RIELAC, Vol.XXXIV 3/2013 p.74-88 Septiembre - Diciembre ISSN: 1815-5928

RIELAC
Revista de Ingeniería Electrónica, Automática y Comunicaciones

Sistema de control de acceso e interbloqueo para el Centro de Inmunología Molecular

Marcel Pedreira Marcel, **Valery Moreno Vega**

RESUMEN / *ABSTRACT*

En el presente trabajo se muestra el diseño y desarrollo de un sistema de control de acceso e interbloqueo en el Centro de Inmunología Molecular debido a que los sistemas comerciales de este tipo instalados en dicho centro no cumplen todas las necesidades. El sistema propuesto consta de dos tarjetas electrónicas: el controlador de puertas y el controlador de interbloqueo, ambas fueron desarrolladas a base de microcontrolador PIC de Microchip programados usando el compilador PCW de CCS. Estas tarjetas son capaces de comunicarse con dispositivos de lectura de código de barras, proximidad, biométricos o cualquier otro que transmita por protocolo Wiegand. Además deben ser configuradas para que operen de manera deseada, para ello fue desarrollada una aplicación software de parametrización utilizando Qt como framework, e implementando prácticas eficientes de ingeniería de software. Esta aplicación se comunica con los controladores vía RS232 con protocolo Modbus.

Palabras claves: Control de acceso, Interbloqueo, Microcontrolador, Modbus, Wiegand, Ingeniería de software

This paper shows the design and development of an access control and interlocking system for the Molecular Immunology Center because the commercial systems installed in this center do not meet all needs. The proposed system consists of two electronic boards: the door controller and the interlock controller, both were developed based on Microchip's PIC microcontroller programmed using CCS PCW compiler. These cards are able to communicate with devices like barcode reading, proximity, biometric or any other that transmit by Wiegand protocol. They must also be configured to operate in a desired manner and therefore a software application was developed using Qt as framework, and implementing effective practices of software engineering. This application communicates with the controllers via RS232 using Modbus protocol.

Key words: Access control, Interlocking, Microcontroller, Modbus, Wiegand, Software engineering

Access control and interlocking system for the Molecular Immunology Center

INTRODUCCION

El concepto de control de acceso se ha utilizado en varios contextos, y frecuentemente se asocia a conceptos de seguridad informática y redes de computadoras. Sin embargo, en este trabajo se hace alusión al sistema vinculado a las puertas de una edificación. Bajo estos principios, un sistema de control de acceso administra el ingreso a áreas restringidas y evita que personas no autorizadas o indeseables tengan la libertad de acceder a determinados locales. Así mismo se puede tener conocimiento de la asistencia del personal, horarios de ingreso y egreso, y también poder tener un control histórico de entradas de personas a todas las áreas.

Los sistemas de control de acceso están formados por diversos dispositivos o componentes. Los controladores de puerta constituyen el elemento fundamental o de inteligencia en un sistema de control de acceso. Por otra parte, los dispositivos de lectura se utilizan para la identificación del usuario a través de alguna tecnología, como puede ser código de barras,

radiofrecuencia o biometría. Los elementos de acción final se encargan del bloqueo físico de las puertas, éstos pueden ser hembrillas o cerraduras electrónicas, electroimanes, etc. Además, estos sistemas presentan, normalmente, un software de supervisión que opera sobre todos los controladores dispuestos en red, y de esta forma realizar labores de monitoreo y control. Es posible también integrar circuitos cerrados de televisión, sistemas contra incendios, contra intrusos y mecanismos de interbloqueo. Mediante estos últimos, se puede bloquear el acceso a un área común entre dos puertas cuando una de ellas se ha abierto.

El Centro de Inmunología Molecular (CIM), perteneciente al polo científico en el oeste de la capital cubana, es una institución especializada en la investigación, desarrollo y manufactura de productos biofarmacéuticos para la lucha contra el cáncer a partir del cultivo de células de mamíferos de acuerdo con las regulaciones de las buenas prácticas de producción (GMP). Este Centro, al igual que otros del polo dedicados a la biotecnología, posee áreas de producción denominadas salas limpias especialmente diseñadas para obtener bajos niveles de contaminación. Estas salas tienen sus parámetros ambientales estrictamente controlados: partículas en el aire, temperatura, humedad, flujo de aire, presión interior, iluminación etc.[1]

El sistema de control de acceso juega un importante papel en lograr tal objetivo. Actualmente en el CIM se encuentran en funcionamiento dos sistemas de control de acceso e interbloqueo de diferentes fabricantes: Kantech y Siemens. Estos sistemas tienen las siguientes desventajas en su aplicación:

- Arquitectura cerrada. Existe incompatibilidad en cuanto a protocolos de comunicación, bases de datos etc. Por tal motivo la información esta fragmentada o dividida lo que dificulta una gestión a nivel central.
- Soporte técnico. Se han presentado dificultades con la obtención de las tarjetas de proximidad y actualmente se encuentran agotadas existiendo trabajadores con necesidad de entrar a determinada área que no cuentan con el medio para hacerlo.
- A pesar de ser sistemas bien eficientes y robustos, se han dado circunstancias particulares en las que no son eficaces o incapaces de resolver el problema usando el sistema por si solo con las opciones que trae por definición.

Por este motivo se ha planteado la necesidad de implementar un nuevo sistema de control de acceso e interbloqueo, que sea desarrollado totalmente en el CIM. Éste permitirá una gestión central de la información y un mayor nivel de configuración, personalización y expansión, permitiendo suplir las necesidades no resueltas por los sistemas de pago instalados en la actualidad.

En este trabajo se describe el proceso de desarrollo del sistema propuesto para dar solución a la problemática anterior. En la sección 2 se da una descripción, a grandes rasgos, de dicho sistema haciendo hincapié en los elementos que lo componen así como los protocolos utilizados. En la sección 3 se detalla el diseño de los controladores, dígase hardware y firmware, requerido para su funcionamiento. La sección 4 aborda todo el proceso de desarrollo de la aplicación software de parametrización de cada uno de los controladores. Finalmente, en la sección 5 se describen los resultados obtenidos en la implementación del sistema a escala piloto.

DESCRIPCIÓN GENERAL DEL SISTEMA PROPUESTO

La solución presentada consiste en un sistema compuesto por dos elementos fundamentales: el controlador de puerta y el controlador de interbloqueo. Ambos son tarjetas electrónicas con finalidades diferentes dentro del sistema.

La función del controlador de puerta es, como su nombre lo indica, el control de las puertas. Un controlador de este tipo puede controlar un máximo de dos puertas. De esta forma se encarga de recibir la solicitud de acceso que viene del elemento de identificación, determinar si el código arribado tiene acceso por la puerta especificada según previa configuración y, de ser positiva o válida, entonces desbloquear la puerta en cuestión. En la Figura 1 se muestra la conexión de este controlador. Cabe destacar que éste es capaz de recibir la codificación enviada desde cualquier lector que implemente protocolo Wiegand formato 26. Este protocolo es abierto y se considera un estándar dentro de los sistemas de control de acceso. De esta forma se garantiza la comunicación con una amplia gama de dispositivos de lectura que existen en la actualidad.

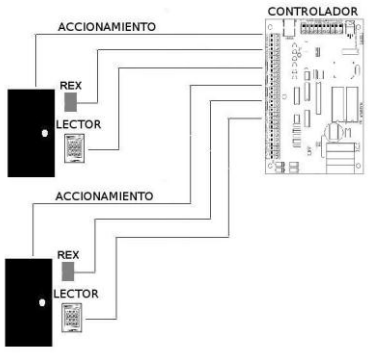

Figura 1. Conexión del Controlador de Puerta.

Otro elemento asociado al sistema es un dispositivo para detectar el caso de que un usuario desee salir de un local a través de alguna puerta controlada, es decir, por la parte contraria en donde se encuentra situado el lector. Estos casos se denominan REX (Request for EXit) en la mayoría de los sistemas de control de acceso, y para ello se incluye en el sistema un elemento hardware que notifique, a través de una señal, esta acción al controlador. Este elemento puede ser un sensor de presencia o un simple botón.[2]

El controlador de interbloqueo se encarga de agregar la capacidad de interbloqueo al sistema. Este mecanismo consiste en el bloqueo de una o más puertas tras la detección de la apertura de una puerta determinada. Esto se realiza, por supuesto, con una previa configuración del controlador, donde se establece esta relación o interacción entre las puertas. Este sistema puede tener muchos usos o aplicaciones y, en el caso del CIM, se utiliza para disminuir los riesgos de contaminación cruzada, concepto asociado a las GMP en el que una persona saliendo del área limpia pueda inducir contaminación a aquella que está entrando. La Figura 2 muestra la conexión de este dispositivo.

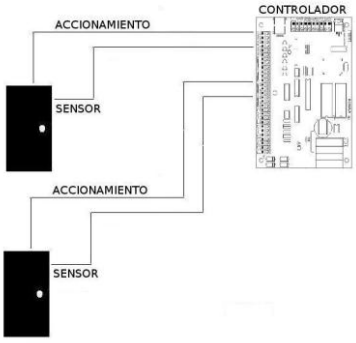

Figura 2. Conexión del Controlador de Interbloqueo.

Estos dispositivos deben ser configurados, o parametrizados, para que realicen la función deseada. Para ello se desarrolla una herramienta de software que se comunique con éstos a través de un puerto RS-232 utilizando una PC. El protocolo de comunicación entre la aplicación y los dispositivos es Modbus. Se implementó este protocolo abierto para que los controladores sean capaces de comunicarse con el mayor número de softwares posible, facilitando de esta manera futuras implementaciones o facilidades al sistema, como la posibilidad de supervisión central a través de un SCADA, por citar un ejemplo.

Las dos tarjetas están basadas en microcontroladores PIC de Microchip, por la gran difusión que existe de estos dispositivos y por la disponibilidad de éstos con sus herramientas asociadas para implementar y verificar aplicaciones de este tipo en el CIM. La programación de los microcontroladores se hizo utilizando el compilador PCW de CCS, por motivos de conocimiento del autor y también por disponibilidad legal de dicho software en el Centro. Las pruebas de la configuración de hardware de los dispositivos se realizaron utilizando el PICSCHOOL, herramienta consistente en una maqueta o protoboard para la puesta a punto de aplicaciones utilizando microcontroladores PIC. La aplicación de parametrización fue desarrollada utilizando el framework Qt, por ser una poderosa plataforma de código abierto muy difundida en estos momentos.

Protocolo Wiegand

Este protocolo es unidireccional y permite el traspaso de datos desde un lector hacia el controlador. A pesar de que fue concebido para los lectores de efecto Wiegand, dada su eficiencia en este ámbito, ha sido implementado también en lectores con otra tecnología, como los de proximidad o radiofrecuencia e, incluso, en teclados numéricos. El protocolo establece líneas de datos, alimentación y señalización. Las líneas de señalización son las usadas para el manejo de la bocina y los leds, que normalmente poseen estos lectores, mientras que la transmisión de datos se realiza a través de tres hilos:

- DATA1: Línea para enviar los unos lógicos.
- DATA0: Línea para enviar los ceros lógicos.
- GND: Línea de tierra de referencia de ambos.

La Figura 3 representa gráficamente este sistema de transmisión. En estado de reposo las líneas DATA1 y DATA0 están a nivel alto. Para transmitir un bit en '1' se manda un pulso a nivel bajo de 50 μs de duración por la línea DATA1, mientras DATA0 permanece en alto. Por el contrario, para transmitir un bit en '0' lo que se hace es enviar un pulso bajo, de la misma duración, pero por la línea DATA0 mientras la línea DATA1 permanece en alto. La separación entre cada pulso y el siguiente es de unos 2 ms.

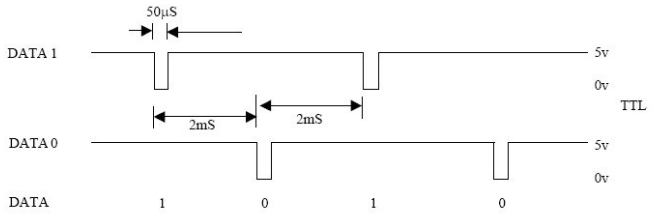

Figura 3. Tren de pulsos 1010 en el protocolo Wiegand

Mediante este sistema se puede transmitir cualquier número de bits. Existe, sin embargo, un consenso internacional para utilizar un determinado número de bits y la interpretación de los mismos. El formato más utilizado, y por tanto el empleado en este proyecto, es el Wiegand26, que especifica que el primer bit (B0) es la paridad par de los primeros 12 bits transmitidos (B1:12), los ocho siguientes (B1:B8) constituyen un entero de 8 bits denominado FacilityCode, los 16 bits siguientes (B9:B24) representan un entero de 16 bits llamado UserCode y el último bit (B25) es la paridad impar de los últimos 12 bits transmitidos (B13:B24).[2]

Protocolo Modbus

El protocolo Modbus está basado en la arquitectura maestro/esclavo. Diseñado en 1979 por la empresa Modicon para su gama de PLCs, se ha convertido en un protocolo de comunicaciones estándar de facto en la industria y es el que goza de mayor disponibilidad para la conexión de dispositivos inteligentes. Ésto es debido fundamentalmente a las siguientes razones:

- Es público.
- Su implementación es fácil y requiere poco desarrollo.
- Maneja bloques de datos sin suponer restricciones.

En ocasiones, en la literatura se refiere a Modbus como una estructura de mensaje, debido a que no especifica una capa física, por lo que se puede implementar usando diferentes normas, como RS232, RS422, RS485 e, incluso, sobre Ethernet. Existen dos variantes con diferentes representaciones numéricas de los datos y detalles del protocolo ligeramente desiguales: Modbus RTU y Modbus ASCII. También existe una versión Modbus TCP, pero más bien es el formato RTU estableciendo la transmisión mediante paquetes TCP/IP. La variante utilizada en este proyecto es la RTU.
En una red Modbus cada dispositivo tiene una dirección única. Cualquier dispositivo puede enviar órdenes, aunque lo habitual es permitirlo solo al maestro y los esclavos se limitan a responder las solicitudes del mismo. Un mensaje Modbus enviado desde el

maestro hacia un esclavo contiene la dirección del esclavo, el comando u orden a realizar, los datos asociados al comando, y una información redundante o check sum para asegurar la integridad de la recepción. El protocolo establece una gran variedad de comandos o funciones, y en este proyecto solo son utilizados los asociados a la modificación del valor de varios de los registros del esclavo (writeMultipleRegisters) y la solicitud del contenido de dichos registros (readHoldingRegisters). Cabe destacar que el estándar oficial establece registros de 16 bits entre otras especificaciones.[3]
Luego, como ya se refirió anteriormente, en este proyecto se utiliza protocolo Modbus RTU sobre RS232, donde la PC es el dispositivo maestro y los controladores son los esclavos, aunque se conecta uno a la vez. Por este motivo, no hay necesidad de especificar direcciones para cada uno de los esclavos, pues todos los mensajes enviados por el maestro son broadcast o de difusión (dirección 0) y solo serán recepcionados por el único esclavo conectado al mismo. No obstante, la implementación de este protocolo brinda la posibilidad de expandir el sistema resultante para de una manera fácil, sin muchos cambios significativos, introducir un sistema de supervisión o SCADA y conectar los controladores conformando una red Modbus.

DESARROLLO DE LOS CONTROLADORES

Como se señaló previamente, las tarjetas que representan los controladores que componen el sistema propuesto están basadas en microcontroladores PIC. En este caso se utilizó un 16F877 para cada tarjeta. Éste es un microcontrolador de 8 bits, perteneciente a la gama media de esta familia y que cumple perfectamente con los requerimientos de ambas aplicaciones, dígase cantidad de memoria, número de entradas y salidas entre otros aspectos. A continuación se expone la configuración circuital de las mencionadas tarjetas, así como el firmware programado en los microcontroladores para su adecuado funcionamiento.

Hardware del controlador de puerta

En la Figura 4 se muestra un esquema de la configuración de hardware de la tarjeta que representa el controlador de puerta.

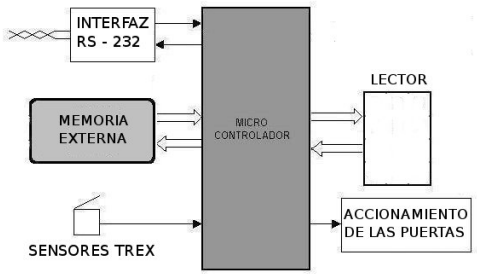

Figura 4. Diagrama en bloques del Controlador de Puerta.

Se utilizaron las líneas del nible alto del Puerto B para la recepción de los datos enviados desde los dos lectores que se pueden conectar a un controlador pues, según lo descrito anteriormente sobre el protocolo Wiegand, cada lector utiliza dos líneas para la transmisión de los datos: DATA0 y DATA1. Se conectó de esta forma para usar la interrupción por cambio de estado en las cuatro entradas más significativas del puerto B. De esta forma, cualquier variación en alguno de estos pines interrumpe la operación del microcontrolador coincidiendo con el arribo de algún bit perteneciente al tren de pulsos correspondiente a un código enviado por un lector. Otras seis líneas de entrada/salida del microcontrolador, esta vez configuradas como salidas, fueron utilizadas para el manejo de los lectores. En este caso se reservaron para la conexión de tres hilos de señalización que proveen la mayoría de los lectores que implementan el mencionado protocolo para la activación de los leds rojo y verde, así como la bocina.
La memoria externa se conectó usando interface I^2C, y es la encargada de almacenar los códigos de los usuarios autorizados a acceder a alguna de las puertas pertenecientes al controlador en cuestión. Se decidió que la memoria fuera externa por dos razones fundamentales:

1. La memoria EEPROM interna del microcontrolador no cumple con las necesidades de espacio.
2. Facilita el mantenimiento y reconfiguración del sistema al ser posible la exportación/importación de los datos.

La estructura de esta memoria se muestra en la Figura 5. Como se puede apreciar, se utilizan cuatro localizaciones para un código de usuario. En la primera se guarda el Facility Code, que como se dijo anteriormente es un número de 8 bits. A continuación se almacena el User Code, que es de 16 bits, por tanto se usan dos localizaciones para ello, en la primera de éstas la parte alta y en la

segunda la parte baja de este número. Finalmente la cuarta localización del código la ocupa la política de acceso, que no es más que un número que identifica el tipo de acceso que tiene el usuario propietario del código en cuestión. En la Tabla 1 se relacionan los tipos de acceso y su correspondiente número de política de acceso.

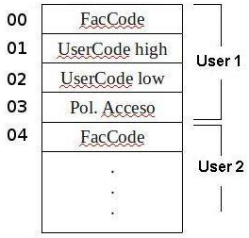

Figura 5. Estructura de la memoria EEPROM I^{2}C del controlador de puerta.

Tabla 1. Identificación de la política de acceso.

Identificador	Acceso
0	Acceso a ninguna puerta.
1	Acceso a Puerta 1.
2	Acceso a Puerta 2.
3	Acceso a las dos puertas.

El número máximo de usuarios para un controlador se fijó a 256, cantidad que satisface perfectamente las necesidades de la aplicación. Luego, con la estructura antes expuesta, una memoria EEPROM I^{2}C de 1Kx8 localizaciones es necesaria. Para este objetivo se utilizó el circuito integrado ST24W08M6R que constituye una memoria con las características requeridas.
El canal asincrónico del puerto serie, o USART del microcontrolador, se reservó para la conexión de la tarjeta a una PC y realizar la configuración o parametrización del controlador por esta vía. Es por ésto que las líneas del puerto serie fueron acopladas a un MAX232, circuito integrado que adapta los niveles de voltaje que maneja el microcontrolador a los específicos de la norma RS-232. Por esta vía se comunica la PC con el microcontrolador usando protocolo Modbus.
Se usaron dos líneas más de entrada/salida del microcontrolador (configuradas como salidas) para el accionamiento de las puertas. Este puede ser basado en diversos tipos de cierre electrónico: hembrilla, electroimán etc. y, en cualquier caso cabe destacar que estos elementos son alimentados con 24VDC o 12VDC. Luego se acopló a estas dos salidas del microcontrolador el circuito necesario para el accionamiento a base de relays.
Otras tres líneas de entrada/salida del microcontrolador (esta vez configuradas como entradas) fueron usadas para detectar las señales REX que, como se dijo anteriormente, pueden proceder de un botón o un sensor de presencia. En cualquier caso debe conectarse el dispositivo sensor a una de las entradas destinadas para ésto, en dependencia de la puerta que se trate. La configuración de hardware del controlador de puerta incluye un circuito de acondicionamiento de estas señales digitales a las entradas del microcontrolador, así como una lógica para que una activación de cualquiera de estas señales interrumpa al mismo por la línea correspondiente a la interrupción externa.[4-5]

Firmware del controlador de puerta

El papel del controlador de puerta es la recepción del código enviado desde cualquiera de los dos lectores a través del protocolo Wiegand y compararlo con los almacenados en la memoria externa para que, en caso de encontrarse coincidencia y se aplique la política de acceso, se proceda a dar el acceso a la puerta en cuestión. El firmware del microcontrolador se obtuvo utilizando el compilador PCW de CSS, como ya se refirió anteriormente, utilizando lenguaje C para la programación. Este compilador posee diversas facilidades para el programador, en este trabajo se utilizaron los drivers que brinda esta herramienta para el manejo de dispositivos I^{2}C y para la implementación de un dispositivo Modbus esclavo respectivamente. Luego el programa está seccionado fundamentalmente en tres funciones:

1. La función principal: main.

2. La función de atención por interrupción por cambio de estado en cualquiera de los cuatro pines más significativos del puerto B.
3. La función de atención por interrupción externa.

En la Figura 6 se muestra el diagrama de flujo de la función principal**.** En esta función lo que se hace es encuestar constantemente por el arribo de algún mensaje Modbus. En caso afirmativo, se identifica cuál es el comando enviado desde el maestro a través del mensaje en cuestión, y se realiza la acción correspondiente, que no es más que la lectura o escritura de los registros, y se envía una respuesta al maestro. Los comandos que se utilizan en esta aplicación son readHoldingRegister, para la lectura por parte del maestro de los registros que almacenan los códigos de usuario del controlador conectado, y writeMultipleRegisters en caso de que se quiera modificar el contenido de estos registros.[3]

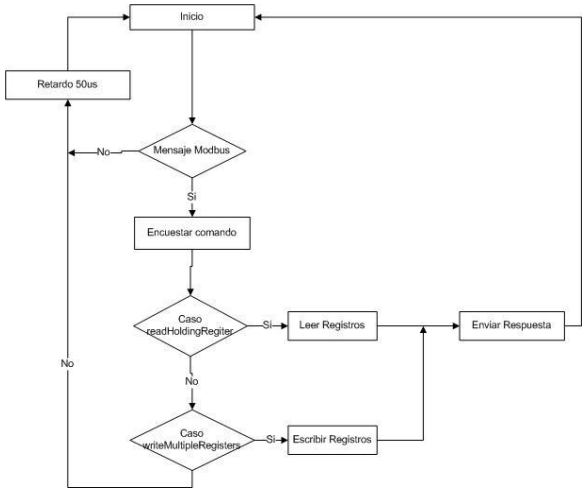

Figura 6. Diagrama de flujo de la función principal del firmware del controlador de puerta.

La Figura 7 ofrece un diagrama de flujo de la función de atención a la interrupción por cambio de estado en los cuatro bits más significativos del puerto B. El objetivo fundamental de esta función es la actualización del buffer de código Wiegand, pues se ejecutará con la llegada de algún bit correspondiente a dicho código. Esta interrupción puede presentarse tanto por un flanco de caída como por uno de subida en cualquiera de los pines más significativos del puerto B**.** Luego, analizando el funcionamiento del protocolo Wiegand, expuesto con anterioridad, se puede inferir que el arribo de un bit por alguna de las líneas de datos, representado por un pulso a nivel bajo, provocara dos interrupciones por cambio de estado en el pin conectado a dicha línea. Por este motivo se decidió realizar la actualización del buffer de código Wiegand solo con la detección del flanco de bajada del pulso correspondiente a la llegada de un bit, por tanto es necesario primeramente leer el puerto y descartar los posibles casos en que la interrupción haya sido motivada por un flanco de subida. En caso afirmativo, es decir que la interrupción haya sido motivada por un flanco de bajada, se identifica el pin por el que se produce el pulso para de esta forma actualizar el buffer de código con ’1’ o ’0’ según corresponda (DATA0 o DATA1) y establecer la puerta de donde viene el código. Finalmente, se comprueba si se llegó al final del código, en cuyo caso se debe determinar si el código arribado está autorizado, es decir, si está almacenado en la memoria de códigos de usuario. En caso afirmativo, se concede el acceso si se aplica la política de acceso.
La función de atención a la interrupción externa lo único que hace es determinar por cual de las dos puertas se produjo el REX y luego permitir el acceso. Se entiende por conceder acceso por una puerta, la acción de desbloqueo de la misma por parte del controlador por un período de tiempo de cinco segundos.

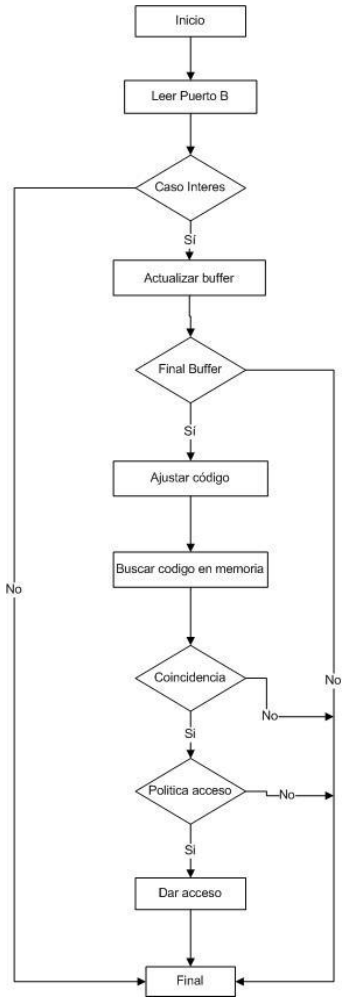

Figura 7. Diagrama de flujo de la función de interrupción por Puerto B del firmware del controlador de puerta.

Hardware del controlador de interbloqueo

En la Figura 8 se muestra un esquema de la configuración de hardware de la tarjeta que representa el controlador de interbloqueo.

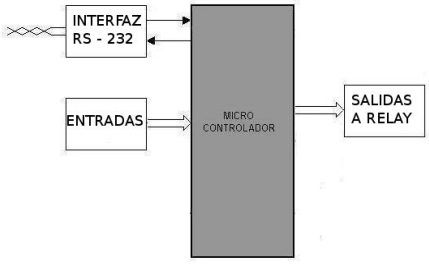

Figura 8. Diagrama en bloques del Controlador de Interbloqueo.

Como se puede apreciar, el hardware del controlador de interbloqueo es mucho más simple comparado con el del controlador de puerta**.** Está formado, en su estructura básica, por ocho entradas digitales con los correspondientes circuitos de acondicionamiento a las entradas del microcontrolador, y ocho salidas a relay. Al igual que el controlador de puerta, se usa el canal USART del microcontrolador para la comunicación con la PC a través de protocolo Modbus, por tanto también es necesario el empleo del circuito integrado MAX232 explicado anteriormente.[4-5]

Firmware del controlador de interbloqueo

La función básica del controlador de interbloqueo no es más que encuestar las entradas y, en dependencia de lo leído y la configuración previa realizada, activar las salidas correspondientes. La configuración se realiza a través de una aplicación de parametrización que corre en una PC y que se conecta a la tarjeta vía RS-232, la cual es guardada en la memoria EEPROM interna del microcontrolador. La estructura de esta memoria se muestra en la Figura 9. Como se puede apreciar, en las localizaciones de la misma se almacena un número que codifica las salidas que deben activarse cuando se activa alguna entrada. De esta forma, en las dos primeras localizaciones se encuentra la codificación de las salidas para la entrada 0, a continuación la de la entrada 1 y así sucesivamente. Se usaron dos localizaciones para cada entrada debido a que el protocolo Modbus utiliza registros de 16 bits.

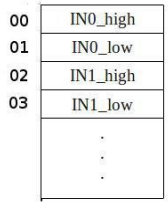

Figura 9. Estructura de la memoria EEPROM del microcontrolador del controlador de interbloqueo.

El algoritmo de la función principal del programa (main) se representa en la Figura 10. Está basado fundamentalmente en la constante lectura del puerto que constituye las entradas del controlador para luego mapearla con la configuración almacenada en la memoria y, de esta forma, conformar el resultado que luego se saca por el puerto de salida. A continuación se encuesta por la llegada de un mensaje por protocolo Modbus y, en caso afirmativo, se procede de la mima forma que en el programa del controlador de puerta. Es decir, se identifica el comando enviado desde el maestro a través del mensaje recibido, se realiza la acción correspondiente y se envía una respuesta al maestro.

DESARROLLO DE LA APLICACIÓN DE PARAMETRIZACIÓN

Para la configuración de los controladores fue necesario desarrollar una aplicación de software para que, de manera fácil, amigable e intuitiva, el operador llevara a cabo dicha parametrización. Esta aplicación debe comunicarse haciendo uso del puerto serie RS-232 de la PC con los controladores para, de esta forma, escribir y leer la configuración implementando protocolo Modbus. El desarrollo se llevó a cabo usando el framework Qt, por las razones antes expuestas, con C++ como lenguaje de programación.

Se decidió implementar una aplicación de parametrización diferente para cada controlador por razones de comodidad y claridad. No obstante las dos son muy similares, y solamente difieren en pequeños aspectos en cuanto a la forma de manejar los datos y representar la información. Esto se ilustrará mejor a medida que se vaya abordando el proceso de desarrollo de las aplicaciones, lo cual se describe a continuación.

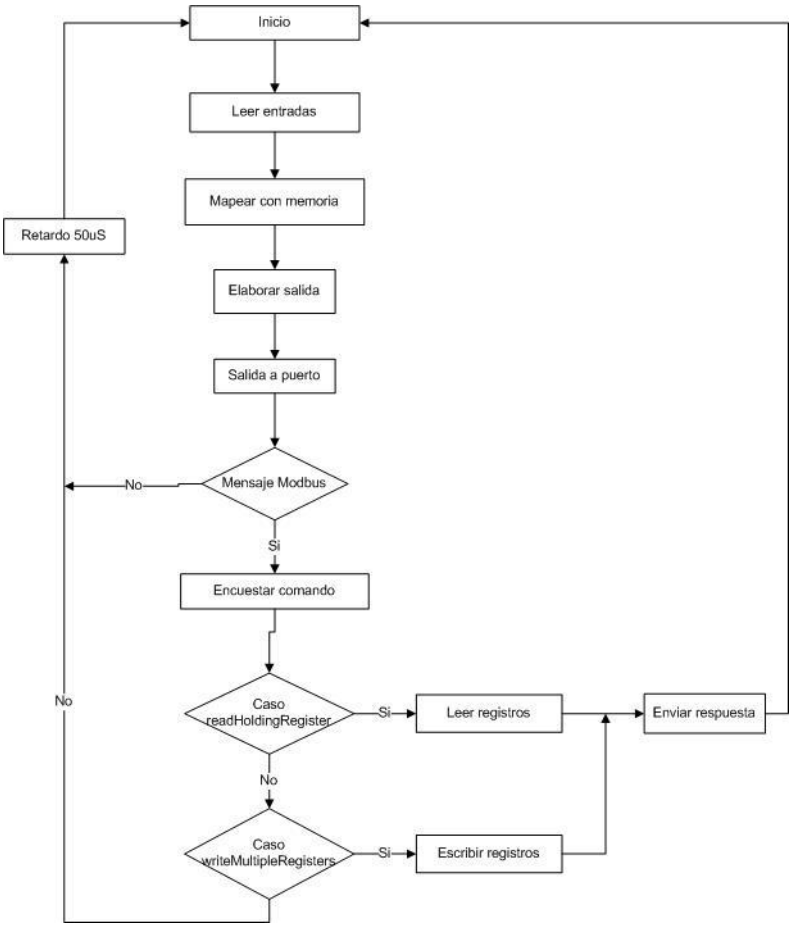

Figura 10. Diagrama de flujo de la función principal del firmware del controlador de interbloqueo.

Aplicación de parametrización del controlador de puerta

Para el caso del controlador de puerta se procedió, en un inicio, a hacer un levantamiento de los requisitos de usuario.

Requisitos funcionales:
1. La aplicación debe conectarse con el controlador, leer la configuración existente y representar la información.
2. El usuario debe tener la posibilidad de agregar y modificar los códigos, así como la política de acceso.
3. La aplicación debe actualizar el controlador cuando se solicite, para de esta forma descargarle todos los posibles cambios en la configuración realizados por el usuario.

Requisitos no funcionales:
1. Uso de idioma español en todo el proyecto.
2. Utilizar un formato y estilo uniforme en todas las ventanas de la aplicación.

A partir del levantamiento de los requerimientos de usuario se definen, entonces, los casos de uso, cuyo diagrama se muestra en la Figura 11. El actor del sistema es el administrador, que es el único operador del sistema capaz de modificar y monitorizar la configuración de los controladores.
A partir de los casos de uso se pasa entonces a la etapa de análisis donde se definen las clases que formarán parte de la arquitectura como tal del software y las relaciones entre sí. Para el diseño de la aplicación se usó una estructura de dos capas, compuesta por la capa de presentación, que representa la interfaz con el usuario, y la capa de negocios donde se realiza la lógica de la aplicación así como el acceso a datos.

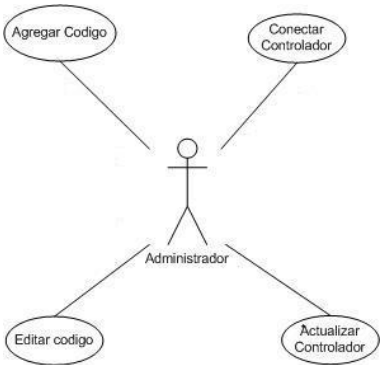

Figura 11. Diagrama de casos de uso de la aplicación de parametrización del controlador de puerta.

La Figura 12 muestra un diagrama de clases de la aplicación. La capa de negocios está formada por una sola clase: TController, que encapsula todas las operaciones a realizar con los controladores. El acceso a datos se realiza usando la librería de código abierto FieldTalk, que posibilita un manejo sencillo del protocolo Modbus sin tener que dominar muchos detalles del mismo. La capa de vista, o presentación, está estrechamente relacionada con el framework de Qt, el cual brinda las clases necesarias para el desarrollo de la interfaz de usuario, o GUI de la aplicación de software. La interfaz está compuesta por una ventana principal, que hereda de la clase QMainWindow de la librería de Qt. Así mismo se crean dos ventanas de forma dinámica del tipo QDialog, para los casos en que el operador desee editar los códigos ó establecer los parámetros de la conexión con el controlador respectivamente.[6-7]

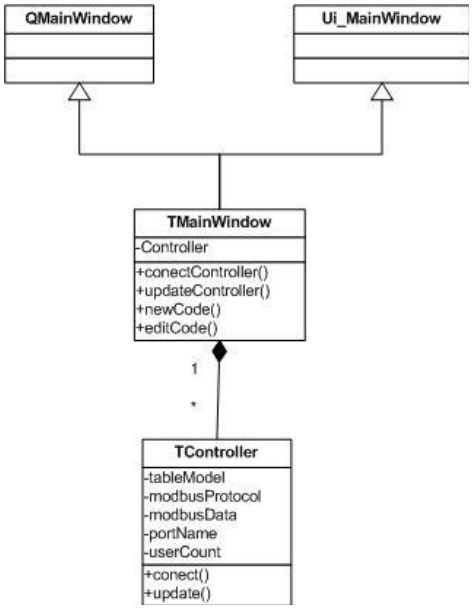

Figura 12. Diagrama de clases de la aplicación de parametrización del controlador de puerta.

Cabe destacar que la clase TController, que es desde donde se produce el acceso a los datos a través de la librería FieldTalk, tiene un miembro que constituye el modelo de los datos, dígase códigos y política de acceso, que se muestran en la vista. Aquí se ha utilizado un patrón de diseño muy difundido en el desarrollo de software llamado modelo vista controlador (M-V-C), donde se separa el procesamiento de los datos de su presentación facilitando así la reusabilidad y el mantenimiento. El framework Qt brinda una implementación más fácil de este patrón, de modo que se crea un modelo de los datos creando un objeto de tipo QStandardItemModel y se le asocia un objeto visual de tipo QTableView a través del cual se presenta y modifica la información

del modelo. De esta forma, en la capa de negocios se hace el procesamiento de los datos y la capa de presentación funge como interfaz con el usuario.[7-8]

Una vez que se tiene el diseño de clases, se hace un análisis más profundo de los casos de uso, pero desde una perspectiva diferente, ya que se analiza la secuencia de interacciones cronológicas entre objetos individuales del sistema para que pueda ejecutarse completamente cada caso de uso.[9]

La Figura 13 muestra la secuencia de eventos para el caso de uso Conectar Controlador. En este caso, en la ventana principal de la aplicación, objeto MainWindow de la clase TMainWindow, se brinda la posibilidad a través de un botón de que el usuario ordene la conexión con el controlador. Inmediatamente se muestra una ventana de diálogo donde se da la posibilidad de especificar los parámetros de la conexión, dígase puerto de la PC, velocidad, etc. Esto último se hace creando un objeto de tipo ConectionDialog heredada de QDialog que presenta los campos asociados a los parámetros mencionados y devuelve sus valores una vez que el usuario acepte en la ventana de diálogo.

Una vez que el objeto MainWindow recibe la información, entonces se crea un objeto de la clase TController y ejecuta el método openProtocol de dicho objeto para abrir la conexión con los parámetros especificados. A continuación, el objeto Controller ejecuta el comando readHoldingRegister perteneciente al protocolo Modbus, a través del cual se lee la configuración actual del controlador conectado. Con esta información, entonces, se procede a actualizar el modelo de los datos, con lo cual se envía una señal o mensaje para la actualización automática de la vista por el mecanismo de Qt para el patrón M-V-C antes mencionado.[10]

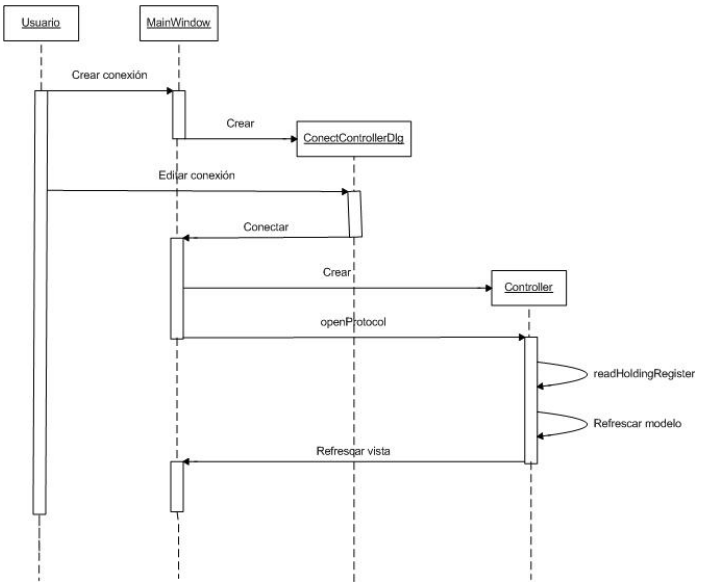

Figura 13. Diagrama de secuencias para el caso de uso Conectar Controlador.

El caso de uso Agregar Código, cuya secuencia se ilustra en la Figura 14, se inicia una vez que el usuario presione el botón correspondiente de la ventana principal. Cuando el objeto MainWindow recibe la señal de este evento crea un objeto de la clase EditDialog heredada de QDialog donde se brindan los campos necesarios para la edición del nuevo código a agregar. Una vez que el usuario termine la edición, entonces se actualiza el modelo de los datos perteneciente al objeto Controller y éste, automáticamente, lanza una señal para la actualización de los mismos en la vista.[7]

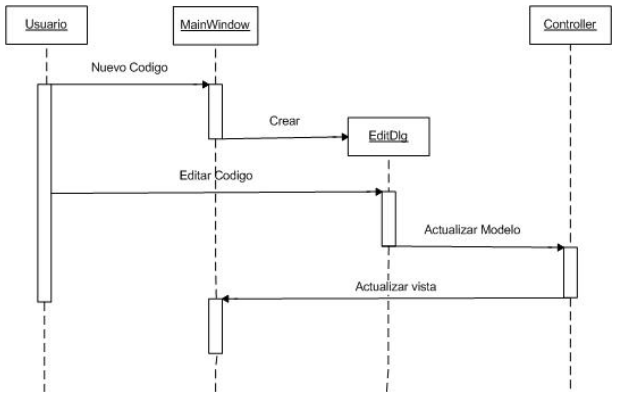

Figura 14. Diagrama de secuencias para el caso de uso Agregar Código.

La secuencia para el caso de uso Editar Código es similar a la de Agregar Código, la diferencia estriba en un ligero cambio en el modo de actualizar los datos en el modelo, pues aquí no se agrega un nuevo registro sino que se determina cual es el registro que se quiere cambiar, y se procede a su modificación.
Por último, la Figura 15 muestra la secuencia de eventos para el caso de uso Actualizar Controlador. Cuando el usuario lo especifique mediante botón en la ventana principal, el objeto MainWindow instancia la función actualizar del objeto Controller. Esta función ejecuta el comando writeMultipleRegisters correspondiente al protocolo Modbus, el cual escribe la configuración de los datos presentes en el modelo en la memoria de configuración del controlador conectado.

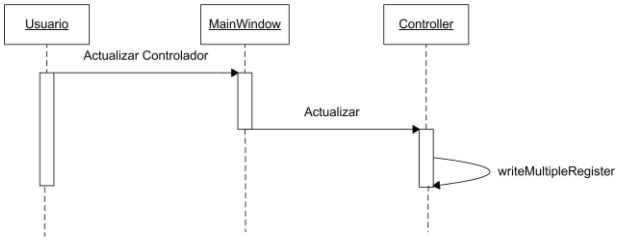

Figura 15. Diagrama de secuencias para el caso de uso Actualizar Controlador.

Aplicación de parametrización del controlador de interbloqueo

En este caso el producto final es similar al del controlador de puerta, y la diferencia estriba en la presentación de la información. Es decir, las ventanas son las mismas excepto que los elementos que las conforman cambian. De esta forma, la ventana principal, en lugar de presentar una tabla para representar los códigos del controlador de puerta, en este caso lo que se tiene es un árbol, donde en cada entrada se muestran las salidas que dependen de ella. Igualmente, esta información está basada en un modelo que forma parte de una clase TController perteneciente a la capa de negocio. De la misma forma, la ventana de diálogo para la edición de la configuración cambia su composición.
Los requisitos de usuario son similares exceptuando el requisito funcional dos del controlador de puerta, pues en el caso del controlador de interbloqueo no se agregan ni se editan códigos, simplemente se relacionan salidas con las entradas. Por tanto el diagrama de casos de uso tiene un ligero cambio al eliminarse los casos de uso agregar y editar código, e incorporar un nuevo caso de uso llamado Editar Configuración, lo cual se ilustra en la Figura 16. La secuencia de eventos para este caso de uso es similar al de agregar y editar código en el controlador de puerta, solo que la ventana de edición es diferente.

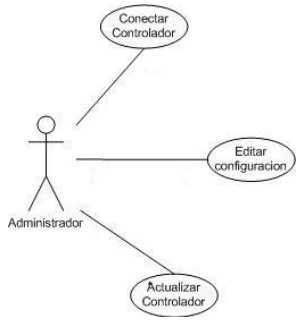

Figura 16. Diagrama de casos de uso de la aplicación de parametrización del controlador de interbloqueo.

RESULTADOS

Después de la correspondiente simulación de los controladores utilizando el software Proteus, se procedió al montaje del hardware de los mismos en PICSCHOOL, que como se dijo anteriormente, consiste en una especie de protoboard donde se pueden probar los prototipos de aplicaciones basadas en microcontroladores PIC. El resultado esperado para el controlador de puerta, era que éste recibiera correctamente el código Wiegand enviado desde un lector de proximidad perteneciente al sistema Kantech. También debía comunicarse correctamente con la aplicación de parametrización utilizando protocolo Modbus y, según la configuración establecida a través de esta aplicación, proveer el acceso a los códigos programados. Así mismo, en el caso del controlador de interbloqueo lo que se perseguía era que respondiera correctamente en la interacción entre las entradas y las salidas, según previa configuración realizada a través de la aplicación de configuración.

Las pruebas se fueron realizando de forma gradual, o por pasos. En el caso del controlador de puerta, primero se comprobó la comunicación Modbus a través de una aplicación de terceros, que se comporta como maestro, ejecutándose en la PC conectada al kit a través del puerto serie RS232. Con este software se pudo comprobar que las respuestas del controlador a los comandos que se le enviaban eran correctas. Seguidamente se procedió a la utilización de la aplicación de parametrización desarrollada y se conectó un lector de proximidad del sistema Kantech, que transmite usando protocolo Wiegand. Se pudo comprobar que los códigos configurados y programados en el controlador a través de la aplicación de parametrización, eran aceptados por el controlador como correctos inmediatamente después de ser ingresados a través del lector, comportándose así correctamente en su papel como controlador de puerta.

De la misma forma, para el controlador de interbloqueo se utilizó el software de terceros para comprobar la comunicación Modbus. Se acoplaron las entradas a los conmutadores presentes en el PICSCHOOL y las salidas a los LEDs. De esta forma se pudo verificar que las salidas se activaban con la activación de las entradas respetando la configuración programada a través del software que se desempeñaba como maestro Modbus. Luego, continuando con las pruebas, se sustituyó el software de terceros por la aplicación de parametrización desarrollada, y se comportó igualmente. Por tanto, se puede concluir que el desarrollo de los controladores a escala piloto se llevó a cabo satisfactoriamente y desempeñando su labor correctamente, como se esperaba.

En cuanto a la utilización de los microcontroladores, en el desarrollo del controlador de puerta se usaron 19 líneas de entrada/salida de las 32 disponibles en el PIC16F877, pero las restantes pueden ser utilizadas en futuras expansiones o adaptaciones si se quiere conectar los controladores en red, por ejemplo. Se usó el canal I^2C, así como el USART y tres fuentes de interrupción. El firmware ocupó el 57% de la memoria ROM o de programa y utiliza el 65% de la RAM, o memoria de datos. En el caso del controlador de interbloqueo, que posee un microcontrolador similar, igualmente se utilizó una parte del total de entrada/salida disponibles, en este caso 18. De la misma forma las líneas restantes pueden ser requeridas para una expansión por necesidad de encuestar un mayor número de entradas, o accionar mayor cantidad de salidas, por citar dos ejemplos. También se usó el canal USART y solo una fuente de interrupción. El firmware ocupó el 20% de la ROM y utiliza un 39% de la RAM.[4]

CONCLUSIONES

En el trabajo se ha expuesto la concepción y desarrollo de un sistema de control de acceso e interbloqueo, lo cual incluyó la puesta a punto, tanto de los controladores como de la aplicación de parametrización. Estos elementos fueron testeados arrojando resultados satisfactorios. El sistema traerá diversas ventajas y beneficios en su implementación en el CIM, como son un mayor soporte técnico y una mayor personalización. El desarrollo de la aplicación de parametrización se hizo teniendo en cuenta una

estructura por capas, así como la utilización del patrón de diseño M-V-C, garantizando un mayor mantenimiento y reutilización del código, elementos claves en una buena ingeniería de software. En el desarrollo de los controladores, igualmente, se tuvieron en cuenta diversos factores para un mayor rendimiento y compatibilidad del sistema, como son:

- Empleo de protocolo Wiegand en la comunicación con los lectores.
- Empleo de protocolo Modbus en la comunicación con la aplicación de parametrización que corre en la PC.
- Empleo de memoria externa para almacenar la configuración en el caso del controlador de puertas, lo cual facilita el mantenimiento o sustitución de los mismos al posibilitar la importación/exportación de los datos.

El sistema obtenido funciona de forma autónoma. No obstante, la forma en la que fue diseñado brinda posibilidades de expansión sin mucha complicación, para integrarlo en un sistema de supervisión o SCADA con la implementación de una red Modbus.

REFERENCIAS

1. Pedreira, M. "Supervisión del acceso a locales del Centro de Inmunología Molecular". Tesis de diploma. CUJAE, 2008.
2. Cosentino, L. "Control de Accesos, Elementos de Identificación". *RNDS*. 2008, 168 p.
3. Pefhany, S. *Modbus Protocol*. 2000.
4. Microchip Technology Inc. *PIC16F87XA Data Sheet*. 1998.
5. Breijo, E.G. *Compilador C CCS y Simulador Proteus para Microcontroladores Pic.* México: Marcombo, 2008. 276 p. ISBN: 978-970-15-1397-2.
6. Eeles, P. *Layering Strategies*. Cupertino: Rational Software, 2002. 13 p.
7. Thelin, J. *Foundations of Qt Development*. New York: Apress, 2007. 528 p. ISBN: 978-1-59059-831-3
8. Gamma, E. , Helm, R., Johnson, R., Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*. 1997. 431 p.
9. Schmuller, J. *Aprendiendo UML 24 horas.* 2004. 398 p.
10. FieldTalk Inc. *FieldTalk Modbus Master C ++ Library Software manual.* 2009.

Autores

Marcel Pedreira Marcel, Ingeniero en Automática, Centro de Inmunología Molecular, La Habana, Cuba, marcel@cim.sld.cu.

Valery Moreno Vega, Ingeniero en Máquinas Computadoras, Máster en Informática Aplicada, Doctor en Ciencias Técnicas, Profesor Titular, Instituto Superior Politécnico José Antonio Echeverría, La Habana, Cuba, valery@electrica.cujae.edu.cu.