Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Electrónica

Instituto Costarricense de Electricidad y Telecomunicaciones

ICE

Desarrollo de una interfaz hacia el protocolo IEC 870-5 para una unidad terminal remota de un sistema SCADA

Informe de Proyecto de Graduación para optar por el Grado de Bachiller en Ingeniería en Electrónica

Óscar Mauricio Vargas Fallas

Cartago - julio del 2002

RESUMEN

En este documento se explica el diseño y los detalles de implementación de un dispositivo traductor de protocolos para un sistema SCADA; específicamente, para hacer posible la conexión entre una unidad terminal remota (RTU o UTR) que utiliza el protocolo Modbus con un sistema SCADA basado en el protocolo IEC 870-5-101.

El traductor se implementó en una tarjeta de desarrollo basada en un microcontrolador Motorola HC12 e incluye el software necesario para interrogar a la estación remota y reportar el estado de los dispositivos de ésta y los cambios en los mismos a la estación maestra; así como ejecutar en la remota acciones requeridas por la maestra.

El prototipo fue construido y probado con éxito, sin embargo cabe la posibilidad de hacerle mejoras para optimizar su funcionalidad y rendimiento.

Palabras clave: SCADA, Modbus, IEC 870-5, traducción de protocolo, conversión de protocolo.

ABSTRACT

This document explains the design and the details of implementation of a protocol converter device for a SCADA system. Specifically, it was designed to make possible the connection between a remote terminal unit (RTU) which uses Modbus protocol to an IEC 870-5-101 based SCADA system

This converter device was developed on an evaluation board based on the Motorola HC12 microcontroller. It includes the software required to poll the remote station and report the state and values of the devices located on the remote, and the changes on then to the master station; it also executes required commands from the master station on the remote station

A prototype was built and successfully tested. However, improvements can be made to optimize its functionality and performance.

Keywords: SCADA, Modbus, IEC 870-5, protocol translation, protocol conversion.

DEDICATORIA

Al Padre Celestial a quien le debo todo lo que soy.

A mi padre y a mi madre, por haberme dado la vida, por haberme dado su amor, por haber hecho posible que llegara hasta aquí; nunca podré corresponderles tanto.

A mi hermana Kattia, por ser mi mejor amiga, por los tantos, más buenos que malos, momentos que hemos compartido.

A todos aquellos, que con sus obras y hasta con su sangre han trabajado y luchado por hacer de este mundo un mejor lugar.

AGRADECIMIENTOS

Agradezco en primer lugar a Dios por haberme permitido llegar hasta aquí.

A mis padres, pues gracias a su esfuerzo fueron posibles mis estudios.

Agradezco al Ing. Luis Moya por haberme abierto las puertas del Centro de Servicios de Investigación y Desarrollo y darme la posibilidad de realizar ahí el presente Proyecto de Graduación.

Agradezco también la valiosa ayuda de los Ingenieros Fernando Lizana, Enrique Gamboa, Alexánder Mora, y Antonio Hasbun pues sin ella no habría podido concluir satisfactoriamente este proyecto.

También a la Hermana Blanca y a todos aquellos que con sus oraciones y palabras de aliento me dieron su apoyo.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1:
INTRODUCCIÓN

1.1 Descripción de la Empresa

1.1.1 Descripción general

El Instituto Costarricense de Electricidad fue creado como una institución autónoma, bajo el decreto nº 449, el 8 de abril de 1949, promulgado por la Junta Fundadora de la Segunda República, con el objetivo de crear empresa nacional que se encargara de la generación y transmisión de electricidad en Costa Rica, que pudiera llevar el desarrollo eléctrico a todos el país, ya que las compañías eléctricas instaladas en el país en ese entonces estaban en manos privadas; principalmente extranjeras.

Actualmente, el ICE, junto con sus empresas subsidiarias (RACSA y la Compañía Nacional de Fuerza y Luz) proveen de energía eléctrica a la mayor parte del territorio nacional y poseen el monopolio de las telecomunicaciones en el país. Gracias a sus esfuerzos, el país cuenta con una cobertura del 96.8% del territorio nacional en cuanto al servicio de energía eléctrica y un 94% en lo que respecta a telefonía básica.

El Consejo Directivo del ICE está conformado por siete miembros nombrados por el Poder Ejecutivo, incluyendo al presidente ejecutivo; según la Ley de Presidencias Ejecutivas. A su vez, el Consejo Directivo nombra al Gerente General. La Gerencia General se divide en tres subgerencias: Electricidad, Gestión Administrativa y Telecomunicaciones. Éstas a su vez, se dividen en Áreas y Unidades Estratégicas de Negocios (UENs)

1.1.2 Descripción del departamento

El presente proyecto fue realizado en la UEN de Proyectos y Servicios Asociados del sector Energía, en el Centro de Servicio de Investigación y Desarrollo (I+D).

Este centro de servicio tiene como objetivo promover y desarrollar proyectos de innovación tecnológica que contribuyan a mejorar la confiabilidad, calidad y costo del servicio eléctrico brindado por ICE Electricidad. Investigación y Desarrollo tiene varios proyectos, entre ellos el proyecto SCADA, del cual, el presente proyecto forma parte. El objetivo del proyecto SCADA es proveer un sistema moderno de manejo de información y control para los Centros Locales de Operaciones de Redes de distribución eléctrica (CLOR) Otros proyectos que se manejan en el departamento son: Simulador de Sistemas de Potencia, Proyecto SIAHC y Detección de Descargas Atmosféricas entre otros.

En este departamento laboran 39 personas, en su mayoría ingenieros (eléctricos, electrónicos, químicos, mecánicos, industriales e informáticos) También hay profesionales en química y técnicos en electrónica. De estos 39 empleados, 25 trabajan en el edificio principal de la UEN en Sabana Norte, el resto lo hace en el plantel Colima. La jefatura del departamento está a cargo del Ing. Ronald Jiménez.

El Centro de Servicio de Investigación y desarrollo cuenta con varios laboratorios:

a. Laboratorio de Electrónica y Circuitos Impresos

b. Laboratorio de Corrosión de Materiales

c. Laboratorio de Simulación de Sistemas de Potencia

Además, hay áreas de interés en las que en el futuro se podrían ejecutar proyectos como lo son: aprovechamiento del recurso eólico y generación de electricidad con celdas de combustible (Hidrógeno)

1.2 Definición del problema y su importancia

El ICE, sector energía, cuenta con sistemas SCADA (Sistema de control supervisorio y adquisición de datos) para monitorear y controlar los procesos de generación, transmisión y distribución eléctrica. En la figura 1.1 se muestra un esquema de un sistema SCADA. Como se puede observar, hay una estación central que controla y recibe los datos de las Unidades Terminales Remotas (UTR), las cuales pueden estar a grandes distancias de la estación central, enlazadas a ésta por medios alámbricos o inalámbricos. Las UTR reciben y envían señales tanto analógicas como digitales de y hacia el proceso físico involucrado. Cada UTR cuenta con su CPU, memoria y dispositivos necesarios para la comunicación, entrada y salida de datos.

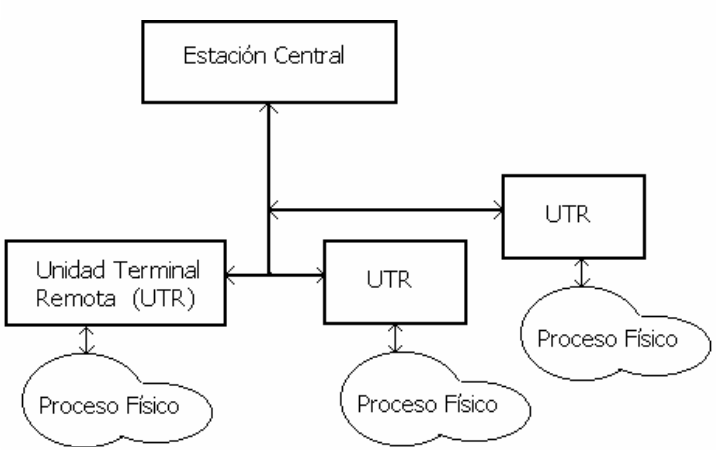


Figura 1.1 Esquema de un sistema SCADA

El ICE está realizando un proceso de modernización de las UTR de sus sistemas SCADA, pues muchas de ellas son ya obsoletas y serán sustituidas por otras, más modernas y aptas para solventar las necesidades de la institución.

El Centro de Servicio de Investigación y Desarrollo está trabajando en el diseño de una UTR que pueda utilizarse para este proceso de reemplazo, a un costo mucho menor que el que resulta de adquirirlas externamente.

El Centro Nacional de Control de Energía opera varios sistemas SCADA, uno de ellos está dedicado al control y monitoreo de las plantas de generación eléctrica, en el cual también se sustituirán estaciones remotas, y se desea integrar a este sistema las UTR desarrolladas por I+D Sin embargo, las nuevas UTR en desarrollo utilizan el protocolo Modbus de Modicon, por lo que no son compatibles con el protocolo IEC 870-5 que es el utilizado por el Centro Nacional de Control de Energía, por lo que las nuevas terminales remotas no podrán utilizarse para este sistema.

Si las nuevas UTR no se ponen en funcionamiento en este sistema SCADA, habrá que dejar las que están actualmente, que ya cumplieron con su vida útil, o habría que adquirir en el mercado unidades terminales remotas diseñadas para trabajar con este protocolo, las cuales tienen un precio muy elevado.

Además, se busca aprovechar los recursos y tecnologías existentes en la empresa, y adaptarlos a las necesidades de la misma.

Así pues, surge la necesidad de diseñar e implementar un dispositivo intérprete para las UTR que se están desarrollando, que sea capaz de recibir las órdenes de la estación central, procesarlas, darle las instrucciones correspondientes a la UTR en un protocolo que ésta pueda entender, recibir de ésta los datos que sean necesarios y enviarlos a la estación maestra en el formato exigido por el protocolo IEC.

De esta forma, se proveería conectividad a las nuevas UTR desarrolladas por el ICE con el protocolo utilizado en el Centro Nacional de Control de Energía, que es además, más poderoso, robusto y confiable que otros protocolos utilizados en sistemas SCADA. Así pues, la solución a este problema contribuiría a mejorar la calidad del servicio eléctrico en las zonas que reciben energía eléctrica generada en plantas eléctricas del ICE, al mejorar los tiempos de respuesta ante averías, y detectarlas antes de que éstas surtan efectos contraproducentes para la institución y sus clientes; todo esto aprovechando los recursos que están disponibles en la institución.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar e implementar un dispositivo capaz de proveer conectividad a las unidades terminales remotas desarrolladas por el departamento de Investigación y Desarrollo, al protocolo IEC 870-5-101, utilizado por el sistema SCADA “Ranger” del Centro Nacional de Control de Energía.

1.3.2 Objetivos específicos

1. Investigar las funciones del protocolo IEC 870-5-101 y determinar cuáles se van a implementar.
2. Investigar las funciones del protocolo Modbus y determinar cuáles se van a implementar.
3. Identificar las características que debe tener el hardware que se utilice para desarrollar la interfaz, basándose en las funciones a implementar.
4. Seleccionar el hardware que mejor cumpla con esas características.
5. Desarrollar un programa de prueba para el microcontrolador.
6. Escribir las rutinas necesarias para que el microcontrolador pueda enviar mensajes a la UTR en protocolo Modbus
7. Escribir las rutinas necesarias para que el microcontrolador pueda leer e interpretar los mensajes de respuesta de la UTR
8. Escribir las rutinas necesarias del microcontrolador, para seleccionar por medio de la PC qué instrucciones se le envían a la UTR y poder desplegar en pantalla los datos enviados por la UTR.
9. Por medio de las rutinas desarrolladas en el objetivo 8, comprobar el funcionamiento de las rutinas de los objetivos 6 y 7.

10. Desarrollar las rutinas para que el microcontrolador identifique e interprete los mensajes provenientes de la estación maestra, codificados en el protocolo IEC 870-5
11. Desarrollar las rutinas necesarias para que la interfaz construya los mensajes de respuesta a la estación maestra.
12. Escribir en Delphi un programa para que una PC simule el comportamiento de la estación maestra, le envíe mensajes al microcontrolador en el protocolo IEC 870-5-101 y despliegue en pantalla las respuestas de éste.
13. Con ayuda del programa desarrollado en el objetivo 12, comprobar el funcionamiento de las rutinas de los objetivos 10 y 11
14. Hacer el diseño del software necesario para integrar las rutinas desarrolladas hasta ahora para que se enlacen en una sola unidad funcional.
15. Evaluar el funcionamiento del prototipo con una UTR Modbus y el programa desarrollado en el objetivo 12.
16. Escribir la documentación donde se explica el diseño del sistema, funciones implementadas, diagramas y otra información de interés para las personas que vayan a trabajar con este dispositivo en el futuro.

CAPÍTULO 2:
ANTECEDENTES

2.1 Estudio del problema a resolver

2.1.1 Modelo de Interconexión

Para el análisis de este problema y su solución, se utilizará el modelo EPA de tres capas, que es una simplificación del modelo OSI de siete capas. En la figura 2.1 se muestran ambos.

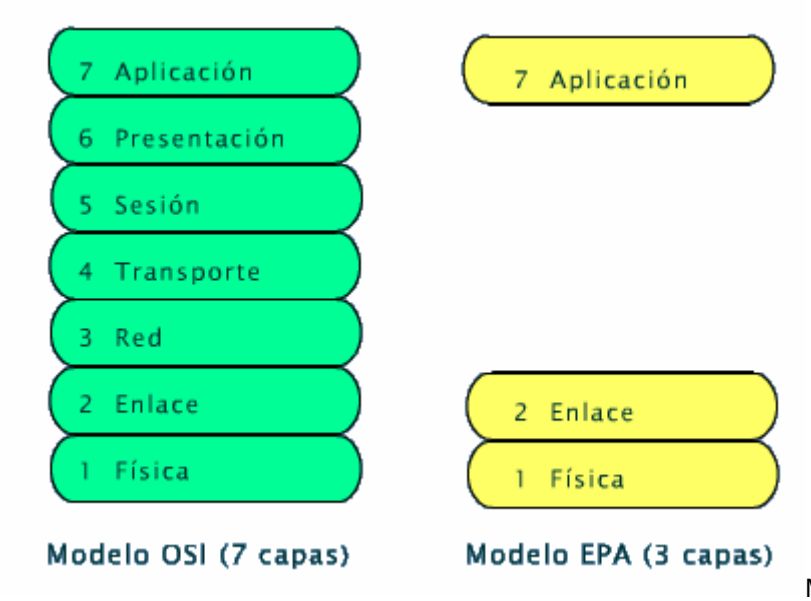


Figura 2.1 Modelos de Interconexión OSI y EPA

Así pues, de ahora en adelante el problema se analizará en tres niveles: capa 1 (física) capa 2 (enlace) y capa 7 (aplicación)

2.1.2 Protocolo Modbus

El protocolo Modbus se rige por el modelo maestro / esclavo, en el cual toda comunicación es iniciada por el dispositivo maestro y el esclavo se limita a responder si y sólo si la orden dada por el maestro así lo indica.

Entre las instrucciones de este protocolo se encuentran: lectura de estado de señales digitales, lectura de estado de señales analógicas, escritura de una señal digital, escritura de varias señales digitales, escritura de una señal analógica y escritura de varias señales analógicas. Además incluye instrucciones de servicio, como lo son: lectura de señales de excepción, reinicio de enlace, etc.

El protocolo Modbus utiliza el estándar RS-232 para la transferencia de datos. Existen dos modos de transmisión: UTR (binario) y ASCII. Para este caso se utiliza el modo UTR. Los dispositivos están mapeados en una memoria de 8 Kb, con un espacio definido para las salidas digitales (llamadas solenoides o bobinas), otro para las entradas analógicas, otro para las entradas digitales y otro para las analógicas.

La adquisición de datos se lleva a cabo mediante un sondeo realizado por la estación maestra: ésta le envía un mensaje a la remota, en el cual se indica la dirección de la entrada (o salida) de la cual se desea saber su valor o estado. La UTR recibe el mensaje, lo decodifica y envía el mensaje de respuesta, el cual contiene la información solicitada por la estación maestra.

De igual forma, para enviar datos de la maestra hacia la esclava, la primera envía un mensaje que indica la acción a realizar y la dirección del dispositivo involucrado. La segunda recibe el mensaje y después de haber ejecutado la orden, envía a la primera un mensaje de confirmación.

Para una descripción detallada del protocolo Modbus, véase el apéndice A1.

2.1.3 Protocolo IEC 870-5-101

El protocolo IEC 870-5 es en realidad una serie de especificaciones para transmisión de datos en sistemas eléctricos. Para esta aplicación en particular se utilizará la norma IEC 870-5-101 que corresponde a UTR controladas remotamente.

Este protocolo cuenta con dos modos: balanceado y desbalanceado. El modo balanceado permite a la remota enviar mensajes en cualquier momento. El modo desbalanceado sigue el esquema maestro / esclavo, donde éste último transmite datos solamente cuando el maestro los solicita. El sistema *Ranger* del Centro de Control de Energía utiliza el modo desbalanceado.

A diferencia del protocolo Modbus, los procedimientos de recolección no son encargados exclusivamente a la estación maestra, sino que la UTR tiene la responsabilidad de definir cuándo y por qué se le reportarán datos. La estación maestra, llamada también primaria, envía un mensaje a la secundaria (esclava) en el cual simplemente solicita datos, ya sean clase 1 o de alta prioridad, como también clase 2 o baja prioridad. La estación remota entonces envía un mensaje con los datos que esta última considere más relevantes. Si todavía resta información importante, la remota lo indica en el mensaje, y esperará hasta que la primaria vuelva a hacer un sondeo para enviarla,

La información en este protocolo se transporta en unidades llamadas mensajes o telegramas. El tamaño de cada uno está dado por su tipo; los hay cortos, de un octeto y largos, de hasta 255 bytes de longitud. En estos últimos puede incluirse un ASDU, que es una unidad donde se empaquetan datos correspondientes a uno o varios puntos o dispositivos en particular de la UTR.

Para una descripción más detallada de este protocolo, véase el apéndice A2.

2.1.4 UTR Modbus

La UTR que se está desarrollando en el Centro de Servicio, a cargo del Ing. Alberto Zamora, está basada en un microcontrolador Basic Stamp 2X. Para la comunicación mediante el protocolo Modbus, cuenta con un módulo, también basado en un Basic Stamp 2X, que interactúa con el CPU principal para obtener acceso a la memoria y así poder leer o escribir datos en esta para enviarlos al maestro o efectuar instrucciones de éste.

Este módulo fue implementado por el Ing. Fernando Lizana para su proyecto de graduación del ITCR, en el primer semestre del 2001. Tiene la particularidad de que no se le implementaron todas las funciones del protocolo, sino solamente las básicas, hecho que se tomó en cuenta para la ejecución de este proyecto.

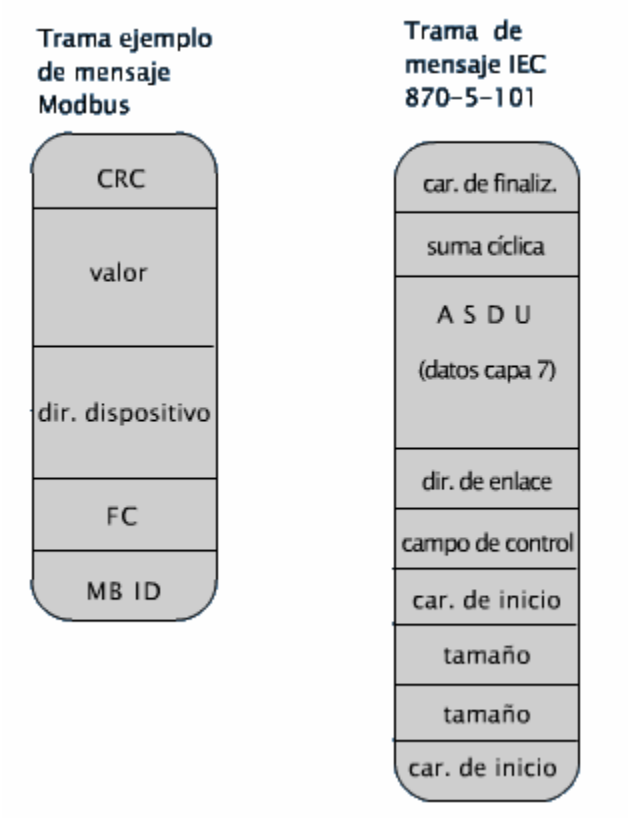
2.1.5 Capa Física

Como ya se vio, el protocolo Modbus utiliza el estándar RS-232 para hacer el enlace en capa 1. Las especificaciones del equipo utilizado en el Centro de Control de Energía con el protocolo IEC 870-5-101 definen también para la capa física una interfaz equivalente a RS-232, por lo que no existen problemas de interconexión en capa 1.

2.1.6 Capa de Enlace

Los procedimientos de comunicación de los protocolos Modbus e IEC 870-5-101 (modo no balanceado) presentan algunas semejanzas como también diferencias. Ambos siguen la modalidad maestro-esclavo, sin embargo existen algunas funciones de capa 2 en el protocolo IEC en las que la estación remota no responde, mientras que en Modbus la UTR siempre responde a los mensajes de la estación maestra. Además, el protocolo Modbus sólo cuenta con un juego de funciones de capa 7, mientras que IEC tiene tanto funciones de capa 2 como de capa 7. En el apéndice A2 se muestran las tablas A2.1 y A2.2 con las funciones de capa 2 para el protocolo IEC.

Las tramas de ambos protocolos sí presentan diferencias considerables. En la figura 2.2 se muestra un ejemplo de la trama de un mensaje de cada protocolo. Para una descripción más detallada de cómo se construyen ambas tramas, véanse los apéndices 1 y 2.



MACROMEDIA FLASH

Figura 2.2 Ejemplos de tramas en ambos protocolos

2.1.7 Capa de Aplicación: Direccionamiento de dispositivos

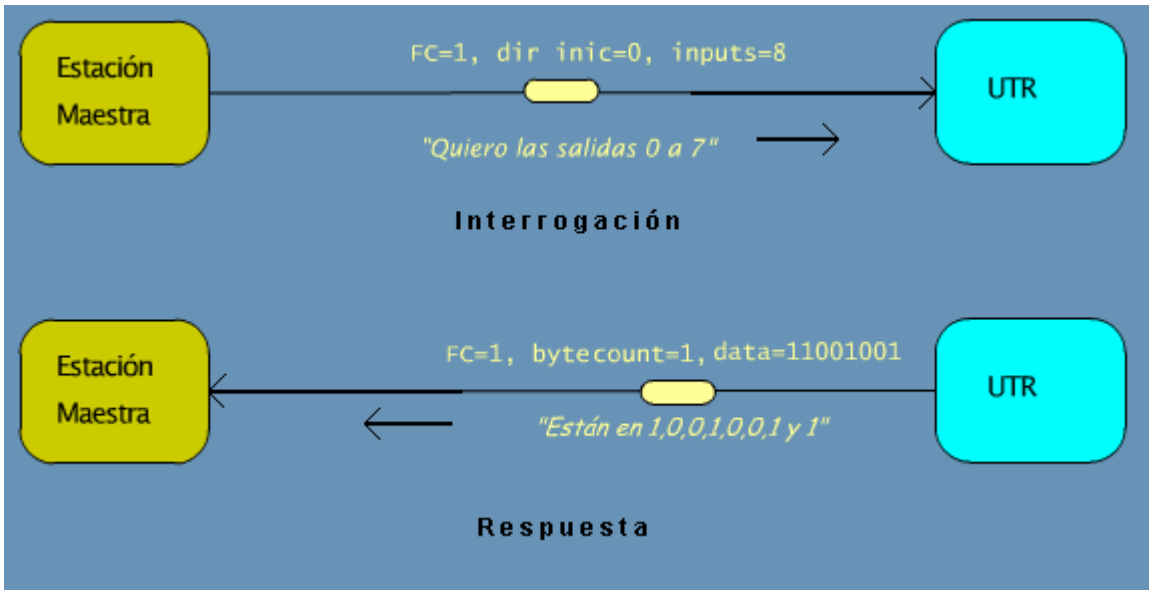
En Modbus, los dispositivos de entrada / salida, se ubican en 4 bloques: en el primero se tienen los tipo *coil* (bobinas o solenoides, utilizados para salidas o entradas digitales), los tipo *input* (entradas, utilizados para dispositivos digitales), los de tipo *holding register* (registros de 16 bits, de lectura y escritura, utilizados principalmente para dispositivos analógicos) y los de tipo *input register* (registros de 16 bits, sólo lectura). Cada tipo cuenta con su propia numeración: de 0 a 2047 para los digitales y de 0 a 255 para los analógicos. Para acceder a ellos, se utilizan funciones Modbus distintas para cada tipo de dispositivo.

En el protocolo IEC, todos los dispositivos se ubican en un mismo bloque, en puntos o direcciones IOA (*Information Object Address)*. El número de puntos que requiere un dispositivo está dado por su tipo: Para los digitales existen los *single point* (punto sencillo) y los *double point* (punto doble) que requieren uno y dos puntos respectivamente. Los analógicos requieren 16 puntos y los valores acumulados (contadores) requieren 24 bits. Todos los puntos de una UTR o de un módulo de ésta tienen en común un parámetro, *common address of ASDU* que es propio de la misma y que junto con la dirección IOA del dispositivo hacen cada punto de la UTR único en el sistema SCADA.

Así pues, se observa que los métodos de direccionamiento son distintos para cada protocolo, por lo que se hace necesario implementar un esquema de redireccionamiento.

2.1.8 Capa de Aplicación: Adquisición de datos

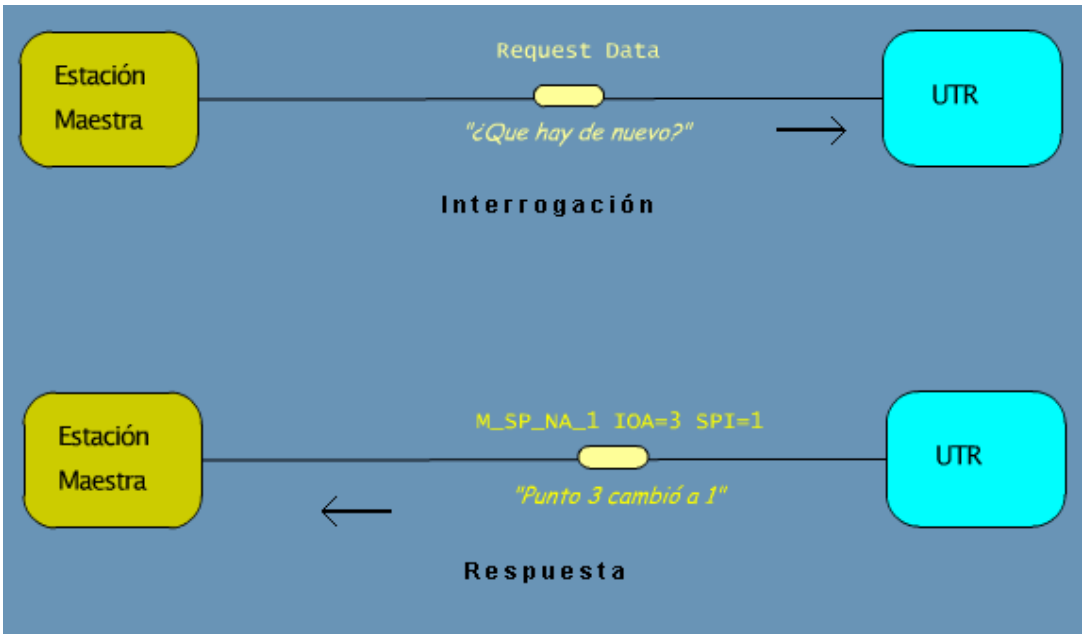
En el protocolo Modbus, la estación maestra debe preguntar por el estado de cada dispositivo para estar al tanto de lo que sucede en el proceso físico. En la figura 2.3 se muestra cómo se da este proceso con ocho dispositivos digitales:



MACROMEDIA FLASH

Figura 2.3 Adquisición de datos en el protocolo Modbus

Con el protocolo IEC, el procedimiento es distinto; la estación maestra no pregunta específicamente por cada dispositivo, sino que hace un *request data* y la UTR responde solamente con los datos que han cambiado, como se muestra en la figura 2.4 Adicionalmente la estación maestra puede hacer una interrogación general, en la que la UTR debe responder con el estado de todos los dispositivos.



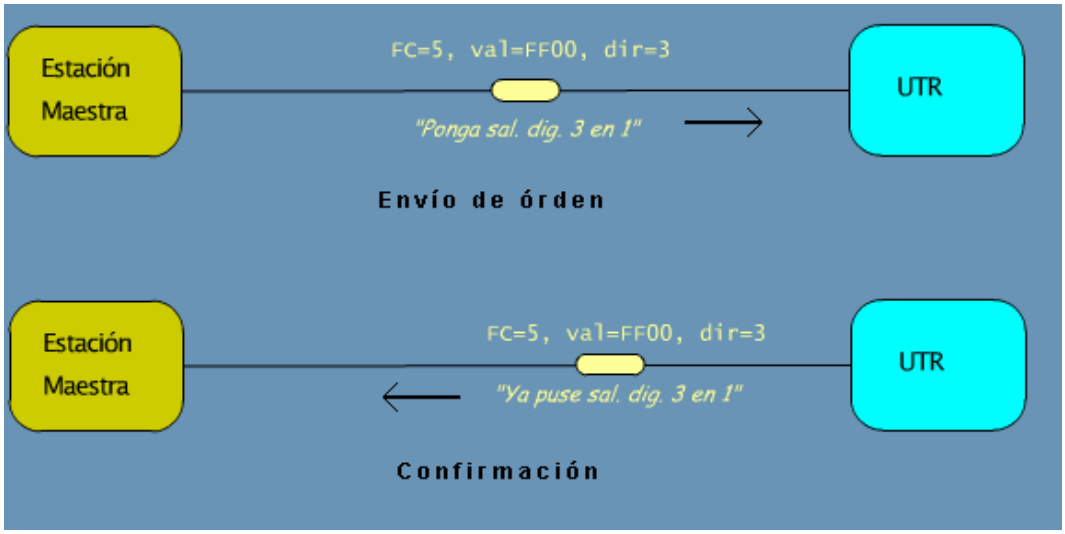
MACROMEDIA FLASH

Figura 2.4 Adquisición de datos con el protocolo IEC 870-5-101

Así pues, se tienen dos protocolos con filosofías distintas para la adquisición de datos, por lo que el dispositivo traductor debe interrogar continuamente a la UTR Modbus para estar al tanto del estado de sus dispositivos de E/S y reportar solamente los cambios a la estación maestra cuando ésta los solicite.

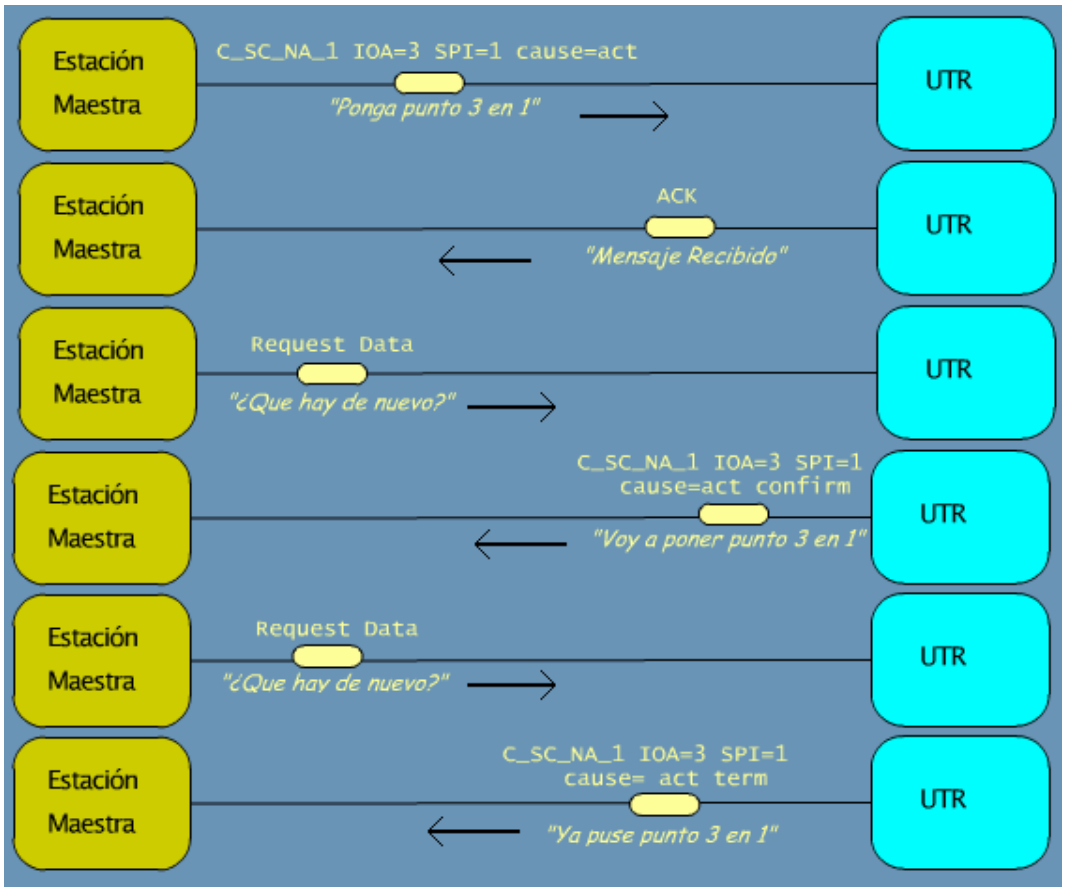
2.1.9 Capa de Aplicación: Ejecución de Órdenes

Aunque en general ambos protocolos tienen filosofías similares para la ejecución de órdenes -la estación maestra envía el mensaje y la remota ejecuta la orden y envía un mensaje de confirmación- los procedimientos presentan diferencias. En Modbus, la estación maestra envía el mensaje con la instrucción y espera el mensaje de confirmación, mientras que con el protocolo IEC la estación maestra envía el mensaje y la remota responde con un "*Acknowledgment*". La remota envía el mensaje de confirmación la próxima vez que la maestra haga un *"request data";* y otro mensaje de terminación de activación en otro *request data.*



MACROMEDIA FLASH

Figura 2.5 Envío de instrucciones con el protocolo Modbus



MACROMEDIA FLASH

Figura 2.6 Envío de instrucciones con el protocolo IEC 870

En las figuras 2.5 y 2.6 se ilustra el procedimiento para la transmisión de instrucciones en los protocolos Modbus e IEC 870-5-101 respectivamente. Obsérvese que el procedimiento con el protocolo IEC tiene una mayor complejidad; además, exige que la UTR guarde los mensajes generados hasta que sean solicitados por la estación maestra, lo que introduce para el dispositivo traductor la necesidad de un espacio de memoria (*buffer*) para almacenar los mensajes de confirmación hasta que sean solicitados por la estación maestra.

2.2 Requerimientos de la empresa

El equipo del proyecto SCADA del Centro de Servicios de Investigación y Desarrollo (I+D) requiere para este proyecto en específico, que se construya un prototipo del sistema encargado de realizar la traducción de protocolos, junto con el software requerido para la configuración y evaluación del mismo.

Este sistema debe ser compatible con la UTR que está desarrollando I+D, con el sistema *Ranger* del Centro de Control de Energía y con los equipos utilizados por ICE Energía para el proceso de transmisión de datos, para lo cual, deben seguirse los estándares del protocolo Modbus para el primer caso (Un conjunto reducido de instrucciones, ya que en la UTR no se implementaron todas), los del protocolo IEC 870-5-101 para el segundo y los del la UIT V.24/V.28 para el tercero.

Además, el diseño debe ser tal que el costo del mismo lo justifique como una alternativa más conveniente que otras opciones, tales como adquirir UTRs compatibles con el protocolo IEC u otras posibles soluciones externas de traducción de protocolo (No se encontraron soluciones de este tipo disponibles en el mercado)

2.3 Solución propuesta

Para este problema en específico, se encontró que la solución óptima consistía en implementar un traductor de protocolo, que actuara como maestro con la UTR mediante el protocolo Modbus y como esclavo ante la estación maestra *Ranger* utilizando el protocolo IEC 870-5-101, como se muestra en la figura 2.7. Así, el conjunto traductor-UTR sería visto por el "*Ranger*" como si fuera solamente una UTR compatible con el protocolo IEC 870.

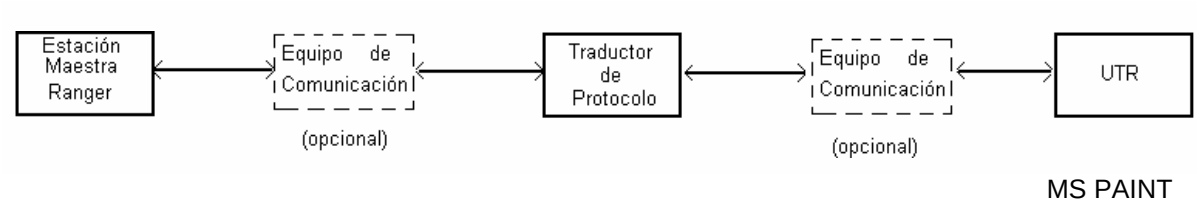


Figura 2.7 Diagrama de Bloques de la solución

Debido a diferencias que se han visto hasta el momento entre ambos protocolos, el dispositivo traductor no se limitaría simplemente a traducir instrucciones, sino que debe ser un dispositivo inteligente con más funciones y características, entre las cuales se pueden citar las siguientes como las más importantes:

a. Generar e interpretar tramas de ambos protocolos
b. Tener un esquema de redireccionamiento de dispositivos
c. Preguntar periódicamente por el estado de cada dispositivo en la UTR, cuando encuentre que alguno ha cambiado, reportarlo a la estación maestra.
d. Ejecutar órdenes de la estación maestra y generar mensaje de confirmación. Mantener un *buffer* para acumular los mensajes de respuesta a la estación maestra hasta que ésta los pida.

En principio, se había planteado una solución distinta al problema: en lugar de implementar un traductor separado de la UTR, se diseñaría un dispositivo periférico del CPU de la UTR, que leería y escribiría los datos directamente en la memoria de ésta; sin embargo, al consultar con otras personas encargadas del desarrollo de la UTR en el Centro de Servicio y analizar las implicaciones de utilizar esta solución, ésta fue descartada y se optó por seguir la solución aquí planteada.

El traductor se encarga de monitorear la línea correspondiente a la estación maestra. Cuando esta le envíe una instrucción, el traductor la recibirá, la decodificará, buscará en su tabla de direcciones a qué dispositivo en la UTR va dirigida y si se requiere, enviará a la UTR la instrucción equivalente en protocolo Modbus. Luego enviará a la estación maestra los mensajes de confirmación correspondientes.

Los valores correspondientes a las entradas (analógicas y digitales) de la UTR son leídos periódicamente, así, si el traductor detecta un cambio en una entrada digital o que una entrada analógica sobrepasó el valor umbral establecido, el evento es reportado a la estación maestra la próxima vez que esta haga un sondeo.

Se definió que el traductor se implementaría mediante un microcontrolador con dos puertos serie, cada uno con su respectiva interfaz RS-232 para la comunicación con la UTR, la estación maestra y el equipo de comunicación. Funciona en dos modos: traductor y configuración. En modo traductor funciona como se ha descrito hasta ahora. En modo configuración, se conectaría por puerto serie con una PC que ejecute un programa emulador de terminal mediante el cual se configuran parámetros del traductor, como dirección IEC, velocidad de transmisión y la tabla de direcciones de los dispositivos de la UTR.

CAPÍTULO 3:
PROCEDIMIENTO METODOLÓGICO

Para materializar la solución propuesta, lo primero que se hizo fue una investigación bibliográfica de los protocolos en cuestión. Para el protocolo IEC 870 se utilizó documentación de equipos que se tenían en el Centro de Control de Energía; sin embargo esta documentación fue insuficiente para tener un dominio completo de este protocolo, por lo que se buscaron otros recursos en Internet, donde se encontró información más detallada. Para el protocolo Modbus se utilizó la información disponible en el sitio web de Modicon (www.modbus.org, www.modicon.com).

Una vez recopilada y estudiada la información mencionada, se elaboró un resumen para cada protocolo. Ambos resúmenes, se incluyen en este informe en los apéndices A1 y A2.

Teniendo ya presentes las características de cada protocolo, se definieron las especificaciones que debía tener el hardware a utilizar para implementar el dispositivo traductor. Se buscaron en Internet las características de los productos ofrecidos por distintos fabricantes y se analizó cuidadosamente la documentación de los que parecían más aptos. Luego de analizar varias propuestas y considerando aspectos técnicos, económicos y logísticos, se escogió la tarjeta ADAPT812DXLT de la empresa canadiense Technological Arts.

Luego se definió el lenguaje y las herramientas de software que se utilizarían para la programación del microcontrolador. Considerando las características de la tarjeta, la documentación y programas de demostración incluidas con la tarjeta, las características y costos de las herramientas de software disponibles, se decidió hacer la programación en lenguaje ensamblador, utilizando el software *Mini-IDE*, desarrollado por MGTEK.

En el momento que se tuvieron disponibles la tarjeta y las herramientas, se procedió a realizar un programa de prueba, utilizando como base el programa de demostración incluido con la tarjeta.

Utilizando este programa como punto de partida, se le fueron agregando progresivamente rutinas que efectuaban labores más complejas: envío y recepción de datos por el puerto serie, utilización del mismo como interfaz de usuario, implementación de menúes, etc.

Así pues, ya se contaba con una plataforma adecuada para comenzar el desarrollo del software concerniente específicamente a este proyecto, por lo que se inició la implementación de envío de datos utilizando el protocolo Modbus. Para probar las rutinas desarrolladas aquí, se utilizó la herramienta *MBSlave* que consiste en una aplicación para *Microsoft Windows* que simula una terminal remota Modbus. Una vez desarrolladas y comprobadas estas rutinas, se siguió con la parte correspondiente a la recepción y decodificación de los mensajes de respuesta provenientes de la UTR. Para poder comprobar el funcionamiento de estas rutinas, se implementó en la memoria de la tarjeta un “espejo” que contenía el estado de los dispositivos de la UTR, que se actualizaba con cada mensaje proveniente de ésta y se le desplegaba al usuario mediante el puerto serie SCI0

Una vez concluida esta parte, se prosiguió con el desarrollo de la parte IEC 870. Para esto, primero se definieron las estructuras de datos para los *buffers*, “espejos“ de dispositivos, arreglos etc. Se definieron variables que debían introducirse y se desarrolló un menú interactivo, donde el “usuario” o administrador introduce parámetros necesarios para la traducción, como número de entradas y salidas, direcciones base, etc.

Una vez concluida la programación de este menú, se integró con las rutinas de envío y recepción de datos en Modbus en un software que hace automáticamente un sondeo de cada dispositivo de la UTR, manteniendo actualizado el espejo de los dispositivos en memoria. Sobre este programa se desarrollaron las rutinas de envío y recepción de mensajes IEC 870 propiamente dichas.

Para evaluar las rutinas de recepción y envío de mensajes en el protocolo IEC 870-5 también se hizo otro programa, pero esta vez para una PC, utilizando un lenguaje de alto nivel, que simulara una Estación Maestra IEC 870-5, le enviara instrucciones al dispositivo traductor y desplegara las respuestas de éste en pantalla. Para esto se utilizó la herramienta de programación *Delphi,* desarrollada por *Borland–Inprise*. Para poder recibir y enviar mensajes a través del puerto serie de la PC se utilizó el componente *nrComm*. Ambas herramientas se encontraban disponibles en el departamento de Investigación y Desarrollo.

Para comprobar que este programa fuera efectivamente una herramienta confiable para evaluar el traductor, se hizo una prueba que consistió en conectar una UTR compatible con el protocolo IEC 870-5-101 como esclava a una PC que estuviera ejecutando el programa.

Una vez comprobado que el dispositivo traductor funcionaba correctamente, mediante el simulador de la estación maestra y el programa *MBSlave,* se procedió a elaborar el manual de usuario del traductor, finalizando con esto la ejecución del proyecto.

CAPÍTULO 4:
DESCRIPCIÓN DEL HARDWARE UTILIZADO

4.1 Descripción general

Para la implementación del traductor, se utilizó la tarjeta de desarrollo ADAPT812DXLT de la empresa canadiense *Technological Arts*. Esta tarjeta está basada en un microcontrolador MC68HC812A4, perteneciente a la familia 68HC12 de Motorola. Aparte de la memoria incluida en el microcontrolador, cuenta con 512 Kb de memoria Flash para programa y 32 Kb de SRAM para datos. También incluye dos puertos serie RS-232 con sus respectivos conectores DB9. La figura 4.1 muestra una fotografía de la misma. A continuación se da más información acerca del microcontrolador y de la tarjeta.

Figura 4.1 Fotografía de la tarjeta Adapt812DXLT

4.2 Microcontrolador Motorola MC68HC812A4

Como ya se indicó, este microprocesador corresponde a la familia HC12 de Motorola, pues el núcleo del mismo lo conforma el CPU12, un procesador de 16 bits utilizado para microcontroladores desde mediados de la década de 1990. Además del núcleo, incluye otros “periféricos” integrados en el chip, como lo son: módulos de comunicación serie, módulo de generación de reloj, módulo de contadores y temporizadores, convertidor A/D de 8 canales. También cuenta internamente con un espacio de memora RAM de 1 Kbyte para datos y un espacio de 4 Kbyte de memoria EEPROM para código.

Este microcontrolador utiliza un empaque LQFP de montaje superficial de 112 pines. Según el modo en que se opere, acepta hasta 4 puertos de 8 bits de entrada / salida para propósito general y puede direccionar hasta 5 Mb de memoria externa.

Típicamente, puede operar en tres modos: sencillo (solamente el chip, sin utilizar memora externa), expandido angosto (utiliza memoria externa mediante un bus de datos de 8 bits) y expandido ancho (utiliza memoria externa mediante un bus de datos de 16 bits)

4.3 Tarjeta Adapt812DXLT

Como ya se mencionó, esta tarjeta tiene como cerebro y elemento principal al microcontrolador 68HC812A4. Mediante un interruptor se puede seleccionar el modo en que éste opera, ya sea sencillo o expandido angosto.

Además del microcontrolador cuenta con un chip de memoria *flash* AMD de 512 Kb, que está unida a la tarjeta por una base que permite cambiarlo si es necesario. También trae un chip de memoria SRAM de 512 Kb, que mediante puentes puede sustituir a la memoria *flash* para almacenar el código del programa. Para almacenar datos y variables, se cuenta con una memoria SRAM de 32 Kb.

Además de estos dispositivos, el módulo también tiene un chip manejador de la interfaz RS-232 y sus respectivos conectores DB-9.; así como un regulador de tensión, que acepta entradas de 7V a 15V CD y mantiene internamente una tensión de 5 V. También cuenta con un botón pulsante para restablecimiento.

Para cargar el programa del usuario en la memoria *flash*, la tarjeta incluye una utilidad que se carga en la memoria EEPROM interna del microcontrolador, que se ejecuta al poner éste en modo sencillo. Esta utilidad recibe el programa en formato S19 por medio del puerto serie SCI0 y lo carga en memoria *flash*. Luego, para ejecutar el programa del usuario, se pone el interruptor del modo en expandido y se le envía la señal de restablecimiento al microcontrolador.

Para la implementación del sistema traductor de protocolo, se utilizó el puerto SCI0 (conector DB-9, abajo en la fotografía de la figura 4.1) para la programación, configuración del traductor y envío / recepción de mensajes IEC, por lo que se le ha denominado a éste, para efectos de este informe, como “Puerto IEC / Config”. El otro puerto serie disponible (SCI1) se utiliza exclusivamente para envío y recepción de datos Modbus y se conecta a la UTR, por lo que se le ha denominado “Puerto Modbus”.

CAPÍTULO 5:
DESCRIPCIÓN DEL SOFTWARE DEL SISTEMA

5.1 Programa del microcontrolador

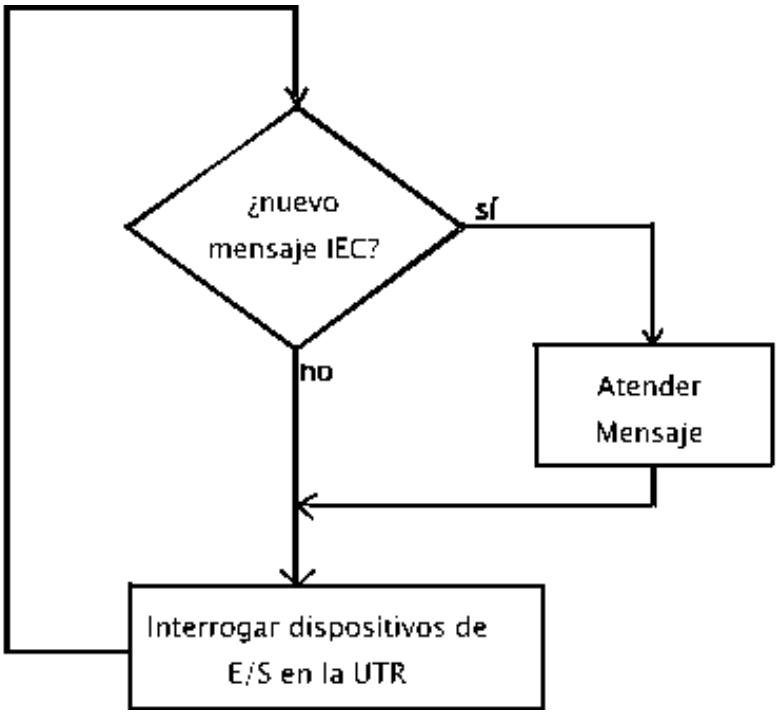
5.1.1 Programa principal

Este programa conforma el núcleo de este proyecto. Se encarga de recibir y decodificar los mensajes de la estación maestra, enviarle los mensajes de vuelta, enviar a la UTR los mensajes de interrogación, recibir y decodificar los mensajes retornados por ésta y asignarle a cada dispositivo su respectiva dirección IEC.

Como también se mencionó ya, este programa interroga continuamente cada uno de los dispositivos de la UTR (entradas y salidas) para detectar si el valor o estado de éstos ha cambiado con respecto al dato almacenado en el espejo. Si esto es así, se activa una bandera específica para el dispositivo; cuando la estación maestra haga un *request data* los dato de este dispositivos serán reportados.

El programa revisa continuamente el puerto IEC para determinar si ha llegado un nuevo mensaje. Mientras esto sucede se va efectuando el muestreo de dispositivos de la UTR, interrogando los dispositivos analógicos de uno en uno y a los digitales ocho a la vez. Cuando llega un nuevo mensaje IEC, el sondeo se suspende y el programa atiende la solicitud de la estación maestra.

En la figura 5.1 se muestra el diagrama de flujo simplificado del programa.



FLASH / PAINT

Figura 5.1 Diagrama de flujo simplificado del programa principal

En la figura 5.2 se muestra un diagrama donde se detallan los procedimientos de adquisición de datos, envío de datos y atención de instrucciones de la estación maestra.

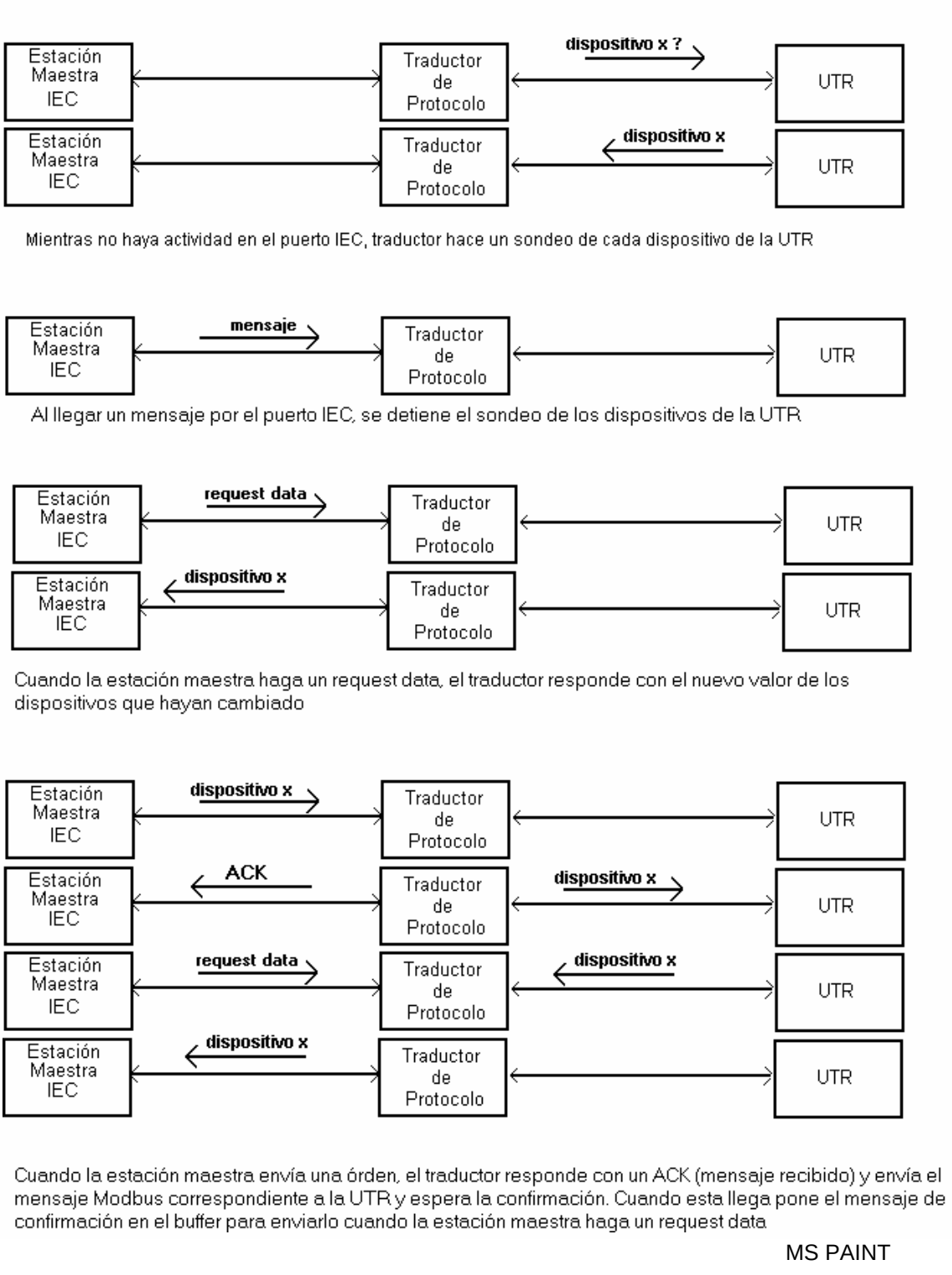


Figura 5.2 Procedimientos seguidos por el traductor

5.1.2 Rutina de configuración

Este programa también incluye una rutina de configuración, que se ejecuta al inicio del programa y que le permite al usuario o administrador del sistema introducir parámetros necesarios para que la el sistema efectúe la traducción. Estos parámetros se organizan en tres grupos: variables de configuración general, variables de configuración IEC 870 y variables de configuración Modbus.

Para acceder a esta rutina, el traductor debe conectarse por el puerto IEC / Config a una PC que tenga disponible un puerto serie con un programa emulador de terminal, como por ejemplo el *Hyperterminal* de *Windows*. En la figura 5.3 se muestra una pantalla del HyperTerminal cuando se ejecuta la rutina de configuración

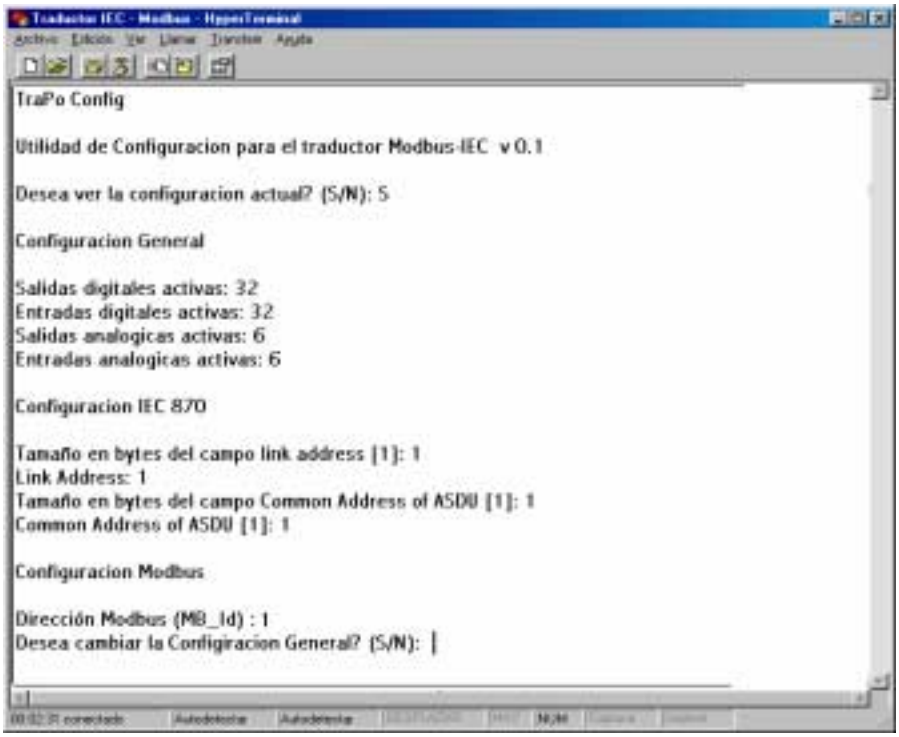


Figura 5.3 Pantalla del programa Hyperterminal al ejecutar la rutina de configuración

Vale la pena mencionar que para el desarrollo de este programa también se tuvo que desarrollar rutinas intermedias para evaluar lo que se había programado hasta ese momento, como lo fueron las rutinas para controlar el envío y recepción de datos Modbus cuando esta parte se estaba desarrollando. También se usaron rutinas similares para ver los mensajes IEC que se estaban enviando.

5.2 Simulador de la estación maestra

Como su nombre lo indica, este programa le envía mensajes al traductor como si fuera la estación maestra y despliega en la pantalla de la PC los mensajes de respuesta de ésta. La finalidad primaria de este software fue, en primera instancia la de comprobar que las rutinas de comunicación con la estación maestra funcionaran de forma correcta, y en segunda instancia, la de comprobar el buen funcionamiento del sistema en conjunto.

Como ya se mencionó, este software fue programado en Dephi 5, lo que permitió una programación visual, orientada a objetos y que aprovecha la interfaz de usuario del sistema operativo Windows de Microsoft.

La interfaz del programa cuenta con una serie de botones, cada uno representa una orden del protocolo IEC 870-5-101. Cuando el usuario presiona uno de ellos con el puntero del ratón, se envía la orden correspondiente. Si para una función en particular se necesita más información, se abre un cuadro de diálogo para que el usuario la suministre.

En las figuras 5.4, 5.5 y 5.6 se muestran pantallas de este programa.

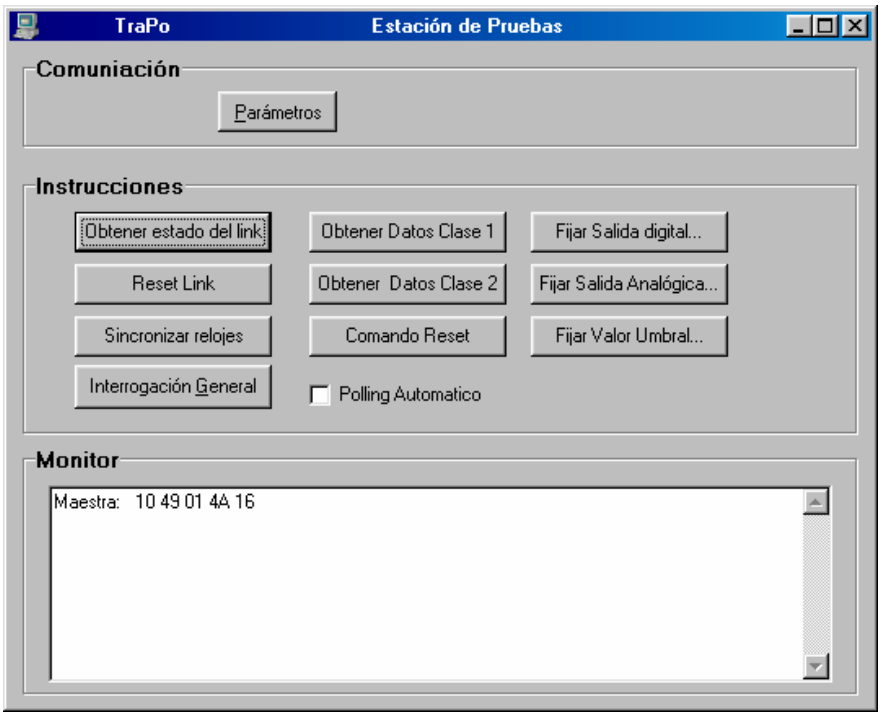


Figura 5.4 Pantalla principal del simulador de la estación Maestra

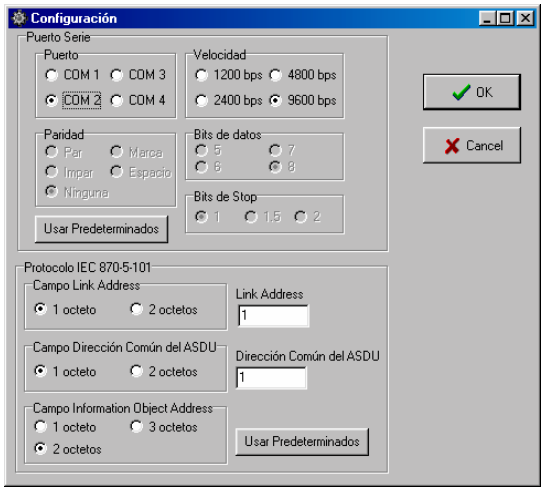


Figura 5.5 Pantalla de configuración del simulador de la estación maestra

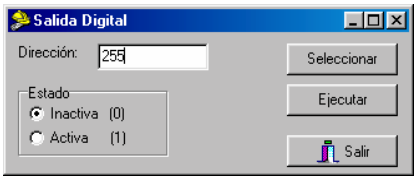


Figura 5.6 Cuadro de diálogo del simulador de la estación maestra

CAPÍTULO 6:
ANÁLISIS Y RESULTADOS

6.1 Explicación del diseño

6.1.1 Programa para el microcontrolador

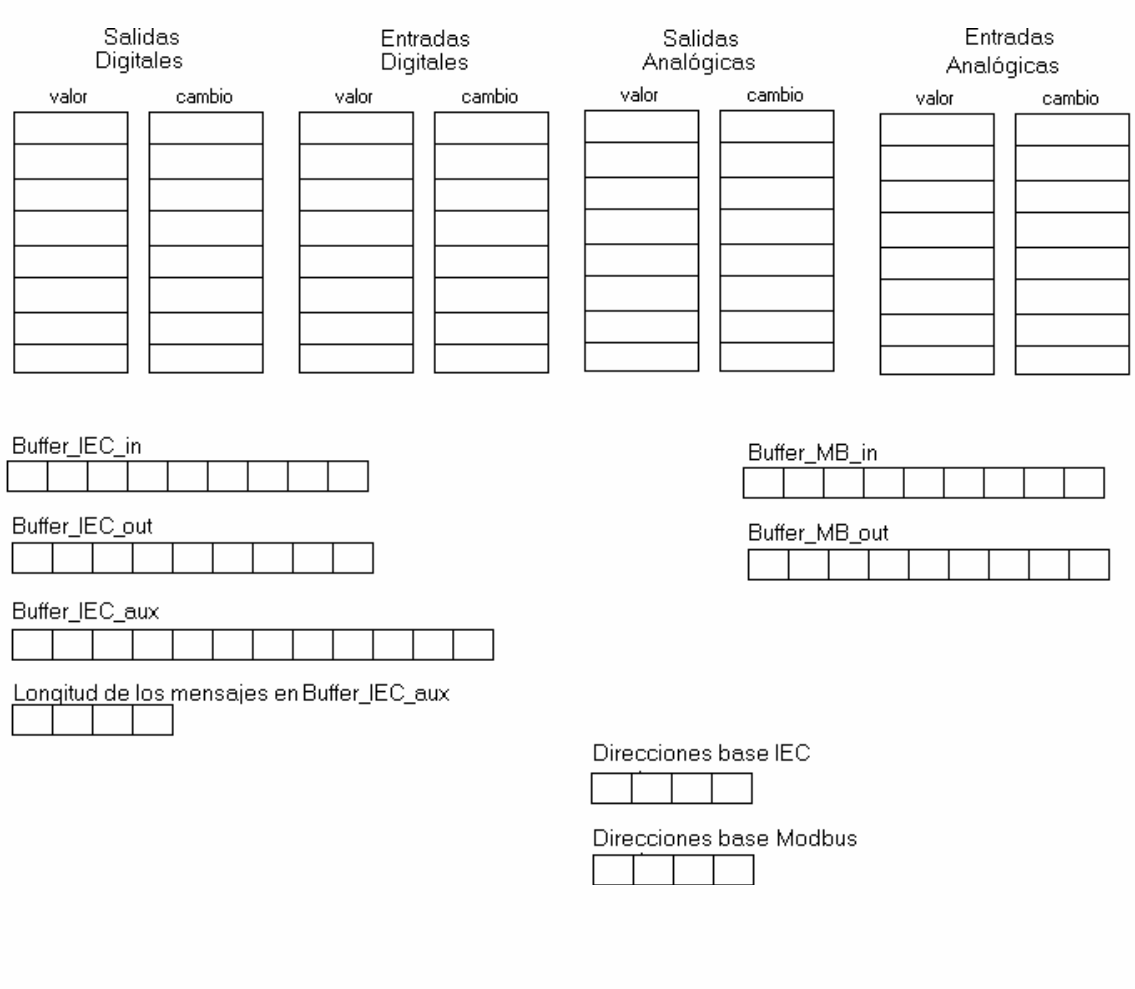
Ya en apartados anteriores se explicaron las funciones de este software, a continuación se dan detalles de cómo se implementó.

En este programa se aprovechó la capacidad del microcontrolador para utilizar paginación de memoria; en este caso, se programó la rutina de configuración en memoria paginada (Página 0) mientras que el resto de las rutinas se colocaron en memoria normal.

Aparte de la memoria RAM requerida para las variables usadas para el programa, se definió un espacio de memoria temporal o *buffer* de entrada y otro de salida de 12 bytes cada uno para el puerto Modbus. También se declararon estructuras similares para el subsistema IEC, sin embargo, estas últimas son más grandes (hasta 180 bytes) pues este protocolo hace un uso extensivo de estos *buffers*. Todas estas estructuras se ubicaron en la memoria RAM interna del microcontrolador.

El espejo de los dispositivos de la UTR se implementó en un arreglo de 64 bytes para dispositivos analógicos (2 bytes para cada entrada / salida, para 32 entradas y 32 salidas) ubicados en memoria RAM interna y de 112 bytes para los dispositivos digitales (1 byte para cada entrada / salida, para un total de 112 entradas y 112 salidas) ubicados en la memoria SRAM externa. También se ubicó en esta memoria un arreglo de 1 byte para cada dispositivo donde se guarda una variable booleana que indica si el valor ha cambiado desde la última vez que se reportó y por tanto debe ser reportado nuevamente a la estación maestra.

En la figura 6.1 se muestra un esquema con todas estas estructuras de datos. También se muestra el *buffer* auxiliar, donde se guardan los mensajes de respuesta IEC para enviarlos cuando la estación maestra pregunte por ellos.



MS PAINT

Figura 6.1 Estructuras de datos utilizadas en el programa

En el capítulo anterior se vio que el programa alternadamente interroga un dispositivo en la UTR y revisa si hay un nuevo mensaje IEC para el traductor, de ser así lo atiende. Este algoritmo se ilustró mediante el diagrama de flujo de la figura 5.1

Al inicio del programa, el dispositivo se pone en modo configuración, para ejecutar la rutina de configuración descrita en el capítulo anterior. Como ya se vio, el programa hace un diálogo con el administrador del sistema para obtener parámetros necesarios para el proceso de traducción de protocolo. Las respuestas que se obtengan se almacenan en memoria RAM como variables booleanas o numéricas, de 1 o 2 bytes.

La recepción de datos tanto por el puerto Modbus (SCI1 en el microcontrolador) como por el puerto IEC (SCI0) se da mediante interrupción: cuando llega un nuevo caracter el microcontrolador llama a una rutina de servicio de interrupción que agrega el dato recibido en el *buffer* correspondiente. Periódicamente, en diferentes puntos del programa se llama a una rutina que revisa si se ha dado un silencio del tiempo de al menos un caracter en el puerto IEC, lo que significa que le mensaje que se estaba recibiendo ha terminado. Si esto sucede, se llama a otra rutina que lo valida, es decir, verifica que la trama esté correcta y que el mensaje tenga como *link address* o dirección de enlace destino la dirección programada en el traductor; si no es así, éste asume que el mensaje es para otra estación remota y lo desecha sin procesarlo. Si el mensaje es válido, se activa la bandera *Mensaje_in_esperando* (nuevo mensaje IEC) para que sea atendido la próxima vez que se pregunte por ella.

El sondeo de los dispositivos de la UTR se hace alternadamente según el número de dispositivos que se haya programado en la configuración del traductor. Si por ejemplo, se especificaron 20 entradas digitales y 6 analógicas, primero se preguntará por las primeras ocho entradas digitales (0 a 7), luego por la primera analógica (0), luego por las siguientes ocho digitales (8 a 15) y luego por la siguiente analógica. Cuando se haya interrogado al último dispositivo, en la interrogación siguiente se continuará con el primero.

Una vez enviado el mensaje de interrogación Modbus, el traductor espera el mensaje de respuesta de la UTR y extrae de éste la información solicitada. Con los datos entrantes se actualiza el espejo en memoria del estado de los dispositivos de la UTR. También se compara con el dato previamente almacenado. Si se ha dado un cambio en el estado de éste, se activa la bandera "cambio" específica para este. También se activa la bandera "cambio" específica para el tipo de dispositivo que cambió, así como la bandera específica para su clase o prioridad.

Cuando se atiende un nuevo mensaje IEC, se revisa primero el código de función del mensaje en capa 2. Según el mensaje se realiza la acción requerida y se envía la confirmación correspondiente. Si el código de función es 3 ó 4, significa que el mensaje contiene un ASDU (mensaje de capa 7). Si es así, se envía un *ACK* (mensaje recibido), se llama a una subrutina donde es procesado; igual que en la capa inferior, se revisa el *type ID* del ASDU y se procede a ejecutar la acción solicitada. Si se trata de una interrogación general, se componen los mensajes correspondientes a los dispositivos de la UTR y se almacenan en *Buffer_iec_aux*. Si el ASDU corresponde a una orden, se hace el mensaje Modbus correspondiente y se envía por el puerto Modbus. Cuando llega la confirmación de la UTR, se hacen los mensajes de confirmación IEC y se guardan en *Buffer_iec_aux.*

Si el mensaje de capa 2 entrante es un *request data* se lee la bandera de cambios para la clase de datos por la que se esté preguntando. Si hay datos para esa clase se arma el mensaje correspondiente con los valores de los dispositivos que han cambiado junto con sus direcciones. Si se pregunta por datos clase 1 y no hay cambios en estos dispositivos pero hay al menos un mensaje en el *Buffer_iec_aux,* se envía el primero que se haya puesto (este *buffer* es de tipo FIFO) Si no hay datos del todo, se envía un mensaje *NACK* (no hay datos disponibles)

Una vez atendido el mensaje entrante, se sigue interrogando dispositivos en la UTR hasta que llegue un nuevo mensaje IEC.

6.1.2 Simulador de la estación maestra

Este software utiliza el componente *nrComm* para acceder al puerto serie, y es inicializado una vez que el usuario llame a la pantalla de configuración e ingrese los parámetros correspondientes. Dado que Delphi es un lenguaje de programación orientado a eventos, las acciones que realice el programa están dadas por las acciones del usuario.

Cuando el usuario pulse un botón correspondiente a una orden IEC 870, se empieza a construir un *buffer* de salida con los datos del mensaje correspondientes, el cual es enviado por el puerto serie y desplegado en pantalla.

El objeto nrComm tiene definido el evento *onAfterReceive* cuando llegan datos al puerto serie. La rutina que se implementó para atender este evento agrega los datos recibidos a un *buffer* de entrada, y cuando detecta que se ha completado la recepción de un mensaje, lo valida, es decir, verifica que la trama esté correcta y lo muestra en la pantalla.

Como el objetivo de este programa es solamente evaluar el desempeño del dispositivo traductor, no se desarrollo la interfaz de usuario de una aplicación SCADA, sino solamente las funciones necesarias para la evaluación del traductor.

6.2 Alcances y limitaciones

Con la implementación de esta solución se cumplió con el objetivo de construir un dispositivo que le diera conectividad a la UTR desarrollada por I+D con el protocolo IEC 870 utilizado por la estación maestra *Ranger*; aunque es importante mencionar que esta compatibilidad es muy básica y su grado de efectividad va a depender de la aplicación en particular. Hay que tomar en cuenta que no es conveniente asignar este sistema a una aplicación donde sea necesario un tiempo de respuesta inmediato, ya que el retardo que existe desde que cambia el valor de un dispositivo de la UTR hasta que el dato llega a la estación maestra, aumenta considerablemente.

También hay que tener presente que no se pueden utilizar funciones del protocolo IEC 870-5 que no existen en Modbus, como es el manejo de contadores, puntos dobles y la transmisión de pulsos de corta y larga duración.

Estas son limitaciones intrínsecas de la solución elegida y no pueden ser eliminadas sin cambiar el esquema utilizado; en el mejor de los casos podría mejorarse el diseño para minimizar su efecto. Si es necesario superar estas limitaciones, debe pensarse en cambiar el enfoque de la solución; aquí podría considerarse en modificar la UTR para que utilice directamente el protocolo IEC 870-5

Sin embargo, otras limitaciones son propias de esta implementación y pueden mejorarse en una optimización posterior. Entre éstas podría mencionarse la volatilidad de la configuración guardada en memoria, pues ésta se pierde al reiniciar el traductor o al cortar la alimentación, y debe ser introducida de nuevo.

También puede considerarse una limitante el hecho de que la configuración no pueda hacerse de forma remota, pues hay que tener acceso al botón de restablecimiento.

Como una limitante, también se debe mencionar el número máximo de dispositivos que el traductor puede manejar. Para esta implementación en particular se hizo soporte para un máximo de 112 entradas digitales, 112 salidas digitales, 32 entradas analógicas y 32 salidas analógicas; esto se debe a que para las interrogaciones generales se hacen mensajes extremadamente grandes (más de 100 bytes) y se encontraron problemas al trabajar con mensajes de este tamaño. También influyó el hecho de que se tuvieron problemas al trabajar con la memoria SRAM externa de la tarjeta Adapt812DXLT, donde no se podía leer datos de una columna, lo que obligó a utilizar la memoria interna del microcontrolador para alojar la mayoría de las variables, lo que resultó en una limitación de espacio significativa.

También es importante destacar que aunque se comprobó el funcionamiento del traductor con una UTR Modbus, solamente se han utilizado simuladores para evaluar el funcionamiento del sistema ante una estación maestra IEC, por lo que su comportamiento en el mundo real, con una estación maestra IEC 870-5-101 con equipo de comunicación especial y una posible configuración tipo *party line* donde se tenga que compartir en enlace con otras UTR no está todavía comprobado.

CAPÍTULO 7:
CONCLUSIONES Y RECOMENDACIONES

7.1 Conclusiones

- El Protocolo IEC 870-5-101 tiene una complejidad mayor a la del protocolo Modbus.
- La solución óptima al problema fue la de implementar un traductor de protocolo.
- La tarjeta ADAPT812DXLT es el dispositivo de hardware que mejor se adapta a la solución para este problema.
- Se cumplió con el objetivo de lograr compatibilidad entre una UTR Modbus y sistema SCADA basado en el protocolo IEC 870-5-101, dentro de ciertas limitaciones
- El sistema tiene limitaciones propias de la implementación, que pueden superarse en futuras implementaciones del diseño
- El sistema tiene limitaciones intrínsicas del diseño, que pueden ser superadas solamente cambiando el enfoque que se le dio a la solución.
- Las limitaciones más significativas de esta solución son: tiempos de retardo y funciones IEC no disponibles en Modbus.
- La conveniencia de utilizar esta aplicación en un sistema SCADA, depende de los requerimientos de la aplicación en particular, poniendo especial atención a las limitaciones de esta solución

7.2 Recomendaciones

Como ya se vio, esta solución tiene ciertas limitaciones, algunas intrínsecas a la solución y otras propias de esta implementación. El personal responsable del proyecto SCADA del departamento de Investigación y Desarrollo deberá evaluarlas para decidir cuáles son más críticas y entonces definir cual será el rumbo que tome la solución al problema: seguir con la propuesta original o idear una nueva estrategia.

De seguir el primer camino, hay muchos aspectos en los que se puede mejorar el traductor, como por ejemplo, modificar la rutina de configuración para que pueda accederse remotamente, y que guarde los datos en memoria EEPROM, de modo que no se borren tan fácilmente. Soporte para manejo de múltiples UTRs al mismo tiempo, soporte para funcionamiento simultáneo en más de un sistema SCADA, manejo de un reloj interno, etc.

También se pueden implementar más funciones del protocolo IEC, pues en este proyecto se hizo una implementación muy básica, como por ejemplo, implementar ASDUs con etiqueta de tiempo o *time tag,* sincronización de reloj o configuración remota mediante transferencias de archivo. También resultaría conveniente agregar otras características que lo hagan mejorar en su robustez y confiabilidad.

Una vez optimizado el diseño, podría construirse el prototipo final, ya no usando la tarjeta Adapt812DXLT, sino fabricando un sistema similar en una PCB con el mismo procesador y componentes equivalentes y que pudiera estar físicamente integrado en la UTR.

De seguir el segundo camino, habría que considerar modificar el diseño de la UTR, lo que puede resultar problemático y costoso para el Centro de Servicio, pero que sería la única alternativa si las circunstancias así lo exigen.

Bibliografía

Statnett SF. Norwegian IEC 870-5-101 User Conventions
1997 <http://www.statnett.no/Files/Open/IEC-R20_1.pdf> (Febrero 2002)

Lizana, Fernando. "Implementación de un traductor de protocolo Modbus para una unidad terminal remota de un sistema SCADA" Informe de Proyecto de Graduación para optar por el Grado de Bachiller en Ingeniería Electrónica. Escuela de Ingeniería en Electrónica, ITCR. Cartago. 2001

BOW Software Inc. Telemetry Integration Enviroment (TIE) IEC 870-5 Remote Interface Functional Specification Versión 1.16 BOW Software. Alberta, Canadá. 1997

Motorola Inc. MC68HC812A4 Advance Information
Rev 5. Motorola Literature. Colorado, EUA. 2001

Motorola Inc. CPU12 Reference Manual
Rev 2.0 Motorola Literature. Colorado, EUA.. 2000

MODICON Inc. Modicon Modbus Protocol Reference Guide Revision J, 1996
MODICON Industrial Automation Systems. Massachussets, EUA

Apéndices

Apéndice A1: Protocolo Modbus

A1.1 Descripción

El protocolo Modbus fue desarrollado por la empresa Modicon para poder comunicar equipos desarrollados por esta compañía, como PLCs. Estrictamente es un protocolo propietario de esta empresa, sin embargo Modicon ha publicado todas las especificaciones del protocolo y da licencias gratuitas del mismo a cualquier organización que decida adoptarlo; como resultado, en la práctica el Modbus es un protocolo abierto, utilizado por muchos fabricantes de equipos.

Existen variantes del protocolo: Modbus, Modbus Plus y Modbus TCP. El primero es la versión original, que se utilizará en este proyecto. El segundo introduce conceptos como capas o enrutamiento, el tercero utiliza el protocolo TCP/IP para comunicar la UTR a través de una LAN o hasta a través de Internet.

El protocolo Modbus define la estructura de los mensajes que serán enviados por la computadora central a las diferentes Unidades Terminales Remotas; describe el proceso que un controlador debe utilizar para solicitar acceso a otros dispositivos, cómo debe responder a los mensajes de los otros dispositivos y cómo deben ser detectados y reportados los errores.

El protocolo Modbus utiliza una técnica de maestro-esclavo cuando el controlador requiere comunicarse con los otros dispositivos. Los dispositivos esclavos responden enviando la información requerida o realizando las acciones indicadas en el mensaje. El controlador maestro puede enviar mensajes a los dispositivos esclavos individualmente o puede enviar mensajes generales. Los esclavos responderán únicamente a los mensajes enviados individualmente.

El protocolo Modbus establece el formato para los mensajes del dispositivo maestro indicando la dirección del dispositivo esclavo, un código definiendo la acción requerida, la información que debe ser enviada y una señal de detección de errores. El esclavo responderá utilizando también este protocolo. Si ocurriera un error en el recibimiento del mensaje, o si el dispositivo esclavo no puede realizar la acción requerida, entonces éste construirá un mensaje de error y lo enviará al dispositivo maestro

Al enviar un mensaje, el código indica al dispositivo direccionado cuál es la acción que debe realizar. Los bytes de información contienen los datos adicionales que pueda requerir el esclavo para realizar la función. Por ejemplo, la función con código 03 solicitará al esclavo que lea los registros de memoria y que responda con su contenido. Los bytes de información deben indicar al esclavo en cuál registro comenzar con la lectura y cuántos registros debe leer. El espacio asignado para detección de errores provee un método para el esclavo para valorar la integridad del contenido del mensaje.

Al responder un mensaje, el esclavo enviará los datos requeridos por el maestro. Si ocurriera un error, el código de función es modificado para indicar que la respuesta es una respuesta de error, y los bytes de información contendrán un código que describirá el error.

Existen dos modos de transmisión serie de los datos, en el protocolo Modbus. Estos son el modo ASCII y el modo UTR. Para el presente proyecto, el modo de transmisión que se utilizó fue el UTR.

A1.2 Estructura de los mensajes

A la hora de enviar un mensaje, el protocolo Modbus especifica un punto de inicio o señal de inicio y un punto final. Esto permite a los dispositivos esclavos saber cuándo inicia y cuándo termina un mensaje, para así leer correctamente los datos enviados. En el modo UTR, el inicio de un mensaje está indicado por una señal nula equivalente a 3,5 caracteres (los caracteres son hexadecimales, es decir, son de 4 bits cada uno). Luego, se transmite la dirección del dispositivo con el que se requiere comunicación. Una vez terminado el mensaje, el protocolo indicará el final del mismo enviando un intervalo de 3,5 caracteres. Luego de esto, un nuevo mensaje puede ser enviado. La tabla A1.1 ilustra la estructura de un mensaje Modbus, enviado mediante la utilización del modo UTR.

Tabla A1.1 Estructurade un mensaje Modbus en modo UTR

Inicio	**Dirección**	**Función**	**Datos**	**Verificación de CRC**	**Finalización**
T1-T2-T3-T4	8 bits	8 bits	n * 8 bits	16 bits	T1-T2-T3-T4

A1.3 Direccionamiento de los dispositivos esclavos

Para direccionar los diferentes dispositivos esclavos, en los mensajes enviados por el maestro se reservan dos caracteres (8 bits). La dirección cero está reservada para el envío de mensajes generales. Los demás estarán ubicados en las direcciones que van desde el 1 hasta el 247 decimal. El dispositivo maestro accede a algún otro poniendo la ubicación en el espacio de dirección del mensaje. El esclavo responde colocando su dirección en el espacio reservado para esto en el mensaje, para indicar así al maestro cuál dispositivo está respondiendo.

A1.4 Instrucciones Modbus

El tercer espacio de un mensaje Modbus está reservado para la función o instrucción que se solicita al esclavo que realice. En las tablas A1.2 y A1.3 se muestran las instrucciones implementadas en el protocolo Modbus

Tabla A1.2 Instrucciones implementadas en el protocolo Modbus

Número	**Nombre**	**Comentario**
1	Lectura de estado de señales digitales	Solicita al esclavo la lectura del estado de un número de señales digitales. La dirección de las mismas es referenciada a 0x (Es decir, direcciones desde 00000 hasta 0FFFFH)
2	Lectura de estado de señales digitales 2	Solicita al esclavo la lectura del estado de un número de señales digitales. La dirección de las mismas es referenciada a 1x
3	Lectura de estado de señales analógicas	Solicita al esclavo la lectura del estado de un número de señales analógicas. La dirección de las mismas es referenciada a 4x
4	Lectura de estado de señales analógicas 2	Solicita al esclavo la lectura del estado de un número de señales analógicas. La dirección de las mismas es referenciada a 3x
5	Escritura de una señal digital	Indica al esclavo el valor que debe colocar en determinada señal digital
6	Escritura de una señal analógica	Indica al esclavo el valor que debe colocar en determinada señal analógica
15	Escritura de varias señales digitales	Indica al esclavo los valores que debe colocar en diferentes señales digitales
16	Escritura de varias señales analógicas	Indica al esclavo los valores que debe colocar en diferentes señales analógicas

Apéndice A2: Protocolo IEC 870-5

A2.1 Introducción

El protocolo IEC 870-5 se utiliza en aplicaciones de Control Supervisorio y Adquisición de Datos (SCADA) según los recomienda la Comisión Electrotécnica Internacional y se basa en la arquitectura de Rendimiento Mejorado (EPA), que es un modelo de tres capas para interconexión de sistemas abiertos, el cual, provee una respuesta más rápida que el modelo OSI de 7 capas. Se compone de:

a. Capa física

b. Capa de enlace de datos

c. Capa de aplicación

A2.2 Capa Física

A nivel físico, la comunicación se da en modo desbalanceado, esto es, siguiendo el esquema maestro-esclavo, en el cual, las UTR envían datos solamente a la estación central y cuando ésta los solicite. Se sigue la recomendación de la UIT V.24/V.28 la cual es equivalente a la interfaz RS-232. Soporta las siguientes velocidades:

a. 100 bit/s

b. 200 bits/s

c. 300 bits/s

d. 600 bits/s

e. 1200 bits/s

f. 2400 bits/s

g. 4800 bits/s

h. 9600 bits/s

El protocolo establece varios formatos de datos. Para este caso particular se utilizará el FT 1.2, en el cual, cada caracter transmitido se compone de:

a. 1 bit de inicio (0)

b. 8 bits de datos, empezando con el LSB

c. 1 bit de paridad

d. 1 bit de paro (1)

Cuando no se transmiten datos la línea permanece en 1 lógico. La transmisión es asincrónica.

A2.3 Capa de enlace de datos

En este nivel se define una trama, llamada LPCI que consiste, para mensajes de longitud fija, en un caracter de inicio, mensaje (denominado LSDU), suma cíclica y caracter de paro, como se muestra en la figura A2.1

Caracter de Inicio
LSDU (datos)
Suma cíclica
Caracter de paro

Figura A2.1 Formato para mensajes de longitud fija

Para los mensajes de longitud variable, la trama consiste en un caracter de inicio, longitud del mensaje (repetida), segundo caracter de inicio, mensaje (LSDU), suma cíclica y caracter de paro. El mensaje propiamente dicho o LSDU, se compone de un campo de control, dirección de destino y los datos correspondientes a la función.

Primer Caracter de Inicio
Longitud L
Longitud L
Segundo Caracter de Inicio
LSDU (datos, L bytes)
Suma cíclica
Caracter de paro

Figura A2.2 Formato para mensajes de longitud variable

También se utilizan mensajes cortos, de un byte de longitud incluyendo trama, que se utilizan para funciones sencillas, como confirmaciones.

Las tablas A2.1 y A2.2 presentan las funciones del protocolo IEC 870-5 en modo no balanceado para capa 2, de la estación primaria a secundaria y de secundaria a primaria, respectivamente:

Tabla A2.1 Funciones de capa 2 del protocolo IEC 870 en dirección primaria

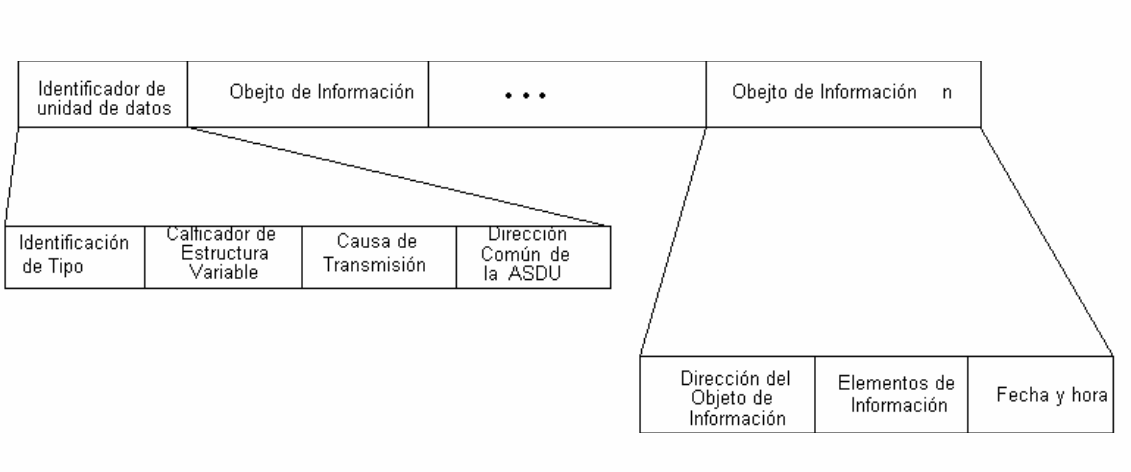
Código de función	Acción
0	Restablecer enlace
1	Restablecer proceso
2	Reservado para modo balanceado
3	Datos de usuario
4	Datos de usuario
5	Reservado
6-7	Reservado para uso especial
8	Solicitar señal de demanda de acceso
9	Solicitar estado del enlace
10	Solicitar datos de usuario clase 1
11	Solicitar datos de usuario clase 2
12-13	Reservado
14-15	Reservado para uso especial

Tabla A2.2 Funciones de capa 2 del protocolo IEC 870 en dirección secundaria

Código de función	Acción
0	ACK Notificación positiva
1	NACK Notificación negativa
2-5	Reservado
6-7	Reservado para uso especial
8	Datos de Usuario
9	NACK Datos no disponibles
10	Reservado
11	Estado del enlace o demanda de acceso
12	Reservado
13	Reservado para uso especial
14	Servicio de enlace no funciona
14-15	Servicio de enlace sin implementar

A2.4 Capa de aplicación

Esta capa define los datos que se envían en el campo (LSDU) de datos visto en la sección anterior. Éste se compone de un campo de control, una dirección de enlace y una Unidad de Datos de Servicios de Aplicación. (ASDU) La ASDU a su vez, consiste en un Identificador de Unidad de Datos, y uno o varios objetos de información, tal como se muestra en la figura A2.3.



MS PAINT

Figura A2.3 Formato de la unidad de datos de servicios de aplicación

El identificador de datos lleva información general y común a los objetos de información, como lo es el identificador de tipo, que indica qué tipo de instrucción o información se está transmitiendo, así como el formato de los datos. El calificador de estructura variable indica si el ASDU incluye objetos de información y si es así, cuántos; si varios datos se almacenan en una secuencia en un objeto o cada uno va en un objeto por separado; y cuántos elementos de información van en el objeto. El campo correspondiente a la causa de la transmisión indica si ésta es espontánea, cíclica, por pedido, etc. La dirección común de la ASDU es un byte que representa la dirección de la UTR.

El objeto de información lleva la dirección del mismo, que corresponde a la dirección del dispositivo (entrada o salida, digital o analógica) que generó la información o que se desea interrogar. Los elementos de información llevan los datos propiamente dichos, seguidos del campo correspondiente a la hora/fecha.

A continuación se presenta un esquema para la estructura de los datos hasta ahora mencionados:

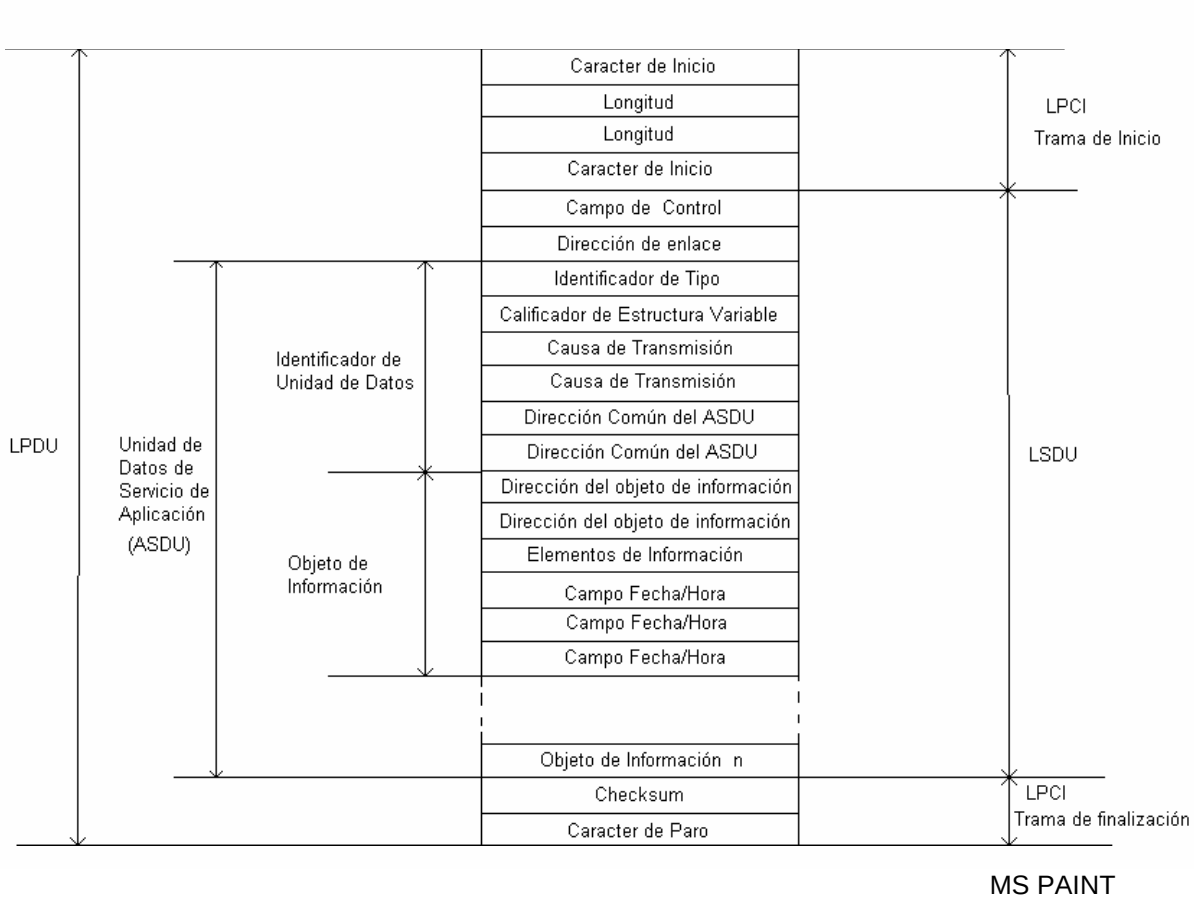


Figura A2.4 Estructura de un mensaje (LPDU) en el protocolo IEC 870-5

A2.5 Procedimientos de comunicación

La norma IEC 870-5-101 especifica los siguientes procedimientos de comunicación:

1. Iniciación de la UTR
2. Adquisición de datos mediante interrogaciones
3. Transmisión de datos cíclicos
4. Adquisición de eventos
5. Interrogación general
6. Sincronización del reloj
7. Transmisión de órdenes
8. Transmisión de totales integrados
9. Carga de parámetros
10. Prueba de procedimiento
11. Transferencia de archivo
12. Adquisición de retardo de transmisión

Cada procedimiento se compone de una serie de mensajes o telegramas intercambiados entre la estación primaria y secundaria. Un procedimiento solo podrá ser iniciado por la estación primaria o maestra, dado que se está trabajando en modo desbalanceado; la estación secundaria solo responderá si el tipo de función o mensaje enviado por la estación primaria así lo requiere. Luego, la estación primaria preguntará por más datos o enviará más órdenes a esa estación secundaria o a otras, repitiendo el ciclo hasta haber completado el procedimiento.

A2.6 Funciones de capa 7 implementadas

Se implementaron las siguientes funciones en capa de aplicación:

a. Iniciación remota de la estación
b. Interrogación general
c. Transmisión de orden directa (set point)
d. Selección y ejecución de instrucción (Instrucción directa solamente)
e. Carga de parámetro, valor umbral

Para implementar estas funciones, fue necesario desarrollar soporte para los siguientes ASDUs:

Información de proceso en dirección de monitor:

<1> := Información de punto simple	M_SP_NA_1
<9> :=Valor medido, normalizado	M_ME_NA_1

Información de proceso en dirección de control

<45> := Orden Simple	C_SC_NA_1
<48> := Orden de puesta de valor normalizado	M_SE_TB_1

Información de sistema en dirección de monitor

<70> := Fin de Iniciación	M_EI_TB_1

Información de sistema en dirección de control

<100> := Orden de Interrogación	C_IC_NA_1
<104> := Orden de Prueba	C_TS_NB_1
<105> := Orden de Reset de proceso	C_RP_NC_1

Parámetro en dirección de control

<110> := Parámetro del valor medido. Normalizado	P_ME_NA_1

Apéndice A3: Manual de usuario

A3.1 Introducción

El sistema Traductor de Protocolo IEC 870 – Modbus es un dispositivo diseñado para hacer posible la conexión entre una Unidad Terminal Remota (UTR) compatible con el protocolo Modbus y un sistema SCADA basado en el protocolo IEC 870-5-101. En particular, se diseñó para enlazar la UTR desarrollada por el Centro de Servicio I+D con la estación maestra *Ranger* del Centro de Control de Energía del ICE. En la siguiente figura se muestra un esquema la conexión entre los dispositivos:

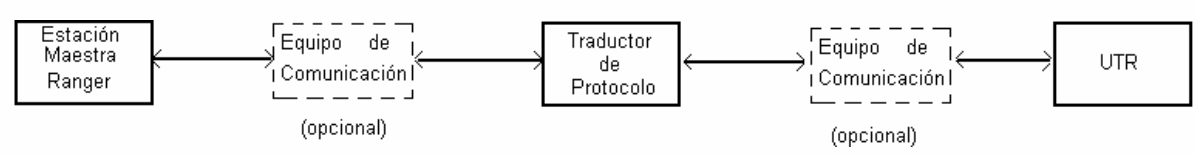


Figura A3.1 Conexión entre los dispositivos involucrados

Así entones, el traductor es visto por la UTR como una estación maestra Modbus, mientras que el conjunto traductor – UTR es visto por la estación maestra IEC 870-5 como una remota IEC.

A3.2 Descripción

El prototipo actual del traductor de protocolo está montado sobre la tarjeta de desarrollo Adapt812DXLT, fabricada por la empresa canadiense *Technological Arts*.

A continuación se muestra la tarjeta Adapt812DXLT con sus partes más importantes:

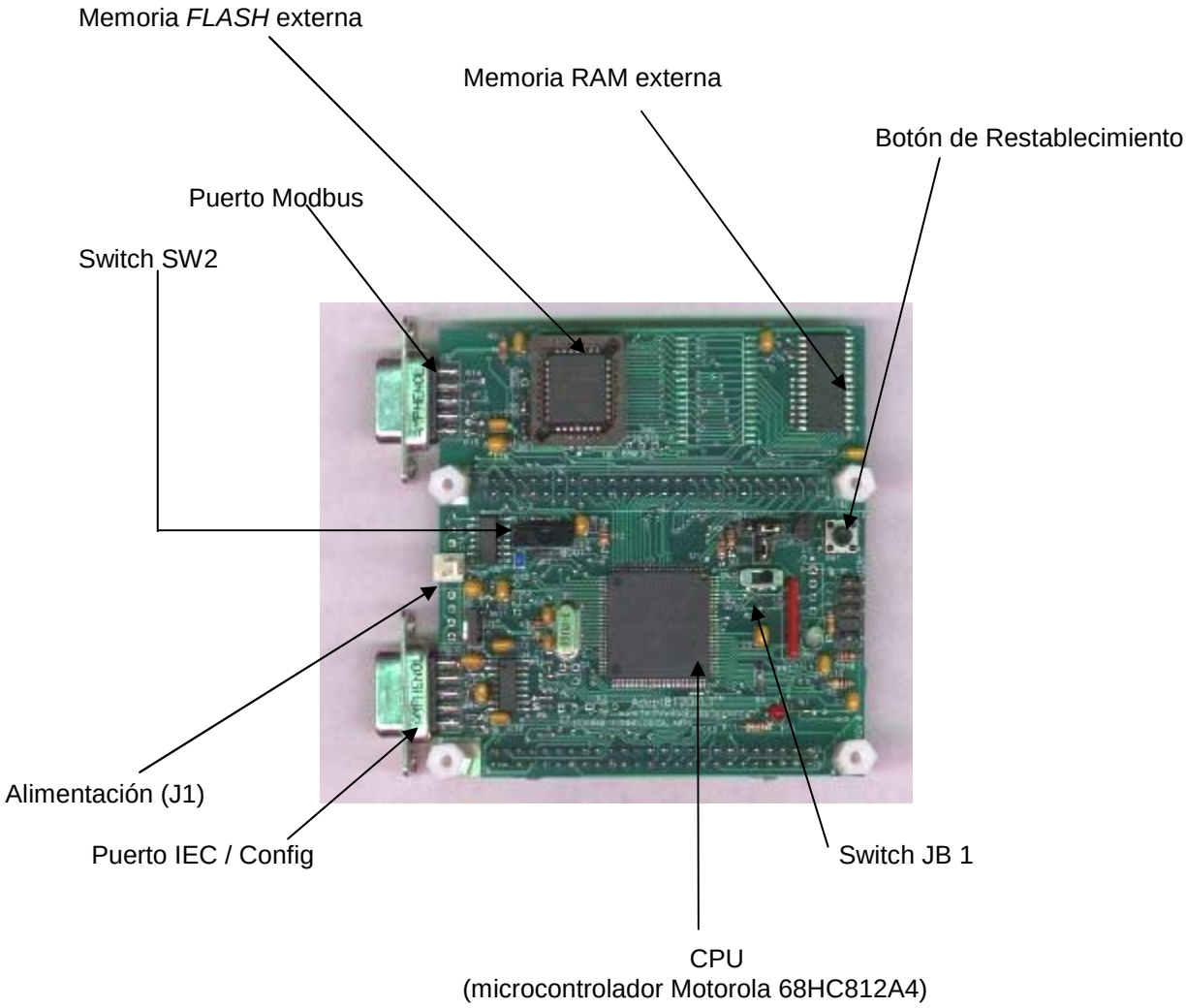


Figura A3.2 Partes de la tarjeta ADAPT812DXLT

Como puede observarse, la tarjeta cuenta con dos conectores tipo DB-9 para puertos RS-232. El primero de ellos, denominado Puerto IEC / Config tiene dos funciones: la de realizar la configuración del sistema por medio de una PC y la de conectarse con la estación maestra IEC. El segundo, llamado puerto Modbus se conecta a la UTR.

La tarjeta tiene dos interruptores: SW2 y JB1. El primero debe estar siempre en la posición *RUN,* mientras que el segundo debe estar siempre en la posición *EXP*. La tensión de alimentación debe suministrarse a través del conector J1, mediante el adaptador de la tarjeta, o un adaptador similar que suministre 9 V CD con el conector incluido con la tarjeta, teniendo presente el positivo se conecta al alambre rojo y el negativo al negro.

A3.3 Instrucciones de uso.

A3.3.1 Advertencias

A continuación se da una guía para la puesta en operación del sistema traductor. Es deseable que la persona que lo vaya a instalar y configurar tenga conocimientos sobre sistemas SCADA, así como de los protocolos Modbus e IEC 870-5-101.

 ¡ATENCIÓN! 

LA TARJETA CONTIENE DISPOSITIVOS SENSIBLES A LA ELECTRICIDAD ESTÁTICA (ESD). FAVOR SEGUIR LAS PRECAUCIONES CORRESPONDIENTES.

A3.3.2 Procedimientos preliminares

Antes de iniciar la instalación y configuración del traductor, es necesario averiguar los siguientes parámetros y tenerlos a mano para los pasos posteriores: cantidad de entradas digitales, entradas analógicas, salidas digitales, salidas analógicas activas en la UTR, dirección inicial para éstas (es necesario que dentro de cada grupo de entradas y salidas, éstas tengan direcciones relativas contiguas, por ejemplo: si hay tres entradas analógicas, la primera puede ubicarse en la dirección 13 si se quiere, pero las demás deberán ocupar las direcciones 14 y 15) dirección o Id Modbus de la UTR (en la UTR de I+D, ésta se programa mediante interruptores tipo DIP)

También es necesario tener los parámetros *Link Address y Common Address of ASDU* para el protocolo IEC 870, así como el número de octetos o bytes utilizados para cada uno de éstos y para el parámetro *Information Object Address*, que indica la dirección de los dispositivos E / S en el protocolo IEC. Al igual que en el caso anterior hay que indicar el valor inicial para cada tipo de dispositivo, pero dado que aquí se manejan direcciones absolutas, se deben ubicar en direcciones diferentes para que no ocurran choques (por ejemplo, si se ubicaron 16 entradas digitales a partir de la dirección 0, los siguientes dispositivos deben ubicarse a partir de la dirección 16) Al igual que para el protocolo Modbus, los dispositivos del mismo tipo deben ponerse en direcciones contiguas.

Es importante mencionar que las direcciones IOA se asignan mediante la cantidad de bits que ocupan: los dispositivos digitales ocupan 1 dirección cada uno, mientras que los analógicos ocupan 16 direcciones, ya que requieren 16 bits (se accesan mediante la dirección inicial del dispositivo, así si hay una entrada analógica ocupando las direcciones 50-65 y la siguiente se ubica en las direcciones 66-81, para escribir un nuevo valor en la primera debe escribirse en la dirección 50); a diferencia de la configuración Modbus, que como utiliza direcciones relativas y separadas por tipo, cada dispositivo ocupa 1 dirección y las direcciones pueden repetirse para dispositivos de diferente tipo. Hay que aclarar que el sistema traductor soporta tres octetos para el parámetro IOA por razones de compatibilidad, pero que internamente trabaja con dos; así, la dirección IOA máxima es 65535.

Por último, hay que tener también el parámetro Valor Umbral (Threshold Value) que representa el valor mínimo que debe cambiar un dispositivo analógico para reportar el cambio a la estación maestra. Se debe tener el valor normalizado (de 0 a 65535). Nótese que éste es solamente un valor inicial, ya que la estación maestra lo puede cambiar luego.

A3.3.3 Instalación y conexiones

Pasos a seguir:

1. Cerciórese que los interruptores SW2 y JB1 estén en las posiciones *RUN y EXP* respectivamente

2. Conecte el cable de alimentación en el conector J1 La tensión de alimentación debe estar entre 7V y 12V CD. Hay que tener presente que una vez realizada la configuración la alimentación no debe interrumpirse en ningún momento, de lo contrario se perderá la configuración almacenada y se reiniciará con la rutina de configuración

3. Conecte la UTR a la tarjeta mediante un cable RS-232 en el puerto Modbus. (Si la interfaz RS-232 de la UTR es tipo DCE, debe utilizarse un cable tipo null-modem, o sea, cruzado tx-rx y viceversa) Verifique que los paráameros de comunicación de la UTR (bits de datos, bits de parada, bits de paridad, velocidad=9600 bps) sean los mismos de la configuración Modbus.

4. Para realizar la configuración, conecte el puerto IEC / Config con el puerto serie de la PC mediante un cable RS-232. Una vez realizada la configuración, desconecte el cable del puerto IEC / Config y en su lugar conecte este puerto con el equipo que la comunica con la estación maestra IEC. Al igual que en el punto anterior, verifique que los parámetros de comunicación coincidan.

A3.3.4 Configuración del sistema

1. Una vez conectado el puerto serie de la PC con el puerto IEC / Config de la tarjeta, ejecute en la PC un programa de emulación de terminal (como *Hyperterminal*) y configúrelo para conectarse con el traductor directo por el puerto serie, a 9600 bps, con 8 bits de datos, paridad par, 2 bits de parada y sin control de flujo. (En *Hyperterminal:* haga una nueva conexión, en “Conectar usando:” seleccione “Directo a Com1” –o el puerto que esté usando-, oprima “Aceptar” y llene el cuadro de diálogo que aparece con los parámetros que se acaban de mencionar y oprima “Aceptar”. Si no está conectado, haga click en el botón de conectar para iniciar la configuración)

2. Oprima el botón de *reset* en la tarjeta. En la pantalla del emulador de terminal debe aparecer lo siguiente:

```
TraPo Config
Utilidad de Configuración para el traductor Modbus-IEC  v 0.1
   Desea ver la configuración actual? (S/N):
```

3. Oprima N para no ver la configuración. Se recomienda responder Sí a la pregunta: “¿Desea cargar valores por defecto?” y luego modificar los parámetros necesarios. Luego, cuando se le pregunte “¿Desea cambiar la configuración general? “ oprima S e introduzca los parámetros necesarios. Haga lo mismo cuando se le pregunte si desea cambiar la configuración Modbus o la configuración IEC 870.

4. Cuando todos los parámetros se hayan ingresado correctamente, oprima S cuando se le pregunte si desea salir de la utilidad de configuración. Ya puede entonces desconectar el traductor de la PC y enlazarlo con la estación maestra; no hay necesidad de ningún paso adicional.

A3.4 Guía para futuras modificaciones

El sistema es un prototipo de un traductor de protocolo; y como tal, está sujeto a modificaciones. Esta sección está dirigida especialmente a quiénes les sea asignada la tarea de mejorar este sistema.

Como recomendación inicial, se sugieren algunos aspectos que podrían mejorarse: capacidad para guardar la configuración en memoria no volátil, capacidad para configurar el sistema sin tener acceso físico a la tarjeta, capacidad para modificar los parámetros de comunicación en ambos puertos, soporte para más dispositivos de entrada / salida, optimización del proceso de *polling* o sondeo, capacidad para manejar varias UTR, soporte para más funciones del protocolo IEC y confección de un prototipo en una PCB.

Se sugiere revisar primero la documentación de la tarjeta ADAPT812DXLT, que puede obtenerse de http:///www.technologicalarts.com así como el manual del microcontrolador 68HC812A4 (MC68HC812A4.PDF) y el manual de referencia del HC12 (CPU12RM.PDF) Ambos documentos pueden descargarse del sitio web de Motorola. También puede revisarse el informe final de este proyecto, presentado a la empresa por el autor de este manual, donde se dan más detalles de la implementación del mismo.

Si se quiere trabajar sobre el primer aspecto mencionado (capacidad para guardar la configuración en memoria no volátil) una posible solución sería utilizar la memoria EEPROM que trae incluida el microcontrolador para guardar datos de configuración, sin embargo, deben tomarse ciertas precauciones para no borrar el *bootstrap*, que se encuentra en esta memoria. Para más detalles véase el manual de usuario de la tarjeta ADAP812DXLT

Para trabajar sobre el programa, el código fuente debería estar disponible donde obtuvo este documento. Está escrito en lenguaje ensamblador para Motorola. Se recomienda trabajar con el programa “MiniIDE” El archivo principal es “TraPo.asm”, aunque la rutina de configuración se dejó por aparte, en el archivo “config.inc”, junto con otras rutinas que se pusieron en otros archivos .inc. Nótese que trabaja con paginación de memoria, y que la rutina de configuración se puso en la página 0. Para cargar el programa en la tarjeta, síganse las instrucciones en el manual de la misma.

Apéndice A4: Especificaciones del sistema

A4.1 Especificaciones Generales:

CPU:	Motorola 68HC812A4
Memoria	32 K SRAM
	512 K *flash*
	4 K EEPROM
Capacidad de manejo de dispositivos:	112 entradas digitales
	112 salidas digitales
	32 entradas analógicas
	32 salidas analógicas

A4.2 Puerto Modbus

Interfaz:	RS-232 o compatible. Conector DB-9 hembra
Parámetros de Comunicación :	9600 8N1, 8N2, 8E2
Funciones Modbus Implementadas:	1,2,3,4,5 y 6

A4.3 Puerto IEC 870-5-101

Interfaz:	RS-232 o compatible. Conector DB-9 hembra
Modo:	No balanceado
Parámetros de Comunicación :	9600 8N1, 8N2, 8E2
Funciones Implementadas:	Dirección primaria: 0,3,4,9,10,11 Dirección secundaria: 0,1,8,9,11

ASDUs Implementados:

M_SP_NA_1	C_IC_NA_1
M_ME_NA_1	C_CI_NA_1
C_SC_NA_1	C_TS_NB_1
C_SE_NA_1	C_RP_NC_1
M_EI_NA_1	P_ME_NA_1

Apéndice A5: Interoperabilidad IEC 870-5-101

A5.1 Configuración de la red (parámetro específico para la red)

☒ Punto a punto
☒ Múltiple punto a punto
☐ Líneas redundantes
☒ Línea compartida multipunto
☒ Estrella multipunto

A5.2 Capa física (parámetro específico para la red)

A5.2.1 Velocidad de transmisión (dirección de control)

	V.24/V.28		X.24/X.27
☐	100 bits/s	☐	2 400 bits/s
☐	200 bits/s	☐	4 800 bits/s
☐	300 bits/s	☐	9 600 bits/s
☐	600 bits/s	☐	19 200 bits/s
☐	1 200 bits/s	☐	38 400 bits/s
☐	2 400 bits/s	☐	56 000 bits/s
☐	4 800 bits/s	☐	64 000 bits/s
☒	9 600 bits/s		

A5.2.2 Velocidad de transmisión (dirección de monitoreo)

	V.24/V.28		X.24/X.27
☐	100 bits/s	☐	2 400 bits/s
☐	200 bits/s	☐	4 800 bits/s
☐	300 bits/s	☐	9 600 bits/s
☐	600 bits/s	☐	19 200 bits/s
☐	1 200 bits/s	☐	38 400 bits/s
☐	2 400 bits/s	☐	56 000 bits/s
☐	4 800 bits/s	☐	64 000 bits/s
☒	9 600 bits/s		

A5.3 Capa de enlace (parámetro específico para la red)

Se utilizan exclusivamente el formato de trama FT 1.2, caracter sencillo 1 y tiempo de espera fijo.

A5.3.1 Procedimiento de transmisión

☐ Transmisión balanceada
☒ Transmisión no balanceada

A5.3.2 Tamaño de la trama

255 Longitud L máxima (número de octetos)

A5.3.3 Campo de dirección del enlace

☐ Ausente (sólo transmisión balanceada)
☒ Un octeto
☒ Dos octetos
☒ Estructurado
☒ No estructurado

A5.4 Capa de Aplicación

A5.4.1 Modo de transmisión para datos de aplicación

Se utiliza exclusivamente el Modo 1 (octeto menos significativo primero) definido en el punto 4.10 de la norma IEC 870-5-4.

A5.4.2 Dirección común del ASDU

☒ Un octeto ☒ Dos octetos

A5.4.3 Dirección de objeto de información

☒ Un octeto ☐ Estructurada
☒ Dos octetos ☒ No estructurada
☒ Tres octetos

A5.4.4 Causa de transmisión

☒ Un octeto ☐ Dos octetos (con dirección de origen)

A5.4.5 Selección de ASDUs estándar

Información de proceso en dirección de monitoreo

☒<1>:=Punto simple	M_SP_NA_1
☐<2>:=Punto simple c/ etiqueta de tiempo	M_SP_TA_1
☐<3>:=Punto doble	M_DP_TA_1
☐<4>:=Punto doble con etiqueta de tiempo	M_DP_TA_1
☐<5>:=Posición de paso	M_ST_NA_1
☐<6>:=Posición de paso con etiqueta de tiempo	M_ST_TA_1
☐<7>:=Cadena de 32 bits	M_BO_NA_1
☐<8>:=Cadena de 32 bits con etiqueta de tiempo	M_BO_TA_1
☒<9>:=Valor medido normalizado	M_ME_NA_1
☐<10>:=Valor medido normalizado con etiq. de t.	M_ME_TA_1
☐<11>:= Valor medido , con escala	M_ME_NB_1
☐<12>:=Valor medido , con escala y etiq. de t.	M_ME_TB_1
☐<13>:=Valor medido de punto flotante	M_ME_NC_1
☐<14>:= Valor medido de punto flotante y et. de t.	M_ME_TC_1
☐<15>:=Totales integrados	M_IT_NA_1
☐<16>:=Totales integrados con etiqueta de tiempo	M_IT_TA_1
☐<17>:=Evento de protección con et. de t.	M_EP_TA1
☐<18>:=Inicio empaquetado de eventos de protección con etiqueta de tiempo	M_EP_TB1
☐<19>:=Información empaquetada de salida de circuito de equipo de protección con etiqueta de tiempo	M_EP_TC_1
☐<20>:=Punto simple empaquetado c/ et. de t.	M_PS_NA_1
☐<21>:=Valor medido sin QD	M_ME_ND_1
☐<30>:=Punto simple con etiqueta CP56Time2a	M_SP_TB_1
☐<31>:=Punto doble con etiqueta CP56Time2a	M_DP_TB_1

Información de proceso en dirección de control	
☒<45>:=Instrucción simple	C_SC_NA_1
☐<46>:= Instrucción doble	C_DC_NA_1
☐<47>:= Instrucción de paso	C_RC_NA_1
☒<48>:=Puesta de valor normalizado	C_SE_NA_1
☐<49>:=Puesta de valor con escala	C_SC_NB_1
☐<50>:=Puesta de valor de punto flotante	C_SC_NC_1
☐<51>:=Cadena de 32 bits	C_BO_NA_1
Información de sistema en dirección de monitoreo	
☒<70>:=Fin de Iniciación	M_EI_NA_1
Información de sistema en dirección de control	
☒<100> := Interrogación general	C_IC_NA_1
☐<101> :=Interrogación de contador	C_CI_NA_1
☐<102> :=Instrucción de lectura	C_RD_NA_1
☐<103> :=Sincronización de reloj	C_CS_NA_1
☒<104> :=Instrucción de prueba	C_TS_NB_1
☒<105> :=Restablecer proceso	C_RP_NC_1
☐<106> :=Adquisición de retardo	C_CD_NA_1
Parámetro en dirección de control	
☒<110>:=Parámetro de valor medido normalizado (solamente valor umbral)	P_ME_NA_1
☐<111>:= Parámetro de valor medido con escala	P_ME_NB_1
☐<112>:= Parámetro de valor de punto flotante	P_ME_NC_1
☐<113>:=Activación de parámetro	P_AC_NC_1

Transferencia de Archivos

☐<120> :=Archivo listo F_FR_NA_1

☐<121> :=Sección lista F_SR_NA_1

☐<122> :=Llamar directorio, seleccionar archivo, llamar archivo, llarmar sección F_SC_NA_1

☐<123> :=Última sección, último segmento F_LS_NA_1

☐<124> :=Confirmar archivo, confirmar sección F_AF_NA_1

☐<125> := Segmento F_SG_NA_1

☐<126> := Directorio F_DR_TA_1

A5.5 Funciones de aplicación básicas

A5.5.1 Iniciación de estación

☒ Iniciación remota

A5.5.2 Interrogación general

☒ Global

☐ Grupo 1 ☐ Grupo 5 ☐ Grupo 9 ☐ Grupo 13

☐ Grupo 2 ☐ Grupo 6 ☐ Grupo 10 ☐ Grupo 14

☐ Grupo 3 ☐ Grupo 7 ☐ Grupo 11 ☐ Grupo 15

☐ Grupo 4 ☐ Grupo 8 ☐ Grupo 12 ☐ Grupo 16

A5.5.3 Sincronización de reloj

☐ Sincronización de reloj

A5.5.4 Transmisión de instrucción

☒ Instrucción directa (punto simple)

☒ Puesta de valor directa

☒ Sin definición adicional

☐ Pulso corto

☐ Pulso largo

☐ Salida persistente

- ☐ Seleccionar y ejecutar instrucción
- ☐ Seleccionar y ejecutar puesta de valor
- ☒ Uso de C-SE ACTTERM

A5.5.5 Transmisión de totales integrados

- ☐ Solicitud de contador
- ☐ Detención de contador sin restablecimiento
- ☐ Detención de contador con restablecimiento
- ☐ Restablecimiento de contador
- ☐ Solicitud general de contador
- ☐ Solicitar contadores de grupo 1
- ☐ Solicitar contadores de grupo 2
- ☐ Solicitar contadores de grupo 3
- ☐ Solicitar contadores de grupo 4

A5.5.5 Carga de parámetros

- ☒ Valor umbral
- ☐ Factor de filtrado
- ☐ Límite inferior para transmisión de valor medido
- ☐ Límite superior para transmisión de valor medido

A5.5.6 Activación de parámetros

- ☐ Act/desact de transmisión cíclica del objeto direccionado

A5.5.7 Transferencia de Archivos

Archivo de configuración de UTR

- ☐ Transferencia de archivo en dirección de monitor
- ☐ Transferencia de archivo en dirección de control.

Apéndice A6: Reporte de costos del proyecto

A6.1 Costo de realización del proyecto

En la tabla A6.1 se detalla el costo de los materiales necesarios para la realización del proyecto.

Tabla A6.3 Materiales Necesarios para el desarrollo del proyecto

Recurso	Cantidad	Costo Total
Tarjeta Adapt812DXLT con accesorios	1	$150
Cable de comunicación serie adicional	1	$10
Suministros de Oficina	varios	$15

En la tabla A6.2 se detalla el costo del equipo necesario para la realización del proyecto.

Tabla A6.2 Equipo necesario para el desarrollo del proyecto

Recurso	Cantidad	Costo Total
Computadora Personal	1	$1200
Escritorio	1	$ 200
Licencia Borland Delphi 5	1	$2000

En la tabla A6.3 se detalla el costo de los servicios personales que se requirieron para la ejecución del proyecto.

Tabla A6.3 Servicios Personales necesarios para el desarrollo del proyecto

Recurso	Cantidad	Costo Total
Estudiante	1	$1250

El costo total del proyecto fue de $4825

A6.2 Costo de producción de un dispositivo traductor

En la tabla A6.4 se da el costo de producción de un dispositivo traductor.

Tabla A6.4 Costo de producción de un dispositivo traductor

Recurso	Cantidad	Costo Total
Tarjeta Adapt812DXLT	1	$100

El costo de producir un dispositivo traductor es de $100

Apéndice A7: Abreviaturas y acrónimos utilizados en este documento

ASDU: Unidad de datos de servicio de aplicación (*Application service data unit*)

CENCE: Centro Nacional de Control de Energía.

IOA: Dirección de objeto de información (*Information Object Address*)

I+D: Centro de Servicios de Investigación y Desarrollo.

ICE: Instituto Costarricense de Electricidad y Telecomunicaciones.

IEC: Comisión Electrotécnica Internacional.

PLC: (*Programable Logic Controller*) Controlador Lógico Programable

RTU: *ver UTR*

SCADA: Control Supervisorio y adquisición de datos (*Supervisory control and data acquisition)*

UEN: Unidad Estratégica de Negocios.

UTR: Unidad Terminal Remota *(en inglés RTU, Remote Terminal Unit)*

Apéndice A8: Glosario de términos

Capa 1, Capa física: Medio físico y tecnologías de señalización para el intercambio de información entre dos o más sistemas.

Capa 2, Capa de enlace: Organización de tramas y control de acceso al medio para el intercambio de información entre dos o más sistemas.

Capa 7, Capa de aplicación: Procedimientos utilizados por el usuario para intercambiar datos que éste pueda entender, de un sistema a otro. Se apoyan en procedimientos de capas inferiores

Delphi: Lenguaje de programación visual basado en Pascal orientado a objetos. Desarrollado por Borland-Inprise.

Dirección de objeto de información: Dirección de cada punto de la UTR en el protocolo IEC 870-5-101

Estación maestra: Computadora que controla el sistema SCADA, recibe datos de las estaciones remotas (UTRs) y les envía instrucciones..

IEC 870-5-101 Protocolo de Comunicación abierto para sistemas SCADA

Microcontrolador: Microprocesador optimizado para labores de control electrónico, generalmente tiene integrados dispositivos útiles a estas tareas, como, contadores, temporizadores, generadores de PWM y cuentan con memoria integrada en el chip, suficiente como para hacer innecesario el uso de memoria de programa o datos externa.

Modbus: Protocolo de Comunicación abierto, de Modicon, utilizado para PLCs y sistemas SCADA

Ranger Estación maestra IEC 870-5-101, fabricada por ABB y utilizada en el Centro de Control de Energía del ICE para el sistema SCADA de generación eléctrica.

Request data: Instrucción en la que la estación maestra IEC solicita a la remota que envíe nuevos datos, ya sean dispositivos que hayan cambiado su valor, datos cíclicos o mensajes almacenados en el buffer.

Sistema SCADA Sistema de Control Supervisorio, administración y adquisición de datos. Se utiliza para monitorear y controlar uno o varios procesos físicos, generalmente distribuidos sobre un área geográfica relativamente grande.

Unidad de datos de servicio de aplicación: Mensaje de capa 7 utilizado por el protocolo IEC 870-5

Unidad terminal remota: Computador de aplicación especial orientado a la toma de datos y control de procesos físicos, subordinado a un computador central