Universidad Nacional de Ingeniería

Área de Conocimiento de Tecnología de la Información y Comunicación

Diseño de una aplicación grafica para codificar y decodificación de variables de dispositivos con protocolo de comunicación Modbus usando el Software de LabVIEW.

Trabajo Monográfico para optar al título de Ingeniero en Electrónica

Elaborado por:

Br. Juan Carlos Brizuela Jiménez Carnet: 2010-32522	Br. Oneyl Augusto Rivas Pravia Carnet: 2010-32980	Br. Josué Otoniel Ortega Acevedo Carnet: 2009-30161	**Tutor:** Ing. Jaime Álvarez Calero

03 de abril de 2024
Managua, Nicaragua

DEDICATORIA

Dedicado a mi madre que es la persona que siempre ha estado a mi lado, la que en todo momento ha permanecido atenta para alentarme con sus palabras, consejos y también con sus hechos, llenándome de valor para afrontar los retos, superar los obstáculos, e inspirándome para cumplir mis sueños.

Igualmente, a mi padre quien me ha inspirado a luchar por mis metas y por mi superación personal, a ser siempre honesto y siempre tener la frente en alto, pero siempre con humildad ya que es Dios quien nos da fuerza y valor e igualmente nos permite alcanzar nuestras metas.

Josué Otoniel Ortega Acevedo

Dedico mi trabajo monográfico primeramente a Dios que ha brindado la salud y sabiduría para poder terminar este tan importante documento, cada paso en mi vida se lo pongo en sus manos.

Dedico muy especialmente a mis padres que me apoyaron en lo largo de la carrera y me ayudaron a enfrentar todos los obstáculos que se presentó durante todo mi periodo universitario, igualmente le dedico a mi Hija e Esposa que me motivaban para finalizar dicho trabajo monográfico, A cada una de las personas que han aportado un poco a mi vida, a través de algún consejo, de alguna palabra de aliento en momentos difíciles. Amigos y demás familiares que siguen en nuestras vidas y otros que nos dejaron demasiado pronto.

Oneyl Augusto Rivas Pravia.

A Dios, primeramente, por darme fuerzas para no rendirme y estar presente siempre, a mis amados padres y hermanos, a mi mamita Norma y Marcelina, a abuelo Eder, a Diana, a mis amigos, colegas y personas queridas quienes forman parte de mi crecimiento personal y profesional.

Juan Carlos Brizuela Jiménez.

AGRADECIMIENTOS

A Dios primeramente ya que sin él no somos nada, porque es el que cada día nos permite levantarnos con energía, y es el que me ha abierto puertas y oportunidades para estar en donde me encuentro ahora y sé que su voluntad es para bien ya que sus bendiciones son abundantes.

Segundo quiero agradecer a mis familiares que siempre han estado para apoyarme y brindarme de sus más sinceros deseos, quienes siempre se alegran con mis avances y mis logros realizados.

Josué Otoniel Ortega Acevedo

Especialmente, le agradezco a Dios su presencia siempre me ha acompañado, como está escrito en el libro de Josué 1:5 "*Como yo fui con Moisés, seré contigo; no te dejaré, ni te desamparare*", es por la gran misericordia de él que podemos culminar esta etapa de mi vida, pero ni con toda nuestra vida tendremos el tiempo suficiente para agradecerle todo lo que hizo, hace y seguirá haciendo por mí.

Agradezco en gran manera a mis padres, esposa e hija, que en todo momento fueron mi fuente de apoyo, durante mis años de estudiante. Les agradecemos sus consejos en la incertidumbre y que sus brazos siempre hayan estado prestos para sostenerme, agradezco que pusieran sus preocupaciones en segundo lugar por ayudarme a las mías.

También agradezco a mis profesores de la universidad que me tuvieron la paciencia y perseverancia de transmitir sus conocimientos a lo largo de mi carrera.

Oneyl Augusto Rivas Pravia

Los que me conocen bien saben que, entre lo despistado y lo poco expresivo que puedo llegar a ser, muchos agradecimientos no quedaran aquí reflejados, pero también han sido importantes y no quedan en el olvido.

Primeramente, a Dios quien estará siempre en cada momento por darme fortaleza y sabiduría, agradezco a mis padres y hermanos quienes siempre serán una guía y ejemplo lo mejor del mundo, mi mamita Norma quien me recibió en sus manos desde que vi la Luz, a familiares que me aportan valores y muestran su apoyo, a esos amigos, colegas y personas que me tienen presente y aman sin ningún interés, agradezco a mi hermano de otra sangre Ovidio Torres por ser un maestro, a mis profesores quienes además de compartir conocimiento aportaron valores, …. Gracias.

Juan Carlos Brizuela Jiménez.

RESUMEN

El presente trabajo monográfico consiste en el desarrollo de una aplicación gráfica para el tratamiento de variables de dispositivos con protocolo de comunicación Modbus. Es dirigido a lectores con interés en los sistemas de control y automatización industrial, estudiantes de ingeniería a fines a sistemas de control y profesionales del sector que tengan interés en aprender una forma de codificar y decodificar variables de dispositivos con protocolo de comunicación Modbus.

Inicialmente se exponen conceptos básicos de la automatización industrial, los sistemas industriales de control, los buses de campo tantos los abiertos como los estandarizados dentro de los cual se definen algunos como protocolo de comunicación HART y FIELBUS y donde también está situado el protocolo de comunicación MODBUS el cual detallaremos en sus diversos modos de transmisión; Modo RTU, Modo ASCII, Modbus plus y Modbus TCP para darnos una idea en qué forma se manifiesta el problema y la solución a la integración con dispositivos de distintos fabricantes, al análisis de cada uno de sus parámetros y la manera de tratar cada uno de estos.

El contenido del documento describe la forma de poder extraer y acondicionar las variables de dispositivos MODBUS que cada fabricante brinda al usuario en sus hojas de datos y demostrar la aplicabilidad de adquisición de datos mediante protocolo Modbus utilizando el software de National Instruments LabVIEW (Laboratory Virtual Instrument Engineering Workbench) se detalla nuestro lenguaje de programación de flujo de datos de forma simple dado a que la ejecución es determinada por la estructura de un diagrama gráfico de bloques (código Fuente LV).

La aplicación se iniciará determinando los parámetros en una tabla del Software de Microsoft EXCEL para ingresar cualquier hoja de variables de dispositivos con protocolo MODBUS, para luego ser tratados automáticamente por la herramienta en LabVIEW. Finalmente se eligieron dos dispositivos de fabricantes diferentes Motor LogicScuareD II Y PM5320 schneider para realizar una prueba piloto y también para dejar una guía de laboratorio en la facultad (FEC) que servirá de referencia académica y que dispongan de un documento que les ilustre el ámbito industrial o posibles proyectos de integración a futuro.

ÍNDICE

Índice de Figuras

Índice de Tablas

I. INTRODUCCIÓN

Uno de los elementos fundamentales que poseen las empresas del mundo moderno es la información. Esta debe de ser concreta, concisa y oportuna, sin importar el dispositivo de procedencia; por esta razón el proceso de integración juega un papel fundamental para garantizar el correcto manejo de dicha información, de lo que depende hoy en día el rendimiento, la productividad, la competitividad y la permanencia de las empresas e industrias en el mundo moderno.

En el área de las comunicaciones en entornos industriales, dada la creciente tendencia de las industrias y empresas de mejorar y controlar procesos de la mejor manera surge la necesidad de realizar un estudio detallado de los protocolos de comunicación industrial que pueden ser utilizados. Uno de estos protocolos es el MODBUS ya que gracias a sus características es hasta la fecha un protocolo estándar utilizado en redes industriales. De ahí surge la necesidad de plasmar este estudio mediante una herramienta que permita utilizar el protocolo MODBUS. Esta herramienta está diseñada para demostrar la aplicabilidad de extracción de variables de cualquier fabricante que utilice el protocolo de comunicación MODBUS y a la vez ser una herramienta práctica para el usuario.

Para desarrollar la herramienta se utiliza el software de National Instruments LabView que permite así programar de una manera que permita que la PC sea configurada como Master y el dispositivo Modbus como esclavo.

En la actualidad, hay un gran número de éxitos productos comerciales utilizables en las industrias, esto debido que es aplicable para controlar una gran variedad de situaciones industriales, lo que cambian en algunos de estos dispositivos es el tipo de lenguaje que este interpreta. Este documento es la propuesta donde se describe el diseño de una herramienta gráfica la cual permita la compatibilidad extracción y comunicación de una gran cantidad de dispositivos que son muy utilizados en las grandes industrias y que tienen en común el protocolo de comunicación MODBUS, la metodología para alcanzarlo, así como los fundamentos teóricos del mismo.

La solución descrita en este documento se desarrollará con mayor detalle durante el avance del proyecto, al igual que la explicación y descripción de la programación usada.

II. OBJETIVOS

2.1. Objetivo general

Diseñar una aplicación gráfica que facilite al usuario la codificación y decodificación de variables de dispositivos con protocolo de comunicación Modbus mediante los softwares LabVIEW y Microsoft EXCEL.

2.2. Objetivos específicos:

- Analizar el protocolo de comunicaciones Modbus para el enlace de la red con elementos de campo.

- Describir los beneficios tecnológicos del protocolo Modbus como una arquitectura a nivel industrial.

- Diseñar una herramienta gráfica de operación en LabVIEW para él envió y recepción de datos de cualquier dispositivo con protocolo de comunicación MODBUS.

- Determinar los parámetros clave en tablas del Software de Microsoft EXCEL para incorporar cualquier hoja de variables de dispositivos con protocolo MODBUS para luego ser tratados automáticamente por la herramienta en LabVIEW.

- Realizar una prueba piloto de la aplicación con dispositivos ModBus (Motor LogicScuareD II y Pm5320 schneider) como una prueba de laboratorio de la facultad (FEC).

III. JUSTIFICACIÓN

La finalidad de esta tesis fue desarrollar un programa gráfico de fácil manejo y que le permita al usuario la posibilidad de poder extraer y acondicionar las variables de dispositivos MODBUS que cada fabricante brinda al usuario en sus hojas de datos y así demostrar la aplicabilidad de adquisición de datos mediante protocolo Modbus utilizando el software de National Instruments LabVIEW. Modbus es un estándar industrial debido a su simplicidad en cuanto a componentes de hardware, sobre todo por ser un protocolo abierto y que actualmente es ampliamente utilizado por diferentes fabricantes.

El sistema desarrollado presenta una posible alternativa de solución al problema de integración de dispositivos de distintos fabricantes, al análisis de cada uno de sus parámetros y la manera de tratar cada uno de estos la cual se hace de manera diferente por cada variable. Este estudio y pruebas realizadas puede permitir a las personas conocer los sistemas de producción de cualquier empresa o industria que mantengan sus operaciones y procesos con controladores lógicos programables y equipos con esta infraestructura de red.

La trascendencia que presta este sistema es que al estar este protocolo (Modbus) presente en la mayoría de estos dispositivos industriales mejora y brinda una herramienta de operación adicional a un sistema de control, así como un aporte significativo para la universidad Nacional de Ingeniería con sus ingenieros recién egresados.

Por lo antes descrito, se observó la necesidad que tienen los sistemas de control y que, dada la creciente tendencia de las empresas e industrias en mejorar y controlar procesos, demostrar como la implementación de esta herramienta influirá determinantemente a que el usuario sea capaz de poder implementar el protocolo sin necesidad de sumergirse en un software de programación, de extracción o al tratamiento de variables. Para lograr este propósito se usó un software común y de fácil manejo como lo es Microsoft EXCEL en donde simplemente se llenan las variables de los parámetros que se establecieron y se explican detalladamente de forma tal que se logre la comunicación MODBUS sin importar el fabricante y que sea la herramienta que se desarrolló en LabVIEW la encargada según lo digitado en la tabla de EXCEL de extraer, codificar y decodificar variables, en resumen, acondicionar las variables de dispositivo MODBUS.

IV. MARCO TEORICO

4. Automatización Industrial

4.1. Introducción

Los protocolos de comunicación industrial son importantes en las empresas de la actualidad y el tener un protocolo de comunicación estándar y un sistema que sea capaz de manejar dicha información facilita que necesidades sean cubiertas ya sea para control o supervisión de equipos, hasta adquisición de datos a distancia de forma fiable y segura, es ahí donde el termino automatización industrial nos dará una visión muchísimo más amplia de lo que puede ayudar esto a una empresa ya que se va a dar en la misma un proceso de mecanización de las actividades industriales para reducir la mano de obra, simplificar el trabajo para que así se de propiedad a algunas máquinas de realizar las operaciones de manera automática; por lo que indica que se va dar un proceso más rápido y eficiente.

Cuando se logra una mayor eficiencia en el sector de equipos y maquinarias, se logrará que la empresa industrial disminuya la producción de piezas defectuosas, por ejemplo, y por lo tanto aumente una mayor calidad en los productos que se logran mediante la exactitud de las maquinas automatizadas; definitivamente cuando en una empresa industrial invierte en tecnologías aumentara su competitividad y no sufrir así el riesgo de quedar rezagado en un mundo que evoluciona constantemente. *[1]*

4.2. Sistemas industriales de control

Todo este entramado de procesos no sería posible coordinarlos sin la existencia de los sistemas físicos capaces de captar, distribuir y almacenar toda la información generada. Es por ello que se hace necesaria la infraestructura de comunicaciones capaz de realizar la integración de sistemas industriales. Tradicionalmente se distinguen tres tipos de sistemas de control industrial: Control centralizado, control hibrido y control distribuido. *[1]*

4.2.1. Control centralizado

Este tipo de control se da en los casos cuando los sistemas son poco complejos donde un proceso puede ser gestionado directamente mediante un único elemente de control encargado de realizar todas las tareas del proceso de producción y que puede incluir un sistema de monitorización y supervisión. Estos sistemas han tenido una tendencia a emplear elementos de control más complejos y potentes manteniendo un único elemento de control como ventajas de esta metodología, no hay necesidad de invertir en un sistema de intercomunicación entre procesos ya que todas las señales están gestionadas por un mismo sistema, también en casos donde los sistemas son poco complejos posee un menor coste económico. Por otro lado, posee numerosas desventajas una de ellas se presentaría cuando el sistema de control falla, toda la instalación queda paralizada, siendo necesario un sistema redundante.

4.2.2. Control distribuido

El concepto de sistemas industriales distribuidos se da cuando los dispositivos conectados a la red forman entre si un anillo, el cual se conecta a través de un pequeño repetidor que interrumpe el canal ver figura 1. La información como tal se transmite mediante paquetes enviados de nodo a nodo desde el nodo emisor hasta el terminal destino, este tipo de topología brinda una excelente transmisión pues evita perdida de paquetes no obstante su un nodo falla toda la red podría dejar de funcionar.

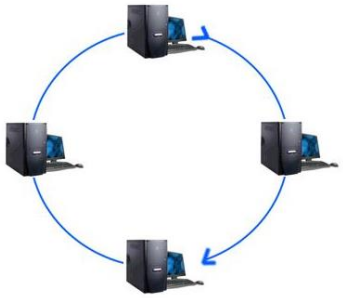


Figura 1- Red Tipo Anillo. *[1]*

4.2.3. Control hibrido

El control hibrido no está muy definido ya que este tipo de control de planta puede considerarse a cualquier estrategia de distribución de elementos de control a medio camino entre el control distribuido y el control centralizado. En numerosas ocasiones no resulta sencillo separar los procesos de manera completamente autónoma, por lo que se debe recurrir a la gestión de varios procesos desde una misma unidad de control pues la complejidad de la separación es mayor que la complejidad que supone su gestión conjunta*. [1]*

4.3. Comunicaciones industriales.

El ejemplo de computadores y autómatas programables como herramienta esencial de tratamiento de la información es habitual, y la implantación de redes de comunicación internas, el uso de comunicaciones industriales se pueden separar dos áreas principales, una comunicación a nivel de campo y una comunicación hacia el SCADA. Según el entorno donde van a ser instaladas, en un ámbito industrial existen varios tipos de redes, la figura 2 pirámide de automatización nos muestra de manera genérica estas interrelaciones y su división en diferentes niveles. *[2]*

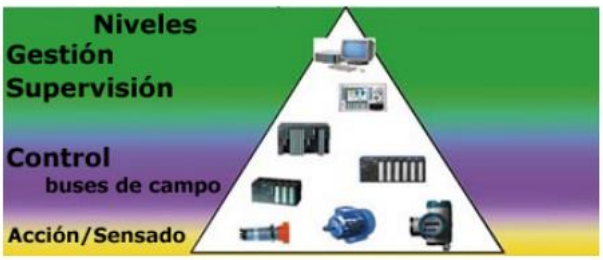


Figura 2- Pirámide de automatización. *[2]*

4.3.1. Nivel de acción / censado (Nivel de célula):

A este también se le conoce como nivel de instrumentación, son todos aquellos equipos que se pueden interconectar y operar en modo secuencial como Robots, máquinas de control numérico, elementos de medida y mando estos elementos están más directamente relacionados con el proceso productivo ya que los actuadores son los encargados de ejecutar las ordenes de los elementos de control para modificar el proceso productivo. Las redes tipo célula deben gestionar mensajes cortos eficientemente, capacidad de manejar tráfico de eventos discretos, bajo coste de instalación entre otras.

4.3.2. Nivel de control (Nivel de campo):

Este es el nivel donde se sitúan los elementos capaces de gestionar los actuadores y sensores de campo, nos referimos a PLC, o equipos de aplicación específica basados en microprocesador como robots, servidores etc. Estos dispositivos permiten que los actuadores y sensores funcionen de forma conjunta para ser capaces de realizar el proceso industrial deseado. Son dispositivos programables, de tal modo que es posible ajustar y personalizar su funcionamiento según las necesidades de cada caso. A pesar de tratarse de procesos aislados, esto no implica que no se empleen buses de comunicación, ya que para procesos que requieran de un gran número de sensores y actuadores es necesario el uso de buses de campo para leer el estado de los sensores, así como múltiples virtudes que brindaría un bus de campo.

4.3.3. Nivel de supervisión (Nivel de planta)

Todos los dispositivos de control existentes en planta es posible automatizarlos si existe un sistema de comunicación adecuado capaz de comunicar estos elementos con otro tipo de dispositivos no dedicados al control sino para la gestión y supervisión, y que habitualmente están constituidos por computadores o sistemas de visualización tales como pantallas industriales. En este nivel es posible visualizar cómo se están llevando a cabo los procesos de planta, y a través de entornos SCADA (Supervisión, Control y Adquisición de Datos) poseer una “imagen virtual de la planta” de modo que ésta se puede recorrer de manera detallada, o bien mediante pantallas de resumen ser capaces de disponer de un “panel virtual” donde se muestren las posibles alarmas, fallos o alteraciones en cualquiera de los procesos que se llevan a cabo.

Mediante este tipo de acciones resulta inmediato disponer de acceso inmediato a cada uno de los sectores de la planta. Para ello, resulta imprescindible la conexión con el nivel de control mediante buses de campo de altas prestaciones, pues a veces resulta necesaria la transmisión de importantes cantidades de datos y la conexión con un gran número de elementos de control. Eventualmente, también es posible modificar los procesos productivos desde los computadores de supervisión. Este nivel sustituye a los grandes paneles y salas de control que durante los años ’70 y ’80 eran habituales en las grandes empresas, pues el computador lo ha sustituido. *[2]*

4.3.4. Nivel de gestión (Nivel de fábrica)

Para redes de oficina, contabilidad y administración, ventas, gestión de pedidos, almacén, etc. El volumen de información intercambiada es muy alto, y los tiempos de respuesta no son críticos. *[2]*

4.4. Sistemas Normalizados, sistemas abiertos.

Debido a la gran aceptación que han tenido los sistemas industriales basados en redes de comunicación, han surgido un gran número de protocolos y sistemas físicos capaces de satisfacer las necesidades del mercado. Para que estos sistemas puedan ser implantados de forma generalizada, deben cumplir diversas especificaciones en cuanto a cumplimiento de normativa se refiere, debiendo superar diferentes pruebas de homologación y cumplir con las normativas de normalización impuestas para ese dispositivo o protocolo. Por otro lado, los fabricantes de equipos de automatización se han centrado en dos grandes áreas de trabajo: la propuesta de nuevos equipos de comunicación basados en protocolos y medios de transmisión preestablecidos, y el desarrollo de nuevos sistemas y protocolos de transmisión propietarios.

En el segundo caso, la empresa fabricante se asegura que el protocolo desarrollado no puede ser empleado por otros fabricantes, debiendo recurrir a su permiso expreso en caso de querer desarrollar dispositivos para dicho protocolo, o bien adquiriendo los dispositivos al fabricante. Con este método, el fabricante se asegura el mercado para ese tipo de dispositivos, pero no permite una compatibilidad con el resto de los sistemas de automatización que pueden existir en una planta industrial. A menudo se dice que este tipo de automatización provoca la existencia de “islas de automatización”. Debido a los problemas que este tipo de metodología puede originar, los clientes de estos sistemas demandan una compatibilidad entre fabricantes y la adopción de sistemas normalizados y abiertos, regulados por organismos internacionales de modo que se facilite el desarrollo de dispositivos a cualquier empresa y ello permita una mayor compatibilidad entre fabricantes y una mayor rapidez en la implantación de nuevas tecnologías en todo el mundo a unos precios razonables.

Por tanto, el concepto de “Sistema abierto” adquiere gran importancia, y según el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) define un sistema abierto como: “Un sistema que incorpora suficientes especificaciones o estándares para interfaces, servicios y formatos como para desarrollar y planificar aplicaciones software capaces de: trasladarse con cambios mínimos a un amplio abanico de sistemas provenientes de uno o más fabricantes, dialogar con otras aplicaciones tanto en el sistema local como en sistemas remotos, e interactuar con los usuarios y programadores de modo que facilita la migración” *[2]*

Existen varios organismos internacionales encargados, entre otras tareas, de definir los estándares capaces de ser usados por todos aquellos desarrolladores interesados, y que proporcione suficiente flexibilidad, robustez y prestaciones

como para ser aceptado globalmente como un estándar “de facto”. Un “estándar” se define como: Documentos consensuados que contienen especificaciones técnicas o criterios precisos para ser empleados de forma sólida como reglas, pautas o definición de características para asegurar que materiales, productos, procesos y servicios se ajustan a la función a la que se destinan. *[2]*

4.4.1. Organismos internacionales más importantes

- International Organisation for Standarisation (ISO). Esta organización internacional y no gubernamental pretende agrupar los estándares que existen y se van creando dentro de cualquier campo de la actividad humana, para de este modo permitir la compatibilidad de los sistemas a lo largo de todo el planeta. Los acuerdos internacionales que incorporen nuevas propuestas son publicados como estándares internacionales ISO.

- International Electrotechnical Comission (IEC). Esta asociación internacional se centra en aspectos más relacionados con los sistemas electrónicos y computadores, por lo que ha servido como base para la creación de la mayoría de los estándares ISO relacionados con los computadores y normativas eléctricas y electrónicas.

- Joint Technical Committee No.1 (JTC1). A partir de 1987, ISO e IEC formaron un comité conjunto que, al igual que ISO e IEC, incorpora los comités nacionales de estandarización, de los que los más conocidos son: BSI (Reino Unido), DIN (Alemania), AFNOR (Francia), ANSI (Estados Unidos), JISC (Japón), CSA (Canadá), AENOR (España). Todos los estándares surgidos a partir de la unión entre ambos comités se preceden del prefijo ISO/IEC. Como ejemplo, al estándar de intercomunicaciones OSI (Open System Interconnection) le corresponde el estándar principal ISO/IEC 7498.

- International Telecommunications Union (ITU). Es el principal organismo regulador de las telecomunicaciones, en concreto el denominado ITU-T, responsable de la estandarización de las telecomunicaciones. Fue creado en 1993 y sustituye al anterior CCITT (International Telegraph and Telephone Consultative Committee). Su principal misión consiste en asegurar la creación de estándares efectivos que aseguren una alta calidad en cualquier aspecto que involucra las telecomunicaciones. *[2]*

4.4.2. Entornos de sistemas abiertos.

Así como los estándares OSI definen los métodos de interconexión entre computadores y sistemas electrónicos, se define un término que abarca OSI e intenta incorporar mayor número de elementos en los llamados “sistemas de comunicaciones abiertos”, los elementos principales de los que se compone son: Gestión, Interfaces de usuario, Interfaces de servicio entre programas, Formatos de datos e información e Interfaces de comunicación. *[2]*

V. 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 bucle de corriente de 4-20mA. Típicamente son redes digitales, bidireccionales, multipunto, montadas sobre un bus serie, que conectan dispositivos de campo como PLCs, transductores, actuadores y sensores. Cada dispositivo de campo incorpora cierta capacidad de proceso, que lo convierte en un dispositivo inteligente, manteniendo siempre un costo bajo. Cada uno de estos elementos será capaz de ejecutar funciones simples de diagnóstico, control o mantenimiento, así como de comunicarse bidireccionalmente a través del bus.

El objetivo es reemplazar los sistemas de control centralizados por redes de control distribuido mediante el cual permita mejorar la calidad del producto, reducir los costos y mejorar la eficiencia. Para ello se basa en que la información que envían y/o reciben los dispositivos de campo es digital, lo que resulta mucho más preciso que si se recurre a métodos analógicos. Además, cada dispositivo de campo es un dispositivo inteligente y puede llevar a cabo funciones propias de control, mantenimiento y diagnóstico.
De esta forma, cada nodo de la red puede informar en caso de fallo del dispositivo asociado, y en general sobre cualquier anomalía asociada al dispositivo. Esta monitorización permite aumentar la eficiencia del sistema y reducir la cantidad de horas de mantenimiento necesarias. *[3]*

1.1. Ventajas de los Buses de Campo

Los buses de campo presentan las siguientes ventajas:

- Minimización del cableado usado lo que abarata costos, facilita la instalación de la red y en la instalación y disminuye el tiempo de mantenimiento.

- Distancias operativas superiores al cableado tradicional.

- Permite la comunicación bidireccional entre los dispositivos de campo y los sistemas de control, pero también entre los propios dispositivos de campo un bus de campo posibilita la adición y retiro de equipos u otros elementos en pleno funcionamiento.

- Facilita el diagnóstico de errores o problemas ocurridos dentro de la red.

- Trabaja con Interfaces abiertas normalizadas, lo que habilita la conexión de equipos de múltiples diseñadores.

De manera general, aunque especialmente para los buses de campo y célula, las ventajas principales que se obtienen en su utilización son: Mejor calidad y cantidad en el flujo de datos, ahorro de coste de cableado e instalación (Figura 3), facilidad en la ampliación o reducción del número de elementos del sistema, reducción de errores en la instalación y numero de terminales en la caja de conexión *[3].*

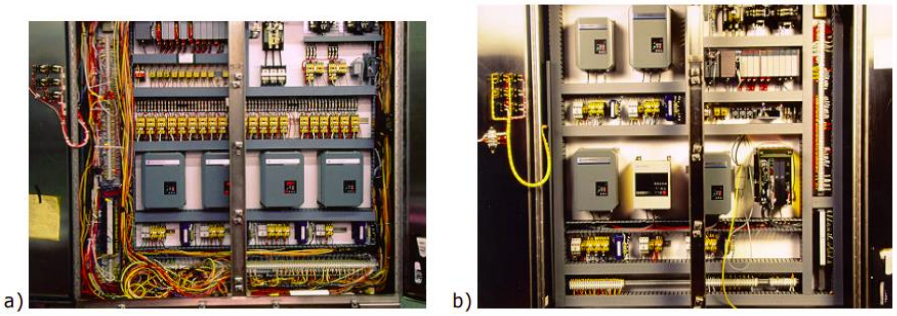


Figura 3- Instalación industrial: a) Sin utilización de buses de campo, b) Con buses de campo. *[3]*

1.2. Clasificación de los Buses de Campo

Debido a la falta de estándares y las grandes expectativas económicas que reporta el control de procesos industriales, un gran número de fabricantes de equipos han desarrollado diversas soluciones, cada una de ellas con distintas prestaciones y campos de aplicación. A continuación, se presenta una clasificación de los buses de campo de acuerdo con su funcionalidad: *[3]*

1.2.1. Buses de alta velocidad y baja funcionalidad

Están diseñados para integrar dispositivos simples como relés, actuadores simples, etc.; que operan en aplicaciones de tiempo real, y se hallan agrupados en una pequeña zona de la planta, típicamente una máquina. Básicamente comprenden las capas físicas y de enlace del modelo OSI. Ejemplos de este tipo de buses son los siguientes:

- CAN: Diseñado originalmente para su aplicación en vehículos.
- SDS: Bus para la integración de sensores y actuadores, basado en CAN.
- ASI: Bus serie diseñado por Siemens para la integración de sensores y actuadores. *[3]*

1.2.2. Buses de alta velocidad y funcionalidad media

Son buses controlan dispositivos de campo complejos, de forma eficiente y a bajo costo, están basados en el diseño de una capa de enlace para el envío eficiente de bloques de datos de tamaño medio. Normalmente incluyen la especificación completa de la capa de aplicación, lo que significa que se dispone de funciones utilizables desde programas basados en PCs para acceder, cambiar y controlar los diversos dispositivos que constituyen el sistema.

1.2.3. Buses de altas prestaciones

Soportan comunicaciones a través de todos los niveles de la producción CIM. Aunque se basan en buses de alta velocidad, pueden presentar problemas debido a la sobrecarga necesaria para alcanzar las características funcionales y de seguridad que se les exigen.
Entre sus características se incluyen las siguientes:

- Redes multi-maestro con redundancia.
- Comunicación maestro-esclavo según el esquema pregunta-respuesta.
- Recuperación de datos desde el esclavo con un límite máximo de tiempo.
- Capacidad de direccionamiento unicast, multicast y broadcast.
- Petición de servicios a los esclavos basada en eventos.
- Comunicación de variables y bloques de datos orientada a objetos.
- Descarga y ejecución remota de programas.
- Altos niveles de seguridad de la red, opcionalmente con procedimientos de autentificación

Algunos ejemplos son de este tipo de buses son: Profibus, WorldFIP, Fieldbus Foundation. *[3]*

1.2.4. Buses para áreas de seguridad intrínseca

La seguridad intrínseca es un tipo de protección por la que el componente en cuestión no tiene posibilidad de provocar una explosión en la atmósfera circundante. Un circuito eléctrico o una parte de un circuito tienen seguridad intrínseca, cuando alguna chispa o efecto térmico en este circuito producidos en las condiciones de prueba establecidas por un estándar no puede ocasionar una combustión. Algunos ejemplos son HART, Profibus PA o WorldFIP. *[3]*

1.2.5. Buses Estandarizados

1.2.5.1. PROFIBUS:

El bus de campo PROFIBUS es una tecnología industrial que permite a los fabricantes diseñar y mantener redes seguras e inteligentes. Está diseñado para proporcionar comunicación entre dispositivos dentro del controlador, así como con equipos externos automatizados. El sistema fue desarrollado por una iniciativa del gobierno alemán a fines de la década de 1980, en conjunto con varias empresas y rápidamente se posicionó dentro de las tecnologías más adoptadas, debido a sus prestaciones y a su estructura estandarizada y abierta. *[4]*

El desarrollo y posterior comercialización ha contado con el apoyo de importantes fabricantes como ABB, AEG, Siemens, Klóckner-Moeller. Está controlado por la PNO (Profibus User Organisation) y la PTO (Profibus Trade Organisation) existen tres perfiles:

- Inicialmente PROFIBUS ofrecía el perfil FMS (**F**ieldbus **M**essage **S**pecification) el cual permitía la comunicación entre diferentes sistemas de automatización con una estructura estándar y orientada a objetos, con grandes volúmenes de datos, pero sin la necesidad de comunicación de tiempo real o determinística. Este perfil cayo en desuso rápidamente al evolucionar la tecnología.

- PROFIBUS DP (Periferia Descentralizada, del inglés **D**ecentralized **P**eriphery), el cual en su versión DP-V0 incorpora, a la base que se tenía de FMS, la posibilidad de intercambiar datos de entradas y salidas de manera determinísticas o en tiempo real, lo cual le permitió ser utilizado para controlar un proceso industrial.

- Por último, el perfil DP-V1 posibilita la adopción de PROFIBUS en automatización de procesos. PROFIBUS PA (de inglés **P**rocess **A**utomation) es un perfil de PROFIBUS especialmente diseñado para conectar sensores (instrumentos de medición de variables físicas como caudal, temperatura, etc.) y actuadores (válvulas) que son utilizados en industrias de proceso, por caso químicas, petroquímicas, etc., con características que permiten su implementación en aquellas zonas con riesgos de explosión (polvos y gases), la posibilidad de reconfigurar parámetros operativos o intercambio de dispositivos sin necesidad de detener el proceso que esta “corriendo”, esto lo logra a través de una nueva capa física conocida como MBP (**M**anchester **B**us **P**owered) la cual permite que comunicación y suministro de alimentación viajen por el mismo para de hilos. *[4]*

1.2.5.2. DeviceNet:

DeviceNet es una red digital, multi-punto para conexión entre sensores, actuadores y sistemas de automatización industrial en general. Esta tecnología fue desarrollada para tener máxima flexibilidad entre los equipos de campo e interoperabilidad entre diferentes fabricantes. Introducido originalmente en 1994 por Allen-Bradley, DeviceNet transfirió su tecnología a ODVA (Open DeviceNet Vendor Association) en 1995. La ODVA es una organización sin fines de lucro compuesta por cientos de empresas alrededor del mundo que mantienen, difunden y promueven la tecnología DeviceNet y otras redes basadas en el protocolo CIP (Common Industrial Protocol). *[5]*

La red DeviceNet está clasificada en el nivel de red llamada devicebus, cuyas características principales son: alta velocidad, comunicación a nivel de byte que incluye comunicación con equipos discretos y analógicos y alto poder de diagnóstico de los dispositivos de la red.

La tecnología DeviceNet es un estándar abierto de automatización con el objetivo de transportar 2 tipos principales de información:

1. Datos cíclicos de sensores y actuadores, directamente relacionados al control.
2. Datos no cíclicos indirectamente relacionados al control, como configuración y diagnóstico.

Los datos cíclicos representan la información intercambiada periódicamente entre el equipo de campo y el controlador. Por otro lado, los no cíclicos son informaciones intercambiadas eventualmente durante la configuración o diagnóstico del equipo de campo.

La capa física y de acceso a la red DeviceNet está basada en la tecnología CAN (Controller Area Network) y las capas superiores en el protocolo CIP (Common Application Layer – Capa de Aplicación Común), que define una arquitectura basada en objetos y conexiones entre ellos. El CAN fue originalmente desarrollado por la BOSCH para el mercado de automóviles europeos para sustituir el cableado costoso por un cable en red de bajo costo en automóviles. Como resultado, el CAN tiene respuesta rápida y confiabilidad alta para aplicaciones principalmente en las áreas automovilística.

Una red DeviceNet puede tener hasta 64 dispositivos donde cada uno ocupa un nodo en la red, direccionados de 0 a 63. Cualquiera de ellos puede ser utilizado, aunque el uso de la dirección 63 no es recomendable, ya que se utiliza para la puesta en marcha ver figura 4 ejemplo de una red DeviceNet. *[5]*

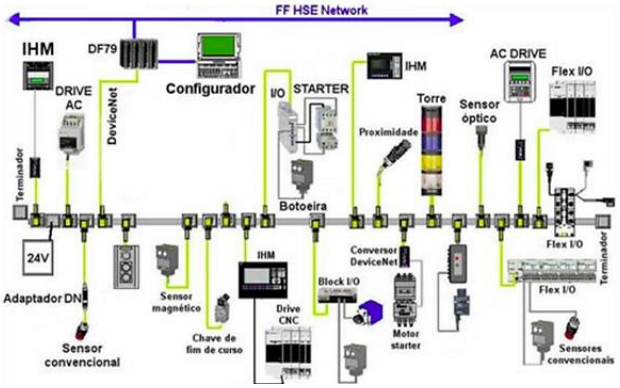


Figura 4- Ejemplo de una red DeviceNet. *[5]*

1.2.5.3. FOUNDATION FIELDBUS

Fundation Fieldbus es un sistema de comunicación digital, serial, bidireccional el cual interconecta los equipos de campo tales como sensores, actuadores y controladores. Fieldbus es una red de área local para instrumentos usado en automatización de procesos con capacidad para distribuir la aplicación de control a través de la red, en la figura 5 se puede observar este tipo de red. *[6]*

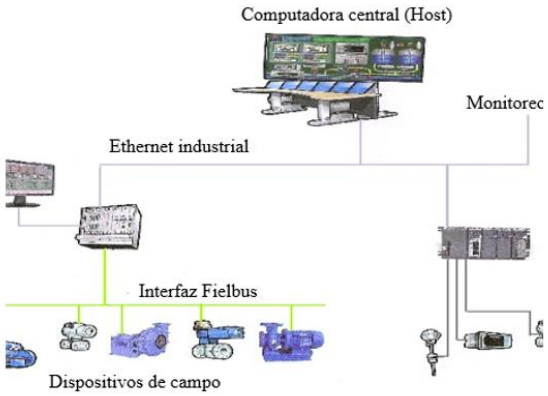


Figura 5- Red FIELBUS. *[6]*

Beneficios del Fieldbus: Fieldbus ofrece una serie de beneficios en cuanto a su instalación, reducción de hardware, interoperabilidad y mantenimiento.Instalación: Fieldbus permite a muchos dispositivos ser conectados a un simple par alámbrico, esto resulta en menos cableado, barreras de seguridad menos intrínsecas y menos hosts, en la figura 6 se muestra la ventaja en la instalación de la red fielbus.

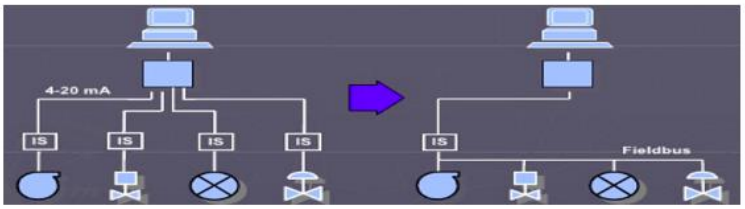


Figura 6- Ventaja de la red Fieldbus. *[6]*

Reducción de hardware: Fieldbus usa los bloques de funciones (funciones de automatización estandarizados) para implementar una estrategia de control. Muchas funciones de sistemas de control tales como PID (con entradas analógicas AI y salidas analógicas AO) pueden ser desempeñados por el dispositivo de campo a través del uso de bloque de funciones, es decir, estos bloques de funciones permiten la distribución de funciones en los dispositivos de campo, esta se utiliza para reducir la cantidad de I/O y equipos de control necesarios incluyendo tarjetas, cabinas y fuentes de poder.

Un bus orientado sobre todo a la interconexión de dispositivos en industrias de proceso continuo. Su desarrollo ha sido apoyado por importantes fabricantes de instrumentación (Fisher-Rosemount, Foxboro). Normalizado como ISA SP50, IEC-ISO 61158 (ISA es la asociación internacional de fabricantes de dispositivos de instrumentación de proceso).

En su nivel H1 (uno) de la capa física sigue la norma IEC 11158-2 para comunicación a 31,25 Kbps, es, por tanto, compatible con Profibús PA, su principal contendiente. Presta especial atención a las versiones que cumplen normas de seguridad intrínseca para industrias de proceso en ambientes combustibles o explosivos. *[6]*

Se soporta sobre par trenzado y es posible la reutilización de los antiguos cableados de instrumentación analógica 4-20 mA. Se utiliza comunicación síncrona con codificación Manchester Bifase-L. El nivel H2 (dos) está basado en Ethernet de alta velocidad (100 Mbps) y orientado al nivel de control de la red industrial. *[6]*

1.2.5.4. SDS

Smart Distributed System, junto con DeviceNet y CANOpen, uno de los buses de campo basados en CAN más extendidos. Se ha utilizado sobre todo en aplicaciones de sistemas de almacenamiento, empaquetado y clasificación automática. Se define una capa física que incluye alimentación de dispositivos en las conexiones. La capa de aplicación define autodiagnóstico de nodos, comunicación por eventos y prioridades de alta velocidad. *[6]*

1.2.5.5. MODBUS

En su definición inicial Modbus era una especificación de tramas, mensajes y funciones utilizada para la comunicación con los PLCs Modicon. Modbus puede implementarse sobre cualquier línea de comunicación serie y permite la comunicación por medio de tramas binarias o ASCII con un proceso interrogación-respuesta simple. Debido a que fue incluido en los PLCs de la prestigiosa firma Modicon en 1979, ha resultado un estándar de facto para el enlace serie entre dispositivos industriales. Modbus Plus define un completo bus de campo basado en técnica de paso de testigo. Se utiliza como soporte físico el par-trenzado o fibra óptica. *[7]*

1.2.5.6. INDUSTRIAL ETHERNET

Protocolo Ethernet/IP (Ethernet Industrial Protocol) es la de un estándar de red de comunicación capaz de manejar grandes cantidades de datos a velocidades de 10 Mbps o 100 Mbps, y hasta 1500 bytes por paquete. La especificación utiliza un protocolo abierto en la capa de aplicación. En la industria es especialmente popular para aplicaciones de control. En definitiva, este tipo de red es fácil de configurar, operar, mantener y ampliar. A su vez, permite la mezcla de productos de 10 Mbps y 100 Mbps, y es compatible con la mayoría de los conmutadores (switch) Ethernet.

Esta tecnología se utiliza con ordenadores personales, mainframes, robots, dispositivos y adaptadores de entrada/salida (E/S), controladores lógicos programables (PLC) y otros dispositivos. La especificación está respaldada por la Industrial Ethernet Association (IEA), ControlNet International (CI) y la Open DeviceNet Vendor Association (ODVA). Recientemente se ha convertido en la tendencia más utilizada en el movimiento de datos en aplicaciones industriales en la planta de producción. Sin embargo, la planta de producción es un entorno muy diferente al doméstico y al de la oficina. *[8]*

Debido a su fiabilidad, rendimiento e interoperabilidad inherentes, Ethernet se ha infiltrado en la planta de producción como el protocolo de comunicación preferido para los sistemas de automatización y control. En los últimos años, Ethernet Industrial ha superado la cuota de mercado de los protocolos de bus de campo tradicionales que normalmente requieren múltiples opciones de cableado.

Para entender qué es el ethernet industrial y cómo funciona es necesario conocer que utiliza esencialmente protocolos industriales especiales, encapsulados en el protocolo Ethernet, de forma que se garantice el envío y la recepción de la información correcta en el momento y el lugar en que se necesita para realizar una operación específica.

Breve Historia de Ethernet: Aunque Bob Metcalfe de Xerox esbozó el concepto original de Ethernet en una servilleta en 1973, su inspiración llegó incluso antes. En aquellos años ALOHAnet era una red de datos inalámbrica creada para conectar varios sistemas informáticos en los campus universitarios hawaianos (en diferentes islas). El reto consistía en permitir que varios nodos de radio con datos independientes se comunicaran entre sí, sin interferir entre ellos. La solución de ALOHAnet era una versión del concepto de detección de colisiones (CSMA/CD). En definitiva, Metcalfe basó su trabajo de doctorado en la búsqueda de mejoras en ALOHAnet. Esto le llevó a trabajar con Ethernet. Años más tarde, el protocolo de comunicación se convirtió en la base del estándar de red IEEE 802.3 que especificó la capa física y la capa de enlace de datos de la funcionalidad de red. *[8]*

1.2.5.7. CONTROLNET

Bus de alta velocidad (5 Mbps) y distancia (hasta 5 Km), muy seguro y robusto promovido por Allen-Bradley. Utiliza cable RG6/U (utilizado en televisión por cable) y se basa en un controlador ASIC de Rockwell. No es soportado por muchos fabricantes y resulta de elevado precio por nodo. Se ha utilizado para interconexión de redes de PLCs y computadores industriales en aplicaciones de alta velocidad y ambientes muy críticos. *[9]*

1.2.5.8. HART

Es un protocolo para bus de campo soportado por la HART Communication Foundation y la Fieldbus Foundation, Su campo de aplicación básico es la comunicación digital sobre las líneas analógicas clásicas de los sistemas de instrumentación, manteniendo éstas en servicio. Sus prestaciones como bus de campo son reducidas.

Utiliza el bus analógico estándar 4-20 mA sobre el que transmite una señal digital modulada en frecuencia (modulación FSK 1200-2200 Hz). Transmite a 1200 bps manteniendo compatibilidad con la aplicación analógica inicial y sobre distancias de hasta 3 Km según como se muestra en figura 7 y figura 8 modulación FSK y la forma de onda superpuesta. *[10]*

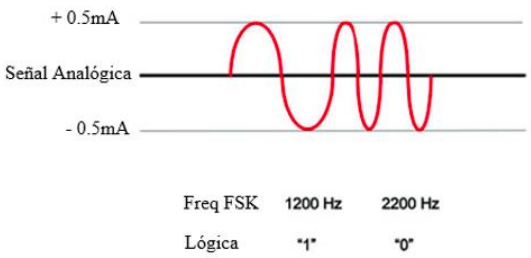


Figura 7- Modulación FSK. *[10]*

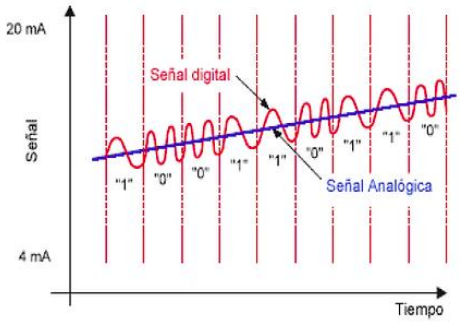


Figura 8- Señal de comunicación digital Hart superpuesta sobre la señal analógica. *[10]*

Beneficios de utilizar la tecnología Hart:

- Amplia variedad y número creciente de productos proporcionados por una lista creciente de proveedores de instrumentos en todo el mundo.
- Hart es el único protocolo de comunicación abierto de su tipo y un estándar en la industria, relativamente fácil de entender y utilizar.
- Los usuarios tienen la libertad de elegir el producto adecuado para su aplicación y la interoperatividad está asegurada por los comandos comunes y la estructura de datos.
- El protocolo Hart proporciona acceso a una gran variedad de información adicional (variables, diagnósticos, calibración, etc.) proporcionados por los dispositivos de campo inteligentes que emplean esta tecnología. Hart permite a los fabricantes de instrumentos de campo incorporar potentes características en sus productos como algoritmos de control PID de diagnóstico y medidas adicionales del proceso. *[10]*

Modo petición/respuesta

Hart usa el modelo de comunicación petición/respuesta, esto significa que, en general, los dispositivos Hart no transmitirán a menos que una petición sea enviada desde el host al dispositivo, por ejemplo, si un dispositivo Hart detecto una condición de falla en sí mismo o en el proceso, el dispositivo Hart no puede comunicar esta información al host a menos que el host solicite específicamente esta información del dispositivo, en la figura 9 se muestra el modo de comunicación petición/respuesta. *[10]*

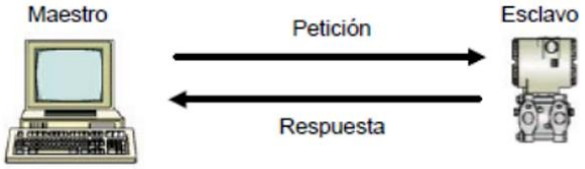


Figura 9- Modo de comunicación petición/respuesta*. [10]*

Modo de comunicación Burst

Los dispositivos Hart pueden enviar una simple variable de información continua de un dispositivo sin necesidad de peticiones hechas por el host, este modo es usualmente empleado para enviar una simple variable tal como una variable de proceso, en la figura 10 se muestra el modo de comunicación Burst.

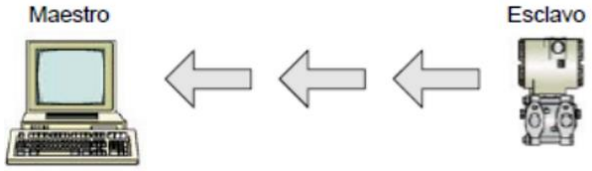


Figura 10- Modo de comunicación Burst. *[10]*

Modo de comunicación Multidrop.

El protocolo Hart también tiene la capacidad de conectar múltiples dispositivos de campo sobre el mismo par de hilos en una configuración de red multidrop como la que se muestra en la figura 11. La corriente a través de cada esclavo se fija a mínimo valor para alimentar el dispositivo. *[10]*

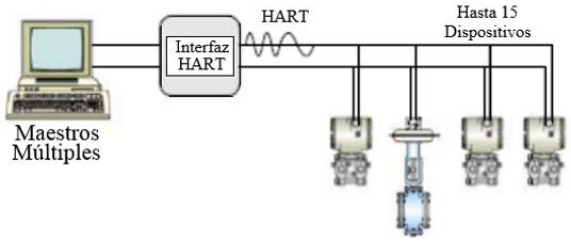


Figura 11- Modo de comunicación Multidrop. *[10]*

VI. PROTOCOLO DE COMUNICACIÓN MODBUS

6.1. Modbus-IDA

La Arquitectura para automatización distribuida (The architecture for distributed automation) Constituida como una asociación comercial sin fines de lucro, Modbus-IDA tiene una sola misión de peso: para ayudar a los proveedores de los usuarios De Modbus, y los desarrolladores de éxito.

Creado inicialmente por Schneider Electric, Modbus constituye ahora un recurso público gestionado por una organización independiente Modbus-IDA, que permite una apertura total de sus especificaciones. Modbus, estándar industrial desde 1979, permite la comunicación entre millones de productos. La IETF, autoridad internacional que gestiona Internet, ha aprobado la creación de un puerto (502) para los productos conectados a Internet/Intranet que utilicen el protocolo de comunicación Ethernet Modbus TCP/IP. *[7]*

6.2. Historia del protocolo Modbus

En 1979 cuando Modicon fabricante PLC-ahora una marca de Telemecanique de Schneider Electric-publicado la interfaz de comunicación Modbus para una red multipunto sobre la base de una arquitectura de maestro / cliente. La comunicación entre los nodos de Modbus se logró con los mensajes. Es un estándar abierto que describió la estructura de mensajería. La capa física de la interfaz Modbus era libre de elegir. RS-232, RS-485 La interfaz Modbus original corría en RS-232, pero la mayoría de las implementaciones más utilizadas Modbus RS-485, ya que permite largas distancias, velocidades más altas y la posibilidad de una red multi-gota. En un corto período de tiempo la empresa Hunderds de los vendedores acabo el Modbus sistema de mensajería en sus dispositivos y Modbus se convirtió en el estándar de facto para las redes de comunicación industrial.

Lo bueno de la norma Modbus es la flexibilidad, pero al mismo tiempo, la aplicación fácil. No sólo los dispositivos inteligentes, como microcontroladores, autómatas programables, etc. son capaces de comunicarse con Modbus, también muchos sensores inteligentes están equipados con una interfaz Modbus a enviar sus datos a los sistemas de acogida. Si bien anteriormente Modbus se utiliza principalmente en las líneas de comunicación por cable de serie, también hay extensiones para el estándar para comunicaciones inalámbricas y de redes TCP / IP. *[7]*

6.3. Introducción al Protocolo MODBUS

La designación Modbus Modicon corresponde a una marca registrada por Inc.Gould 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. No obstante, se suele hablar de MODBUS como un estándar de bus de campo, cuyas características esenciales son las que se detallan a continuación. *[7]*

6.4. Estructura de la red

Medio Físico

El medio físico de conexión puede ser un bus semidúplex (half duplex) (RS-485 o fibra óptica) o dúplex (full duplex) (RS-422, BC 0-20mA o fibra óptica). 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. *[7]*

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.

La única diferencia estriba en que la trama ASCII incluye un carácter de encabezamiento y los caracteres CR y LF al final del mensaje. 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 en la siguiente figura 12:

: (3AH)	Nº Esclavo ($00\text{-}3F_H$)	Código de Operación	Subfunciones, Datos	LRC(16) H L	CR ($0D_H$)	LF ($0A_H$)

Codificación ASCII

Nº Esclavo ($00\text{-}3F_H$)	Código de Operación	Subfunciones, Datos	CRC(P16) H L

Codificación RTU

Figura 12- Codificación de datos MODBUS serial. *[7]*

6.5. El ciclo Petición – Respuesta.

La Petición: El código de función en la petición indica al dispositivo esclavo diseccionado el tipo de acción a realizar. Los bytes de datos contienen cualquier información adicional que el esclavo necesitará para llevar a cabo la función. Por ejemplo, el código de función 03 pedirá al esclavo que lea registros mantenidos (holding regs.) y responda con sus contenidos. El campo de datos debe contener la información que indique al esclavo en qué registro debe comenzar y cuántos ha de leer. El campo de comprobación de error proporciona un método para que el esclavo valide la integridad del contenido del mensaje recibido.

La Respuesta: Si el esclavo elabora una respuesta normal, el código de función contenido en la respuesta es una réplica del código de función enviado en la petición. Los bytes de datos contienen los datos recolectados por el esclavo, tales como valores de registros o estados. Si ocurre un error, el código de función contenido en la respuesta es diferente al código de función enviado en la petición, para indicar que la respuesta es una respuesta de error y los bytes de datos contienen un código que describe el error.

El campo de comprobación de error permite al maestro confirmar que los contenidos del mensaje son válidos. En la siguiente figura 13 se describe el proceso de interrogación y respuesta. *[7]*

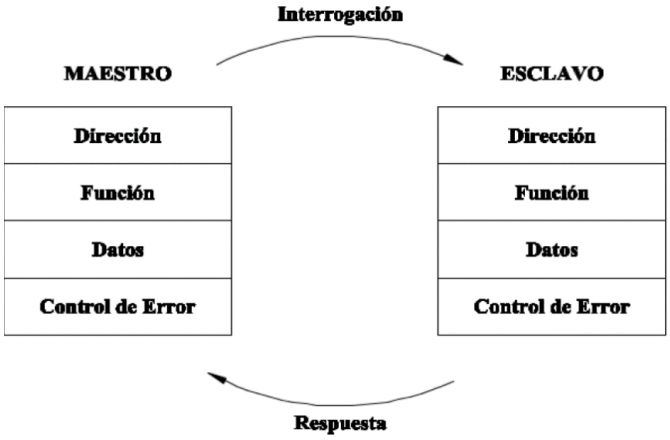


Figura 13- Interrogación Respuesta MODBUS. *[7]*

6.6. Estructura de los mensajes Modbus.

La interfaz de comunicación Modbus se construye alrededor de los mensajes. El formato de estos mensajes Modbus es independiente del tipo de interfaz física utilizada. El RS232 llanura de edad son los mismos mensajes usados como Modbus / TCP a través de Ethernet esto da a la definición de la interfaz Modbus una vida muy larga, el mismo protocolo se puede utilizar independientemente del tipo de conexión. Debido a esto, Modbus da la posibilidad de actualizar fácilmente la estructura de hardware de una red industrial, sin la necesidad de grandes cambios en el software.

El dispositivo también puede comunicarse con varios nodos de Modbus a la vez, incluso si están conectados con diferentes tipos de interfaz, sin necesidad de utilizar un protocolo diferente para cada conexión. Al utilizar los sistemas de red más versátil, como TCP / IP sobre Ethernet, Modbus los mensajes están integrados en los paquetes con el formato necesario para la interfaz física. En ese caso, Modbus y otros tipos de conexiones pueden coexistir en la misma interfaz física, al mismo tiempo. Aunque la estructura de Modbus principal mensaje es "peer-to-peer", Modbus es capaz de funcionar tanto en punto a punto y redes multipunto.

Cada mensaje Modbus tiene la misma estructura cuatro elementos básicos están presentes en cada mensaje la secuencia de estos elementos es el mismo para todos los mensajes, para que sea fácil de analizar el contenido del mensaje Modbus. Una conversación siempre es iniciada por un maestro en la red Modbus.

Un *Modbus maestro* envía un mensaje y dependiendo del contenido del mensaje-*un esclavo* toma acción y responde a ella, no puede haber más de un maestro en una red Modbus. Abordando en el encabezado del mensaje se utiliza para definir el dispositivo que debe responder a un mensaje. Todos los demás nodos de la red Modbus ignorar el mensaje de si el campo de la dirección no coincide con su propia dirección.
Según como se muestra en tabla 1 en las interfaces simples como RS485 o RS232, Modbus los mensajes se envían en forma de plano por la red. En este caso, la red está dedicada a Modbus. *[7]*

Tabla 1- Estructura de los mensajes Modbus. *[7]*

Campo	Descripción
Dirección de dispositivo	Dirección del receptor
Código de función	Código de definir el tipo de mensaje
Datos	Bloque de datos con información adicional
Control de errores	El valor numérico de verificación para detectar errores de comunicación

Modbus serial

Es utilizado como servidor de protocolo de comunicación Modbus vía RS-232 o RS- 422/485, se basa en arquitectura maestro / esclavo una dirección que van desde 1 hasta 247, se asigna a cada dispositivo esclavo. Sólo un maestro está conectado al bus en cualquier momento dado. Dispositivos esclavos, no transmiten información a menos que una solicitud es hecha por el dispositivo maestro y dispositivos esclavo no puede comunicarse con otros dispositivos esclavos. *[7]*

6.7. Modos de transmisión de serie Modbus

Los controladores pueden ser configurados para comunicar sobre redes standard Modbus utilizando cualquiera de los dos modos de transmisión: ASCII o RTU. Los usuarios seleccionan el modo deseado, junto con los parámetros de comunicación del puerto serie (velocidad, paridad, etc.), durante la configuración de cada controlador. El modo y los parámetros serie deben ser los mismos para todos los dispositivos conectados a una red Modbus.

La selección del modo ASCII o RTU tiene que ver únicamente con redes Modbus standard. Define los bits contenidos en los campos del mensaje transmitido en forma serie en esas redes determina cómo debe ser empaquetada y decodificada, la información en los campos del mensaje en otras rede como Modbus Plus, los mensajes Modbus son situados en tramas sin relación con la transmisión serie.*[7]*

6.7.1. Modo ASCII

Cuando los controladores se configuran para comunicar en una red Modbus según el modo ASCII (American Standard Code for Information Interchange), cada byte – 8 bits - en un mensaje se envía como dos caracteres ASCII. La principal ventaja de este modo es que permite intervalos de tiempo de hasta un segundo entre caracteres sin dar lugar a error.

El formato para cada byte en modo ASCII es: Sistema de codificación: Hexadecimal, caracteres ASCII 0-9, A-F. Un carácter hexadecimal contenido en cada carácter ASCII del mensaje. Bits por byte: 1 bit de arranque, 7 bits de datos, el menos significativo se envía primero, 1 bit para paridad Par o Impar, ningún bit para No paridad, 1 bit de paro si se usa paridad, bits si no se usa paridad. *[7]*

6.7.2. Modo RTU

Cuando los controladores son configurados para comunicar en una red Modbus usando el modo RTU (Remote Terminal Unit), cada byte de 8 bits en un mensaje contiene dos dígitos hexadecimales de 4 bits. La principal ventaja de este modo es que su mayor densidad de carácter permite mejor rendimiento que el modo ASCII para la misma velocidad. Cada mensaje debe ser transmitido en un flujo continuo.

El formato para cada byte en modo RTU es:

Sistema de codificación: Binario 8-bits, hexadecimal 0-9, A-F. Dos dígitos hexadecimales contenidos en cada campo de 8 bits del mensaje.
Bits por byte: 1 bit de arranque, 8 bits de datos, el menos significativo se envía primero, 1 bit para paridad Par o Impar; ningún bit para No paridad, 1 bit de paro si se usa paridad; 2 bits si no se usa paridad.
Campo de Comprobación de error: Comprobación Cíclica Redundante (CRC). *[7]*

6.7.3. Modbus plus

Modbus Plus (MB+), es una red de comunicación local de alta velocidad, para aplicaciones de control industrial, responde a una arquitectura cliente / servidor.
Las características de esta son:

- Velocidad de transmisión: 1 Megabit por segundo.
- Cada red soporta hasta 64 nodos o dispositivos direccionales.
- Distancia máxima utilizando repetidora: 1800m (6000ft)
- Medio físico de transmisión es el cable tipo par trenzado enmallado.
- La interfaz se basa en la norma RS-485

La red Modbus Plus estándar, soporta hasta 32 nodos y distancias de esta 450m (1500ft). La longitud puede ser hasta 2000m con el agregado de 3 repetidoras (amplificadores bidireccionales), los cuales son absolutamente “transparentes", para la red y para las aplicaciones sobre la misma, es decir, no constituyen nodos de red. La tensión de alimentación es independiente para cada dispositivo se trata de un protocolo con limitaciones y donde solo es recomendable usarlo en caso de instalaciones donde existan instalaciones de este tipo. *[7]*

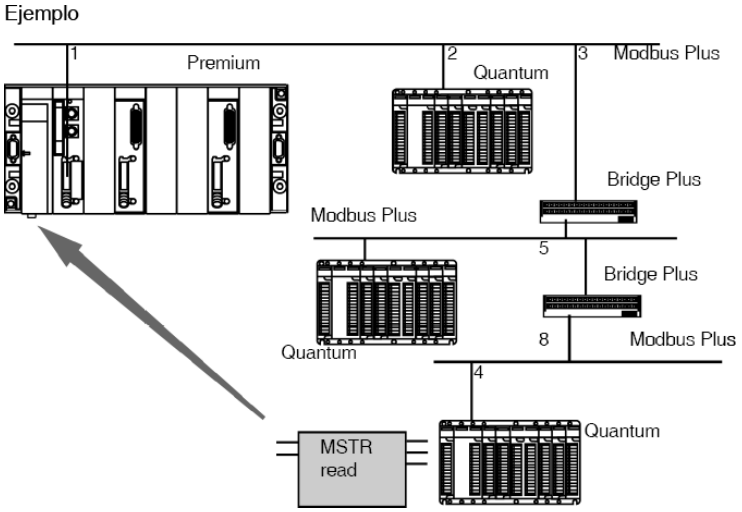


Figura 14- Interrogación Respuesta MODBUS PLUS. *[7]*

En la figura 14 se describe la estación Quantum emite una petición de lectura hacia la estación Premium utilizando una ruta de dirección: 8.5.1.0.0 (routing path). El módulo de función MSTR permite leer o escribir palabras internas de una estación Premium o Atrium.

El parámetro del registro esclavo del módulo de función MSTR indica directamente la dirección de la palabra interna %MW de la aplicación del autómata. Este módulo de función permite igualmente leer o poner a cero los contadores de estadísticas de una estación Premium o Micro. Esta petición se ejecuta a través de la tarjeta PCMCIA, ver detalles en tabla 2 a continuación. *[7]*

Tabla 2- Modbus Plus. *[7]*

Modbus Plus HECHOS	
Tipo de Red:	Maestro / Esclavo de bus de campo basado en RS-485 con fichas que pasa.
Topología:	Topología en línea con segmentos de hasta 32 estaciones.
Instalación:	Cable de par trenzado con 9-D-Sub de polo. Longitud de cable por segmento de hasta 500m ampliable con repetidores de hasta 2.000m.
Velocidad:	2 Mbit / s
Max. Estaciones:	64
Datos:	cíclica I / O y datos de parámetros acíclicos
Funciones de red:	Maestro / Esclavo de bus de campo de la red para aplicaciones de control Real-Time.
Organización de usuario:	Modbus-IDA Grupo de Usuarios

6.7.4. Modbus TCP

MODBUS TCP/IP es una variante o extensión del protocolo Modbus que permite utilizarlo sobre la capa de transporte TCP/IP. De este modo, Modbus-TCP se puede utilizar en Internet, de hecho, este fue uno de los objetivos que motivó su desarrollo (la especificación del protocolo se ha remitido a la IETF=Internet Engineering Task Force). En la práctica, un dispositivo instalado en Europa podría ser direccionado desde EE. UU. o cualquier otra parte del mundo.

MODBUS TCP/IP se ha convertido en un estándar industrial de facto debido a su simplicidad, bajo coste, necesidades mínimas en cuanto a componentes de hardware, y sobre todo a que se trata de un protocolo abierto. En la actualidad hay cientos de dispositivos MODBUS TCP/IP disponibles en el mercado. Se emplea para intercambiar información entre dispositivos, así como monitorizarlos y gestionarlos. También se emplea para la gestión de entradas/salidas

distribuidas, siendo el protocolo más popular entre los fabricantes de este tipo de componentes

La combinación de una red física versátil y escalable como Ethernet con el estándar universal de interredes TCP/IP y una representación de datos independiente de fabricante, como MODBUS, proporciona una red abierta y accesible para el intercambio de datos de proceso. *[7]*

Orientado a conexión.

MODBUS es un protocolo de comunicación sin estado, es decir, cada solicitud del maestro es tratada independientemente por el esclavo y es considerada una nueva solicitud no relacionada a las anteriores, de esta forma haciendo a las transacciones de datos altamente resistentes a rupturas debido a ruido y además requiriendo mínima información de recuperación para ser mantenida la transacción en cualquiera de los dos terminales.

Es interesante analizar por qué el protocolo TCP orientado a la conexión es usado en lugar del protocolo UDP orientado a datagramas. La principal razón es mantener control de una transacción individual encerrándola en una conexión la cual pueda ser identificada, supervisada, y cancelada sin requerir acción específica de parte de las aplicaciones cliente y servidor. Esto da al mecanismo una amplia tolerancia a cambios del desempeño de la red, y permite que herramientas de seguridad tal como firewalls y proxys puedan ser fácilmente añadidos. En adición, TCP permite establecer un gran número de conexiones concurrentes, de este modo el cliente (maestro) puede ya sea reusar una conexión previamente establecida ó crear una nueva, en el momento de realizar una transacción de datos. *[7]*

Codificación de datos.

MODBUS usa una representación. big-endian. Para direcciones y datos. Esto significa que cuando una cantidad numérica más grande que un byte es transmitido, el byte más significante es enviado primero. Así, por ejemplo: 0x1234 será 0x12 0x34.

Interpretación del modelo de datos.

MODBUS basa su modelo de datos sobre una serie de tablas las cuales tienen características distintivas. Las cuatro principales son:

- Entradas discretas. Bit simple, suministrado por un sistema I/O, de solo lectura.
- Salidas discretas. Bit simple, alterable por un programa de aplicación, de lectura-escritura.
- Registros de entrada. Cantidad de 16 bits, suministrado por un sistema I/O, de solo lectura.

- Registros de salida. Cantidad de 16 bits, alterable por un programa de aplicación, de lectura-escritura.

La distinción entre entradas y salidas y entre datos direccionables al bit y direccionables a la palabra, no implica algún comportamiento de la aplicación. Es aceptable y común, considerar las cuatro tablas sobrelapando una con otra, si esta es la interpretación más natural sobre la máquina (esclavo MODBUS) en cuestión. *[7]*

6.8. Ventajas del protocolo Modbus/TCP

- Realizar reparaciones o mantenimiento remoto desde la oficina utilizando un PC, reduciendo así los costes y mejorando el servicio al cliente.
- El ingeniero de mantenimiento puede entrar al sistema de control de la planta desde su casa, evitando desplazamientos.
- Permite realizar la gestión de sistemas distribuidos geográficamente mediante el empleo de las tecnologías de Internet/Intranet actualmente disponibles.
- Es escalable en complejidad. Un dispositivo el cual tiene solo un propósito simple necesita solo implementar uno ó dos tipos de mensaje.
- Es simple para administrar y expandir. No se requiere usar herramientas de configuración complejas cuando se añade una nueva estación a una red Modbus/TCP.
- No es necesario equipo o software propietario de algún vendedor. Cualquier sistema computador ó microprocesador con una pila de protocolos TCP/IP puede usar Modbus/TCP.
- Puede ser usado para comunicar con una gran base instalada de dispositivos MODBUS, usando productos de conversión los cuales no requieren configuración. *[7]*

6.9. Tramas del Mensaje de MODBUS

En cualquiera de los modos de transmisión serie (ASCII o RTU), un mensaje Modbus es situado por el dispositivo que transmite, en una trama que tiene un comienzo y un final conocidos. Esto permite a los dispositivos receptores comenzar en el arranque del mensaje, leer la parte de la dirección y determinar qué dispositivo es direccionado (o todos los dispositivos si es una difusión 'dirección = 0') y conocer cuándo se ha completado el mensaje.

Mensajes parciales pueden ser detectados y establecer errores como resultado. En redes como MAP o Modbus Plus, el protocolo de red manipula la trama de los mensajes con delimitadores de comienzo y final que son específicos de la red. Esos protocolos también manipulan el envío al dispositivo de destino, haciendo innecesario el campo de la dirección Modbus integrado en el mensaje para la transmisión actual. (La dirección modbus es convertida a una dirección de nodo de la red y enrutada por el controlador remitente o sus adaptadores de red. *[7]*

6.10. Estructura del protocolo

A continuación, se describe la forma general de encapsulación de una solicitud o respuesta MODBUS cuando es llevada sobre una red Modbus/TCP. Es importante anotar que la estructura del cuerpo de la solicitud y respuesta, desde el código de función hasta el fin de la porción de datos, tiene exactamente la misma disposición y significado como en las otras variantes MODBUS, tal como: MODBUS serial, Codificación ASCII MODBUS serial, Codificación RTU MODBUS PLUS

Las únicas diferencias en esos otros casos son la especificación de los delimitadores inicial y final del mensaje, el patrón de chequeo de error y la interpretación de la dirección. Todas las solicitudes son enviadas vía TCP sobre el puerto registrado 502. Las solicitudes normalmente son enviadas en forma half-duplex sobre una conexión dada. Es decir, no hay beneficio en enviar solicitudes adicionales sobre una única conexión mientras una respuesta está pendiente. Sin embargo, los dispositivos que desean obtener altas ratas de transferencia pueden establecer múltiples conexiones TCP al mismo destino.

El campo dirección esclavo de MODBUS es reemplazado por un byte identificador de unidad el cual puede ser usado para comunicar a través de dispositivos tales como puentes y gateways, los cuales usan una dirección IP única para soportar múltiples unidades terminales independientes. Los mensajes de solicitud y respuesta en Modbus/TCP poseen un prefijo ó encabezado compuesto por seis bytes como se aprecia en la figura15:

ref	ref	00	00	00	len

Figura 15- Estructura del prefijo de Modbus/TCP*. [7]*

El ref. ref anterior son los dos bytes del campo referencia de transacción, un número que no tiene valor en el servidor, pero son copiados literalmente desde la solicitud a la respuesta a conveniencia del cliente. Este campo se utiliza para que un cliente Modbus/TCP pueda establecer simultáneamente múltiples conexiones con diferentes servidores y pueda identificar cada una de las transacciones.

El tercer y cuarto campo del prefijo representa el identificador de protocolo, un número el cual debe ser establecido a cero. El LEN específica el número de bytes que siguen.

La longitud es una cantidad de dos bytes, pero el byte alto se establece a cero ya que los mensajes son más pequeños que 256. De esta forma, un mensaje Modbus/TCP completo posee una estructura como se muestra en la siguiente tabla #3: *[7]*

Tabla 3- Estructura del mensaje en Modbus/TCP. *[7]*

Posición del Byte	Significado
Byte 0	Identificador de transacción. Copiado por el servidor normalmente 0.
Byte 1	Identificador de transacción. Copiado por el servidor normalmente 0.
Byte 2	Identificador de protocolo = 0
Byte 3	Identificador de protocolo = 0
Byte 4	Campo de longitud (byte alto) =0. ya que los mensajes son menores a 256.
Byte 5	Campo de longitud (byte bajo). Número de bytes siguientes.
Byte 6	Identificador de unidad (previamente "Dirección Esclavo")
Byte 7	Código de función MODBUS.
Byte 8 y más	Los datos necesarios.

Esquema de Encapsulación del Protocolo MODBUS TCP Modbus/TCP simplemente encapsula una trama Modbus en un segmento TCP. TCP proporciona un servicio orientado a conexión fiable, lo que significa que toda consulta espera una respuesta ver figura 16 sobre la trama TCP*:*

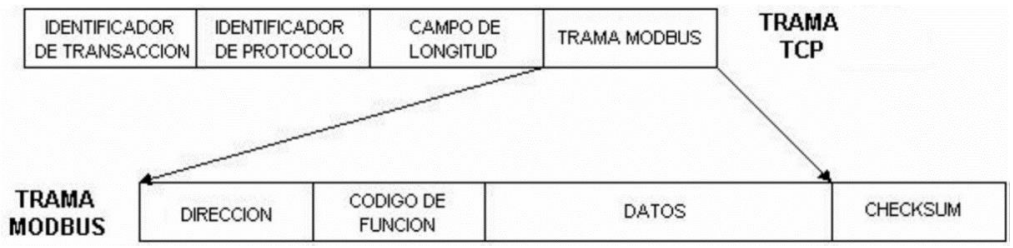


Figura 16- Encapsulamiento de la trama Modbus/TCP. *[7]*

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. *[7]*

6.11. Prestaciones de un sistema MODBUS TCP/IP

Las prestaciones dependen básicamente de la red y el hardware. Si se usa MODBUS TCP/IP sobre Internet, las prestaciones serán las correspondientes a 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 depuración y mantenimiento, evitando así desplazamientos al lugar de la instalación. Si disponemos de una Intranet de altas prestaciones con conmutadores Ethernet de alta velocidad, la situación es totalmente diferente. En teoría, MODBUS TCP/IP, transporta datos hasta 250/(250+70+70) o alrededor de un 60% de eficiencia cuando se trasfieren registros en bloque, y puesto que 10 Base T proporciona unos 1.25 Mbps de datos, la velocidad de transferencia de información útil será:

1.25M / 2 * 60% = 360000 registros por Segundo
En 100BaseT la velocidad es 10 veces mayor.

Esto suponiendo que se están empleando dispositivos que pueden dar servicio en la red Ethernet aprovechando todo el ancho de banda disponible.

En los ensayos prácticos realizados por by Schneider Automation utilizando un PLC Ethernet MomentumTM con entradas/salidas Ethernet, demostró que se podían escanear hasta 4000 bloques I/O por segundo, cada uno con hasta 16 I/O analógicas de 12-bits o 32 I/O digitales (se pueden actualizar 4 bases por milisegundo). Aunque estos resultados están por debajo del límite teórico calculado anteriormente, pero debemos recordar que el dispositivo se probó con una CPU de baja velocidad (80186 a 50MHz con 3 MIPS). Además, el abaratamiento de los ordenadores personales y el desarrollo de redes Ethernet cada vez más rápidas, permite elevar las velocidades de funcionamiento, a diferencia de otros buses que están inherentemente limitados una sola velocidad. *[7]*

6.12. Tramas del Mensaje de MODBUS

En cualquiera de los modos de transmisión serie (ASCII o RTU), un mensaje Modbus es situado por el dispositivo que transmite, en una trama que tiene un comienzo y un final conocidos. Esto permite a los dispositivos receptores comenzar en el arranque del mensaje, leer la parte de la dirección y determinar qué dispositivo es direccionado (o todos los dispositivos si es una difusión 'dirección = 0') y conocer cuándo se ha completado el mensaje. Mensajes parciales pueden ser detectados y establecer errores como resultado. En redes como MAP o Modbus Plus, el protocolo de red manipula la trama de los mensajes con delimitadores de comienzo y final que son específicos de la red. Esos protocolos también manipulan el envío al dispositivo de destino, haciendo innecesario el campo de la dirección Modbus integrado en el mensaje para la transmisión actual. (La dirección modbus es convertida a una dirección de nodo de la red y enrutada por el controlador remitente o sus adaptadores de red. *[7]*

Trama ASCII

En modo ASCII, los mensajes comienzan con un carácter (:) 'dos puntos' (ASCII 3A hex) y terminan con un par de caracteres (CRLF) 'Retorno de Carro + Avance de Línea) (ASCII 0D hex y 0A hex). Los caracteres a transmitir permitidos para todos los demás campos son 0-A, A-F hexadecimal.

Los dispositivos conectados en red monitorizan el bus de red continuamente para detectar un carácter 'dos puntos'. Cuando se recibe, cada dispositivo decodifica el próximo campo (el campo de dirección) para enterarse si es el dispositivo direccionado.

Puede haber intervalos de hasta un segundo entre caracteres dentro del mensaje, si transcurre más tiempo entre caracteres, el dispositivo receptor asume que ha ocurrido un error. Se muestra debajo en tabla 4 una trama de mensaje típica.

Tabla 4- Trama de Mensaje ASCII. *[7]*

ARRANQUE	DIRECCIÓN	FUNCIÓN	DATOS	COMP. LRC	FINAL
1 Carácter	2 Caracteres	2 carácter	N caracteres	2 caracteres	2 carácter CRLF

Excepción: Con los controladores 584 y 984A/B/X, un mensaje ASCII puede terminar normalmente después del campo LRC sin enviar los caracteres CRLF.

En ese caso, debe tener lugar una pausa de al menos 1 segundo. Si esto sucede, el controlador asumirá que el mensaje ha terminado normalmente. *[7]*

Trama RTU

En modo RTU, los mensajes comienzan con un intervalo silencioso de al menos 3.5 tiempos de carácter. Esto es más fácilmente implementado como un múltiplo de tiempos de carácter a la velocidad de transmisión configurada en la red (mostrado como T1-T2- T3-T4 en la figura trama RTU).

El primer campo transmitido es entonces la dirección del dispositivo destinatario. Los caracteres a transmitir permitidos para todos los campos son 0-A, A-F hexadecimal.
Los dispositivos conectados en red monitorizan el bus de red continuamente incluso durante los intervalos 'silencioso'. Cuando el primer campo (el campo de dirección) es recibido, cada Dispositivo lo decodifica para enterarse si es el dispositivo direccionado.

Siguiendo al último carácter transmitido, un intervalo de al menos 3.5 tiempos de carácter señala el final del mensaje. Un nuevo mensaje puede comenzar después de este intervalo. *[7]*

La trama completa del mensaje debe ser transmitida como un flujo continuo. Si un intervalo silencioso de más de 1.5 tiempos de carácter tiene lugar antes de completar la trama, el dispositivo receptor desecha el mensaje incompleto y asume que el próximo byte será el campo de dirección de un nuevo mensaje.

De forma similar, si un nuevo mensaje comienza antes de que transcurran 3.5 tiempos de carácter después de un mensaje previo, el dispositivo receptor lo considerará una continuación del mensaje previo. Esto dará lugar a un error, ya que el valor en el campo final CRC no será válido para el mensaje combinado. Debajo se muestra tabla 5 de una trama de mensaje típica ver tabla #5. *[7]*

Tabla 5- Trama de Mensaje RTU. *[7]*

ARRANQUE	DIRECCIÓN	FUNCIÓN	DATOS	COMPROB LRC	FINAL
T1-T2-T3-T4	8 BITS	8 BITS	N & 8 BITS	16 BITS	T1-T2-T3-T4

Cómo es Manipulado el Campo Dirección

El campo dirección de un mensaje contiene dos caracteres (ASCII) u ocho bits (RTU). Las direcciones de esclavo válidas están en el rango de 0 – 247 decimales. Los dispositivos esclavos individuales tienen direcciones asignadas en el rango 1 – 247. Un maestro direcciona un esclavo situando la dirección del esclavo en el campo dirección del mensaje. Cuando el esclavo envía su respuesta, sitúa su propia dirección en este campo dirección de la respuesta para dar a conocer al maestro qué esclavo está respondiendo.

La dirección 0 es utilizada para dirección difusión, la cual todos los dispositivos esclavos reconocen. Cuando el protocolo Modbus es usado en redes de nivel más alto, las difusiones pueden no estar permitidas o pueden ser reemplazadas por otros métodos. *[7]*

Cómo es Manipulado el Campo Función

El campo código de función de una trama de mensaje contiene dos caracteres (ASCII) u ocho bits (RTU). Los códigos válidos están en el rango de 1 – 255 decimal. De esos, algunos códigos son aplicables a todos los controladores Modicon, mientras que algunos códigos se aplican sólo en algunos modelos y otros están reservados para usos futuros. Cuando un mensaje es enviado desde un maestro a un dispositivo esclavo, el campo del código de función indica al esclavo qué tipo de acción ha de ejecutar. Por ejemplo: leer los estados ON/OFF de un grupo bobinas o entradas discretas; leer el contenido de datos de un grupo de registros; leer el estatus de diagnóstico de un esclavo; escribir en determinadas bobinas o registros; o permitir cargar, salvar o verificar el programa dentro del esclavo.

Cuando el esclavo responde al maestro, utiliza el campo del código de función para indicar bien una respuesta normal (libre de error) o que algún tipo de error ha tenido lugar (denominado respuesta de excepción). Para una respuesta normal, el esclavo simplemente replica el código de función original. Para una respuesta de excepción, el esclavo devuelve un código que es equivalente al código de función original con su bit más significativo puesto a valor 1. Por ejemplo, un mensaje desde un maestro a un esclavo para leer un grupo de registros mantenidos tendría el siguiente código de función:

0000 0011 (Hexadecimal 03)
Si el dispositivo esclavo ejecuta la acción solicitada, sin error, devuelve el mismo código en su Respuesta. Si ocurre una excepción. Devuelve: 1000 0011 (Hexadecimal 83).

Además de la modificación del código de función para una respuesta de excepción, el esclavo sitúa un único código en el campo de datos el mensaje respuesta. Esto indica al maestro qué tipo de error ha tenido lugar, o la razón para la excepción. El programa de aplicación del maestro tiene la responsabilidad de manejar las respuestas de excepción. Procedimientos típicos son: enviar subsiguientes reintentos de mensaje, intentar mensajes de diagnóstico al esclavo y notificar operadores. *[7]*

Contenido del Campo Datos

El campo datos se construye utilizando conjuntos de 2 dígitos hexadecimales, en el rango de 00 a FF hexadecimal. Pueden formarse a partir de un par de caracteres ASCII o desde un carácter RTU, de acuerdo con el modo de transmisión serie de la red. El campo datos de los mensajes enviados desde un maestro a un esclavo, contiene información adicional que el esclavo debe usar para tomar la acción definida por el código de función. Esto puede incluir partes como direcciones discretas y de registros, la cantidad de partes que han de ser manipuladas y el cómputo de bytes de datos contenidos en el campo.

Por ejemplo, si el maestro solicita a un esclavo leer un grupo de registros mantenidos (código de función 03), el campo de datos especifica el registro de comienzo y cuántos registros han de ser leídos. Si el maestro escribe sobre un grupo de registros en el esclavo (código de función 10 hexadecimal), el campo datos especifica el registro de comienzo, cuántos registros escribir, el cómputo de bytes de datos que siguen en el campo datos y los datos que se deben escribir en los registros. Si no ocurre error, el campo datos de una respuesta desde un esclavo al maestro contiene los datos solicitados. Si ocurre un error, el campo contiene un código de excepción que la aplicación del maestro puede utilizar para determinar la próxima acción a tomar. El campo datos puede ser inexistente (de longitud cero) en ciertos tipos de mensajes. *[7]*

6.13. Contenido del Campo Comprobación de Error

Dos tipos de métodos de comprobación de error son utilizados para las redes Modbus Standard, el contenido del campo Comprobación de Error depende del método que esté siendo utilizado. *[7]*

6.13.1. En modo ASCII

Cuando el modo ASCII es usado para trama de carácter, el campo Comprobación de Error Contiene dos caracteres ASCII. Los caracteres de comprobación de error son el resultado de un cálculo Comprobación Longitudinal Redundante (LRC) que es realizado sobre el contenido del mensaje, excluyendo los 'dos puntos' del comienzo y los caracteres CRLF de finalización. Los caracteres LRC son añadidos al mensaje como el último campo que precede a los caracteres **CRLF.** *[7]*

6.13.2. En modo RTU

Cuando el modo RTU es usado para trama de carácter, el campo Comprobación de Error Contiene un valor de 16 bits implementado como dos bytes de 8 bits. El valor de comprobación de error es el resultado de un cálculo Comprobación Cíclica Redundante (CRC) realizado sobre el contenido del mensaje. El campo CRC es añadido al mensaje como último campo del mensaje. La forma de hacerlo es, añadir primero el byte de orden bajo del campo, seguido del byte de orden alto. El byte de orden alto del CRC es el último byte a enviar en el mensaje. *[7]*

6.13.3. Transmitidos los Caracteres en Serie

Cuando los mensajes son transmitidos sobre redes serie standard Modbus, cada carácter o byte es enviado en este orden (izquierda a derecha):

Bit Menos Significativo (LSB)... Bit Más Significativo (MSB) Con trama de carácter ASCII y RTU, la secuencia de bit es según como se ilustra en figura 17.

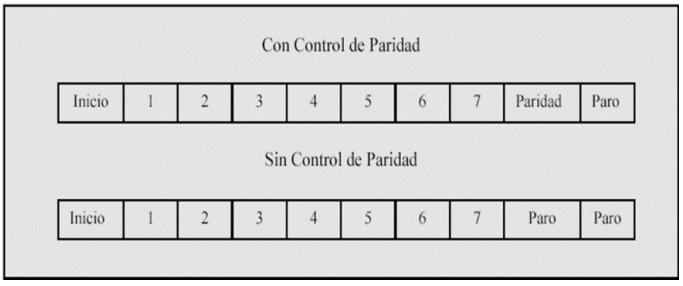


Figura 17- Orden de bits (ASCII) y (RTU). *[7]*

6.13.4. Descripción de las Funciones del Protocolo

En código ASCII, esta palabra es simplemente la suma de comprobación (‘checksum’) del mensaje en módulo 16 expresado en ASCII, la descripción de cada código se detalla en tabla #6. En el caso de codificación RTU el CRC se calcula con una fórmula polinómica según el algoritmo. *[7]*

Tabla 6- Descripción de las Funciones del Protocolo. *[7]*

Función	Código	Tarea
0	00_H	Control de estaciones esclavas
1	01 H	Lectura de n bits de salidas 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 diagnóstico número 1 a 8.
9	09 H	No utilizado
10	0A H	No utilizado
11	0B H	Control de contadores de diagnóstico 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.

6.13.5. Funciones básicas y códigos de operación

Esta función permite ejecutar órdenes de control, tales como marcha, paro, carga y lectura de programas de usuario del autómata. Para codificar cada una de las citadas órdenes se emplean los cuatro primeros bytes del campo de datos.

En caso de las órdenes de marcha y paro, el campo de información de la trama representada en la tabla 7 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 autómata sólo acepta ejecutar subfunciones de la función 00H. *[7]*

Tabla 7- Trama genérica subfunciones control esclavos (cód. Función 00H). *[7]*

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

6.13.6. Métodos para comprobación de errores

Las redes series standard Modbus utilizan dos tipos de comprobación de error. La comprobación de paridad (par o impar) puede ser aplicada opcionalmente a cada carácter. La comprobación de la trama (LRC o CRC) es aplicada al mensaje completo. Ambas comprobaciones, de carácter y de trama de mensaje son generadas en el dispositivo maestro y aplicadas a los contenidos del mensaje antes de la transmisión. El dispositivo esclavo comprueba cada carácter y la trama del mensaje completo durante la recepción.

El maestro es configurado por el usuario para aguardar durante un tiempo de espera predeterminado antes de abortar la transacción. Este intervalo es establecido para ser lo suficientemente largo para que cualquier esclavo responda normalmente. Si el esclavo detecta un error de transmisión, el mensaje no será tenido en cuenta. El esclavo no construirá una respuesta para el maestro. Así el tiempo de espera expirará y permite al programa del maestro tratar el error.

Otras redes tales como MAP y Modbus Plus utilizan comprobación de trama a un nivel por encima del contenido Modbus del mensaje. En esas redes, el campo de comprobación LRC o CRC del mensaje Modbus no se aplica. En caso de error de transmisión, el protocolo de Comunicación específico a esas redes notifica al dispositivo que inició la comunicación que ha ocurrido un error y le permite reintentar o abortar de acuerdo a cómo ha sido configurado. *[7]*

6.13.7. Control de Paridad

Los usuarios pueden configurar los controladores para Control de paridad Par o Impar, o Sin Control de paridad. Esto determinará cómo será iniciado el bit de paridad en cada carácter.

Si se especifica cualquier control de paridad Par o Impar, se contabilizará la cantidad de bits que tienen valor 1 en la porción de datos de cada carácter (siete bits de datos para modo ACSII, u ocho para RTU). Al bit de paridad habrá de darse valor 0 o 1, para que se obtenga finalmente un número par o impar, respectivamente, de bits con valor 1.

Por ejemplo, estos 8 bits de dato forman parte de una trama de carácter RTU: 1100 0101 La cantidad de bits de valor 1 en el dato es cuatro. Si se utiliza Control de Paridad Par, el bit de paridad de la trama debe establecerse a valor 0, haciendo que la cantidad de bits de valor 1 siga siendo un número par (cuatro). Si se utiliza Control de Paridad Impar, el bit de paridad deberá tener valor 1, resultando una cantidad de bits de valor 1, impar (cinco). Cuando el mensaje es transmitido, el bit de paridad es calculado y aplicado a la trama de cada carácter. El dispositivo receptor cuenta la cantidad de bits de valor 1 y establece un error si no coincide la paridad con la configurada para ese dispositivo (todos los dispositivos en la red Modbus deben ser configurados para usar el mismo método de Control de paridad).

La comprobación de paridad sólo detecta si un número impar de bits se han alterado en una trama de carácter durante la transmisión. Por ejemplo, si se utiliza control de paridad Impar y dos bits de valor 1 de un carácter que tiene en origen 3 bits con valor 1, han quedado falseados (pasan a valor 0) durante la transmisión, el resultado es todavía un cómputo impar de bits de valor 1 (y por lo tanto el error no es detectado por este método). Si se especifica control No Paridad, no se transmite bit de paridad y no se hace comprobación de paridad. Se transmite un bit de paro adicional para rellenar la trama de carácter. *[11]*

6.13.8. Comprobación LRC

En modo ASCII, los mensajes incluyen un campo de comprobación de error que está basado en un método de Comprobación Longitudinal Redundante (LRC). El campo LRC controla el contenido del mensaje, a excepción de los ‘:’ del comienzo y el par CRLF. Es aplicado con independencia de cualquier método de control de paridad utilizado para los caracteres individuales del mensaje.
El campo LRC es un byte, conteniendo un valor binario de ocho bits. El valor LRC es calculado por el dispositivo emisor, que añade el LRC al mensaje. El dispositivo receptor calcula el LRC durante la recepción del mensaje y compara el valor calculado con el valor recibido en el campo LRC. Si los dos valores no son iguales, resulta un error.

El valor LRC se calcula sumando entre sí los sucesivos bytes del mensaje, descartando cualquier acarreo y luego complementando a dos el valor resultante. Se realiza sobre el contenido del campo de mensaje ASCII excluyendo el carácter ‘:’ de comienzo del mensaje y excluyendo el par CRLF de final de mensaje. En la lógica de programación de controladores, la función CKSM calcula el LRC en base al contenido del mensaje. *[11]*

6.13.9. Comprobación CRC

En modo RTU, los mensajes incluyen un campo de comprobación de error que está basado en un método Comprobación de Redundancia Cíclica (CRC). El campo CRC controla el contenido del mensaje completo. Se aplica con independencia de cualquier método de control de paridad utilizado para los caracteres individuales del mensaje.

El campo CRC es de dos bytes, conteniendo un valor binario de 16 bits. El valor CRC es calculado por el dispositivo emisor, que añade el CRC al mensaje. El dispositivo receptor calcula el CRC durante la recepción del mensaje y compara el valor calculado con el valor recibido en el campo CRC. Si los dos valores no son iguales, resulta un error.

Para calcular el valor CRC Modbus se precarga un registro de 16 bits, todos ellos a 1. Luego comienza un proceso que toma los sucesivos bytes del mensaje y los opera con el contenido del registro y actualiza éste con el resultado obtenido. Sólo los 8 bits de dato de cada carácter son utilizados para generar el CRC. Los bits de arranque y paro y el bit de paridad, no se tienen en cuenta para el CRC.

Durante la generación del CRC, se efectúa una operación booleana OR exclusivo (XOR) a cada carácter de 8 bits con el contenido del registro. Entonces al resultado se le aplica un desplazamiento de bit en la dirección de bit menos significativo (LSB), rellenando la posición del bit más significativo (MSB) con un cero. El LSB es extraído y examinado. Si el LSB extraído fuese un 1, se realiza un XOR entre el registro y un valor fijo preestablecido (*). Si el LSB fuese un 0, no se efectúa un el XOR. Este proceso es repetido hasta haber cumplido 8 desplazamientos. Después del último desplazamiento (el octavo), el próximo byte es operado XOR con el valor actual del registro y el proceso se repite con ocho desplazamientos más, como se ha descrito más arriba y así con todos los bytes del mensaje.

El contenido final del registro, después de que todos los bytes del mensaje han sido procesados, es el valor del CRC. Cuando el CRC es añadido al mensaje, primero se añade el byte de orden bajo seguido del byte de orden alto. *[11]*

Modbus maestro/esclavo

El usuario establece para cada producto “esclavo” conectado a la red Modbus un número identificativo del 1 al 247 denominado dirección Modbus.

El “maestro”, por ejemplo, un servidor Web integrado en un armario eléctrico, consulta simultáneamente todos los productos con un mensaje en el que se incluye la dirección de destino, el código de función, la ubicación de la memoria en el producto y la cantidad de información (253 bytes máximo).

Sólo un producto con la dirección correspondiente responde a la solicitud de datos. El intercambio sólo se lleva a cabo con la iniciativa del maestro (en este caso, el servidor Web): se trata del procedimiento operativo Modbus maestro-esclavo o detalle de modelo OSI descrito en figura 18. *[7]*

Layer	ISO/OSI Model	
7	Application	MODBUS Application Protocol
6	Presentation	Empty
5	Session	Empty
4	Transport	Empty
3	Network	Empty
2	Data Link	MODBUS Serial Line Protocol
1	Physical	EIA/TIA-485 (or EIA/TIA-232)

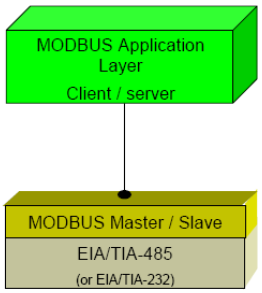


Figura 18- Protocolo MODBUS y Modelo ISO/OSI. *[7]*

6.14. Tablas de registros MODBUS

Las normas de presentación de los registros Modbus son:

Los registros están agrupados de acuerdo con el módulo con el que están relacionados en dependencia y a elección de cada fabricante. Como una forma básica de ordenamiento podría ser: Unidad de control, Módulos de entrada y salidas, Módulos lógicos y resultados, Interfaz e interacción con equipo, Escrituras de datos.

Para cada módulo, los registros están agrupados en tablas de información relacionada lógicamente. Las tablas se presentan ordenando las direcciones de menor a mayor. *[12]*

Tabla 8- Formato de tabla habitual. *[12]*

Dirección	Registro	LE	X	Unidad	Tipo	Rango	A/E	Descripción

- **Dirección:** una dirección de registro de 16 bits en formato hexadecimal. La dirección responde a los datos utilizados en la trama Modbus.
- **Registro:** un número de registro de 16 bits en formato decimal (registro = dirección + 1).
- **RW:** estado del registro de lectura-escritura.
- **L:** el registro puede leerse mediante las funciones Modbus
- **E**: puede escribirse en el registro mediante las funciones Modbus.
- **LE:** el registro puede leerse y puede escribirse en él mediante las funciones Modbus.
- **LC:** el registro puede leerse por medio de la interfaz de comandos.
- **EC:** puede escribirse en el registro por medio de la interfaz de comandos.

- **X:** el factor de escala. Una escala de 10 significa que el registro contiene el valor multiplicado por 10. Así, el valor real es el valor del registro dividido por 10.

Ejemplo: El registro 1054 contiene la frecuencia del sistema. La unidad es Hz y el factor de escala es 10.

- Si el registro devuelve 503, esto significa que la frecuencia del sistema es 503/10 = 50,3 Hz.
- Unidad: la unidad en la que se expresa la información.
- Tipo: tipo de datos de codificación (consulte la descripción de los tipos de datos a continuación).
- Rango: los valores permitidos para esta variable, normalmente un subconjunto de lo que permite el formato.
- A/E: tipo de medida de la unidad de control
- Tipo A (Amperímetro): medidas de corriente
- Tipo E (Energía): medidas de corriente, tensión, alimentación y energía
- Descripción: proporciona información sobre el registro y las restricciones que se aplican.

Tabla 9- Tipos de datos. *[12]*

Tipos de datos	Descripción	Rango
INT16U	Entero sin signo de 16 bits	Entre 0 y 65535
INT16	Entero con signo de 16 bits	Entre –32768 y +32767
INT32U	Entero sin signo de 32 bits	De 0 a 4294967295
INT32	Entero con signo de 32 bits	Entre –2147483648 y +2147483647
INT64	Entero con signo de 64 bits	Entre -9223372036854775808 y +9223372036854775807
FLOAT32	Entero con signo de 32 bits con un punto flotante	2^{-126} (1.0) y 2^{127} (2 - 2^{-23})
OCTET STRING	Cadena de texto	1 byte por carácter
DATETIME	Fecha y hora en formato IEC 60870-5	–
ULP DATE	Fecha y hora en formato ULP	–

Definiendo la tabla 09 se inicia explicando un Formato Big-Endian

Las variables INT32, INT32U, INT64 e INT64U se almacenan en formato big-endian: el registro más significativo se transmite en primer lugar y el menos significativo en último lugar.

Las variables INT32, INT32U, INT64 e INT64U están formadas por variables INT16U.

Las fórmulas para calcular el valor decimal de estas variables son:

- INT32: (0-bit31)x231 + bit30x230 + bit29x229 + ...bit1x21 + bit0x20
- INT32U: bit31x231 + bit30x230 + bit29x229 + ...bit1x21 + bit0x20
- INT64: (0-bit63)x263 + bit62x262 + bit61x261 + ...bit1x21 + bit0x20
- INT64U: bit63x263 + bit62x262 + bit61x261 + ...bit1x21 + bit0x20.

Ejemplo 1: La energía activa total del conjunto de datos estándar es una variable INT64 codificada en los registros 32096 a 32099.

Si los valores de los registros son:

- Registro 32096 = 0
- Registro 32097 = 0
- Registro 32098 = 0x0017 o 23
- Registro 32099 = 0x9692 o 38546 como variable INT16U y -26990 como variable INT16 (use el valor INT16U para calcular el valor de la energía activa total).

Entonces, la energía activa total es igual a 0x248 + 0x232 + 23x216 + 38546x20 = 1545874 Wh.

Ejemplo 2: La energía reactiva del conjunto de datos heredado es una variable INT32 codificada en los registros 12052 a 12053.

Si los valores de los registros son:

- Registro 12052 = 0xFFF2 = 0x8000 + 0x7FF2 o 32754
- Registro 12053 = 0xA96E o 43374 como variable INT16U y -10606 como variable INT16 (use el valor INT16U para calcular el valor de la energía reactiva).

Entonces, la energía reactiva es igual a (0-1)x231 + 32754x216 + 43374x20 = -874130 kVARh. *[12]*

Tipo de datos: FLOAT32:

El tipo de dato FLOAT32 se representa en la precisión única IEEE 754 (IEEE estándar para la aritmética de coma flotante). Un valor N se calcula como se muestra en tabla 10 y a continuación: *[12]*

N = (-1) S x 2E-127 x (1+M)

Tabla 10- Tipo de datos: FLOAT32. *[12]*

Coeficiente	Significa	Descripción	Número de bits
S	Señal	Define la señal del valor: 0 = positivo 1 = negativo	1 bit
E	Exponente	Entero binario 127 de exceso añadido. Si 0 < E < 255, el exponente real es: e = E - 127.	8 bits
M	Mantisa	Significante binario normalizado de magnitud	23 bits

Ejemplo:

0 = 0 00000000 00000000000000000000000

-1.5 = 1 01111111 10000000000000000000000

con:

S = 1

E = 01111111 = 127

M = 10000000000000000000000 = 1x2-1 + 0x2-2 +...+ 0x2-23 = 0.5

N = (-1) x 20 x (1+0.5) = -1.

Tipo de datos: DATETIME: DATETIME es un tipo de datos que permite codificar la fecha y hora definidas según el estándar IEC 60870-5 cada registro se explica en tabla 11 a continuación: *[12]*

Tabla 11- DATETIME. *[12]*

Registro	Tipo	Bit	Rango	Descripción
1	INT16U	0-6	0x00–0x7F	Año: Entre 0x00 (00) y 0x7F (127) corresponde a los años entre 2000 y 2127.
		7-15	–	Reservado
2	INT16U	0-4	0x01–0x1F	Día
		5-7	–	Reservado
		8-11	0x00–0x0C	Mes
		12-15	–	Reservado
3	INT16U	0-5	0x00–0x3B	Minutos

		6-7	–	Reservado
		8-12	0x00–0x17	Horas
		13-15	–	Reservado
4	INT16U	0-15	0x0000–0xEA5F	Milisegundos

Calidad de marcas de tiempo DATETIME: La calidad de las marcas de tiempo codificadas con el tipo de datos DATETIME puede indicarse en el registro que sigue a los 4 registros de la marca de tiempo. En este caso, la calidad de la marca de tiempo se codifica de según la descripción de tabla 12:

Tabla 12- Calidad marca de tiempo. *[12]*

Bit	Descripción
0-11	Reservado
12	Sincronización externa: **o**0 = No válida **o**1 = Válida
13	Sincronización: **o**0 = No válida **o**1 = Válida
14	Fecha y hora configuradas: **o**0 = No válida **o**1 = Válida
15	Reservado

Calidad de bits en registros: La calidad de cada bit de un registro codificado con el tipo de datos INT16U como una enumeración de bits puede indicarse en el registro precedente al registro en cuestión.

Ejemplo: La calidad de cada bit del registro 32001, el estado del interruptor automático se proporciona en el registro precedente, el 32000.

La calidad de los datos correspondientes al bit 0 del registro 32001, el contacto de señalización de estado OFF, se proporciona en el bit 0 del registro 32000, varios casos en tabla #13.

- bit 0 del registro 32000 = calidad de la indicación de estado OFF
- bit 0 del registro 32001 = contacto de indicación de estado OFF. *[12]*

Tabla 13- Calidad de cada bit del registro. *[12]*

Si	Entonces
Si el bit 0 del registro 32000 = 1 Y el bit 0 del registro 32001 = 0	El contacto OFF indica que el dispositivo está abierto.
Si el bit 0 del registro 32000 = 1 Y el bit 0 del registro 32001 = 1	El contacto OFF indica que el dispositivo está cerrado.
Si el bit 0 del registro 32000 = 0	La indicación del contacto OFF no es válida.

Tipo de datos: ULP DATE: ULP DATE es un tipo de datos utilizado para codificar fechas y horas. En tabla 14 se presenta el tipo de datos ULP DATE. *[12]*

Tabla 14- Tipo de datos: ULP DATE. *[12]*

Registro	Tipo	Bit	Rango	Descripción
1 2	INT32U	–	0x00000000– 0xFFFFFFFF	Número de segundos desde el 1 de enero de 2000
3	INT16U	–	–	Complemento en milisegundos
		0–9	–	Codifica los milisegundos
		10-11	–	No se utiliza
		12	0–1	Estado de sincronización externa de la interfaz de comunicación IFM o IFE 0 = La interfaz de comunicación no se ha sincronizado externamente en las últimas 2 horas. 1 = La interfaz de comunicación se ha sincronizado externamente en las últimas 2 horas.
		13	0–1	Estado de sincronización interna del módulo ULP 0 = El módulo ULP no se ha sincronizado internamente. 1 = El módulo ULP se ha sincronizado internamente.
		14	0–1	La fecha absoluta se establece desde el último encendido. 0 = No 1 = Sí
		15	–	Reservado

Contador de fecha ULP: La fecha en formato ULP se cuenta en número de segundos desde el 1 de enero de 2000. En caso de pérdida de alimentación de un módulo IMU, el contador de hora se restablece y se reiniciará en el 1 de enero de 2000. Si se produce una sincronización externa después de una pérdida de alimentación, el contador de hora se actualiza y convierte la fecha de sincronización al número correspondiente en segundos desde el 1 de enero de 2000. *[12]*

Principio de conversión de fecha ULP: Para convertir la fecha de número de segundos desde el 1 de enero de 2000 a la fecha actual, se aplican estas normas:

- 1 año no bisiesto = 365 días. ⁄1 año bisiesto = 366 días

Los años 2000, 2004, 2008, 2012... (múltiplos de 4) son años bisiestos (excepto el año 2100).

- 1 día = 86.400 segundos. ⁄ 1 hora = 3.600 segundos. ⁄ 1 minuto = 60 segundos

En la tabla 15 siguiente se describen los pasos que se deben seguir para convertir la fecha de número de segundos desde el 1 de enero de 2000 a la fecha actual y en tabla 16 se detallan los valores no aplicables para el ULP:

Tabla 15- Principio de conversión de fecha ULP. *[12]*

Paso	Acción
1	Calcular el número de segundos desde el 1 de enero de 2000: S = (contenido del registro 1 x 65536) + (contenido del registro 2)
2	Calcular el número de días desde el 1 de enero de 2000: D = valor entero del cociente de S / 86.400 Calcular el número restante de segundos: s = S - (D x 86,400
3	Calcular el número de días pasados del presente año: d = D - (NL x 365) - (L x 366) con NL = número de años no bisiestos desde el año 2000 y L = número de años bisiestos desde el año 2000
4	Calcular el número de horas: h = valor entero del cociente de s / 3600 Calcular el número restante de segundos: s' = s - (h x 3600)
5	Calcular el número de minutos: m = valor entero del cociente de s' / 60 Calcular el número restante de segundos: s'' = s' - (m x 60)
6	Calcular el número de milisegundos: ms = (contenido del registro 3) Y 0x03FF
7	Resultado: **O** La fecha actual es la fecha = d + 1. Por ejemplo, si d = 303, la fecha actual corresponde al día 304° del año, que corresponde al 31 de octubre 2007. **O** La hora actual es h:m:s'':ms

Tabla 16- Valores no aplicables Contador de fecha ULP. [12]

Tipo de datos	Valores no aplicables y fuera de servicio
INT16U	65535 (0xFFFF)
INT16	-32768 (0x8000)
INT32U	4294967295 (0xFFFFFFFF)
INT32	0x80000000
INT64U	0Xffffffffffffffff
INT64	0x8000000000000000
FLOAT32	0xFFC00000

6.15. Ejemplos de dispositivos con registros MODBUS.

Motor Logic® Plus II: Es un relé de protección de sobre corriente de estado sólido, se toma un extracto de algunos registros modbus de este equipo en tabla 17: [*13]*

Tabla 17- Tabla de registros Modbus Motorlogic*. [13]*

16 Bit Memory		Code and Description	Notes[1]
Hex	Register		
192	40403	WARNSTAT Warning status bits	Bit 0: Average current below UC warning threshold Bit 1: High phase above OC warning threshold Bit 2: Ground fault current above the GF warning threshold Bit 3: Current unbalance above the CUB warning threshold
193	40404	MLP2STAT2 MLPII addition status bits	Bit 10: Modbus remote display watchdog/idle condition Bit 11: Reserved[2] Bit 12: Modbus watchdog condition Bit 13: DeviceNet remote display watchdog/idle condition Bit 14: DeviceNet master in idle mode Bit 15: DeviceNet watchdog condition
194	40405	INFOPWRUP Power up (1–7) and information (8–15) bits	Bit 0: Power on reset Bit 1: Software reset indicator Bit 2: Brown-out reset Bit 3: Watchdog reset Bit 4: Stack over flow reset Bit 5: Stack under flow reset Bit 6: ROM (program code) check sum failure Bit 7: EEPROM check sum failure Bit 8: Phase rotation is ABC Bit 9: Phase rotation is ACB Bit 15: Demonstration unit
195	40406	INSTFLT Instantaneous fault bits	Bit 0: Reverse phase fault Bit 1: PTC overtemperature fault Bit 3: Undercurrent fault Bit 4: Overcurrent fault Bit 5: Ground fault Bit 6: Current unbalance fault Bit 7: Single phase fault Bit 8: PTC above warning level Bit 9: PTC shorted Bit 10: (Internal) reference voltage error Bit 11: No AC power
196	40407	PENDFLT Pending Fault bits	Bit 0: Reverse phase fault Bit 1: PTC overtemperature fault Bit 3: Undercurrent fault Bit 4: Overcurrent fault Bit 5: Ground fault Bit 6: Current unbalance fault Bit 7: Single phase fault Bit 9: PTC shorted Bit 10: (Internal) Reference voltage error Bit 11: No AC power

1 For registers that have bit information, reading a "1" in any bit means that the statement for that bit alone is true.

2 Do not write to reserved bit locations.

- **6000 Servo Tank Gauge Medidor de nivel por detección de densidad de líquidos.**

La implementación del protocolo MODBUS en el 6000 STG permite el paso de variables medidas y calculadas, información de configuración y diagnóstico en registros de datos. Los datos se envían en estos registros como valores de punto flotante, valores de palabra, códigos numéricos relacionados con listas de configuración, palabras de resumen de estado (bits empaquetados) o indicadores de estado individuales (bits individuales). *[14]*

Un maestro y hasta 31 6000 STG pueden conectarse en múltiples puntos en un único bus de comunicación EIA (RS) 485. Las funciones MODBUS implementadas en el 6000 STG se enumeran en la tabla 18:

Tabla 18- funciones MODBUS implementadas en el 6000 STG. *[14]*

Function Code	Function	Information Type	MODBUS Nomenclature
03	Read	Word, code, status word, floating point	Read output registers
04	Read	Word, code, status word, floating point	Read input registers
06	Write	Word, Code, status word	Preset single register
16	Write	Word, Code, status word, floating point	Force multiple registers

Parámetros El puerto EIA (RS) 485 debe configurarse para una velocidad de transmisión (velocidad en baudios). Los valores permitidos son 1200, 2400, 4800, 9600 y 19200 bits por segundo. Este elemento debe configurarse mediante la pantalla local. En la siguiente tabla 19 se proporciona un resumen de la información de configuración requerida por el 6000 STG para implementar MODBUS.

Tabla 19- Configuración requerida por el STG para implementar Modbus. *[14]*

Configuration Item	Valid Entries	MODBUS Configurable	Local Display Configurable
MODBUS address	1...247	No	Yes
Baud rate	600 1200 2400 4800 9600 19200	No	Yes
Parity	Odd Even None	No	Yes
Word type	Unsigned Signed	No	No

Secuencia de transmisión MODBUS

La comunicación MODBUS del receptor 6000 STG Servo Host deberá tener una secuencia ideal para realizar una transmisión de datos fluida y evitar pérdidas de datos innecesarias, a continuación, una porción de registro Modbus en tabla 20:

Tabla 20- Tabla de registros Modbus 6000 Servo Tank. *[14]*

MODBUS Register *1	Code	Data Item	Data Range	Unit (default)	Note	Data Format	GVH *2	Remarks
1	03/04	Displacer Position	0 ~ 65535.0	mm		Float	000	*3
2								
3	03/04	Liquid Level	0~65535.0	mm		Float	008	*3
4								
5	03/04	Liquid Temperature	-200.0 ~ 360.0	°C	10^{-1}	Integer	010	*4
6	03/04	Average Gas Temperature	-200.0 ~ 360.0	°C	10^{-1}	Integer	013	*4
7	03/04	HART Device 1 Data				Float	011	
8								
9	03/04	Hart Device 2 Data				Float	012	
10								
11	03/04	Water Bottom Level	0~65535.0	mm		Float	014	*3
12								
13	03/04	Upper Density	0~3.2767	g/ml	10^{-4}	Word	005	
14	03/04	Middle Density	0~3.2767	g/ml	10^{-4}	Word	006	
15	03/04	Lower Density	0~3.2767	g/ml	10^{-4}	Word	007	
16	03/04	Upper Interface Level	0~65535.0	mm		Float	002	*3
17								
18	03/04	Middle Interface Level	0~65535.0	mm		Float	003	*3
19								
20	03/04	Tank Bottom Level	0~65535.0	mm		Float	004	*3
21								

- **Micropilot FMR20 MODBUS RS485 Radar sin contacto Para sólidos a granel:**

Tabla 21 sobre Integración en el sistema mediante protocolo Modbus, información sobre el Modbus RS485. *[15]*

Tabla 21- Información básica Micropilot FMR20 MODBUS RS485. *[15]*

Ajuste	Opciones	Por defecto
Bits de datos	7,8	**8**
Paridad	Par, impar, ninguno	**Par**
Bits de stop	1,2	**1**
Velocidad de transmisión	1200, 2400, 4800, 9600, 19200	**9600**
Protocolo	RTU, ASCII	**RTU**
Dirección	1 ... 200	**200**
Intervalo de interrogación mínimo	500 ms	

- Variables medidas mediante protocolo Modbus

Los 8 parámetros de proceso más importantes se asignan como parámetros de burst a las primeras direcciones en el rango de direcciones de Modbus. Esto significa que la lectura de estos parámetros puede realizarse en una transmisión de medición. Todos los parámetros están disponibles en el formato Float32.

La dirección del registro debe incrementarse en uno (dirección del registro +1) cuando se utiliza el maestro Modbus Memograph M RSG45 o Fieldgate FXA30b. Esto puede aplicarse también para otros maestros, Se detalla una pequeña porción de registros en tabla 22. *[15]*

Tabla 22- Tabla de registros Modbus Micropilot FMR20. *[15]*

Dirección Modbus	Nombre del parámetro	Descripción	unidad SI
5000	MODB_PV_VALUE	Nivel linealizado (PV)	Depende del tipo de linealización
5002	MODB_SV_VALUE	Distancia (SV)	m
5004	MODB_TV_VALUE	Amplitud relativa de ecos (TV)	dB
5006	MODB_QV_VALUE	Temperatura (QV)	°C
5008	MODB_SIGNALQUALITY	Calidad de señal	-
5010	MODB_ACTUALDIAGNOSTICS	Número de diagnóstico actual	-
5012	MODB_LOCATION_LONGITUDE	Coordenada de longitud	°
5014	MODB_LOCATION_LATITUDE	Coordenada de latitud	°

VII. INTERFAZ MAQUINA USUARIO.

7.1. Sistema GUI

Un sistema GUI es un programa informático que actúa de interfaz de usuario, utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Como una solución ideal se propone una interfaz HMI usando el software de LabVIEW Microsoft Excel junto con el protocolo de comunicación Modbus la cual permitirá crear una herramienta GUI practica para el usuario.

Cuando hablamos de una interfaz gráfica de usuario GUI, nos referimos a la cara visible de los programas tal y como se presenta a los usuarios para que interactúen con la máquina. La interfaz gráfica implica la presencia de un monitor de ordenador o pantalla constituida por una serie de menús e iconos que representan las opciones que el usuario puede tomar dentro del sistema. *[16]*

En el contexto del proceso de interacción persona-ordenador, la interfaz gráfica de usuario es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con el sistema operativo. *[17]*

Este diseño será una herramienta grafica en LabVIEW que permitirá la extracción, visualización y comunicación entre dispositivos con protocolo de comunicación Modbus, la cual tendrá la función de ser el intermediario entre el dispositivo, sistema operativo y el usuario final, esta herramienta será la encargada de decodificar, codificar, extraer y hacer todos los requerimientos necesarios para lograr tratar la variable y así poder utilizarla ya sea como un sistema de supervisión y extracción de datos o un sistema de adquisición y control de datos. Como una herramienta adicional que permita al usuario facilitarle el tratamiento de la variable o parámetro MODBUS que se quiera trabajar, se elaborará una tabla en el software de Microsoft EXCEL la cual contendrá una lista de parámetros estándar necesarios para poder manejar y lograr entender el funcionamiento de la comunicación en Modbus sin importar el fabricante. Todo esto requiere diseño de ingeniería.

Las características básicas de una buena herramienta de interfaz podrían sintetizarse en:

- Facilidad de comprensión, aprendizaje y uso.
- Representación fija y permanente de un determinado contexto de acción (fondo)
- El objeto de interés ha de ser de fácil identificación
- Las interacciones se basarán en acciones físicas sobre elementos de código visual o auditivo (iconos, botones, imágenes, mensajes de texto o sonoros, barras de desplazamiento y navegación...) y en selecciones de tipo menú con sintaxis y órdenes

- Las operaciones serán rápidas, incrementales y reversibles, con efectos inmediatos
- Existencia de herramientas de Ayuda y Consulta
- Tratamiento del error bien cuidado y adecuado al nivel de usuario. *[17]*

7.2. Software de Desarrollo de Sistemas NI LabVIEW

LabVIEW (Laboratory virtual Instrument Engineering Workbench) LabVIEW es un entorno de programación gráfica que los ingenieros utilizan para desarrollar sistemas pruebas automatizadas de investigación, validación y producción, está orientado a desarrollar aplicaciones para instrumentación que integran una serie de librerías para comunicación con instrumentos electrónicos como GPIB, RS232O RS485 con tarjetas de adquisición y acondicionamiento como VXI O SCXI, comunicaciones en red TCP/IP, UDP, o en los estándares de software COM, OLE, DDE, DLL o activeX para Windows, así como Apple Events para MacOS o PIPE para UNIX.(Holguin, Pérez y Orozco, 2002).

El software LabVIEW es ideal para cualquier sistema de medidas y control y el corazón de la plataforma de diseño de NI. Al integrar todas las herramientas que los ingenieros y científicos necesitan para construir una amplia variedad de aplicaciones en mucho menos tiempo. *[18]*

En este caso se usará LabVIEW como una herramienta GUI para lograr el acondicionamiento y extracción de datos de dispositivos MODBUS cualquiera, en siguiente figura (Fig. 19) se ejemplifica un entorno en LabVIEW.

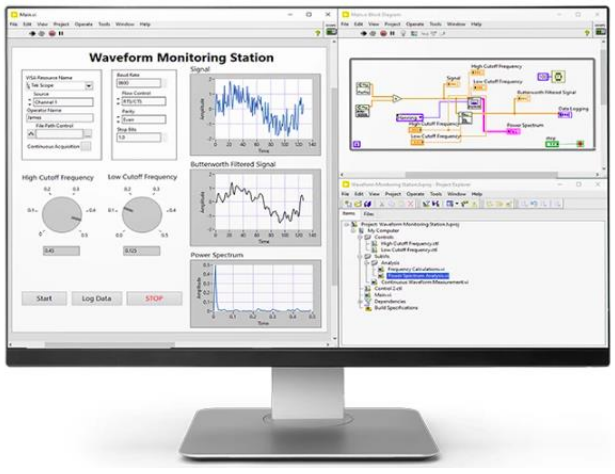


Figura 19- Entorno de LabVIEW habitual. *[18]*

Los programas desarrollados con LabVIEW se llaman Instrumentos Virtuales, o VIs, y su origen provenía del control de instrumentos, aunque hoy en día se ha expandido ampliamente no solo al control de todo tipo de electrónica (Instrumentación electrónica) sino también a su programación embebida,

comunicaciones, matemáticas, etc. Un lema tradicional de LabVIEW es: "La potencia está en el Software", que con la aparición de los sistemas multinúcleo se ha hecho aún más potente. Entre sus objetivos están el reducir el tiempo de desarrollo de aplicaciones de todo tipo (no solo en ámbitos de Pruebas, Control y Diseño) y el permitir la entrada a la informática a profesionales de cualquier otro campo. LabVIEW consigue combinarse con todo tipo de software y hardware, tanto del propio fabricante -tarjetas de adquisición de datos, PAC, Visión, instrumentos y otro Hardware- como de otros fabricantes. *[18]*

Presenta facilidades para el manejo de:

Interfaces de comunicaciones:

- Puerto serie
- Puerto paralelo
- GPIB
- PXI
- VXI
- TCP/IP, UDP, DataSocket
- Irda
- Bluetooth
- USB
- OPC.

Capacidad de interactuar con otros lenguajes y aplicaciones:

- DLL: librerías de funciones
- .NET
- ActiveX
- Multisim
- Matlab/Simulink
- AutoCAD, SolidWorks, etc
- Herramientas gráficas y textuales para el procesado digital de señales.
- Visualización y manejo de gráficas con datos dinámicos.
- Adquisición y tratamiento de imágenes.
- Control de movimiento (combinado incluso con todo lo anterior).
- Tiempo Real estrictamente hablando.
- Programación de FPGAs para control o validación.
- Sincronización entre dispositivos.

Como se ha dicho es una herramienta gráfica de programación, esto significa que los programas no se escriben, sino que se dibujan, facilitando su comprensión. Al tener ya pre-diseñados una gran cantidad de bloques, se le facilita al usuario la creación del proyecto, con lo cual en vez de estar una gran cantidad de tiempo en programar un dispositivo/bloque, se le permite invertir mucho menos tiempo y dedicarse un poco más en la interfaz gráfica y la interacción con el usuario final. *[18]*

Cada VI consta de dos partes diferenciadas:

- Panel Frontal: El Panel Frontal es la interfaz con el usuario, la utilizamos para interactuar con el usuario cuando el programa se está ejecutando. Los usuarios podrán observar los datos del programa actualizados en tiempo real (como van fluyendo los datos, un ejemplo sería una calculadora, donde tú le pones las entradas, y te pone el resultado en la salida). En esta interfaz se definen los controles e indicadores.
- Diagrama de Bloques: Es el programa propiamente dicho, donde se define su funcionalidad, aquí se colocan íconos que realizan una determinada función y se interconectan (el código que controla el programa --. Suele haber una tercera parte icono/conector que son los medios utilizados para conectar un VI con otros VIs.

En el panel frontal, encontraremos todo tipos de controles o indicadores, donde cada uno de estos elementos tiene asignado en el diagrama de bloques una terminal, es decir el usuario podrá diseñar un proyecto en el panel frontal con controles e indicadores, donde estos elementos serán las entradas y salidas que interactuarán con la terminal del VI. Podemos observar en el diagrama de bloques, todos los valores de los controles e indicadores, como van fluyendo entre ellos cuando se está ejecutando un programa VI. *[18]*

La Figura 20. muestra un Diagrama de Bloques de un programa en el que se genera un array de 100 elementos aleatorios, a continuación se hace la FFT de este array y se muestra a continuación:

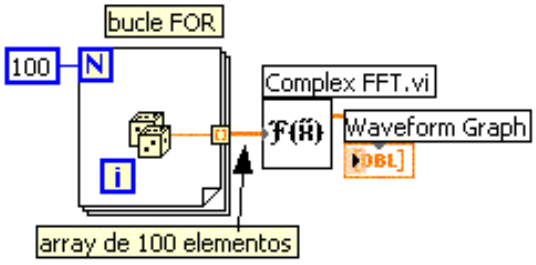


Figura 20- Diagrama de Bloques bucle FOR. *[18]*

7.3. Software de Microsoft Excel.

Excel es una hoja de cálculo que nos permite manipular datos numéricos y de texto en tablas formadas por la unión de filas y columnas. Pero ¿qué es una hoja de cálculo?

Una hoja de cálculo es lo que utilizaban los contadores para llevar registros, esto se utilizaba mucho antes de que aparecieran las computadoras. Las hojas de cálculo en programas informáticos aparecieron desde la década de 1960 y fueron desarrolladas para simular las hojas de trabajo contables, que se utilizaba en ese entonces y de esa manera automatizaban el trabajo contable. *[19]*

Historia.

Microsoft comercializó originalmente un programa para las hojas de cálculo llamado Multiplan en 1982, que fue muy popular en los sistemas CP/M, pero en los sistemas MS-DOS perdió popularidad frente al Lotus 1-2-3. Microsoft publicó la primera versión de Excel para Mac en 1985, y la primera versión de Windows (numeradas 2-05 en línea con el Mac y con un paquete de tiempo de ejecución de entorno de Windows) en noviembre de 1987.

Excel ofrece una interfaz de usuario ajustada a las principales características de las hojas de cálculo, en esencia manteniendo ciertas premisas que pueden encontrarse en la hoja de cálculo original, VisiCalc: el programa muestra las celdas organizadas en filas y columnas (intersección de las filas y columnas), y cada celda contiene datos o una fórmula, con referencias relativas, absolutas o mixtas a otras celdas.

Excel fue la primera hoja de cálculo que permitió al usuario definir la apariencia (las fuentes, atributos de carácter y celdas). También introdujo recomputación inteligente de celdas, donde celdas dependientes de otra celda que han sido modificadas, se actualizan al instante (programas de hoja de cálculo anterior recalculaban la totalidad de los datos todo el tiempo o esperaban para un comando específico del usuario). Excel tiene una amplia capacidad gráfica, y permite a los usuarios realizar, entre otras muchas aplicaciones, listados usados en combinación de correspondencia.

Cuando Microsoft primeramente empaquetó Microsoft Word y Microsoft PowerPoint en Microsoft Office en 1993, rediseñó las GUI de las aplicaciones para mayor coherencia con Excel, producto insignia de Microsoft en el momento. *[19]*

Como bien mencionamos en este proyecto se implementará el uso del software de Microsoft Excel y NI LabVIEW como herramientas graficas finales (GUI) para los usuarios, se detallará la manera de tratamiento de registros MODBUS de forma directa y los resultados en el programa desarrollado.

VIII. TRATAMIENTO DE REGISTROS MODBUS

8.1. Diseño de hojas de cálculos necesarias para ejecución del programa

Por un lado, Modbus RTU usa RS-485 o RS-232, por otro lado, Modbus TCP utiliza Ethernet. Si está buscando una conexión física, es recomendable elegir el modelo que coincida con la interfaz eléctrica del equipo que desea conectar. Si está eligiendo un dispositivo de E/S, es recomendable elegir uno que coincida con su red.

Como bien se define en estudio del protocolo MODBUS se requiere definir la configuración inicial del equipo previamente establecido por cada fabricante pero que siguen el siguiente patrón según tabla 23:

Tabla 23- Archivo de EXCEL #1. Configuración MODBUS: Hoja #1 configuración serial inicial. [Fuente propia]

CONFIGURACION SERIAL				
Puerto	**Baud Rate**	**Data Bits**	**Parity**	**Stop Bits**

8.1.1. Puerto

Los puertos serie se utilizan para conectar físicamente los dispositivos asíncronos con un sistema. Se encuentran en la parte de atrás de la unidad del sistema, integrados o utilizando un adaptador de varios puertos como, por ejemplo, los adaptadores asíncronos de 2 puertos, 8 puertos, 16 puertos y 128 puertos.

Para comprender el funcionamiento de un puerto serie, es necesario examinar primero las comunicaciones paralelo. Un puerto paralelo estándar utilizar ocho conectores o cables para transmitir los bits de datos de forma simultánea, formando un solo carácter. La ilustración siguiente figura 21 muestra la transmisión en paralelo de la letra a. *[20]*

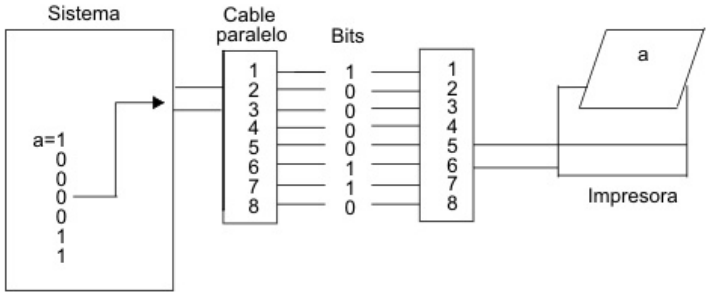


Figura 21- Puerto de comunicaciones paralelo. *[20]*

Los puertos serie requieren un solo conector, o cable, para enviar el mismo carácter de datos al dispositivo. Para conseguirlo, los datos se convierten del formato paralelo (enviado por el sistema), a formato secuencial, en el que los bits se organizan uno tras otro en una serie. Los datos se transmiten entonces al dispositivo, enviando el bit menos significativo (o el bit cero) en primer lugar. Una vez que el dispositivo remoto recibe los datos, éstos se vuelven a convertir al formato paralelo. La ilustración siguiente figura 22 muestra la transmisión serie de la letra a.

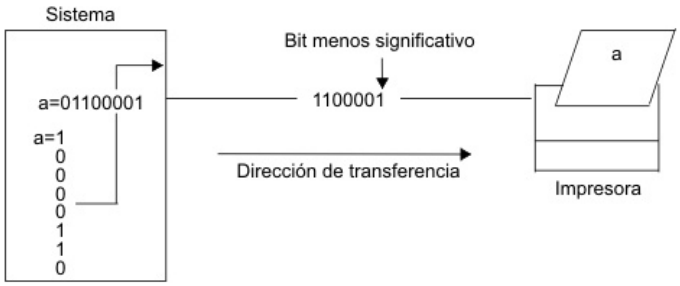


Figura 22- Puerto de comunicaciones serie. *[20]*

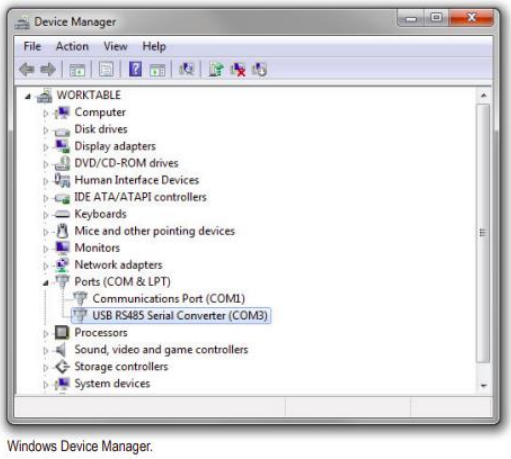
En esta columna se definirá el puerto serial de la computadora con la cual nos comunicaremos con el dispositivo final. Actualmente un puerto serie en un ordenador se han descontinuado por lo que se tiene que optar con convertidores de puertos de serial a USB o el puerto físico de entrada a nuestro computador. Con fines prácticos se seleccionó el convertidor USB a RS485 Microflex siendo un convertidor fiable para conexiones locales con equipos con comunicación RS485.

Al realizar la conexión con este convertidor al computador se requiere la instalación del driver correspondiente el cual en nuestro caso está disponible y completamente gratuito en línea figuras 23 paso a paso de instalación de driver:

Figura 23- Instalación de Driver Convertidor serial. *[21]*

Al finalizar la correcta instalación del Driver y conectar nuestro cable al puerto USB de nuestro computador se define en primera instancia el número de puerto serial

habilitado ver figura 24 el cual será el número que se digitará en primera tabla de Excel (Tabla 23).



Windows Device Manager.

Figura 24- Administrador de dispositivos Windows puerto serial. [Fuente propia]

Extra: Como bien se conoce con Windows podemos realizar cambio en el número de puerto virtual ingresando al menú de configuración.

8.1.2. BAUD RATE (Tasa de baudios)

La velocidad en baudios es el número de veces por segundo que una señal de comunicaciones serie cambia de estado; el estado puede ser un nivel de voltaje, una frecuencia o un ángulo de fase de frecuencia.

Si la señal cambia una vez para cada bit de datos, entonces un bps equivale a un baudio. Por ejemplo, un módem a 300 baudios cambia de estado300 veces por segundo La velocidad o tasa de baudios que se digitará en la columna (Tabla 24) será definida por la ficha técnica del dispositivo y su velocidad de respuesta de información, la mayoría de las veces dependiendo de la distancia, cantidad y el sistema de adquisición de datos se puede configurar esta pueden ser: 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 y 115200. *[20]*

8.1.3 DATA BITS (bits de datos)

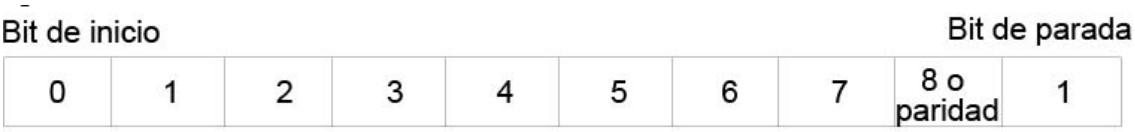
Bits que configuran los caracteres excluyendo a los de control de comunicaciones, el tipo de datos BIT contiene una serie de caracteres de longitud variable de dígitos binarios. Generalmente, se utiliza para representar los datos binarios arbitrarios que no contienen un número exacto de bytes. Un literal de serie de caracteres de bits consta de la letra B, seguida de una serie de dígitos binarios encerrada entre comillas simples, como en el ejemplo siguiente: *[20]*

B'0100101001'

Se puede especificar cualquier número de dígitos, que puede ser 0 ó 1. La B inicial puede especificarse en mayúsculas o minúsculas. Como estudiamos tenemos dos opciones estándares que podemos digitar en nuestra estructura, las cuales también se definen en el manual del dispositivo las cuales pueden ser **7 o 8.**

8.1.4. Parity (Bits de paridad)

El bit de paridad, a diferencia de los bits de inicio y de parada, es un parámetro opcional que se utiliza en las comunicaciones serie para determinar si el dispositivo remoto está recibiendo correctamente el carácter de datos que se transmite. *[20]*



El bit de paridad puede tener una de las cinco especificaciones siguientes:

Ninguna: Especifica que el sistema local no debe crear un bit de paridad para los caracteres de datos que se están transmitiendo. También indica que el sistema local no comprueba el bit de paridad de los datos recibidos de un sistema principal remoto.

Par: Especifica la suma del número total de unos binarios de un solo carácter es un valor par. En caso negativo, el bit de paridad debe ser un 1 para asegurarse de que el número total de unos binarios sea par.

Por ejemplo, si la letra a (1100001 binario) se transmite bajo la paridad par, el sistema de envío suma el número de unos binarios que, en este caso, es tres y deja el bit de paridad en un 1 para mantener un número par de unos binarios. Si la letra A (1000001 binario) se transmite bajo las mismas circunstancias, el bit de paridad sería un 0, por lo que el número total de unos binarios se mantendría como un número par.

Impar: Funciona bajo las mismas directrices que la paridad par, con la excepción de que el número total de unos binarios debe ser un número impar.

Espacio: Especifica que el bit de paridad siempre será un cero binario. Otro término utilizado para la paridad de espacio es rellenado de bits, que se deriva de su utilización como rellenador de los datos de siete bits que se transmiten a un dispositivo que sólo acepte datos de ocho bits. Estos dispositivos interpretan el bit de paridad de espacio como un bit de datos adicional para el carácter transmitido.

Marca: Funciona bajo las mismas directrices que la paridad de espacio, con la excepción de que el bit de paridad siempre es un 1 binario. El bit de paridad de marca sólo actúa como rellenador. *[20]*

8.1.5. Stop bits (Bits de inicio, parada y marcha)

Los bits de inicio y de parada se utilizan en la comunicación asíncrona con el fin de temporizar la sincronización de los caracteres de datos que se transmiten. *[20]* En nuestra tabla se define **1 o 0** dependiendo siempre de hoja de datos del equipo.

Sin la utilización de estos bits, los sistemas emisores y receptores no sabrán dónde termina un carácter y empieza el siguiente.

Otro bit que se utiliza para separar caracteres de datos durante la transmisión es el bit RS de marca (o desocupado). Este bit, un 1 binario, se transmite cuando la línea de comunicaciones está desocupada y no se está enviando o recibiendo ningún carácter. *[20]*

Cuando el sistema recibe un bit de inicio (binario 0), se entiende que un número fijo del bit de carácter (determinado por el parámetro bits por carácter), e incluso un bit de paridad (determinado por el parámetro parity), sigue al bit de inicio. A continuación, el sistema recibe un bit de parada (binario 1). En el ejemplo siguiente figura 25 está presente el bit de paridad y el bit por carácter es 7.

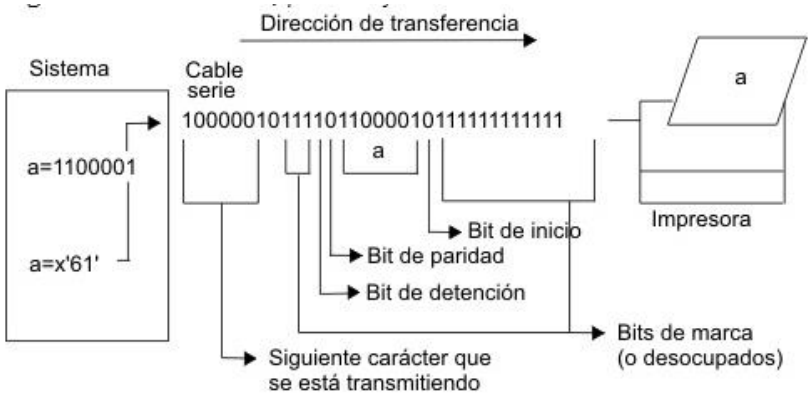


Figura 25- Bits de inicio, parada y marca. *[20]*

Al finalizar la llenar de datos en nuestra primera tabla (Tabla 24 Tabla de Excel número 1 configuración serial inicial) se creará una segunda tabla en el mismo archivo de Excel que se utilizará para listar los dispositivos con comunicación serial a utilizar, tabla 24 esta se titulará: DISPOSITIVOS SERIAL.

Tabla 24- Archivo de EXCEL #1 Configuración MODBUS Hoja #2 Dispositivos serial. [Fuente propia]

DISPOSITIVOS SERIAL				
Dispositivo	**Equipo**	**Esclavo**	**Puerto**	**¿Realizar?**

8.2. Descripción de tabla “Dispositivos Serial”

1. **Dispositivos**: Definiremos el equipo de cualquier marca con el cual trabajaremos detallando modelo y detalles.

Equipo seleccionado para este estudio: **SQUARE-D Motor Logic Plus II**

2. **Equipo:** Se incluye un TAG de trabajo simulando un equipo de campo industrial. **TAG de campo definido E-107.**

3. **Esclavo:** Como ya hemos estudiado Las direcciones de esclavo válidas están en el rango de **0 – 247** decimales. Los dispositivos esclavos individuales tienen direcciones asignadas en el rango 1 – 247. El número máximo de estaciones previsto es de 63 esclavos más una estación maestra en una red industrial habitual. Esta dirección en configurable en nuestro dispositivo en dependencia de nuestra Red.

4. **Puerto:** Esta columna se confirmará el puerto de pasarela previamente definido por el cual nuestro dispositivo se conectará a nuestro ordenador.

5. **Realizar:** En nuestra segunda hoja de Excel podemos llenar un sinnúmero de equipos, pero esta columna dará la pauta para que el programa de LadVIEW ejecute sus rutinas. Se digitará: **1 o 0.**

Al finalizar de llenar la segunda tabla 24 (Hoja de Excel #2 Dispositivos serial) seguimos con la tercera tabla 25 dentro del mismo archivo de Excel que se utilizará para listar los dispositivos con comunicación TCP-IP a utilizar, esta se titulará: **DISPOSITIVOS RED.**

Tabla 25- Archivo de EXCEL #1 Configuración MODBUS, Hoja #3 Dispositivos Red. [Fuente propia]

DISPOSITIVOS RED					
Dispositivos	**Equipos**	**ID**	**Puerto**	**IP**	**¿Realizar?**

8.3. Descripción de tabla “Dispositivos RED”

1. **Dispositivos**: Definiremos el equipo de cualquier marca con el cual trabajaremos detallando modelo y detalles. Equipo seleccionado para este estudio: schneider electrics PM5320

2. **Equipo:** Se incluye un TAG de trabajo simulando un equipo de campo industrial.

   TAG de campo definido cualquiera E-107.

3. **ID:** Entonces es habitual que en la configuración de Modbus TCP se siga asignando el ID del esclavo (número entero normalmente entre **1 y 255).**

4. **Puerto**: Estándar para Modbus TCP es el 502, pero el número de puerto a menudo puede reasignarse si se desea.

5. **IP:** Una dirección IP es una dirección única que identifica a un dispositivo en Internet o en una red local. IP significa “protocolo de Internet”, que es el conjunto de reglas que rigen el formato de los datos enviados a través de Internet o la red local.

Una dirección IP es una cadena de números separados por puntos. Las direcciones IP se expresan como un conjunto de cuatro números, por ejemplo, 192.158.1.38. Cada número del conjunto puede variar de 0 a 255. Por lo tanto, el rango completo de direcciones IP va desde 0.0.0.0 hasta 255.255.255.255.

Las direcciones IP no son aleatorias. La Autoridad de números asignados de Internet (Internet Assigned Numbers Authority, IANA), una división de Internet Corporation para números y nombres asignados (Internet Corporation for Assigned Names and Numbers, ICANN), genera y asigna matemáticamente las direcciones IP. ICANN es una organización sin fines de lucro que se estableció en los Estados Unidos en 1998 para ayudar a mantener la seguridad de Internet y permitir que todos puedan utilizarla. *[22]*

6. **Realizar**: En nuestra tercera hoja de Excel podemos llenar un sinnúmero de equipos, pero esta columna dará la pauta para que el programa de LadVIEW ejecute sus rutinas. Se digitará: 1 o 0.

Como siguiente paso diseñaremos un segundo archivo de Excel tabla 26 en el cual se detallarán los registros Modbus de los dispositivos que trabajaremos:

Tabla 26- Archivo de EXCEL #2 DISPOSITIVOS Hoja #1 SQUARE-D Motor Logic Plus II. [Fuente propia]

1- Address	2-Codificacion	3-Dispositivo	4-Parámetro	5-Unidades

6-Factor	7-Extraer?	8-Cantidad de Bits	9.Orden de elementos	10-Número de elementos

11-Representación Negativos	12-Decimales	13-Offset	14-Length	15-Comentarios

Como hemos seleccionado el equipo **SQUARE-D Motor Logic Plus II** iniciaremos el proceso de llenado de nuestra TABLA de datos con un registro inicial explicando paso a paso la forma correcta de interpretar las tablas, para esto hemos estudiado la hoja de datos del dispositivo disponible en la Web (Instruction Bulletin, Motor Logic® Plus II Network Programming Guide, 30072-451-21B, 07/2006),

8.4. Descripción de Hoja Excel “Dispositivos”

1. **Address (Registros):** Como antes se había mencionado los registros son números de 16 bits en formato decimal (Registro = dirección + **1).**

De acuerdo con la hoja de datos del fabricante y la tabla de registros de nuestro equipo seleccionado extracto de tabla de registros figura 26:

16 Bit Memory Hex	Register	Code and Description	Notes[1]
192	40403	WARNSTAT Warning status bits	Bit 0: Average current below UC warning threshold Bit 1: High phase above OC warning threshold Bit 2: Ground fault current above the GF warning threshold Bit 3: Current unbalance above the CUB warning threshold
193	40404	MLP2STAT2 MLPII addition status bits	Bit 10: Modbus remote display watchdog/idle condition Bit 11: Reserved[2] Bit 12: Modbus watchdog condition Bit 13: DeviceNet remote display watchdog/idle condition Bit 14: DeviceNet master in idle mode Bit 15: DeviceNet watchdog condition

Figura 26- Registro MODBUS Motor Logic plus II [13]

En la columna **1-Address** para obtener el registro 40403 digitaremos: **402** como ya hemos estudiado la trama de solicitud PDU los registros se abordan empezando por cero, por ejemplo: el registro 1 es la dirección 0.

2. Codificación:

En esta columna definiremos el tipo de trama con el cual el registro será tratado las opciones frecuentes se han enlistado como validación de datos con el siguiente orden: **Binario; Octal; Decimal; Hexadecimal**

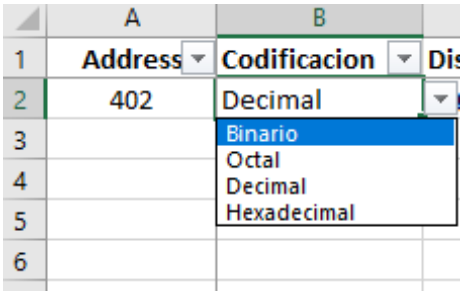


Figura 27- Validación de datos definidos. [Fuente propia]

En la columna 2-Codificacion según figura 27 para el registro 40403 podemos seleccionar: **HEXADECIMAL: cuando digitamos el registro como 192** o **DECIMAL: cuando digitamos 402.** En nuestro caso seleccionaremos **DECIMAL.**

3. Dispositivo.

Definiremos el equipo de cualquier marca con el cual trabajaremos detallando modelo y detalles. Equipo seleccionado para este estudio: **SQUARE-D Motor Logic Plus II**

4. Parámetro:

Esta opción será la descripción detallada del registro extraído o información solicitada al dispositivo: Amperajes, voltajes, TAG, fecha etc. Para nuestro caso de ejemplo imagen 8.6. el registro 40403 la solicitud es: **Warning Status Bits**

5. **Unidades:**

Definimos unidad de medida a una referencia convencional que se usa para medir la magnitud física de un determinado objeto, sustancia o fenómeno: Celcius, Frecuencia, presión, densidad etc. El registro de ejemplo que seleccionamos en particular no posee una unidad de medida solo es informativa en este caso se dejara vacío.

6. **Factor**:

Corresponde a el factor de una escala, significa que el registro contiene el valor multiplicador así, el valor real es el valor del registro dividido por el numero multiplicador. El registro de ejemplo seleccionado no posee un número multiplicador así que se digitara: **1**

7. **Extraer:**

En esta hoja de Excel podemos llenar todos los registros MODBUS, pero esta columna dará la pauta para que el programa de LadVIEW ejecute sus rutinas. Se digitará: **1** si se desea extraer **y 0** si se decide no extraer.

8. **Cantidad de Bits:**

Los registros de MODBUS son de 2 bytes (16 bits), El máximo número de registros leídos en una sola trama es de 64. Entonces digitaremos: **16** en nuestra tabla de Excel.

9. **Orden de elementos:**

En dependencia del equipo a trabajar el orden de los bytes de la red permitirá establecer el orden de Bytes de estos parámetros en MSB (Bits más significativo) o LSB (Bits menos significativo) primero. También hemos enlistado en nuestra hoja como una validación de datos con el siguiente orden: **MSB; LSB.** Para nuestro registro seleccionado la hoja de datos nos indica que es un **MSB,** ver figura 28.

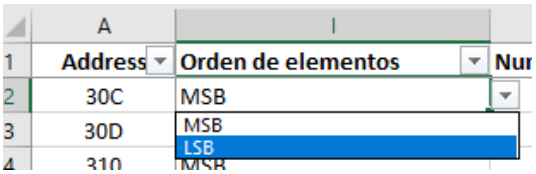


Figura 28- Validación de datos definidos orden de elementos. [Fuente propia]

10. **Número de elementos:**

Esta columna se refiere a la cantidad de registros involucrados en el parámetro a extraer, esto como siempre en dependencia a cada dispositivo y a la longitud de la palabra, en nuestro ejemplo está claro que el número de elementos involucrados es 1 ya que solo utiliza 2 bytes para transmisión y los siguientes 2 bytes corresponden a otro parámetro según se detalla en figura 29:

16 Bit Memory Hex	16 Bit Memory Register	Code and Description	Notes[1]
192	40403	WARNSTAT Warning status bits	Bit 0: Average current below UC warning threshold Bit 1: High phase above OC warning threshold Bit 2: Ground fault current above the GF warning threshold Bit 3: Current unbalance above the CUB warning threshold
193	40404	MLP2STAT2 MLPII addition status bits	Bit 10: Modbus remote display watchdog/idle condition Bit 11: Reserved[2] Bit 12: Modbus watchdog condition Bit 13: DeviceNet remote display watchdog/idle condition Bit 14: DeviceNet master in idle mode Bit 15: DeviceNet watchdog condition

Figura 29- Registro MODBUS Motor Logic plus II número de elementos. *[13]*

Para nuestro ejemplo digitaremos: **1**

Para reforzar este concepto notaremos que en la siguiente figura 30 registro DECS 250, el cual es otro dispositivo con comunicación MODBUS utiliza 4 BYTES para envió de un parámetro el registro 41000 y 41001.

Group	Name	Register	Type	Bytes	R/W	Unit	Range
Field Voltage Meter	V_x	41000	Float	4	R	Volt	-1000 – 1000
Field Current Meter	I_x	41002	Float	4	R	Amp	0–2000000000

DECS-250 Modbus® Communication

Figura 30- Registro DECS 250 de 4 bytes. *[23]*

11. Representación Negativos:

Como ya conocemos los tipos de datos pueden ser números positivos, números negativos, con signo etc, La tabla de registros Modbus debe especificar el formato que se usa para representar el valor del registro. También hemos enlistado en nuestra hoja como una validación de datos las opciones disponibles con el siguiente orden: **Punto Flotante SINT32; Complemento A1; Complemento A2; Ninguno.**

Para nuestro registro seleccionado la hoja de datos nos indica que es: **Ninguno.**

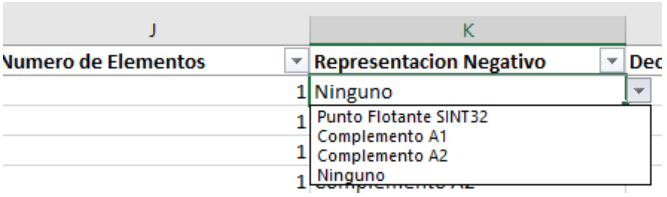


Figura 31- Validación de datos definidos representación negativa. [Fuente propia]

11.1. Punto flotante SINT32 negativo:

Dado que Modbus utiliza registros de 16 bits para mantener valores, los números de punto flotante de 32 bits deben dividirse entre dos registros. Modbus no declara un estándar sobre cómo representar números de punto flotante, por lo que es posible que un dispositivo maneje los números de punto flotante de manera

diferente a LabVIEW. Describiremos las diferentes formas en que los números de punto flotante se pueden representar con dos registros: Cómo se codifican los datos reales (punto flotante) y de 32 bits en los mensajes de Modbus RTU (en inglés). La figura 32 a continuación muestra cómo se definen los números de punto flotante utilizando el estándar IEEE 754. Los bits menos significativos se almacenan en el primer registro: *[24]*

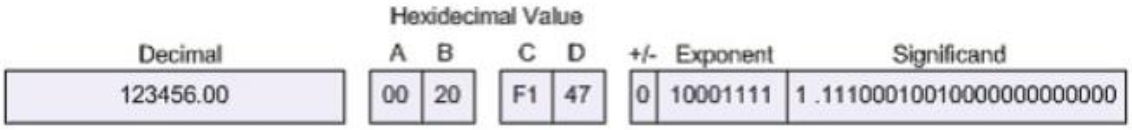


Figura 32- Los valores menos significativos primero. *[24]*

De esta forma, si intenta almacenar el número 123456.00 en el registro F400001, los bytes "A B" se almacenan en el primer registro, F40001. Los bytes "C D" se almacenan en el segundo registro, F400002. Con LabVIEW 2011 y versiones anteriores, **NO** es así como LabVIEW representa los números de punto flotante en los registros Modbus cuando se usa un Modbus Slave I/O Server. Si su dispositivo espera los bits menos significativos primero, sus datos aparecerán como el valor decimal en la figura 33.

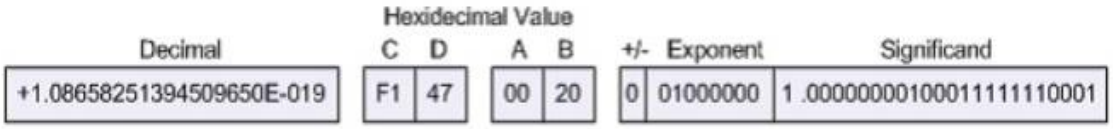


Figura 33- Los valores más significativos primero. *[24]*

11.2. Complemento A1:

Para comenzar veamos que dice las comunicaciones binarias sobre el complemento A1: Es un concepto fundamental en la teoría de números y la lógica binaria. En un sistema numérico binario, cada número puede ser representado como una secuencia de bits, donde cada bit puede tener un valor de 0 o 1.**El complemento a 1 de un número binario** se refiere a la secuencia de bits que se obtiene al invertir todos los bits de ese número. Por ejemplo, el complemento a 1 de la secuencia binaria 0101 es 1010.

El complemento a 1 es útil en la computación y la electrónica, donde se utiliza para representar números negativos en un sistema de representación de números con signo. En este sistema, el complemento a 1 se utiliza para representar el valor negativo de un número positivo, permitiendo la representación de todos los valores numéricos en un rango de números negativos y positivos.

Además, el complemento a 1 también se utiliza en la realización de operaciones aritméticas, como la suma y la resta, en el procesamiento de datos y la transmisión de información. *[25]*

11.3. Complemento A2:

El complemento es por lo general la forma más usada en computación para representar un número entero negativo. Si se trata de un número positivo simplemente se añaden 0s a la izquierda del número binario para completar los espacios de acuerdo con el número total de bits; en cambio, el procedimiento para un número negativo es primero obtener el complemento A1 de ese número y sumarle un uno mediante la suma binaria.

Cabe mencionar que al igual que el complemento A1 el primer bit de la izquierda (Bit del signo) nos indican si el número es positivo o negativo: 0 es positivo y 1 es negativo.

Una forma más fácil de proceder a convertir el binario en complemento A2 es: Encontrar el primer ´´1´´ partiendo de derecha a izquierda a invertir todos los 0s y los 1s que se encuentran a su izquierda. *[25]*

12. Decimales:

Cantidad de decimales o de precisión entregada por el equipo con comunicación MODBUS, también detallada en tabla de registro MODBUS. Para nuestro ejemplo como se trata de un bit en general es un número entero y será: **0** para números sin punto decimal y más de **1** dependiendo del parámetro y hoja de dato extraído.

13. Offset:

El offset solo se puede seleccionar cuando definimos al tipo de variable como “BIT WORD” esto quiere decir que de una consulta MODBUS que nos devuelve 16 bits podremos quedarnos con el resultado de 1 bit. El valor de este parámetro va de 0 a 15 siendo 0 el LSB y 15 el MSB. el bit de parada especifico. Llenaremos nuestra tabla de Excel siempre definiendo el parámetro tratado vamos a seleccionar el Bit 1: **Bit 1: High phase above OC warning threshold,** nuestra fila contendrá el digito: **1**

14. Length:

Con este parámetro podemos definir la cantidad de registros que queremos consultar, esta longitud puede ser 1 o 2 según el tipo de consulta.

Si la consulta es del tipo “Input Status” o “Coil Status” la longitud será forzosamente 1, ósea no será permitida su modificación.

Si la consulta es del tipo “Input Register” o “Holding Register” la longitud puede ser 1 o 2, implica que como resultado de la consulta tendremos 16 o 32 bits respectivamente.

Tipos de Consultas

A continuación, veremos los tipos de consultas en tabla 27 que podemos crear y según sus parámetros a que canal las vamos a poder agregar.

Tabla 27- Tipos de Consultas. *[7]*

Tipo de Registro	Longitud	Tipo de Variable	Offset	Canal
Input Status	1	BIT	-	Entradas Digitales
Coil Status	1	BIT	-	Salidas Digitales
Holding Register	1	BIT WORD	0 a 15	Entradas Digitales
		INTEGER	-	Entradas Analógicas
		INTEGER SIGNED	-	Entradas Analógicas
	2	INTEGER 32 BIT SIGNED	-	Entradas de Pulsos
		INTEGER 32 BIT INV SIGNED	-	Entradas de Pulsos
Input Register	1	BIT WORD	0 a 15	Entradas Digitales
		INTEGER	-	Entradas Analógicas
		INTEGER SIGNED	-	Entradas Analógicas
	2	INTEGER 32 BIT SIGNED	-	Entradas de Pulsos
		INTEGER 32 BIT INV SIGNED	-	Entradas de Pulsos

Comentarios: Espacio reservado para notas personales o claves para poder tratar este registro.

IX. ANÁLISIS DE RESULTADOS.

9.1. Desarrollo del código de programación para tratamiento de registros

Se alcanzaron dos propósitos del porque se seleccionó y se detalló la forma correcta de llenar las tablas en Excel descritas anteriormente. El primero, conocer que los equipos con protocolo de comunicación MODBUS poseen tablas de registros que describen tanto información básica hasta detallada, son gratuitas y con esto se logra poder interpretarlas, como segundo propósito las hojas de datos una vez llenas predisponen cada uno de los parámetros a que sea más fácil de integrarlos y que el código sea más eficiente.

Fue necesario el estudio del protocolo y determinar cada parámetro tanto en Excel como en LabVIEW Para poder desarrollar el código, este dispone de tres etapas o eventos como se les conoce en LABVIEW las cuales se describen a continuación:

- Evento Timeout.
- Evento Actualizar Value Change.
- Evento Actualizar tabla, User Event.

Ejecución de estructuras LABVIEW:

WHILE LOOP.

1. Estructura de eventos: (0) **TIMEOUT**
    1.1. Estructura de casos 0: TRUE and FALSE.
    - ✓ TRUE: Genera un evento de usuario. Enviar a actualizar tabla Excel.
    - ✓ False: Ninguna Acción.

    1.2. Estructura de casos 1: TRUE and FALSE.
    - ✓ TRUE: Ninguna acción.
    - ✓ FALSE: FOOR LOOP, ejecutando extracción de data: RTU mediante SubVi. Cluster_Config.vi.

    1.3. Estructura de casos 2: TRUE and FALSE.
    - ✓ TRUE: Ninguna acción.
    - ✓ FALSE: FOOR LOOP, ejecutando extracción de data: TCP mediante SubVi. Cluster_config_TCP.vi.
2. Estructura de eventos: (1) **ACTUALIZAR, Valué Change.**
    2.1. Estructura de casos 0: TRUE and FALSE.
    - ✓ TRUE: Genera un evento de usuario. (Actualizar tabla Excel)
    - ✓ FALSE: Ninguna Acción.

3. Estructura de eventos: (2) **ACTUALIZAR TABLA, User Event.**

    3.1. Stacked Sequence Structure: (0) Rutas digitales de PC:

- ✓ Configuración, mediante SubVi. Extraer_Configuracion.vi
- ✓ Rutas de dispositivos, mediante SubVi. Listado_Dispositivos.vi

    3.2. Stacked Sequence Structure: (1) Arreglos de estructuras.

- ✓ Stacked Sequence Structure (0). Ejecuta SubVi. Extraer_Modbus_RTU.
- ✓ Stacked Sequence Structure (1). Ejecuta SubVi. Extraer_Modbus_TCP.

    3.3. Stacked Sequence Structure: (2) Boton principal: ACTUALIZAR.

9.1.1. Ejecuta evento “Timeout”.

Se establecieron tres ciclos los cuales se ejecutan dependiendo de la selección y cambios de usuario. La primera instrucción deberá ser el ejecutar la estructura de extracción de datos de archivos en EXCEL, el control booleano de LABVIEW “Correr Continuamente” se encontrará en ON, se corre el caso “True” del “Case Structure”.

Los controles textuales, indicadores numéricos, tales como: “Parámetro”, “Dispositivo” se pondrán invisibles que se encuentra dentro del Clúster “Lista de Parámetros Elegidos”.

- ✓ Habilita el control numérico “Offset” y el control Enum “Orden de Elementos” del Clúster “Cluster Configuración Manual”.
- ✓ Combo Box “Parámetro” ubicado dentro de Clúster “Cluster Configuración Manual”:

    1. Limit to Single Line = True.
    2. Update While typing? = True.

- ✓ Combo Box “Dispositivo” ubicado dentro de Clúster “Cluster Configuración Manual”:

    1. Limit to Single Line = True.
    2. Update While typing? = False.

- ✓ Control VISA Resourse Name “Puerto” ubicado dentro de Clúster “Cluster Extraccion”.

    1. Allow Undefined Names = False.

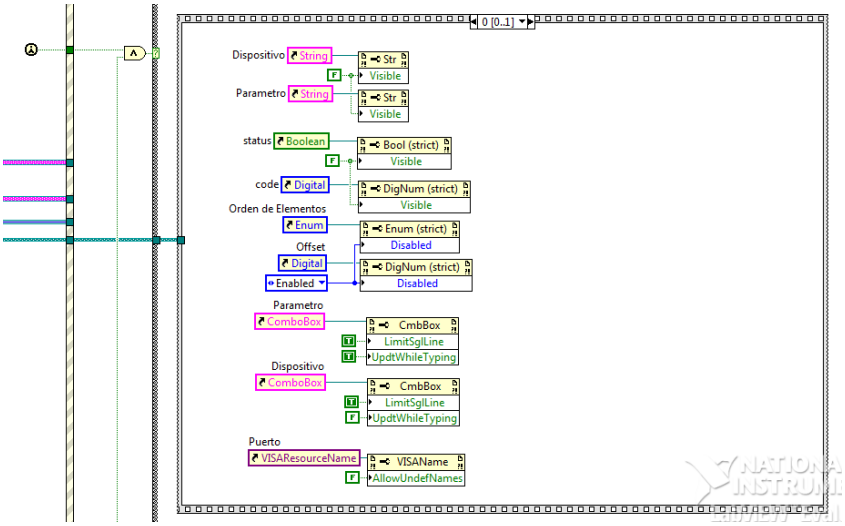


Figura 34- Código de programa ejecuta evento “Timeout. [Fuente propia]

- ✓ Coloca el valor inicial de ruta del archivo de Excel en el Control “Ruta archivos Excel”, la cual será:

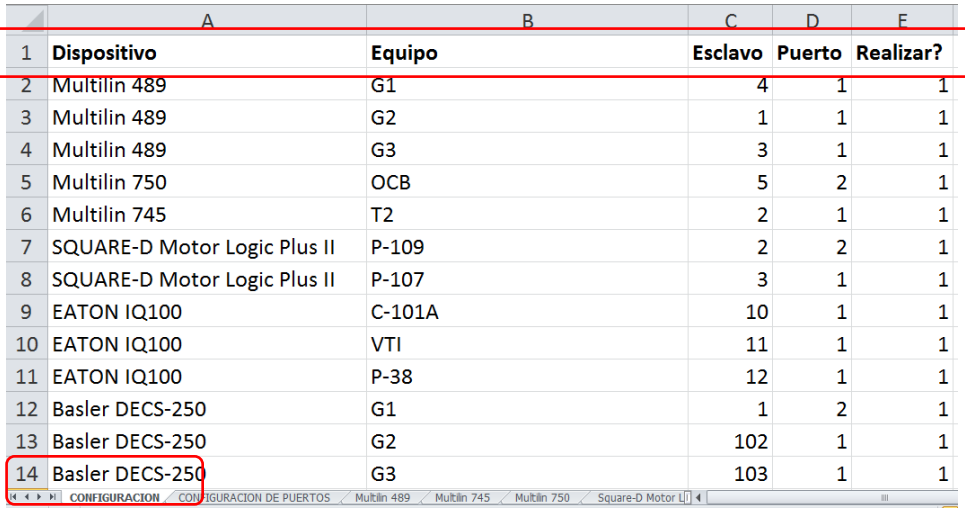
  <Ruta del VI> \Dispositivos MODBUS.xlsx

- ✓ Envía al Evento de Usuario “**Actualización tabla Excel**”

9.1.2. Evento de Usuario “Actualización Tabla Excel”

1. Evalúa si la ruta escrita en el control “Ruta archivos Excel” existe.
   Si la ruta existe, abre el archivo de Excel, y extrae la información de la siguiente forma:
   Primera hoja: “CONFIGURACION”

- ✓ Primera fila la coloca como encabezado de Columna de la tabla “Parámetro de Equipos”.
- ✓ De la fila 2 hasta la última fila, son los datos escritos en la tabla “Configuración Puertos”, en el siguiente orden según la columna.
  1. Dispositivo.
  2. Equipo.
  3. Esclavo.
  4. Puerto.
  5. ¿Realizar? Ver figura 35:

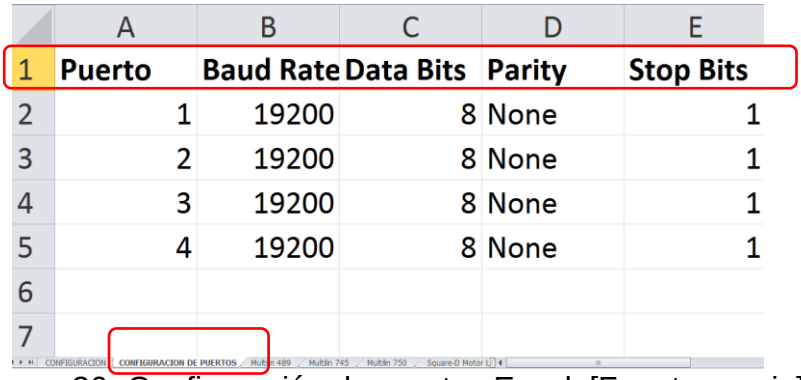


	A	B	C	D	E
1	**Dispositivo**	**Equipo**	**Esclavo**	**Puerto**	**Realizar?**
2	Multilin 489	G1	4	1	1
3	Multilin 489	G2	1	1	1
4	Multilin 489	G3	3	1	1
5	Multilin 750	OCB	5	2	1
6	Multilin 745	T2	2	1	1
7	SQUARE-D Motor Logic Plus II	P-109	2	2	1
8	SQUARE-D Motor Logic Plus II	P-107	3	1	1
9	EATON IQ100	C-101A	10	1	1
10	EATON IQ100	VTI	11	1	1
11	EATON IQ100	P-38	12	1	1
12	Basler DECS-250	G1	1	2	1
13	Basler DECS-250	G2	102	1	1
14	Basler DECS-250	G3	103	1	1

Figura 35- Evento de Usuario “Actualización tabla Excel”. [Fuente propia]

Segunda hoja: “CONFIGURACION DE PUERTOS”

- ✓ Primera fila la coloca como encabezado de Columna de la tabla “Configuración Puertos”.
- ✓ De la fila 2 hasta la última fila, son los datos escritos en la tabla “Configuración Puertos”, en el siguiente orden según la columna.
    1. Puerto.
    2. Baud Rate.
    3. Data Bits.
    4. Parity.
    5. Stop Bits. Ver imagen 36:



	A	B	C	D	E
1	**Puerto**	**Baud Rate**	**Data Bits**	**Parity**	**Stop Bits**
2	1	19200	8	None	1
3	2	19200	8	None	1
4	3	19200	8	None	1
5	4	19200	8	None	1
6					
7					

Figura 36- Configuración de puertos Excel. [Fuente propia]

Las demás hojas se unen una a continuación de la otra para armar una sola tabla con todos los dispositivos, y los escribe en la Tabla “Rutas de Dispositivos”, en el siguiente orden según la columna.

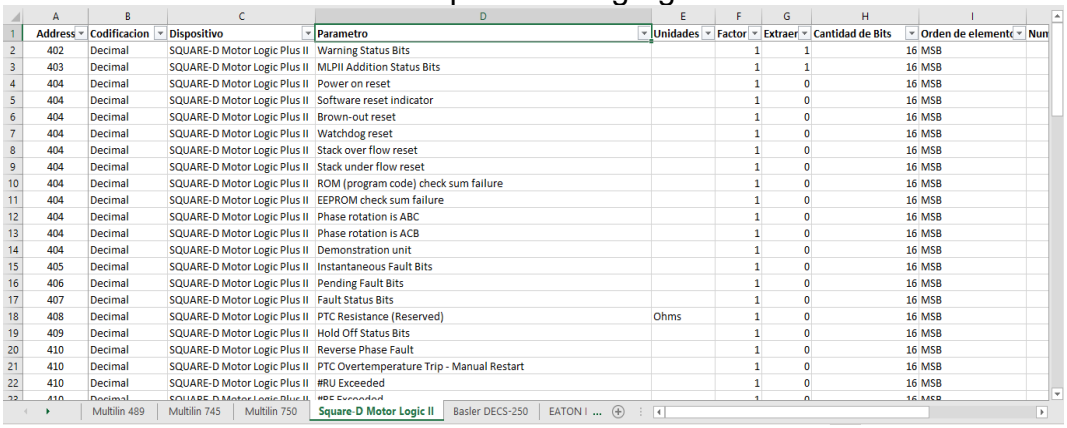
1. Address.
2. Codificación.
3. Dispositivo.
4. Parámetro.
5. Unidades.

6. Factor.
7. ¿Extraer?
8. Cantidad de Bits.
9. Orden de elementos.
10. Número de Elementos.
11. Representación Negativo.
12. Decimales.
13. Offset.
14. Length.
15. Comentarios.

2. Envía al Evento de Usuario "**Actualización Tabla de Parámetros**".

9.1.3. Evento de usuario "Actualización Tabla de Parámetros"

1. Modifica la Tabla "Rutas de Dispositivos" agregando al final de las columnas



	A	B	C	D	E	F	G	H	I
1	Address	Codificacion	Dispositivo	Parametro	Unidades	Factor	Extraer	Cantidad de Bits	Orden de element	Num
2	402	Decimal	SQUARE-D Motor Logic Plus II	Warning Status Bits		1	1	16	MSB	
3	403	Decimal	SQUARE-D Motor Logic Plus II	MLPII Addition Status Bits		1	1	16	MSB	
4	404	Decimal	SQUARE-D Motor Logic Plus II	Power on reset		1	0	16	MSB	
5	404	Decimal	SQUARE-D Motor Logic Plus II	Software reset indicator		1	0	16	MSB	
6	404	Decimal	SQUARE-D Motor Logic Plus II	Brown-out reset		1	0	16	MSB	
7	404	Decimal	SQUARE-D Motor Logic Plus II	Watchdog reset		1	0	16	MSB	
8	404	Decimal	SQUARE-D Motor Logic Plus II	Stack over flow reset		1	0	16	MSB	
9	404	Decimal	SQUARE-D Motor Logic Plus II	Stack under flow reset		1	0	16	MSB	
10	404	Decimal	SQUARE-D Motor Logic Plus II	ROM (program code) check sum failure		1	0	16	MSB	
11	404	Decimal	SQUARE-D Motor Logic Plus II	EEPROM check sum failure		1	0	16	MSB	
12	404	Decimal	SQUARE-D Motor Logic Plus II	Phase rotation is ABC		1	0	16	MSB	
13	404	Decimal	SQUARE-D Motor Logic Plus II	Phase rotation is ACB		1	0	16	MSB	
14	404	Decimal	SQUARE-D Motor Logic Plus II	Demonstration unit		1	0	16	MSB	
15	405	Decimal	SQUARE-D Motor Logic Plus II	Instantaneous Fault Bits		1	0	16	MSB	
16	406	Decimal	SQUARE-D Motor Logic Plus II	Pending Fault Bits		1	0	16	MSB	
17	407	Decimal	SQUARE-D Motor Logic Plus II	Fault Status Bits		1	0	16	MSB	
18	408	Decimal	SQUARE-D Motor Logic Plus II	PTC Resistance (Reserved)	Ohms	1	0	16	MSB	
19	409	Decimal	SQUARE-D Motor Logic Plus II	Hold Off Status Bits		1	0	16	MSB	
20	410	Decimal	SQUARE-D Motor Logic Plus II	Reverse Phase Fault		1	0	16	MSB	
21	410	Decimal	SQUARE-D Motor Logic Plus II	PTC Overtemperature Trip - Manual Restart		1	0	16	MSB	
22	410	Decimal	SQUARE-D Motor Logic Plus II	#RU Exceeded		1	0	16	MSB	

Multilin 489 | Multilin 745 | Multilin 750 | Square-D Motor Logic II | Basler DECS-250 | EATON I ...

Figura 37- Código del programa encargado de modifica la Tabla. [Fuente propia]

Al llegar a este punto las variables están lista para su extracción y dependiendo de la opción digitada en columna EXTRAER esta será realizada en SUBVI extracción de data RTU o TCP.

7. Se colocan en invisibles (Ámbar) ver figura 38 las tablas RTU y TCP que no estén seleccionadas y que no posean dato de extracción.

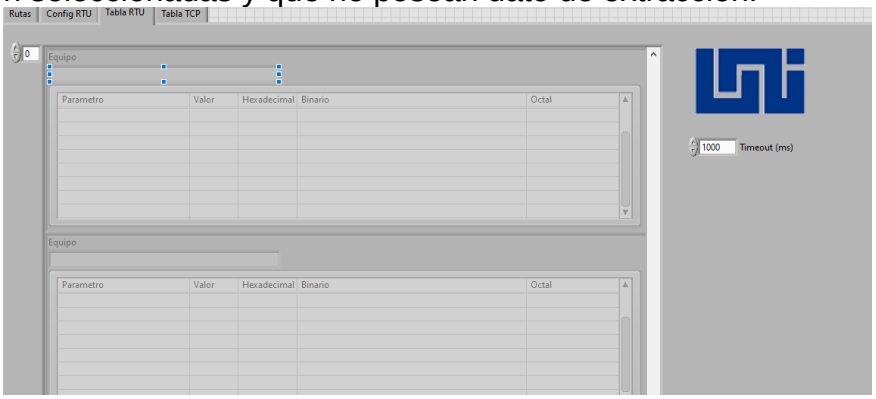


Figura 38- Pantalla del programa ejecutable tablas RTU. [Fuente propia]

8. En caso de que exista parámetro a extraer y el equipo este correctamente conectado programa establecerá enlace en bucle con el dispositivo y nuestra PC será un dispositivo Maestro, tabla de parámetros será visible y dato será visualizado en línea.

9. En tabla de extracción RTU se crearon 5 columnas:

1- Parámetro: Descripción específica del parámetro extraído que se escribió en tabla de EXCEL.
2- Valor: Valor real extraído en línea del dispositivo seleccionado.
3- Hexadecimal, Binario, Octal: Se dejarán datos crudos en línea de como seria la forma en que se trasmite y reciben estos parámetros de manera digital sin acondicionamiento de variable.

10. En tabla de extracción TCP se crearon 4 columnas:

1- Valor: Valor real extraído en línea del dispositivo seleccionado.
2- Hexadecimal, Binario, Octal: Se dejarán datos crudos en línea de como seria la forma en que se trasmite y reciben estos parámetros de manera digital sin acondicionamiento de variable ver figura 39:

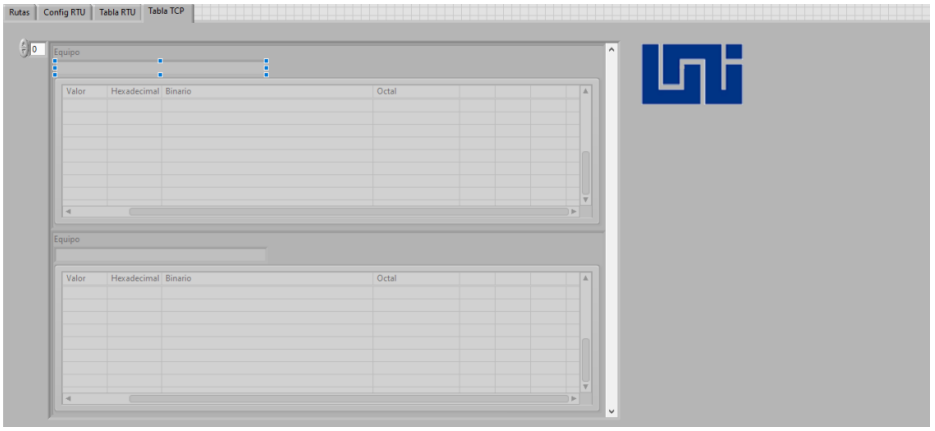
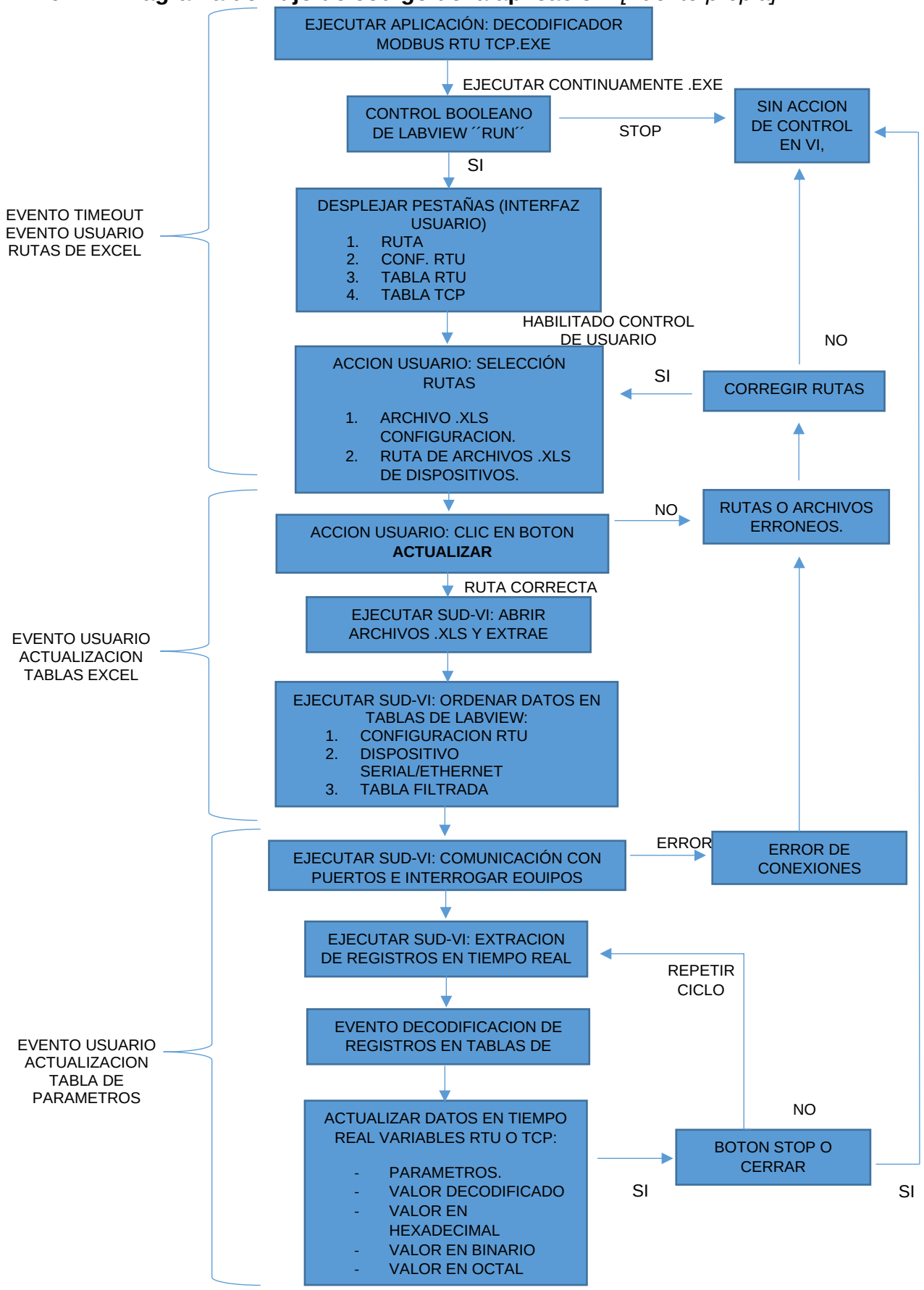


Figura 39- Pantalla del programa ejecutable tablas TCP. [Fuente propia]

9.2. Diagrama de flujo de código de la aplicación. *[Fuente propia]*



9.3. Aplicación ejecutable en LABVIEW.

El código del programa completo estará en anexos al final de este documento, al completar el proceso de extracción de datos de manera exitosa y optimizada, se creó un ejecutable que es capaz de correr con mínimos recursos en cualquier ordenador. Una de las grandes ventajas que ofrece LabVIEW, además de la gran diversidad de herramientas al momento de desarrollar, es que se puede crear un ejecutable de la aplicación desarrollada, lo cual permitirá utilizar dicha aplicación sin la necesidad de abrir LabVIEW y ejecutarla sin necesidad de tener la licencia de este, los detalles y procedimientos empleados para lograr crear el ejecutable se detalla a continuación:

9.3.1. Requerimientos Software

- NI LabVIEW 13.0 Aplication Builder o superior

9.3.2. Requerimiento de Hardware

- Computadora con LabVIEW 2013 o superior.

Al completar la programación de la aplicación la cual nombramos **Decodificador MODBUS**, se logró crear el ejecutable para dicha aplicación y una vez teniendo el ***Aplication Builder*** en LabVIEW ejecutarla de forma gratuita sin licencias de desarrollador y así ser utilizarlas en nuestra prueba piloto y también en las prácticas de laboratorio FEC.

9.3.3. Desarrollo

1. Abrir *NI LabVIEW* y crear un nuevo proyecto, dando clic en el botón *Create Project*, mostrado en la Figura 40.

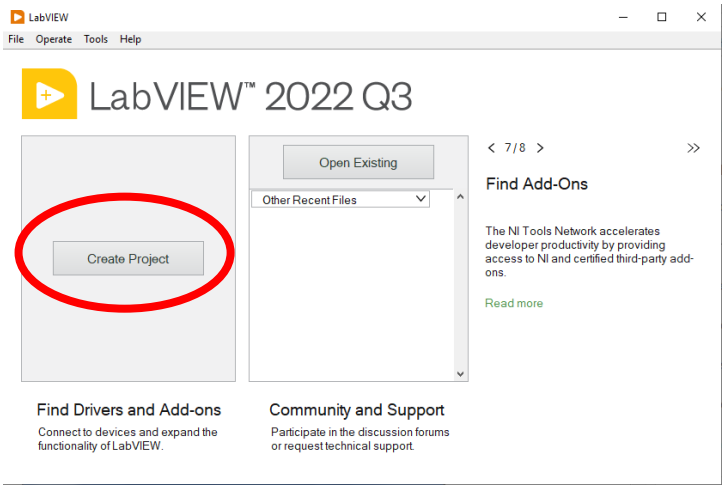


Figura 40- Crear un nuevo proyecto en LabView. [Fuente propia]

2. Seleccionar la opción de ***Blank Project*****,** como se muestra en la figura 41. y dar clic en el botón *Finish*.

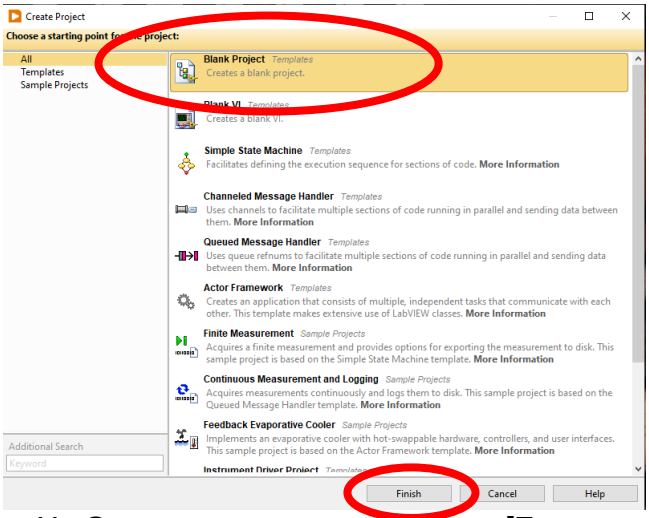


Figura 41- Crear un proyecto en blanco. [Fuente propia]

3- Dar clic secundario en *My Computer*, y seleccionar New/VI como se muestra en la Figura 42. Esto creará un nuevo VI vacío.

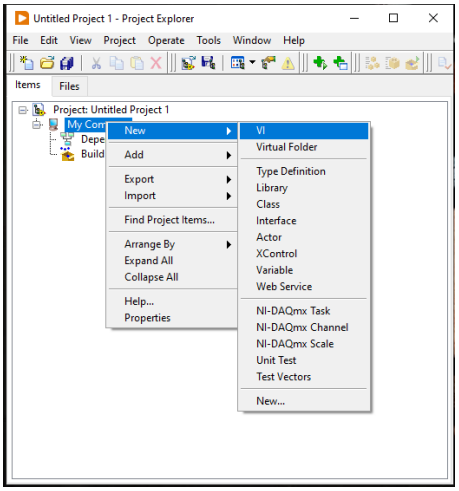


Figura 42- Creando un nuevo VI en un proyecto. [Fuente propia]

4- Agregar un ciclo while que abarque todos los elementos de nuestro panel frontal, y cuya condición de término esté conectada a una constante booleana con el valor de false, como se muestra en la Figura 43.

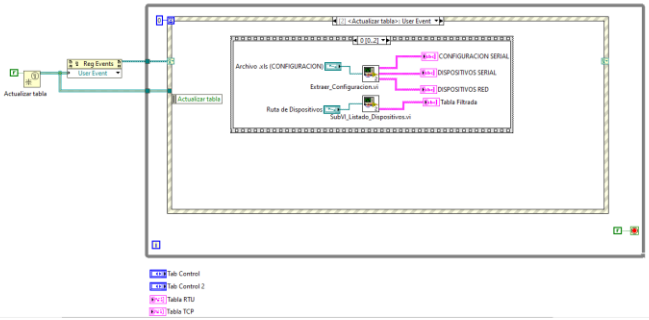


Figura 43- Creando ciclo while de nuestra aplicación. [Fuente propia]

5- Regresar al panel frontal, y personalizarlo a su propio gusto, como se muestra en la Figura 44.

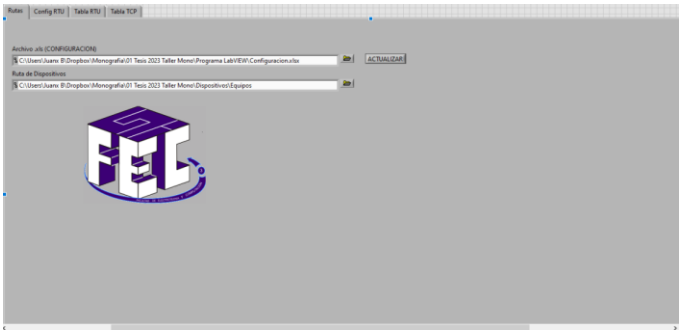


Figura 44- Panel principal desarrollado. [Fuente propia]

6- Correr el VI y verificar su funcionamiento. Guardar el VI y el proyecto como **Decodificador MODBUS.**
7- Abrir el proyecto llamado MODBUS RTU TCP que creó.
8- En la pantalla del proyecto, dar clic en Tools y luego seleccionar en *Build Application (.EXE).*

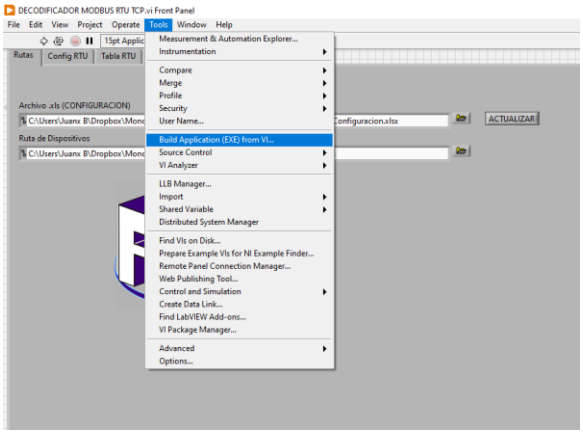


Figura 45- Creando un BUILD. [Fuente propia]

9- Se cambio el nombre de la aplicación **DECODIFICADOR MODBUS RTU TCP** como se muestra en la figura 48. y agregamos ruta de destino.

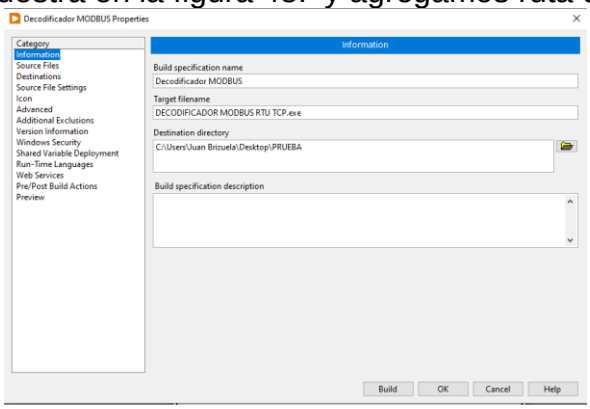


Figura 46- Cambiando el nombre de la aplicación. [Fuente propia]

10-En la pestaña de Source Files, agregar el VI generado como se muestra en la figura 47. Dar clic en OK.

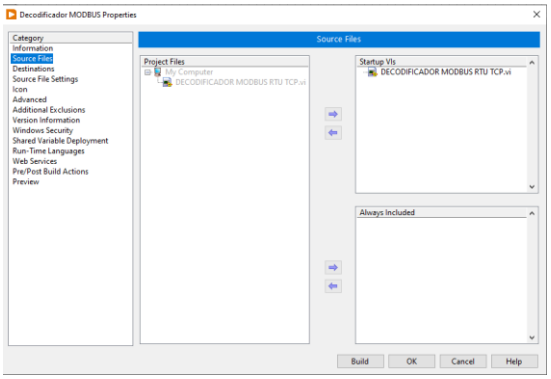


Figura 47- Agregando Decodificador MODBUS RTU TCP.vi como Startup VIs. [Fuente propia]

11-Regresar a la pantalla del proyecto, dar clic secundario en la aplicación previamente creada y seleccionar la opción *Build*, como se muestra en la figura 48.

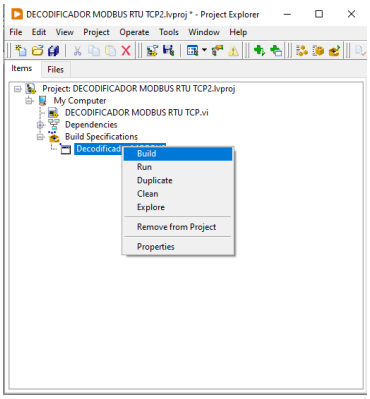


Figura 48- Construyendo ejecutable en LabVIEW. [Fuente propia]

12-Cerrar *LabVIEW* guardando cambios, entrar a la ruta destino y ejecutar la aplicación DECODIFICADOR MODBUS RTU TCP.exe

13- Verificar que aplicación funciona sin la necesidad de abrir *LabVIEW*.

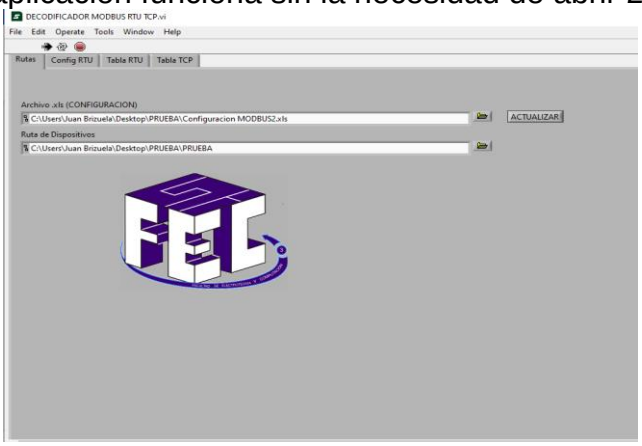


Figura 49- Aplicación Ejecutable finalizada. [Fuente propia]

9.4. Diseño de prototipo y hardware del proyecto.

Para que la prueba piloto se pudiera implementar fue necesario el diseño, construcción e implementación de topología de red en un prototipo que cuenta con un equipo con protocolo de comunicación MODBUS TCP y otro con MODBUS RTU. También se pretende que esta aplicación y prototipo puedan aportar en futuras prácticas de laboratorios las cuales también se diseñaron en este documento.

9.4.1. Listado y costos de equipos utilizados.

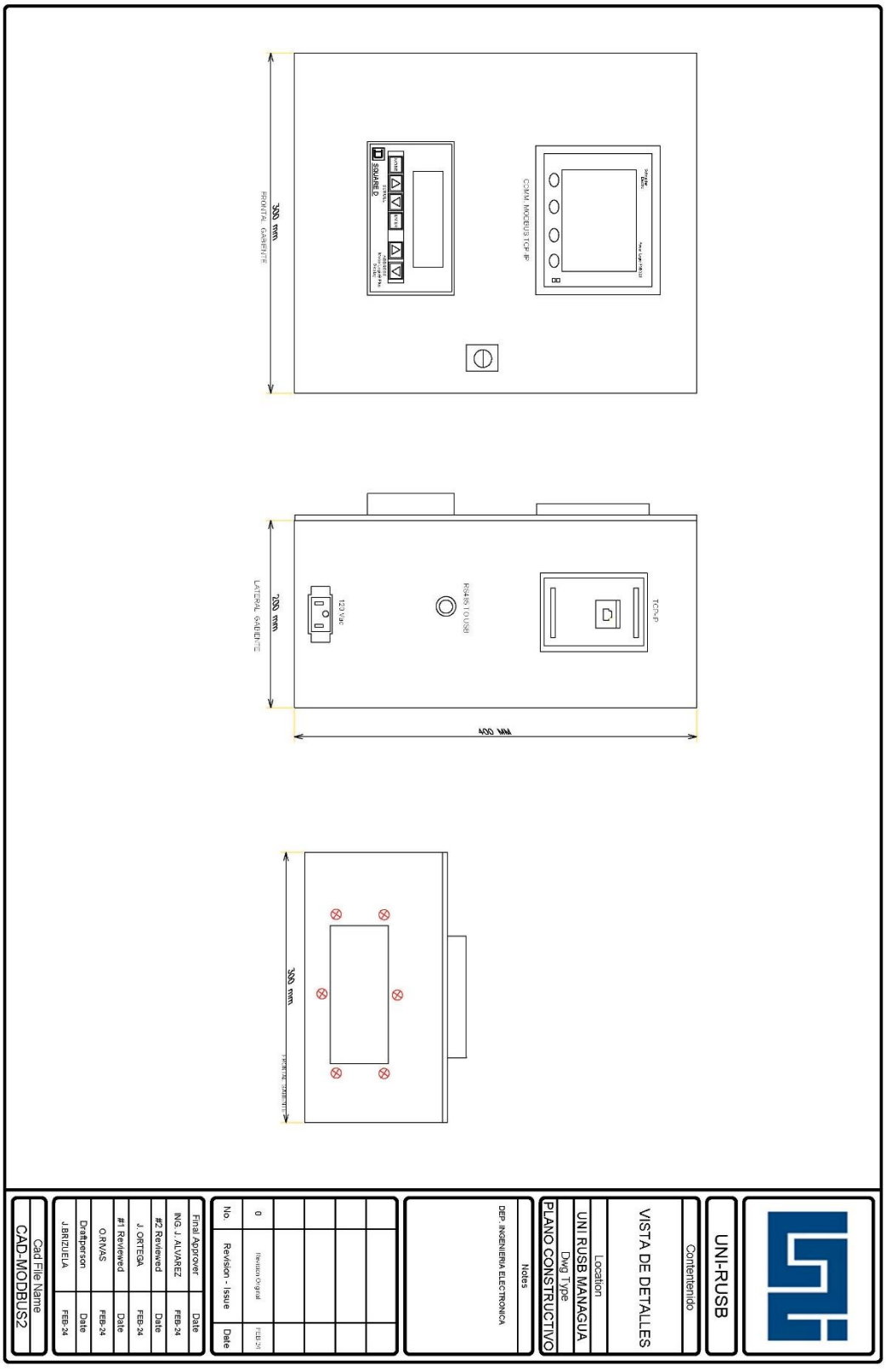
Tabla 28- Listado y costos de hardware implementado. *[Fuente propia]*

Ítem	Materiales / Equipos	Cant.	Costo aprox.
1	Gabinete eléctrico NewTrend 300x400x200	1	$ 48.05
2	JACK QUICKPORT CAT6A RJ-45 LEVITON:CAT6A: AZUL Mas protector pared.	1	$ 7.6
3	Convertidor Microflex 101-0020 USB a RS-485	1	$ 90
4	PATCH CORD CAT6 7FT AZUL NEXXT	1	$ 6.18
5	Riel din o Carril simétrico perforado metros	1	$ 5
6	BORNE DE PASO, UT 2,5 32A, 0.14-4MM2(26-12AWG), TORNILLO, A=5.2MM	40	$ 12
7	Borne de tierra para carril, Conexión por tornillo.	12	$ 4
8	Puente t/peine 10 terminales paso 5mm rojo.	1	$ 1.8
9	Puente t/peine 10 terminales paso 5mm azul.	1	$ 1.8
10	Fuente de alimentación UNO-PS/1AC/24DC/ 30W	1	$ 14.90
11	Breaker riel din 1 Polo 1 Amperio Curva C Acti9 iC6	2	$ 16
12	Cableado de control flex multifilar #14 color rojo	10	$ 4
13	Cableado de control flex multifilar #14 color negro	10	$ 4
14	Cableado de control flex multifilar #14 color blanco	10	$ 4
15	Medidor de potencia Schneider eléctrico Power Logic.PM5300	1	$822
16	Motor Logic® Plus II Programmable Solid-State Overload Relay, Class 9065 solid-state overload relay	1	$ 650
Total.			$ 1,689.53

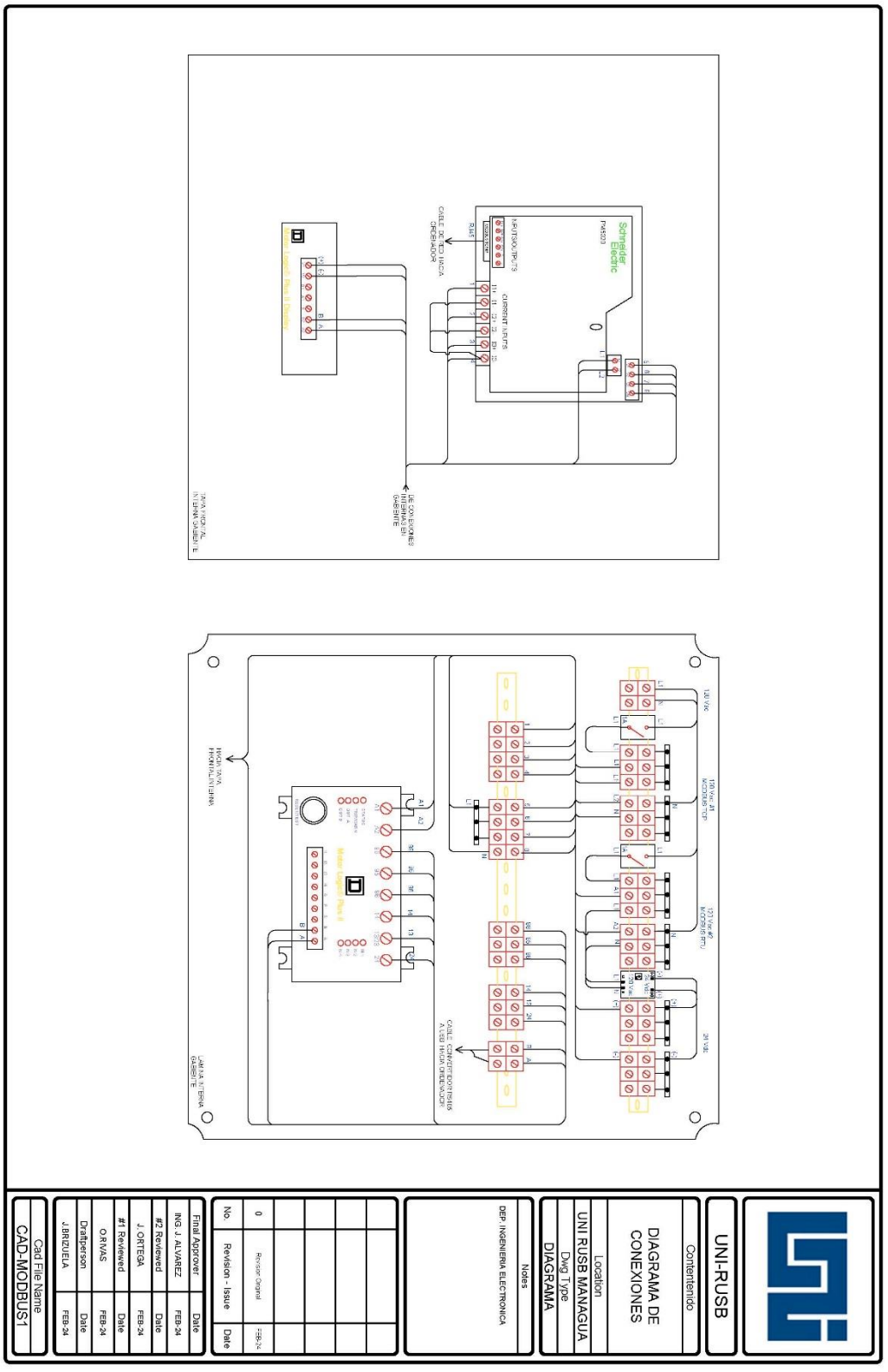
9.4.2. Diseño constructivo y cableado del prototipo

Al completar el proceso de selección de los equipos, se realizado el diseño y conexionado del hardware como equipos seleccionados contamos con **Medidor de potencia Schneider eléctrico METSEPM5320.Power Logic.PM5300** y **Motor Logic® Plus II Programmable Solid-State Overload Relay,** a continuación, se utilizó el software de AutoCAD 2021 para este propósito y los planos finales fueron los siguientes:

9.4.2.1. Plano AutoCAD #1, vistas frontales y detalles constructivos



9.4.2.2. Plano AutoCAD #2, Conexiones eléctricas y bus de campo MODBUS.



9.4.3. Detalle del prototipo final.

Hardware final del proyecto que será un complemento a la aplicación ejecutable de extracción de variables se pueden apreciar en figura 50 y 51:

Figura 50- Fotografía externa de prototipo de laboratorio. [Fuente propia]

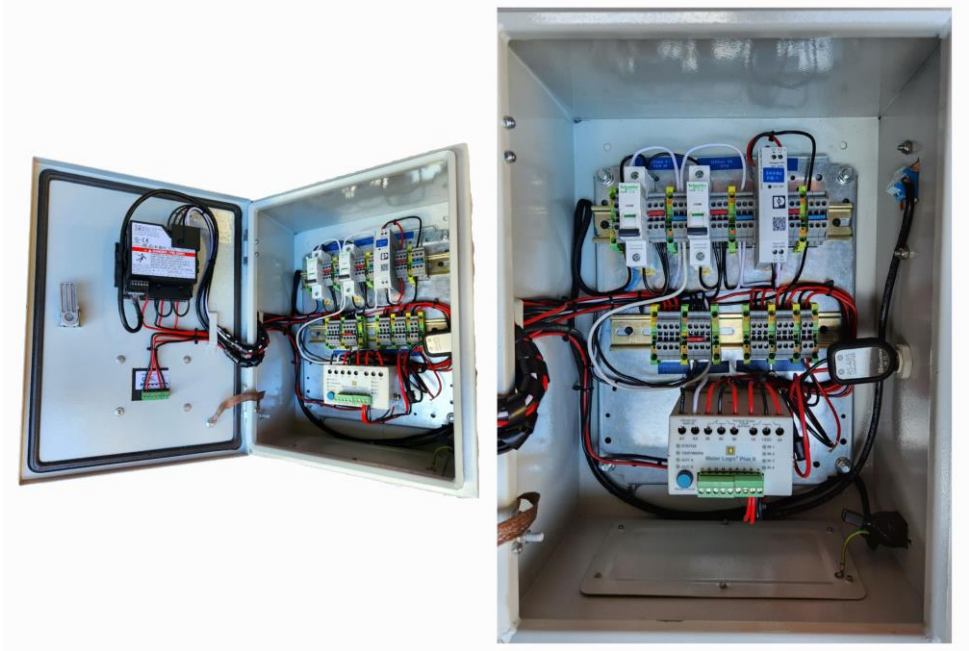


Figura 51- Fotografía interna y detales de cableado. [Fuente propia]

9.5. Resultados finales de prueba piloto.

Cuando se finalizó tanto la aplicación ejecutable y el prototipo de pruebas, se ha demostrado la extracción de variables con una prueba piloto seleccionando variables al azar con los equipos: #1 **Motor Logic® Plus II RTU** y #2 **Medidor Power Logic.PM5300 TCP** de forma exitosa, detalles de prueba a continuación:

1. Se verificó la correcta conexión de comunicación RTU en terminales de red bornes A y B del dispositivo con el convertidor USB a RS485 Microflex y la correcta conexión de comunicación del cable TCP del dispositivo.

2. Se instaló el driver correspondiente Driver Microflex: 101-0019 y verificar en el ordenador que puerto COM se habilita al conectar el convertidor "Administrador de dispositivos". Ejemplo: COM1.

3. Se Configuró el puerto habilitado COM1 haciendo clic derecho y seleccionando "propiedades" con los siguientes parámetros del dispositivo: Baud Rate: 19200, Data Bits: 8, Parity: None, Stop Bits: 1 y también se verifico la dirección IP del ordenador o tarjeta de red en rango de IPs con el dispositivo. (Panel de control\Redes e Internet\Conexiones de red→ clic derecho en puerto de red → Propiedades→ En pestaña funciones de red seleccionar →Habilitar el protocolo de internet versión 4 (TCP\IPv4) en este punto se digito una dirección IP estática en mismo segmento que PM5320 IP estática: 169.254.189.34.

4. Se verificó la correcta conexión de alimentación de ambos dispositivos en bornes A1 y A2 y se habilitaron los autómatas o minibreakers de control A1 y A2 para posterior conectar toma 120 Vac del gabinete.

5. Se completo la instalación del Driver LVRumTimeEng de national Instruments y NI Modbus Library by NI - Toolkit for LabVIEW.

6. Se creo una carpeta en el escritorio titulada "Prueba 1 RTU y TCP" y editar dos documentos de Excel compatibles para (Libro de Excel 97-2003 (*xls)) El Primero se tituló "Configuración puertos Modbus", se creó una sud-carpeta titulada Dispositivos y se editaron los siguientes documentos de Excel titulados "SQRTU-1" y el segundo "PMTCP-1".

7. Se Lleno el documento de Excel "Configuración puertos Modbus" el cual contendrá 3 hojas de cálculo con el siguiente formato: 1er hoja de cálculo titularla: CONFIGURACION SERIAL ejemplo tabla 29. Se relleno con los siguientes parámetros:

Tabla 29- Hoja de cálculo Configuración de puertos prueba 1. [Fuente propia]

Puerto	Baud Rate	Data Bits	Parity	Stop Bits
1	19200	8	None	1

- 2da hoja de cálculo titularla: DISPOSITIVOS SERIAL ejemplo tabla 30. Se relleno con los siguientes parámetros:

Tabla 30- Hoja de cálculo Listado de dispositivos serial prueba 1. [Fuente propia]

Dispositivo	Equipo	Esclavo	Puerto	¿Realizar?
SQUARE-D Motor Logic Plus II	SQRTU-1	3	1	1

- 3er hoja de cálculo titularla: DISPOSITIVOS RED ejemplo tabla 31. Se relleno con los siguientes parámetros:

Tabla 31- Hoja de cálculo Listado de dispositivos TCP prueba 1. [Fuente propia]

Dispositivo	Equipo	ID	Puerto	IP	¿Realizar?
PM5320	PMTCP-1	255	502	169.254.189.34	1

8. Con el apoyo de la hoja de datos o manual del dispositivo Motor Logic Plus II y PM5320 se procedieron a llenar las hojas de cálculo donde especificaremos los parámetros o características operativas de nuestro dispositivo para la prueba.

9. Se Lleno el documento de Excel: “SQRTU-1” y el segundo “PMTCP-1”, ejemplo de diseño tabla 33, siguiendo el siguiente formato:

1- Address.
2- Codificación.
3- Dispositivo.
4- Parámetro.
5- Unidades.
6- Factor.
7- Extraer.
8- Cantidad de Bits.
9- Orden de elementos.
10- Número de elementos.
11- Representación Negativo.
12- Decimales.
13- Offset.
14- Length.
15- Comentarios.

Tabla 32- Archivo de EXCEL #2 DISPOSITIVOS Hoja #1. [Fuente propia]

Address	Codificación	Dispositivo	Parámetro	Unidades

Factor	¿Extraer?	Cantidad de Bits	Orden de elementos	Número de elementos

Representación Negativos	Decimales	Offset	Length	Comentarios

10. Para esta prueba solo extrajimos 6 parámetros para el dispositivo RTU:
    - Registro 40411, Address 19A, Bit 12: Test Trip.
    - Registro 40414, Address 19D, Bit 0: Output A is On or Closed
    - Registro 40414, Address 19D, Bit 1: Output B is On or Closed.
    - Registro 40430, Address 1AD, Corriente Fase C
    - Registro 40431, Address 1AE, Corriente Fase B
    - Registro 40432, Address 1AF, Corriente Fase A.

11. Se lleno hoja de Excel según manual de dispositivo ejemplo 1:

    - Address: 19A
    - Codificación: Hexadecimal.
    - Dispositivo: SQUARE-D Motor Logic Plus II
    - Parámetro: Test Trip
    - Unidades: Ninguno.
    - Factor: 1 (Multiplicador de la variable)
    - Extraer: 1 es que si y 0 es que no. Digitar 1
    - Cantidad de Bits: 16 bits (Numero de bits del mensaje)
    - Orden de elementos: MSB o LSB, digitar MSB.
    - Número de elementos: 1 o 2 registros, digitar: 1
    - Representación Negativo: Punto Flotante SINT32; Complemento A1; Complemento A2; Ninguno, digitar: Ninguno.
    - Decimales: digital 0 para este ejemplo.
    - Offset: número de bits de la trama requerido: 12
    - Length: Longitud de bits después del Offset: 1
    - Comentarios: registro de lectura prueba 1.

12. Para la prueba con el dispositivo TCP solo extrajimos 6 parámetros:
    - Registro 403000, Address BB7, Current A
    - Registro 403002, Address BB9, Current B
    - Registro 403004, Address BBB, Current A
    - Registro 403028, Address BD3, Voltage A-N
    - Registro 403030, Address BD5, Voltage B-N
    - Registro 403032, Address BD7, Voltage C-N

13. Se lleno la segunda hoja de Excel según manual de dispositivo:

    - Address: **BD3**
    - Codificación: **Hexadecimal.**
    - Dispositivo: **PM5320.**
    - Parámetro: **Voltaje A-N**
    - Unidades: **Vac**
    - Factor: **0.5** (Multiplicador de la variable)
    - Extraer: 1 es que si y 0 es que no. Digitar **1**
    - Cantidad de Bits: **32** bits (Numero de bits del mensaje)
    - Orden de elementos: MSB o LSB, digitar: **MSB.**
    - Número de elementos: 1 o 2 registros, digitar**: 1**
    - Representación Negativo: Punto Flotante SINT32; Complemento A1; Complemento A2; Ninguno, digitar: **Punto Flotante SINT32**
    - Decimales: digital **2** para este ejemplo.
    - Offset: número de bits de la trama requerido: **0**
    - Length: Longitud de bits después del Offset: **8**
    - Comentarios: **Registro de lectura prueba 2.**

14. Se ejecuto el programa **DECODIFICADOR MODBUS RTU TCP.exe.**

15. En la primera ventana del ejecutable (ver figura 52) se selecciona la ruta de los archivos de Excel creados, Configuración puertos Modbus y ruta de carpeta de las hojas de dispositivos y se da clic en botón **actualizar.**

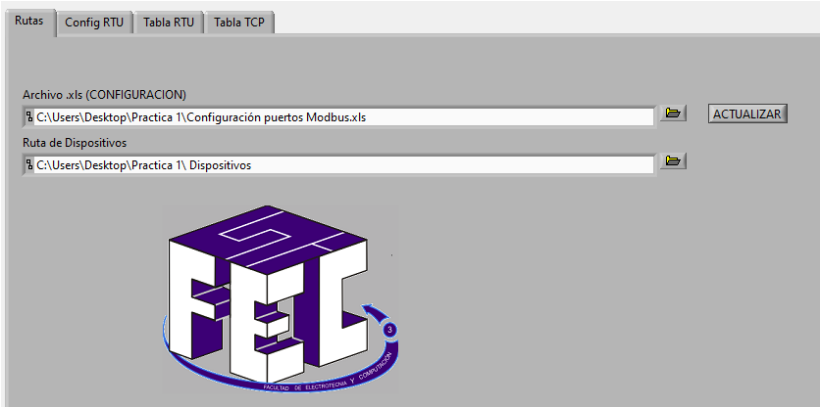


Figura 52- Selección de rutas y hojas de cálculo iniciales prueba 1. [Fuente propia]

16. En la pestaña del ejecutable Config RTU (ver figura 53) se extrajeron los archivos de Excel seleccionados y se rellenaron automáticamente en tablas de LabVIEW, se confirmó cada una de las tablas, PUERTOS, DISPOSITIVOS SERIAL/ETHERNET y TABLA FILTRADA fueron correctamente rellenadas.

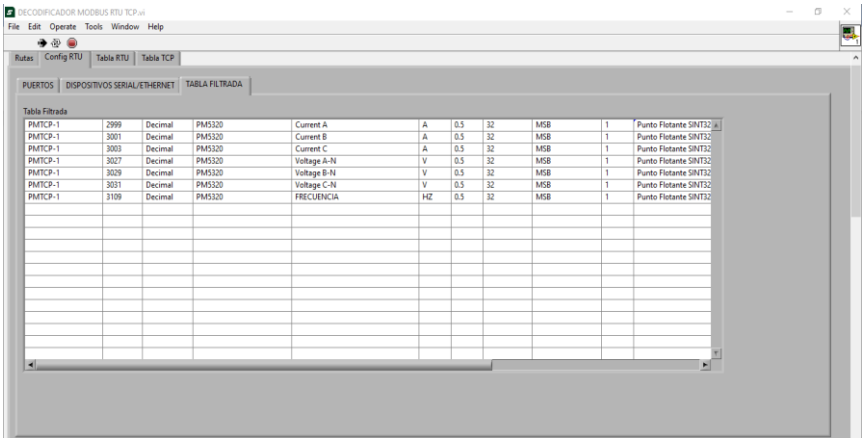
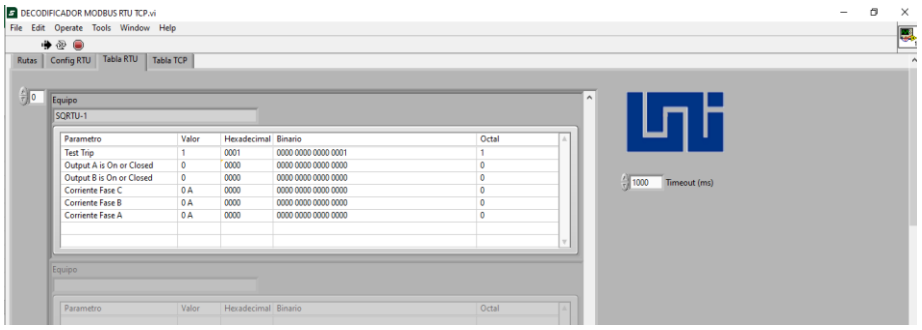


Figura 53- Extracción de variables de EXCEL. [Fuente propia]

17. Finalmente, la aplicación en la pestaña Tabla RTU o en la Tabla TCP (ver figura 54) nos entregó con éxito cada parámetro seleccionado en tiempo real del equipo, presentándonos en diferentes tramas digitales el mismo dato cuando no es tratado, Hexadecimal, binario y Octal.

A)



B)

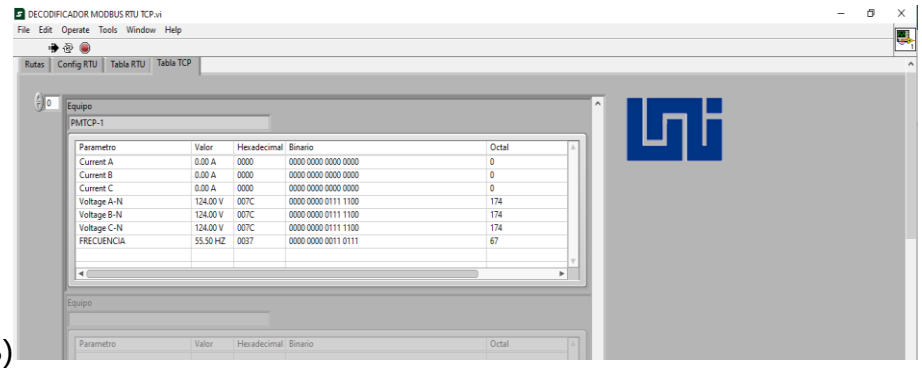


Figura 54- A) Extracción y decodificación variables en línea RTU. B) Extracción y decodificación variables en línea TCP. [Fuente propia].

La optimización de recursos fue una prioridad, por lo que se logró alcanzar una compilación de menos de 72K de tamaño en disco y un total de 169.5K de Código programable aun cuando la aplicación no era ejecutable, al crear la aplicación ejecutable bajamos el consumo de memoria y recursos, detalles en figura 55 y 56.

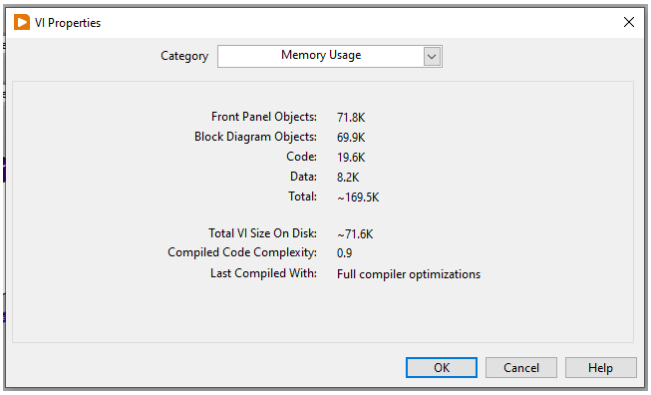


Figura 55- Información de Memoria utilizada en nuestro programa. [Fuente propia]

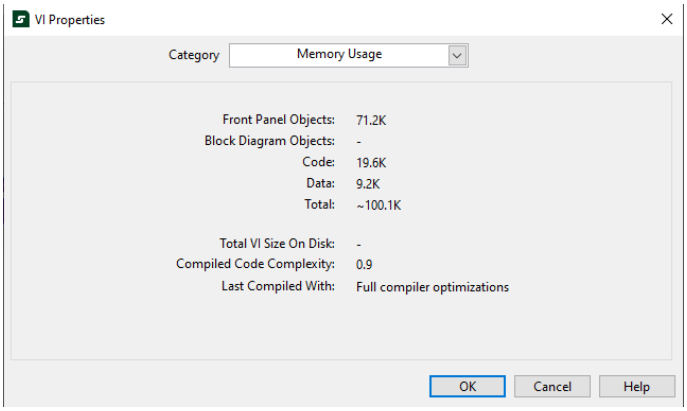


Figura 56- Información de Memoria utilizada con aplicación, reducción de memoria. [Fuente propia]

Al completar la prueba piloto de una manera eficiente y segura simulando los procedimientos de los laboratorios de la UNI, se completó el diseño dos prácticas de laboratorio las cuales tienen conceptos básicos del protocolo, detalla la manera de tratar los registros Modbus y servirá de referencia académica para conocimientos básicos de buses de campo a posibles proyectos de integración a futuro.

La práctica de laboratorio sigue los estándares de las guías que se utilizan actualmente en departamento de ingeniería de la facultad, tratando de simplificar este estudio de tesis, por esta razón también las imágenes y tablas adjuntas en laboratorios no siguen la secuencia de índice de contenido del documento y se encuentran en **anexos.**

X. CONCLUSIONES

En el presente documento se diseñó una aplicación ejecutable para poder extraer y tratar variables cualesquiera de equipos con protocolo de comunicación Modbus y que la variabilidad de marcas y tablas de registros no sea un problema al momento de integración para futuros estudios o proyectos. Para que el estudio y guías realizadas tengan un aporte significativo dentro de la formación de la Universidad Nacional de Ingeniería es importante que los estudiantes refuercen y conozcan de buses de campo industrial, los diferentes métodos para transmitir datos digitales, como se ejecuta e interpretan, así como incrementar los conocimientos básicos de las telecomunicaciones digitales que son la base para cualquier protocolo.

Se completo el análisis de estudio del protocolo de comunicación MODBUS en todas sus tramas y redes de transmisión, así como sus beneficios como arquitectura, esto permitió diseñar la aplicación en LABVIEW que en conjunto a software Microsoft Excel se puedan enviar y recibir datos de equipos con este protocolo incorporado.

Se determinaron todos los parámetros y se diseñó una tabla en el software Microsoft Excel para preparar cualquier variable que se desee extraer de cualquier dispositivo y finalmente el programa ejecutable ser el encargado del tratamiento de variables finales con los mínimos recursos de un ordenador.

Se realizaron pruebas pilotos con los equipos Motor LogicScuareD II y pm5320 schneider con la aplicación ejecutable diseñada y tanto la extracción y decodificación de variables fueron exitosas, con la finalidad que estas pruebas pilotos tengan más relevancia se completó el diseño de dos prácticas de laboratorio y un prototipo de pruebas para ser utilizados en la facultad de electrónica.

El trabajo en general se diseñó para los estudiantes de Ingeniería electrónica, telecomunicaciones o carreras a fines que estén interesados en el estudio o análisis de equipos que cuenten con este protocolo de comunicación o un bus de campo similar, ayudando a expandir conocimiento para entornos industriales.

XI. RECOMENDACIONES

El trabajo realizado les servirá a los estudiantes de ingeniería a fines para tener claros los posibles escenarios en las prácticas profesionales en las empresas del ámbito industrial y de comunicaciones, sin embargo, una interesante mejora sería que la Universidad pueda contar con equipos y buses de campo variados y poder no solo conocer y diagnosticar problemas de comunicación industrial si no también aportar conocimiento e innovación.

Una posible continuidad a este trabajo puede ser contar con más guías de laboratorios fusionando las comunicaciones y transmisión de datos con tecnología actual. Los equipos y aplicación que se donarán a la facultad podrán ser utilizados para estos propósitos, así como para complementar clases de digitales y comunicaciones, uno de estos que es el PM5320 un analizador de redes de energía, podría ser de complemento en laboratorios de Ing. Eléctrica.

Las tecnologías evolucionan un claro ejemplo en las comunicaciones móviles, para una futura investigación sobre este mismo tema se podría implementar comunicación Modbus con controladores Arduino o PLCs de control modernos, implementando sistemas SCADA con LABVIEW y como hemos mencionado los protocolos industriales como MODBUS no creemos que pierdan su posición dominante en un futuro y creemos que evolucionen como el caso de las móviles y debemos ser parte de este proceso.

También se recomienda el uso de equipos de otra marca como referencia en las simulaciones para reforzar conocimientos, tomar las medidas de seguridad necesarias en cada momento al conocer equipos nuevos, forma correcta de conexiones tanto en potencia como en comunicación.

XII. BIBLIOGRAFÍA

[1] Álvarez, M. (2010). Implementación de la arquitectura para automatización distribuida MODBUS-IDA para el laboratorio de automatización industrial. Riobamba-Ecuador: Escuela Superior Politécnica de Chimborazo. Disponible en: http://www.dspace.espoch.edu.ec/handle/123456789/2277

[2] Gallo, T. & Herrera, D. (2009). Diseño e implementación de una red industrial utilizando protocolo MODBUS y comunicación inalámbrica con tecnología Allen Bradley para monitoreo y control local y remoto de las estaciones de nivel, flujo y presión en el laboratorio de redes industriales y control de procesos de la ESPE extensión Latacunga. Latacunga: Universidad de las Fuerzas Armadas ESPE Extensión Latacunga.

[3] Sistemas Industriales Distribuidos. (2010). Universidad de Valencia. Disponible en: https://www.uv.es/rosado/courses/sid/sid.html

[4] Protocolo de comunicación PROFIBUS. Disponible en: https://www.profibus.com.ar/

[5] Protocolo DeviceNET Fieldbus System. Disponible en: https://www.bihl-wiedemann.de/en/company/technological-foundations/bus-systems/devicenet

[6] Introducción a Fundation Fieldbus plant web Emerson process Management.

[7] Información de Modbus organization, Inc. Disponible en: https://www.modbus.org/

[8] Ingeniería de la Automatización Industrial" Autor: Ramón Piedrahita Moreno.

[9] Protocolos de comunicación industrial librería de dispositivos. Disponible en: https://www.fieldcommgroup.org/

[10] Protocolo de comunicación Hart. Disponible en: https://www.instrumentacionuc.wixsite.com/facultad-ingenieria/copia-de-p-comunicacion

[11] Comunicación serie. Disponible en: https://www.ibm.com/docs/es/aix/7.1?topic=communications-serial-communication

[12] MasterPact MTZ - Comunicación Modbus DOCA0105ES-09 07/2022. Disponible en: https://www.productinfo.schneider-electric.com/mtzmodbuscommguide/mtz-modbus-comm-guide/ES/DOCA0105ES-09.pdf

[13] Motor Logic® Plus II Network Programming Guide boletín # 30072-451-21B 07/2006

[14] 6000 Servo Tank Gauge Intelligent tank gauge with high accuracy performance Installation and Operations Manual # boletín OM007NVAE1103

[15] Micropilot FMR20 MODBUS RS485 Radar sin contacto Para sólidos a granel. Disponible en: https://www.es.endress.com/es/instrumentacion-campo/medicion-nivel/medicion-nivel-radar-aguas?t.tabId=product-overview

[16] Marrero, C. (2006). Diseño Gráfico y Comunicación Visual. Tenerife: Universidad de La Laguna.

[17] Wesley, A. (1993). Designing the User Interface. Hoboken: Pearson.

[18] NATIONAL INSTRUMENTS CORP. Disponible en: https://www.ni.com/es-cr.html

[19] Tareas básicas en Excel. Disponible en: https://support.microsoft.com/es-es/office/tareas-b%C3%A1sicas-en-excel-dc775dd1-fa52-430f-9c3c-d998d1735fca

[20] Rautenberg, Hans (2005). “Sistemas Numéricos”. Diseño de circuitos digitales. Concepción, Chile Universidad de Concepción.

[21] USB to RS-485 Converter Manual (PDF). Disponible en: https://cdn.shopify.com/s/files/1/0555/4461/files/101-0020_USB-RS485_Converter_Manual_R4.pdf?v=1622057482

[22] Definición de concepto IP Address disponible en https://latam.kaspersky.com/resource-center/definitions/what-is-an-ip-address.

[23] DECS-250 Digital Excitation Control System Instruction Manual. Publication 9440300990 Rev U Aug 2022.

[24] Soporte Modbus información de punto flotante. Disponible en: https://www.ni.com/es/support.html.

[25] Curso en Linea de electrónica Digital I. “Representacion de números enteros y de punto flotante”. Universidad Nacional de Columbia. Disponible en: http//www.virtual.unal.edu.co/cursos/ingenieria/2000477/lecciones/010201.htm

[26] Marti Campoy, Antonio. “Representación de números enteros: el convenio complemento a uno”. Universidad Politécnica de Valencia. Disponible en: https://riunet.upv.es/bitstream/handle/10251/38421/complemento_a_uno.pdf?sequence=1

[27] Holguín G., Orozco A., & Pérez S. (2002). Curso Básico de LabVIEW 6i. Pereira: Universidad Tecnológica de Pereira.

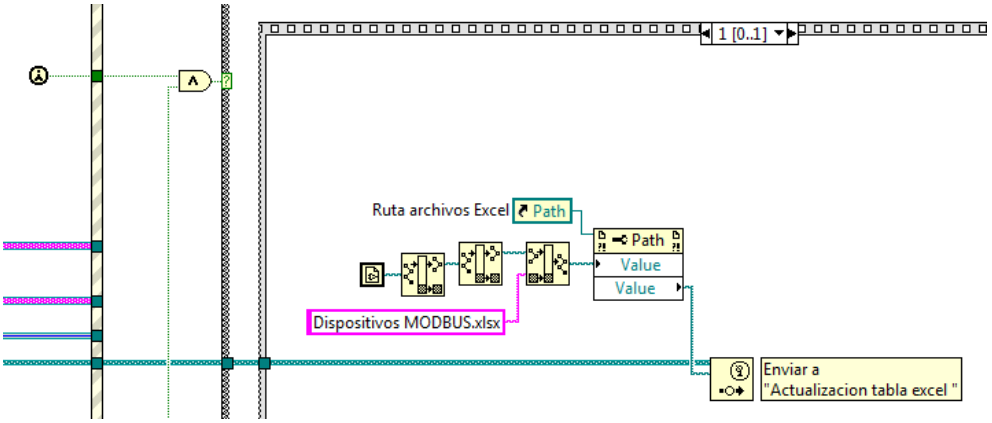
[28] National Instruments. (2003). LabVIEW Getting Started with LabVIEW. Disponible en: http://www.ni.com/pdf/manuals/323427a.pdf

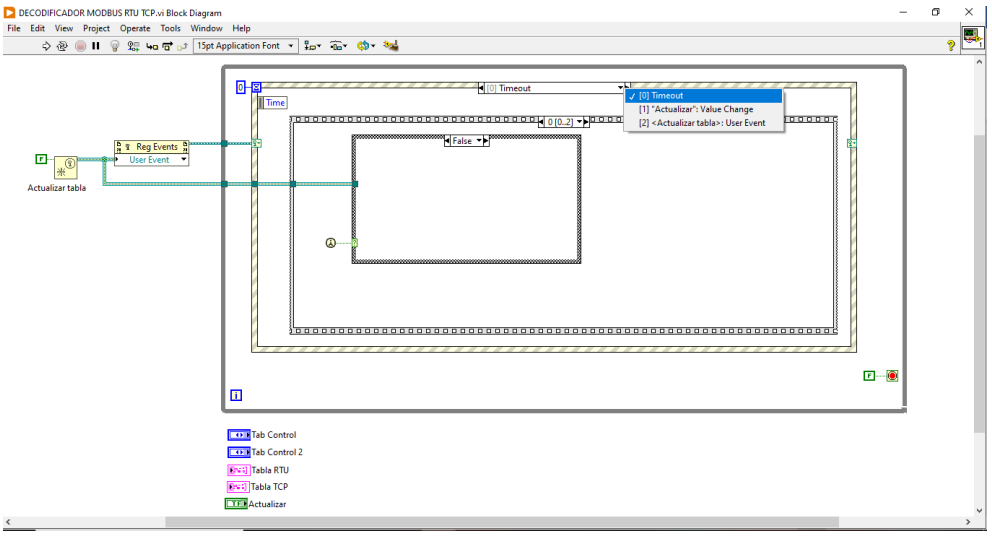
[29] LabVIEWTM Core 1 Manual de Curso, Versión del software actual 2011 Edición de agosto 2011 Número de parte 325290C-0116

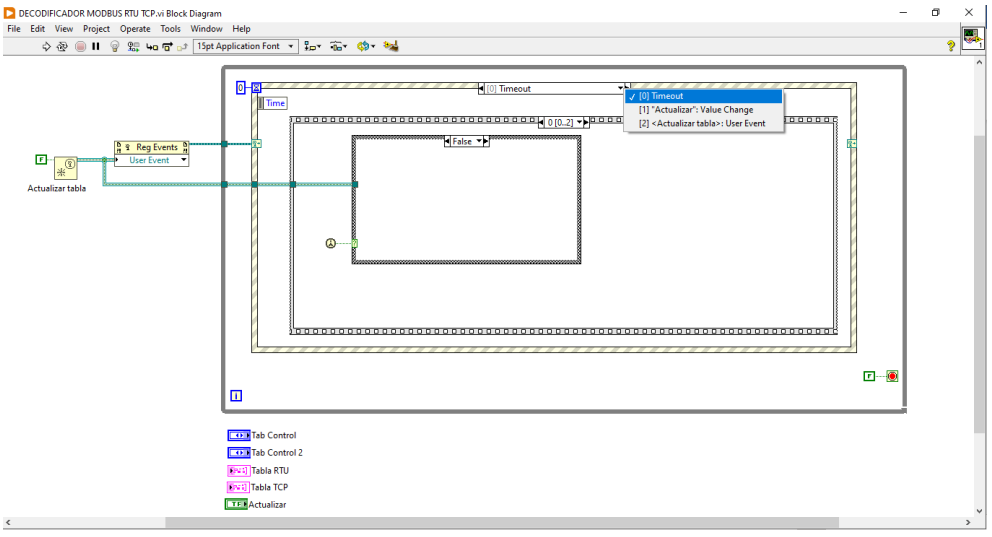
[30] LabVIEWTM Core 2 Manual de Curso, Versión del software actual 2011 Edición de agosto 2011 Número de parte 325292C-0116.

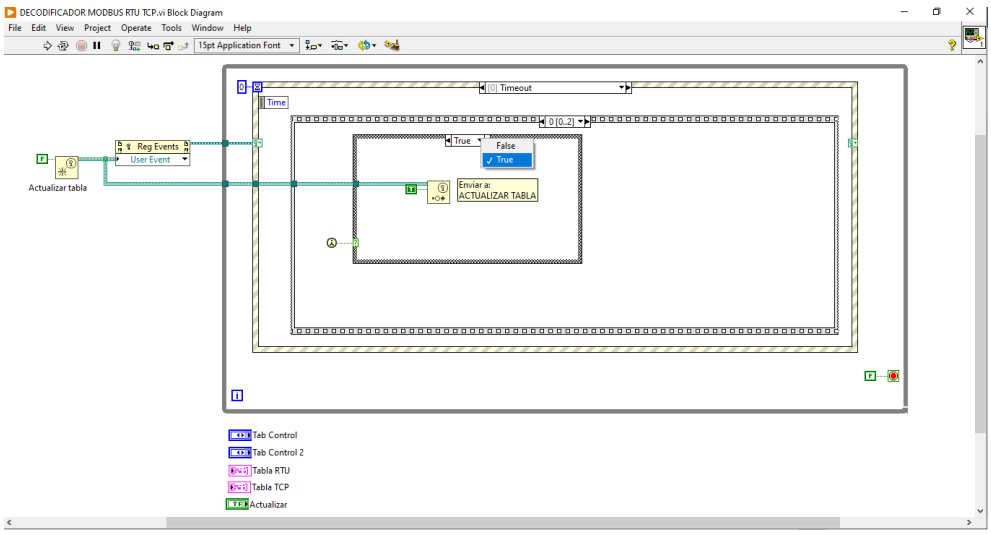
XIII. ANEXOS

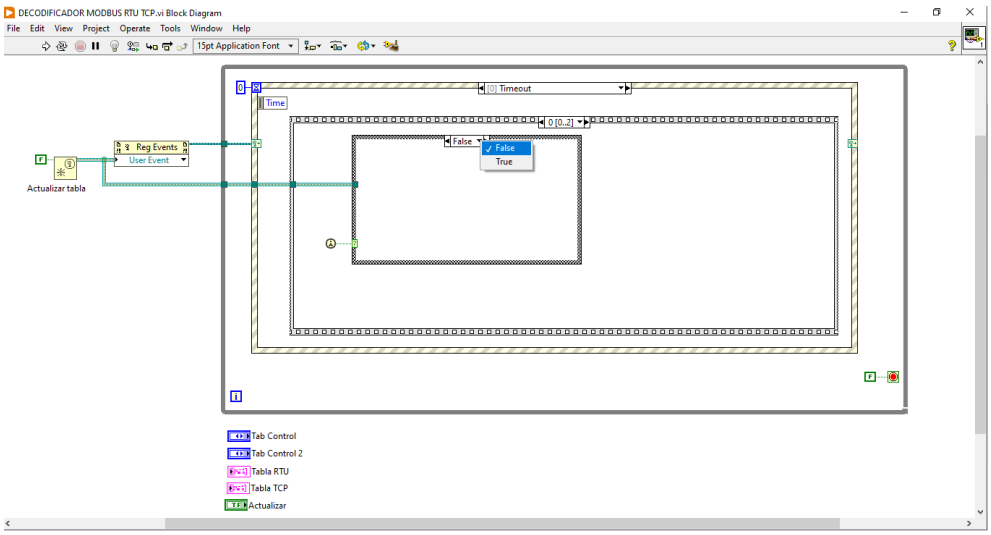
13.1. Código de programación de aplicación:

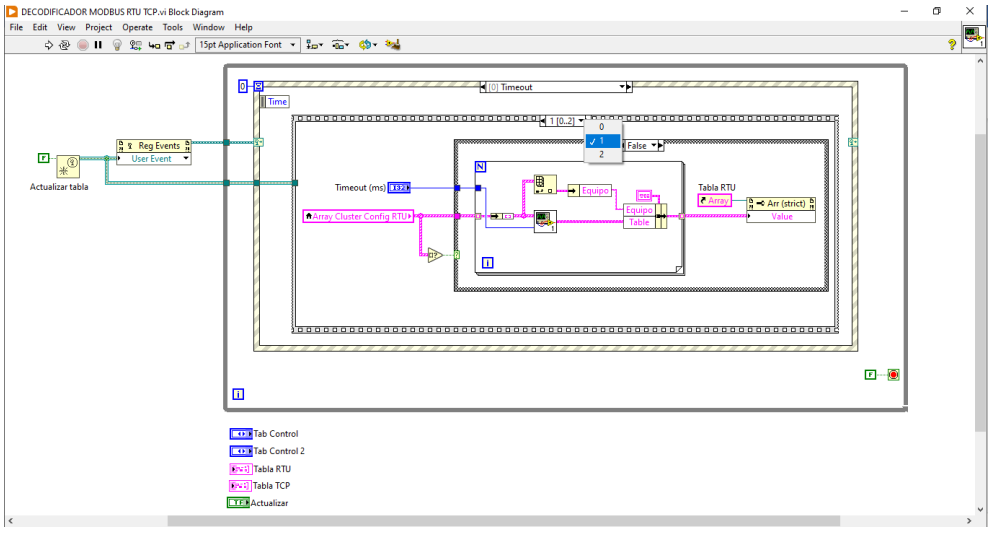


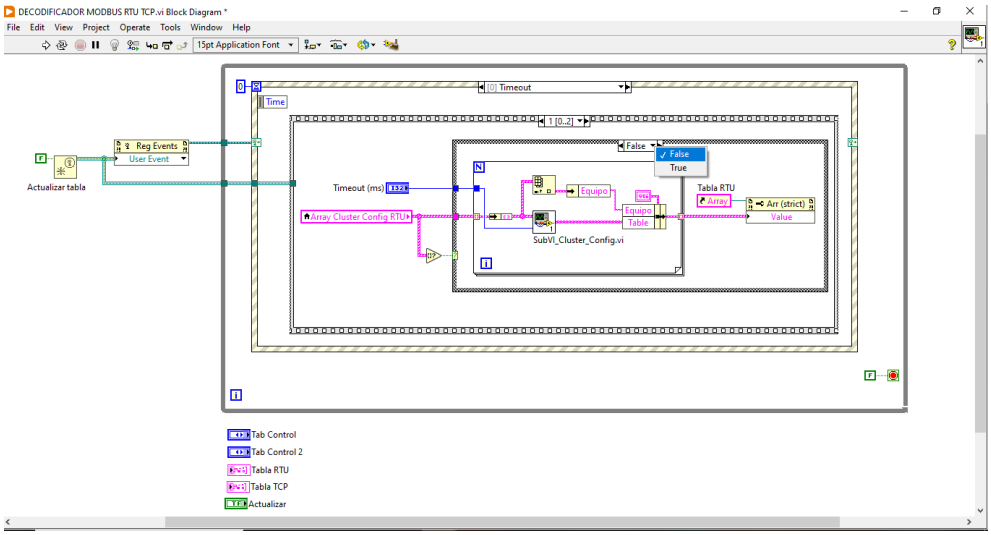


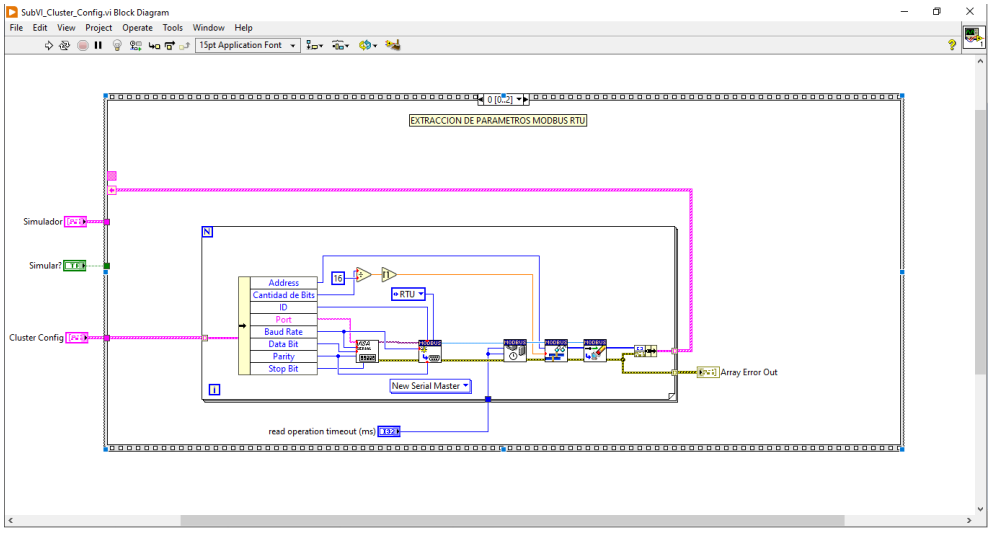


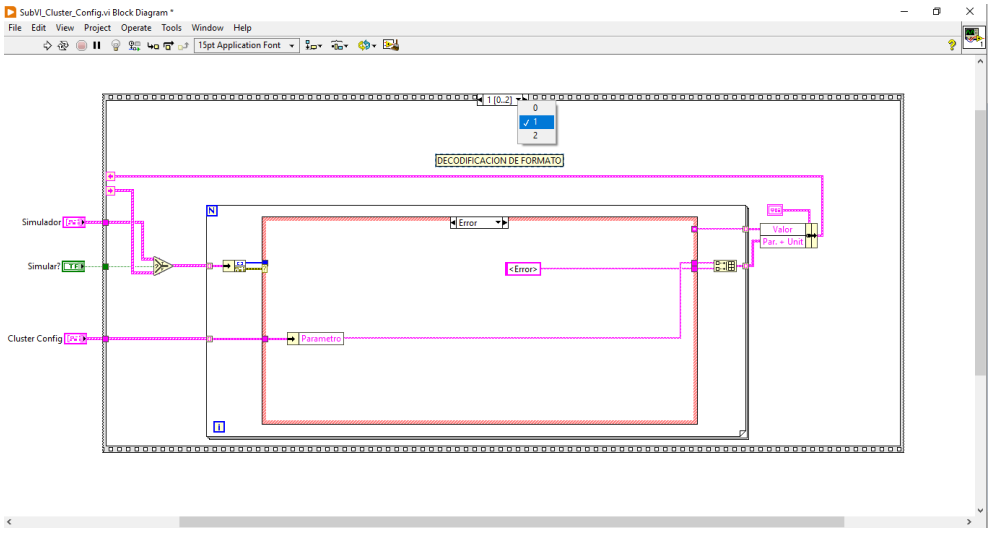

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
[0] Timeout
Time
0 [0..2]
True
False
Reg Events
User Event
Actualizar tabla
Enviar a:
ACTUALIZAR TABLA
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

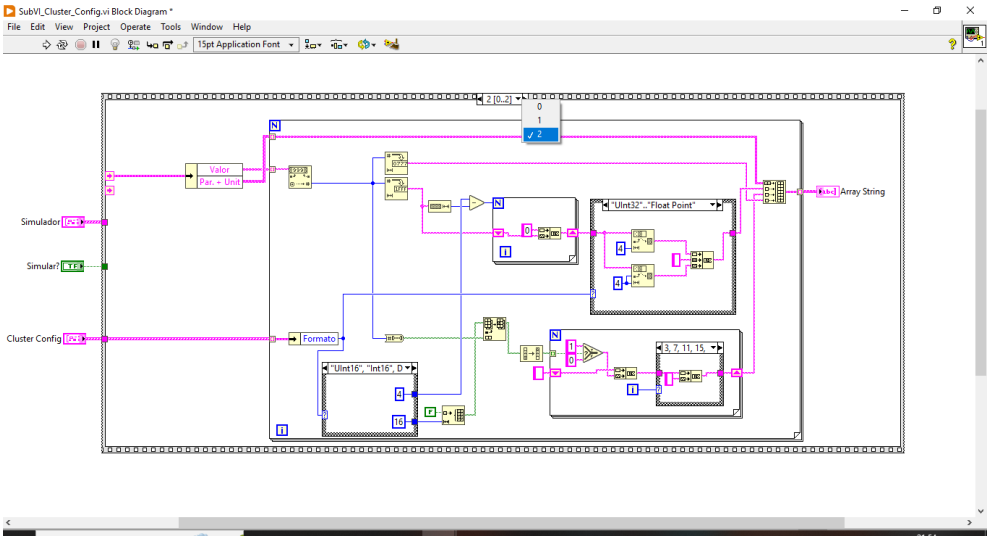

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
[0] Timeout
Time
0 [0..2]
False
Reg Events
User Event
Actualizar tabla
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

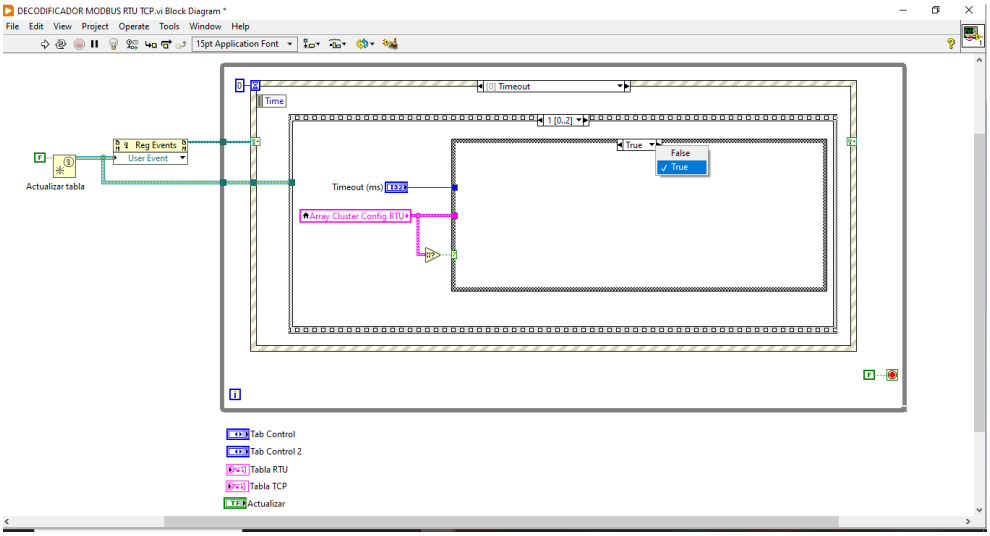

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
[0] Timeout
Time
1 [0..2]
0
1
2
False
Reg Events
User Event
Actualizar tabla
Timeout (ms)
Array Cluster Config RTU
Equipo
Tabla RTU
Array
Arr (strict)
Value
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

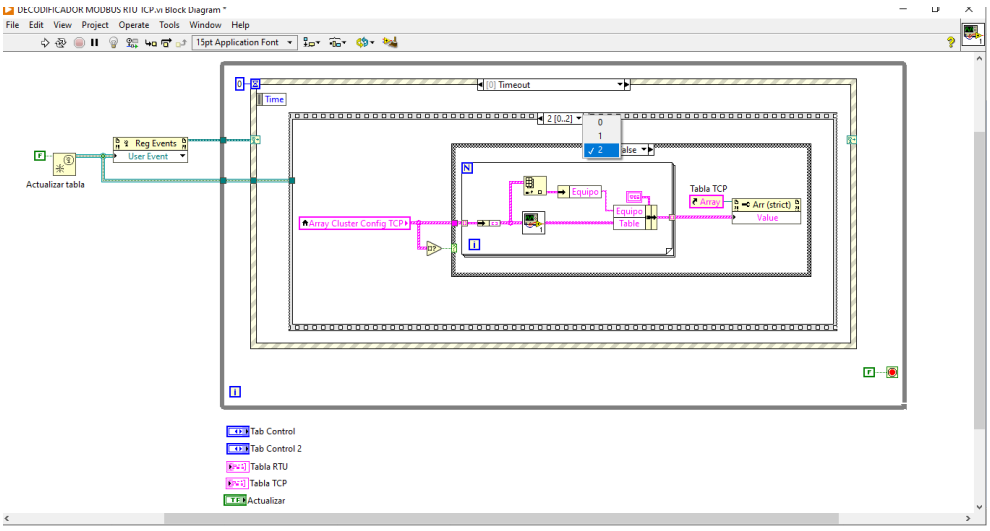

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Timeout
Time
Reg Events
User Event
Actualizar tabla
Timeout (ms)
Array Cluster Config RTU
False
True
Equipo
Tabla RTU
Array
Arr (strict)
Value
SubVI_Cluster_Config.vi
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

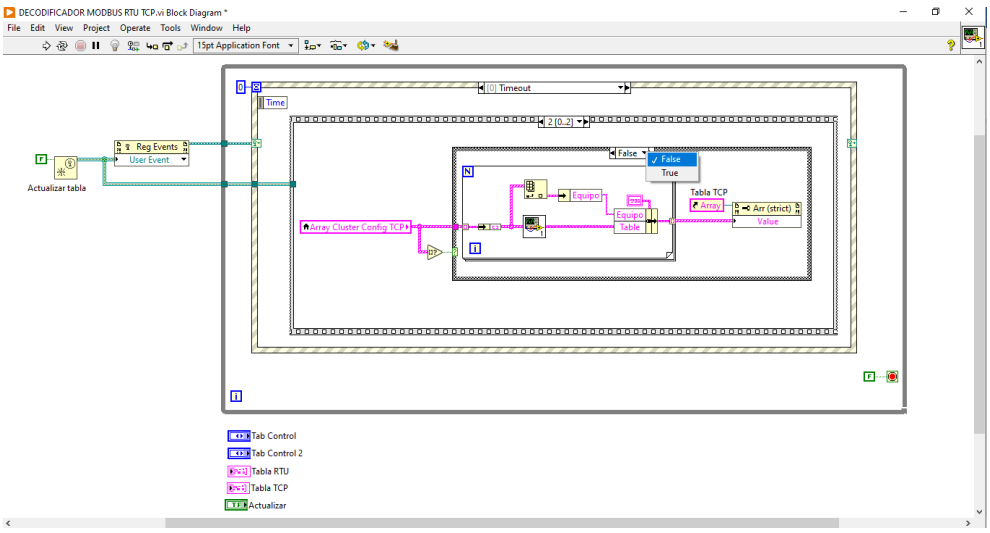

SubVI_Cluster_Config.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
EXTRACCION DE PARAMETROS MODBUS RTU
Simulador
Simular?
Cluster Config
Address
Cantidad de Bits
ID
Port
Baud Rate
Data Bit
Parity
Stop Bit
RTU
New Serial Master
Array Error Out
read operation timeout (ms)

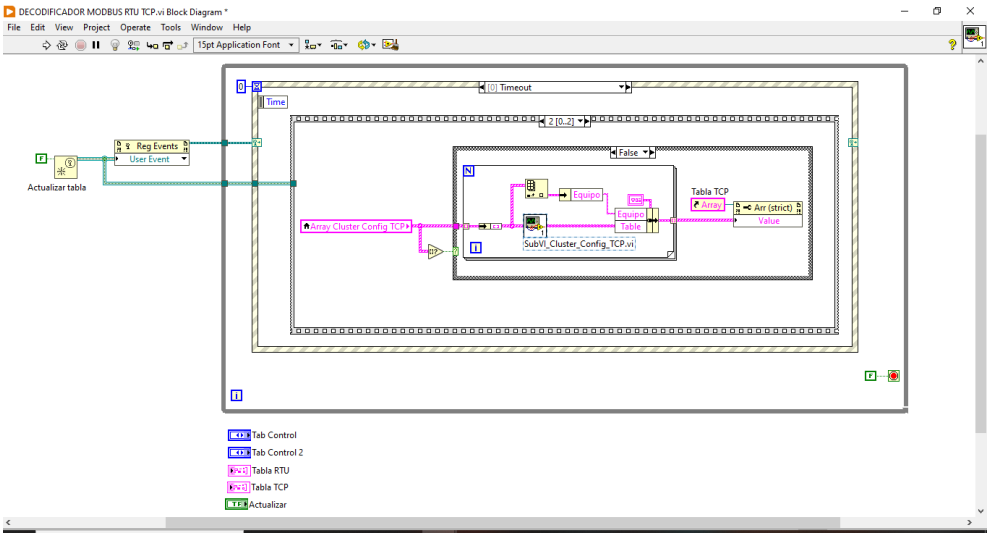

SubVI_Cluster_Config.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
DECODIFICACION DE FORMATO
Simulador
Simular?
Cluster Config
Error
<Error>
Valor
Par. + Unit
Parametro

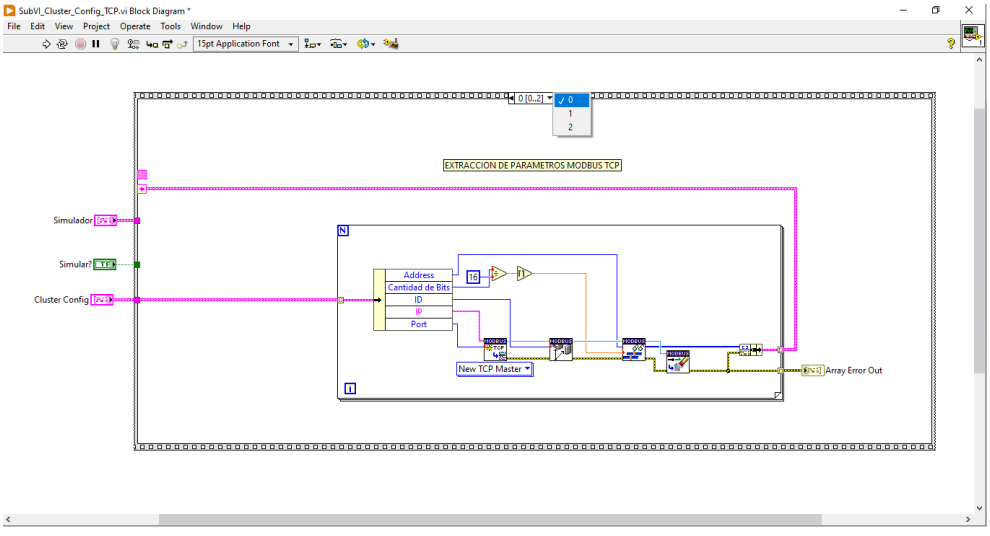

SubVI_Cluster_Config.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Valor
Par. + Unit
Simulador
Simular?
Cluster Config
Formato
"UInt16", "Int16", D
"UInt32"..."Float Point"
[3, 7, 11, 15,
Array String

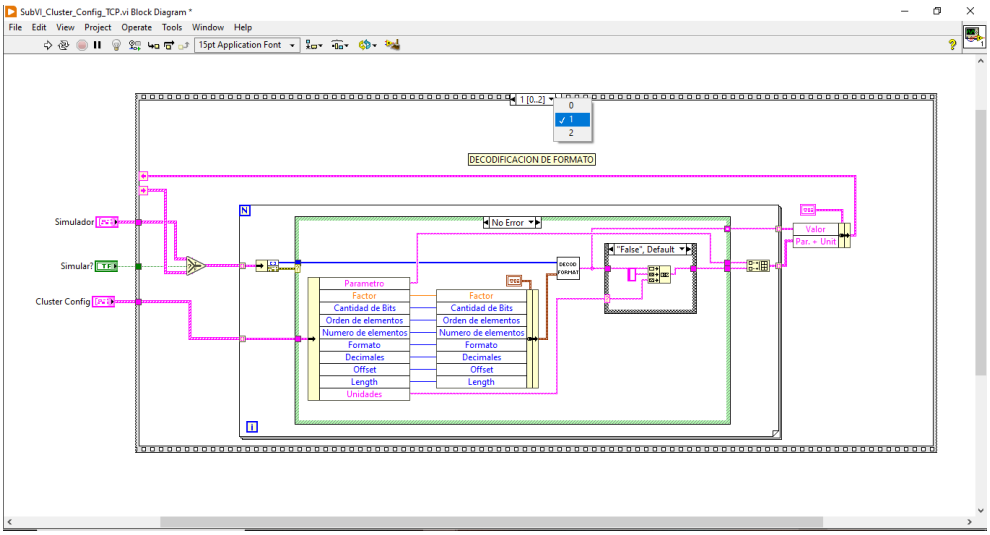

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Actualizar tabla
Reg Events
User Event
Time
Timeout
True
False
Timeout (ms)
Array Cluster Config RTU
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

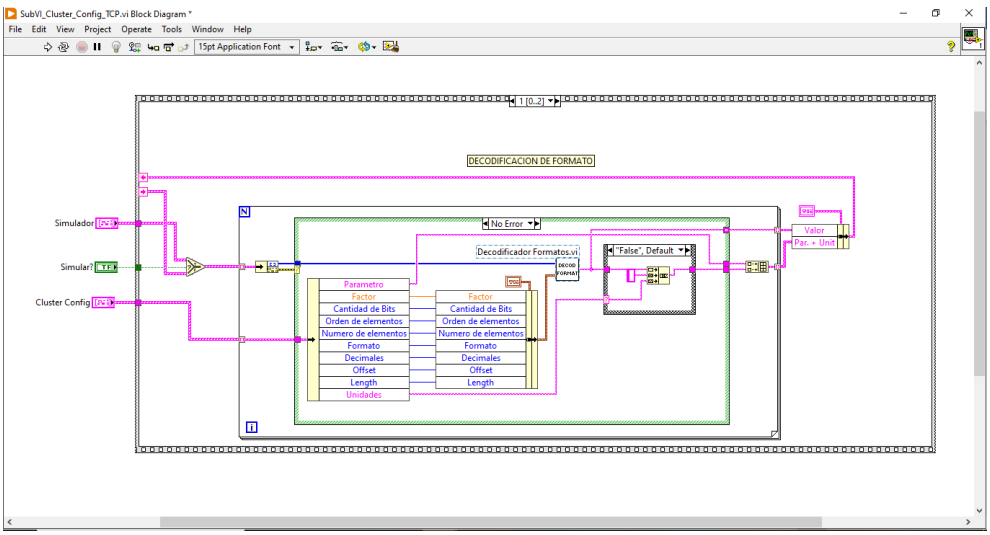

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Actualizar tabla
Reg Events
User Event
Time
Timeout
Array Cluster Config TCP
Equipo
Tabla TCP
Array
Arr (strict)
Value
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

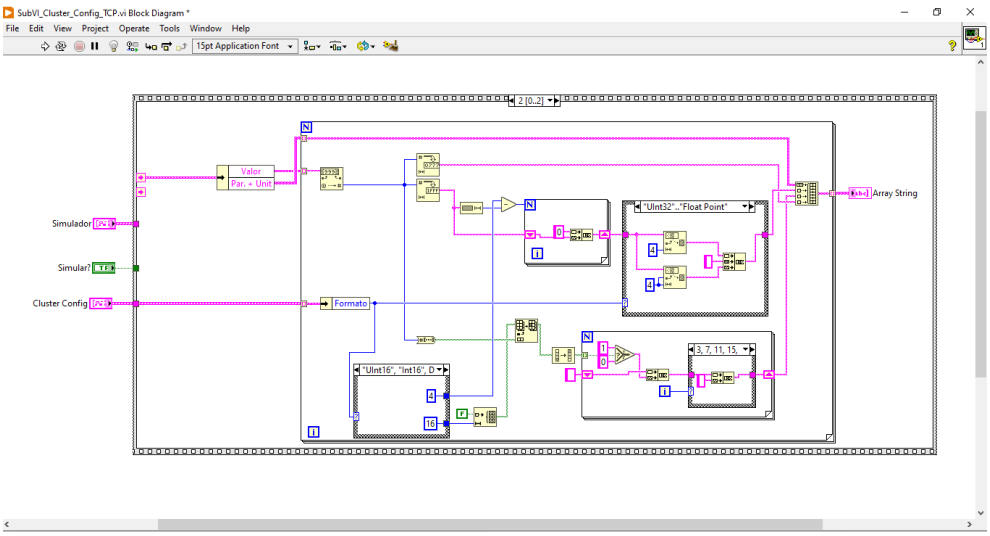

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Reg Events
User Event
Actualizar tabla
[0] Timeout
Time
2 [0..2]
False
True
Equipo
Array Cluster Config TCP
Tabla TCP
Array
Arr (strict)
Value
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

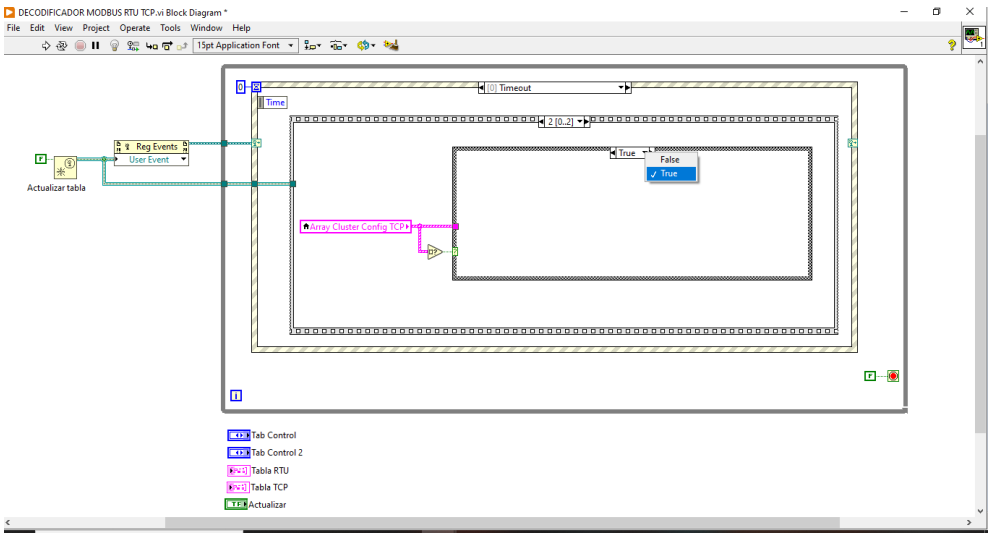

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Reg Events
User Event
Actualizar tabla
[0] Timeout
Time
2 [0..2]
False
Equipo
Array Cluster Config TCP
SubVI_Cluster_Config_TCP.vi
Tabla TCP
Array
Arr (strict)
Value
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

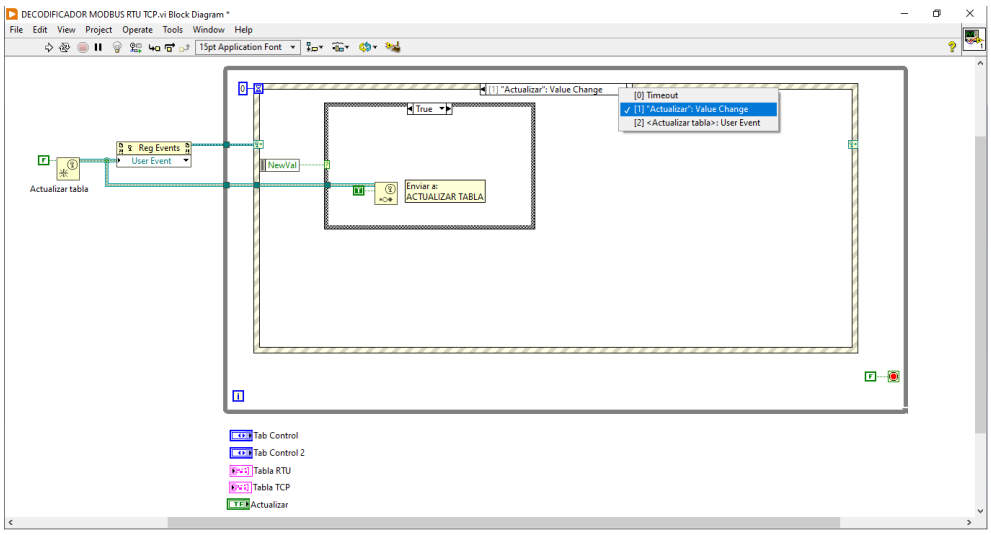

SubVI_Cluster_Config_TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
0 [0..2]
EXTRACCION DE PARAMETROS MODBUS TCP
Simulador
Simular?
Cluster Config
Address
Cantidad de Bits
ID
IP
Port
16
New TCP Master
Array Error Out

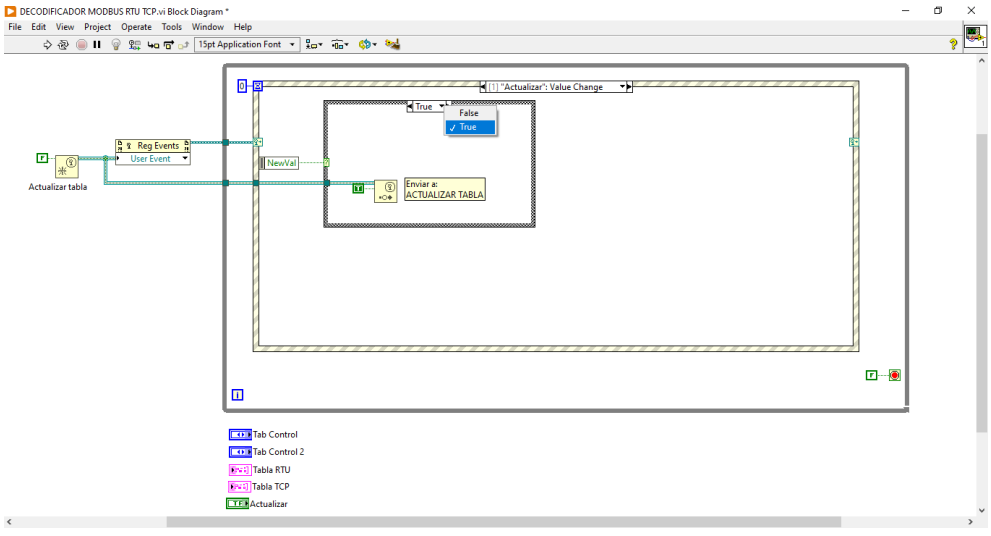

SubVI_Cluster_Config_TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
1 [0..2]
0
1
2
DECODIFICACION DE FORMATO
Simulador
Simular?
Cluster Config
No Error
"False", Default
Valor
Par. + Unit
Parametro
Factor
Cantidad de Bits
Orden de elementos
Numero de elementos
Formato
Decimales
Offset
Length
Unidades

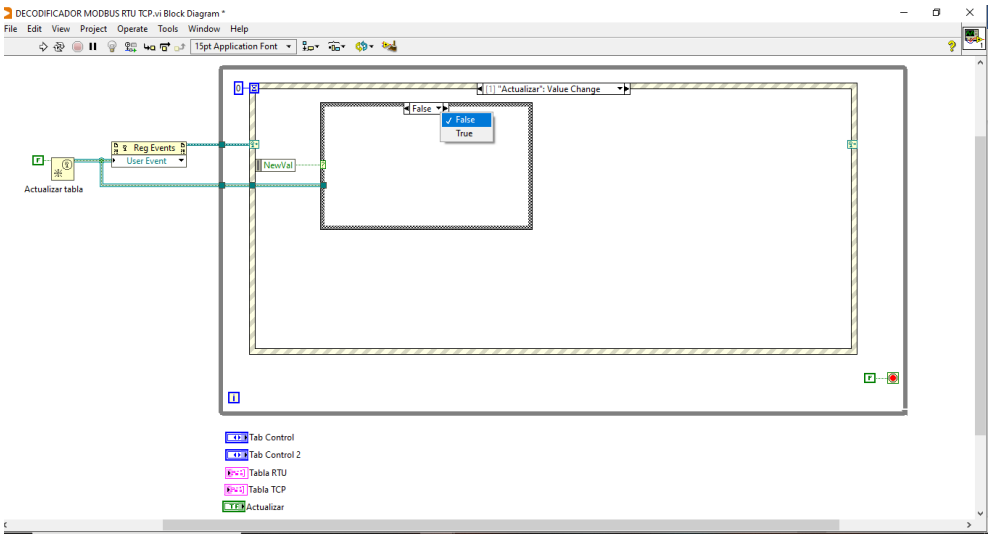

SubVI_Cluster_Config_TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
1 [0..2]
DECODIFICACION DE FORMATO
Simulador
Simular?
Cluster Config
No Error
Decodificador Formatos.vi
"False", Default
Valor
Par. + Unit
Parametro
Factor
Cantidad de Bits
Orden de elementos
Numero de elementos
Formato
Decimales
Offset
Length
Unidades

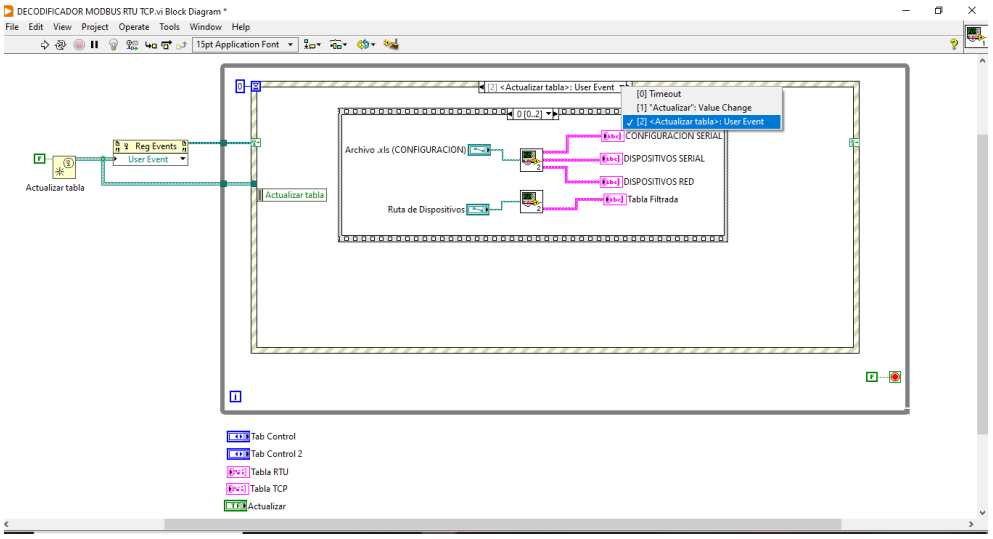

SubVI_Cluster_Config_TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
2 [0..2]
Valor
Par. + Unit
Simulador
Simular?
Cluster Config
Formato
"UInt32"..."Float Point"
"UInt16", "Int16", D
3, 7, 11, 15,
Array String

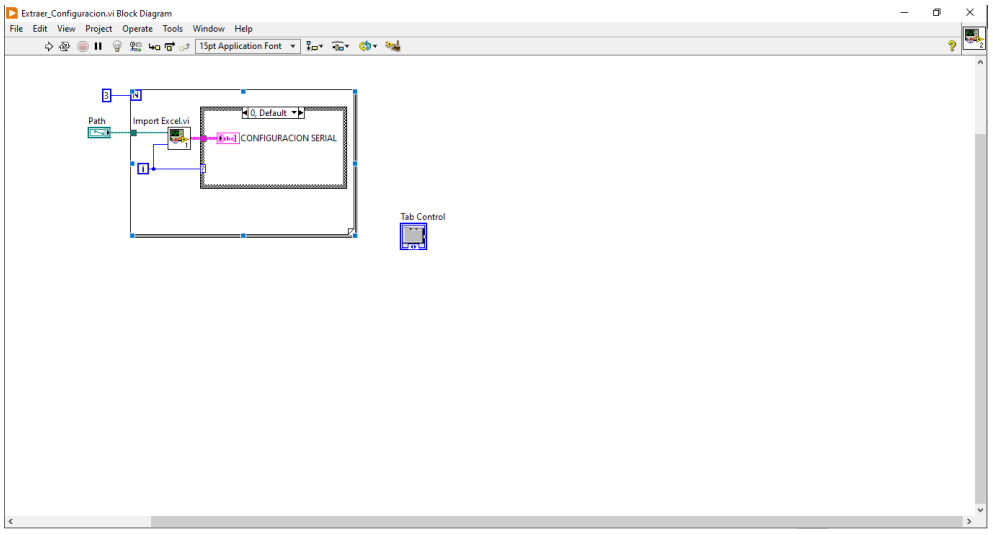

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
[0] Timeout
Time
2 [0..2]
True
False
True
Reg Events
User Event
Actualizar tabla
Array Cluster Config TCP
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

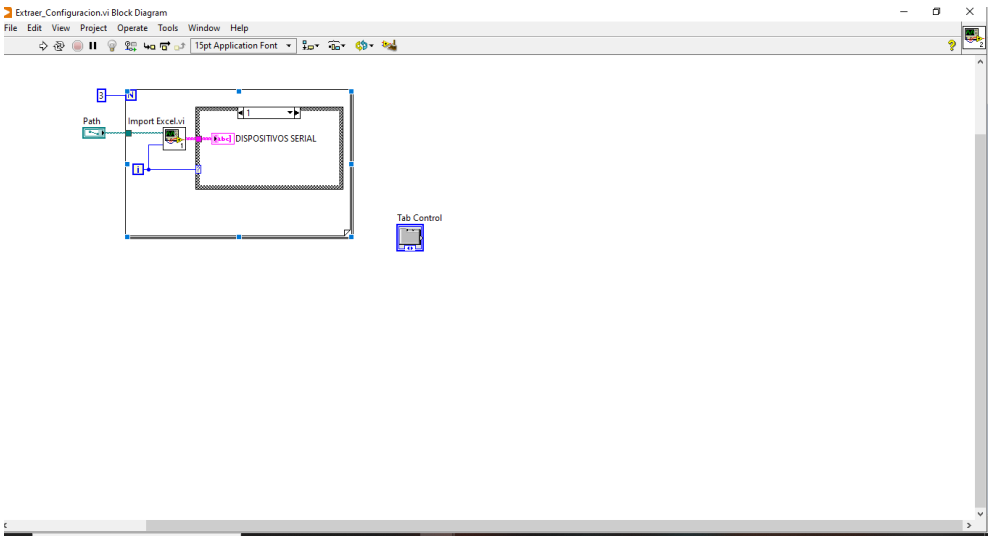

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
[1] "Actualizar": Value Change
[0] Timeout
[1] "Actualizar": Value Change
[2] <Actualizar tabla>: User Event
True
Reg Events
User Event
Actualizar tabla
NewVal
Enviar a:
ACTUALIZAR TABLA
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

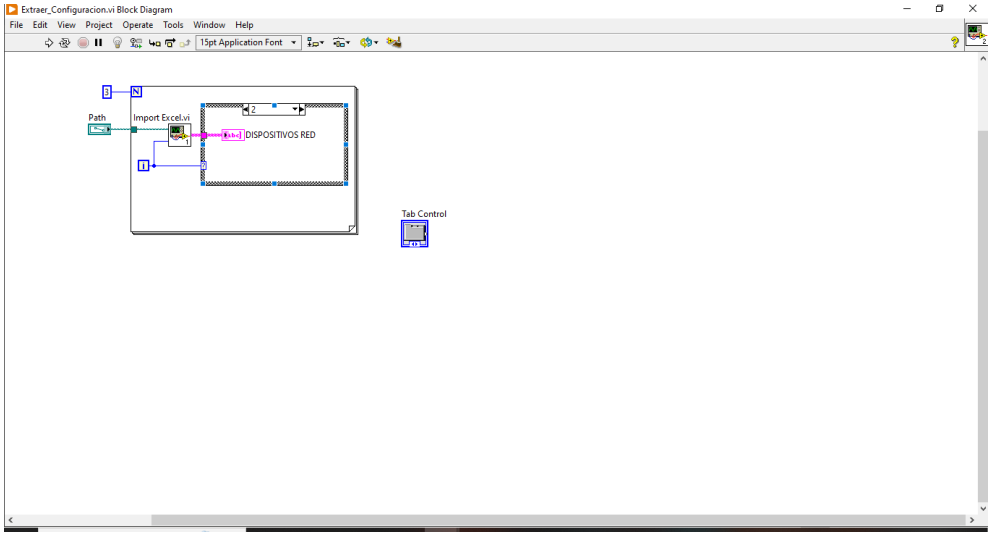

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
[1] "Actualizar": Value Change
True
False
True
Reg Events
User Event
Actualizar tabla
NewVal
Enviar a:
ACTUALIZAR TABLA
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

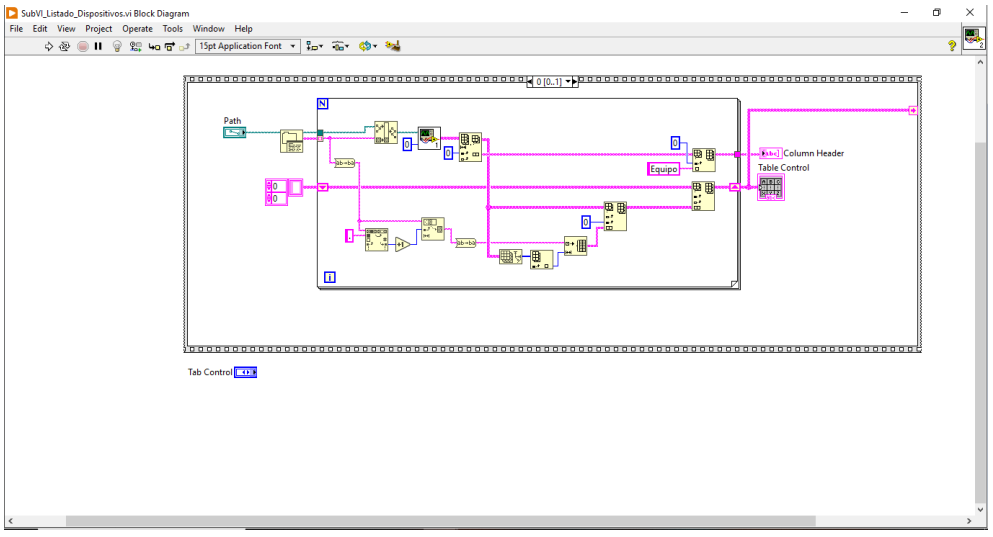

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
[1] "Actualizar": Value Change
False
True
Reg Events
User Event
Actualizar tabla
NewVal
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

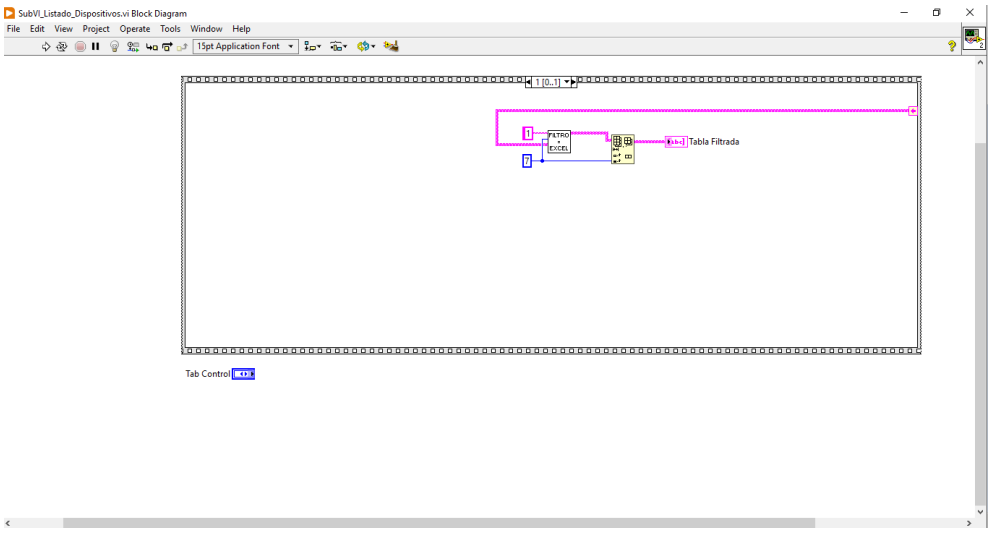

DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
[2] <Actualizar tabla>: User Event
[0] Timeout
[1] "Actualizar": Value Change
[2] <Actualizar tabla>: User Event
0 [0..2]
Reg Events
User Event
Actualizar tabla
Archivo .xls (CONFIGURACION)
Ruta de Dispositivos
CONFIGURACION SERIAL
DISPOSITIVOS SERIAL
DISPOSITIVOS RED
Tabla Filtrada
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

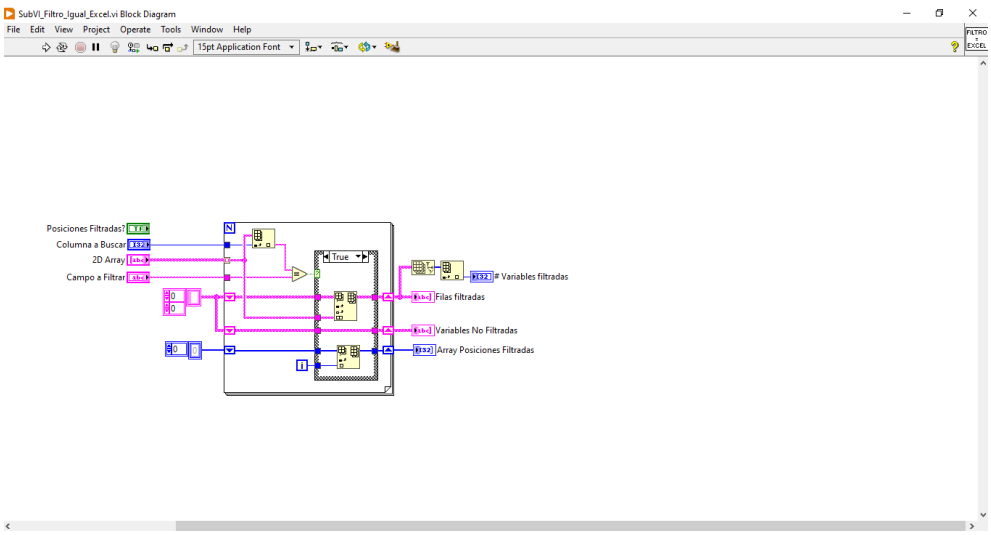

Extraer_Configuracion.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
Path
Import Excel.vi
0, Default
CONFIGURACION SERIAL
Tab Control

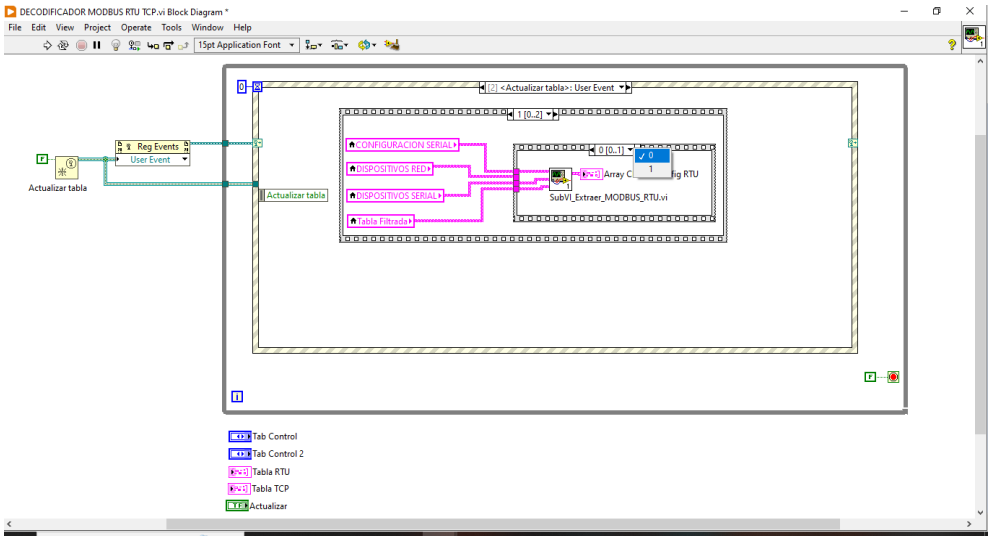

Extraer_Configuracion.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
Path
Import Excel.vi
DISPOSITIVOS SERIAL
Tab Control

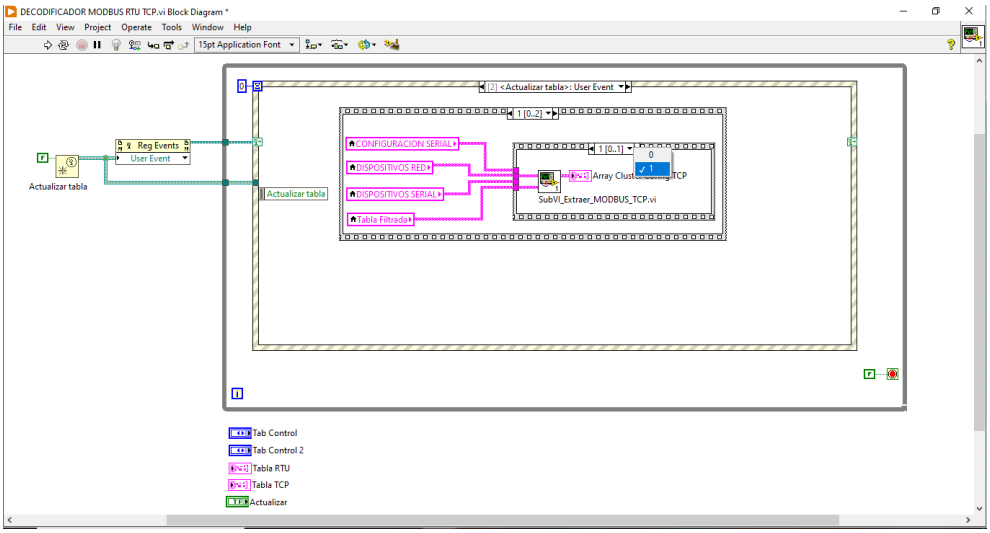

Extraer_Configuracion.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
Path
Import Excel.vi
DISPOSITIVOS RED
Tab Control

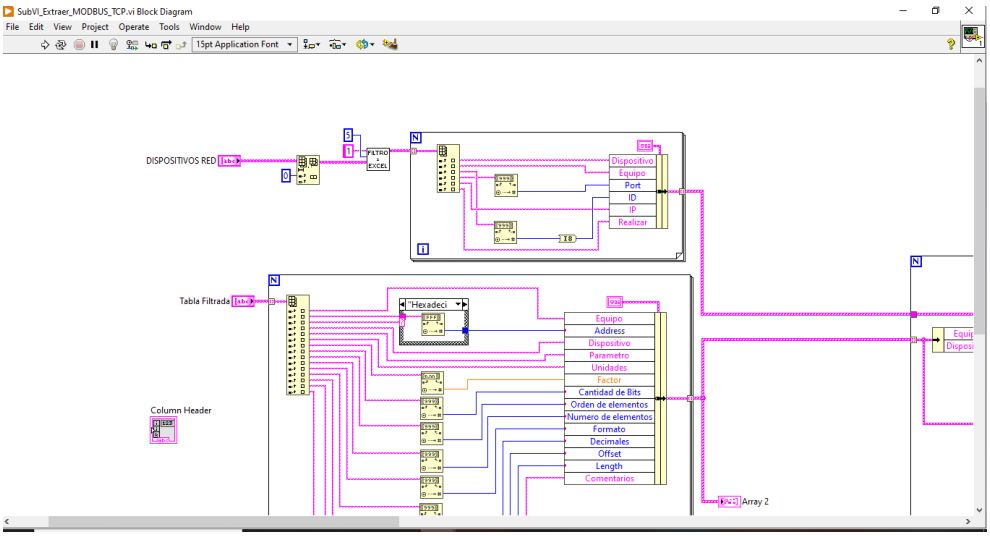

SubVI_Listado_Dispositivos.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
Path
Equipo
Column Header
Table Control
Tab Control

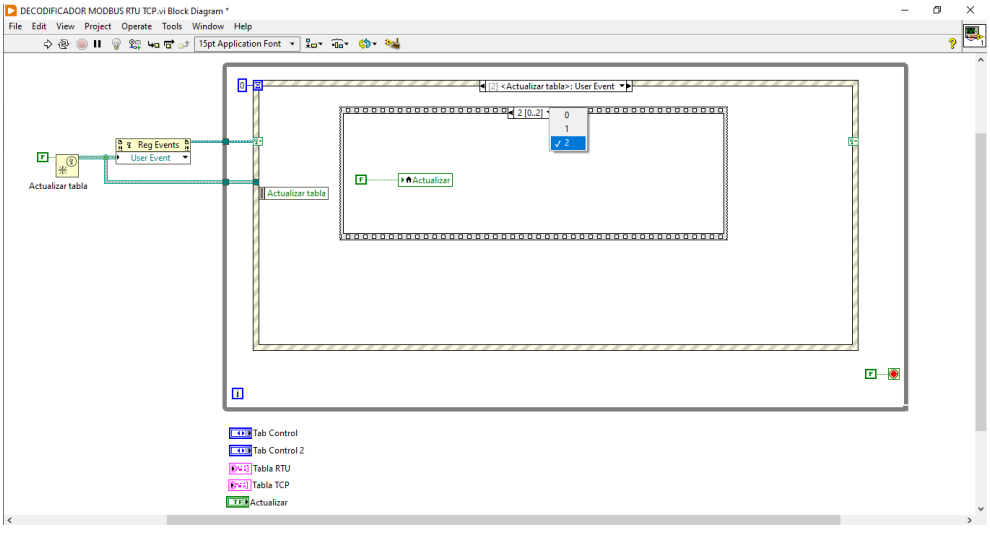

SubVI_Listado_Dispositivos.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
1 [0..1]
FILTRO EXCEL
Tabla Filtrada
Tab Control


SubVI_Filtro_Igual_Excel.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
Posiciones Filtradas?
Columna a Buscar
2D Array
Campo a Filtrar
True
Variables filtradas
Filas filtradas
Variables No Filtradas
Array Posiciones Filtradas


DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Actualizar tabla
Reg Events
User Event
<Actualizar tabla>: User Event
CONFIGURACION SERIAL
DISPOSITIVOS RED
DISPOSITIVOS SERIAL
Tabla Filtrada
SubVI_Extraer_MODBUS_RTU.vi
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar


DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Reg Events
User Event
Actualizar tabla
<Actualizar tabla>: User Event
CONFIGURACION SERIAL
DISPOSITIVOS RED
DISPOSITIVOS SERIAL
Tabla Filtrada
SubVI_Extraer_MODBUS_TCP.vi
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar


SubVI_Extraer_MODBUS_TCP.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
DISPOSITIVOS RED
FILTRO EXCEL
Dispositivo
Equipo
Port
ID
IP
Realizar
Tabla Filtrada
"Hexadeci
Equipo
Address
Dispositivo
Parametro
Unidades
Factor
Cantidad de Bits
Orden de elementos
Numero de elementos
Formato
Decimales
Offset
Length
Comentarios
Column Header
Array 2


DECODIFICADOR MODBUS RTU TCP.vi Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Reg Events
User Event
Actualizar tabla
<Actualizar tabla>: User Event
Actualizar
Tab Control
Tab Control 2
Tabla RTU
Tabla TCP
Actualizar

13.2. Propuesta de Laboratorio 1 Digital Modbus RTU

Universidad Nacional de Ingeniería

Facultad de Electrotecnia y Computación

Departamento de Electrónica

Laboratorio: 1
Materia: Electrónica digital

I. Tema práctico.

- *Uso de aplicación VI LABVIEW sobre protocolo Modbus para la recepción de variables de dispositivos por conexión serial.*

II. Objetivos

- Estudiar el principio de funcionamiento del protocolo Modbus.
- Comprender los conceptos de Modbus RTU.
- Comprender e interpretar tablas de registros Modbus para completar practica de laboratorio utilizando el dispositivo ModBus (Motor LogicScuareD II).

III. Medios a utilizar

- Computadora.
- Driver LVRumTimeEng de national Instruments.
- Aplicación vi “DECODIFICADOR MODBUS RTU TCP”.
- Documento .exe con las tablas de variables de dispositivos.
- Driver 101-0019 & 101-0020 RS-485 Converter - USB Driver Version 5.1 Driver Package (32 & 64 bit) for Windows 11, 10, 8, 7, Vista, XP, and 2000

IV. Componentes/Dispositivos a utilizar

Cantidad	Componente
1	Motor Logic Plus II
1	Cable serie USB a RS485

V. Medidas de seguridad

- En esta práctica de laborátorio, no energice ningún dispositivo sin la autorización.
- Al culminar la practica, desconecte el cable de alimentación antes de desconectar los dispositivos. (Hay que recordar que existen 120 o 220 Ac a la entrada).

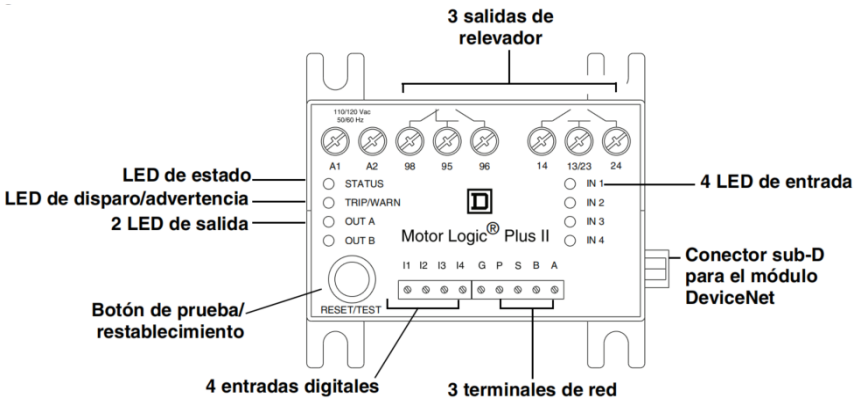
VI. Introducción

El sistema de comunicación Modbus es el protocolo de comunicación más extendido en la automatización industrial, El protocolo Modbus está basado en una arquitectura maestro/esclavo utilizado para transmitir información a través de redes en serie o ethernet entre dispositivos electrónicos de automatización en su mayoría.

En este laboratorio se presenta una aplicación o programa VI (instrumentos virtuales) creada en LabVIEW para la extracción de variables del equipo Motor Logic Plus II por comunicación Modbus,

VII. Desarrollo de la Práctica

Siendo el equipo Motor Logic® Plus II nuestro primer dispositivo para esta práctica.



1. Verificar correcta conexión de comunicación RTU en terminales de red bornes A y B del dispositivo con el convertidor USB a RS485 Microflex.
2. Instalar driver correspondiente Driver Microflex: 101-0019 y verificar en nuestro ordenador que puerto COM se habilita al conectar el convertidor “Administrador de dispositivos”. Ejemplo: COM1
3. Configurar el puerto habilitado COM1 haciendo clic derecho y seleccionando “propiedades” con los siguientes parámetros del dispositivo: Baud Rate: 19200, Data Bits: 8, Parity: None, Stop Bits: 1.
4. Verificar la correcta conexión de alimentación del dispositivo en bornes A1 y A2 y habilitar el autómata o minibreaker de control A1 para posterior conectar toma 120 Vac del gabinete.

5. Completar la instalación del Driver LVRumTimeEng de national Instruments.

6. Crear una carpeta en el escritorio nombrarla “Practica 1” y editar dos documentos de Excel compatibles para (Libro de Excel 97-2003 (*xls)) Primero titulado “Configuración puertos Modbus”, crear una subcarpeta titulada Dispositivos y editar el segundo documento de Excel titularlo “SQRTU-1”.

7. Llenar documento de Excel “Configuración puertos Modbus” el cual contendrá 3 hojas de cálculo con el siguiente formato:

   - 1er hoja de cálculo titularla: CONFIGURACION SERIAL. Sera llenada con los siguientes parámetros:

Puerto	Baud Rate	Data Bits	Parity	Stop Bits
1	19200	8	None	1

   - 2da hoja de cálculo titularla: DISPOSITIVOS SERIA. Sera llenada con los siguientes parámetros:

Dispositivo	Equipo	Esclavo	Puerto	¿Realizar?
SQUARE-D Motor Logic Plus II	SQRTU-1	3	1	1

   - 3er hoja de cálculo titularla: DISPOSITIVOS RED. Dejarla vacía esta será utilizada en práctica de laboratorio 2.

Dispositivo	Equipo	ID	Puerto	IP	¿Realizar?

Una vez que ya tengamos la hoja de datos o manual del dispositivo Motor Logic Plus II y procedemos a llenar nuestra hoja de cálculo donde especificaremos los parámetros o características operativas de nuestro dispositivo para la práctica. Cada dispositivo con protocolo de comunicación Modbus posee una tabla de registro con cada parámetro de este, es de fácil acceso ya sea digital en la página oficial del dispositivo o en el físico acompañado con el manual del dispositivo existen casos especiales de equipos que te permitirán crear tu propia tabla de registros según las necesidades.

8. Llenar el documento de Excel: SQRTU-1”. Siguiendo el siguiente formato:
   16-Address.
   17-Codificación.
   18-Dispositivo.
   19-Parámetro.
   20-Unidades.
   21-Factor.
   22-Extraer.
   23-Cantidad de Bits.
   24-Orden de elementos.
   25- Número de elementos.
   26- Representación Negativo.
   27- Decimales.
   28- Offset.
   29- Length.
   30-Comentarios.

Address	Codificación	Dispositivo	Parámetro	Unidades

Factor	¿Extraer?	Cantidad de Bits	Orden de elementos	Número de elementos

Representación Negativos	Decimales	Offset	Length	Comentarios

9. Para esta práctica solo extraeremos 6 parámetros: Verificarlos en manual o hoja de datos del analizador SQUARE-D Motor Logic Plus II
   - Registro 40411, Address 19A, Bit 12: Test Trip.
   - Registro 40414, Address 19D, Bit 0: Output A is On or Closed
   - Registro 40414, Address 19D, Bit 1: Output B is On or Closed.
   - Registro 40430, Address 1AD, Corriente Fase C
   - Registro 40431, Address 1AE, Corriente Fase B
   - Registro 40432, Address 1AF, Corriente Fase A.

Iniciar a llenar hoja de Excel según manual de dispositivo ejemplo 1:

- Address: 19A
- Codificación: Hexadecimal.
- Dispositivo: SQUARE-D Motor Logic Plus II
- Parámetro: Test Trip
- Unidades: Ninguno.

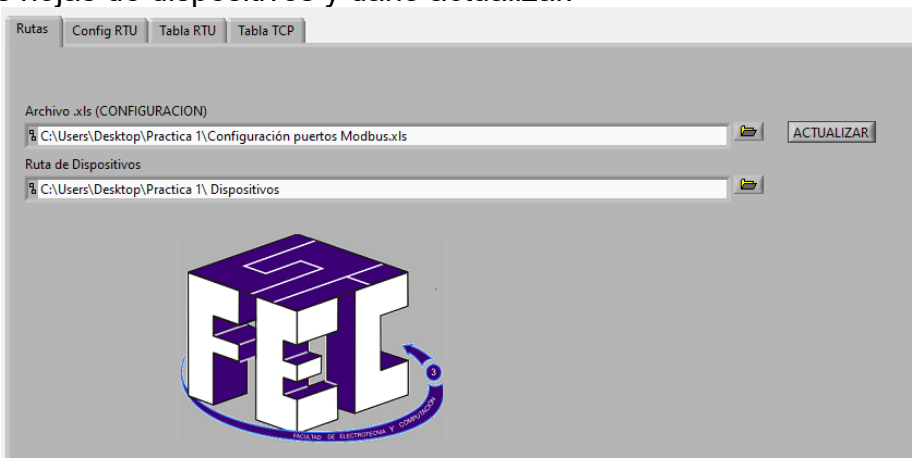
- Factor: 1 (Multiplicador de la variable)
- Extraer: 1 es que si y 0 es que no. Digitar 1
- Cantidad de Bits: 16 bits (Numero de bits del mensaje)
- Orden de elementos: MSB o LSB, digitar MSB.
- Número de elementos: 1 o 2 registros, digitar: 1
- Representación Negativo: Punto Flotante SINT32; Complemento A1; Complemento A2; Ninguno, digitar: Ninguno.
- Decimales: digital 0 para este ejemplo.
- Offset: número de bits de la trama requerido: 12
- Length: Longitud de bits después del Offset: 1
- Comentarios: registro de lectura práctica 1.

10. Completar los restantes 5 registros según manual de dispositivo y ejemplo 1.

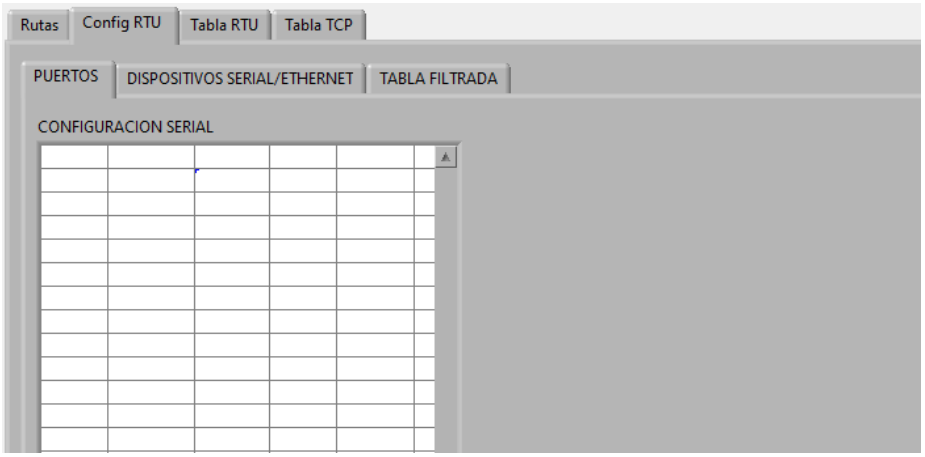
En este punto ya somos capaces de extraer variables y datos del dispositivo mediante su puerto de comunicación.

11. Ejecutar programa DECODIFICADOR MODBUS RTU TCP.exe.

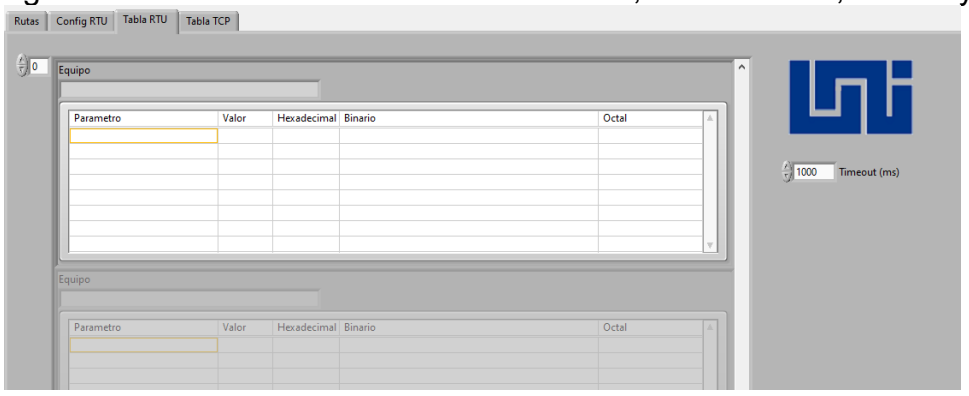
12. Primera ventana del ejecutable será donde seleccionaremos la ruta de los archivos de Excel creados, Configuración puertos Modbus y ruta de carpeta de las hojas de dispositivos y darle actualizar.



13. En la pestaña del ejecutable Config RTU se extraerán los archivos de Excel editados y rellenarán automáticamente en tablas del ejecutable verificar cada una de las tablas, PUERTOS, DISPOSITIVOS SERIAL/ETHERNET y TABLA FILTRADA.



14. Finalmente, la pestaña Tabla RTU nos entregara los parámetros extraídos y decodificados en línea del equipo, presentándonos en diferentes tramas digitales el mismo dato cuando no es tratado, Hexadecimal, binario y Octal.



VII. Trabajo Previo

El trabajo previo consiste en:

1. Investigar sobre el Protocolo de comunicación Modbus
2. Traer la hoja de datos del dispositivo Motor Logic plus II
3. Responder las preguntas de control

VIII. Preguntas de Control

1. ¿Qué entiende por protocolo modbus?
2. Que es el termino RTU y TCP.
3. ¿Que son las VIs?

VII. Conclusiones

- Compare los resultados de los valores tomados en las tablas y cuál sería la relación y diferencia entre los diferentes tipos de presentación de variables.

13.3. Propuesta de Laboratorio 2 Digital Modbus TCP.

Universidad Nacional de Ingeniería

Facultad de Electrotecnia y Computación

Departamento de Electrónica

Laboratorio:	2
Materia:	Electrónica digital

I. Tema práctico.

- *Uso de aplicación VI LABVIEW sobre protocolo Modbus para la recepción de variables de dispositivos por conexión TCP IP.*

II. Objetivos

- Comprender los conceptos de Modbus TCP IP.
- Comprender e interpretar tablas de registros Modbus para completar practica de laboratorio utilizando el dispositivo Modbus (schneider PM5320).

III. Medios a utilizar

- Computadora.
- Driver LVRumTimeEng de national Instruments.
- Aplicación vi “DECODIFICADOR MODBUS RTU TCP”.
- Documento .exe con las tablas de variables de dispositivos.

IV. Componentes/Dispositivos a utilizar

Cantidad	Componente
1	schneider PM5320
1	Cable ethernet RJ45

V. Medidas de seguridad

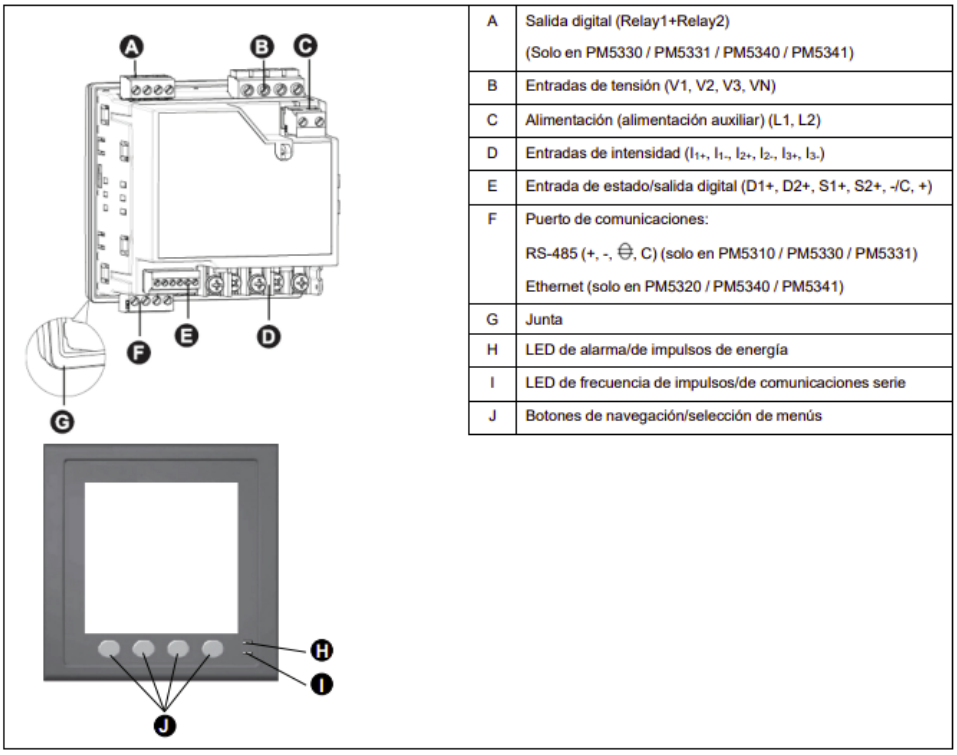
- En esta práctica de laborátorio, no energice ningún dispositivo sin la autorización.
- Al culminar la practica, desconecte el cable de alimentación antes de desconectar los dispositivos. (Hay que recordar que existen120 o 220 VAc a la entrada).

VI. Introducción

En este laboratorio se presenta una aplicación o programa VI (instrumentos virtuales) creada en LabVIEW para la extracción de variables del equipo PM5320 por comunicación Modbus TCP.

VII. Desarrollo de la Práctica

Siendo el equipo analizador de energías Schneider PM5320 nuestro dispositivo para esta práctica.



A	Salida digital (Relay1+Relay2) (Solo en PM5330 / PM5331 / PM5340 / PM5341)
B	Entradas de tensión (V1, V2, V3, VN)
C	Alimentación (alimentación auxiliar) (L1, L2)
D	Entradas de intensidad (I_{1+}, I_{1-}, I_{2+}, I_{2-}, I_{3+}, I_{3-})
E	Entrada de estado/salida digital (D1+, D2+, S1+, S2+, -/C, +)
F	Puerto de comunicaciones: RS-485 (+, -, ⏚, C) (solo en PM5310 / PM5330 / PM5331) Ethernet (solo en PM5320 / PM5340 / PM5341)
G	Junta
H	LED de alarma/de impulsos de energía
I	LED de frecuencia de impulsos/de comunicaciones serie
J	Botones de navegación/selección de menús

1. Verificar correcta conexión de comunicación del cable TCP del dispositivo.

2. Verificar la dirección IP de nuestro ordenador o tarjeta de red esté en rango de IPs con el dispositivo que vaya a realizar las peticiones. (Panel de control\Redes e Internet\Conexiones de red→ clic derecho en puerto de red → Propiedades→ En pestaña funciones de red seleccionar →Habilitar el protocolo de internet versión 4 (TCP\IPv4) en este punto digitar una dirección IP estática en mismo segmento que PM5320 que posee una IP estática: 169.254.189.34.

3. Verificar la correcta conexión de alimentación del dispositivo en bornes C L1 y L2 y habilitar el autómata o minibreaker de control A2 para posterior conectar toma 120 Vac del gabinete.
4. Completar la instalación del Driver LVRumTimeEng de national Instruments.

5. Crear una carpeta en el escritorio nombrarla “Practica 2” y editar dos documentos de Excel compatibles para (Libro de Excel 97-2003 (*xls)) Primero titulado “Configuración puertos Modbus”, crear un sud carpeta titulada Dispositivos y editar el segundo documento de Excel titularlo “PMTCP-1”.
6. Llenar documento de Excel “Configuración puertos Modbus” el cual contendrá 3 hojas de cálculo con el siguiente formato:

- 1er hoja de cálculo titularla: CONFIGURACION SERIAL. Previamente llenada en práctica lab.1

Puerto	Baud Rate	Data Bits	Parity	Stop Bits
1	19200	8	None	1

- 2da hoja de cálculo titularla: DISPOSITIVOS SERIA. Previamente llenada en práctica lab.1

Dispositivo	Equipo	Esclavo	Puerto	¿Realizar?
SQUARE-D Motor Logic Plus II	SQRTU-1	3	1	0

- 3er hoja de cálculo titularla: DISPOSITIVOS RED. Sera llenada con los siguientes parámetros:

Dispositivo	Equipo	ID	Puerto	IP	¿Realizar?
PM5320	PMTCP-1	255	502	169.254.189.34	1

Una vez que ya tengamos la hoja de datos o manual del dispositivo PM5320 procedemos a llenar nuestra hoja de cálculo donde especificaremos los parámetros o características operativas de nuestro dispositivo para la práctica. Cada dispositivo con protocolo de comunicación Modbus posee una tabla de registro con cada parámetro de este, es de fácil acceso ya sea digital en la página oficial del dispositivo o en el físico acompañado con el manual del dispositivo existen casos especiales de equipos que te permitirán crear tu propia tabla de registros según las necesidades.

7. Llenar el documento de Excel: “PMTCP-1”. Siguiendo el siguiente formato:
   31-Address.
   32-Codificación.
   33-Dispositivo.
   34-Parámetro.
   35-Unidades.
   36-Factor.
   37-Extraer.
   38-Cantidad de Bits.
   39-Orden de elementos.
   40- Número de elementos.
   41- Representación Negativo.
   42- Decimales.
   43- Offset.
   44- Length.
   45-Comentarios.

Address	Codificación	Dispositivo	Parámetro	Unidades

Factor	¿Extraer?	Cantidad de Bits	Orden de elementos	Número de elementos

Representación Negativos	Decimales	Offset	Length	Comentarios

8. Para esta práctica solo extraeremos 6 parámetros: Verificarlos en manual o hoja de datos del analizador PM5320.
   - Registro 403000, Address BB7, Current A
   - Registro 403002, Address BB9, Current B
   - Registro 403004, Address BBB, Current A
   - Registro 403028, Address BD3, Voltage A-N
   - Registro 403030, Address BD5, Voltage B-N
   - Registro 403032, Address BD7, Voltage C-N

Iniciar a llenar hoja de Excel según manual de dispositivo ejemplo 1:

- Address: **BD3**
- Codificación: **Hexadecimal.**
- Dispositivo: **PM5320.**

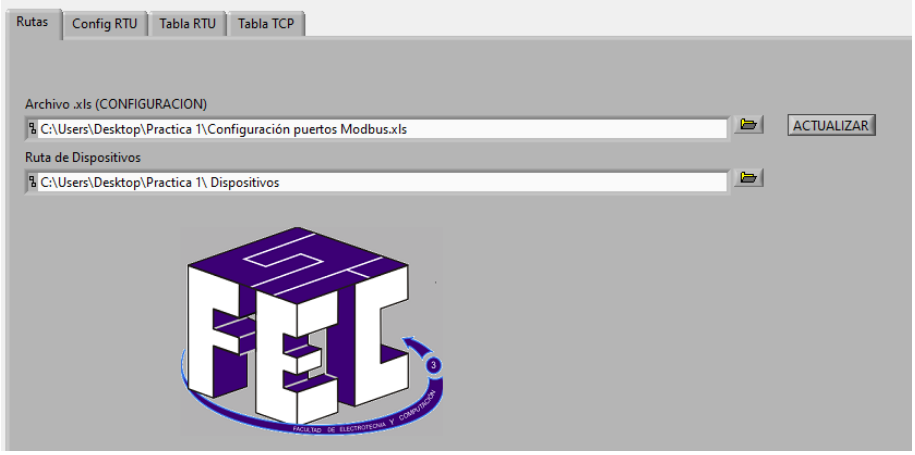
- Parámetro: **Voltaje A-N**
- Unidades: **Vac**
- Factor: **0.5** (Multiplicador de la variable)
- Extraer: 1 es que si y 0 es que no. Digitar **1**
- Cantidad de Bits: **32** bits (Numero de bits del mensaje)
- Orden de elementos: MSB o LSB, digitar: **MSB.**
- Número de elementos: 1 o 2 registros, digitar**: 1**
- Representación Negativo: Punto Flotante SINT32; Complemento A1; Complemento A2; Ninguno, digitar: **Punto Flotante SINT32**
- Decimales: digital **2** para este ejemplo.
- Offset: número de bits de la trama requerido: **0**
- Length: Longitud de bits después del Offset: **8**
- Comentarios: **Registro de lectura práctica 2.**

9. Completar los restantes 5 registros según manual de dispositivo y ejemplo 1.

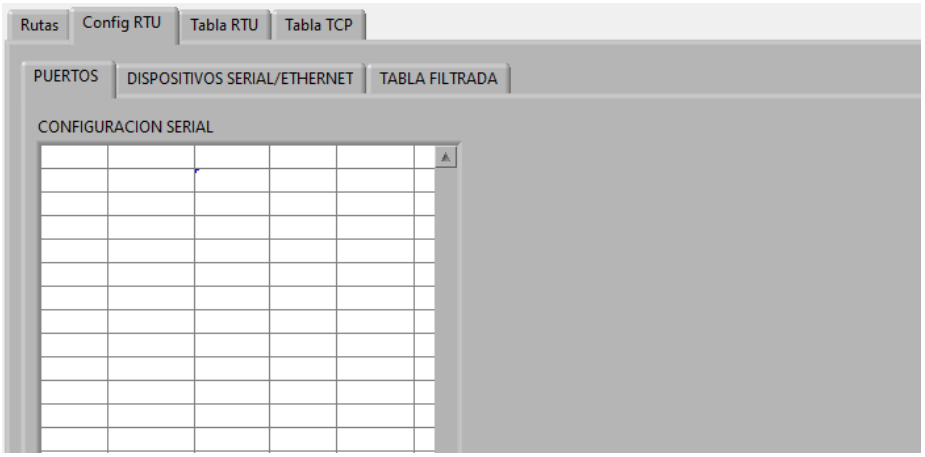
En este punto ya somos capaces de extraer variables y datos del dispositivo mediante su puerto de comunicación.

10. Ejecutar programa DECODIFICADOR MODBUS RTU TCP.exe.

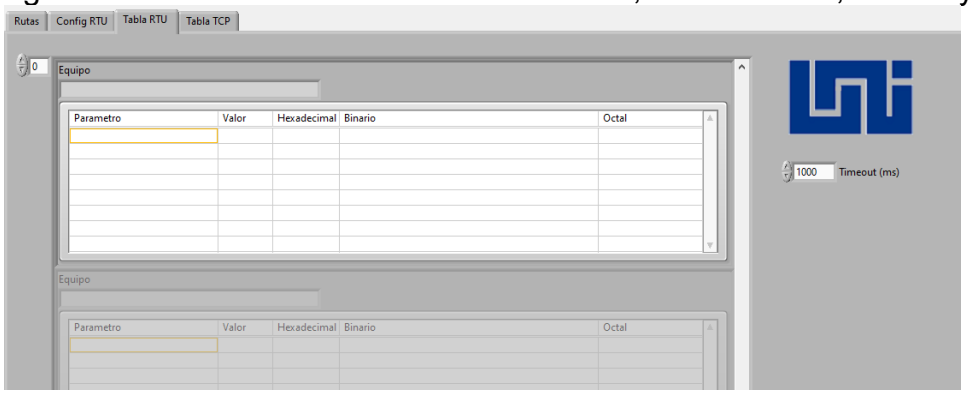
11. Primera ventana del ejecutable será donde seleccionaremos la ruta de los archivos de Excel creados, Configuración puertos Modbus y ruta de carpeta de las hojas de dispositivos y darle actualizar.



12. En la pestaña del ejecutable Config RTU se extraerán los archivos de Excel editados y rellenarán automáticamente en tablas del ejecutable verificar cada una de las tablas, PUERTOS, DISPOSITIVOS SERIAL/ETHERNET y TABLA FILTRADA.



13. Finalmente, la pestaña Tabla TCP nos entregara los parámetros extraídos y decodificados en línea del equipo, presentándonos en diferentes tramas digitales el mismo dato cuando no es tratado, Hexadecimal, binario y Octal.



VII. Trabajo Previo

El trabajo previo consiste en:

4. Investigar sobre el Protocolo de comunicación Modbus
5. Traer la hoja de datos del dispositivo PM5320

VII. Conclusiones

- Compare los resultados de los valores tomados en las tablas y cuál sería la relación y diferencia entre los diferentes tipos de presentación de variables.