

ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

“Implementación de un Sistema de Control Domiciliario (Hardware y Software) basado en el Protocolo X10 y el Sistema de Mensajería Corta (SMS) utilizando el Kit de Desarrollo de Software de Nokia: SDK Beta 3.0 para conectividad Móvil-PC, y Microsoft Visual Basic 6.0”

TESIS DE GRADO

Previa a la obtención del Título de:

**INGENIERO EN ELECTRÓNICA Y
TELECOMUNICACIONES**

Presentada por:

Joffre Ramiro Pesántez Verdezoto

GUAYAQUIL – ECUADOR

Año: 2005

AGRADECIMIENTO

A todas las personas que colaboraron en la realización de este trabajo en especial a los ingenieros: Hernán Córdova y Pedro Vargas.

DEDICATORIA

A DIOS

A MIS PADRES

A MI HERMANO

A MIS AMIGOS

TRIBUNAL DE GRADO

Ing. Miguel Yapur A.
SUBDECANO DE LA FIEC

Ing. Pedro Vargas G.
DIRECTOR DE TESIS

Ing. Sara Ríos O.
VOCAL

Ing. Washington Medina M
VOCAL

DECLARACIÓN EXPRESA

“La responsabilidad del contenido de esta Tesis de Grado, me corresponden exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL”

(Reglamento de Graduación de la ESPOL)

Joffre Pesántez Verdezoto

RESUMEN

Desde hace ya algún tiempo, a nivel comercial, X10 un protocolo que opera sobre de la instalación eléctrica doméstica existente, sin la necesidad de modificar la infraestructura de la instalación eléctrica ha sido utilizado para lograr un cierto grado de automatización de dispositivos electrodomésticos. Los dispositivos transmisores X10 envían una señal de un nivel bajo de voltaje codificado que se sobrepone en la señal de 120 voltios de la alimentación de corriente CA.

Este proyecto consiste en un sistema de control domiciliario que permite la interacción entre el usuario y los dispositivos eléctricos de su hogar desde cualquier terminal móvil celular que soporte el envío de mensajes escritos. El uso del protocolo X10, hace que el sistema sea muy adaptable a la mayoría de los módulos transmisores y receptores para el control de dispositivos eléctricos del hogar que se encuentran en el mercado.

Con este proyecto se logra que la convergencia de dos sistemas: El Sistema de Mensajería Corta y los Sistemas de Control Domiciliarios basados en el protocolo X10, lo que permite darle al usuario control remoto a sus

dispositivos electrodomésticos al agregar los mensajes escritos de texto como un elemento de control adicional.

Haciendo uso de un computador como controlador del sistema, las interfaces y las librerías del Kit de Desarrollo de Software de Noia, SDK Beta 3.0 se logra capturar los mensajes que llegan al teléfono móvil local y luego ejecutar los comandos X10 recibidos, todo esto dentro del marco de la aplicación SMSControl la cual fue desarrollada en Visual Basic 6.0 y donde interactúan todos estos elementos de software. Es así como se logra la convergencia.

Cabe señalar que es la primera vez que en nuestra facultad se desarrolla una aplicación a partir del sistema de mensajería corta SMS, el cual solo había sido usado hasta ahora solo como un mecanismo para enviar mensajes escritos entre dos usuarios. Con este proyecto se abre una serie de posibles aplicaciones para el sistema SMS que pueden ser desarrolladas en nuestra facultad como por ejemplo: monitoreo y diagnóstico remoto de pacientes, monitoreo y control remoto de procesos industriales, monitoreo y envío de alarmas en redes de computadoras y redes de telecomunicaciones, etc.

ÍNDICE GENERAL

	Pág.
RESUMEN.....	VI
ÍNDICE GENERAL.....	VIII
ABREVIATURAS.....	XIII
ÍNDICE DE FIGURAS.....	XIV
ÍNDICE DE TABLAS.....	XVI
INTRODUCCIÓN.....	1
CAPÍTULO 1	
1. EL SISTEMA DE MENSAJERÍA CORTA (SMS: SHORT MESSAGING SYSTEM).....	3
1.1. Mensajes de texto en los sistemas celulares: Evaluación rápida de la tecnología SMS.....	3
1.1.1. Definición.....	3
1.1.2. Introducción.....	4
1.1.3. SMS sobre diferentes redes celulares.....	7
1.1.4. Beneficios y aplicaciones de SMS.....	8
1.2. Elementos de Red y Arquitectura.....	11

1.2.1. Entidades Externas al Sistema (ESME: External Short Message Entities).....	11
1.2.2. El Centro de Conmutación de Mensajes Cortos (SMSC: Short Message Switching Center)	12
1.2.3. Punto de Transferencia de Señal (STP: Signal Transfer Point: Punto de Transferencia de Señal).....	13
1.2.4. HLR	13
1.2.5. Registro de Localización de Visitantes (VLR: Visitor Locator Register).....	14
1.2.6. El medio (Air Interface)	14
1.2.7. Las Estaciones Bases (BS: Base Stations)	14
1.2.8. El Dispositivo Móvil (MD: Mobile Device)	14
1.2.9. Elementos de Señalización	16
1.2.10. Elementos de Servicio.....	17
1.2.11. Servicios de Suscriptor	18
1.2.12. Proceso de envío y recepción de mensajes.....	21

CAPÍTULO 2

2. EL PROTOCOLO X-10	26
2.1. Sistemas de Control domiciliarios: Evaluación rápida de la tecnología X10.....	26
2.1.1. Introducción.....	26

2.1.2. Antecedentes históricos	27
2.1.3. Elementos de un sistema de control domiciliario.....	29
2.2. Teoría de transmisión para X10.....	36
2.3. Variantes del protocolo X10	39
2.3.1. El protocolo CM17A.....	40
2.3.2. El protocolo CM11A.....	43

CAPÍTULO 3

3. CONECTIVIDAD FÍSICA Y LÓGICA ENTRE EL MÓVIL Y LA PC.....	46
3.1. Conectividad física	46
3.1.1. Cable DLR-3P	46
3.1.2. Los protocolos FBUS y MBUS.....	50
3.2. Conectividad lógica	54
3.2.1. Nokia Connection Manager	54
3.2.2. El Kit de desarrollo de Software de Nokia: SDK beta 3.0	55

CAPÍTULO 4

4. SISTEMA DE CONTROL DOMICILIARIO: HARDWARE	69
4.1. Elementos físicos del sistema de Control Domiciliario	69
4.1.1. El teléfono remoto, el teléfono local, la red de telefonía móvil..	70
4.1.2. La PC de gestión local, el cable de conexión con el móvil	71
4.1.3. La Interface Firecracker: RS-232 to Wireless y el Tranceiver	

TM751	72
4.1.4. Los módulos receptores X10 para lámparas LM465, para focos WS12A , y de Potencia HD245-C.....	76
4.1.5. El sensor de movimientos EAGLE-EYE	79
4.1.6. El módulo CM11A.....	81
CAPÍTULO 5	
5. SISTEMA DE CONTROL DOMICILIARIO: SOFTWARE.....	84
5.1. Desarrollo de la aplicación SMS Control en Visual Basic 6.0	84
5.1.1. El formulario frmAgregarUsuario	85
5.1.2. El formulario frmAgregarDispositivo	87
5.1.3. El formulario frmConfigSMSControl.....	79
5.1.4. El formulario frmConfigAlarmas.....	88
CAPÍTULO 6	
6. PUESTA A PRUEBA DEL SISTEMA	110
6.1. Dificultades en la implementación.....	110
6.2. Simulación de un módulo receptor X10	113
6.3. Problemas con X10.....	119
CONCLUSIONES Y RECOMENDACIONES	126

ANEXOS

ANEXO A: Tabla de Datos del Protocolo CM17A.....	131
ANEXO B: Tabla de Datos del Protocolo CM11A.....	140
ANEXO C: Tabla de costos de los dispositivos del sistema	142
ANEXO D: Código fuente del programa	144
BIBLIOGRAFÍA.....	180

ABREVIATURAS

SMS:	Short Message System
GSM:	Global System for Mobile Communication
TDMA:	Time Division Multiplexing Access
CDMA:	Code Division Multiplexing Access
VMS:	Voice Message System
SMSC:	Short Message Service Center
STP:	Signal Transfer Point
VLR:	Visitor Local Registration
MSC:	Mobile Switching Center
MT-SM:	Mobile Terminated – Short Message
MO-SM:	Mobile Originated – Short Message
ESME:	External Short Message Entities
SDK:	Software Development Kit
RTS:	Request to Send
DTR:	Data Terminal Ready

ÍNDICE DE FIGURAS

	Pág
Figura 1.1 Arquitectura Básica de un Sistema de Mensajería Corta	12
Figura 1.2 Infraestructura de Red.....	20
Figura 1.3 Ejemplo de un MT-SM Escenario GSM	21
Figura 1.4 Ejemplo de un MT-SM Escenario IS – 41	22
Figura 1.5 Ejemplo de un MO-SM Escenario GSM.....	23
Figura 1.6 Ejemplo de un MO-SM Escenario IS-41	24
Figura 2.1 Configuración CM17A (Firecracker).....	29
Figura 2.2 Configuración CM11A (bidireccional).....	30
Figura 2.3 Diagrama de Bloques de un transceiver X10 típico.....	33
Figura 2.4 Diagrama de bloques de un receptor X10 típico.....	33
Figura 2.5 Esquema de transmisión de bits X10.....	36
Figura 2.6 Transmisión de códigos de función y de dispositivo.....	37
Figura 2.7 Transmisión complementaria de cada bit.....	38
Figura 2.8 Formato para la transmisión inalámbrica.....	40
Figura 3.1 Pinout en el puerto del móvil Nokia 7160.....	48
Figura 3.2 Formato de trama FBUS.....	54
Figura 3.3 Interfase gráfica del software Nokia Connection Manager.....	55
Figura 3.4 Componentes de la librería NokiaCLMessaging.....	63

Figura 4.1	Esquema del sistema a nivel de hardware	69
Figura 4.2	La interfase Firecracker.....	73
Figura 4.3	Transmisión de bits a la interfase.....	75
Figura 4.4	El transceiver TM751.....	76
Figura 4.5	El módulo de lámparas LM465.....	77
Figura 4.6	El módulo para focos WS12A.....	78
Figura 4.7	El módulo de potencia HD245.....	78
Figura 4.8	Funcionamiento del sensor EAGLE-EYE.....	79
Figura 4.9	El sensor de movimientos EAGLE-EYE.....	80
Figura 4.10	La interfase CM11A bidireccional	83
Figura 5.1	Interfase gráfica del formulario frmAgregarUsuario.....	86
Figura 5.2	Interfase gráfica del formulario frmAgregarDispositivo.....	87
Figura 5.3	Lógica de control en el formulario frmConfigSMSControl.....	96
Figura 5.4	Elementos de la librería NokiaCICall.....	104
Figura 6.1	Detector de cruce por cero	114
Figura 6.2	Fuente de poder	115
Figura 6.3	Etapa extractora de señal.....	115
Figura 6.4	Señal de entrada a la etapa extractora.....	117
Figura 6.5	Señal de salida del los filtros activos.....	118
Figura 6.6	Señal de salida del amplificador secundario.....	118
Figura 6.7	Señal de salida de la etapa comparadora.....	119
Figura 6.8	Atenuación de la señal X10.....	124

ÍNDICE DE TABLAS

	Pág
Tabla 1 SMS sobre diferentes redes.....	7
Tabla 2 Configuración de los parámetros seriales	44
Tabla 3 Conexión física entre la interfase y la PC	44
Tabla 4 Características y función de cada pin de la interfase del móvil..	48
Tabla 5 Conexión Serial.....	50
Tabla 6 Métodos de la interfase IShortMessageItem.....	64
Tabla 7 Lista de teléfonos compatibles con el SDK Beta 3.0 de Nokia....	67
Tabla 8 Estados de las líneas RTS y DTR.....	74
Tabla 9 Configuración inicial del control MSCComm.....	89
Tabla 10 Valores del parámetro command del método ExecWait ()	100

INTRODUCCIÓN

La alimentación eléctrica monofásica que llega a la caja de breakers se distribuye entre las diferentes tomas que alimentan los interruptores de luz, tomas de corriente, y aparatos. Esto hace que el cableado eléctrico domiciliario que interconecta de forma física los diferentes dispositivos eléctricos como luces, lámparas, electrodomésticos, se convierta en una pequeña red domiciliaria. Haciendo uso de la red existente, es factible enviar señales de control a través del cableado eléctrico con la finalidad de obtener un cierto grado de control sobre los diferentes dispositivos eléctricos del hogar. En el mercado, actualmente ya existen módulos que permiten controlar los dispositivos eléctricos a través del cableado eléctrico e inclusive permiten realizar el control a través de un computador, sin embargo, no tenemos un sistema que incorpore un elemento adicional de control: los mensajes escritos enviados desde un celular remoto.

Este proyecto realiza precisamente eso, utilizar el Sistema de Mensajería Corta, SMS, para controlar remotamente los dispositivos eléctricos existentes en un domicilio; es decir, el usuario puede interactuar con los dispositivos eléctricos de su hogar, tales como: lámparas, iluminación, acondicionadores de aire, etc, controlar su encendido y apagado de manera remota a través de cualquier teléfono móvil que soporte el sistema de mensajería corta (SMS).

El objetivo específico de este proyecto consiste en el desarrollo de una aplicación en Visual Basic 6.0 que reciba las sentencias de control enviadas desde el teléfono celular remoto mediante SMS (Short Messaging System) y envíe las palabras de control del protocolo X10 a través de uno de los puertos seriales, haciendo uso de la interfase RS-232 a RF: Firecracker ó la interfase CM11-A (bidireccional), hasta los módulos receptores X10, los cuales están conectados a los dispositivos eléctricos del hogar. Los mensajes escritos con las sentencias de control enviadas desde el teléfono celular remoto son recibidas en el teléfono celular local: Nokia, modelo 6360, el cual se encuentra conectado de manera física a uno de los puertos seriales del computador de gestión local a través del cable DLR3-P de Nokia.

La aplicación hará uso de varios métodos, y procedimientos que se encuentran en las librerías del Kit de Desarrollo de Software de Nokia: SDK beta 3.0. Estas serán usadas para lograr la detección y la lectura de los mensajes escritos que lleguen hasta el teléfono celular local. Para lograr la conectividad lógica y sincronización entre el teléfono celular local y la PC de gestión local, se hará uso del programa Nokia Connection Manager.

CAPÍTULO 1

1. SISTEMA DE MENSAJERÍA CORTA (SMS: SHORT MESSAGING SYSTEM)

1.1. Mensajes de texto en los sistemas celulares: evaluación rápida de la tecnología sms

Se tratará de abordar los conceptos básicos sobre SMS: Qué es un mensaje escrito. Cúal es la arquitectura de red detrás de este sistema, ¿Cómo se realiza la transferencia desde el móvil hasta las entidades de red y viceversa.

1.1.1. Definición

El Sistema de Mensajería Corta (SMS: Short Messaging System) es un servicio inalámbrico aceptado globalmente que permite el envío y recepción de mensajes alfanuméricos entre dos usuarios de un sistema de telefonía móvil, además del envío y recepción de mensajes alfanuméricos entre un

suscriptor móvil y sistemas externos, como es el caso del correo electrónico, servicios de paging. El mensaje no puede ser mayor a 160 caracteres alfanuméricos sin contener imágenes o gráficos. Los mensajes SMS son soportados por redes GSM, TDMA, y CDMA actuales.

1.1.2. Introducción

El Sistema de Mensajería Corta aparece en la escena de los sistemas de telefonía móvil en el año de 1991 en Europa. El estándar europeo del sistema digital de telefonía móvil mejor conocido como GSM: Global System for Mobile Communication, ya incluía el servicio de mensajería corta desde su lanzamiento.

En Norteamérica, SMS fue disponible solo en redes digitales, es decir, redes que utilicen técnicas de acceso múltiple, como TDMA: Time División Multiplexing Access, y CDMA: Code División Multiplexing Access. Estas redes fueron construidas primeramente por empresas como: BellSouth Mobility, PrimeCo, y Nextel, entre otros.

La consolidación de las redes digitales de telefonía permitió servicios de ámbito internacionales además de soportar más de

una tecnología inalámbrica. Esto originó la aparición de la demanda de nuevo servicios. Para satisfacer estos requerimientos se creó el Centro de Servicios de Mensajería Corta (SMSC: Short Messaging Service Center) basado en sistemas de Redes Inteligentes (IN Intelligent Networks).

Los usuarios del servicio hacen uso del SMSC: Short Messaging System Center como un sistema de almacenamiento y envío de sus mensajes de texto. La red inalámbrica provee los mecanismos requeridos para encontrar a la estación destino, y transporta el mensaje desde el SMSC hasta la misma.

En contraste con otros sistemas de transmisión de texto, los elementos del servicio de mensajería corta están diseñados para garantizar el envío seguro de los mensajes hasta su destino. Adicionalmente, SMS soporta varios mecanismos de entrada que permiten la interconexión de diferentes fuentes de mensajes con diferentes destinos, es decir, que pertenezcan a diferentes redes móviles de distintos operadores.

Una característica distintiva de este servicio es que cualquier suscriptor móvil puede recibir o enviar un mensaje corto de

texto en cualquier momento, independientemente si una llamada o la transmisión de datos está en progreso (en algunas implementaciones, dependiendo de las capacidades del MSC o del SMSC). SMS también garantiza el envío de los mensajes por la red. Si se produce alguna falla en la MSC destino, el mensaje es almacenado en la SMSC hasta que el móvil destinatario esté nuevamente disponible.

SMS se caracteriza por el envío de paquetes fuera de banda y transferencia de mensajes en un bajo ancho de banda, lo que resulta en una manera muy eficiente de transmitir ráfagas de paquetes cortos de datos. Las aplicaciones iniciales de SMS se enfocaban en la eliminación de los pagers alfanuméricos permitiendo servicios de notificación de propósito general en dos vías, primariamente con correo de voz. Con el desarrollo de la tecnología, una variedad de servicios fueron introducidos, incluyendo correo electrónico, fax, paging integrado, banca interactiva, servicios de información como revisión de stock, y aplicaciones de integración con Internet.

Las aplicaciones inalámbricas para datos incluían la descarga del módulo de identidad del suscriptor (SIM: Subscriber Identity

Module) para activación, débitos, profile-editing propósitos, Puntos de Venta Inalámbricos (POSs: Wireless Points of Sale) y otras aplicaciones para servicios de campo como lecturas automáticas de medidores, monitoreo remoto, y servicios basados en localización. Adicionalmente, se ofrece la integración con servicios de Internet como Mensajería basada en el Web y otras aplicaciones como mensajería instantánea, juegos en línea, y chatting.

1.1.3. SMS sobre diferentes redes

Desde su ingreso al estándar GSM, SMS también ha sido incorporado en muchos otros estándares: Nordic Mobile Telephone (NMT), Code Division Multiple Access (CDMA) y Personal Digital Cellular (PDC) en Japón.

Cada uno de estos estándares implementa SMS de maneras ligeramente diferentes:

Tabla 1. SMS sobre diferentes redes

Tipo Red móvil	Tipo	Disponibilidad SMS	Longitud del mensaje	Desarrollo
GSM 900	Digital	Si	160	Amplio
GSM 1800	Digital	Si	160	Amplio
GSM 1900	Digital	Si	160	Norteamérica
TACS/ETACS	Analógico	No	Ninguna	Ninguno

NMT	Analógico	Sí	Ninguna	Europa Oriental
TDMA/D-AMPS	Digital	Sí	Ninguna	Norteamérica
NAMPS	Analógico	Sí	46	Norteamérica
CDMA	Digital	Sí	256	Norteamérica
PHS	Digital	Si	Ninguna	Japón
PDC	Digital	Sí	Ninguna	Japón
IDEN/NEXTEL	Digital	Sí	140	Norte y Suramérica
TETRA/Dolphin	Digital	Sí	256	Europa
Globalstar	Satelital	Sí	160	Global

1.1.4. Beneficios y aplicaciones de SMS

Una vez que los servicios de voz ya fueron desarrollados en un ambiente móvil, SMS provee un poderoso vehículo para lograr la diferenciación en el servicio. Si el mercado lo permite, SMS puede llegar a ser una importante fuente de ingresos para el proveedor de servicios.

Los beneficios de SMS para sus suscriptores de centran en la conveniencia, flexibilidad, e integración de los servicios de voz y datos. Desde esta perspectiva, el beneficio primario consiste en la habilidad para usar al móvil como una extensión de su computadora personal. SMS elimina la necesidad de tener un dispositivo pager y un teléfono celular para recibir las llamadas

de voz. De manera mínima, SMS provee los siguientes beneficios:

- Envío de notificaciones y alarmas
- Envío garantizado de los mensajes
- Confiabilidad, mecanismo de comunicación de bajo costo para información concisa.
- Capacidad de mostrar los mensajes y retornar llamadas de una manera selectiva.
- Aumento de la productividad del suscriptor.

Sistemas más sofisticados pueden ofrecer:

- Envío de mensajes a múltiples suscriptores a la vez
- Habilidad para recibir distintos tipos de información
- Generación de correo electrónico
- Creación de grupos de usuarios
- Integración con otros servicios basados en el Web

Los beneficios que provee SMS al proveedor de servicios son los siguientes:

- Habilidad para incrementar el ingreso por usuario (a la vez de aumentar el número de llamadas en la red como respuesta a las notificaciones de paging)

- Una alternativa a los servicios de paging, pudiendo reemplazar o complementar a la oferta existente de estos servicios.
- Habilidad de incorporar acceso a datos para usuarios corporativos.
- Nuevos ingresos, producto de la adición de servicios de valor agregado como correo electrónico, correo de voz, fax, servicios basados en el Web, servicios de notificación de stock, horarios de aerolíneas, etc.
- Provisión de servicios administrativos como envío de notificación de saldos de sus suscriptores, descarga inalámbrica del SIM, etc.
- Protección de recursos medulares de la red como canales de voz
- Soporte WAP, Wireless Application Protocol.
- Todos estos beneficios tienen un retorno rápido, con costos incrementales modestos en períodos de pago cortos, lo cual hace que SMS sea una inversión atractiva para los operadores.

1.2. Elementos de red y arquitectura

En este apartado trataremos de presentar los elementos que conforman un Sistema y como esta se relacionan entre si para proveer la funcionalidad del sistema.

1.2.1. Entidades externas al sistema (ESME: External Short Messaging Entities)

Una ESME es un elemento que puede recibir o enviar un mensaje de texto. La entidad externa (ESME: External Short Message Entity) puede estar localizada en una red fija, en otro centro de servicio, o puede ser un dispositivo móvil.

- **VMS (Voice Messaging System).**- Es responsable de recibir, almacenar y reproducir los mensajes de voz destinados al abonado que no puede recibir la llamada debido a diversas razones (móvil fuera del área de servicio, móvil apagado u otras). También tiene la capacidad de enviar los correos de voz a los usuarios de SMS.
- **Web.**- Debido al crecimiento de los servicios Web, es casi obligatoria la conectividad entre SMS y la Web, por lo cual los servidores de Internet son considerados también entidades externas al sistema.

- **Correo electrónico.-** Esta es una de los servicios de Internet más utilizados en la actualidad, por lo que se demanda que un sistema SMS tenga la capacidad de enviar notificaciones de e-mail y soportar la recepción de e-mail, es decir, el SMSC debe soportar la interconexión con los servidores de correo electrónico.
- **Otros.-** Existen muchos otros mecanismos para enviar mensajes al SMSC: redes de paging, software especializado para PC basado, etc.

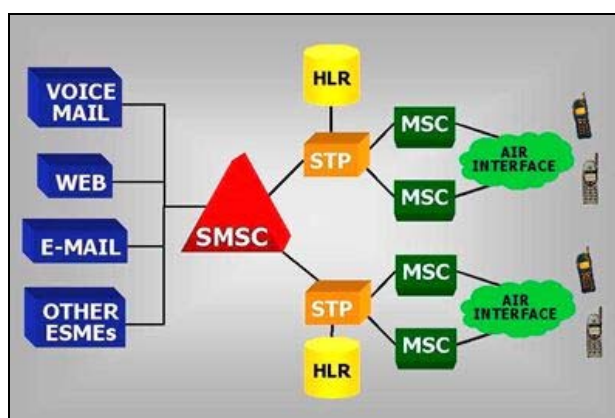


Figura 1.1. Arquitectura Básica de un Sistema de Mensajería Corta

1.2.2. SMSC (Short Message System Center)

El SMSC es una combinación de hardware y software responsable recibir, almacenar y enviar mensajes cortos de texto entre las entidades de mensajería corta, ESME y dispositivos móviles. El SMSC debe tener una alta confiabilidad, capacidad para suscriptores, y eficiencia en mensajería. Debe

ser escalable para poder acomodar a la creciente demanda de suscriptores SMS en la red.

Otro factor a considerar es la facilidad de operación y mantenimiento de las aplicaciones, además de la flexibilidad para activar nuevos servicios y actualizaciones de software.

1.2.3. Punto de transferencia de señal (STP: Signal Transfer Point)

El STP es un elemento de red normalmente disponible en desarrollos de IN (Redes Inteligentes) que permite conexiones IS-41 sobre enlaces del sistema de señalización 7 (SS7), con múltiples elementos de la red.

1.2.4. HLR

El HLR es una base de datos usada para el permanente almacenamiento y manejo de los perfiles de usuario. El HLR provee la información de ruteo para el usuario indicado. Inclusive si la estación destino no está disponible cuando el mensaje ha sido enviado. El HLR informa al SMSC cuando la estación destino es reconocida por la red como accesible y entonces se envía el mensaje.

1.2.5. Registro de locación de visitante (VLR: Visitor Locator Register)

Este es una base de datos que contiene información temporal acerca de los suscriptores domiciliados en un HLR que se encuentran ingresando a otro HLR. Esta información es requerida por el MSC para dar servicio a los suscriptores visitantes.

1.2.6. Centro de switcheo móvil (MSC: Mobile Switching Center)

El MSC realiza las funciones de conmutación el sistema y controla las llamadas entre los teléfonos y sistemas de datos. El MSC envía los mensajes cortos al suscriptor móvil específico a través de la estación base apropiada.

1.2.7. El Medio (Air Interface)

El medio está definido en todas las tecnologías inalámbricas (CDMA o TDMA) como el rango de frecuencias utilizado para transmitir y recibir las señales de voz y datos desde el MSC hasta los dispositivos móviles.

1.2.8. Las Estaciones Bases (BS: Base Stations)

Todas las funciones relacionadas con la transmisión de las señales electromagnéticas de radio entre el MSC y los

dispositivos móviles, son efectuadas por las estaciones base. Las BS consisten en controladores y transceivers también conocidos como sitios de celda o simplemente celdas. El BSC: Base Station Controller puede controlar a una o más estaciones base y esta a cargo del manejo de sus propios recursos cuando un suscriptor se mueve de un sector de la celda a otro, sin importar si este nuevo sector es limítrofe con otras celdas.

1.2.9. El Dispositivo Móvil (MD: Mobile Device)

Es la terminal capaz de recibir y originar los mensajes de textos. Comúnmente estos dispositivos son teléfonos celulares digitales pero recientemente estas capacidades han sido dadas a otros dispositivos como PDAs y computadoras de mano. La infraestructura inalámbrica está basada en señalización SS7. SMS hace uso de la Mobile Application Part (MAP), la cual define los métodos y mecanismos de la comunicación en redes inalámbricas que emplean SS7 y sus capacidades de Transacción (TCAP: Transactional Capabilities Application Part). La capa de servicio SMS hace uso de MAP y TCAP para permitir la transferencia de mensajes cortos entre entidades iguales. Las capacidades de las terminales varían dependiendo de la tecnología inalámbrica soportada por la terminal.

1.2.10. Elementos de señalización

La capa MAP define las operaciones necesarias para soportar SMS. Los estándares Americanos e Internacionales han definido una capa MAP usando los servicios de SS7 TCAP. El estándar Americano es publicado por la TIA y es referido como IS-41. El estándar internacional es definido por la ETSI: European Telecommunications Standards Institute y se denomina como GSM MAP.

Las siguientes operaciones básicas MAP son requeridas para proveer el servicio SMS de extremo a extremo:

- **Petición de Información de Ruteo.-** Antes de intentar enviar el mensaje, el SMSC debe recibir la información de ruteo para determinar el MSC correspondiente al dispositivo móvil destino. Esto se realiza a través de una petición del HLR del móvil destino, lo cual se efectúa mediante el uso del mecanismo: *SMSrequest* y *SendRoutingInfoForShortMsg* en IS-41 y GSM respectivamente.
- **Envío de Mensajes Punto a Punto.-** Después de que la dirección del MSC ha sido obtenida del HLR de la estación, la operación de envío provee un servicio confirmado de entrega. La operación trabaja en conjunto con la estación

base mientras el mensaje está siendo llevado desde la MSC a la MS. Esta operación es realizada mediante el uso de los mecanismos: *short message delivery–point-to-point* (SMD–PP) y *forwardShortMessage* mechanisms en IS–41 y GSM, respectivamente.

- **Indicación de Espera.-** Esta operación se realiza cuando la operación de envío falla debido a ciertas causas como cuando el móvil destino no ha sido registrado y provee la capacidad de que el HLR notifique al SMSC cuando el móvil indicado está nuevamente disponible. Esto se realiza mediante los siguientes mecanismos: *SMS_notification* indicator y *set_message_waiting_data* en IS–41 y GSM, respectivamente.
- **Alerta del Centro de Servicio.-** Permite que el HLR informe al SMSC que el móvil está ya reconocido como disponible. Esto se logra mediante los mecanismo de: *SMS_notification* y *alert_service_center* en IS–41 y GSM, respectivamente.

1.2.11. Elementos de Servicio

SMS está compuesto de varios elementos de servicio involucrados en el envío y recepción de mensajes cortos

- **Expiración de mensaje.-** El SMSC almacenará y reintentará el envío de los mensajes de receptores inalcanzables hasta que la entrega se haya completado satisfactoriamente o hasta que el tiempo de expiración configurado se haya cumplido.
- **Prioridad.-** Este es un elemento de información proveído por un SME que indica la urgencia del mensaje y lo diferencia de los mensajes con prioridad normal.
- **Intercalación de mensajes.-** El SMSC almacena los mensajes por un período no mayor al tiempo de expiración (se asume que el tiempo de intercalación es menor que el tiempo de expiración asociado al mensaje), y después de que el tiempo de intercalación expira, el mensaje será enviado a un sistema alternativo como una red paging o un servidor de correo.
- Además, SMS provee una etiqueta de tiempo que reporta el tiempo de llegada del mensaje al SMSC y un indicador sobre la existencia o no de más mensajes para enviar.

1.2.12. Servicios del suscriptor

SMS provee dos servicios básicos punto a punto:

- Mensaje Corto originado en el móvil (Mobile-originated short message MO–SM)
- Mensaje corto terminado en el móvil (Mobile-terminated short message MT–SM)

El mensaje corto MO es transportado desde el móvil hasta el SMSC y puede ser enviado a otro suscriptor móvil o para suscriptores de servicios de paging o para usuarios de una red IP.

El mensaje corto MT es transportado desde el SMSC hasta el móvil y puede ser enviado desde el SMSC por otro suscriptor móvil o por otras fuentes como óbice-mail, paging, etc.

Algunas de las potenciales aplicaciones de la tecnología SMS, utilizando MT–SM y MO–SM son:

- **Servicios de Notificación.-** Son las aplicaciones más ampliamente desarrolladas. Como ejemplos de notificación se incluyen los siguientes:
- **Mensajes de notificación Voz/Fax.-** el cual indica que un mensaje de voz o un mail-fax ha llegado a su casillero de voz.

- **Notificación de E-mail.-** El cual indica que un nuevo correo electrónico está presente es su buzón o algún recordatorio sobre algún evento está presente en su servicio de recordatorios/agenda.
- **Servicios de Información.-** Varios servicios de información pueden ser implementados; reportes del tiempo, reportes de tráfico, entretenimiento (información sobre horarios de salas de cine, teatros, conciertos, etc), información financiera, etc

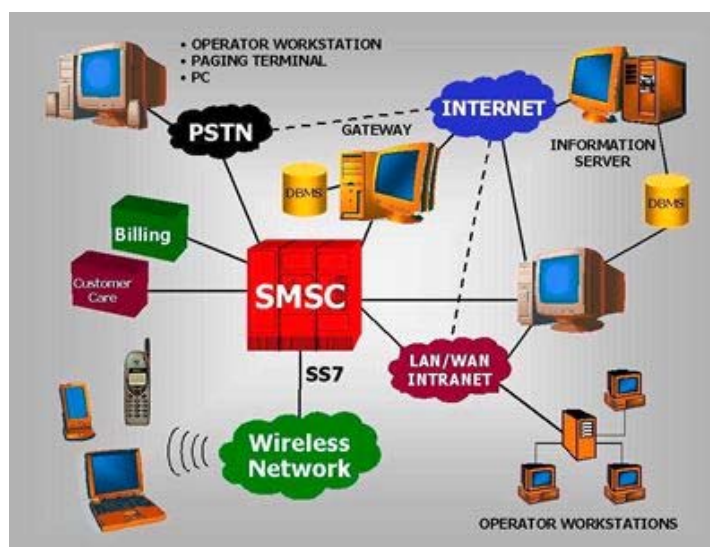


Figura 1.2. Infraestructura de Red

- **Integración Wap.-** SMS permite la notificación de nuevos mensajes Wap a los suscriptores, pero también pueden ser usado como un mecanismo de transporte para los mensajes Wap. Estos mensajes contiene diversos tipos de

información incluyendo; bases de datos, la WWW o servidores de correo, etc.

- **Servicios Móviles de Datos.-** El SMSC puede también proveer servicios inalámbricos de datos pero de corta escala; consulta de stock en inventarios, procesamiento de pedidos, manejo de contactos con clientes, banca móvil, chat móvil, servicios de búsqueda, servicios de notificación de pagos vencidos, etc.

1.2.13. Proceso de envío y recepción de mensajes

MT-SM Escenario GSM

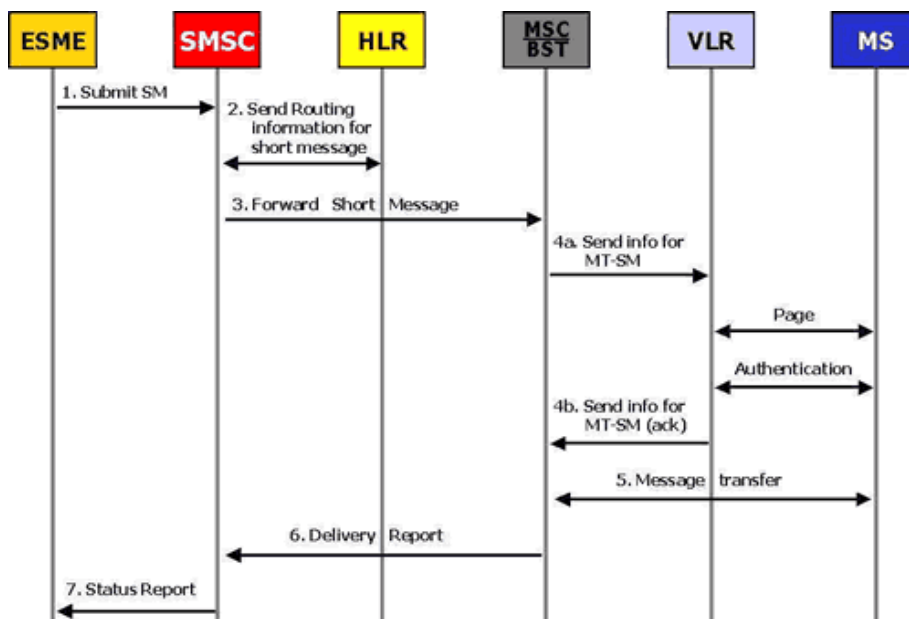


Figura 1.3. Ejemplo de un MT-SM Escenario GSM

1.- El SM es enviado desde el ESME al SMSC

- 2.- Después de completar sus procesos interno el SMSC pregunta al HLR y recibe información de ruteo del suscriptor móvil.
- 3.- El SMSC envía el SM al MSC usando la operación SM delantera
- 4.- El MSC recibe información desde el VLR. Esta operación puede incluir un proceso de autenticación
- 5.- El MSC transfiere el mensaje corto al MS (mobile suscriber)
- 6.- El MSC retorna al SMSC el resultado de la operación SM delantera. Si el ESME lo pide, el SMSC le envía un reporte indicando el éxito del envío del mensaje.

MT-SM Escenario IS – 41

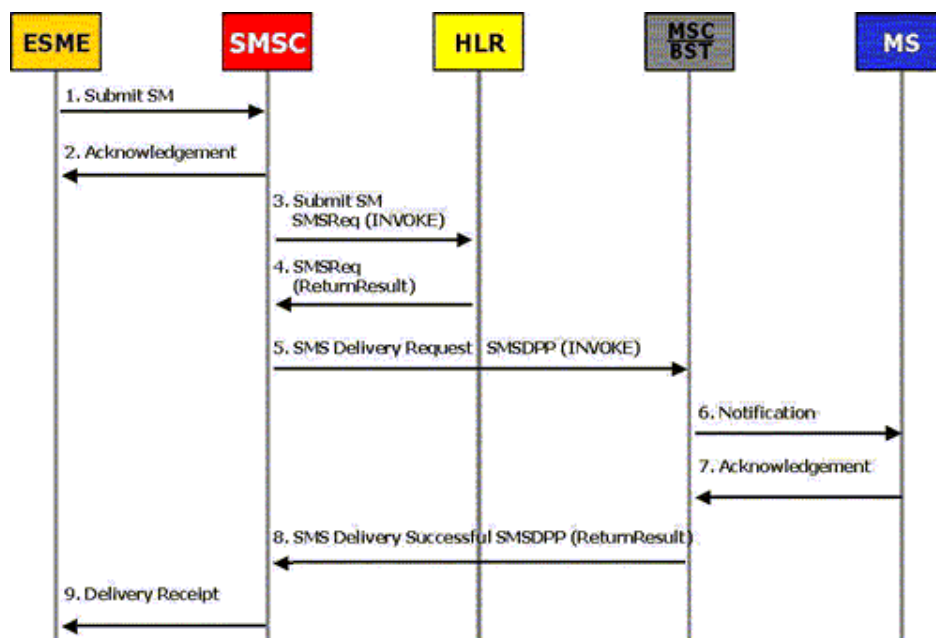


Figura 1.4 Ejemplo de un MT-SM Escenario IS – 41

- 1.- El mensaje es enviado desde el ESME al SMSC
- 2.- El SMSC envía un acuse de recibo al ESME, indicando la recepción del SM.
- 3.- Después de completar sus procesos interno, el SMSC interroga al HLR.
- 4.- El HLR envía información de rutas del suscriptor móvil destino.
- 5.- El SMSC envía el mensaje al MSC haciendo uso de la operación SMSDPP Invoke.
- 6.- El MSC transfiere el mensaje al MS.
- 7.- El MS retorna un acuse de recibo al MSC.
- 8.- El MSC retorna al SMSC el resultado de la operación SMSDPP Invoke. Se es requerido por el ESME, el SMSC envía un reporte indicando el éxito de la operación.

MO-SM Escenario GSM

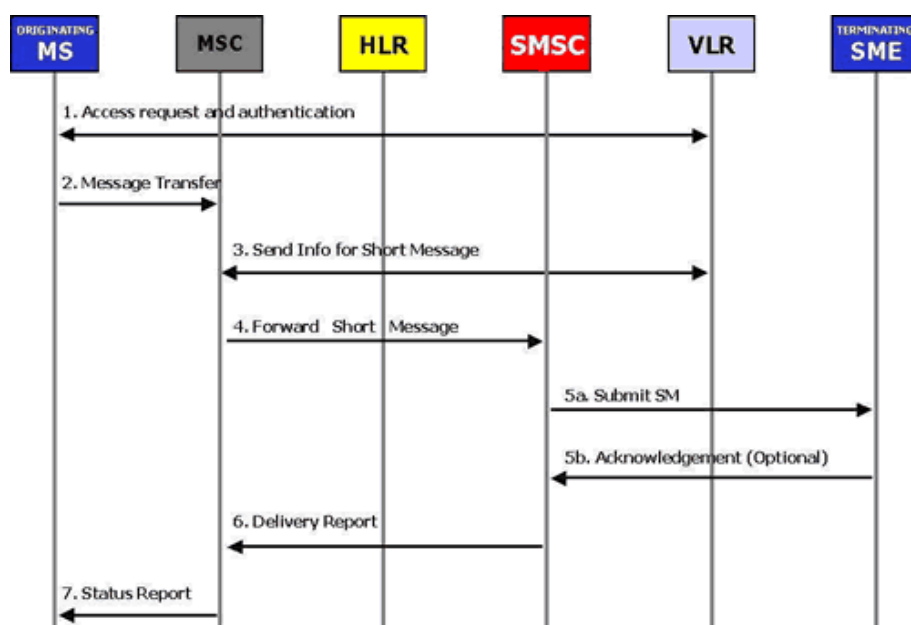


Figura 1.5 Ejemplo de un MO-SM Escenario GSM

- 1.- El MS está activo y registrado en el sistema
- 2.- El MS transfiere el SM al MSC.
- 3.- El MSC interroga al VLR para saber si el mensaje no viola los servicios o las restricciones impuestas al suscriptor.
- 4.- El MSC envía el SM al SMSC haciendo uso de la operación forwardShortMessage.
- 5.- El SMSC envía el SM al ESME (el acuse de recibo es opcional)
- 6.- El SMSC indica al MSC el éxito de la operación forwardShortMessage. El MSC retorna al MS el resultado de la operación MO-SM.

MO-SM Escenario IS-41

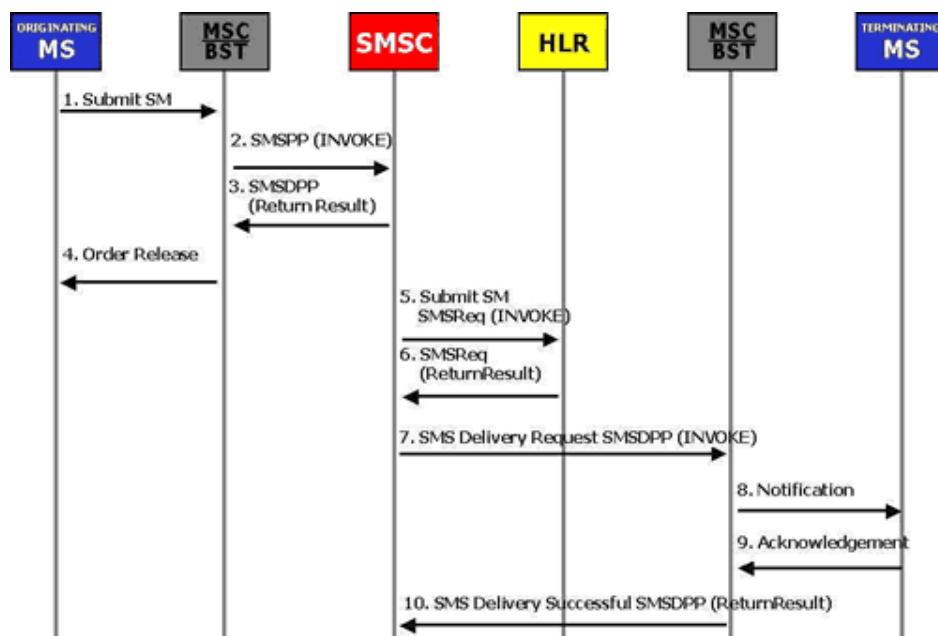


Figura 1.6 Ejemplo de un MO-SM Escenario IS-41

- 1.- El MS transfiere el SM al MSC.
- 2.- El MSC interroga al SMSC local para verificar que le mensaje no viole ninguna restricción impuesta.
- 3.- El MSC envía el mensaje al SMSC local haciendo uso de la operación SMSPP Invoke.
- 4.- El SMSC envía un acuse de recibo al MSC.
- 5.- El MSC retorna una notificación al MS.
- 6.- El SMSC pregunta al HLR por la localización del destinatario.
- 7.- El HLR retorna esta información.
- 8.- El SMSC envía el mensaje al MSC utilizando el MS destino.
- 9.- El SMSC envía el mensaje al MS.
- 10.- El MS envía un acuse de recibo al MSC notificando sobre el éxito de la operación SMSDPP.
- 11.- El MSC retorna al SMSC el resultado de la operación MO-SM.

CAPÍTULO 2

2. EL PROTOCOLO X-10

2.1. Sistemas de control domiciliarios: evaluación rápida de la tecnología X10

En este apartado presentamos los aspectos importantes sobre el protocolo X10, empezando por revisar sus orígenes, su implementación y en especial su teoría de transmisión.

2.1.1. Introducción

X10 es un protocolo que opera sobre de la instalación eléctrica doméstica existente, sin la necesidad de modificar la infraestructura de la instalación eléctrica. Los dispositivos transmisores X10 envían una señal de un nivel bajo de voltaje codificado que se sobrepone en la señal de 120 voltios de la alimentación de corriente CA. Cualquier dispositivo receptor X10 conectado a una de las tomas de alimentación eléctrica

detectará esta señal. Sin embargo, sólo el receptor cuya identificación (ID) o dirección corresponda con la misma dirección existente en la señal, responderá.

Este sistema de comunicación permite direccionar hasta 256 códigos posibles, configurando en cada receptor o emisor el código al que su artefacto responderá, por ejemplo A-1, donde A es el código de casa y 1 el número de unidad, si por ejemplo, se requiere que un conjunto de luces responda con el mismo código, debe configurar el mismo código en todos los receptores conectados a la misma.

2.1.2. Antecedentes históricos

La tecnología X10[1], basada en corrientes portadoras, fue desarrollada entre 1.976 y 1.978 por los ingenieros de Pico Electronics Ltd, en Glenrothes, Scotland. X10 surgió de una familia de chips denominada los proyectos X (series X) del cual el desarrollo de un chip para enviar datos sobre corrientes portadoras era el proyecto número 10. Esta empresa comenzó a desarrollar este proyecto con la idea de obtener un circuito que pudiera ser insertado en un sistema mayor y controlado remotamente. En colaboración con BSR, una empresa dedicada

a los sistemas de audio, comenzaron a construir los dispositivos X10.

El primer módulo podía controlar cualquier dispositivo a través de la red eléctrica doméstica (120 o 220 V y 60 o 50 Hz) modulando pulsos de 120 KHz (0 = sin pulso, 1 = pulso). Con un simple protocolo de direccionamiento, podían ser localizados un total de 256 dispositivos en la red.

El protocolo soporta 16 grupos de direcciones denominados códigos de casa (desde la A hasta la P), y otras 16 direcciones para cada código de dispositivos, denominadas códigos de unidad. La comunicación se realizaba por cadenas de control, que son sucesiones de unos y ceros que completan los comandos. En su primera versión tan sólo existían seis operaciones, encender, apagar, aumentar, disminuir, todo apagado y todo encendido. Estas señales son recibidas en todos los módulos, pero sólo el módulo con la misma dirección que la indicada en el mensaje de control realizará alguna operación. El mensaje completo tiene 48 bits.

Posteriormente, los códigos de operación fueron extendidos a 256 con una cabecera especial, e incluso, la cantidad de información que porta un mensaje puede ser mayor de 48 bits si es usado el código de datos extendidos en la cabecera de control del mensaje, sin embargo, solo los dispositivos que soporten estos códigos extendidos responderán a estos.

2.1.3. Elementos de un sistema de control domiciliario

Un sistema de control domiciliario básico está compuesto principalmente por la red eléctrica domiciliaria, un controlador, un transceiver y los módulos receptores X10. La siguiente figura nos muestra los elementos que conforman un sistema de control domiciliario típico basado en tecnología X10:

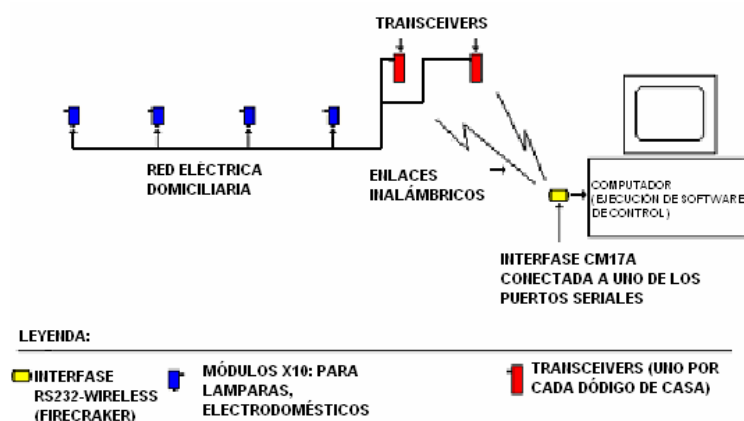


Figura 2.1. Configuración CM17A (Firecracker)

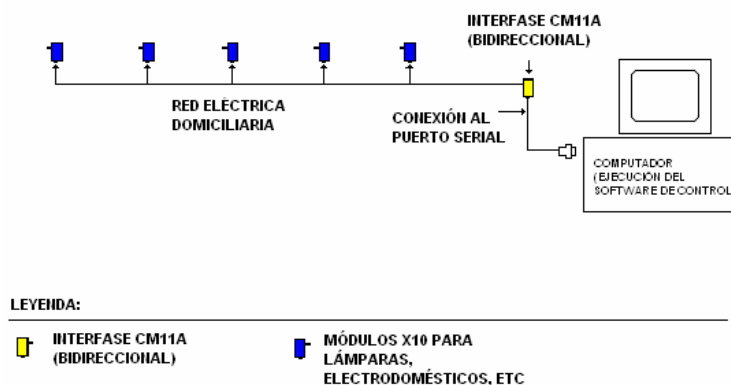


Figura 2.2. Configuración CM11A (bidireccional)

Como podemos ver, el controlador del sistema está conformado por la interfase y el computador, donde se ejecuta el software de control, aunque hay implementaciones que se realizan con pequeños controladores, basados en chips programables, que permiten el control remoto de los módulos y la programación de ciertas rutinas básicas de encendido y apagado de luces y electrodomésticos. Cuando el controlador está basado en un computador, es necesaria la presencia de una interfase para poderlo interconectar con el resto de los elementos del sistema. Estas interfases pueden ser unidireccionales o bidireccionales. Las primeras solo permiten el envío de los comandos X10 hasta los módulos para que sean ejecutados, en cambio, las interfases bidireccionales adicionalmente permiten el monitoreo del estado de los módulos X10. Como ya dijimos anteriormente, este proyecto incorporará el control mediante

mensajes escritos a los sistemas de control domiciliario basados en PC que ya se venden actualmente.

Transceivers

De estos dispositivos tenemos dos clases; los que reciben los comandos de forma inalámbrica y trabajan en conjunto con la interfase Firecracker, y los que reciben los comandos de forma alámbrica, el cual está incluido dentro la interfase bidireccional CM11A. En ambos casos, los transceivers toman la señal recibida y la acondicionan para ser transmitida por la red eléctrica domiciliaria.

Las transmisiones de los comandos X10 por la red eléctrica domiciliaria, deben estar sincronizadas con el cruce por cero de la señal AC de alimentación eléctrica, y debe ser lo más próxima posible al cruce por cero real. Los chips PL513 y TW523 están diseñados para conectarse a casi cualquier otro circuito microcontrolador que envíe los códigos X10 para ser inyectados en la señal AC. Por esta razón, los chips PL513 y TW523 deben proveer una señal de cruce por cero de referencia para el microcontrolador.

Para realizar la transmisión, una señal cuadrada representando la detección del cruce por cero es provista por el chip PL513/TW523, esta señal está dentro de un rango de ± 50 us (100 us) del punto del cruce por cero de la señal AC. La señal de salida del microcontrolador debe estar dentro del rango de hasta +50 us de este cruce por cero de referencia, es decir, la señal digital debe estar durar como mínimo un milisegundo para ser interpretada y debe ser recibida entre $-50 \mu\text{s}$ y $+100 \mu\text{s}$ del cruce por cero real de la señal AC. La señal X10 generada por el microcontrolador es aplicada a los chips PL513 ó TW523, los cuales modulan esta señal con una portadora cuya frecuencia es de 120 Khz y la acoplan de forma capacitiva a la red de alimentación eléctrica.

Un nivel alto por al menos de 1 ms que coincida con el cruce por cero, es interpretado como un "1" binario por los chips PL513 ó TW523 y permiten que la señal producida por el oscilador de 120 khz sea inyectada a la línea de alimentación.

Un nivel bajo por al menos de 1 ms que coincida con el cruce por cero, es interpretado como un "0" binario por los chips PL513 ó TW523 y evitan que la señal producida por el oscilador de 120 khz sea inyectada a la línea de alimentación

Pueden existir varios de estos módulos en un mismo sistema, sin embargo, deben estar configurados con distintos códigos de casa, de tal manera que puedan enviar los comandos de control a su respectivo conjunto de módulos receptores agrupados en un mismo código de casa.

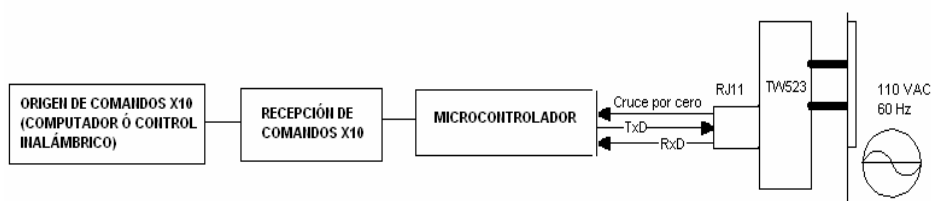


Figura 2.3 Diagrama de bloques de un transceiver X10 típico

Receptores

Como los transceivers, pueden comunicarse con 256 direcciones distintas. Cuando se usan computadoras como controladores del sistema, ciertos de estos dispositivos pueden reportar su estado.

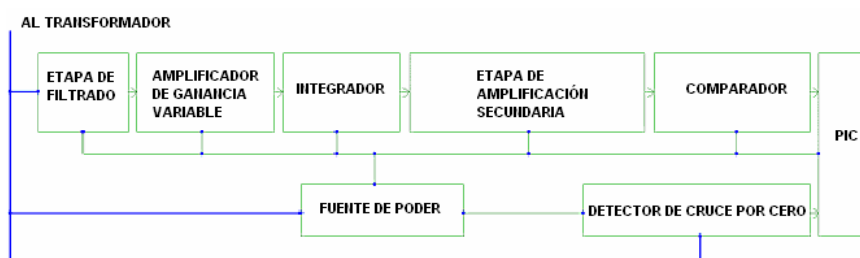


Figura 2.4. Diagrama de bloques de un receptor X10 típico

La figura 5 nos muestra el diagrama de bloques de un típico módulo receptor X10. El microcontrolador toma la señal digital que se obtiene luego de la etapa comparadora y ejecuta la acción indicada, es decir el micro controla a una etapa de interruptores, y de esta manera se logra el control del encendido y apagado de lámparas y electrodomésticos. En el capítulo 7 se muestra la implementación de cada bloque y la simulación del receptor.

Controladores

En gran parte, el controlador define el sistema entero. Ante todo, define el protocolo del sistema. Además, ajusta el envoltorio del funcionamiento del sistema en términos de expansión, flexibilidad y fiabilidad.

Ahora, vamos a definir un controlador como un dispositivo transmisor X-10 programable de software. Añadí 'software' a la definición con el fin de señalar que estos están conformados por un computador y por una interfase, para excluir consolas como mini-controladores, los cuales no trabajan en conjunto con un computador.

Existen controladores domiciliarios básicos, medios y de alto rendimiento, los cuales son implementados con la operación conjunta de las interfases respectivas y un computador. Este proyecto implementa el sistema de dos maneras: haciendo uso de un controlador básico fundamentado en la interfase Firecracker; y haciendo uso de un controlador de nivel medio, en base a la interfase CM11A bidireccional.

Los controladores básicos por lo general son unidireccionales y no pueden monitorear el estatus de los módulos X10, es decir, con ellos no podemos implementar el disparo de alarmas se seguridad, tal es el caso de Firecracker.

Los controladores de nivel medio, permiten el monitoreo de los dispositivos X10, y de esta manera se puede implementar un sistema básico de seguridad, tal es el caso de la interfase CM11A. Además estos pueden detectar posibles colisiones, las cuales pueden ocurrir cuando la interfase envíe un comando justo en el momento que los módulos X10 estén informando su estatus.

Los controladores de alto rendimiento redondean el análisis. Están caracterizados por hardware que realizan muchas de las funciones que controladores más pequeños sólo pueden hacer a través de su ordenador central o no hacerlo en absoluto. Estos pretenden parecer profesionales y están valorados de acuerdo con esto.

2.2. Teoría de transmisión para X10

X10 comunica al receptor y el transmisor mediante el envío y recepción de señales sobre el cableado de alimentación eléctrica. Estas señales envuelven ráfagas cortas las cuales representan la información digital. Los transmisores X10 están sincronizados con el cruce por cero de la señal AC de la línea de poder. Se trata de transmitir los bits lo más cerca posible del cruce por cero, de hecho se tienen aproximadamente 200 microsegundos para hacerlo.

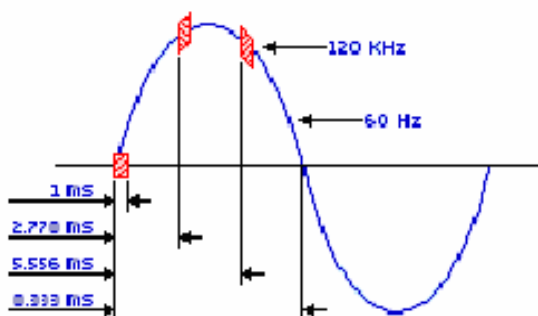


Figura 2.5 Esquema de transmisión X10

Un 1 binario es representado por una ráfaga de 1 milisegundo de duración con una frecuencia de 120Khz justo en el cruce por cero. En cambio, un cero binario se representa por la ausencia de dicha ráfaga. Ese 1 binario debe ser transmitido tres veces durante la mitad del ciclo de la señal AC de 60 Hz.



Figura 2.6 Transmisión de códigos de función y de dispositivo

La transmisión completa de un código X10 necesita 11 ciclos de la señal A.C. Los dos primeros ciclos son para el código de inicio de mensaje, 1110 (este código nunca cambia). Los cuatro siguientes son el código de casa, y los cinco siguientes con el código de unidad o de función.

Este bloque completo es transmitido dos veces, separadas cada una por tres ciclos de la señal AC. Esto es debido a que los módulos receptores X10 requieren un silencio de al menos 3 ciclos de la señal AC entre cada bloque de 11 bits y la única excepción a la regla son los códigos de bright y dim, estos son transmitidos de manera continua y

deben ser transmitidas de forma completa sin ningún ciclo intermedio, es decir de forma completa (al menos dos veces deben ser transmitidos estos bloques)

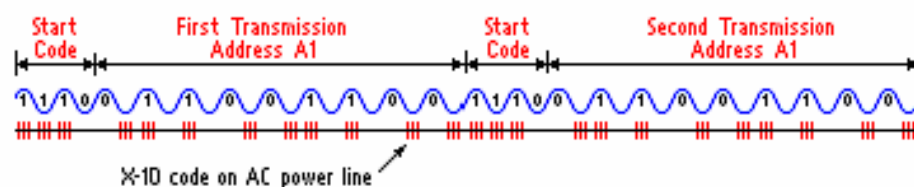


Figura 2.7 Transmisión complementaria de cada bit

Dentro de cada bloque de datos, cada cuatro o cinco bits deben ser transmitidos en una forma complementaria real durante ciclos alternados de la señal AC, es decir, la transmisión de un “1” binario se define como la transmisión de un pulso inmediatamente seguida por la ausencia de un pulso. De manera análoga, la transmisión de un “0” binario se define como la ausencia de un pulso inmediatamente seguida por la presencia de un pulso. Solamente el código de inicio es la excepción de esta regla, ya que siempre será 1110.

En resumen, lo que primero acontece es el envío del código de casa junto con el código del dispositivo. Como dijimos anteriormente este se debe enviar dos veces separadas cada una por tres ciclos de la señal AC, este tiempo entra cada bloque de bits es necesario para poder resetear los registros de desplazamiento presentes dentro del

microcontrolador de los módulos X10. Luego que ya ha sido direccionado el dispositivo destino, este se encuentra esperando por el código de función para ser ejecutado. Solo el dispositivo que haya sido direccionado previamente, se encuentra esperando por el código de función. Cabe señalar que el envío de cada bit se transmite tres veces para poder sincronizarse con cualquiera de las fases del sistema de alimentación.

2.3. Variantes del Protocolo X10:

Básicamente existen dos variantes del protocolo X10 que están vigentes actualmente en las distintas interfaces de comunicación con la PC:

- El protocolo CM17A
- El protocolo CM11A

El primero es utilizado por la interfase Firecracker para enviar de manera inalámbrica los comandos de control recibidos desde la PC, hasta el transceiver TM751, el cual transforma estos comandos y los inyecta a la red eléctrica domiciliaria para que sean ejecutados. El segundo es utilizado por la interfase CM11A (bidireccional), la cual conecta el puerto serial con la toma de alimentación eléctrica. Esta interfase recibe los datos enviados por la PC por el puerto serial y los

transforma directamente para que sean inyectados a la red eléctrica domiciliaria.

2.3.1. El protocolo CM17A

Como dijimos anteriormente la comunicación entre la interfase Firecracker y el módulo TM751 es inalámbrica. Las frecuencias utilizadas para la transmisión RF de los comandos pueden ser:

- 310 MHz - "A" – En Norteamérica
- 418 MHz - "U" – Europa y el Reino Unido
- 433.92 MHz - "E" – Europa

Excepto por la frecuencia de portadora, el protocolo de radiofrecuencia que X10 utiliza para transmisiones estándares es idéntico al protocolo de infrarrojos NEC IR, el cual es ampliamente utilizado para control remoto de electrodomésticos como televisores, radio grabadoras, equipos de sonido, etc. La estructura de una trama NEC se muestra a continuación:

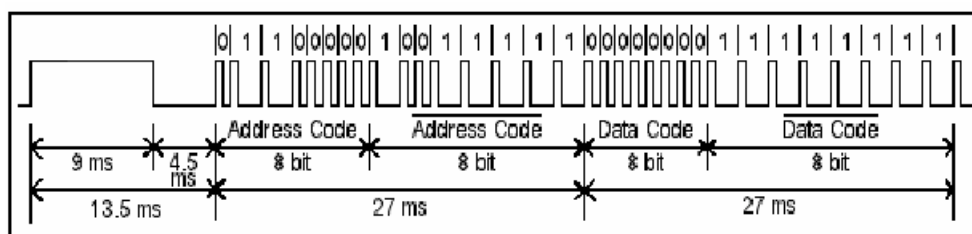


Figura 2.8 Formato para la transmisión inalámbrica

Cada trama empieza con una ráfaga de bits de 9 milisegundos de duración seguida por un silencio de 4.5 milisegundos, esto es necesario para sincronizar la comunicación con el receptor, el cual debe adaptarse dinámicamente a la potencia de la señal.

Después de este inicio, siguen 32 bits con un 1 binario representado por un espacio de 2.25 milisegundos entre dos picos de la señal y el 0 binario representado por un espacio de 1.125 milisegundos entre dos picos de la señal. El pico del último pulso, es decir el pulso número 33 marca el final del envío de los bits, luego sigue un silencio de aproximadamente 40 milisegundos. La mayoría de los transmisores X10 envían al menos 5 copias de esos 32 bits de control separados por 40 milisegundos, aunque algunos no envían dicha repetición.

En lugar de enviar la dirección y los bits de datos, X10 envía 2 bytes de datos, donde cada byte es seguido inmediatamente por su byte complementario para el control de errores. Para cada byte, el bit 7 se recibe primero y el bit 0 al final.

En la Figura 2.8, el primer byte recibido es 0110-0000, el segundo es 1001-1111, el tercero es 0000-0000, y el cuarto es

1111-1111-. Luego de examinar la complementariedad, los dos bytes recibidos son; 0110-0000 y 0000-0000 ó 0x60 y 0x00.

La Figura 2.8, describe la transmisión de códigos IR. Si se tratara de la transmisión de código X10, los dos bytes mostrados deberían representar 2 bytes de control según el protocolo CM17A.

En conclusión, el protocolo CM17A sirve para la transmisión de comandos X10 de manera inalámbrica hasta un dispositivo transceiver. La implementación física del protocolo X10 es idéntica a NEC IR, el cual ya ha sido utilizado para control remoto de dispositivos electrodomésticos.

Cabe señalar que esas palabras de control, antes de ser enviadas al transceiver TM751, son recibidas desde el puerto serial de la PC en la interfase Firecracker, utilizando para esto el mismo protocolo CM17A. La comunicación entre la interfase Firecracker y la PC es descrita en el capítulo 4.

En el anexo D se muestra una tabla con todas las palabras de control del protocolo CM17A.

2.3.2. El protocolo CM11A: Códigos de casa, códigos de dispositivos y códigos de función

Para implementar los códigos de casa se tienen 4 bits, por lo tanto son posibles 16 códigos de casa los cuales son identificados en los módulos por las letras desde la A hasta la P. Luego se tienen los códigos de dispositivos los cuales son implementados por 5 bits, por lo que deberían tenerse hasta 32 dispositivos, sin embargo, el bit menos significativo está siempre en cero para poder diferenciar de los códigos de función, los cuales también constan de 5 bits, con la diferencia que el bit menos significativo está siempre en uno.

Entonces se pueden direccionar hasta 16 dispositivos por cada código de casa. Una vez direccionado, el módulo X10 responderá a cualquier código de función. El módulo volverá a resetearse luego de detectar la primera nueva dirección después de haber recibido un código de función. En el anexo E se muestran las palabras de control para el protocolo CM11A.

Parámetros Seriales

Los parámetros seriales para la comunicación entre la interfase CM11A y la PC son los siguientes:

TABLA 2. Configuración de los parámetros seriales

Tasa de Baudios:	4,800bps
Paridad:	None
Bits de datos:	8
Bits de datos:	1

Para la conexión serial física se tienen las siguientes conexiones de cable:

TABLA 3. Conexión física entre la interfase y la PC.

Señal	Conector DB9	Conector RJ11
SIN	Pin 2	Pin 1
SOUT	Pin 3	Pin 3
GND	Pin 5	Pin 4
RI	Pin 9	Pin 2

Donde; SIN: Serial input to PC (output from the interface), SOUT: Serial output from PC (input to the interface), GND: Signal ground, RI: Ring signal (input to PC).

Transmisión Estándar X10

Una transmisión estándar X10 desde la PC hasta la interfase CM11A, se refiere a la comunicación del código de casa más el

código de dispositivo ó el código de función. El siguiente es el formato del diálogo entre la PC y la interfase:

- Primeramente la PC envía hasta la interfase una cabecera de 2 bytes
- Cuando la interfase recibe la cabecera enviada desde la PC, esta sumará todos los bytes y retornará un byte de suma de comprobación hasta la PC.
- La PC deberá entonces retornar una acuse de recibo, enviando un valor de 0x00 a la interfase para indicar que la transmisión está por empezar.
- Finalmente, la interfase envía 1 byte: 0x55, indicando que está lista para recibir (ready to receive). Luego de esto se realiza la transmisión de los códigos de casa más el código de dispositivo ó el código de función.

CAPÍTULO 3

3. CONECTIVIDAD FÍSICA Y LÓGICA ENTRE EL MÓVIL Y LA PC

3.1. Conectividad física

La mayoría de los teléfonos Nokia pueden comunicarse con un computador mediante los siguientes medios: enlace infrarrojo, cable DKU-5 (móvil-USB), DAU-9P (móvil – Puerto serie), DLR-3P (móvil- Puerto serie). En nuestro caso, para poder enviar al computador los mensajes de texto que contienen los comandos de control recibidos desde el teléfono celular remoto en el móvil local, un Nokia 7160, es necesaria la utilización del cable DLR-3P de Nokia.

3.1.1. El Cable DLR-3P

Los cables DB-9 y DLR-3P de Nokia son muy similares, ambos sirven para conectar los modelos de móviles 51XX, 61XX y 71XX con el puerto serie de un computador.

DAU-9P.- Este cable es dual, es decir, puede funcionar como MBUS o FBUS, más adelante hablaremos de estos dos protocolos. El cambio de un tipo de transmisión a otro puede hacerlo de forma automática o mediante un interruptor. Se puede utilizar para subir logos y melodías y liberar teléfonos. Este cable se utiliza con los Nokia 51xx/61xx y el software de módem del "Nokia Data Suite". Es posible utilizar dichos teléfonos para conectarse a Internet, utilizando dicho software.

DLR-3P.- Este es el cable que se utiliza con los teléfonos Nokia de la serie 62xx/71xx para intercambiar datos a mayor velocidad y conectarse a Internet. Como esos teléfonos tienen hardware propio para la función de módem, los drivers del "Nokia Data Suite" (módem software) no funcionan. Este cable también se puede usar con el LogoManager, el cual es una aplicación para subir y bajar logos y melodías. Existe otra versión del cable, más antigua y menos compatible denominada DLR-3.

Se intentó construir el cable DLR-3P, sin embargo esto no pudo realizarse debido a que este posee un microcontrolador que

actúa como un multiplexor RS-232 handshake entre la PC y el móvil utilizando para esto un protocolo propietario: FBUS. El cable se conecta en uno de sus extremos con el móvil. La siguiente figura presenta los pines presentes en el móvil para el respectivo conector.



Figura 3.1. Pinout en el puerto del móvil Nokia 7160

La siguiente tabla nos muestra la función de cada uno de los pines y los niveles de voltaje respectivos:

TABLA 4. Función de cada pin de la interfase del teléfono móvil

Pin	Name	Función	sonstiges	in / out
1	CHARGE	Charging voltage (Voltaje de carga)	0..8.5V, 0..850mA	in
2	CCONTROL	Charging control (Control de carga)	PWM, 32..37 Hz, 1..99%	out
3	MIC /XMIC	Analog audio in (Entrada analógica de audio)	1Vpp / 100 Ohm	in
		Analog audio in (Headset)	200mVpp / 2.5 KOhm / BIAS 100..600uA	in
		Accessory mute	> 2.5V not muted < 1.5V muted	out

		Headset detection	> 1.5V no Headset / Data mode	in
			< 1.3V Headset connected	
4	AGND	Analog ground (Referencia analógica) Power Source for DLR3-cable	6210 : 2.8 V, 7110 :3.3 V	out
5	EAR / XEAR	Analog audio out (Salida analógica de audio)	U = 80mV - 1.0V	out
		Accessory detection	< 0.2V : Headset connected 0.5V: Accessory connected 2.8V : no accessory connected	in
		PTT-Button	connected to AGND when pressed	in
6	MBUS	Bidir. serial bus (Bus bidireccional)	9600 bps	in / out
7	FBUS RX	Serial data in (Entrada serial de data)	9600 - 230.400 bps < 0.8V : '0' > 2.0V : '1'	in
8	FBUS TX	Serial data out (Salida serial de datos)	9600 - 230.400 bps < 0.8V : '0' @ max. 4mA > 1.7V : '1' @ max. 4mA	out
9	SGND	Signal ground / Charging ground (Referencia para las señales, referencia para carga)		

El otro extremo del cable posee un conector tipo hembra que se conecta con el puerto serial. A continuación mostramos los pines utilizados por el conector:

TABLA 5. Conexión Serial

Señales	DB9 pin	DB25 pin
GND	5	7
TxD	3	2
RxD	2	3
RTS	7	4
DTR	4	20
DCD	1	8

La conectividad física se complementa mediante los protocolos FBUS y MBUS de Nokia. En general, cada fabricante tiene su propio mecanismo para comunicarse con los computadores u otros dispositivos, por lo que estos protocolos utilizados por Nokia no se pueden usar con ningún otro dispositivo móvil de otro fabricante.

3.1.2. Los Protocolos FBUS y MBUS

Los métodos de comunicación llamados se usan FBUS y MBUS se utilizan en los teléfonos móviles Nokia para la transmisión de

los datos, reparación de daños a nivel de software del móvil, y ajustes.

MBUS.- Realiza la comunicación a través de un bus bidireccional implementado con un hilo desde un solo pin en el puerto del teléfono móvil para poder enviar y recibir datos en el móvil, por lo tanto, la comunicación es half-dúplex. Con MBUS es posible la comunicación con casi todos los teléfonos móviles Nokia para el servicio y propósitos de ajuste, sirve para subir logos y melodías, pero es un método muy lento de comunicación.

Alcanza velocidades de transmisión de hasta 9600bps y solo trabaja en modo half-duplex. Solo se usan dos pines del puerto del teléfono móvil; uno para los datos y otro para la referencia (GND). Los parámetros seriales para la comunicación a través del M-BUS son: 8 bits de datos, paridad impar, un bit de parada.

FBUS.- Es una más nueva solución y ofrece una comunicación en modo full-dúplex de gran velocidad para conexión entre el teléfono y la computadora. El servicio y funcionamientos de ajuste que se hacen tradicionalmente utilizando MBUS también

están disponibles vía FBUS, pero de manera mucho más rápida. La comunicación utiliza un pin para transmitir, FBUS-TX y otro para la recepción, FBUS-RX, además de un pin de referencia. Es muy similar a un puerto serial estándar. La comunicación se realiza a 115.200bps y los parámetros seriales son; 8 bits de datos, no paridad, y un bit de parada.

La señal DTR del puerto serial debe estar configurada siempre en alto, puesto que es esta señal la que alimenta la pequeña interfase existente en el cable. Esta interfase está compuesta por un convertidor de niveles de voltaje RS-232 a TTL, además del microcontrolador que maneja la comunicación con el teléfono, esta era una de las limitantes para la construcción del cable por nuestros propios medios.

Como dijimos anteriormente, estos protocolos son propietarios de Nokia, y por este motivo, no es posible obtener las palabras de control que permiten la comunicación entre el móvil y el computador. Sin embargo, se pudo averiguar, aunque sin detalles, como se realiza la comunicación entre la PC y el móvil a través de FBUS.

Primeramente, se sincroniza el microcontrolador de la interfase del cable con el computador, para hacerlo, se envía varias veces, aproximadamente 128, una cadena de caracteres. Esto se hace para alistar al bus para el envío posterior de las tramas de bits.

Formato de trama.- Todas las tramas enviadas a través del cable DLR-3P empiezan con el carácter 0x1E. Este es el ID para las tramas FBUS. Si se utiliza el enlace infrarrojo, el ID varía a 0x1C.

El siguiente byte indica la dirección de destino, si estamos enviando datos hasta el móvil, la dirección de destino es la del móvil, que es siempre 00. Luego se tiene la dirección de origen, en el caso de enviar datos, esta será la del computador, la cual se fija en 0x0C. El tercer byte indica el tipo de mensaje o comando. Los bytes 4 y 5 indican la longitud del mensaje, donde el byte 4 es el byte más significativo, y el byte 5 es el byte menos significativo. En el byte 6 empieza el segmento de datos. Por último, se tiene la finalización de la trama enviada, compuesta por un byte para el número de secuencia y el byte para la comprobación de errores.

FRAME ID	DESTINATION ADDRESS	SOURCE ADDRESS	TYPE	FRAME LENGTH	DATA	SEQUENCE NUMBER	FCS
----------	---------------------	----------------	------	--------------	------	-----------------	-----

Figura 3.2. Formato de trama FBUS

3.2. CONECTIVIDAD LÓGICA

Como se logra la comunicación entre el móvil y la PC? En este apartado se explica como el software Nokia Connection Manager nos permite la conexión móvil-PC no solo vía cable DLR-3P sino haciendo uso de otros medios como Bluetooth e infrarrojo.

3.2.1. NOKIA CONNECTION MANAGER

Este es un software que facilita la comunicación entre el móvil y el computador, permitiendo la conectividad lógica para distintos medios físicos: enlace infrarrojo, cable móvil-puerto serial, bluetooth y cable móvil-puerto USB.

Permite la fácil configuración del puerto que se va a utilizar y la configuración automática de los parámetros de comunicación para el puerto seleccionado.

En el caso de la comunicación vía el puerto serial, el Nokia Connection Manager se encarga de la sincronización entre el

microcontrolador existente en la interfase del cable DLR-3P y el computador.

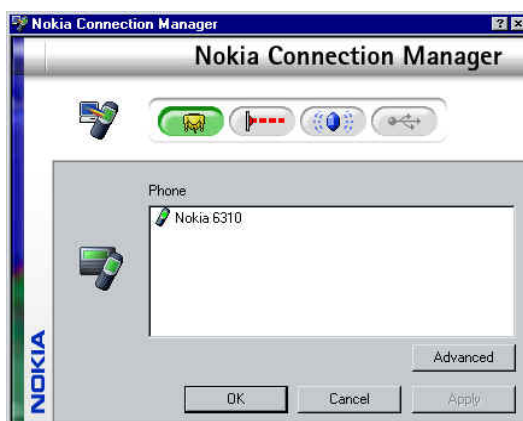


Figura 3.3. Interfase gráfica del software Nokia Connection Manager

3.2.2. EL KIT DE DESARROLLO DE SOFTWARE DE NOKIA: SDK BETA 3.0

El Nokia PC Connectivity SDK es una fácil y sofisticada interfase para la programación de teléfonos móviles Nokia. Consiste en un conjunto de librerías, cada una de las cuales realiza un determinado conjunto de tareas relacionada con la funcionalidad del teléfono.

Todas las librerías son implementadas como librerías COM (Component Object Libraries), este es el nombre que la tecnología orientada a objetos de Microsoft da a su estándar para la integración entre componentes de software.

Una aplicación cliente explota estas librerías a través de objetos, también llamadas librerías de tipos en algunos contextos. Una librería de objetos puede ser considerada como una descripción binaria de la librería de componentes. Muchos ambientes de desarrollo pueden soportar estas librerías: Visual Basic, Visual C++, Delphi, Visual J++, y otros.

Cada librería en el SDK de Nokia contiene uno o más entidades funcionales llamadas componentes. Estos componentes de software pueden ser reutilizados y presenta su funcionalidad a través de un conjunto definido de interfaces. Una aplicación cliente crea una instancia del componente, configura una referencia a la interfase deseada y accede a los métodos a través de esta referencia. Una interfase contiene una colección de propiedades, métodos y funciones relacionadas con una funcionalidad específica, agrupadas bajo una sola denominación. Las interfaces son divididas en dos categorías

de acuerdo al lugar donde los métodos sean invocados. En este sentido las interfaces son de salida y de entrada.

Los métodos de las interfaces de entrada son implementadas en los componentes objetos y reciben llamadas de clientes externos. El objeto realiza el servicio solicitado y retorna el resultado al cliente. La mayoría de las interfaces en estas librerías son interfaces de entrada las cuales son llamadas por la aplicación cliente. Los métodos o eventos de las interfaces de salida son implementadas en el sink del cliente y estos reciben la llamada del objeto. El objeto define la interfase que desea usar, y el cliente la implementa, de esta manera, las interfaces de salida permiten que el objeto pueda responder a su cliente. Las interfaces de salida también son usualmente utilizadas para notificar al cliente cuando algo importante está ocurriendo en su medio ó para informar al cliente cuando una operación de modo asíncrono ha sido completada. Las interfaces de salida son también llamadas puntos de conexión, interfaces de eventos, interfaces de notificación, o interfaces fuentes. A continuación las librerías instaladas del SDK de Nokia:

Library NokiaBTMM

C:\Archivos de programa\Archivos
comunes\Nokia\Transports\NCLBTMM.dll

Nokia Bluetooth Media Module

Descripción: Librería para el manejo de conexión bluetooth
entre el teléfono y la PC.

Library NokiaCLCalendar

C:\Archivos de programa\Archivos
comunes\Nokia\Adapters\NclCal.dll

NokiaCL Calendar

Descripción: Librería para el manejo de calendario (disponible
solo en ciertos modelos)

Library NokiaCLCall

C:\Archivos de programa\Archivos
comunes\Nokia\Adapters\NclCall.dll

NokiaCL Call

Descripción: Librería para el manejo de llamadas de voz.

Library NokiaCLCapabilityService

C:\Archivos de programa\Archivos
comunes\Nokia\Services\NclCapability.dll

NokiaCL Capability Service

Descripción: No disponible.

Library NokiaCLFileTransfer

C:\Archivos de programa\Archivos
comunes\Nokia\Services\NclFT.dll

NokiaCL File Transfer

Descripción: Para la transferencia de archivos entre el móvil y la
PC.

Library NokiaCLMessaging

C:\Archivos de programa\Archivos
comunes\Nokia\Adapters\NclMsg.dll

NokiaCL Messaging

Descripción: Permite el envío y recepción de mensajes escritos,
además del manejo de la memoria SMS.

Library NokiaCLSettings

C:\Archivos de programa\Archivos
comunes\Nokia\Adapters\NclSet.dll

NokiaCL Settings4

Descripción: Librería que permite manipular las configuraciones
del teléfono

Library NokiaCLSMLWrapper

C:\Archivos de programa\Archivos
comunes\Nokia\Services\NclSMLWrap.dll

NokiaCL SML Wrapper

Descripción: No disponible.

Library NokiaCLTaskJournal

C:\Archivos de programa\Archivos
comunes\Nokia\Adapters\NclTJ.dll

NokiaCL Task Journal

Descripción: No disponible.

Library NokiaCLVoice

C:\Archivos de programa\Archivos
comunes\Nokia\Adapters\NCLVoice.dll

NokiaCL Voice

Descripción: Para el manejo de mensajes de voz

Library NokiaCLWAP

C:\Archivos de programa\Archivos
comunes\Nokia\Adapters\NclWAP.dll

NokiaCL WAP

Descripción: Para el manejo de las configuraciones WAP.

Library NokiaRS232MM

C:\Archivos de programa\Archivos
comunes\Nokia\Transports\NCLRSMM.dll

Nokia RS-232 Media Module

Descripción: Permite la comunicación sería entre el teléfono y el
PC.

Library NokiaUSBMM

C:\Archivos de programa\Archivos
comunes\Nokia\Transports\NCLUSBMM.dll

Nokia Media Module USB

Descripción: Para la comunicación USB entre el teléfono y la
PC.

Library PhonebookAdapterDS3

C:\Archivos de programa\Archivos

comunes\Nokia\Adapters\SCM3aS.dll

Nokia Phonebook Adapter (Data Suite 3.7)

Descripción: Para el manejo de la memoria del directorio telefónico, marcación rápida, y grupos.

Library SMS3ASuiteLib

C:\Archivos de programa\Archivos

comunes\Nokia\Adapters\Sms3aS.dll

Nokia SMS Adapter (PC Suite 3.7)

Descripción: Permite el envío y recepción de mensajes escritos, además del manejo de la memoria SMS (solo para teléfonos GSM)

Library STTINGS3A_SLib

C:\Archivos de programa\Archivos

comunes\Nokia\Adapters\Stngs3AS.dll

Nokia Settings Adapter for PC Suite 3.7

Descripción: Librería que permite manipular las configuraciones del teléfono.

Para la realización de este proyecto se hizo uso de los objetos pertenecientes a la librería NokiaCLMessaging. Esta librería contiene los componentes para el manejo de los mensajes SMS, la configuración SMS de los teléfonos Nokia para los sistemas GSM y TDMA. Con esta librería es posible la creación de aplicaciones que manejen mensajes SMS. La siguiente figura muestra la organización de los objetos y componentes en esta librería:

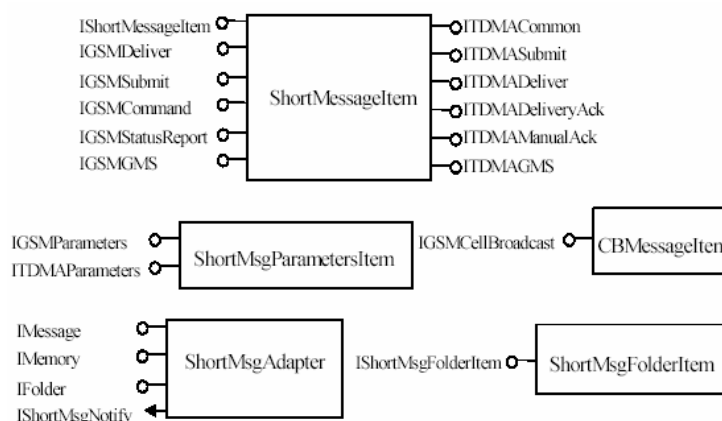


Figura 3.4. Componentes de la librería NokiaCLMessaging

De estos componentes, la librería que utilizamos fue la librería ShortMessageAdapter de la cual se usaron las interfases IMessage e IMemory, y ShortMessageItem, de la cual se usó la interfase IShortMessageItem. En la primera interfase están

implementados los métodos para enviar, recibir y guardar los mensajes escritos. Estos métodos son:

TABLA 6. Métodos de la interfase IShortMessageItem

MÉTODO	DESCRIPCIÓN
SendSMS	Envía mensaje SMS
ReadSMS	Recibe mensaje SMS
SaveSMS	Guarda mensaje SMS
RequestCB	Habilita o deshabilita el monitoreo de mensajes de broadcast de celda
EnableNotifications	Habilita el monitoreo de eventos SMS
DisableNotifications	Deshabilita el monitoreo de eventos SMS

De todos estos métodos se utilizaron los dos primeros: SendSMS y ReadSMS, además de GetMemoryStatus, el cual es parte de la interfase IMemory. Para poder utilizar estos métodos, tenemos primero que hacer referencia a la librería ShortMessageAdapter, además de definir las variables que vamos a utilizar como el tipo de datos definido en la librería.

SendSMS.- Envía un mensaje escrito. Este método es parte de NokiaCIMessaging.ShortMessageAdapter.Imessage.

La sintaxis de este método es la que sigue:

SendSMS ([in]*ShortMessageRouteType* SMSRouteType, [in]
pIShortMessageItem);

Donde SMSRouteType es un valor o constante que indica el ruteo que va a ser utilizado para el ruteo del mensaje escrito; *pIShortMessageItem* es el objeto mensaje al cual antes de ser enviado deben primero definírsele ciertas propiedades como: Destination Number, Validity, etc.

ReadSMS.- Lee el mensaje desde la ubicación específica de la memoria del teléfono. Este método es miembro de NokiaCIMessaging.ShortMessageAdapter.IMessage. La sintaxis de este método es la que sigue:

ReadSMS ([in]*ShortMessageMemory* SMSMemory, [in]
ByteFolderID, [in]*ShortLocation*, [out,retval]
pIShortMessageItem);

Donde SMSMemory es una constante indicando el tipo de memoria que va a ser accedido: Default o Sim. *ByteFolderID* debe señalar el ID de la carpeta donde se encuentra el mensaje. *ShortLocation*, indica el índice del mensaje en dicha carpeta y finalmente *pIShortMessageItem* es el objeto mensaje que es recuperado desde la memoria.

GetMemoryStatus.- Este método obtiene información de la memoria Default o Sim del móvil. Es miembro de NokiaCIMessaging.ShortMessageAdapter.Imemory. La sintaxis es la siguiente:

```
GetMemoryStatus ([in] ShortMessageMemory SMSMemory,  
[out] MemoryCapacity, [out] Messages, [out] UnReadMessage);
```

Donde SMSMemory es una constante indicando el tipo de memoria que va a ser accedido: Default o Sim. MemoryCapacity, salida que indica la capacidad de almacenamiento de dicha memoria. Messages, salida que indica la cantidad de mensajes escritos almacenados en la memoria. UnreadMessages, salida que indica el número de mensajes escritos que no han sido leídos.

Este último método se utilizó con la finalidad de estar verificando continuamente el número de mensajes no leídos, de tal manera que cuando este número aumentaba, esto significaba que un mensaje nuevo había llegado y procedíamos a revisarlo. Cabe indicar que a pesar de tener el software de conectividad móvil-computador y el medio adecuado, no todos los modelos de móviles Nokia son compatibles con estas librerías de desarrollo de aplicaciones, es por esto que se tuvo que implementar la detección de los mensajes de llegada de la

forma descrita arriba, es decir, usando el método GetMemoryStatus, ya que el método EnableNotifications no era compatible con nuestro modelo de teléfono. La siguiente tabla muestra los modelos que son compatibles con la mayoría de las librerías de desarrollo del SDK, digo la mayoría puesto que para algunos modelos, aún ciertas interfases y métodos no son compatibles.

TABLA 7. Lista de teléfonos compatibles con el SDK Beta 3.0 de Nokia

Teléfono	Tipo	Conección	Tipo de Red
Nokia 3320	NPC-1	IRDA	TDMA
Nokia 3360	NPW-1	IRDA	TDMA
Nokia 6210	NPE-3	IRDA/DLR-3P	GSM
Nokia 6250	NHM-3	IRDA/DLR-3P	GSM
Nokia 6310	NPE-4	IRDA/DLR-3P	GSM
Nokia 6310i	NPL-1	IRDA/DLR-3P	GSM
Nokia 6340	NPM-2	IRDA/DLR-3P	GSM
Nokia 6360	NPW-2	IRDA/DLR-3P	TDMA
Nokia 6370	NHP-2	IRDA/DLR-3P	GSM
Nokia 6385	NHP-2	DKU-5	GSM

Nokia 6510	NPM-9	DKU-5	GSM
Nokia 6590	NSM-9	DKU-5	GSM
Nokia 6610	NHL-4U	IRDA/DLR-3P	GSM
Nokia 6650	NHM-1	IRDA/DLR-3P	GSM
Nokia 7110	NSE-5	IRDA/DLR-3P	GSM
Nokia 7160	NSW-5	IRDA/DLR-3P	TDMA
Nokia 7190	NSB-5	IRDA/DLR-3P	GSM
Nokia 7210	NHL-4U	IRDA/DLR-3P	GSM
Nokia 8210	NSM-3	IRDA	GSM
Nokia 8290	NSB-7	IRDA	GSM
Nokia 8310	NHM-7	IRDA	GSM
Nokia 8390	NSB-8	IRDA	GSM
Nokia 8810	NSB-6	IRDA	GSM
Nokia 8850	NSM-2	IRDA	GSM
Nokia 8890	NSB-6	IRDA	GSM
Nokia 8910	NHM-4	IRDA	GSM

Es importante indicar que los sistemas operativos con el Nokia PC Connectivity SDK 3.0 Beta son: Windows 95, Windows 98, Windows 98SE, Windows NT4.0 and Windows 2000, y Windows XP Professional.

CAPÍTULO 4

4. SISTEMA DE CONTROL DOMICILIARIO: HARDWARE

4.1. Elementos físicos del sistema de control domiciliario

El sistema que estamos implementando tiene elementos físicos y de software. La siguiente figura nos muestra un esquema de los elementos de hardware presentes en el sistema de control basado en la interfase unidireccional Firecracker.

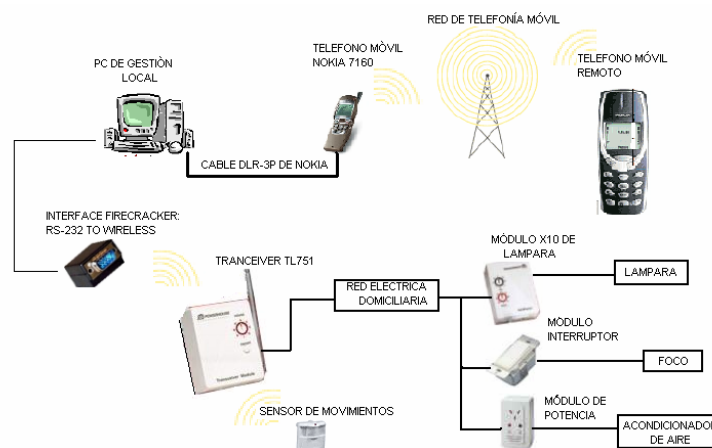


Figura 4.1. Esquema del sistema a nivel de hardware

4.1.1. El teléfono remoto, el teléfono local, la red de telefonía móvil

Como ya dijimos, en el mercado actualmente existen dispositivos para obtener un cierto grado de control sobre los electrodomésticos y luces presentes en un domicilio, si embargo, nuestro proyecto al proveer un elemento adicional de control: los mensajes escritos, tiene que hacer uso de la red de telefonía móvil celular.

Recordemos que el teléfono local es un modelo TDMA Nokia 7160, por lo que solo podrá ser activado en un proveedor que tenga una red TDMA, como es el caso de Porta y BellSouth. Sin embargo, el teléfono remoto puede ser de cualquier marca y trabajar en la red GSM de Porta ó en la red CDMA de BellSouth, puesto que como se vio al inicio, los mensajes escritos son almacenados en el SMSC hasta que son enviados hasta el móvil destino, aún cuando este se encuentre en una red de diferente tecnología que el móvil origen. También vimos que la transmisión de los mensajes es segura en cuanto a la llegada del mensaje a su destino, haciendo que el control remoto del sistema sea efectivo.

4.1.2. La PC de gestión local, el cable de conexión con el móvil

El computador para la gestión local del sistema se conforma como el controlador del mismo, en él se ejecuta el software de control, sin embargo el computador trabaja en conjunto con la interfase X10, que como vimos puede ser unidireccional o bidireccional, como es el caso de Firecracker o CM11A, respectivamente.

Además el computador de gestión local debe realizar la función de interfase de usuario, permitiéndole ingresar las variables para configuración del sistema e identificación de los dispositivos a ser controlados. El computador es el encargado de conectar el sistema con el mundo exterior a través del teléfono móvil local. El teléfono y el computador se interconectan a través del cable DLR-3P de Nokia, del cual se hizo ya una descripción en capítulos anteriores.

El computador debe poseer las características físicas para poder albergar cualquiera de los sistemas operativos compatibles con el software de conectividad Nokia Connection Manager y con el SDK de Nokia, puesto que el software de

control esta basado en dichas librerías. Entre las características físicas necesarias tenemos:

- Procesador Pentium 133 MHz o mayor
- 64 MB RAM o más.
- 1 GB de espacio libre o más.
- Monitor VGA ó SVGA
- 12X CD-ROM, como mínimo.
- Dos puertos seriales disponibles

Se puede observar, que entre las características físicas del computador, tenemos la necesidad de dos puertos seriales, esto se debe a que uno de ellos tendrá conectado el cable DLR-3P y el restante se conectará a la interfase Firecracker ó en su defecto a la interfase CM11A bidireccional.

4.1.3. La interface firecracker: RS-232 to wireless y el tranciever TM751

La interfase Firecracker es un pequeño dispositivo DB9 que se conecta al puerto serial de la PC. Esta se hace uso de un protocolo impar para comunicarse con la PC. La interfase Firecracker utiliza la señal Data Terminal Ready (DTR) y Ready To Send (RTS) para la alimentación de energía y leer los datos

enviados por la PC respectivamente. Debido a que no utiliza las demás señales del puerto serial, es posible conectar otros dispositivos en el mismo puerto serial de la PC, siempre y cuando no utilicen las señales descritas arriba, sin embargo es probable que el software que esté utilizando no le permita usar el mismo puerto para comunicarse con dos dispositivos diferentes.

Debido a que la interfase Firecracker usa la señal DTR para su alimentación eléctrica, es necesario que esta esté siempre en un nivel alto de voltaje (+5 Voltios) para mantener a la interfase encendida.

Por otro lado, la interfase Firecracker se comunica de manera inalámbrica con el módulo transceiver TM751, enviándole los comandos X10 recibido por parte del computador.



Figura 4.2. La interfase Firecracker

Protocolo

Como se mencionó arriba, la interfase se comunica con el computador haciendo uso de un protocolo de comunicación impar. Hay cuatro diferentes estados en los que pueden estar las dos líneas utilizadas del puerto serial. Cuando ambas líneas, DTR y RTS están en alto, la interfase se encuentra en el modo de espera, StandBy, y cuando ambas líneas están en un nivel bajo, la interfase se encuentra en el modo de Reset.

Para enviar un 'uno' lógico a la interfase se debe poner la señal DTR en bajo, y para enviar un 'cero' lógico se debe poner la señal RTS en un nivel bajo.

TABLA 7. Estados de las líneas RTS y DTR

Estado	RTS	DTR
StandBy	1	1
1 lógico	1	0
0 lógico	0	1
Reset	0	0

No existe temporización para el envío de los comandos, aunque es recomendable mantener un estado por al menos 5 ms.

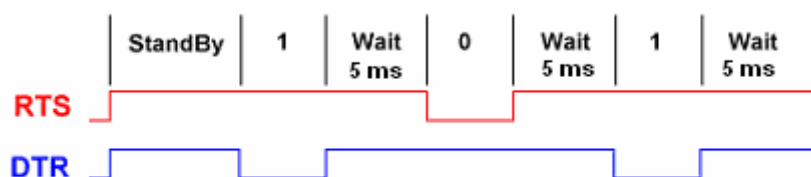


Figura 4.3. Transmisión de bits a la interfase

Cada transmisión esta conformada por 40 bits. Primeramente se envían una cabecera de 16 bits, luego 16 bits de datos y finalmente 8 bits de finalización. Los bits de inicio y finalización tienen siempre los mismos valores de D5 AA Hex y AD Hex respectivamente, pero los bits de datos varían dependiendo que palabra de control que se envíe, según el protocolo CM17A.

El transceiver TM751

Este módulo es un a parte integral del sistema de control domiciliario. Se encarga de recibir de forma inalámbrica los comandos X10 enviados por el computador a través de la interfase Firecracker, y acondicionarlos para ser enviados por la red eléctrica domiciliaria hasta los diferentes módulos X10, los cuales están conectados a las lámparas, luces y electrodomésticos. Si se requiere integrar un número de módulos que superen los 16, entonces se tendrá que usar otro transceiver y configurarlo con un distinto código de casa.



Figura 4.4. El transceiver TM751

La frecuencia de operación es de 310Mhz, y gracias a que la transmisión de los comandos se realiza utilizando una potencia muy baja, esta no interfiere con otros dispositivos de radiofrecuencia que pueden estar presentes en el hogar.

4.1.4. Los módulos receptores X10 para lámparas LM465, para focos WS12A, y de potencia HD245-C

El módulo de lámparas LM465

Este módulo puede se encendido, apagado, o modificar luminosidad a través de cualquier controlador compatible con X10. Responde a los comandos “All lights On” y “All Units OFF”, el cual enciende todos las lámparas, y apaga todas las unidades sin importar cual sea el código de dispositivo asignado. Este módulo también posee una característica

llamada control local que permite al usuario controlar de manera manual el encendido y apagado de la lámpara.

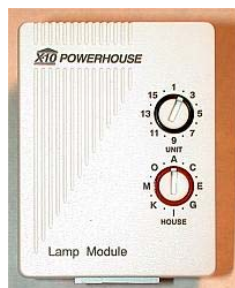


Figura 4.5 El módulo para lámparas LM465

Los módulos para focos WS12A

Este módulo está diseñado para controlar cargas incandescentes dentro de un rango de 40W y 500W. Cargas menores de 40W, actuarán de forma errática. Este módulo puede reemplazar un simple interruptor de pared que controle una lámpara de techo o una lámpara externa. Un botón pulsador se provee en cada interruptor para permitir el control local de encendido y apagado, además del control de intensidad.



Figura 4.6. El módulo para focos WS12A

Los módulos de potencia HD245

Estos módulos se conectan a la toma de 220V, 15 ó 20A, y controlan cualquier carga que trabaje con 220V y 15A ó menos. Están diseñados para trabajar en sistemas monofásicos de 110/220V ó 120/240V, puesto que este tipo de sistema está presente en la mayoría de las instalaciones eléctricas domiciliarias. Cabe señalar que estos módulos no trabajan en sistemas trifásicos.



Figura 4.6 Módulo de potencia HD245

Pueden ser usados para cargas como aires acondicionados, calentadores de agua, y jacuzzis. Por razones de seguridad,

estos módulos no responden al comando "All Lights On", prender todas las luces, sin embargo, si responden al comando "All Units Off", todas las unidades apagadas. Este módulo no posee control local.

4.1.5. El Sensor de Movimientos EAGLE-EYE

El sensor de movimientos EagleEye CH3005E envía una señal de radio frecuencia (RF) al transceiver TM751. El transceiver recibe la señal del sensor y convierte la misma en una señal eléctrica, que es enviada a través del cableado eléctrico domiciliario existente. La señal es recibida por los módulos de lámparas y electrodomésticos que son los encargados de controlar el encendido, apagado y control de intensidad de los mismos. Esta misma señal de actividad llega también a la interfase bidireccional, disparándose un evento planificado en una macro anteriormente cargada en la interfase.



Figura 4.7. Funcionamiento del sensor Eagle-eye

Este sensor está construido con una carcasa que lo protege los elementos, para monitorear ambientes exteriores o expuestos. Permite la desactivación de la foto celda interna a través de los botones de programación, para poderlo utilizar las 24 horas del día con luz o si luz solar. Tiene la facilidad de encender la iluminación, artefactos o bien macros de programación con solo su presencia, y luego desactivar los mismos a los 60 segundos de haber dejado la habitación o la zona de detección como mínimo, y 255 minutos como máximo, es decir, el tiempo de demora es ajustable entre 1 minuto y 255 minutos.



Figura 4.8. El sensor de movimientos Eagle-eye

Características:

- Hasta 20' pies de distancia de transmisión.
- Tecnología infrarroja pasiva
- Dimensiones 2.5 x 2.5 pulgadas
- Requiere 2 AAA baterías

Si configuramos el sensor Active Eye con un único código, por ejemplo; código de casa A y código de unidad 1, y si caminamos dentro de la zona de detección del sensor, cualquiera de los módulos X10 configurados con el ID A1 se activarán y si estos estuvieran controlando las luces, entonces encenderán las mismas durante el tiempo configurado en el sensor Antive Eye.

4.1.6. El Módulo CM11A

Este módulo no es realmente un receptor X10. Como ya habíamos dicho, CM11A es una interfase bidireccional, que al trabajar en conjunto con el computador, se conforman como un controlador de nivel medio para el sistema de control domiciliario.

Esta interfase puede ser programada con a través del puerto serie del computador. La rutinas de eventos y macros de activación, ajuste y desactivación de artefactos en forma temporizada o secuencial, se programan y luego se descargan en la memoria interna de la interfase, manteniendo luego, independencia entre el computador y la interfase, es decir una vez almacenada las secuencias y rutinas podemos desconectar

el controlador de la PC y conectarlo en cualquier tomacorriente su casa, y este estará preparado para controlar toda el domicilio. La interfase monitorea las señales X10 presentes en la red eléctrica.

Cuando se detecta una señal X10, la interfase captura los datos y los almacena en un buffer temporal. Este buffer puede almacenar hasta diez transmisiones X10. Luego estos datos son leídos por el computador a través del puerto serial haciendo uso del protocolo adecuado, del cual se habló en los capítulos anteriores.

La interfase tiene forma cúbica y se conecta a una toma de alimentación de 110V. Se debe procurar que tanto la interfase como el computador se encuentren conectados en la misma toma, para asegurar que ambos posean la misma referencia, y así no tener errores en los datos transmitidos. La interfase viene con un cable que posee en uno de sus extremos un conector RJ11 y en el otro extremo un conector DB9 para conectarse al puerto serial del computador.



Figura 4.9. La interfase CM11A bidireccional

La interfase trabaja con baterías, las cuales le proveen de una alimentación de respaldo en caso de que el computador ya no este encendido ó en caso de que ocurra un apagón temporal.

La interfase trae consigo un CD, el cual posee software que permite programar las macros, monitorear los dispositivos X10, generar señales de salida. Para nuestro proyecto no se utilizarán estos programas puesto que realizaremos nuestras propias rutinas en Visual Basic.

CAPÍTULO 5

5. SISTEMA DE CONTROL DOMICILIARIO: SOFTWARE

5.1. Desarrollo de la Aplicación SMSControl en Visual Basic 6.0

Este es el punto donde se centró nuestro trabajo, desarrollar una aplicación que incorpore los mensajes escritos SMS como elemento de control, además de utilizar las facilidades de los distintos protocolos X10: CM17A y CM11A.

Se desarrollaron dos aplicaciones, una basada en el protocolo CM17A utilizando la interfase unidireccional Firecracker, por lo que estamos hablando de un controlador de nivel básico. La segunda basada en el protocolo CM11A utilizando la interfase bidireccional del mismo nombre. Ambas aplicaciones en su esquema general son muy similares. Ambas utilizan las mismas librerías del SDK de Nokia, sin embargo solo la aplicación basada en CM11A realiza la notificación

remota del estado de los dispositivos, además de alarmas, a través del envío de mensajes escritos hasta el celular remoto.

La aplicación SMSControl basada en la interfase Firecracker

Esta aplicación fue desarrollada en Microsoft Visual Basic 6.0, por ser una herramienta de desarrollo fácil de utilizar, además que permite la utilización de las librerías de manera más cómoda, mediante la utilización de objetos.

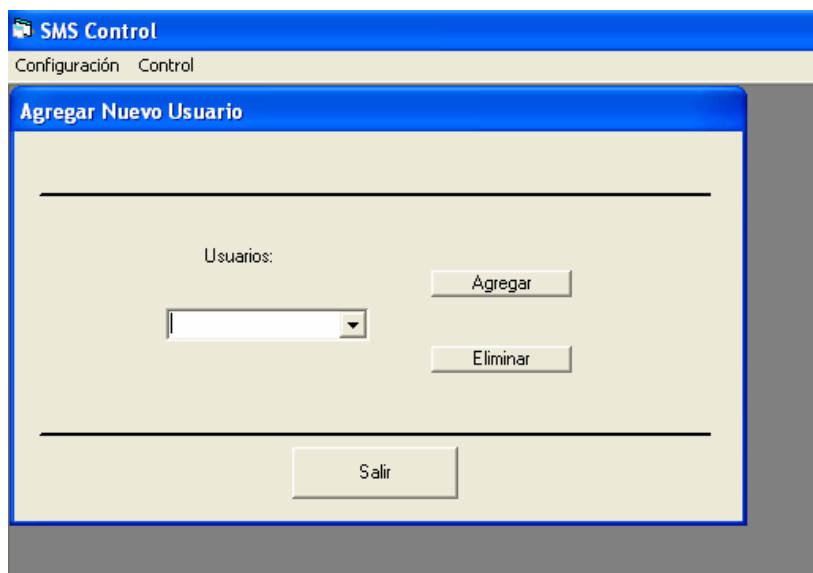
La aplicación fue organizada en formularios, los cuales dividen el trabajo en dos áreas bien diferenciadas: la configuración de los dispositivos a ser controlados y la construcción de la lista de los usuarios del sistema. La configuración de los dispositivos consiste en la inserción de los mismos en una lista junto con sus características: nombres de pila, código de casa en la que se encuentran y código del dispositivo. Esto se realiza en el formulario frmAgregarDispositivo. La lista de usuarios se implementa en el formulario frmAgregarUsuario.

5.1.1. El Formulario frmAgregarUsusario

Este formulario es el encargado de recibir a todos los usuarios del sistema, estos serán identificados utilizando el número de su teléfono celular, es decir, se deberá llenar una lista con todos

los números telefónicos que podrán hacer uso del sistema. En este formulario se valida que los números ingresados no sean nulos, es decir, necesariamente se debe ingresar un número telefónico, además de validarse el ingreso de números telefónicos que pertenezcan a una red móvil celular, es decir cuya numeración empiece con "09". Además el formulario permite la eliminación de usuarios ya ingresados.

Cuando el usuario remoto envíe un mensaje al sistema, haciendo uso de esta lista previamente llenada, se compara uno a uno con los elementos existentes para validar el número de donde proviene el mensaje.



The image shows a screenshot of a web application window titled "SMS Control". The window has a blue header bar with the text "SMS Control" and a menu bar with "Configuración" and "Control". Below the menu bar is a sub-header "Agregar Nuevo Usuario". The main content area is a light beige color and contains a form with the following elements: a label "Usuarios:" above a text input field with a dropdown arrow; a button labeled "Agregar" to the right of the input field; a button labeled "Eliminar" below the "Agregar" button; and a button labeled "Salir" at the bottom center of the form area.

Figura 5.1. Interfase del formulario frmAgregarUsuario

5.1.2. El Formulario frmAgregarDispositivo

Este formulario permite agregar los dispositivos a ser controlados. Estos se colocan en una lista junto con sus características: nombre de pila, código de casa y código de dispositivo, es decir su ID. El nombre de pila es el nombre que el usuario utilizará para hacer referencia al dispositivo cuando envíe un mensaje escrito hasta el celular local. Los parámetros de código de casa y código de unidad deben coincidir con los parámetros configurados físicamente en cada módulo X10 correspondiente a cada dispositivo a ser controlado. Se valida que no se ingrese un registro nulo, además del ingreso de nombres y códigos de casa y ID de dispositivos repetidos.

Agregar Dispositivo X10

1.- Escriba el nombre de pila del dispositivo, ejemplo: mi radio, lampara de dormitorio, aire acondicionado, etc.

Nombre:

2.- Seleccione el código de casa:

Código de casa:

3.- Seleccione el ID:

Número Identificador:

4.- Para agregar el nuevo record, presione Agregar record:

Dispositivos X10 existentes:

Agregar Record Eliminar Record

Salir

Figura 5.2. Interfase gráfica del formulario frmAgregarDispositivo

5.1.3. El Formulario frmConfigSMSControl

Este es el formulario medular de la aplicación, pues en él esta programada la ejecución de las tareas más importantes:

- Selección del puerto serial a ser utilizado por la interfase Firecracker.
- Activación total del sistema.
- Monitoreo de la recepción de los mensajes escritos enviados por el móvil remoto.
- Lectura de estos mensajes desde la memoria por defecto del móvil local.
- Validación del número telefónico desde el cual fue enviado el mensaje, comparando con los elementos de la lista de usuarios del formulario frmAgregarUsuario.
- Validación del comando recibido en el mensaje enviado por el móvil remoto, comparando con los elementos de la lista de dispositivos del formulario frmAgregarDispositivos.
- Utilización del protocolo CM17A para codificar la palabra de control correspondiente.
- Envío bit por bit del comando X10 según el protocolo CM17A.

Para la selección del puerto serial y su respectiva configuración, se utilizó el control "Microsoft Comm Control". Usualmente cuando se usa este control, es necesario configurar la tasa de baudios, paridad y número de bits de inicio y de parada, sin embargo como la interfase Firecracker no utiliza una conexión estándar, no es necesario configurar estos parámetros. Lo que si es necesario es mantener la línea DTR en alto, recordemos que esta señal brindará la tensión para alimentar a la interfase, además de la línea RTS.

Tabla 9. Configuración inicial del control MSComm

Operación	Descripción
MSComm1.CommPort = 1	Seleccionar el número de puerto
MSComm1.DTREnable = True	Configura la línea DTR en alto
MSComm1.RTSEnable = True	Configurar la línea RTS en alto
MSComm1.PortOpen = True	Abrir el puerto
Pause 50	Espera
Reset	Resetea la interfase (función particular)

La activación y el monitoreo del sistema se realizan a través del control Timer. Este control permite el la ejecución de cierto bloque de código luego de transcurrido una cantidad

determinada de tiempo. De esta manera, una vez activado este control, se revisará periódicamente el número de mensajes no leídos presentes en el teléfono.

La lectura del mensaje recibido se realiza utilizando el método ReadSMS, del cual se da una descripción más detallada en capítulos anteriores. El mensaje capturado es un objeto el cual tiene varias propiedades entre ellas tenemos: dirección de origen, prioridad con la que fue enviado el mensaje, el mensaje propiamente dicho. Obteniendo la dirección de origen, es decir, el número telefónico del móvil remoto, se procede a validar comparándolo con los elementos de la lista de usuario del formulario frmAgregarUsuario.

Como sabemos, el mensaje recibido contiene el comando que va a ser ejecutado, el cual está compuesto por el nombre de pila del dispositivo y la acción a realizarse: encender "1", o apagar "0". Para validarlo, se procede a comparar con la lista de dispositivos X10 existentes del formulario frmAgregarDispositivo. Luego se esto se procede a codificar la orden recibida haciendo uso del protocolo CM17A. La palabra de control X10 se forma utilizando el código de casa más el ID

del dispositivo, es decir, el código de dispositivo, junto con el código de función: encendido o apagado. Para esto se utiliza los códigos que fueron ingresados junto con el nombre de pila de los dispositivos en la lista del formulario frmAgregarDispositivo. Una vez codificada la orden, se procede al envío de la misma para el cual se hace uso de las funciones:

- Send_1
- Send_0
- Send_Byte
- Send_Header
- Send_Footer
- Reset
- Pause

Como habíamos visto, inicialmente ambas líneas, DTR y RTS se encontraban el alto, colocando a la interfase en estado de espera. Para enviar un uno era necesario que la señal DTR pasa a un nivel bajo y este nivel se mantenga por al menos 5 ms.

```
Public Sub Send_1( )           'Nombre
MSComm1.DTREnable = False 'Configura DTR en un nivel
                             bajo
```

```

Pause 1                                'Estado de espera
MSComm1.DTREnable = True 'Configura DTR en un nivel alto
Pause 1
End Sub

```

Para el envío de un cero lógico, primeramente se configura la señal RTS en un nivel bajo, manteniéndola en este estado por al menos 5 ms, y luego se la vuelve a configurar en alto. Esto se realiza acorde al protocolo que se utiliza para comunicar el computador con la interfase Firecracker, el cual se revisó anteriormente.

```

Public Sub Send_0( )                    'Nombre
MSComm1.RTSEnable = False 'Configura RTS en un nivel
                                bajo
Pause 1                                ' Estado de espera
MSComm1.RTSEnable = True  'Configura RTS en un nivel
                                alto
Pause 1
End Sub

```

Haciendo uso de estas dos funciones, la función Send_Byte se implementa de una manera muy sencilla:

```

Public Sub SendByte(iByte As Long)

Dim i As Integer

For i = 0 To 7           ' Lazo for para los 8 bits
If (iByte And &H80) = &H80 Then ' Si b8 es 1 entonces
Send_1                 ' Envía 1
Else                   ' Si b8 es 0 entonces
Send_0                 ' Envía 0
End If
iByte = iByte * 2     ' Desplazamiento a la
izquierda
Next i                 ' Siguiete bit
End Sub

```

Según el protocolo de comunicación entre la interfase y el computador, para iniciar la transmisión enviamos primeramente uno dos bytes de inicio: D5 y AA. Para Esto hacemos uso de la función Send_Header, la cual se implementa fácilmente utilizando la función Send_Byte.

```

Public Sub SendHeader()

SendByte &HD5           ' Envía D5

SendByte &HAA          ' Envía AA

```

```
End Sub
```

Análogamente, para finalizar la transmisión es necesario el envío del byte de finalización: AD. Esto se realiza con la función Send_Footer.

```
Public Sub SendFooter()
SendByte &HAD           ' Envía AD
End Sub
```

La función Reset, permite que la interfase vuelva a encontrarse en un estado previo para la realización de una nueva transmisión. Para esto, según el protocolo de comunicación, debemos poner tanto DTR como RTS en un nivel lógico bajo y mantenerlas en este estado al menos por 5 ms, luego de esto debemos volverlas a poner en alto.

```
Public Sub Reset()
MSComm1.RTSEnable = False   ' RTS y DTR
MSComm1.DTREnable = False   ' se configuran en bajo
Pause 50                     ' Pausa para el estado de
espera
```

```
MSComm1.RTSEnable = True      ' RTS y DTR
MSComm1.DTREnable = True      ' se configuran en bajo
Pause 50                        ' Pausa
End Sub
```

Como habremos podido observar, la función Pause es utilizada para brindar los 5ms necesarios para cada cambio de comandos especificados por el protocolo de comunicación. Esta función es implementada haciendo uso de la función sleep () del sistema.

```
Public Sub Pause(milli As Long)
Sleep (milli)
End Sub
```

La siguiente figura resume la lógica de programación del formulario frmConfigSMSControl, el cual es la base para el funcionamiento de nuestro sistema de control domiciliario basado en Firecracker.

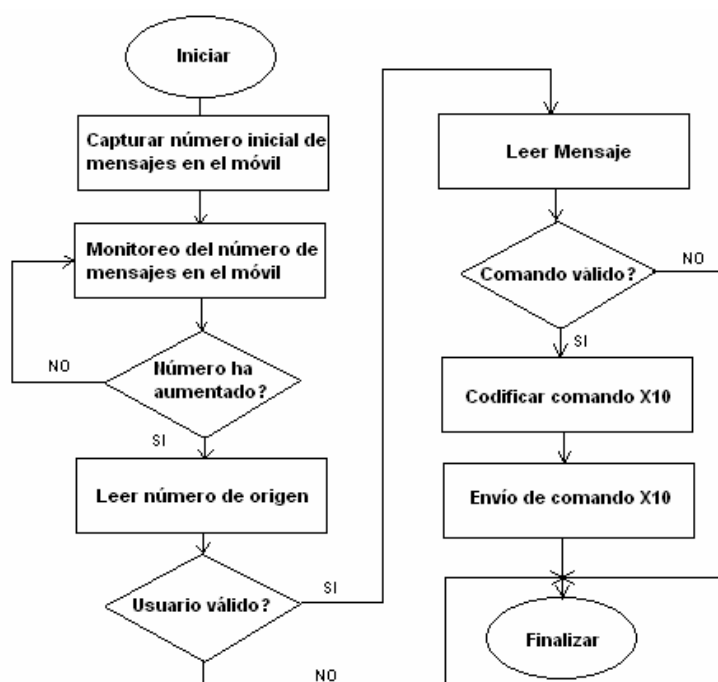


Figura 5.3. Lógica de control en el formulario
frmConfigSMSControl

La aplicación SMSControl basada en la interfase CM11A bidireccional

La aplicación SMSControl basada en la interfase CM11A bidireccional es una variación de la aplicación basada en Firecracker. Como hemos dicho, el tipo de interfase que se utilice define el protocolo que utilizará el sistema, es decir, si se utilizara CM17A ó CM11A según sea el caso de las interfases Firecracker ó CM11A respectivamente. Los formularios frmAgregarUsuario, y frmAgregarDispositivos se mantienen

iguales a los de la aplicación basada en Firecracker, sin embargo el formulario frmSMSConfigControl sufrió varias transformaciones para hacer que el sistema sea bidireccional.

Además se agregó un nuevo formulario frmConfigAlarmas para brindarle a la aplicación muchas más funcionalidades como las de enviar un reporte del estado de los dispositivos cada cierto intervalo de tiempo, realizar una llamada de emergencia en caso de la activación sospechosa de sensor eagle-eye, enviando un tono de emergencia hasta el teléfono remoto, y la programación de la activación y desactivación de los dispositivos X10 a una determinada hora ingresada por el usuario.

Variaciones en el formulario frmConfigSMSControl

La variación fundamental realizada en este formulario es la incorporación del protocolo CM11A en lugar del protocolo CM17A. Para esto se hizo uso del control Active X: CONTROLCM. Este control vino junto con los dispositivos X10 adquiridos, y permite el desarrollo de aplicaciones en Visual Basic de una manera rápida y fácil.

Características del control Active X: CONTROLCM

Este control contiene tres propiedades, cinco métodos y un evento. De todos estos, se utilizaron los métodos; Init, ExecWait, y los eventos; X10Event y X10SingleEvent.

- **El método Init.-** Este método se invoca luego de haber configurado los parámetros del puerto de comunicaciones, tales como, número de puerto, velocidad de transmisión, bits de datos, bits de parada, etc. Este método se sincroniza con la interfase para configurarle la fecha actual. Este método retorna "0" si el puerto de comunicaciones se ha configurado con éxito y la interfase se ha encontrado. Retorna "1" si el puerto de comunicaciones se ha configurado con éxito pero la interfase no se ha detectado. Retorna "OTHER" si ni siquiera se ha podido inicializar el puerto de comunicaciones. La sintaxis de este método es Init ().
- **El método ExecWait().-** Este método se invoca para enviar los comandos X10. Los comandos se colocan en una cola interna, puesto que tomará algunos segundos enviarlos por la red eléctrica. El control posee un buffer que permite almacenar hasta 50 comandos. Este método no retorna ningún valor. La sintaxis es:

Exec (housecode as string, devicecode as string, command as integer, [optional]brightness as integer, optional data1 as integer, optional data2 as integer).

Donde: housecode es el código de casa del dispositivo a ser controlado, los valores posibles son desde "A" hasta "P"; devicecode es el identificador del dispositivo a ser controlado, los valores pueden ser desde "1" hasta "16", command es el comando x10 a ser enviado, los valores posibles muestran en la tabla IX; brightness es un parámetro opcional que se utiliza en conjunto con los comandos dim y bright, es un valor entero que puede ir desde cero hasta cien el cual indica el nivel de intensidad al que se desea poner las lámparas o equivalentes, data1 y data 2 también son dos parámetros opcionales que se utilizan cuando se está enviando comandos extendidos X10, los cuales no son reconocidos por todos los dispositivos X10. Cabe señalar que este método no retornará hasta que el comando X10 se haya enviado exitosamente hasta la interfase CM11A ó hasta que el contador de intentos de envío haya llegado a su máximo valor de cuatro intentos, este valor puede ser configurado haciendo uso de la propiedad "SendRetryCount". Cada intento de envío toma

aproximadamente dos segundos. Este método retorna un “0” si se ha realizado con éxito ó “1” si no fue posible el envío del comando hasta la interfase

Tabla 10. Valores posibles del parámetro command perteneciente al método ExecWait ()

VALORES	DESCRIPCIÓN
0	ALL UNITS OFF
1	ALL LIGHTS ON
2	ON
3	OFF
4	DIM
5	BRIGHT
6	ALL LIGHTS OFF
7	EXTENDED CODE
8	HAIL REQUEST
9	HAIL ACK
10	PRESET DIM 1
11	PRESET DIM 2
12	X DATA XFER
13	STATUS ON
14	STATUS OFF
15	STATUS REQUEST
-1	Only send the address

- **El evento X10Event ()** .- Este evento es activado cada vez que un comando X10 es detectado, en la red eléctrica, por la interfase CM11A. La sintaxis es la siguiente:

X10Event(devices as string, housecode as string, command as integer, extra as string, data2 as string)

Donde: devices es una cadena de caracteres que indica si se han detectado más de un comando sobre la línea de alimentación eléctrica, cada dispositivo esta separado por una coma del siguiente dispositivo, ejemplo: "B1 B2 B3", cabe señalar que no anotarán diferentes códigos de casa en una misma cadena de caracteres; housecode es una cadena de caracteres conteniendo un solo código de casa; command es un entero indicando el comando X10 detectado según la tabla IX; extra es una cadena de caracteres que indica si algún parámetro de control de intensidad fue detectado, data 2 es una cadena de caracteres que contiene los comandos extendidos X10 que se hayan detectado.

- **El evento X10Single Event.**- Este evento es accionado cuando un dispositivo X10 es direccionado. Recordemos que antes de que se envíe el comando a ser ejecutado, el protocolo CM11A indica que se debe direccionar el dispositivo que va a recibir el comando X10. Este evento

detecta este tipo de señales sobre la red de alimentación.

La sintaxis es:

X10SingleEvent (devices as string, housecode as string, command as integer, extra as string, data2 as string)

Donde: *devices* es una cadena de caracteres como "A1"; *housecode* es el código de casa detectado; *command* contiene el valor de -1 indicando que solo se ha enviado la dirección del dispositivo; *extra* es una cadena de caracteres vacía al igual que *data2*.

Las funciones que se incorporan al formulario `frmConfigSMSControl`, a través del control `Active X CONTROLCM11`, son las siguientes:

- Selección del puerto serial a ser utilizado por la interfase CM11A, e inicialización del mismo, esto se hace utilizando el método `Init()` descrito anteriormente.
- Utilización del protocolo CM11A para codificar la palabra de control correspondiente y envío del comando hasta el dispositivo especificado. Esto se realiza utilizando el método `ExecWait()`.
- Registro de todos los comandos enviados durante el tiempo que el sistema haya permanecido activo, además de los eventos acaecidos durante este tiempo.

- Detección de los comandos X10 enviados por la interfase TM751 hasta los módulos X10 luego del disparo del sensor eagle-eye. Esto se logra a través de los eventos X10Event() y X10SingleEvent().
- Realización de llamada de emergencia y envío de tono de alarma (DTMF) hasta todos los teléfonos que el usuario haya ingresado, en caso de activación del sensor de movimientos. Para esto, se utilizó una librería adicional del SDK Beta 3.0 de Nokia: NokiaCLCall.
- Envío de mensajes escritos hasta el celular remoto con el reporte del estado de los dispositivos X10.

La librería NokiaCLCall

Esta librería contiene componentes para el manejo de las llamadas de voz en los teléfonos móviles de Nokia. Los componentes de estas librerías contienen interfases para crear, recibir y manejar llamadas de voz. Sin embargo no todos los teléfonos pueden hacer uso de estas librerías pues están restringidas solo para se usadas por los siguientes modelos de teléfonos: 3320, 3360, 6210, 6250, 6360, 6310, 7110, 7160, 7190, 8310. Los dos elementos fundamentales de esta librería son: CallAdapter y CallItem. El primer elemento permite la manipulación de eventos como: llamada entrante y envío de

tonos DTMF (Data Tone Multiple Frequencies). En cambio CallItem es el objeto que contendrá las propiedades de la llamada activa, por ejemplo: número de origen, número de destino, estado de la llamada, etc. Estas propiedades se agrupan en interfases llamadas: SubBlocks. Como estas propiedades serán utilizadas por teléfonos TDMA como GSM, entonces, se tiene una interfase SubBlock por cada sistema: ITDMASubBlocks e IGSMSubBlocks, además de una interfase adicional compuesto por propiedades comunes a los dos sistemas: IcommonSubBlocks.

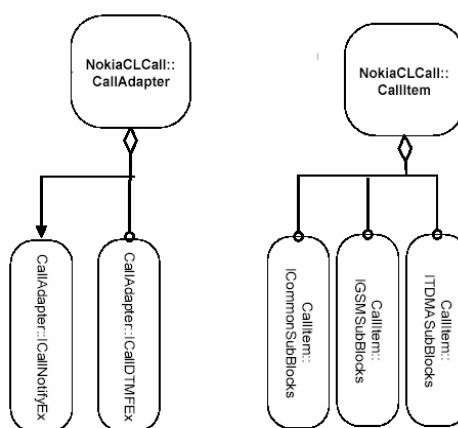


Figura 5.4. Elementos de la librería NokiaCICall

De todas las interfases de la librería, nuestro proyecto utilizó los métodos CreateCall y DTMFSend además de los eventos CallAnswered y CallReleased.

- **El método CreateCall ().**- Este método es miembro de la librería `NokiaCICall.CallAdapter.ICallBasicEx` y permite la creación de la llamada de voz. Como podemos ver este método pertenece al conjunto de métodos de la interfase para la realización de acciones básicas de llamadas de voz. La sintaxis es la siguiente: `CreateCall ([in] CallItem)`, donde `CallItem` es el objeto que contiene las propiedades de la llamada activa que va a ser creada mediante el método.
- **El método DTMFSend ().**- Este método es miembro de la librería `NokiaCLCall.CallAdapter.ICallDTMFEx` y permite acciona el envío de tonos DTMF sobre una llamada activa. Su sintaxis es:

`DTMFSend ([in] VARIANT_BOOL DTMF bDTMFDigit, [in] ICallItem, [out] unsigned char CallID)`

Donde: `bDTMFDigit` es un valor de entrada que debe ser configurado en "0" para permitir el envío de los dígitos DTMF. `ICallItem` es el objeto que tiene las propiedades de las llamada activa, este debe tener previamente la configuración de los dígitos DTMF que van a ser enviados. `CallID` es un valor de salida que indica el identificador de la llamada activa.

- **El Evento CallAnswered ()**.- Ocurre cuando la llamada ha sido atendida por la red de telefonía. Se realiza esta detección tanto para llamadas entrantes como salientes al teléfono local. Es miembro de la librería: NokiaCLCall.CallAdapter.ICallNotifyEx. La sintaxis es la siguiente: CallAnswered ([in] unsigned char *byCallID*); donde *byCallID* es el identificador de la llamada activa.
- **El Evento CallReleased ()**.- Ocurre cuando la llamada ha sido terminada tanto por el usuario local, el usuario remoto ó por la red de telefonía. Es miembro de la librería NokiaCLCal.CallAdapter.ICallNotifyEx. La sintaxis es: CallReleased ([in]unsigned char *byCallID*); donde *byCallID* es el identificador de la llamada activa.

Llamada de emergencia y envío de tonos DTMF

La funcionalidad de llamada de emergencia y envío de tonos DTMF, como ya habíamos dicho, hace uso de estos dos métodos y eventos. Básicamente, al detectarse el evento de disparo del sensor eagle-eye, se crea el objeto CallItem, se le agrega las propiedades necesarias, como número de origen, número de destino, dígitos DTMF a ser enviados; inmediatamente se hace uso del método CreateCall para crear la llamada. Si la llamada es atendida por la red de telefonía

móvil o fija, según sea el caso, entonces, se dispara el evento CallAnswered, originándose una llamada activa que desencadena el envío de los dígitos DTMF que anteriormente fueron especificados dentro de las propiedades del objeto CallItem. Ya sea que el usuario remoto, ó la red terminen la llamada, el evento CallReleased es disparado haciendo que se termine el envío de los dígitos DTMF.

5.1.4. El Formulario frmConfigAlarmas

Este formulario adicional solo está presente en la aplicación SMSControl, basada en la interfase CM11A. Este formulario permite que el usuario programe las tareas de encendido y apagado de dispositivos X10 además de accionar el envío de reporte de estado de los dispositivos y habilitar la realización de llamadas de emergencia y envío de los dígitos DTMF. Este formulario fue diseñado como un cuadro de dialogo compuesto por tres fichas:

- Configuración de día
- Notificación de estado
- Llamada de emergencia

La ficha Configuración de día.- Esta ficha le permite al usuario ingresar la hora a la que desee activar los dispositivos, así

como la hora a la que se desactivarán los mismos. El formato de hora es am / pm. Además el usuario puede seleccionar los dispositivos que se accionarán de una lista de dispositivos existentes que hayan sido previamente ingresados al configurar inicialmente el sistema.

La ficha Notificación de estado.- Esta ficha le permite al usuario accionar el envío de reportes del estado de los dispositivos X10 existentes. El usuario puede ingresar el intervalo de tiempo, en minutos, existente entre cada envío. Este reporte se enviará a todos los usuarios registrados del sistema.

La ficha Llamada de Alarma.- En esta ficha se le pregunta al usuario si desea que se realice una llamada en caso de activación del sensor eagle-eye enviando un tono de emergencia. Si el usuario acciona esta función, deberá ingresar todos los números telefónicos a los que se llamará en caso de emergencia, estos números pueden pertenecer tanto a la red móvil como a la red fija.

Básicamente el formulario posee un temporizador que hace que la aplicación esté verificando cada cierto intervalo de tiempo la

hora del sistema. Cuando la hora del sistema alcanza la hora ingresada por el usuario para el encendido de los dispositivos X10, entonces se ejecuta el envío de los comandos X10. Luego de esto se sigue verificando la hora del sistema y cuando esta alcanza la hora de desactivación previamente ingresada por el usuario, se realiza el envío de los comandos de apagado hasta todos los dispositivos que el usuario haya seleccionado previamente en la ficha Configuración de día. Además, mediante el temporizador se logra el envío de los reportes de estado de los dispositivos X10 cada vez que se agote el intervalo de tiempo, en minutos, ingresado por el usuario.

CAPÍTULO 6

6. PUESTA A PRUEBA DEL SISTEMA

Este capítulo lo consideramos de suma importancia, puesto que en él hacemos un resumen de las dificultades que se tuvo en el momento de diseño, implementación y finalmente en la posterior utilización del sistema, además de las conclusiones y las recomendaciones que se tomaron como resultados de dichas pruebas, haciendo de este capítulo una fuente inicial para la implementación de futuras mejoras para el sistema.

6.1. Dificultades en la implementación

Empezaremos esta sección indicando las dificultades que se ha experimentando durante la etapa de diseño del sistema.

Dificultades

Cuando surgió la idea de incorporar los mensajes escritos como un nuevo elemento de control para los sistemas de control domiciliarios

basados en computadora y el protocolo X10, surgieron los siguientes obstáculos:

- La inexistencia de un solo tipo de interfase y conectores para la comunicación física entre los teléfonos móviles y el computador.
- La inexistencia de un solo protocolo para la comunicación entre teléfonos móviles y el computador.

Se tuvo entonces que escoger la marca de teléfonos con que se iba a trabajar. Debíamos escoger una marca que nos facilite algún mecanismo para lograr la comunicación lógica entre el móvil y el celular. Nokia nos ofrecía todo eso a través un kit para desarrollo de aplicaciones basadas en la conectividad móvil PC, además del cable para conectividad física DLR-3P. Sin embargo no todos los modelos de teléfonos móviles Nokia eran compatibles con el kit, por lo que se tuvo que adquirir uno de los modelos que tenga compatibilidad tanto con el cable DLR-3P como con el kit. Luego durante la etapa de desarrollo surgieron otros obstáculos:

- No todos los métodos y funciones de las librerías funcionaban correctamente con el teléfono móvil Nokia 7160 adquirido.
- El método `SendSMS` y evento `MessageReceived`, de vital importancia para el sistema, no eran reconocidos por el teléfono

Nokia 7160, a pesar que en las especificaciones del kit se aseguraba compatibilidad con este modelo de teléfono.

- La imposibilidad de enviar la reproducción de un mensaje de voz previamente grabado en el computador, en el caso de la activación del sensor eagle-eye, a través de la llamada de emergencia activa, haciendo uso del cable DLR-3P.

Para solucionar el problema de la no funcionalidad del evento MessageReceived, hicimos uso del método GetMemoryStatus para detectar cuando el número de mensajes recibidos aumentaba. Esto se explica con más detalle en capítulo anterior. Sin embargo para lograr la funcionalidad del método SendSMS tuvimos que adquirir otro modelo de móvil: Nokia 6360, el cual tenía un sistema operativo superior al modelo 7160.

Al realizar la ficha de llamada de emergencia del formulario frmConfigAlarmas, buscábamos que el sistema sea capaz de realizar una llamada a ciertos números telefónicos previamente ingresados por el usuario en caso de la activación del sensor eagle-eye; además de la reproducción de un archivo de audio (.wav) previamente grabado en el computador y el envío del mismo a través de la llamada activa utilizando el mismo cable DLR-3P. Sin embargo el conector del cable DLR-3P no nos permitía el uso del pin para entrada de señales de

audio (audio in) puesto que cubría todos los pines a pesar de que realmente solo utiliza 3 hilos para la comunicación FBUS. Entonces se procedió al envío de tonos DTMF para indicar la emergencia en lugar del archivo de voz. Estos tonos son producidos por el teléfono.

6.2. Simulación de un módulo receptor X10

Ante de realizar a tomar cualquier medición se procedió a realizar la simulación de un módulo receptor típico X10. Como habíamos visto en el capítulo 3, este receptor consta de tres etapas:

- Fuente de poder
- Detector de cruce por cero
- Etapa extractora de señal

Donde, la etapa extractora de señal está compuesta a su vez por las siguientes etapas:

- Etapa de Filtrado
- Etapa Amplificadora de Ganancia Variable.
- Integrador
- Etapa Amplificadora Secundaria
- Etapa Comparadora.
- Microcontrolador

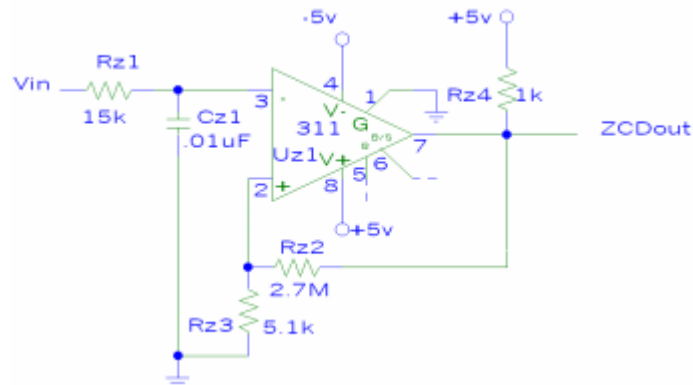


Figura 6.1. Detector de cruce por cero

Esta etapa utiliza un Schmitt Trigger con la finalidad de minimizar el ruido, este es esencialmente un comparador con histéresis; con un voltaje de umbral de 0.01 voltios para el ascenso de la señal de entrada y 0 voltios para el descenso de la señal de entrada. El umbral de subida es muy pequeño para lograr reaccionar rápidamente al cruce por cero de la señal de 60 Hz. Un filtro pasa bajas precede al detector de cruce por cero. Su frecuencia de corte es de 1 kHz para no permitir el ingreso del ruido de frecuencias mayores presente en la línea, además este filtro provee de la atenuación necesaria para la señal de 12 voltios pico a pico proveniente de la salida del transformador, para no sobrecargar el amplificador operacional el cual está siendo alimentado por +5V/-5V. La salida de esta etapa es enviada directamente hasta el microcontrolador.

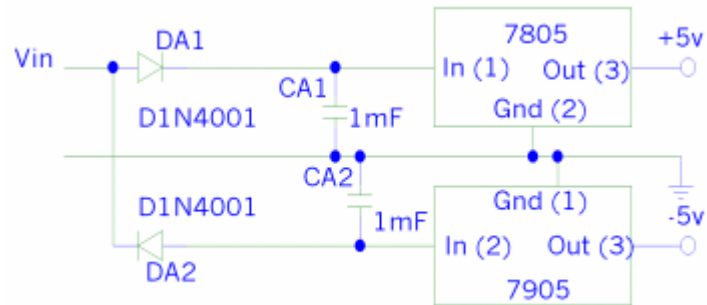


Figura 6.2. Fuente de poder

La fuente de poder provee del voltaje necesario a todos los amplificadores operacionales que se utilizan en el receptor. La señal V_{in} proviene del transformador y las señal $ZDOut$ se inyecta en el microcontrolador, el cual lo utiliza para sincronizarse con la señal que se obtiene de la etapa extractora de señal.

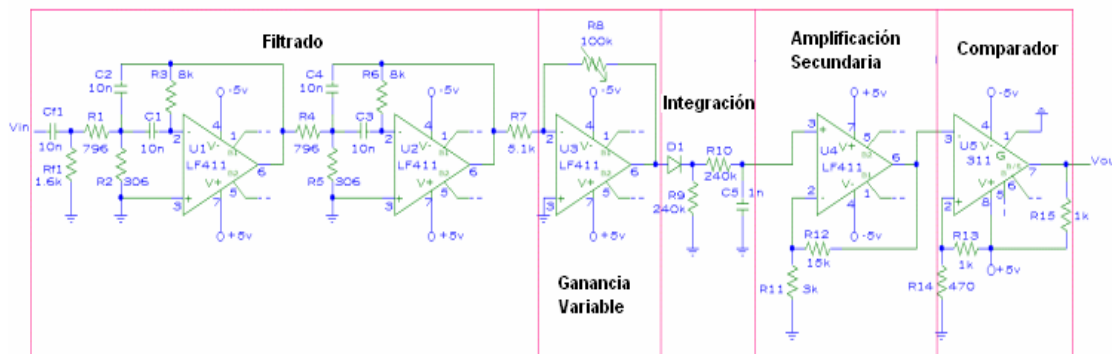


Figura 6.3. Etapa extractora de señal

La primera parte de la etapa extractora de señal, consiste en un bloque de filtrado; un sencillo filtro RC pasa altos, el cual previene que la señal de 12 voltios pico a pico llegue hasta los amplificadores

operacionales, los cuales solo están alimentados con +5V/-5V. Este filtro está diseñado para permitir el paso de señales con frecuencias superiores a los 100 kHz. Luego tenemos dos filtros pasabanda activos con una frecuencia central de 120 kHz con un factor Q igual a 3, lo cual implica una banda de paso entre los 100 kHz y 140 kHz, además, estos proveen una ganancia aproximadamente de 5. Se están utilizando dos filtros pasa banda para lograr una buena atenuación del ruido de 60 Hz producido por el transformador. La etapa de ganancia variable está constituida por un amplificador operacional en una configuración de retroalimentación negativa con un potenciómetro de $100\text{k}\Omega$ y una resistencia en la entrada de $5\text{k}\Omega$, obteniéndose una ganancia máxima de 20. En seguida, tenemos un diodo, el cual rectifica la señal, luego esta señal en una red RC integradora. Esta red RC está diseñada para que el capacitor logre descargarse antes de la llegada de otra ráfaga de señal de 120 kHz. La segunda etapa amplificadora aumenta la señal recibida por la red RC utilizando un amplificador no inversor con una ganancia de 6. Finalmente tenemos la etapa comparadora con un nivel de comparación de 1.6 voltios. La señal obtenida del comparador tiene forma cuadrada y es enviada al microcontrolador. El diseño de todo el módulo receptor es propiedad de la marca X10. Simularemos este diseño en PSPICE y compararemos las señales obtenidas con las

mediciones con la finalidad de entender como funciona la recepción X10. La señal esperada a la entrada de la etapa extractora de señal tiene la siguiente forma:

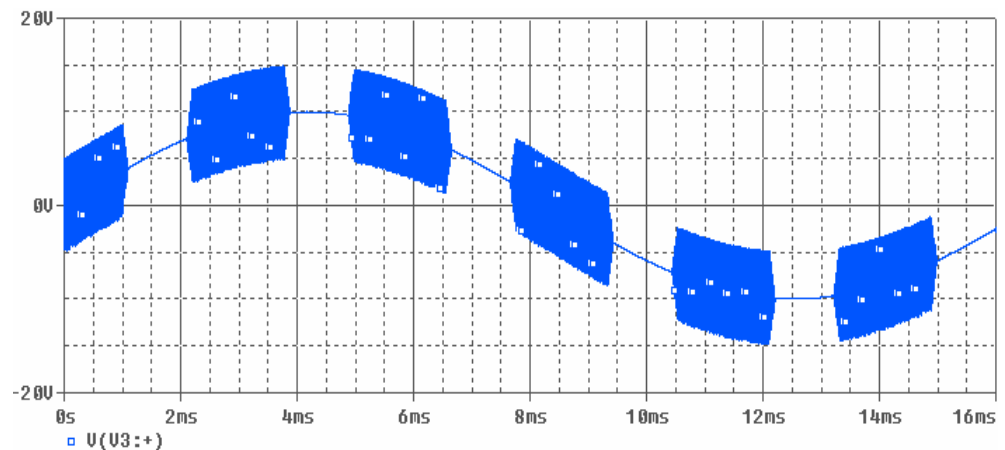


Figura 6.4. Señal de entrada a la etapa extractora

Como podemos ver, la señal de entrada no es más que la suma de la señal de 60 Hz con la señal X10, la cual a su vez, es la multiplicación de una señal cuadrada con un periodo seis veces menor al de la señal de 60 Hz con una señal de 5 voltios pico a pico y una frecuencia de 120 kHz. La señal cuadrada tiene una amplitud de un voltio y su ancho de pulso es de tan solo la tercera parte de su periodo, es decir, se mantiene tan solo por aproximadamente por 1 ms en alto y 2 ms en bajo. Esta señal es la que ingresa al filtro pasa alto, y luego a los dos filtros activos los cuales la atenúan en una proporción importante la señal por lo que es necesaria la etapa de amplificación variable. La siguiente figura muestra la señal a la salida del filtro activo, como

podemos observar, la señal de 120 kHz que inicialmente era de 5 voltios pico a pico, ahora tan solo es de 0.8 voltios pico a pico.

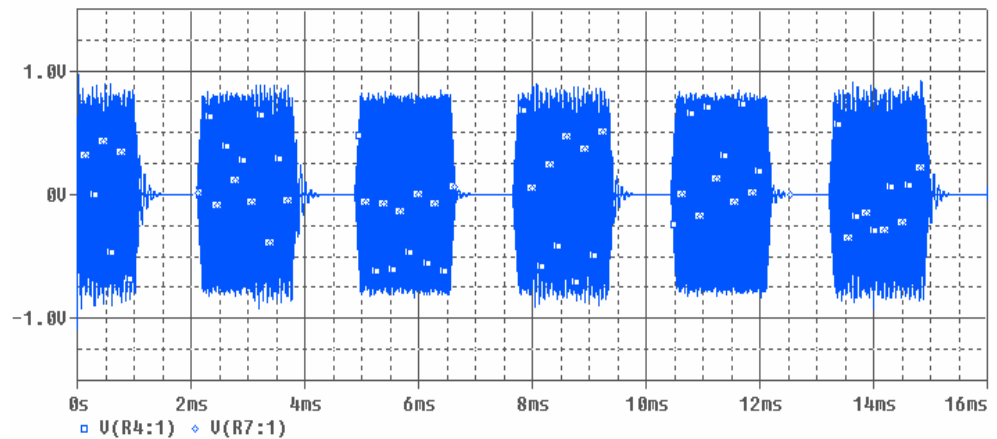


Figura 6.5. Señal de salida de los filtros activos

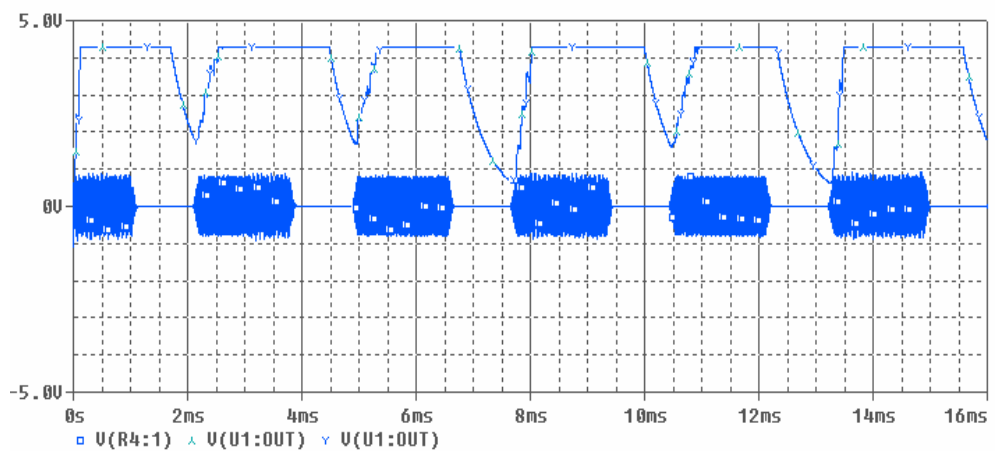


Figura 6.6. Señal de salida del amplificador secundario

La figura 32 muestra la señal de salida del amplificador secundario, ubicado luego de la etapa integradora, podemos ver una señal aproximadamente cuadrada. Además podemos observar que

efectivamente el capacitor de la red RC integradora, logra descargarse justo antes de la llegada de una nueva ráfaga de la señal de 120 kHz.

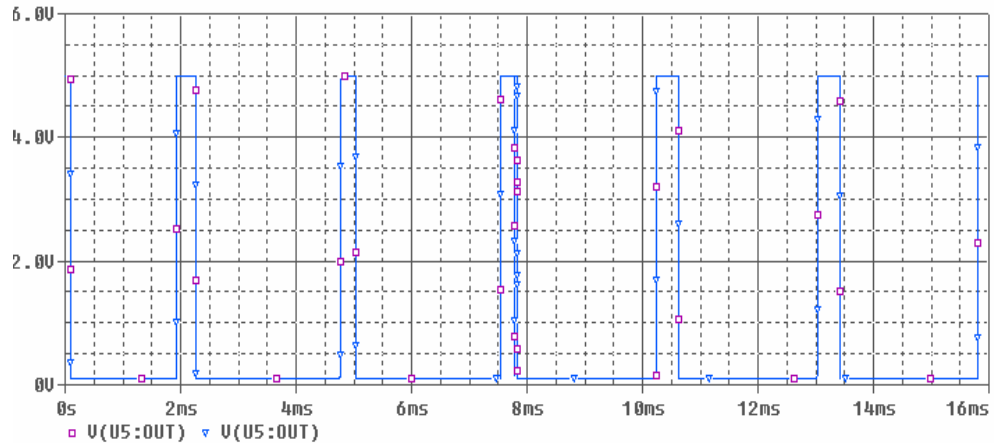


Figura 6.7 Señal de salida de la etapa comparadora

Esta figura nos muestra los pulsos obtenidos luego de la etapa comparadora, sin embargo estos pulsos están desfasados con respecto a las ráfagas de la señal de 120 kHz, es decir, estos pulsos están presentes cuando la ráfaga no lo está, por este motivo, la programación en el microcontrolador debe considerar la lógica negativa de estos pulsos. Esto se hace para evitar cualquier falta sincronización que pueda ser producida por el retardo inherente en cada elemento del circuito extractor de señal.

6.3. Problemas con X10

A pesar de sus múltiples beneficios, el envío de datos a través de la alimentación eléctrica no puede ser tan sencillo como parece, a simple

vista se pueden plantear preguntas como: ¿Qué ocurre si dos tramas son enviadas al mismo tiempo? ¿Qué ocurre si el receptor está muy alejado del emisor de comandos?

Colisiones.- Estas ocurren cuando el transceiver intenta enviar y recibir comandos X10 al mismo tiempo. Como sabemos la línea de alimentación eléctrica solo permitirá el viaje de una señal a la vez, es decir, o bien transmitimos un comando X10 o bien recibimos el mismo, pero no podemos hacer las dos cosas simultáneamente. Cuando el transceiver está enviando un comando y simultáneamente por la línea de alimentación eléctrica ya está viajando otra señal, ocurrirá una colisión. Esto ocurre generalmente en un ambiente donde se tengan varios transceivers. Además el chip TW523 solo puede almacenar una sola transmisión completa en su buffer, entonces, si la línea AC se satura de señales X10, el chip TW523 no podrá procesarlas y las descartará.

Problemas con sistemas vecinos.- Como las señales X10 usan las líneas de alimentación eléctrica, estas pueden salir del sistema local e ingresar a sistemas implementados en un domicilio contiguo y viceversa. Esto producirá que los dispositivos controlados por nuestro sistema comiencen a actuar de forma errática. Los diseñadores del sistema X10, anticipando que varios domicilios contiguos estarían

usando el sistema, crearon el concepto de código de casa, esto permite que diferentes domicilios tengan su propio código de casa el cual es diferente al de las demás y evita que señales de un domicilio interfieran con las de otro. Por ejemplo si todos los módulos de un domicilio están utilizando el código de casa A, estos no interferirán con un sistema contiguo que esté utilizando el código de casa B. Como sabemos, existen 16 códigos de casa disponibles y cada uno puede utilizarse con 16 dispositivos X10. Cabe señalar que esto no evita que las señales X10 puedan entrar o salir de un sistema local.

Existen maneras de evitar que las señales X10 salgan de su domicilio. La más popular es la utilización de filtros acoplados que atenuaran severamente las señales X10 que intenten ingresar o salir de un sistema local. Este filtro atenuador es colocado en la línea neutra del sistema de alimentación, censando la presencia de señales X10 y generando un voltaje de offset para neutralizarlas. Simultáneamente actúa como un acoplador pasivo el cual permite que las señales X10 viajen entre las fases del sistema de alimentación eléctrica. Cualquier señal que logres salir o entrar al sistema local, tendrá un amplitud muy pequeña lo que no le permitirá interferir dentro del sistema local. Si el domicilio contiguo también está utilizando este tipo de filtro, el problema de interferencia de señales está solucionado. Comercialmente existen varios fabricantes de este tipo de filtros.

Sin embargo, ¿qué podemos hacer acerca de la interferencia de señales de un sistema contiguo RF con nuestro sistema local si también está basado en RF? Esto es un serio problema ya que algún persona extraña que configure su control de mando remoto con nuestro código de casa y que esté próximo al sistema local, es decir a menos de 100 pies, podrá enviar señales X10 y controlar el funcionamiento de los dispositivos que estén en nuestro sistema. Para evitar este tipo de inconvenientes es necesario que el controlador del sistema sea un computador el cual podrá detectar cualquier señal de control que no sea enviada por el mismo y enviará mensajes de alerta a los usuarios indicando la presencia de señales intrusas.

El problema de la atenuación de las señales X10.- El ruido eléctrico es cualquier señal que se encuentre en el sistema de alimentación eléctrica que no se una señal de 120 voltios a 60 Hz. Cuando un transmisor X10 envía una señal a un módulo receptor, la señal resultante es la superposición de la señal de alimentación eléctrica y la señal X10, técnicamente hablando, la señal X10 puede ser considerada como ruido eléctrico, sin embargo sabemos que cumple un propósito específico y no es de tipo aleatoria sino más bien determinística. Nuestra señal X10 estará en problemas cuando se encuentre con otras señales que tengan su misma frecuencia pero que tenga amplitudes mayores a ella, esto producirá que el módulo

receptor no pueda discriminar la señal X10 y no pueda ejecutar el comando que le fue enviado. Las fuentes más comunes de señales que interfieren con las señales X10 son:

- Intercomunicadores inalámbricos que funcionan conectándose a las tomas de alimentación eléctrica.
- Luces fluorescentes.
- Los mismos transmisores X10 averiados.
- Fuentes de poder de televisores.
- Motores de dispositivos como refrigeradores, licuadoras batidoras, etc.

Los dispositivos descritos arriba emiten este tipo de ruido. Es muy difícil detectar la fuente de este tipo de ruido a no ser que se empiece probando uno por uno a los dispositivos de la lista, es decir, dejamos funcionando solo uno de ellos teniendo apagado los demás y vemos la respuesta del sistema, y así sucesivamente hasta detectar cual de los dispositivos es el que hace que nuestro sistema trabaje de forma errática. Haciendo uso de un osciloscopio pudo observar como la señal X10 es afectada al ser conectada en la misma regleta donde está conectado un televisor. Como se indica anteriormente, la fuente de poder de televisores de modelos antiguos toman la señal de alimentación eléctrica tal cual la reciben y no realizan ningún filtrado

de entrada como para discriminar tan solo señales alternas a una frecuencia de 60 Hz, al proceder de esta manera, si un transmisor X10 está conectado en una misma regleta o en una toma de corriente muy cercana a la que este conectada un televisor de estas características, la señal que envíe será absorbido por dicha fuente de poder y no alcanzará a llegar a los dispositivos receptores. Lo mismo ocurre si tenemos un receptor conectado cerca de donde está conectado este tipo de televisores.

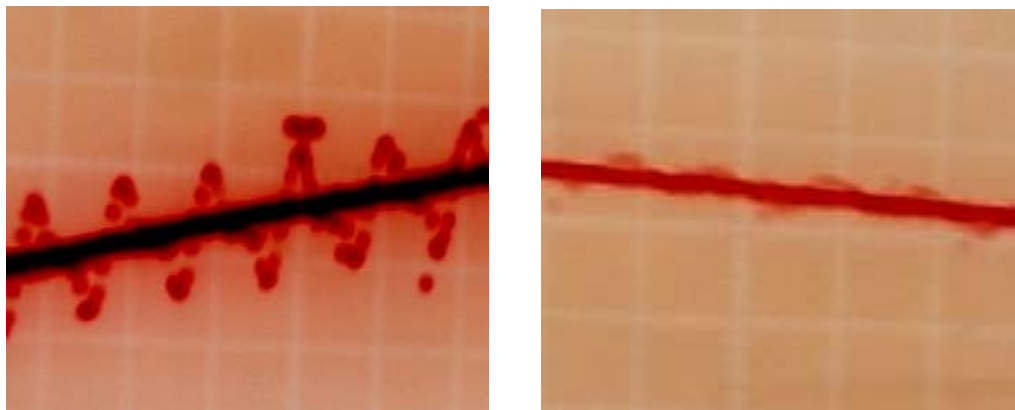


Figura 6.8. Atenuación de la señal X10

En el lado derecho de la figura 6.8 se muestra la señal X10 recibida por un receptor el cual está conectado a una de las tomas de una regleta; en el lado izquierdo, la señal X10 cuando se ha conectado un televisor a esta misma regleta. Se puede observar que la señal X10 se atenúa, disminuyéndose su amplitud en casi un setenta por ciento. Inicialmente la señal tenía una amplitud aproximada de 4 voltios y

luego de ser afectada, tan solo tienen una amplitud de 0.8 voltios. Esto produce que los comandos X10 no sean interpretados correctamente y los dispositivos controlados por los mismos, actúen de forma errática. No es el alcance de este capítulo el diseñar soluciones para este problema, sino más bien presentar todas las posibles dificultades por las que puede atravesar un sistema basado en X10, y comprobándolas de manera práctica.

CONCLUSIONES Y RECOMENDACIONES

Luego de implementar y haber puesto a prueba este sistema, tenemos suficientes fundamentos para poder sacar nuestras propias conclusiones sobre las fortalezas y debilidades de un sistema de control basado X10 de manera general, y de manera particular en nuestro sistema de control domiciliario basado en el mismo protocolo, el cual incorpora los mensajes escritos SMS como otro elemento de control.

De manera general podemos concluir que los sistemas X10 gozan de las siguientes fortalezas:

1. No requieren de ningún cableado adicional para implementar una red de dispositivos.
2. Dependiendo del tipo de controlador del sistema, los sistemas X10 no son complejos de implementar ya que no requieren que el operador tenga grandes conocimientos de electrónica. Son dispositivos del tipo plug-and-play que permiten la instalación de un sistema de control de forma casi inmediata.

3. Los dispositivos X10 tanto transmisores como receptores y tranceivers son completamente comerciales y fáciles de adquirir en tiendas de dispositivos electrónicos electrónica. El protocolo X10 es un protocolo de comunicación maduro, que ya ha sido implementado desde hace varios años de manera comercial
4. Al tener palabras de control sencillas, los dispositivos X10 funcionan de manera correcta a no ser que existan fuentes de ruido en algún punto de la red de alimentación en el domicilio en que se instala.
5. X10 permite la fácil interacción con un computador a través de sus interfaces tanto bidireccionales como unidireccionales, lo que permite inclusive poner en línea el control de dispositivos hogareños a través de Internet.
6. La facilidad de la modificación de los módulos X10 al saber todo su protocolo, además de la facilidad para desarrollar rutinas de control en un computador, lo hacen atractivo y flexible para desarrolladores de soluciones de control, como es el caso de este proyecto.
7. Con lo que respecta a sus debilidades de manera general podemos concluir que:

8. X10 no es escalable y no fue diseñado como un sistema de gran envergadura, es decir que permita el envío de órdenes más complejos e inclusive la transmisión de datos entre dispositivos finales, es decir, X10 no es implementado en plantas industriales y en ambientes hostiles, donde factores generadores de ruido eléctrico están presentes. X10 no trata de ser competencia para soluciones basadas en PLC, las cuales son más complejas y robustas.

9. Los dispositivos X0 como habíamos analizado son propenso al comportamiento erróneo cuando la señal X10 es transmitido en ambientes que tienen la presencia de ruido eléctrico o si los transmisores están junto a la toma de algún dispositivo cuya fuente de poder no filtre señales en el orden de 120 Khz como ocurre con algunos televisores de modelos anteriores. Este comportamiento erróneo también ocurre con el ingreso de señales X10 generadas en otro sistema contiguo.

De manera particular, podemos concluir que nuestro sistema X10 el cual utiliza SMS como elemento adicional de control permitió el desarrollo de una aplicación en Visual Basic sin muchos inconvenientes y la facilidad de la programación por módulos, nos permitió la utilización de la herramienta SDK Beta 3.0 de Nokia para realizar la convergencia del sistema X10 con SMS.

Como indicamos arriba, esta flexibilidad para el desarrollo de soluciones, es lo que hace que X10 sea extremadamente atractivo. Sin embargo podemos acotar que una de las principales debilidades de este sistema X10 es que el teléfono móvil local solo puede ser Nokia e inclusive de un pequeño grupo de modelos. Como ya explicamos anteriormente, esto se debió a que el SDK de Nokia resultó no ser tan flexible, además la interfase entre el teléfono y el computador es un cable de datos también propietario de Nokia, y el protocolo de comunicación entre el móvil y la PC no es abierto, lo cual limita el desarrollo de aplicaciones basadas en SMS. Sin embargo, la tendencia actual es el soporte WAP en los móviles, lo cual permite que nuestro sistema domiciliario sea puesto en línea con gran facilidad y accesado remotamente a través de celulares de cualquier marca que tengan soporte WAP.

Cabe señalar que el desarrollo de este prototipo permitió la interacción de estas dos tecnologías X10 y SMS aunque no al 100 por ciento debido a la falta de flexibilidad del SDK de Nokia como dijimos anteriormente, sin embargo permitió mostrar de manera sencilla como lograr esta interacción y aprovecharla para llevar el control domiciliario más allá de los límites de una casa y de un control remoto de un televisor.

Después de ver las dificultades que se enfrentaron en este proyecto, se recomienda:

1. El desarrollo de nuevos elementos de control para sistema basados en X10 se realicen haciendo uso de WAP y celulares que soporten esa tecnología. Cada vez es mayor el uso que se le puede dar a la navegación en los móviles, y podríamos decir que en un corto tiempo los mensajes escritos solo quedarán relegados a tareas sencillas como notificaciones de algunos eventos, pero un control bidireccional más estable solo se logrará haciendo uso de TCP/IP a través de WAP, de tal manera que el computador que es el controlador del sistema X10 pueda estar en línea mediante la asignación de una dirección pública de su ISP, y pueda ser accesado en tiempo real y de manera bidireccional por el usuario remoto para realizar tareas cada vez más complejas como la descarga del record de eventos acontecidos durante la ausencia en el domicilio, el record de alarmas accionadas y si vamos más lejos, sabemos que esto puede ser utilizado a nivel de operaciones de una empresa, permitiendo las consultas realizadas por un usuario remoto a la base de datos de su empresa, subiendo información a esa base de datos, etc. Podemos entonces concluir indicando que SMS, es útil, ciertamente lo es, pero su campo de acción como vimos está limitado.

ANEXO A

TABLA DE DATOS DEL PROTOCOLO CM17A

A1 ON	01100000	00000000	I1 ON	11100000	00000000
A1 OFF	01100000	00100000	I1 OFF	11100000	00100000
A2 ON	01100000	00010000	I2 ON	11100000	00010000
A2 OFF	01100000	00110000	I2 OFF	11100000	00110000
A3 ON	01100000	00001000	I3 ON	11100000	00001000
A3 OFF	01100000	00101000	I3 OFF	11100000	00101000
A4 ON	01100000	00011000	I4 ON	11100000	00011000
A4 OFF	01100000	00111000	I4 OFF	11100000	00111000
A5 ON	01100000	01000000	I5 ON	11100000	01000000
A5 OFF	01100000	01100000	I5 OFF	11100000	01100000
A6 ON	01100000	01010000	I6 ON	11100000	01010000
A6 OFF	01100000	01110000	I6 OFF	11100000	01110000
A7 ON	01100000	01001000	I7 ON	11100000	01001000
A7 OFF	01100000	01101000	I7 OFF	11100000	01101000
A8 ON	01100000	01011000	I8 ON	11100000	01011000
A8 OFF	01100000	01111000	I8 OFF	11100000	01111000
A9 ON	01100100	00000000	I9 ON	11100100	00000000
A9 OFF	01100100	00100000	I9 OFF	11100100	00100000
A10 ON	01100100	00010000	I10 ON	11100100	00010000
A10 OFF	01100100	00110000	I10 OFF	11100100	00110000
A11 ON	01100100	00001000	I11 ON	11100100	00001000
A11 OFF	01100100	00101000	I11 OFF	11100100	00101000
A12 ON	01100100	00011000	I12 ON	11100100	00011000
A12 OFF	01100100	00111000	I12 OFF	11100100	00111000
A13 ON	01100100	01000000	I13 ON	11100100	01000000
A13 OFF	01100100	01100000	I13 OFF	11100100	01100000
A14 ON	01100100	01010000	I14 ON	11100100	01010000
A14 OFF	01100100	01110000	I14 OFF	11100100	01110000
A15 ON	01100100	01001000	I15 ON	11100100	01001000
A15 OFF	01100100	01101000	I15 OFF	11100100	01101000
A16 ON	01100100	01011000	I16 ON	11100100	01011000
A16 OFF	01100100	01111000	I16 OFF	11100100	01111000
A BRIGHT	01100000	10001000	I BRIGHT	11100000	10001000
A DIM	01100000	10011000	I DIM	11100000	10011000

B1 ON	01110000 00000000	J1 ON	11110000 00000000
B1 OFF	01110000 00100000	J1 OFF	11110000 00100000
B2 ON	01110000 00010000	J2 ON	11110000 00010000
B2 OFF	01110000 00110000	J2 OFF	11110000 00110000
B3 ON	01110000 00001000	J3 ON	11110000 00001000
B3 OFF	01110000 00101000	J3 OFF	11110000 00101000
B4 ON	01110000 00011000	J4 ON	11110000 00011000
B4 OFF	01110000 00111000	J4 OFF	11110000 00111000
B5 ON	01110000 01000000	J5 ON	11110000 01000000
B5 OFF	01110000 01100000	J5 OFF	11110000 01100000
B6 ON	01110000 01010000	J6 ON	11110000 01010000
B6 OFF	01110000 01110000	J6 OFF	11110000 01110000
B7 ON	01110000 01001000	J7 ON	11110000 01001000
B7 OFF	01110000 01101000	J7 OFF	11110000 01101000
B8 ON	01110000 01011000	J8 ON	11110000 01011000
B8 OFF	01110000 01111000	J8 OFF	11110000 01111000
B9 ON	01110100 00000000	J9 ON	11110100 00000000
B9 OFF	01110100 00100000	J9 OFF	11110100 00100000
B10 ON	01110100 00010000	J10 ON	11110100 00010000
B10 OFF	01110100 00110000	J10 OFF	11110100 00110000
B11 ON	01110100 00001000	J11 ON	11110100 00001000
B11 OFF	01110100 00101000	J11 OFF	11110100 00101000
B12 ON	01110100 00011000	J12 ON	11110100 00011000
B12 OFF	01110100 00111000	J12 OFF	11110100 00111000
B13 ON	01110100 01000000	J13 ON	11110100 01000000
B13 OFF	01110100 01100000	J13 OFF	11110100 01100000
B14 ON	01110100 01010000	J14 ON	11110100 01010000
B14 OFF	01110100 01110000	J14 OFF	11110100 01110000
B15 ON	01110100 01001000	J15 ON	11110100 01001000
B15 OFF	01110100 01101000	J15 OFF	11110100 01101000
B16 ON	01110100 01011000	J16 ON	11110100 01011000
B16 OFF	01110100 01111000	J16 OFF	11110100 01111000
B BRIGHT	01110000 10001000	J BRIGHT	11110000 10001000
B DIM	01110000 10011000	J DIM	11110000 10011000

C1 ON	01000000	00000000	K1 ON	11000000	00000000
C1 OFF	01000000	00100000	K1 OFF	11000000	00100000
C2 ON	01000000	00010000	K2 ON	11000000	00010000
C2 OFF	01000000	00110000	K2 OFF	11000000	00110000
C3 ON	01000000	00001000	K3 ON	11000000	00001000
C3 OFF	01000000	00101000	K3 OFF	11000000	00101000
C4 ON	01000000	00011000	K4 ON	11000000	00011000
C4 OFF	01000000	00111000	K4 OFF	11000000	00111000
C5 ON	01000000	01000000	K5 ON	11000000	01000000
C5 OFF	01000000	01100000	K5 OFF	11000000	01100000
C6 ON	01000000	01010000	K6 ON	11000000	01010000
C6 OFF	01000000	01110000	K6 OFF	11000000	01110000
C7 ON	01000000	01001000	K7 ON	11000000	01001000
C7 OFF	01000000	01101000	K7 OFF	11000000	01101000
C8 ON	01000000	01011000	K8 ON	11000000	01011000
C8 OFF	01000000	01111000	K8 OFF	11000000	01111000
C9 ON	01000100	00000000	K9 ON	11000100	00000000
C9 OFF	01000100	00100000	K9 OFF	11000100	00100000
C10 ON	01000100	00010000	K10 ON	11000100	00010000
C10 OFF	01000100	00110000	K10 OFF	11000100	00110000
C11 ON	01000100	00001000	K11 ON	11000100	00001000
C11 OFF	01000100	00101000	K11 OFF	11000100	00101000
C12 ON	01000100	00011000	K12 ON	11000100	00011000
C12 OFF	01000100	00111000	K12 OFF	11000100	00111000
C13 ON	01000100	01000000	K13 ON	11000100	01000000
C13 OFF	01000100	01100000	K13 OFF	11000100	01100000
C14 ON	01000100	01010000	K14 ON	11000100	01010000
C14 OFF	01000100	01110000	K14 OFF	11000100	01110000
C15 ON	01000100	01001000	K15 ON	11000100	01001000
C15 OFF	01000100	01101000	K15 OFF	11000100	01101000
C16 ON	01000100	01011000	K16 ON	11000100	01011000
C16 OFF	01000100	01111000	K16 OFF	11000100	01111000
C BRIGHT	01000000	10001000	K BRIGHT	11000000	10001000
C DIM	01000000	10011000	K DIM	11000000	10011000

D1 ON	01010000	00000000	L1 ON	11010000	00000000
D1 OFF	01010000	00100000	L1 OFF	11010000	00100000
D2 ON	01010000	00010000	L2 ON	11010000	00010000
D2 OFF	01010000	00110000	L2 OFF	11010000	00110000
D3 ON	01010000	00001000	L3 ON	11010000	00001000
D3 OFF	01010000	00101000	L3 OFF	11010000	00101000
D4 ON	01010000	00011000	L4 ON	11010000	00011000
D4 OFF	01010000	00111000	L4 OFF	11010000	00111000
D5 ON	01010000	01000000	L5 ON	11010000	01000000
D5 OFF	01010000	01100000	L5 OFF	11010000	01100000
D6 ON	01010000	01010000	L6 ON	11010000	01010000
D6 OFF	01010000	01110000	L6 OFF	11010000	01110000
D7 ON	01010000	01001000	L7 ON	11010000	01001000
D7 OFF	01010000	01101000	L7 OFF	11010000	01101000
D8 ON	01010000	01011000	L8 ON	11010000	01011000
D8 OFF	01010000	01111000	L8 OFF	11010000	01111000
D9 ON	01010100	00000000	L9 ON	11010100	00000000
D9 OFF	01010100	00100000	L9 OFF	11010100	00100000
D10 ON	01010100	00010000	L10 ON	11010100	00010000
D10 OFF	01010100	00110000	L10 OFF	11010100	00110000
D11 ON	01010100	00001000	L11 ON	11010100	00001000
D11 OFF	01010100	00101000	L11 OFF	11010100	00101000
D12 ON	01010100	00011000	L12 ON	11010100	00011000
D12 OFF	01010100	00111000	L12 OFF	11010100	00111000
D13 ON	01010100	01000000	L13 ON	11010100	01000000
D13 OFF	01010100	01100000	L13 OFF	11010100	01100000
D14 ON	01010100	01010000	L14 ON	11010100	01010000
D14 OFF	01010100	01110000	L14 OFF	11010100	01110000
D15 ON	01010100	01001000	L15 ON	11010100	01001000
D15 OFF	01010100	01101000	L15 OFF	11010100	01101000
D16 ON	01010100	01011000	L16 ON	11010100	01011000
D16 OFF	01010100	01111000	L16 OFF	11010100	01111000
D BRIGHT	01010000	10001000	L BRIGHT	11010000	10001000
D DIM	01010000	10011000	L DIM	11010000	10011000

E1 ON	10000000	00000000	M1 ON	00000000	00000000
E1 OFF	10000000	00100000	M1 OFF	00000000	00100000
E2 ON	10000000	00010000	M2 ON	00000000	00010000
E2 OFF	10000000	00110000	M2 OFF	00000000	00110000
E3 ON	10000000	00001000	M3 ON	00000000	00001000
E3 OFF	10000000	00101000	M3 OFF	00000000	00101000
E4 ON	10000000	00011000	M4 ON	00000000	00011000
E4 OFF	10000000	00111000	M4 OFF	00000000	00111000
E5 ON	10000000	01000000	M5 ON	00000000	01000000
E5 OFF	10000000	01100000	M5 OFF	00000000	01100000
E6 ON	10000000	01010000	M6 ON	00000000	01010000
E6 OFF	10000000	01110000	M6 OFF	00000000	01110000
E7 ON	10000000	01001000	M7 ON	00000000	01001000
E7 OFF	10000000	01101000	M7 OFF	00000000	01101000
E8 ON	10000000	01011000	M8 ON	00000000	01011000
E8 OFF	10000000	01111000	M8 OFF	00000000	01111000
E9 ON	10000100	00000000	M9 ON	00000100	00000000
E9 OFF	10000100	00100000	M9 OFF	00000100	00100000
E10 ON	10000100	00010000	M10 ON	00000100	00010000
E10 OFF	10000100	00110000	M10 OFF	00000100	00110000
E11 ON	10000100	00001000	M11 ON	00000100	00001000
E11 OFF	10000100	00101000	M11 OFF	00000100	00101000
E12 ON	10000100	00011000	M12 ON	00000100	00011000
E12 OFF	10000100	00111000	M12 OFF	00000100	00111000
E13 ON	10000100	01000000	M13 ON	00000100	01000000
E13 OFF	10000100	01100000	M13 OFF	00000100	01100000
E14 ON	10000100	01010000	M14 ON	00000100	01010000
E14 OFF	10000100	01110000	M14 OFF	00000100	01110000
E15 ON	10000100	01001000	M15 ON	00000100	01001000
E15 OFF	10000100	01101000	M15 OFF	00000100	01101000
E16 ON	10000100	01011000	M16 ON	00000100	01011000
E16 OFF	10000100	01111000	M16 OFF	00000100	01111000
E BRIGHT	10000000	10001000	M BRIGHT	00000000	10001000
E DIM	10000000	10011000	M DIM	00000000	10011000

F1 ON	10010000	00000000	N1 ON	00010000	00000000
F1 OFF	10010000	00100000	N1 OFF	00010000	00100000
F2 ON	10010000	00010000	N2 ON	00010000	00010000
F2 OFF	10010000	00110000	N2 OFF	00010000	00110000
F3 ON	10010000	00001000	N3 ON	00010000	00001000
F3 OFF	10010000	00101000	N3 OFF	00010000	00101000
F4 ON	10010000	00011000	N4 ON	00010000	00011000
F4 OFF	10010000	00111000	N4 OFF	00010000	00111000
F5 ON	10010000	01000000	N5 ON	00010000	01000000
F5 OFF	10010000	01100000	N5 OFF	00010000	01100000
F6 ON	10010000	01010000	N6 ON	00010000	01010000
F6 OFF	10010000	01110000	N6 OFF	00010000	01110000
F7 ON	10010000	01001000	N7 ON	00010000	01001000
F7 OFF	10010000	01101000	N7 OFF	00010000	01101000
F8 ON	10010000	01011000	N8 ON	00010000	01011000
F8 OFF	10010000	01111000	N8 OFF	00010000	01111000
F9 ON	10010100	00000000	N9 ON	00010100	00000000
F9 OFF	10010100	00100000	N9 OFF	00010100	00100000
F10 ON	10010100	00010000	N10 ON	00010100	00010000
F10 OFF	10010100	00110000	N10 OFF	00010100	00110000
F11 ON	10010100	00001000	N11 ON	00010100	00001000
F11 OFF	10010100	00101000	N11 OFF	00010100	00101000
F12 ON	10010100	00011000	N12 ON	00010100	00011000
F12 OFF	10010100	00111000	N12 OFF	00010100	00111000
F13 ON	10010100	01000000	N13 ON	00010100	01000000
F13 OFF	10010100	01100000	N13 OFF	00010100	01100000
F14 ON	10010100	01010000	N14 ON	00010100	01010000
F14 OFF	10010100	01110000	N14 OFF	00010100	01110000
F15 ON	10010100	01001000	N15 ON	00010100	01001000
F15 OFF	10010100	01101000	N15 OFF	00010100	01101000
F16 ON	10010100	01011000	N16 ON	00010100	01011000
F16 OFF	10010100	01111000	N16 OFF	00010100	01111000
F BRIGHT	10010000	10001000	N BRIGHT	00010000	10001000
F DIM	10010000	10011000	N DIM	00010000	10011000

G1 ON	10100000	00000000	O1 ON	00100000	00000000
G1 OFF	10100000	00100000	O1 OFF	00100000	00100000
G2 ON	10100000	00010000	O2 ON	00100000	00010000
G2 OFF	10100000	00110000	O2 OFF	00100000	00110000
G3 ON	10100000	00001000	O3 ON	00100000	00001000
G3 OFF	10100000	00101000	O3 OFF	00100000	00101000
G4 ON	10100000	00011000	O4 ON	00100000	00011000
G4 OFF	10100000	00111000	O4 OFF	00100000	00111000
G5 ON	10100000	01000000	O5 ON	00100000	01000000
G5 OFF	10100000	01100000	O5 OFF	00100000	01100000
G6 ON	10100000	01010000	O6 ON	00100000	01010000
G6 OFF	10100000	01110000	O6 OFF	00100000	01110000
G7 ON	10100000	01001000	O7 ON	00100000	01001000
G7 OFF	10100000	01101000	O7 OFF	00100000	01101000
G8 ON	10100000	01011000	O8 ON	00100000	01011000
G8 OFF	10100000	01111000	O8 OFF	00100000	01111000
G9 ON	10100100	00000000	O9 ON	00100100	00000000
G9 OFF	10100100	00100000	O9 OFF	00100100	00100000
G10 ON	10100100	00010000	O10 ON	00100100	00010000
G10 OFF	10100100	00110000	O10 OFF	00100100	00110000
G11 ON	10100100	00001000	O11 ON	00100100	00001000
G11 OFF	10100100	00101000	O11 OFF	00100100	00101000
G12 ON	10100100	00011000	O12 ON	00100100	00011000
G12 OFF	10100100	00111000	O12 OFF	00100100	00111000
G13 ON	10100100	01000000	O13 ON	00100100	01000000
G13 OFF	10100100	01100000	O13 OFF	00100100	01100000
G14 ON	10100100	01010000	O14 ON	00100100	01010000
G14 OFF	10100100	01110000	O14 OFF	00100100	01110000
G15 ON	10100100	01001000	O15 ON	00100100	01001000
G15 OFF	10100100	01101000	O15 OFF	00100100	01101000
G16 ON	10100100	01011000	O16 ON	00100100	01011000
G16 OFF	10100100	01111000	O16 OFF	00100100	01111000
G BRIGHT	10100000	10001000	O BRIGHT	00100000	10001000
G DIM	10100000	10011000	O DIM	00100000	10011000

H1 ON	10110000 00000000	P1 ON	00110000 00000000
H1 OFF	10110000 00100000	P1 OFF	00110000 00100000
H2 ON	10110000 00010000	P2 ON	00110000 00010000
H2 OFF	10110000 00110000	P2 OFF	00110000 00110000
H3 ON	10110000 00001000	P3 ON	00110000 00001000
H3 OFF	10110000 00101000	P3 OFF	00110000 00101000
H4 ON	10110000 00011000	P4 ON	00110000 00011000
H4 OFF	10110000 00111000	P4 OFF	00110000 00111000
H5 ON	10110000 01000000	P5 ON	00110000 01000000
H5 OFF	10110000 01100000	P5 OFF	00110000 01100000
H6 ON	10110000 01010000	P6 ON	00110000 01010000
H6 OFF	10110000 01110000	P6 OFF	00110000 01110000
H7 ON	10110000 01001000	P7 ON	00110000 01001000
H7 OFF	10110000 01101000	P7 OFF	00110000 01101000
H8 ON	10110000 01011000	P8 ON	00110000 01011000
H8 OFF	10110000 01111000	P8 OFF	00110000 01111000
H9 ON	10110100 00000000	P9 ON	00110100 00000000
H9 OFF	10110100 00100000	P9 OFF	00110100 00100000
H10 ON	10110100 00010000	P10 ON	00110100 00010000
H10 OFF	10110100 00110000	P10 OFF	00110100 00110000
H11 ON	10110100 00001000	P11 ON	00110100 00001000
H11 OFF	10110100 00101000	P11 OFF	00110100 00101000
H12 ON	10110100 00011000	P12 ON	00110100 00011000
H12 OFF	10110100 00111000	P12 OFF	00110100 00111000
H13 ON	10110100 01000000	P13 ON	00110100 01000000
H13 OFF	10110100 01100000	P13 OFF	00110100 01100000
H14 ON	10110100 01010000	P14 ON	00110100 01010000
H14 OFF	10110100 01110000	P14 OFF	00110100 01110000
H15 ON	10110100 01001000	P15 ON	00110100 01001000
H15 OFF	10110100 01101000	P15 OFF	00110100 01101000
H16 ON	10110100 01011000	P16 ON	00110100 01011000
H16 OFF	10110100 01111000	P16 OFF	00110100 01111000
H BRIGHT	10110000 10001000	P BRIGHT	00110000 10001000
H DIM	10110000 10011000	P DIM	00110000 10011000

ANEXO B

Tabla de Datos del Protocolo CM11A

HOUSE CODES					KEY CODES					
	H1	H2	H4	H8		D1	D2	D4	D8	D16
A	0	1	1	0	1	0	1	1	0	0
B	1	1	1	0	2	1	1	1	0	0
C	0	0	1	0	3	0	0	1	0	0
D	1	0	1	0	4	1	0	1	0	0
E	0	0	0	1	5	0	0	0	1	0
F	1	0	0	1	6	1	0	0	1	0
G	0	1	0	1	7	0	1	0	1	0
H	1	1	0	1	8	1	1	0	1	0
I	0	1	1	1	9	0	1	1	1	0
J	1	1	1	1	10	1	1	1	1	0
K	0	0	1	1	11	0	0	1	1	0
L	1	0	1	1	12	1	0	1	1	0
M	0	0	0	0	13	0	0	0	0	0
N	1	0	0	0	14	1	0	0	0	0
O	0	1	0	0	15	0	1	0	0	0
P	1	1	0	0	16	1	1	0	0	0
			All Units Off			0	0	0	0	1
			All Lights On			0	0	0	1	1
			On			0	0	1	0	1
			Off			0	0	1	1	1
			Dim			0	1	0	0	1
			Bright			0	1	0	1	1
			All Lights Off			0	1	1	0	1
			Extended Code			0	1	1	1	1
			Hail Request			1	0	0	0	1 ^①
			Hail Acknowledge			1	0	0	1	1
			Pre-Set Dim			1	0	1	X	1 ^②
			Extended Data (analog)			1	1	0	0	1 ^③
			Status-on			1	1	0	1	1
			Status-off			1	1	1	0	1
			Status Request			1	1	1	1	1

ANEXO C

Tabla de costos de los dispositivos del sistema

Cantidad	Dispositivo	Precio	Código
1	Kit Firecracket SDK	\$ 69.00	CH2003
2	Modulo de Potencia C/ Rele 1500 W	\$ 98.00	CH3003
1	Sensor de movimientos Eagle-Eye	\$ 25.00	CH3005
1	CM11A Uinterface Universal Bidireccional	\$ 38.00	CH2000_1
1	Módulo llave decora	\$58.00	CH3002D
1	RR501-C Tranceiver	\$ 29.00	CH200_2
1	LM465 Modulo para lamparas	\$ 25.00	CH200_3
1	Telefono Nokia 6360	\$ 60.00	
1	Cable de Datos DLR-3P de Nokia	\$ 50.00	KDL3P

TOTAL**\$ 422.00**

ANEXO D

Código Fuente del Programa


```
!*****
```

```
'frmAgregarUsuario
```

```
!*****
```

```
Option Explicit
```

```
Dim nuevousuario As String
```

```
Dim Mibase As Database 'Declaramos variables tipo base y tipo recordset
```

```
Dim Mirecord As Recordset
```

```
Dim CadenaporBuscar As String
```

```
Dim Consulta As String
```

```
Private Sub Agregar_Click()
```

```
Dim i As Integer
```

```
nuevousuario = Usuarios.Text
```

```
nuevousuario = Left(nuevousuario, 2)
```

```
If (nuevousuario <> "09") Then 'Validamos que los números ingresados sean  
de teléfonos móviles
```

```
Usuarios.Text = ""
```

```
MsgBox "Número inválido"
```

```
GoTo Salir
```

```
End If
```

```
nuevousuario = Usuarios.Text
```

```
nuevousuario = Right(nuevousuario, 7)
```

```
nuevousuario = "005939" + nuevousuario
```

```
i = 0
```

```
Do
```

```
    If (nuevousuario = Usuarios.List(i)) Then GoTo Error 'Verifica que no se  
trate de un número ya existente
```

```
    i = i + 1
```

```
Loop While i < Usuarios.ListCount
```

```
Usuarios.AddItem nuevousuario 'Agrega nuevo usuario a la lista de Usuarios
```

```
    ' Agrega un nuevo record
```

```
    Mirecord.AddNew
```

```
    Mirecord("Usuarios") = nuevousuario
```

```
    Mirecord.Update
```

```
    Usuarios.Text = ""
```

```
Exit Sub
```

```
Error:
```

```
MsgBox "Usuario ya existente"
```

```
Salir:
```

```
End Sub
```

```
Private Sub Eliminar_Click()
Dim index As Integer
```

```
If Usuarios.ListIndex = -1 Then GoTo Salir 'verifica que exista al menos un
usuario para eliminar
index = Usuarios.ListIndex
CadenaporBuscar = Usuarios.List(index) ' Obtenemos la cadena por buscar
Consulta = "select * from Usuarios where Usuarios like " & "" &
CadenaporBuscar & ""
Set Mirecord = Mibase.OpenRecordset(Consulta)
If Mirecord.RecordCount = 0 Then GoTo Salir 'No se encontró registro
    Mirecord.Delete
    Mirecord.MoveNext
    MsgBox ("Registro eliminado con éxito")
Salir:
Usuarios.RemoveItem Usuarios.ListIndex
End Sub
```

```
Private Sub Form_Load()
On Error GoTo ErrorTrap
```

```
Set Mibase = OpenDatabase("Configuración.mdb") 'Abrimos la base datos
Set Mirecord = Mibase.OpenRecordset("Usuarios") 'Cargamos el recordset
Call FillFields 'Llamamos al método que carga los datos existentes desde la
base hasta la lista de Usuarios
ErrorTrap:
Exit Sub
End Sub
```

```
Private Sub Salir_Click()
frmAgregarUsuario.Hide
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
Mibase.Close
End Sub
```

```
Sub FillFields()
' este método llenará los datos en la lista de Usuarios
inicio:
```

```

    Usuarios.AddItem Mirecord.Fields("Usuarios")
    Mirecord.MoveNext
  If Mirecord.EOF Then
    Beep
    Mirecord.MoveLast
    Exit Sub
  Else
    GoTo inicio
  End If
End Sub

```

```

*****

```

```

'frmAgregarDispositivo

```

```

*****

```

```

Option Explicit
Dim dispositivo As String
Dim identificador
Dim housecode As Long
Dim ID As Long
Public códigos As String
Dim Mibase As Database 'Declaramos variables tipo base y tipo recordset
Dim Mirecord As Recordset
Dim CadenaporBuscar As String
Dim Consulta As String

```

```

Private Sub cmdSalir_Click()
frmAgregarDispositivo.Hide
End Sub

```

```

Private Sub Form_Load()
Combo1.AddItem "A"
Combo1.AddItem "B"
Combo1.AddItem "C"
Combo1.AddItem "D"
Combo1.AddItem "E"
Combo1.AddItem "F"
Combo1.AddItem "G"
Combo1.AddItem "H"
Combo1.AddItem "I"
Combo1.AddItem "J"
Combo1.AddItem "K"
Combo1.AddItem "L"
Combo1.AddItem "M"

```

```

Combo1.AddItem "O"
Combo1.AddItem "P"
*****
Combo2.AddItem "1"
Combo2.AddItem "2"
Combo2.AddItem "3"
Combo2.AddItem "4"
Combo2.AddItem "5"
Combo2.AddItem "6"
Combo2.AddItem "7"
Combo2.AddItem "8"
Combo2.AddItem "9"
Combo2.AddItem "10"
Combo2.AddItem "11"
Combo2.AddItem "12"
Combo2.AddItem "13"
Combo2.AddItem "14"
Combo2.AddItem "15"
Combo2.AddItem "16"

```

On Error GoTo ErrorTrap

```

Set Mibase = OpenDatabase("Configuración.mdb") 'Abrimos la base datos
Set Mirecord = Mibase.OpenRecordset("Dispositivos") 'Cargamos el recordset

```

Call FillFields 'Llenamos la lista de Dispositivos Existentes con los datos de la base

```

ErrorTrap:
Exit Sub
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
Mibase.Close
End Sub

```

```

Sub FillFields()
' Este procedimiento llena la lista de Dispositivos Existentes y de
Identificadores con los datos de la base
inicio:

```

```

Dispositivos.AddItem Mirecord.Fields("Nombre de pila")
Identificadores.AddItem Mirecord.Fields("Identificador")

```

```

Codigosdecasa.AddItem Mirecord.Fields("Código de Casa")
Mirecord.MoveNext
If Mirecord.EOF Then
    Mirecord.MoveLast
    Exit Sub
Else
    GoTo inicio
End If
End Sub

```

```

Private Sub cmdEliminarRecord_Click()
Dim index As Integer

If Dispositivos.ListIndex = -1 Then GoTo Salir 'Verifica que exista al menos un
registro para eliminar
index = Dispositivos.ListIndex
CadenaporBuscar = Dispositivos.List(index) ' Obtenemos la cadena por
buscar
Consulta = "select * from Dispositivos where [Nombre de pila] = " & "" &
CadenaporBuscar & ""
Set Mirecord = Mibase.OpenRecordset(Consulta)
If Mirecord.RecordCount = 0 Then GoTo Salir 'No de encontró registro
    Mirecord.Delete
    Mirecord.MoveNext
    MsgBox ("Registro eliminado con éxito")
Salir:
Identificadores.RemoveItem Dispositivos.ListIndex
Dispositivos.RemoveItem Dispositivos.ListIndex
End Sub

```

```

Private Sub cmdAgregarRecord_Click()
Dim i As Integer
'Verificamos que se llenen todos los campos

If (Combo1.Text = "") Then
MsgBox "Favor llenar todos los campos"
GoTo Salir
End If

```

```

If (txtNombreDispositivo = "") Then
MsgBox "Favor llenar todos los campos"
GoTo Salir

```

End If

```
If (Combo2.Text = "") Then
MsgBox "Favor llenar todos los campos"
GoTo Salir
End If
```

```
dispositivo = txtNombreDispositivo.Text
```

```
i = 0
```

```
Do
```

```
  If (dispositivo = Dispositivos.List(i)) Then GoTo Error 'Verificamos que no
se ingrese un registro ya existente
  i = i + 1
```

```
Loop While i < Dispositivos.ListCount
```

```
  identificador = Combo2.Text
```

```
  i = 0
```

```
Do
```

```
  If (identificador = Identificadores.List(i)) Then GoTo Error1 'Verificamos
que no se ingrese un registro ya existente
  i = i + 1
```

```
Loop While i < Identificadores.ListCount
```

```
Identificadores.AddItem identificador
```

```
Dispositivos.AddItem dispositivo
```

```
códigos = Combo1.Text
```

```
Codigosdecasa.AddItem códigos
```

```
'Agrega un nuevo record
```

```
  Mirecord.AddNew
```

```
  Mirecord("Nombre de pila") = dispositivo
```

```
  Mirecord("Código de Casa") = códigos
```

```
  Mirecord("Identificador") = identificador
```

```
  Mirecord.Update
```

```
  'cmdAgregar.Enabled = False
```

```
Dispositivos.Text = "" 'Limpiamos las cajas de texto
```

```
txtNombreDispositivo.Text = ""
```

```
Combo1.Text = ""
```

```
Combo2.Text = ""
```

```

CodigosdeCasa.Text = ""
Identificadores.Text = ""
Exit Sub
Error:
MsgBox "Nombre de pila ya existente"
Exit Sub

```

```

Error1:
MsgBox "Identificador ya existente"
Salir:
End Sub

```

```

!*****
'frmConfigSMSControl
!*****

```

```

Option Explicit
Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)
Private smsMemObj As NokiaCLMessaging.IMemory
Private submitObj As NokiaCLMessaging.ITDMASubmit
Private smsObj As NokiaCLMessaging.ShortMsgAdapter
Private messageItemObj As NokiaCLMessaging.ShortMessageItem
Private deliverObj As NokiaCLMessaging.ITDMADeliver
Public i0 As Integer 'Variable que contendrá el número inicial de mensajes
existentes en el móvil

```

```

Private Sub A2_OFF_Click()
SendHeader
SendByte &H60
SendByte &H30
SendFooter
End Sub

```

```

Private Sub A2_ON_Click()
SendHeader
SendByte &H60
SendByte &H10
SendFooter
End Sub

```

```

Private Sub Combo3_Click()
Dim puerto As Integer
On Error GoTo ErrorTrap

```

```

If Combo3.Text = "COMM 1" Then
    puerto = 1
    MSComm1.CommPort = 1
    MSComm1.PortOpen = True
End If
If Combo3.Text = "COMM 2" Then
    puerto = 2
    MSComm1.CommPort = 2
    MSComm1.PortOpen = True
End If
If Combo3.Text = "COMM 3" Then
    puerto = 3
    MSComm1.CommPort = 3
    MSComm1.PortOpen = True
End If
If Combo3.Text = "COMM 4" Then
    puerto = 4
    MSComm1.CommPort = 4
    MSComm1.PortOpen = True
End If
ErrorTrap:
    If Err.HelpContext = 0 Then
        MSComm1.PortOpen = False
MSComm1.CommPort = puerto ' Utilizamos COM4
MSComm1.DTREnable = True ' Configuramos DTR como HIGH
MSComm1.RTSEnable = True ' Configuramos RTS como HIGH
MSComm1.PortOpen = True ' Finalmete abrimos el puerto
Pause 50 ' Esperamos un momento
Reset ' Reseteamos FireCracker

        MsgBox "Puerto:" & Combo3.Text & " seleccionado"
        End If

    If Err.HelpContext = 8005 Or Err.HelpContext = 8002 Then
        MsgBox "Source: " & Err.Source & Chr$(10) _
            & "Error Description: Número de puerto en uso o no válido" & Chr$(10) &
            "HelpContext: " & Err.HelpContext
        End If
End Sub

Private Sub Form_Load()

Set smsMemObj = New NokiaCLMessaging.ShortMsgAdapter
Set smsObj = New NokiaCLMessaging.ShortMsgAdapter

```



```

    Set messageItemObj = New NokiaCLMessaging.ShortMessageItem
    Combo3.AddItem "COMM 1"
    Combo3.AddItem "COMM 2"
    Combo3.AddItem "COMM 3"
    Combo3.AddItem "COMM 4"
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
    Set smsMemObj = Nothing
    Set smsObj = Nothing
    Set messageItemObj = Nothing
End Sub

```

```

Private Sub ReadSMS_Click()
    Timer1.Interval = 10000 ' Establecer el intervalo del timer.
    Dim i As Integer
    Dim capacity As Integer
    Dim messages As Integer
    Dim usuario As Integer
    Dim unreadmessages As Integer

```

```

    Call smsMemObj.GetMemoryStatus(SHORTMESSAGE_MEMORY_ME,
    capacity, messages, unreadmessages)
    i0 = messages ' Cargamos i0 con el número inicial de mensajes existentes en
    el móvil
End Sub

```

```

Private Sub Salir_Click()
    frmConfigSMSControl.Hide
End Sub

```

```

Private Sub Timer1_Timer()
    Dim códigos
    Dim cdi As String
    Dim Comando As String
    Dim capacity As Integer
    Dim messages As Integer
    Dim unreadmessages As Integer
    Dim mensaje As Variant
    Dim usuario As Variant
    Dim sw As Boolean
    Dim housecode As String
    Dim X10HOUSECODE As Long
    Dim i As Integer ' Contadores para el manejo de índices de las listas

```

```

Dim j As Integer
Dim k As Integer
Dim tope As Integer
Dim tope2 As Integer
On Error GoTo Leerotravez
mensaje = " "
usuario = " "

```

```

Call smsMemObj.GetMemoryStatus(SHORTMESSAGE_MEMORY_ME,
capacity, messages, unreadmessages)
i = messages
If (i <= i0) Then GoTo Salida

```

Leerotravez:

```

Set messageItemObj = smsObj.ReadSMS(SHORTMESSAGE_MEMORY_DEFAULT, 0, i)
Set deliverObj = messageItemObj.TypeProperties
mensaje = deliverObj.Message
usuario = deliverObj.OriginatorAddress

```

'-----Validamos el número entrante, verificando la lista de usuario
tope = frmAgregarUsuario.Usuarios.ListCount 'Obtenemos el número de
datos existentes en la lista

```

j = -1
Do While j <= tope - 1
j = j + 1
If usuario = frmAgregarUsuario.Usuarios.List(j) Then
Beep
Exit Do
End If
Loop

```

'-----Validamos el comando recibido en la lista de comandos almacenados

```

k = -1
Do
k = k + 1
If (UCase(mensaje) = UCase(frmAgregarDispositivo.Dispositivos.List(k)
+ " " + "1")) Then
Text2.Text = UCase(mensaje)
Beep
GoTo salida1
End If

```

```

        If (UCase(mensaje) = UCase(frmAgregarDispositivo.Dispositivos.List(k)
+ " " + "0")) Then
            Text2.Text = UCase(mensaje)
            Beep
            GoTo salida1
        End If
Loop While k <= frmAgregarDispositivo.Dispositivos.ListCount - 1
GoTo Salida
salida1:
' -----
Comando = mensaje
Comando = UCase(Right(Comando, 1))

If Comando = "1" Then
sw = True
End If
If Comando = "0" Then
sw = False
End If
cdi = frmAgregarDispositivo.Identificadores.List(k)
housecode = frmAgregarDispositivo.Codigosdecasa.List(k)
Call GetActionCode(sw, cdi, housecode)
Salida:
        cdi = Empty
        k = Empty
        i0 = i
End Sub

Public Sub Send_1()
MSComm1.DTREnable = False      ' Configuramos DTR como Low
Pause 1                        ' Nos ponemos en estado de espera
MSComm1.DTREnable = True       ' Configuramos DTR nuevamente como
High
Pause 1
End Sub

Public Sub Send_0()
MSComm1.RTSEnable = False      ' Configuramos RTS como Low
Pause 1                        ' Nos ponemos en estado de espera
MSComm1.RTSEnable = True       ' Configuramos RTS nuevamente como
High
Pause 1

```

```
End Sub
```

```
Public Sub Send2Byte(iByte As Long)
Dim i As Integer
For i = 0 To 15          ' Lazo para todos los 8 bits
If (iByte And &H8000) = &H8000 Then ' Si b15 es 1 entonces
Send_1          ' Enviamos 1
Else            ' Si b15 es 0 entonces
Send_0          ' Enviamos 0
End If
iByte = iByte * 2      ' desplazamos los bits un lugar a la izquierda
Next i                ' Seguimos con el siguiente bit
End Sub
```

```
Public Sub SendByte(iByte As Long)
Dim i As Integer
For i = 0 To 7          'Lazo para todos los 8 bits
If (iByte And &H80) = &H80 Then ' Si b8 es 1 entonces
Send_1          'Enviamos 1
Else            ' Si b15 es 0 entonces
Send_0          'Enviamos 0
End If
iByte = iByte * 2      'desplazamos los bits un lugar a la izquierda
Next i                'Seguimos con el siguiente bit
End Sub
```

```
Public Sub Reset()
MSComm1.RTSEnable = False ' Configuramos RTS y DTR
MSComm1.DTREnable = False ' como Low para especificar un Reset
Pause 50                 ' Nos permitimos un tiempo para el Reset
MSComm1.RTSEnable = True  ' Configuramos RTS y DTR como high
MSComm1.DTREnable = True  ' para ponernos en modo de StandBy
Pause 50                 ' Otra vez una pausa
End Sub
```

```
Public Sub SendHeader()
SendByte &HD5          ' Enviamos D5
SendByte &HAA         ' Enviamos AA
End Sub
```

```
Public Sub SendFooter()
SendByte &HAD          ' Enviamos AD
End Sub
```

```
Public Sub Pause(milli As Long)
Sleep (milli)
End Sub
```

```
Public Sub GetActionCode(sw As Boolean, cdi As String, housecode As
String)
Dim ID As Long
Dim X10HCODE As Long
Dim X10HOMECODE As Long
Dim X10COM As Long
' -----Obtnemos el código de casa
Select Case housecode
Case "A"
X10HCODE = &H60
Case "B"
X10HCODE = &H70
Case "C"
X10HCODE = &H40
Case "D"
X10HCODE = &H50
Case "E"
X10HCODE = &H80
Case "F"
X10HCODE = &H90
Case "G"
X10HCODE = &HA0
Case "H"
X10HCODE = &HB0
Case "I"
X10HCODE = &HE0
Case "J"
X10HCODE = &HF0
Case "K"
X10HCODE = &HC0
Case "L"
housecode = &HD0
Case "M"
X10HCODE = &H0
```

```

Case "N"
X10HCODE = &H10
Case "O"
X10HCODE = &H20
Case "P"
X10HCODE = &H30
End Select

```

```

Select Case cdi
Case "1", "2", "3", "4", "5", "6", "7", "8"
ID = &H0
Case "9", "10", "11", "12", "13", "14", "15", "16"
ID = &H4
End Select

```

```
X10HOMECODE = X10HCODE + ID
```

' -----Seleccionamos el codigo de accion ON - OFF

```

If (sw) Then
Select Case cdi
  Case "1"
    X10COM = &H0
  Case "2"
    X10COM = &H10

```

```

  Case "3"
    X10COM = &H8
  Case "4"
    X10COM = &H18
  Case "5"
    X10COM = &H40
  Case "6"
    X10COM = &H50
  Case "7"
    X10COM = &H48
  Case "8"
    X10COM = &H58
  Case "9"
    X10COM = &H0
  Case "10"
    X10COM = &H10
  Case "11"
    X10COM = &H8
  Case "12"
    X10COM = &H18

```

```
Case "13"  
X10COM = &H40  
Case "14"  
X10COM = &H50  
Case "15"  
X10COM = &H48  
Case "16"  
X10COM = &H58  
End Select  
End If  
If (Not sw) Then  
Select Case cdi  
Case "1"  
X10COM = &H20  
Case "2"  
X10COM = &H30  
Case "3"  
X10COM = &H28  
Case "4"  
X10COM = &H38  
Case "5"  
X10COM = &H60  
Case "6"  
X10COM = &H70  
  
Case "7"  
X10COM = &H68  
Case "8"  
X10COM = &H78  
Case "9"  
X10COM = &H20  
Case "10"  
X10COM = &H30  
Case "11"  
X10COM = &H28  
Case "12"  
X10COM = &H38  
Case "13"  
X10COM = &H60  
Case "14"  
X10COM = &H70  
Case "15"  
X10COM = &H68  
Case "16"
```

```

    X10COM = &H78
    End Select
    End If
' -----Enviamos el código de casa más el código de operación
SendHeader
SendByte X10HOMECODE
SendByte X10COM
SendFooter
End Sub

```

```

!*****

```

```

'frmConfigSMSControl version CM11A

```

```

!*****

```

```

Option Explicit
Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)
Public released As Boolean
Private WithEvents pCallAdapter As NokiaCLCall.CallAdapter
Attribute pCallAdapter.VB_VarHelpID = -1
Private pIncomingCall As NokiaCLCall.CallItem
Private pCallItem As NokiaCLCall.CallItem
Private pCallInHold As NokiaCLCall.CallItem
Private bytOldCallId As Byte
Private bytIncomingCallId As Byte
Private blnIsCallInHold As Boolean
Private bytConferenceCall As Byte
Private lngEventCounter As Long
Private lngNumberOfCalls As Long
Private smsMemObj As NokiaCLMessaging.IMemory
Private submitObj As NokiaCLMessaging.ITDMASubmit
Private smsObj As NokiaCLMessaging.ShortMsgAdapter
Private messageItemObj As NokiaCLMessaging.ShortMessageItem
Private deliverObj As NokiaCLMessaging.ITDMADeliver
Private Declare Function WaitForSingleObject Lib "kernel32" (ByVal hHandle
As Long, ByVal dwMilliseconds As Long) As Long
Public i0 As Integer
Private Sub Command1_Click()
On Error GoTo ErrorTrap
Dim name As String
Dim msgStatus As NokiaCLMessaging.ShortMessageStatus
Dim nameIndicator As NokiaCLMessaging.SMSNameIndicator
Dim msgType As NokiaCLMessaging.ShortMessageType

```



```

'Call smsMemObj.ReadSMS(SHORTMESSAGE_MEMORY_DEFAULT, 0,
16, nameIndicator, msgtype, msgStatus, name)
Set          messageItemObj          =
smsObj.ReadSMS(SHORTMESSAGE_MEMORY_DEFAULT, 0, 1)
    Set deliverObj = messageItemObj.TypeProperties
    'Text1.Text = messageItemObj.MessageStatus

    'Text2.Text = deliverObj.message
    'Text3.Text = deliverObj.OriginatorAddress
'Text1.Text = name
  ' Text2.Text = nameIndicator
  'Text3.Text = msgStatus
Exit Sub
ErrorTrap:
  MsgBox "Source: " & err.Source & Chr$(10) _
    & "Error Description: " & err.Description _
    & Chr$(10) & "HelpContext: " & err.HelpContext
End Sub
Private Sub Form_Load()
  On Error GoTo ErrorTrap
Set smsMemObj = New NokiaCLMessaging.ShortMsgAdapter
  Set smsObj = New NokiaCLMessaging.ShortMsgAdapter
  Set messageItemObj = New NokiaCLMessaging.ShortMessageItem
  Combo7.ListIndex = 0
Set pCallAdapter = New NokiaCLCall.CallAdapter

  pCallAdapter.EnableNotifications
  blnIsCallInHold = False
  'cmdHangUp.Enabled = False
  lngNumberOfCalls = 0
  bytConferenceCall = 0
  Exit Sub
ErrorTrap:
  ShowError
End Sub
Private Sub Form_Unload(Cancel As Integer)
  Set smsMemObj = Nothing
  Set smsObj = Nothing
  Set messageItemObj = Nothing
  pCallAdapter.DisableNotification
  Set pCallAdapter = Nothing
  Set pCallItem = Nothing
End Sub

```

```

Private Sub ReadSMS_Click()
Timer1.Interval = 10000 ' Establecer el intervalo.
Dim i As Integer
Dim capacity As Integer
Dim messages As Integer
Dim usuario As Integer
Dim unreadmessages As Integer
Call smsMemObj.GetMemoryStatus(SHORTMESSAGE_MEMORY_ME,
capacity, messages, unreadmessages)
i0 = messages
End Sub

```

```

Private Sub Salir_Click()
frmConfigSMSControl.Hide
End Sub
Private Sub Timer1_Timer()
Dim códigos
Dim cdi As String
Dim Comando As String
Dim capacity As Integer
Dim action As Integer
Dim MiTiempo As Variant
Dim MiFecha As Variant
Dim caracter As String
Dim messages As Integer
Dim unreadmessages As Integer
Dim mensaje As Variant
Dim usuario As Variant
Dim sw As Boolean
Dim housecode As String
Dim X10HOUSECODE As Long
Dim i As Integer
Dim j As Integer
Dim k As Integer
Dim tope As Integer
Dim tope2 As Integer
    MiFecha = Date
    MiTiempo = Time ' Devuelve la hora del sistema actual.
    caracter = Chr(10)
On Error GoTo Leertravez
mensaje = " "
usuario = " "

```

```

Call smsMemObj.GetMemoryStatus(SHORTMESSAGE_MEMORY_ME,
capacity, messages, unreadmessages)
i = messages
If (i <= i0) Then GoTo Salida
Leerotravez:
    Set messageItemObj = smsObj.ReadSMS(SHORTMESSAGE_MEMORY_DEFAULT, 0, i)
    Set deliverObj = messageItemObj.TypeProperties
    mensaje = deliverObj.message
    usuario = deliverObj.OriginatorAddress

'-----Validamos el número entrante, verificando la lista de usuario

tope = frmAgregarUsuario.Usuarios.ListCount
j = -1
Do While j <= tope - 1
    j = j + 1
    If usuario = frmAgregarUsuario.Usuarios.List(j) Then
        Beep
        'MsgBox "Match"
        Exit Do
    End If
Loop
'-----Validamos el comando recibido en la lista de comandos almacenados
k = -1
Do
    k = k + 1
    If (UCase(mensaje) = UCase(frmAgregarDispositivo.Dispositivos.List(k)
+ " " + "1")) Then
        Beep
        'Text2.Text = UCase(mensaje)
        'MsgBox "Match"
        GoTo salida1
    End If
    If (UCase(mensaje) = UCase(frmAgregarDispositivo.Dispositivos.List(k)
+ " " + "0")) Then
        Beep
        'Text2.Text = UCase(mensaje)
        'MsgBox "Match"
        GoTo salida1
    End If
Loop While k <= frmAgregarDispositivo.Dispositivos.ListCount - 1
GoTo Salida
salida1:

```

```

'-----
Comando = mensaje
Comando = UCase(Right(Comando, 1))
If Comando = "1" Then
sw = True
'MsgBox "Comando es On"
End If
If Comando = "0" Then
sw = False
'MsgBox "Comando es OFF"
End If

'Text1.Text = k
'Text2.Text = frmAgregarDispositivo.Identificadores.List(k)
TxtDC = frmAgregarDispositivo.Identificadores.List(k)
TxtHC = frmAgregarDispositivo.Codigosdecasa.List(k)

If (sw) Then
    action = 2
    If (controlcm1.ExecWait(TxtHC, TxtDC, action, Val(TxtDim),
Val(TxtDim), Val(TxtData2))) Then
        'Beep
        'MsgBox "Fallo al enviar" & TxtHC & TxtDC & action
        Else
        Beep
        'MsgBox "Exito al enviar" & TxtHC & TxtDC & i
        TxtEvent = TxtEvent + "Acción: " & TxtHC & TxtDC & action & " " &
MiFecha & " " & MiTiempo & caracter
        End If
        If Not (frmAgregarDispositivo.Dispositivos.List(k) = "") Then
        reporte.AddItem frmAgregarDispositivo.Dispositivos.List(k) + Comando
        End If

'controlcm1.Exec TxtHC, TxtDC, action, Val(TxtDim), Val(TxtDim),
Val(TxtData2)
Else
    action = 3
    If (controlcm1.ExecWait(TxtHC, TxtDC, action, Val(TxtDim),
Val(TxtDim), Val(TxtData2))) Then
        'MsgBox "Fallo al enviar" & TxtHC & TxtDC & action
        Else
        Beep

```

```

        MsgBox "Exito al enviar" & TxtHC & TxtDC & i
        TxtEvent = TxtEvent + "Acción: " & TxtHC & TxtDC & action & " " &
MiFecha & " " & MiTiempo & caracter
        End If
        If Not (frmAgregarDispositivo.Dispositivos.List(k) = "") Then
            reporte.AddItem frmAgregarDispositivo.Dispositivos.List(k) + Comando
'Agregamos acción y evento al reporte
        End If
    End If
    Salida:
        cdi = Empty
        k = Empty
        i0 = i
    End Sub

```

```

Private Sub Butlnit_Click()
    Dim err As Integer
    controlcm1.comport = Val(TxtComPort)
    Butlnit.Caption = "Espere..."
    err = controlcm1.Init
    If err <> 0 Then
        MsgBox "Error inicializando CM11A: puerto en uso ó no válido",
vbCritical + vbOKOnly
        Butlnit.Caption = "Inicializar"
    Else
        Butlnit.Caption = "finalizado"
        Butlnit.Enabled = False
    End If
End Sub

```

```

Private Sub Command4_Click()
    TxtEvent = ""
End Sub

```

```

Private Sub controlcm1_X10Event(devices As String, housecode As String,
command As Integer, extra As String, data2 As String)
    Dim myCommonBlock As NokiaCLCall.ICommonSubBlocks
    Dim myCallMode As CallMode
    Dim arrModelInfo() As CallModelInfo ' SAFEARRAY
    Dim optMI As Byte
    Dim callnbr As String

```

```

    Dim arrSize As Long
    arrSize = 0
On Error GoTo ErrorTrap
    TxtEvent = TxtEvent + "Evento: " + housecode + " " + devices +
Str(command) + " dimval: " + extra + "D2: " + data2 + vbCrLf
If (command = 3) And (frmConfigAlarmas.ActivacionSensor = True) Then
    Beep
'MsgBox "Alguien a activado el sensor de movimientos"
If (frmConfigAlarmas.NúmerosTelefónicos911.ListCount > 0) Then
    'Read the number from a field in the form
    callnbr = frmConfigAlarmas.NúmerosTelefónicos911.List(0)
    Set pCallItem = pCallAdapter.CreateCallItem
    ' Or the optional way:
    ' Set myCallItem = New NokiaCLCall.CallItem
    arrSize = arrSize + 1
    ReDim Preserve arrModelInfo(arrSize)
    arrModelInfo(arrSize - 1) = CA_CALL_GSM_MODE_SIM_CALL
    optMI = 0
    myCallMode = CA_CALL_MODE_SPEECH
    ' Make the QueryInterface:
    Set myCommonBlock = pCallItem
    Call myCommonBlock.SetDTMFString(494)
    Call myCommonBlock.SetMode(arrModelInfo, optMI, myCallMode)
    Call myCommonBlock.SetDestAddress( _
        CA_CALL_NBR_PLAN_PRIVATE, _
        CA_CALL_NBR_TYPE_UNKNOWN, _
        CA_CALL_PRESENTATION_ALLOWED, _
        CA_CALL_SCREEN_NETW_PROVIDED, _
        callnbr)
    'cmdHangUp.Enabled = True
    'cmdHoldCall.Enabled = True

    'If (blnIsCallInHold = True) Then
        'cmdCreateConferenceCall.Enabled = True
        'cmdSwapCalls.Enabled = True
    'End If
    lngNumberOfCalls = lngNumberOfCalls + 1

    ' this should be before enabling the command buttons
    ' but since this function always raises an error
    ' this is the only solution to do this
    Call pCallAdapter.CreateCall(pCallItem)
Exit Sub
ErrorTrap:

```

```
ShowError
Exit Sub
End If
End If
End Sub
```

```
Private Sub controlcm1_X10SingleEvent(devices As String, housecode As
String, command As Integer, extra As String, data2 As String)
    If frmConfigAlarmas.ActivacionSensor = True Then
        TxtEvent = TxtEvent + "Single Event: " + housecode + " " + devices + " " +
Str(command) + vbCrLf
    End If
End Sub
```

```
Private Sub txtEvent_Change()
If TxtEvent.Text = "Call Released" Then
released = True
End If
End Sub
```

```
Public Sub Pause(milli As Long)
Sleep (milli)
End Sub
```

```
Private Sub ShowEvent(ByVal strEvent As String)
    lngEventCounter = lngEventCounter + 1
    TxtEvent.SelStart = Len(TxtEvent)
    TxtEvent.SelLength = 0
    TxtEvent.SelText = (lngEventCounter & ". " & strEvent) & vbNewLine
End Sub
```

```
' This function is used to show error messages from all the forms
Public Sub ShowError()
    MsgBox "Source: " & err.Source & Chr$(10) _
        & "ErrorDescription: " & err.Description _
        & Chr$(10) & "HelpContext: " & err.HelpContext
End Sub
```

```

Private Sub pCallAdapter_CallReleased(ByVal pICallItem As
NokiaCLCall.ICallItem)
    ShowEvent ("Call Released")
End Sub

```

```

Private Sub pCallAdapter_CallCreating(ByVal pICallItem As
NokiaCLCall.ICallItem)
    ShowEvent ("Call Creating")
End Sub

```

```

Private Sub pCallAdapter_CallAnswered(ByVal byCallID As Byte)
On Error GoTo TrapError
Dim myDTMFInterfase As NokiaCLCall.ICallDTMFEx
Dim myCommonBlock As NokiaCLCall.ICommonSubBlocks
Dim sbList() As SubBlockType
Dim i As Long
Dim bFound As Boolean
Dim byDTMFRet As Byte
On Error GoTo TrapError
released = False

```

```

    ShowEvent ("Call Answered")
    Pause (2000)

```

```

bFound = False
sbList = pCallItem.GetSubBlockList
For i = 0 To UBound(sbList)

```

```

    If sbList(i) = CA_DTMF_STRING Then
        bFound = True
        Exit For
    End If
Next i

```

```

If bFound = True Then
Set myDTMFInterfase = pCallAdapter
Pause (150)
Do
Pause (250)
byDTMFRet = myDTMFInterfase.DTMFSend(False, pCallItem)
Pause (250)

```



```

Loop Until released = True
released = False
Set myDTMFInterfase = Nothing
End If
Set myCommonBlock = Nothing
Exit Sub
TrapError:
Exit Sub
MsgBox err.Description
End Sub

```

```

!*****

```

```

'frmConfigAlarmas

```

```

!*****

```

```

Option Explicit
Public ActivacionSensor As Boolean
Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)
Private pSMSAdapter As NokiaCLMessaging.ShortMsgAdapter
Private pIGSMSSubmit As NokiaCLMessaging.IGSMSSubmit
Private pSMSMessageItem As NokiaCLMessaging.ShortMessageItem
Private smsMemObj As NokiaCLMessaging.IMemory
Private submitObj As NokiaCLMessaging.ITDMASubmit
Private smsObj As NokiaCLMessaging.ShortMsgAdapter
Private messageItemObj As NokiaCLMessaging.ShortMessageItem
Private deliverObj As NokiaCLMessaging.ITDMADeliver
Private intervalo As String
Dim AlarmTime 'tiempo de alarma de encendido
Dim AlarmTime2 'tiempo de alarma de apagado
Dim AlarmTime3 ' tiempo de alarma de reportes SMS
Const conMinimized = 1
Dim correcto As Boolean
Dim Mibase As Database 'Declaramos variables tipo base y tipo recordset
Dim Mirecord As Recordset

Private Sub AgregarRegistro911_Click()
If Not (IsNumeric(NúmerosTelefónicos911.Text)) Then GoTo Error
'Validamos los números ingresados
If Not (Len(NúmerosTelefónicos911.Text) = 9) Then GoTo Error
NúmerosTelefónicos911.AddItem NúmerosTelefónicos911.Text
NúmerosTelefónicos911.Text = ""
Exit Sub

```

```

Error:
MsgBox "Número no válido"
NúmerosTelefónicos911.Text = ""
End Sub

```

```

Private Sub AgregarDiario_Click()
Dim i As Integer
If Not (DispositivosExistentes1.Text = "") Then
i = 0
Do
    If (DispositivosExistentes1.Text = DispositivosSeleccionados1.List(i)) Then
GoTo Error 'Validamos que no se seleccione dos veces el mismo dispositivo
        i = i + 1

```

```

Loop While i < DispositivosSeleccionados1.ListCount
DispositivosSeleccionados1.AddItem DispositivosExistentes1.Text
End If
Exit Sub
Error:
MsgBox "Dispositivo ya seleccionado"
End Sub

```

```

Private Sub Aplicar1_Click()
Dim i As Integer
'Verifica si la hora ingresada es valida
If Not (IsDate(hora1.Text + ":" + minutos1.Text + " " + AMPM.Text)) Then
GoTo Error
If Not (IsDate(hora2.Text + ":" + minutos2.Text + " " + AMPM1.Text)) Then
GoTo Error
If DispositivosSeleccionados1.ListCount = 0 Then GoTo Error
AlarmTime = CDate(hora1.Text + ":" + minutos1.Text + " " + AMPM.Text)
'Hora de activación de dispositivos
If (AlarmTime < Time) Then GoTo Error
correcto = True
AlarmTime2 = CDate(hora2.Text + ":" + minutos2.Text + " " + AMPM1.Text)
'Hora de desactivación de dispositivos
Label8.Caption = AlarmTime
Exit Sub
Error:
MsgBox "Error en uno de los campos ingresados"
End Sub

```

```

Private Sub Aplicar3_Click()
'Verifica si el intervalo de duración en minutos es válido
If Not (IsNumeric(duración3.Text)) Then GoTo Error
If (CInt(duración3.Text) > 59) Then GoTo Error
If (CInt(duración3.Text) < 10) Then GoTo Error
If (AlarmTime = "" And AlarmTime2 = "") Then GoTo Error
intervalo = duración3.Text
AlarmTime3 = "0" + ":" + intervalo + ":" + "00" 'Primer intervalo de envío de
reporte
AlarmTime3 = AlarmTime + CDate(AlarmTime3)
Label8.Caption = AlarmTime3
Exit Sub
Error:
MsgBox "Intervalo de tiempo muy corto o muy grande"
duración3.Text = ""
Exit Sub
Error1:
MsgBox "Llenar primero hora de activación y desactivación"
End Sub

```

```

Private Sub Eliminar911_Click()
If NúmerosTelefónicos911.ListIndex = -1 Then GoTo Salir
NúmerosTelefónicos911.RemoveItem NúmerosTelefónicos911.ListIndex
NúmerosTelefónicos911.Text = ""
Salir:
End Sub

```

```

Private Sub Form_Load()
Dim i As Integer
NúmerosTelefónicos911.Enabled = False
duración3.Enabled = False
ActivacionSensor = False
AlarmTime = ""
AlarmTime2 = ""
AlarmTime3 = ""
Set smsMemObj = New NokiaCLMessaging.ShortMsgAdapter
Set smsObj = New NokiaCLMessaging.ShortMsgAdapter
Set messageItemObj = New NokiaCLMessaging.ShortMessageItem
AMPM.AddItem "am"
AMPM.AddItem "pm"
AMPM1.AddItem "am"
AMPM1.AddItem "pm"

```

```

' El control TabStrip se llama "TabStrip2".
' El control Frame se llama "Configuración".
For i = 1 To Configuración.Count
With Configuración(i)
.Move TabStrip2.ClientLeft, _
TabStrip2.ClientTop, _
TabStrip2.ClientWidth, _
TabStrip2.ClientHeight
End With
Next i
' Trae al frente el primer control fraTab.
Configuración(1).ZOrder 0
On Error GoTo ErrorTrap

```

```

Set Mibase = OpenDatabase("Configuración.mdb") 'Abrimos la base datos
Set Mirecord = Mibase.OpenRecordset("Dispositivos") 'Cargamos el
recordset

```

```

Call FillFields 'Funcion para sacar los elementos de la tabla correspondiente
y ponerlos en la lista
ErrorTrap:
Exit Sub
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
Mibase.Close
Set smsMemObj = Nothing
Set smsObj = Nothing
Set messageItemObj = Nothing
ActivacionSensor = False
End Sub

```

```

Private Sub Option1_Click()
duración3.Enabled = True
End Sub
Private Sub Option2_Click()
duración3.Enabled = False
End Sub
Private Sub Option3_Click()
NúmerosTelefónicos911.Enabled = True

```

```

ActivacionSensor = True
End Sub
Private Sub Option4_Click()
NúmerosTelefónicos911.Enabled = False
End Sub
Private Sub Salir1_Click()
frmConfigAlarmas.Hide
End Sub
Private Sub Salir2_Click()
frmConfigAlarmas.Hide
End Sub
Private Sub Salir3_Click()
frmConfigAlarmas.Hide
End Sub
Private Sub TabStrip2_Click()
Configuración(TabStrip2.SelectedItem.Index).ZOrder 0
End Sub

```

```

Private Sub SetCaptionTime()
Caption = Format(Time, "Medium Time") ' Presenta la hora con el formato
Medium Time.
End Sub
Private Sub Timer1_Timer()
Dim flag As Boolean
Dim i As Integer
Dim j As Integer
Dim k As Integer
Dim DC As String
Dim HC As String
Dim MiFecha As Variant
Dim MiTiempo As Variant
Dim caracter As String
Static AlarmSounded As Integer
Static AlarmSounded2 As Integer
Static AlarmSounded3 As Integer
Dim longitud
Dim report As String
Dim message As String
Dim smsEntry As NokiaCLMessaging.ShortMessageItem

```

'Verifica hora de activación y ejecuta los comandos X10

!*****

```

If lblTime.Caption <> CStr(Time) Then
  ' Ahora el número de segundo es diferente del mostrado.
  If Time >= AlarmTime And Not AlarmSounded And (correcto = True)
Then
  Beep
  MsgBox "Alarma a las " & Time, , "Alarma"
  AlarmSounded = True
  For i = 0 To DispositivosSeleccionados1.ListCount
  For j = 0 To frmAgregarDispositivo.Dispositivos.ListCount
  If (DispositivosSeleccionados1.List(i) =
frmAgregarDispositivo.Dispositivos.List(j)) Then
  frmConfigSMSControl.TxtDC =
frmAgregarDispositivo.Identificadores.List(j)
  frmConfigSMSControl.TxtHC =
frmAgregarDispositivo.Codigosdecasa.List(j)
  If
(frmConfigSMSControl.controlcm1.ExecWait(frmConfigSMSControl.TxtHC,
frmConfigSMSControl.TxtDC, 2, Val(frmConfigSMSControl.TxtDim),
Val(frmConfigSMSControl.TxtDim), Val(frmConfigSMSControl.TxtData2)))
Then
  MsgBox "Fallo al enviar" & HC & DC
  Else
  MiFecha = Date
  MiTiempo = Time
  caracter = Chr(10)

  MsgBox "Exito al enviar" & HC & DC
  frmConfigSMSControl.TxtEvent = frmConfigSMSControl.TxtEvent
+ "Acción: " & DispositivosSeleccionados1.List(i) &
frmConfigSMSControl.TxtDC & 2 & " " & MiFecha & " " & MiTiempo &
caracter
  If Not (DispositivosSeleccionados1.List(i) = "") Then
  frmConfigSMSControl.reporte.AddItem
DispositivosSeleccionados1.List(i) + "ON"
  End If
  End If
  End If
  Next j
  Next i
  AlarmSounded = True
  Exit Sub

```

```

End If
  Elself Time < AlarmTime Then
    AlarmSounded = False
  End If

'Verifica la hora de desactivación y ejecuta los comandos
*****
If lblTime.Caption <> CStr(Time) Then
  ' Ahora el número de segundo es diferente del mostrado.
  If Time >= AlarmTime2 And Not AlarmSounded2 Then
    Beep
    MsgBox "Alarma a las " & Time, , "Alarma"
    AlarmSounded2 = True
    For i = 0 To DispositivosSeleccionados1.ListCount
      For j = 0 To frmAgregarDispositivo.Dispositivos.ListCount
        If (DispositivosSeleccionados1.List(i) =
frmAgregarDispositivo.Dispositivos.List(j)) Then
          frmConfigSMSControl.TxtDC =
frmAgregarDispositivo.Identificadores.List(j)
          frmConfigSMSControl.TxtHC =
frmAgregarDispositivo.Codigosdecasa.List(j)
          If
(frmConfigSMSControl.controlcm1.ExecWait(frmConfigSMSControl.TxtHC,
frmConfigSMSControl.TxtDC, 3, Val(frmConfigSMSControl.TxtDim),
Val(frmConfigSMSControl.TxtDim), Val(frmConfigSMSControl.TxtData2)))
Then
            MsgBox "Fallo al enviar" & HC & DC
            Else
            MiFecha = Date
            MiTiempo = Time
            caracter = Chr(10)

            MsgBox "Exito al enviar" & HC & DC
            frmConfigSMSControl.TxtEvent = frmConfigSMSControl.TxtEvent
+ "Acción: " & DispositivosSeleccionados1.List(i) &
frmConfigSMSControl.TxtDC & 2 & " " & MiFecha & " " & MiTiempo &
caracter
            If Not (DispositivosSeleccionados1.List(i) = "") Then
            frmConfigSMSControl.reporte.AddItem
DispositivosSeleccionados1.List(i) + "ON"
            End If
            End If
          End If
        End If
      End If
    End If
  End If

```

```

        Next j
    Next i
    AlarmSounded2 = True
    Exit Sub
End If
    ElseIf Time < AlarmTime2 Then
        AlarmSounded2 = False
    End If

```

'Verifica el tiempo para el envío de reportes X10

!*****

```

If lblTime.Caption <> CStr(Time) Then
    ' Ahora el número de segundo es diferente del mostrado.
    If Time >= AlarmTime3 And Not AlarmSounded3 Then
        Beep
        MsgBox "Alarma a las " & Time, , "Alarma"
        AlarmSounded3 = True

        If (frmConfigSMSControl.reporte.ListCount = 0) Then 'Si no
tenemos nada que reportar
            Exit Sub
        End If
        k = frmConfigSMSControl.reporte.ListCount - 1
        Do
            report = report + " " + frmConfigSMSControl.reporte.List(k)
'Concatenamos todos los estados en un solo string para que sea enviado
            longitud = Len(report)
            k = k - 1
            Loop Until (longitud = 157 Or k < 0)
            On Error GoTo ErrorTrap
            Set smsEntry = New NokiaCLMessaging.ShortMessageItem
            smsEntry.Type = SHORTMESSAGE_TYPE_TDMA_SUBMIT
            Set submitObj = smsEntry.TypeProperties
            submitObj.AcknowledgeRequest = 0
            submitObj.CallbackName = "Joffre"
            submitObj.CallbackNumber = "094467056"
            submitObj.DeliveryTimeDaylightSaving = 0
            submitObj.DeliveryTimeMinutes = 0
            submitObj.DeliveryTimeSeconds = 0
            submitObj.DeliveryTimeZone = 1
            submitObj.message = report
            'submitObj.message = "Dios ayudame por favor"

```



```

        ' submitObj.MessageReference = 35
        submitObj.OriginatorAddress = "094467056"
        submitObj.DestinationAddress = "094554244"
        submitObj.Privacy = 0
        submitObj.UpdateStatus = 1
        submitObj.Urgency = 3
        'submitObj.ValidityPeriodTimeDaylightSaving =
MESSAGE_TIME_DAYLIGHT_SAVING_TIME
        Call
smsObj.SendSMS(SHORTMESSAGE_ROUTE_TYPE_GPRS, submitObj)
        MsgBox ("Message sent with reference number: ")
        AlarmSounded3 = True
        If (AlarmTime3 < AlarmTime2) Then
            AlarmTime3 = AlarmTime3 + CDate("0" + ":" + intervalo + ":" +
"00")
            Label8.Caption = AlarmTime3
        End If
        Exit Sub
    End If
    ElseIf Time < AlarmTime3 Then
        AlarmSounded3 = False
    End If
Exit Sub
ErrorTrap:
MsgBox "Source: " & err.Source & Chr$(10) _
    & "Error Description: " & err.Description _
    & Chr$(10) & "HelpContext: " & err.HelpContext
End Sub

Private Sub Timer2_Timer()
Static AlarmSounded3 As Integer
End Sub

Sub FillFields()
    ' Este procedimiento llena las listas con los datos provenientes de la base
inicio:
    DispositivosExistentes1.AddItem Mirecord.Fields("Nombre de pila")
    Mirecord.MoveNext
    If Mirecord.EOF Then
        Beep
        Mirecord.MoveLast
    
```

```

Exit Sub
Else
GoTo inicio
End If
End Sub

```

```

*****

```

```

'SMS Control
*****
Option Explicit
Private Sub MDIForm_Unload(Cancel As Integer)
'Termina la Aplicación
Unload frmConfigSMSControl
Unload frmAgregarUsuario
Unload frmAgregarDispositivo
Unload frmSplash
Unload Me
End Sub

```

```

Private Sub mnuControlSMS_Click()
frmConfigSMSControl.Show 'Muestra el formulario ConfigSMSControl
End Sub

```

```

Private Sub mnuDispositivoX10_Click()
frmAgregarDispositivo.Show 'Muestra el formulario AgregarDispositivo
End Sub

```

```

Private Sub mnuUsuariosSMS_Click()
frmAgregarUsuario.Show 'Muestra el formulario AgregarUsusario
End Sub

```

```

*****

```

```

'SMSControl CM11A
*****

```

```

Option Explicit

```

```

Private Sub MDIForm_Unload(Cancel As Integer)
'Termina la Aplicación
Unload frmConfigSMSControl

```

```
Unload frmAgregarUsuario
Unload frmAgregarDispositivo
Unload frmSplash
Unload frmConfigAlarmas
Unload Me
End Sub
```

```
Private Sub mnuControlSMS_Click()
'Muestra el formulario apropiado
frmConfigSMSControl.Show
End Sub
```

```
Private Sub mnuDispositivoX10_Click()
'Muestra el formulario apropiado
frmAgregarDispositivo.Show
End Sub
```

```
Private Sub mnuRutinasyAlarmasx10_Click()
'Muestra el formulario apropiado
frmConfigAlarmas.Show
End Sub
```

```
Private Sub mnuUsuariosSMS_Click()
'Muestra el formulario apropiado
frmAgregarUsuario.Show
End Sub
```

BIBLIOGRAFÍA

1. Wayne Tomasi, Sistemas de Comunicaciones Electrónicas, 2da Edición, Pearson Education
2. Leon Couch, Digital and Analog Communicatons Systems, 5ta Edición, Prentice Hall
3. Corporate Author, Microsoft Visual Basic 6.0 Reference Library, Microsoft Press (August 1, 1998)
4. Michael Harvorson, Microsoft Visual Basic 6.0 Professional Step-By-Step, Microsoft Press; Bk&CD-Rom edition (July 15, 1998)
5. Teoría del Protocolo X10
<http://www.smarthomeusa.com/info/x10theory/x10theory/>
6. Dispositivos X10
<http://www.x10.com>
7. Problemas con X10
<http://www.smarthomeusa.com/info/trouble/trouble/>
8. Herramientas de desarrollo de software de Nokia
<http://www.forum.nokia.com>
9. Conceptos básicos sobre SMS
http://www.iec.org/online/tutorials/wire_sms