ITCA FEPADE
TÉCNICOS E INGENIEROS

ISBN: 978-99983-69-02-3 (Impreso)
ISBN: 978-99983-69-15-3 (Ebook, pdf)

INFORME FINAL DE INVESTIGACIÓN

DISEÑO Y CONSTRUCCIÓN DE UNIDAD TERMINAL REMOTA DE BAJO COSTO PARA UN LABORATORIO DE PRUEBAS DE SISTEMAS ELECTRONEUMÁTICOS

APLICACIÓN EN ITCA-FEPADE CENTRO REGIONAL SANTA ANA

DOCENTE INVESTIGADOR PRINCIPAL
ING. CARLOS LEVI CARTAGENA LOBOS

DOCENTE COINVESTIGADOR
ING. DAVID ERNESTO CORTEZ PÉREZ

TÉCNICO EN INGENIERÍA ELÉCTRICA Y TÉCNICO EN DESARROLLO DE SOFTWARE CENTRO REGIONAL ITCA-FEPADE SANTA ANA

ENERO 2023

GOBIERNO DE EL SALVADOR
MINISTERIO DE EDUCACIÓN, CIENCIA Y TECNOLOGÍA

ITCA EDITORES

ESCUELA ESPECIALIZADA EN INGENIERÍA ITCA-FEPADE
DIRECCIÓN DE INVESTIGACIÓN Y PROYECCIÓN SOCIAL
SANTA TECLA, LA LIBERTAD, EL SALVADOR, CENTRO AMÉRICA

Comisión de Acreditación de la Calidad Académica
INSTITUCION ACREDITADA

ITCA FEPADE
TÉCNICOS E INGENIEROS

ISBN: 978-99983-69-02-3 (Impreso)
ISBN: 978-99983-69-15-3 (Ebook, pdf)

INFORME FINAL DE INVESTIGACIÓN

DISEÑO Y CONSTRUCCIÓN DE UNIDAD TERMINAL REMOTA DE BAJO COSTO PARA UN LABORATORIO DE PRUEBAS DE SISTEMAS ELECTRONEUMÁTICOS

APLICACIÓN EN ITCA-FEPADE CENTRO REGIONAL SANTA ANA

DOCENTE INVESTIGADOR PRINCIPAL
ING. CARLOS LEVI CARTAGENA LOBOS

DOCENTE COINVESTIGADOR
ING. DAVID ERNESTO CORTEZ PÉREZ

TÉCNICO EN INGENIERÍA ELÉCTRICA Y TÉCNICO EN DESARROLLO DE SOFTWARE CENTRO REGIONAL ITCA-FEPADE SANTA ANA

ENERO 2023

GOBIERNO DE EL SALVADOR
MINISTERIO DE EDUCACIÓN, CIENCIA Y TECNOLOGÍA

ITCA EDITORES

ESCUELA ESPECIALIZADA EN INGENIERÍA ITCA-FEPADE
DIRECCIÓN DE INVESTIGACIÓN Y PROYECCIÓN SOCIAL
SANTA TECLA, LA LIBERTAD, EL SALVADOR, CENTRO AMÉRICA

Comisión de Acreditación de la Calidad Académica
INSTITUCION ACREDITADA

Rector
Ing. Carlos Alberto Arriola Martínez

Vicerrector Académico
Ing. Christian Antonio Guevara Orantes

Director de Investigación y Proyección Social
Ing. Mario W. Montes Arias

Dirección de Investigación y Proyección Social
Ing. David Emmanuel Ágreda Trujillo
Inga. Ingrid Janeth Ulloa de Posada
Téc. Alexandra María Cortez Campos
Sra. Delmy Roxana Reyes Zepeda

Director Centro Regional Santa Ana
Ing. Manuel Antonio Chicas Villeda

629.804
C322d slv
Cartagena Lobos, Carlos Levi, 1984-
Diseño y construcción de unidad terminal remota de bajo costo para un laboratorio de pruebas de sistemas electro-neumáticos [recurso electrónico] / Carlos Levi Cartagena Lobos, David Ernesto Cortez Pérez. -- 1ª ed. -- Santa Tecla, La Libertad, El Salv. : ITCA Editores, 2023.
1 recurso electrónico, (56 p. : il. col. ; 28 cm.)

Datos electrónicos (1 archivo : pdf, 5 MB). -- https://www.itca.edu.sv/produccion-academica/

ISBN: 978-99983-69-02-3 (Impreso)
ISBN: 978-99983-69-15-3 (Ebook, pdf)

1. Dispositivos electromecánicos - Sistemas de comunicación. 2. Máquinas electroneumáticas. 3. Control automático. I. Cortez Pérez, David Ernesto, 1995- coaut. II. Título.

Autor
Ing. Carlos Levi Cartagena Lobos

Co Autor
Ing. David Ernesto Cortez Pérez

Tiraje: 13 ejemplares
Año 2023

Este documento técnico es una publicación de la Escuela Especializada en Ingeniería ITCA–FEPADE; tiene el propósito de difundir la Ciencia, la Tecnología y la Innovación CTI, entre la comunidad académica, el sector empresarial y la sociedad, como un aporte al desarrollo del país. Para referirse al contenido debe citar el nombre del autor y el título del documento. El contenido de este Informe es responsabilidad de los autores.

Atribución-No Comercial
Compartir Igual
4.0 Internacional

Esta obra está bajo una licencia Creative Commons. No se permite el uso comercial de la obra original ni de las posibles obras derivadas, cuya distribución debe hacerse mediante una licencia igual que la sujeta a la obra original.

Escuela Especializada en Ingeniería ITCA-FEPADE
Km 11.5 carretera a Santa Tecla, La Libertad, El Salvador, Centro América
Sitio Web: www.itca.edu.sv
TEL: (503)2132-7423

CONTENIDO

1. INTRODUCCIÓN

Una Unidad Terminal Remota RTU permite procesar de manera continua estados y eventos para gestionar los equipos físicos en una instalación industrial. Además, facilita la comunicación con los sistemas SCADA. Las RTU son los equipos más utilizados para el telecontrol de instalaciones distribuidas y/o desatendidas. Si se compara un PLC con una RTU, hay algunas diferencias importantes: los PLC están diseñados para uso industrial, donde se prioriza el tiempo de respuesta en el control de los procesos en milisegundos. En cambio, las RTU están pensadas para la gestión de infraestructuras, donde principalmente se deben cumplir dos grandes requisitos: autonomía la fiabilidad de los datos. Por ello, ofrecen múltiples métodos de conexión (Ethernet, serie, GPRS/3G/4G, radio) y diferentes protocolos de comunicación (Modbus, SNMP, IEC-104, DNP3).

El objetivo de este proyecto fue diseñar y construir una Unidad Terminal Remota RTU de bajo costo para un entrenador del laboratorio de pruebas de sistemas electroneumáticos. El proyecto se ejecutó utilizando el diseño experimental llevando a cabo diferentes actividades. Como primera etapa se diseñó el prototipo tomando el enfoque modular incluyendo los elementos esenciales que conforman las unidades terminales remotas, es decir que se diseñó un módulo de entradas de señales digitales, un módulo para la salida de señales digitales y un módulo central para el procesamiento de datos y manejo de las comunicaciones alámbricas e inalámbricas de la RTU. En la segunda etapa se realizó el testeo de la RTU en los laboratorios de ITCA-FEPADE, para garantizar el correcto funcionamiento de la RTU.

Como resultado se obtuvo el diseño innovador de creación propia de una Unidad Terminal Remota RTU de bajo costo tomando elementos del hardware abierto, con la que se puede establecer la comunicación con un PLC S7 1200, mediante el protocolo MODBUS TCP/IP. Además, permite el envío y recepción de comandos de forma inalámbrica bajo el estándar Zigbee desde cualquier parte del mundo.

Con el desarrollo de la investigación queda demostrado que la fiabilidad en la comunicación entre un PLC S7 1200 de Siemens y una RTU construida con hardware abierto se logra si se usa un protocolo de comunicación industrial como el MODBUS TCP/IP.

Para validar el funcionamiento, se realizaron diferentes pruebas experimentales, como medir la latencia en la transmisión de datos desde una conexión local y una conexión remota usando el VPN. Se diseñó una aplicación con LabView para manipular un cilindro neumático desde cualquier parte del mundo, obteniendo resultados satisfactorios.

Los resultados al detalle, los diseños y las pruebas experimentales para validar la RTU se pueden encontrar en el desarrollo del contenido de este informe.

2. Planteamiento del Problema

2.1. Definición del Problema

El Centro Regional Santa Ana de la Escuela Especializada en Ingeniería ITCA-FEPADE cuenta con 1 módulo para prácticas de electroneumática básica. Este módulo se utiliza para prácticas de laboratorio en el campo de la mecatrónica, realizando operaciones de control y programación mediante autómatas programables. Bajo la configuración actual la única forma de poder establecer una conexión con un PLC es mediante un cable, lo que dificulta el desarrollo de prácticas cuando se trabaja con varios grupos. Por lo que se requiere de un dispositivo que provea de flexibilidad para integrar hardware externo sin necesidad de utilizar un cable. Se requiere de una solución tecnológica que sea fácil de implementar empleando estándares industriales de comunicación para el envío de comandos para el control del sistema electroneumático que se desea probar. En ese sentido, surge la siguiente interrogante: ¿Cómo transferir comandos de control de forma inalámbrica desde un autómata programable para realizar operaciones en un módulo de pruebas para sistemas electroneumáticos?

2.2. Antecedentes/Estado de la Técnica

En el campo industrial existen una variedad de soluciones para conectar un PLC, para el control de un proceso de forma remota, una de las soluciones más difundidas en el entorno industrial son las RTU o Unidad Terminal Remota, que es una pequeña y robusta computadora que proporciona inteligencia en el campo para permitir que un autómata programable o PC se comunique con los instrumentos en los sistemas SCADA. Es una unidad independiente de adquisición y control de datos. Actualmente, existen en el mercado una variedad de RTU que utilizan hardware propietario por ejemplo: SIMATIC RTU 3010C, RTU3030 y RTU3031C de la marca SIEMENS, RTU -LP-ST de la marca SENECA entre otras.

Se han realizado algunos proyectos académicos de diseño de RTU de bajo costo como el propuesto por el Ing. Luis Montalvo: ”Diseño y construcción de unidades terminales remotas RTU de bajo costo para sistemas de control, supervisión y adquisición de datos (SCADA)”, donde describe las partes principales que conforman una RTU, utilizando un sistema micro procesado. [1].

2.3. Justificación

En los talleres de práctica, los grupos en promedio son de 20 personas, por lo que un módulo para prácticas de mecatrónica genera cuellos de botella, limitando las prácticas que se pueden llevar a cabo y reduciendo el tiempo de prácticas por persona debido a la configuración actual del módulo. Una solución para solventar en

parte este problema sería contar con unidades terminales remotas con conectividad inalámbrica, lo que proporcionaría flexibilidad cuando se desarrollan prácticas de programación de autómatas programables. De esta manera, varios grupos podrían programar procesos para el módulo de electroneumática básica, realizando un control de forma remota. El docente podría decidir fácilmente, a través de una computadora y un comando, qué grupo está autorizado para usar el módulo y probar el programa desarrollado.

3. Objetivos

3.1. Objetivo General

Diseñar y construir una Unidad Terminal Remota RTU de bajo costo para un laboratorio de pruebas de sistemas electroneumáticos.

3.2. Objetivos Específicos

1. Diseñar una Unidad Terminal Remota RTU tomando elementos de hardware abierto.
2. Diseñar el firmware de la RTU para que establezca la comunicación con un PLC S7 1200 mediante el protocolo MODBUS.
3. Validar el funcionamiento de la RTU y la red diseñada mediante pruebas de laboratorio con dispositivos electroneumáticos.

4. Hipótesis

La fiabilidad en la comunicación entre un hardware abierto y un PLC 1200 se garantiza si se usa el protocolo de comunicación MODBUS TCP/IP.

5. Marco Teórico

5.1. Concepción y Diseño de las Unidades Terminales Remotas

La adquisición de datos por parte de la estación maestra requiere que la unidad terminal remota (RTU)pueda leer tres tipos distintos de señales. [1]:

Entrada Digital. Contacto libre de potencial que puede adoptar una de dos posiciones, abierto o cerrado. El contacto puede representar el estado sensor, switch, relé o alarma

Entrada de acumulación de pulsos. Es un registro que cuenta el número de veces que un contacto libre de potencial ha sido abierto y cerrado en un período determinado. Los pulsos pueden provenir de un sensor digital.

Entrada Analógica. Utilizada para leer valores de voltaje o corriente DC de naturaleza analógica. Las señales analógicas normalmente provienen de transductores que generan voltajes o corrientes normalizados (0-5 Vdc,0-10 Vdc,4-20mA) que representan magnitudes físicas como temperatura, presión, humedad relativa, masa, etc. La ejecución de tele comandos, provenientes de la estación maestra, tendrá efecto sobre dos tipos distintos de señales. [1]

El módulo de salidas puede manejar las siguientes señales:

Salida Digital. Contacto libre de potencial que puede adoptar una de dos posicione, abierto o cerrado. El contacto puede comandar de eta manera la operación de relés y contactores para el control de altas potencias.

Salida Analógica. Valor de voltaje o corriente de naturaleza analógica, que puede ser utilizada para modificar el Setpoint de un controlador analógico.

Modulo modem. Para establecer la comunicación entre la estación maestra y la RTU se utiliza un modem que permita el envío de comandos empleando protocolos de comunicación industrial. Los protocolos más usados son: El protocolo MODUS TCP y MODBUS RTU.

5.2. El Protocolo MODBUS

MODBUS es un protocolo de comunicaciones originalmente ubicado en la capa de aplicación del modelo OSI, que ofrece comunicación cliente/servidor entre distintos dispositivos y medios. Ha sido el protocolo preferido en la industria desde 1979 [2]. “El protocolo MODBUS se ha expandido para incluir implementaciones a través de protocolo serial, TCP/IP y el User Datagram Protocol (UDP)” [3]. Luego, se describen las características más importantes del protocolo MODBUS sobre una línea serial, que se encuentra ubicado en la segunda capa del modelo OSI. [4]

MODBUS en la capa de enlace

El protocolo MODBUS sobre una línea serial es un protocolo jerárquico de tipo maestro/esclavo, en el que uno de los nodos, el maestro, envía solicitudes explicitas a los nodos esclavos para que sean procesadas y respondidas. Los nodos esclavos no pueden comunicarse entre sí, y no transmiten datos sin la previa solicitud del maestro. El maestro solo puede iniciar un intercambio de información a la vez.

Cada uno de los esclavos es identificado con una dirección, que debe ser única para cada bus de datos. Estas

direcciones pueden ir desde la 1 hasta la 247. Las direcciones entre la 248 y la 255 están reservadas por el protocolo, mientras que la dirección 0 se encuentra designada para el modo de broadcast. El maestro no posee una dirección específica.

En la capa de aplicación la trama MODBUS está conformada por una unidad de datos de protocolo (PDU), mostrada en la figura 1, que contiene la función que desea implementar el maestro y, de haberlos, los datos asociados a ella. Al insertar el protocolo MODBUS en la capa de enlace se le asocian dos nuevos elementos a la PDU: La dirección del esclavo y la comprobación de redundancia cíclica (CRC), como se muestra en la figura 2.

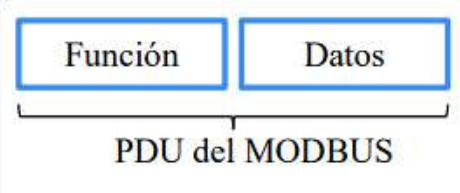


Figura. 1: PDU del MODBUS en la capa de aplicación.

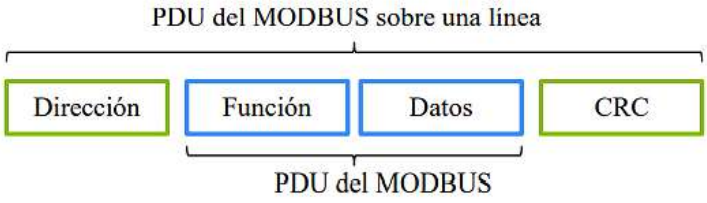


Figura. 2: PDU del MODBUS en la capa de enlace.

El maestro tiene dos maneras de hacer solicitudes a los esclavos: el modo unicast y el modo broadcast [2]. En el modo unicast, el maestro envía una petición a un único esclavo, el cual procesa y responde la petición. Mientras que, en el modo de broadcast, el maestro envía un mensaje a todos los esclavos empleando la dirección 0. Los esclavos procesan la petición, pero no emiten una respuesta. De acuerdo con el contenido de los bits de la PDU el protocolo MODBUS sobre una línea serial presenta dos modos de trasmisión: el modo RTU y el modo ASCII [4].

MODBUS RTU

En el MODBUS RTU cada byte enviado está conformado por dos caracteres hexadecimales de 4 bits. Para enviar un byte se emplea 1 bit de inicio, los 8 bits que conforman el byte a enviar, 1 bit de paridad y 1 bit de parada. Los bytes de datos son enviados del bit menos significativo (LSB) al bit más significativo (MSB). [4]

La trama de MODBUS RTU sigue con el esquema básico de la PDU de MODBUS sobre una línea serial, y está limitada a una longitud máxima de 256 bytes. En el MODBUS RTU, el sistema determina que se finalizó el envío de una trama mediante un tiempo de silencio en la red. Este tiempo queda especificado de acuerdo con la velocidad de transmisión como 3.5 veces el tiempo de envío de un byte por la red. Para velocidades sobre los 19200Bd el protocolo recomienda una espera mínima de 2ms [3]. En la Figura 3 se presenta un esquema con la PDU del MODBUS RTU y los periodos de espera necesarios.

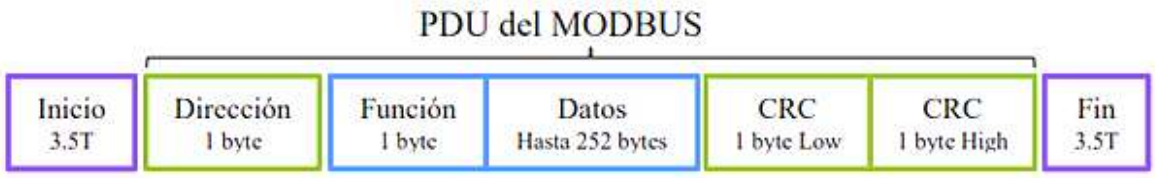


Figura. 3: Trama MODBUS RTU.

Las distintas funciones que se pueden operar en una red MODBUS pueden ser funciones de uso público, definidas por el usuario y funciones reservadas. Las funciones de uso público están definidas en el protocolo y son únicas, pero no todos los dispositivos MODBUS tienen habilitadas todas las funciones públicas. [2]. En la Tabla 1 se encuentran las funciones de uso público definidas para el protocolo MODBUS RTU.

Tabla 1: Definición de códigos de funciones públicas.

Función			Código
Acceso a Datos	Acceso a Bits	Leer entrada discreta	02
		Leer bobinas	01
		Escribir una bobina	05
		Escribir múltiples bobinas.	15
	Acceso de 16 Bits	Leer registro de entrada	04
		Leer múltiples registros	03
		Escribir un registro	06
		Escribir múltiples registros	16
		Leer/Escribir múltiples registros	23
		Enmascarar escritura de registro	22
		Leer FIFO	24
	Acceso a Archivos	Leer archivo	20
		Escribir archivo	21
Diagnóstico		Leer estatus de excepción	07
		Diagnóstico	08
		Obtener contador de eventos	11
		Obtener log de eventos	12
		Reportar ID	17
		Leer identificación de dispositivo	43

Los esclavos emplean los códigos de excepción para indicar al maestro que ha ocurrido un error, originado por varias razones, como por ejemplo el recibir una solicitud incompleta, o que el esclavo no tenga implementado el código de la función solicitada. En la Tabla 2 se presentan los códigos de excepciones más utilizados. [3]

Tabla -2: Códigos de excepciones.

Código de excepción	Descripción
01	El código de la función recibida no está habilitado
02	La solicitud intento acceder a una dirección no válida
03	La solicitud tenía datos incorrectos
04	Ocurrió un error irrecuperable al procesar la solicitud

Modelo de datos

Según su tamaño podemos clasificar en dos los tipos de datos que maneja el protocolo: bits individuales y registros de 2 Bytes. Los bits individuales para entradas y salidas digitales, mientras que los registros de 2 Bytes son para variables que requieran mayor tamaño. En la siguiente tabla se muestran los tipos de datos disponibles [6]:

Tabla 3: Modelo de datos del protocolo MODBUS.

Tipo de objeto	Acceso	Tamaño
Discrete input	Solo leer	1 bit
Coil	Leer/escribir	1 bit
Input register	Solo leer	16 bits
Holding register	Leer/escribir	16 bits

- Discrete input o entrada discreta: puede ser generado por un sistema de entrada/salida.
- Coil o bobina: puede ser modificado por un programa de aplicación.
- Input register o registro de entrada: puede ser generado por un sistema de entrada/salida.
- Holding register o registro de salida: puede ser modificado por un programa de aplicación.

Modelo de direccionamiento

Las direcciones de memoria para acceder a los datos de un dispositivo MODBUS están ordenadas según los tipos de datos [6]:

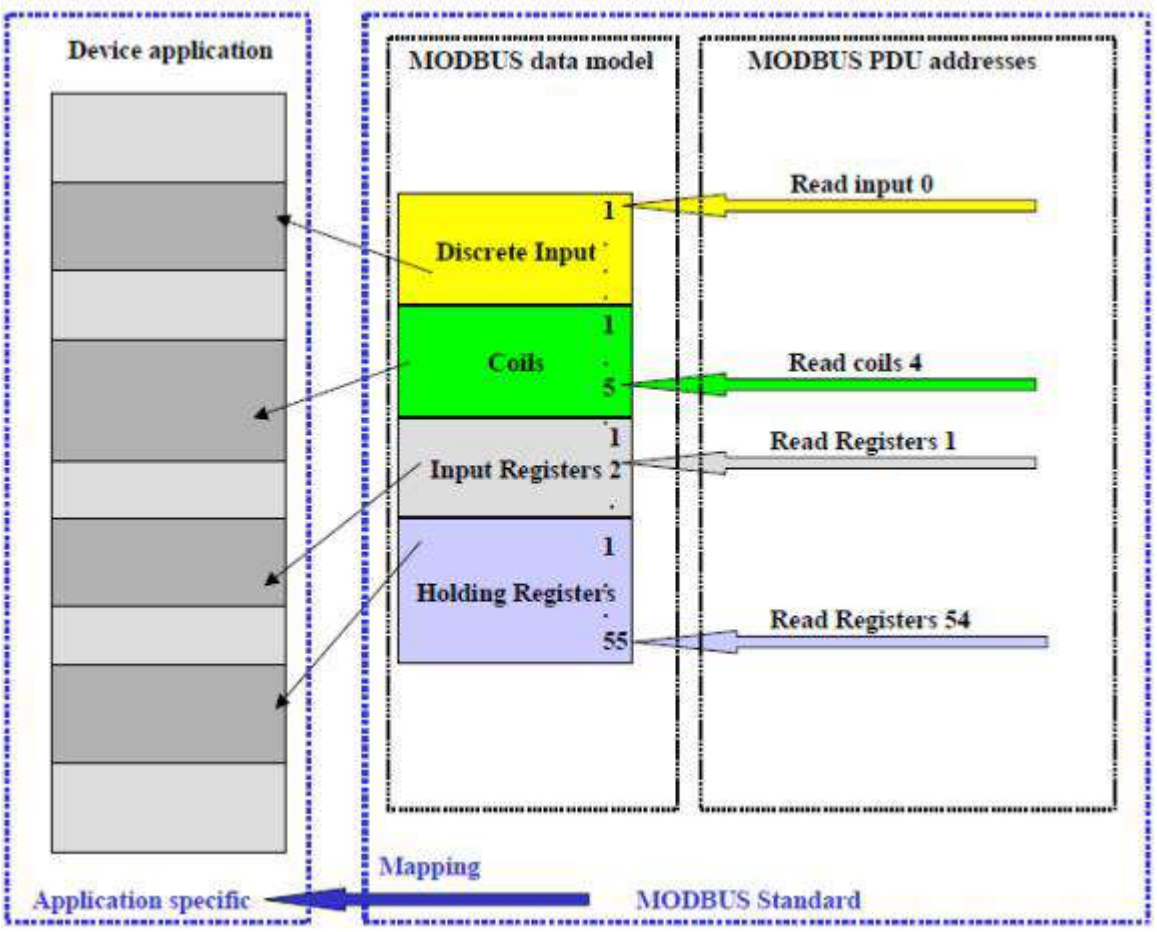


Figura 4: Modelo de direccionamiento en el protocolo MODBUS.

Tal como muestra la figura, existen cuatro bloques que corresponden al modelo de datos. Todas las direcciones se refieren a cero de manera que, si queremos leer el elemento N de un determinado bloque, se direccionará como N-1. El mapeado de las direcciones MODBUS con las direcciones reales del dispositivo es independiente del estándar MODBUS, dependiendo totalmente del fabricante.

MODBUS en la capa física

A nivel de la capa física, el protocolo MODBUS sobre una línea serial puede ser implementado por distintas interfaces. La más común es el RS485 de par trenzado, aunque también está disponible para el estándar RS485 de cuatro cables y en el RS232 para conexiones cortas de tipo punto a punto [4].

En el estándar MODBUS de par trenzado todos los dispositivos están conectados en paralelo por tres cables: el común y el bus de datos conformado por el par trenzado. La velocidad para la transmisión de datos suele ser de 9600Bd y los dispositivos MODBUS también deben implementar la velocidad de 19200Bd. [4]

MODBUS TCP/IP

MODBUS TCP/IP es una variante de la familia MODBUS de protocolos de comunicación simples y neutrales para la supervisión y el control de equipos de automatización. Específicamente, cubre el uso de la mensajería MODBUS en un entorno 'Intranet' o 'Internet', utilizando los protocolos TCP/IP. El uso más común de los protocolos en este momento es para la conexión Ethernet con un PLC, módulos de E/S y "puertas de enlace" a otros buses de campo o redes de E/S simples.

Modbus TCP es la evolución del protocolo Modbus (1979), que permite la implementación de este protocolo sobre redes Ethernet, aumentando el grado de conectividad. Es muy semejante al formato RTU, pero estableciendo la transmisión mediante paquetes TCP/IP. Modbus TCP es un protocolo de comunicaciones situado en el nivel 7 del Modelo OSI, basado en la arquitectura maestro/esclavo RTU (Remote Terminal Unit) o de cliente/servidor TCP/IP (Transmission Control Protocol /Internet Protocol). Esta "versión" del protocolo encapsula la trama base del protocolo Modbus en la capa de aplicación TCP/IP, multiplexando entradas / salidas en un Host/Server en base a dirección IP y un puerto de destino (Sockets). Modbus/TCP simplemente encapsula una trama Modbus en un segmento TCP [5] Figura 5.

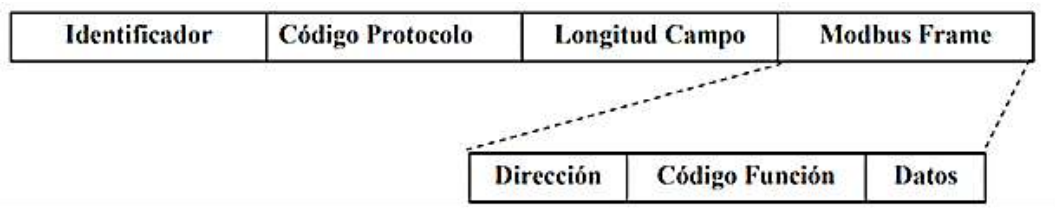


Figura 5: Estructura de trama MODBUS TCP/IP

- Byte 0 Identificador de transacción.
- Byte 1 Identificador de transacción.
- Byte 2 Código de protocolo.
- Byte 3 Código de protocolo.
- Byte 4 Longitud Campo (byte alto) = 0. Los mensajes son menores a 256.
- Byte 5 Longitud Campo (byte bajo). Número de bytes siguientes.
- Byte 6 Dirección Dispositivo (dirección esclavo remoto.).
- Byte 7 Código de Función MODBUS.
- Byte 8 y más. Los datos necesarios para transmitir.

Esta técnica de consulta/respuesta encaja perfectamente con la naturaleza Maestro/Esclavo de Modbus, añadido a la ventaja del determinismo que las redes Ethernet conmutadas ofrecen a los usuarios en la industria. El empleo del protocolo abierto Modbus con TCP proporciona una solución para la gestión desde unos pocos a decenas de miles de nodos.

En los sistemas informáticos se ha establecido un puerto de destino específico identificado con el número 502. Lo mismo que ocurre con otras aplicaciones de Internet:

- Puerto 50: SMTP (Sample Mail Transfer Protocol). Mensajería.
- Puerto 80: HTTP (Hiper Texto Transfer Protocol). Hipermedia.
- Puertos 20,21: FTP (File Transfer Prococol). Archivos.
- Puerto 502: Modbus TCP (Modbus TCP Protocol). Automatización.

Los dispositivos modernos de control tienen directamente conexión a Ethernet con Modbus TCP, no obstante, la trama Modbus RTU se puede trasladar al Modbus TCP, con un Conversor externo de Protocolo (Pasarela / Gateway), que lo encapsula en el protocolo TCP/IP sobre una red Ethernet [5]. Figura 6.

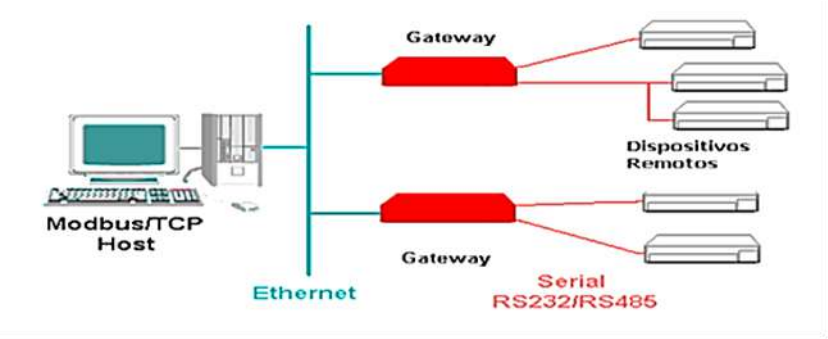


Figura 6: Conversión de MODBUS RTU a MODBUS TCP.

Todas las solicitudes son enviadas vía TCP sobre el puerto registrado 502. Las solicitudes normalmente son enviadas en forma half-duplex. Para realizar la conexión Ethernet entre dispositivos se necesita el cable de red UTP y conectores RJ45. Las prestaciones de un sistema MODBUS TCP/IP sobre Internet dependen básicamente de la red y del hardware, pero se corresponden con los tiempos de respuesta en Internet, que no siempre serán las deseables para un sistema de control. Sin embargo, pueden ser suficientes para la comunicación destinada a la supervisión general, depuración y mantenimiento de los procesos industriales, evitando así costosos desplazamientos al lugar de las instalaciones.

Arquitectura de Red

Con Modbus TCP se implementan redes Ethernet, utilizando la topología en Estrella, donde todos los dispositivos están conectados a través de un dispositivo intermedio. Este dispositivo puede ser un Switch. La estrella es la topología usada en redes corporativas y actualmente se adopta en casi todas las aplicaciones de automatización [5].

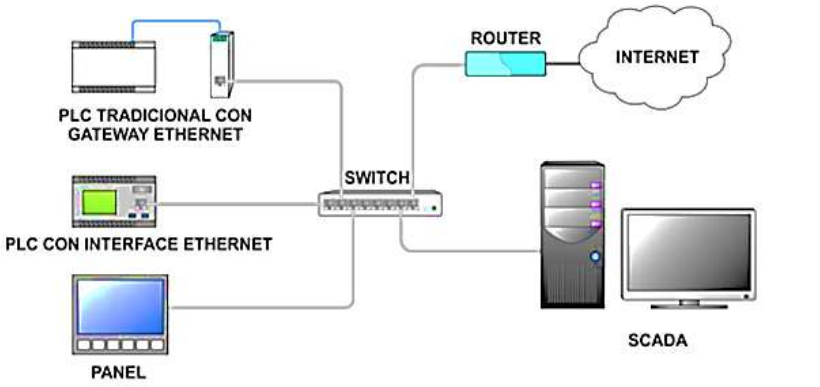


Figura 7: Topología de conexión tipo estrella.

El campo Dirección del esclavo en MODBUS es reemplazado por un byte identificador de unidad, el cual puede ser usado para comunicarse con múltiples unidades terminales independientes, a través de dispositivos tales como puentes o Gateways, los cuales usan una dirección IP única.

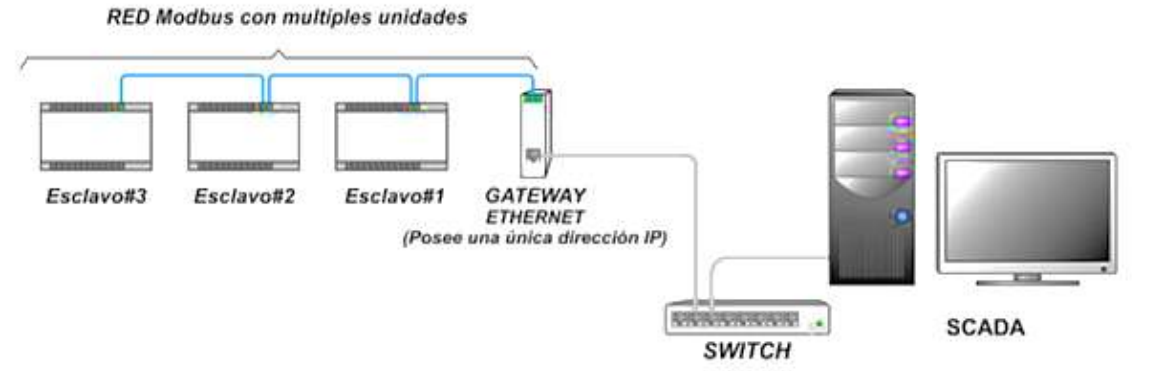


Figura 8: Modelo de conexión con múltiples unidades terminales.

Módulo CPU de una RTU

El módulo CPU constituye el cerebro de la RTU y tiene a cargo las siguientes funciones:

- Recibir, a través del módulo Modem los comandos provenientes de la estación maestra del sistema ESCADA.
- Interpretar dichos comandos y ejecutarlos, a través de la lectura u operación de los puntos de entrada/salida de los módulos correspondientes.
- Enviar los datos requeridos a la estación maestra, a través del módulo MODEM.

6. METODOLOGÍA DE INVESTIGACIÓN

El proyecto se ejecutó con el diseño experimental con diferentes actividades que se detallan a continuación.

- **Diseño y construcción de prototipo de RTU.**

  El prototipo se diseñó siguiendo un enfoque modular, que incluye los elementos esenciales que conforman las unidades terminales remotas. Esto implica el diseño de un módulo de entradas de señales digitales, otro módulo para la salida de señales digitales y un módulo central para el procesamiento de datos y manejo de las comunicaciones alámbricas e inalámbricas de la RTU.

- **Testeo de la RTU en los laboratorios de ITCA.**

  Para garantizar que la unidad terminal remota funciona correctamente y que soluciona el problema para el cual fue concebida, se realizaron diferentes experimentos utilizando de base el laboratorio de electroneumática. Se realizaron pruebas para verificar si los comandos de control se enviaban correctamente en una red conformada por hardware abierto, PLC y computadoras con un sistema ESCADA experimental, utilizando diferentes protocolos de comunicación especialmente MODBUS TCP y Zigbee para la comunicación inalámbrica.

  La descripción de las actividades específicas se detalla en la siguiente matriz metodológica:

Tabla 4: Matriz Metodológica.

OBJETIVOS	ACTIVIDADES	RESULTADOS
1. Diseñar una unidad terminal remota tomando elementos del hardware abierto.	A1. Diseño de circuitos esquemáticos para la RTU A2. Diseño de PCB de la RTU	R1. Circuito esquemático del módulo RTU R2. Placas de circuitos impresos de la unidad terminal remota.
2. Diseñar el firmware de la RTU para que establezca la comunicación con un PLC S7 1200 mediante el protocolo MODBUS. TCP.	A1. Construcción de librerías para el manejo de las comunicaciones en la RTU A2. Programación de bloques de comunicación en Tía Portal para un PLC 1200 A3. Realizar depuración del firmware	R1. Microcontrolador con firmware programado para establecer la comunicación con un PLC 1200 R2. Programas de aplicación para utilizar la RTU con un PLC 1200
3. Validar el funcionamiento de la RTU y la red diseñada mediante pruebas de laboratorio con dispositivos electroneumáticos.	A1. Diseño de aplicaciones experimentales para poner a prueba la red y los protocolos de comunicación utilizando la RTU y el PLC 1200 para el control de un sistema electroneumático.	R1. Guías para la realización de prácticas con el equipo electroneumático, utilizando una RTU para el envío de comandos de forma remota.

7. RESULTADOS

7.1. DISEÑO DE LA RTU

Descripción general del hardware

El hardware de la unidad terminal remota está dividido en 5 módulos: el módulo de alimentación, el módulo de comunicación, el módulo de control, módulo de entradas digitales y el módulo de salidas digitales. El montaje se realizó utilizando una placa prediseñada en la que se colocaron los diferentes componentes requeridos para cada módulo. En la figura 9 se muestra un diagrama de bloques para tener una visión general del hardware de la RTU diseñada.

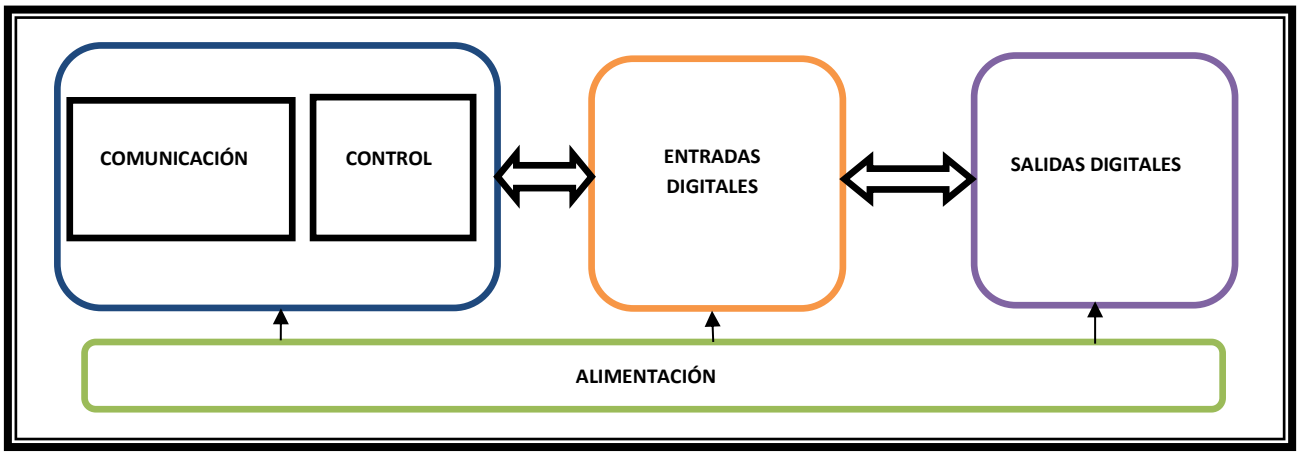


Fig. 9: Diagrama de bloques de la RTU.

Para facilitar los cambios en la funcionalidad de la RTU, se adoptó el enfoque de diseño modular, permitiendo que el producto final se ensamblar por capas. En la figura 10 se muestra una imagen del ensamble final de la RTU.

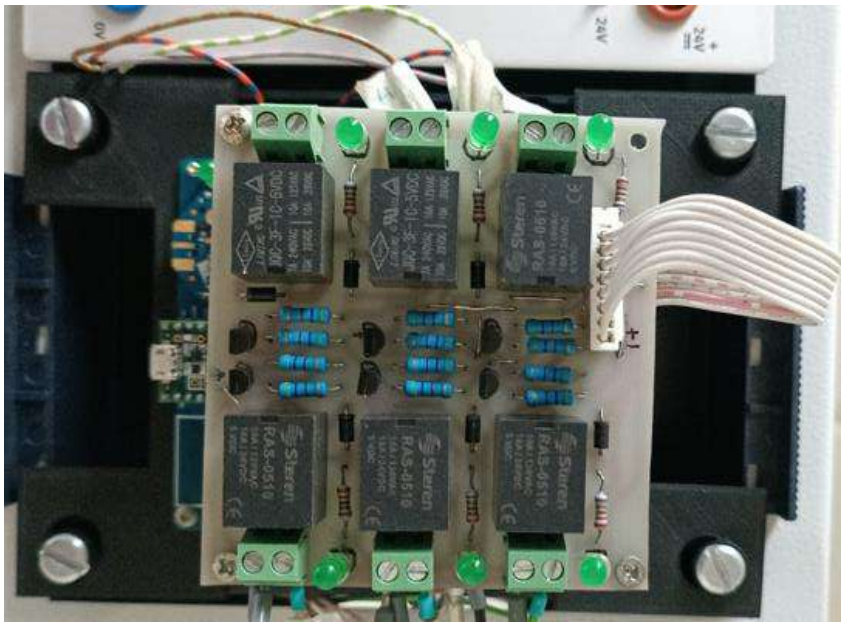


Fig. 10: Ensamble de la RTU.

Descripción del módulo de alimentación

La tarjeta se alimenta con una fuente externa de 24V, por lo que es necesario regular los niveles de tensión apropiados para el funcionamiento de los circuitos integrados utilizados en el proyecto, para adecuar el nivel de tensión a 5v se utilizó el regulador R-78E-05 de la marca RECOM. En la figura 11 y 12 se muestra una imagen y el esquemático de conexión del regulador para aplicaciones estándar.

Fig.11: Regulador de 5v.

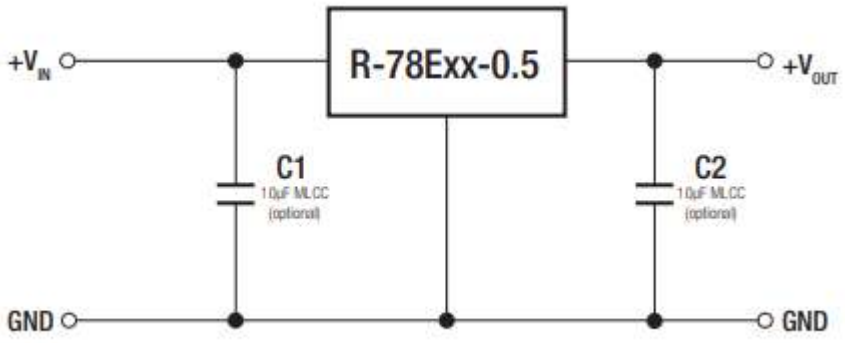


Fig. 12: Diagrama de conexión para aplicaciones estándar

Descripción del módulo de comunicación

La función del módulo de comunicación es permitir que la RTU pueda enviar y recibir comandos de control desde otros dispositivos de forma remota. Con el objetivo de facilitar la integración de la RTU a los módulos electroneumáticos con los que cuenta ITCA-FEPADE, se decidió diseñar el módulo de comunicación para que funcione de forma inalámbrica y para ello se utilizaron antenas Xbee S2C con formato tipo látigo (figura 13), utilizando el estándar ZigBee en el cual se definen un conjunto de protocolos para el armado de redes inalámbricas de corta distancia y baja velocidad de datos. Opera en las bandas de 868 MHz, 915 MHz y 2.4 GHz y puede transferir datos hasta 250Kbps.

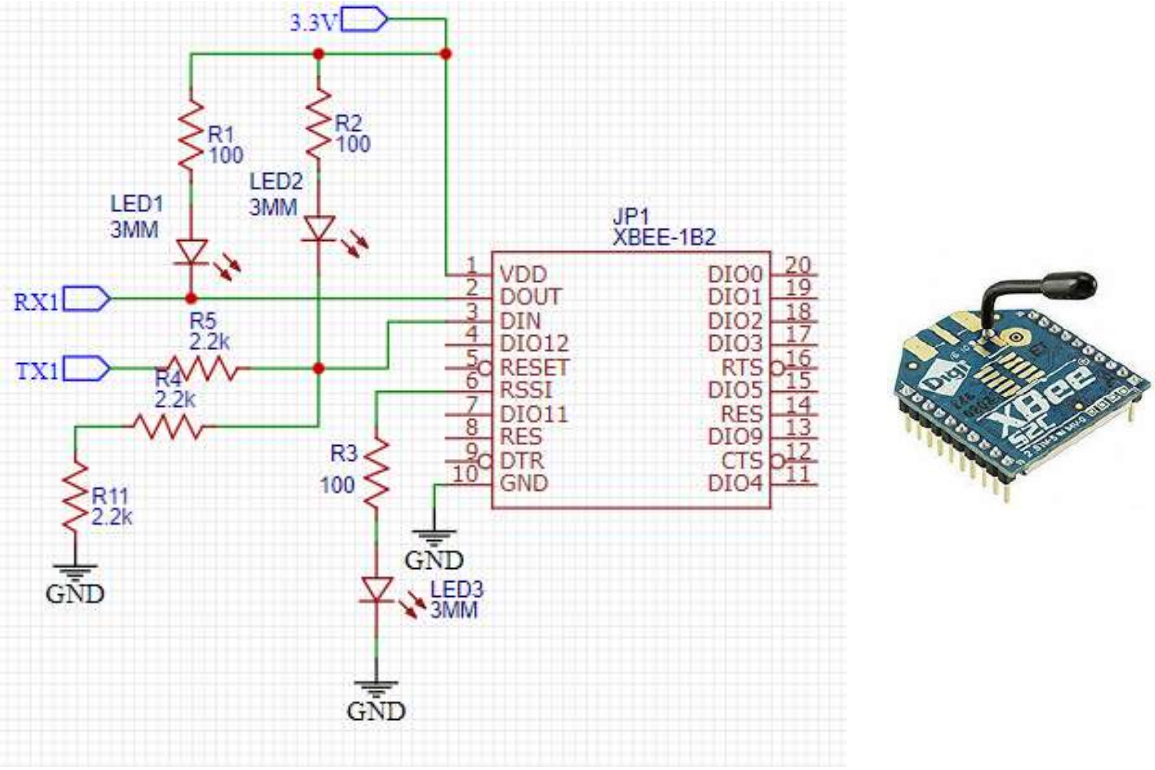


Fig. 13: Antena Xbee S2C y circuito esquemático de las conexiones.

Uno de los objetivos del proyecto es que la RTU se pueda comunicar con un PLC 1200 y para ello se utilizó una pasarela open hardware que convierte los datos enviados bajo el protocolo ZigBee al protocolo MODBUS TCP. El hardware utilizado para la pasarela es un Shield Ethernet W5100, un Arduino uno R3 y un Shield Xbee, el ensamble de la pasarela se muestra en la figura 14.

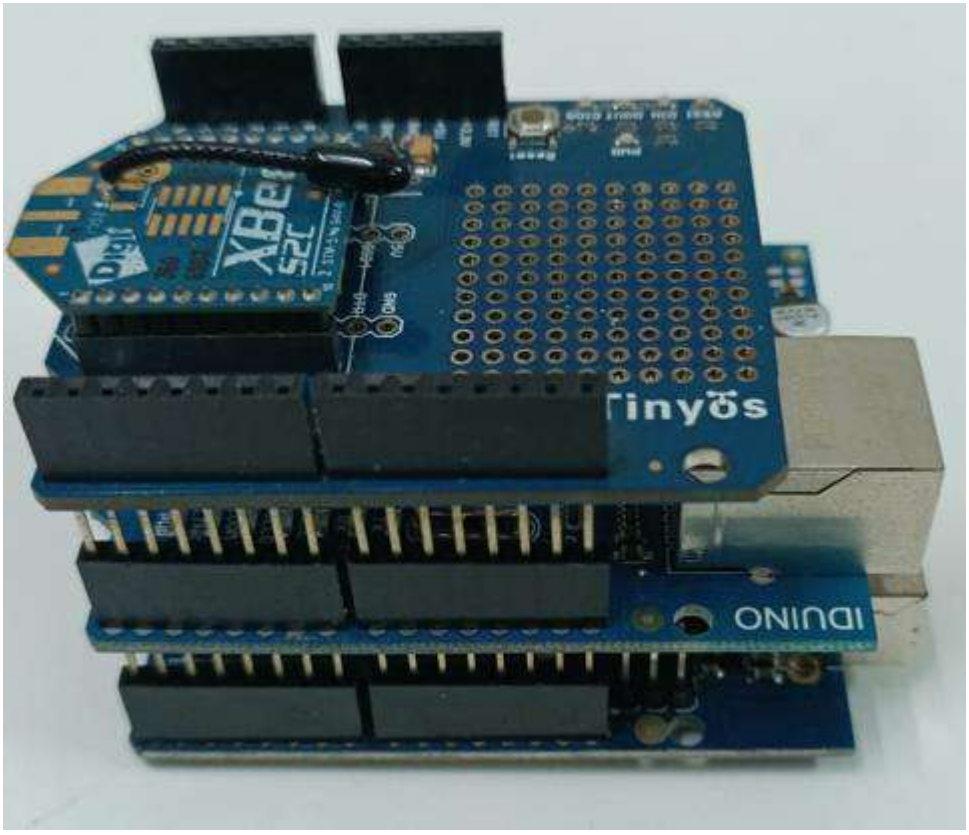


Fig. 14. Pasarela de ZigBee a MODBUS.

Descripción del módulo de control

La función del módulo de control es gestionar el flujo de datos que provienen de los otros componentes de la RTU, comoel módulo de comunicaciones, módulo de entradas y salidas digitales. El núcleo principal del módulo de control está compuesto por una tarjeta Teensy 3.5. Esta tarjeta, desarrollada por PJRC, consiste en una placa repleta de funciones diseñada para prototipado y viene precargada con un gestor de arranque. Esto permite una fácil programación usando conexión USB en placa sin necesidad de un programador externo. El diseño esquemático del módulo de control se muestra en la fig. 15.

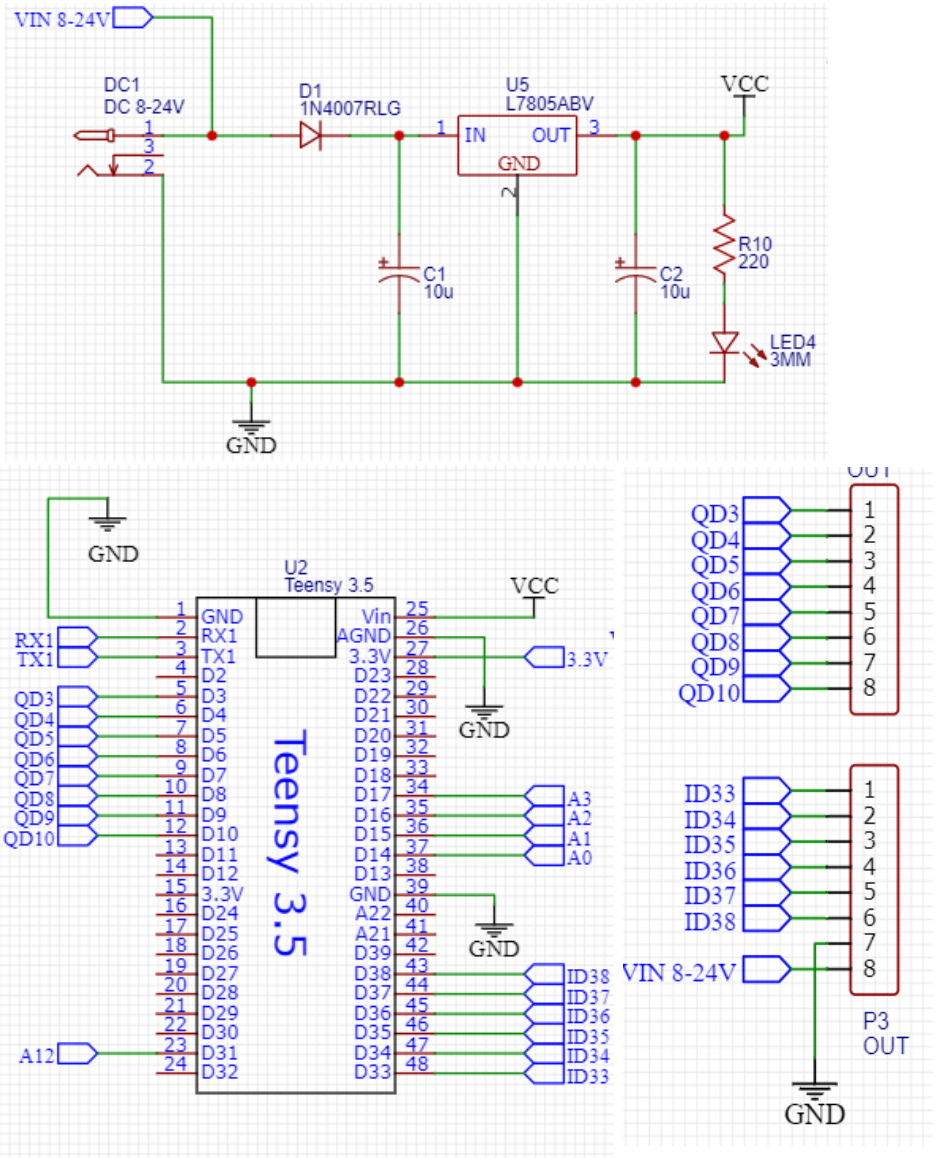


Fig. 15: Diseño esquemático del módulo de control.

Descripción del módulo de entradas digitales

La función del módulo de entradas digitales es acondicionar las señales discretas que provengan de sensores digitales y pulsadores. El módulo electroneumático con el que cuenta ITCA-FEPADE trabaja a 24V DC y para evitar daños a los circuitos internos de la RTU, se diseñó un sistema opto acoplado (Fig. 16), permitiendo de este modo que las señales que ingresan con una tensión de 24V se conviertan en señales a 5V.

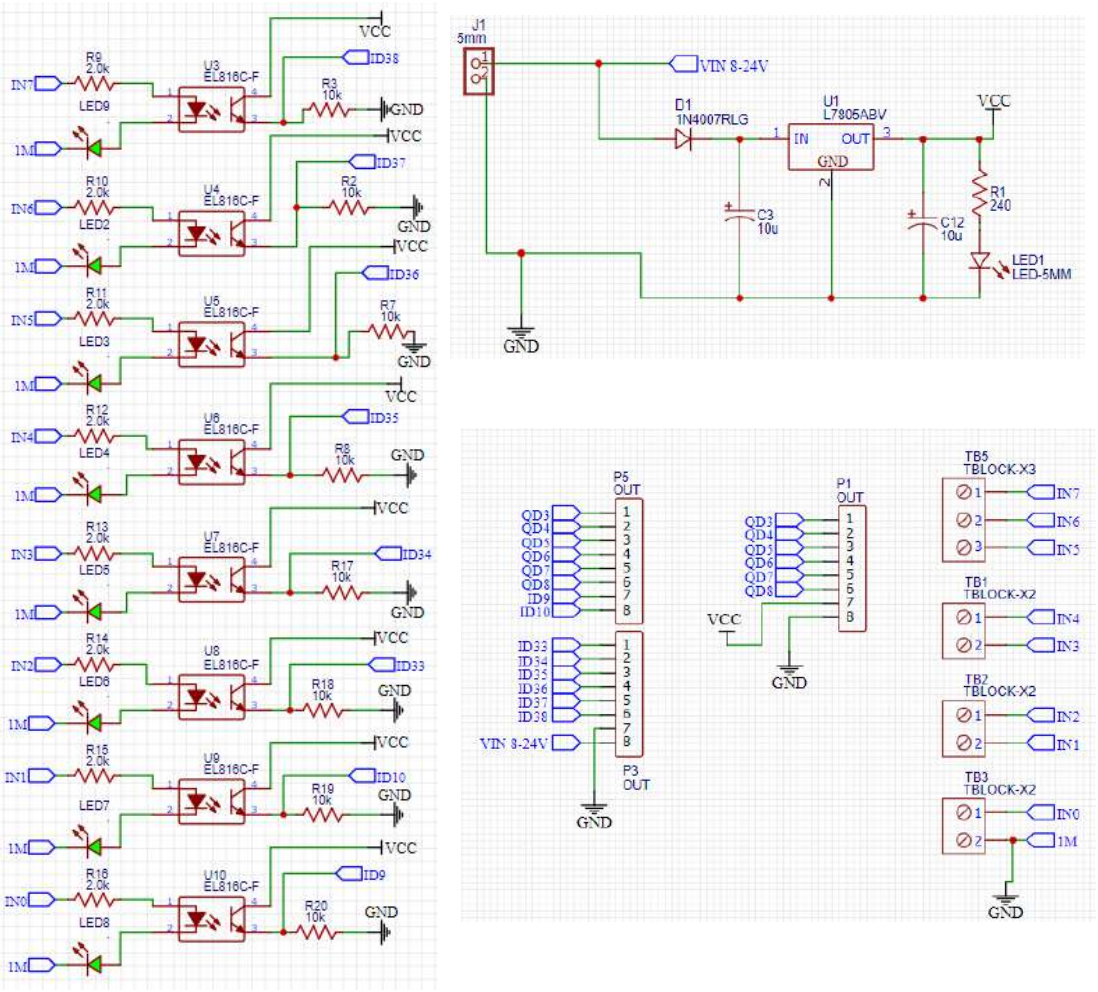


Fig. 16: Esquemático del módulo de entradas digitales.

Descripción del módulo de salidas digitales

El módulo de salidas digitales se diseñó para que pudiera manejar señales de corriente alterna y corriente directa, con cargas máximas de 10 A/120VAC y 10A /24V DC, para cumplir con estos requisitos se optó por una interfaz de acople de salidas usando relés electromecánicos. El diseño resultante se muestra en la figura 17.

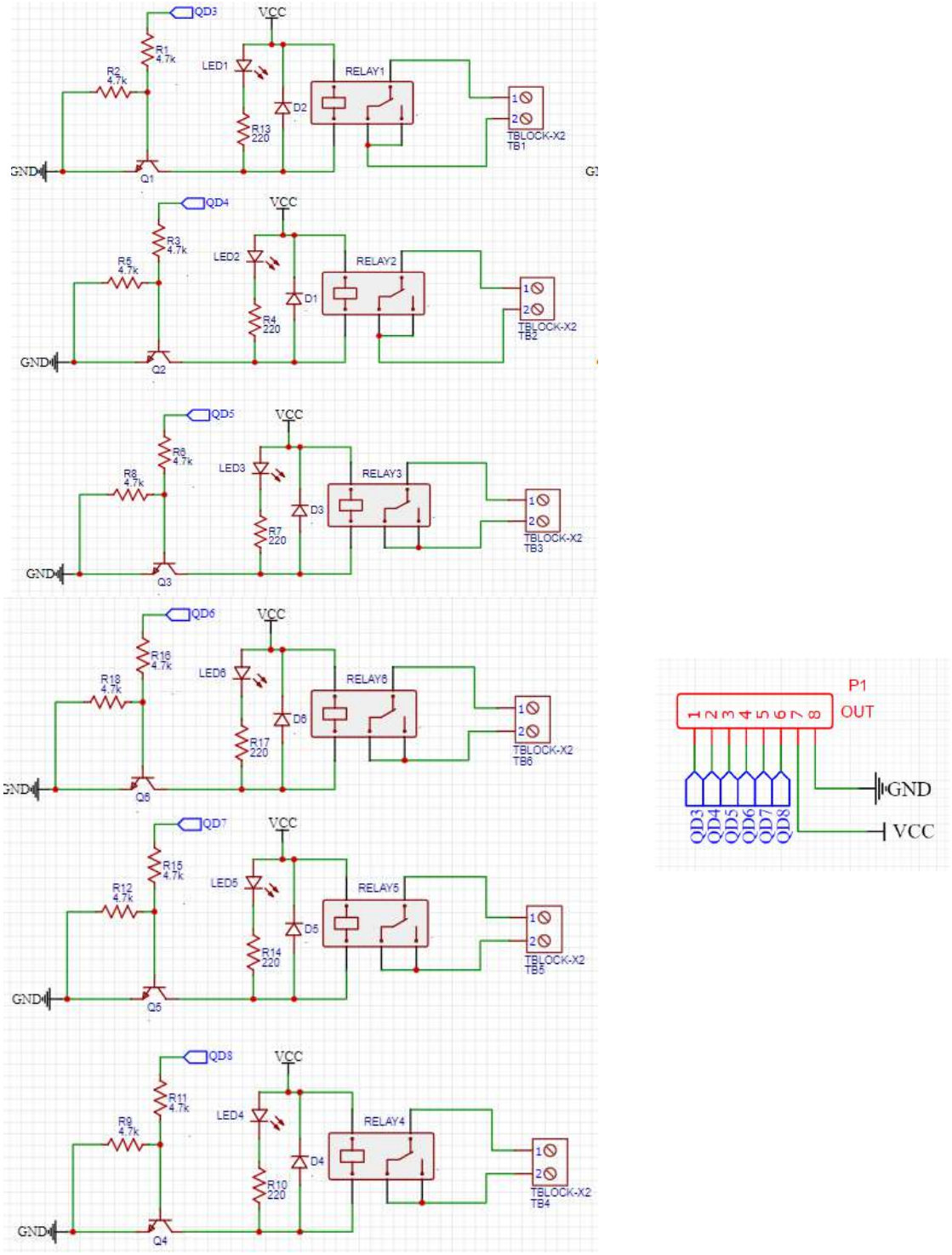


Fig.17. Esquemático del módulo de salidas digitales.

7.2. Diseño del Firmware

El firmware es el programa básico que se encarga de controlar lo que tiene que hacer el hardware que conforma la RTU, además tiene que asegurar que el funcionamiento básico sea el correcto. Para lograr establecer la comunicación con un PLC 1200 se programaron dos firmwares, uno para la pasarela que convierte los mensajes enviados del estándar Zigbee al protocolo MODBUS TCP y otro que se encarga de gestionar las entradas y salidas de la RTU.

Firmware de prueba para establecer la comunicación con el PLC 1200 mediante MODBUS TCP

```
#include <SoftwareSerial.h>
#include <ModbusIP.h>
# include <Modbus.h>
# include <Ethernet.h>
# include <SPI.h>
const int SENSOR_IREG = 100;
const int SENSOR_IREG_DOS = 200;
SoftwareSerial Xbee(2,3) ;        //2 RX, 3 TX
String comando=" ";
////////////////////////////////////////
void LEER_PUERTO_SERIAL (){
  if (Serial.available () > 0)
  {
 comando = "“;          //vacío lo leído
    do {
      char caracter_leido;
      delay(5);
      caracter_leido = Serial.read();
    comando += caracter_leido;
    } while (Serial.available() > 0);

}
////////////////////////////////////////
```

```
if (Xbee.available () > 0)
 {
 comando = ""; //vacío lo leído
   do {
     char caracter_leido;
     delay(5);
     caracter_leido = Xbee.read();
   comando += caracter_leido;
   }  while (Xbee.available() > 0);
Serial.print("COMANDO= ");
Serial.println(comando);
}
 }
////////////////////////////////////////////////
char dato=' ';
int setPoint;
//const int setPoint_IR=100;
//Modbus Registers Offsets (0-9999)
const int LAMP1_COIL = 100;
const int LAMP2_COIL=102;
const int LAMP3_COIL=104;
const int Q1_COIL = 100;
const int Q2_COIL=102;
const int Q3_COIL=104;
//Pines usados
const int ledPin1 = 3;
const int ledPin2 = 4;
const int ledPin3 = 5;
ModbusIP mb;
```

```
//////////////////////////////////////
void DisplaySetPointValues(){
String Printstr;
Printstr="SetPoint Value="+ String(setPoint);
Serial.println(Printstr);
}
////////////////////////////////////////////////////////////////////////////////////////////
**************************************************************************
void setup() {
Serial.begin(9600);
Xbee.begin(9600);
byte mac[]={0xDE,0xAD,0xAE,0xEF,0xFD,0xED};
//byte mac[]={0xDE,0xAD,0xBE,0xEF,0xFE,0xED};
byte ip[]={192,168,140,101};
mb.config(mac,ip);
//mb.addIreg(setPoint_IR);
pinMode(ledPin1, OUTPUT);
pinMode(ledPin2, OUTPUT);
pinMode(ledPin3, OUTPUT);
// Add LAMP1_COIL register - Usar addCoil() salidas digitales
mb.addCoil(LAMP1_COIL);
mb.addCoil(LAMP2_COIL);
mb.addCoil(LAMP3_COIL);
mb.addCoil(Q1_COIL);
mb.addCoil(Q2_COIL);
mb.addCoil(Q3_COIL);
mb.addIreg(SENSOR_IREG);
mb.addIreg(SENSOR_IREG_DOS);
}
```

```
////////////////////////////////////////////////////////////////////////////////////////
**************************************************************************
void loop() {
 //if (Serial.available()>0){dato=Serial.read();}
 mb.task();
 LEER_PUERTO_SERIAL();
 //setPoint=(dato);
 //setPoint=analogRead(A0);
 //mb.Ireg(setPoint_IR,setPoint);
 //DisplaySetPointValues();
 //Attach ledPin to LAMP1_COIL register
  digitalWrite(ledPin1, mb.Coil(Q1_COIL));
 digitalWrite(ledPin2, mb.Coil(Q2_COIL));
  digitalWrite(ledPin3, mb.Coil(Q3_COIL));
 ////////////////////////////////////////////////////
  if (comando.startsWith( "I1.1") ){sensorPin=1;}
  if (comando.startsWith( "I1.0")){sensorPin=2;}
  if (comando.startsWith( "I2.1") ){sensorPin=3;}
  if (comando.startsWith( "I2.0")){sensorPin=4;}
  if (comando.startsWith( "I3.1") ){sensorPin=5;}
  if (comando.startsWith( "I3.0")){sensorPin=6;}
  if (comando.startsWith( "I4.1") ){sensorPin=7;}
  if (comando.startsWith( "I4.0")){sensorPin=8;}
  if (comando.startsWith( "I5.1") ){sensorPin=9;}
  if (comando.startsWith( "I5.0")){sensorPin=10;}
  if (comando.startsWith( "I6.1") ){sensorPin=11;}
  if (comando.startsWith( "I6.0")){sensorPin=12;}
  ////////////////////////////////////////////////
```

```
  mb.Ireg(SENSOR_IREG, sensorPin);//**********************************<<<<
   mb.Ireg(SENSOR_IREG_DOS,sensorPin);//**********************************<<
   if(mb.Coil(Q1_COIL)==1){Serial.println("Q1.1");}
   if(mb.Coil(Q1_COIL)==0){Serial.println("Q1.0");}
   if(mb.Coil(Q2_COIL)==1){Serial.println("Q2.1");}
   if(mb.Coil(Q2_COIL)==0){Serial.println("Q2.0");}
   if(mb.Coil(Q3_COIL)==1){Serial.println("Q3.1");}
   if(mb.Coil(Q3_COIL)==0){Serial.println("Q3.0");}
  delay(50);
}
```

Para que la RTU pueda establecer la comunicación con el PLC y manipular de forma remota las direcciones Q0.0, Q0.1, Q0.2, Q0.3, Q0.4 y Q0.5 del registro de memoria del PLC 1200 se realizó el programa en el entorno de desarrollo Tía Portal, estableciendo el PLC como un cliente MODBUS, el código desarrollado se muestra a continuación.

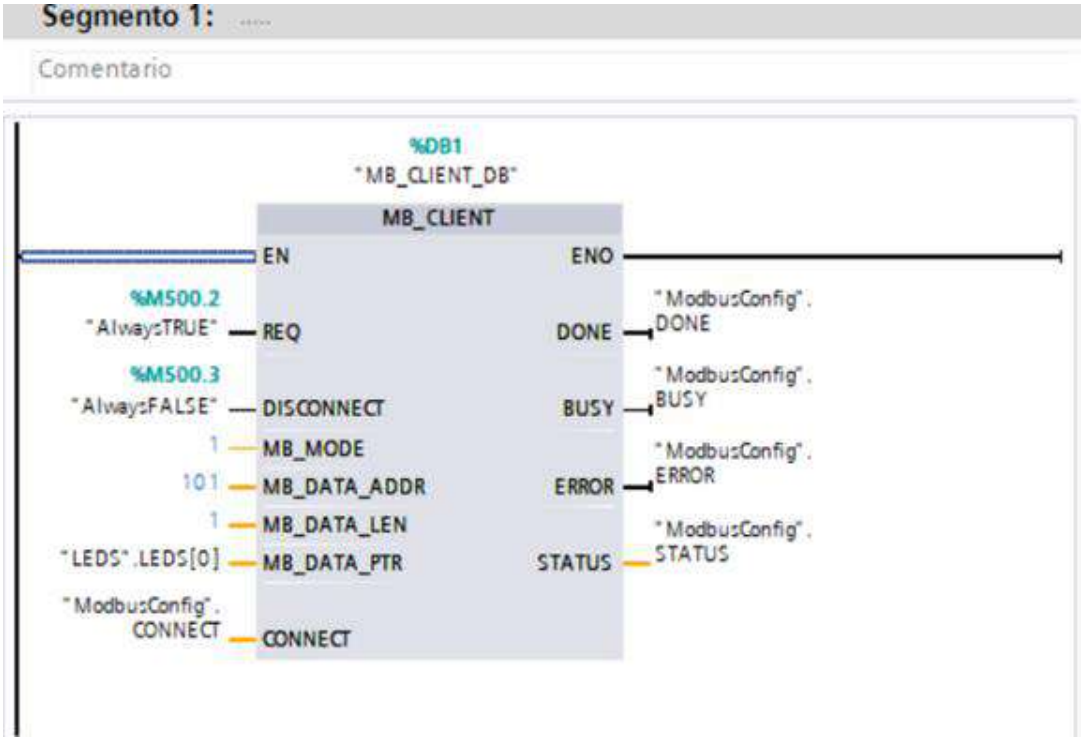


Fig. 18: Configuración de MB_CLIENT_DB para la dirección de datos 101

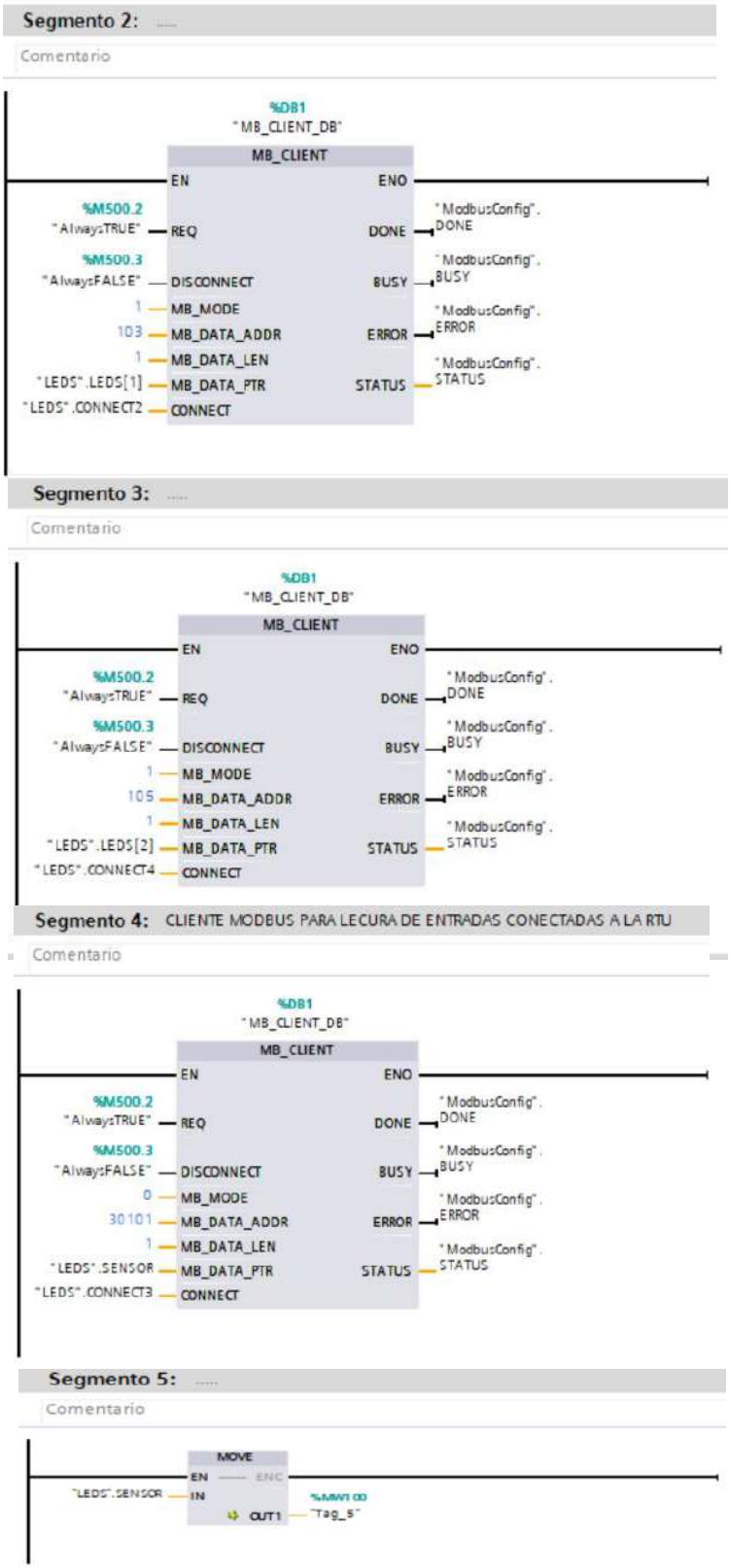


Fig. 19: Configuración de MB_CLIENT_DB para la dirección de datos 103,105 y 30101.

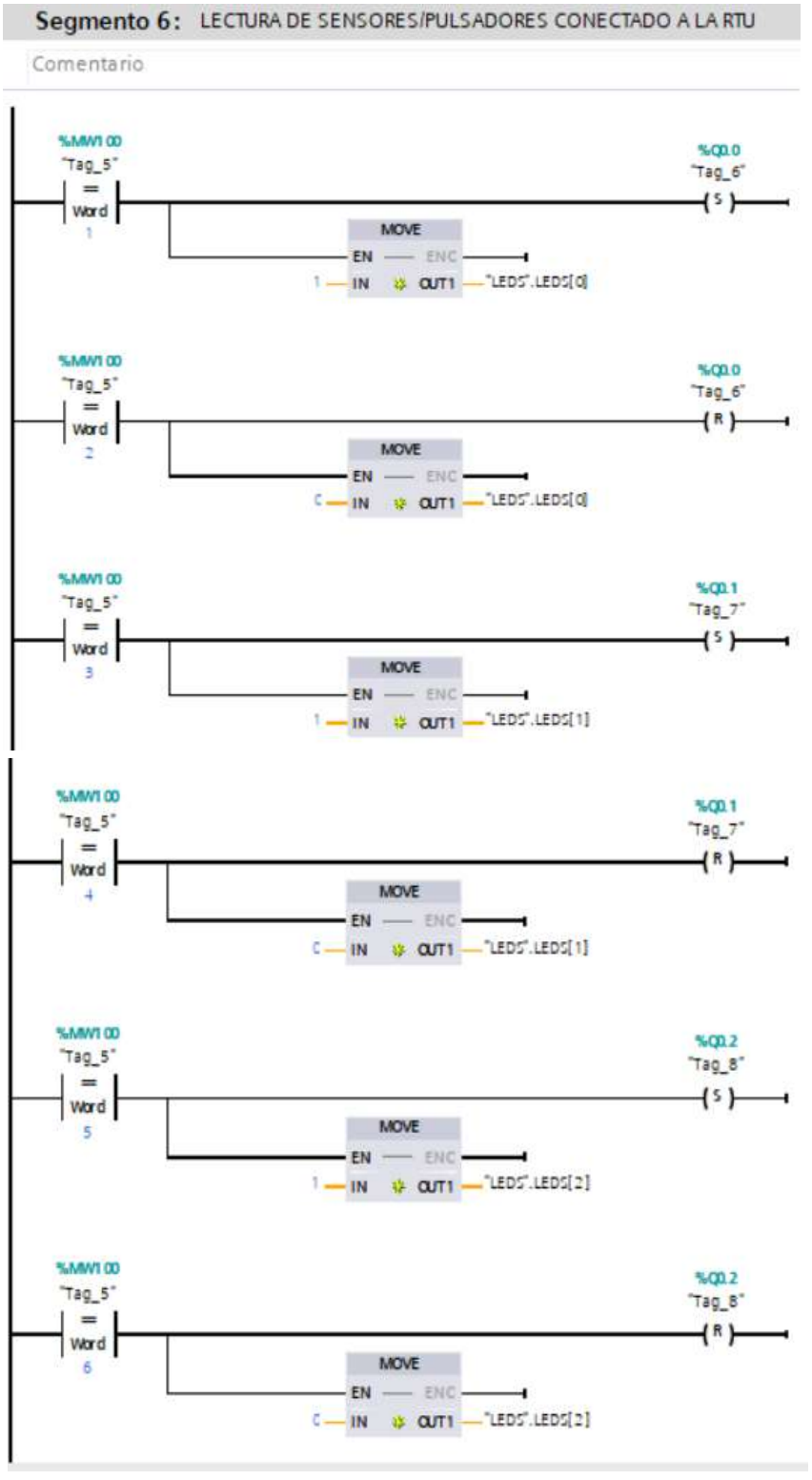


Fig. 20: Lectura de sensores/pulsadores conectados a la RTU con retroalimentación.

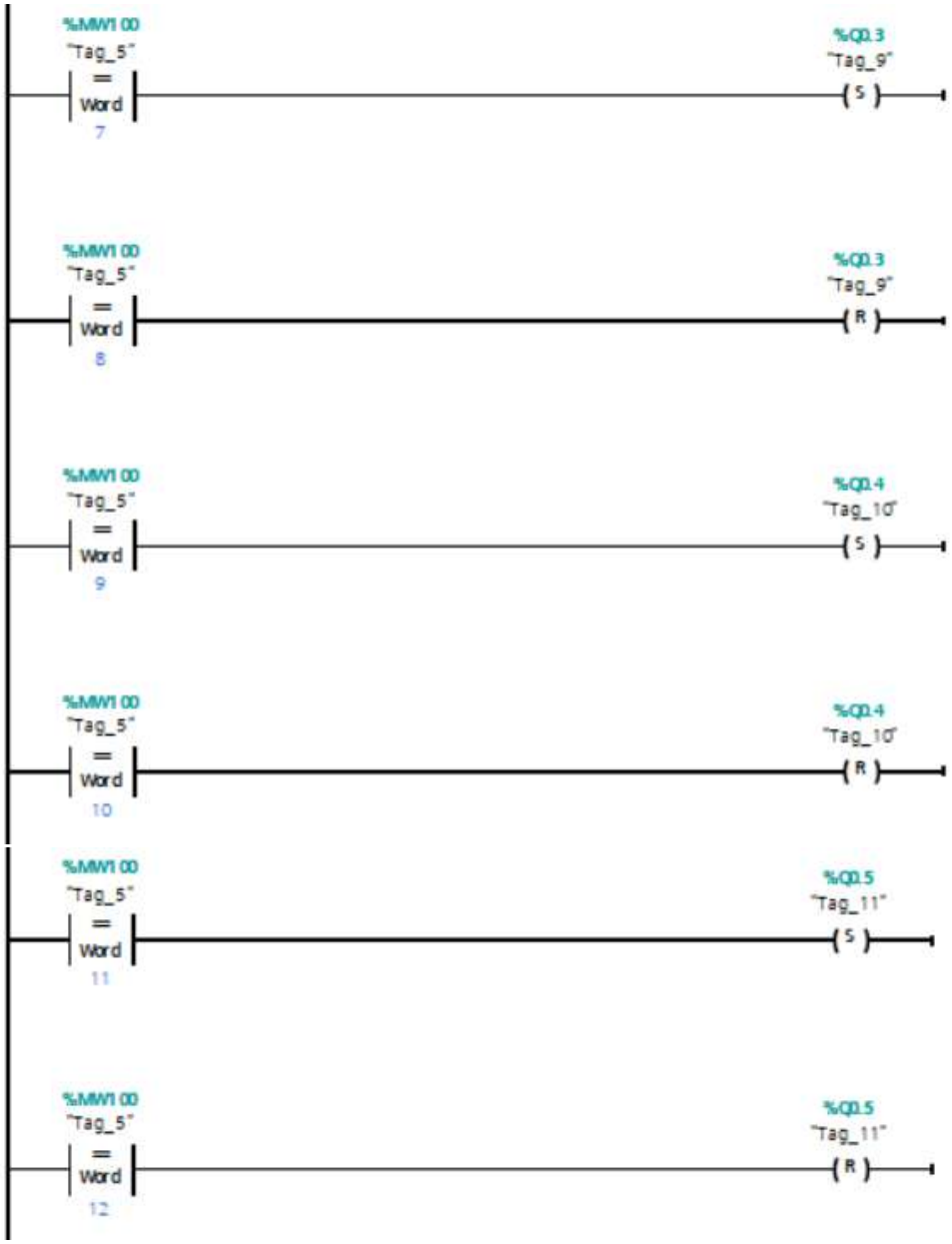


Fig. 21: Lectura de sensores/pulsadores conectados a la RTU sin retroalimentación.

El cliente MODBUS debe configurarse tomando de referencia los parámetros definidos por SIEMENS para el manejo del protocolo MODBUS, la descripción del MB_CLIENT y los parámetros de configuración se muestran a continuación.

Descripción MB_CLIENT

La instrucción "MB_CLIENT" permite la comunicación como cliente Modbus TCP a través de la conexión PROFINET. La instrucción "MB_CLIENT" permite establecer una conexión entre el cliente y el servidor, enviar órdenes Modbus y recibir respuestas, así como controlar la desconexión del cliente Modbus TCP.

Para el S7-1200 con versión de firmware V4.0 puede utilizarse la instrucción "MB_CLIENT" hasta la versión de librería V3.1. Con el S7-1200, a partir de la versión de firmware V4.1 y el S7-1500, puede utilizarse la instrucción "MB_CLIENT" de todas las versiones de librería. La conexión puede realizarse a través de la interfaz local de la CPU o CM/CP. Para utilizar esta instrucción no se requiere ningún módulo de hardware adicional.

Conexiones múltiples a cliente

Un cliente Modbus TCP puede admitir varias conexiones TCP (el número máximo de conexiones depende de la CPU utilizada). El total de conexiones de una CPU, incluidos los clientes Modbus TCP y los servidores, no debe exceder el número máximo de conexiones admitido. Las conexiones Modbus TCP también pueden ser utilizadas conjuntamente por instancias de "MB_CLIENT" y/o "MB_SERVER".

En algunas conexiones de cliente deben respetarse las siguientes reglas:

- Cada conexión "MB_CLIENT" debe utilizar un DB de instancia unívoco.
- Para cada conexión "MB_CLIENT" debe especificarse una dirección IP unívoca del servidor.
- Cada conexión "MB_CLIENT" requiere una ID de conexión unívoca.

Para cada DB de instancia de la instrucción debe utilizarse la correspondiente ID de conexión. Los ID de conexión y los DB de instancia se agrupan por pares y deben ser unívocos para cada conexión. Según la configuración del servidor, se requerirán o no números unívocos de puerto IP.

Parámetros

La tabla siguiente muestra los parámetros de la instrucción "MB_CLIENT":

Tabla 5: Parámetros de la instrucción "MB_CLIENT"

Descripción MB_CLIENT

REQ	Input	BOOL	Orden Modbus al servidor TCP Modbus El parámetro REQ se controla por nivel. Así, mientras la entrada esté activada (REQ=true), la instrucción enviará órdenes de comunicación. • Al iniciar la orden Modbus se bloquea el DB de instancia para otros clientes. • Las modificaciones de los parámetros de entrada no se hacen efectivas hasta que no hay respuesta del servidor o hasta que no se devuelve un mensaje de error. • Si durante una orden Modbus en curso se vuelve a activar el parámetro REQ, a continuación no se ejecuta ninguna otra transferencia.
DISCONNECT	Input	BOOL	Mediante este parámetro se controla el establecimiento de la conexión y la desconexión con el servidor Modbus: • 0: Establecer conexión de comunicación con el interlocutor configurado en el parámetro CONNECT (ver parámetro CONNECT). • 1: Deshacer la conexión. Durante la desconexión no se ejecuta ninguna otra función. Tras deshacer la conexión correctamente, el parámetro STATUS devuelve el valor 0003. Si el parámetro REQ está activado mientras se establece la conexión, la orden Modbus se envía de inmediato.
MB_MODE	Input	USINT	Selección del modo de orden Modbus (lectura, escritura o diagnóstico) o selección directa de una función Modbus.
MB_DATA_ADDR	Input	UDINT	en función de MB_MODE
MB_DATA_LEN	Input	UINT	Longitud de datos: Número de bits o palabras para el acceso a los datos (ver Parámetros MB_MODE, MB_DATA_ADDR y MB_DATA_LEN).
MB_DATA_PTR	InOut	VARIANT	Puntero hacia un búfer de datos para los datos que se van a recibir desde el servidor Modbus o que se van a enviar al servidor Modbus.

CONNECT	InOut	VARIANT	Puntero hacia la estructura de la descripción de la conexión Se pueden utilizar las siguientes estructuras (tipos de datos de sistema): • TCON_IP_v4: contiene todos los parámetros de direccionamiento necesarios para establecer una conexión programada. Si se utiliza TCON_IP_v4, la conexión se establece al llamar la instrucción "MB_CLIENT". • TCON_Configured: contiene los parámetros de direccionamiento de una conexión configurada. En el caso de utilizar TCON_Configured se emplea una conexión existente, establecida después de que la CPU cargara la configuración hardware.
DONE	Out	BOOL	El bit del parámetro de salida DONE se pone a "1" en cuanto se ejecuta sin errores la última orden Modbus.
BUSY	Out	BOOL	• 0: Ninguna orden Modbus en proceso • 1: La orden Modbus se está ejecutando El parámetro de salida BUSY no se activa al establecer o al deshacer la conexión.
ERROR	Out	BOOL	• 0: Ningún error • 1: Con errores. La causa del error se indica mediante el parámetro STATUS.
STATUS	Out	WORD	Información de estado detallada de la instrucción.

La configuración del MB CLIENT para establecer la comunicación mediante MODBUS entre la RTU y el PLC 1200 se muestra en la tabla 6.

Tabla 6: Variables de configuración de MB- CLIENT.

ModbusConfig

Nombre	Tipo de datos	Valor de arranq...	Remanen...	Accesible d...	Escrib...	Visible en ...	Valor de a...
Static			☐	☐	☐	☐	☐
REQ	Bool	false	☐	☑	☑	☑	☐
DISCONNECT	Bool	false	☐	☑	☑	☑	☐
MB_MODE	USInt	0	☐	☑	☑	☑	☐
MB_DATA_ADDR	UDInt	101	☐	☑	☑	☑	☐
MB_DATA_LEN	UInt	1	☐	☑	☑	☑	☐
CONNECT	TCON_IP_v4		☑	☑	☑	☑	☑
InterfaceId	HW_ANY	64	☑	☑	☑	☑	☐
ID	CONN_OUC	1	☑	☑	☑	☑	☐
ConnectionType	Byte	16#0B	☑	☑	☑	☑	☐
ActiveEstablished	Bool	true	☑	☑	☑	☑	☐
RemoteAddress	IP_V4		☑	☑	☑	☑	☑
RemotePort	UInt	502	☑	☑	☑	☑	☐
LocalPort	UInt	0	☑	☑	☑	☑	☐
DONE	Bool	false	☐	☑	☑	☑	☐
BUSY	Bool	false	☐	☑	☑	☑	☐
ERROR	Bool	false	☐	☑	☑	☑	☐
STATUS	Word	16#0	☐	☑	☑	☑	☐

Firmware para la gestión de entradas y salidas en la RTU

Con el propósito de gestionar las entradas y las salidas de la RTU se diseñó el siguiente firmware:

```
#define  Xbee Serial1
# define Q1 3 //**************
# define Q2 4
# define Q3 5 // SALIDAS *****
# define Q4 6
# define Q5 7
# define Q6 8//***************
////////////////////////////////////
# define I1 9 //*********************
# define I3 10
# define I2 33
# define I5 34 // ENTRADAS**********
# define I4 35
# define I8 36
# define I7 37
```

```
digitalWrite(Q1,0);
  digitalWrite(Q2,0);
  digitalWrite(Q3,0);
  digitalWrite(Q4,0);
  digitalWrite(Q5,0);
  digitalWrite(Q6,0);
}
////////////////////////////////////
void CONTROL_DE_CARGAS(){
 if (comando.startsWith( Q1_ON)) {digitalWrite(Q1,1);}
 if (comando.startsWith( Q1_OFF)){digitalWrite(Q1,0);}

 if (comando.startsWith( Q2_ON)) {digitalWrite(Q2,1);}
 if (comando.startsWith( Q2_OFF)){digitalWrite(Q2,0);}

 if (comando.startsWith( Q3_ON)) {digitalWrite(Q3,1);}
 if (comando.startsWith( Q3_OFF)){digitalWrite(Q3,0);}

 if (comando.startsWith( Q4_ON)) {digitalWrite(Q4,1);}
 if (comando.startsWith( Q4_OFF)){digitalWrite(Q4,0);}

 if (comando.startsWith( Q5_ON)) {digitalWrite(Q5,1);}
 if (comando.startsWith( Q5_OFF)){digitalWrite(Q5,0);}

 if (comando.startsWith( Q6_ON)) {digitalWrite(Q6,1);}
 if (comando.startsWith( Q6_OFF)){digitalWrite(Q6,0);}
}
```

```
# define I6 38//********************
////////////////////////////////////
# define Q1_ON   "Q1.1"
# define Q1_OFF  "Q1.0"
# define Q2_ON   "Q2.1"
# define Q2_OFF  "Q2.0"
# define Q3_ON   "Q3.1"
# define Q3_OFF  "Q3.0"
# define Q4_ON   "Q4.1"
# define Q4_OFF  "Q4.0"
# define Q5_ON   "Q5.1"
# define Q5_OFF  "Q5.0"
# define Q6_ON   "Q6.1"
# define Q6_OFF  "Q6.0"
String comando=" ";
byte valor_de_entradas_0=0;
byte valor_de_entradas_1=0;
////////////////////////////////////
void Init_puerto_digital(){
  pinMode(Q1,OUTPUT);
  pinMode(Q2,OUTPUT);
  pinMode(Q3,OUTPUT);
  pinMode(Q4,OUTPUT);
  pinMode(Q5,OUTPUT);
  pinMode(Q6,OUTPUT);
  pinMode(13,OUTPUT);
  pinMode(I1,INPUT);
  pinMode(I2,INPUT);
  pinMode(I3,INPUT);
  pinMode(I4,INPUT);
  pinMode(I5,INPUT);
  pinMode(I6,INPUT);
  pinMode(I7,INPUT);
  pinMode(I8,INPUT);
```

```
//////////////////////////////////////////////////////
 void LEER_PUERTO_SERIAL(){
  if (Serial.available () > 0)
  {
 comando = ""; //vacío lo leído
    do {
      char caracter_leido;
      delay(5);
      caracter_leido = Serial.read();
    comando += caracter_leido;
    }  while (Serial.available() > 0);

}

if (Xbee.available () > 0)
  {
 comando = ""; //vacío lo leído
    do {
      char caracter_leido;
      delay(5);
      caracter_leido = Xbee.read();
    comando += caracter_leido;
    }  while (Xbee.available() > 0);
Serial.print("COMANDO= ");
Serial.println(comando);
}
  }
```

```
byte LECTURA_DE_ENTRADAS(){
  boolean dato=0;
  byte valor_de_entradas=0;
  boolean sensor[]={0,0,0,0,0,0,0,0};
  int entradas[]={I8,I7,I6,I5,I4,I3,I2,I1};
  for (int i=0;i<8;i++){
    dato=digitalRead(entradas[i]);
    sensor[i]=dato;
                  }
     valor_de_entradas= (sensor[0]*128+sensor[1]*64+sensor[2]*32+sensor[3]*16+
                      sensor[4]*8+sensor[5]*4+sensor[6]*2+sensor[7]*1);
     return valor_de_entradas;
                    }
////////////////////////////////////////////////////////////////////////////////////////////
void setup() {
  Init_puerto_digital();// configuración del puerto de señales digitales
  Serial.begin(115200);
  Xbee.begin(9600);
  Serial.println("{\"RTU_id\":\"RTU1\",\"Value\":" + (String)LECTURA_DE_ENTRADAS() + "}");
  Xbee.println("{\"RTU_id\":\"RTU1\",\"Value\":" + (String)LECTURA_DE_ENTRADAS() + "}");
}
////////////////////////////////////////////////////////////////////////////////////////////
void loop() {
CONTROL_DE_CARGAS();
LEER_PUERTO_SERIAL() ;
valor_de_entradas_1=LECTURA_DE_ENTRADAS();
String json_data1 = "{\"RTU_id\":\"RTU1\",\"Value\":" + (String)valor_de_entradas_1 + "}";
if(valor_de_entradas_1!=valor_de_entradas_0){
                                Serial.println(json_data1);
                                Xbee.println(json_data1);
                                valor_de_entradas_0=valor_de_entradas_1;
                                delay(250);
                                }        }
```

7.3. Diseño de Aplicaciones

Aplicación para medir la latencia de la red de comunicaciones.

Considerando que la latencia es el tiempo de retraso en el proceso de envío y recepción de datos a través de una red, con el propósito de determinar la viabilidad de utilizar la RTU en aplicaciones donde el tiempo de respuestas en las comunicaciones es crítico. Se realizaron dos experimentos: el primer experimento consistió en conectar el módulo RTU a una red LAN en las instalaciones de ITCA-FEPADE y el segundo se realizó con una conexión remota usando un VPN y el IXROUTER de la marca IXON. Para realizar las pruebas de latencia se diseñó una aplicación con el entorno de desarrollo LabVIEW (fig.22), utilizando el protocolo de comunicación MODBUS TCP.

Fig. 22. Aplicación para medir la latencia de red

Resultado de la prueba de latencia

Los resultados de los experimentos fueron: para una conexión de la RTU a una red local (LAN), se obtuvo una latencia promedio de 52.9 milisegundos y para la conexión remota, usando una VPN IXON, la latencia promedio fue de 994.0 milisegundos. Los datos del muestreo se presentan en las siguientes tablas:

Tabla 7: Latencias para conexión en una red LAN

PRUEBAS DE LATENCIA CONECTADO DESDE RED INTERNA DE ITCA		
FECHA DE LA PRUEBA: 10/11/2023		
#	HORA	LATENCIA REGISTRADA (ms)
1	10:45	53
2	11:00	53
3	11:15	54
4	11:30	52
5	11:45	53
6	12:00	54
7	01:00	55
8	01:15	51
9	01:30	52
10	01:45	54
11	02:00	54
12	02:15	53
13	02:30	52
14	02:45	50
15	03:00	51
16	03:15	54
17	03:30	52
18	03:35	49
19	03:45	58
20	04:00	54
	PROMEDIO:	**52.9**

Tabla 8: Latencias para conexión remota usando una VPN

PRUEBAS DE LATENCIA CONECTADO DESDE RED EXTERNA DE ITCA		
FECHA DE LA PRUEBA: 10/11/2023		
#	HORA	LATENCIA REGISTRADA (ms)
1	10:45	1146
2	11:00	865
3	11:15	910
4	11:30	878
5	11:45	972
6	12:00	908
7	01:00	1063
8	01:15	977
9	01:30	1001
10	01:45	893
11	02:00	893
12	02:15	938
13	02:30	954
14	02:45	908
15	03:00	887
16	03:15	912
17	03:30	1003
18	03:35	1573
19	03:45	1101
20	04:00	1097
	PROMEDIO:	**994.0**

Aplicación para comunicación remota vía VPN usando el protocolo MODBUS TCP

Para verificar si era posible manipular un cilindro neumático de doble efecto, usando el módulo electroneumático conectado a la RTU, se diseñó un experimento en el laboratorio el cual se describe a continuación.

Descripción general

El experimento consistió en diseñar una aplicación en LabVIEW que permita manipular un cilindro neumático de doble efecto desde cualquier parte del mundo. Al presionar un botón virtual (ON) sale el vástago del cilindro y al presionar otro botón (OFF) el vástago del cilindro se retrae.

La solución al problema planteado se dividió en las siguientes partes:

A) Diseño de la red de comunicaciones.
B) Diseño del diagrama de mando.
C) Diseño del diagrama de fuerza.
D) Diseño de la aplicación en LabVIEW.
E) Diseño del programa para el PLC 1200
F) Montaje de los componentes.

Los resultados del experimento se muestran a continuación.

Diseño de la red de comunicaciones

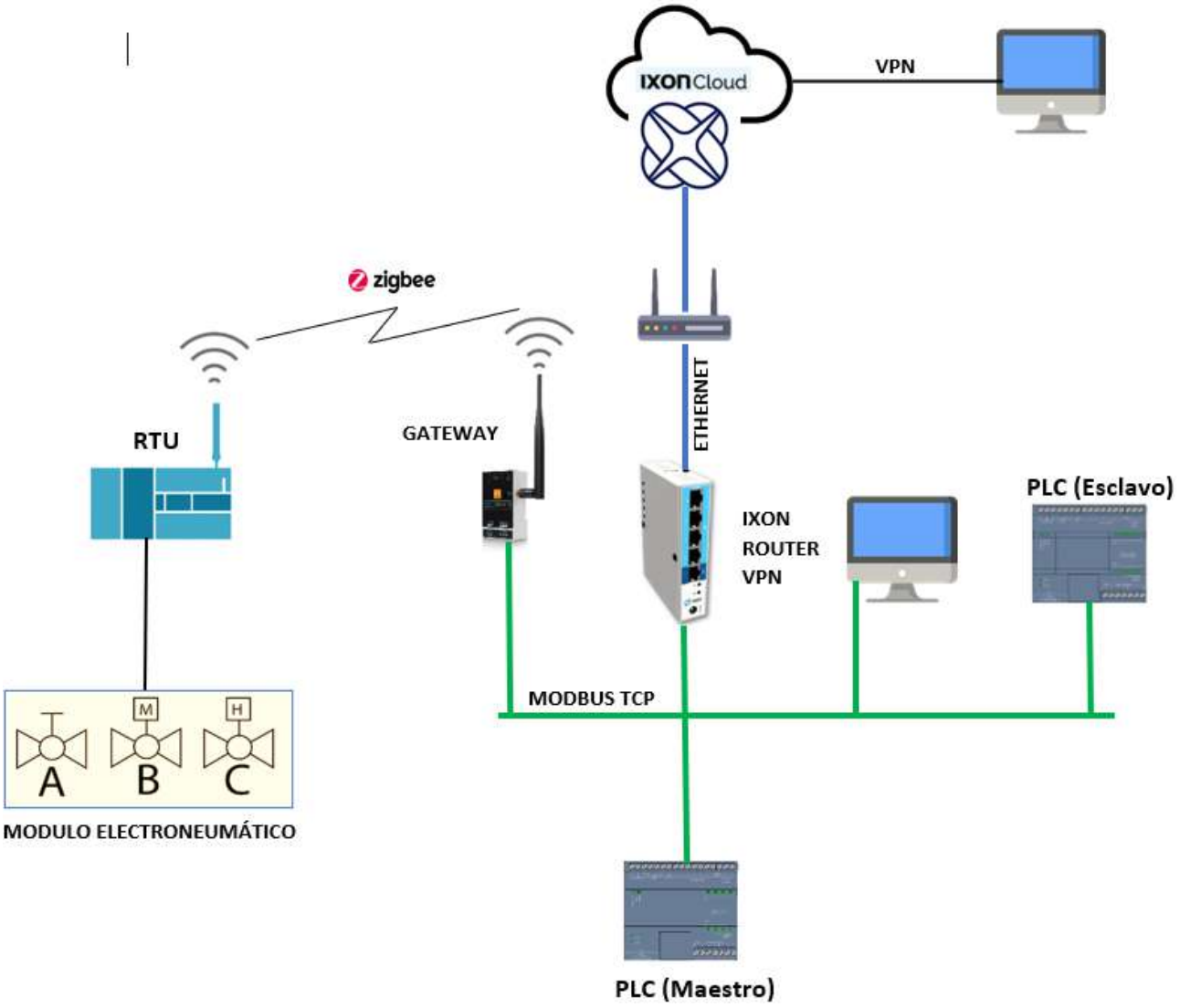


Fig. 23: Diseño de red para el envío y recepción de datos

Diseño del diagrama de mando

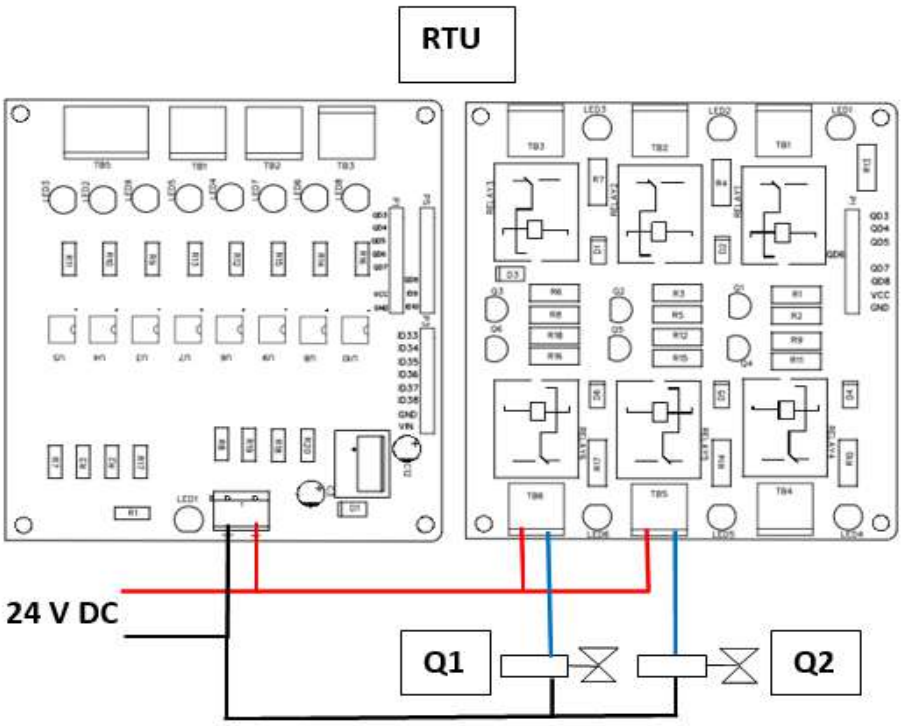


Fig. 24: Diagrama de conexión de las electroválvulas con la RTU

Diseño del diagrama de fuerza

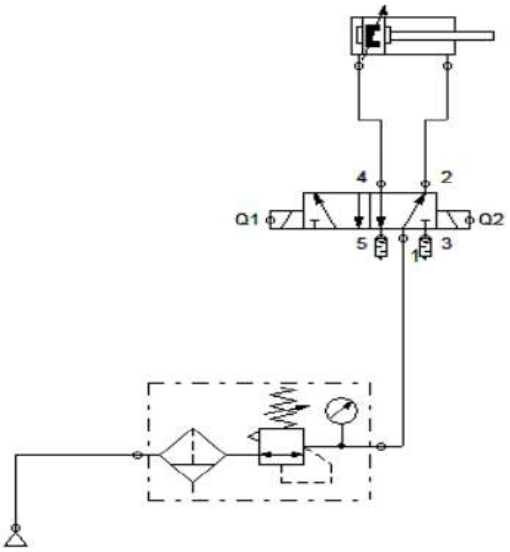


Fig. 25:Circuito para el control de un cilindro de doble efecto

Diseño de la aplicación en LabVIEW

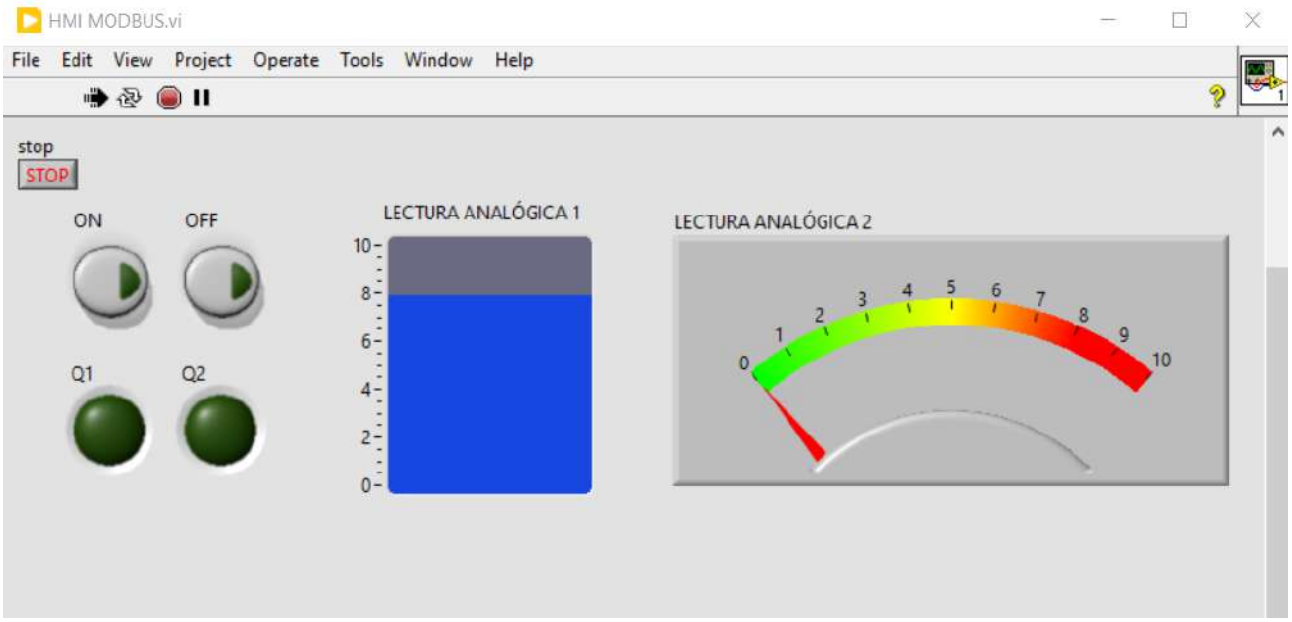


Fig. 26 Interfaz para control y monitoreo remoto.

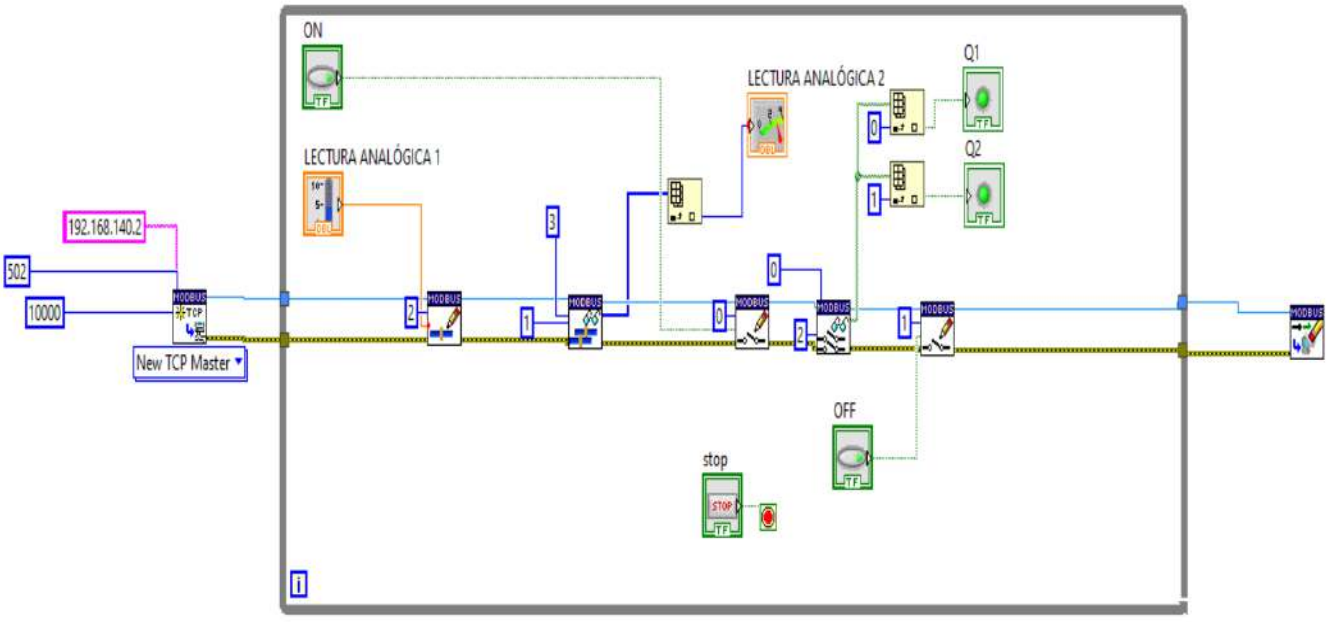


Fig. 27. Código de la interfaz de control y monitoreo remoto.

Diseño del programa para el PLC S7 1200

Usando el protocolo MODBUS el PLC se configura como servidor.

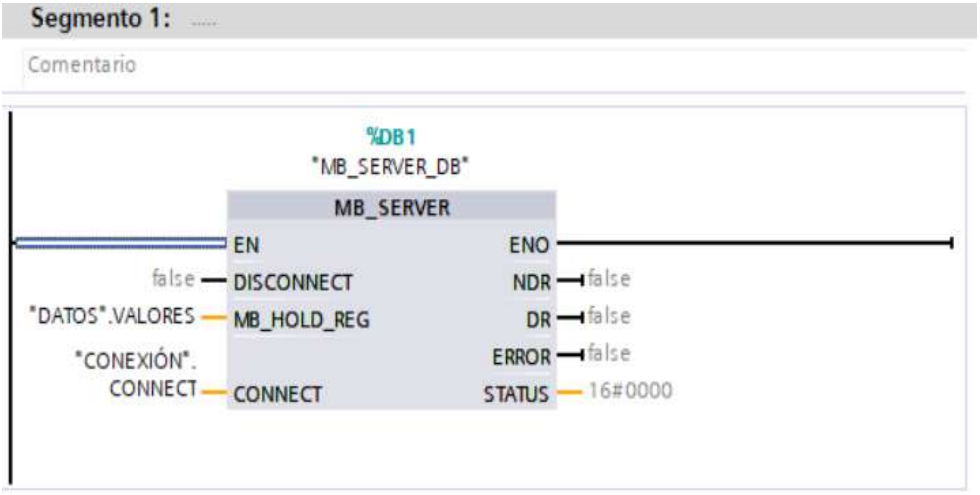


Fig. 28: Configuración del MB_SERVER_DB

Se programaron los siguientes segmentos para probar el envío y recepción de valores numéricos, que representan fuentes de señales analógicas

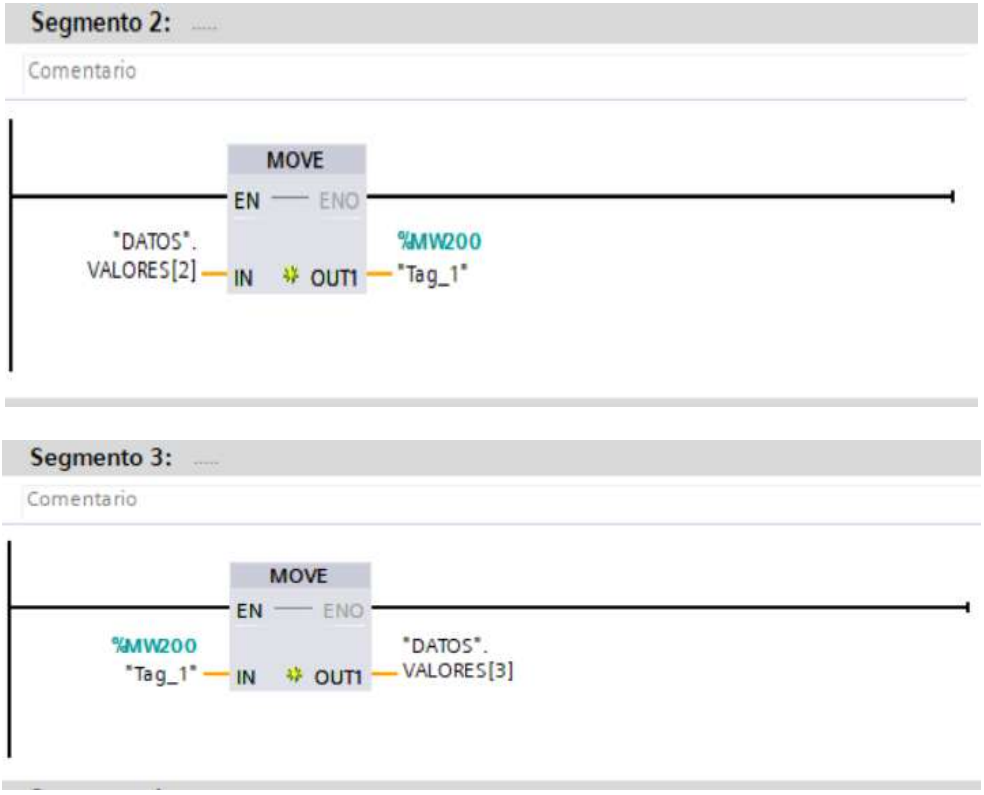


Fig. 29: Registro de valores numéricos.

Montaje de los componentes

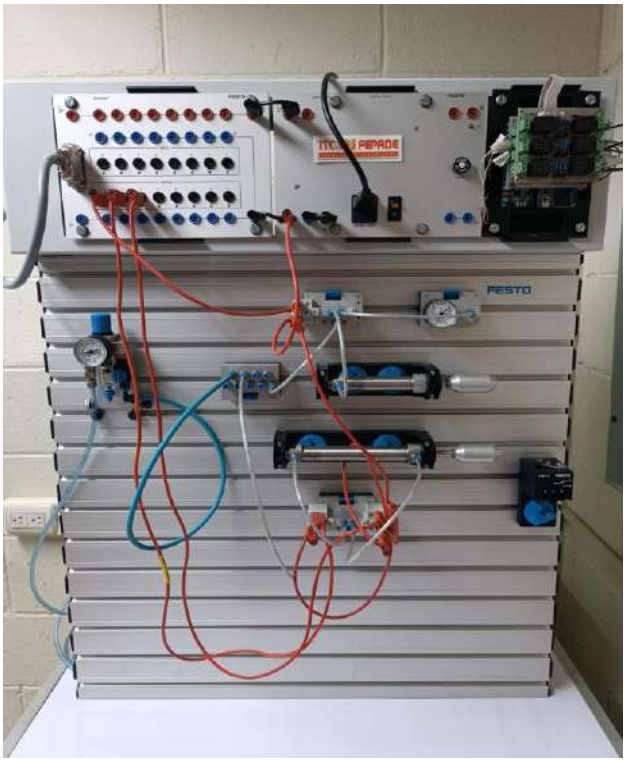


Fig. 30: Vista del montaje del circuito electroneumático

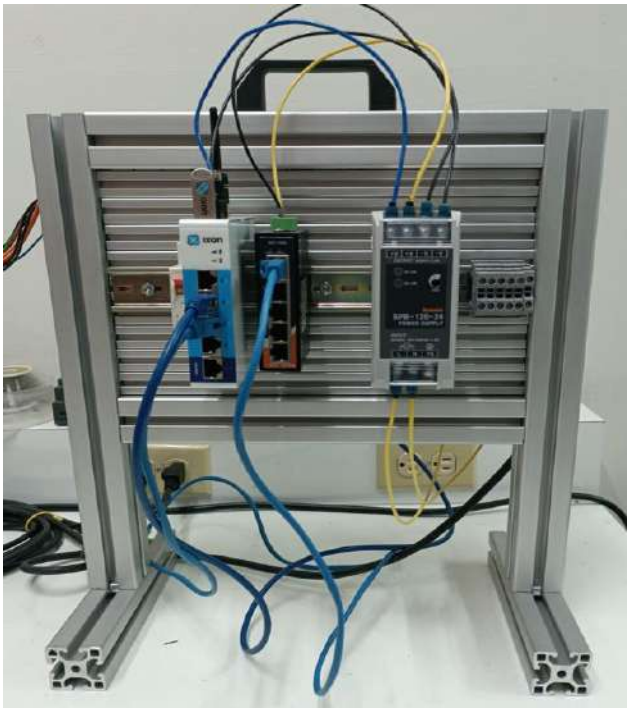


Fig. 31: Montaje de red de comunicación

8. CONCLUSIONES

1. Con el desarrollo de la investigación queda demostrado que la fiabilidad en la comunicación entre un PLC S7 1200 de Siemens y una RTU construida con hardware abierto se logra si se usa un protocolo de comunicación industrial como el MODBUS TCP/IP.
2. Para aplicaciones donde el tiempo de respuesta es crítico, hay que considerar la latencia en la comunicación, usando diferentes configuraciones de red. En este proyecto, de forma experimental, se obtuvo una latencia de 52.9 milisegundos con conectividad local, y 994 milisegundos con conectividad remota. Por lo tanto, se puede concluir que en aplicaciones donde el tiempo de respuesta sea crítico, es recomendable usar la RTU en una red local. Para aplicaciones donde la latencia en las comunicaciones no es crítica, la RTU se puede usar con conectividad remota mediante un VPN.

9. RECOMENDACIONES

1. Evaluar diferentes configuraciones de RED para reducir la latencia en el proceso de comunicación.
2. En futuras actualizaciones de la RTU, incorporar un módulo de entradas y salidas analógicas; ya que con el diseño actual únicamente se pueden procesar señales digitales.

10. GLOSARIO

- **GATEWAY.** Un Gateway (puerta de enlace) es un dispositivo (con frecuencia un computador) que permite interconectar redes con protocolos y arquitecturas diferentes a todos los niveles de comunicación. Su propósito es traducir la información del protocolo utilizado en una red al protocolo usado en la red de destino. A diferencia del Gateway, que no recolecta datos históricos ni realiza reportes de excepción, la RTU almacena los registros horarios en una base de datos y envía reportes obligatorios a través de los canales de comunicación predeterminados.
- **LATENCIA.** En redes informáticas de datos, la latencia de red es la suma de retardos temporales dentro de una red. Un retardo es producido por la demora en la propagación y transmisión de paquetes dentro de la red.

- **MODBUS.** Es un protocolo de comunicación abierto, utilizado para transmitir información a través de redes en serie entre dispositivos electrónicos. El dispositivo que solicita la información se llama maestro Modbus y los dispositivos que suministran la información son los esclavos Modbus.
- **SISTEMAS ELECTRONEUMÁTICOS.** El término electroneumático se define a partir de las palabras electro que significa eléctrico y neumático que significa presión de aire. Por lo tanto, un sistema electroneumático es una integración de la electricidad y los componentes mecánicos con fuente de aire comprimido.
- **UNIDAD DE TRANSMISIÓN REMOTA RTU.** Es un dispositivo electrónico que permite controlar un determinado número de entradas/salidas y enviarlas a un sistema de control superior, generalmente un PLC, o bien directamente a un SCADA. Las funciones de una RTU varían entre obtener información de telemetría y/o alterar el estatus de las aplicaciones conectadas al sistema en base a los datos de entrada.

  Una RTU recolecta información directamente de los sensores, medidores y equipamiento de campo. Normalmente, están localizadas cerca de los procesos monitoreados y transfieren información a los sistemas de control; están diseñadas para operar en forma segura en ambientes hostiles, protegidas de la erosión, humedad, polvo y de otros contaminantes atmosféricos. Algunas aplicaciones requieren de RTU redundantes que permiten la continuidad del servicio, aun cuando las unidades primarias dejen de funcionar.

11. REFERENCIAS BIBLIOGRÁFICAS

[1] Montalvo Luis. Diseño y construcción de unidades terminales remotas (RTU) de bajo costo para sistemas de control, supervisión y adquisición de datos (SCADA): Conference Paper, julio 1991

[2] MODBUS Organization, MODBUS Application Protocol Specification V1.1b3 [Consulta 2017].

[3] National Instruments, Información Detallada sobre el Protocolo Modbus < http://www.ni.com/white-paper/52134/es/> [Consulta 2017].

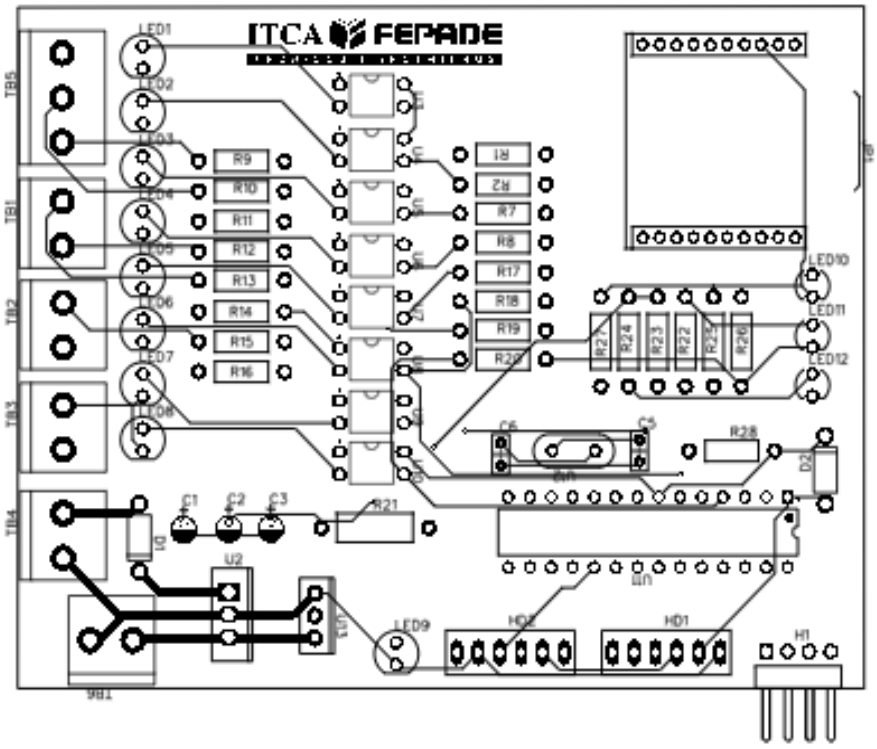
[4] MODBUS Organization, MODBUS over Serial Line Specification and Implementation Guide V1.02 [Consulta 2017].

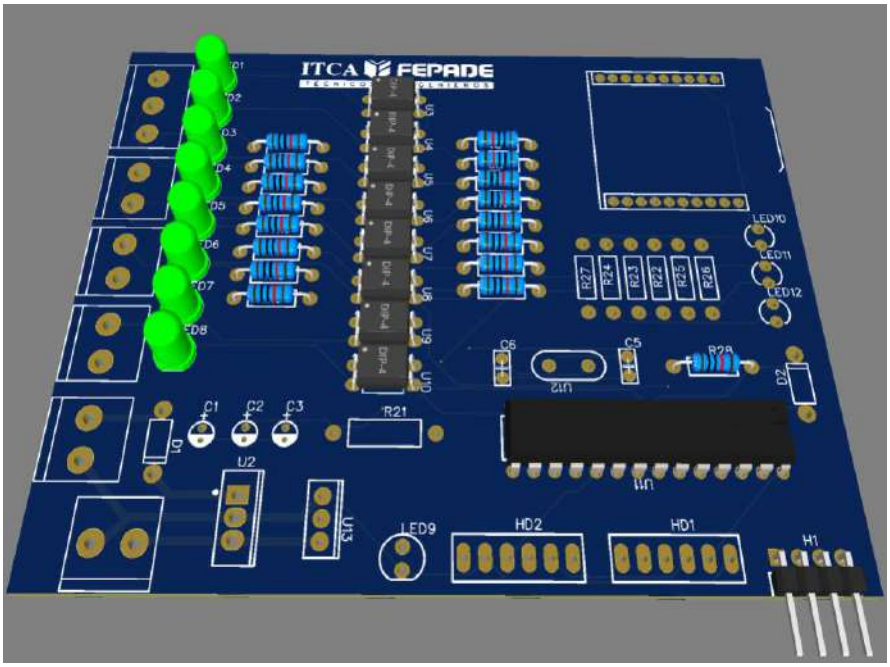
[5] Facultad de Ingeniería, Universidad Nacional de Cuyo: Electrónica general y aplicada, Carpeta de trabajos prácticos, 2016- TP N° 14.

[6] Rabadán Barastegui, J.J. (2017). Diseño y desarrollo de una red MODBUS RTU basada en Arduino. (Trabajo Fin de Grado Inédito). Universidad de Sevilla, Sevilla.

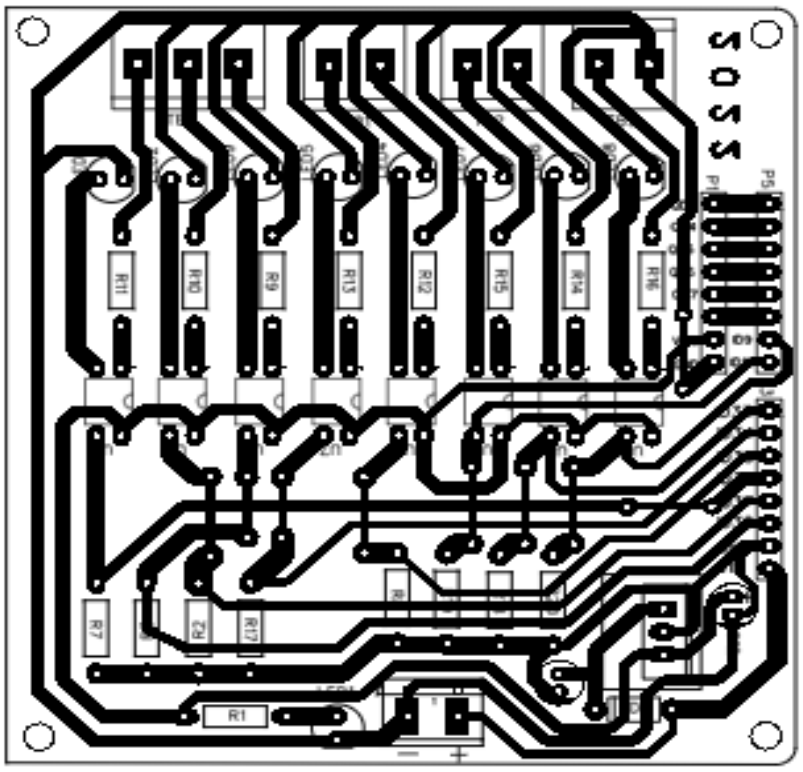
12. ANEXOS

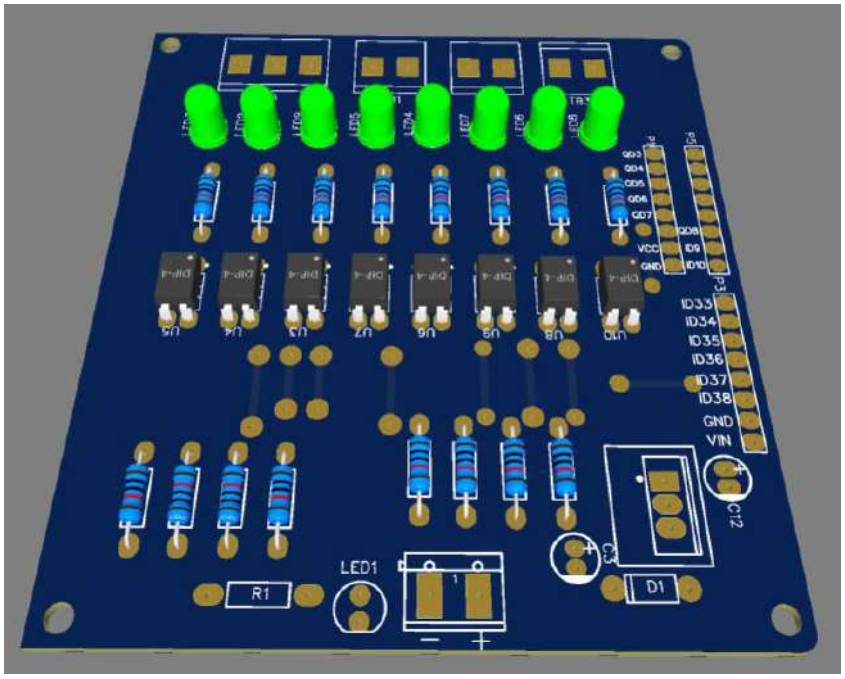
12.1. ANEXO 1. MÓDULO DE ENTRADAS DE LA RTU VERSIÓN 1



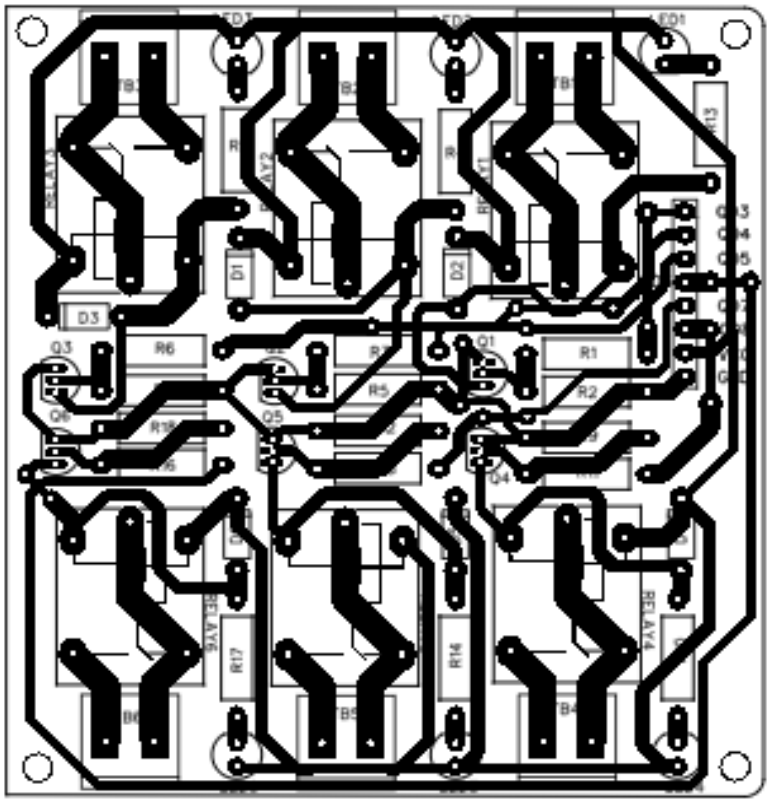


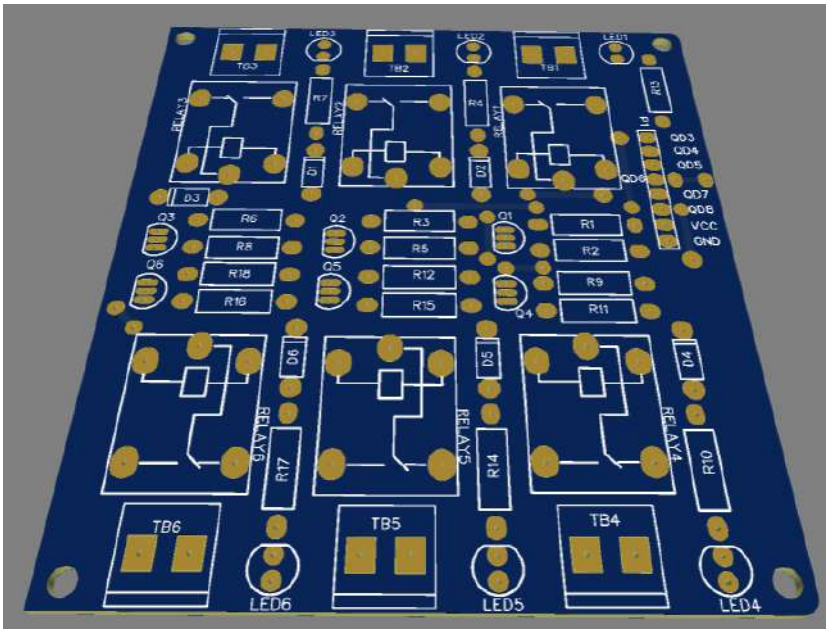
12.2. ANEXO 2. MÓDULO DE ENTRADAS DE LA RTU VERSIÓN 2



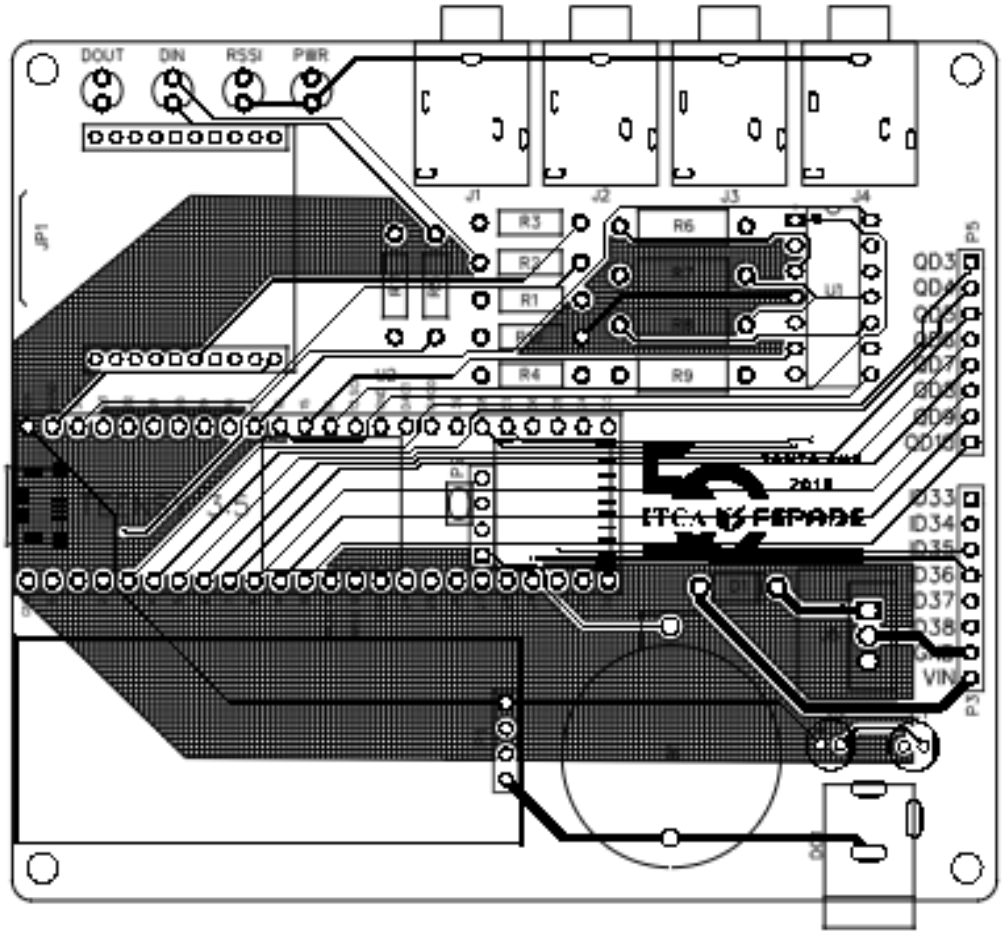


12.3. ANEXO 3. MÓDULO DE SALIDAS DE LA RTU

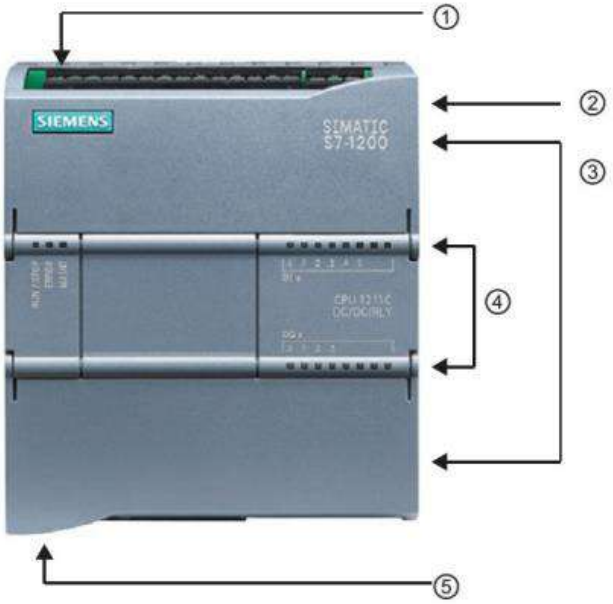




12.4. ANEXO 4. MÓDULO DE CONTROL Y COMUNICACIÓN INALÁMBRICA



12.5. ANEXO 5. PLC S7 1200



① Conector de corriente
② Ranura para Memory Card (debajo de la tapa superior)
③ Conectores extraíbles para el cableado de usuario (detrás de las tapas)
④ LEDs de estado para las E/S integradas
⑤ Conector PROFINET (en el lado inferior de la CPU)

DATOS TECNICOS

MODELO: SIMATIC S7-1200, CPU 1215C,
TIPO: AC/DC/RLY,
NÚMERO DE PARTE: 6ES7 215-1BG40-0XB0
VERSIÓN DE FIRMWARE: V4.2
ENTRADAS INTEGRADAS: 14 DI 24V DC ;
SALIDAS INTEGRADAS: 10 DO 24 V DC, 2A;
ENTRADAS ANALÓGICAS: 2 AI DE 0 - 10V DC,

SALIDAS ANALÓGICAS: 2 AO DE 0 - 20V MA,
VOLTAJE DE ALIMENTACIÓN: AC 85 - 264 V AC CON 47 -63 HZ
MEMORIA DE PROGRAMA/DATOS: 100KB
TIPO DE INTERFAZ: PROFINET
NORMA FÍSICA: ETHERNET

12.6. ANEXO 6. ENTORNO DE DESARROLLO TIA PORTAL

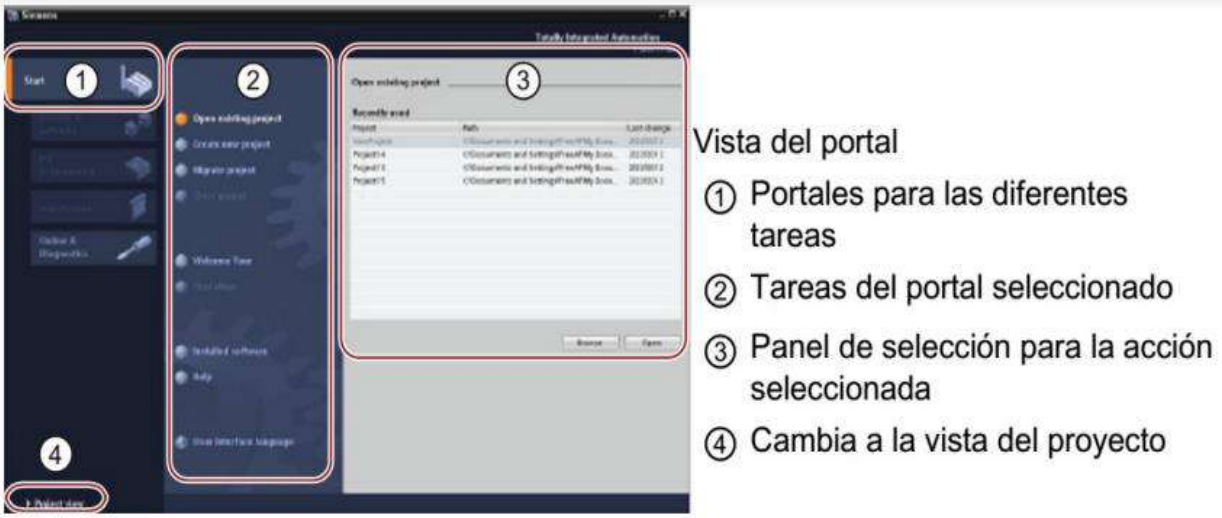
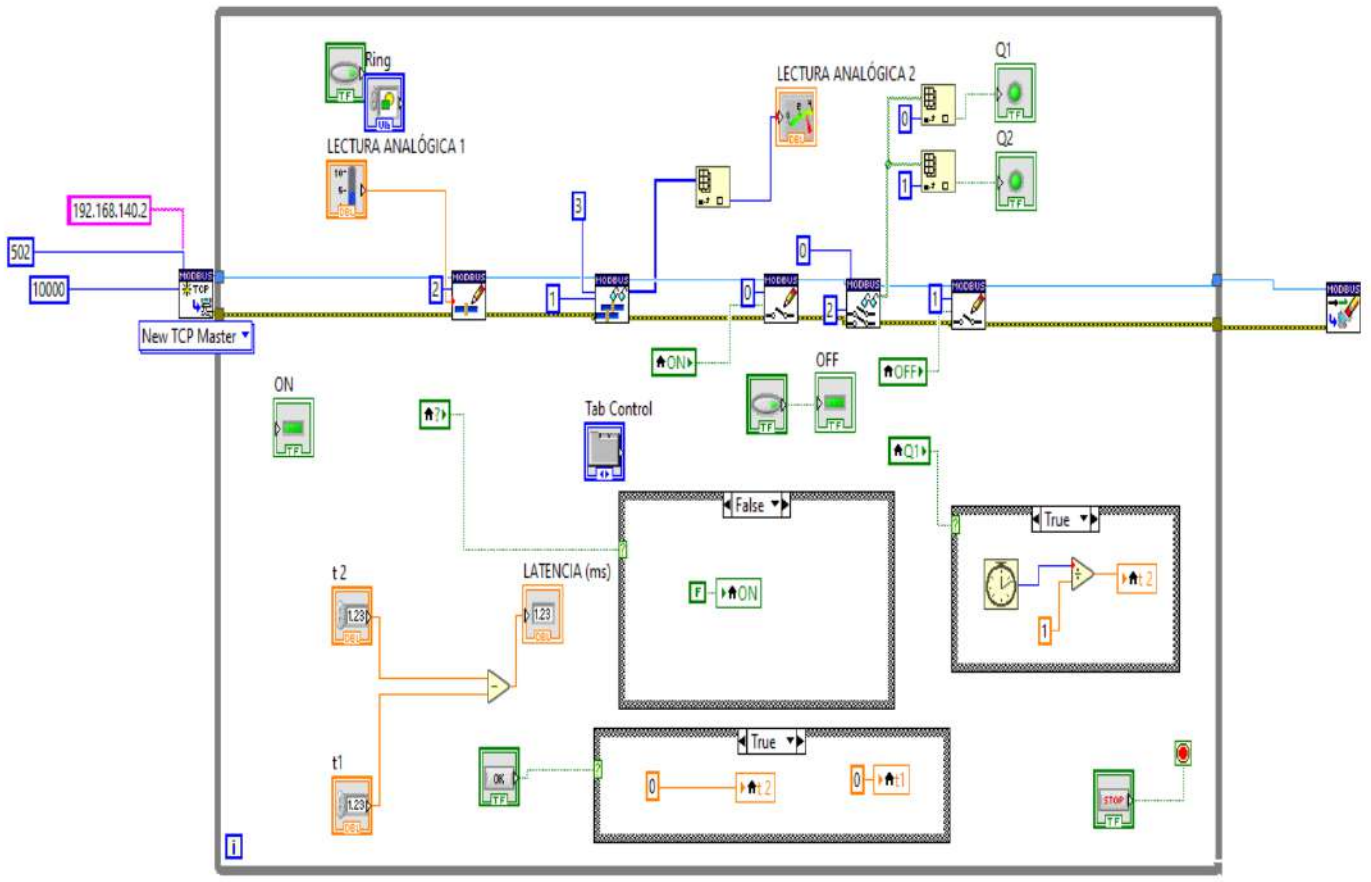


Figura 2. Vista del portal.

12.7. ANEXO 7. CÓDIGO LABVIEW PARA LA PRUEBA DE LATENCIA



12.8. ANEXO 8. EQUIPO PARA MONTAJE DE CIRCUITOS ELECTRONEUMÁTICOS

ITCA FEPADE
TÉCNICOS E INGENIEROS

SEDE CENTRAL Y CENTROS REGIONALES EL SALVADOR

La Escuela Especializada en Ingeniería ITCA-FEPADE, fundada en 1969, es una institución estatal con administración privada, conformada actualmente por 5 campus: Sede Central Santa Tecla y cuatro centros regionales ubicados en Santa Ana, San Miguel, Zacatecoluca y La Unión.

1. SEDE CENTRAL SANTA TECLA
Km. 11.5 carretera a Santa Tecla, La libertad.
Tel.: (503) 2132-7400

2. CENTRO REGIONAL SANTA ANA
Final 10a. Av. Sur, Finca Procavia.
Tel.: (503) 2440-4348

3. CENTRO REGIONAL ZACATECOLUCA
Km. 64.5, desvío Hacienda El Nilo sobre autopista a Zacatecoluca.
Tel.: (503) 2334-0763 y 2334-0768

4. CENTRO REGIONAL SAN MIGUEL
Km. 140 carretera a Santa Rosa de Lima.
Tel.: (503) 2669-2298

5. CENTRO REGIONAL LA UNIÓN
Calle Sta. María, Col. Belén, atrás del Instituto Nacional de La Unión
Tel.: (503) 2668-4700

www.itca.edu.sv

ISBN: xxx-xxxx-xx-xx-x (Impreso)
ISBN: xxx-xxxx-xx-xx-x (E-book)