Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

HOMOLOGACIÓN DE PROTOCOLO DE COMUNICACIÓN MODBUS RTU A PROTOCOLO MODBUS TCP-IP UTILIZANDO UN CONTROLADOR LÓGICO PROGRAMABLE

José Alexander Vásquez Palma
Asesorado por el Ing. Luis Raúl Velásquez Herrera

Guatemala, junio de 2018

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

HOMOLOGACIÓN DE PROTOCOLO DE COMUNICACIÓN MODBUS RTU A PROTOCOLO MODBUS TCP-IP UTILIZANDO UN CONTROLADOR LÓGICO PROGRAMABLE

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

JOSÉ ALEXANDER VÁSQUEZ PALMA
ASESORADO POR EL ING. LUIS RAÚL VELÁSQUEZ HERRERA

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO ELECTRÓNICO

GUATEMALA, JUNIO DE 2018

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Pedro Antonio Aguilar Polanco
VOCAL I	Ing. Angel Roberto Sic García
VOCAL II	Ing. Pablo Christian de León Rodríguez
VOCAL III	Inga. José Milton de León Bran
VOCAL IV	Br. Oscar Humberto Galicia Nuñez
VOCAL V	Br. Carlos Enrique Gómez Donis
SECRETARIA	Inga. Lesbia Magalí Herrera López

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Pedro Antonio Aguilar Polanco
EXAMINADOR	Ing. Guillermo Antonio Puente Romero
EXAMINADORA	Inga. María Magdalena Puente Romero
EXAMINADOR	Ing. Marvin Marino Hernández Fernández
SECRETARIA	Inga. Lesbia Magalí Herrera López

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

HOMOLOGACIÓN DE PROTOCOLO DE COMUNICACIÓN MODBUS RTU A PROTOCOLO MODBUS TCP-IP UTILIZANDO UN CONTROLADOR LÓGICO PROGRAMABLE

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica, con fecha 6 de junio del 2017.

José Alexander Vásquez Palma

Guatemala, 09 de febrero del 2018

Ing. Otto Andrino
Director Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería
Universidad de San Carlos de Guatemala

Respetable Ingeniero Andrino

Por medio de la presente informo a usted que, como asesor del Trabajo de Graduación del estudiante universitario **JOSÉ ALEXANDER VASQUEZ PALMA**, quien se identifica con carné universitario No. **2011-22924**, procedí a revisar la tesis de cuatro capítulos, cuyo título es: **"HOMOLOGACION DE PROTOCOLO DE COMUNICACIÓN MODBUS RTU A PROTOCOLO MODBUS TCP-IP UTILIZANDO UN CONTROLADOR LÓGICO PROGRAMABLE"**. El cual encuentro satisfactorio.

En tal virtud, LA DOY POR APROBADA, solicitándole darle el trámite correspondiente.

Sin otro particular, es grato suscribirme de usted.

Atentamente,

Luis Raúl Velásquez Herrera
INGENIERO ELECTRÓNICO
COLEGIADO No. 11933

Ing. Luis Raúl Velásquez Herrera
Colegiado Activo No. 11933

FACULTAD DE INGENIERIA

REF. EIME 20. 2018.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen el Asesor, con el Visto Bueno del Coordinador de Área, al trabajo de Graduación del estudiante: JOSÉ ALEXANDER VÁSQUEZ PALMA titulado: HOMOLOGACIÓN DE PROTOCOLO DE COMUNICACIÓN MODBUS RTU A PROTOCOLO MODBUS TCP-IP UTILIZANDO UN CONTROLADOR LÓGICO PROGRAMABLE, procede a la autorización del mismo.

Ing. Otto Fernando Andrino González

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

GUATEMALA, 12 DE ABRIL 2018.

FACULTAD DE INGENIERIA

Guatemala, 7 de marzo de 2018

Señor Director
Ing. Otto Fernando Andrino González
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC.

Señor Director:

Por este medio me permito dar aprobación al Trabajo de Graduación titulado: **HOMOLOGACIÓN DE PROTOCOLO DE COMUNICACIÓN MODBUS RTU A PROTOCOLO MODBUS TCP-IP UTILIZANDO UN CONTROLADOR LÓGICO PROGRAMABLE**, desarrollado por el estudiante **José Alexander Vásquez Palma**, ya que considero que cumple con los requisitos establecidos.

Sin otro particular, aprovecho la oportunidad para saludarlo.

Atentamente,

ID Y ENSEÑAD A TODOS

Ing. Julio César Solares Peñate
Coordinador de Electrónica

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

de Guatemala

Facultad de Ingeniería
Decanato

DTG. 207.2018

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Mecánica Eléctrica, al Trabajo de Graduación titulado: **HOMOLOGACIÒN DE PROTOCOLO DE COMUNICACIÒN MODBUS RTU A PROTOCOLO MODBUS TCP-IP UTILIZANDO UN CONTROLADOR LÒGICO PROGRAMABLE,** presentado por el estudiante universitario: **Josè Alexander Vàsquez Palma,** y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Pedro Antonio Aguilar Polanco
Decano

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANO
FACULTAD DE INGENIERIA

Guatemala, junio de 2018

/gdech

ACTO QUE DEDICO A:

Al creador	Por ser una importante influencia en mi carrera, entre otras cosas.
Mis padres	José Pedro Vásquez Guacamaya y Francisca Idaura Palma Alarcón, mis soportes incondicionales en mi vida.
Mis hermanos	Idaura Elizabeth y Cristian Eduardo, por ser mis compañeros de vida.
Mis sobrinos	Camila Ramírez, Alexandra Ramírez, Ángela Vásquez, Sebastián Torres y Ana Torres, para que sea un buen ejemplo en su dirección educativa.
Mis primos	Diana, Karin y José Sanchinelli por su apoyo en todo momento de mi vida.
Mis tíos	Maura Palma y Hugo Sanchinelli por su apoyo y motivación en mi carrera.
Señor	Roberto Vásquez por su motivación hacia mi persona en mi carrera.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por ser de gran influencia en mi carrera.
Facultad de Ingeniería	Por su formación académica a lo largo de mi carrera.
Mis amigos de la facultad	Luis Sierra, Rony Hernández, Marcos Reyes, Kevin Sáenz, Axel Monzón, Leo Mollineo, Mónica Meneses, Jorge Vásquez, Rudy Boj, Néstor Herrera, Silvia Ramírez Girón, Luis Diego Maldonado, Esther Pineda, Guillermo Chocano y Alejandra Santiago, por hacer amena mi vida a lo largo de mi carrera.
Luis Raúl Velásquez	Por su apoyo en la culminación de mi carrera.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
ATM	*Automated teller machine*
bps	*bits per second*
CD	*Compact disc*
CRC	*Cyclic redundancy check*
DVD	*Digital video disc*
f	Frecuencia
GHz	Gigahercios
HF	*High frequency*
HTTP	*Hipertext transfer protocol*
Hz	Hercios
IP	*Internet protocol*
mA	miliamperio
MHz	Megahercios
s	segundos
T	Periodo
t	Tiempo
TCP	*Transmission control protocol*
V	Voltio

GLOSARIO

bit	Es la unidad de información más pequeña. Un *bit* sólo puede tener uno de dos valores: encendido o apagado.
bps	Son las siglas en inglés de *bits per second*, cuyo significado en español se traduce como bits por segundo.
DeviceNet	Es una red digital, multipunto para conexión entre sensores, actuadores y sistemas de automatización industrial en general.
Fieldbus	Es el nombre de una familia de protocolos industriales de redes informáticas utilizados para redes de control industrial en tiempo real, estandarizado como Norma IEC 61158.
Gateways	Es un dispositivo, con frecuencia un ordenador que permite interconectar redes con protocolos y arquitecturas diferentes a todos los niveles de comunicación.
Hercios	La unidad de frecuencia del sistema internacional de unidades es conocida como hertz o hercio en

	castellano. Está íntimamente relacionada con la propagación de las ondas electromagnéticas.
Impulso	En la física son variaciones en la intensidad o en la tensión de una corriente de tipo pulsatorio, y suelen durar tan sólo unos microsegundos y mostrar una onda angulosa.
Onda	Forma de propagación a través del espacio de los campos eléctricos y magnéticos producidos por las cargas eléctricas en movimiento.
scada	Es una aplicación software de control de producción, que se comunica con los dispositivos de campo y controla el proceso de forma automática desde la pantalla del ordenador.
Series	Una serie es un conjunto de cosas una detrás de otra, en un cierto orden.
Throughput	Es la tasa promedio de éxito en la entrega de un mensaje sobre un canal de comunicación.

RESUMEN

Unos de los fuertes objetivos de la automatización industrial es la reducción de costos financieros, la efectividad de las resoluciones de problemas y la precisión de los procesos, si se posiciona a la parte de cómo es la comunicación en estos sistemas se hablará de pulsos digitales, señales análogas, protocolos de comunicación, módulos de comunicación de distintas marcas, diferentes tipos de conectores y trasmisión de datos. En el siguiente trabajo de graduación se muestra específicamente los protocolos de comunicación RTU y tcp/ip, se hará un planteamiento de una homologación entre estos dos, para fines de reducción de costos financieros y optimización de procesos, teniendo en cuenta que RTU es un protocolo muy eficiente pero muy antiguo, una homologación a un protocolo más reciente hará que sea compatible con diversas marcas y aumentará la solución de problemas en el campo laboral, con estas configuraciones se eliminará procesos inadecuados como añadir dispositivos por cada interfaz a un solo interprete para todas las interfaces.

Se hablará de como se encapsulará la información digital por medio de un controlador lógico programable para manipular instrucciones de maestro – esclavo y otras configuraciones, también se mencionará como están constituidos estos protocolos y su funcionamiento interno. Se utilizará un software para manipular el controlador lógico programable, en este caso se utilizará la marca SIEMENS para fines didácticos. Se describirá paso a paso el proceso de homologación tanto textual como ilustrativo para fines didácticos y comodidad del lector.

OBJETIVOS

General

Desarrollar y documentar una homologación de protocolo de comunicación modbus rtu a modbus tcp-ip por medio de un controlador lógico programable.

Específicos

1. Representar los protocolos de comunicación en circuitos digitales.

2. Representar el protocolo de comunicación modbus en sus estados cliente / servidor.

3. Configurar un controlador lógico programable para interpretación de protocolos de comunicación.

4. Representar el software para la comunicación de dispositivos lógicos programables

5. Documentar la configuración de comunicación de protocolos de hardware y software de un controlador lógico programable.

INTRODUCCIÓN

Los protocolos de comunicación en el mundo de la red informática son sumamente importantes, poseen las estructuras de cómo se trasporta información ya sea inalámbrica o alámbrica, la información emitida puede ser símbolos de accionamiento, interpretación del algún lenguaje, lengua o solo una simple instrucción, su medio de trasmisión puede ser digital o analógico dependiendo de la necesidad o soluciones requeridas. Los protocolos son utilizados en todo el mundo y en casi todas las áreas comerciales y no comerciales. Se encuentran protocolos de fuente abierta, es decir, que no tienen cobre por patente, pero al utilizar varios protocolos de comunicación se vio necesario crear una homologación entre ellos.

Modbus rtu es protocolo de comunicación relativamente antiguo, por lo que hay que actualizarlo a la tecnología del día al día, como un intérprete para el protocolo tcp/ip. Es más versátil y fácil de utilizar para ello se utilizará codificación de la información mediante la configuración maestro-esclavo. Los protocolos de comunicación fueron tan eficientes que son utilizados en la industria como en el área de automatización, especialmente hablando sobre plc´s, variadores de frecuencia, entre otros. Uno de los protocolos de comunicación antiguo utilizado para automatización es modbus, el cual no posee muchas restricciones, es muy eficiente y utilizado en muchos implementos electrónicos, hay que recordar que modbus solo es un protocolo más no su trasmisión física, hay varias formar de trasmisión de datos, pero este caso se hablará específicamente sobre modbus rtu y modbus tcp-ip, como hacer una homologación o interpretación entre ellos para mayor eficiencia y evitar problemas en la industria de la automatización.

1. COMUNICACIONES DIGITALES Y PROTOCOLOS DE COMUNICACIÓN

En las comunicaciones digitales es importante tener conocimientos y conceptos básicos para implementar, razonar e interpretar los protocolos de comunicación.

A continuación se dará a conocer conceptos de señales digitales, análogas, como funciona un controlador programable y como serán implementados dentro de este trabajo de graduación.

1.1. Señales digitales

El término digital se deriva de la forma en que las computadoras realizan las operaciones contando dígitos. Durante muchos años las aplicaciones de la electrónica digital se limitaron a los sistemas informáticos. Hoy día la tecnología digital tiene aplicación en un amplio rango de áreas además de la informática. Aplicaciones como la televisión, los sistemas de comunicaciones, de radar, sistemas de navegación y guiado, sistemas militares, instrumentación médica, control de procesos industriales y electrónica de consumo, usan todos ellos técnicas digitales. A lo largo de los años, la tecnología digital ha progresado desde los circuitos de válvulas de vacío hasta los transistores discretos y los circuitos integrados, conteniendo algunos de ellos millones de transistores. Pero en este caso solo interesa la parte de industria enfocada a la parte de automatización.

Una magnitud digital es aquella que toma un conjunto de valores discretos. La mayoría de las cosas que se pueden medir cuantitativamente aparecen en la naturaleza en forma analógica.

En las aplicaciones de electrónica la representación digital presenta ciertas ventajas sobre la analógica. La principal ventaja es que el dato digital puede ser procesados y transmitidos de forma más fiable y eficiente que los datos analógicos. También los datos digitales disfrutan de una ventaja importante cuando es necesario su almacenamiento. Por ejemplo, cuando la música se convierte a formato digital puede almacenarse de manera más compacta y reproducirse con mayor precisión y claridad de lo que es posible en formato analógico. El ruido (fluctuaciones de tensión no deseadas) no afecta a los datos digitales, tanto como a las señales analógicas.

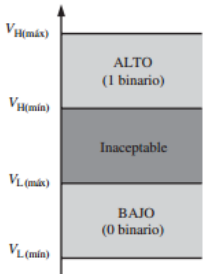
La electrónica digital utiliza sistemas y circuitos en los que sólo existen dos estados posibles. Estos estados se representan mediante dos niveles de tensión diferentes: alto *(high*) y bajo (*low*), también pueden representarse mediante niveles de corriente, *bits* y relieves en un *cd* o en un *dvd,* entre otros. En los sistemas digitales como las computadoras, las combinaciones de los dos estados denominadas códigos se emplean para representar números, símbolos, caracteres alfabéticos y otros tipos de datos. El sistema de numeración de dos estados se denomina binario y los dos dígitos que emplea son 0 y 1. Un dígito binario se denomina *bit*.

1.1.1. Dígitos binarios

Cada uno de los dos dígitos del sistema binario 1 y 0 se denomina *bit*, que es la contracción de las palabras *binary digit* (dígito binario). En los circuitos digitales se emplean dos niveles de tensión diferentes para representar los dos

bits. Por lo general, el 1 se representa mediante el nivel de tensión más elevado, que se denomina nivel alto (*high*) y 0 se representa mediante el nivel de tensión más bajo, que se denomina nivel bajo (*low*).

Figura 1. **Rango de niveles lógicos de una tensión para un circuito digital**



Fuente**:** FLOY, Thomas, *Fundamentos de sistemas digitales*. p. 8.

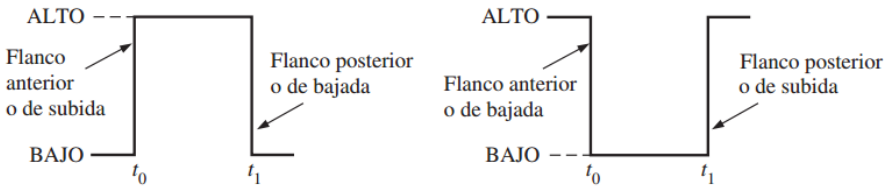
1.2. Formas de ondas digitales

Las formas de onda digitales consisten en niveles de tensión que varían entre los estados o niveles alto y bajo. La figura 2 (a) muestra que un impulso positivo se genera cuando la tensión (o la intensidad) pasa de su nivel normalmente bajo hasta su nivel alto y luego vuelve otra vez a su nivel bajo. El impulso negativo de la figura 2 (b) se genera cuando la tensión pasa de su nivel normalmente alto a su nivel bajo y vuelve a su nivel alto. Una señal digital está formada por una serie de impulsos.

El impulso como se muestra en la figura 2, tiene dos flancos: un flanco anterior que se produce en el instante t =0 y uno flanco posterior que se produce en el instante posterior t1. Para un impulso positivo, el flanco anterior es de subida y el flanco posterior es de bajada. Los impulsos mostrados en la figura 2 son ideales porque se supone que los flancos de subida y de bajada ocurren en un tiempo cero (instantáneamente). En la práctica estas transiciones

no suceden de forma instantánea, aunque para la mayoría de las situaciones digitales se puede suponer que son impulsos ideales.

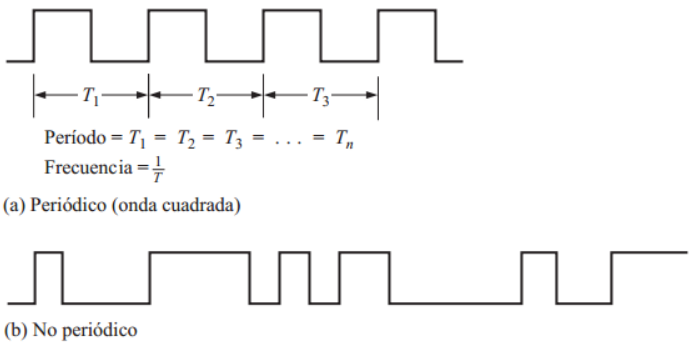
Figura 2. **Impulsos ideales**



Fuente**:** FLOY, Thomas, *Fundamentos de sistemas digitales*. p. 8.

Características de la forma de onda, la mayoría de las formas de onda que se pueden encontrar en los sistemas digitales están formadas por series de impulsos, algunas veces denominados también trenes de impulsos, y pueden clasificarse en periódicas y no periódicas. Un tren de impulsos periódico es aquel que se repite a intervalos de tiempo fijos; se denomina período (T). La frecuencia es la velocidad a la que se repite y se mide en hercios (Hz). Por supuesto, un tren de impulsos no periódico no se repite intervalos de tiempo fijos y puede estar formado por impulsos de distintos anchos o impulsos que tienen intervalos distintos de tiempo entre los pulsos.

Figura 3. **Ejemplos de ondas digitales**



Fuente**:** FLOY, Thomas, *Fundamentos de sistemas digitales.* p. 8.

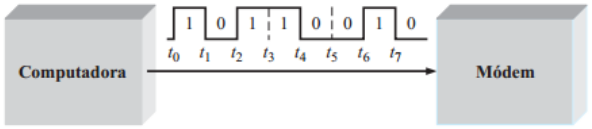
La información binaria que manejan los sistemas digitales aparece en forma de señales que representan secuencias de bits. Cuando la señal está a nivel alto, quiere decir que está presente un 1 binario cuando la señal está a nivel bajo, lo indica un 0 binario. Cada bit dentro de una secuencia ocupa un intervalo de tiempo definido, denominado período de *bit.*

1.3. Transmisión de datos digitales

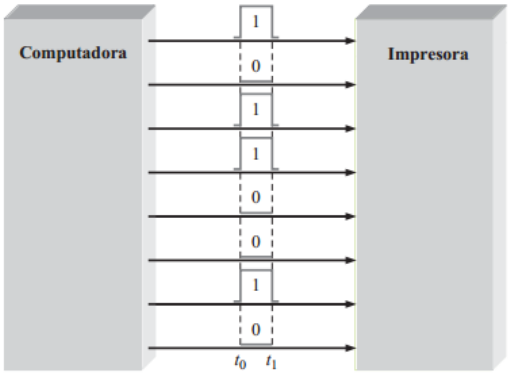
Los datos son grupos de bits que transportan algún tipo de información. Los datos binarios que se representan mediante señales digitales, deben transferirse de un circuito a otro dentro de un sistema digital o desde un sistema a otro, para servir a un propósito determinado. Por ejemplo, los números almacenados en formato binario en la memoria de una computadora se deben transferir a la unidad central de procesamiento de la computadora para poder sumarse. El resultado de la suma debe entonces transferirse a la pantalla para visualizarse o enviarse de nuevo a la memoria. En los sistemas informáticos, como se muestra en la figura 4, los datos binarios pueden transferirse de dos formas: en serie y en paralelo.

Cuando los bits se transmiten en serie de un punto a otro, se envían bit a bit a través de una sola línea, como se muestra en la figura 4 para el caso de una transmisión computadora a módem. Durante el intervalo de tiempo de t0 a t1, se transmite el primer bit. Durante el intervalo de tiempo de t1 a t2, se transmite el segundo bit, y así sucesivamente. Por tanto, la transmisión de ocho de bits en serie precisa ocho intervalos de tiempo Cuando los bits se transmiten en paralelo, todos los bits de un grupo se envían por líneas separadas al mismo tiempo. Como se muestra en la figura 4, para el ejemplo de transmisión de ocho bits desde una computadora a una impresora, existe una línea para cada bit. Para transferir ocho bits en paralelo sólo se necesita un intervalo de tiempo frente a los ocho que se precisan en la transferencia en serie.

Figura 4. **Trasmisión de datos digitales**



(a) Transferencia serie de 8 bits de datos binarios desde una computadora a un módem. El primer intervalo es de t_0 a t_1.



(b) Transferencia en paralelo de 8 bits de datos binarios desde una computadora a una impresora. t_0 es el instante inicial.

Fuente: Comunicaciones digitales www.tsc.uc3m.es/~antonio/libro_comunicaciones. p. 9 consulta: 30 de marzo 2017.

En resumen la ventaja de una transmisión en serie de datos binarios es que sólo se necesita una línea. En la transmisión en paralelo se necesitan tantas líneas como número de bits que hay que transmitir al mismo tiempo. Uno de los inconvenientes de la transmisión en serie es que tarda más tiempo en transferir un número de bits dado que la transmisión en paralelo. Por ejemplo, si un *bit* puede transferirse en un 1 µs, entonces para transmitir 8 *bits* en serie se necesitan 8 µs, pero sólo 1 µs para hacerlo en paralelo. Una desventaja de la transmisión en paralelo es que se precisan más líneas.

1.4. Canales de trasmisión de datos

La existencia de un canal es la razón que justifica la existencia de un sistema de comunicaciones, y la forma en que un canal degrada la señal que se coloca a su entrada es la guía para el diseño del sistema de comunicaciones. Por esta razón, conviene qué analizar al menos los tipos de canales más importantes.

Un primer tipo de canales son aquellos en que el medio físico empleado para la transmisión es un conductor eléctrico, tal como el par de hilos de cobre que conectan un terminal telefónico con la central o centralita a la que está asociado, el cable coaxial que conecta el módem de una red de cable con su distribuidor correspondiente, los ocho hilos de un cable de pares trenzados de los utilizados para conectar un ordenador al conmutador de la red de área local o los cuatro hilos paralelos de un cable USB (Universal Serial Bus).

Los principales mecanismos de degradación que introduce este medio son la atenuación de la señal puesta a su entrada y la adición del denominado ruido térmico. La atenuación va a depender, entre otros factores, de la longitud de los hilos o el cable, pero no de la longitud en unidades de medida como el metro, sino en longitudes de onda de la señal que lo atraviesa (supuesto que es una sinusoide), lo que implica que sinusoides de distintas frecuencias van a sufrir distintas atenuaciones.

Para una señal en general, este efecto se va a traducir en una distorsión de su forma de onda. El segundo efecto, el ruido térmico o ruido de Johnson, se produce por el movimiento desordenado de los electrones en un material

conductor y es de naturaleza aleatoria. Es, en la mayoría de los casos, el factor que limita más severamente la fiabilidad de la comunicación.

Un segundo tipo de canal es el denominado canal radioeléctrico o canal radio, que engloba todas aquellas transmisiones electromagnéticas en el espacio libre. Suele dividirse en canal radio fijo (si tanto la antena transmisora como la receptora están en una ubicación física fija) y canal radio móvil (si al menos una de las antenas está en movimiento). Ejemplos de estos canales son los radioenlaces fijos en HF (*high frequency*, de 3 a 30 MHz), el canal radio móvil en el sistema de telefonía móvil GSM en las bandas de 900 o 1 800 MHz o el canal entre un satélite de comunicaciones y un receptor terrestre en la banda K a (entre18 y 40 GHz).

Los mecanismos de propagación electromagnética son distintos en bandas distintas de frecuencia, pero en espacio libre el efecto es sólo una atenuación (aunque puede ser distinta a frecuencias distintas). El principal mecanismo de distorsión de la forma de onda es debido a las múltiples reflexiones y refracciones que pueden sufrir las ondas electromagnéticas en su propagación, lo que se conoce con el nombre de propagación multitrayecto, que provoca que en la antena receptora aparezcan réplicas de la onda original a distintos retardos y con distintas amplitudes. Adicionalmente, en el canal radio móvil (o en canales fijos si cambian las condiciones de propagación). La comparación de magnitudes se realiza mediante un circuito lógico denominado comparador. Su propósito es comparar dos cantidades e indicar si son iguales o no. Por ejemplo, si se tiene dos números y se desea saber si son iguales o no; en el caso de que no sean iguales, se desea saber cuál es el mayor.

1.5. Multiplexación

La multiplexación se refiere a la habilidad para transmitir datos que provienen de diversos pares de aparatos (transmisores y receptores) denominados canales de baja velocidad en un medio físico único (denominado canal de alta velocidad).

Un multiplexor es el dispositivo de multiplexado que combina las señales de los transmisores y las envía a través de un canal de alta velocidad. Un demultiplexor es el dispositivo de multiplexado a través del cual los receptores se conectan al canal de alta velocidad.

Figura 5. **Canal de multiplexación y demultiplexación**

Fuente**:** http://www.tyr.unlu.edu.ar/TYR-publica/04-Multiplexacion.pdf. Consulta: 02 de abril de 2018.

1.6. Tipos de velocidades

Cuando se trata de medir la velocidad de transmisión de un módem hay abundante confusión. La tasa de transferencia, rendimiento o *throughput* es el término para todo el proceso.

Se refiere a cuántos datos se mueven durante una cierta cantidad de tiempo.

Puesto que el módem es sólo una parte del proceso del movimiento de datos, adquirir un módem más rápido puede no resultar en conseguir acelerar el traslado de los datos. Hay dos clases de cosas diferentes a medir:

- El proceso digital y el proceso análogo-digital.
- La velocidad de la transmisión digital se mide en *bits* por segundo (bps).

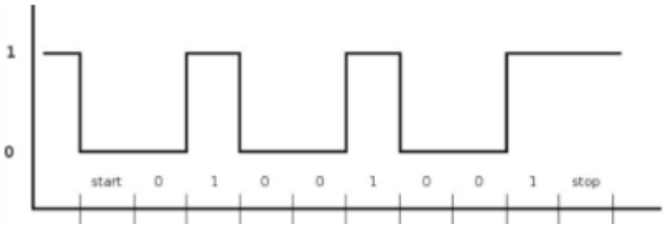
Son velocidades comunes de los módems: 28,8 Kbps, 33,6 Kbps, y 56 Kbps donde la K significa mil. Los dispositivos completamente digitales son mucho más rápidos. Entre más rápido, desde luego es mejor. Una velocidad de 2400 bps enviaría un texto de 20 páginas tapeado a un espacio, en 5 minutos.

1.6.1. Velocidad de transmisión

Bits por segundo. Es el número efectivo de bits/s que se transmiten en una línea por segundo. Un módem de 600 baudios puede transmitir a 1 200, 2 400 o incluso a 9 600 bps. Los aparatos de módem que se comercializan actualmente se diferencian notablemente entre sí en cuanto al tipo (internos o

externos) y, sobre todo, en cuanto al índice de velocidad de transmisión de datos que pueden alcanzar. Aunque las velocidades de transmisión son frecuentemente expresadas en baudios (el número de cambios de frecuencia en un segundo), ese término ya no se utiliza y en su lugar se utiliza otro más exacto: *bits* por segundo (bps).

Figura 6. **Gráfica de trasmisión de datos digitales**



Fuente**:** http://tecomunicacion.webcindario.com/Archivos/Unidad%20III.pdf. consultado: 04 de abril de 2017.

1.6.2. Velocidad de modulación (baudios)

Se define como el máximo número de cambios de estado dela señal por unidad de tiempo.

La velocidad de modulación se mide en baudios (número de *bits*/s).

VM= número de bits/tiempo.

La velocidad de modulación también se puede llamar velocidad de señalización.

La velocidad en baudios es el número de elementos de señalización por segundo. 1 baudio = 1 *bit* por segundo si cada elemento de señal transporta 1 bit. Se usa en transmisiones asíncronas. Las velocidades de transmisión de datos se miden normalmente en kilobits por segundo (Kbps). Un bit es simplemente la representación del estado eléctrico, óptico o electromagnético de la línea: tensiones, corrientes o alguna forma de señal radioeléctrica u óptica. El baudio es la unidad informática que se utiliza para cuantificar el número de cambios de estado o eventos de señalización que se producen cada segundo durante la transferencia de datos. La velocidad de transferencia de datos puede medirse en baudios o en símbolos/segundo.

1.7. Modos de transmisión, simplex, *half*-dúplex y *full* dúplex

- Dúplex: se denomina canal de comunicación al recorrido físico que es necesario establecer para que una señal eléctrica, óptica, electro óptico, se pueda desplazar entre dos puntos

- Simplex: se denomina simplex al método de transmisión en que una estación siempre actúa como fuente y la otra siempre actúa como colector. Este método permite la transmisión de información, en un único sentido. Un ejemplo de servicio simplex, es el que brindan las agencias de noticias a sus asociados.

- Half-dúplex: se denomina semidúplex (*half-dúplex*) al método de transmisión en que una estación A en un momento de tiempo, actúa como fuente y otra estación corresponsal B actúa como colector; y en el momento siguiente, la estación B actuará como fuente y la A como colector. Este método permite la transmisión en las dos direcciones,

aunque en momentos diferentes, es decir, que nunca pueden hablar ambas partes simultáneamente.

- Full-dúplex: se denomina dúplex (*full dúplex*) al método de transmisión en que dos estaciones A y B, actúan como fuente y colector transmitiendo y recibiendo información simultáneamente. Este método permite la transmisión en las dos direcciones en forma simultánea.

1.8. Protocolos de comunicación

Lo fundamental de la comunicación de datos es resolver el problema de llevar la información de un punto A hacia un punto B sin errores, utilizando red con la codificación correspondiente para su trasmisión. Para esto se utilizan canales de comunicación que establecen la unión entre los puntos A y B. En dichos puntos estarán los equipos transmisores y receptores de datos y sus convertidores encargados de la codificación y decodificación. Los sistemas de comunicación no responden ni reaccionan ante el contenido de la información. Un componente importante en el sistema de comunicación es el protocolo de comunicación.

- El protocolo: se define como las reglas para la transmisión de la información entre dos puntos. Un protocolo de red de comunicación de datos es un conjunto de reglas que gobierna el intercambio ordenado de datos dentro de la red.

Los elementos básicos de un protocolo de comunicaciones son: un conjunto de símbolos llamados conjunto de caracteres, un conjunto de reglas para la secuencia y sincronización de los mensajes construidos a partir del conjunto de caracteres y los procedimientos para determinar cuándo ha

ocurrido un error en la transmisión y como corregir el error. El conjunto de caracteres se formará de un subconjunto con significado para las personas (usualmente denominado como caracteres imprimibles) y otro subconjunto que transmite información de control (usualmente denominado caracteres de control). Hay una correspondencia entre cada carácter y los grupos de símbolos usados en el canal de transmisión, que es determinado por el código. Muchos códigos estándar con sus respectivas equivalencias de grupos de unos y ceros (bits) han sido definidos con el paso de los años. El conjunto de reglas a seguir por el emisor y el receptor propicia: que haya un significado con secuencias permitidas y a tiempo, entre los caracteres de control y los mensajes formados a partir de los símbolos. La detección de error y los procedimientos de corrección permiten la detección y la recuperación ordenada de los errores causados por factores fuera del control de la terminal en cada extremo.

Para que exista comunicación en ambos puntos al extremo de un canal se deben emplear la misma configuración de protocolos.

Los protocolos gestionan dos niveles de comunicación distintos. Las reglas de alto nivel definen como se comunican las aplicaciones, mientras que las de bajo nivel definen como se transmiten las señales.

El protocolo de bajo nivel es básicamente la forma en que las señales se transmiten, transportando tanto datos como información y los procedimientos de control de uso del medio por los diferentes nodos. Los protocolos de bajo nivel más utilizados son: Ethernet, Token ring, Token bus, FDDI, CDDI, HDLC, Frame, Relay y ATM.

El protocolo de red determina el modo y organización de la formación (tanto los datos como los controles) para su transmisión por el medio físico con

el protocolo de bajo nivel. Los protocolos de red más comunes son: IPX/SPX, DECnet, X.25, tcp/ip, AppleTalk y NetBEUI.

En un circuito de comunicación de datos, la estación que transmite en el momento se llama estación maestra, y la estación que recibe se llama esclava. En una red centralizada, la estación primaria controla cuándo puede transmitir cada estación secundaria. Cuando transmite una estación secundaria se convierte en estación maestra, y la estación primaria es ahora la esclava. El papel de estación maestra es temporal, y la estación primaria determina cuál estación es maestra. Al principio, la estación primaria es maestra. La estación primaria solicita por turno a cada estación secundaria interrogándola. Una interrogación es una invitación de la primaria a una secundaria para que transmita un mensaje. Las estaciones secundarias no pueden interrogar a una primaria. Cuando una primaria interroga a una secundaria, inicia un cambio de dirección de línea; la secundaria interrogada ha sido designada como maestra y debe responder. Si la primaria selecciona una secundaria, ésta se identifica como receptora. Una selección es una interrogación, por parte de una primaria o una secundaria, para determinar el estado de la secundaria (es decir, lista para recibir o no lista para recibir un mensaje). Las estaciones secundarias no pueden seleccionar a la primaria. Las transmisiones de la primaria van a todas las secundarias, y depende de las estaciones secundarias la decodificación individual de cada transmisión, y la determinación de si es para ellas. Cuando una secundaria transmite, sólo manda a la primaria.

Los protocolos de enlace de datos se clasifican en general como: asíncronos o síncronos. Por regla, los protocolos asíncronos usan un formato de datos asíncronos y módems asíncronos, mientras que los protocolos síncronos usan un formato de datos síncronos y módems síncronos.

Toda comunicación ya sea cara a cara o por una red está regida por reglas predeterminadas que se denominan protocolos. Estos protocolos son específicos de las características de la conversación. En la comunicación personal diaria, las reglas que se utilizan para estar en comunicación, como una llamada telefónica, no son necesariamente las mismas que los protocolos para utilizar otro medio, como enviar una carta.

A nivel humano, algunas reglas de comunicación son formales y otras simplemente se entienden o están implícitas, de acuerdo a los usos y costumbres. Para que los dispositivos se puedan comunicar en forma exitosa, un nuevo conjunto de aplicaciones de protocolos debe describir los requerimientos e interacciones precisos.

Las suites de protocolos de *networking* describen procesos como los siguientes:

- El formato o la estructura del mensaje
- El método por el cual los dispositivos de *networking* comparten información sobre las rutas con otras redes
- Cómo y cuándo se transmiten mensajes de error y del sistema entre los dispositivos
- La configuración y la terminación de sesiones de transferencia de datos

El protocolo individual en una suite de protocolos puede ser específica para el vendedor y exclusiva. Exclusiva, en este contexto, significa que una compañía o proveedor controla la definición del protocolo y cómo funciona. Algunos protocolos exclusivos los pueden utilizar distintas organizaciones con permiso del propietario. Otros, sólo se pueden implementar en equipos fabricados por el proveedor exclusivo.

Interacción de protocolos: un ejemplo del uso de una suite de protocolos en comunicaciones de red es la interacción entre un servidor y un explorador web. Esta interacción utiliza una cantidad de protocolos y estándares en el proceso de intercambio de información entre ellos. Los distintos protocolos trabajan en conjunto para asegurar que ambas partes reciben y entienden los mensajes. Algunos ejemplos de estos protocolos son:

- Protocolo de aplicación: el protocolo de transferencia de hipertexto (http) es un protocolo común que rige la forma en que interactúan un servidor web y un cliente web. http define el contenido y el formato de las solicitudes y respuestas intercambiadas entre el cliente y el servidor. Tanto el cliente como el software del servidor web implementan el http como parte de la aplicación. http se basa en otros protocolos para regular la forma en que se transportan los mensajes entre el cliente y el servidor.

- Protocolo de transporte: el protocolo de control de transmisión (tcp) es el transporte que administra las conversaciones individuales entre servidores y clientes web. tcp divide los mensajes http en pequeñas partes denominadas segmentos, para enviarlas al cliente de destino. También es responsable de controlar el tamaño y los intervalos a los que se intercambian los mensajes entre el servidor y el cliente.

- Protocolo de internetwork: el protocolo de *internetwork* más común es el protocolo de internet (ip). El ip es responsable de tomar los segmentos formateados del tcp, encapsularlos en paquetes, asignar las direcciones apropiadas y seleccionar la mejor ruta al host de destino.

- Protocolos de acceso a la red: los protocolos de acceso a la red describen dos funciones principales, la administración de enlace de datos

y la transmisión física de datos en los medios. Los protocolos de administración de enlace de datos toman los paquetes IP y los formatean para transmitirlos por los medios. Los estándares y protocolos de los medios físicos rigen de qué manera se envían las señales por los medios y cómo las interpretan los clientes que las reciben. Los transceptores de las tarjetas de interfaz de red implementan los estándares apropiados para los medios que se utilizan.

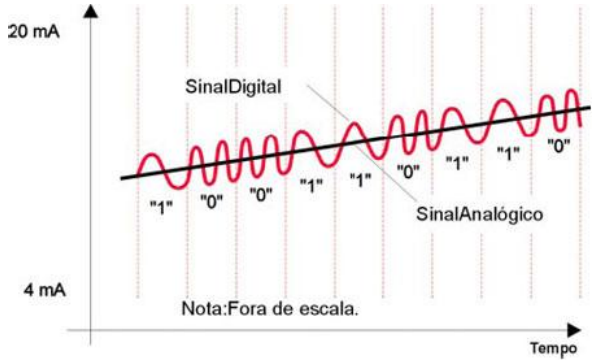
Los buses de datos que permiten la integración de equipos para la medición y control de variables de proceso, reciben la denominación genérica de buses de campo. Un bus de campo es un sistema de transmisión de información (datos) que simplifica enormemente la instalación y operación de máquinas y equipamientos industriales utilizados en procesos de producción. El objetivo de un bus de campo es sustituir las conexiones punto a punto entre los elementos de campo y el equipo de control a través del tradicional lazo de corriente de 4V-20mA o 0 a 10V DC, según corresponda. Generalmente son redes digitales, bidireccionales, multipunto, montadas sobre un bus serie, que conectan dispositivos de campo como PLC's, transductores, actuadores, sensores y equipos de supervisión.

Varios grupos han intentado generar e imponer una norma que permita la integración de equipos de distintos proveedores. Sin embargo, hasta la fecha no existe un bus de campo universal. Los buses de campo con mayor presencia en el área de control y automatización de procesos son:

- *Hart*
- *Profibus*
- *Fieldbus Foundation*

El protocolo *hart* (*high way-addressable-remote-transducer*) agrupa la información digital sobre la señal analógica típica de 4 a 20 mA DC. La señal digital usa dos frecuencias individuales de 1 200 y 2 200 Hz, que representan los dígitos 1 y 0 respectivamente y que en conjunto forman una onda sinusoidal que se superpone al lazo de corriente de 4-20 miliamperios.

Figura 7. **Diagrama de señal *hart***



Fuente: http://www.smar.com/espanol/har. Consulta: 04 de abril de 2018.

Como la señal promedio de un onda sinusoidal cero, no se añade ninguna componente DC a la señal analógica de 4-20 mA, lo que permite continuar utilizando la variación analógica para el control del proceso.

Profibus (*process field bus*) norma internacional de bus de campo de alta velocidad para control de procesos normalizada en Europa por EN 50170.

Existen tres perfiles:

- Profibus do (*decentralized periphery)*: orientado a sensores /actuadores enlazados a procesadores (PLCs) o terminales.

- Profibus *pa* (*rocess automation*): para control de proceso, cumple normas especiales de seguridad para la industria química (IEC11158-2, seguridad intrínseca).

- Profibus *fms* (*fieldbus message specification*): para comunicación entre células de proceso o equipos de automatización.

- Foundation (*fieldbus*) (FF): es un protocolo de comunicación digital para redes industriales, específicamente utilizado en aplicaciones de control distribuido. Puede comunicar grandes volúmenes de información, ideal para aplicaciones con varios lazos complejos de control de procesos y automatización. Está orientado principalmente a la interconexión de dispositivos en industrias de proceso continuo.

Los dispositivos de campo son alimentados a través del bus *fieldbus* cuando la potencia requerida para el funcionamiento lo permite.

- Modbus: es un protocolo de transmisión para sistemas de control y supervisión de procesos (sacada) con control centralizado, puede comunicarse con una o varias estaciones remotas (rtu) con la finalidad de obtener datos de campo para la supervisión y control de un proceso. La interface de capa física puede estar configurada en: RS-232, RS-422, RS-485.

En modbus los datos pueden intercambiarse en dos modos de transmisión:

- Modo RTU
- Modo ASCII

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

En la tabla I se muestra una breve comparación entre algunos protocolos y buses de comunicación.

Tabla I. **Comparación de protocolos**

Profibus DP	línea, estrella y anillo	par trenzado fibra óptica	127/segm	Hasta 1.5M y 12M	0.1 segm 24 fibra	Master/Slave peer to peer
Profibus PA	línea, estrella y anillo	par trenzado fibra óptica	14400 /segm	31.5K	0.1 segm 24 fibra	Master/Slave peer to peer
Profibus FMS		par trenzado fibra óptica	127/segm	500K		Master/Slave peer to peer
Foundation Fieldbus HSE	estrella	par trenzado fibra óptica	240 p/segm 32.768 sist	100M	0.1 par 2 fibra	Single/multi master
Foundation Fieldbus H1	estrella o bus	par trenzado fibra óptica	240 p/segm 32.768 sist	31.25K	1.9 cable	Single/multi master
LonWorks	bus, anillo, lazo, estrella	par trenzado fibra óptica coaxial, radio	32768 /dom	500K	2	Master/Slave peer to peer
Interbus-S	segmentado	par trenzado fibra óptica	256 nodos	500K	400/segm 12.8 total	Master/Slave
DeviceNet	troncal/puntual c/bifurcación	par trenzado fibra óptica	2048 nodos	500K	0.5 6 c/repetid	Master/Slave, multi-master, peer to peer
AS-I	bus, anillo, arbol, estrella	par trenzado	31 p/red	167K	0.1, 0.3 c/rep	Master/Slave
Modbus RTU	línea, estrella, arbol, red con segmentos	par trenzado coaxial radio	250 p/segm	1.2 a 115.2K	0.35	Master/Slave
Ethernet Industrial	bus, estrella, malla-cadena	coaxial par trenzado fibra óptica	400 p/segm	10, 100M	0.1 100 mono c/switch	Master/Slave peer to peer
HART		par trenzado	15 p/segm	1.2K		Master/Slave

Fuente: http://proyectointerfasesitslp.blogspot.com/2010/12/protocolos-de-redes-industriales.html. Consulta: 05 de abril de 2017.

1.9. Modelo tcp/ip y OSI

El primer modelo de protocolo en capas para comunicaciones de *internetwork* se creó a principios de la década de los setenta y se conoce con el nombre de modelo de internet. Define cuatro categorías de funciones que deben existir para que las comunicaciones sean exitosas. La arquitectura de la suite de protocolos tcp/ip sigue la estructura de este modelo. Por esto es común que al modelo de internet se le conozca como modelo tcp/ip.

El modelo tcp/ip es usado para comunicaciones en redes y como todo protocolo, describe un conjunto de guías generales de operación para permitir que un equipo pueda comunicarse en una red. tcp/ip provee conectividad de extremo a extremo, especificando cómo los datos deberían ser formateados, direccionados, transmitidos, enrutados y recibidos por el destinatario.

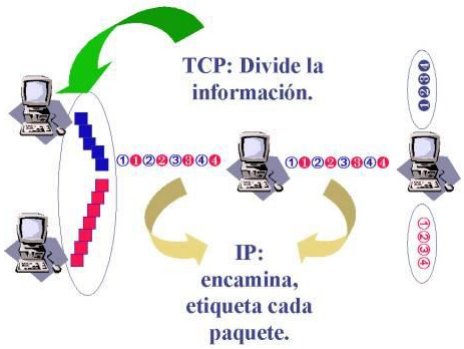
La mayoría de los modelos de protocolos describen un *stack* de protocolos específicos del proveedor. Sin embargo, puesto que el modelo tcp/ip es un estándar abierto, una compañía no controla la definición del modelo. Las definiciones del estándar y los protocolos tcp/ip se explican en un foro público y se definen en un conjunto de documentos disponibles al público. Estos documentos se denominan solicitudes de comentarios (RFC). Contienen las especificaciones formales de los protocolos de comunicación de datos y los recursos que describen el uso de los protocolos.

Este modelo está constituido por cuatro capas, aplicación, transporte, Internet y acceso a ala red.

- Aplicación: representa datos para el usuario más por el control de la codificación y el diálogo.
- Trasporte: admite la comunicación entre distintos dispositivos de distintas redes.

- Internet: determina la mejor ruta a través de la red.
- Acceso a la red: controla los dispositivos del hardware y los medios que forman la red.

Figura 8. **Modelo tcp/ip funcionamiento básico**



Fuente: http://javieralexandergf.blogspot.com/2016/03/protocolo-tcpip.html. Consulta: 05 de abril de 2017.

El modelo tcp/ip describe la funcionalidad de los protocolos que forman la suite de protocolos tcp/ip. Estos protocolos que se implementan en los hosts emisores y receptores, interactúan para brindar una entrega extremo a extremo de las aplicaciones a través de la red.

Un proceso de comunicación completo incluye estos pasos:

- Creación de datos en la capa de aplicación del dispositivo final de origen.
- Segmentación y encapsulación de datos a medida que pasan por el *stack* de protocolos en el dispositivo final de origen.
- Generación de datos en los medios en la capa de acceso a la red del *stack*.
- Transportación de los datos a través de *internetwork*, la cual está compuesta por medios y por cualquier dispositivo intermediario.

- Recepción de los datos en la capa de acceso a la red del dispositivo final de destino.
- Desencapsulación y ensamblaje de los datos a medida que pasan por el *stack* en el dispositivo de destino.
- Transmisión de estos datos a la aplicación de destino en la capa de aplicación del dispositivo final de destino.

1.10. Modelo OSI

Inicialmente el modelo OSI fue diseñado por la Organización Internacional para la Estandarización (ISO, *international organization for standardization*) para proporcionar un esquema sobre el cual crear una suite de protocolos de sistemas abiertos. La visión era que este conjunto de protocolos se utilizara para desarrollar una red internacional que no dependiera de sistemas propietarios.

Lamentablemente la velocidad a la que fue adoptada la internet con base en tcp/ip y la velocidad a la que se expandió ocasionaron que el desarrollo y la aceptación de la suite de protocolos OSI quedarán atrás. Aunque pocos de los protocolos que se crearon mediante las especificaciones OSI se utilizan ampliamente en la actualidad, el modelo OSI de siete capas ha hecho más contribuciones al desarrollo de otros protocolos y productos para todo tipo de redes nuevas.

Como modelo de referencia, el modelo OSI proporciona una amplia lista de funciones y servicios que se pueden presentar en cada capa. También describe la interacción de cada capa con las capas directamente por encima y por debajo de él. Aunque el contenido de este curso se estructura en torno al

modelo OSI, el eje del análisis son los protocolos identificados en el *stack* de protocolos tcp/ip.

Tenga en cuenta que mientras las capas del modelo tcp/ip se mencionan sólo por el nombre, las siete capas del modelo OSI se mencionan con frecuencia por número y no por nombre.

1.10.1. Capa de aplicación

La capa de aplicación es la capa del modelo OSI más cercana al usuario; suministra servicios de red a las aplicaciones del usuario. Difiere de las de más capas debido a que no proporciona servicios a ninguna otra capa OSI, sino solamente a aplicaciones que se encuentran fuera del modelo OSI. Algunos ejemplos de aplicaciones son los programas de hojas de cálculo, de procesamiento de texto y los de las terminales bancarias. La capa de aplicación establece la disponibilidad de los potenciales socios de comunicación, sincroniza y establece acuerdos sobre los procedimientos de recuperación de errores y control de la integridad de los datos.

1.10.2. Presentación

La capa de presentación garantiza que la información que envía la capa de aplicación de un sistema pueda ser leída por la capa de aplicación de otro. De ser necesario, la capa de presentación traduce entre varios formatos de datos utilizando un formato común. Si desea recordar la capa 6en la menor cantidad de palabras posible, se debe pensar en un formato de datos común.

1.10.3. Sesión

Como su nombre lo implica, la capa de sesión establece, administra y finaliza las sesiones entre dos hosts que se están comunicando. La capa de sesión proporciona sus servicios a la capa de presentación. También sincroniza el diálogo entre las capas de presentación de los dos hosts y administra su intercambio de datos. Además de regular la sesión, la capa de sesión ofrece disposiciones para una eficiente transferencia de datos, clase de servicio y un registro de excepciones acerca de los problemas de la capa de sesión, presentación y aplicación.

1.10.4. Transporte

La capa de transporte segmenta los datos originados en el host emisor y los reensambla en una corriente de datos dentro del sistema del host receptor. El límite entre la capa de transporte y la capa de sesión puede imaginarse como el límite entre los protocolos de aplicación y los protocolos de flujo de datos. Mientras que las capas de aplicación, presentación y sesión están relacionadas con asuntos de aplicaciones, las cuatro capas inferiores se encargan del transporte de datos. La capa de transporte intenta suministrar un servicio de transporte de datos que aísla las capas superiores de los detalles de implementación del transporte. Específicamente, temas como la confiabilidad del transporte entre dos hosts es responsabilidad de la capa de transporte. Al proporcionar un servicio de comunicaciones, la capa de transporte establece, mantiene y termina adecuadamente los circuitos virtuales. Al proporcionar un servicio confiable, se utilizan dispositivos de detección y recuperación de errores de transporte. Si desea recordar a la capa 4 en la menor cantidad de palabras posible, se debe pensar en calidad de servicio y confiabilidad.

1.10.5. Capa de red

La capa de red es una capa compleja que proporciona conectividad y selección de ruta entre dos sistemas de hosts que pueden estar ubicados en redes geográficamente distintas. Si desea recordarla capa 3 en la menor cantidad de palabras posible, se debe pensar en la selección de ruta, direccionamiento y enrutamiento.

1.10.6. Enlace de datos

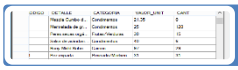
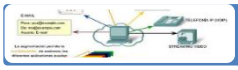
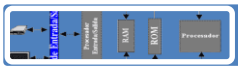
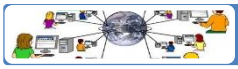
La capa de enlace de datos proporciona tránsito de datos confiable a través de un enlace físico. Al hacerlo la capa de enlace de datos se ocupa del direccionamiento físico (comparado con el lógico), la topología de red, el acceso a la red, la notificación de errores, entrega ordenada de tramas y control de flujo. Si desea recordar la capa 2 en la menor cantidad de palabras posible, piense entramas y control de acceso al medio.

1.10.7. Capa física

La capa física define las especificaciones eléctricas, mecánicas, de procedimiento y funcionales para activar, mantener y desactivar el enlace físico entre sistemas finales. Las características tales como niveles de voltaje, temporización de cambios de voltaje, velocidad de datos físicos, distancias de transmisión máximas, conectores físicos y otros atributos similares son definidos por las especificaciones de la capa física. Si desea recordar la capa 1 en la menor cantidad de palabras posible, piense en señales y medios.

En la tabla II se muestran las 7 capas del modelo OSI, con un breve ejemplo de cada una de ellas.

Tabla II. **Modelos OSI ejemplo**

	APLICACION: SMTP,FTP,HTTP	(7)
	PRESENTACIÓN: ASN1	(6)
	SESIÓN: NetBIOS	(5)
	TRANSPORTE: TCP, UDP	(4)
	RED: ARP, GATEWAY,ROUTER.	(3)
	ENLACE DE DATOS: ETHERNET, TOKENRING	(2)
	FISICA: CABLE COAXIAL, PAR TRENZADO	(1)

Fuente: elaboración propia.

1.11. Protocolos de comunicación y el *gateway*

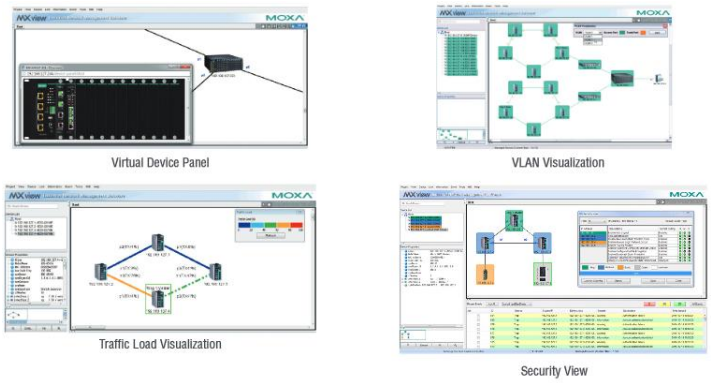
Se sabe que los *gateways* pertenecen a la capa número tres del modelo OSI, se hará referencia ya que este tema será de ayuda en las desventajas y ventajas al realizar la homologación. En el mercado de la automatización se encuentran varios métodos de soluciones de comunicación, estos módulos pueden funcionar como traductores o intérpretes. Estos módulos poseen su propia red para comunicarse remotamente o no, como por ejemplo utilizando telnet.

El *gateway* o puerta de enlace es normalmente un equipo informático configurado para dotar a las máquinas de una red local (*lan*) conectadas a él de un acceso hacia una red exterior, generalmente realizando para ello operaciones de traducción de direcciones ip (*nat: network address translation*).

Esta capacidad de traducción de direcciones permite aplicar una técnica llamada ip *masquerading* (enmascaramiento de ip), usada muy a menudo para dar acceso a internet a los equipos de una red de área local compartiendo una única conexión a internet, y por tanto, una única dirección ip externa.

Se tiene un ejemplo claro como la marca MOXA, tiene modo gráfico para configuración, donde se administra las direcciones IP de comunicación y se puede ver en tiempo real el trasporte de los paquetes de datos.

Figura 9. **Modo gráfico de módulos de homologación**



Fuente: https://www.moxa.com/product/Modbus_TCP_Gateway.htm. consulta: 06 de abril de 2017.

Figura 10. **Módulo MOXA de modbus a tcp/ip**

Fuente: https://www.commgear.com/moxa-nport-6110-series-, de RTU a tcp/ip. consulta: 06 de abril de 2017.

Por lo tanto un *gateway* permite el paso de paquetes de información de cada protocolo de comunicación, y unos especiales como este, además de trasmitir el protocolo hacen una homologación física para evitar problemas de comunicación, algo negativo de esto sería que depende de la RED o del equipo de la cantidad de interfaces, este módulo sería repetitivo y eso haría ascender cualquier presupuesto.

2. PROTOCOLO DE COMUNICACIÓN MODBUS

Modbus es un protocolo de solicitud-respuesta implementado usando una relación maestro-esclavo. En una relación maestro-esclavo, la comunicación siempre se produce en pares, un dispositivo debe iniciar una solicitud y luego esperar una respuesta y el dispositivo de inicio (el maestro) es responsable de iniciar cada interacción. Por lo general, el maestro es una interfaz humano-máquina (*hmi*) o sistema scada y el esclavo es un sensor, controlador lógico programable (plc) o controlador de automatización programable (pac). El contenido de estas solicitudes y respuestas, y las capas de la red a través de las cuales se envían estos mensajes, son definidas por las diferentes capas del protocolo.

Modbus es un protocolo de tipo petición/respuesta, por lo que en una transacción de datos se puede identificar al dispositivo que realiza una petición como el cliente o maestro, y al que devuelve la respuesta como el servidor o esclavo de la comunicación. En una red modbus se dispone de un equipo maestro que puede acceder a varios equipos esclavos. Cada esclavo de la red se identifica con una dirección única de dispositivo.

2.1. Protocolo RS-485

La *Electronics Industries Association* (EIA), en 1983 autorizó un nuevo estándar de transmisión diferencial llamado RS-485. Este estándar es similar en muchos aspectos al popular estándar EIA RS-422, de hecho RS-485 se puede considerar como el resultado de la expansión del RS-422, para permitir drivers y receptores múltiples multiterminal, compartiendo la misma línea de datos de transmisión. El protocolo RS485 tiene una gran demanda de uso industrial

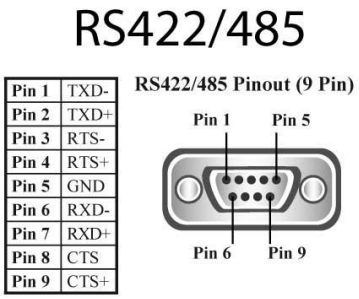
debido a que es capaz de manejar corrientes elevadas y permite que los dispositivos electrónicos sean de fácil acceso.

El estándar RS-485, como el estándar RS-422, especifica solamente las características eléctricas del driver y del receptor para ser utilizado en la línea de transmisión, pero no especifica o recomienda ningún protocolo de software, es decir, cuando se decide utilizar el protocolo RS-485 para interconectar dispositivos se deben de seguir las especificaciones de hardware que marca y que más adelante se describen, pero si se necesita otro tipo de software se deja a criterio del usuario o de la aplicación a desarrollar. El estándar EIA RS-485 ha tenido mucha aceptación.

Los usuarios son ahora capaces de hacer redes de área local económicas y enlaces en comunicaciones multiterminales utilizando cables de par trenzado y el protocolo de su opción. Dicha aceptación del estándar RS-485 está también reflejado por el hecho de que otras normas la refieren cuando se especifica un enlace de datos multiterminal, ANSI (*american national standard institute*), Normas IPI (*intelligent peripheral nterface*) y SCSI (*small computer systems interface*), han utilizado el estándar RS-485 como la base para la interconexión en modo diferencial.

En la figura 11 se hace un ejemplo de conexión de 9 pines para una configuración modbus.

Figura 11. **Diagrama de conexiones de 9 pines de un RS485**



Fuente**:** http://nilza.net/mainpage/detail/modbus-rs485-wiring-diagram. Consulta: 10 de abril de 2017.

El estándar IPI especifica la interconexión entre controladores de disco y adaptadores de *host* a velocidades de transmisión de 2,5 mega baudios sobre un enlace de datos hasta 50 metros NRZ (*non return to zero*). El estándar SCSI especifica la interconexión entre computadores personales, drives de disco, impresoras, escáner, y otros periféricos a velocidades de transmisión de 4 mega baudios sobre un enlace de 25 metros. Hasta la introducción del estándar RS-485, el estándar RS-422 fue la interconexión estándar más ampliamente aceptada para la transmisión de datos en modo diferencial.

El estándar RS-485 fue creado para permitir la conexión de varios dispositivos a través de una línea común. Cada dispositivo representa una unidad de carga (*ul = unit load*) y se denomina nodo. Una red RS-485 puede contener hasta 32 UL, donde cada UL representa una carga de 12 K. utilizando receptores de alta impedancia, la cantidad de nodos puede aumentarse hasta 256.

2.1.1. Generalidades y estructuras de conexión RS-485

El protocolo RS-485 para ser usado necesita un adaptador RS-232 que puede usarse en cualquier puerto serial de cualquier computadora o un

dispositivo entrada/salida. Su alta velocidad y su poco consumo de poder, permite al convertidor poder usarlo con el puerto serie RS-232 o cualquier alimentación externa de 3,3 hasta 40 volts. Las características con que cuenta este protocolo son:

- Puerto de energía para conexión externa
    - Conector DE-9F (DB9) con tres tipos de cable para el conector RJ11
- Entrada / salida de un *unbuffered*
    - Rango de datos de entrada es igual al rango de los datos de salida
    - Opciones eléctricas:
    - Solo transmisión
    - Solo recepción
    - RTS-hardware control de flujo TX/RX en convertidor DCE
    - CTS-hardware control de flujo TX/RX en convertidor DTE
    - Re vertidor de señal TX y RX
    - Opciones mecánicas:
    - Convertidor hembra con *thumbscrews*(tornillo manual)
    - Convertidor macho con *thumbscrews*
    - Convertidor hembra con *Jackscrews*(tornillo no manual)
    - Convertidor macho con *Jackscrews*

El cableado puede hacerse por medio de cualquier RJ11 4P/4C (4 pines / 4 conectores), 6P/4C (6 pines / 4 conectores), o el estándar 4P/AC (4pines / 4 conectores) con un acoplador de teléfono o una placa de pared *Jack*.

También se puede acoplar un RJ-45 ethernet o *jacks*. El voltaje de entrada debe de ser de 3,3 volts y hasta 40 volts. RS 485 es un dispositivo que usa una señal de par trenzado (tp), dos cables trenzados en ellos mismos. Se

está hablando de transmisión de datos balanceado, o diferencia en transmisión de voltaje. Si podría etiquetar los cables TP A y el otro B, la señal se encuentra inactiva cuando el voltaje A es negativo y el voltaje en B son positivo.

Sin embargo, sí el cable A es positivo y el voltaje B es en negativo la señal se encuentra activa. Por supuesto, la diferencia entre el cable A y B es importante. Este dispositivo es usado para comunicaciones multipunto. Mas dispositivos pueden ser conectados a un cable de señal simple, similar a la red Ethernet, cual usa cable coaxial. La mayoría de los sistemas RS-485 usan arquitectura maestro / esclavo, en donde cada esclavo cuenta con una dirección única y responde solamente a paquetes con la dirección de esa unidad. Estos paquetes están generados por el maestro, que periódicamente jala todos los dispositivos esclavos conectados.

Existen dos versiones del RS-485, el par trenzado simple opar trenzado doble que a continuación se especificarán.

Par trenzado simple: en esta versión todos los dispositivos son conectados a un par trenzado simple. Así, estos deben de tener los *driver* con sus salidas de estado triple (incluyendo al maestro). La comunicación pasa a través de la línea simple en ambas direcciones. Es importante evitar más dispositivos se transmitan a la vez (problemas de software).

Doble par trenzado: aquí el maestro no necesita la salida de estado triple, desde los dispositivos esclavos transmites arriba del segundo par trenzado, que es la intención para mandar datos desde el esclavo hacia el maestro.

Esta solución a menudo permite implementar comunicación multipunto en el sistema, donde originalmente se diseña (HW así como SW) para el RS-232. Por supuesto, el software del maestro necesita de ser modificado para mandar paquetes de preguntas a los dispositivos esclavos.

El rendimiento de procesamiento de datos creciente es evidente engrandes volúmenes. Algunas veces puedes ver el sistema RS485 en un sistema multipunto. Es virtualmente idéntico al RS-422; no es usada la salida de los *drivers* del alto esto de impedancia. La única diferencia en hardware del circuito RS-485 y RS-422 es la habilidad de mandar la salida en estado de alta impedancia.

2.1.2. Sistemas de transferencia de datos RS-485 modbus

La interfaz RS485 ha sido desarrollada analógicamente a la interfaz RS422 alta velocidad a grandes distancias y encuentra creciente aplicación en el sector industrial. Pero mientras que la RS422 sólo permite la conexión unidireccional de hasta 10 receptores en un transmisor, la RS485 establece un bus bidireccional con hasta 32 participantes. Físicamente las dos interfaces RS422 y RS485 pueden instalarse tanto como sistema de 2 hilos o de 4 hilos. Dado que varios transmisores trabajan en una línea común, tiene que garantizarse con un protocolo que en todo momento esté activo como máximo un transmisor de datos. Los otros transmisores tienen que encontrarse en ese momento en estado ultra ohmio.

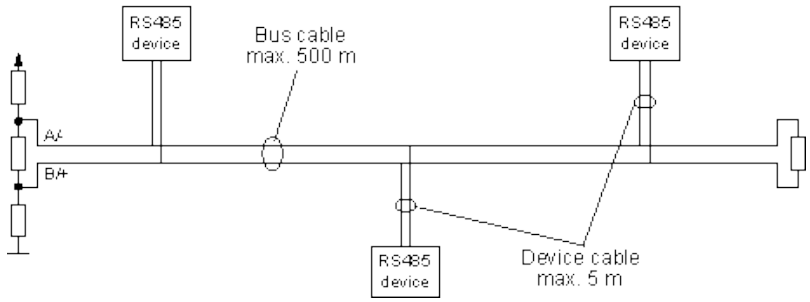
La Norma RS-485 define solamente las especificaciones eléctricas para receptores y transmisores de diferencia en sistemas de bus digitales. La norma ISO 8482 estandariza además adicionalmente la topología de longitud máx. de 500 metros.

Bus de 2 hilos RS485: se puede montar una red con dos hilos simplificando de esta forma considerablemente las conexiones en el bus de comunicación. Las redes de 2 hilos RS2485 solo pueden comunicarse de forma dual-no simultáneo (*half-dúplex*), lo que conlleva a un tipo de comunicación de pregunta-respuesta.

La ventaja de estas redes es que se puede montar una estructura multimaestro (*multi-master*), es decir, cada nodo tiene la facultad de comunicarse con cualquier otro. Evidentemente todo tiene un precio, las reglas de arbitraje y la detección de colisiones se hace imprescindible en este tipo de redes. La elección de hilos que puede manejar el protocolo RS485 es completamente dependiente del uso que tenga en el proceso y del papel que llevará a cabo.

En la figura 12 se mostrará la conexión modbus con dos hilos.

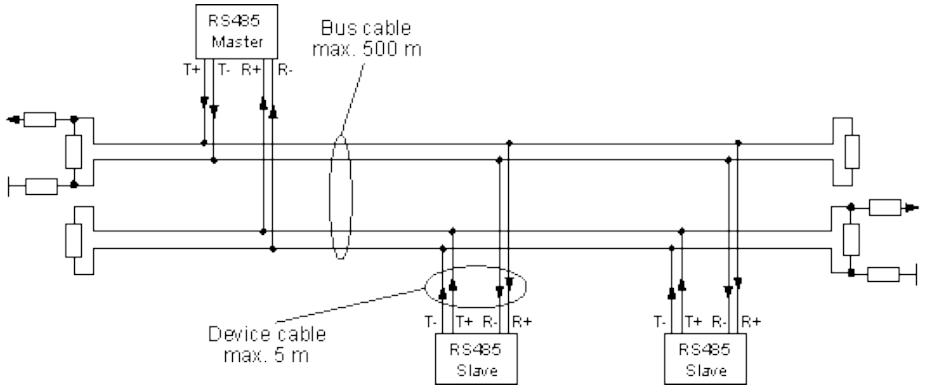
Figura 12. **Diagrama de conexión modbus de dos hilos (*half*-dúplex)**



Fuente: https://www.wut.de/e-6wwww-11-apes-000.php. Consulta: 10 de abril de 2017.

En la figura 13 se muestra la conexión de modbus de 4 hilos.

Figura 13. **Diagrama de conexión modbus 4 hilos (*full*-dúplex)**



Fuente**:** https://www.wut.de/e-6wwww-11-apes-000.php. Consulta: 10 de abril de 2017.

Bus de 4 hilos RS485: se pueden montar redes de cuatro hilos con una comunicación dual-simultánea (*full-dúplex*) con una conexión semejante a la conexión punto a punto. Aunque la comunicación entre dos nodos puede ser(*full-dúplex*) solo un nodo de la red tiene la facultad de comunicarse con todos los demás, a este nodo se le denomina maestro (*master*), al resto de nodos se les denomina esclavos (*slave*). Los modos esclavos no pueden establecer una comunicación entre sí, estos se han de comunicar siempre con el master.

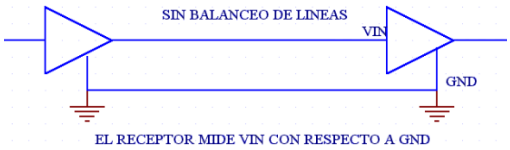
2.1.3. RS-485 balanceo de líneas

La razón por la que RS-485 puede transmitir a largas distancias, es porque utiliza el balanceo de líneas. Cada señal tiene dedicados un par de cables, sobre uno de ellos se encontrará un voltaje y en el otro se estará su complemento, de esta forma, el receptor responde a la diferencia entre voltajes.

La ventaja de las líneas no balanceadas es su inmunidad al ruido. En cuanto a las líneas balanceadas la TIA/EIA-485 designa a estas dos líneas como A y B. En el controlador TX, una entrada alta TTL causa que la línea A sea más positiva (+) que la línea B, mientras que un bajo en lógica TTL causa que la línea B sea más positiva (+) que la línea A. Por otra parte en el controlador de recepción RX, si la entrada A es más positiva que la entrada B, la salida lógica TTL será 1 y si la entrada B es más (+) que la entrada A, la salida lógica TTL será un 0.[12]

En la figura 14 se hace una analogía de una línea NO balanceada.

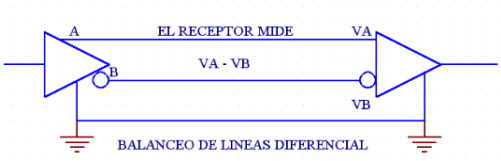
Figura 14. **Diagrama línea no balanceada por ruido**



Fuente: http://www.i-micro.com/pdf/articulos/rs-485.pdf. Consulta:10 de abril de 2017.

En la figura 15 pondremos un línea balanceada con inmunidad al ruido.

Figura 15. **Diagrama de línea balanceada inmune al ruido**



Fuente: http://www.i-micro.com/pdf/articulos/rs-485.pdf. Consulta: 10 de abril de 2017.

Usando un método de transmisión simétrico en combinación con cables de pares de baja capacidad y amortiguación (*twistedpair*) pueden realizarse conexiones muy eficaces a través de una distancia de hasta 500m con ratios de transmisión al mismo tiempo altas. El uso de un cable TP de alta calidad evita por un lado la diafonía entre las señales transmitidas y por el otro reduce adicionalmente al efecto del apantallamiento, la sensibilidad de la instalación de transmisión contra señales perturbadoras entremezcladas.

En conexiones RS485 es necesario un final de cable con redes de terminación para obligar al nivel de pausa en el sistema de bus en los tiempos en los que no esté activo ningún transmisor de datos.

Los datos en serie, como en interfaces RS422, se transmiten sin relación de masa como diferencia de tensión entre dos líneas correspondientes. Para cada señal a transmitir existe un par de conductores que se compone de una línea de señales invertida y otra no invertida.

La línea invertida se caracteriza por regla general por el índice A o -, mientras que la línea no invertida lleva B o +.

El receptor evalúa solamente la diferencia existente entre ambas líneas, de modo que las modalidades comunes de perturbación en la línea de transmisión no falsifican la señal útil. Los transmisores RS485 ponen a disposición bajo carga un nivel de salida de ±2V entre las dos salidas; los módulos de recepción reconocen el nivel de ±200mV como señal válida.

La asignación del nivel de tensión diferencial para el estado lógico está definido como sigue:

A - B < -0,3V = marck = OFF = lógico 1

A - B > +0,3V = space = ON = lógico 0

2.2. Transacciones sobre redes modbus

Los puertos estándar modbus en los controladores modicon usan una interface serial compatible RS232 que define el *pin-out* de los conectores, el cableado, los niveles de señal, las tasas de transmisión en baudios y comprobación de la paridad. Los controladores pueden ser conectados en re directamente o a través de módems. Los controladores se comunican usando una técnica maestro-esclavo, en el que sólo un dispositivo (el maestro) puede iniciar transacciones (llamadas consultas). Los otros dispositivos (los esclavos) responden al suministrar los datos solicitados para el maestro, o realizando la acción solicitada en la consulta. Los accesorios típicos maestros incluyen los procesadores del *host* y los paneles de programación. Los esclavos típicos incluyen controladores programables. El maestro puede dirigirse a esclavos individuales o iniciar una difusión de mensajes para todos los esclavos. Los esclavos devuelven un mensaje (llamado una respuesta) a las preguntas que se les envían individualmente. No se devuelven respuestas a las consultas de difusión del maestro.

El protocolo modbus establece el formato para la consulta del maestro colocando en ella el dispositivo (o emisión) dirección, un código de función que define la acción solicitada, cualquier dato a ser enviado, y un campo de comprobación de errores. El mensaje de respuesta del esclavo también se construyó utilizando el protocolo modbus. Contiene campos que confirman la acción tomada, cualquier dato que sea devuelto, y un campo de comprobación de errores. Si se produce un error en la recepción del mensaje o si el esclavo

no puede realizar la acción solicitada, el esclavo construirá un mensaje de error y lo enviará como su respuesta.

2.3. El ciclo pregunta-respuesta

La consulta: el código de función en la consulta le dice al dispositivo esclavo direccionado qué tipo de acción a realizar. Los *bytes* de datos contienen la información adicional que el esclavo necesita para realizar la función. Por ejemplo, el código de función 03 consultará al esclavo para leer registros de las explotaciones y responder con su contenido. El campo de datos debe contener la información que indica que el esclavo debe registrarse para empezar en el número de registros y de leer. El campo de comprobación de error proporciona un método para el esclavo para validar la integridad de los contenidos del mensaje.

La respuesta: si el esclavo hace una respuesta normal, el código de función en la respuesta es un eco de código de función de la consulta. Los *bytes* de datos contienen los datos recogidos por el esclavo, como los valores de registro o de su estatuto. Si se produce un error, el código defunción se a modificado 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 que el maestro para confirmar que el contenido del mensaje son válidos.

2.4. Modbus RTU

Cuando los controladores están configurados para comunicarse en una red modbus mediante el modo RTU (*remote terminal unit*), cada *byte* de 8-bitsen un mensaje contiene dos caracteres hexadecimales de 4-bits.La principal ventaja de este modo es que su mayor densidad de caracteres permite un mejor rendimiento de datos que ASCII para la misma velocidad de transmisión. Cada mensaje debe ser transmitido en un flujo continuo. El formato para cada *byte* en modo RTU es: codificación de sistema: 8-bit binario, hexadecimal 0-9, A-F, dos caracteres hexadecimales contenidos en cada campo del mensaje de 8-bit.

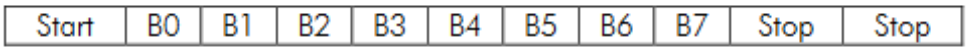
Bits por *byte*: 1 bit de inicio, 8 bits de datos, bit menos significativo enviado, 1 bit para la paridad par /impar; sin bit de no paridad, 1 bit deparada si se utiliza paridad, 2 bits si no hay paridad. Campo de comprobación de error: comprobación de redundancia cíclica (CRC).

2.4.1. Modos de transmisión

En la especificación del protocolo están definidos dos modos de transmisión: ASCII y RTU. Los modos definen la forma como son transmitidos los bytes del mensaje. No es posible utilizar los dos modos de transmisión en la misma red.

En el modo RTU cada palabra transmitida posee 1 start bit, ocho bits de datos, 2 stop bits, sin paridad. De este modo, la secuencia de bits para la transmisión de un byte es la siguiente:

Figura 16. **Secuencia de bits**



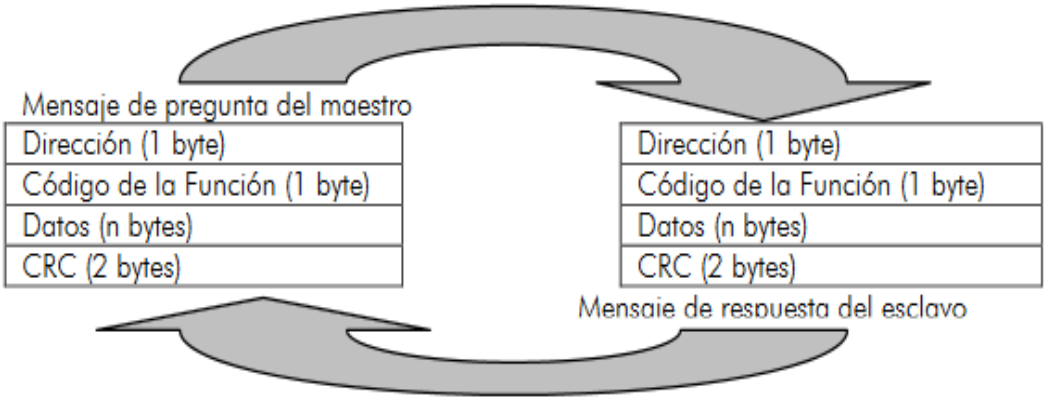
Fuente: http://ecatalog.weg.net/files/wegnet/WEG-srw01-manual-de-la-comunicacion-modbus-rtu-10000521680-4.0x-manual-espanol.pdf. Consulta: 11 de abril de 2017.

En el modo RTU cada byte de datos es transmitido como siendo una única palabra con su valor directamente en hexadecimal. El SRW 01 utiliza solamente este modo de transmisión para comunicación, no poseyendo, por lo tanto, comunicación en el modo ASCII.

2.4.2. Estructura de los mensajes en el modo RTU

La red modbus - RTU utiliza el sistema maestro-esclavo para el intercambio de mensajes. Permite hasta 247 esclavos, más solamente un maestro. Toda comunicación inicia con el maestro haciendo una solicitación a un esclavo, y este contesta al maestro el que fue solicitado. En ambos los telegramas (pregunta y respuesta), la estructura utilizada es la misma: dirección, código de la función, datos y *checksum*. Solo el contenido de los datos posee tamaño variable.

Figura 17. **Diagrama de bloques de maestro y esclavo de RTU**



Fuente**:** http://ecatalog.weg.net/files/wegnet/WEG-srw01-manual-de-la-comunicacion-modbus-rtu-10000521680-4.0x-manual-espanol.pdf. Consulta: 11 de abril de 2017.

A continuación se explicará cada uno de los bloques durante el proceso de transmisión de información.

- Dirección: el maestro inicia la comunicación enviando un byte con la dirección del esclavo para el cual se destina el mensaje. Al enviar la respuesta, el esclavo también inicia el telegrama con la su propia dirección, posibilitando que el maestro conozca cual esclavo está enviándole la respuesta. El maestro también puede enviar un mensaje destinado a la dirección 0 (cero), lo que significa que el mensaje es destinado a todos los esclavos de la red (*broadcast*). En este caso, ninguno esclavo irá contestar al maestro.

- Código función: este campo también contiene un único byte, donde el maestro especifica el tipo de servicio o función solicitada al esclavo (lectura, escrita, entre otros). De acuerdo con el protocolo, cada función es utilizada para acceder un tipo específico de dato. En el SRW 01, los datos están dispuestos como registradores del tipo *holding* (*words*), o del

tipo *coil/input discrete* (*bits*), y, por lo tanto, el relé solo acepta funciones que manipulan estos tipos de datos.

- Campo de datos: campo con tamaño variable. El formato y el contenido de este campo dependen de la función utilizada y de los valores transmitidos. Este campo está descrito juntamente con la descripción de las funciones.

- CRC: la última parte del telegrama es el campo para el chequeo de errores de transmisión. El método utilizado es el CRC-16 (*cycling redundancy check).* Este campo es formado por dos bytes, donde primero es transmitido el byte menos significativo (CRC-), y después el más significativo (CRC+).El cálculo del CRC es iniciado cargándose una variable de 16 bits (referenciado a partir de ahora como variable CRC) con el valor FFFFh. Después se debe ejecutar los pasos de acuerdo con la siguiente rutina:

    - Se somete al primer byte del mensaje (solamente los bits de datos -*start bit, paridad y stop bit* no son utilizados) a una lógica XOR (O exclusivo) con los 8 *bits* menos significativos de la variable CRC, retornando el resultado en la propia variable CRC;
    - Entonces, la variable CRC es desplazada una posición a la derecha, en dirección al *bit* menos significativo, y la posición del bit más significativo es rellenada con 0 (cero);
    - Luego de este desplazamiento, el *bit* de *flag* (*bit* que fue desplazado para fuera de la variable CRC) es canalizado, ocurriendo lo siguiente:

- Si el valor del bit fuera 0 (cero), nada es hecho.

- Si el valor del bit fuera 1 (uno), el contenido de la variable CRC es sometida a una lógica XOR con un valor constante de A001h y el resultado es regresado a la variable CRC
  - Se repiten los pasos 2 y 3 hasta que ocho desplazamientos tengan sido hechos.
  - Se repiten los pasos de 1 a 4, utilizando el próximo byte del mensaje, hasta que todo el mensaje tenga sido procesado. El contenido final de la variable CRC es el valor del campo CRC que es transmitido en el final del telegrama. La parte menos significativa es transmitida primero (CRC-) y en seguida la parte más significativa (CRC+).

2.4.3. Funciones y registros

La tabla III muestra las funciones más utilizadas en las peticiones y respuestas de modbus, con sus códigos.

Tabla III. **Códigos más comunes en modbus**

Códigos de función más comunes de MODBUS			
Código decimal	**Código hexadecimal**	**Función**	**Tipo de datos**
1	16#01	Leer estado de marcas y salidas digitales (bobinas)	Bit
2	16#02	Leer estado de entradas digitales	Bit
3	16#03	Leer registros	Entero 16 bits
4	16#04	Leer entradas analógicas	Entero 16 bits
5	16#05	Forzar valor de una salida digital (bobina)	Bit
6	16#06	Establecer valor de un registro	Entero 16 bits
15	16#0F	Forzar múltiples marcas o salidas digitales (bobinas)	Bit
16	16#10	Establecer múltiples registros	Entero 16 bits

Fuente**:** https://rua.ua.es/dspace/bitstream/10045/18990/1/AA-p3.pdf. Consulta: 11 de abril de 2017.

El formato de los campos de función y de datos de las tramas de modbus depende de la función utilizada. Esta última trama es un caso especial de respuesta enviada por un esclavo cuando tiene problemas para atender una petición.

Figura 18. **Registros de modbus**

Tabla de correspondencias de registros de MODBUS					
Función	**Dirección MODBUS**	**Dirección en dispositivo**	**Dirección IEC61131**	**Tipo de registro**	**Tipo de acceso**
16#01, 16#05, 16#0F	0 a 9.999	1 a 10.000	%M0, %M1…	Salidas o registros de aplicación digitales (bits)	Lectura y escritura
16#02	0 a 9.999	10.001 a 20.000	%I0, %I1…	Entradas digitales (bits)	Lectura
16#04	0 a 9.999	30.001 a 40.000	%IW0, %IW1…	Entradas analógicas (entero)	Lectura
16#03, 16#06, 16#10	0 a 9.999	40.001 a 50.000	%MW0, %MW1…	Registro general de la aplicación (entero)	Lectura y escritura

Fuente: https://rua.ua.es/dspace/bitstream/10045/18990/1/AA-p3.pdf. Consulta: 11 de abril de 2017.

2.5. Método de verificación de error

Las redes serial estándar modbus utilizan dos tipos de comprobación de errores. La comprobación de paridad (par o impar) puede aplicarse opcionalmente a cada carácter. La trama de control (LRC o CRC) se aplica a todo el mensaje. Tanto el carácter de comprobación de verificación de trama y mensaje son generados en el dispositivo maestro y aplican el contenido del mensaje antes de la transmisión. El dispositivo esclavo comprueba cada carácter y la trama de mensaje durante toda la recepción. El maestro está configurado por el usuario para esperar durante un intervalo de tiempo de espera predeterminado antes de abortar la transacción. Este intervalo se fija con el tiempo suficiente para que cualquier esclavo pueda responder con normalidad. Si el esclavo detecta un error de transmisión, el mensaje no actúa

sobre ellos. El esclavo no construirá una respuesta al maestro. Así, el tiempo de espera expira y permite que el programa del maestro maneje el error.

Tenga en cuenta que si un mensaje dirigido a un dispositivo esclavo no existente también causará un *time out*. En virtud de las redes, el campo del mensaje modbus LRC o CRC de verificación no se aplica. En el caso de un error de transmisión, los protocolos de comunicación específicos a esas redes notifican al dispositivo de origen que ha ocurrido un error, y permiten que se vuelva a intentar o abortar de acuerdo con la forma en que ha sido configurada.

Si el mensaje es entregado, pero el dispositivo esclavo no puede responder, un error de tiempo de espera puede ocurrir y puede ser detectado por el programa del maestro.

2.5.1. Control de paridad

Los usuarios pueden configurar los controladores para la comprobación de paridad par o impar, o sin comprobación de paridad. Esto determinará cómo el bit de paridad se establecerá en cada personaje. Si cualquiera de paridad par o Impar especifica la cantidad de bits 1 se contará en la parte de datos de cada carácter (siete bits de datos para el modo ASCII, y ocho para RTU). El bit de paridad a continuación se establece en un 0 o 1 para dar como resultado un total de par o impar de bits 1. Por ejemplo, estos ocho bits de datos están contenidos en una trama de caracteres RTU: 1100 0101. La cantidad total de 1 bits en la trama es de cuatro. Si se utiliza paridad par, el bit de la trama paridad será un 0, por lo que la cantidad total de bits 1 todavía es un número par (cuatro). Si se utiliza paridad impar, el bit de paridad será un 1, por lo que una cantidad impar (cinco). Cuando se transmite el mensaje, el bit de paridad se calcula y se aplicará a la estructura de cada carácter. El dispositivo de

recepción cuenta la cantidad de bits 1 y establece un error sino son el mismo que el configurado para ese dispositivo (todos los dispositivos en la red modbus deben estar configurados para utilizar el método de verificación de paridad mismo).

Tenga en cuenta que la comprobación de paridad sólo puede detectar un error si un número impar de bits son recogidos o dejados en un marco de carácter durante la transmisión. Por ejemplo, si la comprobación de paridad impar se emplea, y dos bits 1 se reducen de un carácter que contiene tres bits de 1, el resultado es todavía un conteo impar de bits 1. Si no se especifica la comprobación de paridad, el bit de paridad no se transmite y sin verificación de paridad se pueden hacer. Un bit de parada adicional se transmite al llenar lat rama de caracteres.

2.5.2. Control LRC

En el modo ASCII los mensajes incluyen un campo de comprobación de errores que se basan en un método de comprobación de redundancia longitudinal (LRC). El campo LRC comprueba el contenido del mensaje, sin contar los dos puntos al comienzo y finalización par CRLF. Se aplica independientemente de cualquier método de comprobación de paridad utilizado para los caracteres individuales del mensaje.

El campo LRC es de un byte contiene un valor binario de 8-bits. El valor de LRC es calculado por el dispositivo de transmisión, que agrega el LRC al mensaje. El dispositivo de recepción calcula una LRC durante la recepción del mensaje, y compara el valor calculado con el valor real que recibió en el campo LRC. Si los dos valores no son iguales, se producirá un error. El LRC se calcula mediante la suma de los sucesivos bytes de 8 bits del mensaje, descartando

cualquier acarreo, y luego los dos complementan el resultado. Se realiza en el contenido del campo de mensaje ASCII excluyendo el carácter colon con que comienza el mensaje, y excluyendo el par CRLF al final del mensaje. En la lógica de escalera, la función CKSM calcula un LRC de los contenidos del mensaje.

2.5.3. Control CRC

En el modo RTU los mensajes incluyen un campo de comprobación de errores que se basa en un método comprobación de redundancia cíclica (CRC). El campo CRC comprueba el contenido de todo el mensaje. Se aplica independientemente de cualquier método de comprobación de paridad utilizado para los caracteres individuales del mensaje. El campo CRC es de dos bytes, contiene un valor binario de 16-bit. El valor CRC se calcula por el dispositivo de transmisión, que agrega el CRC para el mensaje.

El dispositivo receptor vuelve a calcular un CRC durante la recepción del mensaje, y compara el valor calculado con el valor real recibido en el campo CRC. Si los dos valores no son iguales, se producirá un error. El CRC es iniciado por primera pre cargado de un registro de 16-bit con todos 1. Entonces se inicia un proceso de aplicación de sucesivas de 8-bits del mensaje con el contenido actual del registro. Sólo los ocho bits de datos en cada carácter se utilizan para la generación de la CRC. Iniciar y detener bits y el bit de paridad, no se aplican a la CRC.

Durante la generación del CRC, cada carácter de 8-bits es OR exclusivo con el contenido del registro. A continuación el resultado se desplaza en la dirección del bit menos significativo (LSB), con un cero llena en la posición del bit más significativo (MSB). El LSB se extrae y se examina. Si el LSB es un 1, el

registro es entonces OR exclusivo con un valor predeterminado, fijo. Si el LSB es un 0, no se produce un OR exclusivo. Este proceso se repite hasta que ocho cambios se han realizado.

Después del último cambio (octava), el siguiente byte de 8-bits es OR exclusivo con el valor actual del registro, y el proceso se repite para ocho desplazamientos más como se describe anteriormente. Los contenidos finales del registro, después de que todos los bytes del mensaje han sido aplicados, es el valor de CRC.

Cuando el CRC se anexa al mensaje, el byte de orden inferior se añade primero, seguido por el byte de orden superior. En la lógica de escalera, la función CKSM calcula un CRC de los contenidos del mensaje.

2.6. Modbus tcp/ip

Es la evolución más utilizada/conocida, una versión del protocolo modbus que permite la implementación de este protocolo sobre redes ethernet ip, en consecuencia, aumenta el grado de conectividad. Es muy semejante al formato RTU, pero estableciendo la transmisión mediante paquetes tcp/ip (puerto del sistema 502, identificador *asa-appl-proto*). Esta versión del protocolo encapsula la trama base del protocolo modbus en la capa de aplicación tcp/ip de forma sencilla.

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 EEUU o cualquier otra parte del mundo.

Las ventajas para los instaladores o empresas de automatización son innumerables:

- 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.

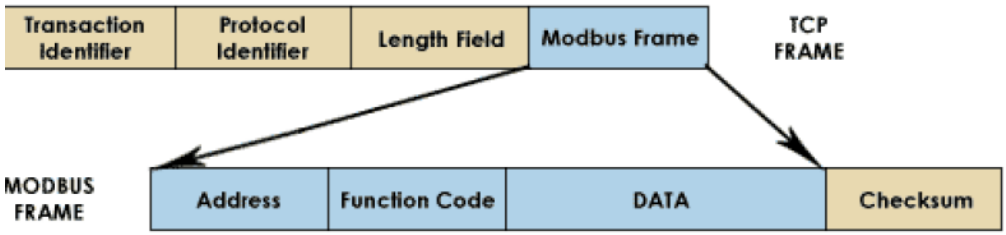
Modbus tcp-ip se ha convertido en un estándar industrial de facto debido a su simplicidad, bajo coste, necesidades mínimas en cuanto a componentes de hardware, y sobre todo a que se trata de un protocolo abierto. En la actualidad hay cientos de dispositivos modbus tcp-ip disponibles en el mercado. Se 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 inter redes 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.

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.

En la figura 19 se observará la encapsulación de la trama modbus tcp.

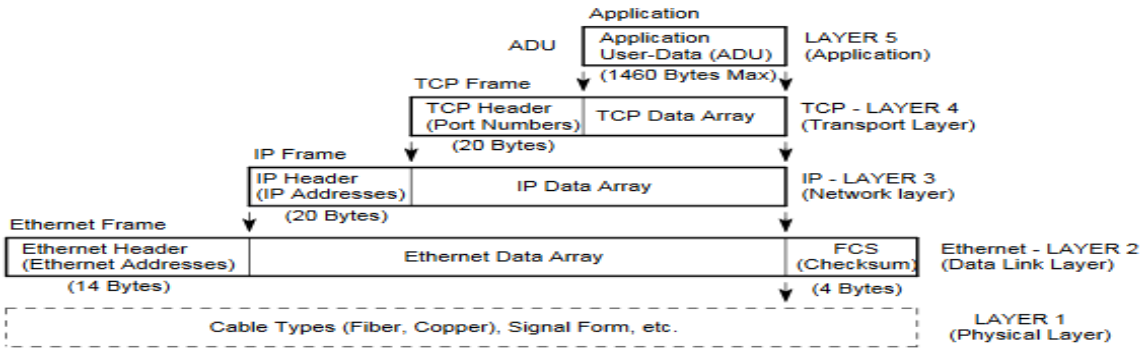
Figura 19. **Diagrama de bloques de modbus tcp**



Fuente**:** http://uhu.es/antonio.barragan/content/modbus-tcp. Consulta: 11 de abril de 2017.

En la figura 20 será una construcción de un paquete de datos tcp/ip-ethernet.

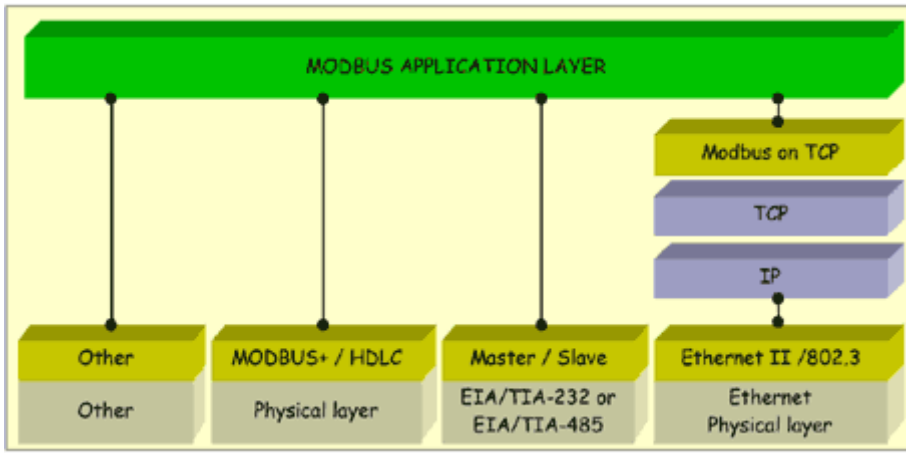
Figura 20. **Diagrama de bloques de construcción de paquetes de datos tcp/ip**



Fuente**:** https://www.prosoft-technology.com/kb/assets/intro_modbustcp.pdf. Consulta:11 de abril de 2017.

En siguiente es un ejemplo secuencial de modbus tcp

Figura 21. **Secuencia de aplicación de modbus sobre tcp**



Fuente**:** http://www.anybus.jp/technologies/modbustcp2.shtml. Consulta: 11 de abril 2017.

3. CONTROLADOR LÓGICO PROGRAMABLE

En el siguiente capítulo se dará conocer qué es un controlador lógico programable en la parte de la industria, cómo se va a relacionar dicho controlador con los protocolos de comunicación, también para fines didácticos, se conocerán los aspectos básicos del controlador lógico programable Siemens S7-1200, entradas y salidas, módulos, estructura de comunicación interna, sus conexiones físicas con respectivos protocolos de comunicación y algunos tipos de lenguajes de programación y configuración del mismo.

3.1. Controlador lógico programable (PLC)

Según lo define la Asociación Nacional de Fabricantes Eléctricos de los Estados Unidos un PLC – *Programable Logic Controller* (controlador lógico programable) es un dispositivo digital electrónico con una memoria programable para el almacenamiento de instrucciones, permitiendo la implementación de funciones específicas como ser: lógicas, secuenciales, temporizadas, de conteo y aritméticas; con el objeto de controlar máquinas y procesos.

También se puede definir como un equipo electrónico, el cual realiza la ejecución de un programa de forma cíclica. La ejecución del programa puede ser interrumpida momentáneamente para realizar otras tareas consideradas más prioritarias, pero el aspecto más importante es la garantía de ejecución completa del programa principal.

Estos controladores son utilizados en ambientes industriales donde la decisión y la acción deben ser tomadas en forma muy rápida, para

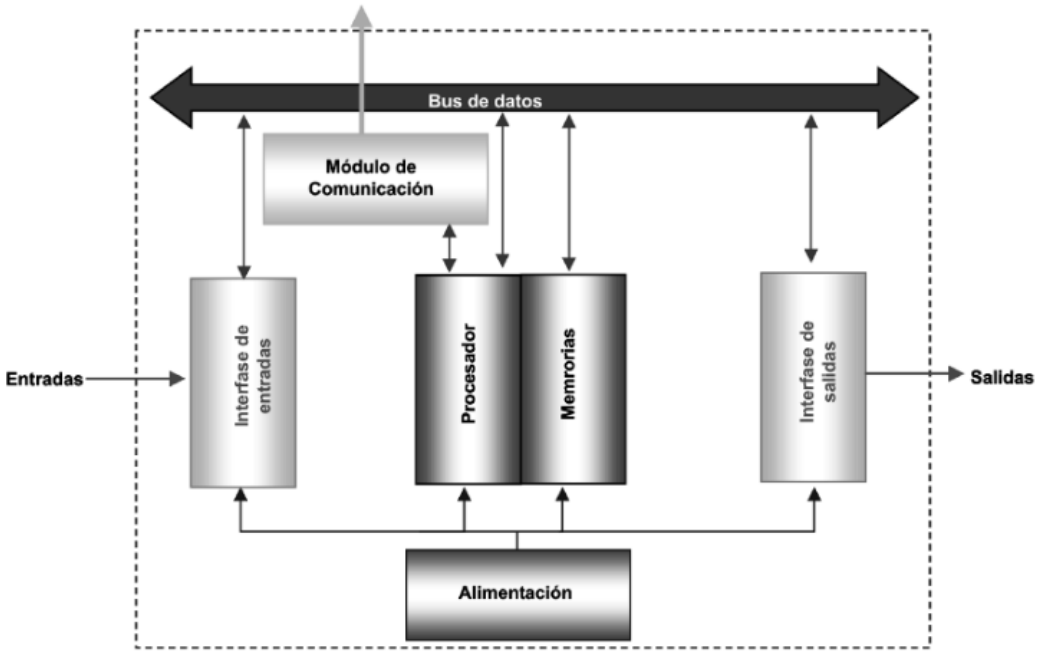
responder en tiempo real. Los PLC son utilizados donde se requieran tanto controles lógicos como secuenciales o ambos a la vez.

3.2. Estructura de un PLC

Un PLC básico está compuesto por tres partes:

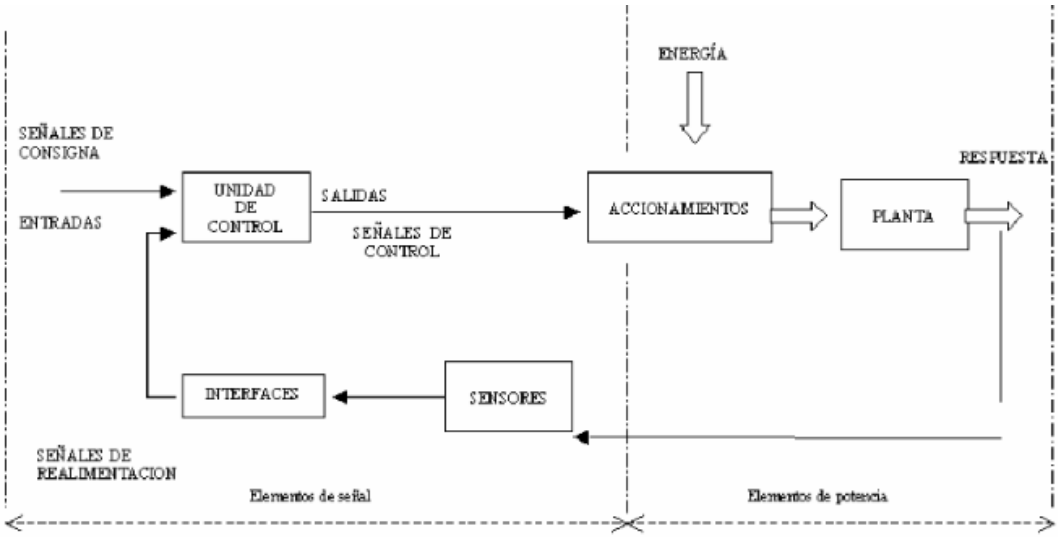
- CPU.
- Interfaces de entrada.
- Interfaces de salida.

Figura 22. **Estructura básica de PLC**



Fuente: http://www.microautomacion.com/capacitacion/Manual061ProgramablePLC.pdf. Consulta: 15 de abril de 2017.

Figura 23. **Diagramación de ejemplo de proceso de PLC en planta**



Fuente: https://www.dsi.fceia.unr.edu.ar/downloads/digital_I/Apunte_PLC.pdf. p.1. Consulta: 15 de abril de 2017.

3.2.1. Procesador

La función principal que tiene es procesar los datos y transferirlos a los otros elementos del computador. Estas tareas se llevan a cabo mediante la ejecución de instrucciones.

Entre sus tareas principales están:

- Ejecutar el programa realizado por el usuario.
- Administración de la comunicación entre el dispositivo de programación y la memoria, y entre el microprocesador y los bornes de entrada/salida.
- Ejecutar los programas de autodiagnósticos.

Para realizar todas estas tareas, el procesador necesita un programa escrito por el fabricante, llamado sistema operativo. Este programa no es

accesible por el usuario y se encuentra grabado en una memoria que no pierde la información ante la ausencia de alimentación, es decir, en una memoria no volátil.

3.2.2. Memoria

Los PLC tienen que ser capaces de almacenar y retirar información, para ello cuentan con memorias. Las memorias son miles de cientos de localizaciones donde la información puede ser almacenada. Estas localizaciones están muy bien organizadas. En las memorias el PLC debe ser capaz de almacenar:

- Datos del proceso:
    - Señales de entradas y salidas.
    - Variables internas, de bit y de palabra.
    - Datos alfanuméricos y constantes.
- Datos de control:
    - Instrucciones de usuario, programa.
    - Configuración del autómata.

Los tanto el sistema operativo como el programa de aplicación, las tablas o registros de entradas/ salidas y los registros de variables o *bits* internos están asociados a distintos tipos de memoria.

La capacidad de almacenamiento de una memoria suele cuantificarse en *bits*, *bytes* grupo de 8 bits), o *words* (grupo de 16 *bits*).

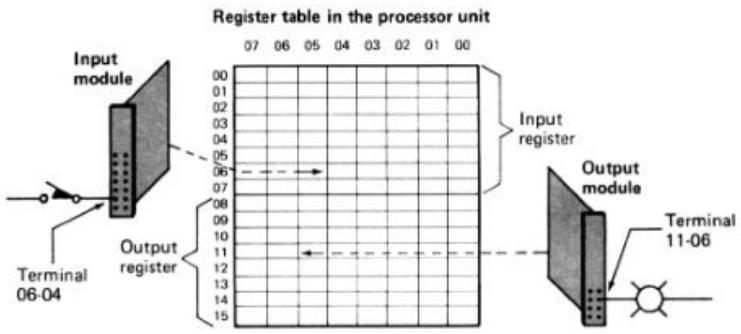
El sistema operativo viene grabado por el fabricante. Como debe permanecer inalterado y el usuario no debe tener acceso a él, se guarda en una memoria como las *ROM* (*read only memory*), que son memorias cuyo contenido no se puede alterar inclusive con ausencia de alimentación.

3.2.3. Tipos de memoria

Hay varios tipos de memorias como se harpa mención a continuación.

- La memoria de datos: también llamada tabla de registros, se utiliza tanto para grabar datos necesarios a los fines de la ejecución del programa, como para almacenar datos durante su ejecución o retenerlos luego de haber terminado la aplicación. Este tipo de memorias contiene la información sobre el estado presente de los dispositivos de entrada y salida. Si un cambio ocurre en los dispositivos de entrada o salida, ese cambio será registrado inmediatamente en esta memoria.

Figura 24. **Relación entre las terminales de salida, terminales de entrada y localización de registro de entradas/salidas**

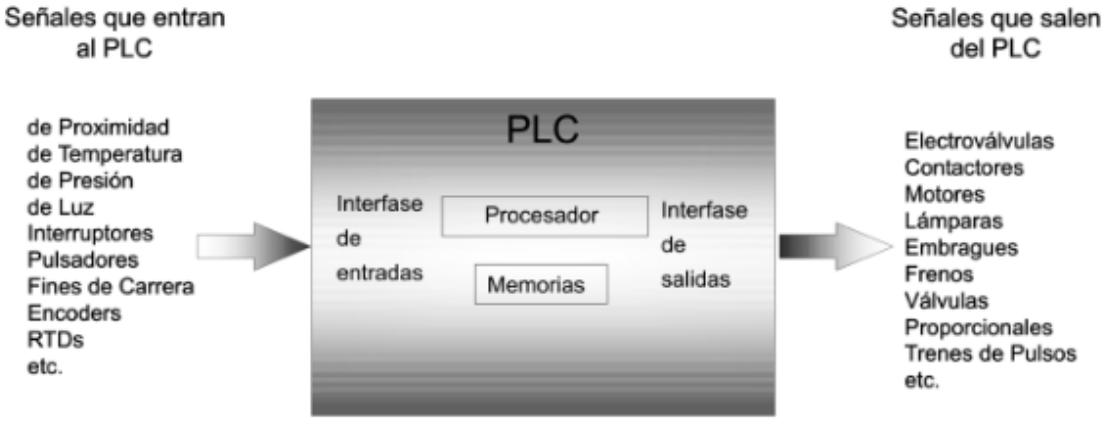


Fuente: http://www.microautomacion.com/capacitacion/Manual061ProgramablePLC.pdd. Consulta: 15 de abril de 2017.

En resumen esta memoria es capaz de guardar información originada en el microprocesador incluyendo: tiempos, unidades de conteo y relés internos.

- Memoria de usuario: es la memoria utilizada para guardar el programa. El programa construido por el usuario debe permanecer estable durante el funcionamiento del equipo, además debe ser fácil de leer, escribir o borrar. Por eso es que se usa para su almacenamiento memorias tipo *ram*, o *EEPROM*. A estas memorias se la llama memoria del usuario o memoria de programa. En el caso de usar memorias tipo RAM será necesario también el uso de pilas, ya que este tipo de memoria se borra con la ausencia de alimentación. En el caso de usar memorias EEPROM la información no se pierde al quitar la alimentación.

Figura 25. **Tipos de señales I/O de un PLC básico**



Fuente: http://www.microautomacion.com/capacitacion/Manual061ProgramablePLC.pdf. Consulta: 15 de abril de 2017.

3.2.4. Entradas y salidas

Dispositivos de entrada**:** los dispositivos de entrada y salida son aquellos equipos que intercambian (o envían) señales con el PLC. Cada dispositivo de entrada es utilizado para conocer una condición particular de su entorno, como temperatura, presión, posición, entre otras.

Entre estos dispositivos podemos encontrar:

- Sensores inductivos magnéticos, ópticos, pulsadores, termocoplas, termoresistencias, *encoders*, entre otros.

Dispositivos de salida: los dispositivos de salida son aquellos que responden a las señales que reciben del PLC, cambiando o modificando su entorno.

Entre los dispositivos típicos de salida podemos hallar:

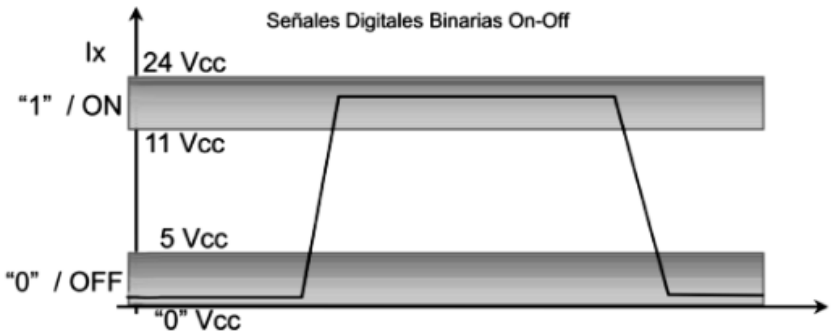
- Contactores de motor
- Electroválvulas
- Indicadores luminosos o simples relés

Generalmente los dispositivos de entrada, los de salida y el microprocesador trabajan en diferentes niveles de tensión y corriente. En este caso las señales que entran y salen del PLC deben ser acondicionadas a las tensiones y corrientes que maneja el microprocesador, para que éste las pueda reconocer. Ésta es la tarea de las interfaces o módulos de entrada o salida. Las entradas se pueden clasificar en:

Entradas digitales: también llamadas binarias u *on-off,* son las que pueden tomar sólo dos estados: encendido o apagado, estado lógico 1 o 0.

Los módulos de entradas digitales trabajan con señales de tensión. Cuando por un borne de entrada llega tensión, se interpreta como 1 y cuando llega cero tensión se interpreta como 0. Existen módulos o interfaces de entradas de corriente continua para tensiones de 5, 12, 24 o 48 Vcc y otros para tensión de 110 o 220 Vca.

Figura 26. **Interpretación de una señal digital binaria**



Fuente: http://www.microautomacion.com/capacitacion/Manual061ProgramablePLC.pdf.p.12. Consulta: 15 de abril de 2017.

Hay que recordar que las señales digitales en contraste con las señales analógicas no varían en forma continua, sino que cambian en pasos o en incrementos discretos en su rango. La mayoría de las señales digitales utilizan códigos binarios o de dos estados. Las entradas discretas, tanto las de la corriente continua como las de la corriente alterna, están compuestas por una estructura típica que se puede separar en varios bloques:

Figura 27. **Estructura entradas discretas**

Fuente: elaboración propia.

- Rectificador: en el caso de una entrada de corriente alterna, convierte la señal en continua. En el caso de una señal de corriente continua, impide daños por inversión de polaridad.

- Acondicionador de señal: elimina los ruidos eléctricos, detecta los niveles de señal para los cuales conmuta el estado lógico, y lleva la tensión al nivel manejado por la CPU.

- Indicador de estado: en la mayoría de los PLC existe un indicador luminoso por cada entrada. Este indicador (casi siempre un *led*) se encenderá con la presencia de tensión en la entrada y se apagará en caso contrario.

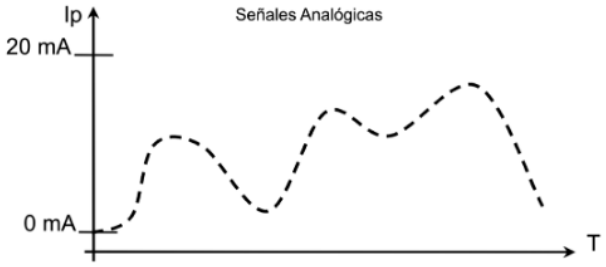
- Aislación: en la mayoría de los PLC las entradas se encuentran aisladas para que, en caso de sobretensiones externas, el daño causado no afecte más que a esa entrada, sin perjudicar el resto del PLC.

- Circuito lógico de entrada: es el encargado de informar a la CPU el estado de la entrada cuando éste lo interrogue.

- Entradas analógicas: estos módulos o interfaces admiten como señal de entrada valores de tensión o corriente intermedios dentro de un

rango, que puede ser de 4-20 mA, 0-5 VDC o 0-10 VDC, convirtiéndola en un número. Este número es guardado en una posición de la memoria del PLC.

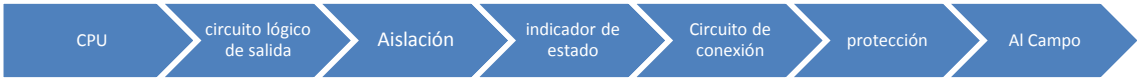
Figura 28. **Ejemplo de señal analógica para un PLC**



Fuente: http://www.microautomacion.com/capacitacion/Manual061ProgramablePLC.pdf. p1. Consulta: 15 de junio de 2017.

Los módulos de entradas analógicas son los encargados de traducir una señal de tensión o corriente proveniente de un sensor de temperatura, velocidad, aceleración, presión, posición, o cualquier otra magnitud física que se quiera medir en un número para que el PLC la pueda interpretar. En particular es el conversor analógico digital (A/D) el encargado de realizar esta tarea. Una entrada analógica con un conversor A/D de 8 bits podrá dividir el rango de la señal de entrada en 256 valores (28).

Figura 29. **Estructura de proceso de señal analógica en PLC**



Fuente: elaboración propia.

- Circuitos lógicos de salida: es el receptor de la información enviada por la CPU.

- Aislación: cumple la misma función que en las interfaces de entrada.

- Indicador de estado: también tiene la misma función que en la entrada.

- Circuitos de conexión: está compuesto por el elemento de salida al campo que maneja la carga conectada por el usuario. Existen tres tipos de circuitos de conexión que se describirán más adelante.

- Protección: son internas al PLC y pueden ser fusibles en serie con los contactos de salida, alguna protección electrónica por sobrecarga o algún circuito RC. Recordar que en caso de que más de una salida use un solo borne de referencia, es éste el que lleva asociada la protección. Por lo cual si esta protección actúa dejarán de funcionar todas las salidas asociadas a ese borne común.

3.3. Controlador lógico programable S7-1200 Siemens

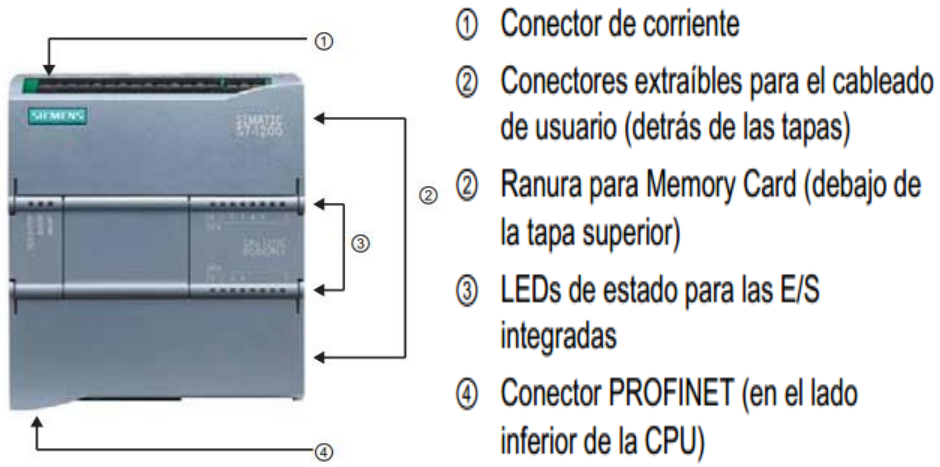
En este capítulo se dará a conocer el controlador lógico programable S7-1200 para fines didácticos, tipos de protocolos que soporta, funcionamiento básico, módulos externos entre otras características.

Este controlador es básico en la gama de Siemens en la parte de automatización industrial, puede controlar múltiples tareas con su juego de instrucciones, una de las ventajas de elegir este controlador fue por su fuente de alimentación integrada, un microprocesador integrado a la carcasa, cambio de salidas según entradas, lógica Booleana, instrucciones de contaje,

temporizadores y módulos de expansión. Tiene todo lo básico de controlador programable para poder realizar trabajo demostrativo.

El S7-1200 posee una gran ventaja para el proyecto práctico, cuenta con módulos de entradas de comunicación de red RS485 y RS232. Esto facilitará la comunicación modbus más adelante. A continuación se colocará una imagen física del controlador con sus partes básicas.

Figura 30. **PLC básico S7-1200**



Fuente: Manual Siemens p.11 Consulta: 15 de junio de 2018.

A continuación se hará mención de algunas características básicas de s7-1200, para su correcto funcionamiento en este trabajo de graduación.

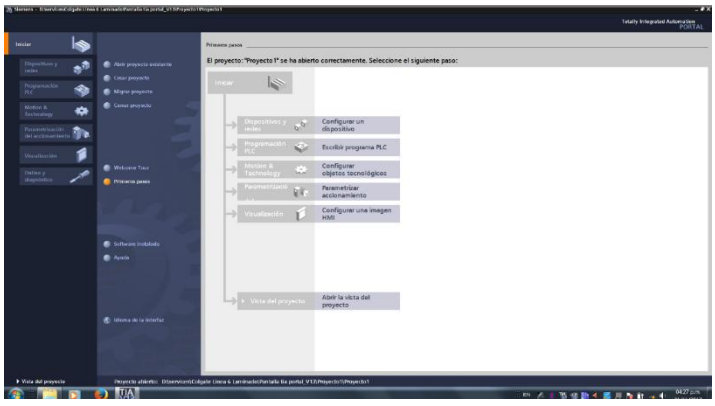
- Memoria de trabajo 25KB – 50KB.
- Memoria de carga 1 MB – 2MB.
- Memoria remanente 2KB.
- 6 entradas digitales y 4 salidas digitales.
- 2 entradas analógicas.
- 8 ampliación de módulos.

- 3 módulos de comunicación de lado izquierdo.
- Velocidad de ejecución de funciones matemáticas con números reales 18 microsegundos / instrucción.
- Velocidad de ejecución Booleana 0,1 microsegundos / instrucción.

También cuenta con un software llamado STEP 7 Basic, con el cual se harán todas las modificaciones posibles en el controlador para llegar a nuestro objetivo, es amigable con el usuario, para desarrollar y editar la lógica requerida en este trabajo, este software está atado Windows, por su puesto se necesita licencia para poder instalar dicho programa, en este caso se usará un PG s7 que traer todas las licencias requeridas.

Ya instalado el software, tendrá una presentación gráfica, dentro de este trabajo de investigación se llamará TIA PORTAL. A continuación se mostrará una captura de pantalla de dicho software.

Figura 31. **Tia portal**



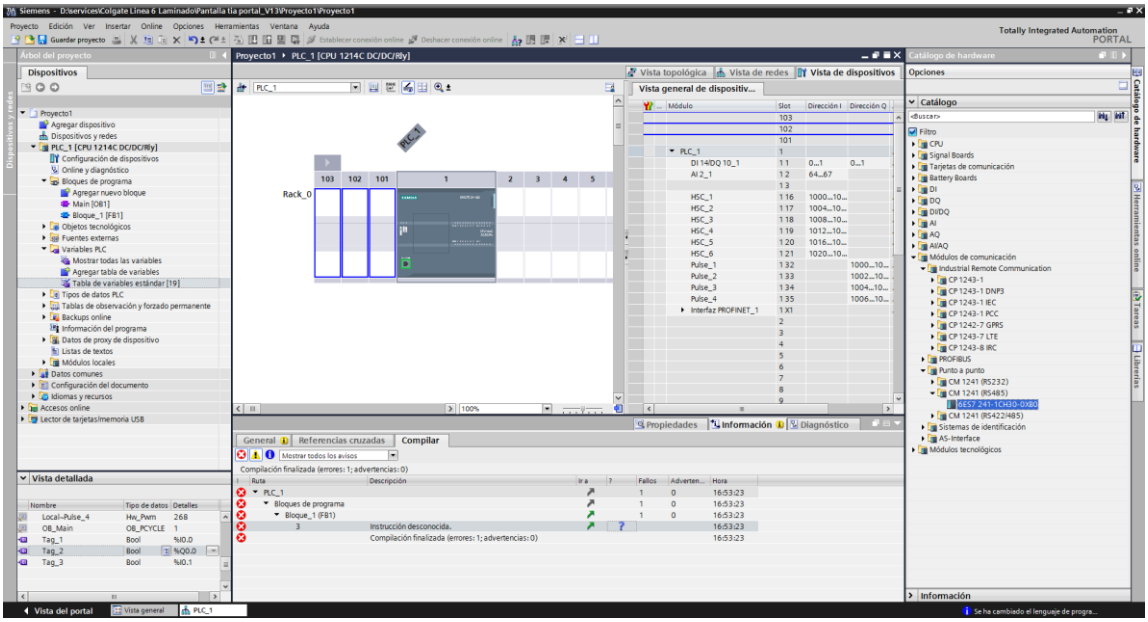
Fuente: elaboración propia.

Este software tiene dos vistas principales que son la del proyecto y una parte descriptiva del proyecto, en la parte de proyectos el usuario puede elegir

lo que desee configurar ya sea un PLC como este caso u otros productos de la marca como pantallas, medidores de energía, entre otros.

A continuación se dará a conocer como es la vista en el área de proyecto de tia portal.

Figura 32. **Tia portal área de trabajo**



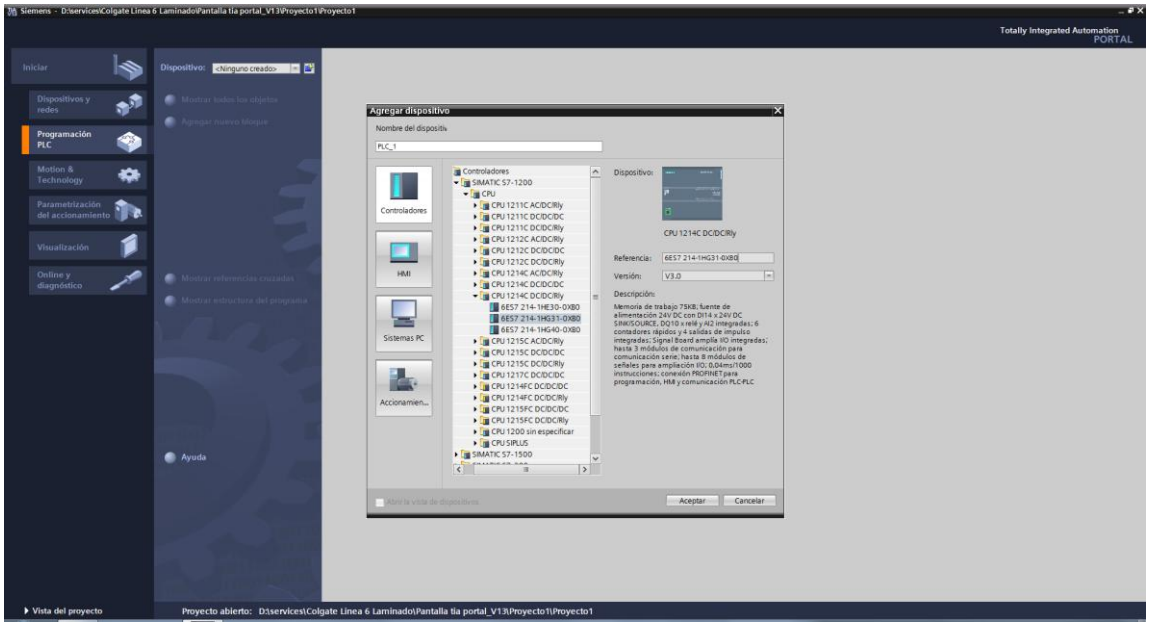
Fuente: elaboración propia, empleando software Tia portal.

Como se puede observar en la imagen XXX trata de mostrar en forma digital el elemento físico para sus configuraciones deseadas, más adelante se hablará más detalladamente sobre ello. Hay que recordar que este software posee parámetros para funcionar en una computadora convencional como por ejemplo tiene que ser mínimo debe tener un procesador mínimo i5.

3.3.1. Insertar una CPU a Tia portal

Para agregar un dispositivo al proyecto se necesitan varios pasos, los cuales se resumirán a continuación de manera gráfica y descriptiva.

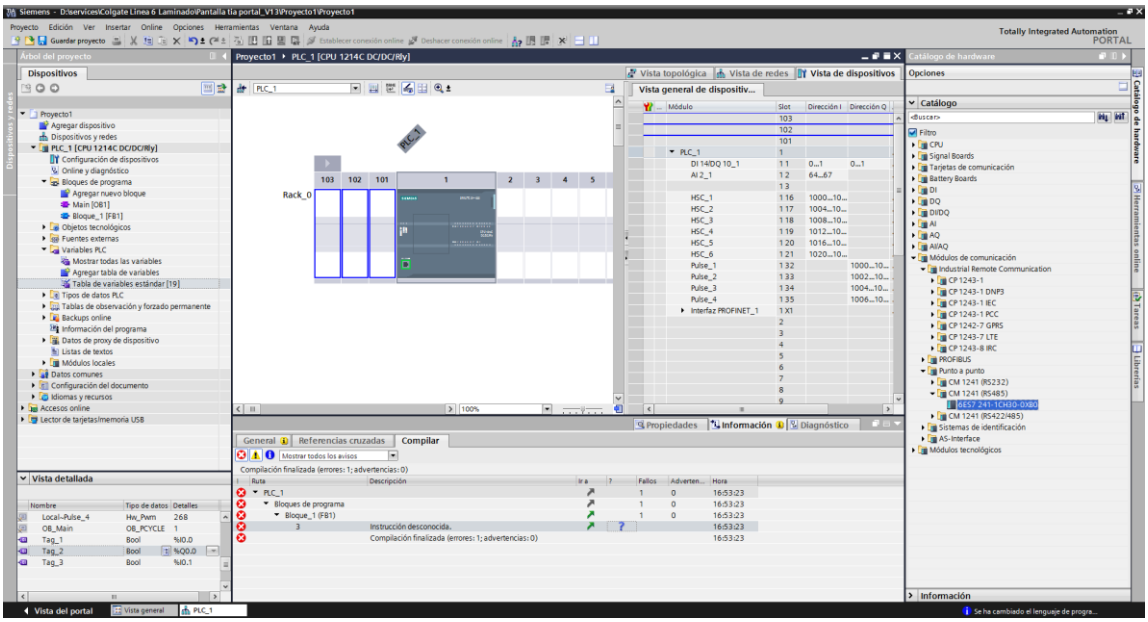
Figura 33. **Tia portal CPU´s**



Fuente: elaboración propia, empleando software Tia portal.

Como primer paso se tiene que elegir la CPU a trabajar / luego agregar dispositivo, en este caso será s7-1200, hay varios modelos, por lo tanto seleccionar el mejor se adapte a sus necesidades, se usará 3 ya que es el *firmware* de actualización, se presiona aceptar y nos muestra la siguiente pantalla de trabajo:

Figura 34. **Selección CPU´s Tia portal**



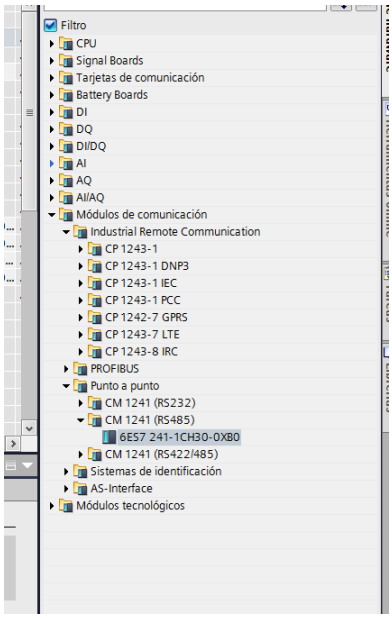
Fuente: elaboración propia, empleando software Tia portal.

Como se puede observar se tiene el dispositivo añadido a la área de proyectos y es de forma sencilla para comenzar a configurar y programar el equipo, como se mencionó anteriormente, tia portal está diseñada para que sea amigable con el usuario de manera eficiente.

3.3.2. Agregar módulos de comunicación a S7-1200 Tia portal

En esta sección se explicará como agregar una módulo de expansión a *software* Tia portal, dentro de Siemens llaman a todo módulo de comunicación-CM y SM-los módulos de señal, por lo tanto, solo se usará los CM por fines didácticos y además son los que se usará para hacer el trabajo de investigación, hay que recordar que el S7-1200 solo tiene módulo de comunicación CM RS485 y CM RS232 a la CPU, estos módulos se conectan al lado izquierdo, los del lado derecho son módulos de señales SM.

Figura 35. **Menú de CM´s Tia portal**

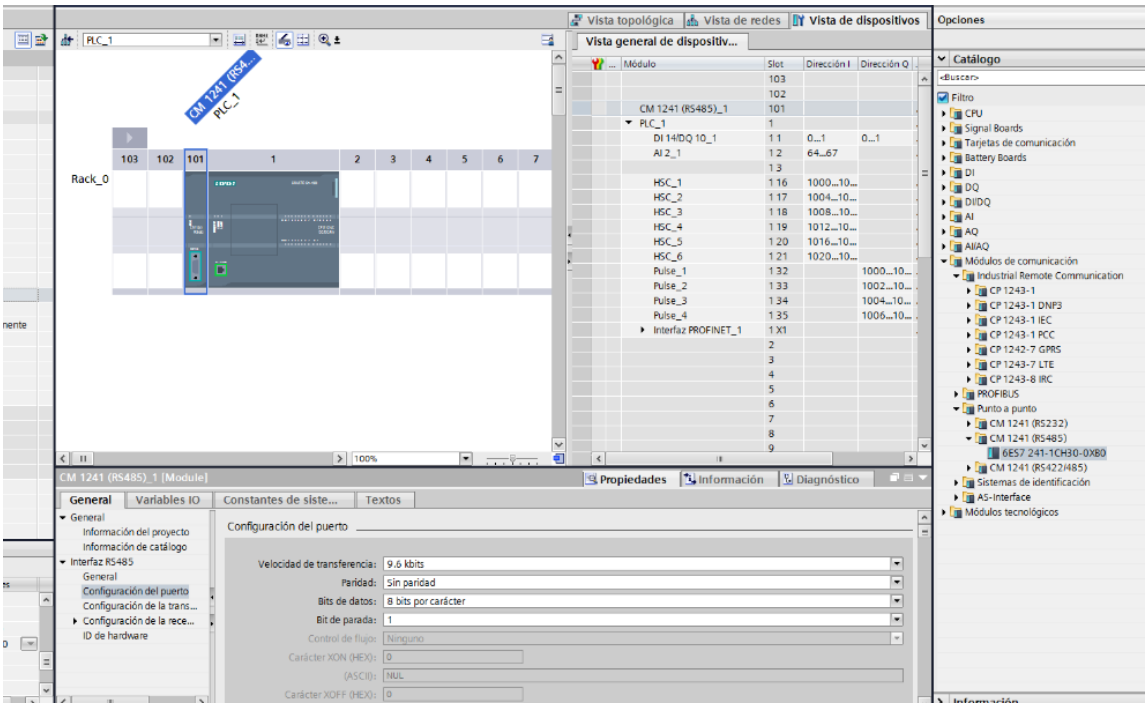


Fuente: elaboración propia, empleando software Tia portal.

Como siguiente paso se va a la parte derecha de la pantalla de Tia portal, luego a catálogo y se presiona módulos de comunicación, posteriormente se selecciona la carpeta punto a punto y por último CM 1241 (RS485).

Al tener ubicada la carpeta donde se encuentra el módulo RS485, se procede a dar le click sostenido al ítem y arrastrarlo a la posición deseada en el *rack* donde se encuentra el controlador S7-1200. Y se obtendrá la instalación del módulo en el proyecto.

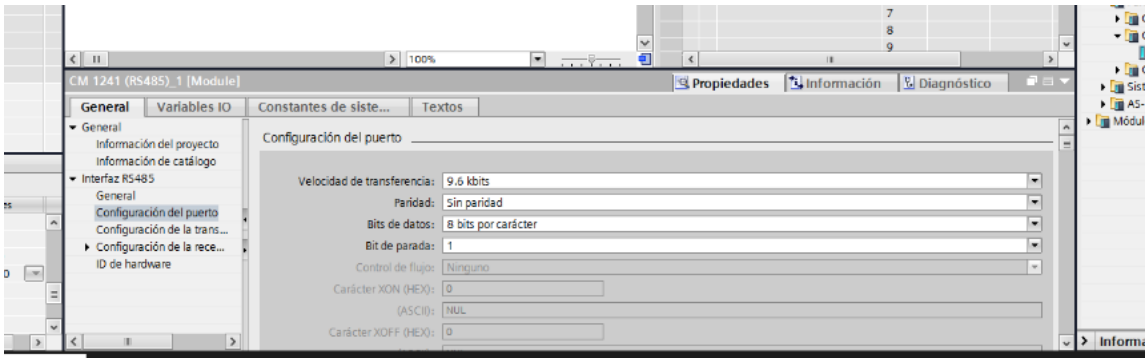
Figura 36. **Área de trabajo Tia portal**



Fuente: elaboración propia, empleando software Tia portal.

Al seleccionar el módulo RS485 se tendrán que hacer unas configuraciones básicas para realizar el trabajo de investigación, todas estas configuraciones aparecen en la parte inferior de la sección de los módulos.

Figura 37. **Configuraciones básicas PLC**



Fuente: elaboración propia, empleando software Tia portal.

Como por ejemplo velocidad de transferencia, paridad, bits de datos , bit de parada, entre otros. Estos ítems servirán más adelante para la configuración y programación de nuestro plc S7-1200.

3.3.3. Ethernet / tcp/ip PLC S7-1200 (profinet)

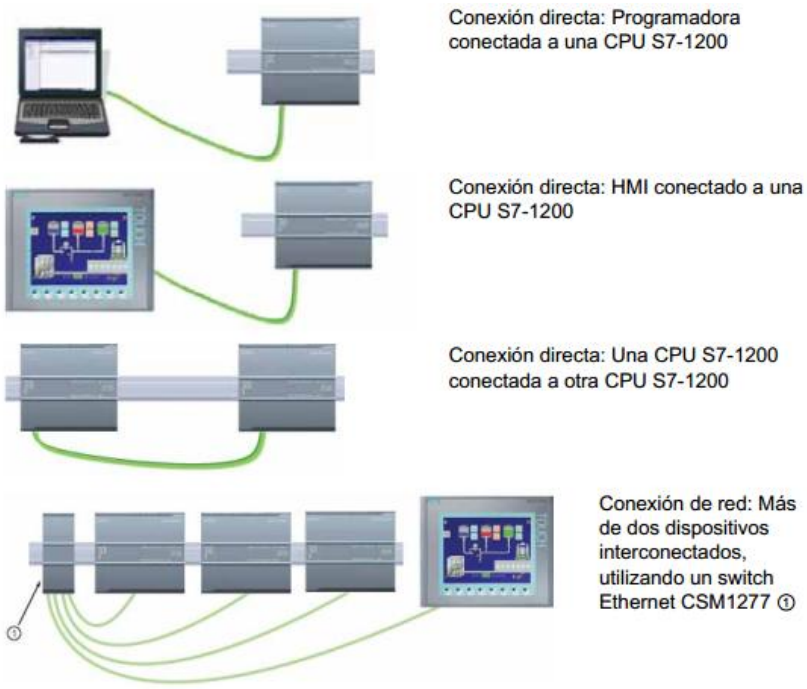
Internamente del S7-1200 se incorpora un puerto profinet el cual es compatible con las normas ethernet y de comunicación tcp/ip, usualmente será el conector RJ-45 para hablar en forma general.

Al tener profinet, el PLC se vuelve más ventajoso refiriéndose a la parte de comunicación, una de ellas es que utiliza el conector RJ-45, conector universal y de fácil uso, teniendo tcp el dispositivo puede comunicarse con otros dispositivos que no sean de la misma marca, basta con que hablen el mismo lenguaje . Se encuentran dos maneras de comunicación vía profinet:

- Conexión directa: esta conexión se refiere cuando se conectan y comunican el dispositivo donde se controla la programación y una sola CPU.
- Conexión de red: esta conexión hace referencia al conectarse más de dos dispositivos.

En la figura 38 se observa unos ejemplos gráficos.

Figura 38. **Conexiones nivel hardware PLC**



Fuente: Manual S7-1200 Siemens p. 241.

Para la comunicación entre el dispositivo programador y una CPU se debe considerar lo siguiente:

- La programadora tenga entrada Ethernet.
- Un cable RJ-45 en buen estado.
- No se requiere interlocutor entre los dos puntos ya que solo es una CPU.
- En este caso tener instalado el software Tia portal si fuese una PC.
- Y la configuración de software para comunicación con el hardware, más adelante se observa cómo hacerlo.

Para la conexión física entre la PC o programadora y la CPU se utiliza el cable ethernet anteriormente mencionado, no importando si es recto o cruzado ya que la CPU tiene le opción auto-crossover, el cual reconoce cómo está configurado el cable. En la figura 39 se mostrará gráficamente en donde se encuentra la entrada profinet.

Figura 39. **Hardware conexión profinet**

① Puerto PROFINET

Hay una descarga de tracción opcional disponible para reforzar la conexión PROFINET.

Fuente: Manual Siemens S7-1200 p. 243.

Ahora bien, hay que recordar que el fin de la comunicación profinet / tcp-ip es que se utilizará todos los beneficios de este conector universal, así como sus propias estatutos de este, como por ejemplo, al tratar de comunicar ambos dispositivos se tiene que tomar en cuenta que se encuentren en la misma LAN,

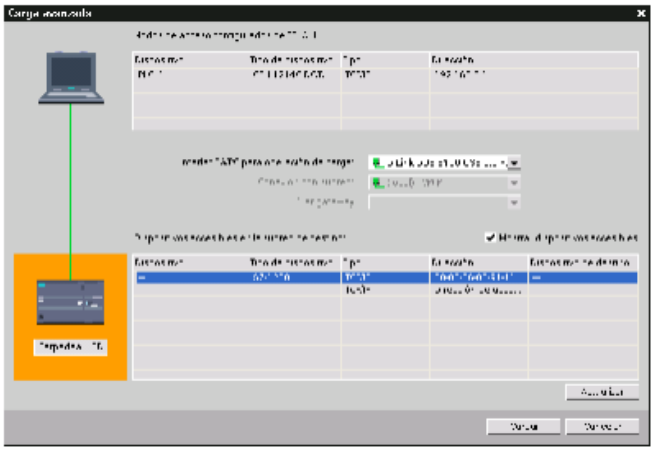
que tenga dirección IP única y que posean la misma máscara, lo que se quiere decir es que hay que saber un poco de redes básicas para configurar este dispositivo. A continuación se verá una imagen de como configurar la IP y máscara de red del dispositivo PLC, para ello se selecciona en la ventana de propiedades, dirección ethernet y se observará la ventana para configurarlo. Hay otras opciones como, seleccionar una ip *online* pero no se verá en este trabajo de investigación, también existe una opción en la cual se puede comprobar la conexión del dispositivo PLC con la programadora, en esta caso será un PC , por lo tanto se deberá entrar en la pc en el cmd y verificar si podemos llegar al plc por medio de ping´s.

Figura 40. **Configuración IP PLC**

Fuente: elaboración propia, empleando software Tia portal.

Una manera de comprar la configuración correcta de las configuraciones, es cargar el proyecto en la CPU, al hacer esto todas las direcciones IP son reconocidas por la CPU y mostrará todo en verde, para dar a entender que la conexión es un éxito. A continuación se observará la figura 41.

Figura 41. **Comprobación de conexión nivel software de un PLC**



Fuente: Manual S7-1200 Siemens p. 251.

Figura 42. **Menú comprobación de conexión**

① La segunda de dos redes Ethernet de esta programadora

② Dirección IP de la única CPU S7-1200 de esta red Ethernet

Fuente: Manual Siemens S7-1200 p. 252.

3.3.4. Configuraciones generales s7-1200

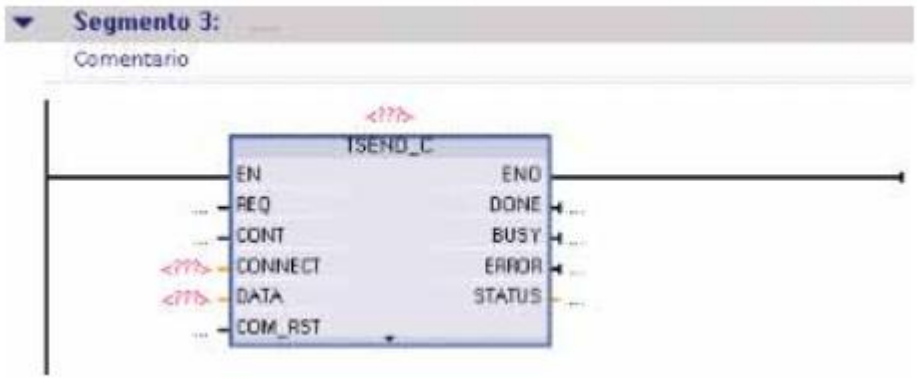
Después de configurar las partes del hardware y colocar las partes correctas adecuadamente se entrará un poco al lo que es el software, se describirán algunas configuraciones generales, así como algunas instrucciones y parámetros.

Se tratará de la comunicación de bloque de transferencia, más adelante se detallará sobre este bloque, el funcionamiento del plc se da por transmitir y recibir información, es decir *full*-dúplex,para que profinet funcione adecuadamente, habrá que configurar estos parámetros ya sean para destino o para algo local. A continuación se explicarán algunas instrucciones como ejemplo

.

Instrucción TSEND_C: crea una conexión con un interlocutor. La conexión se configura, establece y vigila automáticamente hasta que la instrucción ordene que sea desconectada. La instrucción TSEND_C combina las funciones de las instrucciones TCON, TDISCON y TSEND.

Figura 43. **Bloque de instrucción PLC S7-1200**



Fuente: Manual Siemens S7-1200 p. 258.

Los parámetros de comunicación se configuran en el diálogo propiedades de la instrucción TSEND_C. Este diálogo aparece en el lado inferior de la página cuando se ha seleccionado alguna parte de la instrucción TSEND_C.

El plc es capaz de reconocer el profinet estándar eso quiere decir que será universal y fácil de entender, los protocolos ethernet soportados por este son RFC 1006 y tcp.

ISO *on* tcp (RFC 1006):ISO *on* tcp es un mecanismo que permite portar aplicaciones ISO a la red tcp/ip. Este protocolo tiene las características siguientes:

- Protocolo de comunicación eficiente vinculado estrechamente al hardware.
- Adecuado para cantidades de datos medianas y grandes (hasta 8192 bytes).
- A diferencia de tcp, los mensajes tienen un indicador de fin y están orientados a los mensajes.
- Apto para *routing*, puede utilizarse en WAN.
- Las longitudes de datos dinámicas son posibles.
- Es necesario programar la gestión de datos debido a la interfaz de programación *SEND* / *RECEIVE*.

Transport control protocol (tcp): tcp es un protocolo estándar descrito por RFC 793: *transmission control protocol*. El objetivo principal de tcp es ofrecer un servicio de conexión seguro y fiable entre pares de procesos. Este protocolo tiene las características siguientes:

- Protocolo de comunicación eficiente puesto que está vinculado estrechamente al hardware.
- Adecuado para cantidades de datos medianas y grandes (hasta 8192 *bytes*).
- Ofrece numerosas prestaciones más a las aplicaciones, en particular:
  - Recuperación de errores
  - Control de flujo
  - Fiabilidad
- Protocolo orientado a la conexión.
- Puede utilizarse muy flexiblemente con sistemas de terceros que soporten únicamente tcp.
- Apto para *routing.*
- Las aplicaciones se direccionan usando números de puerto.
- La mayoría de los protocolos de aplicación (p. ej. TELNET y FTP) utilizan TCP.
- Es necesario programar la gestión de datos debido a la interfaz de programación *SEND / RECEIVE*.

A continuación se verá las configuraciones generales.

Tabla IV. **Tabla de configuraciones generales de un PLC**

Parámetro	Definición
General	
Punto final: Interlocutor	Nombre asignado a la CPU interlocutora (receptora)
Interfaz	Nombre asignado a las interfaces
Subred	Nombre asignado a las subredes
Dirección	Direcciones IP asignadas
Tipo de conexión	Tipo de protocolo Ethernet
ID de conexión	Número de ID
Datos de conexión	Ubicación de almacenamiento de datos de las CPUs local e interlocutora
Establecimiento de conexión activo	Botón de opción para seleccionar la CPU local o interlocutora como conexión activa
Detalles de dirección	
Puerto (decimal)	Puerto de la CPU interlocutora en formato decimal

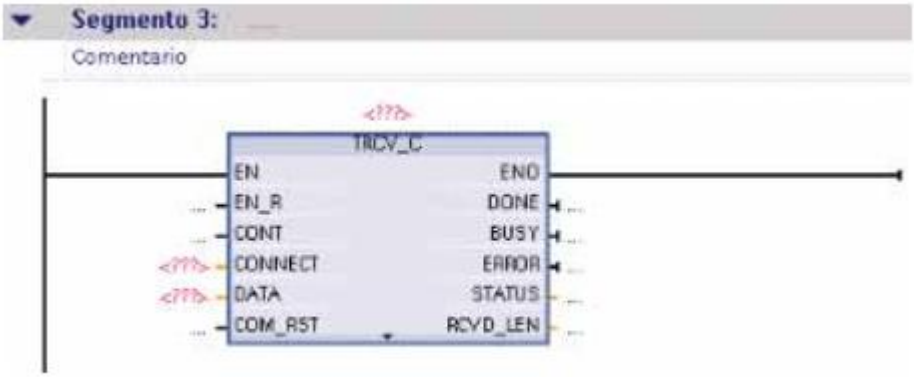
Fuente: Manual S7-1200 p. 261.

Ahora para las instrucciones de recepción tener como ejemplo TRCV_C.

Instrucción TRCV_C : crea una conexión con un interlocutor. La conexión se configura, establece y vigila automáticamente hasta que la instrucción ordene que sea desconectada. La instrucción TRCV_C combina las funciones de las instrucciones TCON,TDISCON y TRCV.

Desde la configuración de la CPU en STEP 7 *basic* es posible configurar cómo la instrucción TRCV_C debe recibir los datos. Para comenzar inserte la instrucción en el programa desde la carpeta Comunicación en las Instrucciones avanzadas. La instrucción se visualizará junto con el diálogo Opciones de llamada en el que se asigna un DB para almacenar los parámetros de la instrucción TRCV_C.

Figura 44. **Bloque de instrucción TRCV_C**



Fuente: Manual Siemens S7-1200 p. 262.

Los parámetros de comunicación se configuran en el diálogo propiedades de la instrucción TRCV_C. Este diálogo aparece en el lado inferior de la página cuando se ha seleccionado alguna parte de la instrucción TRCV_C.

En la tabla V se mostrará las configuraciones generales de esta instrucción.

Tabla V. **Configuraciones generales de instrucción TRCV_C**

Parámetro	Definición
General	
Punto final: Interlocutor	Nombre asignado a la CPU interlocutora (receptora)
Interfaz	Nombre asignado a las interfaces
Subred	Nombre asignado a las subredes
Dirección	Direcciones IP asignadas
Tipo de conexión	Tipo de protocolo Ethernet
ID de conexión	Número de ID
Datos de conexión	Ubicación de almacenamiento de datos de las CPUs local e interlocutora
Establecimiento de conexión activo	Botón de opción para seleccionar la CPU local o interlocutora como conexión activa
Detalles de dirección	
TSAP[1] (ASCII)	TSAPs de las CPUs local e interlocutora en formato ASCII
ID TSAP	TSAPs de las CPUs local e interlocutora en formato hexadecimal

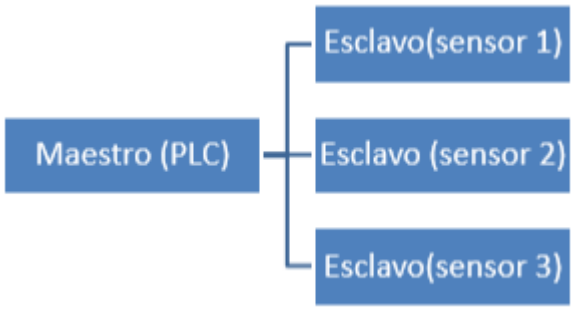
Fuente: Manual Siemens S7-1200 p. 264.

3.3.5. Protocolo USS

El protocolo de universal de interface en serie (USS) se utiliza en temas de accionamientos, un ejemplo muy utilizado es el acoplamiento en serio de una jerarquía entre un maestro y varios sistemas esclavos, en otras palabras permite la comunicación de un punta a llamado maestro a un punto B llamado esclavo, ya que no es posible una comunicación directa en sistemas esclavos, para este tipo de comunicación se utiliza la tecnología semi-dúplex, solo uno puede trasmitir o recibir, No simultáneamente, a continuación se mostrará un diagrama de bloques sencillo para una mejor entendimiento de maestro-esclavo.

Esta librería contiene 1 FB y 3 FC, los cuales soportan USS, un módulo CM 1241 RS 485 soporta como máximo 16 accionamientos. Cuando están configurados en una red USS el almacenamiento es temporal por cada PTP instalado.

Figura 45. **Ejemplo de PLC maestro con varios esclavos**



Fuente: elaboración propia.

Instrucciones, USS_DRV, USS_PORT, USS_RPM y USS_WPM, servirán para controlar todos los accionamientos de la red.

A continuación se resume un poco sobre cada instrucción de accionamiento.

USS_DVR: intercambia datos con el accionamiento creando peticiones e interpretando las respuestas del accionamiento. Para cada accionamiento debe utilizarse un bloque de función propio. No obstante, todas las funciones USS asociadas con una red USS y el módulo de comunicación ptp deben utilizar el mismo bloque de datos instancia. Es preciso crear el nombre del DB cuando se inserta la primera instrucción USS_DRV. Este DB creado al insertar la instrucción por primera vez se reutiliza posteriormente.[25]

USS_PORT: la instrucción USS_PORT gestiona la comunicación en la red USS. Generalmente, el programa contiene sólo una función USS_PORT por cada módulo de comunicación ptp. Cada llamada de esta función gestiona una transferencia hacia o desde un accionamiento. El programa debe ejecutar la función USS_PORT con suficiente frecuencia para impedir *timeouts* del accionamiento. Todas las funciones USS asociadas a una red USS y a un módulo de comunicación ptp deben utilizar el mismo bloque de datos instancia. USS_PORT se llama generalmente desde un OB de alarma de retardo para impedir *time outs* del accionamiento y para que las actualizaciones de datos USS más recientes estén disponibles para las llamadas de USS_DRV.

Cuando la instrucción USS_DRV se ejecuta por primera vez, el accionamiento que indica la dirección USS (parámetro DRIVE) se inicializa en el DB instancia. Después de esta inicialización, las ejecuciones siguientes de USS_PORT pueden iniciar la comunicación con el accionamiento .Si se

modifica el número del accionamiento, el PLC debe cambiar de *stop* a *run* con objeto de inicializar el DB instancia. Los parámetros de entrada se configuran en el búfer de mensajes USS TX y las salidas se leen de un búfer de respuesta válido anterior (si existe). Durante la ejecución de USS_DRV no se transmiten datos. Los accionamientos se comunican cuando se ejecuta USS_PORT. USS_DRV configura únicamente los mensajes que deben enviarse e interpreta los datos que puedan haberse recibido de una petición anterior.

USS_WPM: la instrucción USS_WPM modifica un parámetro en el accionamiento. Todas las funciones USS asociadas a una red USS y a un módulo de comunicación ptp deben utilizar el mismo bloque de datos. USS_WPM debe llamarse desde el OB principal.

MB_COMM_LOAD: la instrucción MB_COMM_LOAD configura un puerto del módulo de comunicación punto a punto (ptp) CM1241 RS485 o CM 1241 RS232 para la comunicación vía el protocolo modbus rtu.

Anteriormente se menciona el estado maestro – esclavo a continuación se menciona algunas reglas para que se cumplan siempre en modbus.

- MB_COMM_LOAD debe ejecutarse para configurar un puerto antes de que la instrucción MB_MASTER pueda comunicarse con ese puerto.
- Si un puerto debe utilizarse para iniciar peticiones de maestro modbus, MB_SLAVE nopodrá utilizar este puerto. Una o más instancias de ejecución de MB_MASTER pueden utilizarse en este puerto.
- Las instrucciones modbus no utilizan eventos de alarma de comunicación para controlar el proceso de comunicación. El programa debe consultar la instrucción MB_MASTER para transmitir y recibir condiciones completas.

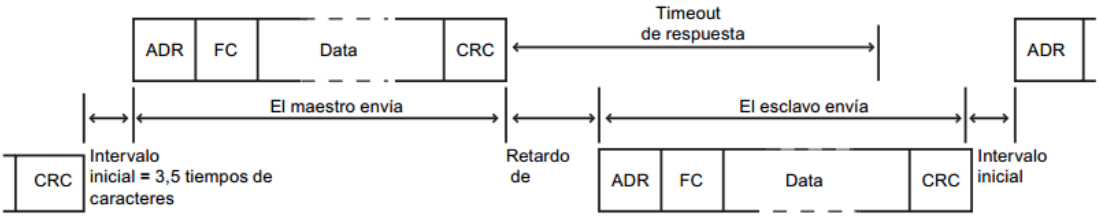
- Si el programa opera un maestro modbus y utiliza MB_MASTER para enviar una petición a un esclavo, MB_MASTER se deberá seguir ejecutando hasta que se devuelva la respuesta del esclavo.
- Todas las ejecuciones de MB_MASTER para un determinado puerto deben llamarse desde un mismo OB (o clase de prioridad de OB).

Así pues el esclavo también posee reglas a seguir como las siguiente.

-
- MB_COMM_LOAD debe ejecutarse para configurar un puerto antes de que la instrucción MB_SLAVE pueda comunicarse con ese puerto.
- Si un puerto debe responder como esclavo a un maestro modbus, MB_MASTER no podrá utilizar este puerto. Sólo se puede utilizar una instancia de MB_SLAVE en un determinado puerto.
- Las instrucciones modbus no utilizan eventos de alarma de comunicación para controlar el proceso de comunicación. El programa debe controlar el proceso de comunicación consultando la instrucción MB_SLAVE para comprobar si se han finalizado las operaciones de transmisión y recepción.
- La instrucción MB_SLAVE debe ejecutarse periódicamente a una frecuencia que permita responder sin demora a las peticiones entrantes de un maestro modbus.
- MB_SLAVE se debería llamar en cada ciclo desde un OB de ciclo.

A continuación se dará conocer en forma gráfica los pasos para la interacción maestro-esclavo.

Figura 46. **Gráfica de interacción maestro-esclavo**



Fuente: Manual Siemens S7-1200 p. 233.

3.3.6. Otros protocolos

Entre otros protocolos del S7-1200 se tiene profibus, DNP3 y MPI. A continuación se hará un breve resumen de estos protocolo para que el lector sepa de sus existencias dentro del controlador lógico programable.

3.3.6.1. Profibus

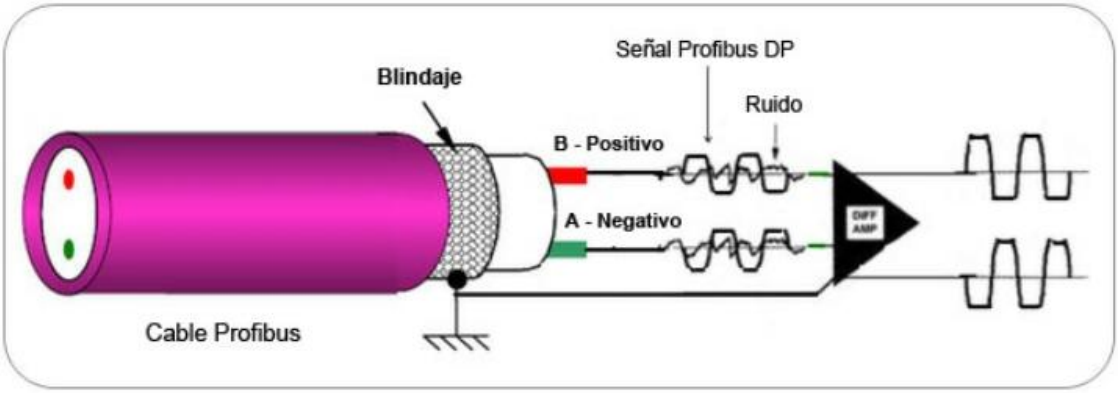
Este protocolo es *open source*, es decir es libre y no pertenece a ninguna compañía, es simple de utilizar y tiene las mayores instalaciones en toda Europa, el tipo de comunicación establece que es de bus serie. Se utilización permite que sea un sistema multimaestro y permita el funcionamiento de varios sistemas esclavos en el área de automatización. En la actualidad existen 3 tipos:

- DP: automatización en fabricas, plug & play, eficiente y económico.
- FMS: gran variedad de aplicaciones de campo, comunicación multimaestro, se utiliza más para propósitos generales.
- PA: alimentación por bus, seguridad intrínseca, usado en automatización de procesos.

La instalación de este tipo de comunicación es adaptable tanto como controladores PLC, componentes de red, *drivers motors*, sensores, actuadores, puerta de enlaces, entre otros, posee más de 800 dispositivos compatibles en todo la industria. Su velocidad estándar es de 31,25 Kbps pero puede llegar hasta 12 Mbps, esto es muy ventajoso si se habla solamente de dos cables entre dispositivos, ya que ninguno es el dueño de este protocolo un gran porcentaje en el mundo de la automatización es compatible, esto también hace que sea muy seguro al hablar de seguridad transmisión, también puede usar fibra óptica como medio de comunicación. También este protocolo se vasa en la ISO 9506 que es un estándar en el modelo OSI, su transmisión máxima es de 1200 m. Prácticamente está orientada a dispositivos de campo.

En la figura 47 se mostrará una imagen física del conector.

Figura 47. **Cable profibus**



Fuente: http://www.smar.com/blog-profibus. Consulta 20 de junio 2017.

3.3.6.2. Protocolo DNP3

Distributed network protocol versión 3 (DNP3) este protocolo fue creado para componentes de vida longeva y transmisiones en la cual la recepción no sea muy confiable, también se sabe que este protocolo es *open source* (es gratis) para las compañías norteamericanas, una de las ventajas de este protocolo que no es sensible a las EMF. También otorga diversos modos de salidas, transferencia segura de archivos, direccionamiento sobre 65000 dispositivos en enlaces simples, sincronización de tiempo y eventos de estampado de tiempos, entre otras ventajas.

Este protocolo está basado en IEC 870-5 por lo tanto, es normal ver lo en aplicaciones de SCADA, incluyendo la comunicación RTU y IED, maestro y esclavo, punto a punto y aplicaciones en la red.

En la tabla VI se mostrará una imagen del protocolo DNP3 respecto modelo OSI.

Tabla VI. **Comparación de protocolos OSI y DNP3**

Modelo OSI	DNP3
Capa de Aplicación	Capa de Aplicación
Capa de Presentación	-
Capa de Sesión	-
Capa de Transporte	Función de Transporte
Capa de Red	-
Capa de Enlace de Datos	Capa de Enlace de Datos
Capa Física	Capa Física (medio)

Fuente: http://jupiter.utm.mx/~tesis_dig/11905.pdf. Consulta: 20 de junio de 2017.

3.3.6.3. Protocolo MPI

Multi-point-interface (MPI) es el protocolo más sencillo de comunicar y configurar entre las gamas s7 de siemens, a pesar que no se requieren tarjetas adicionales como los otros protocolos eso la hace más económica pero insegura hablando informáticamente, este protocolo es muy parecido a modbus pero la ventaja de este es que guarda la información al dejar de funcionar por un corte no deseado y después de vuelve a su estado restablecido.

La interface MPI sólo permite que una PC o PG, los cuales son dispositivos de programación de las estaciones que integran la red, tenga el acceso a la vez. Las direcciones MPI de la red se disponen de la siguiente manera:

Tabla VII. **Direccionamientos MPI**

Dirección	**Dispositivo**	**Descripción**
0	op	Panel de operaciones
1	Pc / PG	Dispositivo de programación
2 hasta el 31	cpu	PLC

Fuente: elaboración propia.

Las direcciones 0 y 1 están reservadas como direcciones fijas y no se programan, sin embargo, si se conectan dos PC de manera simultánea en la red, esta falla, a menos que de manera directa se configure la dirección de la otra PC en la dirección 0.

Cuando se habla de la misma marca siemens tiene ciertas ventajas en estos protocolos, una de ellas sería la comunicación en CPU a través de una solo interface MPI sin necesidad de programación.

A pesar de tener estas ventajas también posee varios contras, una de ellas que sólo es compatible con la marca siemens, esto no lo hace favorable al usuario ya que lo limita de opciones en su red, otra desventaja en número de equipos conectadas a la red, ya que son 32 participantes, pero solamente 5 CPU pueden intercambiar datos por ejemplo.

Las CPU´s S7 tienen integradas en el sistema operativo funciones de comunicación simples como la comunicación por GD. De esta forma una CPU puede intercambiar datos con otras CPU´s a través del interface MPI, sin necesidad de programación.

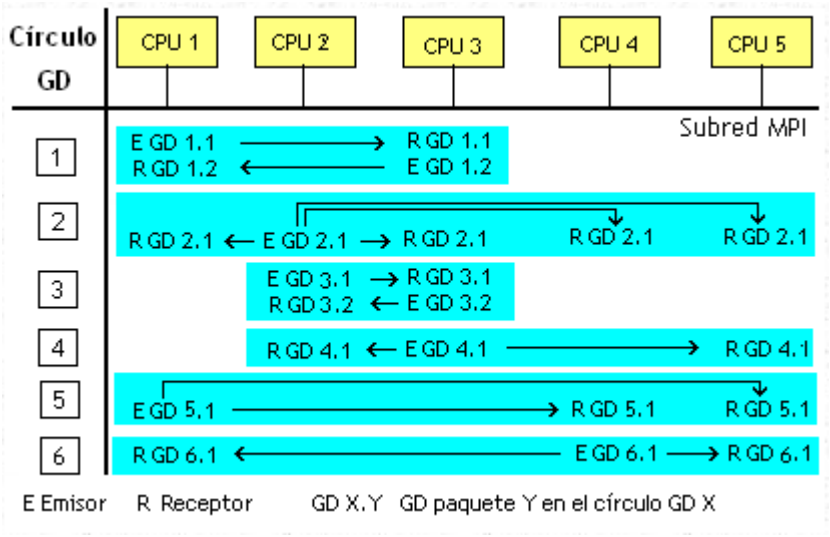
Se entiende bajo datos globales (comunes) las entradas, salidas, marcas y áreas en bloques de datos que se intercambian vía MPI entre dos o más CPU´s S7-300/400.

Opcionalmente es posible definir un factor de ciclo que especifica tras cuántos ciclos de programa deben emitirse / recibirse los datos. Por ejemplo para un factor de 8 (SR 8), la transferencia de datos toma lugar cada 8 ciclos del procesador. El factor puede ser de 1 a 255 (dependiendo del CPU).

Todos los datos globales que van del mismo emisor al mismo receptor se agrupan en un paquete GD. Este se envía dentro de un telegrama. El paquete GD se identifica por un número de paquete GD. Si se supera la longitud máxima de un paquete GD de emisión, se utiliza un nuevo círculo GD.

Las CPU´s que participan en el intercambio de datos de un determinado paquete GD forman un círculo GD. Si en una subred MPI hay otras CPU´s que intercambian otros paquetes GD, forman un segundo círculo GD. Varios círculos pueden acceder a la misma CPU, es decir, tiene la capacidad de pertenecer a varios círculos GD, como se muestra en la figura 18.

Figura 48. **Gráfica de intercambio de datos CPU**



Fuente: VASQUEZ LOPEZ, Virgilio http://homepage.cem.itesm.mx/vlopez/mpi.htm. Consulta: 20 de junio de 2017.

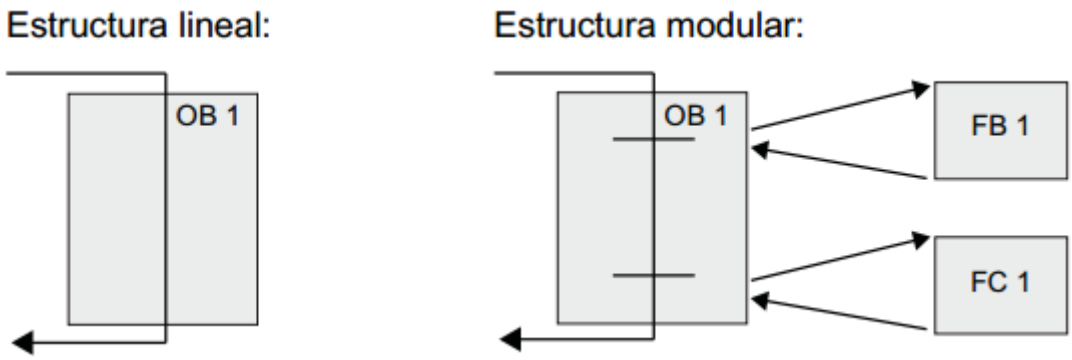
3.3.7. Principios básicos sobre estructura de programación

A continuación se mostrará el tipo de jerarquía, tipos de bloques, como interactúan entre ellos, directrices de la marca SIEMENS, agregar valores iníciales y lenguajes de programación. Estos conceptos serán la base para comenzar el trabajo de investigación utilizando un S7-1200, será el principio para hacer la estructura de programación del software del PLC como en la imaginación para hacer futuros proyectos.

En la estructura del programa se definirán por bloques como OB, FB y FC. que serán los bloques bases en los cuales se basará toda la programación, se dará un explicación breve sobre el funcionamiento de estos bloques y cómo usarlos para las necesidades.

Como primer paso se tiene que elegir si se quiere que el programa sea línea o modular, el línea como su nombre lo indica ejecuta todas las instrucciones una de siguiente de la otra de forma ordenada. Y el modular llama bloques específicos para ejecutarlas y hacer tareas determinadas, la manera de cómo trabaja este tipo es que cada bloque tiene tareas subordinadas, la estructura de este programa se formando cuando desde un bloque lógico llamamos a otro bloque.

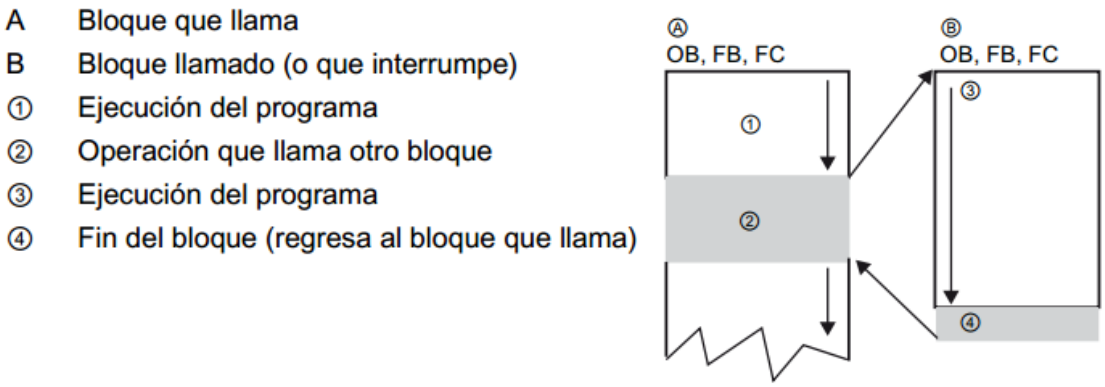
Figura 49. **Estructuras de programación PLC**



Fuente: Manual S7-1200 Siemens p. 89.

Al querer diseñar FB (bloque de función) y FC (una función) que ejecuten tareas específicas se crean programas modulares, y cuando un bloque llama a otro bloque estos pueden ser reutilizados en el programa, a continuación se mostrará una imagen para una mejor interpretación sobre la interacción sobre los bloques.

Figura 50. **Algoritmo de ejecución de una instrucción PLC**

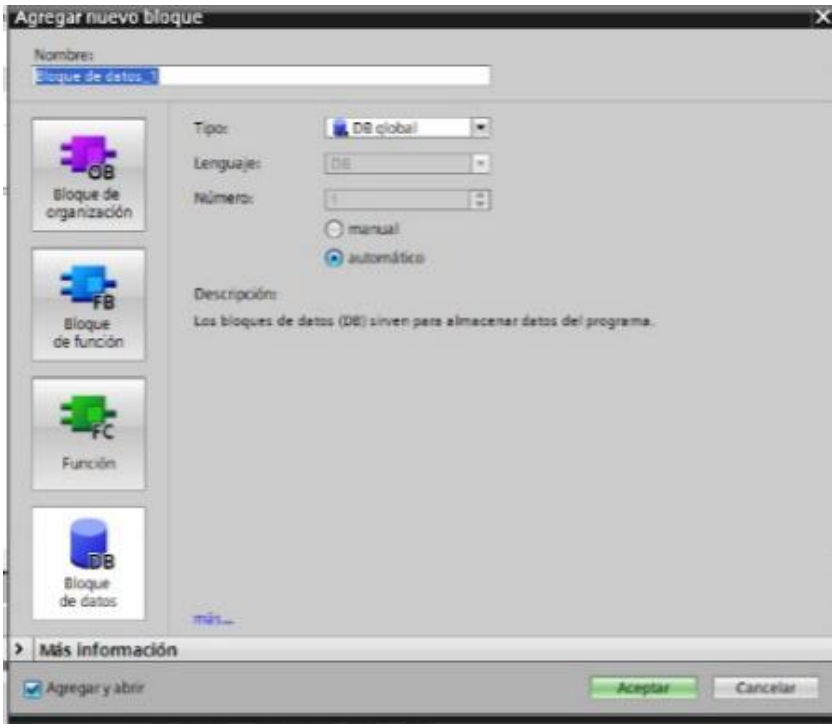


Fuente: Manual Siemens S7-1200 p. 90.

En otras palabras cuando un bloque llama a otro bloque, el bloque el cual es llamado, la CPU ejecuta el programa en el bloque llamado y al finalizar, reanuda la ejecución del bloque quien hizo la llamada, de esta manera es que ejecuta nuestro controlador lógico programable.

Para agregar bloques al programa por Tia portal (visto anteriormente) se puede ingresar directamente agregar de programa y nos aparecerá una pantalla que describirá cada bloque, y se tendrá que elegir el que se encaje a las necesidades. A continuación se observa en la figura 51.

Figura 51. **Tipos de bloques Tia portal**



Fuente: elaboración propia, empleando software Tia portal.

3.3.7.1. Bloques de organización (OB)

Como su nombre lo indica son los que organizan los demás bloques, estos serán los responsables de toda la estructura inicial del nuestro programa, este bloque es el intermediario entre el sistema operativo y el programa de usuario. El OB de ciclo contiene el programa principal. Es posible incluir más de un OB de ciclo en el programa de usuario. En estado operativo RUN, los OBs de ciclo se ejecutan en el nivel de prioridad más bajo y pueden ser interrumpidos por todos los demás tipos de procesamiento del programa. El OB de arranque no interrumpe el OB de ciclo, puesto que la CPU ejecuta el OB de arranque antes de pasar al estado operativo RUN.

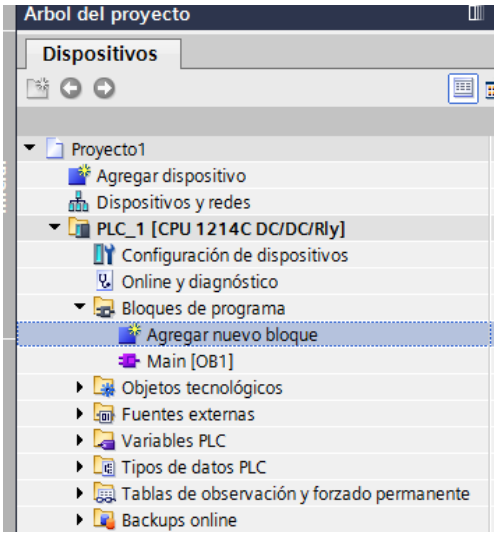
Tras finalizar el procesamiento de los OBs de ciclo, la CPU vuelve a ejecutarlo inmediatamente. Esta ejecución cíclica es el tipo de procesamiento normal que se utiliza para los controladores lógicos programables. En numerosas aplicaciones, el programa de usuario entero está contenido en un solo OB de ciclo. Es posible crear otros OBs para ejecutar funciones específicas, tales como tareas de arranque, procesamiento de alarmas y tratamiento de errores o ejecución de un código de programa específico en determinados intervalos. Estos OBs interrumpen la ejecución de los OBs de ciclo .A continuación se observa en la figura 52.

Figura 52. **Bloque de organización**

Fuente: Manual Siemens S7-1200 p. 92.

A continuación se mostrará una imagen de donde se seleccionó originalmente esta ventana.

Figura 53. **Menú de creación de un bloque**



Fuente: elaboración propia, empleando software Tia portal.

3.3.7.2. Función (FB)

Una función (FC) es un bloque lógico que por lo general realiza una operación específica en un conjunto de valores de entrada. La FC almacena los resultados de esta operación en posiciones de memoria.

Las FCs se utilizan para realizar las tareas siguientes:

- Para ejecutar operaciones estándar y reutilizables, en cálculos matemáticos.

- Para ejecutar funciones tecnológicas, por ejemplo controles individuales con operaciones lógicas binarias.

Una FC también se puede llamar varias veces en diferentes puntos de un programa. Esto facilita la programación de tareas que se repiten con frecuencia. Una FC no tiene ningún bloque de datos instancia asociado (DB). La FC usa la pila de datos locales para los datos temporales utilizados para calcular la operación. Los datos temporales no se almacenan. Para almacenar los datos de forma permanente es preciso asignar el valor de salida a una posición de memoria global, por ejemplo el área de marcas o un DB global.

3.3.7.3. Bloque de función

Un bloque de función (FB) es un bloque lógico que utiliza un bloque de datos instancia para sus parámetros y datos estáticos. Los FBs tienen una memoria variable ubicada en un bloque de datos o DB instancia. El DB instancia ofrece un bloque de memoria asociado a esa instancia (o llamada) del FB y almacena datos una vez que haya finalizado el FB. Es posible asociar distintos DBs de instancia a diferentes llamadas del FB. Los DB instancia permiten utilizar un FB genérico para controlar varios dispositivos. El programa se estructura de manera que un bloque lógico llame un FB y un DB instancia. La CPU ejecuta luego el código del programa en ese FB y almacena los parámetros del bloque y los datos locales estáticos en el DB instancia. Cuando finaliza la ejecución del FB, la CPU regresa al bloque lógico que ha llamado el FB. El DB instancia conserva los valores de esa instancia del FB. Estos valores están disponibles para las llamadas posteriores al bloque de función, bien sea en el mismo ciclo o en otros ciclos.

3.3.7.4. Bloques de datos (DB)

Los bloques de datos (DB) se crean en el programa de usuario para almacenar los datos de los bloques lógicos. Todos los bloques del programa de

usuario pueden acceder a los datos en un DB global. En cambio un DB instancia almacena los datos de un bloque de función (FB) específico. Un DB se puede definir de manera que sea de sólo lectura.

Los datos almacenados en un DB no se borran cuando finaliza la ejecución del bloque lógico asociado. Hay dos tipos de DBs, los cuales son:

- Un DB global almacena los datos de los bloques lógicos en el programa. Cualquier OB,FB o FC puede acceder a los datos en un DB global.

- Un DB instancia almacena los datos de un FB específico. La estructura de los datos en un DB instancia refleja los parámetros (*input, output e inOut*) y los datos estáticos de lFB. (La memoria temporal del FB no se almacena en el DB instancia.)

3.3.8. Seleccionar lenguaje de programación

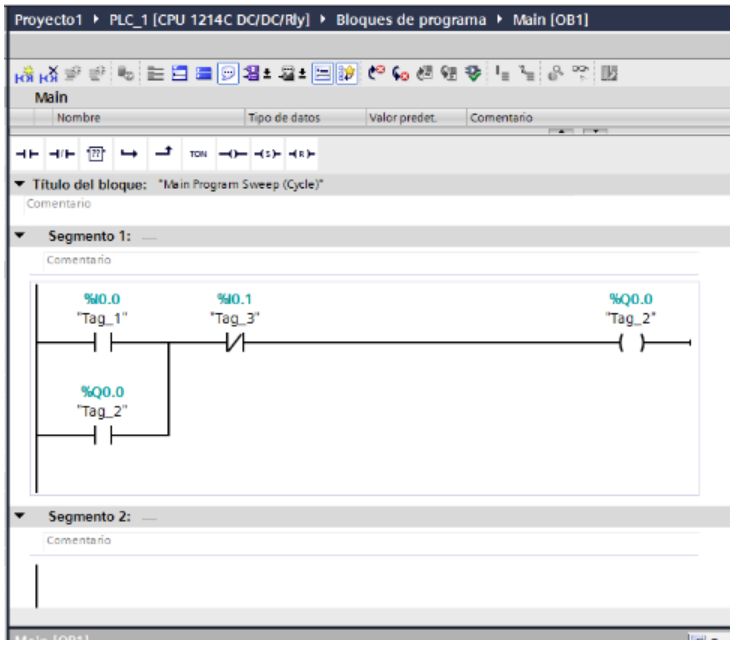
Cuando se refiere al lenguaje de programación es el mecanismo lógico que se usará para programar, es decir, cada lenguaje tiene su interpretación según diferentes tipos de simbología e interacciones visuales con el usuario. Las que soporta el PLC serán KOP, FUP Y SCL. Estos tres tipos se describirán brevemente en el trabajo de investigación.

3.3.8.1. Lenguaje de esquema de contactos (KOP)

Este lenguaje va orientada a diagramas eléctricos ya que es muy similar a los diagramas de contactos abiertos, cerrado y temporizadores. Cada rama tendrá un rol diferente en el programa y estarán conectadas a la rama raíz que será de alimentación, este lenguaje es tan versátil que proporciona funciones

como matemáticos, temporizadores, contadores y transferencias. A continuación se observa una ilustración de este lenguaje en la figura 54.

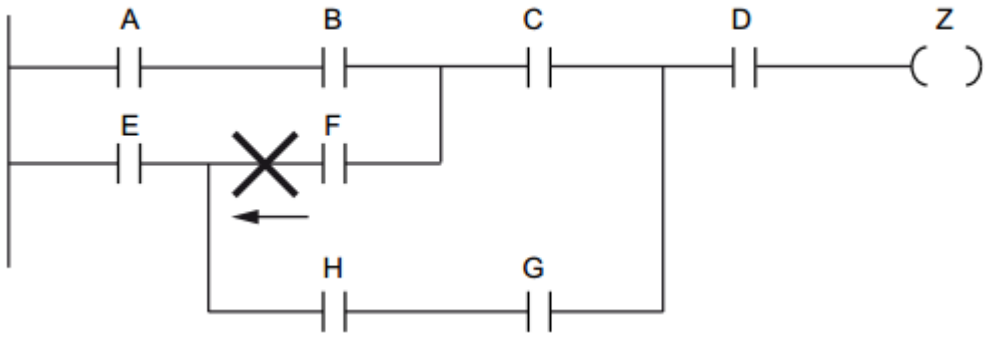
Figura 54. **Esquema KOP**



Fuente: elaboración propia, empleando software Tia portal.

Pero también tiene ciertas reglas como por ejemplo no comenzar una nueva rama dentro de otra, como se observa en la figura 55.

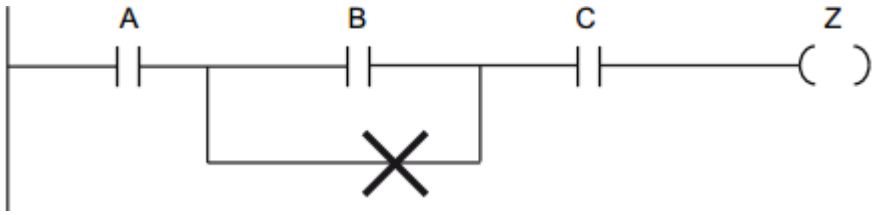
Figura 55. **Errores típicos kop 1**



Fuente: Manual Siemens S7-1200 p. 97.

O de esta manera

Figura 56. **Errores típicos kop 2**

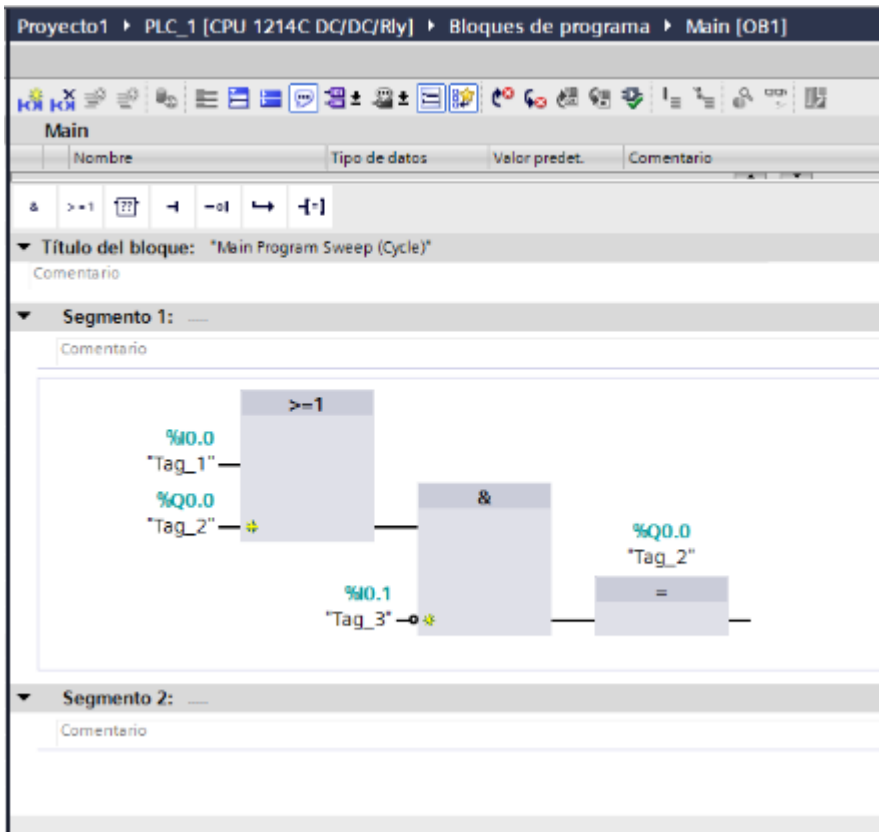


Fuente: Manual Siemens S7-1200 p. 97.

3.3.8.2. Programación de diagramas de funciones (FUP)

Este lenguaje de programación también es visual, este se caracteriza por que se utiliza la lógica de la algebra booleana, tanto como sus reglas como su simbología son perceptibles en todo el programa. Unas de las ventajas sobre este tipo de lenguaje que se puede ver la programación que sea ha hecho desde el inicio al fin, y así, es más fácil detectar un error o corrección dentro del programa. Como el objetivo de este trabajo de investigación no es sobre lenguajes no se entrará en mucho detalle, a continuación se verá un ejemplo gráfico de este lenguaje en la figura 57.

Figura 57. **Esquema FUP**

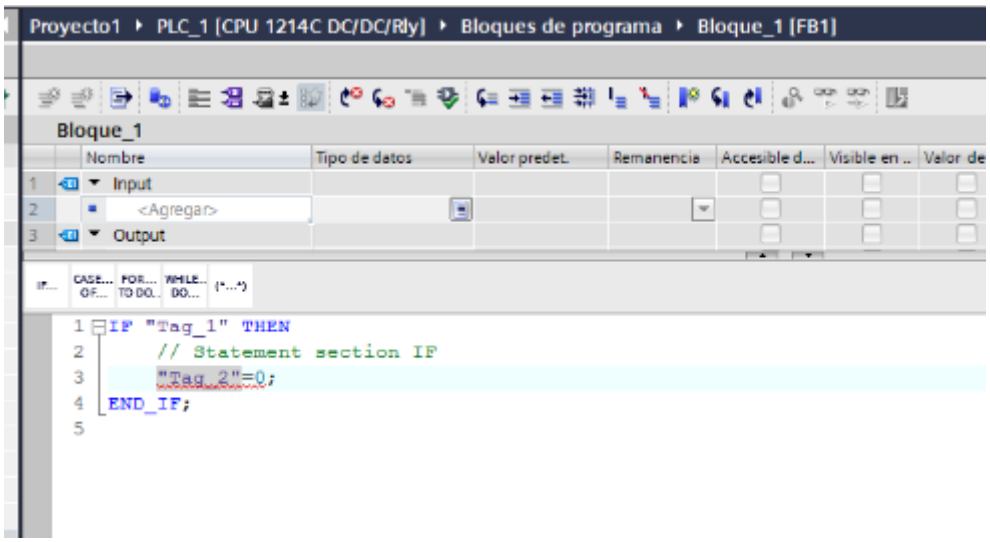


Fuente: elaboración propia, empleando software Tia portal.

3.3.8.3. Estructurado control de lenguaje (SCL)

Este tipo de lenguaje es para la programación de algo muy especificó tareas muy difíciles de concretar, programaciones que no son funcionales en todos los ámbitos, también usando este tipo de lenguaje se optimiza el rendimiento de la CPU ya que ahorra memoria. Posee algoritmos muy complejos, y algunos casos simplifican códigos. Todo su lenguaje esta basado en alto nivel de programación Pascal, este lenguaje es propio de siemens y cada año le añaden funciones nuevas, para la actualización. A continuación se observa en la figura 58.

Figura 58. **Esquemas SCL**



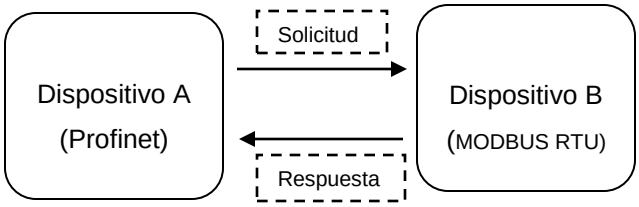
Fuente: elaboración propia, empleando software Tia portal.

4. HOMOLOGACIÓN DE PROTOCOLOS RTU Y TCP/IP

Se tomarán dos escenarios, el primero será un sistema cliente- servidor, el servidor será un dispositivo A y el cliente será un dispositivo B. En el Cual el dispositivo A será el encargado de hacer la homologación modbus-rtu a modbus-tcp-ip, ya que cuenta con la posibilidad de utilizar ambos protocolos. El dispositivo B cuenta sólo con mosbus rtu. El segundo escenario será que un dispositivo C pida información al dispositivo A en protocolo tcp/ip. Por lo tanto, la dificultad será pasar todo lo del dispositivo B-modbus rtu al dispositivo C-tcp/ip.

La estructura anterior se representará en la figura 59.

Figura 59. **Escenario 1**



Fuente: elaboración propia.

En la figura 60 se mostrará el escenario 2, en la cual el dispositivo C pedirá información del dispositivo B.

Figura 60. **Escenario 2**

Fuente: elaboración propia.

Como se observa ninguno en los dos escenarios son protocolos compatibles para que la información se traslade correctamente, por lo tanto se explicará primero el escenario 1 y posteriormente el escenario 2. Al terminar la explicación de los dos escenarios se entenderá la homologación entre los dos protocolos.

4.1. Comunicación PLC a modbus RTU S7-1200

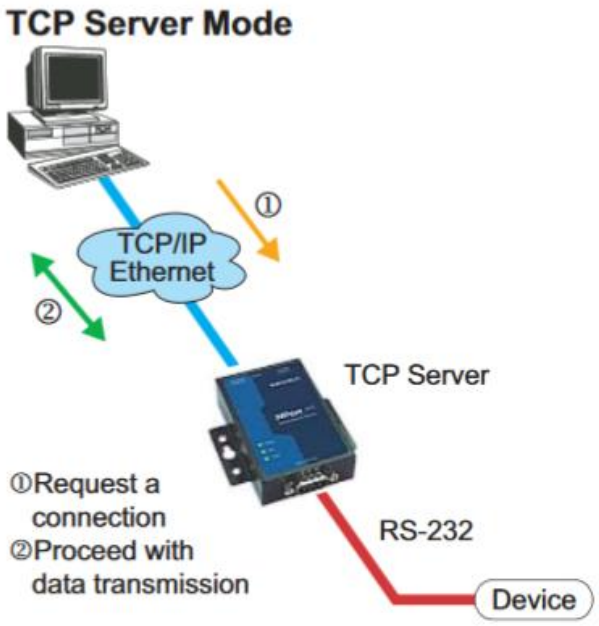
Hay que recordar que para fines didácticos se utilizó el S7-1200, la ventaja de este PLC es que posee módulo expandibles para comunicación entre otros dispositivos. El siguiente procedimiento es compatible con cualquier tipo de marca, que posean estos protocolos de comunicación. Es decir el dispositivo A puede ser Siemens, Dispositivo B mitsubishi y el dispositivo C podría ser Emerson, sólo para dar una idea.

Una solución típica en las industrias es obtener un intérprete entre los dispositivos, como por ejemplo los MOXA, estos son pequeños dispositivos que poseen dos orificios en cada lado opuesto de conexión, cada orificio corresponde a un tipo de conexión requerido dependiendo la función necesitada, la mayoría en forma de cajas de 5x5 cm. En otros casos se mandan

a manufacturar intérpretes dedicados, el problema del último es que son precios muy elevados.

En la figura 61 se mostrará una presentación gráfica de uno de los dispositivos mencionados anteriormente.

Figura 61. **Esquema funcionamiento MOXA**



Fuente: Manual Productos Moxa / https://www.moxa.com/doc/manual/nport/5100/V1.0/NPort_5100_Series_Users_Manual_v1.pdf. Consulta: 20 de junio de 2017.

Otros de los inconvenientes para algunas ocasiones es la seguridad informática, ya que estos pueden funcionar como servidores, con una IP configurada, por lo tanto las hace susceptibles a posibles ataques. También secciona la red haciéndola un poco más complicada si la red fuera muy grande.

Estos son problemas muy comunes en la industria de automatización, el costo es despreciable si solo se toma en cuenta una red pequeña de comunicación, pero al multiplicar esa pequeña red por una 100 veces, los gastos incrementan exponencialmente. Por ello viene la solución con el S7-1200.

Con el S7-1200 se puede colocar un módulo externo al PLC que no sobre pasa el 0,01 % del costo de un proyecto de automatización. Al tener instalado el módulo externo, procederá configurar lo con sus parámetros deseados, y ya se podrá comunicar el dispositivo A con el dispositivo B. Resumiendo la solución del primer escenario, dentro del PLC, se tendrá que programar y configurar de manera que pida los registros del dispositivo B, al mandar los registros en el bus, se almacenarán los daros en el PLC, y así, podrán tener comunicación dispositivo A y dispositivo B.

Para la solución del escenario 2 se procederá a guardar la información en el dispositivo A con los registros enviados del dispositivo B, y guardarlos en variables conocidas, para manipularlas en un futuro. Al configurarlas correctamente en las variables ya se pueden convertir los registros de entrada modbus rtu a registros de salida tcp/ip.

Después de resumir el procedimiento que se hará, se iniciará las soluciones reales en forma gráfica, serán unas series de pasos a seguir para realizar la solución correcta. Para ello se utilizará una PG Siemens, diseñada especialmente para el uso de proyectos reales y virtuales como Tia portal.

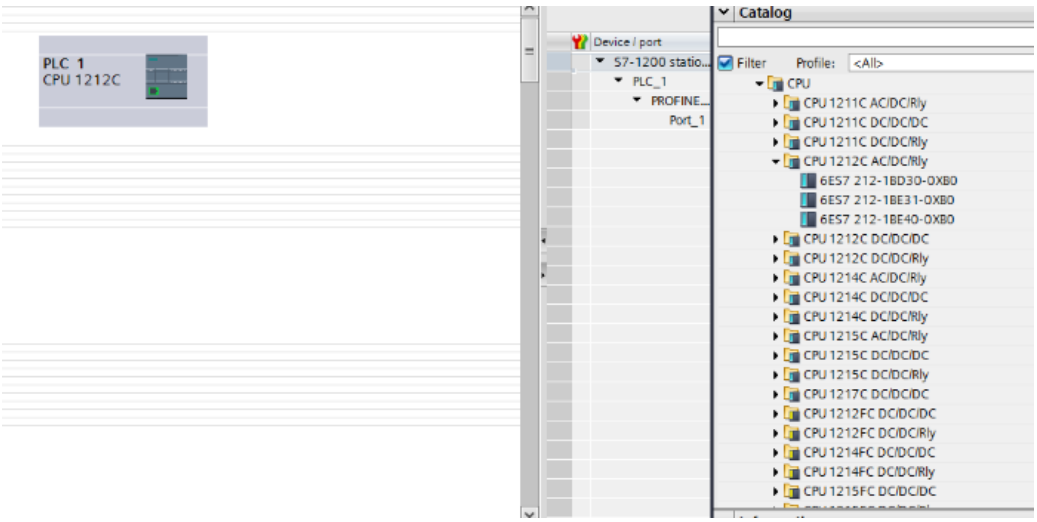
Se procederá una crear un nuevo proyecto en Tia portal, y a seleccionar la CPU requerida en este caso será una CPU S7-1200 / 6ES7 212, hay que tener en cuenta que habrán varios modelos, ya algunos son de salidas AC otras de

DC, otras que traen su fuente incluida, otras que se alimentan con corriente alterna, entre otros. Por lo tanto hay que poner mucha atención al modelo seleccionado, para que compile apropiadamente con el equipo real.

Se procederá a configurar el hardware tanto físicamente como virtualmente para hacer el trabajo de investigación.

Se seleccionará la CPU anteriormente mencionada y se arrastrará al riel de la pantalla. A continuación se mostrará gráficamente.

Figura 62. **Selección de CPU Tia portal Siemens**

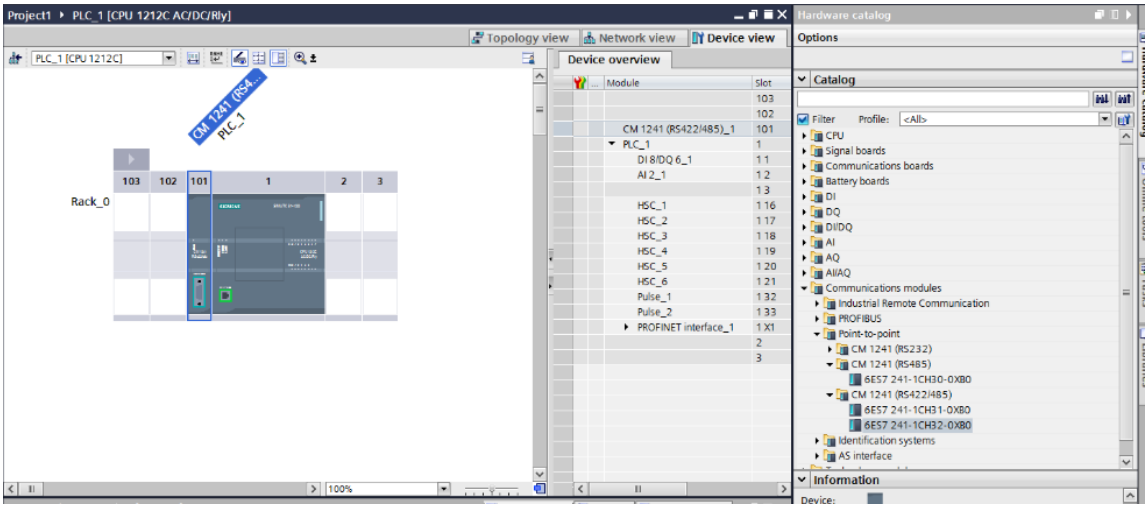


Fuente: elaboración propia, empleando software Tia portal.

Lo siguiente será la selección del módulo correcto para la interpretación de modbus rtu, en este caso se seleccionó un CM 1241 que corresponden a las entradas RS422 y RS 485, hay que recordar que CM significa módulo de comunicación, y para esta ocasión será una entrada RS 485 la que se usará

para este trabajo de investigación. En la figura 63 se mostrará la selección del ítem mencionado.

Figura 63. **Selección de módulo CM RS485 Tia portal**

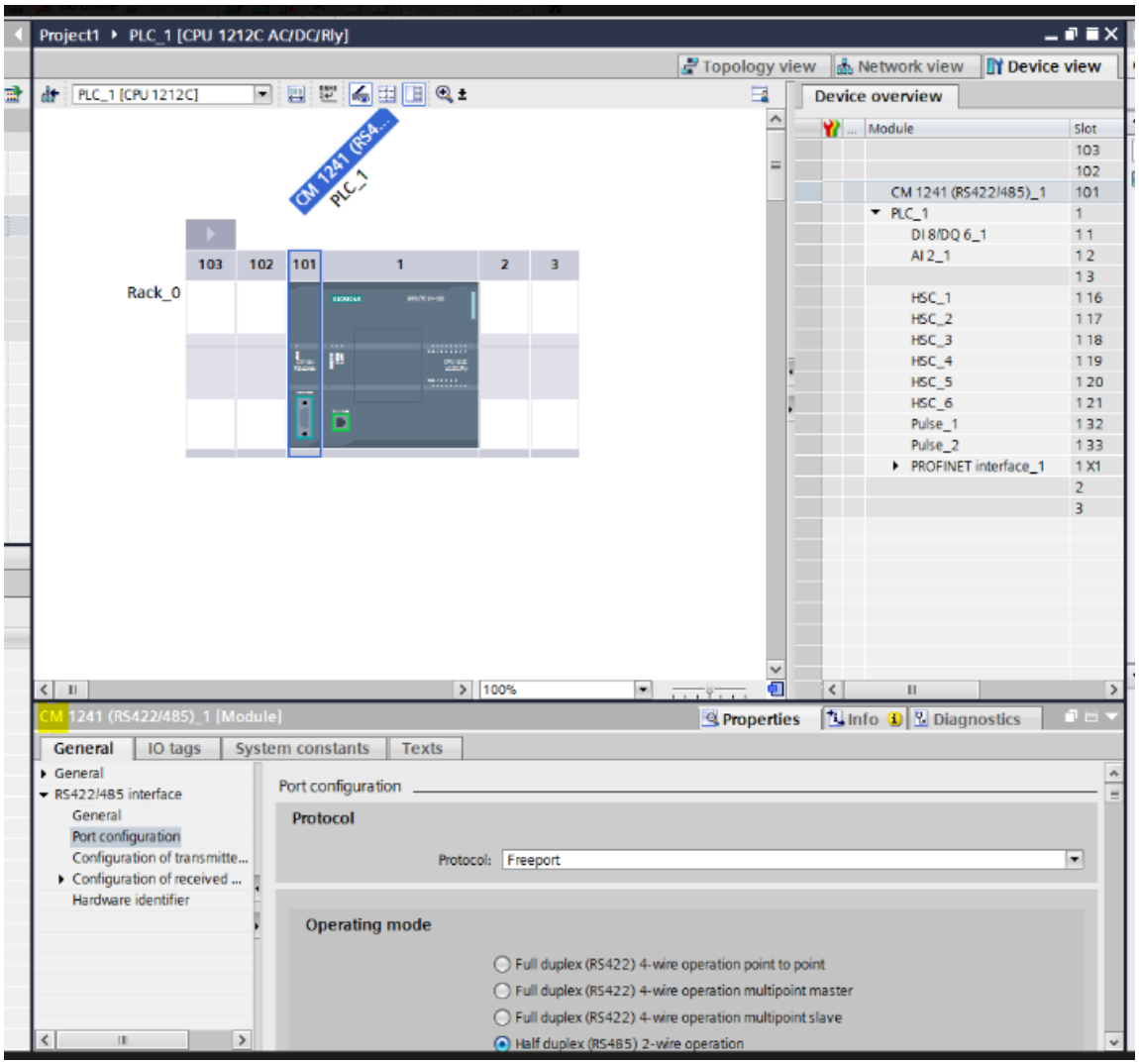


Fuente : elaboración propia, empleando software Tia portal.

A partir de este punto se trabajará en vista de *device view* para mejor interpretación del proyecto, como el siguiente paso, hay que recordar que este será el dispositivo A, como siguiente paso se configurarán los parámetros del módulo, como velocidad de transmisión, entre otros. Recordar que todo debe estar sincronizado.

Estas configuraciones aparecen cuando se selecciona el módulo con un click, en la parte inferior aparecerá un menú de configuraciones como se muestra en la figura 64.

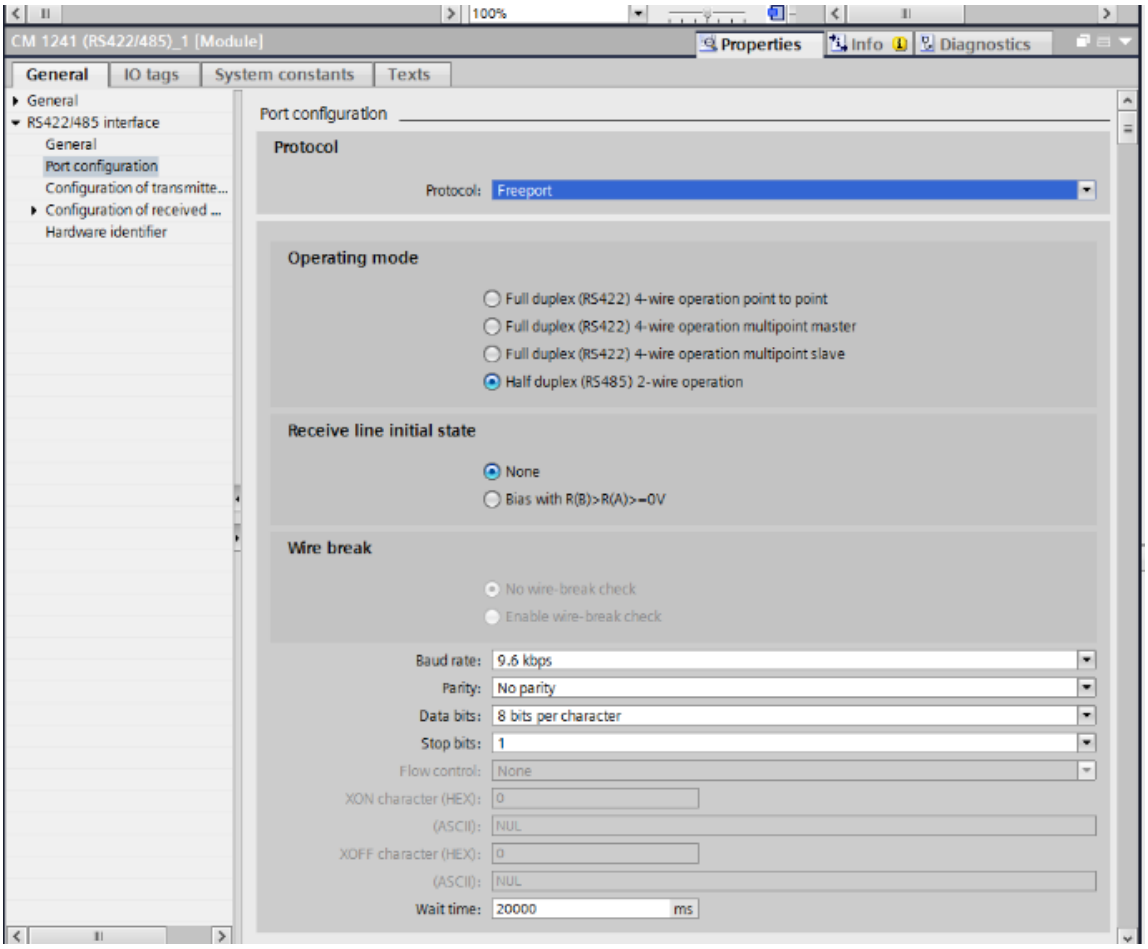
Figura 64. **Panel de configuraciones generales de un CM**



Fuente: elaboración propia, empleando software Tia portal.

Al ampliar el menú aparecerá lo siguiente.

Figura 65. **Configuración CM PLC Siemens S7-1200**



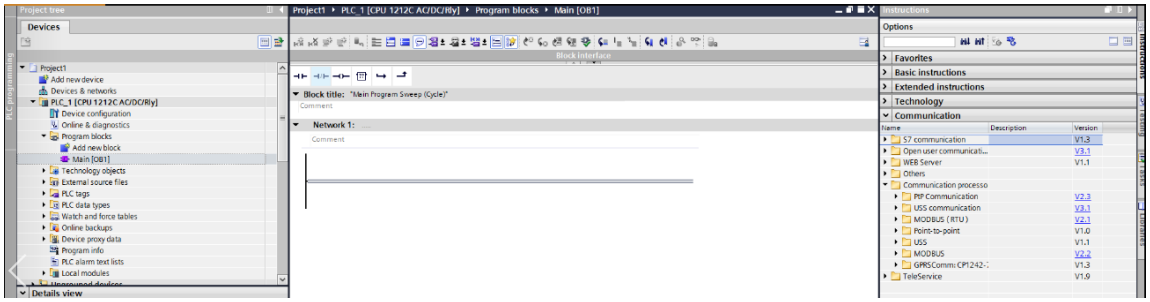
Fuente: elaboración propia, empleando software Tia portal.

Esta ventana se observa las configuraciones que se deberán seleccionar, como que tiene que ser *half-dúplex,* estados iniciales-ninguno, estará a 9,6 Kbps, no tendrá paridad, 8 *bits* per carácter *stopbits* de 1, y un tiempo de espera de 20000 ms.

Al tener todo el hardware configurado, se traslada a la parte del software, en este punto se tiene que tener las conexiones del cable a los módulos, bien colocados, la fuente correcta de alimentación y seguros.

Se procederá a crear un bloque de organización, y dirigirse a la sección de *comunication*. Se mostrará en la figura 66.

Figura 66. **Configuración de software**

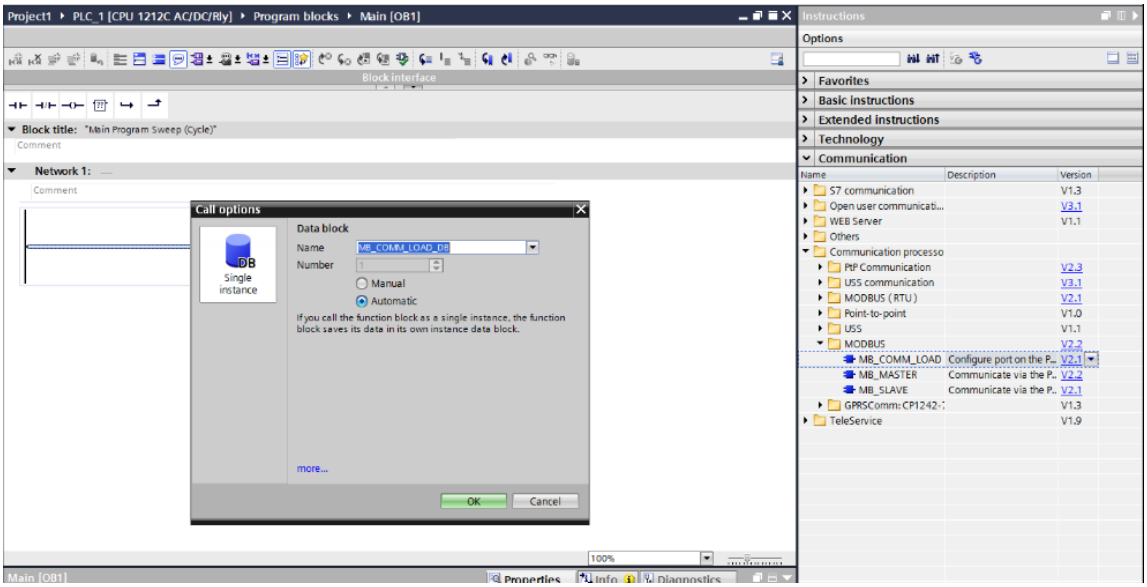


Fuente: elaboración propia, empleando software Tia portal.

Y se dirige a la carpeta *comunication* processo/modbus.

En esa sección se encontrarán 3 tipos de bloques MB_COMM_LOAD, MB_Master, MB_SLAVE. Se selecciona el primero MB_COMM con click derecho, enseñará varias opciones se selecciona nuevo, desplegará otra ventana y se elegirá bloque de datos, se seleccionará automático y finalmente la opción OK.

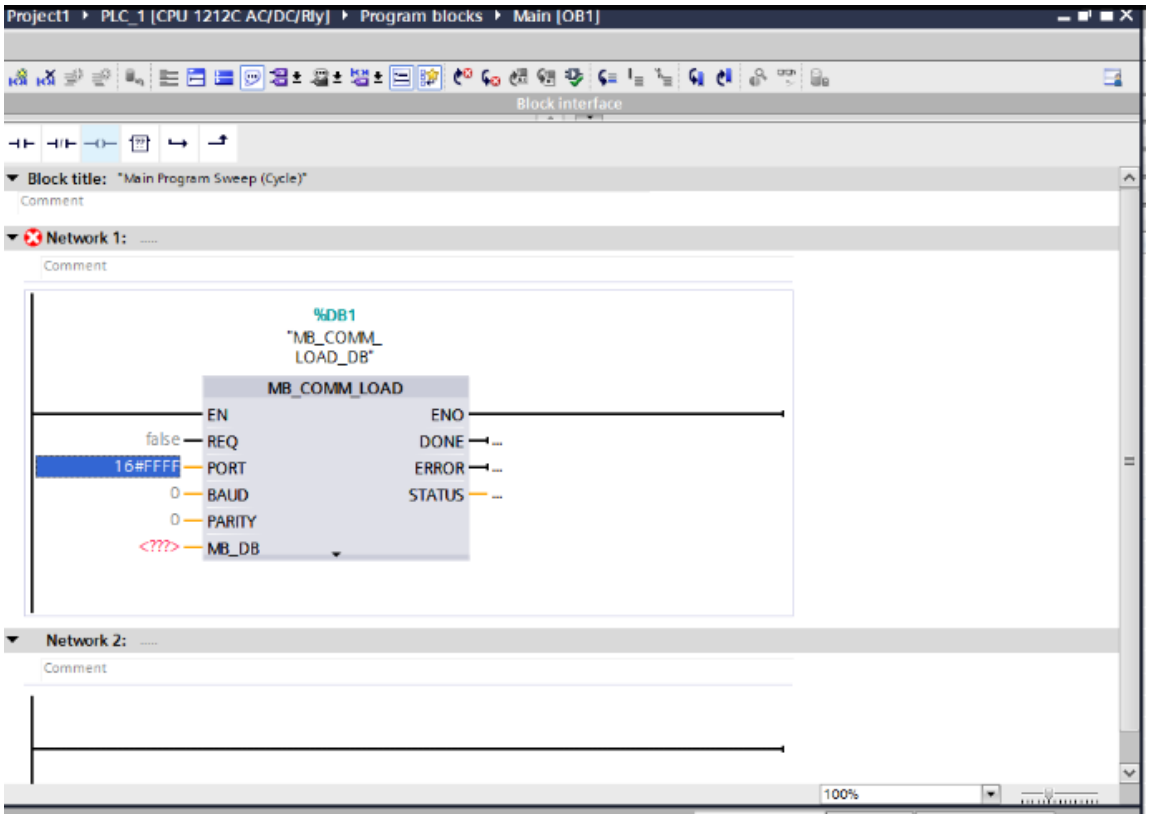
Figura 67. **Creación de bloque de datos Comm_load PLC S7-1200 Siemens**



Fuente: elaboración propia, empleando software Tia Portal.

Por defecto arrastrará un bloque a la primera rama de programación, se mostrará en la figura 68.

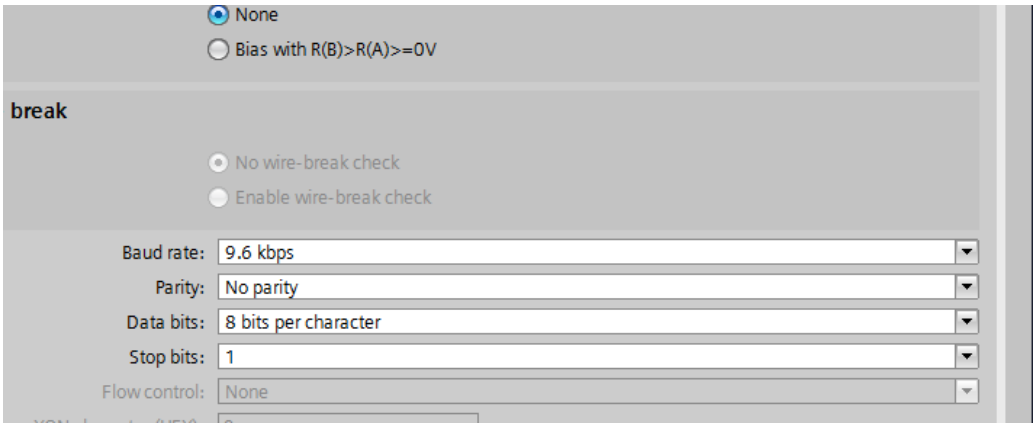
Figura 68. **Bloque de dato Tia portal S7-1200**



Fuente: elaboración propia, empleando software Tia portal.

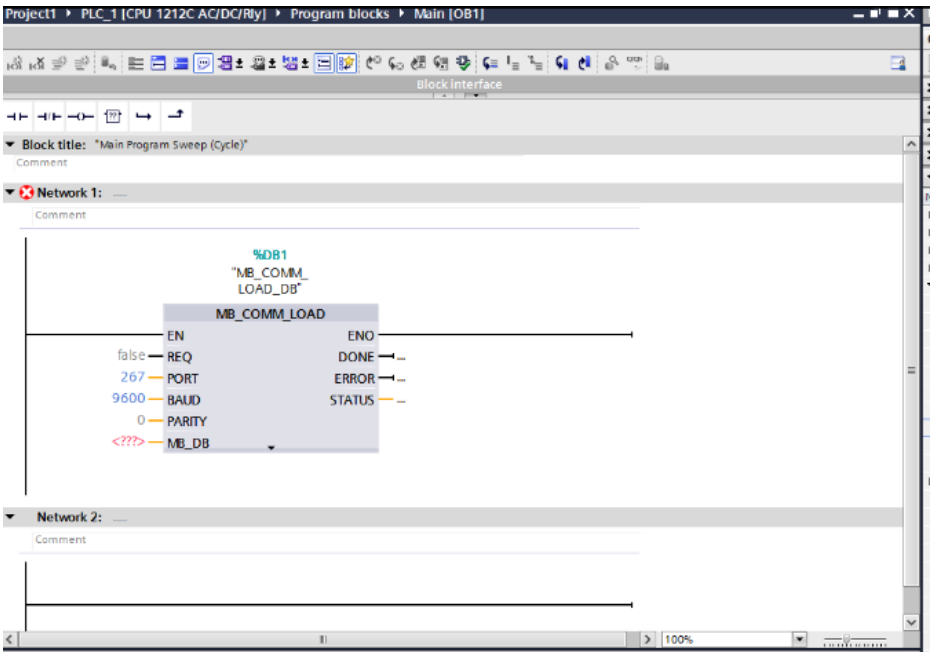
Al Observar toda la columna izquierda se encuentran todas las configuraciones por establecer como EN, REQ, PORT, entre otras. Se dejarán algunos parámetros por default como REQ *false*. Para el parámetro PORT hace referencia al puerto modbus que se instaló, para saber qué número de puerto establecido el programa se dirige a la vista de *device view*, posteriormente dirigirse en configuraciones RS422/485 y seleccionar, identificador de hardware, de esa manera se mostrará el número de puerto que se asignó Tia portal, en este caso es 267. Para los baudios serán 9 600. En paridad, hay que recordar que anteriormente lo se configuró en cero. Por lo tanto también será cero.

Figura 69. **Panel de configuración bloque de datos Comm_Load**



Fuente: elaboración propia, empleando software Tia portal.

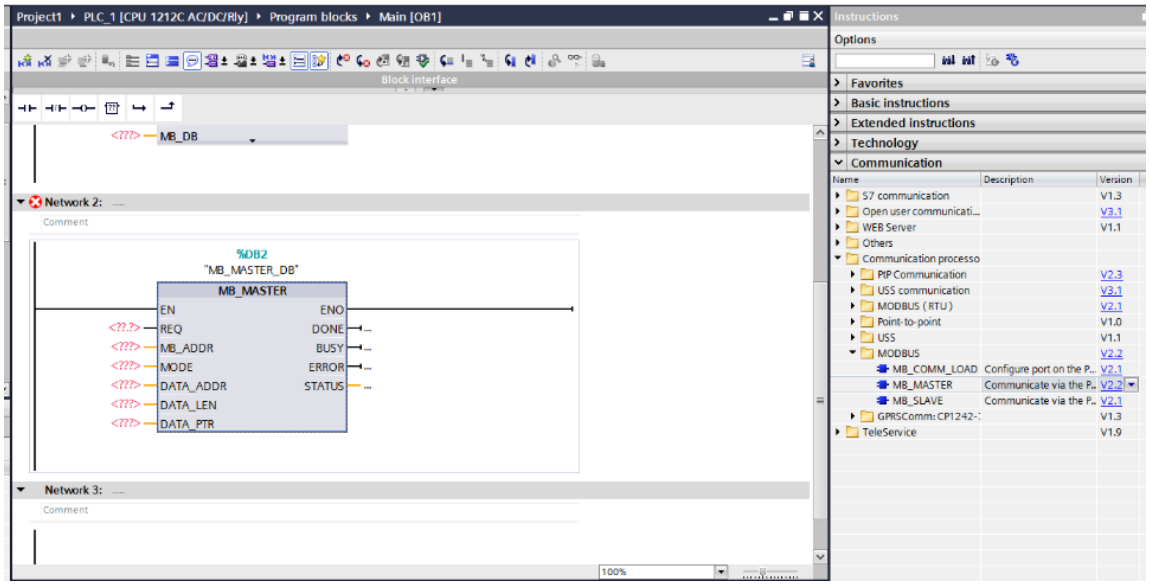
Figura 70. **Parámetros configurados bloque Comm_load**



Fuente: elaboración propia, empleando software Tia portal.

En la ubicación de la misma pantalla después de realizar el bloque de datos de carga de comunicación, se procederá a crear y arrastrar el DB de MB_Master, siempre como bloque de datos. Y se colocará en la segunda rama de programación lader. Como se muestra en la figura 71.

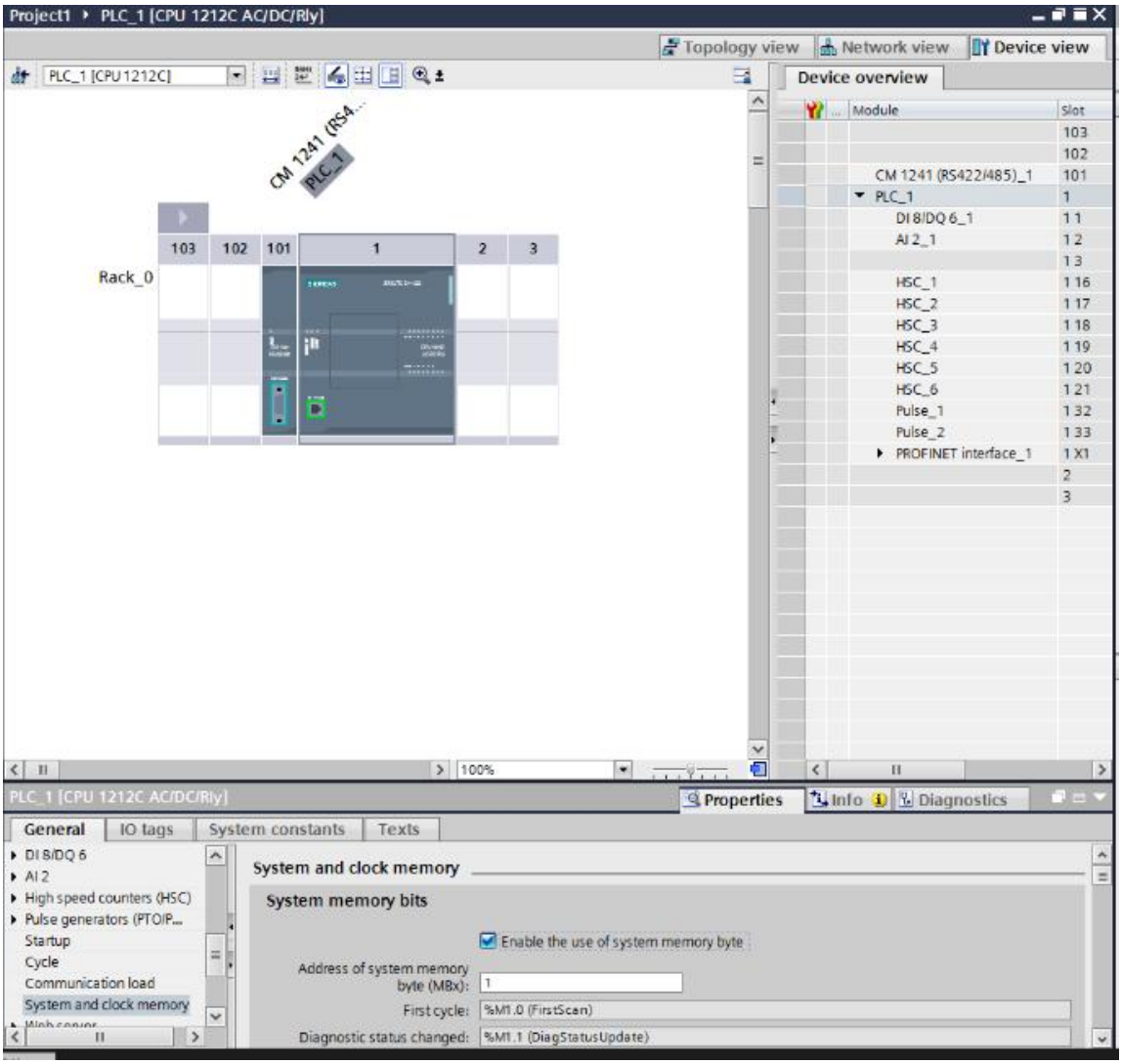
Figura 71. **Bloque MB_master**



Fuente: elaboración propia, empleando software Tia portal.

Antes de comenzar a configurar este bloque hay que configurar el *clockmemory bits*, el cual será el encargado de pedir los registros cada ciclo que se configure, por ejemplo cada segundo, cada minuto, entre otros. Para ello habrá que dirigirse a *device view* y se hace clic a la CPU. Como se muestra en la figura 77.

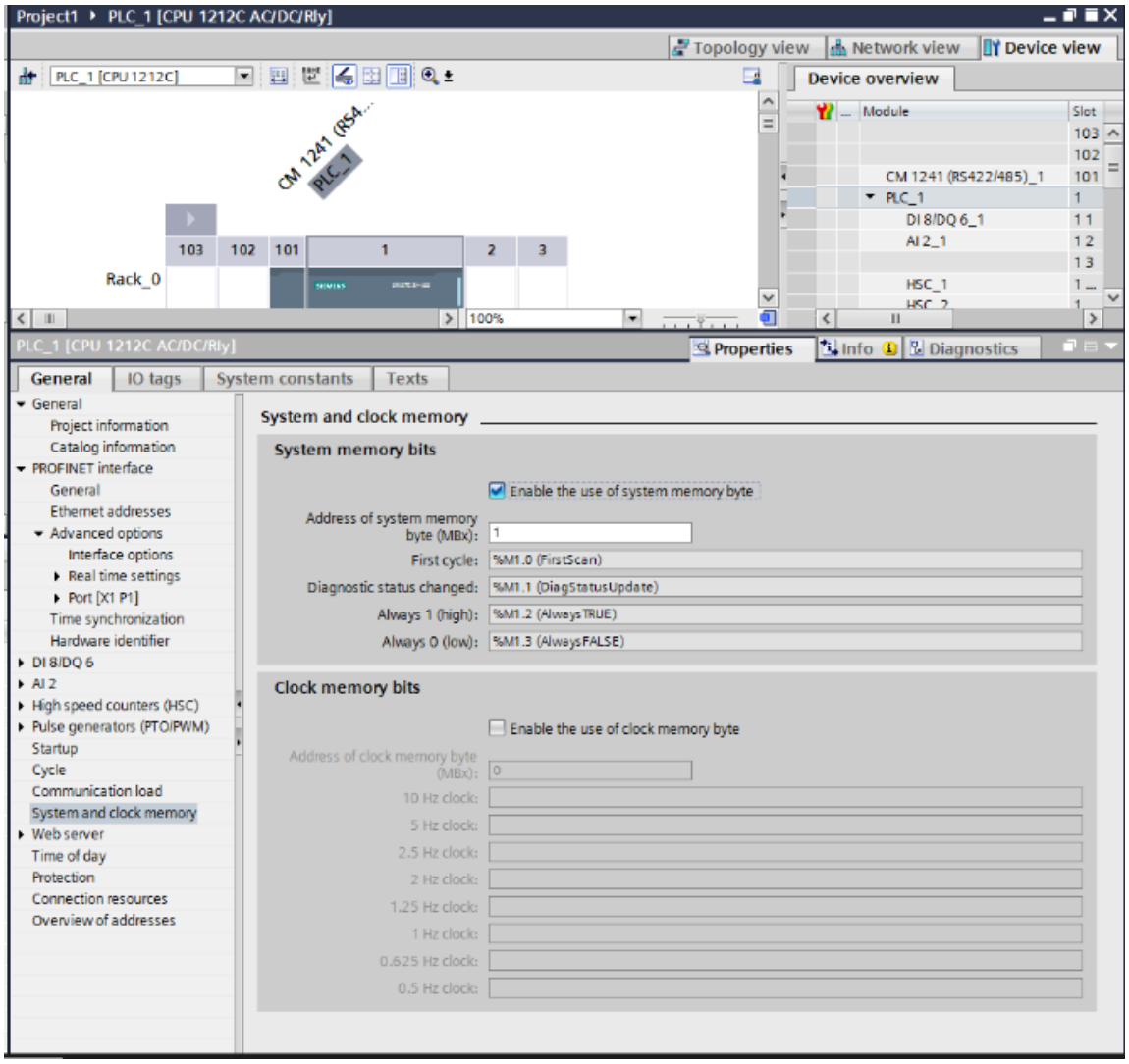
Figura 72. **Configuración de *clock memory* 1**



Fuente: elaboración propia, empleando software Tia portal.

Expandir el menú que se encuentra en la parte inferior y se mostrará más opciones a configurar. Posteriormente dirigirse a *system and clock memory* Aquí se encontrará la opción habilitar *clock memory*, como se muestra en la figura 73.

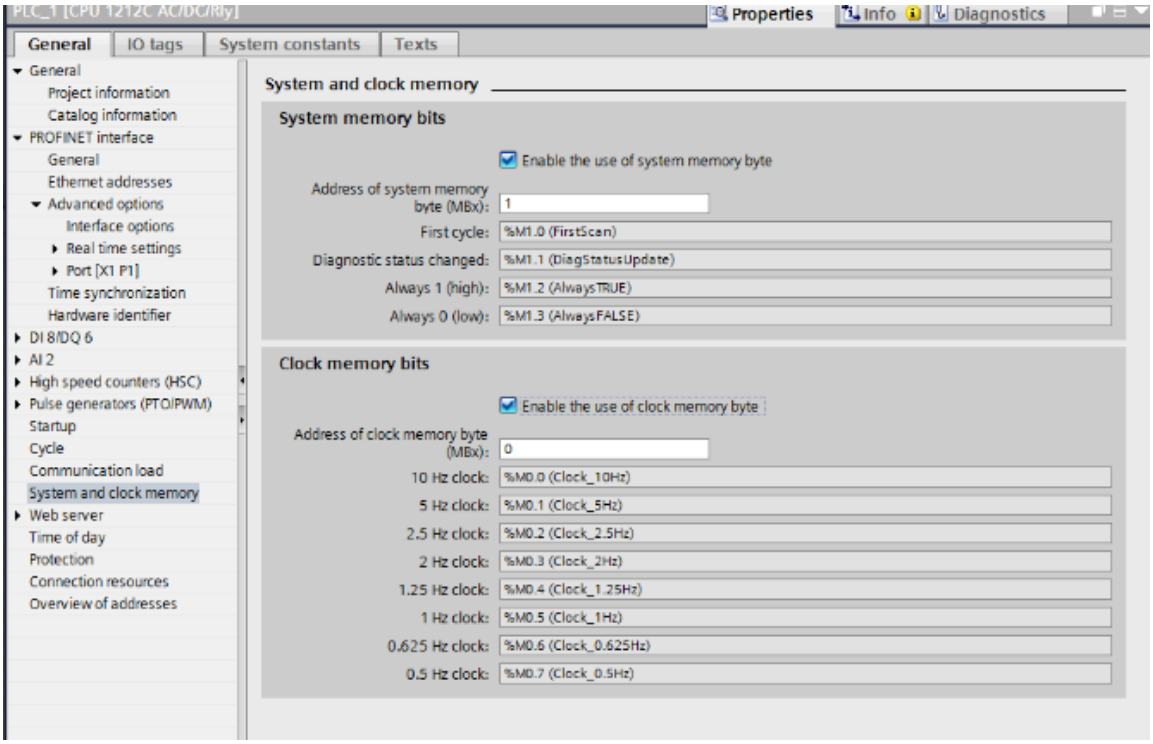
Figura 73. **Configuración *clock memory* 2**



Fuente: elaboración propia, empleando software Tia portal.

Se procederá presionar click en el cuadro habilitar, se lo dejará así por default esta cada segundo, y eso es funcional para este trabajo de investigación. Se mostrará en la figura 74.

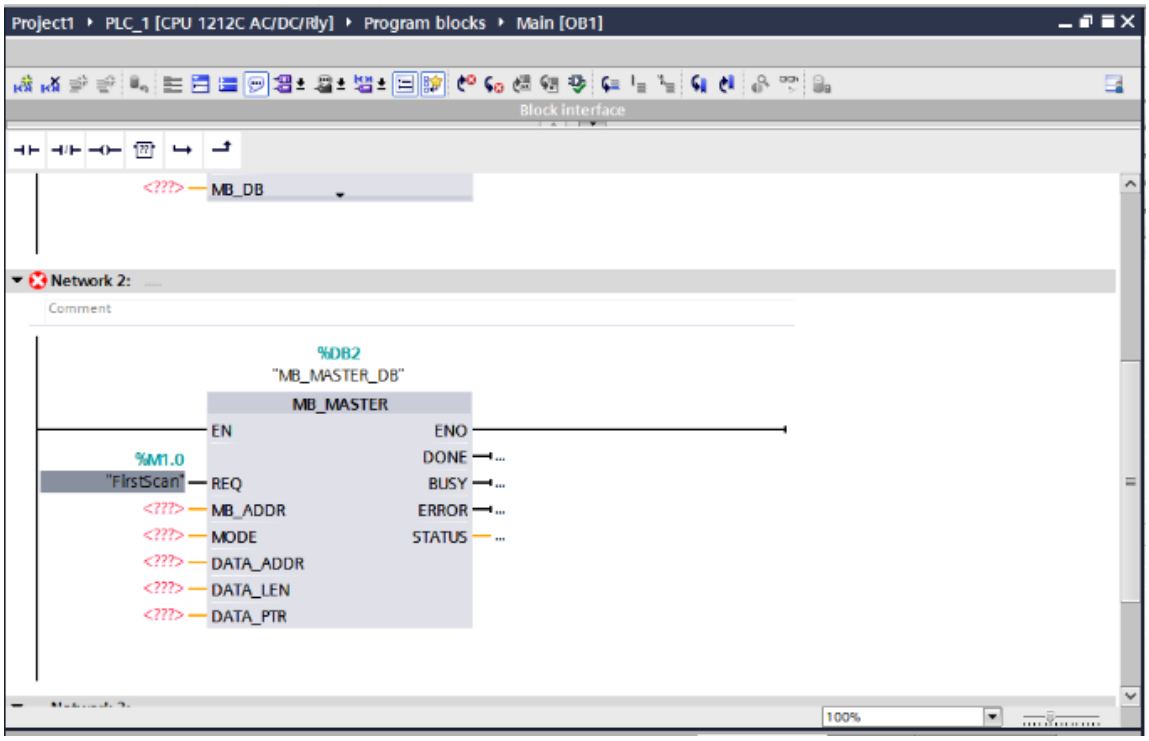
Figura 74. **Configuración de ciclo de ejecución**



Fuente: elaboración propia, empleando software Tia portal.

Regresando al bloque *master* se comenzará configurar todos los parámetros necesarios. Como por ejemplo en REQ se escribirá FIRST SCAN. Hay que tener en cuenta que si no se realizan todos los pasos anteriores no aparecerá está opción.

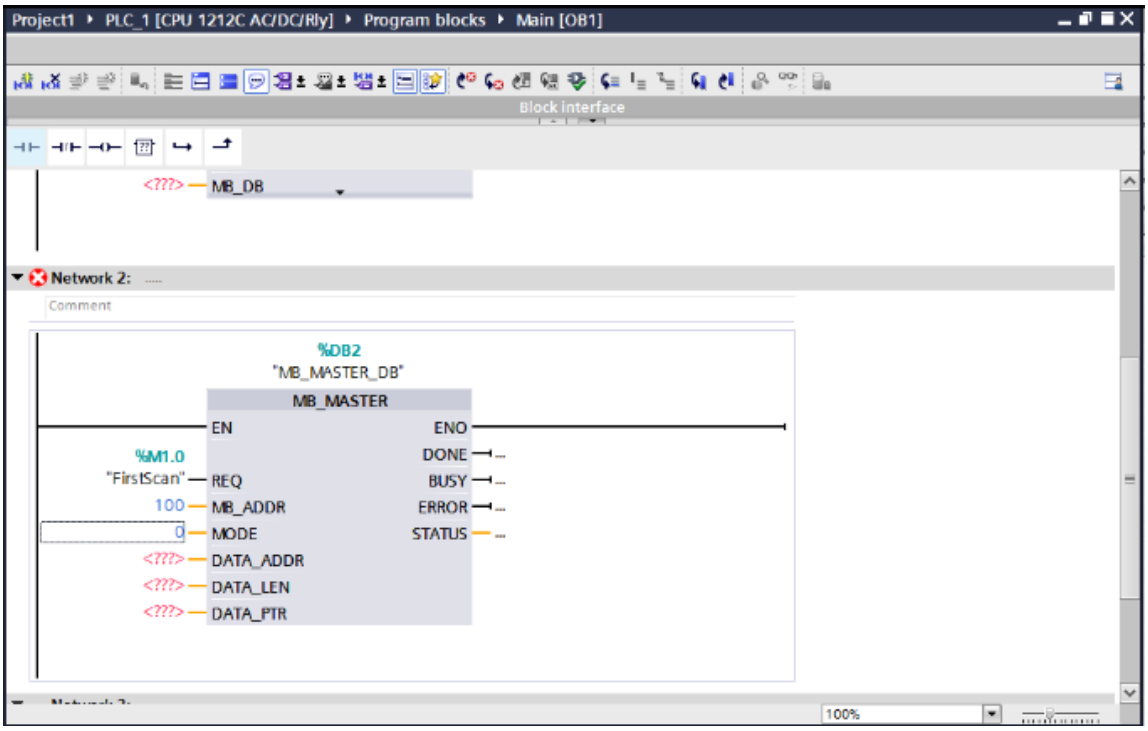
Figura 75. **Configuración parámetros bloque 1**



Fuente: elaboración propia, empleando software Tia portal.

Como siguiente paso el parámetro MB_ADDR sólo se refiere a registro de salida para el esclavo, en este caso puede ser del 1 al 256, es decir solo se puede tener esa cantidad de posiciones, en este caso se usará un número aleatorio, el cual será 100 , *mode* se pondrá 0.

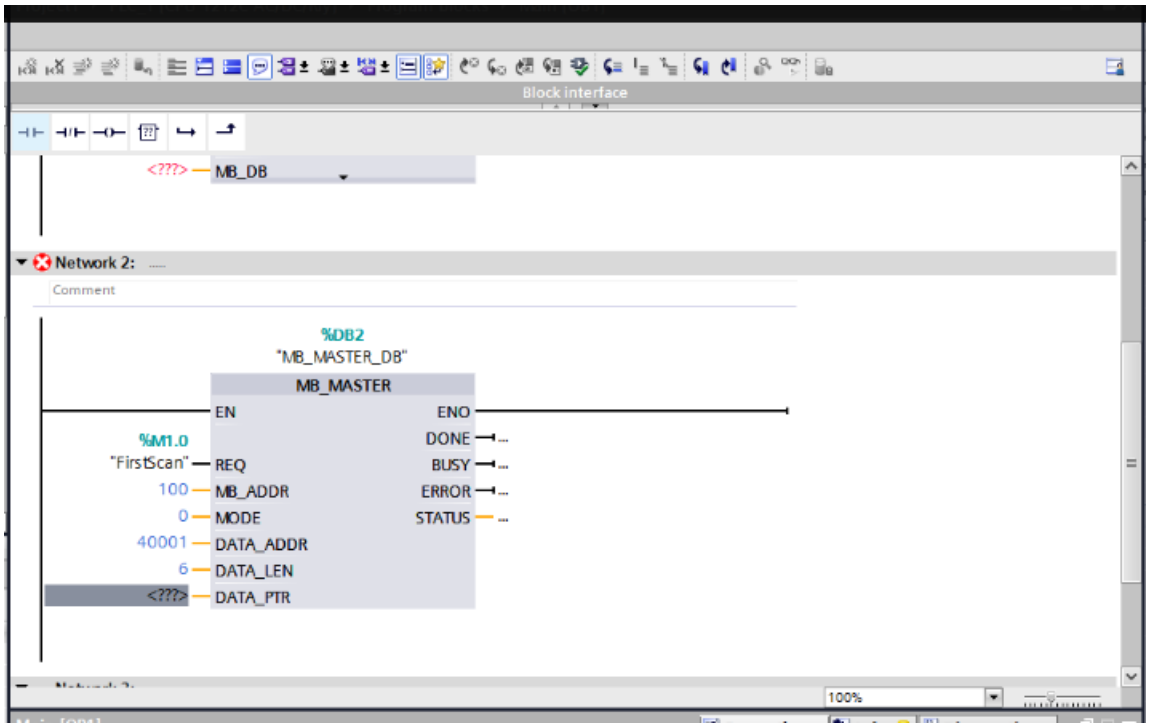
Figura 76. **Configuración parámetros bloque master 2**



Fuente: elaboración propia, empleando software Tia portal.

El siguiente parámetro es DATA_ADDR, en este caso hace referencia al registro que pedirá solicitud al bloque maestro, es decir, suponer que el dispositivo B es un sentron Pack 1 300, también es de la gama de Siemens, pero la importancia de este, es que se desea saber voltajes de un sistema trifásico, esos registros empiezan desde el 40 000 para arriba, por lo tanto, dispositivo A solicitará los estados de voltaje cada segundo, al dispositivo B, en este caso será el registro 40 001. Por lo mismo se pondrá ese número de registro en DATA_ADDR.

Figura 77. **Configuración de parámetros MB_master_DB**

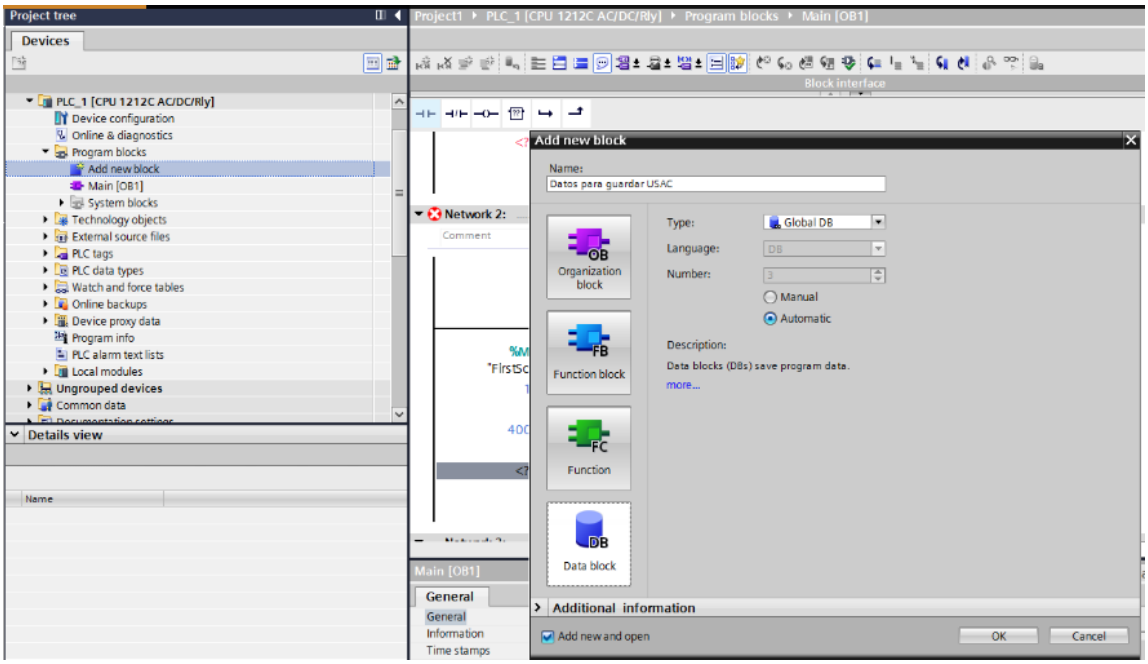


Fuente: elaboración propia, empleando software Tia portal.

El siguiente parámetro será DATA_LEN, este hace referencia a la interpretación de cuantos bits se van a utilizar para la lectura de lo deseado, como por ejemplo 300v, 200v, entre otros. Para ellos este utilizará dos palabras por medición solicitada, para fines de ejemplificación, se medirán para un sistema trifásico, por lo tanto serian 3 mediciones, entonces serían 6 palabras en total, esa será la longitud de la solicitud. Data_PTR por el momento se deja deshabilitada.

Como siguiente paso dirigirse a bloques de programa, y se añadirá otro bloque, pero este bloque será de datos, es decir un DB, se pondrán datos para diferenciar de los demás, en este caso se le colocará datos para guardar USAC, como se muestra a continuación.

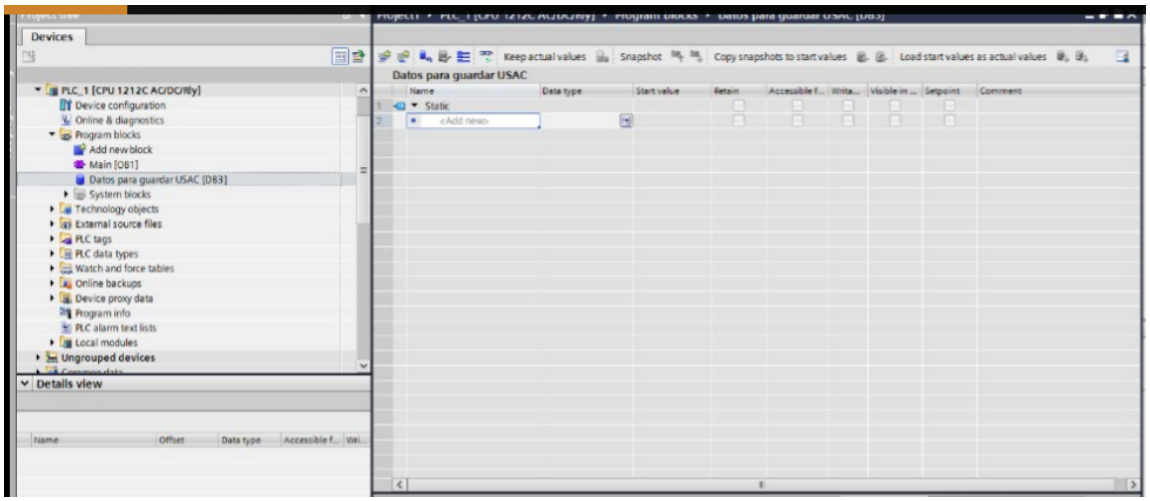
Figura 78. **Creación de bloques de datos main**



Fuente: elaboración propia, empleando software Tia portal.

Al crear el DB se desplegará una lista, se podrá poner un nombre a cada variable y declarar si será un entero, real, decimal, entro otros. Como se muestra en la figura 79.

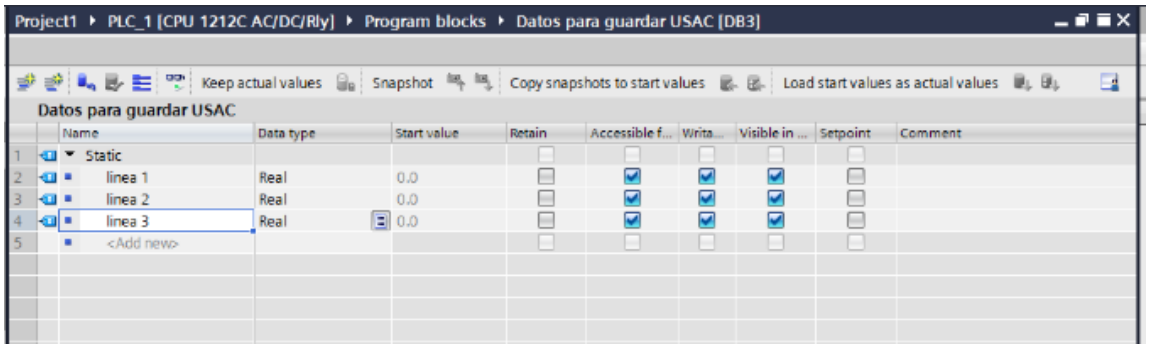
Figura 79. **Configuración de variables DB**



Fuente: elaboración propia, empleando software Tia portal.

En este caso serán las mediciones de 3 líneas vivas o energizadas, por lo tanto, se procederá a colocarles nombre, línea 1, línea 2, línea 3. Y se declararán que serán variables que pertenecen a los números reales. Como se muestra en la figura 80.

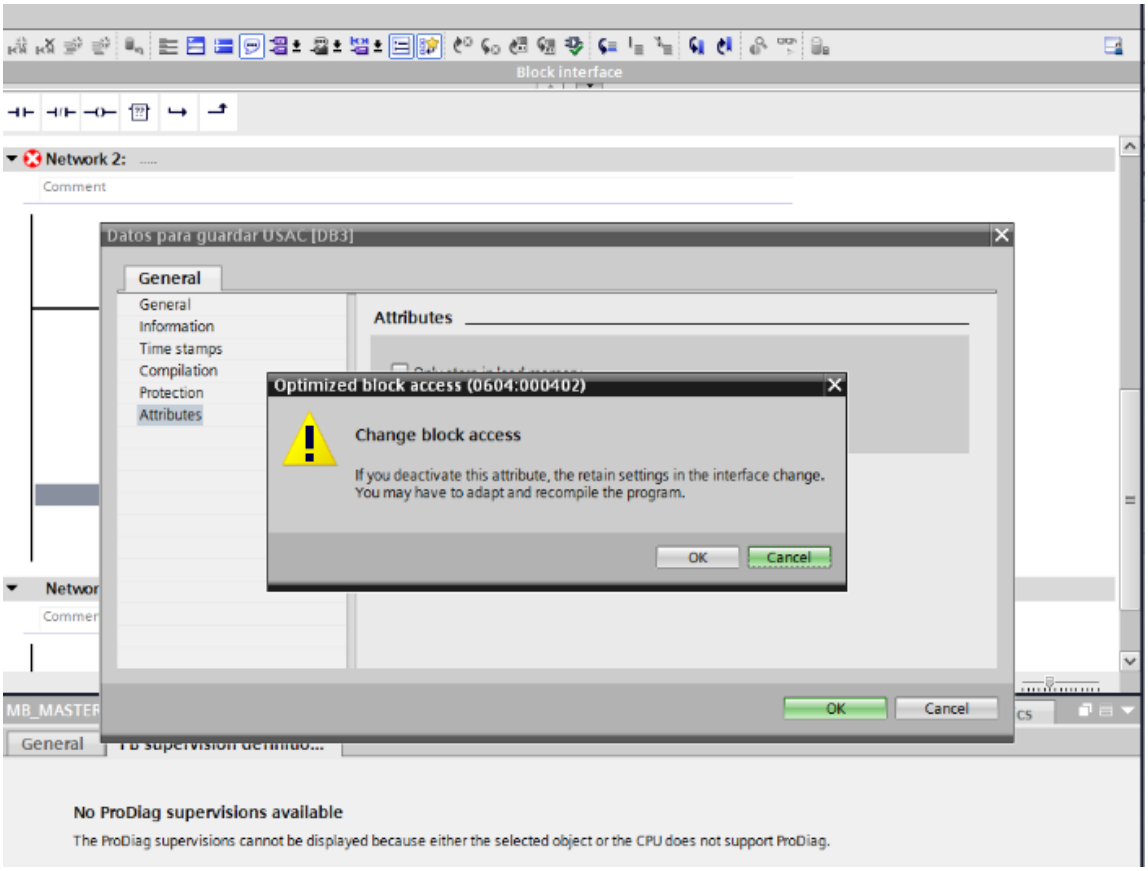
Figura 80. **Creación de variables bloque de datos**



Fuente: elaboración propia, empleando software Tia portal.

A este punto ya creadas las variables se procederá a presionar click derecho sobre el bloque de datos con el nombre datos para guardar USAC, y seleccionar solamente optimizador de bloques de acceso, las demás se dejarán deseleccionadas, se pulsará el ítem OK. Y se mostrará una advertencia, como se muestra en la figura 81.

Figura 81. **Advertencia de cambios de bloque de acceso**

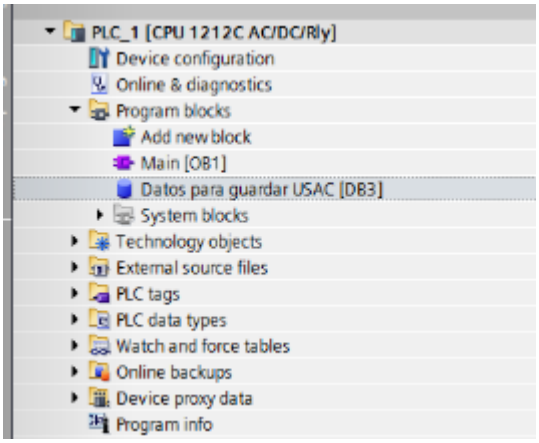


Fuente: elaboración propia, empleando software Tia portal.

Al finalizar este paso se procederá a regresar a la rama 2 en la cual está el bloque OB maestro, hay que recordar que el último parámetro no se configuró,

no se realizó, al crear ya el bloque de datos permitirá hacerlo, de manera que se arrastrará el bloque de datos (DB) y se colocará en DATA_PTR.

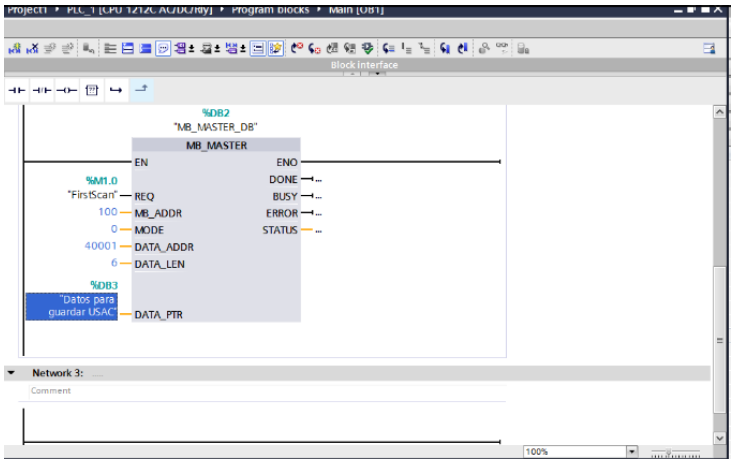
Figura 82. **Menú de bloques**



Fuente: elaboración propia, empleando software Tia portal.

Y se mostrará en la figura 83.

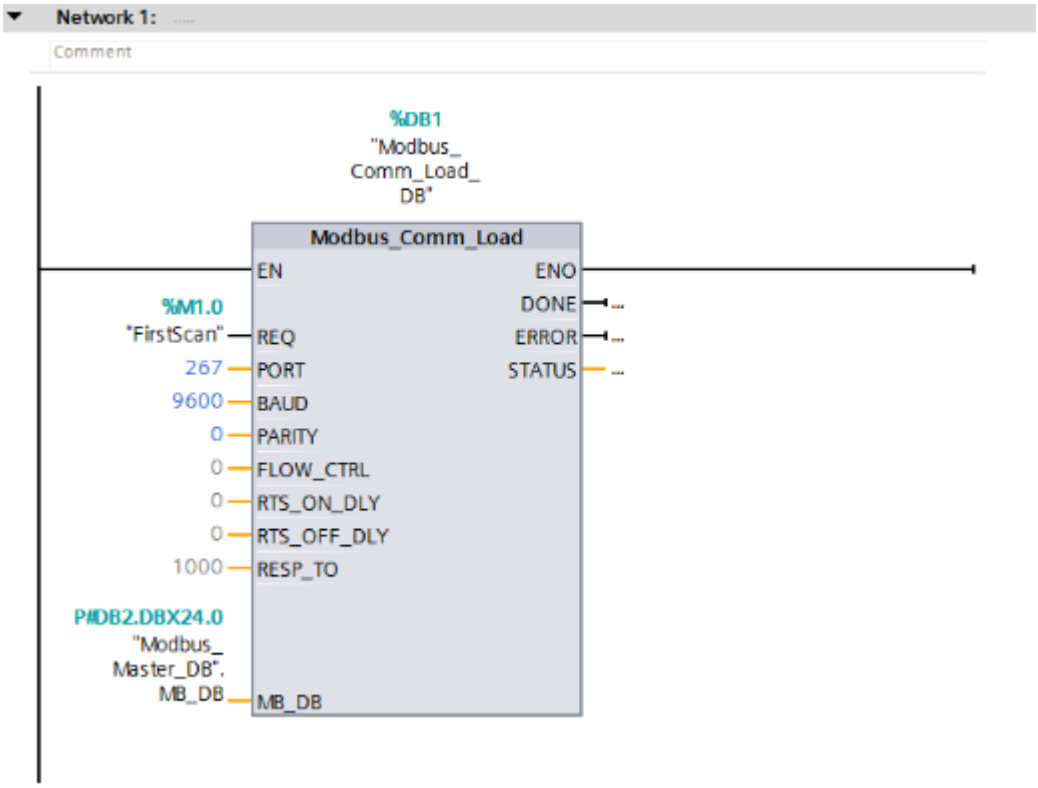
Figura 83. **Bloque con último parámetro configurado de bloque dos**



Fuente: elaboración propia, empleando software Tia portal.

Llegando a este punto solo faltaría una configuración para terminar de configurar en su totalidad el almacenamiento de datos del dispositivo B al dispositivo A para ello dirigirse a la rama 1, en la cual se encuentra el bloque de MB_COMM_LOAD, y se configurará el último parámetro MB_DB, como se muestra en la figura 84.

Figura 84. **Configuración de parámetro MB_DB del bloque Comm_load**



Fuente: elaboración propia, empleando software Tia portal.

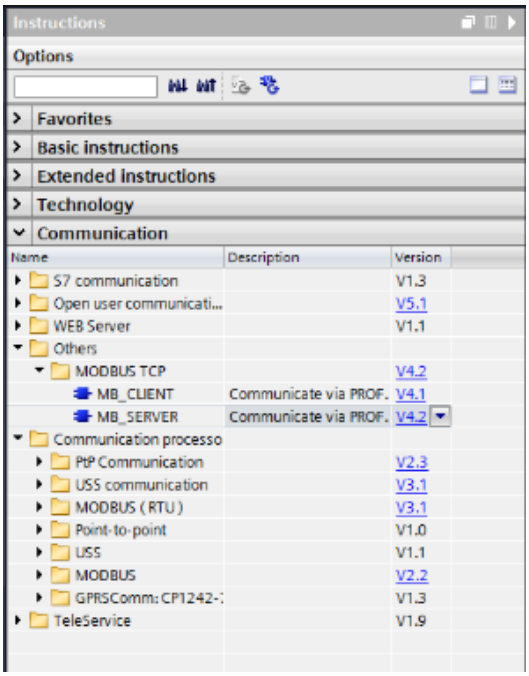
Se puede observar que en el último parámetro se colocó el modbus *master*, al terminar este segmento se finaliza el primer escenario. Se tiene las variables dentro del controlador lógico programable, cada segundo.

4.2. Respuesta de solicitud de protocolo dentro de un PLC S7-1200

Al tener todas las variables almacenadas dentro de nuestro controlador lógico programable, es posible controlar las variables y convertirlas a cualquier protocolo de salida deseado, como por ejemplo en el escenario dos se tiene como objetivo que un dispositivo C se solicite información del dispositivo B por medio del dispositivo A por medio de protocolo tcp-ip. A continuación se explicará el procedimiento para lograrlo.

Recopilando, al tener todas las variables de lectura guardadas en el PLC se procederá que colocar un MB_SERVER, que se encuentra en la sección de *communication* / *others* / modbus tcp / MB_SERVER. Se mostrará en la figura 85.

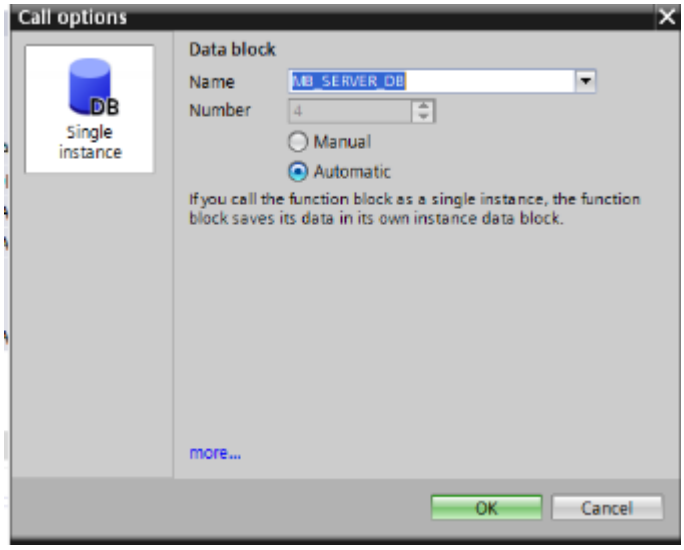
Figura 85. **Menú de bloque de MB_SERVER**



Fuente: elaboración propia, empleando software Tia portal.

Como los demás bloques creados anteriormente, se arrastrará esta opción al campo lader, y se mostrará en la figura 86.

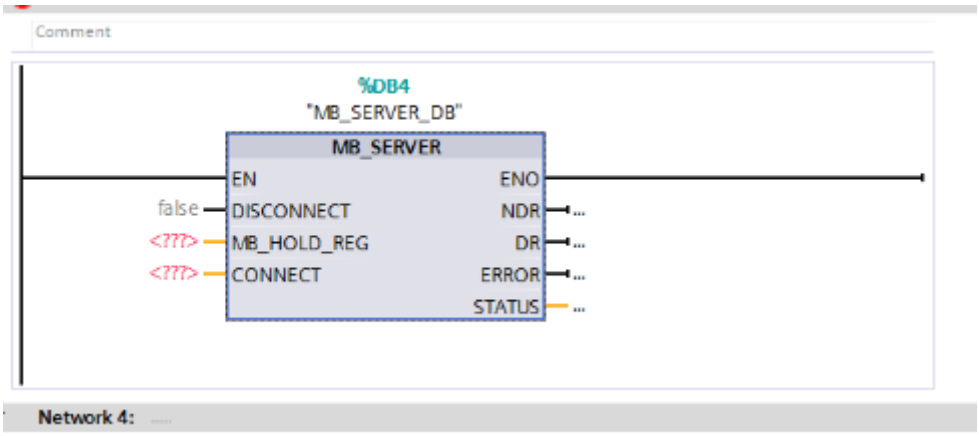
Figura 86. **Creación de bloque de datos MB_SERVER**



Fuente: elaboración propia, empleando software Tia portal.

Se observa que es otro bloque de datos con la diferencia que este será un bloque es maestro se procede a presionar la opción ok. Y aparecerá un bloque al cual se tendrá que configurar parámetros predeterminados para que encuentre una dirección IP de salida. Como se muestra en la figura 87.

Figura 87. **Bloque MB_SERVER_DB**



Fuente: elaboración propia, empleando software Tia portal.

Al tener este bloque se procederá a crear un bloque de datos predeterminado, este se configurará de acuerdo a los siguientes parámetros:

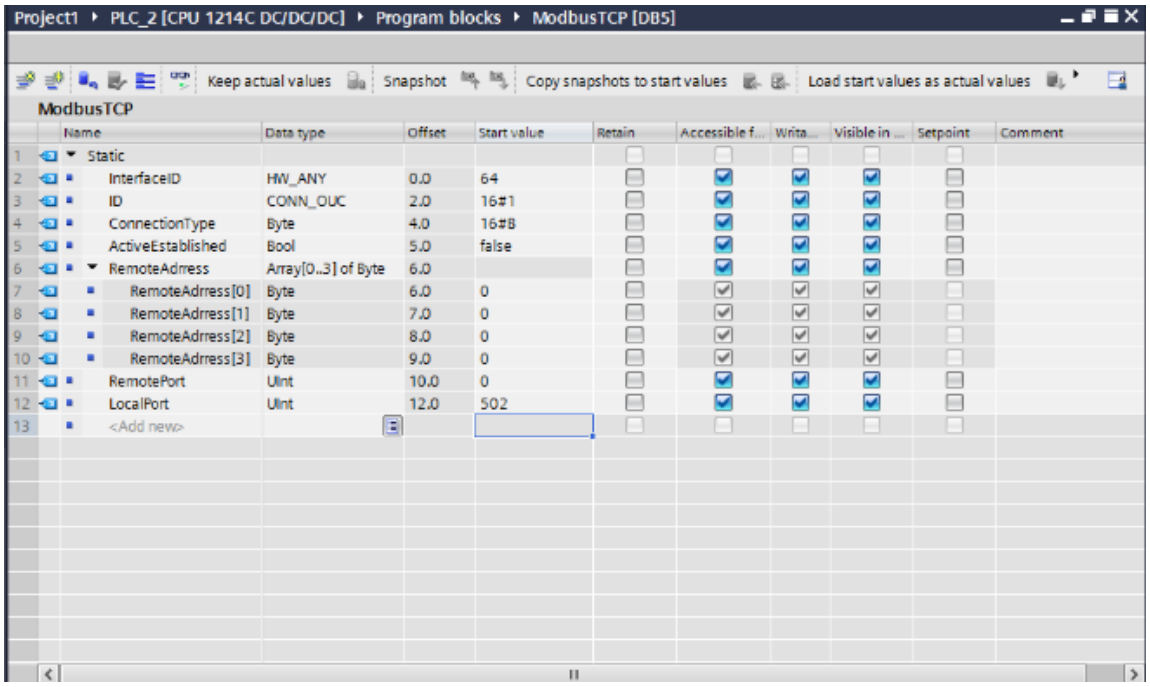
Tabla VIII. **Tabla de configuración para encontrar rutas de salida con IP**

Byte	Parameter	Data type	Start value	Description
0 ... 1	InterfaceID	HW_ANY	-	Hardware identifier of the local interface (value range: 0 to 65535).
2 ... 3	ID	CONN_OUC	-	Reference to this connection (value range: 1 to 4095). The parameter uniquely identifies a connection within the CPU. Each individual instance of the instruction "MB_SERVER" must use a unique ID.
4	ConnectionType	BYTE	11	Connection type Select 11 (decimal) for TCP. Other connection types are not permitted. If another connection type (e.g. UDP) is used, a corresponding error message is output at the STATUS parameter of the instruction.
5	ActiveEstablished	BOOL	FALSE	ID for the manner in which the connection is established Select FALSE for passive connection establishment.
6 ... 9	RemoteAddress	ARRAY [1..4] of BYTE	0.0.0.0	IP address of the connection partner, for example, for 192.168.0.1: • addr[1] = 192 • addr[2] = 168 • addr[3] = 0 • addr[4] = 1 If the instruction "MB_SERVER" is to accept connection requests from any connection partner, use "0.0.0.0" as IP address.
10 ... 11	RemotePort	UINT	0	Port number of the remote connection partner (value range: 1 to 49151). If the instruction "MB_SERVER" is to accept connection requests from any remote partner, use "0" as port number.
12 ... 13	LocalPort	UINT	502	Port number of the local connection partner (value range: 1 to 49151). The number of the IP port defines which IP port is monitored for connection requests of the Modbus client. The following TCP port numbers must not be used for the passive connection of the "MB_SERVER" instruction: 20, 21, 25, 80, 102, 123, 5001, 34962, 34963 and 34964.

Fuente: elaboración propia, empleando software Tia portal.

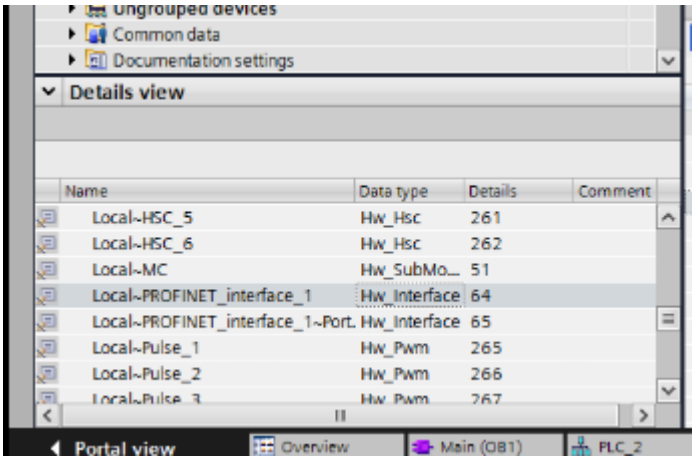
Se colocarán estos parámetros dentro de un bloque de datos con el fin que el parámetro *connect* de nuestro bloque maestro encuentre por default un receptor, y dará información a cualquiera que solicite.

Figura 88. **Creación de variables dentro del bloque de datos**



Fuente: elaboración propia, empleando software Tia portal.

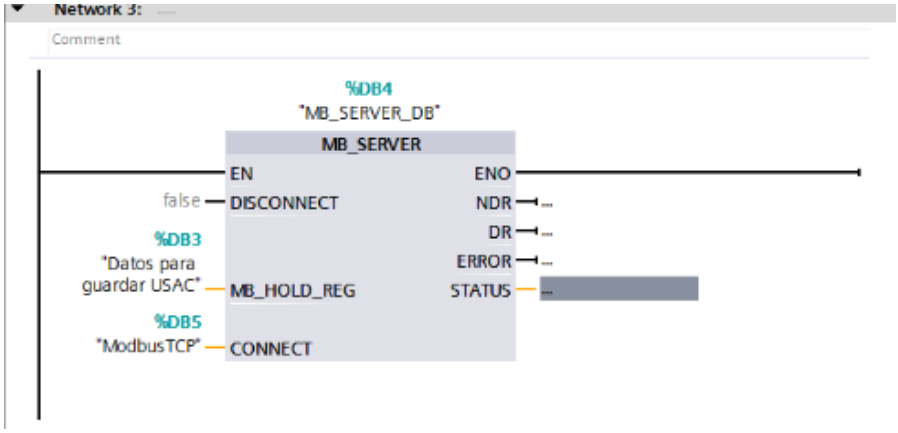
Figura 89. **Menú donde se ubica el *star value* de la variable interfaces ID**



Fuente: elaboración propia, empleando software Tia portal.

Al configurar todas las variables predeterminadas dentro del bloque de datos, se procederá configurar todos los parámetros del bloque maestro creado como se muestra a continuación.

Figura 90. **Bloque MB_SERVER_DB configurado correctamente**



Fuente: elaboración propia, empleando software Tia portal.

Las configuraciones del último bloque permitirán que cualquier dispositivo que se encuentre en la red del controlador lógico programable responda cualquier solicitud en el protocolo tcp/ip. Se debe tener cuidado con la configuración de puerto de salida del mismo, ya que no responderá sino se configura correctamente.

Al examinar ambas ramas de los bloques de datos se puede notar que comparten la misma base de datos, por ello es posible pasar las variables de un lugar a otro, para realizar tareas deseadas. Como por ejemplo en este caso podría ser, cuando una línea disminuya exponencialmente que active una alarma o entre una generadora con transferencia. Con esto se evitará los interpretes de en medio reduciendo costos innecesarios en nuestra red de automatización industrial y evitando más subredes dentro de la misma.

Al terminar los dos escenarios, unirlos e interpretarlos, se puede observar que se generó un *gateway*, en la cual varios dispositivos finales piden una solicitud y el controlador lógico programable responde cada solicitud con los datos correctos y sus protocolos de comunicación correspondiente.

CONCLUSIONES

1. Finalmente, si es posible crear y desarrollar una homologación de protocolo modbus tcp/ip a protocolo modbus RTU dentro de un controlador lógico programable, convirtiéndose el PLC en un *gateway* de la red ejemplificada.

2. La programación y la configuración de algunas funciones lógicas dentro de un controlador lógico programable representan el orden de entrada y salidas de los protocolos de comunicación.

3. Al tener una salida de modbus RTU y una entrada de modbus tcp/ip se Cumplió el estado cliente servidor dentro de la red ejemplificada y al mismo tiempo se puede aprovechar dicha homologación para realizar otras tareas dentro del PLC.

4. Obteniendo un enlace sin interrupciones de diferentes puntos es posible realizar diferentes tipos de homologaciones de protocolos como profibus y profinet.

5. Se pudo realizar una documentación detallada de las configuraciones del software, por medio de capturas de pantalla del Tia portal.

6. Al utilizar Tia portal como las configuraciones del software y las Conexiones correctas con los dispositivos asignados como hardware, fue posible la documentación global de ambas partes del trabajo de investigación.

RECOMENDACIONES

1. Al seleccionar otros dispositivos para repetir este trabajo de investigación, confirmar que los módulos a utilizar sean compatibles con PLC elegido.

2. Dentro del PLC hay variedad de campos para configuraciones lógicas, se recomienda a utilizar lader, por mejor interpretación y visión de todas las ramas configuradas en el proyecto.

3. Al tener en cuenta varios dispositivos que forman la red, asegurarse que todos posean los mismos parámetros en la fuente de alimentación, de esa manera economizarán y unificarán toda la red.

4. Asegurar que el controlador lógico programable elegido tenga físicamente los puertos correspondientes a los protocolos deseados a homologar.

5. Al configurar la solicitud de las variables de los registros, fijarse de cuantas palabras (16 bits) ocupa dentro de la parte lógica, de lo contrario la solicitud será rechazada.

6. Si se utilizara un PLC diferente de la marca Siemens verificar si cuenta con modo gráfico para facilitar las configuraciones correctas.

BIBLIOGRAFÍA

1. *COUCH, Leon. Sistemas de comunicación digitales y analógicos.* 7ª ed. Estados Unidos Florida. Pearson Educación S.A. 2008. 354 p.

2. FLOY, Thomas*. Fundamentos de sistemas digitales.* 9ª ed. España. Pearson educación S.A. 2006. 256 p.

3. RODRÍGUEZ, Artés, *Comunicaciones digitales.* España, Madrid. 450 p.

4. WAYNE, Tomasi*, Sistemas de comunicación electrónicas.* 4ª ed. Estados Unidos, Arizona. Pearson Educación S.A. 2003. 288 p.

5. Anónimo. *Modbus Com. [en línea].* <http://ecatalog.weg.net/files/wegnet/WEG-srw01-manual-de-la comunicacion-modbus-rtu-10000521680-4.0x-manual-espanol.pdf>. [Consulta: 10 de agosto de 2017].

6. Anónimo. *Módulos Modbus. [en línea].* <http://nilza.net/mainpage/detail/modbus-rs485-wiring-diagram>. [Consulta: 10 de abril de 2017].

7. Anónimo. *Telecomunicaciones y sus Módulos de comunicación. [en línea].* <http://tecomunicacion.webcindario.com/Archivos/Unidad%20III.pdf>. [Consulta: 10 de agosto de 2017].

8. Anónimo. *Automatización industrial y tipos de transmisores. [en línea].* <http://www.aie.cl/files/file/comites/ca/articulos/agosto->. [Consulta: 10 de agosto de 2017].

9. Anónimo. *Controladores lógicos programables aplicados en la industria. [en línea].* <http://www.microautomacion.com/capacitacion/Manual061ControladorLgicoProgramable:>. [Consulta: diciembre 2017].

10. Anónimo. *Módulos externos en la automatización industrial. [en línea].* <http://www.ptolomeo.unam.mx:8080/xmlui/bitstream/handle/132.248.52.100/734/A6.pdf?sequence=6>. [Consulta: 10 de agosto de 2017].

11. Anónimo. *Automatización industrial modelo 1200 Siemens. [en línea].* <https://w5.siemens.com/spain/web/es/industry/automatizacion/simatic/Documents/S71200-MANUAL%20DEL%20SISTEMA.PDF:>. [Consulta: 02 de enero de 2018].

12. Anónimo. *Automatización y puntos accesos Moxa network. [en línea].* <https://www.commgear.com/moxa-nport-6110-series->. [Consulta: 10 de abril de 2017].

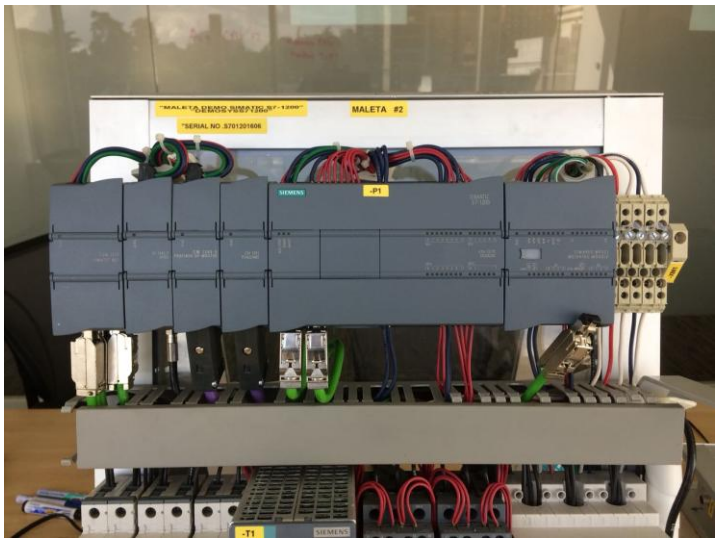
13. Anónimo. *Modbus Rtu y TCPIP. [en línea].* <www.micro.com.ar>. [Consulta: 10 de agosto 2017].

APÉNDICES

Apéndice 1. **Hardware del trabajo de investigación**

Fuente: elaboración propia.

Apéndice 2. **PLC S7-1200 hardware con módulos**



Fuente: elaboración propia.