Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Electrónica

Compañía Nacional de Fuerza y Luz S.A.

Centro de Control de Energía, Uruca

Diseño de una interfaz de comunicación aplicada a una red de monitoreo SCADA con protocolo ModBus y una DPU con protocolo ASCII. Sistema MISERC

Informe de Proyecto de Graduación para optar por el título de Ingeniero en Electrónica con el Grado Académico de Licenciatura

Adriano Enrique Espinosa Carreira

Costa Rica, Cartago, Mayo del 2003

Resumen

El Centro de Control de Energía (CCE) de la Compañía Nacional de Fuerza y Luz S.A. tiene implementado un sistema SCADA (Supervisory Control and Data Acquisition) para monitorear en forma automática la red de distribución de energía eléctrica en el área metropolitana de San José. El CCE cuenta con diversas unidades que se conectan a este sistema pero existen unidades como los DPU (Distribution Protection Unit, unidades de protección de distribución) serie 245D/445H/445V que su conexión no es permitida ya que se comunican por medio de SRC (Serial Relay Commands, comandos de interrupción seriales) un protocolo basado en transferencias de datos en formato ASCII, diferente al ModBus que usa el sistema SCADA.

El diseño y construcción de una tarjeta de comunicación con su sistema de control llamado MISERC, servirá de interfaz entre los diferentes protocolos permitiendo que los DPU se puedan comunicar al sistema SCADA. Esta tarjeta se comunicará con cada DPU, haciendo que trabajen como cualquier otro esclavo conectado al sistema SCADA.

Se implementó en su totalidad el protocolo ModBus, permitiendo que cualquier sistema que haga uso de este protocolo se pueda comunicar con los DPU y a su ves se implementaron 45 de los 64 SRC disponibles para el control de los DPU. Para el desarrollo de la tarjeta de comunicación se utilizó el más moderno microcontrolador de la familia PIC16F87X de 40 pines, de la empresa MICROCHIP. El micro PIC16F877 es el traductor de ambos protocolos y permite el control del flujo de datos desde y hacia el sistema SCADA de una manera rápida y económica ya que dicho microcontrolador no requiere de muchos periféricos para su funcionamiento.

Palabras clave: SCADA, DPU, SRC, ModBus, ASCII, PIC.

Summary

The Control center of Energy (CCE) of the Compañía Nacional de Fuerza y Luz S.A. has implemented a SCADA system (Supervisory Control and Data Acquisition) to monitor the electrical energy distribution network in the metropolitan area of San José. The CCE counts on diverse units that are connected to this system but the series 245D/445H/445V DPU (Distribution Protection Unit) does not allow the communication since they communicate by means of SRC (Serial Commands Relay) a protocol based on data transfers in ASCII format, different from the protocol that the SCADA system uses, ModBus.

The design and construction of a communication card with its system called MISERC, will serve as interface between the different protocols allowing the DPU to communicate with Modbus. This card will communicate with each DPU, causing that each DPU will works like any other slave connected to the SCADA system.

The ModBus protocol was implemented in its totality, allowing that any system that uses this protocol can communicate with the DPU. MISERC has 45 of the 64 SRC commands available for the control of the DPU. For the development of the communication card, it has been used the most modern microcontroller. Microcontroler PIC16F877 is the translator of both protocols and allows the control of the data flow from and towards the SCADA system. The use of this chip yields a fast and economic product since this microcontroller does not require external peripheral for his operation.

Key words: SCADA, DPU, SRC, ModBus, ASCII, PIC.

Indudablemente tengo que agradecer a muchas personas que sin la ayuda de ellos hubiera sido imposible la finalización de este proyecto. A la señora Grettel por permitirme invadir su oficina y dejarme desarmar por enésima vez la computadora del Tribunal, a Marco mejor conocido como Xytras, hermano me salvaste al prestarme el PIC16F877, porque el mío se quemó la semana anterior a la presentación del sistema.

Como olvidar a las personas de la Escuela de Electrónica que ya se cansaron de verme por los corredores. A la secretaria de la escuela de Electrónica, Sonia, gracias por todos los levantamientos de requisitos, sin ti, estuviera llevando aún Diseño Lógico. Profesor Paulino, le prometo que ya no haré más ruido en los laboratorios, gracias por tomarme en cuenta. Profesor Badilla, gracias por creer en mi, este proyecto si que se complico. Y sobre todo, agradezco la paciencia que como profesor guía me tuvo, Don Roberto le comento que ya terminé el proyecto, Y SI FUNCIONA.

Gracias a la CNFL en especial a Don Jose Salazar al brindarme las opciones para poder terminar el proyecto. Le cuento que ese DPU conoce Cartago, Tres Rios y todo San José.

Todo mi esfuerzo está dedicado a mi familia, aunque lejos, los llevo siempre en mi corazón.

Si no fuera por la fe que siempre han tenido en mí,

Si no fuera por que siempre me han ayudado,

Si no fuera por ustedes . . .

Gracias Papá, Gracias Mamá.

Índice General

Índice de Figuras

Índice de Tablas

Capítulo 1. Introducción

1.1. Descripción de la empresa

La Compañía Nacional de Fuerza y Luz, Sociedad Anónima, (CNFL S.A.) se dedica a satisfacer las necesidades de energía en forma oportuna, eficiente y eficaz, en el área bajo su servicio. Además desarrolla proyectos de inversión como el proveerse del equipo necesario para afrontar el crecimiento de la demanda de servicio eléctrico y también desarrolla acciones tendientes a la protección del ambiente y la conservación de la energía.

La misión empresarial de la CNFL S.A. es: “contribuir al desarrollo económico y social del país mediante el suministro adecuado de la energía eléctrica en el área servida por la empresa, en busca tanto de la excelencia en la utilización de los recursos, como en la calidad del servicio al cliente”.

Esta empresa fue pionera en los sistemas de alumbrados públicos a nivel mundial cuando el 9 de agosto de 1884 se puso en marcha la primera planta hidroeléctrica del país, situada en Barrio Aranjuez, se encendieron 25 luminarias en las principales vías públicas de la Ciudad de San José. De esta forma, nuestra capital se constituyó en la tercera ciudad del mundo y la primera en Latinoamérica, en ser iluminada gracias a la energía eléctrica. A partir de entonces, el desarrollo que empezó a tener floreciente actividad propició el surgimiento de pequeñas y grandes empresas dedicadas a la producción y distribución de energía, y años más tarde, la llegada al país de grandes transnacionales.

Fue en 1941 que la CNFL adquiere su estructura actual, como resultado de la fusión de tres empresas eléctricas: Compañía Nacional de Electricidad, The Costa Rican Electric Light and Traction Company Limited y Compañía Nacional Hidroeléctrica.

Su capital mayoritario, 98% de las acciones, fue comprado por el Instituto Costarricense de Electricidad, el ICE, el 30 de abril de 1968 a la Transnacional norteamericana Electric Bond and Share Co con lo cual se finiquitó su nacionalización. Actualmente la empresa, tras 61 años de operación cuenta con 1300 empleados y 342 000 clientes (los usuarios se estima que superan el millón).

El Centro de Control de Energía (CCE) es el departamento donde se desarrolló el proyecto. Está dedicado al monitoreo y control de la red de distribución de energía eléctrica, la cual cuenta con un tendido de redes que cubren 934 Km^2 del gran Área Metropolitana (Alajuela, Heredia, Cartago y San José).

Bajo la dirección del CCE se encuentra el Despacho de Carga, en el cual se realiza el monitoreo de la red de distribución por medio de un sistema SCADA (Supervisory Control and Data Acquisition), es decir un sistema de supervisión, control y adquisición de datos.

Este departamento se encarga de realizar proyectos que tienen que ver con la red, su mantenimiento y extensión. El CCE se encarga de agregar dispositivos adicionales a la red y de llevar la información desde los lugares donde dicha información es recopilada, hasta el Despacho de Carga, donde, por medio de pantallas y monitores relacionados con el sistema SCADA, los operadores observan el estado de los diferentes circuitos y realizan las operaciones de cierre y apertura de circuitos.

Para realizar sus funciones este departamento cuenta con recursos de hardware y software, tales como el sistema SCADA que le permite realizar el monitoreo desde numerosos Centros de Comunicación Remota, encargados de recopilar la información necesaria.

El CCE se ubica en La Uruca, contiguo a Automatra, donde laboran alrededor de 36 personas, entre ellas varios ingenieros, de los cuales se encuentra el Ingeniero Eric Esquivel quien es el Jefe del Centro de Control de Energía.

1.2. Definición del problema y su importancia

En el CCE utiliza un sistema SCADA con protocolo de comunicación ModBus. Con este sistema se lleva a cabo el monitoreo de la red de distribución de energía eléctrica.

El SCADA está conectado a una serie de dispositivos, de los que toma la información necesaria para llevar a cabo el monitoreo de dicha red. Estos dispositivos por ende, deberían tener un protocolo de comunicación ModBus, lo cual no es así en su totalidad, ya que hay dispositivos como los DPU que se comunican por medio del SRC. Por este motivo, se hace necesaria una tarjeta convertidora de protocolos, para poder conectar estos dispositivos a la red de monitoreo.

La ventaja de conectar el DPU al sistema SCADA consiste en que el DPU es un sensor de vital importancia tanto en lo que respecta a seguridad e información para el CEE. En lo que concierne a seguridad, los DPU se encargan de sensar fallas en la red de distribución de energía y abrir circuitos en caso de que se dé alguna falla. Los DPU son instrumentos muy importantes ya que protege equipos de alto costo tales como transformadores, alimentadores, etc. Además los DPU evitan que una

falla se propague causando la salida de funcionamiento de un sector más grande de la red eléctrica perjudicando a más personas.

Los DPU pueden ser interrogados por medio de un teclado numérico que viene en la carátula frontal de la unidad o por medio de un puerto serie que se comunica en ASCII. En el caso de que se quiera operar la unidad, por ejemplo, conocer una corriente o cerrar un circuito, un operador debería ir hasta el lugar donde se encuentran los DPU e ingresar los comandos en el teclado numérico. Con un medio de comunicación inalámbrica y un convertidor de protocolos se evita ese inconveniente, ya que permite que los operadores del Despacho de Carga operen los dispositivos a distancia y conozcan su estatus. Además el Centro de Control requiere una gran variedad de datos del dispositivo tales como corrientes, demandas pico, etc.

Haciendo uso del puerto serie de la DPU y una adecuada tarjeta convertidora de protocolos, se puede conectar esta unidad al sistema SCADA por medio de un Radio Modem, así desde la Estación Maestra se podrá tener control de la información de los DPU. De este modo el monitoreo de la red de distribución de energía eléctrica será más eficiente, evitando de este modo, tener que ir hasta donde se encuentra la unidad para poder extraer la información o para operar dicho dispositivo.

Actualmente para operar la DPU, se necesita hacer una instalación que aparte de costosa es poco eficiente ya que se tiene que utilizar una estación remota de poste, la cual tiene solo un canal de comunicación digital. Por este motivo no es posible interrogar por medio de comunicación digital a través de dicha estación la información almacenada en todos los DPU. También se tiene que realizar un cableado entre la estación remota de poste y la DPU el cual consta de muchos cables. La figura 1.1 muestra dicha instalación.

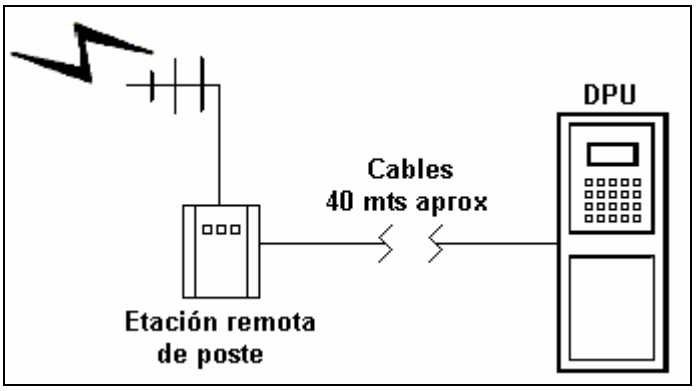


Figura 1.1 Instalación actual de una DPU a la red de monitoreo

Para mejorar la instalación que se muestra en la figura 1.1, se tiene que eliminar la estación remota (la cual tiene un costo que ronda los miles de dólares) y el cableado entre este y la DPU. Para poder realizar dicha tarea se tiene que utilizar la tarjeta convertidora de protocolos y un radio transmisor los cuales se deben instalar directamente a la DPU, la figura 1.2 muestra como se realizaría la comunicación entre la DPU y la red de monitoreo del CCE.

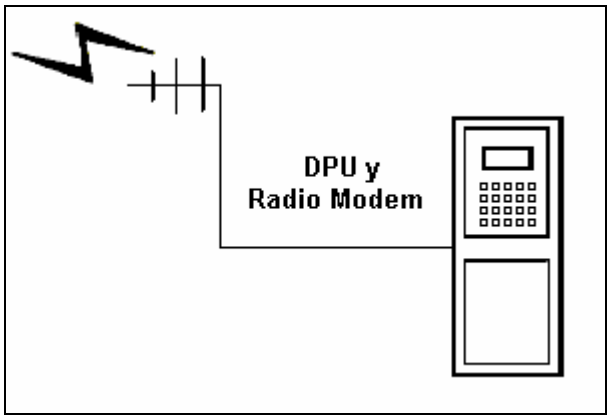


Figura 1.2 Instalación ideal entre la DPU y la red de monitoreo

La figura 1.3 es un diagrama de bloques que muestra con más detalle la instalación que se muestra en la figura 1.2. En este diagrama se evidencia la comunicación

bidireccional entre el sistema SCADA y la DPU, también se muestra la RTU (Radio Terminal Unit, Radio Modem) que es la unidad encargada de hacer la transmisión radial desde las subestaciones de distribución de energía donde están colocadas los DPU's al Despacho de Carga del CCE donde se encuentra el sistema SCADA.

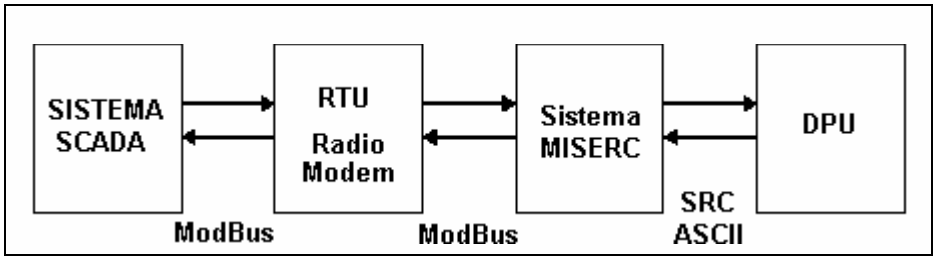


Figura 1.3 Diagrama de bloques entre el DPU y el sistema SCADA

1.3. Objetivos alcanzados

La finalización de este proyecto se logró al terminar los siguientes objetivos:

- Diseño del sistema de control llamado MISERC. El cual comprende:
    1. control de datos desde y hacia el sistema SCADA por medio del protocolo ModBus.
    2. control de datos desde y hacia el DPU por medio del estándar ASCII usando comandos SRC.
- Diseño de la tarjeta controladora, la cual ejecutará el firmware del sistema MISERC.
- La transferencia tecnológica al CCE sobre el sistema MISERC

Capítulo 2. Antecedentes

2.1. Problema a resolver

El CCE tiene las unidades DPU que sirven para el control de voltajes y corrientes de la red eléctrica del área metropolitana. Las unidades DPU serie 245D/445H/445V al ser muy viejas, no contaron con el desarrollo de una interfaz de protocolos de alto nivel como lo es el protocolo ModBus. La empresa ABB, creadora de los DPU serie 245D/445H/445V tiene tarjetas que manejan el protocolo ModBus, pero al ser muy costosas representan un gasto innecesario para unidades muy viejas.

En anteriores proyectos que respondían a ser una interfaz entre protocolos, se utilizaron componentes parecidos a este proyecto y una filosofía de programación muy diferente limitando su posterior desarrollo o mantenimiento del sistema creado.

Los aspectos técnicos que este proyecto mejoró en gran medida en comparación a los otros fueron:

- Se redujo en un circuito integrado los componentes finales de la tarjeta controladora.
- Se implementó interruptores DIP para la configuración automática de la dirección del DPU esclavo, eliminando la reprogramación de cada DPU.
- Se crearon las rutinas de CRC (Cyclic Redundancy Check, revisado redundante cíclico) del revisado y generación de errores del protocolo ModBus.
- Se implementaron 4 tipos de excepciones o errores a nivel del protocolo ModBus.

- Se implementaron 2 tipos más de chequeo de error del protocolo ModBus, esto fue posible ya que solo se implementaron funciones básicas del protocolo ModBus.
- Se implementó correctamente el protocolo ModBus en 6 funciones.
- Se implementaron 45 comandos SRC del DPU.

Al tener que programar en ensamblador el sistema MISERC se tomó en cuenta la filosofía de procedimientos modulares, permitiendo un rápido mantenimiento.

Se dejo abierto la posibilidad de implementar una reprogramación de la tarjeta "en caliente" por medio del protocolo ModBus con solo crear la función de transferencia de archivos. No se implemento esta función ya que excede los objetivos planteados.

2.2. Requerimientos de la empresa

En reuniones celebradas por parte del jefe del CCE y el encargado del Sistema SCADA se determinaron los siguientes requerimientos que debe contar el sistema de control:

Construir una tarjeta controladora de protocolos.

Implementar 25 comandos SRC[1] del DPU

- Lecturas de corrientes (15 comandos)
- Lecturas y escrituras de recierres. (7 comandos)
- Comando de abrir y cerrar (3 comandos)

Realizar la transferencia tecnológica al CCE sobre el sistema a elaborar

[1] Ver Apéndice A.2

2.3. Solución propuesta

Al analizar diferentes proyectos que anteriormente se habían elaborado para resolver el mismo problema de tener diferentes protocolos, se pudo determinar las mejoras que se le agregaron a este proyecto y a la vez implementar una nueva filosofía de programación en ensamblador que sea totalmente paramétrica y modular para que en el futuro sea más fácil darle mantenimiento al sistema elaborado. Por tanto el diseño del programa de interfaz del microcontrolador se rediseño en su totalidad partiendo de un sistema de programación nuevo y una tarjeta controladora mejorada.

La idea principal fue la de desarrollar una tarjeta económica y de alto rendimiento que sirva de interfaz o puente entre el protocolo ModBus del sistema SCADA y los SRC en su formato ASCII de los DPU. Al utilizar componentes de control económicos y que no dependan de tantos periféricos para su funcionamiento permitirá la mejor solución al problema. Se investigó en el mercado internacional sobre microcontroladores económicos y a la ves que representen un alto rendimiento de procesamiento, dando como resultado el microcontrolador PIC16F877[2] en su modelo de 40 pines, 20 megahertz, de la empresa MICROCHIP. Teniendo como base un microcontrolador que tiene un rendimiento de 5 MIPS (millones de instrucciones por segundos) a un costo de US$ 12.00 y un conjunto de instrucciones de solo 35 comandos, se empezó a construir la tarjeta controladora y el firmware de interfaz, MISERC.

El uso de este microcontrolador en especial resultó en un conjunto de integrados periféricos mínimos, por tanto el costo final del sistema en cuanto a hardware es menos US$ 100.00. Este precio es ínfimo en comparación al costo (alrededor de los

[2] Ver Anexo B.1

miles de dólares) de una tarjeta de la empresa ABB que soporte el protocolo ModBus-SRC.

Al utilizar el microcontrolador PIC16F877, la empresa MICROCHIP ofrece además del ensamblador nativo del microcontrolador un lenguaje de alto nivel, C. Otras empresas además de soportar el ensamblador nativo del PIC ofrecen un lenguaje más sencillo que C, este lenguaje es BASIC. Para desarrollar el sistema se tomó en cuenta esta variedad de lenguajes seleccionando el ensamblador nativo del PIC para programar el sistema de control del PIC. Esto debido a que en ensamblador todas las rutinas se elaboran con un alto rendimiento, caso contrario pasaría si se utilizara un lenguaje de alto nivel como C o BASIC.

2.4. Protocolos de comunicación

2.4.1. Protocolo ModBus

El protocolo ModBus está definido como una PDU (Protocol Data Unit, unidad de protocolo de información) que son independientes de las capas de comunicación. La estructura del protocolo contiene campos que hacen referencia a la información que se está enviando.

Para que el protocolo ModBus se pueda usar en diferentes redes se introdujo campos adicionales lo que forman la ADU (Application Data Unit, unidad de aplicación de información). Ver Figura 2.1.

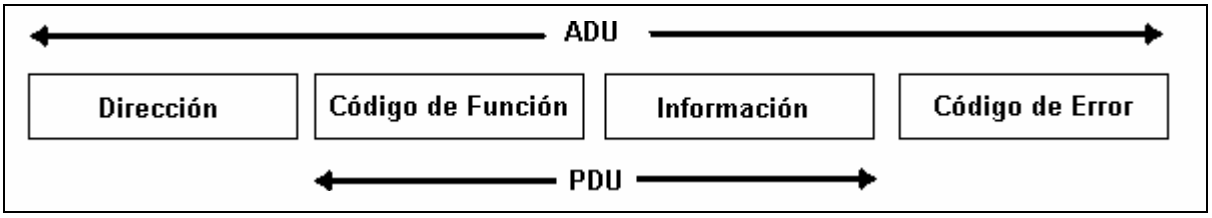


Figura 2.1 Ventana general ModBus

La empresa estadounidense MODICON, creadora del protocolo ModBus es la que fija los estándares que pueda tener dicho protocolo a nivel internacional. El ADU del ModBus es construido por el cliente que inicia una transacción ModBus.

La función le indica al servidor que tipo de acción tomar. Este tipo de comunicación es conocido como maestro esclavo, siendo el maestro el cliente y el esclavo el servidor. Ambos, el cliente como el servidor, son unidades de comunicación que de antemano tienen una velocidad de transferencia de datos predefinida.

El primer campo que esta definido por un byte (Ver Figura 2.1) indica la dirección del servidor al cual se está accesando. Un byte da por resultado 256 direcciones, por tanto s pueden tener 256 servidores conectados a la red usando protocolo ModBus.

El siguiente byte indica el código de función a ejecutar por el servidor, este también da una combinación de 256 comandos a ejecutar. El tercer campo el cual está definido por dos bytes indica la información que el cliente le manda al servidor para que este se actualice.

El último campo, que son dos bytes es donde se ubica un paquete de revisado de error. . El protocolo ModBus usa el CRC16. El CRC16 es un método de revisado de error de 16 bits que pretende eliminar que un ADU errado por diversos factores llegue a ejecutarse en el servidor.

El servidor cuando revisa el ADU y se da cuanta de que está correcto, empieza a ejecutar los comandos necesarios para luego enviar al cliente una respuesta basada en el mismo formato.

El protocolo ModBus además de manejar el revisado de errores por medio del CRC16, tiene errores de excepciones. Los principales son:

- Código 1: Función no implementada.
- Código 2: Registro diseccionado esta fuera de rango.
- Código 3: Estructura del ADU es errada.
- Código 4: Error al ejecutar la función por el lado del servidor

El modelo de información que maneja el protocolo ModBus está basado en 4 tablas. Ver tabla 2.1

Tabla 2.1 Tipos de información del protocolo ModBus

Tablas Primarias	Tipo de Objeto	Tipo de acceso
Entradas discretas	Largo de un bit	Solo lectura
Coils	Largo de un bit	Lectura y escritura
Registro de Entrada	Largo de 16 bits	Solo lectura
Registros retenedores	Largo de 16 bits	Lectura y escritura

La cantidad de registros suman 65536, esto debido a que el tamaño del paquete que indica la dirección del registro es de 16 bits; independiente del largo del tipo de objeto.

El protocolo ModBus está predefinido con 127 funciones. La mayoría de las funciones predefinidas son de conocimiento público. Las principales funciones, las cuales fueron implementadas en este proyecto son las de lectura y escritura de

registros. Ver el capítulo 5 una explicación más detallada de las funciones implementadas en el sistema MISERC.

2.4.2. Protocolo SRC

El protocolo SRC esta basado en el envío de datos en forma ASCII a la terminal remota que en este caso es el DPU. La estructura de datos esta compuesta por caracteres alfanuméricos que serán interpretados y ejecutados por el DPU. Una lista de los comandos que el DPU puede ejecutar verlos en el apéndice A2.

Para iniciar una transferencia SRC es necesario que el DPU este sincronizado a la misma velocidad que el cliente. Al empezar a mandar los datos, es preferible iniciar con el caractér 17 decimal de la tabla ASCII. Este caractér una vez enviado por primera ves, no es necesario enviarlo por cada comando que se transmita al DPU. Este carácter le indica al DPU que inicie la transmisión de datos al cliente. Luego de enviar el caractér 17 se procede a enviar los caracteres ASCII que forman el comando a ejecutar.

Al no tener un largo definido en la extensión del comando a ejecutar, el DPU se quedará esperando el carácter 13 decimal que le indica que se terminó de enviar el comando. Acto seguido el DPU empieza la transferencia de datos o instrucciones en formato ASCII del comando ejecutado.

El SRC está implementado por el protocolo RS232, el cuál pertenece a la capa física de los niveles de comunicación.

2.4.3. Protocolo RS232C

En el nivel más bajo de las capas de comunicación se encuentra la capa física. En este nivel se encuentra el protocolo RS232, el cual está definido por niveles de voltaje que van de -12voltios para un nivel lógico de 1 hasta los +12voltios para un nivel lógico de un 0.

La estructura del protocolo RS232 esta definido por la transmisión de un bit de inicio, 7 u 8 bits de datos, 1 bit de paridad (opcional), y de 1 a 2 bits de finalización de transmisión. Ver figura 2.2.

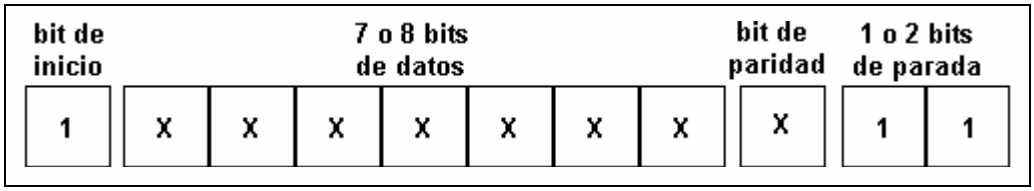


Figura 2.2 Estructura del protocolo RS232

Los puertos de entrada y salida del protocolo RS232 tienen cantidades diferentes de pines de entrada/salida para los niveles de voltajes transmitidos.

El estándar RS232C tiene 25 pines, los cuales los principales son los pines enumerados que van del pin 1 al pin 7. Ver figura 2.3.

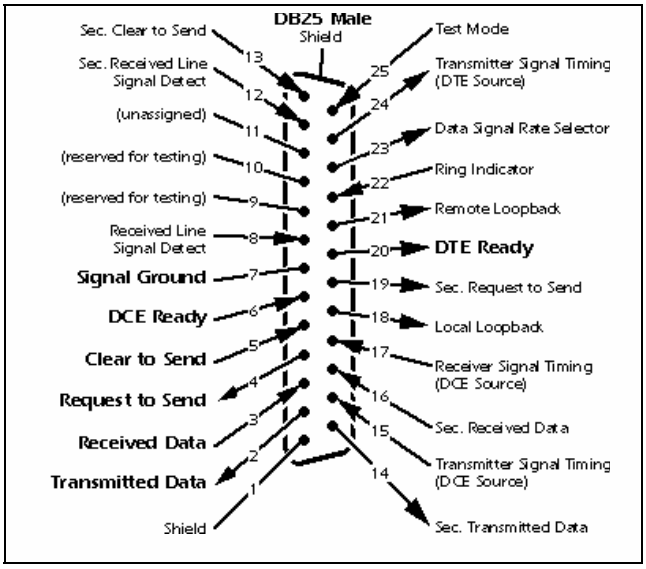


Figura 2.3 Ubicación de los pines 1 al 7

Para permitir una conexión con la menor cantidad de pines entre dos terminales remotas se debe hacer una conexión DTE (Data Terminal Equipment, equipo terminal de información). Está conexión fue la utilizada en la realización del proyecto. Ver figura 2.4. Con está configuración solo se necesitan 3 pines de I/O (entrada/salida). El pin TX es por el cual se transmite los datos, RX por el cual se reciben los datos, GND es el nivel de voltaje 0 el cual sirve de referencia de voltajes entre las terminales remotas.

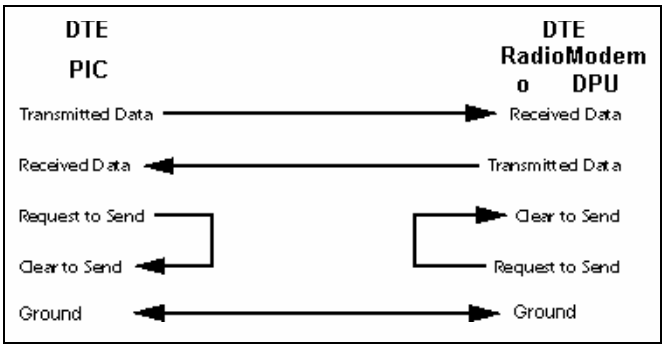


Figura 2.4 Conexión para dos DTE

Capítulo 3. Procedimiento metodológico

La siguiente metodología fue la empleada para cumplir con los objetivos expuestos:

1. Recopilación de información de los protocolos ModBus y ASCII:
Esta etapa se llevará a cabo con ayuda de los manuales presentes en el Centro de Control de Energía y con apoyo del Internet en caso de que el material presente en la empresa no sea suficiente.

2. Reconocimiento del sistema SCADA utilizado en el CCE:
Esta etapa se llevará a cabo con ayuda de los manuales presentes en el Centro de Control de Energía y con la ayuda de los operarios del sistema, también se verá la red de monitoreo funcionando por lo que se deberá hablar con el operario encargado de manipular dicha red.

3. Etapa de análisis del funcionamiento de la Unidad de Protección de Distribución:
Esta parte se llevará a cabo utilizando el manual Circuit-Shield Distribution Protection Unit (DPU) de ABB (Asea Brown Boveri), con la ayuda de otros manuales que pueda tener el CCE y con la información que se encuentre en la Web.

4. Estudio y análisis de los comandos que debe enviar y recibir el sistema de comunicación.
Para llevar a cabo esta etapa se debe analizar con el Jefe del Centro de Control de Energía cuales son los datos más importantes que se necesitan de la DPU para que estos sean visualizados en la red de monitoreo.

5. Estudio de las características de hardware del microcontrolador (PIC16F877):
Esta etapa se llevará a cabo con ayuda de los catálogos de microprocesadores de Microchip y con información de los CD's que Microchip provee a los interesados.

6. Etapa de diseño de la tarjeta convertidora de protocolos

El diseño se implementará en un “proto board” y se alambraran los componentes necesarios para poder hacer las pruebas necesarias de la interfaz de comunicación.

7. Probar el diseño de la tarjeta convertidora de protocolos

Una vez concluido el diseño y ensamblaje de los componentes en la “proto board”, se probará el diseño con un programa pequeño de comunicación previamente elaborado. Esto con la finalidad de comprobar el buen funcionamiento de los componentes. Tiempo 1 semana.

8. Etapa para diseñar e implementar el firmware del sistema.

Se diseña el programa para la tarjeta convertidora de protocolos que incluyen las rutinas del sistema que le permitan a la Estación Maestra recibir y enviar información a la DPU.

9. Etapa de planificación para llevar a cabo pruebas a las nuevas implementaciones tanto de software como de hardware:

Para llevar a cabo este punto hay que hacer un estudio preliminar de los resultados esperados y determinar cuales son los puntos críticos que se llevarán a cabo.

10. Realización de las pruebas del sistema:

Para llevar a cabo este punto se necesitará el uso del grabador del microcontrolador PIC 16F877A además de la DPU para probar las características implementadas.

11. Etapa de análisis de los resultados obtenidos en las pruebas del sistema desarrollado:

Después de haber realizado las pruebas se hará uso de bases de datos con la información que tenga la empresa de mediciones hechas anteriormente con el DPU, esto con el fin de comprobar el funcionamiento de las implementaciones que se hayan aplicado al sistema.

12. Etapa de corrección del ¨Firmware¨ del sistema:

De existir alguna falla en el sistema se procederá a realizar esta etapa. Aquí se realiza el rediseño de las rutinas y programas implementados. Se utilizará el mismo lenguaje para efectuar las correcciones al programa del microcontrolador.

13. Implementar un prototipo del sistema (construir el impreso de la tarjeta):
En esta etapa se realizara un prototipo del sistema, para llevarlo a cabo se ocuparan los chips utilizados en las pruebas para montarlos en un circuito impreso.

14. Etapa para llevar a cabo las pruebas finales del sistema:
Esta etapa se desarrollará una vez identificadas y corregidas las fallas en el sistema. Al igual que en el punto 16 de la metodología se hará uso de la DPU para probar las características implementadas.

15. Preparar la documentación final del proyecto:
Esta etapa se desarrollará una vez que se haya logrado poner el sistema en funcionamiento al 100%. La documentación final son todos lo documentos que se entregarán a la empresa, desde el informe final hasta los diversos manuales de usuario que se utilicen en el sistema.

16. Presentación del proyecto a la empresa y las capacitaciones necesarias para el buen uso del mismo:
Se realizará una presentación a la empresa en el ámbito administrativo y se darán capacitaciones del sistema a los operarios que se encargarán de manipular el sistema.

Capítulo 4. Descripción del hardware utilizado

4.1. Diagrama general de bloques

El sistema de comunicación se encontrará en el medio de la comunicación entre el sistema SCADA y el DPU. Esta configuración le permite al sistema MISERC realizar la función de interfaz o bien, de traductor e interpretador de las funciones que el sistema SCADA envíe por medio del protocolo ModBus para procesarlas y enviar los comandos en formato SRC al DPU. De forma análoga, el DPU mandará datos en formato ASCII al sistema MISERC que este procesará construyendo un paquete ModBus para su envío al sistema SCADA. Ver figura 1.3.

La comunicación entre el sistema SCADA y el sistema MISERC es half duplex. Esto quiere decir que el sistema MISERC jamás responderá cuando se este recibiendo datos del sistema SCADA. De igual forma, la comunicación entre el sistema MISERC y el DPU es half duplex.

4.2. Diagrama de buses de la tarjeta de comunicación

La tarjeta de comunicación contiene un circuito integrado el cual le permite comunicarse con solo un DTE a la vez. Esto debido a que el microcontrolador PIC16F877 solo posee un puerto de comunicación (USART). Para permitir la comunicación entre dos DTE se utilizó la lógica de multiplexación de puertos por medio de un habilitador de buses de comunicación. El CI 74LS125 al tener 4 buffers permitió la disponibilidad de habilitar o deshabilitar dos buffers a la ves y lograr la comunicación por medio de los pines RX, TX del PIC16F877 y el DPU o bien al sistema SCADA. En esta configuración que solo necesita un solo CI para la

multiplexación de buses necesita 2 líneas de control del PIC16F877 para seleccionar el par de buffers del 74LS125 que se habiliten o deshabiliten. Ver figura 4.1.

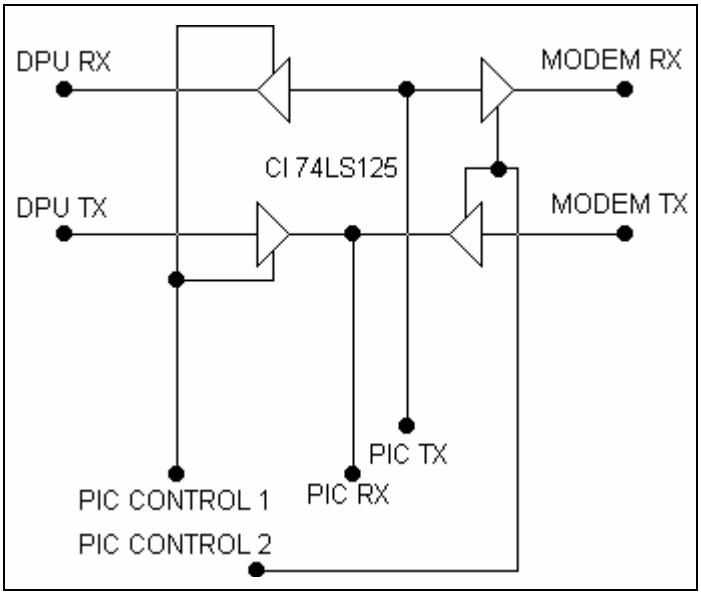


Figura 4.1 Multiplexación de los buses de comunicación

4.3. Diagrama esquemático de la tarjeta de comunicación

El diseño final de la tarjeta de comunicación al tener una cantidad mínima de componentes periféricos permite que la depuración del hardware sea muy sencilla y a la vez representa un costo muy bajo para su construcción.

A diferencia de los trabajos anteriores que resolvían de manera programada la selección de la dirección de la tarjeta, este proyecto incluyó la selección de la dirección por medio de un DIP SWITCH de 8 contactos. Está configuración permite la selección de 256 direcciones para cada DPU que posea una tarjeta con el sistema MISERC. Esta configuración soporta el formato de 8 bits de direcciones del protocolo ModBus.

Capítulo 5. Descripción del software utilizado

5.1. Lenguaje de programación

Para el desarrollo del sistema MISERC se utilizó el lenguaje ensamblador para PIC que provee la empresa MICROCHIP como lenguaje por defecto para el microcontrolador PIC16F877. El lenguaje esta implementado en un software de desarrollo que está en su versión 6.20 del programa MPLAB IDE de la misma empresa. Existen lenguajes de alto nivel como el BASIC para PIC y C para PIC que ofrecen la versatilidad que lenguajes de muy bajo nivel como el lenguaje ensamblador no posee. Pero al dar un mayor rendimiento, se eligió el lenguaje ensamblador nativo del nativo como lenguaje de programación.

El MPLAB IDE además de ofrecer una estructura de revisión de sintaxis de lo programado, ofrece un simulador del programa como si el verdadero microcontrolador lo estuviese ejecutando.

La diferencias entre lenguajes de alto nivel y los de bajo nivel ofrecen entre si ventajas y desventajas. La principal ventaja de un lenguaje se convierte en la mayor desventaja de otro. En este caso la rapidez con se escribe un programa en un lenguaje de alto nivel (BASIC, C) produce en lenguaje máquina un programa muy extenso, caso contrario pasa que el mismo programa escrito en un lenguaje de bajo nivel (ensamblador) produce un programa más compacto pero más difícil de entender. La ventaja de tener un programa más compacto ofrece en este caso una mayor rapidez en las ejecuciones que tiene que procesar el microcontrolador. El tiempo de ejecución es parte vital en los procesos que manejan los microcontroladores. Además el espacio al ser muy limitados (costosos para el

programador) en los microcontroladores nos quita la alternativa de darnos el lujo de programarlos en lenguajes de alto nivel.

5.2. Programa de comunicación RS232

En el mercado existen muchos programas de comunicación básica que soportan el protocolo RS232 para las computadoras personales (PC) los cuales ofrecen muchas características de configuración. El sistema operativo Windows en sus diferentes versiones tiene el programa Hyper Terminal que ofrece la conectividad bajo RS232.

Para el desarrollo del proyecto se utilizó el programa RS232 diseñado en lenguaje Pascal v7.0 para DOS usando rutinas en ensamblador para procesadores x86 INTEL. El programa fue construido anteriormente a la realización de este proyecto. Este programa al no tener características de configuración complejas le ofrece al usuario una vista de lo que llega a la USART de la PC en formato ASCII sin procesar y de lo que la PC envía por el puerto de comunicación también en formato ASCII sin procesar. Esta característica le ofrece al usuario la posibilidad de detectar errores de comunicación de muy bajo nivel sin tener que usar un osciloscopio. Ver figura 5.1.

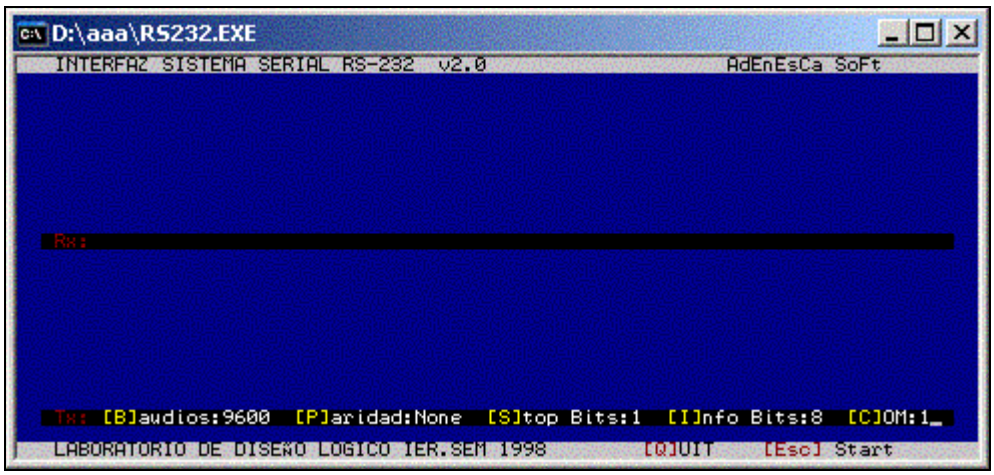


Figura 5.1 Vista del programa RS232

5.3. Programa de comunicación ModBus

Para establecer una comunicación usando el protocolo ModBus y verificar el envío de paquetes con el revisado de errores CRC16, se utilizó el MODBUS POLL.

La versión que se bajó de Internet de la dirección www.wittecom.com es una versión freeware. Esta versión tiene una limitante de usa por 30 días. El programa ofrece un revisado del protocolo ModBus al sistema MISERC como si fuera el sistema SCADA. Esto provee una herramienta más de depuración para el desarrollo del software. Debido a que el programa soporta por completo el protocolo ModBus, se puede dar un revisado de todas las funciones desarrolladas en el sistema MISERC. Ver figura 5.2.

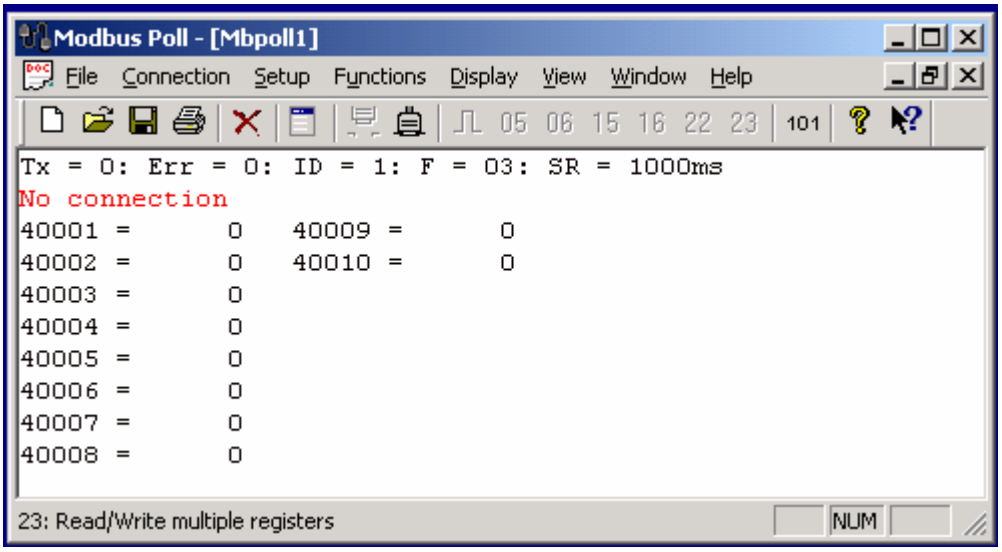


Figura 5.2 Vista del programa MODBUS POLL

5.4. Diseño del esquemático de la tarjeta

El esquemático, o bien el diseño final de la tarjeta de comunicación fue realizada en la versión Trial del software Protell 99 SP6.1. Este programa ofrece una herramienta de diseño parecida a los programas de diseño gráfico como el AUTO CAD. Entre

sus ventajas esta la opción de visualizar la tarjeta en una vista de tercera dimensión a tamaño real. El PCB, el archivo resultante del esquemático de la tarjeta indica la posición lógica de los componentes que se usan en el desarrollo del hardware. Ver Figura 5.3.

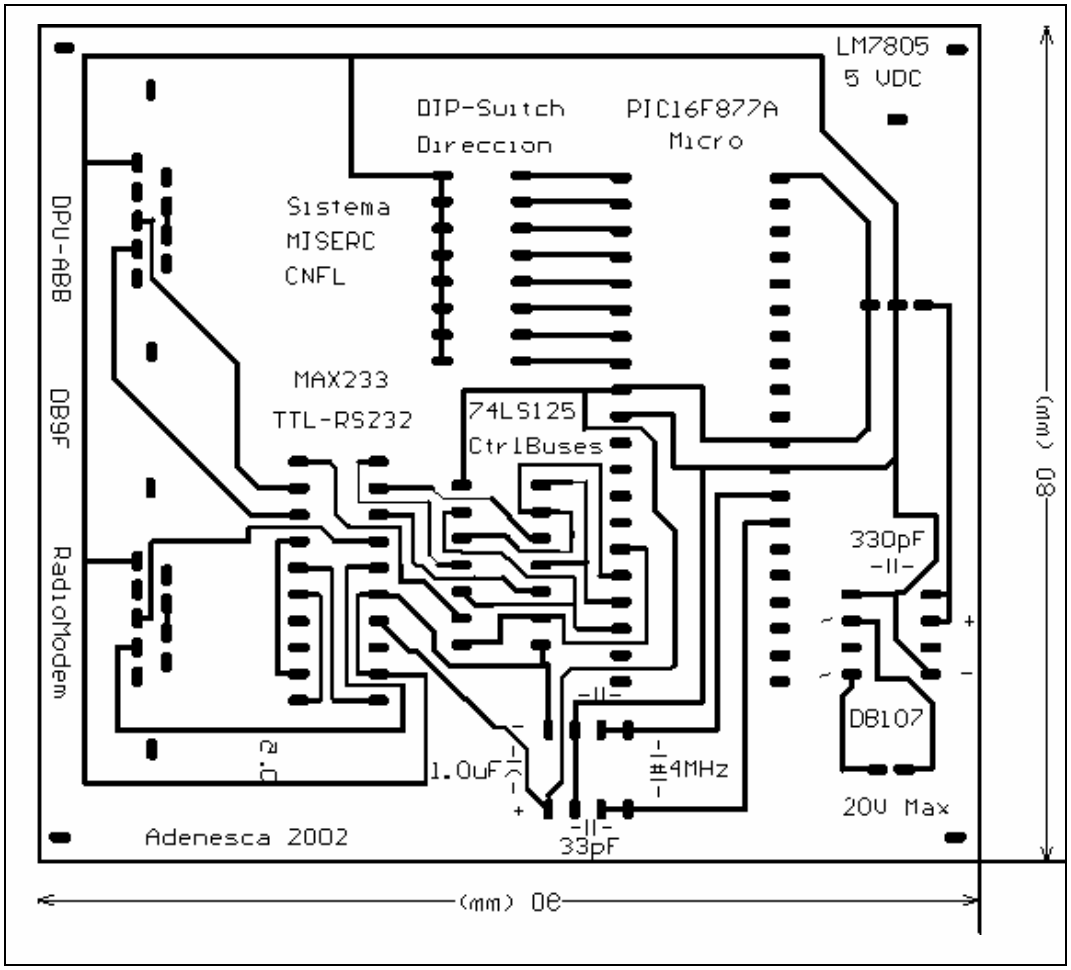


Figura 5.3 Impreso de la tarjeta de comunicación

Capítulo 6. Análisis y resultados

6.1. Explicación del diseño de hardware

El diseño de la tarjeta como se ha explicado anteriormente, fue diseñado tomando en cuenta consideraciones de desempeño, economía y versatilidad de programación.

El desempeño está relacionado con el caudal de procesamiento que el microcontrolador puede ejecutar. Para realizar tarjetas en serie la mejor opción estaría representada por tarjetas con un costo menor.

La versatilidad de programación fue el factor más preponderante en cuanto a la elección de este diseño de hardware. Debido a que el lenguaje de programación del PIC esta muy relacionado a la estructura interna del CI, este representó una curva de aprendizaje muy corta dando como resultado que el problema de aprender un nuevo lenguaje y estructura interna no fueran problemas a resolver para el proyecto.

El conjunto de circuitos integrados que formarán parte de los circuitos integrados periféricos al PIC16F877 serán: LM7805[3] (fuente de 5 voltios), 74LS125[4] (control de puertos), MAX233A[5] (convertidor TTL-RS232). Se utilizarán dos DB9 (conectores puerto serie), un DIP-Switch (interruptores), 2 capacitores, un puerto de entrada de

[3] Ver Anexo B.2

[4] Ver Anexo B.3

[5] Ver Anexo B.4

alimentación de 20 voltios de dos pines, un puerto de dos pines para el cristal y el impreso[6].

6.2. Explicación del diseño de software

El MISERC, **M**odBus **I**nterface to **SE**rial **R**elay **C**ommands es el sistema desarrollado a solventar la carencia de comunicación entre diferentes protocolos.

El MISERC al ser implementado en ensamblador puede ser algo muy tedioso de depurar. Por esa razón se implementó la filosofía de módulos con parámetros a lo largo del programa. Cada módulo o procedimiento tiene una función en particular y específica por tanto si se desea ampliar las funciones del sistema se puede hacer tomando en cuenta que ya existen funciones básicas que procesan procedimientos de revisado de errores, selección de puertos etc. La idea es simplificar un posible mantenimiento por una persona que desconozca el proyecto a profundidad.

El programa principal, que a continuación esta escrito:

```
;*********************************************************************BEGIN*BEGIN
BEGIN           CALL  PICINI        ; INICIALIZA EL PIC
INI         CALL  DPUINI            ; INICIALIZA LA TX/RX CON EL DPU
RXM         CALL  MODEMRX  ; START RX DE PAQUETES DE MODEM
            CALL  CRCCHK    ; CHK=CRC
            BTFSC       BOOL,CRC
            GOTO RXM
            CALL  EXC1y3       ; CHK=DIR EXC(1,3)
            BTFSC       BOOL,EXC
            GOTO RXM
            CALL  EXC2         ; CHK=EXC(2) BOOL[RdWr]
```

[6] Ver Figura 5.3

```
                BTFSC       BOOL,EXC
                GOTO RXM
                CALL  DPURdPro   ; READ & PROCESS DATA FROM DPU
                BTFSC       BOOL,EXC
                GOTO INI
                CALL  MODEMTX
                GOTO INI
 END
;*****************************************************************END*END*END
```

minimiza las llamadas a procedimientos dentro de otro y permite ver el flujo directo del proceso de los datos por parte del sistema MISERC.

El procedimiento PICINI configura el microcontrolador tomando en cuenta salidas, entradas, velocidad de transmisión, variables de entorno global. El procedimiento DPUINI, inicializa la transmición del sistema MISERC al DPU. MODEMRX se encarga de recibir los paquetes que envía el sistema SCADA. El procedimiento CRCCHK, revisa si el código de error es el correcto o no. Los procedimientos EXC1y3 y EXC2 se encargan de evaluar si lo recibido por el sistema MISERC está correcto. Para que el sistema MISERC se comporte de una manera independiente en cuanto al proceso de cada función se implementó una variable tipo bool, cierto o falso. La implementación de este tipo de banderas permitió que cualquier procedimiento interno del sistema MISERC se de cuenta del estado global del microprocesador sin tener que volver a ejecutar código repetido.

El procedimiento DPURdPro se encarga de llamar a sub-procedimientos en donde están ubicados los comandos para el DPU, además se encarga de formar la estructura ModBus de respuesta al sistema SCADA. Finalmente el procedimiento MODEMTX se encarga de enviar el paquete ADU al sistema SCADA.

6.3. Equivalencias de los Registros con los comandos SRC del DPU

Se estableció un mapa de registros del DPU de acuerdo al estándar ModBus, quedando de la siguiente manera.

Tabla 6.1 Dirección correspondiente a funciones 1 y 5

Dirección ModBus (decimal)	Dirección Física (hexa)	REGISTROS 0X Coil Status		REGISTROS 0X Coil Status	
		Función1 Leer 1 bit de lectura/escritura	**CMDSER**	**Función5** Escribir 1 bit de lectura/escritura	**CMDSER**
01	00	TGTMOD	A4	TGTMOD	A4
02	01	SELECT16	A5	SELECT16	A5
03	02	PVOTE	A8	PVOTE	A8
04	03	GNDLOC	A9	GNDLOC	A9
05	04	TCMODE	A10	TCMODE	A10
06	05	SELECT17	A11	SELECT17	A11
07	06	B1	B1	B1	B1
08	07	B2	B2	B2	B2
09	08			TGTRST	A2
10	09			RSTPEAKA	041
11	0A			RSTPEAKB	042
12	0B			RSTPEAKC	043
13	0C			RSTPEAKN	040
14	0D			CLRKSIA	081
15	0E			CLRKSIB	082
16	0F			CLRKSIC	083

Tabla 6.2 Dirección correspondiente a funciones 2 y 4

Dirección ModBus (decimal)	Dirección Física (hexa)	REGISTROS 1X Input Status			REGISTROS 3X Input Register	
		Función2 Leer 1 bit de solo lectura	**DESCRIPCIÓN**	**CMDSER**	**Función4** Leer 16 bits de solo lectura	**CMDSER**
01	00	ST	ROM	51	IA	D1
02	01	ST	RAM	51	IB	D2
03	02	ST	CNTRL POW	51	IC	D3
04	03	ST	+5V +/-15V	51	IN	D0
05	04	ST	ANALOG OFFSET	51	DIA	DD1
06	05	ST	TRIP CIRCUIT	51	DIB	DD2
07	06	ST	A/D CONVERTER	51	DIC	DD3
08	07	ST	ANALOG GAIN	51	DIN	DD0
09	08	SY	52A	52	PEAKA	41
10	09	SY	52B	52	PEAKB	42
11	0A	SY	TRQ CTRL PHSE	52	PEAKC	43

Dirección ModBus (decimal)	Dirección Física (hexa)	REGISTROS 1X Input Status			REGISTROS 3X Input Register	
		Función2 Leer 1 bit de solo lectura	DESCRIPCIÓN	CMDSER	Función4 Leer 16 bits de solo lectura	CMDSER
12	0B	SY	TRQ CTRL GRND	52	PEAKN	40
13	0C	SY	I NPUT 16 CNTCT	52	KSIA	81
14	0D	SY	TRIP CNTCT MON	52	KSIB	82
15	0E	SY	43A	52	KSIC	83
16	0F	SY	INST CUTOFF	52		

CMDSER = Comando Serie (SRC)

Tabla 6.2 Continuación

Tabla 6.3 Dirección correspondiente a funciones 3 y 6

Dirección ModBus (decimal)	Dirección Física (hexa)	REGISTROS 4X Holding Register Función3 Leer 16 bits de lectura/escritura	CMDSER	REGISTROS 4X Holding Register Función6 Escribir 16 bits de lectura/escritura	CMDSER	Descripción del Registro
01	00	DEMAND	5D	DEMAND	5D	DEMAND
02	01	ED79CO	A6	ED79CO	A6	ED79C0
03	02	CLTIME	A7	CLTIME	A7	CLTIME
04	03	COCTI	A12	COCTI	A12	COCTI
05	04	BKROP	70	BKROP	70	BKROP
06	05	OCOP	71	OCOP	71	OCOP
07	06	BFI	BF	BFI	BF	BF
08	07	BTC	B0	BTC	8747/7478	B0/TRIP/CLOSE
#1						
16	0F	DERS	5678 (SHOW)	DERS	FF (EDIT)	PHASE CT
17	10	DERS	5678 (SHOW)	DERS	FF (EDIT)	GROUND CT
18	11	DERS	5678 (SHOW)	DERS	FF (EDIT)	51 CURVE
19	12	DERS	5678 (SHOW)	DERS	FF (EDIT)	51 PICKUP
20	13	DERS	5678 (SHOW)	DERS	FF (EDIT)	**ALT 51 PICKUP$_{\#2}$**
21	14	DERS	5678 (SHOW)	DERS	FF (EDIT)	51 TIME DIAL
22	15	DERS	5678 (SHOW)	DERS	FF (EDIT)	50 CURVE
23	16	DERS	5678 (SHOW)	DERS	FF (EDIT)	50 PICKUP
24	17	DERS	5678 (SHOW)	DERS	FF (EDIT)	50PH TIME DELAY
25	18	DERS	5678 (SHOW)	DERS	FF (EDIT)	50H PICKUP
26	19	DERS	5678 (SHOW)	DERS	FF (EDIT)	51N CURVE
27	1A	DERS	5678 (SHOW)	DERS	FF (EDIT)	51N PICKUP
28	1B	DERS	5678 (SHOW)	DERS	FF (EDIT)	**ALT 51N PICKUP$_{\#2}$**
29	1C	DERS	5678 (SHOW)	DERS	FF (EDIT)	51N TIME DIAL
30	1D	DERS	5678 (SHOW)	DERS	FF (EDIT)	50N CURVE
31	1E	DERS	5678 (SHOW)	DERS	FF (EDIT)	50N PICKUP
32	1F	DERS	5678 (SHOW)	DERS	FF (EDIT)	50N TIME DELAY
33	20	DERS	5678 (SHOW)	DERS	FF (EDIT)	50NH PICKUP
34	21	DERS	5678 (SHOW)	DERS	FF (EDIT)	79-0 RESET TIME
35	22	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-1 PICKUP$_{\#3}$*
36	23	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-1 TIME DELAY$_{\#3}$*

Dirección ModBus (decimal)	Dirección Física (hexa)	REGISTROS 4X Holding Register Función3 CMDSER Leer 16 bits de lectura/escritura		REGISTROS 4X Holding Register **Función6 CMDSER** Escribir 16 bits de lectura/escritura		Descripción del Registro
37	24	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-2 PICKUP*$_{\#3}$
38	25	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-2 TIME DELAY*$_{\#3}$
39	26	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-3 PICKUP*$_{\#3}$
40	27	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-3 TIME DELAY*$_{\#3}$
41	28	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-4 PICKUP*$_{\#3}$
42	29	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-4 TIME DELAY*$_{\#3}$
43	2A	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-5 PICKUP*$_{\#3}$
44	2B	DERS	5678 (SHOW)	DERS	FF (EDIT)	*79-5 TIME DELAY*$_{\#3}$

CMDSER = Comando Serie (SRC)
#1 El direccionamiento continuo (dirección ModBus) solo es permitido dentro del rango [01 … 08] y en [16 … 44] , al tratar de leer o escribir en los registros [9 … 15] producirá el error Exception 2.
#2 ALT 51 PICKUP y ALT 51N PICKUP son habilitados por el comando B2
#3 Estos registros son dinámicos dentro del DPU.
Nota: El comando DERS no fue implementado dentro del sistema MISERC
Tabla 6.3 Continuación

6.4. Rango de datos de los Registros del DPU implementados en el protocolo ModBus

Dependiendo del tipo de registro, así puede variar su contenido, por tanto se diseñó un mapa de los datos que podrán ser enviados al DPU.

Tabla 6.4 Rango de datos para registros de tipo 0X y 1X

Dirección ModBus (decimal)	Dirección Física (hexa)	Rango de Datos Rango	/Incremento	Descripción de REGISTROS 0X		Descripción de REGISTROS 1X
01	00	0 … 1	/1	TGTMOD	A4	ROM
02	01	0 … 1	/1	SELECT16	A5	RAM
03	02	0 … 1	/1	PVOTE	A8	CNTRL POW
04	03	0 … 1	/1	GNDLOC	A9	+5V +/-15V
05	04	0 … 1	/1	TCMODE	A10	ANALOG OFFSET
06	05	0 … 1	/1	SELECT17	A11	TRIP CIRCUIT
07	06	0 … 1	/1	B1	B1	A/D CONVERTER
08	07	0 … 1	/1	B2	B2	ANALOG GAIN
09	08	0 … 1	/1	TGTRST	A2	52A
10	09	0 … 1	/1	RSTPEAKA	041	52B
11	0A	0 … 1	/1	RSTPEAKB	042	TRQ CTRL PHSE
12	0B	0 … 1	/1	RSTPEAKC	043	TRQ CTRL GRND

Dirección ModBus (decimal)	Dirección Física (hexa)	Rango de Datos Rango	/Incremento	Descripción de REGISTROS 0X		Descripción de REGISTROS 1X
13	0C	0 … 1	/1	RSTPEAKN	040	INPUT 16 CNTCT
14	0D	0 … 1	/1	CLRKSIA	081	TRIP CNTCT MON
15	0E	0 … 1	/1	CLRKSIB	082	43A
16	0F	0 … 1	/1	CLRKSIC	083	INST CUTOFF

Tabla 6.4 Continuación

Tabla 6.5 Rango de datos para registros de tipo 3X

Dirección ModBus (decimal)	Dirección Física (hexa)	Rango de Datos Rango	/Incremento	Descripción de REGISTROS 3X	
01	00	0 … 65536	/1	IA	D1
02	01	0 … 65536	/1	IB	D2
03	02	0 … 65536	/1	IC	D3
04	03	0 … 65536	/1	IN	D0
05	04	0 … 65536	/1	DIA	DD1
06	05	0 … 65536	/1	DIB	DD2
07	06	0 … 65536	/1	DIC	DD3
08	07	0 … 65536	/1	DIN	DD0
09	08	0 … 65536	/1	PEAKA	41
10	09	0 … 65536	/1	PEAKB	42
11	0A	0 … 65536	/1	PEAKC	43
12	0B	0 … 65536	/1	PEAKN	40
13	0C	0 … 65536	/1	KSIA	81
14	0D	0 … 65536	/1	KSIB	82
15	0E	0 … 65536	/1	KSIC	83

Tabla 6.6 Rango de datos para registros de tipo 4X

Dirección ModBus (decimal)	Dirección Física (hexa)	Rango de Datos Rango	/Incremento	Descripción de REGISTROS 4X
01	00	15 … 30	/15	DEMAND
02	01	1 … 200	/1	ED79C0
03	02	0 … 200	/1	CLTIME
04	03	18 … 999	/1	COCTI
05	04	0 … 9999	/1	BKROP
06	05	0 … 9999	/1	OCOP
07	06	5 … 60	/1	BF
08	07	0 … 12_{B0}	/1	B0/TRIP/CLOSE

Dirección ModBus (decimal)	Dirección Física (hexa)	Rango de Datos Rango	/Incremento	Descripción de REGISTROS 4X
#1				
16	0F	1 … 999	/1	PHASE CT
17	10	1 … 999	/1	GROUND CT
18	11	0 … 7	/1	51 CURVE
19	12	1 … 12	/0.1	51 PICKUP
20	13	0.2 … 2.4	/0.02	**ALT 51 PICKUP$_{\#2}$**
21	14	1 … 10 , F	/0.1	51 TIME DIAL
22	15	0 … 2	/1	50 CURVE
23	16	0.5 … 20X , F	/0.1	50 PICKUP
24	17	0 … 1	/0.01	50PH TIME DELAY
25	18	0.5 … 20X , F	/0.1	50H PICKUP
26	19	0 … 7	/1	51N CURVE
27	1A	1 … 12	/0.1	51N PICKUP
28	1B	0.2 … 2.4	/0.02	**ALT 51N PICKUP$_{\#2}$**
29	1C	1 … 10 , F	/0.1	51N TIME DIAL
30	1D	0 … 2	/1	50N CURVE
31	1E	0.5 … 20X , F	/0.1	50N PICKUP
32	1F	0 … 1	/0.01	50N TIME DELAY
33	20	0.5 … 20X , F	/0.1	50NH PICKUP
34	21	4 … 200	/1	79-0 RESET TIME
35	22	0 … 4	/1	*79-1 PICKUP$_{\#3}$*
36	23	0.1 … 200 , L	/0.1	*79-1 TIME DELAY$_{\#3}$*
37	24	0 … 4	/1	*79-2 PICKUP$_{\#3}$*
38	25	0.1 … 200 , L	/0.1	*79-2 TIME DELAY$_{\#3}$*
39	26	0 … 4	/1	*79-3 PICKUP$_{\#3}$*
40	27	0.1 … 200 , L	/0.1	*79-3 TIME DELAY$_{\#3}$*
41	28	0 … 4	/1	*79-4 PICKUP$_{\#3}$*
42	29	0.1 … 200 , L	/0.1	*79-4 TIME DELAY$_{\#3}$*
43	2A	0 … 4	/1	*79-5 PICKUP$_{\#3}$*
44	2B	L		*79-5 TIME DELAY$_{\#3}$*

#1 El direccionamiento continuo (dirección ModBus) solo es permitido dentro del rango [01 … 08] y en [16 … 44] , al tratar de leer o escribir en los registros [9 … 15] producirá el error Exception 2. Los registros [16 … 44] no los soporta el sistema MISERC.
#2 ALT 51 PICKUP y ALT 51N PICKUP son habilitados por el comando B2
#3 Estos registros son dinámicos dentro del DPU.
Tabla 6.6 Continuación

6.5. Alcances y limitaciones

El sistema MISERC al implementar más de los comandos pedidos cumplió con excelencia esa especificación.

Se pudo realizar el revisado de error por medio del CRC, disminuyendo así posibles errores de transmición.

El protocolo ModBus soporta funciones que le indican al DTE una escritura múltiple de datos por función ModBus en los registros del DPU. Esta función es interesante, pero quita el estándar de tener un largo igual para las demás funciones implementadas en el sistema MISERC. Hay que tener en cuenta que el sistema MISERC soporta una función de escritura de un único registro por función por tanto habría que llamar tantas veces la función como registros que se deseen escribir.

Hay otras funciones aún más complejas que el protocolo ModBus soporta, pero estas funciones se extralimitan al alcance de este proyecto.

Una limitación que podría tener el sistema es a la hora de actualizar el firmware. La desventaja de la actualización sería la de quitar el microcontrolador de la tarjeta de comunicación, dando como resultado la in operabilidad de una unida DPU ya que se desconectaría del sistema SCADA. La opción para solventar esta desventaja sería la de implementar la función de transferencia de archivos por medio del protocolo ModBus. Al implementar está función se puede mandar vía la red SCADA un nuevo firmware que se actualizará automáticamente en el microcontrolador. Esta opción la tienen los microprocesadores de la familia PICxxF.

Capítulo 7. Conclusiones y recomendaciones

7.1. Conclusiones

- El DPU es un dispositivo que además de servir como medidor de corrientes, provee medidas de seguridad a la red eléctrica.
- Al poseer el DPU un puerto RS232, se le puede implementar un sistema de interrogación o programación de sus registros.
- Para lograr una comunicación con el sistema SCADA, el protocolo ASCII no es eficiente.
- La implementación del Protocolo ModBus provee las herramientas de comunicación necesarias para lograr una comunicación eficiente con el DPU.
- El Microcontrolador PIC16F877 provee la versatilidad adecuada para implementar un protocolo de alto nivel como el ModBus.

7.2. Recomendaciones

- Implementar las funciones 15 y 16 del Protolo ModBus, esto para agilizar la transferencia de datos.
- Implementar el uso del comando EDIT y SHOW del DPU para lograr una utilización casi completa de todos los comandos del DPU.
- Diseñar el procedimiento de Interrupción por Delay, para permitir diferentes tamaño de paquete y poder implementar las funciones 15 y 16.
- Usar un microcontrolador más pequeño, logrando así un menor espació en el diseño de la tarjeta de comunicación.

Bibliografía

1. Texas Instruments, US. 1997. Designer´s guide and Databook. 1 disco compacto, 8mm.

2. Texas Instruments, US. 1997. Logic Selector Guide and Databook. 1 disco compacto, 8mm.

3. ABB. Buyer´s Guide 98/99 Protection, monitoring and Control descriptive Literature. ABB. Coral Springs, USA. 1999

4. Microchip, US. 2000. Microchip technical library CD-ROM, second edition 2000. 2 discos compactos, 8mm.

5. Microchip, US. 2000. PICmicro Microcontrollers (MCUs), PIC16F877 Family, PIC16F877 datasheet and MPLAB version 5.7 homepage (en línea). Disponible en http://www.microchip.com

6. FieldNet AS. 2001. Modbus Protocol homepage (en línea). Disponible en http://www.fieldnet.no/fbacad/chapter_1.htm

7. WinTECH Software, US. 1997. ModScan32 Application Description homepage (en línea). Disponible en http://www.win-tech.com/html/modscan32.htm

Apéndices

Apéndice A.1 Glosario

ADU: Application Data Unit, unidad de aplicación de información

ASCII: Estándar, codificación de caracteres

Baudios: Medida utilizada para la velocidad de una transmisión digital, la cual se da en bits por segundo.

Bit: Unidad míinima del sistema binario, representa uno o cero

Byte: Conjunto de 8 bits

CCE: Centro de Control de Energía

CNFL S.A.: Compañía Nacional de Fuerza y Luz, Sociedad Anónima

CPU: Unidad de procesamiento central.

CRC16: Chequeo de redundancia cíclica, es un método para determinar errores en transmisiones digitales de datos, este genera un valor de 16 bits el cual es añadido al mensaje en el campo de chequeo de error, donde se envían primero el byte menos significativo y luego el más significativo.

DIP SWITCH: Componente en paquete dual en línea que posee interruptores.

DPU: Distribution Protection Unit, unidades de protección de distribución.

Estación remota de poste: Es una RTU colocada en un punto específico, la cual se encarga de una aplicación específica.

Firmware: Microprograma de un procesador o microcontrolador.

Frame: Estructura de un mensaje en una transmisión digital.

MISERC: ModBus Interface to SErial Relay Commands. Sistema traductor de protocolos entre un DPU y un sistema que se comunique por ModBus.

Modbus ASCII: Protocolo de comunicación donde cada byte en un mensaje se envía como dos caracteres ASCII. Utiliza un chequeo de redundancia longitudinal para determinar errores en la transmisión.

Modbus RTU: Protocolo de comunicación donde cada byte en un mensaje se envía como dos caracteres hexadecimales. Utiliza un chequeo de redundancia cíclica para determinar errores en la transmisión.

Protocolo: Set de reglas que hacen la comunicación en las redes más eficiente. Determina el formato y transmisión de los datos.

LSB: Bit menos significativo, es aquel que está más a la derecha y que tiene el menor valor posicional.

MSB: Bit más significativo, es aquel que está más a la izquierda y que tiene el mayor valor posicional.

PCB: Tarjeta de circuito impreso, está formada por pistas conductoras de corriente, las cuales conectan varios tipos de dispositivos (chips, resistencias, capacitores, interruptores, bases para montar chips, etc.) entre sí, también tiene agujeros para colocar y soldar los dispositivos.

RAM: Memoria de acceso aleatorio, es una memoria semiconductora volátil, en la cual se garantiza que los tiempos de acceso son iguales sin importar la ubicación de la información.

RTU: Unidad terminal de radio, es un dispositivo de comunicación de datos, puede ser unidireccional o bidireccional, funciona como interfaz para ejecutar comandos o recibir datos en una red de distribución eléctrica.

SCADA: Sistema de supervisión, control y adquisición de datos, es una red que se encarga de monitorear la red de distribución eléctrica de la CNFL.

Transformador: Máquina eléctrica que se utiliza para transformar un voltaje de un valor a otro.

USART: Transmisor y receptor asíncrono sincrónico universal, puerto programable que se encarga de controlar la transferencia y recepción de datos asíncronos o sincrónicos por el puerto serie.

Apéndice A.2 Comandos SRC del DPU que son utilizados

Dirección ModBus (decimal)	Dirección Física (hexa)	Comando SRC	CMDSER	Uso de Función	Tipo de Registro	Descripción del Registro
01	00	TGTMOD	A4	F1 F5	RW	Display/Edit Target Mode
02	01	SELECT16	A5	F1 F5	RW	Display/Edit A5 Input Mode
03	02	PVOTE	A8	F1 F5	RW	Display/Edit Phase Inst. Vote Mode
04	03	GNDLOC	A9	F1 F5	RW	Display/Edit Lockout on 51N Trip
05	04	TCMODE	A10	F1 F5	RW	Display/Edit TCC Output Contact Mode
06	05	SELECT17	A11	F1 F5	RW	Display/Edit Instantaneous Cutout Input Mode
07	06	B1	B1	F1 F5	RW	Display/Edit Zone Sequence Coordination Mode
08	07	B2	B2	F1 F5	RW	Display/Edit Alternate Minimum Trip Mode
09	08	TGTRST	A2	F5	OW	Reset Front Panel Targets
10	09	RSTPEAKA	041	F5	OW	Clear Peak Demand Current A
11	0A	RSTPEAKB	042	F5	OW	Clear Peak Demand Current B
12	0B	RSTPEAKC	043	F5	OW	Clear Peak Demand Current C
13	0C	RSTPEAKN	040	F5	OW	Clear Peak Demand Current D
14	0D	CLRKSI-A	081	F5	OW	Clear KSI, Phase A
15	0E	CLRKSI-B	082	F5	OW	Clear KSI, Phase B
16	0F	CLRKSI-C	082	F5	OW	Clear KSI, Phase C
01-08	00-07	ST	51	F2	OR	Display Self-Check Status
09-16	08-0F	SY	52	F2	OR	Display Contact Input Status
01	00	IA	D1	F4	OR	Display Current, Phase A
02	01	IB	D2	F4	OR	Display Current, Phase B
03	02	IC	D3	F4	OR	Display Current, Phase C
04	03	IN	D0	F4	OR	Display Current, Phase N
05	04	DIA	DD1	F4	OR	Display Demand Current, Phase A
06	05	DIB	DD2	F4	OR	Display Demand Current, Phase B
07	06	DIC	DD3	F4	OR	Display Demand Current, Phase C
08	07	DIN	DD0	F4	OR	Display Demand Current, Phase N
09	08	PEAKA	41	F4	OR	Display Peak Demand Current A
10	09	PEAKB	42	F4	OR	Display Peak Demand Current B
11	0A	PEAKC	43	F4	OR	Display Peak Demand Current C
12	0B	PEAKN	40	F4	OR	Display Peak Demand Current N
13	0C	KSIA	81	F4	OR	Display Accumulated KSI, Phase A
14	0D	KSIB	82	F4	OR	Display Accumulated KSI, Phase B
15	0E	KSIC	83	F4	OR	Display Accumulated KSI, Phase C
01	00	DEMAND	5D	F3 F6	RW	Display/Edit Demand Meter Time Constant
02	01	ED79CO	A6	F3 F6	RW	Display/Edit 79 Cutout Time
03	02	CLTIME	A7	F3 F6	RW	Display/Edit Cold-Load Pickup Time
04	03	COCTI	A12	F3 F6	RW	Display/Edit Failure-to-Close Time Interval

Dirección ModBus (decimal)	Dirección Física (hexa)	Comando SRC	CMDSER	Uso de Función	Tipo de Registro	Descripción del Registro
05	04	BKROP	70	F3 F6	RW	Display/Edit Breaker Operations Counter
06	05	OCOP	71	F3 F6	RW	Display/Edit Overcurrent-trip Counter
07	06	BF	BF (BFI)	F3 F6	RW	Breaker Fail Time Interval (cycles)
08	07	B0/ TRIP/ CLOSE	B0/ 8747/ 7478 (BTC)	F3 F6	RW	Display Circuit Breaker Status/ Trip Circuit Breaker/ Close Circuit Breaker
09-37	08-24	SHOW/EDIT	5678/FF (DERS)	F3 F6	RW	Display Relay Settings/ Edit Relay Setting

RW = Lectura y Escritura OR = Solo Lectura OW = Solo Escritura

CMDSER = Comando Serie

El comando DERS (show, edit) no fue implementado, pero se encuentra “mapeado” dentro de sistema.

Apéndice A.3 Comandos SRC del DPU que NO son utilizados

Comando SRC	**CMDSER**	Descripción del Registro	Razón de no utilizarlo
CT	9A	Display Feeder CT Ratios	Utilizado en SHOW/EDIT
VER	D99	Display Software version	No es necesario
BAUD		Display Baud Rate	Dato fijo a 9600 baudios
TIME	89AB	Display Clock	No es necesario
ID		Display Relay ID	No es necesario
	5B	Display/Edit Baud Rate	Dato fijo a 9600 baudios
NEWT	4567	Display/Edit Clock	No es necesario
SETID		Display/Edit Relay ID	No es necesario
PASS	A1	Change Editing Password	Dato fijo a 0000
TCPWD	A3	Change Trip/Close Password	Dato fijo a 0000
EVENT		Display Entire Event Record	Excede los requerimientos del proyecto
Dbmn	Dbmn	Display Trip “n” of Event “m”	Excede los requerimientos del proyecto
PFLDTA	30	Display Detalis of Last Power Failure	Excede los requerimientos del proyecto
IABC	D123	Display Phase Currents	Utilizado en D1,D2,D3
ALL	33	Display All Currents	Utilizado en D(0-3),DD(0-3),4(0-3)
PEAKT	45	Display Peak Demand Times	Excede los requerimientos del proyecto
333	333	Clear All Peak Demand Current	Utilizado en 040,041,042,043
TEST	8	Test Target Lights or Printer	No es necesario
SYSCAL	0347	Calibrate Current Metering	Excede los requerimientos del proyecto
SHOW/EDIT	SHOW/EDIT	Display Relay Settings/ Edit Relay Setting	Excede los requerimientos del proyecto

Apéndice A.4 Listado del programa principal

```
;-------------------------------------------------------------------------------
;  M.I.SE.R.C.   MODBUS INTERFACE to SERIAL RELAY COMMANDS    FECHA:01/II/2002
;-------------------------------------------------------------------------------
;   DISEÑADO POR :    ADRIANO ENRIQUE ESPINOSA CARREIRA (adenesca@yahoo.com)
;-------------------------------------------------------------------------------
;    INTERFAZ ENTRE LA DPU ABB 245D/445H/445V Y LA RED ModBus DEL CENTRO DE
; CONTROL DE LA COMPAÑIA NACIONAL DE FUERZA Y LUZ, URUCA, SAN JOSE, COSTA RICA
;-------------------------------------------------------------------------------
;                             LIMITACIONES:
;1. 8BYTES DE RX SI MAS O MENOS DESCARTA LA RECEPCION EN SU TOTALIDAD
;2. TX Y RX A 9600
;3. TIMER 0 y 1 CAMBIAN SUS VALORES AL CAMBIAR MHz DEL CLK
;-------------------------------------------------------------------------------
;*******************************************************************************
       LIST P=16F877
       ERRORLEVEL  -302      ;QUITA LOS WARNINGS DE OPERANDO BANK 1,2,3
       INCLUDE <P16F877.INC> ;Fosc = 20 MHz
 __CONFIG _HS_OSC & _WDT_OFF & _PWRTE_ON & _BODEN_OFF & _LVP_OFF & _CPD_OFF & _WRT_ENABLE_OFF
& _DEBUG_OFF & _CP_OFF
;*******************************************************************CONSTANTES
;BAUD9600      EQU     0X81    ; 0X81=129  OJO BRGH=1 CLK=20MHz
;BAUD9600      EQU     0X40    ; 0X40= 64  OJO BRGH=1 CLK=10MHz
BAUD9600       EQU     0X19    ; 0X19= 25  OJO BRGH=1 CLK= 4MHz
;*************************************************************CONSTANTES DE BOOL
RdWr           EQU 7 ;[7] 0=READ F[1,2,3,4]  / 1=WRITE F[5,6]
CRC            EQU 6 ;[6] 0=GOOD CRCCHK      / 1=BAD CRCCHK
T0OVERF        EQU 5 ;[5] 0=NO OVERF TIMER0  / 1=OVERF TIMER0
T1OVERF        EQU 4 ;[4] 0=NO OVERF TIMER1  / 1=OVERF TIMER1
EXC            EQU 3 ;[3] 0=GOOD EXCEPTION   / 1=BAD EXCEPTION
DPUERROR       EQU 2 ;[2] 0=GOOD TX=RX,Delay / 1=BAD TX<>RX,Delay
REGF36         EQU 1 ;[1] 0=REG[0..7],SI ALT / 1=REG[0F..2B],NO ALT
PASSWORD       EQU 0 ;[0] 0=NO       TX CLAVE AL DPU / 1=SI TX CLAVE AL DPU
;*************************************************************VARIABLES BANK0
 CBLOCK 0X20           ; ADDRESS DE VARIABLE
BUFFERRX:      0XF     ; BUFFER DE RX FH=15D=1111B
BUFFEREXC:     0X06    ; BUFFER DE TX DE ERRORES
H_TEMP:        1       ; BCD2BIN FUNCION2
L_TEMP:        1       ; BCD2BIN FUNCION2
H_BYTE:        1       ; BCD2BIN BIN2BCD
L_BYTE:        1       ; BCD2BIN BIN2BCD
BCD2:          1       ; BCD2BIN BIN2BCD MSD
BCD1:          1       ; BCD2BIN BIN2BCD 1/2
BCD0:          1       ; BCD2BIN BIN2BCD LSD
REG4:          1       ;
REG5:          1       ;
REG6:          1       ;
 ENDC
;*************************************************************VARIABLES BANK1
 CBLOCK 0XA0           ; ADDRESS DE VARIABLE
BUFFERTX:      0X50     ; BUFFER DE TX 50H=80D=1010000B
 ENDC
;*********************************************MAX 16 VAR***VARIABLES BANK[0..3]
 CBLOCK 0X70           ; ADDRESS DE VARIABLES
W_TEMP:        1       ; VAR TEMP DE W
STATUS_TEMP:   1       ; VAR TEMP DE STATUS
FSR_TEMP:      1       ; VAR TEMP DE FSR
;PCLATH_TEMP:  1       ; VAR TEMP DE PCLATH (SI USO PAG 1,2,3)
BOOL:          1       ; ALL
CRCH:          1       ; CRCGEN
CRCL:          1       ; CRCGEN
COUNT:         1       ; BIN2BCD CRCGEN
EXCP:          1       ;
CHAR:          1       ;
RORL:          1       ;
INFO:          1       ;
NUMBYTES:      1       ; CONTADOR DE BYTES (RX/TX)  SIN INCLUIR (2 CRC BYTES)
DIR:           1       ;
FUNCION:       1       ;
 ENDC
;*************************************************************************ENCABEZADO
```

```
                ORG     0X00            ; RESET VECTOR LOCATION
                GOTO    BEGIN
                ORG     0X04            ; INTERRUPT VECTOR LOCATION
                ;GOTO   INTERRUPCION
;*****************************************************************ENCABEZADO
;*****************************************************************INTERRUPCION
INTERRUPCION
                MOVWF   W_TEMP          ; GUARDA W
                SWAPF   STATUS,W
                MOVWF   STATUS_TEMP     ; GUARDA STATUS
                MOVFW   FSR
                MOVWF   FSR_TEMP        ; GUARDA FSR
                ;MOVFW  PCLATH
                ;MOVWF  PCLATH_TEMP     ; GUARDA PCLATH

                BTFSC   INTCON,T0IF     ; INT TIMER0
                CALL    TIMER0_OFF
                BTFSC   PIR1,TMR1IF     ; INT TIMER1
                CALL    TIMER1_OFF

                ;MOVFW  PCLATH_TEMP
                ;MOVWF  PCLATH          ; RESTAURA PCLATH
                MOVFW   FSR_TEMP
                MOVWF   FSR             ; RESTAURA FSR
                SWAPF   STATUS_TEMP,W
                MOVWF   STATUS          ; RESTAURA STATUS
                SWAPF   W_TEMP,F
                SWAPF   W_TEMP,W        ; RESTAURA W
 RETFIE
;*****************************************************************INTERRUPCION
;*************************************************************************MACRO
BANK0   MACRO
                BCF     STATUS, RP0             ; BANK 0
 ENDM
;*****
BANK1   MACRO
                BSF     STATUS, RP0             ; BANK 1
 ENDM
;*****
CHK     MACRO
                MOVLW   0X04            ; PRE CARGO w CON EXC=4 POR SI HAY ERROR
                BTFSC   BOOL,DPUERROR
                CALL    EXCEPTION
                BTFSC   BOOL,EXC
 RETURN                                 ; SALE CON ERROR DE EXC
                MOVLW   .13             ; MANDA ENTER
                CALL    SENDTX
 ENDM
;*************************************************************************MACRO
;*************************************************************************INCLUDE
 INCLUDE <Avarios.ASM>; PICINI DPUINI TIMER0(ON/OFF) TIMER1(ON/OFF) MODEM DPU
                       ; MODEM(RX/TX) DPUTX SENDTX SENDBUFFER BCD2BIN BIN2BCD
 INCLUDE <Aerror.ASM> ; EXC1y3 EXC2 EXCEPTION CRCGEN CRCCH CHKTXRX
 INCLUDE <Afuncion.ASM> ; DPURdPro LEER(NUM/CHAR/xCHAR/5CHAR) F36INI F[1..6]
 INCLUDE <Acmd.ASM>           ; Comandos ligados al DPU
                ; CMD DE F1 READ  1  BIT/RW   ; CMD DE F5 WRITE 1  BIT/RW
                ; CMD DE F2 READ  1  BIT/R
                ; CMD DE F3 READ  16 BIT/RW   ; CMD DE F6 WRITE 16 BIT/RW
                ; CMD DE F4 READ  16 BIT/R
;*****************************************************************INCLUDE
;*****************************************************************BEGIN*BEGIN
BEGIN           CALL    PICINI          ; INICIALIZA EL PIC
                ;CALL   TIMER1_OFF      ; OFF DELAY DE TIME OUT
INI             CALL    DPUINI          ; INICIALIZA LA TX/RX CON EL DPU

RXM             CALL    MODEMRX         ; START RX DE PAQUETES DE MODEM
                ;CALL   TIMER1_ON       ; ON DELAY DE TIME OUT

AA              CALL    CRCCHK          ; CHK=CRC
                BTFSC   BOOL,CRC
                GOTO RXM

AB              CALL    EXC1y3          ; CHK=DIR EXC(1,3)
                BTFSC   BOOL,EXC
                GOTO RXM
```

```
AC              CALL    EXC2            ; CHK=EXC(2) BOOL[RdWr]
                BTFSC   BOOL,EXC
                GOTO RXM

AD              CALL    DPURdPro        ; READ & PROCESS DATA FROM DPU
                BTFSC   BOOL,EXC
                GOTO INI

                ;;CALL  TIMER1_OFF      ; OFF DELAY DE TIME OUT
                CALL    MODEMTX
                GOTO INI
 END
;*********************************************************************END*END*END
```

Anexo B.1 Hoja de Dato de PIC16F877

MICROCHIP

PIC16F87XA

28/40-Pin Enhanced FLASH Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High Performance RISC CPU:

- Only 35 single word instructions to learn
- All single cycle instructions except for program branches, which are two-cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory,
  Up to 368 x 8 bytes of Data Memory (RAM),
  Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during SLEEP via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I²C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) 8-bits wide, with external RD, WR and CS controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8 channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (VREF) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced FLASH program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving SLEEP mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

CMOS Technology:

- Low power, high speed FLASH/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low power consumption

Anexo B.2 Hoja de Dato del LM7805

Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Input Voltage (for V_O = 5V to 18V) (for V_O = 24V)	V_I V_I	35 40	V V
Thermal Resistance Junction-Cases	$R_{\theta JC}$	5	°C/W
Thermal Resistance Junction-Air	$R_{\theta JA}$	65	°C/W
Operating Temperature Range (MC78XXCT/LM78XXCT/MC78XXCDT)	T_{OPR}	0 ~ +125	°C
Storage Temperature Range	T_{STG}	-65 ~ +150	°C

Electrical Characteristics (MC7805/LM7805)

(Refer to test circuit ,$0°C < T_J < 125°C$, I_O = 500mA, V_I = 10V, C_I= 0.33µF, C_O= 0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		MC7805/LM7805			Unit
				Min.	Typ.	Max.	
Output Voltage	V_O	T_J =+25 °C		4.8	5.0	5.2	V
		5.0mA ≤ Io ≤ 1.0A, P_O ≤ 15W V_I = 7V to 20V V_I = 8V to 20V		4.75	5.0	5.25	
Line Regulation	ΔV_O	T_J=+25 °C	V_O = 7V to 25V	-	4.0	100	mV
			V_I = 8V to 12V	-	1.6	50	
Load Regulation	ΔV_O	T_J=+25 °C	I_O = 5.0mA to1.5A	-	9	100	mV
			I_O =250mA to 750mA	-	4	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	8	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	0.03	0.5	mA
		V_I= 7V to 25V		-	0.3	1.3	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O= 5mA		-	-0.8	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A=+25 °C		-	42	-	µV
Ripple Rejection	RR	f = 120Hz V_O = 8V to 18V		62	73	-	dB
Dropout Voltage	V_O	I_O = 1A, T_J =+25 °C		-	2	-	V
Output Resistance	R_O	f = 1KHz		-	15	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A =+25 °C		-	230	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Anexo B.3 Hoja de Dato del 74LS125

The SN54125, SN54126, SN74125, SN74126, and SN54LS126A are obsolete and are no longer supplied.

SN54125, SN54126, SN54LS125A, SN54LS126A, SN74125, SN74126, SN74LS125A, SN74LS126A QUADRUPLE BUS BUFFERS WITH 3-STATE OUTPUTS

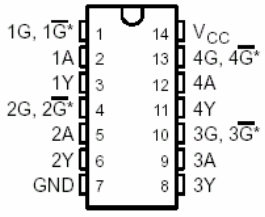
SDLS044A – DECEMBER 1983 – REVISED MARCH 2002

- **Quad Bus Buffers**
- **3-State Outputs**
- **Separate Control for Each Channel**

description

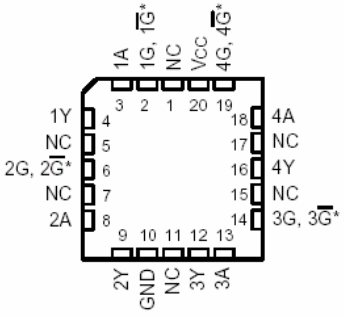
These bus buffers feature three-state outputs that, when enabled, have the low impedance characteristics of a TTL output with additional drive capability at high logic levels to permit driving heavily loaded bus lines without external pullup resistors. When disabled, both output transistors are turned off, presenting a high-impedance state to the bus so the output will act neither as a significant load nor as a driver. The '125 and 'LS125A devices' outputs are disabled when $\overline{G}$ is high. The '126 and 'LS126A devices' outputs are disabled when G is low.

SN54125, SN54126, SN54LS125A, SN54LS126A . . . J OR W PACKAGE
SN74125, SN74126 . . . N PACKAGE
SN74LS125A, SN74LS126A . . . D, N, OR NS PACKAGE
(TOP VIEW)



*$\overline{G}$ on '125 and 'LS125A devices;
G on 126 and 'LS126A devices

SN54LS125A, SN54LS126A . . . FK PACKAGE
(TOP VIEW)



*$\overline{G}$ on '125 and 'LS125A devices;
G on 126 and 'LS126A devices
NC – No internal connection

Anexo B.4 Hoja de Dato del MAX233A

+5V-Powered, Multichannel RS-232 Drivers/Receivers

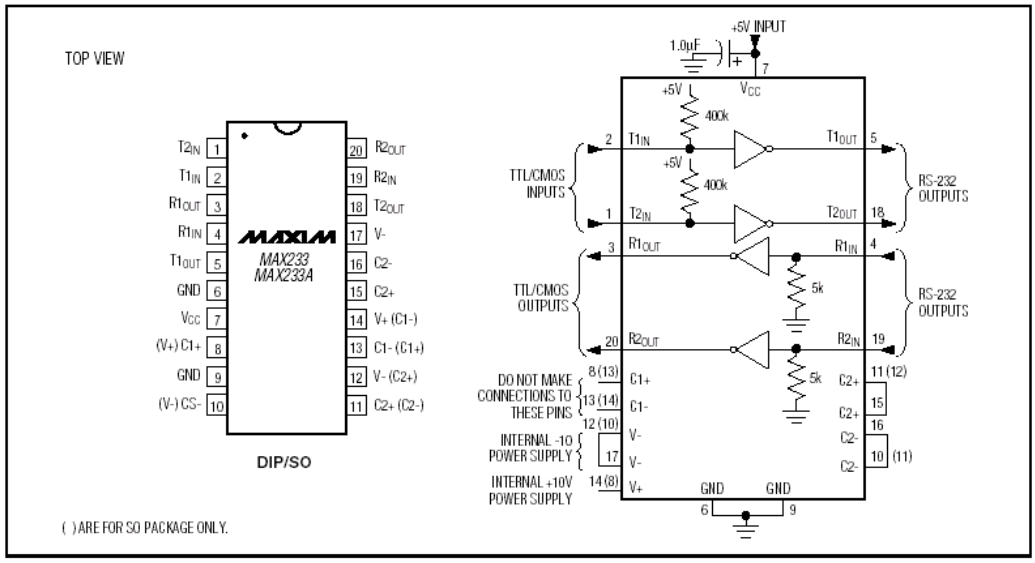


Figure 11. MAX233/MAX233A Pin Configuration and Typical Operating Circuit