

INSTITUTO TECNOLÓGICO DE MÉRIDA

TESIS

**“SISTEMA DE MONITOREO INALÁMBRICO PARA MEDICIÓN DE
PARÁMETROS ENERGÉTICOS EN PLANTA
PRODUCTORA DE MUEBLES”**

PARA OPTAR AL GRADO DE:
**MAESTRO EN INGENIERÍA
OPCIÓN EN MECATRÓNICA**

PRESENTA:

ING. CARLOS HUGO CARMONA GARCÍA

ASESOR:

DR. JOSÉ RAMÓN ATOCHE ENSEÑAT

COASESOR:

M.C. JOSÉ AGUSTÍN HERNÁNDEZ BENÍTEZ

MÉRIDA, YUCATÁN, MÉXICO.
17 DE JUNIO 2016

Autorización de publicación



TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Mérida

DEPENDENCIA: DIV. DE EST. DE POSG. E INV.
NO. DE OFICIO: X-140/2016

ASUNTO: AUTORIZACIÓN DE IMPRESIÓN

MÉRIDA, YUCATÁN A 01 DE JUNIO DE 2016

C. CARLOS HUGO CARMONA GARCÍA
PASANTE DE MAESTRIA EN INGENIERÍA

De acuerdo al fallo emitido por su asesor el Dr. José Ramón Atoche Enseñata, y su coasesor el M.C. José Agustín Hernández Benítez, la comisión revisora integrada por el Dr. Víctor Sandoval Curmina, y el Dr. Agustín Alfonso Flores Novelo, considerando que cubre los requisitos establecidos en el Reglamento de Titulación de los Institutos Tecnológicos le autorizamos la impresión de su trabajo profesional con la TESIS:

"Sistema de monitoreo inalámbrico para medición de parámetros energéticos en planta productora de muebles"

ATENTAMENTE
IN HOC SIGNO VINCES

M.C. MIRIAM H. SÁNCHEZ MONROY
JEFA DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN

C.p. Archivo
MIISM/fja.



S. E. P.
INSTITUTO TECNOLÓGICO
DE MÉRIDA
DIVISIÓN DE ESTUDIOS DE
POSGRADO E INVESTIGACIÓN



SEP Instituto Tecnológico de Mérida, Km.5 Carretera Mérida-Progreso
C.P 97118 Mérida Yucatán, México. Tels. 964
10601, 10201 e-mail:itm@itmerida.mx http://www.itmerida.mx



Agradecimientos

A mi esposa Kristhy que gracias a su confianza, paciencia y apoyo me ha impulsado a terminar este proyecto.

Tu cariño, tu amor y tus grandes manifestaciones de afecto, son una gran bendición de Dios, muchas gracias por la constante motivación y ayudarme a concluir mi proyecto de tesis de la mejor manera posible.

A mi asesor de tesis el Dr. Ramón Atoche, que gracias a sus conocimientos, orientaciones y su metodología de trabajo, han sido fundamentales para el desarrollo del proyecto de tesis. Siempre siguiendo un sentido de seriedad, compromiso, responsabilidad sin los cuales no podría tener una formación integral como especialista.

A mis padres que hasta el día de hoy siguen siendo apoyo y guías de mis pasos, sin su gran ejemplo no sería la persona perseverante por cumplir sus ideales que soy ahora.

Índice de contenido

Autorización de publicación	i
Agradecimientos	ii
Índice de contenido.....	iii
Índice de tablas	v
Índice de ilustraciones.....	vi
Resumen.....	1
Abstract	2
1 Introducción	3
1.1 Presentación	3
1.2 Planteamiento del problema	3
1.3 Objetivos	4
1.3.1 Objetivo general	4
1.3.2 Objetivos específicos	4
1.4 Justificación.....	5
1.5 Estado del arte	6
2 Marco teórico.....	16
2.1 Topologías de red.....	16
2.2 Redes industriales.....	19
2.3 Descripción de funciones del protocolo Modbus.....	22
3 Desarrollo	30
3.1 Diagrama de bloques de solución	30
3.2 Selección de tecnologías.....	32
3.3 Diseño del coordinador.....	34
3.3.1 Alimentación del coordinador	35
3.3.2 Alimentación del XBEE.....	35
3.3.3 Montaje de microcontrolador	36
3.3.4 Comunicación entre Microcontrolador y XBEE.....	38
3.3.5 Conexión USB	40
3.4 Diseño de los dispositivos finales.....	41

3.4.1	Alimentación dispositivo final.....	41
3.4.2	Alimentación microcontrolador	41
3.4.3	Alimentación XBee.....	42
3.4.4	Montaje básico del microcontrolador	42
3.4.5	Conexión entre el microcontrolador y el XBee.....	42
3.4.6	Conexión entre microcontrolador y MAX 485.....	42
3.5	Analizador de redes eléctricas	43
3.6	Herramientas de programación	45
3.6.1	Visual Studio 2012	45
3.6.2	PIC C Compiler	46
3.6.3	X-CTU	47
3.7	Configuración de dispositivo final.....	49
3.7.1	Adquisición de datos	49
3.8	Configuración del módulo XBee.....	51
3.8.1	Configuración del coordinador	52
3.8.2	Configuración del dispositivo final.....	54
4	Resultados	55
5	Conclusiones	59
6	Bibliografía	61
7	Anexos.....	62
7.1	Código de SCADA	62
7.2	Interfaz visual de software SCADA.....	68
7.3	Código dispositivo final	69
7.4	Diagrama electrónico de coordinador	71
7.5	Diagrama electrónico de dispositivo final.....	72

Índice de tablas

TABLA 1.1: FUNCIONES DEL PROTOCOLO MODBUS	11
TABLA 2.1: COMPARATIVA DE PRINCIPALES REDES	22
TABLA 2.2: SUBFUNCIONES DE LAS FUNCIÓN 00H	23
TABLA 2.3: CONTADORES DE DIAGNÓSTICO	27
TABLA 3.1: CONFIGURACIÓN BÁSICA DE NETWORKING & SECURITY	53
TABLA 3.2: CONFIGURACIÓN NETWORKING & SECURITY DISPOSITIVO FINAL	54

Índice de ilustraciones

FIGURA 1.1: DATOS ENERGÉTICOS POR ÁREA	5
FIGURA 1.2: DIFERENCIA ENTRE ARQUITECTURA DE RED INDUSTRIAL Y RED COMERCIAL [1].....	6
FIGURA 1.3: NIVELES DE UNA RED INDUSTRIAL [3].....	7
FIGURA 1.4: PRINCIPALES REDES INDUSTRIALES [5]	8
FIGURA 1.5 : TIPOS DE CODIFICACIONES MODBUS	10
FIGURA 1.6: CÁLCULO DE CRC EN CODIFICACIÓN RTU	12
FIGURA 1.7: PRINCIPIO DE MEDICIÓN DE UN TRANSDUCTOR FLUX GATE [7]	15
FIGURA 2.1: TOPOLOGÍA DE BUS	16
FIGURA 2.2: TOPOLOGÍA DE ESTRELLA.....	17
FIGURA 2.3: TOPOLOGÍA DE ÁRBOL	18
FIGURA 2.4: TOPOLOGÍA EN MALLA	18
FIGURA 2.5: TOPOLOGÍA EN ANILLO	19
FIGURA 2.6: TRAMA GENÉRICA DE LAS SUBFUNCIONES DE CONTROL DE ESCLAVOS.....	23
FIGURA 2.7: PETICIÓN Y RESPUESTA DE LA FUNCIÓN 01H Y 02H	24
FIGURA 2.8: PETICIÓN Y RESPUESTA DE LA FUNCIÓN 03H Y 04H	24
FIGURA 2.9: PETICIÓN Y RESPUESTA DE LA FUNCIÓN 05H	25
FIGURA 2.10: PETICIÓN Y RESPUESTA DE LA FUNCIÓN 06H	25
FIGURA 2.11: LECTURA DE UN OCTETO	26
FIGURA 2.12: CONTROL DE CONTADORES	26
FIGURA 2.13: CONTENIDO DE CONTADOR 9	27
FIGURA 2.14: ESCRITURA DE BITS.....	28
FIGURA 2.15: ESCRITURA DE PALABRAS.....	28
FIGURA 2.16: TRAMA DE MENSAJE DE ERROR	28
FIGURA 3.1: DIAGRAMA DE BLOQUES DE LA SOLUCIÓN	30
FIGURA 3.2: INDICADOR LED DE ALIMENTACIÓN	35
FIGURA 3.3: ALIMENTACIÓN XBEE	36
FIGURA 3.4: CIRCUITO DE RESET (MCLR)	36
FIGURA 3.5: CONFIGURACIÓN CRISTAL DE CUARZO	37
FIGURA 3.6: CIRCUITO DE PROGRAMACIÓN DE MICROCONTROLADOR	38
FIGURA 3.7: ADAPTADOR DE VOLTAJE PARA XBEE	39
FIGURA 3.8: CONEXIÓN USB	40
FIGURA 3.9: CIRCUITO REGULADOR DE VOLTAJE A LA ENTRADA	41
FIGURA 3.10: CONEXIÓN DE MAX485.....	43
FIGURA 3.11: ANALIZADOR DE REDES ELÉCTRICAS	44
FIGURA 3.12: DIAGRAMA DE CONEXIÓN ANALIZADOR DE REDES	44
FIGURA 3.13: ENTORNO VISUAL STUDIO 2012	46
FIGURA 3.14: ENTORNO DE TRABAJO PIC C	47
FIGURA 3.15: XBEE EXPLORER	48
FIGURA 3.16: ESQUEMA DE COMUNICACIÓN ESCLAVO	49
FIGURA 3.17: CONFIGURACIÓN DE COMUNICACIÓN CON MÓDULO XBEE	51
FIGURA 3.18: VENTANA DE CONFIGURACIÓN DE RED X-CTU	52
FIGURA 4.1: INTERFAZ SCADA CON GRÁFICA DE BARRAS.....	55
FIGURA 4.2: INTERFAZ SCADA CON CADENA DE TEXTO RECIVIDA	56
FIGURA 4.3: UBICACIÓN DE DISPOSITIVOS.....	57

FIGURA 4.4: RESULTADOS DE PRUEBA A 40M. GRÁFICO	58
FIGURA 4.5: RESULTADOS DE PRUEBA A 40M. TEXTO	58
FIGURA 7.1: INTERFAZ VISUAL SCADA.....	68
FIGURA 7.2: DIAGRAMA DE COORDINADOR.....	71
FIGURA 7.3: DIAGRAMA DE DISPOSITIVO FINAL	72

Resumen

Las redes de sensores inalámbricos son una tecnología para una amplia variedad de aplicaciones en ingeniería debido a su fácil instalación y mantenimiento. Usada inicialmente en aplicaciones militares, ahora se ha extendido a muchas áreas industriales y de interés social, como controles del proceso de producción, monitorización de la salud, domótica o control de tráfico.

Este proyecto está enfocado en el diseño de una red inalámbrica para una planta productora de muebles. El diseño consiste en centralizar la medición de voltaje, corriente, potencia, frecuencia y factor de potencia, mediante tecnología inalámbrica y crear un software específico para poder supervisar los datos obtenidos.

Para añadir mayor funcionalidad al proyecto, se ha creado una aplicación para PC, cuya función es permitir visualizar los datos obtenidos desde un punto con la posibilidad de almacenar la información en un base de datos para su posterior análisis.

Entre las principales tecnologías usadas para llevar a cabo la realización de este proyecto se puede destacar la comunicación inalámbrica ZigBee (IEEE 802.15.4), la comunicación bus USB PC a USB PIC18F2550, programación en Visual Studio C# y bases de datos MySQL.

Abstract

Wireless Sensor Networks are technology for a wide variety of engineering applications due to its easy installation and maintenance. Initially used in military applications, has now spread to many areas of industrial and social interest, such as production process controls, health monitoring, home automation and traffic control.

This project focuses on the design of a wireless network for a cabinet factory. The design is to centralize energy and power management data using wireless technology and create specific software to monitor data.

To add more functionality to the project, has created a PC application whose function is to allow displaying and insert into database the measured energy and power data to analyze later .

The main technologies used to carry out this project can highlight the wireless ZigBee (IEEE 802.15.4), the communication bus PIC18F2550 USB to PC USB, programming in Visual Studio C# and MySQL databases.

1 Introducción

1.1 Presentación

El proyecto presenta la necesidad de una fábrica de cocinas y closets para tener una correcta medición energética, todo esto con la finalidad de reducir los pagos a la compañía de luz así como poder tener las bases para costear los gastos energéticos por cada producto.

En primera instancia se describe el planteamiento general del problema poniendo así un marco de referencia de tal modo que el lector tiene un contexto general de la problemática a resolver así como una descripción de cómo se encuentra la tecnología actual en este tipo de proyectos.

En el desarrollo se muestra el proceso para realizar la medición mediante un analizador de redes eléctricas en un punto y que es repetible para cada punto de medición adicional en un futuro. Se podrá observar los pasos que se siguieron para resolver el tema, así como los diagramas técnicos de conexión y comunicación.

1.2 Planteamiento del problema

Las empresas en México han sido testigos como la energía eléctrica ha pasado de representar un factor poco representativo en su estructura de costos a ser un tema sumamente importante en la misma, todo esto debido al incremento paulatino en su precio. Se tiene que afrontar el reto de disminuir la participación de la energía en los costos de las diferentes áreas, o por lo menos no aumentar mes a mes.

Para que esto se pueda dar, es preciso conocer la cantidad de energía que se utiliza en cada uno de los procesos que conforman la producción y determinar las acciones pertinentes para abaratar los costos, sin reducir o comprometer la calidad ni la cantidad de producción.

Otra problemática encontrada es que al no saber cuánto se consume de energía eléctrica con cada uno de los procesos o estaciones de trabajo, es difícil medir el impacto del costo de la energía en el valor del producto final. Visto de esta manera, se puede tener un análisis detallado para poder calcular cuánto cuesta en materia de energía eléctrica la fabricación de un mueble.

Pueden existir consumos “fantasmas” durante el proceso de producción que pueden ser identificados e inclusive podría cambiar la perspectiva con respecto al uso de las máquinas, es

decir, podría tener un mayor consumo de energía un proceso de producción diferente en lugar de lo que se piensa actualmente.

Por último se tiene la evaluación de la calidad de la energía y que impacto tiene en la producción de tal forma que las variaciones de energía suministradas por la Comisión Federal de Electricidad puedan ser monitoreadas y controladas en el proceso productivo.

Para alcanzar todo lo anterior es necesario implementar un programa de medición y ahorro de energía cuyo objetivo central es el uso eficiente de la energía.

1.3 Objetivos

1.3.1 Objetivo general

Desarrollar una interfaz de adquisición de datos inalámbrica para obtener información energética de máquinas industriales (Voltaje, corriente, potencia, factor de potencia, frecuencia) con el almacenaje en una base de datos y el despliegue visual de la información en un software para computadora.

1.3.2 Objetivos específicos

- Analizar las redes de comunicación Modbus RTU y su implementación con micro controladores
- Implementar una red tipo estrella mediante el uso de radio frecuencia para obtener comunicación inalámbrica.
- Seleccionar analizador de redes electricas capaz de trabajar en ambientes industriales a los niveles de voltaje y corriente de la fábrica.
- Determinar tecnología necesaria para realizar la medición de datos energéticos.
- Desarrollar un software capaz de almacenar la información generada por la estación de medición.

1.4 Justificación

La necesidad parte de la solicitud de la fábrica por reducir costos en el tema del consumo energético, cada vez se vuelve crucial para las empresas moderar y/o eficientar sus consumos, otro punto tratado es el requerimiento de costear la energía necesaria para transformar la materia prima en cada proceso productivo ya que esto es de suma importancia para fijar un precio correcto a los productos teniendo márgenes de ganancia optimos.

Otra ventaja que dará a futuro el proyecto, es la generación de gráficas estadísticas del comportamiento electrico de la planta llevando un registro de control para cerciorar que las cargas que generan un consumo en la fábrica no sean operadas en horas inadecuadas, un claro ejemplo es que el router CNC que requiere 4 motores de 40 HP no sea operado en horario pico de consumo.

Se requiere un dispositivo flexible, es decir que el mismo dispositivo pueda repetirse para cada una de las estaciones de trabajo y zonas de la empresa, que en un futuro se lleve a otros departamentos adicionales a producción, por ejemplo, medir el consumo de aires acondicionados del departamento de administración. A un mediano plazo la meta es tener un reporte como el que se muestra en la Figura 1.1.



Figura 1.1: Datos energéticos por área

1.5 Estado del arte

La comunicación en la industria moderna cada vez se vuelve un punto más importante e imprescindible. Lo común es encontrar siempre sistemas que están conformados de diferentes fabricantes y con diferentes funcionalidades dependiendo del nivel de complejidad de los sistemas; que pese a que se pueden encontrar en puntos diferentes dentro de la misma planta, se requiere que trabajen en conjunto y coordinadamente para tener un proceso con resultados satisfactorios y óptimos. El objetivo principal es tener todo completamente integrado en un solo sistema y para dar solución a esto se crearon las redes industriales. Seguido la creación de las redes tenemos los sistemas de comunicación y control industrial donde se integran sistemas de supervisión control y adquisición de datos (SCADA), sistemas de control distribuidos (DCS) y otros sistemas de control como pueden ser controladores lógicos programables (PLC) [1]. Tener todos los sistemas mencionados anteriormente nos hace llegar a tener lo que se denomina Manufactura Integrada por Computadora (CIM).

En los últimos años, las aplicaciones industriales basadas en sistemas de comunicación digital se han incrementados pero no solo eso sino que algunos fabricantes han optado por utilizar estándares con la finalidad de poder integrar a un mismo sistema productos de su competencia [2], los sistemas industriales cada vez son más complejos a diferencia de las redes comerciales en las cuales se demandan menos elementos como se muestra en la Figura 1.2:

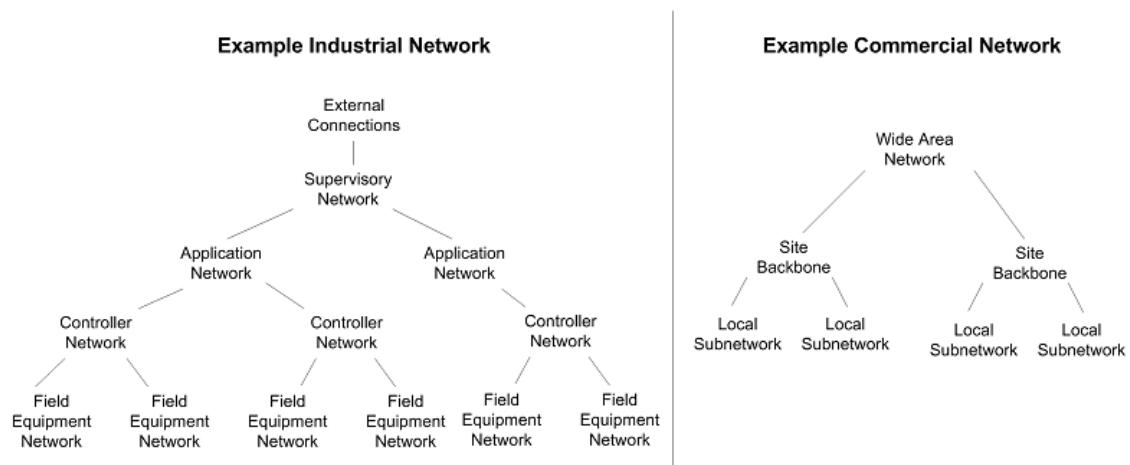


Figura 1.2: Diferencia entre arquitectura de red industrial y red comercial [1]

Las ventajas de tener una red industrial donde coexisten una serie de equipos y dispositivos dedicados al control de una máquina o de un proceso son las siguientes:

- Poder visualizar y supervisar todo el proceso productivo.
- Adquirir datos de manera instantánea.
- Mejorar el rendimiento del proceso.
- Intercambio de datos entre diferentes sectores del proceso y departamentos.
- Programación y/o monitoreo a distancia.

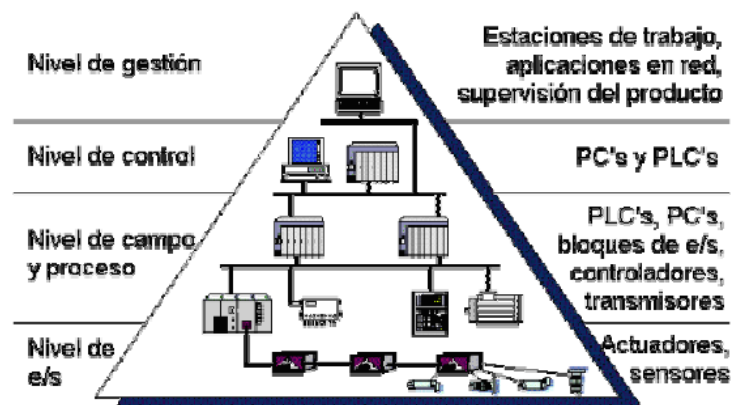


Figura 1.3: Niveles de una red industrial [3]

En una red industrial coexisten todo tipo de dispositivos, los cuales se pueden agrupar jerárquicamente y de este modo tener una pirámide que muestre más claramente la relación entre cada uno de los elementos que conforman la red (Figura 1.3).

Se definen cuatro niveles de manera tradicional en una red industrial [4]:

- **Nivel de gestión:** Es el nivel más elevado, puede o no encontrarse en el mismo sitio donde está la fábrica y es el encargado de integrar los niveles subsecuentes. Las máquinas conectadas a los sistemas de este nivel suelen ser estaciones de trabajo que funcionan como puente entre el proceso productivo y el área de gestión así como el enlace con los sistemas de Planeación de Recursos Empresariales (ERP) [1]. En este nivel la red comúnmente implementada son del LAN (Red de área local) o WAN (Red de área amplia).
- **Nivel de control:** Es el encargado de enlazar y dirigir distintas zonas de trabajo, en este nivel se sitúan autómatas de gama alta y servidores y/o computadores encargados del

diseño, control de calidad, configuración, etc. En este nivel la red comúnmente implementada es LAN.

- **Nivel de campo y proceso:** En este nivel se pueden encontrar autómatas compactos, multiplexores de entrada y salida, sistemas de control, etc. Es en este nivel donde las redes industriales son más efectivas ya que pueden estar presentes uno o varios autómatas modulares actuando como maestros o esclavos. A este nivel se manejan diferentes tipo de buses (serán explicados más adelante), el cual será elegido dependiendo de las necesidades, economía y fabricante mayoritario en el sistema.
- **Nivel de entradas y salidas:** Es el nivel que se encuentra inmerso en el proceso, aquí están todos los sensores y actuadores encargados de manejar el proceso productivo y tomar todas las medidas necesarias para que el proceso fluya de manera adecuada. A este nivel lo común y recomendado es tener buses sencillos y sistemas de periferia descentralizada.

En la actualidad encontramos diferentes protocolos de comunicación (más adelante se definirá lo que es un protocolo) dependiendo del tipo de aplicación, donde los que se aplican actualmente en la mayor parte de la industria son los que aparecen en Figura 1.4.

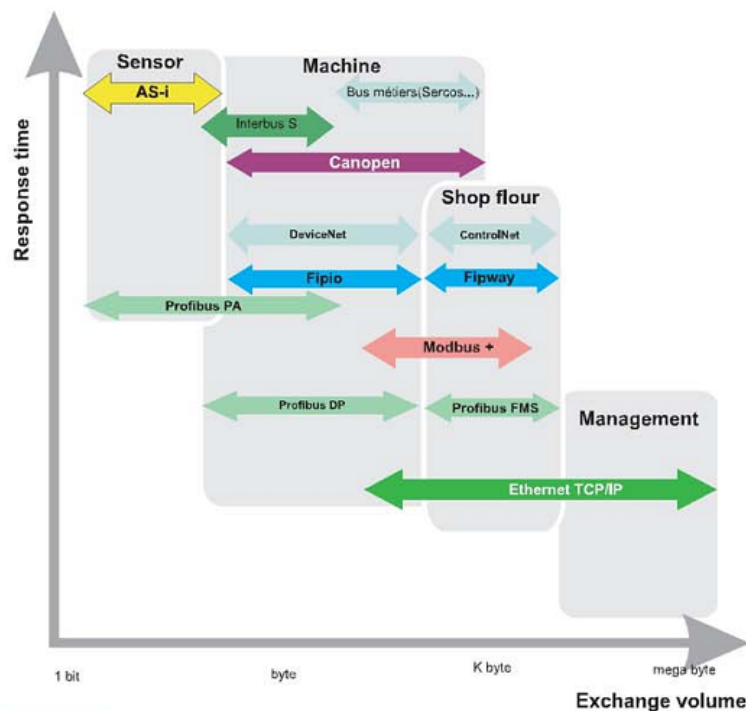


Figura 1.4: Principales Redes Industriales [5]

Dependiendo de la cantidad de información que se intercambia entre los elementos de la red y la velocidad necesaria de respuesta es la que se elegirá. A continuación se describirán algunas características de las principales redes:

Modbus TCP:

Es un protocolo Ethernet completamente abierto, al desarrollar una aplicación con este protocolo no se requiere ningún componente propietario o licencia de compra y el protocolo se puede utilizar en cualquier dispositivo que soporte un estándar TCP/IP.

El Ethernet Modbus TCP tiene un protocolo muy sencillo y una salida de 100 Mbps lo cual garantiza excelentes prestaciones al momento de comunicar y transmitir información entre autómatas permitiendo que sea la red más usada en aplicaciones de tiempo real. Otra ventaja importante y superior a otras redes que utilizan TCP/IP es que es compatible con Modbus tradicional, lo cual permite direccionar mensajes a estos dispositivos sin tener que cambiar de protocolo. Otra ventaja es que al ser IP se puede utilizar el direccionamiento para poder controlar o comunicarse con un dispositivo en cualquier parte del mundo sin importar la distancia.

Modbus:

La designación Modbus Modicon corresponde a una marca registrada por Gould Inc. Como en tantos otros casos, la designación no corresponde propiamente al estándar de red, incluyendo todos los aspectos desde el nivel físico hasta el de aplicación, sino a un protocolo de enlace (nivel OSI 2). Puede, por tanto, implementarse con diversos tipos de conexión física y cada fabricante suele suministrar un software de aplicación propio, que permite parametrizar sus productos. [6]

No obstante, se suele hablar de MODBUS como un estándar de bus de campo, cuyas características esenciales son las que se detallan a continuación:

Estructura de red

Medio Físico

El medio físico de conexión puede ser un bus semidúplex (half duplex) (RS-485 o fibra óptica) o dúplex (full duplex).

La comunicación es asíncrona y las velocidades de transmisión previstas van desde los 75 baudios a 19.200 baudios. La máxima distancia entre estaciones depende del nivel físico, pudiendo alcanzar hasta 1200 m sin repetidores.

Acceso al Medio

La estructura lógica es del tipo maestro-esclavo, con acceso al medio controlado por el maestro.

El número máximo de estaciones previsto es de 63 esclavos más una estación maestra.

Los intercambios de mensajes pueden ser de dos tipos:

- Intercambios punto a punto, que comportan siempre dos mensajes: una demanda del maestro y una respuesta del esclavo (puede ser simplemente un reconocimiento («acknowledge»).
- Mensajes difundidos. Estos consisten en una comunicación unidireccional del maestro a todos los esclavos. Este tipo de mensajes no tiene respuesta por parte de los esclavos y se suelen emplear para mandar datos comunes de configuración, reset, etc.

Protocolo

La codificación de datos dentro de la trama puede hacerse en modo ASCII o puramente binario, según el estándar RTU (Remote Transmission Unit). En cualquiera de los dos casos, cada mensaje obedece a una trama que contiene cuatro campos principales, según se muestra en la Figura 1.5. La única diferencia estriba en que la trama ASCII incluye un carácter de encabezamiento («:»=3AH) y los caracteres CR y LF al final del mensaje.

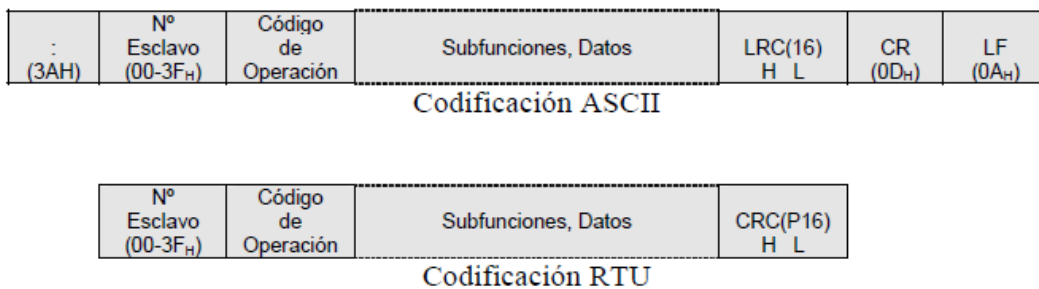


Figura 1.5 : Tipos de codificaciones Modbus

Pueden existir también diferencias en la forma de calcular el CRC, puesto que el formato RTU emplea una fórmula polinómica en vez de la simple suma en módulo 16. Con independencia de

estos pequeños detalles, a continuación se da una breve descripción de cada uno de los campos del mensaje:

Número de esclavo (1 byte):

Permite direccionar un máximo de 63 esclavos con direcciones que van del 01H hasta 3FH. El número 00H se reserva para los mensajes difundidos.

Código de operación o función (1 byte):

Cada función permite transmitir datos u órdenes al esclavo. Existen dos tipos básicos de órdenes:

- Ordenes de lectura/escritura de datos en los registros o en la memoria del esclavo.
- Ordenes de control del esclavo y el propio sistema de comunicaciones .

La Tabla 1.1 muestra la lista de funciones disponibles en el protocolo MODBUS con sus correspondientes códigos de operación.

Tabla 1.1: Funciones del protocolo Modbus

Función	Código	Tarea
0	00 _H	Control de estaciones esclavas
1	01 _H	Lectura de n bits de salida o internos
2	02 _H	Lectura de n bits de entradas
3	03 _H	Lectura de n palabras de salidas o internos
4	04 _H	Lectura de n palabras de entradas
5	05 _H	Escritura de un bit
6	06 _H	Escritura de una palabra
7	07 _H	Lectura rápida de 8 bits
8	08 _H	Control de contadores de diagnósticos número 1 a 8
9	09 _H	No utilizado
10	0A _H	No utilizado
11	0B _H	Control del contador de diagnósticos número 9
12	0C _H	No utilizado
13	0D _H	No utilizado
14	0E _H	No utilizado
15	0F _H	Escritura de n bits
16	10 _H	Escritura de n palabras

Campo de subfunciones/datos (n bytes):

Este campo suele contener, en primer lugar, los parámetros necesarios para ejecutar la función indicada por el byte anterior. Estos parámetros podrán ser códigos de subfunciones en el caso de órdenes de control (función 00H) o direcciones del primer bit o byte, número de bits o palabras a leer o escribir, valor del bit o palabra en caso de escritura, etc.

Palabra de control de errores (2 bytes):

En código ASCII, esta palabra es simplemente la suma de comprobación ('checksum') del mensaje en módulo 16 expresado en ASCII. En el caso de codificación RTU el CRC se calcula con una fórmula polinómica según el algoritmo mostrado en la Figura 1.6.

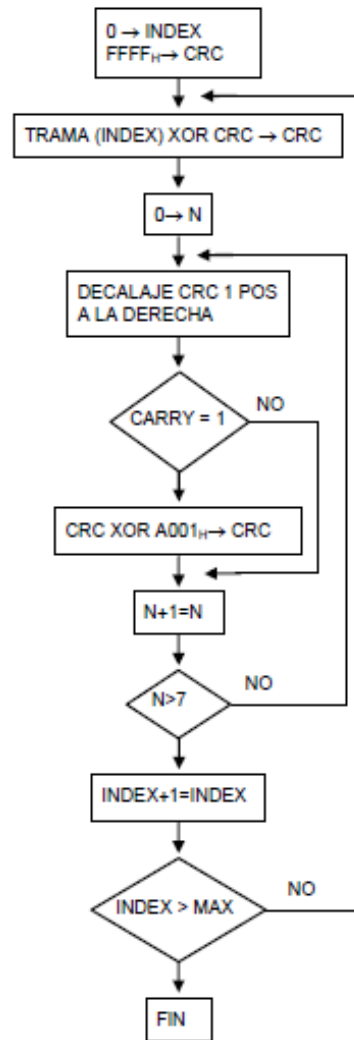


Figura 1.6: Cálculo de CRC en codificación RTU

CANOpen:

El motivo principal para utilizar esta red son sus prestaciones y su flexibilidad de adaptación, su modo de operación es del tipo productor/consumidor, lo cual hace que los datos solo se transfieren si es necesario o en una escala de tiempo específica.

Esta red es una variante del CAN donde sus principales ventajas es que nos encontramos frente a una red muy fiable ya que cuando un dispositivo genera un mensaje, en automático asigna una prioridad a ese mensaje para ser atendido, la pérdida de información por temas de colisiones es imposible ya que siempre se espera al próximo estado a recibir, lo cual la hace completamente fiable y es por ello que es frecuentemente utilizada en la industria médica. Las redes CAN cuentan con una gran inmunidad al ruido e interferencias electromagnéticas, esto hace que las máquinas o una planta trabajen con suficiente precisión. La velocidad de transferencia es otro de sus puntos fuertes, ya que una pérdida de tiempo en la industria se transforma en pérdida de dinero. CANOpen es la red más confiable cuenta con excelentes mecanismos para detección de errores y corrección de los mismos, la probabilidad de que un error pase sin ser detectado es uno cada mil años.

El bajo coste de implementación la hace muy atractiva ya que al ser implementada en grandes cantidades ha hecho que los costos de los productos que manejan se abaraten en promedio implementar este tipo de red es entre un 10 y 20% más económico que otros buses de campo.

AS-Interface:

En el más bajo nivel de la pirámide donde se encuentran los actuadores y sensores, es donde la cantidad de información que se transmite es mínima (binaria) pero donde la velocidad de transmisión debe ser instantánea. Este tipo de bus transporta la señal y potencia en un único cable de dos hilos lo cual hace que la instalación y mantenimiento sea mucho más sencillo [4].

Es una tecnología abierta compatible con los fabricantes de automatismos líderes del mercado lo cual garantiza la inter operatividad de fabricantes en una misma solución. El método de funcionamiento es Maestro/Esclavo, donde el maestro suele ser algún autómatas.

Actualmente esta red se encuentra en su segunda versión (V2) donde el cambio más grande que sufrió es permitir hasta 62 esclavos por maestro y la capacidad de poder mezclar dispositivos analógicos y digitales.

Entre sus principales ventajas nos encontramos lo siguiente:

- Su simplicidad de cableado al requerir un único cable para conectar todos los actuadores y sensores.
- Los costos pueden llegar a reducir hasta un 40% debido a los tiempos de diseño de la solución así como los tiempos de instalación e implementación.
- Ahorro de espacio gracias a los dispositivos que implementan este bus son más compactos y no se requiere dispositivos intermediarios.
- La seguridad en las conexiones, esto gracias a la implementación de un sistema Poka Yoke donde los errores por mala conexión del cableado son imposibles, además de tener inmunidad a las interferencias electromagnéticas.

Medición de corriente

Respecto a la medición de corriente en la actualidad se siguen utilizando básicamente los mismos métodos desde ya hace muchos años basados en principios eléctricos, magnéticos, ópticos o utilizando el comportamiento de algunos materiales frente a los campos magnéticos. En la actualidad adicional a los métodos tradicionales existe un principio de medición denominado “Transformador Flux Gate” [7] el cual se basa en una estructura física similar al transformador de efecto Hall, basado en la detección del estado de saturación de un circuito magnético, el cual es excitado por una señal exterior. La inyección de corriente en un devanado auxiliar crea un campo magnético de compensación que restaura la simetría del ciclo de histéresis. La corriente inyectada compensa el campo magnético creado por la corriente medida, y su valor es proporcional a la corriente.

Este tipo de sistemas es apto para la medición de sistemas de corriente continua y de corriente alterna con una gran precisión y de gran valor, y con frecuencias de hasta 100 kHz. En la Figura 1.7 se muestra el principio de funcionamiento del transductor Flux Gate.

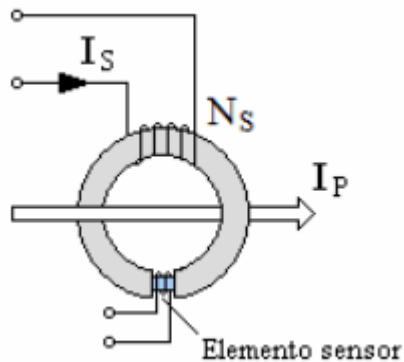


Figura 1.7: Principio de medición de un transductor Flux Gate [7]

2 Marco teórico

2.1 Topologías de red.

Unas de las principales topologías que existen en las redes inalámbricas son:

- Bus
- Estrella
- Árbol
- Malla
- Anillo

Red en Bus:

Topología de red en la que todas las estaciones están conectadas a un único canal de comunicaciones por medio de unidades interfaz y derivadores. Las estaciones utilizan este canal para comunicarse con el resto. La topología de bus tiene todos sus nodos conectados directamente a un enlace y no tiene ninguna otra conexión entre nodos. Físicamente cada host está conectado a un cable común, por lo que se pueden comunicar directamente, aunque la ruptura del cable hace que los hosts queden desconectados. La topología de bus permite que todos los dispositivos de la red puedan ver todas las señales de todos los demás dispositivos, lo que puede ser ventajoso si desea que todos los dispositivos obtengan esta información. Sin embargo, puede representar una desventaja, ya que es común que se produzcan problemas de tráfico y colisiones, que se pueden paliar segmentando la red en varias partes. Es la topología más común en pequeñas LAN, con hub o switch final en uno de los extremos.



Figura 2.1: Topología de Bus

Red en estrella:

Red en la cual las estaciones están conectadas directamente al servidor u ordenador y todas las comunicaciones se han de hacer necesariamente a través de él. Todas las estaciones están conectadas por separado a un centro de comunicaciones, concentrador o nodo central, pero no están conectadas entre sí. Esta red crea una mayor facilidad de supervisión y control de información ya que para pasar los mensajes deben pasar por el hub o concentrador, el cual gestiona la redistribución de la información a los demás nodos. La fiabilidad de este tipo de red es que el malfuncionamiento de un ordenador no afecta en nada a la red entera, puesto que cada ordenador se conecta independientemente del hub, el costo del cableado puede llegar a ser muy alto. Su punto débil consta en el hub ya que es el que sostiene la red.



Figura 2.2: Topología de estrella

Red de árbol:

Topología de red en la que los nodos están colocados en forma de árbol. Desde una visión topológica, la conexión en árbol es parecida a una serie de redes en estrella interconectadas.

Es una variación de la red en bus, la falla de un nodo no implica interrupción en las comunicaciones. Se comparte el mismo canal de comunicaciones.

Cuenta con un cable principal al que hay conectadas redes individuales en bus.



Figura 2.3: Topología de árbol

Red en malla:

La Red en malla es una topología de red en la que cada nodo está conectado a uno o más de los otros nodos. De esta manera es posible llevar los mensajes de un nodo a otro por diferentes caminos.

Si la red de malla está completamente conectada no puede existir absolutamente ninguna interrupción en las comunicaciones. Cada servidor tiene sus propias conexiones con todos los demás servidores.



Figura 2.4: Topología en malla

Red en anillo:

Topología de red en la que las estaciones se conectan formando un anillo. Cada estación está conectada a la siguiente y la última está conectada a la primera. Cada estación tiene un receptor y un transmisor que hace la función de repetidor, pasando la señal a la siguiente estación del anillo.

Cabe mencionar que si algún nodo de la red se cae (termino informático para decir que está en mal funcionamiento o no funciona para nada) la comunicación en todo el anillo se pierde.



Figura 2.5: Topología en anillo

2.2 Redes industriales.

Muchas veces escuchamos en la industria la palabra protocolos de comunicación sin tener claro de que estamos hablando. Con el objeto de familiarizar a los lectores, se expondran sus principales características y fundamentos de los más utilizados. En principio un protocolo de comunicación es un conjunto de reglas que permiten la transferencia e intercambio de datos entre los distintos dispositivos que conforman una red. Estos han tenido un proceso de evolución gradual a medida que la tecnología electrónica ha avanzado y muy en especial en lo que se refiere a los microprocesadores. Un importante número de empresas en nuestro país presentan la existencia de islas automatizadas (células de trabajo sin comunicación entre sí), siendo en estos casos las redes y los protocolos de comunicación Industrial indispensables para realizar un enlace entre las distintas etapas que conforman el proceso.

La irrupción de los microprocesadores en la industria han posibilitado su integración a redes de comunicación con importantes ventajas, entre las cuales figuran:

- Mayor precisión derivada de la integración de tecnología digital en las mediciones.

- Mayor y mejor disponibilidad de información de los dispositivos de campo.
- Diagnóstico remoto de componentes

La integración de las mencionadas islas automatizadas suele hacerse dividiendo las tareas entre grupos de procesadores jerárquicamente anidados, esto da lugar a una estructura de redes Industriales, las cuales es posible agrupar en tres categorías:

- Buses de campo
- Redes LAN
- Redes LAN-WAN

En este documento se referirá a los protocolos de comunicación más usados en la industria y se hará una pequeña comparativa de los mismos (Tabla 2.1). Los buses de datos que permiten la integración de equipos para la medición y control de variables de proceso, reciben la denominación genérica de buses de campo. Un bus de campo es un sistema de transmisión de información (datos) que simplifica enormemente la instalación y operación de máquinas y equipamientos industriales utilizados en procesos de producción. El objetivo de un bus de campo es sustituir las conexiones punto a punto entre los elementos de campo y el equipo de control a través del tradicional lazo de corriente de 4-20mA o 0 a 10V DC, según corresponda. Generalmente son redes digitales, bidireccionales, multipunto, montadas sobre un bus serie, que conectan dispositivos de campo como PLC's, transductores, actuadores, sensores y equipos de supervisión. Varios grupos han intentado generar e imponer una norma que permita la integración de equipos de distintos proveedores. Sin embargo, hasta la fecha no existe un bus de campo universal. Los buses de campo con mayor presencia en el área de control y automatización de procesos son:

- Profibus
- Fieldbus Foundation
- Modbus
- DeviceNet

PROFIBUS (Process Field Bus) Norma internacional de bus de campo de alta velocidad para control de procesos normalizada en Europa por EN 50170.

Existen tres perfiles:

- Profibus DP (Decentralized Periphery). Orientado a sensores/actuadores enlazados a procesadores (PLCs) o terminales.
- Profibus PA (Process Automation). Para control de proceso, cumple normas especiales de seguridad para la industria química (IEC 1 1 15 8-2, seguridad intrínseca).
- Profibus FMS (Fieldbus Message Specification). Para comunicación entre células de proceso o equipos de automatización.

FOUNDATION FIELDBUS (FF) es un protocolo de comunicación digital para redes industriales, específicamente utilizado en aplicaciones de control distribuido. Puede comunicar grandes volúmenes de información, ideal para aplicaciones con varios lazos complejos de control de procesos y automatización.

Está orientado principalmente a la interconexión de dispositivos en industrias de proceso continuo. Los dispositivos de campo son alimentados a través del bus Fieldbus cuando la potencia requerida para el funcionamiento lo permite.

MODBUS es un protocolo de transmisión para sistemas de control y supervisión de procesos (SCADA) con control centralizado, puede comunicarse con una o varias Estaciones Remotas (RTU) con la finalidad de obtener datos de campo para la supervisión y control de un proceso.

La Interfaces de Capa Física puede estar configurada en:: RS-232, RS-422, RS-485. En Modbus los datos pueden intercambiarse en dos modos de transmisión:

- Modo RTU
- Modo ASCII

DEVICENET Red de bajo nivel adecuada para conectar dispositivos simples como sensores fotoeléctricos, sensores magnéticos, pulsadores, etc y dispositivos de alto nivel (PLC, controladores, computadores, HMI, entre otros). Provee información adicional sobre el estado de la red, cuyos datos serán desplegados en la interfaz del usuario.

Tabla 2.1: Comparativa de principales redes

Nombre	Topología	Soporte	Dispositivos Máx	Vel. De transmisión	Distancia máx (KM)	Tipo de comunicación
Profibus DP	Bus, estrella y anillo	Par trenzado, fibra óptica	127	Hasta 12M	0.1Par 24 fibra	Master/Slave
Profibus PA	Bus, estrella y anillo	Par trenzado, fibra óptica	14400	31.5K	0.1Par 24 fibra	Master/Slave
Profibus FMS	Estrella	Par trenzado, fibra óptica	127	500K	0.1Par 24 fibra	Master/Slave
Fieldbus	Estrella	Par trenzado, fibra óptica	240	100M	0.1Par 2 fibra	Single/Multi Master
DeviceNet		Par trenzado, fibra óptica	2048	500K	0.5	Master/Slave/Multi Master
Modbus	Bus, estrella, árbol	Par trenzado, coaxial.	255	1.2K a 115.2K	1.2	Master/Slave

2.3 Descripción de funciones del protocolo Modbus.

Como se observa en la Tabla 1.1 del Estado del arte existen una serie de funciones propias del protocolo Modbus las cuales contienen el mensaje enviado por el maestro y la respuesta generada por el esclavo, algunas de ellas pueden contener sub funciones, a continuación se describe cada una de las funciones.

Función 0:

Esta función permite ejecutar órdenes de control, tales como marcha, paro, carga y lectura de programas de usuario del autómatas. Para codificar cada una de las citadas órdenes se emplean

los cuatro primeros bytes del campo de datos. La trama resultante es la representada en la Figura 2.6 y la interpretación de los códigos de subfunción se especifica en la Tabla 2.2.

En caso de las órdenes de marcha y paro, el campo de «información» de la trama representada en la Figura 2.6 está vacío y, por tanto, el mensaje se compone simplemente de 6 bytes de función más 2 bytes de CRC. La respuesta del esclavo a estas órdenes es un mensaje idéntico al enviado por el maestro. Cabe señalar, además, que después de un paro el autómatas sólo acepta ejecutar subfunciones de la función 00H.

Nº Esclavo (00-3FH)	00 _H	Código Subfunción SF0 SF1	Datos Subfunción D0 D1	Información	CRC(16) H L
---------------------------	-----------------	---------------------------------	------------------------------	-------------	----------------

Figura 2.6: Trama genérica de las subfunciones de control de esclavos.

Tabla 2.2: Subfunciones de las función 00H

Código subfunción SF0 SF1		Datos subfunción D0 D1		Tarea
00 _H	00 _H	00 _H	00 _H	Paro del esclavo sin inicializar
00 _H	01 _H	00 _H	00 _H	Marcha del esclavo sin inicializar
00 _H	02 _H	00 _H	00 _H	Marcha e inicialización del esclavo
00 _H	03 _H	00 _H	XX _H	Lectura de la secuencia XX de programa de usuario en el esclavo
00 _H	04 _H	YY _H	XX _H	Carga de una secuencia de programa de usuario en el esclavo Petición: YY = secuencia a cargar, XX= próxima secuencia Respuesta: XX= código error, YY= 00

Funciones 1 y 2:

Lectura de bits del autómeta. La trama es la indicada en la Figura 2.7. La forma de direccionamiento de los bits es a base de dar la dirección de la palabra que los contiene y luego la posición del bit. Obsérvese también que la respuesta es dada siempre en octetos completos.

Petición del maestro

Nº Esclavo (00-3F _H)	01 _H o 02 _H	Dirección 1 ^{er} Bit PP PB	Nº de Bits NN NN	CRC H L
----------------------------------	---	--	---------------------	------------

PPP = Dirección de la palabra (hex), B= Dirección del bit dentro de la palabra 0 a F_H.

Respuesta del esclavo

Nº Esclavo (00-3F _H)	01 _H o 02 _H	Nº Octetos leídos NN NN	1 ^{er} Octeto B7..B0	Otros Octetos Hasta máx. 256	CRC H L
----------------------------------	---	----------------------------	----------------------------------	------------------------------------	------------

Figura 2.7: Petición y respuesta de la función 01H y 02H

Funciones 3 y 4:

Lectura de palabras del autómeta. La trama es la indicada en la Figura 2.8. Obsérvese que la petición indica el número de palabras a leer, mientras que en la respuesta se indica el número de octetos leídos.

Petición del maestro

Nº Esclavo (00-3F _H)	03 _H o 04 _H	Dirección 1ª Palabra PP PP	Nº de Palabras NN NN	CRC H L
----------------------------------	---	-------------------------------	-------------------------	------------

PPPP = Dirección de la palabra (hex)

Respuesta del esclavo

Nº Esclavo (00-3F _H)	03 _H o 04 _H	Nº Octetos leídos NN NN	1 ^{er} Palabra H L	Otras Palabras Hasta máx. 128 H L,H L,HL,....	CRC H L
----------------------------------	---	----------------------------	--------------------------------	---	------------

Figura 2.8: Petición y respuesta de la función 03H y 04H

Función 5:

Escritura de un bit. La trama es la indicada en la Figura 2.9. El direccionamiento del bit se efectúa tal como se ha indicado para las funciones 1 y 2.

Petición del maestro

Nº Esclavo (00-3F _H)	05 _H	Dirección Bit PP PB	XX _H	00 _H	CRC H L
----------------------------------	-----------------	---------------------	-----------------	-----------------	---------

PPP = Dirección de la palabra (hex), B= Dirección del bit dentro de la palabra 0 a F_H.

Respuesta del esclavo

Nº Esclavo (00-3F _H)	05 _H	Dirección Bit PP PB	XX _H	00 _H	CRC H L
----------------------------------	-----------------	---------------------	-----------------	-----------------	---------

XX_H = 00H para bit = 0 y XX_H = FF_H para bit = 1

Figura 2.9: Petición y respuesta de la función 05H

Función 6:

Escritura de una palabra. La trama es la indicada en la Figura 2.10.

Petición del maestro

Nº Esclavo (00-3F _H)	06 _H	Dirección Palabra PP PP	Valor Palabra DD DD	CRC H L
----------------------------------	-----------------	-------------------------	---------------------	---------

Respuesta del esclavo

Nº Esclavo (00-3F _H)	06 _H	Dirección Palabra PP PP	Nº de Palabras DD DD	CRC H L
----------------------------------	-----------------	-------------------------	----------------------	---------

Figura 2.10: Petición y respuesta de la función 06H

Función 7:

Petición de lectura rápida de un octeto. La trama es la mostrada en la Figura 2.11. Obsérvese que la petición no tiene campo de dirección, esto es debido a que el octeto legible por esta función es fijo en cada esclavo y viene fijado en su configuración.

Petición del maestro

Nº Esclavo (00-3F _H)	07 _H	CRC H L
--	-----------------	------------

Respuesta del esclavo

Nº Esclavo (00-3F _H)	07 _H	Valor Octeto DD	CRC H L
--	-----------------	-----------------------	------------

Figura 2.11: Lectura de un octeto

Función 8:

Petición del contenido y control de los 8 primeros contadores de diagnóstico de un esclavo (véase Tabla 2.3). Las tramas de petición y respuesta pueden verse en la Figura 2.12.

Petición del maestro

Nº Esclavo (00-3F _H)	08 _H	Código Subfunción SF0 SF1	Dato Subfunción D0 D1	CRC H L
--	-----------------	---------------------------------	-----------------------------	------------

Respuesta del esclavo

Nº Esclavo (00-3F _H)	08 _H	Código Subfunción SF0 SF1	Valor Contador H L	CRC H L
--	-----------------	---------------------------------	--------------------------	------------

Figura 2.12: Control de contadores

Tabla 2.3: Contadores de diagnóstico

Subfunción Nº	Código		Datos		Tarea
			D0	D1	
0	00 _H	00 _H	XY _H	ZT _H	El esclavo envía el eco XYZT de petición como test.
3	00 _H	03 _H	ZZ _H	00 _H	Modifica el carácter de fin de trama en modo ASCII por ZZ _H
10	00 _H	0A _H	00 _H	00 _H	Puesta a cero de los contadores
11	00 _H	0B _H	00 _H	00 _H	Lectura del contador 1
12	00 _H	0C _H	00 _H	00 _H	Lectura del contador 1
13	00 _H	0D _H	00 _H	00 _H	Lectura del contador 1
14	00 _H	0E _H	00 _H	00 _H	Lectura del contador 1
15	00 _H	0F _H	00 _H	00 _H	Lectura del contador 1
18	00 _H	12 _H	00 _H	00 _H	Lectura del contador 1

Función 11:

La petición del contenido del contador de diagnóstico número 9, no se realiza por la función 8, sino por la función 11. Las tramas de petición y respuestas son las indicadas por la Figura 2.13.

Petición del maestro

Nº Esclavo (00-3F _H)	0B _H	CRC H L
--	-----------------	------------

Respuesta del esclavo

Nº Esclavo (00-3F _H)	0B _H	00 00	Valor Contador H L	CRC H L
--	-----------------	-------	--------------------------	------------

Figura 2.13: Contenido de contador 9

Función 15:

Escritura de bits del autómeta. La trama es la indicada en la Figura 2.14. La forma de direccionamiento es análoga a la indicada para las funciones 1 y 2.

Petición maestro

Nº Esclavo (00-3F _H)	0F _H	Dirección 1º Bit PP PB	Nº de Bits NN NN	Nº de Octetos M	Valor de los bits 8xM valores	CRC H L
----------------------------------	-----------------	------------------------	------------------	-----------------	-------------------------------	---------

Respuesta del esclavo

Nº Esclavo (00-3F _H)	0F _H	Dirección 1º Bit PP PB	Nº de Bits NN NN	CRC H L
----------------------------------	-----------------	------------------------	------------------	---------

Figura 2.14: Escritura de bits

Función 16:

Escritura de palabras del autómata. La trama es la indicada en la Figura 2.15.

Petición maestro

Nº Esclavo (00-3F _H)	10F _H	Dirección 1ª Palabra PP PP	Nº de Palabras NN NN	Nº de Octetos M	Valor de las palabras HL, HL, ...	CRC H L
----------------------------------	------------------	----------------------------	----------------------	-----------------	-----------------------------------	---------

Respuesta del esclavo

Nº Esclavo (00-3F _H)	10 _H	Dirección 1ª Palabra PP PP	Nº de Palabras NN NN	CRC H L
----------------------------------	-----------------	----------------------------	----------------------	---------

Figura 2.15: Escritura de palabras

Mensajes de error:

Puede ocurrir que un mensaje se interrumpa antes de terminar. Cada esclavo interpreta que el mensaje ha terminado si transcurre un tiempo de silencio equivalente a 3,5 caracteres. Después de este tiempo el esclavo considera que el carácter siguiente es el campo de dirección de esclavo de un nuevo mensaje.

Cuando un esclavo recibe una trama incompleta o errónea desde el punto de vista lógico, envía un mensaje de error como respuesta, excepto en el caso de mensajes de difusión. La trama del mensaje de error es la indicada en la Figura 2.16.

Respuesta del esclavo

Nº Esclavo (00-3F _H)	Código Función	Código Error	CRC H L
----------------------------------	----------------	--------------	---------

Figura 2.16: Trama de mensaje de error

Si la estación maestra no recibe respuesta de un esclavo durante un tiempo superior a un límite establecido, declara el esclavo fuera de servicio, a pesar de que al cabo de un cierto número de ciclos hace nuevos intentos de conexión.

3 Desarrollo

3.1 Diagrama de bloques de solución

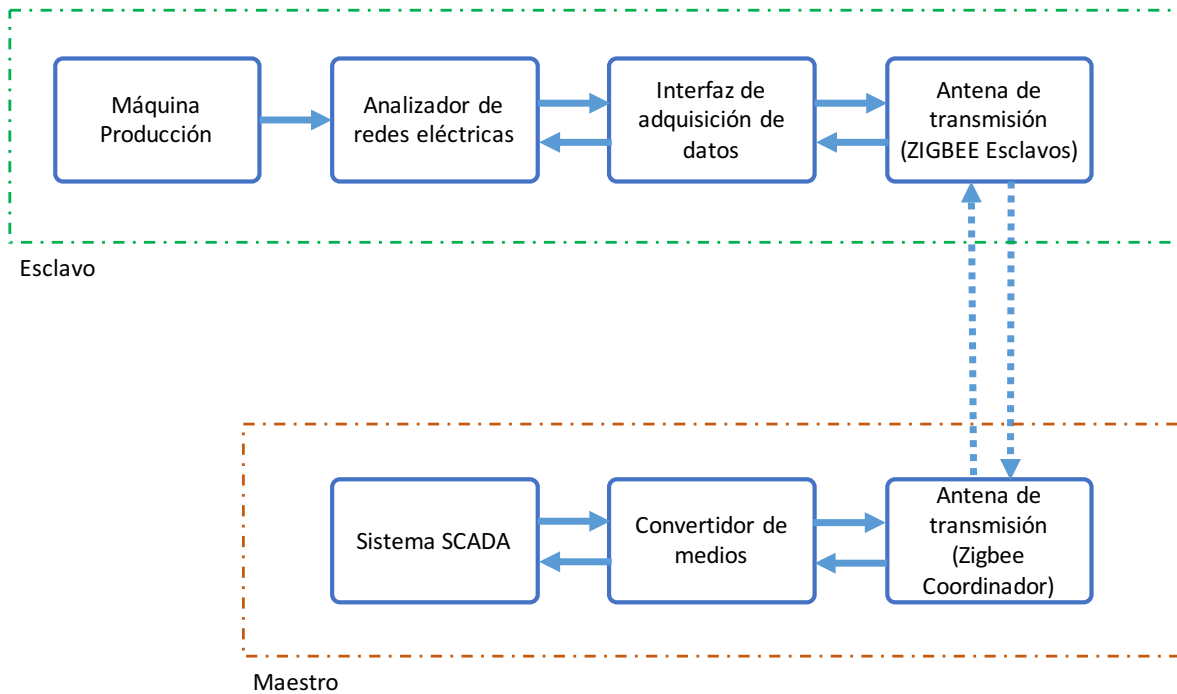


Figura 3.1: Diagrama de bloques de la solución

En la Figura 3.1 se muestra el diagrama de bloques de la solución final, los primeros cuatro bloques de la parte superior corresponden al elemento “esclavo” el cual es repetible para cada elemento del cual se requiere obtener la información de consumo energético por máquina; en la parte inferior encontramos 3 bloques los cuales corresponden al dispositivo “maestro” que obtiene la información y la almacena en un sistema SCADA (Supervisory Control And Data Acquisition).

El primer bloque denominado Máquina Producción corresponde a cualquier elemento de la fábrica al cual se requiera analizar su consumo energético, pueden ser motores, elementos térmicos, sistemas de ventilación y/o refrigeración, luminaria, etcétera.

Analizador de redes eléctricas, es el elemento encargado de medir las variables de voltaje, corriente, factor de potencia, potencia reactiva, frecuencia el cual cuenta con una salida de datos en formato Modbus RTU.

La *Interfaz de adquisición de datos* se comprende de un micro controlador y convertidores de medios que tienen la finalidad de obtener información del analizador de redes eléctricas mediante una red Modbus RTU y pasar dicha información a un formato de comunicación serial.

El último bloque de la sección esclavo corresponde a la *antena de transmisión de datos (ZIGBEE esclavo)* necesaria para hacer llegar la información de la interfaz de adquisición al dispositivo coordinador.

Para el caso de los bloques correspondientes al maestro, tenemos la *antena de transmisión de datos (ZIGBEE maestro)* que tiene la función de recibir información proveniente de los esclavos , así como hacer llegar comandos de control a los esclavos.

El *convertidor de medios* es un dispositivo que tiene como función el convertir la información serial proveniente de la antena ZIGBEE a USB para que pueda ser interpretado por el sistema SCADA.

Como elemento final del bloque maestro se encuentra el *sistema SCADA* que corresponde a una PC que mediante un software adquiere y almacena los datos de las máquinas de producción con la posibilidad de generar reportes históricos que ayuden a la toma de decisiones de la fábrica con respecto a temas energéticos.

3.2 Selección de tecnologías.

Topología de estrella

Unas cuantas razones de elección de esta topología son el bajo coste de implementación (por requerir poca electrónica en su desarrollo) y no ser necesario disponer de una comunicación entre dispositivos finales. El uso de cualquiera de las otras topologías complicaría el desarrollo del proyecto innecesariamente, además de encarecerlo considerablemente para después no llegar a aprovechar todas sus funciones.

ZIGBEE

La tecnología inalámbrica escogida es ZigBee porque hace posible que dispositivos electrónicos de bajo consumo puedan realizar comunicaciones inalámbricas. Es especialmente útil para redes de sensores en entornos industriales, médicos y sobre todo domóticos. Las comunicaciones con ZigBee se realizan en la banda libre de 2.4GHz. A diferencia de bluetooth, este protocolo no utiliza FHSS (Frequency Hooping Spread Spectrum), sino que realiza las comunicaciones a través de una única frecuencia, es decir, de un canal. Normalmente puede escogerse un canal de entre 16 posibles.

El alcance depende de la potencia de transmisión del dispositivo, así como también del tipo de antena utilizada (cerámica, dipolo, etc.). El alcance normal con antena dipolo, en línea vista, es de aproximadamente (tomando como ejemplo el caso de MaxStream, en la versión de 1mW de potencia) 100m y en interiores de unos 30m. La velocidad de transmisión de datos de una red ZigBee es de hasta 256kBps. Una red ZigBee (teóricamente) puede estar formada por un máximo de 65535 equipos, es decir, el protocolo está preparado para poder controlar en la misma red 65535 dispositivos.

Las características que hacen decantarse por el uso de ZigBee son:

- Bajo coste económico.
- Menor consumo energético comparado con Wi-Fi y Bluetooth.
- Uso de bandas de radio libres y sin necesidad de licencias.
- Instalación barata y simple (No es necesaria mucha electrónica para implementarlo).
- 65535 posibles nodos.

PIC18F2550

En la realización del proyecto se utiliza el PIC18F2550 porque el microcontrolador reúne las características necesarias actuales y podría ajustarse sin problemas a características futuras.

Una de las características más importantes de este microcontrolador es que dispone de conexión USB y posibilidad de establecer comunicaciones serie. Además de disponer de 13 entradas de conversión analógicas lo cual es ideal en caso de necesitar controlar gran cantidad de variables físicas.

Las características que hacen decantarse por el PIC18F2550 son las siguientes:

- Cumple con el estándar USB V2.0.
- 32kb de memoria de programa Flash.
- Baja velocidad (1,5 Mb/s) y alta velocidad (12 Mb/s).
- Soporta transferencias de Control, Asíncronas, Interrupción y Bulk.
- Soporta un máximo de 32 Endpoints (16 Bidireccionales).
- 1 kByte de RAM con Acceso dual para USB.
- Transmisor USB en el propio chip con Regulador de Voltaje.
- Modo Idle y Sleep con 5,8 μ A y 0,1 μ A de consumo respectivamente.

Además de disponer de una muy buena conectividad con el exterior:

- Alta corriente de salida: 25 mA.
- 24 pines entradas / salidas.
- 13 entradas convertidoras analógico / digital.
- Convertidor analógico digital de 10 bits.

Se considera que es un microprocesador potente tanto para el desarrollo actual como para posteriores ampliaciones del proyecto.

SCADA

Para el desarrollo del proyecto era necesario crear un software para el PC que nos permita visualizar el estado y las medidas adquiridas por los dispositivos finales. El usuario a través de este software puede visualizar numérica y gráficamente todos los datos enviados por el analizador de redes eléctricas tales como voltaje, corriente, factor de potencia, frecuencia, potencia.

El software SCADA necesita operar a través del puerto USB, la finalidad es poder enviar / recibir datos del coordinador. El coordinador es el encargado de gestionar la comunicación con los dispositivos finales. En este proyecto únicamente es necesario un software capaz de gestionar los datos de entrada para su posterior visualización de los datos obtenidos por medio de USB.

El software SCADA además nos permite tener conectividad en una red local, esto es posible porque se ha interconectado el SCADA con una base de datos mySQL. El SCADA es el encargado de abrir la base de datos mySQL e ir la rellenando conforme se van procesando los datos recibidos por el coordinador. De esta forma podemos consultar los datos del proceso en cualquier punto de la fábrica.

3.3 Diseño del coordinador

El Hardware coordinador, como se ha explicado anteriormente, es el encargado de gestionar el flujo de datos entre los dispositivos finales y el PC. Para la realización del Hardware coordinador se han tenido que tener en cuenta los siguientes aspectos.

3.3.1 Alimentación del coordinador

La alimentación del hardware coordinador procede directamente del bus USB del PC. El microprocesador PIC18F2550 tiene un rango de tensión de alimentación de 2 a 5,5V, por lo cual la alimentación que ofrece el puerto USB es ideal debido a que no tiene que ser tratada para poder alimentarlo.

Para ver fácilmente si la placa recibe tensión del puerto USB del PC se ha implementado un led, el cual permitirá ver que la placa esta recibiendo alimentación.

En la Figura 3.2 se puede observar el circuito implementado.

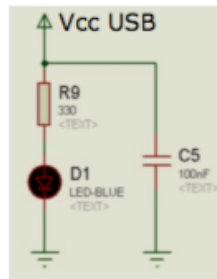


Figura 3.2: Indicador LED de alimentación

Los cálculos de la resistencia R9 se observan en la Ecuación 1.

$$R9 = \frac{V_{CC}}{I_{LED}} = \frac{5V}{15 mA} = 333,33 \Omega = 330\Omega$$

Ecuación 1: Cálculo de R9

Como se puede ver en la Figura 3.2 se ha dispuesto un condensador C5 de 100nF conectado desde la Vcc hasta Gnd. Este condensador tiene la función de proteger al microcontrolador de posibles ruidos e interferencias electromagnéticas que podrían ocasionar el mal funcionamiento.

3.3.2 Alimentación del XBEE

La alimentación del modulo XBee tiene un rango de 2,8V a 3,4V. Se debe tener en cuenta que los módulos XBee son muy sensibles a la fluctuación de la tensión de alimentación, es por ello que debe utilizarse un estabilizador de tensión para asegurarse de no superar los 3,4V bajo ningún concepto. De lo contrario el modulo XBee quedaría completamente inservible. En el

diseño se ha utilizado el estabilizador de tensión MCP1700-3302E/TO para poder ajustar la tensión de alimentación deseada de acuerdo a la Figura 3.3 .

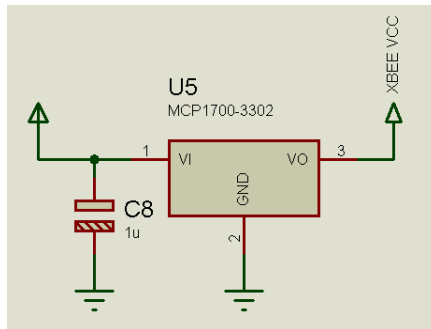


Figura 3.3: Alimentación XBEE

3.3.3 Montaje de microcontrolador

El circuito de reset ha sido implementado para no tener que desconectar el bus USB en caso de sufrir algún malfuncionamiento por parte de coordinador.

El circuito implementado se puede observar en la Figura 3.4.

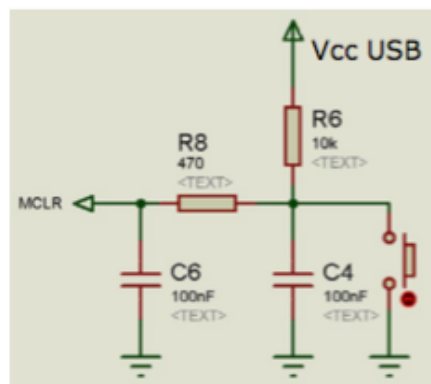


Figura 3.4: Circuito de Reset (MCLR)

La función del circuito es ofrecer 5V continuos al pin 1 del microcontrolador. En el momento en que se pulsa el botón de reset en el pin 1 aparecerán 0V y el microcontrolador realizara el reset de su CPU. La función del condensador C6 es evitar algún posible reset accidental por interferencias en el ambiente y el condensador C4 evita rebotes por el accionamiento mecánico del pulsador.

Otro montaje básico es el circuito oscilador, es un circuito muy simple pero de vital importancia para el buen funcionamiento del microcontrolador. Dependiendo del cristal de cuarzo elegido se le indicara al microcontrolador la velocidad a la que deberá trabajar.

El circuito utilizado se puede observar en la Figura 3.5:

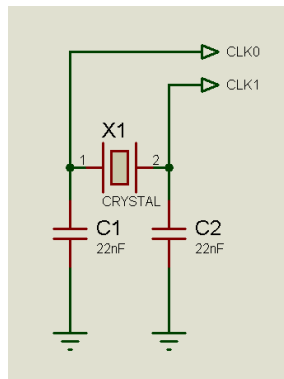


Figura 3.5: Configuración cristal de cuarzo

Una de las grandes virtudes del diseño realizado es que no es necesario desconectar ni realizar ningún reset después de la programación del coordinador. El diseño se ha realizado de forma que se pueda reprogramar el código del microprocesador sin tener que desmontar el hardware coordinador del proceso.

En la Figura 3.6 se puede observar el circuito que permite realizar la programación del microcontrolador.

El circuito programador esta formado por una resistencia de 100Ω y un zócalo donde va conectado el programador. La función de la resistencia R7 es proteger al programador de un exceso de corriente proveniente del circuito que pudiera causar daños en el programador.

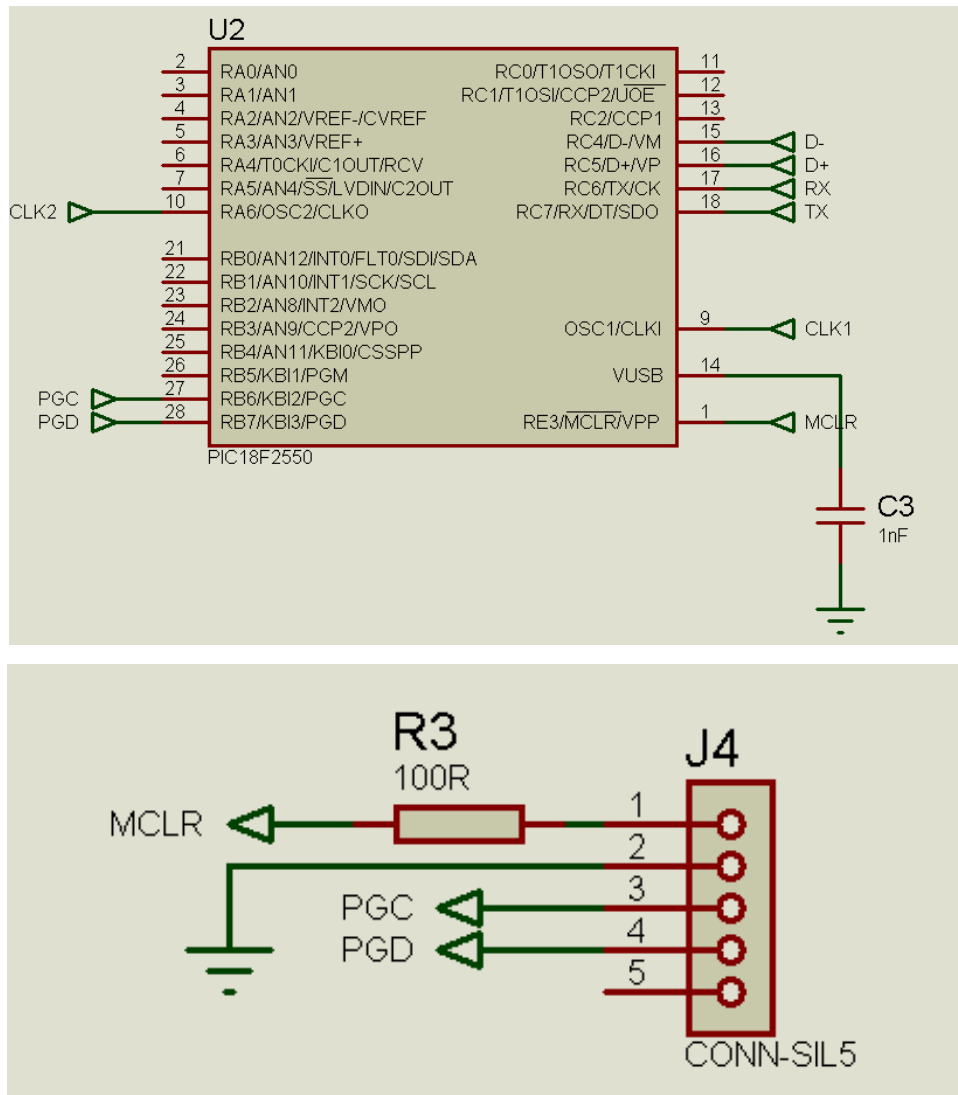


Figura 3.6: Circuito de programación de microcontrolador

3.3.4 Comunicación entre Microcontrolador y XBEE

Para realizar la conexión serie se debe tener en cuenta que el microcontrolador esta alimentado a 5V y el modulo XBee a 3,3V. En el caso del microcontrolador no debe preocupar porque su rango de alimentación es de 2 a 5V, por lo cual los datos llegaran con una tensión de 3,3V que están dentro del rango y los detectara sin problemas. En el caso del XBee se debe tener cuidado con la entrada Rx, debido a que se le estarán inyectando los datos a 5V cuando la tensión de trabajo es de 3,3V, eso quiere decir que se deberá montar un circuito atenuador para adaptar los niveles de tensión y evitar futuros posibles daños al XBee. El chip seleccionado es el 74HC125,

no se decidió utilizar un divisor de voltaje ya que en la práctica se tuvo pérdida de información en algunos casos.

El circuito usado se puede visualizar en la Figura 3.7.

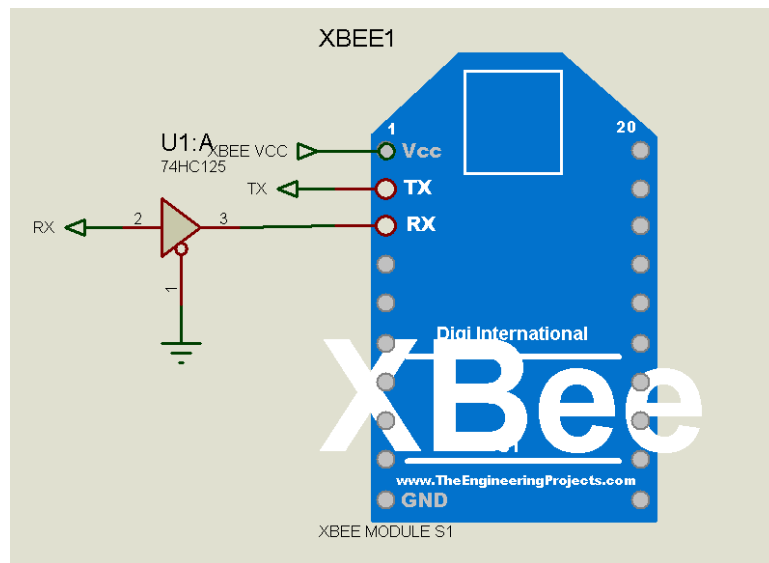


Figura 3.7: Adaptador de voltaje para XBEE

3.3.5 Conexión USB

Para realizar la conexión USB no es necesario implementar ningún circuito de adaptación de señales, el microcontrolador está preparado para poder realizar la conexión de forma directa en sus pines 15 y 16, por tal motivo se conecta directamente al puerto USB tal como se muestra en la Figura 3.8.

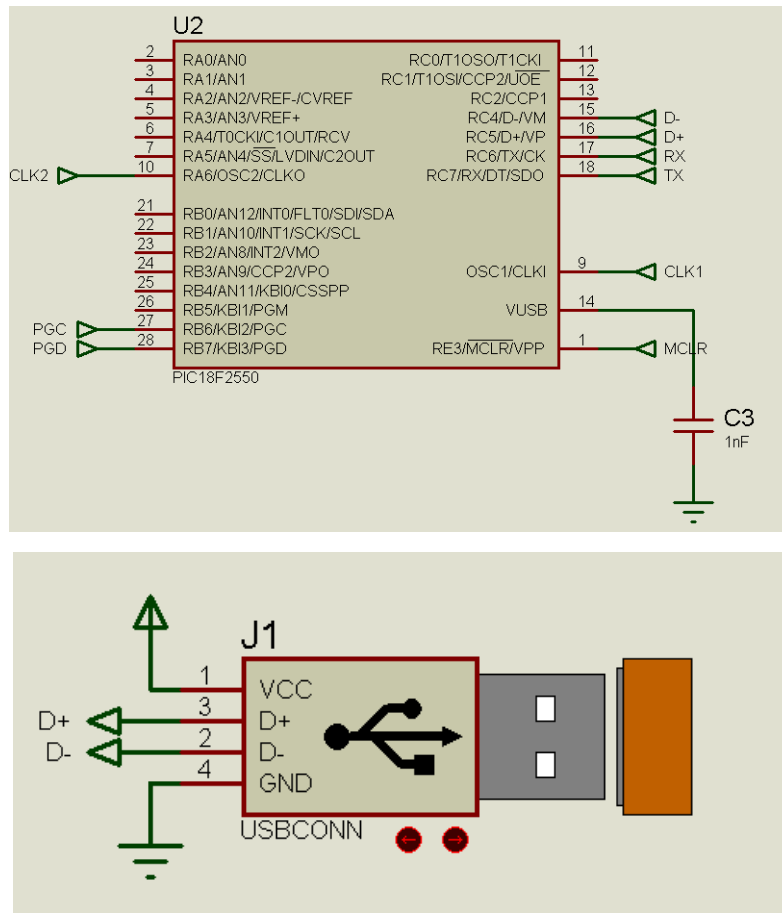


Figura 3.8: Conexión USB

3.4 Diseño de los dispositivos finales

Los dispositivos finales, son los encargados de adquirir los datos de los analizadores de energía eléctrica y transmitirlos al coordinador. Para la realización de los dispositivos finales se han tenido en cuenta los siguientes aspectos.

3.4.1 Alimentación dispositivo final

La alimentación de los dispositivos finales es diferente a la alimentación del hardware coordinador. En el caso del hardware coordinador, la alimentación procede directamente del bus USB y no se tienen limitaciones de energía, siempre y cuando no se superen los 500mA que puede abastecer como máximo el bus USB.

En el caso de los dispositivos finales se requiere tener una alimentación externa la cual debe regularse para no tener variaciones de energía. Para abastecerles de la energía necesaria para su funcionamiento se utilizan una fuente conmutada de voltaje de 110V AC a 12 V DC.

3.4.2 Alimentación microcontrolador

Para adaptar la tensión de la fuente conmutada a la tensión de 5V se utiliza un regulador de tensión LM7805, adicional al circuito se agrega un capacitor electrolítico C7 a la entrada a manera de filtro de interferencias provenientes de la entrada, en la Figura 3.9 se puede ver el circuito de alimentación principal.

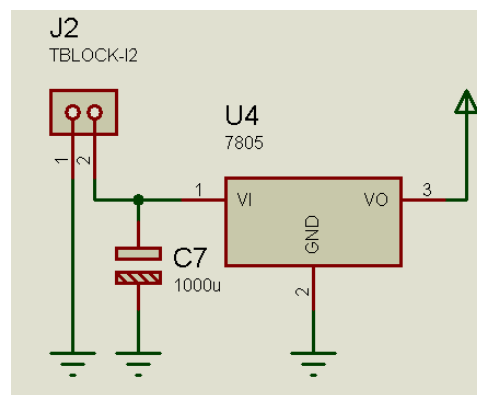


Figura 3.9: Circuito regulador de voltaje a la entrada

3.4.3 Alimentación XBee

El circuito de alimentación del módulo XBee es el mismo que se ha utilizado en el hardware coordinador. Se pueden consultar las características en el apartado 3.3.2 Alimentación del XBEE.

3.4.4 Montaje básico del microcontrolador

El montaje básico del microcontrolador esta formado por el circuito de reset, circuito oscilador y circuito programador. Estos circuitos son los mismos que se han desarrollado para el hardware coordinador, si se desea consultar dichos circuitos pueden verse en el apartado 3.3.3 Montaje de microcontrolador de este documento.

3.4.5 Conexión entre el microcontrolador y el XBee

El circuito de conexión entre el microcontrolador y el XBee es el mismo que se ha utilizado en el hardware coordinador. Se pueden consultar las características en el apartado 3.3.4 Comunicación entre Microcontrolador y XBEE de este documento.

3.4.6 Conexión entre microcontrolador y MAX 485

El analizador de redes eléctricas tiene la capacidad de transmitir sus datos mediante una red de tipo MODBUS, la cual puede ser implementada en el microcontrolador por medio de software, pero para el enlace físico se requiere un transductor de medios ya que la comunicación de este tipo de redes se trabajo por medio del estándar RS-485.

El circuito integrado seleccionado para realizar la conversión de medios es el MAX485 ya que no requiere componentes externos para poder trabajar.

En la Figura 3.10 se muestra el circuito final de conexión, en los canales A y B se aprecian las resistencias R4 y R5 que corresponden a resistencias de Pull Up y Pull Down, esto debido a que en ausencia de transmisión de datos, los transmisores están en alta impedancia.

Las resistencias de polarización hacen que los receptores “vean” una diferencia de potencial entre los terminales A y B. [8]

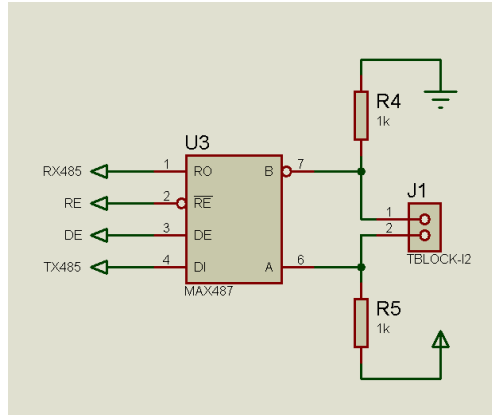


Figura 3.10: Conexión de MAX485

Los diagramas finales completos se pueden ver en los anexos en los apartados:

- Diagrama electrónico de coordinador
- Diagrama electrónico de dispositivo final

3.5 Analizador de redes eléctricas

La obtención de los datos energéticos se da por medio de un analizador de redes eléctricas el cual cuenta con el protocolo de comunicación MODBUS para transmitir su información.

El dispositivo cuenta con montaje en panel tipo DIN lo que hace que su instalación en los tableros de control sea muy fácil. Tiene la capacidad de obtener todos los siguientes datos en conexiones trifásicas: corrientes, voltajes, potencia activa, potencia reactiva, potencia aparente, frecuencia y factor de potencia. El dispositivo corresponde a la Figura 3.11.



Figura 3.11: Analizador de redes eléctricas

Su principal ventaja es que para la medición de corriente puede ser de hasta 100A y no requiere de elementos externos tal cual como se muestra en la Figura 3.12.

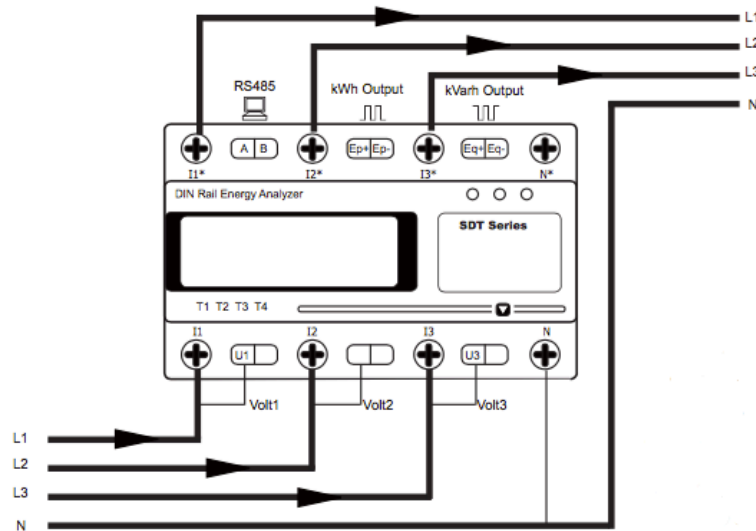


Figura 3.12: Diagrama de conexión analizador de redes

Cuenta con un display de 7 dígitos para poder observar la información en sitio así como con interfaz RS-485 para comunicación con dispositivos externo.

3.6 Herramientas de programación

3.6.1 Visual Studio 2012

Visual C# 2012 es el software elegido para realizar el SCADA, forma parte de Microsoft Visual Studio que es un entorno de desarrollo integrado (IDE por su acrónimo inglés) para sistemas operativos Windows. Soporta diversos lenguajes de programación tales como Visual Basic, Visual C++, Visual C#, Visual F#, etc.

Visual Studio permite a los desarrolladores crear tanto aplicaciones, como lugares web o cualquier servicio web que soporte la plataforma .net. Así mismo se pueden llegar a crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web o dispositivos móviles. El entorno IDE de Visual Studio consiste en un editor de código (programa donde se escribe el código fuente), un depurador (programa que corrige errores del código fuente para que pueda ser correctamente compilado), un compilador (programa que traduce el código fuente a lenguaje máquina), y un constructor de interface gráfica o GUI, que es una forma de programar donde no es necesario escribir el código para desarrollar la parte gráfica del programa, sino que se puede realizar de forma visual.

Las ventajas que proporciona son diversas ya que permite un rápido aprendizaje, integra el diseño y la implementación de formularios de Windows en una sola interface y permite utilizar con facilidad la plataforma Windows, ya que tiene acceso prácticamente total a las librerías del sistema operativo. Existe una gran cantidad de recursos y ayuda en la web que hace posible tener soporte en caso de ser necesario.

En la Figura 3.13 se puede observar la ventana de trabajo de Visual C# 2012.

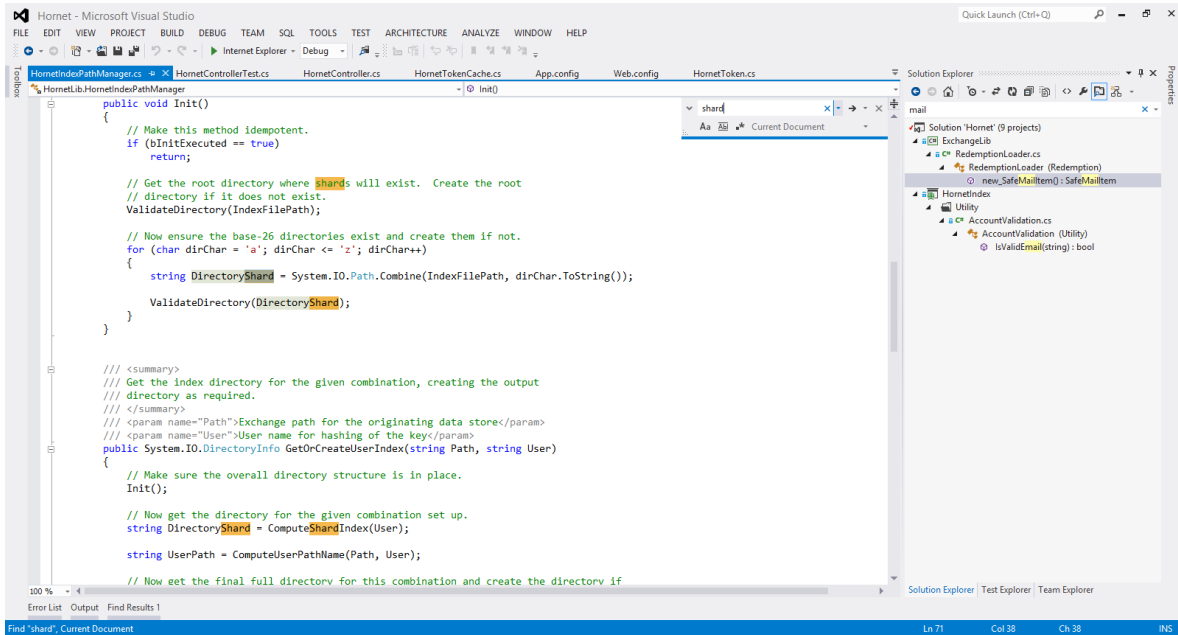


Figura 3.13: Entorno Visual Studio 2012

3.6.2 PIC C Compiler

Para la realización y compilación del programa del microcontrolador (.hex) se ha utilizado el programa PIC C Compiler. Este software permite programar el código del microcontrolador mediante el uso de lenguaje C facilitando la programación enormemente si comparamos lo costoso que sería usar código ASM (Ensamblador por sus siglas en inglés). Adicionalmente cuenta con una gran cantidad de ejemplos y drivers listos para implementar y hacer más rápido la solución.

En la Figura 3.14 podemos observar la ventana de trabajo de PIC C Compiler.

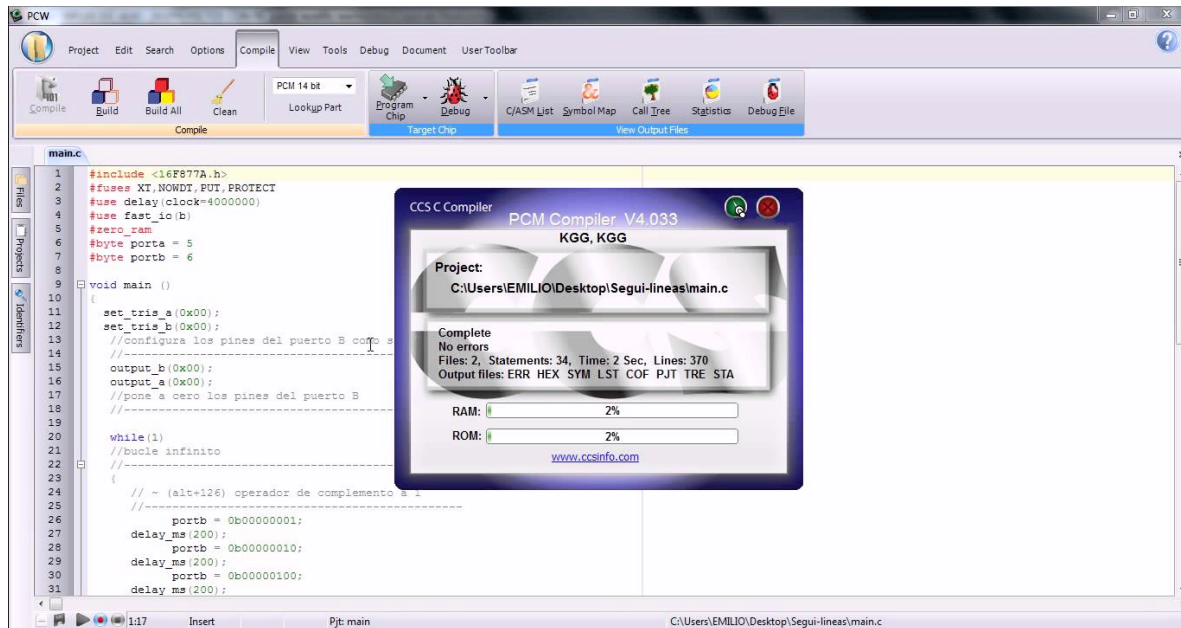


Figura 3.14: Entorno de trabajo PIC C

3.6.3 X-CTU

X-CTU es el programa que proporciona Digi para la configuración de los módulos XBee. Es un programa capaz de configurar cualquier modulo XBee (no importa la Serie) con los parámetros de funcionamiento deseados dentro de las posibilidades de configuración del modulo XBee.

Las características que ofrece X-CTU son las siguientes:

- Soporte para todos los productos MaxStream.
- Ventana de terminal integrada.
- Capacidad de descarga y actualización del Firmware del modulo.
- Visualización caracteres ASCII o hexadecimal en la ventana del terminal.
- Composición de paquetes de prueba de envío en ASCII o hexadecimal.
- Guardar y recuperar configuraciones de módulos de uso común.
- Detección automática del modulo conectado.
- Restauración de parámetros del modulo a estado de fabrica.

Además de la utilización del programa X-CTU es necesario disponer de un hardware especial para conectarse con el modulo XBee. El hardware utilizado es el “XBee Explorer USB”. Es

un hardware adaptador para poder conectar los módulos XBee al puerto USB del PC. Este hardware funciona con todos los módulos XBee incluida la Serie 1, la 2.5 tanto las versiones estándar como la Pro. En la Figura 3.15 se puede observar el hardware XBee Explorer USB.

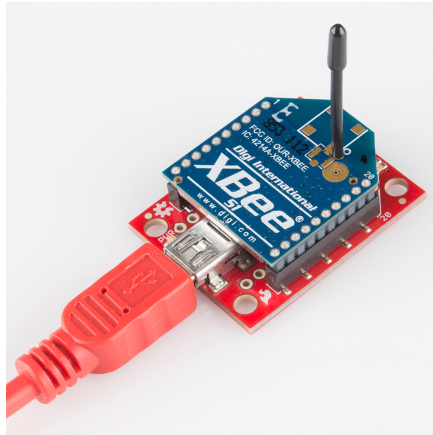


Figura 3.15: XBee Explorer

3.7 Configuración de dispositivo final

Los dispositivos finales únicamente pueden comunicarse con el Hardware coordinador a través de los módulos inalámbricos.

En la Figura 3.16 podemos observar el funcionamiento del dispositivo final.

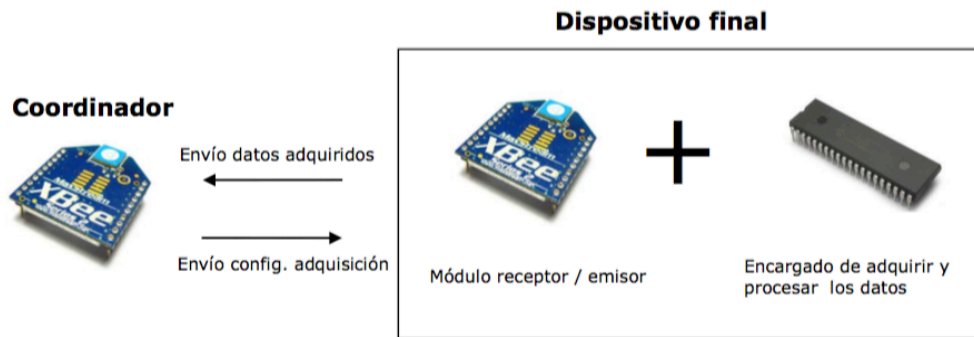


Figura 3.16: Esquema de comunicación Esclavo

Para conseguir la comunicación al microcontrolador se le debe indicar que use la comunicación serie, se indica que la velocidad usada en la comunicación es de 9600 baudios, los pines usados para el envío y recepción de datos entre el microprocesador y el modulo XBee, y la cantidad de datos que debe usar en la transmisión (8 bits).

La función principal de los dispositivos finales es enviar al hardware coordinador los datos adquiridos por el analizador de redes eléctricas con las variables de voltaje, corriente, factor de potencia, frecuencia, potencia.

3.7.1 Adquisición de datos

La información que genera el analizador de redes eléctricas viaja por el protocolo MODBUS el cual ha sido implementado en el código del microcontrolador haciendo uso de la librería modbus.c proporcionada por la asociación creadora del protocolo www.modbus.org la cual se encuentra al final de este documento en los anexos.

Haciendo uso de la instrucción 03H del protocolo Read Holding Registers se logra obtener todos los parámetros del analizador de redes:

```
int8 nodo=0x01; //Slave Adress
int16 start_address=0x00; //Start Address to read data
int16 quantity=0x4C; //Data quantity to read
```

```
modbus_read_holding_registers(nodo,start_address,quantity);
```

Esta información es almacenada en un buffer para posteriormente ser transmitida mediante el Xbee:

```
fprintf(XBEE,"%X", cadena[j]);
```

El código completo se encuentra en el apartado de anexos.

3.8 Configuración del módulo XBee

Como ya se ha mencionado anteriormente el programa utilizado para configurar los XBee es el X-CTU.

Con la configuración que se muestra a continuación se consigue configurar los dispositivos con la topología punto - multipunto, es decir, el coordinador puede comunicarse con todos los dispositivos finales que estén dentro de su red, pero los dispositivos finales solo pueden comunicarse con el coordinador.

Para programar los módulos XBee solamente es necesario utilizar la pestaña de Radio Configuration.

El primer paso es seleccionar la configuración de la comunicación del PC con el modulo XBee. Para que puedan entenderse sin problemas se le debe indicar la velocidad de comunicación (Baud), el control de flujo usado (Flow Control), la cantidad de datos a enviar (Data bits), si usamos paridad (Parity) y cuantos bits conforman el indicador de fin de comunicación (Stop Bits).

En la Figura 3.17 se puede ver la pestaña con la configuración necesaria para comunicar con los módulos XBee.

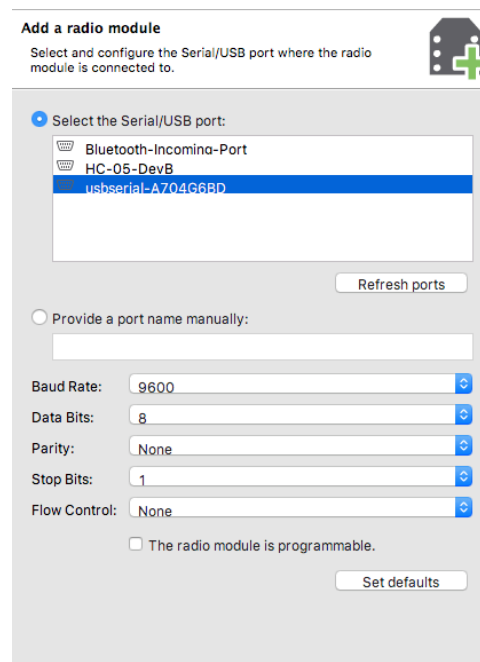


Figura 3.17: Configuración de comunicación con Módulo Xbee

3.8.1 Configuración del coordinador

En la siguiente imagen se puede ver la pestaña *Radio Configuration* con la configuración necesaria para el correcto funcionamiento del módulo XBee ubicado en el coordinador.

Radio Configuration [- 0013A20040EA52D0]

Read Write Default Update Profile

Product family: XBP24 Function set: XBEE P...2.15.4 Firmware version: 10ee

▼ Networking & Security
Modify networking settings

i	CH Channel	C					
i	ID PAN ID	1986					
i	DH Destination Address High	0					
i	DL Destination Address Low	FFFF					
i	MY 16-bit Source Address	0					
i	SH Serial Number High	13A200					
i	SL Serial Number Low	40EA52D0					
i	MM MAC Mode	802.15.4 + MaxStream header w/					
i	RR XBee Retries	0					
i	RN Random Delay Slots	0					
i	NT Node Discover Time	19	x 100 ms				
i	NO Node Discover Options	0					
i	CE Coordinator Enable	Coordinator f11					
i	SC Scan Channels	1FFE	Bitfield				
i	SD Scan Duration	4	exponent				
i	A1 End Device Association	000b f01					
i	A2 Coordinator Association	000b f01					
i	AI Association Indication	0					
i	EE AES Encryption Enable	Disable f01					
i	KY AES Encryption Key						
i	NI Node Identifier						

Figura 3.18: Ventana de configuración de red X-CTU

En el apartado *Networking & Security* es donde se indica cual es el funcionamiento del modulo XBee deseado y la configuración del mismo como por ejemplo el canal de comunicación, el PAN ID, etc.

En la Figura 3.18 se puede ver la configuración necesaria para el correcto funcionamiento, a modo de resumen se muestra la Tabla 3.1 con la configuración básica para el correcto funcionamiento del coordinador.

Tabla 3.1: Configuración básica de *Networking & Security*

Channel	C
PAN ID	1986
Destination Address High	0
Destination Address Low	FFFF
Serial Number High	13A200
Serial Number Low	40EA52D0

Channel C es el canal que se ha asignado al modulo XBee para realizar la comunicación, al indicarle al modulo que trabaje en el Channel C se esta trabajando en el rango de frecuencia de: 2,4075 – 2,4125GHz.

El parámetro PAN ID es el identificador de la red, en este caso se ha indicado que usen el identificador 1986.

Los parámetros de Destination Address High y Destination Address Low están configurados con el valor cero y FFFF respectivamente, esto le indica al modulo XBee del coordinador que la información se envía a todos los módulos de los dispositivos finales que estén en el mismo Channel y que contengan la misma PAN ID.

Los parámetros Serial Number High y Serial Number Low son los datos identificativos del módulo XBee colocados en el coordinador (No son modificables por el usuario). Si se quisiera enviar única y exclusivamente datos a un dispositivo final, se debería cambiar los ceros de la

Destination Address High y Destination Address Low, por los datos identificativos del modulo XBee al que se quiera enviar la información. Esto se vera en el siguiente apartado en el cual se trata el envío de los datos de los dispositivos finales al hardware coordinador.

3.8.2 Configuración del dispositivo final

Los pasos preliminares para la configuración del XBee del dispositivo final son los mismos que los tratados en el apartado 3.8.1 Configuración del coordinador.

En la Tabla 3.2 se puede ver el resumen de la configuración necesaria para el correcto funcionamiento del módulo XBee ubicado en el dispositivo final.

Tabla 3.2: Configuración Networking & Security dispositivo final

Channel	C
PAN ID	1986
Destination Address High	13A200
Destination Address Low	40EA52D0
Serial Number High	13A200
Serial Number Low	40715D36

Channel C y el parámetro PAN ID son los mismos que se colocaron el dispositivo coordinador como se mostró en el apartado anterior 3.8.1 Configuración del coordinador.

Los parámetros de Destination Address High y Destination Address Low están configurados con los valores Serial Number High y Serial Number Low del coordinador, de esta forma el dispositivo final únicamente enviara la información al modulo del coordinador.

Los parámetros Serial Number High y Serial Number Low son los datos identificativos del módulo XBee colocados en el dispositivo final (No son modificables por el usuario).

4 Resultados

Al momento de a redacción de este documento se tienen los siguientes resultados:

El primer procedimiento de prueba consiste en colocar el dispositivo maestro a 5 metros del dispositivo esclavo, dicha prueba se realizó por 18 horas sin interrupción, en las cuales el dispositivo maestro recibió las tramas de comunicación de manera correcta en un 97.5% de las lecturas realizadas, el procedimiento para corroborar una lectura correcta es mediante la medición de la cadena recibida y la verificación de no encontrarse trunca, en la Figura 4.1 y en la Figura 4.2 se muestra el software SCADA con las dos opciones de visualización que se cuenta donde en la primera imagen se aprecia la información de voltaje en barras tridimensionales y en la segunda imagen la cadena de texto recibida del dispositivo esclavo.

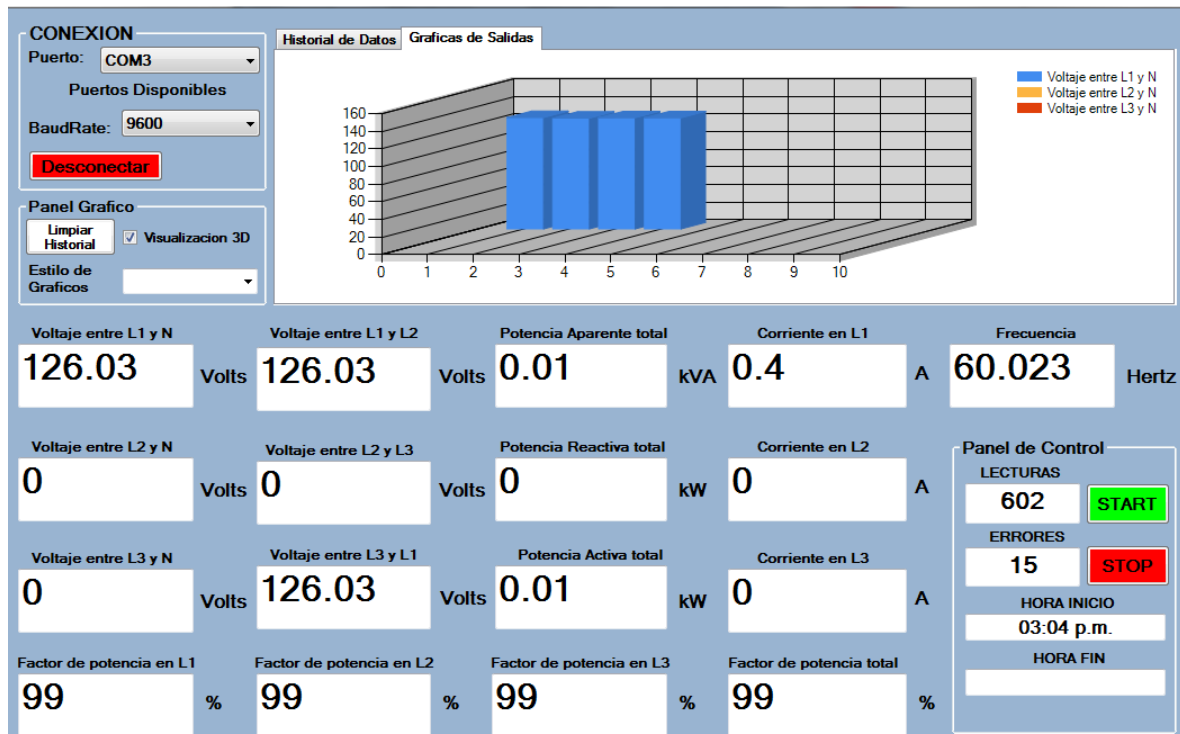


Figura 4.1: Interfaz SCADA con gráfica de barras

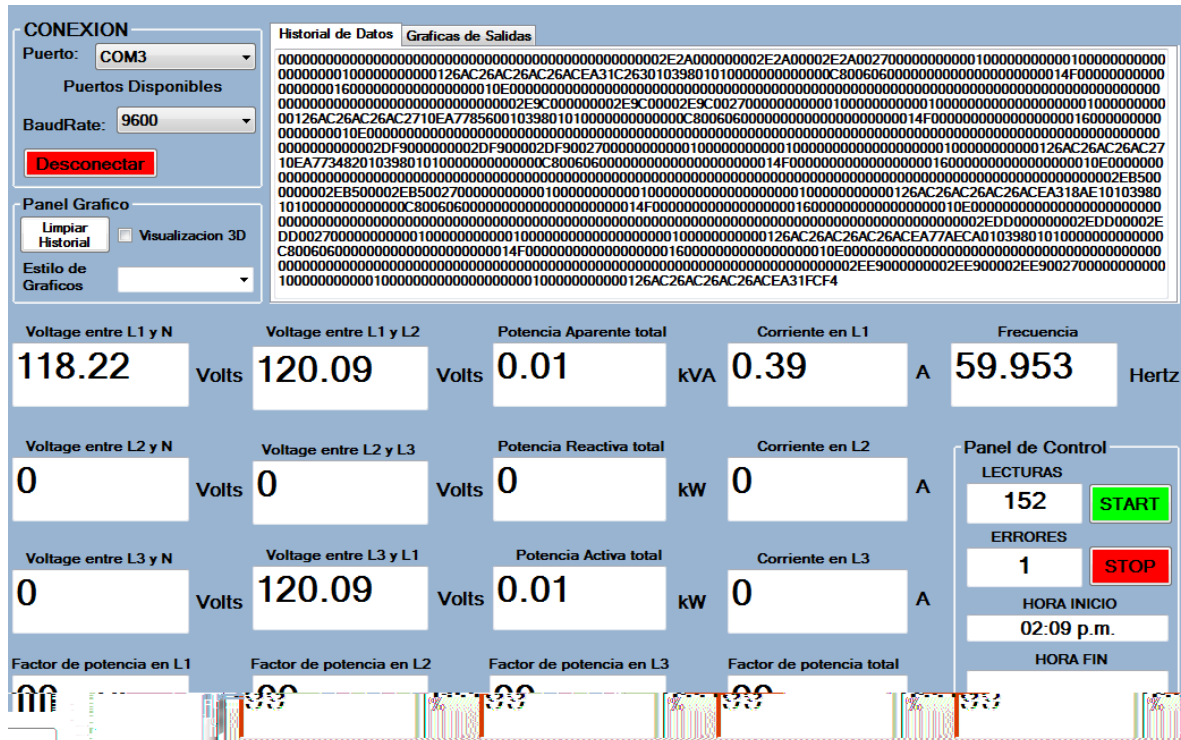


Figura 4.2: Interfaz SCADA con cadena de texto recibida

Como segundo objeto de prueba se determinó considerar dos cosas, distancia máxima de transmisión de datos así como la interferencia por obstáculos en la señal de radiofrecuencia entre los dispositivos maestro y esclavo. La prueba de distancia se realizó a 40 metros entre ambos dispositivos, esto debido a que corresponde a la distancia máxima que existe entre las máquinas de producción y de la ubicación del dispositivo maestro, en la Figura 4.3 se muestra un croquis de la ubicación de ambos dispositivos. Los recuadros en color azul corresponden a las máquinas a medir del departamento de producción.



Figura 4.3: Ubicación de dispositivos

En este caso se tuvo un porcentaje de repetitibilidad de recepción de datos de un 85.3%, esto debido a que se colocaron los dispositivos con diversos obstáculos como muros de concreto y cristal, esto pudo afectar la comunicación entre ellos y por ende los resultados finales. El rango de repetitividad deseado es de mínimo un 85% en los datos con un intervalo de medición de datos de 10 segundos

En la Figura 4.4 y la Figura 4.5 se muestra las imágenes de los datos obtenidos del software SCADA en sus dos variantes tal cual como se mostró en la primera prueba.

4. Resultados

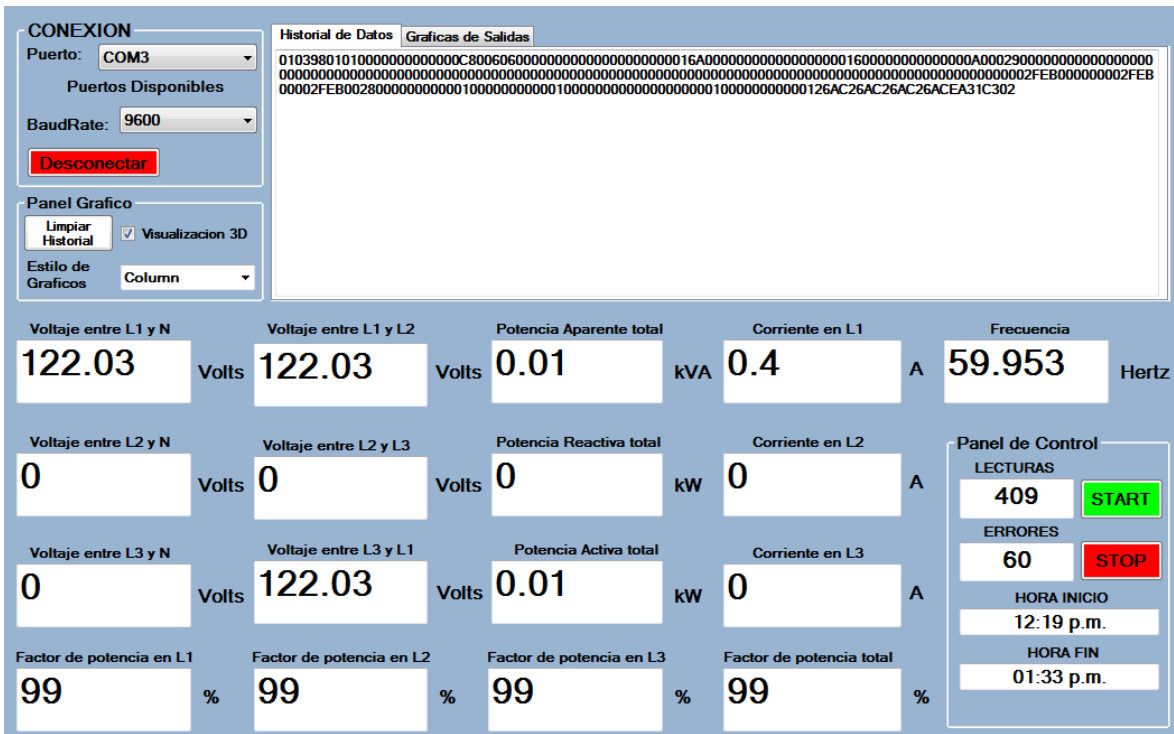


Figura 4.4: Resultados de prueba a 40m. Gráfico

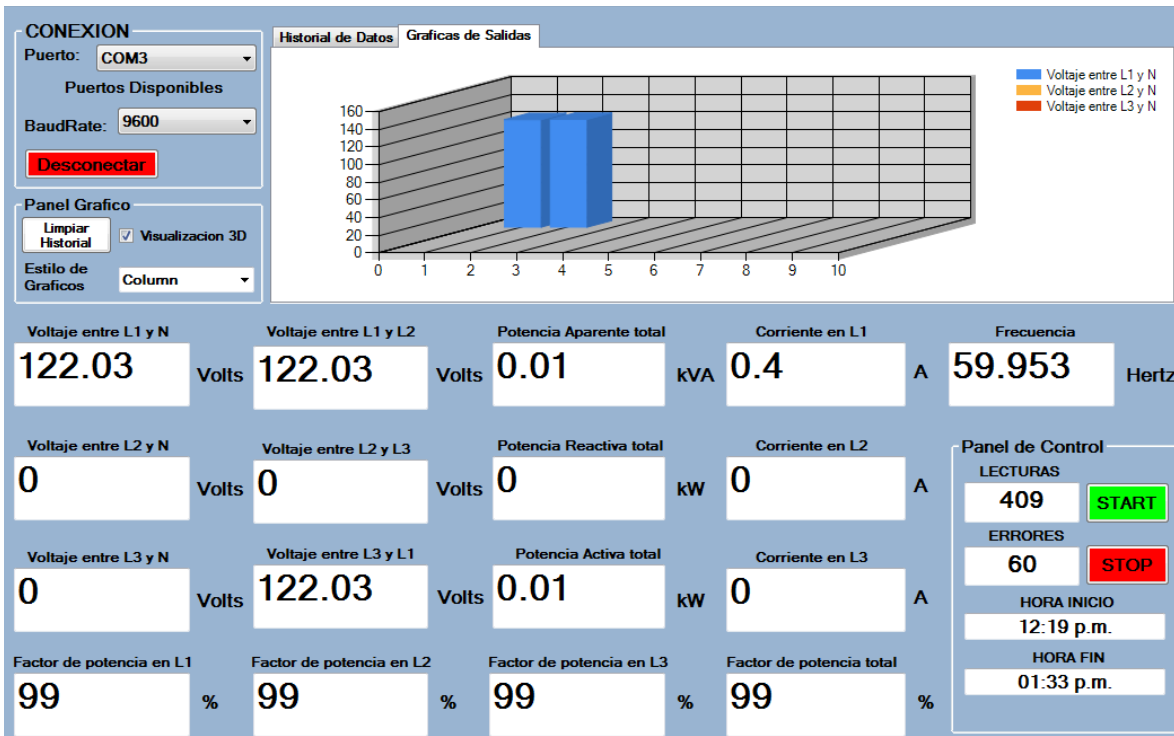


Figura 4.5: Resultados de prueba a 40m. Texto

5 Conclusiones

La medición de energía y su optimización es un proceso crítico para cualquier empresa que se dedique a la transformación de materias primas ya que en promedio el costo de energía eléctrica que impacta en el costo del producto oscila entre un 8 y 15%. Esto orilla a que sea completamente necesario tener monitoreada la energía que consume cada uno de los procesos de transformación de los materiales.

Como resultado del desarrollo se pudo obtener una interfaz de comunicación que ejecuta el protocolo Modbus en el modo de transmisión RTU utilizando un microcontrolador de propósito general con CPU de 8 bits transmitiendo la información por medio de una antena ZigBee. Este procesador soporta las funciones generales del protocolo Modbus y se conecta a una red con topología en bus según el estándar RS-485 y en topología estrella para el caso de la comunicación inalámbrica. El tipo de conexión a la red es Half-Duplex. Todo esto con la capacidad de transmitir inalámbricamente la información energética de máquinas industriales (Voltaje, corriente, potencia, factor de potencia, frecuencia).

Este trabajo puede tener aplicación en el estudio de redes industriales con Modbus como protocolo de comunicación, y permite implementar estas redes con propósitos académicos. Además de la ventaja de ser un sistema inalámbrico ayuda a tener un ahorro significativo en los tiempos de implementación así como en costes de instalación del proyecto.

Actualmente existen soluciones similares en el mercado que pueden ser muy costosas y que los tiempos de implementación son un poco más amplios respecto a la solución propuesta, esto gracias a que no existe actualmente una solución de hardware y software que pueda implementar la comunicación inalámbrica en un ambiente industrial teniendo seguridad en los datos que se transmiten por medio de esta.

Las interfaces de adquisición se encuentran listas para continuar desarrollando sobre ellas y tener aplicaciones futuras, como por ejemplo, poder alertar de manera remota cuando se activo un paro de emergencia, medir variables de temperatura por medio de una entrada analógica.

El diseño de la solución SCADA nos ayuda a tener la información disponible, de momento en una red local pero con la capacidad de ser compartida por WEB y tener disponibles todos los datos estando fuera de la fábrica y tomar decisiones de manera remota, la base de datos se encuentra montada en un servidor, el cual trabaja 24 horas con redundancia de datos y respaldo de energía por lo que se minimiza la pérdida de datos.

La precisión del dispositivo de medición es de un $\pm 2\%$ con respecto a los valores reales de voltaje y corriente y se logra una repetitibilidad del 97%, esto hace que la solución sea confiable para la industria y que dará información paralela a la facturación de la Comisión Federal de Electricidad.

Esta herramienta al ser aplicada en todas las máquinas de la producción dará como resultado:

- Identificar las áreas de oportunidad que ofrecen potencial de ahorro de energía.
- Analizar datos sobre consumos energéticos mensuales, costos de energía y cantidades producción para mejorar el entendimiento de los factores que contribuyen a la variación de los índices energéticos de la planta.
- Obtener el balance energético global de la planta, así como balances energéticos de los equipos y líneas de producción en consumos de energía para su cuantificación.
- Determinar y evaluar económicamente los volúmenes de ahorro alcanzables y las medidas técnicamente aplicables para lograrlo.
- Tener indicadores para la evaluación periódica de los consumos eléctricos.

6 Bibliografía

- [1] Keith Stouffer, Joe Falco, and Karen Scarfone, "Guide to Industrial Control Systems (ICS) Security," *National Institute of Standards and Technology*, p. 155, 2011.
- [2] Brendan Galloway and Gerhard Hancke, "Introduction to Industrial Control Networks," 2012.
- [3] Diego Zornoza. (2013, Octubre) Blog de Redes Industriales. [Online].
<http://diegozornoza.blogspot.mx/2013/10/redes-de-comunicacion.html>
- [4] José Ignacio Armesto Quiroga, "Instalación de Sistemas de Automatización y Datos," in *Dpto. Ingeniería de Sistemas y Automática*, Vigo, 2007.
- [5] "Guía de soluciones de automatización," in *Guía de soluciones de automatización.*: Schneider Electric, 2009, pp. 198-231.
- [6] Manuel Jiménez Buendía, "Comunicaciones Industriales," in *Univ Politécnica de Cartagena. Departamento de Tecnología Electrónica*. Cartagena, Colombia: Universidad Politécnica de Cartagena, 2006, p. 10.
- [7] G. Velasco et al., "Transductor Flux gate para la Medida de Corriente Continua y Corriente Alterna de Bajo Consumo," *Departamento Técnico – Grupo Premo S.A*, p. 6.
- [8] Nota de Aplicación EQUITEL. (2008, 2008) EQUITEL. [Online].
http://www.equitel.es/datos/Documentos/NotasAplicacion/RS485_NA_es_v3.pdf
- [9] Dominique Paret, *Multiplexed Networks for Embedded Systems: CAN, LIN, FlexRay, Safe by-Wire*. Francia: John Wiley & Sons, 2007.
- [10] Eduardo Garcia Breijo, *Compilador C CCS y Simulador Proteus Para Micro Controladores PIC 1a Edición*. México: Alfaomega, 2008.
- [11] Julio Deifilio Pañi Arevalo and Pablo Geovanny Pinos Guantamber, "Compendio del sistema de transmisión de datos CAN-BUS," Quito Ecuador, 2007.
- [12] "Guía de soluciones de automatización," in *Guía de soluciones de automatización.*: Schneider Electric, 2009, pp. 199-231.
- [13] Javier Barragán Piña. Tutoría virtual de Javier Barragán Piña. Topologías de red. [Online].
<http://uhu.es/antonio.barragan/content/5topologias>
- [14] Aquilino Rodriguez Penin, *Comunicaciones Industriales Guía Práctica.*: Marcombo, 2008.

7 Anexos

7.1 Código de SCADA

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO.Ports;
using System.Threading;

namespace Monitoreo_Corriente_Voltaje
{
    public partial class Form1 : Form
    {
        private int[] Bauds = { 1200, 2400, 4800, 9600, 19200, 38400, 57600 };
        string[] ports = null;
        private SerialPort serialport;
        char[] valores = new char[32];

        public Form1()
        {
            InitializeComponent();
            ports = SerialPort.GetPortNames();
            UpdateCheckPorts();

            CheckForIllegalCrossThreadCalls = false;
            chart1.ChartAreas[0].AxisX.Minimum = 0;
            chart1.ChartAreas[0].AxisX.Maximum = 15;
            chart1.ChartAreas[0].AxisX.Interval = 1;

            chart1.ChartAreas[0].AxisY.Minimum = 0;
            chart1.ChartAreas[0].AxisY.Maximum = 12;
            chart1.ChartAreas[0].AxisY.Interval = 1;

            cBoxBauds.DataSource = Bauds; cBoxBauds.SelectedItem = cBoxBauds.Items[3];
        }

        private void UpdateCheckPorts()
        {
            if (ports.Length == 0)
            {
                cBoxPortNames.DataSource = null;
                lblPortsFound.Visible = true;
                lblPortsFound.Text = "Puertos No Disponibles";
                btnConnect.Enabled = false;
            }
        }
    }
}
```

```
        else
        {
            cBoxPortNames.DataSource = SerialPort.GetPortNames();
            lblPortsFound.Visible = true;
            lblPortsFound.Text = "Puertos Disponibles";
        }
    }

    private void btnConnect_Click(object sender, EventArgs e)
    {
        if ((sender as Button).Text == "Conectar")
        {
            if (cBoxPortNames.SelectedItem != null && cBoxBauds.SelectedItem !=
null)
            {
                int baud = Convert.ToInt32(cBoxBauds.SelectedItem);
                string portName = cBoxPortNames.SelectedItem.ToString();
                try
                {
                    serialport = new SerialPort(portName, baud);
                    serialport.Open();
                    serialport.ReadTimeout = 200;
                    serialport.WriteTimeout = 200;
                    MessageBox.Show("Dispositivo conectado", "Notification",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                    btnConnect.Text = "Desconectar";
                    btnConnect.BackColor = Color.Red;
                    serialport.DataReceived += serialport_DataReceived;
                    timer1.Start();
                }
                catch
                {
                    MessageBox.Show("Conexion fallida, Puerto NO inicializado",
"Notification", MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            else
            {
                MessageBox.Show("Verificar los campos obligatorios, Puerto y
Baudios", "Verificacion", MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }
        else if ((sender as Button).Text == "Desconectar")
        {
            serialport.DataReceived -= serialport_DataReceived;
            if (serialport.IsOpen)
            {
                serialport.DiscardInBuffer();
                serialport.DiscardOutBuffer();
                serialport.Close();
            }
            MessageBox.Show("Dispositivo desconectado", "Notification",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            btnConnect.Text = "Conectar";
        }
    }
}
```



```
        btnConnect.BackColor = Color.Lime;
        textBoxComandos.Clear();
        timer1.Stop();
    }
}

private static Decimal ParseHexString(string hexNumber)
{
    long result = 0;
    long.TryParse(hexNumber, System.Globalization.NumberStyles.HexNumber, null,
out result);
    return result;
}

void serialport_DataReceived(object sender, SerialDataReceivedEventArgs e)
{
    try
    {
        Thread.Sleep(205);
        int num = 157;
        char[] buffer = new char[157];

        try { serialport.Read(buffer, 0 , num); }
        catch { }
        for (int k = 0; k < 157; k++)
        {
            textBoxComandos.AppendText(buffer[k].ToString());
        }

        valores[0] = buffer[157 - 108];
        valores[1] = buffer[157 - 107];
        valores[2] = buffer[157 - 106];
        valores[3] = buffer[157 - 105];
        valores[4] = buffer[157 - 8];
        valores[5] = buffer[157 - 7];
        valores[6] = buffer[157 - 6];
        valores[7] = buffer[157 - 5];
        valores[8] = buffer[157 - 96];
        valores[9] = buffer[157 - 95];
        valores[10] = buffer[157 - 94];
        valores[11] = buffer[157 - 93];
        valores[12] = buffer[157 - 88];
        valores[13] = buffer[157 - 87];
        valores[14] = buffer[157 - 86];
        valores[15] = buffer[157 - 85];

        valores[16] = buffer[157 - 84];
        valores[17] = buffer[157 - 83];
        valores[18] = buffer[157 - 82];
        valores[19] = buffer[157 - 81];
        valores[20] = buffer[157 - 12];
        valores[21] = buffer[157 - 11];
        valores[22] = buffer[157 - 10];
        valores[23] = buffer[157 - 9];
        valores[24] = buffer[157 - 60];
    }
}
```

```
        valores[25] = buffer[157 - 59];
        valores[26] = buffer[157 - 58];
        valores[27] = buffer[157 - 57];
        valores[28] = buffer[157 - 28];
        valores[29] = buffer[157 - 27];
        valores[30] = buffer[157 - 26];
        valores[31] = buffer[157 - 25];
        textBoxVoltaje.Text = (((256 * ParseHexString((valores[0].ToString() +
valores[1].ToString())) + ParseHexString((valores[0].ToString() +
valores[1].ToString())) / 100).ToString());
        textBoxFreq.Text = (((256 * ParseHexString((valores[4].ToString() +
valores[5].ToString())) + ParseHexString((valores[6].ToString() +
valores[7].ToString())) / 1000).ToString());
        textBoxL1L2.Text = (((256 * ParseHexString((valores[8].ToString() +
valores[9].ToString())) + ParseHexString((valores[10].ToString() +
valores[11].ToString())) / 100).ToString());
        textBoxL1L3.Text = (((256 * ParseHexString((valores[12].ToString() +
valores[13].ToString())) + ParseHexString((valores[14].ToString() +
valores[15].ToString())) / 100).ToString());
        textBoxCorriente.Text = (((256 * ParseHexString((valores[16].ToString()
+ valores[17].ToString())) + ParseHexString((valores[18].ToString() +
valores[19].ToString())) / 100).ToString());
        textBoxPF.Text = (((256 * ParseHexString((valores[20].ToString() +
valores[21].ToString())) + ParseHexString((valores[22].ToString() +
valores[23].ToString())) / 100).ToString());
        textBoxPACT.Text = (((256 * ParseHexString((valores[24].ToString() +
valores[25].ToString())) + ParseHexString((valores[26].ToString() +
valores[27].ToString())) / 100).ToString());
        textBoxPAPT.Text = (((256 * ParseHexString((valores[28].ToString() +
valores[29].ToString())) + ParseHexString((valores[30].ToString() +
valores[31].ToString())) / 100).ToString());
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.ToString());
    }
}

private void buttonClearH_Click(object sender, EventArgs e)
{
    textBoxComandos.Clear();
}

private void checkBox3D_CheckedChanged(object sender, EventArgs e)
{
    if (checkBox3D.Checked == true)
    {
        chart1.ChartAreas[0].Area3DStyle.Enable3D = true;
    }
    else
    {
        chart1.ChartAreas[0].Area3DStyle.Enable3D = false;
    }
}
```

```
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    if (comboBox1.SelectedItem.ToString() == "Point")
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Point;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Point;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Point;
    }
    else if (comboBox1.SelectedItem.ToString() == "Line")
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Line;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Line;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Line;
    }
    else if (comboBox1.SelectedItem.ToString() == "Spline")
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Spline;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Spline;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Spline;
    }
    else if (comboBox1.SelectedItem.ToString() == "Bar")
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Bar;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Bar;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Bar;
    }
    else if (comboBox1.SelectedItem.ToString() == "StepLine")
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.StepLine;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.StepLine;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.StepLine;
    }
    else if (comboBox1.SelectedItem.ToString() == "Column")
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Column;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Column;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Column;
    }
}
```

```
    }
    else if (comboBox1.SelectedItem.ToString() == "Area")
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Area;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Area;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Area;
    }
    else if (comboBox1.SelectedItem.ToString() == "Range")
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Range;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Range;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Range;
    }
    else
    {
        chart1.Series["Sensor 1"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Radar;
        chart1.Series["Sensor 2"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Radar;
        chart1.Series["Sensor 3"].ChartType =
System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Radar;
    }
}

private void cBoxPortNames_Click(object sender, EventArgs e)
{
    UpdateCheckPorts();
}
}
```

7.2 Interfaz visual de software SCADA

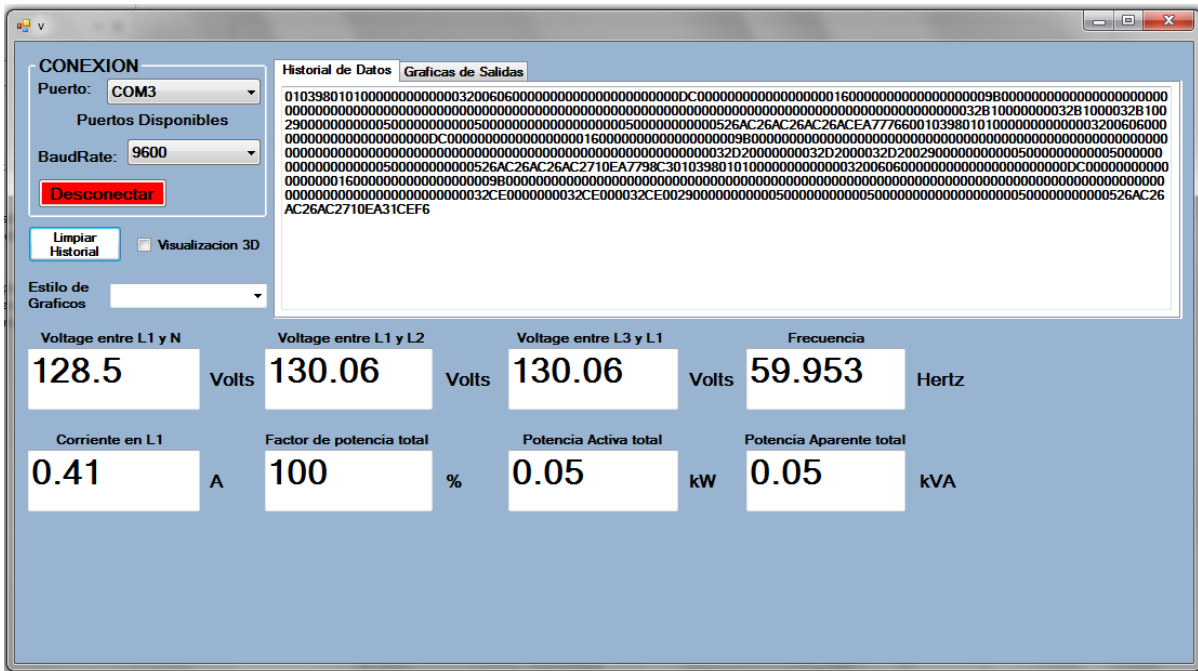


Figura 7.1: Interfaz visual SCADA

7.3 Código dispositivo final

```

#include <18F2550.h>
#fuses HSPLL,NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,PLL2,CPUDIV1,VREGEN,NOPBADEN, NOPUT
#use delay(clock=4800000) // Frecuencia máxima de trabajo.

#include "usb_cdc.h" // Descripción de funciones del USB.
#include "usb_desc_cdc.h" // Descriptores del dispositivo USB.
#include <ctype.h>
#include <stdio.h>

#define MODBUS_TYPE MODBUS_TYPE_MASTER
#define MODBUS_SERIAL_RX_BUFFER_SIZE 255
#define MODBUS_SERIAL_BAUD 9600
#define MODBUS_SERIAL_PARIDAD N //E:EVEN, N:NONE, O:ODD
#define MODBUS_SERIAL_ENABLE_PIN pin_b2 // Controls DE pin
#define MODBUS_SERIAL_RX_ENABLE pin_b3 // Controls RE pin
#define MODBUS_SERIAL_TIMEOUT 100000 //us
#define MODBUS_SERIAL_RX_PIN pin_c7
#define MODBUS_SERIAL_TX_PIN pin_c6
#define LED pin_b1

#include "./modbus.c"
#use rs232(baud=9600, xmit=PIN_C0, rcv=PIN_C1, stream=XBEE)

int8 nodo=0x01; //Slave Adress
int16 start_address=0x00; //Start Address to read data
int16 quantity=0x4C; //Data quantity to read
char rdato;
char cadena[159];
int8 index=0;

void main (void)
{
    modbus_init();
    modbus_read_holding_registers(nodo,start_address,quantity);

    while (TRUE)
    {
        if(kbhit(MODBUS_SERIAL)) //If the response is received
        {
            rdato = getch();
            if(index == 0)
            {
                if (rdato==0x01)
                {
                    cadena[index]=rdato;
                    index++;
                }
            }
            else if (index > 0 && index<159)
            {
                cadena[index]=rdato;
                index++;
            }
        }
    }
}

```

```
    }
    output_high(LED);
}

if (index >= 159)
{
    for (int j=2; j<159;j++)
    {
        fprintf(XBEE,"%X", cadena[j]);
    }
    for(int k = 0; k< 100; k++)
    {
        delay_ms(100);
    }
    modbus_init();
    modbus_read_holding_registers(nodo,start_address,quantity);
    index=0;
    output_low(LED);
}
}
```

7.4 Diagrama electrónico de coordinador

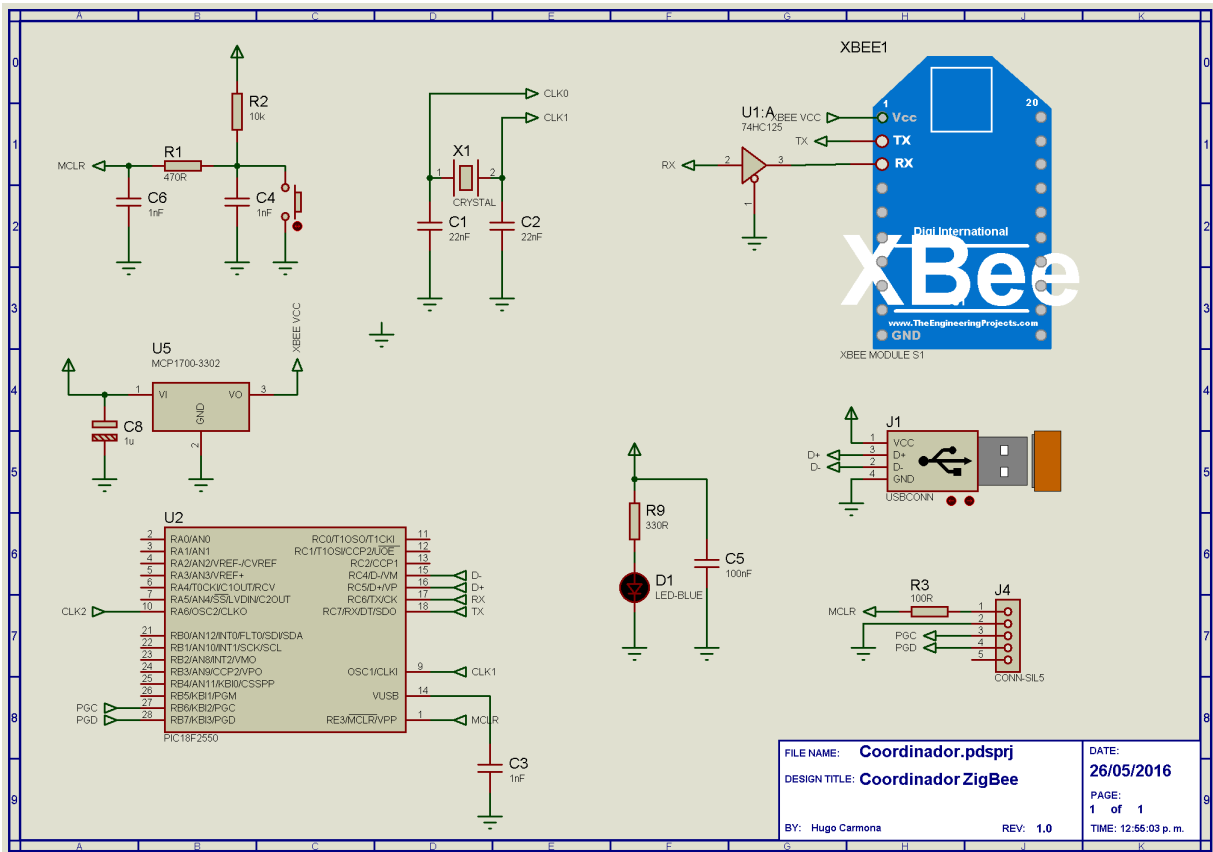


Figura 7.2: Diagrama de coordinador

7.5 Diagrama electrónico de dispositivo final

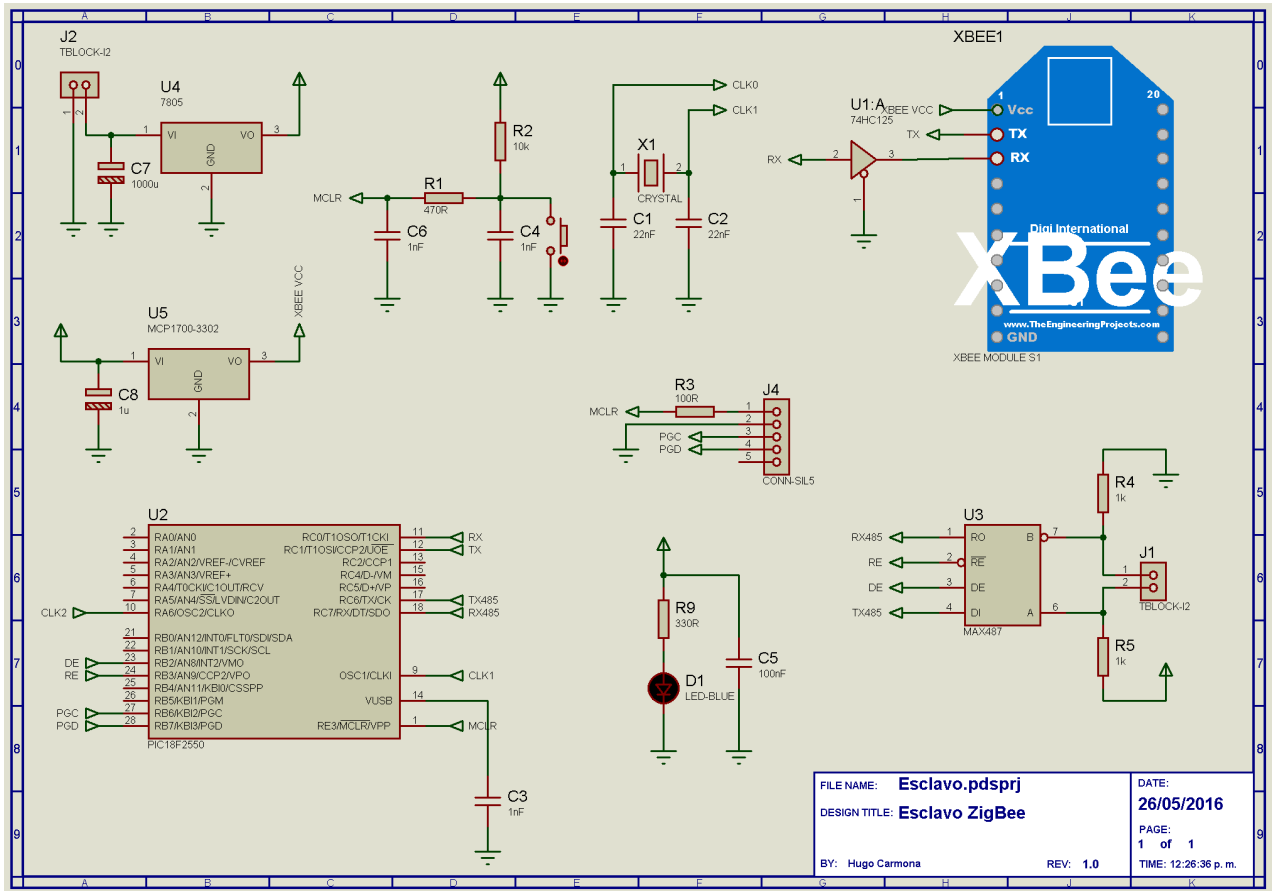


Figura 7.3: Diagrama de dispositivo final