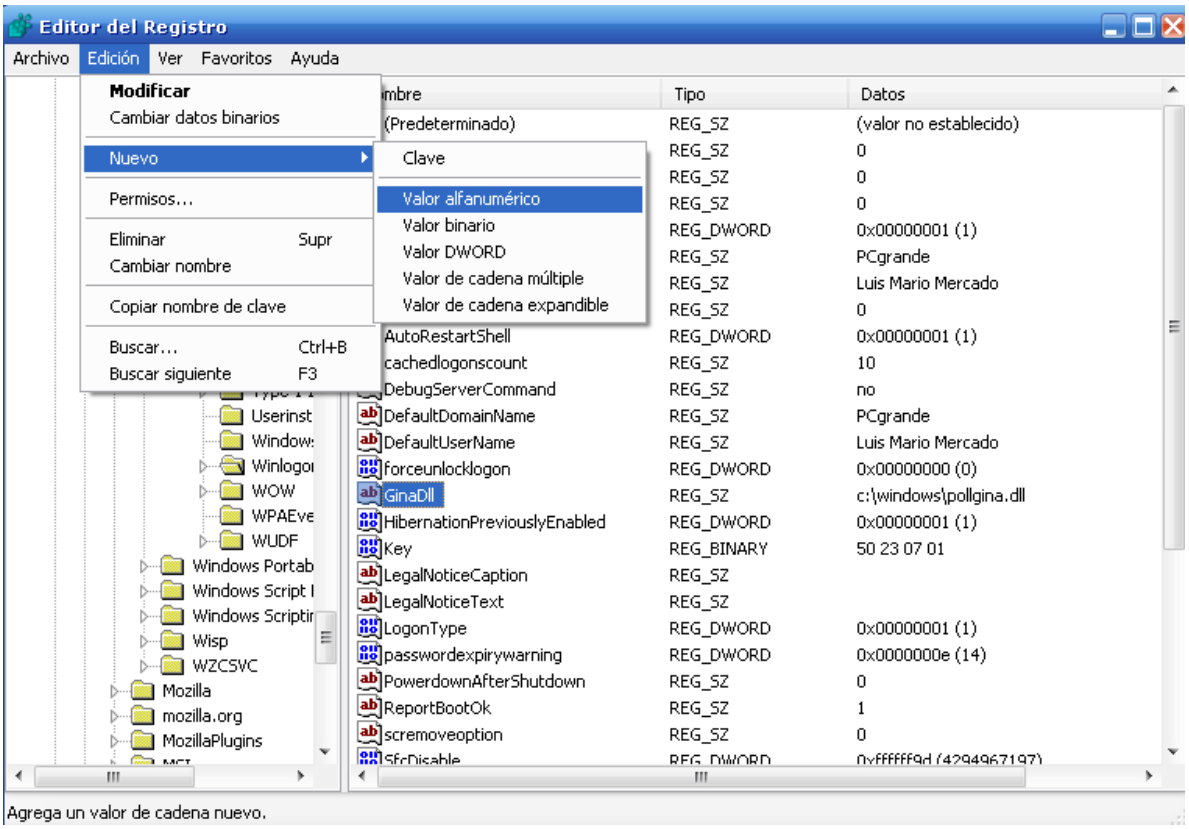


Figura 10.2.3 Ruta del archivo rfidpoll.dll

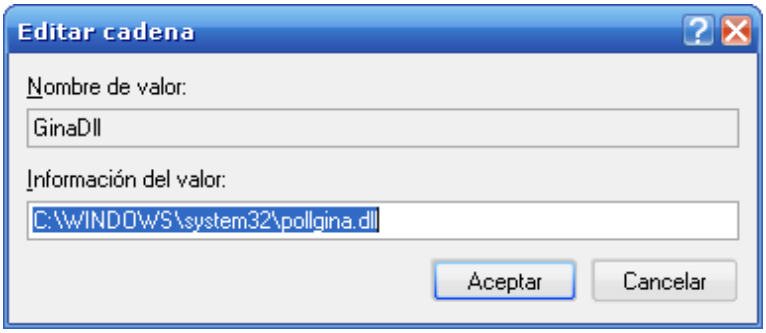


Figura 10.2.4 Cadena con la ruta del archivo pollgina.dll

4. Agregar nuevas claves de registro las cuales contendrán configuraciones adicionales de RFIDGina. Navegar en el registro hasta la siguiente clave:

   HKEY_LOCAL_MACHINE\SOFTWARE

5. Agregar una nueva clave llamada PollGina.

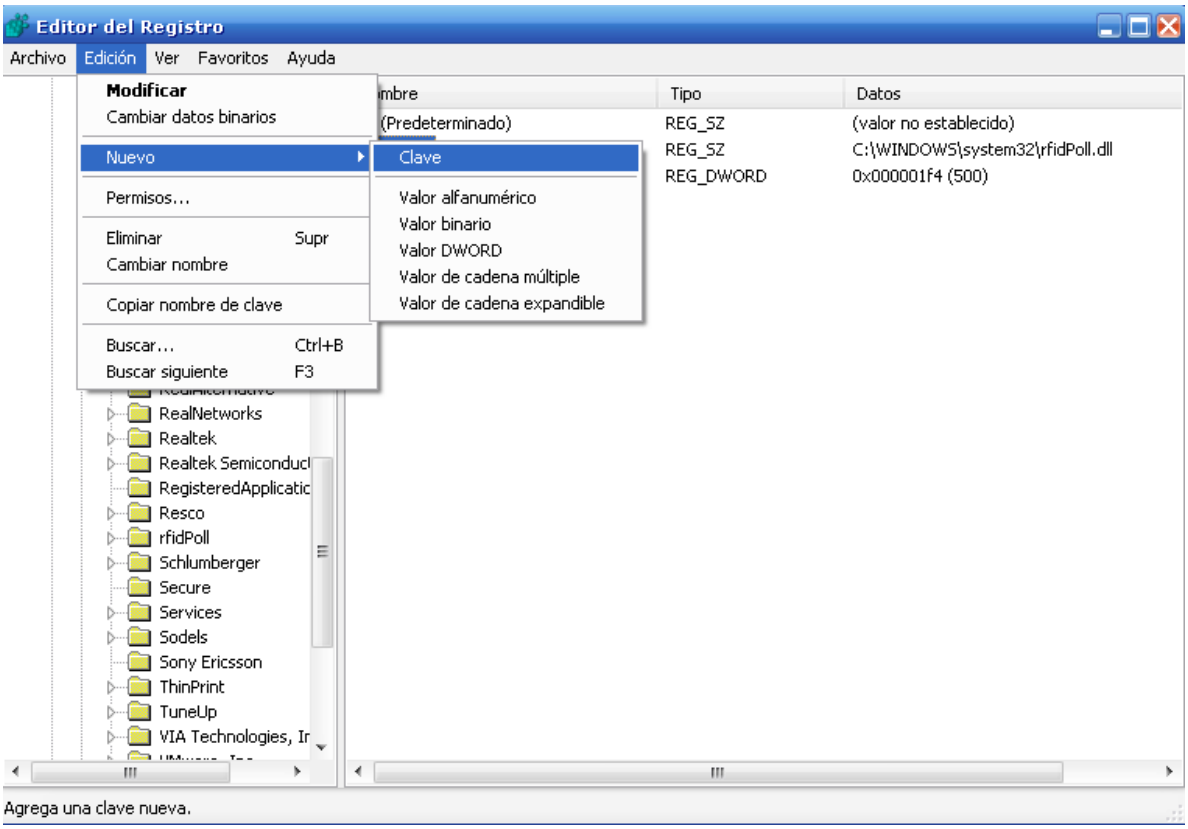


Figura 10.2.5 Agregando la clave PollGina.

6. Agregar un nuevo valor alfanumérico a la clave PollGina llamada pollLib. El valor de esta cadena deberá ser la ruta al archivo rfidPoll.dll, similar a lo que se realizo con GinaDll en el paso 3.

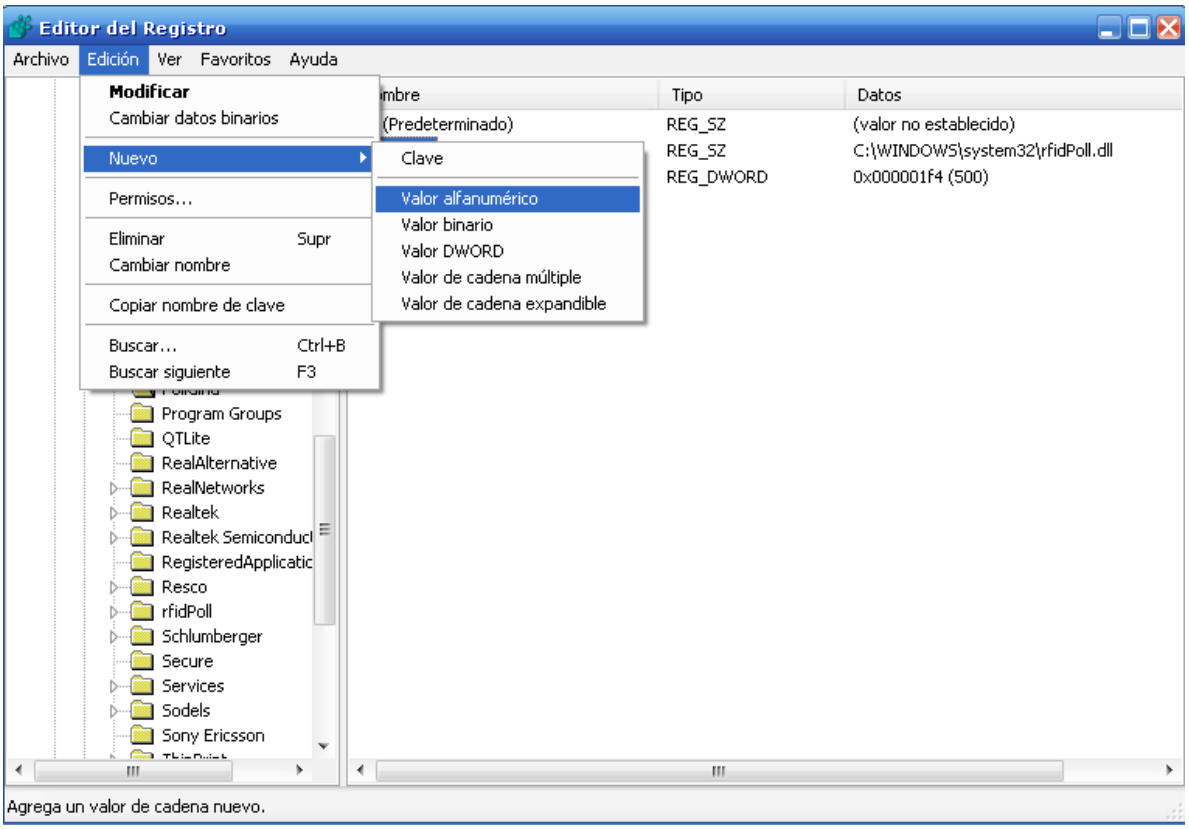


Figura 10.2.6 Agregando un valor alfanumérico a la clave PollGina.

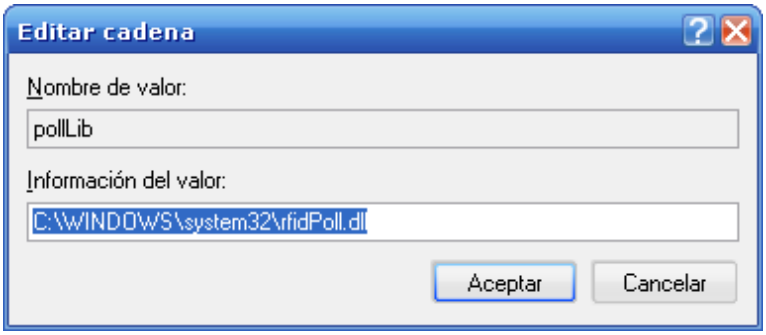


Figura 10.2.7 Cadena con la ruta del archivo rfidpoll.dll.

7. Agregar un valor DWORD a la clave pollGina, no un valor alfanumérico. Nombrar a este nuevo valor pollTime. Introducir un valor de 500 y asegurarse que la base sea decimal, como se muestra en la figura.

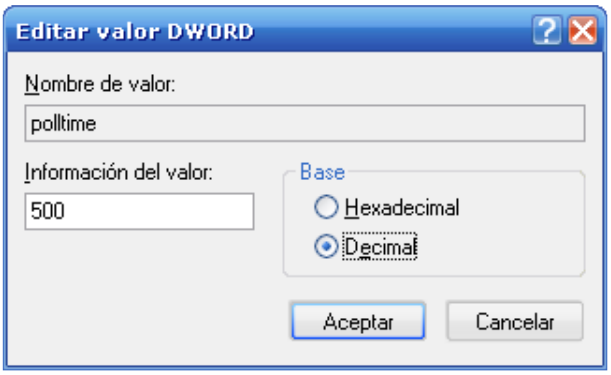


Figura 10.2.8 Agrando un valor pollTime
a la clave pollGina.

Después de haber realizado los pasos arriba mencionados, se habrá reemplazado satisfactoriamente el GINA estándar de Windows y configurado la librería RFID de polling, posteriormente se procederá a asignar los nombres de usuarios y contraseñas a las etiquetas RFID.

Configurar las etiquetas RFID para la autenticación

Para asociar etiquetas RFID a cuentas de usuario se procede a hacer lo siguiente:

- Asegurarse que la tarjeta lectora este en modo "marcación de Tarjeta"
- Conectar la tarjeta lectora al computador por la interfaz USB.
- Ejecutar el archivo RFIDAuth.exe, se desplegara una ventana como la de la figura abajo.

Figura 10.2.9 Programa RFIDAuth.exe

- Para asociar una etiqueta a una cuenta de usuario de hace clic en el botón Agregar Tag; aparecerá una nueva ventana en la cual se hace la asignación del tag a la cuenta de usuario. Deslizar la etiqueta por el sistema lector y automáticamente se llena el campo de tag ID luego llenar los demás campos con la información pertinente y hacer clic en OK.

Figura 10.2.10 Formulario de Tag y cuenta de usuario de Windows XP.

La información se guarda automáticamente en el registro de Windows, asociando la etiqueta deslizada por el lector a la cuenta de usuario ingresada.

Las etiquetas RFID que han sido ingresadas al registro y asociadas a los usuarios, se presentan en la ventana principal del programa RFIDAuth.exe.

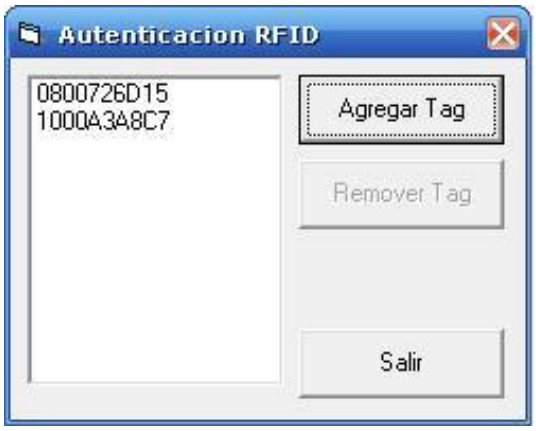


Figura 10.2.11 Programa RFIDAuth.exe con tags asociados a cuentas de usuario.

Habiendo realizado todos los pasos anteriores, se procederá a apagar el computador para conectar el sistema lector RFID y posteriormente se encenderá nuevamente. Al llegar al momento en que el computador pregunta por el nombre de usuario y contraseña, como se muestra en la figura abajo:

Figura 10.2.12 Inicio de Sesión en Windows XP.

Teniendo la pantalla mostrada arriba, se procederá a deslizar una etiqueta asociada a una cuenta de usuario por el sistema lector RFID, inmediatamente habiendo realizado lo anterior, los campos de nombre de usuario y contraseña se llenan automáticamente y se procede a la carga del escritorio del usuario asociado a la etiqueta que se deslizo por el sistema RFID.

Figura 10.2.12 Iniciando Sesión en Windows XP.

10.3 IMPLEMENTACION DEL SISTEMA PARA MARCACION DE HORAS DE TRABAJO

10.3.1 Sistema de información Principal

El sistema está desarrollado en Visual Basic 6.0, dicho sistema posee la capacidad de mostrar formularios con la información contenida en la base de datos de empleados, con facilidades para editar, borrar, actualizar e ingresar nueva información a la base de datos.

Grabará en la base de datos las horas de entradas y salidas de empleados a la empresa. Así mismo información personal de los trabajadores.

Para esto, el programa en diseñado en Visual Basic 6.0, se enlazara a la base de datos de empleados, mediante una conexión ADO.

10.3.2 Instalación de una instancia de SQL SERVER 2005[41]

Para instalar una instancia del motor de base de datos de SQL Server 2005, se deben seguir los siguientes pasos:

1. Inicie una sesión con una cuenta que tenga privilegios de administrador. Luego inserte el CD-ROM de SQL Server 2005 en la unidad de CD-ROM.

2. Si esta habilitada la ejecución automática, el programa de instalación de SQL Server 2005 debe iniciar automáticamente. De otra manera, haga doble clic en **splash.hta** dentro de los archivos del CD-ROM.

3. Bajo instalación, haga clic en Componentes de servidor, Herramientas, Libros en pantalla y ejemplo. Se despliega el contrato de licencia para el usuario final. Seleccione Acepto los términos y condiciones de la licencia y haga clic en siguiente.

Figura 10.3.2.1 Programa de instalación de Microsoft SQL Server 2005

[41] Tomado de: "Manual de Instalacion SQL Server 2005". Universidad Don Bosco. Diseñado por Virgilio Reyes.

4. Seleccione el asistente para la instalación de Microsoft SQL Server, haga clic en Siguiente. Entonces el asistente realizará una verificación de la configuración del sistema. Anote cualquier error y tome las acciones correctivas necesarias antes de seguir adelante. Si no se necesita este tipo de acciones, haga clic en Siguiente para proceder a la instalación.

Figura 10.3.2.2 Comprobación de configuración
(programa de instalación de Microsoft SQL Server 2005).

5. En la página información de registro, ingrese su nombre, el nombre de la compañía y haga clic en Siguiente.

6. En la página Componentes para instalar, seleccione los componentes que instalará. Seleccione las siguientes opciones y luego haga clic en Siguiente:

- SQL Server: Le permite instalar una instancia de SQL Server. También puede instalar SQL Server 2005 como parte de un clúster. Si detecta un clúster, la opción Servidor virtual estará seleccionada como opción predeterminada.
- Notification Services: Le permite instalar el motor de notificación y los componentes para generar y enviar notificaciones.
- Componentes de estación de trabajo, Libros en pantalla y Herramientas de desarrollo: Le permite instalar componentes de SQL Native Client, documentación y herramientas.

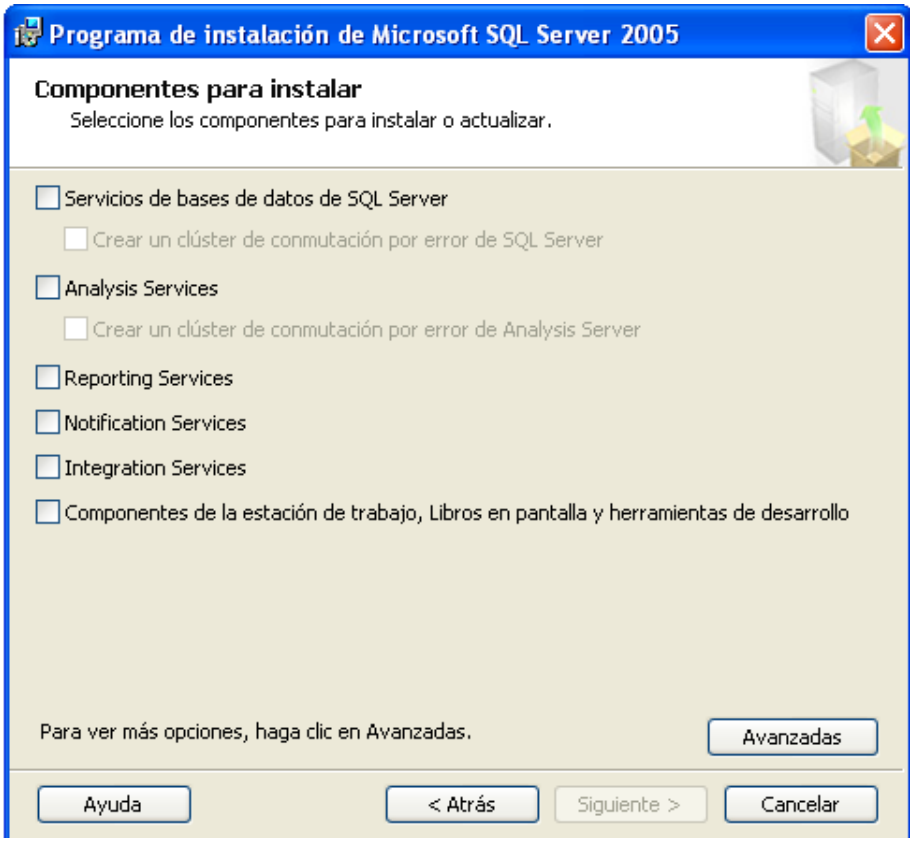


Figura 10.3.2.3 Selección de componentes
(programa de instalación de Microsoft SQL Server 2005).

8. Como se muestra en la figura de abajo, ahora debe determinar el tipo de instancia que instalará. Para nuestro caso debemos usar la instancia predeterminada de SQL Server, seleccione Instancia predeterminada y luego haga clic en Siguiente.

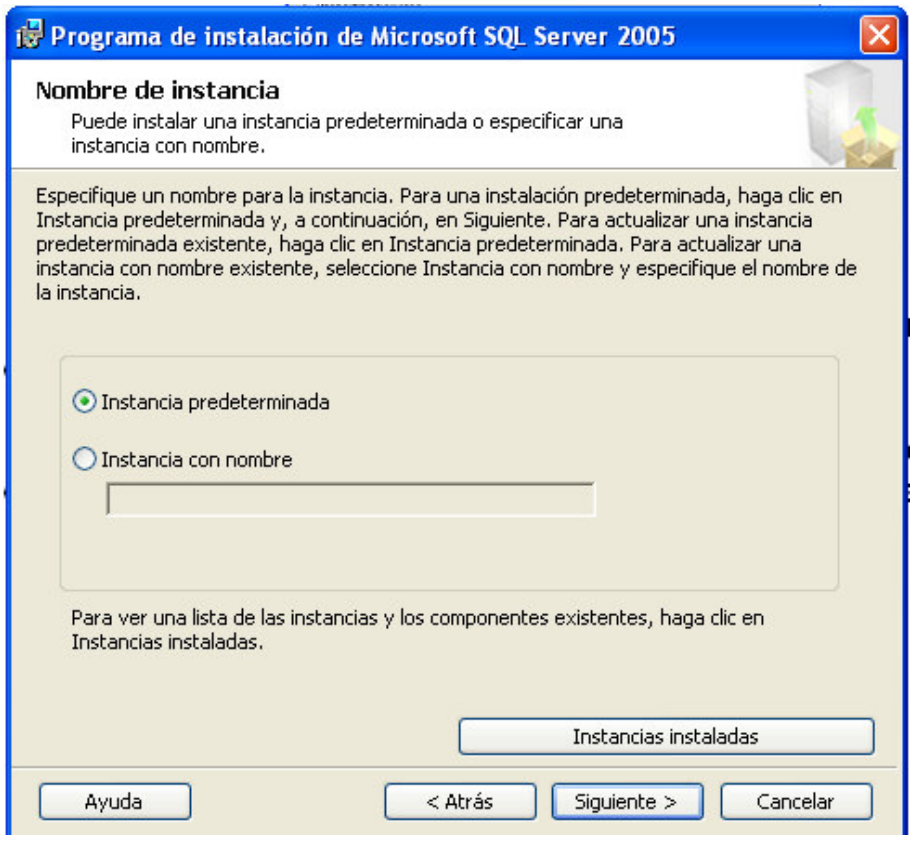


Figura 10.3.2.4 Nombre de Instancia
(programa de instalación de Microsoft SQL Server 2005).

9. En la Página Cuenta de servicio, se determina la manera en que se ejecutarán los servicios de SQL Server y Agente SQL Server. En nuestro caso debemos escoger: Usar la cuenta del sistema integrada con la opción de la pestaña Sistema local.

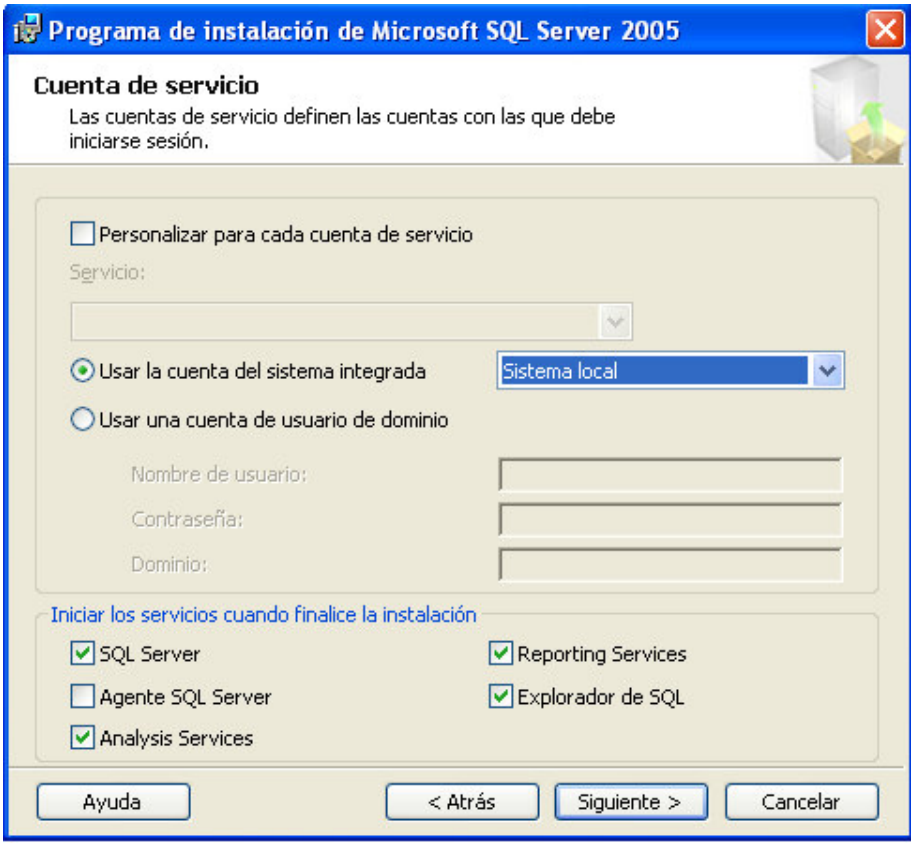


Figura 10.3.2.5 Cuenta de servicio
(programa de instalación de Microsoft SQL Server 2005).

10. Use la página Modo de autenticación para configurar los parámetros de autenticación. La instancia de SQL Server puede ejecutarse bajo autenticación de Microsoft Windows o de Modo Mixto. Con autenticación de Windows, sólo debe usar cuentas de usuario de dominio para autenticar conexiones con la instancia de SQL Server. Con la autenticación de modo Mixto, los usuarios pueden acceder a la instancia de SQL Server empleando cuentas de Usuario de dominio o ID de SQL Server. En nuestro caso debemos seleccionar la autenticación de Windows.

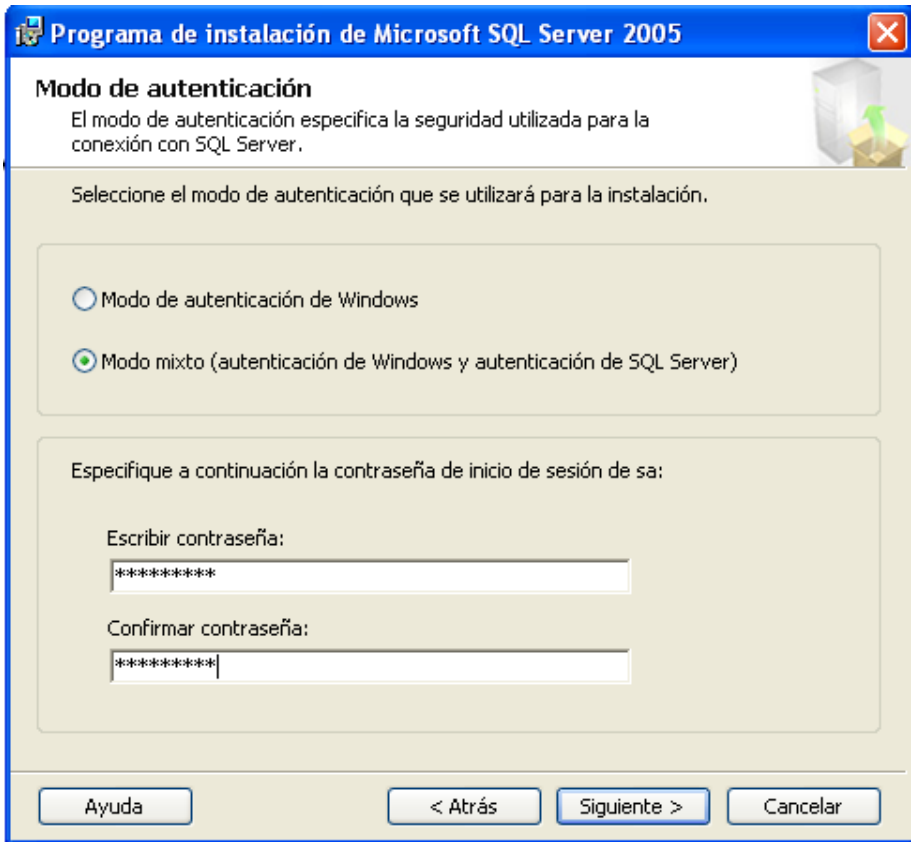


Figura 10.3.2.6 Modo de autenticación
(programa de instalación de Microsoft SQL Server 2005).

11. En la página configuración de intercalación, defina el comportamiento de ordenamiento para el servidor. Si selecciona personalizar para cada cuenta de servicio, puede especificar valores de intercalación diferentes para SQL Server y Analysis Services. Luego usaría las opciones de la lista desplegable para configurar valores separados para SQL Server Analysis Services antes de seguir adelante. En nuestro caso debemos seleccionar solamente las opciones que aparecen en la figura de abajo.

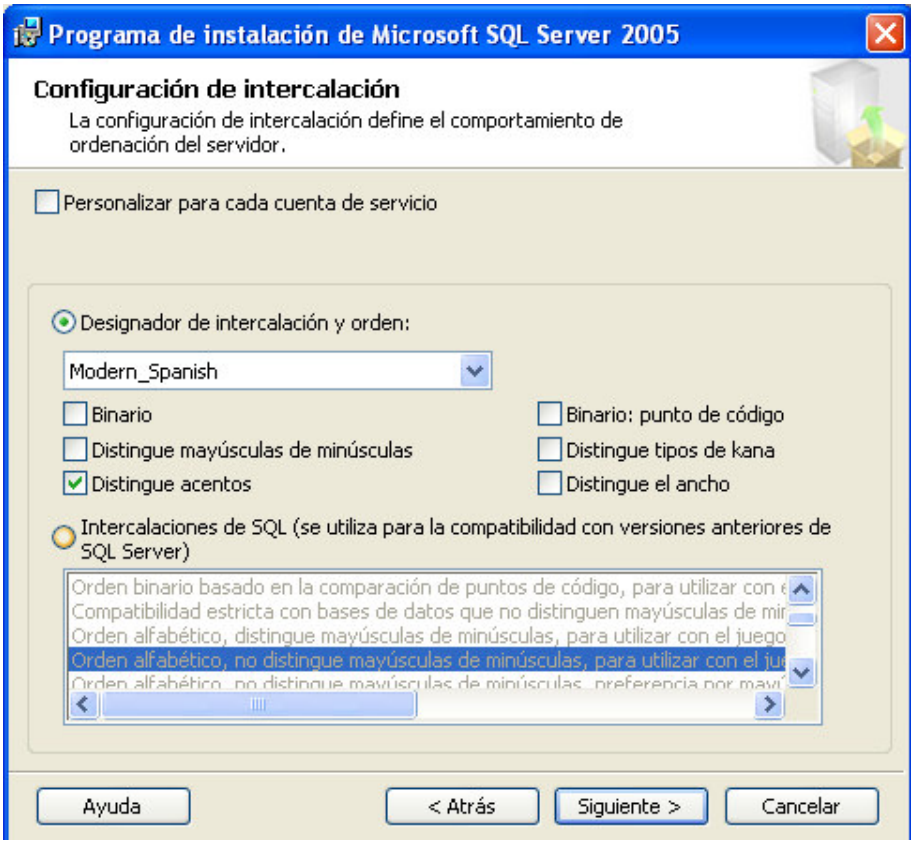


Figura 10.3.2.7 Modo de autenticación
(programa de instalación de Microsoft SQL Server 2005).

12. En la ventana Opciones de instalación del servidor de informes, debemos seleccionar la opción "Instalar la configuración predeterminada".

13. En la ventana Configuración de informes de Errores y uso, no debemos seleccionar nada porque esto es para notificar a Microsoft de los errores que de SQL Server.

14. En la siguiente ventana debemos hacer clic en "Instalar" para empezar el proceso de instalación. La página de Progreso de la instalación da seguimiento a los componentes que se están instalando y al avance de la instalación. Cuando termine la instalación, haga clic en Siguiente, y luego haga clic en finalizar para completar el proceso de instalación.

10.3.3 Base de datos de Empleados

La base de datos de empleados será creada en SQL Server 2005, y estará constituida de tablas, vistas, formularios de búsqueda, etc.

Algunos parámetros principales que se tomaran en cuenta para almacenar la base de datos podemos mencionar:

- Información Personal
- Nombres, Apellidos, Teléfonos, Dirección, etc.
- Información Sobre cargo en la empresa.
- Información sobre Funciones asignadas en la empresa
- Sueldos de los empleados
- Horas Extras de los empleados si las hubiere.
- Horas de entrada y salida a la empresa

Para esta aplicación funcionara, se procedió a hacer una instalación limpia de SQL Server 2005. Se procedió a especificar que Windows sea el cliente de la base de datos, con la finalidad de que la misma computadora este funcionando como servidor de la base de datos.

Habiendo hecho la instalación de SQL Server 2005, se procedió a realizar las tablas, haciendo las conexiones a SQL como servidor de base de datos como se muestra en la figura:

Figura 10.3.3.1 Conexion a Microsoft SQL Server 2005.

Una vez realizada al conexión se abre el programa principal de SQL Server 2005, donde se pueden crear, manejar y manipular las bases de datos.

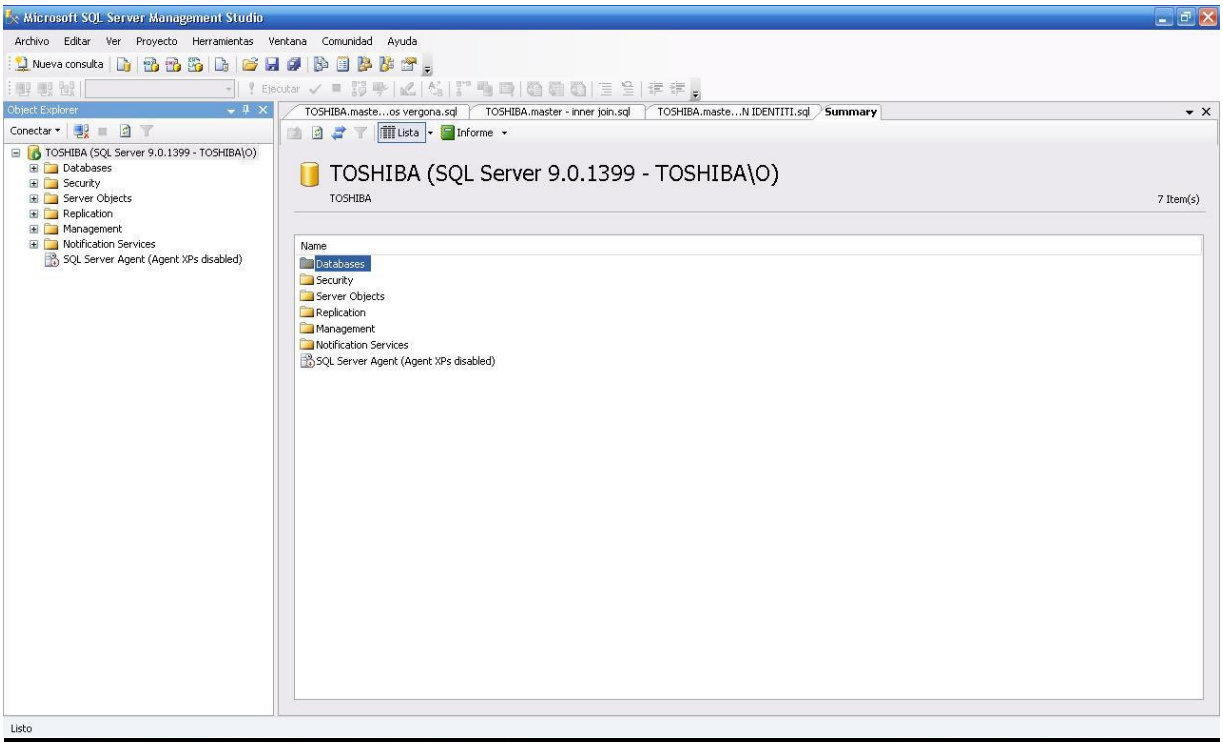


Figura 10.3.3.2 Programa Principal de Microsoft SQL Server 2005.

Luego de haber realizado la instalación de SQL Server 2005, se procede a copiar las tablas realizadas a la carpeta de proyectos del sistema, las cuales serán accesadas desde el programa de marcación de horas de trabajo realizado en Visual Basic 6.0.

10.4 PRUEBAS ESTADISTICAS DEL SISTEMA

Con el fin de registrar el desempeño del sistema se efectuaron pruebas estadísticas para encontrar errores del tipo I y tipo II (falsos positivos y falsos negativos), para nuestro caso estos errores son:

- Error tipo I o falso positivo: es el error que se da al tomar como "autorizado" un tag que es "NO autorizado".

- Error tipo II o falso negativo: es el error que se da al tomar como "No autorizado" un tag que es "autorizado".

La prueba se realizo en condiciones de operación normales (temperatura ambiente, lejos de fuentes de ruido electromagnético y con alimentación del bus USB) con 10 tags en total, 5 estaban autorizados en la memoria eeprom sistema, y 5 no estaban autorizados, se realizo un total de 1000 lecturas, 100 lecturas por cada tag, y los resultados fueron los siguientes.

		Condicion Real	
		Autorizado	NO Autorizado
Resultado Obtenido	Autorizado	100	0 Falsos Positivos
	NO Autorizado	0 Falsos Negativos	100

Tabla 10.2.1 Resultados de las Pruebas estadísticas del sistema RFID.

Los resultados nos dicen que el sistema es muy confiable, puesto que de las 1000 lecturas que se realizaron, ninguna dio resultados incorrectos, esto lo podemos atribuir a que el sistema implementado cuenta con mecanismos para evitar errores en la transmisión de la trama de datos, tanto en la etapa de Radio Frecuencia (14 de los 64 bits del tag son de bits paridad), como en la etapa USB.

11. CONCLUSIONES

- El diseño de la antena para un sistema RFID, debe cumplir con todas las recomendaciones del fabricante del circuito integrado de la etapa lectora (distancias mínimas entre elementos, cumplimiento de tolerancias de elementos, voltajes de alimentación), para lograr el máximo rango de lectura de la antena.

- El sistema desarrollado resultó ser muy confiable en las pruebas para encontrar errores del tipo "falso positivo" y "falso negativo", esta confiabilidad se debe a los sistemas de protección para errores de transmisión, que poseen el protocolo de transmisión de los tags EM4102, así como el estándar USB.

- Los microcontroladores PIC pueden ser utilizados en aplicaciones de identificación, autenticación, acceso y control, no requiriendo hardware adicional (interfases USB, interfases serie, interfases paralelas, DAC, ADC, memorias externas, etc.), solamente administración de todos sus subsistemas mediante su programa interno.

- En todo proyecto se debe tratar de usar la mejor interfaz de usuario posible, para las personas encargadas de manipular el sistema desarrollado, debido a esto, es importante mejorar la forma en la que se introducen los datos y se presentan las variables, para lograr un uso eficiente y más productivo del sistema desarrollado.

- Si se usa un compilador de alto nivel para generar el código en ensamblador, de los microcontroladores PIC, se debe elegir uno que sea 100% compatible con los dispositivos PIC seleccionados para el proyecto. En el sistema RFID

que se desarrolló, el compilador elegido presento problemas de compatibilidad con la serie de dispositivos PIC 18, específicamente el PIC18F4550, por lo que tuvo que utilizarse un PIC16F877 para realizar parte del procesamiento.

- En un sistema RFID los dispositivos Tags y el circuito lector, deben ser seleccionados para cumplir con los requisitos de la aplicación a desarrollar. Los criterios que se deben tomar en cuenta son: precio, orden mínima (muchos elementos RFID son vendidos en ordenes mínimas de cientos), memoria, alimentación (tags pasivos o activos), banda de frecuencia de trabajo (hay equipos que trabajan en frecuencias que son restringidas en nuestro país), y rango de lectura.

- En un sistema de seguridad para un automóvil, el punto de control del motor debe ser elegido teniendo en cuenta que la seguridad de los ocupantes del vehiculo, es la prioridad principal. Siendo esto así, se debe elegir un punto que solo limite el arranque del vehiculo y no su funcionamiento, porque de esta manera estamos asegurando que el vehiculo cuando esta en marcha pueda seguir así, aun cuando el sistema de seguridad desarrollado falle.

- Las librerías y funciones de manejo de protocolo USB que vienen incorporadas en el compilador CCS, son de gran ayuda a la hora de realizar software para microcontrolador que requiera de este tipo de comunicación. Ya que este protocolo es uno de los estándares de comunicación mas usados hoy en día, nuestra tarjeta se suma a este tipo de dispositivos que son bastante amigables a usuarios finales, con la principal característica de no necesita software de instalación.

- EL aislamiento eléctrico y eliminación de ruido es muy necesario para sistemas que se instalen en vehículos automotores, ya que los automóviles por sus

sistemas de carga, que son a base de dispositivos electromecánicos generan mucho ruido eléctrico en las líneas de alimentación, por lo tanto se necesitan filtros electrónicos para aislar este ruido y que este no se introduzca al sistema lector; por otra parte, los automóviles son propensos a cortocircuitos y/o sobrecargas por los mismos dispositivos electromecánicos que lo componen, por lo tanto también es imprescindible contar con dispositivos o elementos que aíslen estas fallas, y así evitar un daño a los sistemas que se instalen en el automóvil.

- La aplicación de inicio de sesión y autenticación en Windows XP es un método alternativo de inicio de sesión para usuarios, así como también provee seguridad a la hora de realizar el inicio de sesión, ya que no es necesario ingresar ninguna tecla y así evitar que alguna persona pueda copiar de alguna manera nuestra información para ingresar a la computadora, con el solo hecho de deslizar la etiqueta sobre el sistema lector, los campos se llenan automáticamente, sin dejar al descubierto nuestra información de inicio de sesión.

- Visual Basic 6.0 es una muy buena herramienta para usuarios que deseen realizar aplicaciones que interactúen con hardware, por lo tanto, el componente HidComm desarrollado por Microchip es una muy buena alternativa para poder manejar el protocolo USB que hoy en día es el protocolo de comunicación mas usado para la transferencia de datos entre computadoras personales y dispositivos de hardware externos.

12. REFERENCIAS

- Fabalabs. *Research paper, comparison of the Enterprise Functionalities of Open Source Database Management Systems* [en línea]. 26 de Abril de 2005 [ref. Marzo de 2006]. Disponible en Web: <http://www.fabalabs.org/research/papers/FabalabsResearchPaper-OSDBMS-Eval.pdf>.
- Finkenzeller, Klaus. *RFID handbook* [en línea]. Segunda Edición, Munich (Alemania): John Wiley and Sons Ltd. 2003 [ref. de Diciembre de 2006]. Disponible en Web: <http://www.rfid-handbook.de>.
- Hervás Lucas, Ramón. *Tecnología y aplicaciones de RFID* [en línea]. Año 2005 [ref. de Diciembre de 2006]. Disponible en Web: <http://mami.uclm.es/rhervas/articulos/Ramon%20Hervas%20-%20RFID%20-%20Curso%20CCMM05.pdf>.
- Livingstone, David. *La tecnología de Identificación por Radio Frecuencia (RFID)* [en línea]. 19 de Mayo de 2005 [ref. de Diciembre de 2006]. Disponible en Web: <http://www.pc-doctor.com.mx/Radio%20Formula/temas/RFID%20ETIQUETAS%20DEL%20FUTURO.htm>.
- Llamazares, Juan Carlos. *¿Cómo Funciona?. Tarjetas identificadoras sin contacto o sistemas RFID* [en línea]. [ref. De Diciembre de 2006]. Disponible en Web: <http://www.ecojoven.com/dos/03/RFID.html>.
- Ormachea Freyre, Fernando. *RFID y ePC: Aplicaciones* [en línea]. Holistica, revista de Ingeniería Industrial Volumen 1/Numero 1. Enero de 2006 [ref. de Diciembre de 2006]. Disponible en Web: <http://pergamo.pucp.edu.pe/holistica/node/8>
- Serra, Jorge. *Trasteando. Weblog sobre programación, desarrollo, redes, Linux, diseño Web, etc.* [no en linea]. 2003-2004 [ref. de Diciembre de 2006]. Disponible en Web: <http://nkieto.f2o.org/blog/archivos/000143.php>.
- Gómez, J.C.; Escudero, G. *Bus serie universal*. [en línea]. Electronica e Informatica. Noviembre 2002. [ref. de Agosto de 2007]. Disponible en Web: <http://www4.inti.gov.ar/GD/4jornadas2002/pdf/citei-046.pdf>.
- Jimenez, Carlos Fernando. *Desarrollando para el puerto USB con la familia 18F2455/2550/4455/4550 y la PICDEM FS USB DEMOSTRATION BOARD de Microchip.* [en línea]. Enero de 2005. [ref. de Agosto de 2007]. Disponible en Web: <http://usuarios.lycos.es/charlytospage/Articulos/Desarrollando_USB_18F2455_2550_4555_4550_PICDEM_.PDF>.

- Aguirre, Emanuel G.; Di Giulio, Pablo A. *Firmware para dispositivo esclavo USB de clase HID.* [en línea]. [ref. de Agosto de 2007]. Disponible en Web: <http://www.cneisi.frc.utn.edu.ar/papers/3642e51321ca65a8eeeaad6a302e.pdf>.
- Periago de la Torre, Miguel. *Compresión y comunicación de datos con un microcontrolador PIC.* [en línea]. Junio de 2007. [ref. de Agosto de 2007]. Disponible en Web: <http://sauron.etse.urv.es/public/propostes/pub/pdf/1033pub.pdf>.
- Reyes, Virgilio. *Manual de Instalacion SQL 2005.* Universidad Don Bosco. Julio 2007. [ref. Octubre 2007].
- Universidad Don Bosco. *Transact SQL.* [en línea]. Julio 2007. [ref. Octubre 2007]. Disponible en Web: <http://www.udb.edu.sv/Academia/Laboratorios/informatica/SQL/Guia_1_SQL.pdf>.
- Universidad Don Bosco. *Indices, Secuencias de comandos y Procesamiento por lotes.* [en línea]. Julio 2007. [ref. Octubre 2007]. Disponible en Web: <http://www.udb.edu.sv/Academia/Laboratorios/informatica/SQL/Guia_2_SQL.pdf>.
- Som, Guillermo. *Bases de datos ADO con Visual Basic 6.0.* [en línea]. 1996 - 2007 [ref. de Julio de 2007]. Disponible en Web: <http://www.elguille.info/vb/bases/ADO/indiceADO.htm>

13. ANEXOS

13.1 CODIGO DEL PROGRAMA EN C DE LA TARJETA LECTORA RFID

PROGRAMA PRINCIPAL DEL PIC18F4455. (RFID18f4455final3.c)

```
//* RFID READER CARD                                                            *
//* Modulo Principal                                                            *
//*                                                                             *
//* Version 1.0 (01/07/07)                                                      *
//* Hecho por Luis Mario Mercado                                                *
//*****************************************************************************/

//Fusibles del PIC (XTAL de 4MHz, USB FULL SPEED):
#include <18F4455.h>
#fuses XTPLL,NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,PLL1,CPUDIV3,VREGEN
#fuses BROWNOUT, BORV28, NOCPD, STVREN, NOIESO, NOFCMEN, NOPBADEN, CCP2B3
#use delay(clock=48000000)
#define USB_USE_FULL_SPEED  true

//Se configura el I/O en modo rapido:
#use fast_io (B)
#use fast_io (C)
#use fast_io (D)

//Definiciones dinamicas requeridas y usadas por usb.h:
#DEFINE USB_HID_DEVICE  TRUE  //Define que se incluya codigo de manejo de HID

//Activa el Endpoint 1 para hacer transferencias de entrada (al host) del tipo
//interrupcion:
#define USB_EP1_TX_ENABLE  USB_ENABLE_INTERRUPT   //turn on EP1 for IN bulk/interrupt transfers
#define USB_EP1_TX_SIZE    64  //Longitud de paquete maxima para este endpoint

//Tambien activa el Endpoint 2 para hacer transferencias del mismo tipo:
#define USB_EP1_RX_ENABLE  USB_ENABLE_INTERRUPT
#define USB_EP1_RX_SIZE    64  //Longitud de paquete maxima para este endpoint

//Definiciones dinamicas requeridas y usadas por pic18_usb.h:
//-------------------------------------------------------------
//Se define que se va a usar un pin de sensado de conexion:
#define USB_CON_SENSE_PIN PIN_E3
//  (Sensado de Conexion es como sigue:
//                              100k
//            VBUS-----+----/\/\/\/\/\----- (I/O PIN ON PIC)
//                     |
//                     +----/\/\/\/\/\-----GND
//                              100k
//        (donde VBUS es el pin 1 del conector USB)

//Con las definiciones dinamicas hechas, se incluyen las cabeceras adecuadas
//para el manejo de la interfase USB:
#include <pic18_usb.h>               //Capa de hardware USB del PIC para usb.c

#include <usb_desc_hid8-bytemio.h>   //Configuracion y descriptores USB.
#include <usb.c>             //Maneja los tokens de setup y reporta descriptores
#include <flex_lcdmio.c>  //Driver de LCD con configuracion de pines flexible

#define EEPROMCEIL 248        //Tamaño de eeprom usada. Variable que se utilizara en el programa
#define EEPROMBASE 0x780000  //direccion base de la eeprom del microcontrolador

#include <usbrfid2.c> //libreria usada para mandar datos hacia la PC
#include <eeprom.c>  //Libreria con rutinaspara manejor de datos en eeprom.
#opt 10

int1 interrupt = FALSE;  //desabilitar interrupciones y definir variable global.
static int8 code[6];     //Variable definida como una estructura que contiene 7 bytes.

//Rutina de interrupcion por cambio de estado en pines de puerto B (B7, B6, B5)
//Por cada cambio de estado recibido en PB, sera un dato recibido en PD
//estos datos y punteros son enviados por otro microcontrolador, que decodifica
//los tags y manda la informacion por PD y el puntero de datos por PB
//0000xxxx ==> Dato 1
//0010xxxx ==> Dato 2
//0100xxxx ==> Dato 3
//0110xxxx ==> Dato 4
```

```
//1000xxxx ==> Dato 5
#int_RB
void RB_isr() {
int8 puerto;            //Definicion de variables utilizadas en interrupcion
int8 old_b_state=0xe0;  //y configuracion de puerto B
set_tris_b(0xe0);

   puerto = (input_b() & 0xe0);   //Leer PB y guardar en variable los datos que interesan
   if (old_b_state != puerto) {   //Si el puerto ha cambiado de un estado anterior
                                  //pasar a la siguiente rutina
   switch (input_b() & 0xe0)      //En cada cambio de estado guardar un dato.
{
case 0x00:
   code[0]=input_d();            // Guardar Dato 1 en Variable
   break;
case 0x20:
   code[1]=input_d();            // Guardar Dato 2 en Variable
   break;
case 0x40:
   code[2]=input_d();            // Guardar Dato 3 en Variable
   break;
case 0x60:
   code[3]=input_d();            // Guardar Dato 4 en Variable
   break;
case 0x80:
   code[4]=input_d();            // Guardar Dato 5 en Variable y habilitar interrupcion
   interrupt = TRUE;
break;
default:
   break;

}

   }
      old_b_state=puerto;       //Guarda el estado anterior del PB
}

//Punto de arranque del MCU y definicion de variables usadas en el programa
void main(void) {
int up_down, ok_esc;
int32 tagNum=0;
int8  customerCode=0;
int dir=0, dir2=1;
int8 in_data[6];
int i, j, x;
int8 tmp[6];
int1 admin;
int8 admin2[5];
int8 arriba=0;
int8 dat_eeprom[5];
int8 correla=1;
int8 a,b,oo;
int8 conta, tempo;

//Inicializacion de variables y hardware.

//Variable utilizada para borrar una direccion de EEPROM especifica en rutina de ADMIN
 for(x=0;x<=4;x++) {
dat_eeprom[x]=0xff;
}

//Habilitacion de interrrupciones para el Sistema
   enable_interrupts(INT_RB);
   enable_interrupts(global);

   lcd_init();         //Inicia el modulo LCD.
   usb_init_cs();      //Inicializa el modulo USB del PIC y espera la conexion al bus.
   output_low(pin_b4); //Iniciar Salidas a cero.
   output_low(pin_b3);
   output_low(pin_b2);
   output_high(pin_b1);

    b=0;
    a=0;
    admin=0;
    interrupt=false;

//Bucle infinito del firmware:
while (TRUE) {
```

```
      usb_task(); //Mantiene el sensado de conexion llamando a otras subrutinas de la libreria.

//Comienzo del pograma principal, aca se detecta que aplicacion se esta utilizando.
         switch (input_A() & 0x30){

//*************************************************************************************************************
******************

//Seleccion de situacion; situacion CARRO.
case 0:
         admin=0;
         printf(lcd_putc,"\f   Sistema de");
         printf(lcd_putc,"\n Seguridad-Auto");
         delay_ms(25);

//Rutina para sincronizar los datos de la EEPROM via PC
if (usb_enumerated()) {
 if (usb_kbhit(1)) {
           usb_get_packet(1, in_data, 6);
//=============================================================================================
//Rutina para sincronizar datos. Envia la EEPROM a la PC.
              if (in_data[5]==50) {
                 j=0;
                 tmp[5]=0;
                 while(j<EEPROMCEIL) {          //Bucle que lee toda la EEPROM
                    for(i=0;i<=4;i++) {
                    tmp[i]=read_eeprom(i+j);   //Variable que lleva los datos de la EEPROM
                 }
                 usb_put_packet(1, tmp, 6, USB_DTS_TOGGLE); //Linea que envia cada dato a la pc de uno a
uno.
                 tmp[5]=tmp[5]+1;                           //puntero de direccion para control de datos en
PC
                 j+=8;
                 delay_ms(100);

              }
              }

//=============================================================================================

              //Rutina para sincronizar Tag maestro. Envia # de tag a la PC.
              if (in_data[5]==51) {
                 for(i=0;i<=4;i++) {
                    tmp[i]=read_eeprom(i+240);   //Variable que lleva los datos de la EEPROM
                 }
                 usb_put_packet(1, tmp, 6, USB_DTS_TOGGLE); //Linea que envia el # de tag maestro a la pc.
                 delay_ms(100);

              }
//=============================================================================================
//Rutina para sincronizar datos. Recibe los datos modificados de la EEPROM del PC al Pic
               if (in_data[5]==a){
               escribir_eeprom(in_data,b);  //llamada  a  rutina  para  escribir  datos  en  una  direccion  de
memoria
               delay_ms(10);
               a+=1;
               b+=8;
               if (a>31){  //Condicion de fin de sincronizacion.
               a=0;
               b=0;
               }
               }
               }
}
           if (input(pin_c7)==0 && oo==0){
           oo=1;
           output_high(pin_b4);
           output_low(pin_b3);
           output_low(pin_b2);
           output_high(pin_b1);
           delay_ms(100);
           output_low(pin_b4);
           for (conta=0;conta<=9;conta++){
           delay_ms(2000);
           tempo = tempo + 1;
           }
           output_high(pin_b4);
           delay_ms(100);
           output_low(pin_b4);
           }
```

```
//Rutina ocurre cuando se da una interrupcion (cuando un tag se ha detectado)
        if(interrupt){
        tagNum = make32(code[1],code[2],code[3],code[4]);
        customerCode = code[0];
        enviar_usb(code); //Llamada a rutina para enviar datos a la PC

 //Llamada a rutina para comprobar si un tag se encuentra en memoria
 //si el tag esta en memoria dara acceso y habilitara encendido, sino esta en memoria
 //Denegara acceso y desabilitara o mantendra desabilitados los sistemas de acceso y encendido
         if(check_id(code)==1)
         {
           printf(lcd_putc,"\f TAG: ACEPTADO");
           printf(lcd_putc,"\nNo: %x%lx", customerCode, tagNum);

           output_high(pin_b3); // B3 habilita el encendido del carro.
           output_high(pin_b4); // B4 es para activar solenoide de seguros
           output_high(pin_b2);
           output_low(pin_b1);
           delay_ms(100);      // un pulso de 100m segundo.
           output_low(pin_b4);  // termina pulso de 100m seg para solenoides.


           for (conta=0;conta<=9;conta++){
           delay_ms(2000);
           tempo = tempo + 1;
           }

           if (input(pin_c7)==0){
           output_low(pin_b3);
           output_high(pin_b4);
           output_low(pin_b2);
           output_high(pin_b1);
           delay_ms(100);      // un pulso de 100m segundo.
           output_low(pin_b4);  // termina pulso de 100m seg para solenoides.

           }

           else{
           oo=0;
           output_high(pin_b3);
           output_high(pin_b4);
           output_high(pin_b2);
           output_low(pin_b1);
           delay_ms(100);
           output_low(pin_b4);
           }
           }

         else
         {
           printf(lcd_putc,"\f TAG: DENEGADO");
           printf(lcd_putc,"\nNo: %x%lx", customerCode, tagNum);
           delay_ms(1000);
         }

           interrupt=false;   //Desabilitar interrupcion para esperar otra.
        }
        break;

//*************************************************************************************************************
******************

//Seleccion de situacion; situacion PUERTA-CASA. Cerradura electronica.
   case 16:
        admin=0;
        output_low(pin_b2);
        output_high(pin_b1);
        printf(lcd_putc,"\f   Cerradura");
        printf(lcd_putc,"\n  Electronica ");
        delay_ms(25);

//Rutina para sincronizar los datos de la EEPROM via PC
if (usb_enumerated()) {
 if (usb_kbhit(1)) {
//=============================================================================================
//Rutina para sincronizar datos. Envia la EEPROM a la PC.
           usb_get_packet(1, in_data, 6);
              if (in_data[5]==50) {
                 j=0;
                 tmp[5]=0;
```

```
                while(j<EEPROMCEIL) {         //Bucle que lee toda la EEPROM
                   for(i=0;i<=4;i++) {
                   tmp[i]=read_eeprom(i+j);   //Variable que lleva los datos de la EEPROM
                }
                usb_put_packet(1, tmp, 6, USB_DTS_TOGGLE); //Linea que envia cada dato a la pc de uno a
uno.
                tmp[5]=tmp[5]+1;                           //puntero de direccion para control de datos en
PC
                j+=8;
                delay_ms(100);
             }
             }
//=============================================================================================

             //Rutina para sincronizar Tag maestro. Envia # de tag a la PC.
             if (in_data[5]==51) {
                for(i=0;i<=4;i++) {
                   tmp[i]=read_eeprom(i+240);   //Variable que lleva los datos de la EEPROM
                }
                usb_put_packet(1, tmp, 6, USB_DTS_TOGGLE); //Linea que envia el # de tag maestro a la pc.
                delay_ms(100);

             }
//=============================================================================================
//Rutina para sincronizar datos. Recibe los datos modificados de la EEPROM del PC al Pic
              if (in_data[5]==a){
              escribir_eeprom(in_data,b);  //llamada  a  rutina  para  escribir  datos  en  una  direccion  de
memoria
              delay_ms(10);
              a+=1;
              b+=8;
              if (a>31){                  //Condicion de fin de sincronizacion.
              a=0;
               b=0;
              }
              }
 }
}

//Rutina ocurre cuando se da una interrupcion (cuando un tag se ha detectado)
        if(interrupt){
        tagNum = make32(code[1],code[2],code[3],code[4]);
        customerCode = code[0];
        enviar_usb(code);   //Llamada a rutina para enviar datos a la PC

 //Llamada a rutina para comprobar si un tag se encuentra en memoria
 //si el tag esta en memoria dara acceso y habilitara cerradura, sino esta en memoria
 //Denegara acceso y desabilitara o mantendra desabilitados los sistemas de acceso (cerradura)
         if(check_id(code)==1)
         {
           printf(lcd_putc,"\f TAG: ACEPTADO");
           printf(lcd_putc,"\nNo: %x%lx", customerCode, tagNum);
           output_high(pin_b4);
           output_high(pin_b2);
           output_low(pin_b1);
           delay_ms(1000);
           output_low(pin_b4);
           output_low(pin_b2);
           output_high(pin_b1);

         }

         else
         {
           printf(lcd_putc,"\f TAG: DENEGADO");
           printf(lcd_putc,"\nNo: %x%lx", customerCode, tagNum);
           output_low(pin_b2);
           output_high(pin_b1);
           delay_ms(1000);
         }

           interrupt=false;   //Desabilitar interrupcion para esperar otra.
        }
        break;

//*********************************************************************************************************
*********************
//Seleccion de situacion; situacion MARCACION TARJETA-USB.
//posibilidad de hacer un chesum de datos
//posibilidad de recibir la hora de la pc y mandarla a la LCD
```

```
//Posibilidad de recibir alguna informacion del empleado de la pc a la LCD (nombre, apellido)
   case 32:
         admin=0;
         printf(lcd_putc,"\fSistema Marcador");
         printf(lcd_putc,"\nHoras de Trabajo");
         delay_ms(25);

//Rutina ocurre cuando se da una interrupcion (cuando un tag se ha detectado)
      if(interrupt){
        tagNum = make32(code[1],code[2],code[3],code[4]);
        customerCode = code[0];
        printf(lcd_putc,"\fNo: %x%lx", customerCode, tagNum);
        enviar_usb(code);      //Llamada a rutina para enviar datos a la PC
        delay_ms(1000);
        interrupt=false;      //Desabilitar interrupcion para esperar otra.
        }

        break;

//*************************************************************************************************************
//Seleccion de situacion; situacion AGREGAR TAG-BORRAR TAG. Modo ADMIN
//En esta rutina se utilizaran pulsadores para manipulacion manual del usuario
//seran 4 pulsadores: UP (arriba), DOWN (abajo), OK (aceptar), ESC (Salir)
//Los pulsadores se utilizaran segun el caso y segun instrucciones en la LCD
   case 48:      //Primer caso, si se desea agregar un tag a la memoria o borrar un tag de la memoria

   while(admin==0){
        printf(lcd_putc,"\fDeslizar TAG de");
        printf(lcd_putc,"\nAdministracion");
        delay_ms(25);

   for(i=0;i<=4;i++) {
   admin2[i]=read_eeprom(i+240);
   }

   if (interrupt)
   {
    if   (code[0]==admin2[0]   &&   code[1]==admin2[1]&&   code[2]==admin2[2]   &&   code[3]==admin2[3]   &&
code[4]==admin2[4])
    {
   admin=1;
   }
   else {
        printf(lcd_putc,"\fEl TAG deslizado");
        printf(lcd_putc,"\n  no es valido");
        delay_ms(1000);
   }
    interrupt=false;
   }

}
         for(x=0;x<=4;x++) {
         code[x]=0xff;
         }
        printf(lcd_putc,"\fAgregar: up");
        printf(lcd_putc,"\nBorrar: down");
        delay_ms(25);

        up_down=(input_c() & 0x03);      //Variable para la seleccion de agregar o borrar tag
                                         //pulsador UP o DOWN
//=============================================================================
//Seleccion de rutina, ya sea para agregar, borrar tag o EEPROM
switch (up_down){
//rutina de Agregar tag. Se da cuando se presiona el pulsador UP
            case 1:
            printf(lcd_putc,"\fDeslizar TAG");
            delay_ms(25);

            do{   //Bucle que se mantiene hasta que se presiona un pulsador (OK, ESC)
//Rutina ocurre cuando se da una interrupcion (cuando un tag se ha detectado)
//en esta rutina se espera a que se de la presencia de un tag para poderlo agregar a la EEPROM
//Cuando el tag esta presente, se pide confirmacion de guardar o desechar.
            if(interrupt){
            tagNum = make32(code[1],code[2],code[3],code[4]);
            customerCode = code[0];
            printf(lcd_putc,"\fNo: %x%lx", customerCode, tagNum);
            printf(lcd_putc,"\nAgregar: OK/ESC");
            delay_ms(25);
            interrupt=false;      //Desabilitar interrupcion para esperar otra.
            }
```

```
//rutina de confirmacion de guardado o desechado de Tag presente
//Se hace uso de los otros dos pulsadores en esta rutina
//OK para guardar tag en memoria o ESC para no guardar y volver al inicio de la aplicacion ADMIN

            ok_esc=(input_c() & 0x44);   //Variable para la seleccion de OK, ESC
            } while(ok_esc==0);          //Bucle que se mantiene hasta que se presiona un pulsador (OK, ESC)

//Seleccion de rutina ya sea para grabar un tag o no guardarlo.
            switch (ok_esc){
          //-------------------------------------------------------
//Rutina para guardar un tag en memoria. Cuando se presiono el pulsador OK
               case 4:

               if (code[0]==0xFF && code[1]==0xFF && code[2]==0xFF && code[3]==0xFF && code[4]==0xFF){
               printf(lcd_putc,"\f No se detecto");    //si el tag ya esta guardado se entra a esta rutina
sino
               printf(lcd_putc,"\n  ningun TAG");   //se procede a grabarlo en memoria
               delay_ms(1000);
               break;
               }

               if (check_id(code)==1){      //llamada a rutina para verificar si un tag ya esta en memoria
               printf(lcd_putc,"\f  El TAG ya ha");    //si el tag ya esta guardado se entra a esta rutina
sino
               printf(lcd_putc,"\n sido Agregado");   //se procede a grabarlo en memoria
               delay_ms(1000);
               break;
               }

 j=0;
  while(j<EEPROMCEIL) {          //Toda la eeprom
     for(i=0;i<=4;i++) {
       tmp[i]=read_eeprom(i+j);
     }
     if((tmp[0] | tmp[1] | tmp[2] | tmp[3] | tmp[4])!=0)   // isn't all bits == 0?
     {
        if ((tmp[0]==0xff)&&(tmp[1]==0xff)&&(tmp[2]==0xff)&&(tmp[3]==0xff)&&(tmp[4]==0xff))
        dir=j;
        j+=8;
        dir2=j/8;
               escribir_eeprom(code,dir);      //Si el tag no estaba en memoria se guarda en una localidad
               printf(lcd_putc,"\fDir. %u escrita", dir2);
               delay_ms(1000);
//               dir=dir+8;
//               dir2++;
               break;

     }
   }

         //-------------------------------------------------------
         //Rutina para no guardar un tag en memoria. Cuando se presiono el pulsador ESC
               case 64:
               break;
         //-------------------------------------------------------
               default:
               break;
               }
            break;

 //=============================================================================
//Rutina para borrar tago eeprom completa. Se da cuando se presiona el pulsador DOWN
//se hace una comprobacion de que se desea hacer
//Se puede borrar un tag especifico al presionar el pulsador OK, o la EEPROM completa al presionar ESC
            case 2:
            printf(lcd_putc,"\fDireccion: OK");
            printf(lcd_putc,"\nCompleta: ESC");
            delay_ms(25);

            do{   //Bucle que se mantiene hasta que se presiona un pulsador (OK, ESC)
            ok_esc=(input_c() & 0x44);   //Variable para la seleccion de OK, ESC
            } while(ok_esc==0);       //Bucle que se mantiene hasta que se presiona un pulsador (OK, ESC)

//Seleccion de rutina ya sea para Borrar un tag especifico o la eeprom completa
          switch (ok_esc){
//-----------------------------------------------------------------------------
//Rutina para borrar un tag especifico.
//En esta rutina se selecciona el tag desplazandose por toda la memoria con las teclas UP, DOWN
//cuando se selecciona el tag a borrar, se puede borrar al presionar la tecla de confirmacion OK
//o salir de la rutina con la tecla ESC.
```

```
            case 4:
           delay_ms(500);

        do{      //Bucle que se mantiene hasta que se presiona un pulsador (OK, ESC)
           ok_esc=(input_c() & 0x44);    //Variable para la seleccion de OK, ESC

            up_down=(input_c() & 0x03);   //Variable para la seleccion UP, DOWN desplazarse en la memoria

//.....................................................................
//Rutinas para desplazarse por toda la memoria hacia arriba o abajo, dezplegando los tags
//que se encuentran en las localidades que se van dezplazando.
//Se puede dar una vuelta completa a la memoria.

            if(up_down==1){   //Rutina para dezplazarse en memoria hacia arriba
            delay_ms(300);
            arriba-=8;
            correla--;
            if (correla<1){
            arriba=232;
            correla=30;
            }
            }

            if(up_down==2){   //Rutina para dezplasarse en memoria haci abajo
            delay_ms(300);
            arriba+=8;
            correla++;
            if(correla>30){
            arriba=0;
            correla=1;
            }
            }
//.....................................................................
 // lectura de las localidades de memoria segun dezplazamiento
 //si la localidad esta vacia, se muestra un mensaje de localidad vacia.
 //si no esta vacia, se muestra el No. de Tag.
              for(i=0;i<=4;i++) {
              tmp[i]=read_eeprom(i+arriba);
              }

             //Muestra en LCD si la localidad esta vacia
             if(tmp[0]==0xFF && tmp[1]==0xFF && tmp[2]==0xFF && tmp[3]==0xFF && tmp[4]==0xFF){
              printf(lcd_putc,"\fDir. %u Vacia", correla);
              delay_ms(25);
             }
              else{

              //Muestra en LCD el No. de tag que se encuenta en la localidad seleccionada.
              printf(lcd_putc,"\fNo.%u %x%x%x%x%x",correla, tmp[0],tmp[1],tmp[2],tmp[3],tmp[4]);
              printf(lcd_putc,"\n    OK / ESC");
              delay_ms(25);
              }

        } while(ok_esc==0);   //Bucle que se mantiene hasta que se presiona un pulsador (OK, ESC)

          if (ok_esc==4){   //Si se presiono Ok, se borra el tag seleccionado.
          escribir_eeprom(dat_eeprom, arriba);   //Se borra el tag, escribiendo en memoria la serie
                                                 // FFFFFFFFFF
          printf(lcd_putc,"\fDir. %u Borrada", correla);
          delay_ms(1000);
          }
          else{
          break;
          }
           break;
        //-------------------------------------------------------------------------------

        //Rutina para borrar la EEPROM completa, al presionar la tecla ESC
        //la eeprom se borra grabandole en sus localidades FFFFFFFFFF
              case 64:
               borrar_eeprom();   //llamada a funcion de borrado de eeprom completa
               printf(lcd_putc,"\fEEPROM BORRADA...");
               delay_ms(1000);
                dir=0;
               dir2=1;
               break;
               }
              break;
           default:
           break;
```

```
         }
        break;

        default:
        break;

//=============================================================================
   }
}
}
```

DIVER PARA EL MANEJO DE PANTALLAS LCD 16X2. (flex_lcdmio.c)

```
//Driver para manejar pantallas LCD 16x2
//Descargado de http://www.ccsinfo.com/forum/viewtopic.php?t=24661&highlight=flexlcd
//Modificado para su funcionamiento en el sistema RFID reader card
//Modificado por Luis Mario Mercado

// flex_lcd.c
// These pins are for the Microchip PicDem2-Plus board,
// which is what I used to test the driver.  Change these
// pins to fit your own board.

#define LCD_DB4   PIN_A0
#define LCD_DB5   PIN_A1
#define LCD_DB6   PIN_A2
#define LCD_DB7   PIN_A3

#define LCD_E     PIN_E0
#define LCD_RS    PIN_E1
#define LCD_RW    PIN_E2

// If you only want a 6-pin interface to your LCD, then
// connect the R/W pin on the LCD to ground, and comment
// out the following line.

#define USE_LCD_RW   1

//=======================================

#define lcd_type 2        // 0=5x7, 1=5x10, 2=2 lines
#define lcd_line_two 0x40 // LCD RAM address for the 2nd line


int8 const LCD_INIT_STRING[4] =
{
 0x20 | (lcd_type << 2), // Func set: 4-bit, 2 lines, 5x8 dots
 0xc,                    // Display on
 1,                      // Clear display
 6                       // Increment cursor
 };


//-------------------------------------
void lcd_send_nibble(int8 nibble)
{
// Note:  !! converts an integer expression
// to a boolean (1 or 0).
 output_bit(LCD_DB4, !!(nibble & 1));
 output_bit(LCD_DB5, !!(nibble & 2));
 output_bit(LCD_DB6, !!(nibble & 4));
 output_bit(LCD_DB7, !!(nibble & 8));

 delay_cycles(1);
 output_high(LCD_E);
 delay_us(2);
 output_low(LCD_E);
}

//-------------------------------------
// This sub-routine is only called by lcd_read_byte().
// It's not a stand-alone routine.  For example, the
// R/W signal is set high by lcd_read_byte() before
// this routine is called.

#ifdef USE_LCD_RW
int8 lcd_read_nibble(void)
{
int8 retval;
```

```
// Create bit variables so that we can easily set
// individual bits in the retval variable.
#bit retval_0 = retval.0
#bit retval_1 = retval.1
#bit retval_2 = retval.2
#bit retval_3 = retval.3

retval = 0;

output_high(LCD_E);
delay_cycles(1);

retval_0 = input(LCD_DB4);
retval_1 = input(LCD_DB5);
retval_2 = input(LCD_DB6);
retval_3 = input(LCD_DB7);

output_low(LCD_E);

return(retval);
}
#endif

//-----------------------------------
// Read a byte from the LCD and return it.

#ifdef USE_LCD_RW
int8 lcd_read_byte(void)
{
int8 low;
int8 high;

output_high(LCD_RW);
delay_cycles(1);

high = lcd_read_nibble();

low = lcd_read_nibble();

return( (high<<4) | low);
}
#endif

//-----------------------------------------
// Send a byte to the LCD.
void lcd_send_byte(int8 address, int8 n)
{
output_low(LCD_RS);

#ifdef USE_LCD_RW
while(bit_test(lcd_read_byte(),7)) ;
#else
delay_us(60);
#endif

if(address)
   output_high(LCD_RS);
else
   output_low(LCD_RS);

 delay_cycles(1);

#ifdef USE_LCD_RW
output_low(LCD_RW);
delay_cycles(1);
#endif

output_low(LCD_E);

lcd_send_nibble(n >> 4);
lcd_send_nibble(n & 0xf);
}

//----------------------------
void lcd_init(void)
{
int8 i;

output_low(LCD_RS);
```

```
#ifdef USE_LCD_RW
output_low(LCD_RW);
#endif

output_low(LCD_E);

delay_ms(15);

for(i=0 ;i < 3; i++)
   {
    lcd_send_nibble(0x03);
    delay_ms(5);
   }

lcd_send_nibble(0x02);

for(i=0; i < sizeof(LCD_INIT_STRING); i++)
   {
    lcd_send_byte(0, LCD_INIT_STRING[i]);

    // If the R/W signal is not used, then
    // the busy bit can't be polled.  One of
    // the init commands takes longer than
    // the hard-coded delay of 60 us, so in
    // that case, lets just do a 5 ms delay
    // after all four of them.
    #ifndef USE_LCD_RW
    delay_ms(5);
    #endif
   }

}

//-----------------------------

void lcd_gotoxy(int8 x, int8 y)
{
int8 address;

if(y != 1)
   address = lcd_line_two;
else
   address=0;

address += x-1;
lcd_send_byte(0, 0x80 | address);
}

//-----------------------------
void lcd_putc(char c)
{
 switch(c)
   {
    case '\f':
      lcd_send_byte(0,1);
      delay_ms(2);
      break;

    case '\n':
       lcd_gotoxy(1,2);
       break;

    case '\b':
       lcd_send_byte(0,0x10);
       break;

    default:
       lcd_send_byte(1,c);
       break;
   }
}

//-----------------------------
#ifdef USE_LCD_RW
char lcd_getc(int8 x, int8 y)
{
char value;

lcd_gotoxy(x,y);
```

```
// Wait until busy flag is low.
while(bit_test(lcd_read_byte(),7));

output_high(LCD_RS);
value = lcd_read_byte();
output_low(lcd_RS);

return(value);
}
#endif
```

LIBRERÍA DE FUNCION USADA PARA ENVIAR DATOS A PC. (usbrfid2.c)

```
//* RFID READER CARD                                                            *
//* Rutina Envio datos USB-PC                                                   *
//*                                                                             *
//* Version 1.0 (01/07/07)                                                      *
//* Hecho por Luis Mario Mercado                                                *
//*****************************************************************************/

//Rutina que envia datos contenidos en una variable hacia la PC via USB
void enviar_usb(int8* data){
int16 send_timer=0;


if (usb_enumerated()) {
         if (!send_timer) {
            send_timer=250;
            usb_put_packet(1, data, 6, USB_DTS_TOGGLE);
         }
         send_timer--;
         delay_ms(1);
}
}
```

LIBRERÍA DE FUNCION USADA PARA MANEJAR DATOS EN EEPROM. (eeprom.c)

```
//Funcion para escribir codigo en una localidad de memoria eeprom

void escribir_eeprom(int8* data, int8 direccion) {
  int i;
     for(i=0;i<=4;i++)
     {
       write_eeprom((i+direccion),data[i]);
       delay_ms(10);
     }
 }


// Rutina para chequear Tag con dato guardado en eeprom
int check_id(int8* data) {
  int8 tmp[5];
  int i,j;

  j=0;
  while(j<EEPROMCEIL) {          //till the ceil of EEPROM
     for(i=0;i<=4;i++) {
       tmp[i]=read_eeprom(i+j);
     }
     if((tmp[0] | tmp[1] | tmp[2] | tmp[3] | tmp[4])!=0)   // isn't all bits == 0?
     {
        if ((tmp[0]==data[0])&&(tmp[1]==data[1])&&(tmp[2]==data[2])&&(tmp[3]==data[3])&&(tmp[4]==data[4]))
//check for matching
           return 1;
        j+=8;
     } else
         return 0;
   }
   return 0;

}


//Rutina para borrar la eeprom completa (no la borra, pone FF en todas las localidades).
void borrar_eeprom() {
  int i,j;
```

```
   for(j=0;j<240;j+=8)
   {
     for(i=0;i<=4;i++)
     {
       write_eeprom((i+j),0xFF);
     }
  }
 }
```

PROGRAMA PRINCIPAL DEL PIC16F877A. (RFID16f877final1.c)

```
//* RFID READER CARD                                                            *
//* Modulo de envio y decodificacion de etiquetas.                              *
//*                                                                             *
//* Version 1.0 (01/07/07)                                                      *
//* Hecho por Luis Mario Mercado                                                *
//*****************************************************************************/

//Fusibles del PIC (XTAL de 20MHz):
#include <16f877a.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP,NOPUT,NOBROWNOUT,NOCPD,NOWRT,NODEBUG
#use delay(clock=20000000)

#include <em4095mio16f877.c>   //Driver para manejo de IC EM4095 RFID Basestation
#include <em4102.c>      //Driver para decodificacion de etiquetas con standard EM4102

//Punto de arranque del MCU y definicion de variables usadas en el programa
void main(void) {
   int8 code[5];
   int8 i, j, k;
   int8 dato_org;

//Inicializacion de variables y hardware.
   set_tris_d(0b00000000);
   delay_ms(10);
   output_d(0b00000111);

   rf_init();      //Inicializa el EM4095 en Modo Sleep segun especificaciones del fabricante
   rf_powerUp();   //Activa la antena del EM4095, segun especificaciones del fabricante.

//Bucle infinito del firmware:
while(TRUE)
 {
   if(read_4102(code))  //Lee el Tag y lo decodifica.
   {
//-------------------------------------------------------------------------------
//Rutina para acomodar datos decodificados provenientes de la libreria EM4102

      for (i=0; i<5; i++) {
         dato_org = code[i];
         code[i] = 0;
         for (j=0; j<8; j++) {
            code[i] |= ((dato_org>>(j)) & 0x01);
            if(j<=6) {
             code[i] <<= 1;
            }

         }
      }
//-------------------------------------------------------------------------------
//Rutina para enviar puntero y datos por puerto al otro microcontrolador
for (k=0;k<5;k++){
output_d(k);
output_b(code[k]);
delay_ms(10);
}
   }
 }
}
```

Ciertas librerías a las que se hace llamada desde los programas principales de los microcontroladores, son estándares y sirven para el control de ciertos módulos internos de los microcontroladores, por lo tanto estos archivos de librería vienen grabados en las carpetas del compilador PCWH V4.013; las librerías y/o archivos usados son:

- **ARCHIVO DE CABECERA PARA LOS DISPOSITIVOS PIC16F877A. (PIC16F877A.h)**
  Archivo procedente de la carpeta de instalación "devices" del compilador PCWH V4.013

- **Driver para el manejo del IC EM4095 RFID basestation. (em4095mio16f877.c)**
  Archivo procedente de la carpeta de instalación "drivers" del compilador PCWH V4.013; este archivo fue modificado en cierta medida para cumplir los requisitos que se deseaban para el buen funcionamiento del sistema lector.

- **ARCHIVO MANEJADOR DE TOKENS Y DESCRIPTORES. (USB.c)**
  Archivo procedente de la carpeta de instalación "devices" del compilador PCWH V4.013

- **ARCHIVO DE CABECERA PARA LOS DISPOSITIVOS PIC18F4455. (PIC18F4455.h)**
  Archivo procedente de la carpeta de instalación "devices" del compilador PCWH V4.013

- **Archivo de cabecera para el manejo de la capa de hardware USB del PIC18F4455. (PIC18_USB.h)**
  Archivo procedente de la carpeta de instalación "devices" del compilador PCWH V4.013

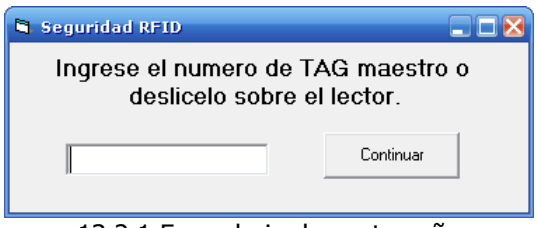
- **Driver para el manejo y decodificación de las etiquetas de RFID EM4102. (em4102.c)**
  Archivo procedente de la carpeta de instalación "drivers" del compilador PCWH V4.013

- **Driver para el manejo de descriptores USB. (usb_desc_hid8-bytemio.h)**
  Archivo procedente de la carpeta de instalación "drivers" del compilador PCWH V4.013; este archivo fue modificado en cierta medida para cumplir los requisitos que se deseaban para el buen funcionamiento del sistema lector.

13.2 PROGRAMAS EN VISUAL BASIC

13.2.1 Programa de sincronización de EEPROM. PIC - PC, PC - PIC

Formulario de Contraseña



13.2.1 Formulario de contraseña

```
'Definicion de variables usadas en el programa
Dim Buffer() As Byte
Dim BufferSize As Long
Dim usb As String
Dim texto As String
Dim A As Integer

'Rutina que se ejecuta al darse el evento clic en el boton de continuar
'se efectua una comparacion entre la variable que guarda el numero de tag
'maestro obtenido de la ultima localidad de memoria y el numero en la caja de
'texto. si los numeros son iguales, accede al programa maestro, sino
'presenta un mensaje de advertencia.
Private Sub cmdpass_Click()
    texto = txtpass.Text
    A = A + 1
    'Rutina que se ejecuta cuando los numeros coinciden
```

```
    'Habilita la forma principal del programa.
    If (usb = texto) Then
    Timer1.Enabled = False
    frmRFID.Timer1.Enabled = True
    frmRFID.HIDComm1.TimeOut = 10
    frmRFID.lblTag.Caption = "Deslizar un TAG"
    frmRFID.Visible = True
    frmConexion.Visible = False
    frmPass.Visible = False
    txtpass.Text = ""
    'Rutina que se ejecuta cuando no hay ningun tag maestro agregado
    'habilita la forma principal haciendo un recordatorio
    'que todavia no hay ningun tag maestro agregado.
    'el tag maestro se guarda en la ultima localidad de memoria.
    ElseIf (usb = "FFFFFFFFFF") Then
    Timer1.Enabled = False
    frmRFID.Timer1.Enabled = True
    frmRFID.HIDComm1.TimeOut = 10

    MsgBox "Todavia no ha agregado ningun TAG maestro. Agreguelo en este momento.", _
    vbInformation, "Importante"

    frmRFID.lblTag.Caption = "Deslizar un TAG"
    frmRFID.Visible = True
    frmConexion.Visible = False
    frmPass.Visible = False
    txtpass.Text = ""
    'Rutina que se ejecuta cuando no hay match entre numero y tag maestro
    Else
    MsgBox "La contraseña es incorrecta o el TAG no es valido", vbCritical, "ERROR"
    txtpass.Text = ""
    End If
End Sub

'Rutina que se ejecuta al darse el evento de que se ha cerrado la forma.
'desconecta usb y termina el programa.
Private Sub Form_Terminate()
frmRFID.HIDComm1.Uninit
frmRFID.HIDComm1.Disconnect
frmRFID.Timer1.Enabled = False
Timer1.Enabled = False
End
End Sub

'Rutina que se ejecuta al darse el evento de carga de la forma.
'Aca se detecta si la tarjeta no esta conectada cuando el programa se abre.
'tambien recibe el numero de tag maestro para compararlo con los tags
'deslizados, y asi poder entrar al programa maestro
Private Sub Form_Load()
'Rutina que detecta si la tarjeta esta conectada cuando se abre el programa
'Si la tarjeta no esta conectada, el programa muestra un aviso y se cierra.
If Not frmRFID.HIDComm1.Connected Then
MsgBox "No se ha detectado ningun lector RFID conectado al equipo", vbCritical, "Error"
frmRFID.HIDComm1.Uninit
frmRFID.HIDComm1.Disconnect
frmRFID.Timer1.Enabled = False
Timer1.Enabled = False
End
End If

'Rutina que recibe el numero de tag maestro localizado en la ultima direccion
'de memoria en el chip, lo guarda en una variable, con la finalidad de compararlo
'para poder ingresar al tag maestro.
frmRFID.HIDComm1.TimeOut = 1000
frmRFID.HIDComm1.Connect
ReDim Buffer(48)
    Buffer(0) = 0
    Buffer(1) = 0
    Buffer(2) = 0
    Buffer(3) = 0
    Buffer(4) = 0
    Buffer(5) = 51
    BufferSize = 6
    frmRFID.HIDComm1.WriteTo Buffer, BufferSize
    Buffer(5) = 0
    Buffer = frmRFID.HIDComm1.ReadFrom(BufferSize)

    usb = Format(Hex(Buffer(0)), "00") & Format(Hex(Buffer(1)), "00") & _
    Format(Hex(Buffer(2)), "00") & Format(Hex(Buffer(3)), "00") & _
    Format(Hex(Buffer(4)), "00")
```

```
    A = A + 1
    Timer1.Enabled = True
    frmRFID.Timer1.Enabled = False
End Sub

'Rutina que se ejecuta al darse el evento timer.
'Detecta si la tarjeta se ha desconectado y tambien detecta
'los tags que se han deslizado por la tarjeta para introducir
'el tag maestro.
Private Sub Timer1_Timer()
'Rutina que detecta si la tarjeta se ha desconectado.
If Not frmRFID.HIDComm1.Connected Then
        frmRFID.Visible = False
        frmConexion.Visible = True
        frmPass.Visible = False
        frmConexion.Label1.Caption = "Por favor conecte la tarjeta"
        frmRFID.HIDComm1.Connect
        Exit Sub
End If

'rutina que envia los tags deslizados al campo de contasena.
'esperando el tag maestro.
frmRFID.HIDComm1.TimeOut = 10
 BufferSize = 6
Buffer = frmRFID.HIDComm1.ReadFrom(BufferSize)
If BufferSize < 6 Then
Exit Sub
End If

txtpass.Text = Format(Hex(Buffer(0)), "00") & Format(Hex(Buffer(1)), "00") & _
Format(Hex(Buffer(2)), "00") & Format(Hex(Buffer(3)), "00") & _
Format(Hex(Buffer(4)), "00")


texto = txtpass.Text
    A = A + 1
    'Rutina que se ejecuta cuando los numeros coinciden
    'Habilita la forma principal del programa.
    If (usb = texto) Then

    'Rutina que presenta en pantalla un message box que desaparece despues de
    '1 segundo y automaticamente carga la forma principal.
    SetTimer hWnd, NV_CLOSEMSGBOX, 1000&, AddressOf TimerProc
    Call MessageBox(hWnd, "Etiqueta de Administracion Correcta", _
    "AutoCerrado", MB_ICONQUESTION Or MB_TASKMODAL)

    'Continuacion de la rutina que habilita la forma principal
    Timer1.Enabled = False
    frmRFID.Timer1.Enabled = True
    frmRFID.HIDComm1.TimeOut = 10
    frmRFID.lblTag.Caption = "Deslizar un TAG"
    frmRFID.Visible = True
    frmConexion.Visible = False
    frmPass.Visible = False
    txtpass.Text = ""
    'Rutina que se ejecuta cuando no hay ningun tag maestro agregado
    'habilita la forma principal haciendo un recordatorio
    'que todavia no hay ningun tag maestro agregado.
    'el tag maestro se guarda en la ultima localidad de memoria.
    ElseIf (usb = "FFFFFFFFFF") Then
    Timer1.Enabled = False
    frmRFID.Timer1.Enabled = True
    frmRFID.HIDComm1.TimeOut = 10

    MsgBox "Todavia no ha agregado ningun TAG maestro. Agreguelo en este momento.", _
    vbInformation, "Importante"

    frmRFID.lblTag.Caption = "Deslizar un TAG"
    frmRFID.Visible = True
    frmConexion.Visible = False
    frmPass.Visible = False
    txtpass.Text = ""
    'Rutina que se ejecuta cuando no hay match entre numero y tag maestro
    Else
    MsgBox "La contraseña es incorrecta o el TAG no es valido", vbCritical, "ERROR"
    txtpass.Text = ""
    End If
End Sub
```

```
'Rutina que se ejecuta al darse el evento de cerrado de la forma.
'Se ejecuta cuando se cierra la forma. cuando se hace clic en el boton
'cerrar X. Se desconecta USB y termina el programa
Private Sub Form_Unload(Cancel As Integer)
frmRFID.HIDComm1.Uninit
frmRFID.HIDComm1.Disconnect
frmRFID.Timer1.Enabled = False
Timer1.Enabled = False
  End
End Sub

'Funcion que detecta cuando se presiona la tecla ENTER
'con la finalidad de poder entrar al programa presionando
'enter despues de introducir la clave.
Private Sub Form_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
SendKeys "{tab}"
SendKeys "{enter}"
KeyAscii = 0
End If
End Sub
```

Formulario Principal (Programa de Sincronización)

13.2.2 Formulario Principal

```
'Programa sincronizador de EEPROM por USB
'Tesis Lector RFID
'Hecho por Luis Mario Mercado
'8-Agosto-2007

'Definicion de Variables usadas.
Dim Buffer() As Byte
Dim BufferSize As Long
Dim numero As String
Dim numero2 As String
Dim uno As Boolean
Dim selrows As Integer
Dim selcols As Integer
Dim selstartrow As Integer
Dim selstartcol As Integer
Dim rowbeg As Integer
Dim colbeg As Integer
Dim arri1 As String
```

```
Dim arri2 As String
Dim aba1 As String
Dim aba2 As String
Dim agregar As Integer
Dim buscar As Integer
Dim num1 As String
Dim num2 As String
Dim j As Integer
Dim nume1, nume2, nume3, nume4, nume5, nume6 As String
Dim tama As Integer
Dim A, b, c As Integer

'Funcion para hacer retardos de tiempo. sleep(t en milisegundos)
Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

'Runtina que se ejecuta al darse el evento de click en el boton Agregar
Private Sub cmdAgregar_Click()

'Asignacion del valor de label a variable Numero
'con la finalidad de hacer una busqueda por toda la tabla
'para no agregar tags repetidos.
numero2 = lblTag.Caption
For j = 1 To 31
grdRFID.row = j
grdRFID.Col = 1
num1 = grdRFID.Text
If numero2 = num1 Then
MsgBox "El Tag ya ha sido agregado", vbExclamation, "Advertencia"
Exit Sub
End If
Next

'realiza la busqueda de tags repetidos, al igual q la anterior, y ademas,
'detecta cuando el boton se presiona y no hay un numero de tag valido
'asi como agrega los tags que si son validos a la tabla.
For buscar = 1 To 31
grdRFID.row = buscar
grdRFID.Col = 1
numero = grdRFID.Text
If numero = numero2 Then
MsgBox "El Tag ya ha sido agregado", vbExclamation, "Advertencia"
Exit Sub
ElseIf numero2 = "Deslizar un TAG" Then
MsgBox "Deslice un Tag para agregarlo", vbInformation, "Informacion"
Exit Sub
ElseIf numero = "" Then
agregar = buscar
grdRFID.row = agregar
grdRFID.Col = 1
grdRFID.Text = numero2
Exit Sub
End If
Next

End Sub

'Sincronizacion de los datos de la tabla con la memoria del microcontrolador
'envia los datos de la tabla al chip
Private Sub cmdEnviar_Click()
'desabilita todos los botones hasta que se ha dado la sincronizacion
cmdAgregar.Enabled = False
cmdBorrar.Enabled = False
cmdArriba.Enabled = False
cmdAbajo.Enabled = False
cmdSalir.Enabled = False
cmdEnviar.Enabled = False
cmdRecibir.Enabled = False
cmdMasterTag.Enabled = False
cmdGuardar.Enabled = False
ProgressBar1.Max = 32
Timer1.Enabled = False

'Recorre todos los datos de la tabla, para enviarlos uno por uno a la memoria del chip
'prepara el usb y envia dato por dato a la memoria del chip
For A = 1 To 32 Step 1
Sleep (100)
grdRFID.row = A
grdRFID.Col = 1
num1 = grdRFID.Text
```

```
If num1 = "" Then
num1 = "ffffffffff"
End If

'prepara los buffers del usb con los datos que se van a mandar. cada buffer
'contienen 8 bits y segun el descriptor de el usb son 6 buffers
ReDim Buffer(48)
BufferSize = 6
Buffer(0) = (ConvertHexToDecimal(Mid(num1, 1, 2)))
Buffer(1) = (ConvertHexToDecimal(Mid(num1, 3, 2)))
Buffer(2) = (ConvertHexToDecimal(Mid(num1, 5, 2)))
Buffer(3) = (ConvertHexToDecimal(Mid(num1, 7, 2)))
Buffer(4) = (ConvertHexToDecimal(Mid(num1, 9, 2)))
Buffer(5) = (A - 1)
HIDComm1.WriteTo Buffer, BufferSize
ProgressBar1.Value = ProgressBar1.Value + 1
Next A
ProgressBar1.Value = 0
MsgBox "Sincronizacion realizada con exito", vbInformation, "Sincronizacion"

'Habilita los botones despues de una sincronizacion exitosa.
cmdAgregar.Enabled = True
cmdBorrar.Enabled = True
cmdArriba.Enabled = True
cmdAbajo.Enabled = True
cmdSalir.Enabled = True
cmdEnviar.Enabled = True
cmdRecibir.Enabled = True
cmdMasterTag.Enabled = True
cmdGuardar.Enabled = True
Timer1.Enabled = True
HIDComm1.TimeOut = 10
End Sub

'Rutina para guardar en hoja de exel.
'se esta utilizando un modulo para lograr que los datos se exporten a exel.
Private Sub cmdGuardar_Click()

'Funcion que exporta a exel los datos del grid.
If Exportar_Excel("c:\AsignacionTags", grdRFID) Then
   MsgBox " Datos exportados ", vbInformation
End If

End Sub

'Rutina que manda el tag seleccionado a la ultima localidad en el grid (localidad de etiqueta
'de administracion).
Private Sub cmdMasterTag_Click()

'rutina que valida cuando la celda esta vacia.
If (grdRFID.HighLight) Then
If grdRFID.Text = "" Then
MsgBox "La celda esta vacia", vbInformation, "Advertencia"
Exit Sub
End If

If grdRFID.Col = 2 Then
MsgBox "Este No es un numero de TAG", vbInformation, "Advertencia"
Exit Sub
End If


'Rutina que valida que solo se seleccione una celda a la vez.
selstartrow = grdRFID.RowSel
rowbeg = grdRFID.row
If Not ((selstartrow - rowbeg) = 0) Then
MsgBox "Solo se puede seleccionar una celda", vbInformation, "Informacion"
Exit Sub
End If

'rutina que efectua el desplazamiento de celdas hacia abajo de uno a uno.
aba2 = grdRFID.Text
grdRFID.Text = ""
grdRFID.row = 31
grdRFID.Col = 1
grdRFID.Text = aba2
End If
End Sub

'Rutina que se ejecuta cuando se cierra una forma.(cuando se da en el boton cerrar X)
```

```
'Se desconecta el usb y termina el programa.
Private Sub Form_Terminate()
HIDComm1.Uninit
HIDComm1.Disconnect
Timer1.Enabled = False
End
End Sub

'Rutina que detecta cuando la tarjeta esta conectada por el puerto usb.
'trabaja en combinacion con una condicion IF que se encuentra mas adelante en el programa
Private Sub HIDComm1_ConnectSuccess(ByVal Status As Long)
    frmRFID.Visible = False
    frmConexion.Visible = False
    frmPass.Visible = True
    Timer1.Enabled = False
    frmPass.Timer1.Enabled = True
    End Sub

'Rutina que se ejecuta al darse el evento de hacer click en el boton abajo.
'Al dar click en el boton, lo que sucede, es que la celda seleccionada
'se desplaza una posicion hacia abajo. Si la celda que esta por debajo de la seleccionada
'posee informacion, se hace un cambio de datos, lo de abajo pasa arriba y lo de arriba abajo.
Private Sub cmdAbajo_Click()
'rutina que valida cuando la celda esta vacia.
If (grdRFID.HighLight) Then
If grdRFID.Text = "" Then
MsgBox "La celda esta vacia", vbInformation, "Advertencia"
Exit Sub
End If

'Rutina que valida que solo se seleccione una celda a la vez.
selstartrow = grdRFID.RowSel
rowbeg = grdRFID.row
If Not ((selstartrow - rowbeg) = 0) Then
MsgBox "Solo se puede seleccionar una celda", vbInformation, "Informacion"
Exit Sub
End If

'Rutina que valida cuando se alcanza la posicion final de la tabla y se quiere seguir bajando
'ademas cuando se quiere seleccionar la posicion 0 que contiene el nombre de la columna
If rowbeg = 31 Then
MsgBox "Accion Invalida", vbCritical, "ERROR"
Exit Sub
ElseIf rowbeg = 0 Then
MsgBox "Seleccione una celda con informacion", vbCritical, "ERROR"
Exit Sub
End If

'rutina que efectua el desplazamiento de celdas hacia abajo de uno a uno.
For selrows = rowbeg To selstartrow
grdRFID.row = selrows
grdRFID.Col = grdRFID.Col
aba1 = grdRFID.Text
grdRFID.row = (selrows + 1)
grdRFID.Col = grdRFID.Col
aba2 = grdRFID.Text
grdRFID.Text = aba1
grdRFID.row = selrows
grdRFID.Col = grdRFID.Col
grdRFID.Text = aba2
Next
grdRFID.row = rowbeg + 1
grdRFID.RowSel = selstartrow + 1
End If
End Sub

'Rutina que se ejecuta al darse el evento de hacer click en el boton arriba.
'Al dar click en el boton, lo que sucede, es que la celda seleccionada
'se desplaza una posicion hacia arriba. Si la celda que esta por arriba de la seleccionada
'posee informacion, se hace un cambio de datos, lo de arriba pasa abajo y lo de abajo arriba.
Private Sub cmdArriba_Click()
'rutina que valida cuando la celda esta vacia.
If (grdRFID.HighLight) Then
If grdRFID.Text = "" Then
MsgBox "La celda esta vacia", vbInformation, "Error"
Exit Sub
End If

'Rutina que valida que solo se seleccione una celda a la vez.
selstartrow = grdRFID.RowSel
```

```
rowbeg = grdRFID.row
If Not ((selstartrow - rowbeg) = 0) Then
MsgBox "Solo se puede seleccionar una celda", vbInformation, "Informacion"
Exit Sub
End If

'Rutina que valida cuando se alcanza la posicion inicial de la tabla y se quiere seguir subiendo
'ademas cuando se quiere seleccionar la posicion 0 que contiene el nombre de la columna
If rowbeg = 1 Then
MsgBox "Accion Invalida", vbCritical, "ERROR"
Exit Sub
ElseIf rowbeg = 0 Then
MsgBox "Seleccione una celda con informacion", vbCritical, "ERROR"
Exit Sub
End If

'rutina que efectua el desplazamiento de celdas hacia arriba de uno a uno.
For selrows = rowbeg To selstartrow
grdRFID.row = selrows
grdRFID.Col = grdRFID.Col
arri1 = grdRFID.Text
grdRFID.row = (selrows - 1)
grdRFID.Col = grdRFID.Col
arri2 = grdRFID.Text
grdRFID.Text = arri1
grdRFID.row = selrows
grdRFID.Col = grdRFID.Col
grdRFID.Text = arri2
Next
grdRFID.row = rowbeg - 1
grdRFID.RowSel = selstartrow - 1
End If
End Sub

'Rutina que se ejecuta cuando se da el evento click en el boton borrar
'Al hacer clic en el boton borrar, se borran los datos que se encuentran
'seleccionados en la tabla. Se puede seleccionar mas de una celda a la vez
Private Sub cmdBorrar_Click()
'rutina que se ejecuta cuando la celda seleccionada es la que
'contiene el texto de la columna.
If (grdRFID.HighLight) Then
If grdRFID.row = 0 Then
grdRFID.row = grdRFID.row
End If

'rutina de borrado, detecta cuantas celdas se han seleccionado y se borran
selstartrow = grdRFID.RowSel
rowbeg = grdRFID.row
For selrows = rowbeg To selstartrow
grdRFID.row = selrows
grdRFID.Col = grdRFID.Col
grdRFID.Text = ""
Next
grdRFID.row = rowbeg
grdRFID.RowSel = selstartrow
End If
End Sub

'Rutina que se ejecuta al darse el evento clic en el boton recibir.
'Al hacer clic en el boton recibir, se desabilitan todos los botones
'mientras se esta efectuando la sincronizacion.
'Se reciben todos los datos de la memoria del chip a la tabla en el programa
Private Sub cmdRecibir_Click()
HIDComm1.TimeOut = 1000
ProgressBar1.Max = 31
cmdAgregar.Enabled = False
cmdBorrar.Enabled = False
cmdArriba.Enabled = False
cmdAbajo.Enabled = False
cmdSalir.Enabled = False
cmdEnviar.Enabled = False
cmdRecibir.Enabled = False
cmdMasterTag.Enabled = False
cmdGuardar.Enabled = False
uno = True
End Sub

'Rutina que se ejecuta al darse el evento clic del boton salir.
'Al hacer clic en el boton salir, se desconecta el puerto usb y se
'termina el programa.
```

```
Private Sub cmdSalir_Click()
HIDComm1.Uninit
HIDComm1.Disconnect
Timer1.Enabled = False
End
End Sub

'Rutina que se ejecuta al darse el evento de carga de la forma
'Se inicializa el puerto usb, y todos los controles de la forma principal
'se cargan todas configuraciones de la tabla.
Private Sub Form_Load()

    Text1.BorderStyle = vbBSNone
    Text1.FontName = grdRFID.FontName
    Text1.FontSize = grdRFID.FontSize
    Text1.Visible = False
    grdRFID.Cols = 3
    grdRFID.Rows = 33
    grdRFID.ColWidth(0) = 500
    grdRFID.ColWidth(1) = 2200
    grdRFID.ColWidth(2) = 4100



'inicializacion de puerto usb
frmRFID.HIDComm1.TimeOut = 1000
frmRFID.HIDComm1.Connect

'rutina que detecta cuando el usb no se ha conectado a la pc.
'esta rutina se ejecuta solo en la carga del programa.
If Not HIDComm1.Connected Then
MsgBox "No se ha detectado ningun lector RFID conectado al equipo", vbCritical, "Error"
HIDComm1.Uninit
HIDComm1.Disconnect
Timer1.Enabled = False
frmPass.Timer1.Enabled = False
End
End If

'Obtencion del tag maestro desde la memoria del chip hasta la posicion
'final de la tabla, con la finalidad de que cuando se sincronice no se
'borre por error.
frmRFID.HIDComm1.TimeOut = 1000
frmRFID.HIDComm1.Connect
ReDim Buffer(48)
    Buffer(0) = 0
    Buffer(1) = 0
    Buffer(2) = 0
    Buffer(3) = 0
    Buffer(4) = 0
    Buffer(5) = 51
    BufferSize = 6
    frmRFID.HIDComm1.WriteTo Buffer, BufferSize
    Buffer(5) = 0
    Buffer = frmRFID.HIDComm1.ReadFrom(BufferSize)
    grdRFID.Col = 1
    grdRFID.row = 31

    grdRFID.Text = Format(Hex(Buffer(0)), "00") & Format(Hex(Buffer(1)), "00") & _
    Format(Hex(Buffer(2)), "00") & Format(Hex(Buffer(3)), "00") & _
    Format(Hex(Buffer(4)), "00")

'Llamada a funciones que inicializan las propiedades de la tabla
'las propiedades de las celdas y los titulos en la tabla.
'tambien las propiedades de color en celdas y la inicializacion de
'las barras de progreso.
HIDComm1.TimeOut = 10
Call CenterCells
'Call Sizecells
Call titles
ProgressBar1.Min = 0
ProgressBar1.Value = 0
HIDComm1.Connect
HIDComm1.TimeOut = 50
lblTag.Caption = "Deslizar un TAG"
grdRFID.row = 32
grdRFID.Col = 1
grdRFID.CellBackColor = &H8000000F
grdRFID.row = 32
grdRFID.Col = 0
```

```
grdRFID.CellBackColor = &H8000000F
grdRFID.row = 32
grdRFID.Col = 2
grdRFID.CellBackColor = &H8000000F
Timer1.Enabled = True
End Sub

'Rutina para poder editar los campos del grid, con la finalidad de poder asociar
'un tag y usuario.
Private Sub GridEdit(KeyAscii As Integer)

'Rutina para especificar que no se puede editar el campo del tag de administracion
If ((grdRFID.Col = 2 Or grdRFID.Col = 1) And grdRFID.row = 32) Then
    MsgBox "Esta celda no se puede editar", vbCritical, "ERROR"
    Exit Sub
    End If

'Rutina que no permite la edicion de los campos en la columna de etiquetas.
    If (grdRFID.Col = 1) Then
    MsgBox "No se puede editar columna de etiquetas", vbCritical, "ERROR"
    Exit Sub
    End If

'Rutina para posicionar el textbox sobre la celda.
    Text1.Left = grdRFID.CellLeft + grdRFID.Left + 480
    Text1.Top = grdRFID.Top + grdRFID.CellTop + 1680
    Text1.Width = grdRFID.CellWidth
    Text1.Height = grdRFID.CellHeight
    Text1.Visible = True
    Text1.SetFocus

    Select Case KeyAscii
        Case 0 To Asc(" ")
            Text1.Text = grdRFID.Text
            Text1.SelStart = Len(Text1.Text)
        Case Else
            Text1.Text = Chr$(KeyAscii)
            Text1.SelStart = 1
    End Select
End Sub

'Rutina que detecta las teclas presionadas, con la finalidad de poder editar
'desplazandose con teclas.
Private Sub Text1_KeyDown(KeyCode As Integer, Shift As Integer)
    Select Case KeyCode

        'Dejar el texto sin cambios.
        Case vbKeyEscape
            Text1.Visible = False
            grdRFID.SetFocus

        'Finalizar la edicion
        Case vbKeyReturn
            grdRFID.SetFocus

        'Moviendose abajo una fila.
        Case vbKeyDown
            grdRFID.SetFocus
            DoEvents
            If grdRFID.row < grdRFID.Rows - 1 Then
                grdRFID.row = grdRFID.row + 1
            End If

        'Moviendose arriba una fila.
        Case vbKeyUp
            grdRFID.SetFocus
            DoEvents
            If grdRFID.row > grdRFID.FixedRows Then
                grdRFID.row = grdRFID.row - 1
            End If

    End Select
End Sub
'No hacer nada al presionar ESC o ENTER.
Private Sub Text1_KeyPress(KeyAscii As Integer)
    If (KeyAscii = vbKeyReturn) Or _
       (KeyAscii = vbKeyEscape) _
            Then KeyAscii = 0
End Sub
```

```
'Rutina para comenzar la edicion al hacer doble click
Private Sub grdRFID_DblClick()
    GridEdit Asc(" ")
End Sub

'Rutina que detecta las teclas presionadas al editar una celda
Private Sub grdRFID_KeyPress(KeyAscii As Integer)
    GridEdit KeyAscii
End Sub

'Rutina que se ejecuta al dejar de editar una celda
Private Sub grdRFID_LeaveCell()
    If Text1.Visible Then
        grdRFID.Text = Text1.Text
        Text1.Visible = False
    End If
End Sub

'Rutina que se ejecuta al momento de la edicion de la celda.
Private Sub grdRFID_GotFocus()
    If Text1.Visible Then
        grdRFID.Text = Text1.Text
        Text1.Visible = False
    End If
End Sub

'Funcion que define las propiedades de las columnas de la tabla
Private Sub CenterCells()
Dim column As Integer
For column = 0 To 1
grdRFID.Col = column
grdRFID.ColAlignment(column) = flexAlignCenterCenter
Next column
End Sub

'Funcion que define la propiedad del tamano de las celdas
'Private Sub Sizecells()
'grdRFID.ColWidth(0) = 500
'grdRFID.ColWidth(1) = 4000
'End Sub

'Funcion que define las propiedades de titulo de las celdas.
'Asigna titulos a las celdas que lo requieran
Private Sub titles()
Dim row As Integer
grdRFID.Col = 0
For row = 1 To 31
grdRFID.row = row
grdRFID.Text = row
Next row
grdRFID.row = 0
grdRFID.Col = 1
grdRFID.Text = "Etiqueta"
grdRFID.row = 0
grdRFID.Col = 2
grdRFID.Text = "Usuario"
grdRFID.CellAlignment = flexAlignCenterCenter
grdRFID.row = 31
grdRFID.Col = 2
grdRFID.Text = "Etiqueta de Administracion"
End Sub

'Evento Timer, que se esta ejecutando cada 10ms independientemente
'de lo que este sucediendo en el programa.
'En este evento se detecta cuando la tarjeta es desconectada
'Tambien en este evento se hace la sincronizacion de la memoria del chip
'a la tabla
Private Sub Timer1_Timer()
'Rutina que detecta cuando se ha desconectado la tarjeta
If Not HIDComm1.Connected Then
        frmRFID.Visible = False
        frmConexion.Visible = True
        frmPass.Visible = False
        frmConexion.Label1.Caption = "Por favor conecte la tarjeta"
        HIDComm1.Connect
        Exit Sub
    End If

'Inicio del usb, manda configuracion de inicio para recibir la memoria del chip
ReDim Buffer(48)
```

```
If uno = True Then
    Buffer(0) = 0
    Buffer(1) = 0
    Buffer(2) = 0
    Buffer(3) = 0
    Buffer(4) = 0
    Buffer(5) = 50
    BufferSize = 6
    HIDComm1.WriteTo Buffer, BufferSize
    Buffer(5) = 0
    'Bucle que se ejecuta hasta que se ha sincronizado toda la memoria del chip
    'con el programa.
    'Cada numero sincronizado, lo va colocando en una celda. el numero de celda es
    'acorde al numero de posicion en la memoria del chip.
    'Tambien se efectua la visualizacion de la barra de progreso.
    While (x < 31)
    Buffer = HIDComm1.ReadFrom(BufferSize)
    If Buffer(5) = x Then
    grdRFID.row = (x + 1)
    grdRFID.Col = 1
    If (Hex(Buffer(0)) & Hex(Buffer(1)) & Hex(Buffer(2)) & Hex(Buffer(3)) & Hex(Buffer(4)) = "FFFFFFFFFF")
Then
    grdRFID.Text = ""
    Else

    grdRFID.Text = Format(Hex(Buffer(0)), "00") & Format(Hex(Buffer(1)), "00") & _
    Format(Hex(Buffer(2)), "00") & Format(Hex(Buffer(3)), "00") & _
    Format(Hex(Buffer(4)), "00")

    End If
    x = x + 1
    End If
    ProgressBar1.Value = ProgressBar1.Value + 1
    Wend
    ProgressBar1.Value = 0
    MsgBox "Sincronizacion realizada con exito", vbInformation, "Sincronizacion"
    uno = False
    'Se habilitan todos los botones de la celda, cuando ya se ha realizado
    'la sincronizacion.
    cmdAgregar.Enabled = True
    cmdBorrar.Enabled = True
    cmdArriba.Enabled = True
    cmdAbajo.Enabled = True
    cmdSalir.Enabled = True
    cmdEnviar.Enabled = True
    cmdRecibir.Enabled = True
    cmdMasterTag.Enabled = True
    cmdGuardar.Enabled = True
    HIDComm1.TimeOut = 10
    Exit Sub
    End If


'Rutina que se esta ejecutando con elo evento timer, para detectar
'los tags que son deslizados, y los presenta en en label junto al
'boton agregar. con la finalidad de poder agregarlos a la tabla, para
'luego sincronizarlos con la memoria del chip
BufferSize = 6
Buffer = HIDComm1.ReadFrom(BufferSize)
If BufferSize < 6 Then
Exit Sub
End If

lblTag.Caption = Format(Hex(Buffer(0)), "00") & Format(Hex(Buffer(1)), "00") & _
Format(Hex(Buffer(2)), "00") & Format(Hex(Buffer(3)), "00") & _
Format(Hex(Buffer(4)), "00")

If (lblTag.Caption = "FFFFFFFFFF") Then
lblTag.Caption = "Deslizar un TAG"
End If
HIDComm1.TimeOut = 10
End Sub

'Rutina que finaliza el programa al hacer clic en la X
'termina el programa y cierra las conexiones
Private Sub Form_Unload(Cancel As Integer)
frmRFID.HIDComm1.Uninit
frmRFID.HIDComm1.Disconnect
frmRFID.Timer1.Enabled = False
  End
```

```
End Sub

'NOTA:
'- El modulo agregado (modProcedures), sirve para hacer conversiones de numeros
'  entre diferentes sistemas.
'- El modulo agregado (export), sirve para exportar el grid a un documento de exel.
'- El modulo agregado (module1), sirve para ejecutar un proceso automatico despues de
'  un tiempo especificado.
```

Formulario de desconexión

13.2.3 Formulario de desconexión

```
'Forma que se ejecuta cuando se ha desconectado la tarjeta de la pc
'muestra en pantalla la forma de conexion, con la leyenda que cita
'"Por favor conecte la tarjeta"

'Rutina que se ejecuta cuando se cierra la forma, cuando se hace
'clic en el boton cerrar X
Private Sub Form_Unload(Cancel As Integer)
frmRFID.HIDComm1.Uninit
frmRFID.HIDComm1.Disconnect
frmRFID.Timer1.Enabled = False
  End
End Sub
```

MODULO PARA CONVERSION DE NUMEROS ENTRE SISTEMAS. (modProcedures.bas)

```
'Modulo descargado desde la direccion
'http://vbasic.astalaweb.com/C%C3%A1lculos%20II/Convierte%20de%20hexadecimal%20a%20decimal.asp
'Adaptado por Luis Mario Mercado.

Const HexA = 10
Const HexB = 11
Const HexC = 12
Const HexD = 13
Const HexE = 14
Const HexF = 15

Public Function ConvertHexToDecimal(ByVal HexValue As String) As Variant
    Dim rValue, A As Long
    Dim Temp, Rev As String

    Rev = StrReverse(HexValue)
    'Numbers are read from right to left, unlike text, which
    'are read from right to left; therefore, the string should
    'be reversed so it can be read like a normal string.

    For A = 1 To Len(HexValue)
        Temp = Mid$(Rev, A, 1)
        'This, along with the for-next loop allows you to
        'read all the characters in the screen 1 at a time

        If Val(Temp) = Temp Then
            'Character is a number
            If A = 1 Then
                'Character is a number, and is the first
                'character in the string
                rValue = Val(Temp)
                'Because the character is the first of the
                'string, there is no value in rValue yet, so
                'you can just assign the value
            Else
                rValue = rValue + (Val(Temp) * (16 ^ (A - 1)))
                'So this adds to rValue.. it takes it's
```

```
                'current value, and adds to it, so the
                'previous value isn't lost.  Because single
                'digit hex values can be up to a value of 15,
                'the decimal value of 10 would be 16.  Now,
                'values are not their own when they are not
                'their own if they aren't the first character
                'of the string.  Here, we use exponents..
                'if 16^0 were 0, then I would have done this
                'diffrently, but because it isn't, it has to
                'be it's position(a) -1.  The -1 is because
                'you don't multiply by 16 on the first
                'character, you start it on the 2nd character.
            End If
        Else
            Select Case LCase$(Temp)
                'Because single digit hex values can go up to
                '15, more characters were needed, so A - F
                'were added in.  A has the value of 10, B has
                'the value of 11, and so forth.  G is not a
                'valid character because it is a 15 value
                'system.  Here, it just goes through the
                'valid letters, and does the same thing it did
                'with numbers.
                Case "a"
                    If A = 1 Then
                        rValue = HexA
                    Else
                        rValue = rValue + (HexA * (16 ^ (A - 1)))
                    End If
                Case "b"
                    If A = 1 Then
                        rValue = HexB
                    Else
                        rValue = rValue + (HexB * (16 ^ (A - 1)))
                    End If
                Case "c"
                    If A = 1 Then
                        rValue = HexC
                    Else
                        rValue = rValue + (HexC * (16 ^ (A - 1)))
                    End If
                Case "d"
                    If A = 1 Then
                        rValue = HexD
                    Else
                        rValue = rValue + (HexD * (16 ^ (A - 1)))
                    End If
                Case "e"
                    If A = 1 Then
                        rValue = HexE
                    Else
                        rValue = rValue + (HexE * (16 ^ (A - 1)))
                    End If
                Case "f"
                    If A = 1 Then
                        rValue = HexF
                    Else
                        rValue = rValue + (HexF * (16 ^ (A - 1)))
                    End If
            End Select
        End If
        DoEvents
        'ALWAYS have this in loops.. you don't know how slow
        'the computer it's being operated on is, the lack of
        'DoEvents could cause the computer freeze up
        'temporerily, or even force you to force shutdown.
    Next A
    ConvertHexToDecimal = rValue
    'Sends the value that has been made out :)
End Function
```

Modulo para exportar los datos de un control FlexGrid a documento de Excell (exporttoexel.bas)

```
'Modulo descargado de la direccion:
'http://www.recursosvisualbasic.com.ar/htm/trucos-codigofuente-visual-basic/
'240-exportar-flexgrid-a-excel.htm
'Adaptado por Luis Mario Mercado.


Public Function Exportar_Excel(Path_Libro As String, _
                               FlexGrid As MSFlexGrid) As Boolean

On Error GoTo errSub

'Variables para la aplicación Excel, el libro y la hoja
Dim o_Excel As Object
Dim o_Libro As Object
Dim o_Hoja  As Object

'Para las filas y columnas del FlexGrid y la Hoja
Dim Fila As Integer
Dim Columna As Integer

If Path_Libro = vbNullString Then
   ' Falta la ruta del libro
   MsgBox " Falta el Path del archivo de Excel "
   Exit Function
End If

Exportar_Excel = False

' crea los objetos y agrga el libro y la hoja
Set o_Excel = New Excel.Application
Set o_Libro = o_Excel.Workbooks.Add
Set o_Hoja = o_Libro.Worksheets.Add

' Recorremos el FlexGrid por filas y columnas
For Fila = 1 To FlexGrid.Rows - 1

    For Columna = 0 To FlexGrid.Cols - 1

        ' Agrega el Valor en la celda indicada  del Excel
        o_Hoja.Cells(Fila, Columna + 1).Value = _
                    FlexGrid.TextMatrix(Fila, Columna)

    Next Columna

Next Fila

' Para guardar el Libro
o_Hoja.SaveAs Path_Libro

' Cierra el libro
o_Libro.Close

' Cierra el excel
o_Excel.Quit

' Elimina las instancias
If Not o_Excel Is Nothing Then
    Set o_Excel = Nothing
End If

If Not o_Libro Is Nothing Then
    Set o_Libro = Nothing
End If

If Not o_Hoja Is Nothing Then
    Set o_Hoja = Nothing
End If

Exportar_Excel = True

Exit Function

'Error

errSub:
```

```
'Cierra la hoja y el la aplicación Excel
If Not o_Libro Is Nothing Then: o_Libro.Close False
If Not o_Excel Is Nothing Then: o_Excel.Quit

'Liberar los objetos
Set o_Excel = Nothing
Set o_Libro = Nothing
Set o_Hoja = Nothing

If Err.Number <> 0 Then
    MsgBox Err.Number & " - " & Err.Description
End If

End Function
```

Modulo para Automatizar el ingreso de teclas (Modulemsgbox.bas)

```
'Modulo descargado desde la direccion
'http://www.lawebdelprogramador.com/news/mostrar_new.php?id=96&texto=Visual+
'Basic+Avanzado&n1=434376&n2=0&n3=0&n4=0&n5=0&n6=0&n7=0&n8=0&n9=0&n0=0
'Adaptado por Luis Mario Mercado.

Public Const NV_CLOSEMSGBOX As Long = &H5000&
Public Declare Function SetTimer& Lib "user32" (ByVal hWnd&, ByVal nIDEvent&, _
ByVal uElapse&, ByVal lpTimerFunc&)
Public Declare Function FindWindow& Lib "user32" Alias "FindWindowA" _
(ByVal lpClassName$, ByVal lpWindowName$)
Public Declare Function LockWindowUpdate& Lib "user32" (ByVal hwndLock&)
Public Declare Function SetForegroundWindow& Lib "user32" (ByVal hWnd&)
Public Declare Function MessageBox& Lib "user32" Alias "MessageBoxA" _
(ByVal hWnd&, ByVal lpText$, ByVal lpCaption$, ByVal wType&)
Public Declare Function KillTimer& Lib "user32" (ByVal hWnd&, ByVal nIDEvent&)
Public Const API_FALSE As Long = 0&

Public Sub TimerProc(ByVal hWnd&, ByVal uMsg&, ByVal idEvent&, ByVal dwTime&)
 KillTimer hWnd, idEvent
 Dim hMessageBox&
 hMessageBox = FindWindow("#32770", "AutoCerrado")
  If hMessageBox Then
   Call SetForegroundWindow(hMessageBox)
   SendKeys "{enter}"
  End If
  Call LockWindowUpdate(API_FALSE)
End Sub
```

13.2.2 Programa para agregar etiquetas al registro de Windows y asociarlas a un usuario

Formulario de autenticación para agregar o remover etiquetas asociadas a usuarios.

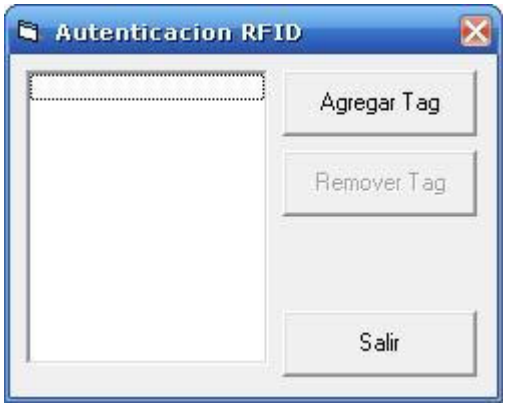


Figura 13.2.4 Formulario de autenticación para agregar o remover etiquetas asociadas a usuarios

```
'Programa descargado de la direccion:
'http://www.rfidtoys.net/forum/forum_posts.asp?TID=4&PN=1
'Modificado y Adaptado por Luis Mario Mercado.
‘Modificado para el funcionamiento con el sistema lector RFID


Option Explicit

'public vars which are updated via update form
Public strUpdateTagID As String
Public strUpdateUsername As String
Public strUpdatePassword As String
Public strUpdateDomain As String


'//////////////////////////////////////////
'                                        //
'               Registry API             //
'                                        //
' //////////////////////////////////////////

Private Type SECURITY_ATTRIBUTES
        nLength As Long
        lpSecurityDescriptor As Long
        bInheritHandle As Long
End Type

' Reg Data Types
Private Const REG_SZ = 1 ' Unicode null terminated string

' Reg Options
Private Const REG_OPTION_RESERVED = 0
Private Const REG_OPTION_NON_VOLATILE = 0

' Reg Key Security
Private Const KEY_ALL_ACCESS = &H3F

' Reg Key Constants
Private Const HKEY_CLASSES_ROOT = &H80000000
Private Const HKEY_CURRENT_USER = &H80000001
Private Const HKEY_LOCAL_MACHINE = &H80000002
Private Const HKEY_USERS = &H80000003
Private Const HKEY_PERFORMANCE_DATA = &H80000004
Private Const HKEY_CURRENT_CONFIG = &H80000005
Private Const HKEY_DYN_DATA = &H80000006

' Reg API function calls
Private Declare Function RegCloseKey Lib "advapi32.dll" (ByVal hKey As Long) As Long
Private Declare Function RegCreateKeyEx Lib "advapi32.dll" Alias "RegCreateKeyExA" (ByVal hKey As Long,
ByVal lpSubKey As String, ByVal Reserved As Long, ByVal lpClass As String, ByVal dwOptions As Long, ByVal
samDesired As Long, lpSecurityAttributes As SECURITY_ATTRIBUTES, phkResult As Long, lpdwDisposition As Long)
As Long
Private Declare Function RegDeleteKey Lib "advapi32.dll" Alias "RegDeleteKeyA" (ByVal hKey As Long, ByVal
lpSubKey As String) As Long
Private Declare Function RegEnumKey Lib "advapi32.dll" Alias "RegEnumKeyA" (ByVal hKey As Long, ByVal
dwIndex As Long, ByVal lpName As String, ByVal cbName As Long) As Long
Private Declare Function RegOpenKeyEx Lib "advapi32.dll" Alias "RegOpenKeyExA" (ByVal hKey As Long, ByVal
lpSubKey As String, ByVal ulOptions As Long, ByVal samDesired As Long, phkResult As Long) As Long
Private Declare Function RegQueryValueExNULL Lib "advapi32.dll" Alias "RegQueryValueExA" (ByVal hKey As
Long, ByVal lpValueName As String, ByVal lpReserved As Long, lpType As Long, ByVal lpData As Long, lpcbData
As Long) As Long
Private Declare Function RegQueryValueExString Lib "advapi32.dll" Alias "RegQueryValueExA" (ByVal hKey As
Long, ByVal lpValueName As String, ByVal lpReserved As Long, lpType As Long, ByVal lpData As String,
lpcbData As Long) As Long
Private Declare Function RegSetValueExString Lib "advapi32.dll" Alias "RegSetValueExA" (ByVal hKey As Long,
ByVal lpValueName As String, ByVal Reserved As Long, ByVal dwType As Long, ByVal lpValue As String, ByVal
cbData As Long) As Long


Private Sub CheckRegistryKey()
    Dim hKey As Long 'registry key handle value
    Dim lRetVal As Long 'return value for API call result codes

    'try to open registry key
    lRetVal = RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\rfidPoll\" & Chr(0), 0, KEY_ALL_ACCESS, hKey)

    'check for errors
    If lRetVal <> 0 Then
        'error opening key
```

```
        'check error type
        If lRetVal = 2 Then
            'key does not exist, create new one
            Call CreateRegistryKey
        Else
            'error was some other kind of problem
            MsgBox "Ocurrio un error al abrir el registro."
            Exit Sub
        End If

    End If

End Sub
Private Sub CreateRegistryKey()
    Dim hKey As Long 'registry key handle value
    Dim lRetVal As Long 'return value for API call result codes
    Dim lRetHKEY As Long 'return value for registry key handle
    Dim lDisposition As Long 'return code so we know if key was created or an existing key opened
    Dim saKeyAttrib As SECURITY_ATTRIBUTES 'holds attributes structure

    'try to open registry key
    lRetVal = RegOpenKeyEx(HKEY_LOCAL_MACHINE, "" & Chr(0), 0, KEY_ALL_ACCESS, hKey)

    'check for errors
    If lRetVal <> 0 Then
        'error opening key
        MsgBox "Ocurrio un error al abrir el registro"
        Exit Sub
    End If

    'create key

    lRetVal = RegCreateKeyEx(hKey, "SOFTWARE\rfidPoll\", 0, Chr(0), REG_OPTION_NON_VOLATILE, _
    0, saKeyAttrib, lRetHKEY, lDisposition)

    'check for errors
    If lRetVal <> 0 Then
        MsgBox "Ocurrio un error tratando de crear las claves del registro."
    End If

    'close registry keys
    Call RegCloseKey(lRetHKEY)
    Call RegCloseKey(hKey)
End Sub


Private Sub LoadTags()
    Dim strKeyName As String 'registry key name return value
    Dim hKey As Long 'registry key handle value
    Dim lRetVal As Long 'return value for API call result codes
    Dim lKeyCount As Long 'counts the number of keys we've enumerated so far

    'clear list box
    lstRFIDTags.Clear

    'open registry key

    lRetVal = RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\rfidPoll\" & Chr(0), 0, _
    KEY_ALL_ACCESS, hKey)

    'check for errors
    If lRetVal <> 0 Then
        MsgBox "Ocurrio un error al abrir la clave de registro requerida."
        Exit Sub 'could not open the registry key!
    End If

    'enumerate subkeys and load them into the listbox
    Do 'loop until we get an error
        strKeyName = Space$(255) 'create room in the variable for API return value

        'call enumeration API
        lRetVal = RegEnumKey(hKey, lKeyCount, strKeyName, 255)

        'check for errors
        If lRetVal <> 0 Then
            'exit loop, we've found them all
            Exit Do
        End If

        'another key found, increase count and add to the listbox
```

```
        lKeyCount = lKeyCount + 1 'increase count
        strKeyName = Trim(Replace(strKeyName, Chr(0), "")) 'trim spaces and remove NULL chars
        lstRFIDTags.AddItem strKeyName 'add to listbox
    Loop

    'close the registry key
    Call RegCloseKey(hKey)
End Sub


Private Sub cmdAddNew_Click()
    Dim saKeyAttrib As SECURITY_ATTRIBUTES
    Dim lDisposition As Long 'return code for key created or existing key opened
    Dim lRetVal As Long
    Dim hKey As Long

    'clear public vars
    strUpdateTagID = ""
    strUpdateUsername = ""
    strUpdatePassword = ""
    strUpdateDomain = ""

    'load update form
    Load frmUpdateTag

    'show form as modal
    frmUpdateTag.Show vbModal

    'form has returned, check public vars
    If strUpdateTagID = "" Then 'CANCEL was clicked
        Exit Sub 'they clicked cancel
    End If

    'add tag to registry

    'create key

    lRetVal = RegCreateKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\rfidPoll\" & strUpdateTagID & _
    Chr(0), 0, Chr(0), REG_OPTION_NON_VOLATILE, KEY_ALL_ACCESS, saKeyAttrib, hKey, 0)

    'check for errors
    If lRetVal <> 0 Then
        MsgBox "Ocurrio un error al crear la clave de registro."
        Exit Sub
    End If

    'set values
    lRetVal = RegSetValueExString(hKey, "Username", 0&, REG_SZ, strUpdateUsername, Len(strUpdateUsername))
    lRetVal = RegSetValueExString(hKey, "Password", 0&, REG_SZ, strUpdatePassword, Len(strUpdatePassword))
    lRetVal = RegSetValueExString(hKey, "Domain", 0&, REG_SZ, strUpdateDomain, Len(strUpdateDomain))

    'close registry key
    Call RegCloseKey(hKey)

    'reload tags
    Call LoadTags
End Sub

Private Sub cmdExit_Click()
    Unload frmRFIDAuth
End Sub

Private Sub cmdRemoveTag_Click()
    Dim lRetVal As Long
    Dim hKey As Long

    'check to see if they user really wants to remove the tag

    If MsgBox("Esta seguro que desea" & vbCrLf & "remover la etiqueta " & _
    lstRFIDTags.List(lstRFIDTags.ListIndex) & "?", vbYesNo, "Remover Etiqueta " & _
    lstRFIDTags.List(lstRFIDTags.ListIndex)) = vbYes Then

        'yes, they want to remove it

        'open key with write access
        lRetVal = RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\rfidPoll\" & _
        lstRFIDTags.List(lstRFIDTags.ListIndex) & Chr(0), 0, KEY_ALL_ACCESS, hKey)

        'check for errors
        If lRetVal <> 0 Then
```

```
            'error opening registry key
            MsgBox "Ocurrio un error al abrir la clave de registro."
            Exit Sub
        End If

        'remove tag from registry
        lRetVal = RegDeleteKey(hKey, Chr(0))

        'check for errors
        If lRetVal <> 0 Then
            'error opening registry key
            MsgBox "Ocurrio un error al remover la clave de registro."
            Exit Sub
        End If

        'close registry key
        Call RegCloseKey(hKey)

        'disable button
        cmdRemoveTag.Enabled = False

        'load tags again
        Call LoadTags
    End If
End Sub

Private Sub Form_Load()
    Dim strReturn As String

    'check to see if the application is already running
    If App.PrevInstance = True Then End

    'ensure registry key has been created
    Call CheckRegistryKey

    'load existing RFID tags from registry into listbox
    Call LoadTags

    'disable buttons
    cmdRemoveTag.Enabled = False
End Sub


Private Sub Form_Unload(Cancel As Integer)
    End
End Sub


Private Sub lstRFIDTags_Click()
    'check to see if there are any tags selected
    If lstRFIDTags.SelCount > 0 Then
        'a tag is selected, enable remove button
        cmdRemoveTag.Enabled = True
    End If
End Sub

Private Sub lstRFIDTags_DblClick()
    Dim strUsername As String
    Dim strPassword As String
    Dim strDomain As String
    Dim lRetVal As Long
    Dim lValueType As Long
    Dim lValueLength As Long
    Dim hKey As Long

    'get values from registry

    'open key with write access

    lRetVal = RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\rfidPoll\" & _
    lstRFIDTags.List(lstRFIDTags.ListIndex) & Chr(0), 0, KEY_ALL_ACCESS, hKey)

    'check for errors
    If lRetVal <> 0 Then
        'error opening registry key
        MsgBox "Ocurrio un error al abirir la clave de registro."
        Exit Sub
    End If

    'get values from key
```

```
    'get size and type of registry value
    lRetVal = RegQueryValueExNULL(hKey, "Username", 0&, lValueType, 0&, lValueLength)
    'check for errors
    If lRetVal <> 0 Then
        MsgBox "Ocurrio un error al leer los valores de registro."
        Exit Sub
    End If
    strUsername = String(lValueLength, 0)  'prep variable space for API return with nulls
    lRetVal = RegQueryValueExString(hKey, "Username", 0, REG_SZ, strUsername, 255) 'get value
    strUsername = Replace(Left$(strUsername, lValueLength), Chr(0), "") 'clean return value

    'get size and type of registry value
    lRetVal = RegQueryValueExNULL(hKey, "Password", 0&, lValueType, 0&, lValueLength)
    'check for errors
    If lRetVal <> 0 Then
        MsgBox "Ocurrio un error al leer los valores de registro."
        Exit Sub
    End If
    strPassword = String(lValueLength, 0)  'prep variable space for API return with nulls
    lRetVal = RegQueryValueExString(hKey, "Password", 0, REG_SZ, strPassword, 255) 'get value
    strPassword = Replace(Left$(strPassword, lValueLength), Chr(0), "") 'clean return value

    'get size and type of registry value
    lRetVal = RegQueryValueExNULL(hKey, "Domain", 0&, lValueType, 0&, lValueLength)
    'check for errors
    If lRetVal <> 0 Then
        MsgBox "Ocurrio un error al leer los valores de registro."
        Exit Sub
    End If
    strDomain = String(lValueLength, 0)  'prep variable space for API return with nulls
    lRetVal = RegQueryValueExString(hKey, "Domain", 0, REG_SZ, strDomain, 255) 'get value
    strDomain = Replace(Left$(strDomain, lValueLength), Chr(0), "") 'clean return value


    'clear public vars
    strUpdateTagID = ""
    strUpdateUsername = ""
    strUpdatePassword = ""
    strUpdateDomain = ""

    'load update form
    Load frmUpdateTag

    'copy values to form
    frmUpdateTag.txtTagID.Text = lstRFIDTags.List(lstRFIDTags.ListIndex)
    frmUpdateTag.txtTagID.Enabled = False
    frmUpdateTag.txtUsername.Text = strUsername
    frmUpdateTag.txtPassword.Text = strPassword
    frmUpdateTag.txtDomain.Text = strDomain

    'show form as modal
    frmUpdateTag.Show vbModal

    'form has returned, check public vars
    If strUpdateTagID <> "" Then 'OK was clicked
        'update tag in registry
        lRetVal = RegSetValueExString(hKey, "Username", 0&, REG_SZ, strUpdateUsername,
Len(strUpdateUsername))
        lRetVal = RegSetValueExString(hKey, "Password", 0&, REG_SZ, strUpdatePassword,
Len(strUpdatePassword))
        lRetVal = RegSetValueExString(hKey, "Domain", 0&, REG_SZ, strUpdateDomain, Len(strUpdateDomain))
    End If

    'close registry key
    Call RegCloseKey(hKey)
End Sub
```

Formulario de asociación de etiquetas a cuentas de usuarios

Figura 13.2.5 Formulario de asociación de etiquetas a cuentas de usuarios

```
'Option Explicit
Dim Buffer() As Byte
Dim BufferSize As Long
Dim uno As Long
Dim texto As String

'rutina para tarjeta Phidgets. (Removido)
''Phidgets objects
'Private WithEvents phManager As PhidgetManager
'Private WithEvents phRFID As PhidgetRFID
Private Sub cmdCancel_Click()
    Unload frmUpdateTag
End Sub


Private Sub cmdOK_Click()
    frmRFIDAuth.strUpdateTagID = txtTagID.Text
    frmRFIDAuth.strUpdateUsername = txtUsername.Text
    frmRFIDAuth.strUpdatePassword = txtPassword.Text
    frmRFIDAuth.strUpdateDomain = txtDomain.Text
    Unload frmUpdateTag
End Sub


Private Sub Form_Load()
 HIDComm1.Connect
 Timer1.Enabled = True
 HIDComm1.TimeOut = 10

 'rutina para tarjeta Phidgets. (Removido)
    'create new PhidgetManager object
 '   Set phManager = New PhidgetManager

End Sub

Private Sub Form_Unload(Cancel As Integer)
    'On Error Resume Next
    Timer1.Enabled = False
    HIDComm1.Uninit

    'rutina para tarjeta Phidgets. (Removido)
    'phRFID.OutputState(2) = False
'    phRFID.OutputState(3) = False
    'destroy Phidgets objects
'    Set phRFID = Nothing
    'Set phManager = Nothing
    'On Error GoTo 0

End Sub

'rutina para tarjeta Phidgets. (Removido)
```

```
'Private Sub phManager_OnAttach(ByVal PHIDGET As PHIDGET.IPhidget)
    'Dim lOutput As Long

    'check to see if it's an RFID reader
    'If PHIDGET.DeviceType = "PhidgetRFID" Then
        'device attached was RFID reader, set reader object
     '   Set phRFID = PHIDGET

        'reset all the outputs to OFF
    '    For lOutput = 0 To phRFID.NumOutputs - 1
 '           phRFID.OutputState(lOutput) = False
   '     Next lOutput

        'turn on RFID reader
  '      phRFID.OutputState(3) = True

        'turn on surface mount LED
 '       phRFID.OutputState(2) = True
'    End If
'End Sub

'Private Sub phRFID_OnTag(ByVal TagNumber As String)
    'set txtTagID value



'End Sub


'Agregado. Rutina para detectar etiquetas deslizadas sobre el sistema lector RFID
Private Sub Timer1_Timer()
    ReDim Buffer(48)
    BufferSize = 6
    Buffer = HIDComm1.ReadFrom(BufferSize)
    If BufferSize < 6 Then
    Exit Sub
    End If

    texto = Format(Hex(Buffer(0)), "00") & Format(Hex(Buffer(1)), "00") & _
    Format(Hex(Buffer(2)), "00") & Format(Hex(Buffer(3)), "00") & _
    Format(Hex(Buffer(4)), "00")

    txtTagID.Text = texto

End Sub

Private Sub txtTagID_KeyPress(KeyAscii As Integer)
     'test for acceptable characters (numbers and letters only)

    If (KeyAscii >= 48 And KeyAscii <= 57) Or (KeyAscii >= 65 And KeyAscii <= 90) Or _
    (KeyAscii >= 97 And KeyAscii <= 122) Or KeyAscii = 8 Then

        'key is acceptable
    Else
        KeyAscii = 0
    End If
End Sub
```

13.2.3 Programa para la marcación de horas de trabajo

Formulario de control de empleados.

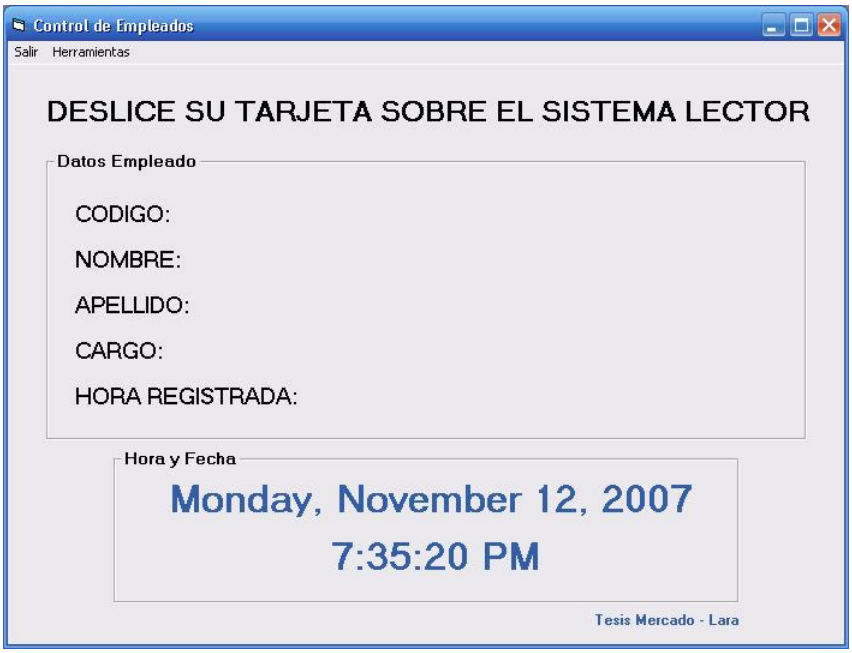


Figura 13.2.6 Formulario de control de empleados

```
'Definicion de Variables a utilizar
Option Explicit
Dim a As Integer
Private cnn As ADODB.Connection
Private rst As ADODB.Recordset
Private dt As ADODB.Recordset
Dim sSelect As String
Dim cn As ADODB.Connection
Dim rs As ADODB.Recordset
Dim rsEmpleados As ADODB.Recordset
Dim StrSql As String

'Codigo asociado al boton de Salir. Al hacer clic sobre este boton, se desconecta el sistema RFID
'y se cierra el programa
Private Sub Apagar_Click()
HIDComm1.Disconnect
Me.Timer1inicio.Enabled = False
Me.Timer2inicio.Enabled = False
Me.Timerhora.Enabled = False
End
End Sub

'Codigo asociado al cargarse el formulario principal
'Llamar la rutina de inicio (start), habilitar el timer de lectura USB, Esconder la informacion de empleado
'Setear la base de datos a utilizar
Private Sub Form_Load()
Call start
Timer1inicio.Enabled = True
Nombre.Visible = False
Apellido.Visible = False
cargo.Visible = False
'Me.tag1.Text = ""
tag1.Visible = False
Timerhora.Enabled = True
Set rsEmpleados = New ADODB.Recordset
Me.Nombre.Text = ""
HIDComm1.TimeOut = 100
End Sub
```

```
Private Sub Form_Terminate()
HIDComm1.Uninit
HIDComm1.Disconnect
Me.Timer1inicio.Enabled = False
Me.Timer2inicio.Enabled = False
Me.Timerhora.Enabled = False
rs.Close
End
End Sub

'Rutina que escribe la fecha y hora de entrada en la tabla registro
Private Sub HorEntrada_Click()
   If Me.Text1.Text <> "" Then
      sSelect = "INSERT INTO Registro([HoraEntrada], [HoraSalida], [NombreEmpleado], " _
         & "[ApellidoEmpleado],[TagEmpleado], [Salario]) Values('" & Format(Now, "dd/mm/yyyy HH:mm:ss") & "', '" & Format(Now,
"dd/mm/yyyy HH:mm:ss") & "', '" & Me.Nombre.Text & "', " _
         & "'" & Me.Apellido.Text & "', '" & Me.tag1.Text & "', " & Format(Me.txtSalario.Text, "#####0.##") & ") "
      cn.Execute sSelect
      Me.Label4.Caption = "HORA DE ENTRADA: "
      Me.hora.Text = Now
      Beep
      Timer2inicio = True
   Else
      MsgBox "Consulte antes un empleado", vbInformation, "Informacion"
   End If
End Sub

'Rutina que escribe la fecha y hora de salida en la tabla registro
Private Sub HorSalida_Click()
   If Me.Text1.Text <> "" Then
      sSelect = "INSERT INTO Registro([HoraEntrada], [HoraSalida], [NombreEmpleado], " _
         & "[ApellidoEmpleado],[TagEmpleado],[Salario]) Values('" & Format(Now, "dd/mm/yyyy HH:mm:ss") & "', '" & Format(Now, "dd/mm/yyyy
HH:mm:ss") & "', '" & Me.Nombre.Text & "', " _
         & "'" & Me.Apellido.Text & "', '" & Me.tag1.Text & "', " & Format(Me.txtSalario.Text, "#####0.##") & ") "
      cn.Execute sSelect
      Me.Label4.Caption = "HORA DE SALIDA: "
      Me.hora.Text = Now
      Beep
      Timer2inicio = True
   Else
      MsgBox "Consulte antes un empleado", vbInformation, "Informacion"
   End If
End Sub

'Rutina que activa el formulario de mantenimiento de empleados
Private Sub mtto_Click()
Mantenimiento.Show
Timer1inicio.Enabled = False
Timer2inicio.Enabled = False
Mantenimiento.Timer1.Enabled = True
End Sub

'Rutina que activa el formulario de Reporte de empleados
Private Sub reporte_Click()
Reportes.Show
Me.Timer1inicio.Enabled = False
Me.Timer2inicio.Enabled = False
End Sub

'Codigo asociado al boton de Salir. Al hacer clic sobre este boton, se desconecta el sistema RFID
'y se cierra el programa
Private Sub salir_Click()
HIDComm1.Uninit
HIDComm1.Disconnect
Me.Timer1inicio.Enabled = False
Me.Timer2inicio.Enabled = False
Me.Timerhora.Enabled = False
rs.Close
End
End Sub


'Rutina para lectura de sistema RFID. Variables necesarias para la adquisicion de datos por USB
Private Sub Timer1inicio_Timer()
```

```
'variables para leer HIDComm
Dim buffer() As Byte
Dim BufferSize As Long
ReDim buffer(64)
Dim a, b, c, d, i As Single

'Leyendo usb
HIDComm1.Connect 'Conectarse al 1er dispositivo
BufferSize = 6   'Cantidad de bytes a aceptar desde el control en cada transaccion
'Guardar los datos recibios en el arreglo buffer
buffer = HIDComm1.ReadFrom(BufferSize)

If BufferSize < 6 Then
Exit Sub
End If

'Leyendo posiciones
Dim byte0, byte1, byte2, byte3, byte4, byte5 As Single
Dim cad0, cad1, cad2, cad3, cad4, cad5, cadena As String
'**************************************
'Arreglo de datos que vienen del control
byte0 = buffer(0)
byte1 = buffer(1)
byte2 = buffer(2)
byte3 = buffer(3)
byte4 = buffer(4)
byte5 = buffer(5)

'Dandole formato a la informacion obtenida por USB
cad0 = CStr(Format(HEX(byte0), "00"))
cad1 = CStr(Format(HEX(byte1), "00"))
cad2 = CStr(Format(HEX(byte2), "00"))
cad3 = CStr(Format(HEX(byte3), "00"))
cad4 = CStr(Format(HEX(byte4), "00"))
cad5 = CStr(Format(HEX(byte5), "00"))

cadena = cad0 & cad1 & cad2 & cad3 & cad4


Text1.Text = cadena
If cadena <> "000000" Then

   StrSql = "select E.TagEmpleado, E.NOmbreEmpleado, E.ApellidoEmpleado, CA.CargoEmpleado, " _
      & "getdate() as HoraEntrada, SE.SueldoEmpleado from DatosEmpleado E, CargoEmpleado CA, " _
      & "SueldoEmpleado SE where E.tagEmpleado=ca.TagEmpleado AND E.TagEmpleado=SE.TagEmpleado " _
      & "and e.TagEmpleado='" & cadena & "' "
   If rsEmpleados.State = 1 Then
     rsEmpleados.Close
   End If
   rsEmpleados.Open StrSql, cn, adOpenForwardOnly, adLockReadOnly
   If rsEmpleados.EOF Then

      MsgBox "Codido de Empleado no encontrado", vbExclamation, "Error"

      Exit Sub
   Else

      Nombre.Visible = True
      Me.Nombre.Text = rsEmpleados.Fields("NombreEmpleado").Value
      Apellido.Visible = True
      Me.Apellido.Text = rsEmpleados.Fields("ApellidoEmpleado").Value
      cargo.Visible = True
      Me.cargo.Text = rsEmpleados.Fields("CargoEmpleado").Value
      tag1.Visible = True
      Me.tag1.Text = rsEmpleados.Fields("TagEmpleado").Value
      Me.hora.Visible = True
      Me.hora.Text = rsEmpleados.Fields("HoraEntrada").Value
      Me.txtSalario.Text = rsEmpleados.Fields("SueldoEmpleado").Value

      'MsgBox "Registro Valido", vbInformation, "Tesis Rmercado"
   End If
   rsEmpleados.Close
   End If
'Detectar si es manana o tarde
If (Format(Time, "AM/PM")) = "AM" Then
```

```
Call HorEntrada_Click
Else
Call HorSalida_Click
End If
HIDComm1.TimeOut = 100

End Sub

'Timer para hacer visible la informacion del empleado por unos segundos
Private Sub Timer2inicio_Timer()
Nombre.Visible = False
Apellido.Visible = False
cargo.Visible = False
tag1.Visible = False
hora.Visible = False
Me.Label4.Caption = "HORA REGISTRADA:"
Timer2inicio.Enabled = False
Timer1inicio.Enabled = True
End Sub

'Rutina para inicializar la conexion hacia la base de datos.
Sub start()

    Set cn = New ADODB.Connection
    Dim sSelect As String

    cn.Open "Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security Info=False;Initial Catalog=EMPLEADOS;Data Source=TOSHIBA"

sSelect = "SELECT PC.IDEmpleado, PC.NombreEmpleado, PC.ApellidoEmpleado, PC.DireccionEmpleado, PC.Telefono, " _
    & "PC.DUIEmpleado, PC.NITEmpleado, PC.SeguroSocial, HR.CargoEmpleado, HR.FuncionEmpleado, HR.ProfesionEmpleado, " _
    & "EH.SueldoEmpleado, EH.HorasExtras, EH.FuncionEmpleado, EH.CuentaEmpleado, TY.Tagempleado " _
    & " FROM dbo.DatosEmpleado AS PC INNER JOIN dbo.CARGOEMPLEADO AS HR ON PC.TagEmpleado = HR.TagEmpleado " _
    & " INNER JOIN dbo.SueldoEmpleado AS EH ON PC.TagEmpleado = EH.TagEmpleado INNER JOIN " _
    & "dbo.TagEmpleado AS TY ON PC.TagEmpleado = TY.TagEmpleado ORDER BY PC.IDEmpleado "

    ' El recordset para acceder a los datos

    Set rs = New ADODB.Recordset

    rs.Open sSelect, cn, adOpenDynamic, adLockOptimistic

    'seteando tabla 1
    Set Nombre.DataSource = rs
    Set id.DataSource = rs
    Set Apellido.DataSource = rs
    'seteando tabla 2
    Set cargo.DataSource = rs
    'seteando tabla 4
    Set tag1.DataSource = rs

    'asignado todos los tetxbox
    Nombre.DataField = "NombreEmpleado"
    id.DataField = "IDEmpleado"
    Apellido.DataField = "ApellidoEmpleado"
    cargo.DataField = "CargoEmpleado"
    tag1.DataField = "tagempleado"
'fin de conexion
End Sub

'Timer para presentar fecha y hora constantemente en el formulario principal
Private Sub Timerhora_Timer()
horareal1.Caption = Format(Date, "long date")
fechareal1.Caption = Time
End Sub

'Rutina para insertar informacion en registro de empleados
'Sub tabla1()
'sSelect = "Insert Into Registro " & "(HoraEntrada, HoraSalida, NombreEmpleado, ApellidoEmpleado," & "TagEmpleado) Values ('" & hora & "','" &
horasalida & "','" & Nombre & "','" & Apellido & "','" & tag1 & "')"
'cn.Execute sSelect
'End Sub

'Rutina que se ejecuta cuando se cierra un formulario (boton X)
Private Sub Form_Unload(Cancel As Integer)
HIDComm1.Uninit
```

```
HIDComm1.Disconnect
Me.Timer1inicio.Enabled = False
Me.Timer2inicio.Enabled = False
Me.Timerhora.Enabled = False
rs.Close
End
End Sub
```

Formulario de Mantenimiento de Empleados

Figura 13.2.7 Formulario de control de empleados

```
'Definicion de Variables a utilizar
Option Explicit
Dim a As Integer
Private cnn As ADODB.Connection
Private rst As ADODB.Recordset
Private dt As ADODB.Recordset
Dim sSelect As String
Dim cn As ADODB.Connection
Dim rs As ADODB.Recordset

'Rutina para limpiar todos los campos de empleado
Sub LimpiarControles()
    '  Este procedimiento limpia los controles del formulario de mantenimiento de empleados
    'Me.ID.Text = ""
    Me.nombre.Text = ""
    Me.Apellido.Text = ""
    Me.Direccion.Text = ""
    Me.Telefono.Text = ""
    Me.DUI.Text = ""
    Me.nit.Text = ""
    Me.Seguro.Text = ""
    Me.Cargo.Text = ""
    Me.funcion.Text = ""
    Me.HorasExtra.Text = ""
    Me.sueldo.Text = ""
    Me.profesion.Text = ""
    Me.numerocuenta.Text = ""
    Me.tag1.Text = ""
End Sub
'Rutina que se ejecuta al presionar el boton Anterior
'Muestra la informacion de un empleado previo al que se muestra en pantalla
```

```
Private Sub anterior_Click()
   If rs.BOF Then
      MsgBox "Este es el Primer Registro.", vbInformation, "Registros"
      rs.MoveLast
   Else
      rs.MovePrevious
   End If
End Sub

'Rutina que se ejecuta al presionar el boton borrar.
'Borra un registro de empleado especifico
Private Sub borrar_Click()
If Me.nombre.Text = "" Or Me.Apellido.Text = "" Or Me.tag1.Text = "" Or Me.Seguro.Text = "" Or Me.nit.Text = "" Or Me.DUI.Text = "" Then
MsgBox "No se puede borrar un registro vacio o incompleto.", vbInformation, "Error"
Exit Sub
End If
   rs.Delete
   MsgBox "Registro eliminado", vbInformation, "Borrando"
   Call LimpiarControles
   Call Conectar_Click
End Sub

'Rutina que se ejecuta al presionar el boton guardar.
'Guarda la informacion de un nuevo empleado en el registro
Private Sub btnGuardar_Click()
If Me.nombre.Text = "" Or Me.Apellido.Text = "" Or Me.tag1.Text = "" Or Me.Seguro.Text = "" Or Me.nit.Text = "" Or Me.DUI.Text = "" Or
Me.sueldo.Text = "" Then
MsgBox "Faltan campos por llenar.", vbExclamation, "Error"
Exit Sub
End If
   Dim StrSql As String
   Dim rsEmpleados As ADODB.Recordset
   Set rsEmpleados = New ADODB.Recordset

   StrSql = "select count(*) CantTag from DatosEmpleado where TagEmpleado='" & Me.tag1.Text & "' "
   rsEmpleados.Open StrSql, cn, adOpenForwardOnly, adLockReadOnly
   If rsEmpleados.Fields("CantTag") > 0 Then
      MsgBox "Este TAG ya esta asignado a otro empleado....", vbInformation, "Informacion"
      Me.tag1.Text = ""
      Exit Sub
   Else
       sSelect = "Insert Into DatosEmpleado " & _
      "(NombreEmpleado,ApellidoEmpleado,DireccionEmpleado,Telefono,DUIEmpleado,NITEmpleado," & _
      "SeguroSocial, TagEmpleado ) Values ('" & nombre & "','" & Apellido & _
      "','" & Direccion & "','" & Telefono & "','" & DUI & "','" & nit & "','" & Seguro & "', '" & Me.tag1.Text & "' )"

      cn.Execute sSelect

      sSelect = "Insert Into CargoEmpleado " & _
      "(CargoEmpleado,FuncionEmpleado," & _
      "ProfesionEmpleado, TagEmpleado) Values ('" & Cargo & "','" & funcion & _
      "','" & profesion & "', '" & Me.tag1.Text & "'  )"
      cn.Execute sSelect

      sSelect = "Insert Into SueldoEmpleado " & _
      "(SueldoEmpleado,HorasExtras," & _
      "FuncionEmpleado, CuentaEmpleado, TagEmpleado) Values ('" & sueldo & "','" & HorasExtra & _
      "','" & funcionempleado & "','" & numerocuenta & "', '" & Me.tag1.Text & "' )"
      cn.Execute sSelect

      sSelect = "Insert Into TagEmpleado " & _
      "(TagEmpleado) Values ('" & tag1 & "')"
      cn.Execute sSelect

       MsgBox "Registro Guardado exitosamente", vbInformation, "Guardado"
       Me.anterior.Enabled = True
       Me.siguiente.Enabled = True
       Me.primero.Enabled = True
       Me.ultimo.Enabled = True
   End If
   Call Conectar_Click

End Sub
'Rutina que se ejecuta al presionar el boton borrar tabla
'Borra todos los registros de empleados contenidos en la tabla.
```

```
Private Sub cmdborrartabla_Click()

Dim respuesta As Integer

respuesta = MsgBox("Esta seguro de realizar esta accion", vbYesNo, "Borrar")
If respuesta = vbYes Then
sSelect = "delete from DatosEmpleado"
   cn.Execute sSelect
   MsgBox "La informacion fue eliminada con éxito", vbInformation, "Borrado"
   Else
   Exit Sub
   End If
End Sub

'Rutina para conectar el sistema a la base de datos.
Private Sub Conectar_Click()
   'Set cn = New ADODB.Connection
   'Dim sSelect As String
   'Dim rs As ADODB.Recordset

'    cn.Open "Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security Info=False;Initial Catalog=EMPLEADOS;Data Source=TOSHIBA"

  ' sSelect = "SELECT PC.IDEmpleado, PC.NombreEmpleado, PC.ApellidoEmpleado, PC.DireccionEmpleado, PC.Telefono, PC.DUIEmpleado,
PC.NITEmpleado, PC.SeguroSocial, HR.CargoEmpleado, HR.FuncionEmpleado, HR.ProfesionEmpleado, EH.SueldoEmpleado, EH.HorasExtras,
EH.FuncionEmpleado, EH.CuentaEmpleado, TY.TagEmpleado FROM DATOSEMPLEADO  AS PC INNER JOIN CargoEmpleado AS HR ON
PC.TagEmpleado = HR.TagEmpleado INNER JOIN SueldoEmpleado AS EH ON PC.TagEmpleado = EH.TagEmpleado INNER JOIN TagEmpleado AS
TY ON PC.TagEmpleado = TY.TagEmpleado ORDER BY PC.IDEmpleado"
  sSelect = "SELECT     TOP (100) PERCENT PC.TagEmpleado, PC.IDEmpleado, PC.NombreEmpleado, PC.ApellidoEmpleado,
PC.DireccionEmpleado, PC.Telefono, PC.DUIEmpleado, " _
      & "PC.NITEmpleado, PC.SeguroSocial, HR.CargoEmpleado, HR.FuncionEmpleado, HR.ProfesionEmpleado, EH.SueldoEmpleado, " _
      & "EH.CuentaEmpleado " _
      & "FROM         dbo.DatosEmpleado AS PC INNER JOIN " _
      & "dbo.CARGOEMPLEADO AS HR ON PC.TagEmpleado = HR.TagEmpleado INNER JOIN " _
      & "dbo.SueldoEmpleado AS EH ON PC.TagEmpleado = EH.TagEmpleado " _
      & "ORDER BY PC.TagEmpleado "

   'Set rs = New ADODB.Recordset

   If rs.State = 1 Then
      Set rs = New ADODB.Recordset
      'rs.Close

   End If

      rs.Open sSelect, cn, adOpenStatic, adLockOptimistic

   'seteando tabla 1
   Set nombre.DataSource = rs
   Set ID.DataSource = rs
   Set Apellido.DataSource = rs
   Set Direccion.DataSource = rs
   Set Telefono.DataSource = rs
   Set DUI.DataSource = rs
   Set nit.DataSource = rs
   Set Seguro.DataSource = rs
   'seteando tabla 2
   Set Cargo.DataSource = rs
   Set funcion.DataSource = rs
   Set profesion.DataSource = rs
   'seteando tabla 3
   Set sueldo.DataSource = rs
   Set HorasExtra.DataSource = rs
   Set numerocuenta.DataSource = rs
   'seteando tabla 4
   Set tag1.DataSource = rs


   'asignado todos los tetxbox
   nombre.DataField = "NombreEmpleado"
   ID.DataField = "IDEmpleado"
   Apellido.DataField = "ApellidoEmpleado"
   Direccion.DataField = "DireccionEmpleado"
   Telefono.DataField = "Telefono"
   DUI.DataField = "DUIEmpleado"
   nit.DataField = "NITEmpleado"
```

```
    Seguro.DataField = "SeguroSocial"
    Cargo.DataField = "CargoEmpleado"
    funcion.DataField = "FuncionEmpleado"
    profesion.DataField = "ProfesionEmpleado"
    sueldo.DataField = "SueldoEmpleado"
'    HorasExtra.DataField = "HorasExtras"
    numerocuenta.DataField = "CuentaEmpleado"
    tag1.DataField = "tagempleado"
End Sub

'Rutina que se ejecuta al cargar el formulario
'Llamar a la rutina conectar
Private Sub Form_Load()
    Set rs = New ADODB.Recordset
    Set cn = New ADODB.Connection
    cn.Open "Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security Info=False;Initial Catalog=EMPLEADOS;Data Source=TOSHIBA"
    Timer1.Enabled = True
    Call Conectar_Click
 End Sub

'Rutina que se ejecuta al presionar el boton modificar
'Guarda en el registro de empleados la informacion que ha sido modificada
Private Sub modificar_Click()
If Me.nombre.Text = "" Or Me.Apellido.Text = "" Or Me.tag1.Text = "" Or Me.Seguro.Text = "" Or Me.nit.Text = "" Or Me.DUI.Text = "" Then
MsgBox "Faltan campos por llenar.", vbExclamation, "Error"
Exit Sub
End If

    sSelect = "Update DatosEmpleado " & _
        "set NombreEmpleado ='" & nombre & "', ApellidoEmpleado='" & Apellido & "', " _
        & "DireccionEmpleado='" & Direccion & "', Telefono='" & Telefono & "', " _
        & "DUIEmpleado='" & DUI & "', NITEmpleado='" & nit & "', " _
        & "SeguroSocial='" & Seguro & "' " _
        & "WHERE TagEmpleado='" & Me.tag1.Text & "' "

        cn.Execute sSelect

        sSelect = "UPDATE CargoEmpleado " _
            & "SET CargoEmpleado='" & Cargo & "',FuncionEmpleado='" & funcion & "', " _
            & "ProfesionEmpleado='" & profesion & "' " _
            & "WHERE TagEmpleado='" & Me.tag1.Text & "' "
        cn.Execute sSelect

        sSelect = "UPDATE SueldoEmpleado " _
            & "SET SueldoEmpleado= " & sueldo & ", FuncionEmpleado='" & funcionempleado & "', " _
            & "CuentaEmpleado='" & numerocuenta & "' " _
            & "WHERE TagEmpleado='" & Me.tag1.Text & "' "

        cn.Execute sSelect
        Call Conectar_Click

    'rs.Update
     MsgBox "Registro Modificado", vbInformation, "Modificando"
     Control_emple.HIDComm1.TimeOut = 100
End Sub

'Rutina que se ejecuta al presionar boton de nuevo.
'Limpia los controles y desabilita los controles de desplazamiento para empleados
Private Sub nuevo_Click()
Me.anterior.Enabled = False
Me.siguiente.Enabled = False
Me.primero.Enabled = False
Me.ultimo.Enabled = False
Call LimpiarControles
End Sub

'Rutina que se ejecuta al presionar el boton primero.
'Muestra la informacion del primer empleado contenido en el registro
Private Sub primero_Click()
rs.MoveFirst
End Sub


'Rutina que se ejecuta al presionar el boton siguiente.
'Muestra la informacion de un empleado posterior al que se muestra en pantalla
```

```
Private Sub siguiente_Click()
   If rs.EOF Then
      MsgBox "Este es el Último Registro.", vbInformation, "Registros"
      rs.MoveFirst
   Else
      rs.MoveNext
   End If
End Sub

Private Sub sueldo_Validate(Cancel As Boolean)
   If Not IsNumeric(Me.sueldo.Text) Then
      MsgBox "Este campo solo acepta números... Favor vuelva a ingresar el salario..", vbCritical, "Error"
      Me.sueldo.Text = ""
      Cancel = True
   End If
End Sub

'Rutina para lectura de datos de sistema RFID
'Se obtienen los datos del USB
Private Sub Timer1_Timer()
Dim buffer() As Byte
Dim BufferSize As Long
ReDim buffer(64)
Dim a, b, c, d, i As Single

Control_emple.HIDComm1.Connect 'Conectarse al 1er dispositivo: Control
BufferSize = 6   'Cantidad de bytes a aceptar desde el control en cada transaccion
'Guardar los datos recibios en el arreglo buffer
buffer = Control_emple.HIDComm1.ReadFrom(BufferSize)

If BufferSize < 6 Then
Exit Sub
End If

'Leyendo posiciones
Dim byte0, byte1, byte2, byte3, byte4, byte5 As Single
Dim cad0, cad1, cad2, cad3, cad4, cad5, cadena As String
'****************************************
'Arreglo de datos que vienen del control
byte0 = buffer(0)
byte1 = buffer(1)
byte2 = buffer(2)
byte3 = buffer(3)
byte4 = buffer(4)
byte5 = buffer(5)

cad0 = CStr(Format(HEX(byte0), "00"))
cad1 = CStr(Format(HEX(byte1), "00"))
cad2 = CStr(Format(HEX(byte2), "00"))
cad3 = CStr(Format(HEX(byte3), "00"))
cad4 = CStr(Format(HEX(byte4), "00"))
cad5 = CStr(Format(HEX(byte5), "00"))

tag1.Text = cad0 & cad1 & cad2 & cad3 & cad4
Control_emple.HIDComm1.TimeOut = 100
End Sub

'Rutina que se ejecuta al presionar el boton primero.
'Muestra la informacion del primer empleado contenido en el registro
Private Sub ultimo_Click()
rs.MoveLast
End Sub

'Sub tabla1()
'sSelect = "Insert Into DatosEmpleado " & _
'"(NombreEmpleado,ApellidoEmpleado,DireccionEmpleado,Telefono,DUIEmpleado,NITEmpleado," & _
'"SeguroSocial) Values ('" & nombre & "','" & Apellido & _
"','" & Direccion & "','" & Telefono & "','" & DUI & "','" & nit & "','" & Seguro & "')"
'
'cn.Execute sSelect
'End Sub
'
'Sub tabla2()
'sSelect = "Insert Into CargoEmpleado " & _
'"(CargoEmpleado,FuncionEmpleado," & _
```

```
'"ProfesionEmpleado) Values ('" & Cargo & "','" & funcion & _
"','" & profesion & "')"
'
'cn.Execute sSelect
'End Sub
'
'Sub tabla3()
'sSelect = "Insert Into SueldoEmpleado " & _
'"(SueldoEmpleado,HorasExtras," & _
'"FuncionEmpleado, CuentaEmpleado) Values ('" & sueldo & "','" & HorasExtra & _
"','" & numerocuenta & "','" & funcionempleado & "')"
'cn.Execute sSelect
'End Sub
'
'Sub tabla4()
'sSelect = "Insert Into TagEmpleado " & "(Byte3," & "Byte4) Values ('" & byte3 & "','" & byte4 & "')"
'cn.Execute sSelect
'End Sub

'Rutina para volver al formulario principal
Private Sub volverinicio_Click()
Mantenimiento.Hide
Timer1.Enabled = False
Control_emple.Timer1inicio = True
Control_emple.Timerhora = True
Call Conectar_Click
Me.anterior.Enabled = True
Me.siguiente.Enabled = True
Me.primero.Enabled = True
Me.ultimo.Enabled = True
End Sub
```

Formulario de Reportes de Empleados

Figura 13.2.8 Formulario de control de empleados

```
'Definicion de variables a utilizar
Option Explicit
Dim a As Integer
Private cnn As ADODB.Connection
Private rst As ADODB.Recordset
Private dt As ADODB.Recordset
Dim sSelect As String
```

```
Dim cn As ADODB.Connection
Dim rs As ADODB.Recordset

'Rutina que se ejecuta al presionar el boton borrar registro
'Se borra todo el registro de horas de trabajo de los empleados
Private Sub cmdborrarregisgtro_Click()
Dim respuesta As Integer

respuesta = MsgBox("Esta seguro de realizar esta accion", vbYesNo, "Borrar")
If respuesta = vbYes Then
sSelect = "delete from Registro"
   cn.Execute sSelect
   Me.MSHFlexGrid2.Clear
   'MsgBox "El registro fue eliminado con éxito", vbInformation, "Borrado"
   'Call Command1_Click
   Else
   Exit Sub
   End If
End Sub

'Rutina que hace la conexion a la base de datos
Private Sub Command1_Click()

   Set cn = New ADODB.Connection

   Dim sSelect As String
   Dim i, j, z As Integer
   Dim HOR As Double
   Dim HEX, SalHora, TotHorEmp, ValorHorEmp, TotalHoras, ValorTotal As Double

   cn.Open "Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security Info=False;Initial Catalog=EMPLEADOS;Data Source=TOSHIBA"

 sSelect = "SELECT TagEmpleado as Cod_Empleado, NombreEmpleado, ApellidoEmpleado, MIN(CONVERT(smalldatetime, HoraEntrada, 103)) AS
FechaEntrada, " _
   & "MAX(CONVERT(smalldatetime, horasalida, 103)) As FechaSalida, MIN(Salario) as Salario " _
   & "From dbo.Registro WHERE HoraEntrada>='" & Format(Me.DTPicker1.Value, "dd/mm/yyyy") & "' and horasalida<='" &
Format(Me.DTPicker2.Value, "dd/mm/yyyy") & " 23:59:00' " _
   & "GROUP BY TagEmpleado, NombreEmpleado, ApellidoEmpleado, " _
   & "CONVERT(VARCHAR(10), HoraEntrada, 103), CONVERT(VARCHAR(10), HoraSalida, 103) ORDER BY NombreEmpleado "

   Set rs = New ADODB.Recordset

   rs.Open sSelect, cn, adOpenStatic

   'Me.MSHFlexGrid2.Rows = 0

   ' Permitir redimensionar las columnas
   Me.MSHFlexGrid1.AllowUserResizing = flexResizeColumns
   Me.MSHFlexGrid2.AllowUserResizing = flexResizeColumns

   ' Asignar el recordset al FlexGrid
   Set MSHFlexGrid1.DataSource = rs
   i = 0 ' columnas
   j = 1 ' filas
   If Not rs.EOF Then
      z = rs.RecordCount + 2
      Me.MSHFlexGrid2.Rows = z
      TotHorEmp = 0
      ValorHorEmp = 0
      TotalHoras = 0
      ValorTotal = 0
   Else
      MsgBox "NO hay registros para el periodo especificado"
      Me.MSHFlexGrid2.Clear
      Exit Sub
   End If
'    MSHFlexGrid2.AllowUserResizing = flexResizeColumns
  ' Me.MSHFlexGrid2.Rows = 5
   Me.MSHFlexGrid2.Cols = 11

   'Me.MSHFlexGrid2.ColHeader(0) = flexColHeaderOn
   'Me.MSHFlexGrid2.Row = 1

   Me.MSHFlexGrid2.Row = 0
   Me.MSHFlexGrid2.Col = 0
```

```
    Me.MSHFlexGrid2.Text = "Codigo"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 1
    Me.MSHFlexGrid2.Text = "Nombres"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 2
    Me.MSHFlexGrid2.Text = "Apellidos"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 3
    Me.MSHFlexGrid2.Text = "Hora Entrada"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 4
    Me.MSHFlexGrid2.Text = "Hora Salida"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 5
    Me.MSHFlexGrid2.Text = "H. Normales"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 6
    Me.MSHFlexGrid2.Text = "Valor/H. Normal"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 7
    Me.MSHFlexGrid2.Text = "H. Extras"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 8
    Me.MSHFlexGrid2.Text = "Valor/H. Extra"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 9
    Me.MSHFlexGrid2.Text = "Total Horas"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter
    Me.MSHFlexGrid2.Row = 0
    Me.MSHFlexGrid2.Col = 10
    Me.MSHFlexGrid2.Text = "Valor Total"
    Me.MSHFlexGrid2.CellAlignment = flexAlignCenterCenter

    Dim colum As Integer
For colum = 0 To 4
Me.MSHFlexGrid2.Col = colum
Me.MSHFlexGrid2.ColAlignment(colum) = flexAlignLeftCenter
Next colum

For colum = 5 To 10
Me.MSHFlexGrid2.Col = colum
Me.MSHFlexGrid2.ColAlignment(colum) = flexAlignCenterCenter
Next colum

    'Me.MSHFlexGrid2.ColHeaderCaption(2, 0) = "Cod_Empleado"
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 1) = "Nombres"
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 2) = "Apellidos"
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 3) = "Hora Entrada"
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 4) = "Hora Salida"
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 5) = "H. ORD"
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 6) = "Valor HOR."
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 7) = "H. Ext."
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 8) = "Valor HEX."
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 9) = "Total Hrs."
    'Me.MSHFlexGrid2.ColHeaderCaption(0, 10) = "Valor Total"

    While Not rs.EOF
        Me.MSHFlexGrid2.Col = i
        Me.MSHFlexGrid2.Row = j
        Me.MSHFlexGrid2.Text = rs.Fields("Cod_Empleado")

        i = i + 1
        Me.MSHFlexGrid2.Col = i
        Me.MSHFlexGrid2.Row = j
        Me.MSHFlexGrid2.Text = rs.Fields("NombreEmpleado")
```

```
        i = i + 1
        Me.MSHFlexGrid2.Col = i
        Me.MSHFlexGrid2.Row = j
        Me.MSHFlexGrid2.Text = rs.Fields("ApellidoEmpleado")

        i = i + 1
        Me.MSHFlexGrid2.Col = i
        Me.MSHFlexGrid2.Row = j
        Me.MSHFlexGrid2.Text = rs.Fields("FechaEntrada")

        i = i + 1
        Me.MSHFlexGrid2.Col = i
        Me.MSHFlexGrid2.Row = j
        Me.MSHFlexGrid2.Text = rs.Fields("FechaSalida")

        HOR = (Hour(rs.Fields("FechaSalida") - rs.Fields("FechaEntrada")) + ((Minute(rs.Fields("FechaSalida") - rs.Fields("FechaEntrada"))) / 60)) - 1
'tiempo de almuerzo
        'HOR = Minute(rs.Fields("FechaSalida") - rs.Fields("FechaEntrada"))
        TotHorEmp = HOR
        TotalHoras = TotalHoras + TotHorEmp

        SalHora = rs.Fields("Salario") / 30 / 8
        If HOR > 8 Then
        'Horas Normales
           i = i + 1
           Me.MSHFlexGrid2.Col = i
           Me.MSHFlexGrid2.Row = j
           Me.MSHFlexGrid2.Text = Format(8, "standard")
        'Valor por horas normales
           i = i + 1
           Me.MSHFlexGrid2.Col = i
           Me.MSHFlexGrid2.Row = j
           Me.MSHFlexGrid2.Text = Format(8 * SalHora, "currency")
           ValorHorEmp = ValorHorEmp + (8 * SalHora)
        'Horas Extras
            i = i + 1
           Me.MSHFlexGrid2.Col = i
           Me.MSHFlexGrid2.Row = j
           Me.MSHFlexGrid2.Text = Format((HOR - 8), "standard")
        'Valor por horas extras
            i = i + 1
           Me.MSHFlexGrid2.Col = i
           Me.MSHFlexGrid2.Row = j
           Me.MSHFlexGrid2.Text = Format((HOR - 8) * SalHora * 2, "currency")
           ValorHorEmp = ValorHorEmp + ((HOR - 8) * SalHora * 2)

        Else
        'Horas normales
           i = i + 1
           Me.MSHFlexGrid2.Col = i
           Me.MSHFlexGrid2.Row = j
           Me.MSHFlexGrid2.Text = Format(HOR, "standard")
        'valor por horas normales
           i = i + 1
           Me.MSHFlexGrid2.Col = i
           Me.MSHFlexGrid2.Row = j
           Me.MSHFlexGrid2.Text = Format(HOR * SalHora, "currency")
           ValorHorEmp = ValorHorEmp + (HOR * SalHora)
        'Horas Extras
            i = i + 1
           Me.MSHFlexGrid2.Col = i
           Me.MSHFlexGrid2.Row = j
           Me.MSHFlexGrid2.Text = Format((0), "standard")
        'Valor por horas extras
            i = i + 1
           Me.MSHFlexGrid2.Col = i
           Me.MSHFlexGrid2.Row = j
           Me.MSHFlexGrid2.Text = Format((0) * SalHora * 2, "currency")
           ValorHorEmp = ValorHorEmp + ((0) * SalHora * 2)
        End If
        i = i + 1
        Me.MSHFlexGrid2.Col = i
        Me.MSHFlexGrid2.Row = j
        Me.MSHFlexGrid2.Text = Format(TotHorEmp, "standard")
```

```
        TotHorEmp = 0

        i = i + 1
        Me.MSHFlexGrid2.Col = i
        Me.MSHFlexGrid2.Row = j
        Me.MSHFlexGrid2.Text = Format(ValorHorEmp, "currency")
        ValorTotal = ValorTotal + ValorHorEmp
        ValorHorEmp = 0

        rs.MoveNext
        j = j + 1
        i = 0
    Wend

End Sub

'Rutina que se ejecuta al presionar el boton volver a inicio
'Se cierra el formulario de reportes y se vuelve al formulario principal
Private Sub Command2_Click()
Reportes.Visible = False
Control_emple.Timer1inicio.Enabled = True
Control_emple.Timer2inicio.Enabled = True
End Sub

Private Sub Command3_Click()
    Dim i As Long, j As Long
    Dim objExcel As Object
    Dim objWorkbook As Object
    On Error Resume Next ' por si se cierra Excel antes de cargar los datos
    Set objExcel = CreateObject("Excel.Application")
    objExcel.Visible = True
    Set objWorkbook = objExcel.Workbooks.Add
    For i = 0 To MSHFlexGrid2.Rows - 1
        MSHFlexGrid2.Row = i
        For j = 0 To MSHFlexGrid2.Cols - 1
            MSHFlexGrid2.Col = j
            objWorkbook.ActiveSheet.Cells(i + 1, j + 1).Value = MSHFlexGrid2.Text
        Next
    Next
    objExcel.Cells.Select
    objExcel.Selection.EntireColumn.AutoFit    ' Ancho de columna
    objExcel.Range("A1").Select
    'objExcel.ActiveWindow.SelectedSheets.PrintPreview ' Previsualizar informe
    Set objWorkbook = Nothing
    Set objExcel = Nothing

End Sub

'Rutina que se ejecuta al cargarse la ventana de reportes
'Llamada a rutina de conexion de base de datos
Private Sub Form_Activate()
'Call Command1_Click
End Sub

'Rutina que se ejecuta cuando se carga el formulario
'Redimensionar el grid para poder mostrar toda la informacion
Private Sub Form_Load()

MSHFlexGrid2.ColWidth(0) = 1600
MSHFlexGrid2.ColWidth(1) = 2200
MSHFlexGrid2.ColWidth(2) = 2500
MSHFlexGrid2.ColWidth(3) = 2700
MSHFlexGrid2.ColWidth(4) = 2700
MSHFlexGrid2.ColWidth(5) = 1650
MSHFlexGrid2.ColWidth(6) = 2100
MSHFlexGrid2.ColWidth(7) = 1300
MSHFlexGrid2.ColWidth(8) = 1800
MSHFlexGrid2.ColWidth(9) = 1500
MSHFlexGrid2.ColWidth(10) = 1600
End Sub
```

13.3 PROGRAMAS EN VISUAL C++

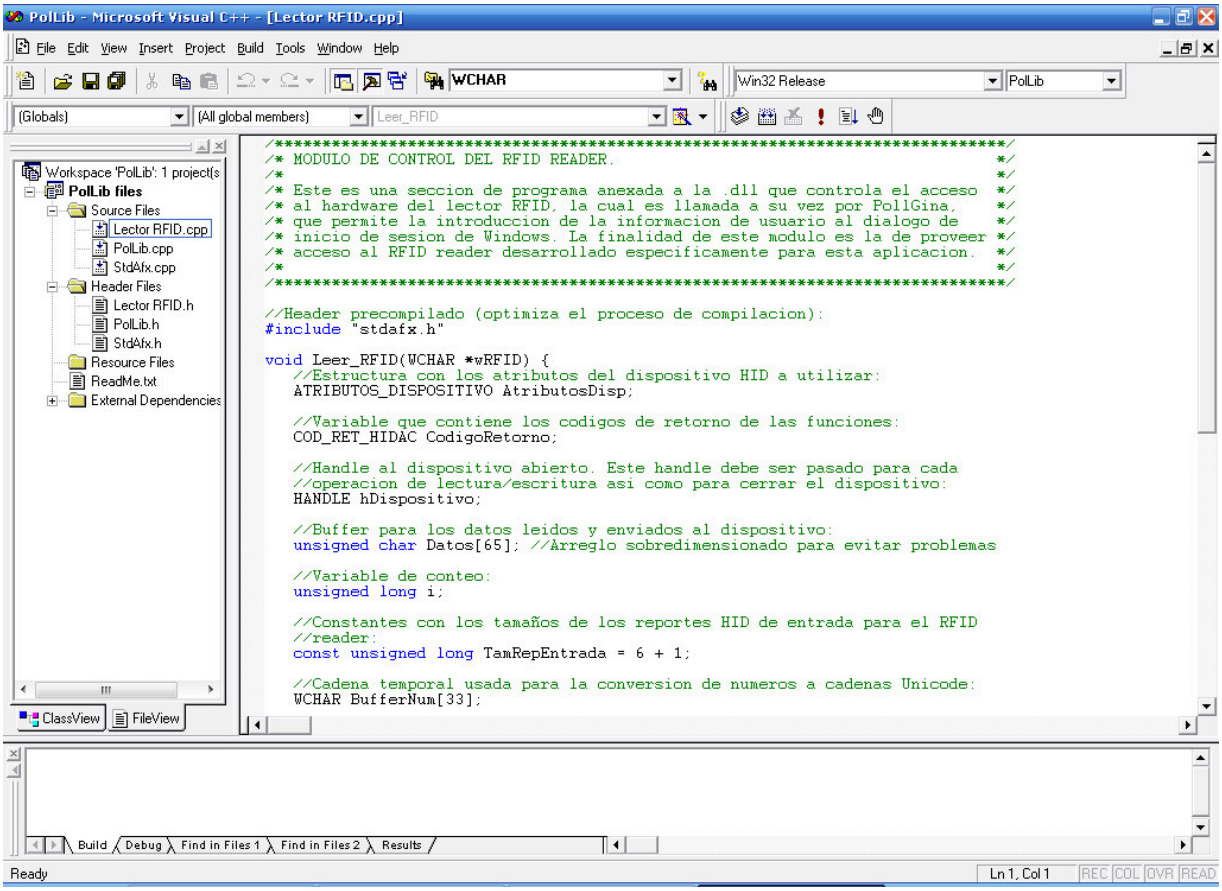


Figura 13.2.9 LIBRERÍA RFID PARA POLLGINA.

Archivos de Código

LECTOR RFID.CPP.

```
/*****************************************************************************/
/* MODULO DE CONTROL DEL RFID READER.                                        */
/*                                                                           */
/* Este es una seccion de programa anexada a la .dll que controla el acceso  */
/* al hardware del lector RFID, la cual es llamada a su vez por PollGina,    */
/* que permite la introduccion de la informacion de usuario al dialogo de    */
/* inicio de sesion de Windows. La finalidad de este modulo es la de proveer */
/* acceso al RFID reader desarrollado especificamente para esta aplicacion.  */
/*                                                                           */
/*****************************************************************************/

//Header precompilado (optimiza el proceso de compilacion):
#include "stdafx.h"

void Leer_RFID(WCHAR *wRFID) {
   //Estructura con los atributos del dispositivo HID a utilizar:
   ATRIBUTOS_DISPOSITIVO AtributosDisp;

   //Variable que contiene los codigos de retorno de las funciones:
   COD_RET_HIDAC CodigoRetorno;

   //Handle al dispositivo abierto. Este handle debe ser pasado para cada
   //operacion de lectura/escritura asi como para cerrar el dispositivo:
```

```
   HANDLE hDispositivo;

   //Buffer para los datos leidos y enviados al dispositivo:
   unsigned char Datos[65]; //Arreglo sobredimensionado para evitar problemas

   //Variable de conteo:
   unsigned long i;

   //Constantes con los tamaños de los reportes HID de entrada para el RFID
   //reader:
   const unsigned long TamRepEntrada = 6 + 1;

   //Cadena temporal usada para la conversion de numeros a cadenas Unicode:
   WCHAR BufferNum[33];

   //Fin de la declaracion de variables
   //---------------------------------
   //Inicio del codigo ejecutable

   //Primero se definen las caracteristicas del dispositivo:
   AtributosDisp.CriterioBusqueda = BUSCAR_VID | BUSCAR_PID | BUSCAR_VERSION;
   AtributosDisp.Vid = 0x0461;
   AtributosDisp.Pid = 0x0020;
   AtributosDisp.Version = 0x0100;

   //A continuacion se abre el dispositivo:
   CodigoRetorno = AbrirDispositivo(&AtributosDisp, &hDispositivo);
   if (CodigoRetorno != CRH_EXITO)
                        //En caso de falla al abrir el dispositivo, el proceso simplemente se
                        //cancela:
      return;

                     //Con el dispositivo abierto, se lee un reporte:
                     CodigoRetorno = RecibirReporteEntrada(hDispositivo, Datos, TamRepEntrada, 5000);
                     if (CodigoRetorno == CRH_EXITO) {
                        for (i=1; i<TamRepEntrada-1; i++) {
                                wsprintf(BufferNum, L"%.2x", Datos[i]);
                                wcscat(wRFID, BufferNum);
                        }
                        //Notese que el primer byte contiene el identificador del reporte, que
                        //regularmente es cero. El reporte real comienza desde el segundo byte.
      //Adicionalmente, el ultimo byte del reporte esta reservado como indice
      //de la lista de RFID's guardados en el firmware.
                     }

   //Finalmente, se cierra el dispositivo:
   CerrarDispositivo(hDispositivo);

   return;
}
```

POLLIB.CPP.

```
/*****************************************************************************/
/* Modulo principal de rfidPoll                                              */
/* Version modificada para utilizar el RFID reader en reemplazo  por el      */
/* Phidget                                                                   */
/*****************************************************************************/

#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <winsock2.h>
#include <windows.h>
#include <string.h>
#include <tchar.h>
#include <wchar.h>

#include "PolLib.h"
#include "Hidac.h"
#include "Lector RFID.h"

/* Removidos para eliminar la funcionalidad de phidget */
//HANDLE m_ThreadHandle;                          // Handle to the thread executing the reads
//unsigned long m_sThreadIdentifier;      // ID of the read thread

WCHAR rfId[12];                                         // Last id read from device
WCHAR *lastUsed;                                // Last id attempted to auth
```

```
// DllMain is called when the dll is first loaded/unloaded
BOOL APIENTRY DllMain( HANDLE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                                             )
{
    switch (ul_reason_for_call)
                      {
                         case DLL_PROCESS_ATTACH:
                         case DLL_THREAD_ATTACH:
                                 lastUsed = NULL;
                                 break;
                         case DLL_THREAD_DETACH:
                         case DLL_PROCESS_DETACH:
                                 if(lastUsed) {
                                         free(lastUsed);
                                         lastUsed = NULL;
                                 }
                                 break;
    }
    return TRUE;
}


/**
 * Read a string from the registry, specifically from root\base_key\sub_key\location
 */
WCHAR * regReadString(HKEY root, LPCTSTR base_key, LPCTSTR sub_key, LPCTSTR location) {
                      LONG lResult = 0;
                      HKEY hKey = NULL;
                      DWORD dwBytesRead = 6000;
                      WCHAR result[6000];
                      WCHAR *ret = NULL;
                      WCHAR full_key[10000];

                      memset(full_key,0,sizeof(full_key));
                      wsprintf(full_key,L"%s\\%s",base_key,sub_key);

                      lResult = RegOpenKeyEx(root,full_key,0,KEY_QUERY_VALUE,&hKey);

                      if(lResult == ERROR_SUCCESS) {
                         lResult = RegQueryValueEx(hKey,location,NULL,NULL,(LPBYTE)&result,&dwBytesRead);

                         if(lResult == ERROR_SUCCESS) {
                                 if(dwBytesRead > 0) {
                                         ret = _tcsdup(result);
                                 }
                         }
                         RegCloseKey(hKey);
                      }
                      return ret;
}

/* Removido para eliminar la funcionalidad de phidget */

/*
int __stdcall RFID_Handler(CPhidgetRFIDHandle RFID, void *userptr, unsigned char *buf)
{
                      wsprintf(rfId,L"%x%x%x%x%x%x%x%x%x%x\n", buf[0]/16,buf[0]%16,buf[1]/16,buf[1]%16,
buf[2]/16,buf[2]%16, buf[3]/16,buf[3]%16, buf[4]/16,buf[4]%16);
                      return 0;
}

// This thread runs in the background, executing successive reads until time is up
DWORD WINAPI RFID_ThreadedRead(LPVOID lpdwParam)
{
                      int result;
                      CPhidgetRFIDHandle RFID = (CPhidgetRFIDHandle )lpdwParam;
                      while (1)
                      {
                         result = CPhidgetRFID_read(RFID);
                         if (result) break;
                      }
                      return 0;
}
*/

// This is our entrypoint, where the PollGina calls us
POLLIB_API void pollFunction(LPTSTR loginUsername,LPTSTR loginPassword,LPTSTR loginDomain) {
```

```
                        int result = 0;
                        WCHAR *username = NULL, *password = NULL, *domain = NULL;

/* Codigo de inicializacion de Phidget removido */

/*                      PhidgetRFIDHandle RFID = 0;

                        CPhidgetRFID_create(&RFID);
                        if(CPhidget_open(RFID, -1)) {
                           return;
                        }
*/
                        // Clear any existing rfid data
                        memset(rfId,0,sizeof(rfId));

/* Inicializacion de la lectura de Phidget Removida */

/*
                        // Set the callback for read events
                        CPhidgetRFID_set_OnTag_Handler(RFID, RFID_Handler, NULL);
                        // Set the output state to 3
                        CPhidgetRFID_setOutputState(RFID, 3, true);

                        // Start our reader thread
                        m_ThreadHandle = CreateThread(NULL, 0, RFID_ThreadedRead,(void *)RFID, 0,
&m_sThreadIdentifier);
                        // Pause for 500 ms, then continue
                        WaitForSingleObject(m_ThreadHandle, 500);
*/

/* Codigo de lectura del RFID reader insertado aqui */
                        Leer_RFID(rfId);   //Llamada a la funcion en el modulo agregado
/* Fin del codigo insertado */

                        // If an id was read
                        if(_tcslen(rfId)) {
                           // And it does NOT match the last used one
                           if(lastUsed) {
                                   if(wcscmp(rfId,lastUsed) == 0) {
                                           // This is the same one as last time, ignore it
                                           goto cleanup;
                                   }
                           }

                           // Strip trailing \n that phidgets puts in...
                           if(wcsstr(rfId,L"\n")) {
                                   *(wcsstr(rfId,L"\n")) = '\0';
                           }

                           // Get appropriate values (if any) for this id
                           username =
regReadString(HKEY_LOCAL_MACHINE,L"Software\\rfidPoll",rfId,L"Username");
                           password =
regReadString(HKEY_LOCAL_MACHINE,L"Software\\rfidPoll",rfId,L"Password");
                           domain = regReadString(HKEY_LOCAL_MACHINE,L"Software\\rfidPoll",rfId,L"Domain");

                           // Username is required, the rest are optional, put the data in the buffers
provided by PollGina
                           if(username) {
                                   wsprintf(loginUsername,L"%s",username);

                                   if(password) {
                                           wsprintf(loginPassword,L"%s",password);
                                   }
                                   if(domain) {
                                           wsprintf(loginDomain,L"%s",domain);
                                   }
                           }
                        }

                        // Cleanup after ourselves.. free memory we've allocated, and leave
cleanup:
                        if(username) {
                           free(username);
                           username = NULL;
                        }
                        if(password) {
                           free(password);
                           password = NULL;
                        }
```

```
                        if(domain) {
                           free(domain);
                           domain = NULL;
                        }

/* Codigo de cierre de Phidget eliminado */

                        // Close the reader
//                      CPhidget_close(RFID);
                        // Delete our context
//                      CPhidget_delete((CPhidgetHandle )RFID);
}
```

STDAFX.CPP.

```
// stdafx.cpp : source file that includes just the standard includes
//                      PolLib.pch will be the pre-compiled header
//                      stdafx.obj will contain the pre-compiled type information

#include "stdafx.h"

// TODO: reference any additional headers you need in STDAFX.H
// and not in this file
```

Archivos de cabecera

LECTOR RFID.H.

```
extern void Leer_RFID(WCHAR *wRFID);
```

POLLIB.H.

```
// The following ifdef block is the standard way of creating macros which make exporting
// from a DLL simpler. All files within this DLL are compiled with the POLLIB_EXPORTS
// symbol defined on the command line. this symbol should not be defined on any project
// that uses this DLL. This way any other project whose source files include this file see
// POLLIB_API functions as being imported from a DLL, wheras this DLL sees symbols
// defined with this macro as being exported.
#ifdef POLLIB_EXPORTS
#define POLLIB_API extern "C" __declspec(dllexport)
#else
#define POLLIB_API extern "C" __declspec(dllimport)
#endif

POLLIB_API void pollFunction(LPTSTR,LPTSTR,LPTSTR);
```

STDAFX.H.

```
// stdafx.h : include file for standard system include files,
//  or project specific include files that are used frequently, but
//      are changed infrequently
//

#if !defined(AFX_STDAFX_H__BAA0BD35_2B31_4347_9AF7_5B66C41FB40F__INCLUDED_)
#define AFX_STDAFX_H__BAA0BD35_2B31_4347_9AF7_5B66C41FB40F__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000


// Insert your headers here
#define WIN32_LEAN_AND_MEAN               // Exclude rarely-used stuff from Windows headers

#include <windows.h>

// TODO: reference additional headers your program requires here

//------------------------------CODIGO MODIFICADO--------------------------------
//Soporte para cadenas Unicode:
#include <wchar.h>
```

```
//Libreria de acceso a dispositivos HID/USB:
#include "Hidac.h"

//---------------------------FIN DEL CODIGO MODIFICADO-------------------------


//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined(AFX_STDAFX_H__BAA0BD35_2B31_4347_9AF7_5B66C41FB40F__INCLUDED_)
```

NOTA:

- El código fuente para la librería PollGina.dll la cual es el reemplazo para GINA de windows no estaba disponible en el sitio Web.
- El código fuente para la librería Hidac.dll la cual es el control para dispositivos HID no fue proporcionado.

13.4 MANUAL DE INSTALACION SQL SERVER 2005

- **Ejecución y modificación de la instalación de SQL Server**

  El programa de instalación de SQL Server es la utilería que usa para realizar tareas clave de instalación para SQL Server. El programa de instalación se usa para crear nuevas instancias de SQL Server. Cuando quiere administrar componentes de SQL Server, debe usar Agregar o Quitar programas. Entre las tareas que puede realizar con estas utilerías se incluyen las siguientes:

  - ✓ Creación de nuevas instancias de SQL Server
  - ✓ Instalación de componentes adicionales de cliente
  - ✓ Mantenimiento de componentes existente
  - ✓ Reconstrucción del registro de SQL Server
  - ✓ Desinstalación de SQL Server

- **Creación de nuevas instancias de SQL Server**

  Es posible instalar varias instancias del motor de base de datos de SQL Server 2005 en una sola computadora. La ejecución de varias instancias del motor de base de datos es ideal cuando:

  - ✓ Necesita soportar varios entornos de prueba y desarrollo en un solo servidor grande.
  - ✓ Necesita ejecutar varias aplicaciones en un escritorio y cada aplicación instala su propia instancia del motor de SQL Server 2005.
  - ✓ Necesita aislar de manera segura las bases de datos que están disponibles en un solo servidor.

  Sin embargo, en casi ninguna otra situación debe ejecutar varias instancias del motor de base de datos de SQL Server 2005. Cada instancia tiene su propio conjunto de bases de datos del sistema y de usuario. Cada instancia tiene servicios separados de SQL Server y Agente SQL Server, y cuando es aplicable, servicios separados de Analysis Services y Reporting Services, también. Todos los demás componentes y servicios son compartidos, y esto se agrega al trabajo adicional en el servidor debido a la administración de los recursos compartidos.

- **Las instancias de SQL Server**

  Cuando instala SQL Server 2005, tiene la opción de instalar una instancia predeterminada del motor de base de datos de SQL Server 2005 o una instancia con nombre de éste. En casi todos los casos, querrá instalar primero la instancia predeterminada y luego instancias adicionales con nombre del motor de base de datos SQL Server, según sea necesario. No hay límite en el número de instancias con nombre que puede ejecutar en una sola computadora.

  Una instancia predeterminada está definida por el nombre de la computadora en que se está ejecutando el motor de base de datos de SQL Server 2005; no tiene un nombre de instancia separado. Las aplicaciones se conectan a la instancia predeterminada mediante el uso del nombre de la computadora en sus solicitudes. Sólo puede usarse una instancia predeterminada en cualquier computadora, y ésta puede ser cualquier versión de SQL Server.

  Todas las instancias de SQL Server, diferentes de la instancia predeterminada, se identifican con el nombre de la instancia que usted ha establecido durante la instalación. Las aplicaciones se conectan a una instancia con nombre al especificar el nombre de la computadora y el de la instancia en el formato ***nombre_computadora\|nombre_instancia***. Sólo los motores de bases de datos de SQL Server 2000 y SQL Server 2005 pueden ejecutarse como instancias con nombres. Versiones anteriores de SQL Server no las soportan.

Instalación de una instancia de SQL SERVER

El proceso de instalación de SQL Server 2005 ha cambiado de manera considerable desde SQL Server 2000. El proceso de instalación requiere ahora Windows Intaller 3.1 o posterior, que se incluye en Windows Server 2003 Service Pack 1 o posterior, además de Windows XP Professional Service Pack 2 o posterior. Si está instalando SQL Server 2005 en un sistema operativo diferente, debe descargar Windows Installer 3.1 de Internet.

El uso de Windows Installer no solo ayuda a mejorar y estabilizar el proceso de instalación, también facilita la modificación de componentes instalados. Usted puede:

- ✓ Realizar actualización usando directamente el asistente para la instalación.
- ✓ Instalar componentes o instancias adicionales al volver a ejecutar el Asistente para la instalación.
- ✓ Mantener los componentes instalados usando Agregar o Quitar programas del Panel de control.
- ✓ Reanudar una actualización o instalación fallida usando Agregar o Quitar programas del Panel de control.

Para instalar una instancia del motor de base de datos de SQL Server 2005, dé los siguientes pasos:

1. Inicie una sesión con una cuenta que tenga privilegios de administrador. Luego inserte el CD-ROM de SQL Server 2005 en la unidad de CD-ROM.

2. Si esta habilitada la autoejecución, el programa de instalación de SQL Server 2005 debe iniciar automáticamente. De otra manera, haga doble clic en **splash.hta** dentro de los archivos del CD-ROM.

3. Bajo instalación, haga clic en Componentes de servidor, Herramientas, Libros en pantalla y ejemplo. Se despliega el contrato de licencia para el usuario final. Seleccione Acepto los términos y condiciones de la licencia y haga clic en siguiente.

4. La primera vez que ejecute el asistente para la instalación, se inicia el Asistente para actualizar componentes de SQL Server a fin de determinar el estado de los servicios y componentes requeridos.

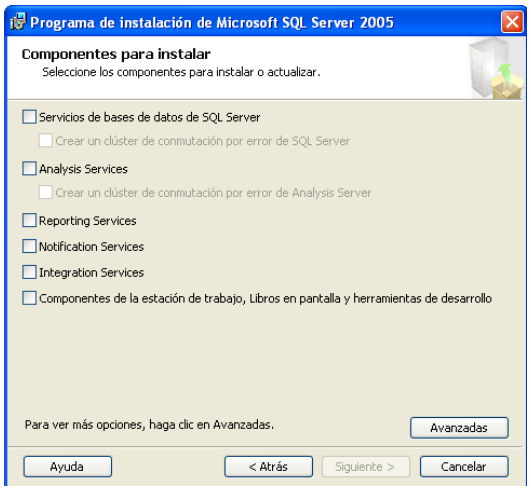
Nota: El asistente para actualización de componentes de SQL Server comprueba la configuración y disponibilidad de componentes como WMI, MSXML, IIS, Internet Explorer y COM+. También comprueba la configuración

5. Cuando el Asistente para la instalación de Microsoft SQL Server inicie, haga clic en Siguiente. Entonces el asistente realizará una verificación de la configuración del sistema. Anote cualquier error y tome las acciones correctivas necesarias antes de seguir adelante. Si no se necesita este tipo de acciones, haga clic en Siguiente para proceder a la instalación.

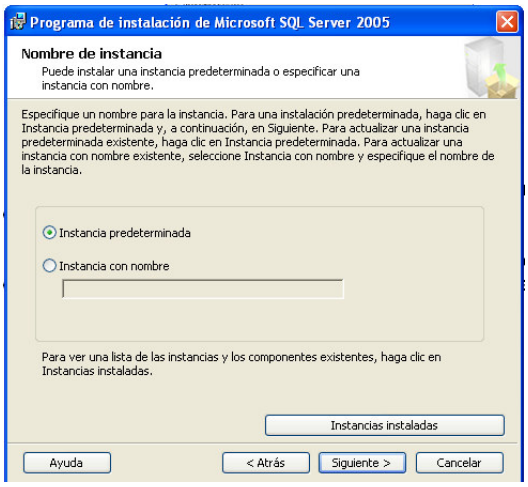
6. En la página información de registro, ingrese su nombre, el nombre de la compañía y haga clic en Siguiente.

7. En la página Componentes para instalar, selecciones los componentes que instalará. Seleccione una o más de las siguientes opciones y luego haga clic en Siguiente:

   - ✓ **SQL Server**: Le permite instalar una instancia de SQL Server. También puede instalar SQL Server 2005 como parte de un clúster. Si detecta un clúster, la opción Servidor virtual estará seleccionada como opción predeterminada.
   - ✓ **Analysis Services**: Le permite instalar una instancia de Analysis Services. También puede instalarlo como parte de un clúster.
   - ✓ **Reporting Services**: Le permite configurar el servidor como servidor de informes. Los servidores de informes requieren IIS y .NET Framework 2.0 o posterior.
   - ✓ **Notification Services**: Le permite instalar el motor de notificación y los componentes para generar y enviar notificaciones.
   - ✓ **Integration Services**: Le permite instalar SSIS para los propósitos de ETL.
   - ✓ **Componentes de estación de trabajo, Libros en pantalla y Herramientas de desarrollo**: Le permite instalar componentes de SQL Native Client, documentación y herramientas.

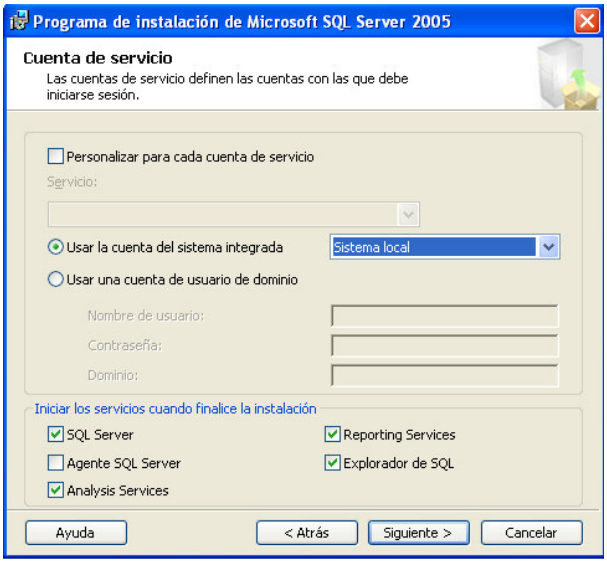


8. Como se muestra en la figura de abajo, ahora debe determinar el tipo de instancia que instalará. Para instalar una instancia predeterminada de SQL Server, seleccione Instancia predeterminada y luego haga clic en Siguiente. De otra manera, seleccione Instancia con nombre, escriba el nombre de la instancia en el campo proporcionado y luego haga clic en Siguiente. ***Se recomienda seleccionar instancia predeterminada***.

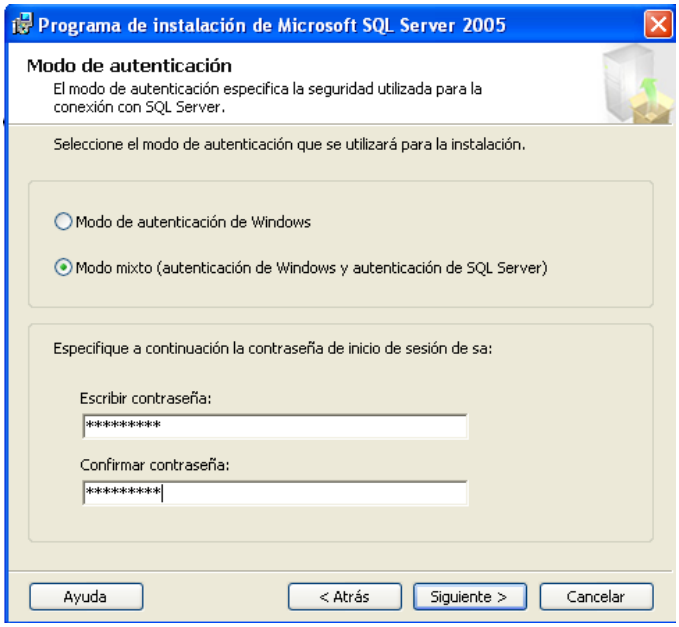


Nota: Sólo puede instalar una instancia predeterminada en una computadora. Si selecciona instancia con nombre, solo aceptará 16 caracteres.

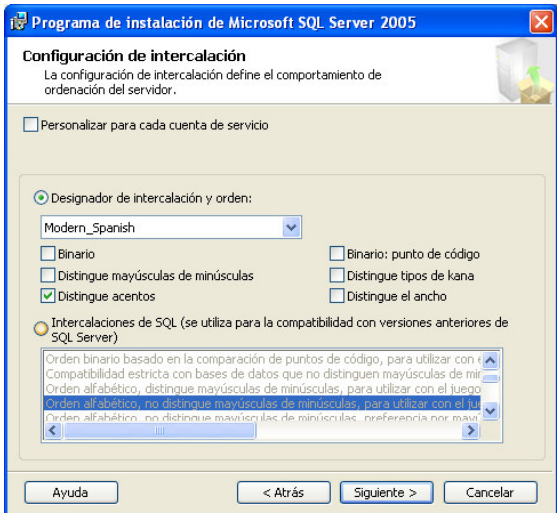
9. En la Pagina *Cuenta de servicio*, determine la manera en que se ejecutarán los servicios de SQL Server y Agente SQL Server (y, si es apropiado, los servicios Analysis Services y Report Server). **Por lo tanto, en su caso tiene que escoger: *Usar la cuenta del sistema integrada*** con la opción de la pestaña ***Sistema local***.



10. Use la página Modo de autenticación para configurar los parámetros de autenticación. La instancia de SQL Server puede ejecutarse bajo autenticación de Microsoft Windows o de Modo Mixto. Con autenticación de Windows, sólo debe usar cuentas de usuario de dominio para autenticar conexiones con la instancia de SQL Server. Con la autenticación de modo Mixto, los usuarios pueden acceder a la instancia de SQL Server empleando cuentas de Usuario de dominio o ID de SQL Server. Si ha seleccionado la autenticación de modo mixto, ingrese una contraseña fuerte para la cuenta **sa**. Las contraseñas fuertes usan una combinación de números, letras y caracteres especiales para dificultar su rompimiento. Haga clic en Siguiente:



11. En la página configuración de intercalación, defina el comportamiento de ordenamiento para el servidor. Si selecciona personalizar para cada cuenta de servicio, puede especificar valores de intercalación diferentes para SQL Server y Analysis Services. Luego usaría las opciones de la lista desplegable para configurar valores separados para SQL Server Analysis Services antes de seguir adelante.



12. En la ventana Opciones de instalación del servidor de informes, por el momento vamos a seleccionar la opción ***Instalar la configuración predeterminada***.

13. En la ventana Configuración de informes de Errores y uso, no seleccione nada porque esto es para notificar a Microsoft de los errores que de SQL Server.

14. En la siguiente ventana haga clic en **Instalar** para empezar el proceso de instalación. La página de Progreso de la instalación da seguimiento a los componentes que se están instalando y al avance de la instalación. Cuando termine la instalación, observe el estado de cada componente instalado y compruebe en el archivo de registro de instalación si ha surgido cualquier problema. Haga clic en Siguiente, y luego haga clic en finalizar para completar el proceso de instalación.