



**FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA ELECTRÓNICA**

TRABAJO DE GRADUACIÓN

**“INVESTIGACION DE METODOS, APLICACIONES Y
FUNCIONAMIENTO DE RED DE COMUNICACIÓN INDUSTRIAL
MODBUS RS485.”**

**PARA OPTAR AL GRADO DE:
INGENIERO EN ELECTRONICA.**

PRESENTADO POR:

Br. Carlos Alexander Aquino Aragón

Br. John Jossue Stanley Vanegas Caballero.

Br. José Ricardo Monge Marroquín.

SANTA TECLA, MAYO DE 2024



AUTORIDADES UNIVERSITARIAS

ING. ROSENDO MAURICIO SERMEÑO PALACIOS
RECTOR

ING. FRANCISCO ALFREDO CARRILLO LARREYNAGA
VICE-RECTOR

ING. ROOSELVET ADOLFO OSORIO
SECRETARIO GENERAL

LIC. CARLOS ALBERTO COCA MUÑOZ
FISCAL

ING. MAURICIO ANTONIO AGUIRRE ORELLANA
DECANO DE LA FACULTAD DE INGENIERÍA

ING. RENÉ EDUARDO CUÉLLAR FIGUEROA
DIRECTOR DE ESCUELA DE INGENIERÍA ELÉCTRICA

SANTA TECLA, MAYO 2024



ASESOR

Ing. Gustavo Raúl Alfaro.

JURADO EVALUADOR

Ing. Francisco Ernesto Colorado Gutiérrez

Inga. Ana Gladis Revelo Verdi

Ing. José Alexander Ramírez Hernández

SANTA TECLA, MAYO 2024

AGRADECIMIENTOS.

Me gustaría agradecer primeramente a Dios por permitirme tomar el camino de avanzar en mis estudios; a mis catedráticos que ayudaron a formar mi carácter como profesional y a la Universidad Técnica Latinoamericana por abrirme las puertas y brindarme la oportunidad de pertenecer a esta gran familia de profesionales. Agradezco por haber puesto su fe en mis habilidades y su disposición para ayudarme la cual ha sido fundamental para la finalización de esta tesis.

CARLOS ALEXANDER AQUINO ARAGON

AGRADECIMIENTOS.

Para mi madre y Niko por ser la luz que me guía y reconforta que ahora al final de mi carrera la cual comencé hace dos décadas después de luchar contra las adversidades y problemas que se dieron a lo largo del camino veo con felicidad que he logrado terminar lo que me propuse.

No ha sido fácil, hubo momentos en que pensé en dejar todo pero puse empeño y esfuerzo ante toda situación y poco a poco avance hasta llegar hasta este momento en el cual me siento satisfecho y agradecido con la vida por haberme permitido lograr llegar hasta aquí.

Gracias totales a todos aquellos que creyeron en que podía hacerlo que con una palabra o un gesto me dieron ánimo, muchas gracias y bendiciones eternas.

JOHN JOSSUE STANLEY VANEGAS CABALLERO

AGRADECIMIENTOS.

En primer lugar agradezco a Dios todopoderoso y a la virgen María, por estar presente a lo largo de mi vida, que me haya permitido continuar con mis estudios, a pesar de las dificultades y adversidades.

Gracias a mi madre Sonia, quien con su amor ha creído en mis capacidades y por apoyarme en todo momento, gracias a mi suegra Imelda, que siempre me ha demostrado con su ejemplo de superación para alcanzar mis sueños, también agradecer de manera especial a mi suegro Juan Escapini que demostró su apoyo incondicional, gracias hasta el cielo +.

A mi amada esposa Karla quien con su amor, paciencia y esfuerzo me brindó su apoyo incondicional para poder cumplir todos mis objetivos personales y académicos y por haber creído en mis capacidades, animándome siempre a seguir adelante permitiéndome cumplir con mis metas y sueños, ella ha sido una persona incondicional.

A mi hija amada Malena que es mi inspiración para salir adelante y que cada logro que obtenga será para brindarle lo mejor.

A mi familia por haberme brindaron sus muestras de cariño me animaron en todo momento a lograr mis metas.

JOSE RICARDO MONGE MARROQUIN

INDICE

INDICE	7
LISTA DE ILUSTRACIONES	17
LISTA DE TABLAS	21
CAPITULO 1. GENERALIDADES.	22
1.1 INTRODUCCIÓN	23
1.2 OBJETIVOS	24
1.2.1. General:	24
1.2.2. Específicos:	24
1.3 DESCRIPCIÓN DEL PROBLEMA	25
1.4 JUSTIFICACIÓN	26
1.5 ALCANCES Y LIMITANTES	27
1.5.1. Alcances.	27
1.5.2. Limitantes.	27
1.6. ORGANIZACIÓN DEL PROYECTO DE GRADUACIÓN	27
1.6.1. Capítulo 1 Generalidades	28
1.6.2. Capítulo 2 Marco Teórico	28
1.6.3. Capítulo 3 Protocolos de comunicación, modelo OSI, TCP/IP y modelo de datos modbus.	28
1.6.4. Capítulo 4 Guías de trabajo del laboratorio de protocolo Modbus de la Utl.	28
1.6.5. Capítulo 5 Conclusiones del proyecto.	29
1.6.6 Capitulo 6 Referencias bibliográficas	29

	8
1.6.7 Capitulo 7 Anexo Construcción del módulo.	29
1.7 METODOLOGÍA DE LA INVESTIGACIÓN	29
1.7.1. La investigación documental:	29
1.7.2. La investigación bibliográfica:	29
CAPITULO 2 MARCO TEORICO	30
2.1 ANTECEDENTE HISTÓRICOS	31
2.2. DESCRIPCIÓN.	32
2.3. CONCEPTUALIZACIÓN.	33
2.4. CONEXIÓN DE INTERFAZ.	35
2.4.1. Dispositivos de entrada.	35
2.4.1.1. Pulsador	35
2.4.1.2 El interruptor	36
2.4.1.2.1 Tipos de interruptores	37
2.4.2 Dispositivos de salida.	37
2.4.2.1 Relé.	37
2.4.2.2 Contactor	38
2.4.2.2.1 Conceptos básicos.	38
2.4.2.3. Electroválvula.	39
2.4.2.4. Indicador luminoso.	40
2.5. RASPBERRY PI	40
2.5.1. Características	41
2.5.2. Cómo funciona una Raspberry Pi	41

	9
2.5.3. Modelos de Raspberry Pi	42
2.5.3.1. Versión Pi Model B	42
2.5.3.2. Versión Pi Model A	43
2.5.3.3. Versión Pi Zero	43
2.5.3.5. Versión Pi Pico	44
2.6. CONTROLADOR LÓGICO PROGRAMABLE.	44
2.6.1. Logo!	46
2.6.1.1 ¿Cómo Funciona el LOGO?	47
2.7. DISPOSITIVOS DE COMUNICACIÓN INALÁMBRICA.	50
2.7.1. Bluetooth.	51
2.8 PROTOCOLOS DE COMUNICACIÓN.	52
2.8.1 HART.	52
2.8.1.1 Comunicación Punto a punto:	53
2.8.1.2 Comunicación tipo Multipunto:	54
2.8.2 Foundation Fielbus.	54
2.8.2.1. Diseño.	55
2.8.2.2. Ventajas.	56
2.8.3. Profibus.	56
2.8.3.1 Cómo funciona.	56
2.8.3.2. Tipos de PROFIBUS	58
2.8.3.2.1 PROFIBUS FMS	58

	10
2.8.3.2.2. PROFIBUS DP	58
2.8.3.2.3. PROFIBUS PA	59
2.8.4. DeviceNet.	60
2.8.5 Contronet	62
2.8.5.1 Características	63
2.8.6. Ethernet IP	63
2.8.6.1 Qué es la capa física de Ethernet	64
2.8.6.2 Qué es la capa de enlace de datos	65
2.8.7 AsiBus.	66
2.8.7.1 Características principales	67
2.8.8 PROFINET	68
2.8.8.1 Características clave de PROFINET	69
2.8.9 Ethercat	70
2.8.9.1 Grupo de Tecnología EtherCAT	71
2.8.9.2 Estándares internacionales	71
2.8.9.3 Principio fundamental	72
2.8.9.4 Representación de datos	72
2.8.9.5 Marco EtherCAT	73
2.8.10 BACnet	74
2.8.11 CANOpen	75
2.8.12. MODBUS.	76
2.8.12.1 ¿Quién inventó Modbus?	77

	11
2.8.12.2 Tipos de protocolo Modbus	77
2.8.12.2.1. Modbus RTU.	78
2.8.12.2.2 Modbus TCP	81
2.8.12.2.3. Modbus ASCII	83
2.8.12.2.4. Modbus Plus.	84
2.9. PROGRAMACIÓN	85
2.9.1. Código maquina	85
2.9.2 Ensamblador	86
2.9.3 Lenguaje C	86
2.9.4 Python	87
2.9.5 Compilador	87
CAPITULO 3. PROTOCOLOS DE COMUNICACIÓN, MODELO OSI, TCP/IP Y	
MODELO DE DATOS MODBUS.	88
3.1. PROTOCOLOS DE COMUNICACIÓN.	89
3.1.1. Descripción general de los protocolos.	89
3.1.1.1. El protocolo.	89
3.1.2. Estándares del modelo OSI.	91
3.1.2.1 ¿Cómo funciona el modelo OSI?	92
3.1.2.2. ¿Para qué sirve el modelo OSI?	92
3.1.2.3 Capas del modelo OSI	93
3.1.2.3.1. Capa física.	93
3.1.2.3.2 Capa de enlace de datos.	94

	12
3.1.2.3.3. Capa de red.	95
3.1.2.3.4. Capa de transporte.	96
3.1.2.3.5. Capa de sesión.	97
3.1.2.3.6. Capa de presentación.	99
3.1.2.3.7. Capa de aplicación.	100
3.1.3. Modelo de arquitectura del protocolo TCP/IP	101
3.1.3.1. Capa de red física	102
3.1.3.2. Capa de vínculo de datos	102
3.1.3.3. Capa de Internet	102
3.1.3.3.1. Protocolo IP	102
3.1.3.3.2. Protocolo ARP	103
3.1.3.3.3. Protocolo ICMP	103
3.1.3.4. Capa de transporte	103
3.1.3.4.1. Protocolo TCP	104
3.1.3.4.2. Protocolo SCTP	104
3.1.3.4.3. Protocolo UDP	104
3.1.3.5. Capa de aplicación	105
3.1.3.5.1. Servicios TCP/IP estándar	105
3.1.3.5.2. Comandos UNIX "r"	106
3.1.3.5.3. Servicios de nombres	106
3.1.3.5.4. Servicio de directorios	106
3.1.3.5.5. Servicios de archivos	107

	13
3.1.3.5.6. Administración de la red	107
3.1.3.5.7. Protocolos de enrutamiento	107
3.1.4. TCP/IP Modbus	107
3.1.4.1. Modbus de control con protocolo TCP	108
3.1.4.2. Operación dúplex completo	112
3.1.4.3. Encabezado TCP	112
3.1.4.4. Uso de conexiones	113
3.2 MODELO DE DATOS DE MODBUS.	115
3.2.1. Funciones de Modbus.	115
3.2.1.1. Funcion Read Holding Register.	117
3.2.1.2. Funcion Write Multiple Coils.	117
3.2.2. La trama Modbus.	118
3.2.2.1. Campos de una trama Modbus	118
3.2.2.2. Trama Modbus TCP	119
CAPÍTULO 4 GUIAS DE LABORATORIO.	120
4.1 FORMATO DE GUÍAS DE LABORATORIO.	121
4.2 GUÍA OPERACIÓN BASICA DE RASPBERRY PI.	122
I. OBJETIVOS.	122
II. INTRODUCCIÓN.	122
Configuración de pines de Raspberry Pi	123
Especificaciones de hardware	124
Sistema operativo	126

	14
Puntos de consideración	126
III. MATERIALES Y EQUIPO.	128
IV. PROCEDIMIENTO.	128
V. PRUEBA DE CONOCIMIENTOS	133
VI. CONCLUSIONES DE LA PRÁCTICA	133
4.3 GUÍA BASICA DE LOGO V8 SIEMENS.	134
I. OBJETIVOS.	134
II. INTRODUCCIÓN.	134
LOGO! es el módulo lógico universal de Siemens.	134
Programar LOGO!	136
Bornes	136
Entradas Digitales	137
Entradas Analógicas	137
Salidas Digitales	137
Salidas Analógicas	138
Bloques y su numeración	138
Representación de un bloque en el display.	138
Esquema de circuito a LOGO!8	139
Realización del circuito mediante LOGO!	139
LOGO! Soft Comfort V8.0	139
Interfaz de programación	140
Interfaz de proyecto	141
III. MATERIALES Y EQUIPO.	142

	15
IV. PROCEDIMIENTO.	143
V. PRUEBA DE CONOCIMIENTOS	156
VI. CONCLUSIONES DE LA PRÁCTICA	158
4.4 GUÍA INICIANDO CON MODBUS.	159
I. OBJETIVOS.	159
II. INTRODUCCIÓN.	159
III. MATERIALES Y EQUIPO.	159
IV. PROCEDIMIENTO.	160
V. PRUEBA DE CONOCIMIENTOS	172
VI. CONCLUSIONES DE LA PRÁCTICA	172
CAPITULO 5 CONCLUSIONES Y RECOMENDACIONES	173
5.1. CONCLUSIONES:	174
5.2. RECOMENDACIONES:	175
CAPITULO 6 REFERENCIA BIBLIOGRAFICA.	177
6.1 REFERENCIAS BIBLIOGRÁFICAS	178
CAPITULO 7 ANEXOS.	181
7.1 DISEÑO Y CONSTRUCCION DEL MÓDULO DEL PROYECTO.	182
7.1.1 Diseño del Módulo.	182
7.1.2. Descripción del modulo	182
7.1.3 Caracterización del modulo	183
7.1.3.1 Estructura física.	183
7.1.3.2 Equipo de control.	185

7.1.3.2.1. Componentes de los equipos de control	185
7.1.3.3. Diagrama eléctrico y conexión	189
7.1.3.4. Programa de aplicación	193
7.1.3.4.1 Estructura de logo	194
7.1.3.4.2. Entradas de logo	195
7.1.3.4.3. Salidas de logo	196
7.1.3.4.4 Configuración	197
7.1.3.4.5. Información	198
7.1.3.4.6. Información de conexión	198
7.1.4 Programa de la aplicación SCADA en PYTHON.	199
7.1.4.1 Imágenes para aplicación SCADA.	211
7.1.4.2. Mapa de Modbus del LOGO!	212

LISTA DE ILUSTRACIONES

<i>Ilustración 1 Contacto pulsador</i>	<i>36</i>
<i>Ilustración 2 Interruptor</i>	<i>36</i>
<i>Ilustración 3 Característica del interruptor</i>	<i>36</i>
<i>Ilustración 4 Interruptor unipolar y bipolar.....</i>	<i>37</i>
<i>Ilustración 5 Símbolo del relé.</i>	<i>37</i>
<i>Ilustración 6 Presentación del relé y configuración de pines</i>	<i>38</i>
<i>Ilustración 7 Presentación del relé y configuración de pines</i>	<i>38</i>
<i>Ilustración 8 Electroválvulas.....</i>	<i>39</i>
<i>Ilustración 9 Indicador luminoso</i>	<i>40</i>
<i>Ilustración 10 Raspberry Pi.....</i>	<i>41</i>
<i>Ilustración 11 Característica.....</i>	<i>41</i>
<i>Ilustración 12 Raspberry Pi Pía 4.....</i>	<i>42</i>
<i>Ilustración 13 Raspberry Pi Zero.....</i>	<i>43</i>
<i>Ilustración 14 Raspberry Pi 400.....</i>	<i>44</i>
<i>Ilustración 15 Raspberry Pi pico</i>	<i>44</i>
<i>Ilustración 16 PLC industrial.....</i>	<i>44</i>
<i>Ilustración 17 Bloque simplificado del PLC.....</i>	<i>46</i>
<i>Ilustración 18 Partes del logo.....</i>	<i>48</i>
<i>Ilustración 19 Autómata s7-1500.....</i>	<i>50</i>
<i>Ilustración 20 Presentación física del módulo bluetooth</i>	<i>52</i>
<i>Ilustración 21 Protocolo Hart</i>	<i>53</i>
<i>Ilustración 22 Protocolo Hart punto a punto</i>	<i>53</i>
<i>Ilustración 23 Protocolo Hart multipunto</i>	<i>54</i>
<i>Ilustración 24 Protocolo foundation fieldbus</i>	<i>55</i>
<i>Ilustración 25 Profibus.....</i>	<i>58</i>

<i>Ilustración 26 Red Devicenet.....</i>	<i>62</i>
<i>Ilustración 27 Red Contronet.....</i>	<i>63</i>
<i>Ilustración 28 Red Ethernet.....</i>	<i>66</i>
<i>Ilustración 29 Profinet.....</i>	<i>70</i>
<i>Ilustración 30 Macro Ethercat.....</i>	<i>74</i>
<i>Ilustración 31 Bacnet permite la conexión de varios dispositivos.....</i>	<i>75</i>
<i>Ilustración 32 CAN y CANOpen en el modelo OSI.....</i>	<i>76</i>
<i>Ilustración 33 La ADU de RTU</i>	<i>79</i>
<i>Ilustración 34 La ADU de TCP.....</i>	<i>82</i>
<i>Ilustración 35 La ADU de ASCII.....</i>	<i>83</i>
<i>Ilustración 36 Arquitectura básica de Modbus</i>	<i>85</i>
<i>Ilustración 37 Código Máquina</i>	<i>86</i>
<i>Ilustración 38 Programa hola mundo con ensamblador.....</i>	<i>86</i>
<i>Ilustración 39 Programa hola mundo con lenguaje C.....</i>	<i>86</i>
<i>Ilustración 40 Programa hola mundo con lenguaje Python.....</i>	<i>87</i>
<i>Ilustración 41 Esquema de funcionamiento de un compilador.....</i>	<i>87</i>
<i>Ilustración 42 Capa modelo OSI</i>	<i>93</i>
<i>Ilustración 43 Medio físico de transporte</i>	<i>94</i>
<i>Ilustración 44 Codificación de la señal</i>	<i>94</i>
<i>Ilustración 45 Enlace de datos</i>	<i>95</i>
<i>Ilustración 46 Capa de red.....</i>	<i>96</i>
<i>Ilustración 47 Capa de transporte.....</i>	<i>97</i>
<i>Ilustración 48 Capa sesión.....</i>	<i>98</i>
<i>Ilustración 49 Función de la capa presentación.....</i>	<i>99</i>
<i>Ilustración 50 Capa de aplicación</i>	<i>101</i>
<i>Ilustración 51 Mensaje Modbus.....</i>	<i>107</i>
<i>Ilustración 52 Cabecera protocolo de aplicación Modbus</i>	<i>108</i>

<i>Ilustración 53 Flujo de trabajo de conexión TCP.</i>	109
<i>Ilustración 54 Conector RJ45 y cable UTP</i>	109
<i>Ilustración 55 Red wi-fi en una oficina formal</i>	111
<i>Ilustración 56 Control de flujo.</i>	111
<i>Ilustración 57 Conexión dúplex completo</i>	112
<i>Ilustración 58 Encabezado TCP y datos encapsulados en el campo de datos IP.</i>	113
<i>Ilustración 59 Conexión TCP desde el protocolo de enlace hasta la finalización</i>	114
<i>Ilustración 60 Secuencia de la trama</i>	119
<i>Ilustración 61 Diseño del modulo</i>	182
<i>Ilustración 62 Medidas del módulo</i>	183
<i>Ilustración 63 elección de madera</i>	184
<i>Ilustración 64 edición y corte de las partes de modulo</i>	184
<i>Ilustración 65 Modulo completamente armado</i>	184
<i>Ilustración 66 Interruptores</i>	185
<i>Ilustración 67 indicadores luminosos</i>	186
<i>Ilustración 68 borneras de conexión, salida del LOGO!</i>	186
<i>Ilustración 69 borneras de conexión de control</i>	186
<i>Ilustración 70 Protección eléctrica, autómato 2 Amperios.</i>	187
<i>Ilustración 71 LOGO!</i>	187
<i>Ilustración 72 Raspberry Pi</i>	188
<i>Ilustración 73 Monitor</i>	188
<i>Ilustración 74 Teclado y Mouse</i>	188
<i>Ilustración 75 Conexión de alimentación eléctrica</i>	189
<i>Ilustración 76 Conexión eléctrica de las entradas del LOGO!</i>	189
<i>Ilustración 77 Conexión de las salidas del logo</i>	190
<i>Ilustración 78 Conexión eléctrica de Raspberry Pi</i>	190
<i>Ilustración 79 Modulo conectado y funcionando cada uno de sus componente</i>	190

<i>Ilustración 80 Conexión interna del módulo de los pilotos</i>	<i>191</i>
<i>Ilustración 81 Conexión eléctrica de los componentes del modulo</i>	<i>191</i>
<i>Ilustración 82 Canaleta utilizada para organizar el cable</i>	<i>191</i>
<i>Ilustración 83 Conexión de señales del Raspberry Pi, monitor, teclado y mouse.</i>	<i>191</i>
<i>Ilustración 84 Switch hub para la conexión de los cables de red</i>	<i>192</i>
<i>Ilustración 85 Conexiones eléctricas modificadas.....</i>	<i>192</i>
<i>Ilustración 86 Canaleta y conexiones de control.....</i>	<i>192</i>
<i>Ilustración 87 Switch de red conectado ordenado los cables ya no están expuestos.....</i>	<i>193</i>
<i>Ilustración 88 Interface grafica</i>	<i>193</i>
<i>Ilustración 89 estructura del LOGO!.....</i>	<i>194</i>
<i>Ilustración 90 Indicadores luminosos desactivado en interface grafica</i>	<i>195</i>
<i>Ilustración 91 Visualización de las entradas en, pantalla del LOGO!.....</i>	<i>195</i>
<i>Ilustración 92 Indicador luminoso activada en interface gráfica.....</i>	<i>195</i>
<i>Ilustración 93 Entrada activa en pantalla del LOGO!.....</i>	<i>195</i>
<i>Ilustración 94 Indicador luminoso desactivado en interface gráfica y en pantalla del LOGO!.....</i>	<i>196</i>
<i>Ilustración 95 Indicador luminoso activado en interface gráfica y pantalla del LOGO!</i>	<i>196</i>
<i>Ilustración 96 Interruptores e indicadores luminosos en interface gráfica desactivados y activados</i>	<i>196</i>
<i>Ilustración 97 Salidas activas desde la interface gráfica</i>	<i>196</i>
<i>Ilustración 98 Iconos de ajustes</i>	<i>197</i>
<i>Ilustración 99 Configuración de IP</i>	<i>197</i>
<i>Ilustración 100 Configuración de red</i>	<i>198</i>
<i>Ilustración 101 Información del grupo.....</i>	<i>198</i>
<i>Ilustración 102 Mensaje cuando no está conectado al dispositivo.....</i>	<i>199</i>
<i>Ilustración 103 Mensaje conectada</i>	<i>199</i>
<i>Ilustración 104 iconos ocupadas en la interface gráfica.....</i>	<i>211</i>

LISTA DE TABLAS

<i>Tabla 1 Pila de protocolo TCP / IP</i>	<i>101</i>
<i>Tabla 2 Cabecera MB</i>	<i>108</i>
<i>Tabla 3 Modelo de bloques de almacenamiento de datos en Modbus.</i>	<i>115</i>
<i>Tabla 4 Extracto de los códigos de funciones públicos.</i>	<i>116</i>
<i>Tabla 5 PDU del Request para el código de función 3</i>	<i>117</i>
<i>Tabla 6 PDU del Response para el código de función 3</i>	<i>117</i>
<i>Tabla 7 PDU de Excepción para el código de función 3</i>	<i>117</i>
<i>Tabla 8 PDU del Request para el código de función 15</i>	<i>118</i>
<i>Tabla 9 PDU del Response para el código de función 15</i>	<i>118</i>
<i>Tabla 10 PDU del Response para el código de función 15</i>	<i>118</i>

CAPITULO 1. GENERALIDADES.

1.1 INTRODUCCIÓN

Desde los inicios de la era industrial, la temperatura ha servido para indicar el estado de las máquinas, equipos, conductores etc. Los técnicos saben que una temperatura anómala, a menudo, es indicativa de un posible problema. Dependiendo de la situación, a veces basta con tocar la máquina con el dorso de la mano para saber si la temperatura se encuentra dentro de los límites aceptables. Naturalmente, este método tiene importantes limitaciones de seguridad, especialmente en el caso de la maquinaria giratoria y los equipos eléctricos.

Se han creado protocolos de comunicación para que puedan transmitir información entre los dispositivos de automatización, facilita la interacción entre ellos.

El protocolo más utilizado en la automatización es modbus, gracias a su simplicidad y facilidad de implementación, y se puede utilizar en diferentes estándares de entorno físico.

La velocidad de comunicación varía en cada uno de estos estándares, así como la longitud máxima de la red y la cantidad máxima de dispositivos conectados, es ampliamente utilizado en la industria y es sin duda uno de los estándares más utilizados es el protocolo ModBus. Este estándar permite trabajar con velocidades de comunicación que pueden alcanzar los 12Mbps y en algunos casos hasta 50Mbps, vale la pena recordar que cuanto mayor es la longitud de la red, menor es la velocidad de comunicación, la distancia máxima de la red es de alrededor de 1200m, y el número máximo de dispositivos en el bus de red es 32.

1.2 OBJETIVOS

1.2.1. General:

Crear módulo didáctico basado en protocolo de comunicación industrial Modbus con enlace Python con el fin de evaluar su viabilidad, eficiencia y aplicaciones en entornos industriales modernos para sistemas SCADA.

1.2.2. Específicos:

- Desarrollar código en software de Python para la implementación con el protocolo Modbus TCP/IP que permita la comunicación con dispositivos PLC y su integración con sistemas SCADA lo cual permita su implementación en la materia de Diseño electrónico.
- Investigar las diferentes librerías de comunicación Modbus TCP/IP, ahondando y evaluando su facilidad de uso, compatibilidad con diversos sistemas industriales y capacidades funcionales.
- Identificar las ventajas y limitaciones de la comunicación Modbus para su aplicación efectiva en entornos SCADA y de automatización.
- Diseñar una interfaz de usuario en Python que permita la visualización y el control de dispositivos industriales a través de la red Modbus, demostrando la integración con un sistema SCADA.
- Elaborar guías prácticas que proporcione las competencias necesarias y detalladas sobre las aplicaciones y el funcionamiento de las redes de comunicación industrial en sistemas SCADA, utilizando el protocolo Modbus y ser usadas en las materias de Autómatas Programables y Procesamiento Digital de señales.

1.3 DESCRIPCIÓN DEL PROBLEMA

En la actualidad con la globalización de tecnología y la necesidad de tener un mejor control en los procesos de fabricación o un servicio de productos, la mayor parte de las empresas usan dispositivos que cumplan normas y calidad para mejorar sus procesos en la industria lo que conlleva a la instalación de varios dispositivos de control, se hace necesario tener una forma que mejore la conexión que sea fácil y seguro su instalación, para poder cumplir con esta necesidad sea creado un protocolo de comunicación que cumple con la robustez y calidad para soportar un ancho de banda que el solicitante necesita para ejecutar su labor. Por lo cual es necesario que los profesionales o técnicos conozcan la forma de cómo poder implementar este protocolo de comunicación. La universidad Técnica Latinoamericana en su función de formación, debe poseer equipo que proporcione las competencias para lograr que sus profesionales estén a la vanguardia en el uso de tecnología para interpretar mejor la realidad que la globalización demanda y que en sus prácticas de laboratorio puedan enlazar a dispositivos usando el protocolo industrial ModBus y sistemas SCADA.

Ya que la carrera de Ingeniería Electrónica está ligada a la tecnología, los estudiantes deben obtener las competencias necesarias con respecto a los contenidos que involucren los protocolos de comunicación, con lo cual en esta investigación se pretenden que los estudiantes aumenten sus competencias en el área de Diseño Electrónico, Sistemas Digitales, Autómatas Programables.

1.4 JUSTIFICACIÓN

Dada la creciente importancia de la automatización y la monitorización en entornos industriales, es crucial explorar y comprender a fondo las tecnologías de comunicación, como el protocolo Modbus, y su implementación en sistemas de supervisión y control como SCADA. Al utilizar Python, un lenguaje de programación versátil y de fácil aprendizaje, esta investigación busca proporcionar una solución eficiente y accesible para la implementación de redes de comunicación industrial. Al comprender mejor el funcionamiento y las aplicaciones de esta tecnología, se espera contribuir al avance y la optimización de los sistemas industriales modernos, mejorando su eficiencia, fiabilidad y seguridad.

Con esta investigación y la fabricación de un equipo se pretende aumentar las competencias de los estudiantes en áreas como:

- Diseño Electrónico
- Sistemas Digitales
- Autómatas Programables.

1.5 ALCANCES Y LIMITANTES

1.5.1. Alcances.

El trabajo de Tesis **APLICACIONES Y FUNCIONAMIENTO DE RED DE COMUNICACIÓN INDUSTRIAL PARA SISTEMAS SCADA POR PROTOCOLO MODBUS**, está efectuado para el pleno y educativo de la Universidad Técnica Latinoamericana, para la realización de prácticas con microcontroladores y sensores de variables digitales y analógicas, que dichas señales puedan comunicarse por medio del protocolo ModBus a elementos ya sea como maestro o esclavo depende de la configuración en el PLC y Logo! que usan normas industriales. Los dispositivos de adquisición son diversificados, e incluso controlados por parte de software.

1.5.2. Limitantes.

El trabajo de investigación está diseñado para el uso y desarrollo con el protocolo de comunicación ModBus mediante un sistema de adquisición de datos SCADA y conectada a una red industrial, que procesará información de comunicación a PLC y Logo, con lo cual se dará enfoque y generará competencias en las materias de Diseño Electrónico, Sistemas Digitales y Autómatas Programables.

El material y el modulo proporcionado está dispuesto para la realización de las guías con dos áreas de trabajo.

1.6. ORGANIZACIÓN DEL PROYECTO DE GRADUACIÓN

A continuación, se da una breve descripción de lo que trata cada uno de los de los capítulos que componen este trabajo de graduación:

1.6.1. Capítulo 1 Generalidades

En este capítulo se detalla las generalidades del proyecto como son los objetivos, la justificación, los alcances y limitaciones que describen la importancia de trabajos como este.

1.6.2. Capítulo 2 Marco Teórico

Se describe los conceptos básicos de dispositivos eléctricos y electrónicos como pueden ser Raspberry Pi, PLC, LOGO V8 que pueden tener integrados en módulos o puertos para establecer una comunicación entre varios dispositivos, también se conocerá los principales protocolos que existen de comunicación a nivel industrial, se comprenderá su estructuras y sus diferentes tipos de conexión, además se da una breve explicación del funcionamiento del lenguaje de programación que se ha utilizado para desarrollar este proyecto, el lenguaje es Python.

1.6.3. Capítulo 3 Protocolos de comunicación, modelo OSI, TCP/IP y modelo de datos modbus.

Se describe con detalle los protocolos de comunicación, así como el estándar del modelo OSI a través de sus diferentes capas que los componen, también se habla de la arquitectura del protocolo TCP/IP. Y los diferentes tipos de protocolos que existen.

1.6.4. Capítulo 4 Guías de trabajo del laboratorio de protocolo Modbus de la Utl.

Se elaboran diferentes guías para poder abordar los temas de manejo de Raspberry Pi, logo8, y fundamentos del protocolo Modbus. Para tener un soporte en las materias Diseño Electrónico II, Ingeniería de Sistemas de Control y Autómatas Programables, de la carrera de Ingeniería Electrónica, manteniendo la estructura del formato de laboratorio de la Universidad Técnica Latinoamericana. El desarrollo de las guías tiene su complementación con el equipo de los dispositivos normados que posee la universidad en su laboratorio de práctica.

1.6.5. Capítulo 5 Conclusiones del proyecto.

Se incluyen las conclusiones finales del trabajo.

1.6.6 Capítulo 6 Referencias bibliográficas

Las referencias bibliográficas son esenciales para asegurar la integridad, la claridad y la calidad en la investigación, y son un elemento clave en la comunicación de ideas y descubrimientos en el ámbito académico.

1.6.7 Capítulo 7 Anexo Construcción del módulo.

Se detallan el diseño y construcción del módulo para la realizar prácticas de laboratorio y poder implementar, aplicaciones diversas para la comunicación a través del protocolo Modbus, donde se realizarán diferentes pruebas con los dispositivos Raspberry Pi que será el encargado de controlar un logov8 siemens, mostrando de forma correcta del uso de la comunicación Modbus.

1.7 METODOLOGÍA DE LA INVESTIGACIÓN

Los métodos utilizados para llevar a cabo la investigación son aplicados y experimentales. Y las técnicas que se utilizaron para recolectar datos fueron:

1.7.1. La investigación documental:

Utilizamos esta técnica por la recopilación de información donde recolectamos, seleccionamos, analizamos y presentamos los enunciados teóricos que sustentan el estudio del proyecto

1.7.2. La investigación bibliográfica:

Son las fuentes donde se han realizado las búsquedas de la información teórica-practica en varios medios como los digitales o en las bibliotecas etc.

CAPITULO 2 MARCO TEORICO

2.1 ANTECEDENTE HISTÓRICOS

La historia de los protocolos de comunicación está relacionada a la historia de los PLC. El PLC (Controlador Lógico Programable) apareció con el propósito de eliminar el enorme costo que significaba el mantenimiento de un sistema de control electromecánico (basado en relés) a finales de los años 60.

Para permitir la comunicación entre los dispositivos de automatización se han desarrollado diversos protocolos de comunicación industriales, estos protocolos se remontan a 1968 cuando la Empresa Bedford Associates invento el Modicon 084 (Modular Digital Controller) siendo el primer controlador sustituyendo a la lógica cableada basada en relé. El protocolo de comunicación más utilizado y se sigue utilizando es el protocolo modbus, pero no quiere decir que es el único, con el paso de los años se han desarrollado diversos protocolos que son los siguientes: Hart, Foundation Fielbus, Profibus, Devicenet, Contronet, Ethernet IP, AsiBus, Profinet, Ethercat, Bacnet, Canopen, etc.

El protocolo Modbus se desarrolló por el año de 1973. El primer sistema que lo hizo servir fue Modicom que permitió eliminar casi todo el cableado de control entre maquinas. A través de éste protocolo se podría coordinar el funcionamiento de todos los componentes de un sistema, programar a distancia, supervisar remotamente, diagnosticar todos los elementos conectados.

Raspberry Pi es una placa de microordenador, que apareció en febrero de 2012, La primera unidad tenía 256 MB de RAM y un procesador a 700 MHz, tenía el característico conector de 26 pines GPIO y salida de video por HDMI o RCA además de un conector de 3.5mm para el audio, el primer modelo carecía de puerto Ethernet.

Raspberry Pi es una placa sencilla, pero suficiente para hacer las tareas más comunes de cualquier ordenador, la informática está al alcance de todos los usuarios e incluso bolsillos. El

microordenador Raspberry Pi ha pasado por varias versiones de mejora, para funcionar utiliza un sistema operativo, Raspbian, basado en Linux y especial para aprovechar el hardware de este microordenador. Además de Raspbian, multitud de sistemas han aparecido que hacen de la Raspberry Pi un microordenador para diferentes fines.

En 1996 Siemens lanzó al mercado un pequeño controlador lógico que supuso toda una revolución en el automatismo tradicional, permitiendo el salto de la lógica cableada a la programada, LOGO8. Es un módulo lógico, es decir, un controlador programable que permite que, sin intervención humana, las máquinas hagan un trabajo. Por tanto, es necesario programarlo para que este haga una tarea, tomando como datos de entrada una serie de señales, y el traduce de pulsadores, manetas, sensores, etc. para un procesamiento en el que como resultado dará una activación o no, de salidas de relé (datos de salida).

2.2. DESCRIPCIÓN.

La mayoría de los sistemas digitales con equipo de pruebas generan y/o adquieren patrones de 1's y 0's para comunicarse con un dispositivo de prueba. Sin embargo, con innovaciones recientes en componentes digitales, estos sistemas requieren un probador digital más sofisticado con habilidades más avanzadas que las de un analizador lógico con dos estados de encendido/apagado. Las velocidades, ahora más altas de los chips y la tendencia de la industria hacia protocolos digitales seriales en lugar de paralelos, requieren las más altas velocidades de muestreo. Además, las presiones de manufactura y de tiempo al mercado están demandando procesamiento a nivel de hardware para completar pruebas más rápido. Mientras tanto, las nuevas familias lógicas, con niveles de voltaje variables y generación de señales diferenciales o comunes, continúan haciendo estas pruebas más complejas. Los dispositivos de Entrada/Salida digital de alta velocidad ofrecen

una gama de estímulo-respuesta diseñadas para cubrir las necesidades de estas aplicaciones de pruebas digitales de alta calidad.

2.3. CONCEPTUALIZACIÓN.

Las señales analógicas y digitales son dos sistemas completamente distintos, pero que a su vez están estrechamente relacionados, las señales analógicas son aquellas donde el voltaje varía en función del valor de una muestra. Las señales digitales pueden tener dos estados bien definidos en un momento dado, High (1) y Low (0), es decir, encendido o apagado. Cuando se trabaja con señales digitales no se incluye rangos de variaciones en el voltaje que se esté aplicando, debido a que solo se toma la existencia o no del voltaje usado. En la actualidad los dispositivos de entrada y salida disponen de señales analógicas y digitales que proporcionan un control preciso por hardware, el cual es esencial para los controladores programables, y dispositivos de comunicación digital. El control bidireccional de datos y la comparación de hardware en tiempo real de la respuesta, facilitan el desarrollo de aplicaciones tales como pruebas de razón de error, análisis de fallas para verificación y validación entre otras.

Los niveles de voltaje crean un sistema digital flexible que puede establecer interfaces con múltiples familias lógicas, o caracterizar los límites inferiores y superiores de dispositivos bajo prueba específicas, incluyendo lógico bajo (0), lógico alto (1), triestado (Hi-Z), definiendo las formas de onda de pruebas digitales y controlan las acciones de un acondicionador digital para las señales físicas a ser tratadas.

Las aplicaciones industriales generalmente involucran requisitos fuera del alcance de un dispositivo típico, muchos sensores y actuadores industriales requieren niveles lógicos de 24 V y pueden operar a diferentes potenciales de voltaje. La seguridad para el sistema de medida o control y la seguridad para el usuario u operador son igualmente importantes. Son ideales para control

industrial y para sistemas de pruebas de manufactura, tales como automatización de fábricas, control embebido de máquinas y verificación de línea de producción. Además, los dispositivos de E/S (entrada/salida), digital e industrial regidas bajo el funcionamiento de un microcontrolador proporcionan un conjunto de características industriales de alta fiabilidad diseñadas para automatizar hasta las aplicaciones más demandantes.

Los dispositivos que sirven como adquisición de datos están preparados para la recepción y transmisión de datos hacia elementos normados que cumplan los estándares industriales, en el entendido que los factores involucrados en ese entorno deben ser superados, entre ellos se encuentran:

- El aislamiento para la reducción del ruido, que ofrece un rango de voltaje extendido y protege el hardware para conexión directa a sensores y actuadores industriales.
- Los estados de encendido programables proporcionan un estado de inicio conocido para una operación segura al estar conectado a bombas, motores y otros actuadores o maquinaria industriales.
- La detección de cambios, con uso mínimo del procesador, dispara automáticamente su aplicación para que realice una operación digital de lectura después de un cambio de estado en una línea digital.
- Los filtros de entrada programables eliminan ruido, rebotes y picos en entradas y proporcionan eliminación de rebotes para conmutadores digitales y relés.
- Las certificaciones industriales de CE, FCC, C-Tick, UL y VDE aseguran la compatibilidad con EMI en la mayoría de las regiones del mundo y certifican que el sistema es seguro para operar.

2.4. CONEXIÓN DE INTERFAZ.

Los elementos de entrada/salida para usos industriales son parte de la gama de productos industriales que incluye dispositivos multifunción que complementan la plataforma de un controlador de automatización programable, que se integra perfectamente con aplicaciones de lógica, movimiento, control de procesos y visión.

Para que los dispositivos de entrada/salida (E/S o I/O), lleven a cabo la tarea de comunicarse con los elementos normados, deben usar las señales adecuadas del estándar, y prácticamente relacionarse únicamente con las líneas de conexión que se hacen llegar a los puertos de entrada o salida de los dispositivos de cliente servidor el cual se enlazara con la ayuda del programa Python.

2.4.1. Dispositivos de entrada.

Los dispositivos de entrada 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.

2.4.1.1. Pulsador

Es un dispositivo eléctrico que permite o impide el paso de la corriente eléctrica momentáneamente, cuando se aprieta o pulsa. El pulsador solo se abre o se cierra cuando el usuario lo presiona y lo mantiene presionado. Al soltarlo vuelve a su posición inicial. El pasó o cierre de la corriente se consigue mediante contactos eléctricos, también llamados bornes, que normalmente son de cobre. Cada contacto eléctrico del pulsador tiene 2 posiciones, abierto y cerrado.

- **Cerrado:** Los 2 bornes están juntos y el pulsador permite el paso de la corriente eléctrica.
- **Abierto:** los 2 bornes están separados y el pulsador corta o no permite el paso de la corriente eléctrica.

El símbolo que representa el pulsador es diferente si es normalmente cerrado o normalmente abierto.

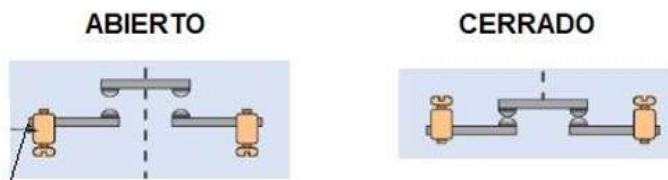


Ilustración 1 Contacto pulsador

2.4.1.2 El interruptor

Es un dispositivo eléctrico que nos permite realizar una función de ON / OFF desde un mando. Su funcionamiento consiste en dejar pasar o no la corriente en un circuito eléctrico. Por ello, su función principal es el encendido y apagado. Mecánicamente un interruptor consta de dos contactos metálicos separados y una parte móvil actuante que suele ser un balancín. Presionando la tecla movemos el balancín y abrimos o cerramos el circuito.



Ilustración 2 Interruptor

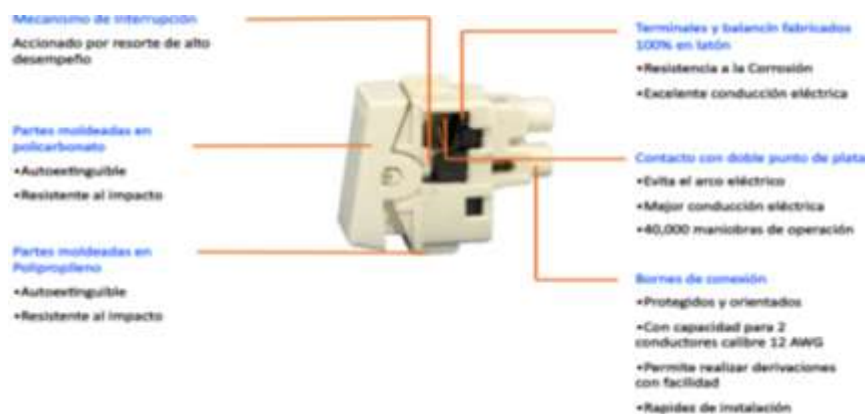


Ilustración 3 Característica del interruptor

2.4.1.2.1 Tipos de interruptores.

Los principales tipos de interruptores que podemos encontrar son los siguientes:

- **Unipolares:** Conectan / desconectan una línea viva de la carga, desde un punto
- **Bipolares:** Conectan / desconectan dos líneas vivas de una carga, desde un punto.



Ilustración 4 Interruptor unipolar y bipolar

2.4.2 Dispositivos de salida.

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

2.4.2.1 Relé.

El aislamiento está provisto únicamente por interruptores individuales sin que exista contacto entre un sistema de una sección con otro, para ello se usa un relé, que es un dispositivo electromecánico utilizado para conmutar circuitos. Es, básicamente, un interruptor controlado por un electroimán. Cuando se somete el electroimán a una corriente se genera un campo magnético que mueve uno o varios contactos permitiéndole al dispositivo abrir o cerrar un circuito.

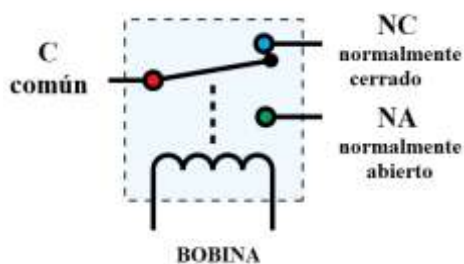


Ilustración 5 Símbolo del relé.

La bobina que se puede apreciar y se debe someter a una tensión que es específica para cada dispositivo, al suceder esto se genera un campo electromagnético, convirtiendo al relé en un electroimán que, por acción de la fuerza del campo atrae contactos metálicos, dispuestos con un

juego de resortes. Al desaparecer dicho campo por suspensión de la energía en su bobina, los contactos regresan a su posición original por el efecto del juego de resortes. Esta operación convierte al relé en un interruptor activado por corriente y a los contactos metálicos en sus terminales.

La configuración de sus pines aparece en la carcasa de los relés, donde se puede encontrar esta información.



Ilustración 6 Presentación del relé y configuración de pines

2.4.2.2 Contactor.

El Contactor es un dispositivo eléctrico que puede cerrar o abrir circuitos en carga o en vacío en los que intervengan cargas de intensidad que pudieran producir algún efecto perjudicial para quien lo accione como por ejemplo en maniobras de apertura y cierre de instalaciones de motores.

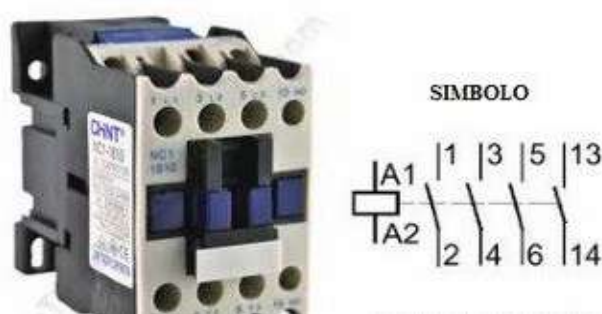


Ilustración 7 Presentación del relé y configuración de pines

2.4.2.2.1 Conceptos básicos.

Los contactores son dispositivos de conmutación eléctricos, similares a un relé. Los contactores pueden manejar niveles de corriente mucho más altos que los relés estándar, lo que los hace útiles para aplicaciones de equipos móviles. El contactor funciona aplicando un voltaje a la bobina, esto

crea un campo magnético que mueve los contactos a la posición cerrada y completa el circuito. Una vez que se elimina el voltaje de la bobina, los contactos se abrirán nuevamente y desconectarán el circuito.

2.4.2.3. Electroválvula.

Las electroválvulas son dispositivos que responden a pulsos eléctricos. Gracias a la corriente que circula a través del solenoide es posible abrir o cerrar la válvula controlando, de esta forma, el flujo de fluidos. A la circular corriente por el solenoide se genera un campo magnético que atrae el núcleo móvil y al finalizar el efecto del campo magnético, el núcleo vuelve a su posición, en la mayoría de los casos, por efecto de un resorte.

Las electroválvulas se utilizan en gran número de sistemas y rubros industriales que manejan fluidos como el agua, el aire, el vapor, aceites livianos, gases neutros y otros. En particular, las electroválvulas suelen implementarse en lugares de difícil acceso ya que pueden ser accionadas por medio de acciones eléctricas. También son utilizadas en vacío o hasta en altas presiones y altas temperaturas.



Ilustración 8 Electroválvulas

Las electroválvulas se aplican a surtidores automáticos de combustibles, irrigación de parques, fuentes de agua danzantes, dosificadores de líquidos o gases, regulación de niveles de líquidos, en máquinas envasadoras, lavaderos automáticos de autos, máquinas de limpieza, procesos de niquelado o galvanizado, en máquinas de café y en muchos lugares más.

2.4.2.4. Indicador luminoso.

Una luz fija emite una señal luminosa permanente, sin interrupciones ni cambios de la intensidad luminosa. Las luces permanentes suelen indicar el estado de un dispositivo o una aplicación. Los colores rojo y verde se utilizan a menudo para este fin. El color verde se utiliza para señalar el funcionamiento o el estado normal de una máquina o aplicación. Del mismo modo, una luz verde puede significar el libre acceso, como en el principio del semáforo, tal y como se utiliza, por ejemplo, en los puertos de carga.



Ilustración 9 Indicador luminoso

2.5. RASPBERRY PI

Es una computadora de formato compacto destinada para hacer accesible la informática a todos los usuarios. La Raspberry Pi también se caracteriza por ser muy utilizada para desarrollar pequeños prototipos y para la formación sobre informática y electrónica. Todos los diseños de Raspberry Pi se basan en el hardware libre y habitualmente se utilizan también sistemas operativos libres basados en GNU/Linux. Para este microordenador se ha desarrollado Raspberry Pi OS (antes conocido como Raspbian) que es una versión personalizada de Debían. Adicionalmente se pueden instalar diferentes sistemas operativos.



Ilustración 10 Raspberry Pi

La primera Raspberry Pi fue lanzada al mercado en febrero de 2012 de la mano de la Raspberry Pi Foundation. Esta organización nace con la idea de promover y enseñar informática en los centros educativos y universidades de Reino Unido. Para hacerlo lo más accesible posible, desarrollaron este pequeño ordenador de bajo coste y grandes posibilidades. Precisamente es en el buen rendimiento en donde radica el gran éxito de la Raspberry Pi. Cuentan además con una gran conectividad y de conexiones GPIO que permiten desarrollar una gran variedad de proyectos educativos.

2.5.1. Características

Los conectores de esta placa son los que se muestran en la siguiente imagen:

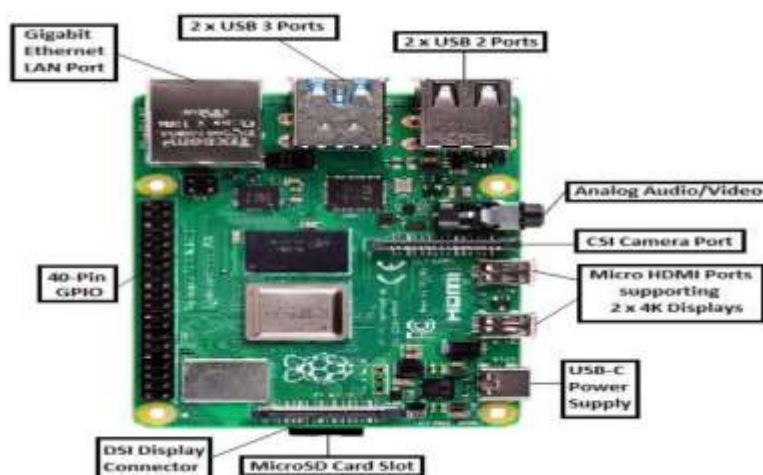


Ilustración 11 Característica

2.5.2. Cómo funciona una Raspberry Pi

El funcionamiento es el mismo que un ordenador de sobremesa, ya que a nivel fundamental tiene los mismos componentes. Las placas Raspberry Pi se basan en un SoC de arquitectura ARM

de bajo consumo y buen rendimiento. Se acompañan por un modelo de memoria RAM, cuya capacidad varía según el modelo. Disponen de varias salidas de vídeo y en las versiones más nuevas se agregan un conector Jack de 4 polos, para entrada de micro y salida de audio. Cuentan con un lector de tarjetas donde se instala el sistema operativo y varios puertos USB. También dispone de una gran cantidad de conectores GPIO para poder desarrollar una gran cantidad de proyectos.

La **Raspberry Pi 4**, que es la versión más moderna cuenta con un conector Gigabit Ethernet y una tarjeta WiFi + Bluetooth integrada. Este modelo más moderno tiene un conector de alimentación del tipo USB-C y dos conectores MIPI, uno para un display y el otro para una cámara.

Para hacerlo funcionar debemos alimentar la placa mediante el conector específico. Debemos cargar el sistema operativo que queramos en la tarjeta de memoria SD, para poder disfrutar de toda su funcionalidad. Tendremos que conectar teclado y ratón, así como una pantalla de visualización de datos para poder interactuar con la placa.



Ilustración 12 Raspberry Pi Pía 4

2.5.3. Modelos de Raspberry Pi

A continuación, dividiremos las Raspberry Pi en los modelos enfocados a sus funcionalidades.

2.5.3.1. Versión Pi Model B

Son las placas de tipo completo que **incluyen puertos Ethernet y puertos USB**. Se podrá encontrar las variantes B+, donde el «+» nos indica variaciones o actualizaciones menores. Las versiones más modernas de estas placas agregan conectividad WiFi y Bluetooth.

2.5.3.2. Versión Pi Model A

Estas placas suelen ser un poco más compactas que las «Model B» y suelen tener un formato cuadrado. Podemos decir que **son versiones sencillas** de las placas Raspberry Pi, ya que tienen limitaciones en conectividad. Las «Model A» suelen tener **solo un puerto USB y carecen de puerto Ethernet**, y son más económicas que las «Model B». Las versiones más modernas de estas placas agregan conectividad WiFi y Bluetooth.

2.5.3.3. Versión Pi Zero

Este tipo de placas tienen un formato súper reducido de formato rectangular y tienen un consumo energético realmente reducido. Todos los modelos disponen de un micro USB y un micro HDMI. Salvo la primera Raspberry Pi Zero, las demás versiones cuentan con conectividad WiFi y Bluetooth.



Ilustración 13 Raspberry Pi Zero

2.5.3.4. Versión Pi 400

Posiblemente la versión más particular que han lanzado de todas las variantes. Tenemos una Raspberry Pi embebida en un teclado, lo cual es una solución muy interesante si queremos tener un ordenador compacto. Dispone de un procesador potente de 4 núcleos, 4GB de memoria RAM, conectividad WiFi + Bluetooth y dos puertos USB 3.0, un puerto USB 2.0, dos microHDMI

e incluso un puerto Gigabit Ethernet. La Raspberry Pi que integra tiene un diseño único específico para ser embebida dentro del teclado.



Ilustración 14 Raspberry Pi 400

2.5.3.5. Versión Pi Pico

No es un ordenador compacto, es más bien un microcontrolador que se basa en el procesador RP2040, desarrollado por la propia compañía. Está pensado para ser usado sobre todo para proyectos del tipo DIY (Do It Yourself, hágatelo tú mismo). Carece de cualquier tipo de puerto de conexión y también de conectividad inalámbrica.



Ilustración 15 Raspberry Pi pico

2.6. CONTROLADOR LÓGICO PROGRAMABLE.



Ilustración 16 PLC industrial

Un autómatas programable es un equipo electrónico que realiza procesos secuenciales a nivel industrial. Monitoriza entradas y salidas (input/output) y toma decisiones lógicas para automatizar procesos o máquinas.

Estos dispositivos cumplen con normas estándares industriales, son componentes robustos que, en algunos casos, pueden aguantar condiciones severas: calor, frío, polvo y humedad extrema. Su lenguaje de programación se entiende fácilmente, por lo que se pueden programar sin mucha dificultad. Son modulares con lo que permiten añadir módulos (E/S digitales, Analógicas, comunicación, pesaje o temperatura), en caso de necesidad.

Están constituidos por partes importantes que los hacen tan versátiles para todas las funciones que cumplen, las más importantes son:

- CPU o procesador: el procesador principal (Unidad Central de Procesamiento o CPU) es un sistema basado en un microprocesador que ejecuta el programa de control después de leer el estado de las entradas de campo y envía comandos a las salidas de campo.
- Sección de E / S: Los módulos de E / S actúan como Interfaz de Datos Reales entre el campo y la CPU. El autómatas conoce el estado real de los dispositivos de campo y controla los dispositivos de campo mediante las tarjetas de E / S correspondientes.
- Dispositivo de programación: se puede conectar una tarjeta de CPU con un dispositivo de programación a través de un enlace de comunicación mediante el puerto de programación de la CPU.
- Estación de operación: Una estación de operación se utiliza comúnmente para proporcionar una visualización al proceso o procesos. Generalmente es un dispositivo separado conectado por comunicaciones a las CPU's de los autómatas.

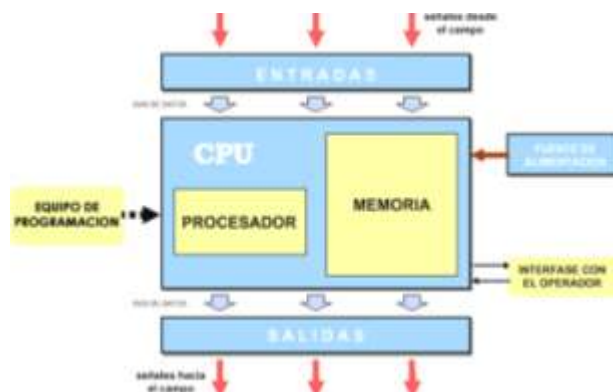


Ilustración 17 Bloque simplificado del PLC

Los autómatas programables, como ya se mencionó, son dispositivos digitales electrónicos con una memoria programable para el almacenamiento de instrucciones, permitiendo la implementación de funciones específicas que pueden ser: lógicas, secuenciales, temporizadas, de conteo y aritméticas; con el objeto de controlar máquinas y procesos. Realiza la ejecución de un programa de forma cíclica, que 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.

2.6.1. Logo!

Es un módulo lógico inteligente destinado para aplicaciones domésticas y de la ingeniería de instalación; También puede ofrecer soluciones para ingeniería de armarios de distribución, así como para ingeniería mecánica y construcción de máquinas.

LOGO! también se utiliza para implementar sistemas de control especiales en invernaderos, para el procesamiento de señales de control y, mediante la conexión de un módulo de comunicaciones, para el control distribuido local de máquinas y procesos; Para aplicaciones de producción en serie de máquinas pequeñas, aparatos y armarios eléctricos (automatismos).

Algunos ejemplos más concretos pueden ser:

- Puede encender una lámpara en intervalos regulares, o bien subir y bajar las persianas mientras está de vacaciones.
- Calefacción central: LOGO! hace que la bomba de circulación funcione sólo si se necesitan realmente agua o calor.
- Sistemas de refrigeración: LOGO! puede descongelar sistemas de refrigeración en intervalos regulares para ahorrar costes de energía.
- Es posible alumbrar cualquier habitáculo en función del tiempo, incluso alumbrado externo según luminosidad.
- El control de cualquier automatismo ahorrando mucho cableado.

2.6.1.1 ¿Cómo Funciona el LOGO?

Toda la programación se realiza de una forma bastante sencilla con las 6 teclas que están situadas en su panel frontal y desde las que podemos introducir el programa. La visualización del programa, estado de entradas y salidas, parámetros, etc., se realiza en una pequeña pantalla LCD de forma gráfica. Otra forma de trabajo es creando el programa en un ordenador externo mediante el software especial que es gratuito de SIEMENS y posteriormente introducir el programa en el LOGO, conectando el ordenador con LOGO, mediante un cable que distribuye la propia SIEMENS. Incluso se puede simular el programa en el ordenador antes de introducirlo en el LOGO.

Este mini PLC está implantado en todo el mundo y puede controlarse a distancia. Tiene tareas de automatización muy sencillas de programar mediante un software intuitivo (LOGO! Soft Comfort) de fácil manejo.

A continuación, puedes ver las partes del LOGO! de Siemens:

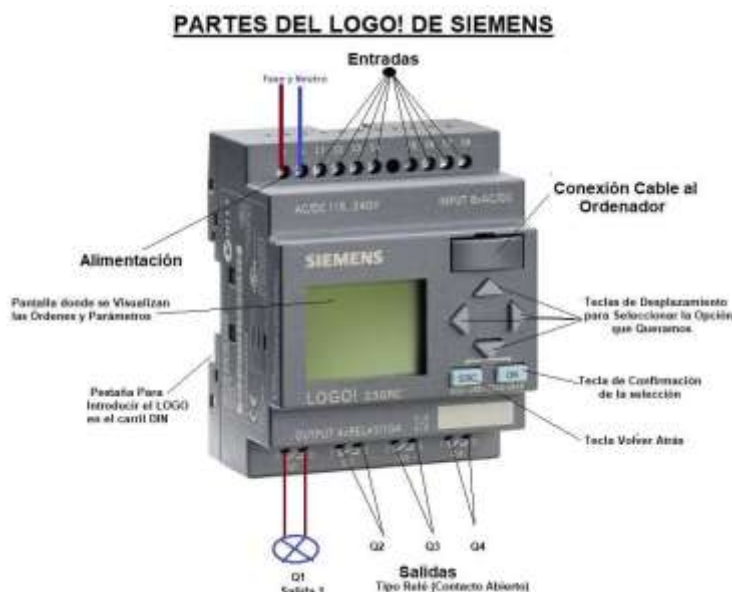


Ilustración 18 Partes del logo

En sus inicios LOGO! de Siemens se utilizó para reemplazar los relés cableados; solo permitía la programación de 30 bloques y disponía 8 funciones básicas y 8 funciones adicionales. Pero con el tiempo y, debido a sus nuevas características se fue ampliando su campo de aplicación con cada actualización y sus capacidades de comunicación. Ahora permite programar 400 bloques y tiene más de 40 funciones.

Al fin y al cabo, este PLC junta los diferentes componentes utilizados con anterioridad por separado (relés, contactores auxiliares, temporizadores y controladores) en un único módulo, reduciendo el espacio de armario y el engorroso cableado del pasado.

Además, es compatible con el protocolo Ethernet Industrial y dispone de un servidor incorporado de gran utilidad en aplicaciones de IIoT. Gracias a la interfaz Ethernet integrada, puede conectar los módulos lógicos con otros componentes de la serie LOGO! de Siemens y formar redes de comunicación avanzadas.

En su última versión (LOGO! 8.3) permite conexión a la nube procesando grandes volúmenes de datos, sin la dependencia de la tarjeta SD integrada en anteriores versiones de LOGO! que se van a poder almacenar en la nube.

El LOGO! por sus especiales características de diseño tiene un campo de aplicación muy extenso. La constante evolución del hardware y software amplía constantemente este campo, para poder satisfacer las necesidades que se detectan en el espectro de sus posibilidades reales.

Su utilización se da fundamentalmente en aquellas instalaciones en donde es necesario un proceso de maniobra, control y señalización. Por tanto, su aplicación abarca desde procesos de fabricación industriales de cualquier tipo a transformaciones industriales, o control de instalaciones, entre otras.

Sus reducidas dimensiones, la extremada facilidad de su montaje, la posibilidad de almacenar los programas para su posterior y rápida utilización, la modificación o alteración de los mismos, hace que su eficacia se aprecie principalmente en procesos en que se producen necesidades de reducir tiempo empleado en la elaboración de proyectos, debido a que no es necesario dibujar previamente el esquema de contactos, es preciso simplificar las ecuaciones lógicas, ya que por lo general la capacidad de almacenamiento del módulo de memoria es lo suficientemente grande.

La lista de materiales queda sensiblemente reducida, y puede dar la posibilidad de introducir modificaciones sin cambiar el cableado ni añadir aparatos, reduciendo el costo de mano de obra de la instalación debido al mínimo espacio del tablero donde se instala el autómata programable, además de aumentar la fiabilidad del sistema, al eliminar contactos móviles, los mismos autómatas pueden indicar y detectar averías.

Para el PLC una excelente ventaja es la posibilidad de gobernar varias máquinas con un mismo autómata, si por alguna razón la máquina queda fuera de servicio, el autómata sigue siendo útil para otra máquina o sistema de producción.

Como inconvenientes se puede hablar, en primer lugar, el costo inicial, en segundo, que hace falta un programador, lo que obliga a adiestrar a uno de los técnicos en tal sentido. Esta capacitación puede ser tomada en distintos cursos, inclusive en universidades.



Ilustración 19 Autómata s7-1500

2.7. DISPOSITIVOS DE COMUNICACIÓN INALÁMBRICA.

Para lograr que dos dispositivos se comuniquen se necesita un método de comunicación y un lenguaje o protocolo común entre ambos dispositivos. La forma más común de establecer lo antes mencionado es utilizando la comunicación serie, que consiste en transmisión y recepción de pulsos digitales a una misma velocidad. El transmisor envía pulsos que representan el dato y el receptor se sincroniza a esa misma velocidad e intercambian información, como ejemplo, si la velocidad es de 9600 bits por segundo, el receptor capturaré el voltaje que le está enviando el transmisor cada $1/9600$ de segundo, interpretará dicho voltaje como un nuevo bit de datos. Para ello se usa el código ASCII como estándar serie, que representa caracteres alfanuméricos.

La comunicación entre sistemas diferentes siempre ha estado en constante estudio por parte de los circuitos electrónicos destinados a este fin, en un sistema de comunicación de datos se propagan

de un punto a otro mediante una señal electromagnética. Una señal analógica es una onda electromagnética que varía constantemente con el tiempo y que, según su espectro puede propagarse a través de una serie de medios como fibra óptica, cable coaxial y otros.

Una señal digital es una secuencia de pulsos de tensión que se pueden transmitir a través de un medio conductor, por lo que la transmisión analógica es una forma de transmitir señales independientemente de su contenido, ósea analógicas o digitales. En el mercado se han incorporado varios estándares para proporcionar una solución general al envío de datos.

2.7.1. Bluetooth.

El Bluetooth es un estándar de comunicación inalámbrica que permite la transmisión de datos a través de radiofrecuencia en la banda de 2.4GHz. Existen muchos módulos para usarlos en aplicaciones prácticas, de acuerdo con el precio y a la operación a realizar, el estándar presenta un bajo consumo y el tamaño pequeño lo hace funcional y versátil. Estos módulos presentan un chip con una placa de desarrollo que tiene pines necesarios para la comunicación serie, con un alcance de 5 a 10 metros, antena incorporada, autenticación con contraseña consumo de corriente de 50mA y otras especificaciones que se muestran en su datasheet. El dispositivo de bluetooth puede conectarse con microcontroladores que incluyan tecnología serie RX-TX e incluso puede comunicarse con dispositivos móviles con sistema Android.

Los modelos más populares son el bluetooth esclavo y el máster/esclavo, cuyas diferencias es que en el modo esclavo es el dispositivo que se conecta al módulo, mientras en el modo maestro es el módulo quien se conecta con el dispositivo. Físicamente los módulos son parecidos solo varían las conexiones:



Ilustración 20 Presentación física del módulo bluetooth

VCC: Alimentación del módulo.

GND: La masa del módulo.

TXD: Transmisión de datos.

RDX: Recepción de datos.

KEY: Poner a nivel alto para configuración.

STATE: conexión del led de salida.

2.8 PROTOCOLOS DE COMUNICACIÓN.

En los procesos industriales, no basta con tener equipos automatizados y equipos industriales para desarrollar operaciones óptimas, también es indispensable los protocolos de comunicación, ya que son ellos quienes permiten el intercambio de datos e información entre el equipo y el usuario.

Un protocolo de comunicación industrial es un conjunto de normas que le permiten a dos identidades que se encuentren en un mismo sistema establecer una comunicación, con el objetivo de pasar información a través de diversas variables. En la actualidad existen gran variedad de protocolos de comunicación

2.8.1 HART.

En 1986 aparece el protocolo de comunicación HART (Highway Addressable Remote Transducer), el cual entrega una solución para la comunicación de instrumentos inteligentes y que

además tiene la virtud de ser compatible con la transmisión analógica convencional en corriente 4-20mA, permitiendo que la señal analógica y las señales de comunicación digital sean transmitidas simultáneamente sobre el mismo cableado. Mediante este protocolo, la información de la magnitud que se mide es transmitida mediante la señal analógica de 4-20mA.

La comunicación puede operar sobre dos tipos de arquitectura de redes:

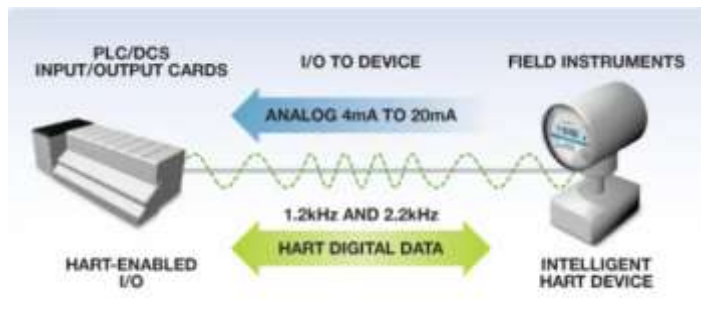


Ilustración 21 Protocolo Hart

2.8.1.1 Comunicación Punto a punto:

La señal convencional de 4-20 mA es usada para comunicar una variable de proceso mientras otras variables adicionales, parámetros de configuración y otras informaciones del dispositivo son transmitidas digitalmente usando el protocolo HART, por lo tanto la señal análoga de 4-20 mA no es afectada por la señal HART y puede ser usada para el monitoreo o control de forma normal.

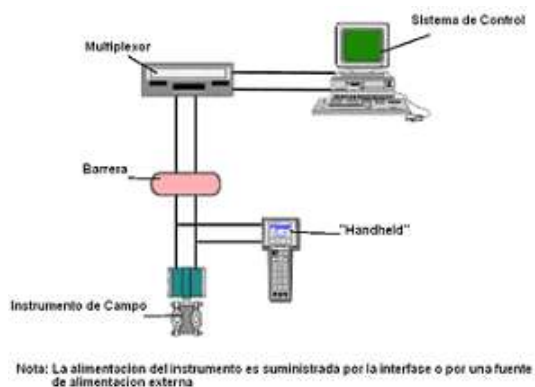


Ilustración 22 Protocolo Hart punto a punto

2.8.1.2 Comunicación tipo Multipunto:

El modo Multipunto requiere solamente un par de alambres y el lazo de control puede tener barreras de seguridad y fuentes de poder auxiliares para hasta 15 aparatos de terreno, donde todos los valores de proceso son transmitidos digitalmente.

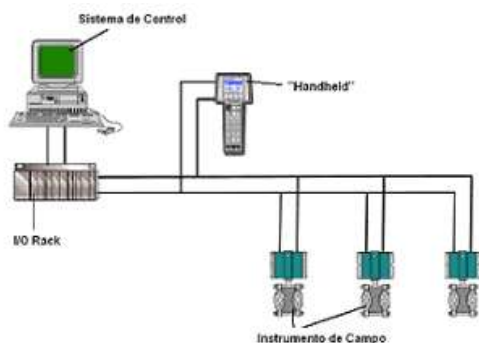
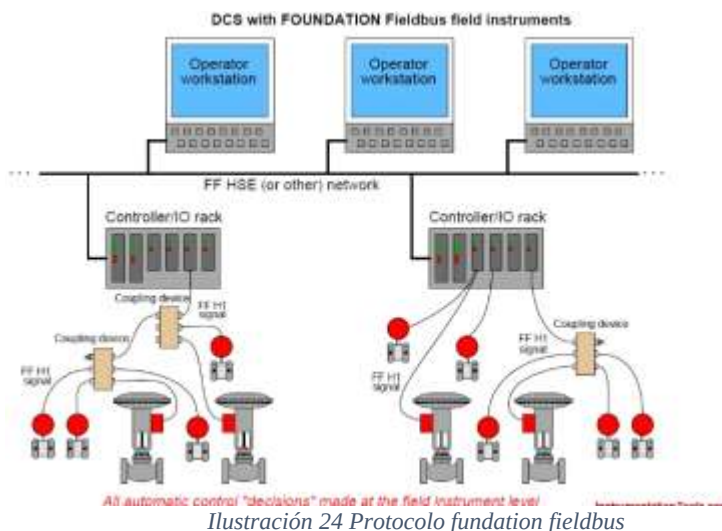


Ilustración 23 Protocolo Hart multipunto

2.8.2 Foundation Fielbus.

Fielbus es un protocolo de comunicación bidireccional que proporciona control de circuito cerrado en tiempo real entre instrumentos de campo inteligentes y sistemas host. Un host puede ser cualquier aplicación de software: desde el dispositivo portátil o portátil de un técnico hasta el control de procesos, la gestión de activos o el sistema de seguridad de una planta. Al ser un sistema de igual a igual, la comunicación puede ocurrir directamente entre dos dispositivos de campo habilitados para bus de campo o entre dispositivos de campo y un sistema de control o gestión de activos. Dada la naturaleza totalmente digital del bus de campo, seguir las pautas de la industria sobre prácticas adecuadas de cableado y terminación garantiza una comunicación confiable.

Aproximadamente el 90% de los problemas con la comunicación digital se deben a errores en la instalación de la capa física.



2.8.2.1. Diseño.

A diferencia de otras arquitecturas digitales, FOUNDATION Fieldbus se diseñó desde cero para permitir estrategias de control en el campo (CIF) en toda la planta. Esto significa que durante una falla del sistema host, los instrumentos de campo pueden comunicarse directamente entre sí para mantener operaciones seguras y continuas. La tecnología incluye una serie de capas físicas (el medio a través del cual se comunica la señal) y una rica plataforma de software que permite que los más de 1000 productos registrados interoperen entre varios fabricantes de dispositivos y sistemas de control, ofreciendo a los usuarios finales la flexibilidad de elegir la mejor opción. Soluciones de primera clase para su aplicación.

FOUNDATION Fieldbus también admite las recomendaciones NAMUR NE107 para gestionar datos de instrumentación inteligente. Esto garantiza que los datos de diagnóstico se gestionen de forma eficaz y que usted solo vea la información que necesita, cuando la necesita.

FieldComm Group es propietario de las especificaciones de FOUNDATION Fieldbus y proporciona servicios de desarrollo de especificaciones, capacitación y registro de productos asociados con la tecnología.

2.8.2.2. Ventajas.

Fieldbus proporciona importantes beneficios para todo tipo de operaciones industriales:

- Control en el campo (comunicación de dispositivo a dispositivo).
- Control de circuito cerrado (retroalimentación del sensor).
- Cableado reducido (múltiples dispositivos por cable).
- Hardware reducido (a través de Virtual Marshalling™).
- Reducción del espacio ocupado por el sistema.
- Recuento reducido de dispositivos (dispositivos multivariables)
- Puesta en marcha más rápida (en comparación con los enfoques tradicionales)
- Restricciones de control más estrictas (a través del control en el campo)
- Diagnóstico mejorado (a través de una mayor disponibilidad de energía y estado de la señal)
- Actualizaciones en línea sencillas (la transferencia sin interrupciones permite realizar actualizaciones desde la comodidad de su consola de trabajo)

2.8.3. Profibus.

Es un estándar de red digital de campo abierto que se puede comunicar entre los sensores de campo y el sistema de control o los controladores. Es importante conocer que PROFIBUS son las siglas de Process Field Bus. Esta red comunicación es adecuada tanto para aplicaciones rápidas y sensibles como para tareas de comunicación complejas.

2.8.3.1 Cómo funciona.

La tecnología de la información desempeñó un papel decisivo en el desarrollo de la automatización industrial y cambió las jerarquías y estructuras de las empresas. En la actualidad,

ha llegado al entorno industrial y a sus diversos sectores, desde las industrias de proceso y fabricación hasta los edificios y sistemas logísticos.

La posibilidad de comunicación entre los dispositivos y el uso de mecanismos estandarizados, abiertos y flexibles son componentes esenciales del concepto de automatización actual. La comunicación se expande rápidamente en la dirección horizontal a nivel de campo, así como en la dirección vertical integrando todos los niveles jerárquicos de un sistema de automatización. A nivel de campo, los periféricos distribuidos, como módulos de E/S, transductores, accionamientos, válvulas y paneles de operación funcionan en los sistemas de automatización, a través de un eficiente sistema de comunicación en tiempo real, el PROFIBUS **DP** o **PA**.

La transmisión de datos de proceso se realiza en ciclos, mientras que las alarmas, los parámetros y los diagnósticos se transmiten sólo cuando es necesario, de forma no cíclica. En lo que respecta a las células, los controladores programables como los PLC y los ordenadores de ingeniería necesitan comunicarse entre sí, por lo tanto requieren que se transfiera un gran paquete de datos en varias y potentes funciones de comunicación.

Además, la integración eficiente en los sistemas de comunicación corporativos existentes, como Intranet, Internet y Ethernet es absolutamente obligatoria. Esta necesidad se satisface mediante el protocolo PROFINET.

La revolución de la comunicación industrial en la tecnología juega un papel fundamental en la optimización de los sistemas de proceso y ha hecho una valiosa contribución para mejorar el uso de los recursos.

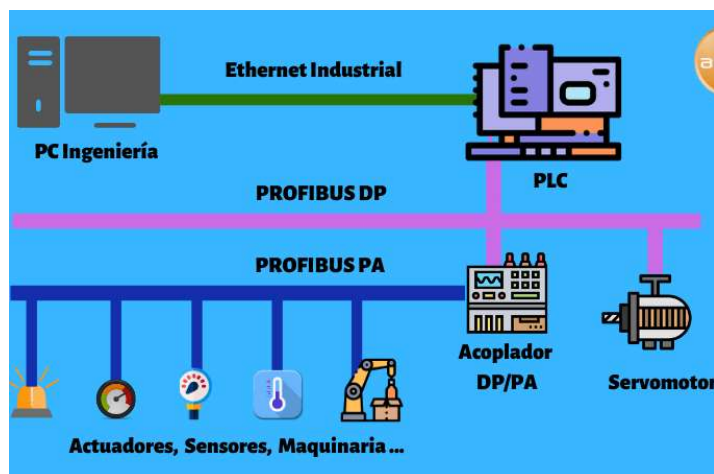


Ilustración 25 Profibus

2.8.3.2. Tipos de PROFIBUS

PROFIBUS, en su arquitectura, y se divide en tres tipos principales:

- PROFIBUS FMS (Fieldbus Message Specification).
- PROFIBUS DP (Periféricos Descentralizados)
- PROFIBUS PA (Automatización de Procesos).

2.8.3.2.1 PROFIBUS FMS

Ofrece al usuario una amplia selección de funciones en comparación con otras variantes. Es la solución en el estándar de comunicación universal que puede utilizarse para resolver tareas de comunicación complejas entre los PLC y los DCS.

Esta variante soporta la comunicación entre sistemas de automatización, además del intercambio de datos entre equipos inteligentes, generalmente utilizados en el nivel de control. Recientemente, como su función principal es la comunicación entre equipos, está siendo sustituida por el protocolo Ethernet Industrial.

2.8.3.2.2. PROFIBUS DP

Es la solución de alta velocidad. Fue desarrollado específicamente para la comunicación entre los sistemas de automatización y los equipos descentralizados. Es aplicable en sistemas de control

donde se enfatiza el acceso a dispositivos distribuidos de E/S y sustituye a los sistemas convencionales de 4 a 20 mA, HART o en transmisiones de 24 voltios.

Utiliza la interfaz estándar de la capa física de comunicación RS-485 o fibra óptica. Requiere menos de dos minutos para transmitir 1 Kbyte de E/S y se utiliza principalmente en sistemas de automatización industrial.

En la actualidad, el 90 por ciento de las aplicaciones en las que interviene PROFIBUS utilizan el PROFIBUS DP. Esta variante está disponible en tres versiones: DP-V0 (1993), DP-V1 (1997) y DP-V2 (2002). El origen de cada versión ha sido producido por el avance tecnológico y la creciente demanda de las aplicaciones.

2.8.3.2.3. PROFIBUS PA

Cumple con los requisitos de la automatización industrial, donde los sistemas de automatización y los sistemas de control de procesos se conectan con los equipos de campo, como transmisores de presión y temperatura, convertidores, posicionadores, etc. También puede sustituir el estándar de 4 a 20 mA.

Existen ventajas potenciales para el uso de esta tecnología, que conlleva ventajas funcionales como la transmisión de información confiable, manejo de estados variables, sistemas de seguridad de fallas, y la característica de equipo de auto-diagnóstico, medición con alta resolución, integración con control de alta velocidad, etc.

Permite la medición y el control a través de una línea y dos cables individuales. También alimenta los equipos de campo en zonas intrínsecamente seguras. Permite el mantenimiento y la conexión/desconexión de los equipos incluso durante el funcionamiento sin interferir en otras estaciones en zonas. Ha sido desarrollado en colaboración con la industria de control y procesos (NAMUR), en cumplimiento de los requisitos especiales de esta área de aplicación:

- El perfil de aplicación original para la automatización de procesos y la compatibilidad de equipos de campo de diferentes fabricantes.
- Adición y eliminación de estaciones de bus incluso en áreas intrínsecas sin afectar a otras estaciones.
- Una comunicación transparente a través de los acopladores del segmento entre el bus de automatización PROFIBUS PA y el bus de automatización industrial PROFIBUS DP.
- Transmisión de potencia y datos en el mismo par de cables basado en la tecnología IEC 61158-2.
- Uso en áreas potencialmente peligrosas con pantalla de protección a prueba de explosiones «intrínsecamente segura» o «intrínsecamente insegura».

El cable de par trenzado se utiliza como fuente de alimentación y comunicación de datos para cada equipo, lo que facilita la instalación y reduce el coste del hardware, lo que se traduce en un menor tiempo de puesta en marcha, un mantenimiento sin problemas, un bajo coste de software de ingeniería y un funcionamiento altamente fiable.

Todas las variantes de PROFIBUS se basan en el modelo de comunicación de red OSI (Open System Interconnection), de acuerdo con la norma internacional ISO 7498.

2.8.4. DeviceNet.

DeviceNet es una red digital, multi-punto para conexión entre sensores, actuadores y sistemas de automatización industrial en general. Esta tecnología fue desarrollada para tener máxima flexibilidad entre los equipos de campo e interoperabilidad entre diferentes fabricantes.

Introducido originalmente en 1994 por Allen-Bradley, DeviceNet transfirió su tecnología a ODVA (Open DeviceNet Vendor Association) en 1995. La ODVA difunden y promueven la tecnología DeviceNet y otras redes basadas en el protocolo CIP (Common Industrial Protocol).

La red DeviceNet está clasificada en el nivel de red llamada devicebus, cuyas características principales son: alta velocidad, comunicación a nivel de byte que incluye comunicación con equipos discretos y analógicos y alto poder de diagnóstico de los dispositivos de la red. La tecnología DeviceNet es un estándar abierto de automatización con el objetivo de transportar 2 tipos principales de información:

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

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

La capa física y de acceso a la red DeviceNet está basada en la tecnología CAN (Controller Area Network) y las capas superiores en el protocolo CIP (Common Application Layer – Capa de Aplicación Común), que define una arquitectura basada en objetos y conexiones entre ellos.

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

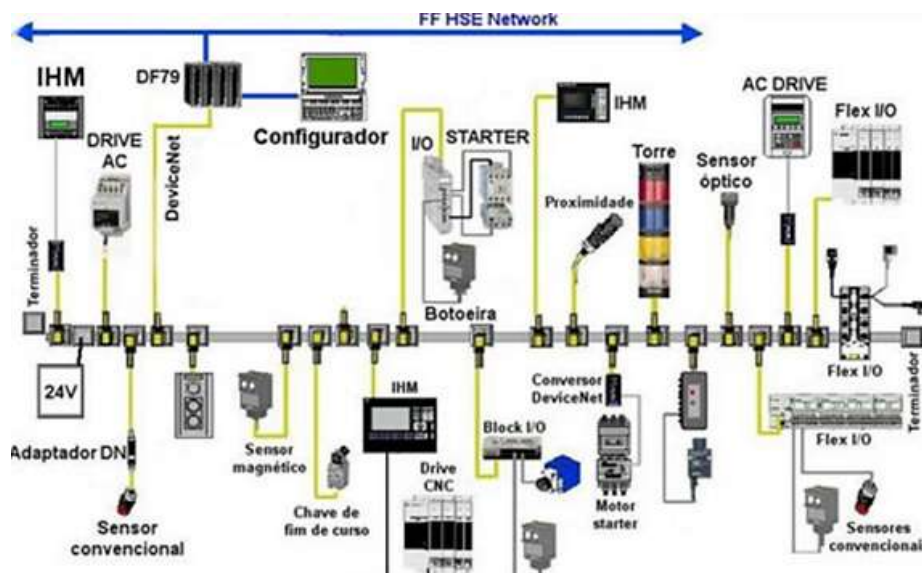


Ilustración 26 Red Devicenet

2.8.5 Contronet.

ControlNet es una red de alta velocidad en tiempo real tanto para transmitir datos de I/O en un tiempo crítico como para transmitir mensajes. Incluye acciones de carga/descarga de programas, configuración de datos y mensajes punto a punto (peer-to-peer), en un solo medio físico. La velocidad de transmisión de ControlNet es de 5 Mbits/s tanto para el control como datos.

ControlNet permite múltiples consoladores para el control de I/O utilizando el mismo cable. Esta red permite una ventaja sobre otras redes, que consiste en permitir un solo máster en una sola red, permitiendo también multicast tanto para las entradas como para los datos punto a punto (peer-to-peer), de esta forma reduce el tráfico en la red incrementando el rendimiento del sistema.

ControlNet es una red determinista y cíclica. Determinista por la capacidad de predecir cuándo los datos son enviados y cíclico porque asegura que las veces en que se transmite son constante y no depende de los dispositivos conectados.

El acceso a la red es controlado por un algoritmo de tiempo llamado Concurrent Time Domain Múltiple Access (CTDMA), las cuales regula la oportunidad de un nodo a transmitir en cada intervalo.

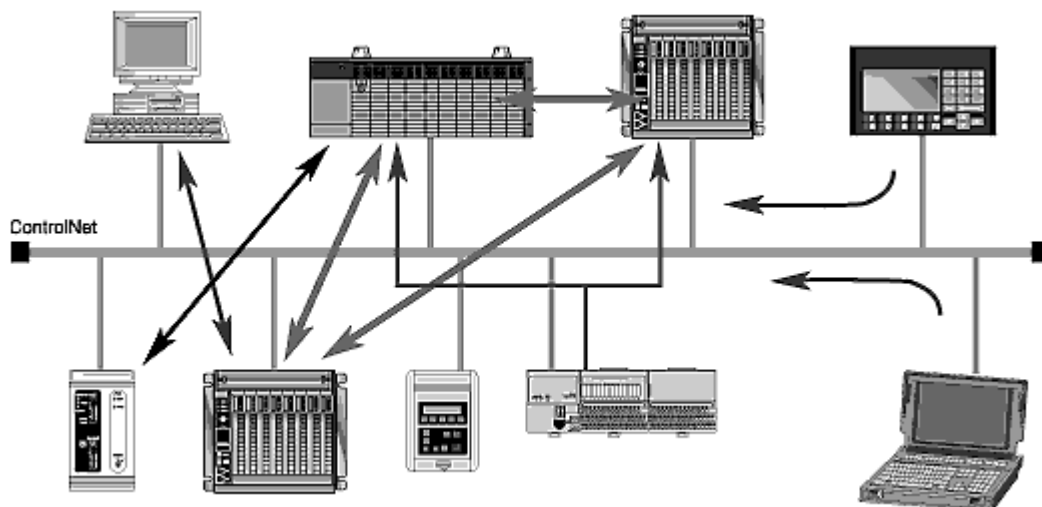


Ilustración 27 Red Contronet

2.8.5.1 Características

- Velocidad de 5 Mbits/s.
- Permite 99 nodos de conexión.
- Cada segmento puede cubrir distancias de hasta 1 000 m.
- El medio de transmisión utilizado puede ser cable coaxial o fibra óptica.
- La cantidad de información enviada es de 510 bytes.
- El tipo de comunicación es del tipo multimaestro.

2.8.6. Ethernet IP

Es una red de comunicación capaz de manejar grandes cantidades de datos a velocidades de 10 Mbps o 100 Mbps, y hasta 1500 bytes por paquete. La especificación utiliza un protocolo abierto en la capa de aplicación.

En la industria es especialmente popular para aplicaciones de control. En definitiva, este tipo de red es fácil de configurar, operar, mantener y ampliar. A su vez, permite la mezcla de productos de 10 Mbps y 100 Mbps, y es compatible con la mayoría de los conmutadores (switch) Ethernet. Esta tecnología se utiliza con ordenadores personales, mainframes, robots, dispositivos y

adaptadores de entrada/salida (E/S), controladores lógicos programables (PLC) y otros dispositivos. La especificación está respaldada por la Industrial Ethernet Association (IEA), ControlNet International (CI) y la Open DeviceNet Vendor Association (ODVA).

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

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

2.8.6.1 Qué es la capa física de Ethernet

En los primeros tiempos, las opciones de Ethernet eran más limitadas que en la actualidad. Dos de las opciones más comunes eran las configuraciones 10Base2 y 10Base5. Ambas funcionaban a 10 Mbps y utilizaban cable coaxial con nodos conectados al cable mediante conectores en T, o a través de interfaces de unidad de conexión (AUI) en una configuración de bus multipunto.

Las redes 10Base2 permitían longitudes de segmento de hasta 185 pies (56´38 metros aprox.) utilizando cable coaxial RG 58 (también llamado Thin Ethernet).

Este tipo de redes 10Base5 ofrecía mayores distancias entre nodos, pero el grueso cable coaxial y las conexiones eran voluminosas y difíciles de trabajar. Más tarde, otra solución en esta categoría de velocidad fue 10Base-FL, que utiliza cableado de fibra óptica y proporciona distancias superiores a los 2.000 pies (609´6 metros).

Otra de las primeras opciones de capa física de 10 Mbps -10Base-T- ganó popularidad rápidamente porque era más fácil de instalar y usaba cable barato de Categoría 3 de par trenzado sin blindaje (UTP). Los nodos (normalmente ordenadores con tarjetas de interfaz de red o NIC) se conectaban en una topología en estrella a un concentrador, que a su vez estaba conectado a otros segmentos de la red. Cada computadora tenía que estar a menos de 100 pies (30,48 metros) del centro. Se utilizaron conectores RJ-45 estándar.

A mediados de la década de 1990, empezó a estar disponible el equipo Ethernet de 100 Mbps, lo que aumentó significativamente la velocidad de transferencia de datos. Las NIC que se ajustarían automáticamente para funcionar a 10Mbps o 100Mbps simplificaron la migración al estándar más rápido.

Hoy en día, prácticamente todas las tarjetas de interfaz de red de ordenadores implementan 100Base-TX. El cable UTP de categoría 5e es el cable estándar utilizado con 100Base-TX y las longitudes de cable son las mismas que para las redes 10Base-T. En la actualidad, las redes coaxiales están siendo reemplazadas cada vez más por medios de fibra óptica, especialmente para enlaces punto a punto. Por ejemplo, 100Base-FX utiliza dos fibras ópticas y permite comunicaciones punto a punto full duplex de hasta 2,000 pies (609´6 metros).

2.8.6.2 Qué es la capa de enlace de datos

La capa de enlace de datos en Ethernet define su método de acceso a los medios. Los enlaces semidúplex, como los conectados en topologías de bus o estrella (10/100Base-T, 10Base2, 10Base5, etc.), utilizan el sentido de portadora y acceso múltiple con detección de colisiones (CSMA/CD). Este método permite que múltiples nodos tengan igual acceso a la red, similar a los primeros sistemas telefónicos de línea compartida en los que los usuarios escuchaban las conversaciones en curso y esperaban hasta que la línea estuviera libre antes de acceder a ella.

Todos los nodos de una red Ethernet monitorean continuamente las transmisiones en los medios. Si un nodo necesita transmitir, espera hasta que la red esté inactiva, entonces comienza la transmisión. Mientras transmite, cada nodo monitorea su propia transmisión y compara lo que 'oye' con lo que está tratando de enviar.

Si dos nodos empiezan a transmitir al mismo tiempo, las señales se superponen, estropeando los originales. Ambos nodos verán una señal diferente a la que están tratando de enviar. Esto se reconoce como una «colisión». Si hay una colisión, cada nodo deja de transmitir y sólo intenta retransmitir después de un retardo preestablecido, que es diferente para cada nodo.

Este método de acceso a los medios de comunicación facilita la adición o eliminación de nodos de una red. Simplemente conectando otro nodo, comenzará a escuchar y transmitir cuando la red esté disponible.

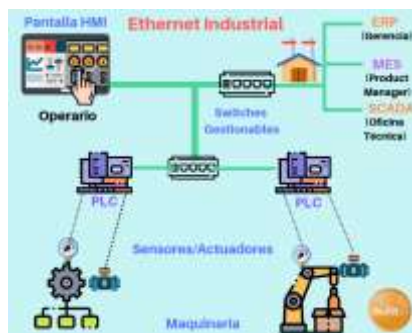


Ilustración 28 Red Ethernet

2.8.7 AsBus.

AS-Interface o AS-i fue diseñado en 1990 como una alternativa económica al cableado tradicional. El Objetivo fundamental fue determinar un sistema de comunicación único para todos los fabricantes de sensores y actuadores. La idea original fue crear una red simple para sensores y actuadores binarios, capaz de transmitir datos y alimentación a través del mismo bus, manteniendo una gran variedad de topologías que faciliten la instalación de los sensores y actuadores en cualquier punto del proceso con el menor esfuerzo posible y que cumpliera con las normativas de

seguridad. Desde entonces, el concepto AS-Interface se ha extendido considerablemente y las especificaciones iniciales se han revisado para adaptar el bus a las nuevas circunstancias y necesidades del mercado

Las especificaciones de AS-i se encuentran actualmente en su versión 3.0, y son de carácter abierto, lo que significa que cualquier fabricante puede obtener una copia de las mismas para elaborar sus productos conforme a dicho estándar.

En 1992 se creó la AS-International Association e.V., cuyas tareas fundamentales son:

- Difusión del concepto AS-i.
- Informar a los usuarios y proporcionar soporte técnico sobre AS-i.
- Estandarización de las normas AS-i y revisión periódica de éstas.
- Certificación de los productos que se adapten a las normas de AS-Interface, de forma que se garantice el correcto funcionamiento de éstos en cualquier red AS-i independientemente del fabricante.

2.8.7.1 Características principales

Las características principales de AS-Interface son:

- Especificaciones abiertas.
- Permite la conexión de sensores y actuadores "No AS-i" mediante módulos activos.
- Ideal para la interconexión de sensores y actuadores binarios.
- A través del cable AS-i se transmiten tanto los datos como la alimentación.
- Cableado sencillo y económico.
- Fácil montaje, con perforación de aislamiento.
- Sistema Monomaestro.
- Gran flexibilidad de topología (árbol).

- Reacción rápida: máximo 5ms para intercambiar datos con hasta 31 esclavos.
- Velocidad de transferencia de datos de 167 Kbits/s.
- Máximo 100m por segmento, con posibilidad de extensión hasta 3 segmentos (300m).
- Permite conectar hasta 124 sensores y 124 actuadores con módulos estándar.
- Permite conectar hasta 248 sensores y 186 actuadores con módulos extendidos.
- Cumple con los requerimientos IP-65/HIP-6 (idóneos para ambientes exigentes) e IP-20 (aplicaciones en cuadro).
- Temperatura de funcionamiento entre -25°C y +85°C.
- Transmisión por modulación de corriente, lo cual garantiza un alto grado de seguridad.
- Según un estudio realizado por la Universidad de Munich, mediante una red AS-i se puede ahorrar entre un 15% y un 30% del coste total.

AS-i se sitúa en la parte más baja de la pirámide de control, conectando los sensores y actuadores con el maestro del nivel de campo. Los maestros pueden ser autómatas o PCs situados en los niveles bajos de control, o pasarelas que comuniquen la red AS-Interface con otras redes de nivel superior, como Profibus o DeviceNet.

2.8.8 PROFINET

PROFINET (PROcess FIeld NETwork) es un protocolo de comunicación desarrollado por la organización PROFIBUS & PROFINET International (PI). Está diseñado para conectar dispositivos industriales con varios tipos de equipamiento productivo, como motores, sensores y otros dispositivos electrónicos. El protocolo fue diseñado para ser independiente del fabricante, lo que significa que los dispositivos de diferentes marcas pueden funcionar juntos sin problemas.

Además, el protocolo se ha optimizado para proporcionar un rendimiento óptimo a través de soluciones innovadoras como la detección automática de equipamiento, herramientas de diagnóstico avanzadas y recuperación rápida en caso de fallas. Estas características hacen que las redes basadas en PROFINET sean ideales para los entornos industriales modernos.

2.8.8.1 Características clave de PROFINET

El estándar PROFINET es una tecnología de red de campo para la comunicación entre equipos industriales y controladores. Está basado en Ethernet y se usa para interconectar dispositivos dentro de sistemas automatizados para el ahorro de tiempo, reducción de costes y mejora del rendimiento. El estándar PROFINET ofrece los siguientes beneficios:

En primer lugar, proporciona alta velocidad de comunicación a partir de 100 Mbps, lo que permite realizar múltiples tareas simultáneamente al mismo tiempo.

También admite protocolos flexibles para permitir a los dispositivos compartir información sin conflictos.

Además, el estándar PROFINET ofrece una arquitectura escalable que permite añadir dispositivos fácilmente según sea necesario. Utiliza estructuras jerárquicas complejas, lo que facilita la gestión del sistema, y también posee capacidades autodescriptivas e interoperables que simplifican el diseño y la configuración de redes.

El estándar PROFINET, al estar basado en Ethernet (IEEE 802.3), es compatible con otros protocolos como TCP-UDP/IP, SNMP, LLDP, DHCP. Esto significa que se puede manejar múltiples protocolos en un único sistema sin complicaciones adicionales.

Además, cuenta con seguridad integrada para proteger sus datos confidenciales contra cualquier intruso externo o ataque malicioso.

Finalmente, el estándar PROFINET ofrece funciones avanzadas como monitorización en tiempo real, diagnóstico remoto y análisis predictivo de fallas para ayudar a los operadores en su procesamiento industrial diario. Estas características contribuyen enormemente a la eficiencia operacional al permitir que los problemas detectados se aborden inmediatamente antes de que cause un daño significativo a los activos o producción del negocio industrial. Además, el estándar PROFINET cuenta con funciones para garantizar la integridad dentro del sistema industrial al permitir verificar regularmente sus componentes contra versiones anteriores no autorizadas o desactualizadas.

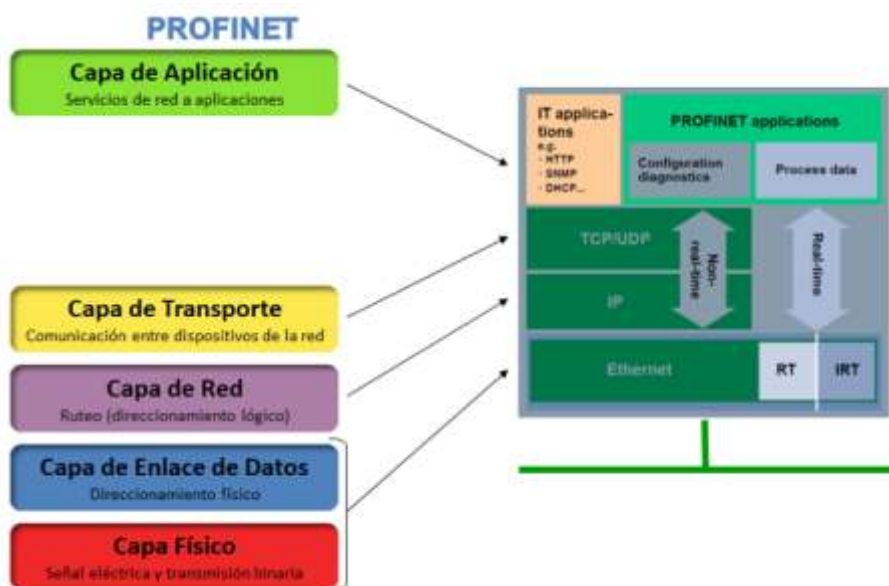


Ilustración 29 Profinet.

2.8.9 Ethercat

Las capas de aplicación de Ethernet utilizadas en los sistemas de automatización industrial y de edificios suelen ser capas de aplicación semidúplex de “baja carga útil”, un mensaje de Ethernet para una capa de aplicación como Modbus TCP, por ejemplo, puede llevar solo un único registro para un dispositivo servidor con poca información como un controlador de válvula. En los dispositivos de carga pequeña como esa, se pierden enormes porciones de ancho de banda ya que no solo se mueven pequeños paquetes de datos, sino que esos mensajes solo se emiten cuando el

dispositivo Cliente o Maestro solicita esa información. Modbus TCP es un ejemplo extremadamente ineficiente, pero los mismos tipos de ancho de banda y problemas de carga útil pueden encontrarse en capas de aplicación como EtherNet / IP y Profinet IO.

Beckhoff, una compañía de automatización alemana, desarrolló un sistema de bus de campo llamado Fast Lightbus para corregir el problema de la utilización del bajo ancho de banda presente en otros protocolos de Ethernet, este protocolo llevó a EtherCAT, que Beckhoff lanzó en 2003.

2.8.9.1 Grupo de Tecnología EtherCAT

En 2004, Beckhoff ayudó a crear un nuevo grupo para promover el protocolo EtherCAT, sus esfuerzos llevaron al Grupo de Tecnología EtherCAT, o ETG, Beckhoff donó los derechos de EtherCAT a la ETG. Las ventajas de la alianza EtherCat Technology Group son numerosas, el ETG puede proporcionar investigación y avance del protocolo EtherCAT, la ETG también publica especificaciones y documentos que permiten a las organizaciones miembros tener mayor éxito en el uso de EtherCAT.

2.8.9.2 Estándares internacionales

La ETG trabaja con la Comisión Electrotécnica Internacional (IEC), sirviendo específicamente como enlace para el grupo de trabajo de comunicaciones digitales. La asociación ha llevado a la estandarización a lo largo de la historia del protocolo EtherCAT.

En 2005, EtherCAT se estandarizó como IEC / PAS 62407, esta norma ahora está obsoleta, aunque no se debe a ninguna falla en EtherCAT. En cambio, EtherCAT se ha integrado en una serie de otros estándares, un signo de su versatilidad.

Las normas de bus de campo IEC IEC 61158 e IEC 61784-2 incluyen EtherCAT a partir de 2007, EtherCAT también se incluye en la norma publicada ISO 15745-4.

2.8.9.3 Principio fundamental

El principio fundamental de EtherCAT es la lectura de paso, pasar a través de la lectura significa que los mensajes no están destinados a un solo nodo y son consumidos por ese nodo, en su lugar, los mensajes se transmiten al siguiente nodo en una cadena a medida que se procesan. Los datos de entrada a un nodo se leen a medida que el mensaje se procesa y los datos de salida se insertan en el mensaje al siguiente nodo.

EtherCAT Master emite un solo mensaje con datos para todos los nodos, a medida que el mensaje se transmite alrededor del anillo y de vuelta hacia el Maestro, cada nodo lee sus entradas y agrega sus salidas al mensaje, cuando el mensaje vuelve al Maestro EtherCAT, cada nodo en la red ha recibido nuevos datos de entrada del Maestro y ha devuelto nuevos datos de salida al Maestro. Sin la deficiencia de pequeñas cargas útiles o mensajes dirigidos a nodos específicos, una red EtherCAT puede lograr la máxima utilización del ancho de banda.

Una red EtherCAT puede compararse con un ferrocarril donde cada estación puede descargar y volver a cargar los vagones del tren mientras el tren se mueve a través de la estación.

2.8.9.4 Representación de datos

Cómo una red representa datos es una característica fundamental de una red. Es un factor distintivo que debe ser la primera pregunta en la mente de cualquiera que esté investigando seriamente una tecnología de red.

La capa de aplicación Ethernet con la representación de datos más sofisticada es CIP. Los dispositivos CIP (Common Industrial Protocol) de ODVA utilizan una representación de datos Objeto / Instancia / Atributo. Al igual que los datos se recogen en los objetos. Los conjuntos comunes de esos objetos se especifican como instancias. Los elementos de datos dentro de esos

objetos son atributos. Los objetos también pueden tener controladores de eventos y estructuras operativas implementadas como máquinas de estado.

La capa de aplicación Ethernet con la representación de datos menos sofisticada es Modbus TCP. Al igual que su primo en serie, todos los dispositivos Modbus TCP representan datos como registros o bobinas sin firma de 16 bits.

Hay versiones de entrada y salida de cada uno. Desafortunadamente, hay poca estandarización o similitud en los dispositivos Modbus. Cada fabricante desarrolla su propio mapa de registro Modbus único.

Profinet IO utiliza una representación de datos similar a Profibus. Cada dispositivo se parece a un bastidor de E / S. Hay un designador de bastidor, una ranura, un canal y un punto.

EtherCAT utiliza una representación de datos muy simple, similar a Modbus. Hay un espacio de datos, un espacio de datos enorme, y a cada dispositivo se le asigna una parte de ese espacio de datos. A medida que los mensajes se transmiten a través de la red, asignan su parte del espacio de datos a los datos en el mensaje de Ethernet.

2.8.9.5 Marco EtherCAT

Para que el chasis EtherCAT siga rodando sin detenerse en cada nodo, el paquete debe contener componentes específicos, como los vagones de un tren. Mirando el chasis EtherCAT, su analogía con un tren es bastante evidente. El encabezado actúa como una locomotora. Los coches, más su contenido, son datos de DOP. El caboose, completo con la información de envío, es el contador de trabajo.

Todas estas partes encajan simplemente en el marco EtherCAT, y el marco encaja simplemente en un marco Ethernet. Ethernet es el medio de transmisión que permite que EtherCAT funcione. El marco EtherCAT simplemente reemplaza el marco IP de un mensaje Ethernet estándar. Por lo

tanto, la trama Ethernet no necesita modificación, lo que nuevamente contribuye a la flexibilidad para EtherCAT.

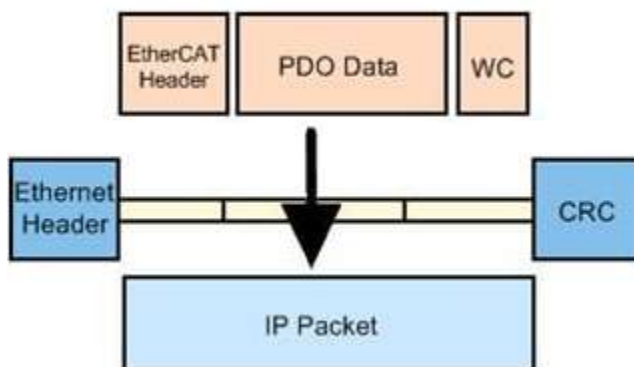


Ilustración 30 Macro Ethercat

2.8.10 BACnet

BACnet significa redes de control y automatización de edificios. Este protocolo de comunicación de datos se utiliza ampliamente en todo el mundo con HVAC, iluminación e incluso sistemas de seguridad dentro de los edificios. BACnet es una forma de comunicación de protocolo de código abierto que permite que los dispositivos BACnet se comuniquen entre sí en la red de un edificio. BACnet fue desarrollado por ASHRAE para promover el uso de protocolos abiertos en contraposición a los protocolos propietarios que alguna vez dominaron la industria.

Antes de que BACnet y otros protocolos de código abierto estuvieran disponibles, los proveedores implementaban un protocolo propietario para su marca de dispositivos (controladores, por ejemplo). Esto permitiría que los controladores de la misma marca y protocolo se comunicaran entre sí en la red de un edificio. Esto acabó creando un monopolio dentro de la industria y los profesionales vieron la necesidad de romper este molde; por lo tanto, ASHRAE desarrolló BACnet.

Los sistemas de control digital directo (DDC) son un ejemplo de dónde se implementa BACnet. Los controladores digitales se comunican entre sí en una red y transmiten datos a un sistema de gestión de edificios (BMS) o un sistema frontal. El sistema front-end es una interfaz gráfica que se utiliza para mostrar una lista de dispositivos BACnet, así como información útil para el usuario final. Esta es una herramienta extremadamente poderosa que se utiliza para monitorear equipos en un edificio. Información como la temperatura ambiente, el punto de ajuste de la habitación, la posición de la compuerta y los valores del flujo de aire se pueden informar en una interfaz gráfica, junto con tendencias inteligentes y alarmas para notificar al personal si un equipo ha fallado o si está funcionando según lo previsto.

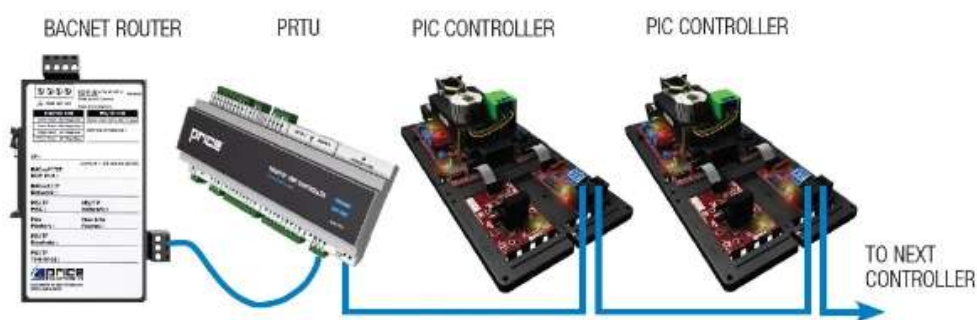


Ilustración 31 Bacnet permite la conexión de varios dispositivos

2.8.11 CANOpen

CANopen es un protocolo de comunicación de alto nivel y una especificación de perfil de dispositivo que se basa en el protocolo CAN (Controller Area Network). El protocolo fue desarrollado para aplicaciones de redes integradas, como redes de vehículos. El CANopen cubre un marco de programación de red, descripciones de dispositivos, definiciones de interfaz y perfiles de aplicación. CANopen proporciona un protocolo que estandariza la comunicación entre dispositivos y aplicaciones de diferentes fabricantes. Se ha utilizado en una amplia gama de industrias, destacando en aplicaciones de automatización y movimiento.

En términos del modelo de sistemas de comunicación OSI, CAN cubre los dos primeros niveles: la capa física y la capa de enlace de datos. La capa de enlace de datos incluye el hecho de que CAN es un protocolo basado en tramas (mensajes).

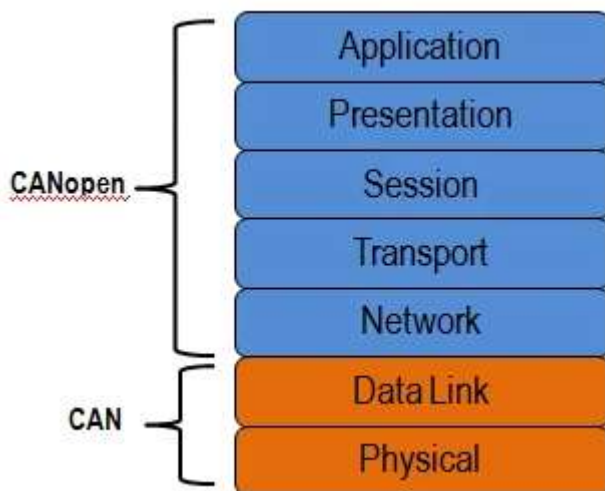


Ilustración 32 CAN y CANOpen en el modelo OSI

CANopen cubre las cinco capas principales: red (direccionamiento, enrutamiento), transporte (confiabilidad de extremo a extremo), sesión (sincronización), presentación (datos codificados de manera estándar, representación de datos) y aplicación. La capa de aplicación describe cómo configurar, transferir y sincronizar dispositivos CANopen.

2.8.12. MODBUS.

Modbus es un protocolo de comunicación abierto, utilizado para transmitir información a través de redes en serie entre dispositivos electrónicos. El dispositivo que solicita la información se llama maestro Modbus y los dispositivos que suministran la información son los esclavos Modbus.

En realidad, esto significa que un dispositivo esclavo no puede ofrecer información; debe esperar a que se le pida. El maestro escribirá datos en los registros de un dispositivo esclavo y leerá los datos de los registros de un dispositivo esclavo.

Por lo tanto, en una red Modbus estándar, hay un maestro y hasta 247 esclavos, cada uno con una dirección de esclavo única de 1 a 247. El maestro también puede escribir información a los esclavos.

Además, esta red de comunicación industrial usa los protocolos RS232/RS485/RS422. Su simplicidad y el hecho de que los fabricantes pueden incorporarlo en sus productos sin cargo alguno han ayudado a que se convierta en el método más popular de conexión de dispositivos electrónicos industriales.

En consecuencia, son estos protocolos abiertos los que muchos fabricantes adaptan para integrar fácilmente sus productos en el mercado.

2.8.12.1 ¿Quién inventó Modbus?

Este protocolo industrial estándar fue creado por Modicon, a finales de los 70 para la comunicación entre controladores lógicos programables (PLC). En la actualidad, Modbus sigue siendo el protocolo más usado para conectar dispositivos industriales.

La especificación del protocolo se publica abiertamente y el uso del protocolo está libre de derechos. Lo que significa que es libre de instalar para los fabricantes en sus equipos sin tener que pagar derechos de autor

2.8.12.2 Tipos de protocolo Modbus

Existen varios tipos de versiones en el protocolo Modbus para el puerto serie y Ethernet, que se utilizan para atender las necesidades específicas de los sistemas de automatización industrial en las empresas. Por ejemplo, Modbus TCP se utiliza para Ethernet, y Modbus RTU y Modbus ASCII para los puertos serie.

Las más comunes son:

- Modbus RTU

- Modbus TCP
- Modbus ASCII
- Modbus Plus

2.8.12.2.1. Modbus RTU.

El protocolo Modbus RTU es un medio de comunicación que permite el intercambio de datos entre los controladores lógicos programables (PLC) y los ordenadores (PC). Los dispositivos electrónicos pueden intercambiar información a través de conexiones en serie utilizando este protocolo

En definitiva, una característica distintiva de esta versión es el uso de codificación binaria y una fuerte verificación de errores CRC. En realidad, es la implementación del protocolo Modbus utilizado con mayor frecuencia en aplicaciones industriales e instalaciones de producción automatizada.

No obstante, este es un protocolo serial relativamente simple que puede ser transmitido a través de la tecnología tradicional UART. Los datos se transmiten en bytes de 8 bits, un bit cada vez, a velocidades en baudios que van de 1200 bits por segundo (baudios) a 115200 bits por segundo. La mayoría de los dispositivos sólo soportan velocidades de hasta 38400 bits por segundo.

Finalmente, una red Modbus RTU tiene un maestro y uno o más esclavos. Cada esclavo tiene una dirección de dispositivo o número de unidad de 8 bits. Los mensajes enviados por el maestro incluyen la dirección del esclavo al que va dirigido el envío.

El esclavo debe responder sólo si se reconoce su dirección, y debe responder dentro de un cierto período de tiempo o el maestro lo llamará un error de «no respuesta».

El Modbus RTU se utiliza frecuentemente:

- Cuando se utilizan dispositivos Modbus similares
- Cuando el protocolo Modbus se entiende bien
- Como estándar de la instalación

La ADU de RTU parece ser mucho más simple.

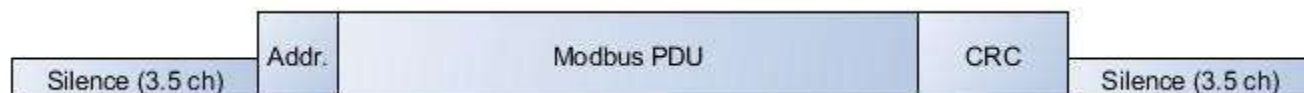


Ilustración 33 La ADU de RTU

A diferencia de la ADU de TCP/IP más compleja, esta ADU incluye solamente dos piezas de información, además de la PDU principal. Primero, una dirección es usada para definir para qué esclavo está diseñada una PDU. En la mayoría de las redes, una dirección 0 define la dirección de "broadcast". Es decir, un maestro puede enviar un comando de salida a la dirección 0 y todos los esclavos deben procesar la solicitud, pero ningún esclavo debe responder. Además de esta dirección, un CRC es usado para asegurar la integridad de los datos.

Sin embargo, la realidad es que la situación en las implementaciones más modernas está lejos de ser simple. Encapsulando el paquete hay un par de tiempos en silencio, es decir, periodos en los que no hay comunicación en el bus. Para una velocidad de transferencia de 9,600, este tiempo es alrededor de 4 ms. El estándar define una longitud mínima de silencio, independientemente de la velocidad de transferencia, de un poco menos de 2 ms.

Primero, esto tiene un inconveniente de rendimiento ya que el dispositivo debe esperar a que el tiempo muerto se cumpla antes de que el paquete pueda ser procesado. Más peligrosa aún, sin embargo, es la introducción de diferentes tecnologías usadas para transferencia serial y velocidades de transferencia mucho más rápidas que cuando se introdujo el estándar. Con un cable convertidor de USB a serial, por ejemplo, usted no tiene ningún control sobre el paquete y la transferencia de

datos. Las pruebas muestran que usar un cable de USB a serial con el controlador NI-VISA introduce grandes intervalos de tamaño variable en el flujo de datos y estos intervalos, períodos de silencio, engañan al código compatible con la especificación al creer que un mensaje se ha completado. Debido a que el mensaje no es completado, esto por lo general conduce a un CRC no válido y al dispositivo que interpreta la ADU como alterada.

Además de los problemas con la transmisión, las tecnologías modernas de controlador abstraen comunicación serial significativa y generalmente requieren un mecanismo de consulta desde el código de la aplicación. Por ejemplo, ni el .NET Framework 4.5 SerialPort Class ni el controlador NI-VISA proporcionan un mecanismo para detectar silencio sobre una línea serial excepto al consultar los bytes en el puerto. Esto resulta en una escala de bajo rendimiento (si la consulta es realizada demasiado lento) o alto uso del CPU (si la consulta es realizada demasiado rápido).

Un método común para resolver estos problemas es romper la capa de abstracción entre la PDU de Modbus y la capa de red. Es decir, el código serial interroga el paquete de la PDU de Modbus para determinar el código de función. Combinado con otros datos en el paquete, la longitud del paquete restante puede ser descubierta y usada para determinar el final del paquete. Con esta información, puede usarse un descanso mucho más largo, lo que permite silencios de transmisión y puede ocurrir consulta a nivel de la aplicación mucho más lentamente. Se recomienda utilizar este mecanismo para un nuevo desarrollo. El código que no emplea esto puede experimentar un número mayor de paquetes "alterados" de lo esperado.

RTU debe transmitirse continuamente sin vacilaciones entre caracteres. Los mensajes Modbus son entramados (separados) por períodos inactivos (silenciosos).

2.8.12.2.2 Modbus TCP

Modbus/TCP se introdujo para aprovechar las infraestructuras LAN actuales. A su vez, aumentó el número de unidades que podían conectarse a la misma red.

Este sistema engloba los bloques de datos de solicitud y respuesta del Modbus RTU en un bloque TCP transmitido a través de redes estándar de Ethernet. Por otro lado, el número de unidades sigue estando incluido y su interpretación varía según la aplicación: la dirección de la unidad o del esclavo no es el medio principal de direccionamiento en el TCP. Entonces, la dirección más importante aquí es la dirección IP.

El puerto estándar para Modbus TCP es el 502, pero el número de puerto a menudo puede reasignarse si se desea. La versión TCP sigue el modelo de referencia de Red OSI. Por otro lado, define las capas de presentación y aplicación en el modelo OSI.

En consecuencia, Modbus TCP hace que la definición de maestro y esclavo sea menos obvia, porque la Ethernet permite la comunicación entre pares. La definición de cliente y servidor son los elementos más conocidos en las redes basadas en Ethernet.

En este contexto, el esclavo se convierte en el servidor y el maestro en el cliente. Puede haber más de un cliente que obtenga datos de un servidor. En términos de Modbus, esto significa que puede haber múltiples maestros, así como múltiples esclavos.

En lugar de definir maestro y esclavo en un dispositivo físico por dispositivo, en esta ocasión es responsabilidad del diseñador del sistema crear asociaciones lógicas entre la funcionalidad de maestro y esclavo.

El Modbus/TCP se utiliza frecuentemente en

- PLC (controladores industriales)
- Sistemas SCADA (visualizaciones y control básico de procesos industriales)

➤ Sensores y actuadores

Las ADUs de TCP consisten en el Encabezado de Protocolo de Aplicación Modbus (MBAP) combinado con la PDU de Modbus. El MBAP es un encabezado de uso general que depende de una capa de red confiable. El formato de esta ADU, incluyendo el encabezado.



Ilustración 34 La ADU de TCP

Los campos de datos del encabezado indican su uso. Primero, incluye un identificador de transacción. Esto es valioso en una red en la que se pueden soportar múltiples solicitudes simultáneamente. Es decir, un maestro puede enviar solicitudes 1, 2, y 3. En algún punto, un esclavo puede responder en el orden 2, 1, 3, y el maestro puede igualar las solicitudes con las respuestas y analizar los datos con precisión. Esto es útil para redes Ethernet.

El identificador de protocolo es normalmente cero, pero usted puede utilizarlo para ampliar el comportamiento del protocolo. El campo de longitud es usado por el protocolo para delinear la longitud del resto del paquete. La ubicación de este elemento también indica la dependencia de este formato encabezado en una capa de red confiable. Debido a que los paquetes TCP tienen verificación de errores integrada y garantizan la coherencia de los datos y la entrega, la longitud del paquete puede ser ubicado en cualquier parte del encabezado. En una red inherentemente menos confiable como una red serial, un paquete podría perderse, teniendo el efecto de que incluso si la escritura de datos leída por la aplicación incluía información válida de transacción y del protocolo, la información de longitud alterada volvería inválido al encabezado. TCP proporciona una cantidad razonable de protección contra esta situación.

El ID de Unidad generalmente no es usado por los dispositivos TCP/IP. Sin embargo, Modbus es un protocolo común en el que se implementan muchos gateways, lo cual convierte al protocolo

Modbus en otro protocolo. En el caso original de uso, un gateway Modbus TCP/IP a serial podría ser usado para permitir la conexión entre las nuevas redes TCP/IP y redes seriales anteriores. En dicho entorno, el ID de Unidad es usado para determinar la dirección del dispositivo esclavo para la que la PDU está destinada.

Finalmente, la ADU incluye una PDU. La longitud de esta PDU está aún limitada a 253 bytes para el protocolo estándar.

2.8.12.2.3. Modbus ASCII

Modbus ASCII es una implementación más antigua que contiene todos los elementos de un paquete RTU, pero expresada completamente en caracteres ASCII imprimibles. Estos son caracteres hexadecimales que contienen 4 bits de datos cada uno.

El uso de esta herramienta permite: analizar la actividad del puerto serie, monitorear múltiples puertos serie a la vez, emular la transmisión de datos a los dispositivos en serie de varios formatos como cadena, binario, decimal y hexadecimal y, por último, la exportación de los datos monitoreados.

Actualmente, este protocolo se considera obsoleto, ya no se utiliza con frecuencia y no está incluido en la especificación formal del protocolo Modbus.

La ADU de ASCII es más compleja que la de RTU, también evita muchos de los problemas del paquete RTU. Sin embargo, tiene algunas de sus propias desventajas.

0x3A ":"	Address (ASCII)	Modbus PDU (ASCII)	LRC (ASCII)	0x0D CR	0x0A LF
-------------	--------------------	-----------------------	----------------	------------	------------

Ilustración 35 La ADU de ASCII

Al resolver el problema de determinar el tamaño del paquete, la ADU de ASCII tiene un inicio y final bien definido y único para cada paquete. Es decir, cada paquete comienza con ":" y termina con un regreso de carro (CR) y alimentación de línea (LF). Además, los APIs seriales como NI-

VISA y el .NET Framework SerialPort Class pueden leer datos fácilmente en un búfer hasta que un carácter específico, como CR/LF, es recibido. Estas características hacen que sea fácil procesar la escritura de datos en la línea serial de manera eficiente en el código de la aplicación moderno.

La desventaja del ADU de ASCII es que todos los datos son transferidos como caracteres hexadecimales codificados en ASCII. Es decir, en lugar de enviar un solo byte para el código de función 3, 0x03, envía los caracteres ASCII "0" y "3" o 0x30 / 0x33.

Esto hace que el protocolo sea más legible, pero también significa que el doble de datos debe ser transferidos a través de la red en serie y que las aplicaciones que envían y reciben deben ser capaces de analizar los valores ASCII.

2.8.12.2.4. Modbus Plus.

Modbus Plus es un protocolo de red con alta velocidad entre pares. Está basado en la comunicación a través de un token bus. En definitiva, es un sistema completo con un medio predefinido y la aplicación de un sistema de comunicación de paso rápido. Modbus es sólo un protocolo, en cambio Modbus Plus es un sistema completo con un medio predefinido y la implementación de la capa física (capa 1 de OSI).

Por consiguiente, es un sistema LAN para aplicaciones de control industrial, que permite a los dispositivos en red intercambiar mensajes para el control y la supervisión de procesos en lugares remotos de la planta industrial.

Por otro lado, el protocolo de mensajería de la capa 7 de Modbus Plus es esencialmente el mismo que se utiliza para Modbus Serial y Modbus/TCP. La capa física se implementa con RS485 y funciona sobre cable de par trenzado blindado. Esto da como resultado la transmisión de datos a 1 Mbps.



Ilustración 36 Arquitectura básica de Modbus

2.9. PROGRAMACIÓN.

Las computadoras son dispositivos complejos, pero están diseñados para hacer una cosa bien: **ejecutar aquello que se les indica**. Solo hay que definir de como se le indica a una computadora lo que queremos que ejecute. Estas indicaciones se llaman técnicamente **instrucciones** y expresan en un lenguaje. Se podría decir que **programar** consiste en escribir instrucciones para que sean ejecutadas por una computadora. El lenguaje que se utiliza se denomina **lenguaje de programación**.

2.9.1. Código maquina

Hasta el momento seguimos con el problema de cómo hacer que una computadora entienda el lenguaje de programación, si se conoce que las computadoras solo pueden entender un lenguaje muy **simple** denominado código máquina. En este lenguaje se utilizan únicamente los símbolos 0 y 1 en representación de los niveles de tensión alto y bajo, que, al fin y al cabo, son los estados que puede manejar un circuito digital. Hablamos de sistema binario. Si tuviéramos que escribir programas de ordenador en este formato sería una tarea ardua, pero afortunadamente se han ido creando con el tiempo lenguajes de programación intermedios que, posteriormente, son convertidos a código máquina. Si intentamos visualizar un programa en código máquina, únicamente obtendríamos una secuencia de ceros y unos

```

00001000 00000010 01111011 10101100 10010111 11011001 01000000 01100010
00110100 00010111 01101111 10111001 01010110 00110001 00101010 00011111
10000011 11001101 11110101 01001110 01010010 10100001 01101010 00001111
11101010 00100111 11000100 01110101 11011011 00010110 10011111 01010110

```

Ilustración 37 Código Maquina

2.9.2 Ensamblador

El primer lenguaje de programación que encontramos es ensamblador. Se observa un ejemplo de código en ensamblador del típico programa que se escribe por primera vez

```

SYS_SALIDA equ 1

section .data
    msg db "Hello, World",0x0a
    len equ $ - msg ;longitud de msg

section .text
global _start ;para el linker
_start: ;marca la entrada
    mov eax, 4 ;llamada al sistema (sys_write)
    mov ebx, 1 ;descripción de archivo (stdout)
    mov ecx, msg ;msg a escribir
    mov edx, len ;longitud del mensaje
    int 0x80 ;llama al sistema de interrupciones

fin: mov eax, SYS_SALIDA ;llamada al sistema (sys_exit)
    int 0x80

```

Ilustración 38 Programa hola mundo con ensamblador

Lo único que hace este programa es mostrar en la pantalla de la computadora la frase «Hello, World», pero además teniendo en cuenta que sólo funcionará para una arquitectura x86.

2.9.3 Lenguaje C

En el lenguaje ensamblador se facilita la tarea de desarrollar las instrucciones en el programa, pero sigue siendo complicado ya que las instrucciones son muy específicas y no proporcionan una semántica entendible. El lenguaje que se desarrollo para suplir los obstáculos del lenguaje ensamblador fue el lenguaje C. este lenguaje permite hacer uso de instrucciones más claras y potentes. El mismo ejemplo anterior del programa «Hello, World» se escribiría así en lenguaje C:

```

#include <stdio.h>

int main() {
    printf("Hello, World");
    return 0;
}

```

Ilustración 39 Programa hola mundo con lenguaje C.

2.9.4 Python

Si seguimos «subiendo» en esta lista de lenguajes de programación, podemos llegar hasta Python. Se dice que es un lenguaje de más alto nivel en el sentido de que sus instrucciones son más entendibles por un humano. Veamos cómo se escribiría el programa «Hello, World» en el lenguaje de programación Python:

```
print('Hello, World')
```

Ilustración 40 Programa hola mundo con lenguaje Python

Hemos pasado de código máquina (ceros y unos) a código Python en el que se puede entender perfectamente lo que estamos indicando al ordenador. Para que la computadora entienda lo que tiene que hacer si le pasamos un programa hecho en Python (o cualquier otro lenguaje de alto nivel). Se debe de contar un programa especial que es un compilador.

2.9.5 Compilador

Los compiladores son programas que convierten un lenguaje «cualquiera» en código máquina. Se pueden ver como traductores, permitiendo a la máquina interpretar lo que queremos hacer

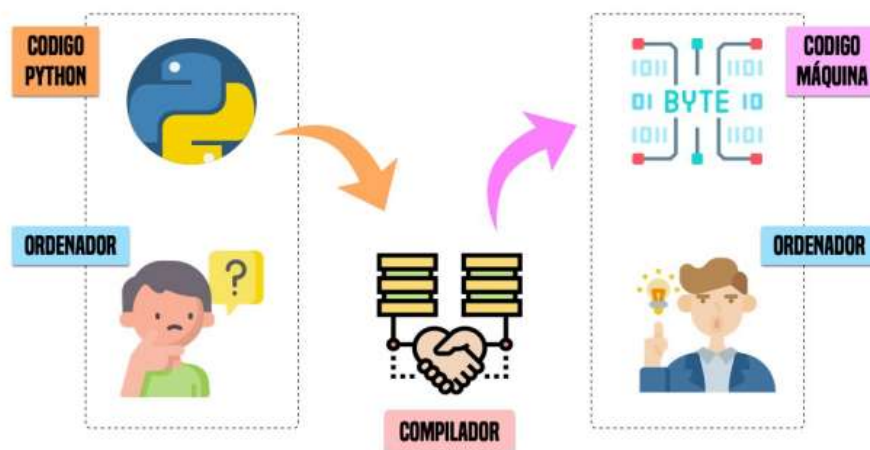


Ilustración 41 Esquema de funcionamiento de un compilador

CAPITULO 3. PROTOCOLOS DE COMUNICACIÓN, MODELO OSI, TCP/IP Y MODELO DE DATOS MODBUS.

3.1. PROTOCOLOS DE COMUNICACIÓN.

3.1.1. Descripción general de los protocolos.

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 redes con la codificación correspondiente para su transmisión. Para esto utilizamos 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.

3.1.1.1. El protocolo.

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 contruidos 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 y otro subconjunto que transmite información 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 información 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. 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.

3.1.2. Estándares del modelo OSI.

Los estándares del modelo OSI (Open System Interconnection) o simplemente modelo OSI, son una metodología teórica de comunicaciones, que buscaba definir cómo se podían comunicar dos ordenadores o dos dispositivos de red. Este modelo se divide en 7 capas OSI, donde cada una es responsable de cierto elemento de los procesos de comunicación en red y le ofrece a la capa siguiente, una herramienta clave.

Fue desarrollado hacia inicio de los años 80 por la Organización Internacional de Estándares ISO (International Organization for Standardization) con el objetivo de establecer un método en el que cualquier dispositivo electrónico que intentara comunicarse con otro en una red de comunicaciones, tuviera que seguir una serie de principios que facilitarían su labor.

De modo que los estándares del modelo OSI funciona como un estándar para los protocolos de red o reglas de comunicación para conectar equipos. Y aunque la red actual de internet pareciera no adaptarse por completo a este sistema o conjunto de reglas, el modelo OSI sigue siendo una herramienta de gran utilidad para la resolución de un gran número de los problemas de red que se presentan.

Por ejemplo, si se presentan casos problemáticos como el de un usuario no puede acceder a la conexión de internet a través de su dispositivo electrónico portátil o si ocurre una interrupción de una determinada página web o red social que le impide el acceso a un gran número de personas, las herramientas y planteamientos de los estándares del modelo OSI podrían resultar la pieza clave para la resolución de estos inconvenientes, debido a que sería más fácil la identificación de la causa del problema y su ubicación dentro de alguna de las capas de este modelo, lo cual se traduce en una reducción considerable del trabajo innecesario y del tiempo empleado en solucionar este conflicto en la red.

3.1.2.1 ¿Cómo funciona el modelo OSI?

El funcionamiento del Modelo OSI depende directamente de sus siete capas, en las que descompone el complicado proceso de la comunicación digital. Al compartimentarlo, asigna a cada capa funciones muy específicas, dentro de una estructura jerárquica fija.

Así, cada protocolo de comunicación emplea estas capas en su totalidad o sólo algunas de ellas, pero al obedecer este conjunto de reglas, garantiza que la comunicación entre las redes sea eficaz y sobre todo que se de en los mismos términos.

3.1.2.2. ¿Para qué sirve el modelo OSI?

El Modelo OSI es fundamentalmente una herramienta conceptual, de organización de las telecomunicaciones. Universaliza la manera en que la información es compartida entre redes

informáticas o sistemas computarizados, independientemente de su origen geográfico, empresarial u otras condiciones que podrían dificultar la comunicación de los datos.

El Modelo OSI no es una topología de red, ni un modelo de red en sí mismo, ni una especificación de protocolos; simplemente es una herramienta que define la funcionalidad de los protocolos, para conseguir un estándar de comunicación, o sea, lograr que todos los sistemas hablen el mismo idioma.

3.1.2.3 Capas del modelo OSI

Las siete capas o niveles del modelo OSI son los siguientes:



Ilustración 42 Capa modelo OSI

3.1.2.3.1. Capa física.

La capa más baja del modelo, se encarga de la topología de red y las conexiones globales entre la computadora y la red, refiriéndose tanto al medio físico y características eléctricas de los diversos componentes; de la velocidad de transmisión, si esta es uni o bidireccional. También de aspecto mecánicos de las conexiones y terminales, incluyendo la interpretación de las señales eléctricas.

Esta capa es la responsable de la conversión de bits y bytes (1 y 0) a una representación física impulsos eléctricos, ondas o señales ópticas y de la reconversión de estas representaciones en bits en el lado de la recepción, estos son algunos de sus componentes: tarjeta de red, medios de transmisión (físico e inalámbrico), switches, hubs, conectores.



Ilustración 43 Medio físico de transporte

El nivel físico recibe una trama binaria que debe convertir a una señal eléctrica, electromagnética u otra dependiendo del medio, de tal forma que a pesar de la degradación que pueda sufrir en el medio de transmisión vuelva a ser interpretable correctamente en el receptor. En el caso mas sencillo el medio es directamente digital, como en el caso de fibra óticas, dado que por ellas se transmiten pulsos de luz.

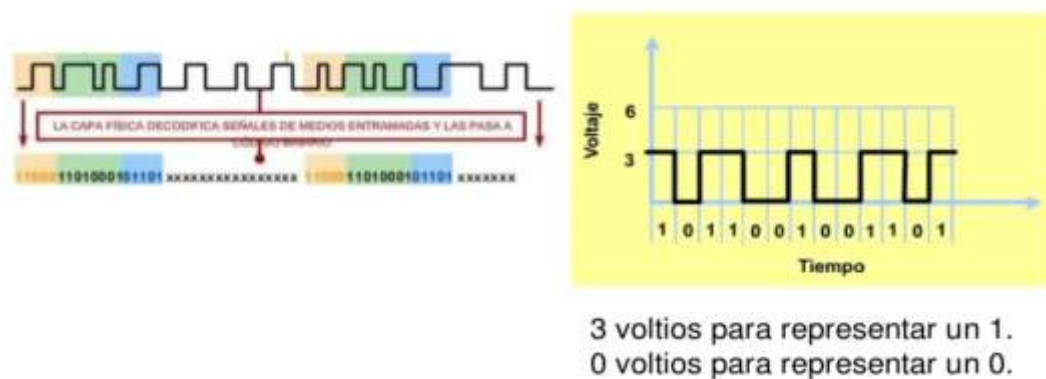


Ilustración 44 Codificación de la señal

3.1.2.3.2 Capa de enlace de datos.

Se ocupa del direccionamiento físico, detección de errores, estructura de la trama, acceso al medio y control del flujo durante la comunicación, siendo parte de la creación de protocolos básicos para regular la conexión entre los sistemas informáticos. Se ocupa de la determinación de como negociar los accesos a un único canal de datos, cuando varios anfitriones intentan utilizarlo

al mismo tiempo. Todos los componentes deben de tener una dirección física en redes LAN se llama MAC.

Esta capa tiene como funciones la detección de errores, determina si han surgido problemas en un paquete durante la transmisión (utiliza FCS) se debe de tener los dos en la misma transmisión de los datos la misma trama por ej. 802.2,802.3,802.5 para que exista comunicación, también traslada los mensajes desde la capa física a la capa de red, se encarga del direccionamiento local, la capa de datos se divide en 2 partes:

- Control lógico de enlace LLC (logical link control).
- Control de acceso al medio MAC (medium Access control)

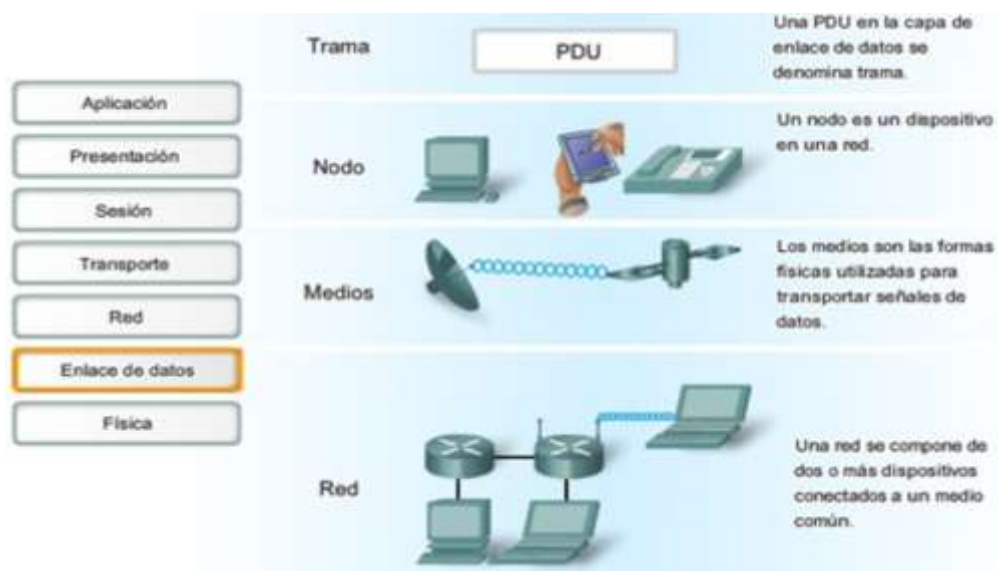


Ilustración 45 Enlace de datos

3.1.2.3.3. Capa de red.

Es la capa que se encarga de la identificación del enrutamiento existente entre las redes involucradas, así, las unidades de datos pasan a denominarse “paquetes” y pueden clasificarse conforme al protocolo de enrutamiento o protocolo enrutable que utilizan.

Los primeros seleccionan las rutas (RIP, IGRP, EIGRP, entre otras) y los segundos viajan con los paquetes (IP, IPX, APPLETTALK, etc.). El objetivo de esta capa es garantizar que los datos lleguen a su destino, incluso si ello implica utilizar dispositivos intermedios, como encaminadores o enrutadores.

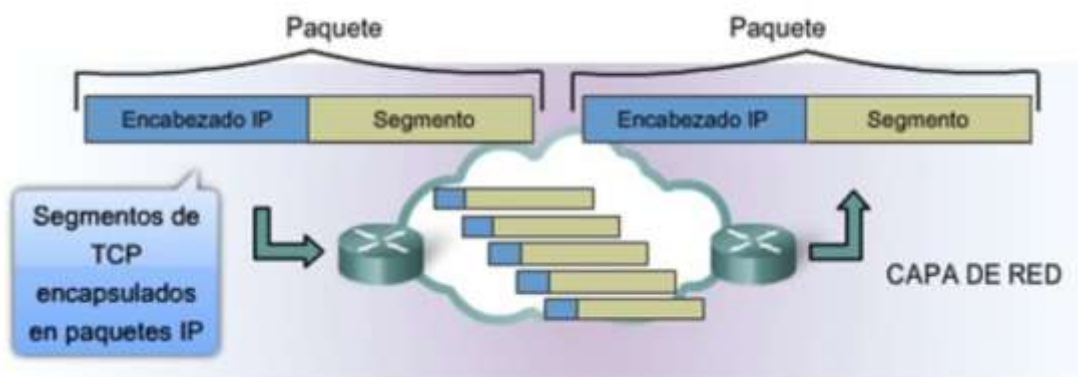


Ilustración 46 Capa de red

3.1.2.3.4. Capa de transporte.

La capa de transporte garantiza que los mensajes se entregan sin errores, en secuencia y sin pérdidas o duplicaciones. Libera a los protocolos de capas superiores de cualquier cuestión relacionada con la transferencia de datos entre ellos y sus pares.

El tamaño y la complejidad de un protocolo de transporte depende del tipo de servicio que pueda obtener de la capa de transporte. Para tener una capa de transporte confiable con una capacidad de circuito virtual, se requiere una mínima capa de transporte. Si la capa de red no es confiable o solo admite datagramas, el protocolo de transporte debería incluir detección y recuperación de errores extensivos.

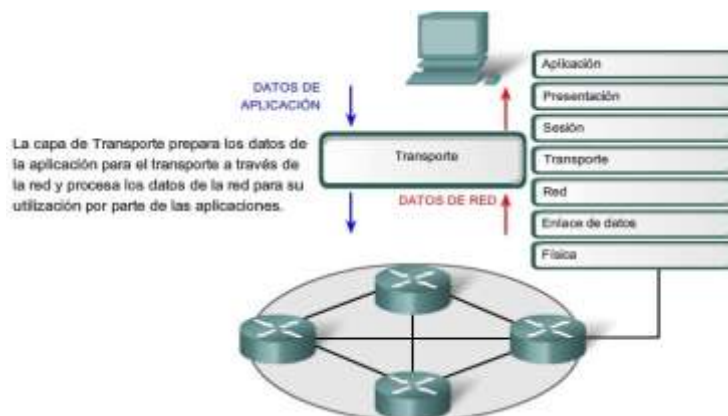


Ilustración 47 Capa de transporte

La capa de transporte proporciona:

- Segmentación de mensajes: acepta un mensaje de la capa (de sesión) que tiene por encima, lo divide en unidades más pequeñas (si no es aún lo suficientemente pequeño) y transmite las unidades más pequeñas a la capa de red. La capa de transporte en la estación de destino vuelve a ensamblar el mensaje.
- Confirmación de mensaje: proporciona una entrega de mensajes confiable de extremo a extremo con confirmaciones.
- Control del tráfico de mensajes: indica a la estación de transmisión que "dé marcha atrás" cuando no haya ningún búfer de mensaje disponible.
- Multiplexación de sesión: multiplexa varias secuencias de mensajes, o sesiones, en un vínculo lógico y realiza un seguimiento de qué mensajes pertenecen a qué sesiones (consulte la capa de sesiones).

3.1.2.3.5. Capa de sesión.

Es la encargada de establecer el enlace de comunicación o sesión entre las computadoras emisora y receptora. Esta capa permite que los usuarios de diferentes máquinas puedan establecer sesiones entre ellos. Una sesión permitirá al usuario acceder a un sistema de tiempo compartido a

distancia, o transferir un archivo entre dos máquinas. En este nivel se gestiona el control del diálogo. Además, esta capa se encarga de la administración de testigos y los objetos entre el origen y destino de los datos. En esta capa se ubican los gateways y el software. Las funciones esenciales son:

- Esta encargada de proporcionar sondas sonoras y gestiones de testigos.
- Establece, administra y finaliza las sesiones entre dos host que se están comunicando.
- Restaura la sesión a partir de un punto seguro y sin pérdida de datos.
- Sincroniza el dialogo entre las capas de presentación de los host y administra su intercambio de datos.
- Ofrece disposiciones para una transferencia de datos eficiente.
- Fichas de Maneja.
- Hacer puntos de control.
- Cronometra y controla el flujo.
- Coordinar el intercambio de información entre sistemas mediante técnicas de conversación o diálogo.
- Puede ser usado para realizar un login a un sistema de tiempo compartido remoto.
- Permite que los usuarios de diferentes máquinas puedan establecer sesiones entre ellos.

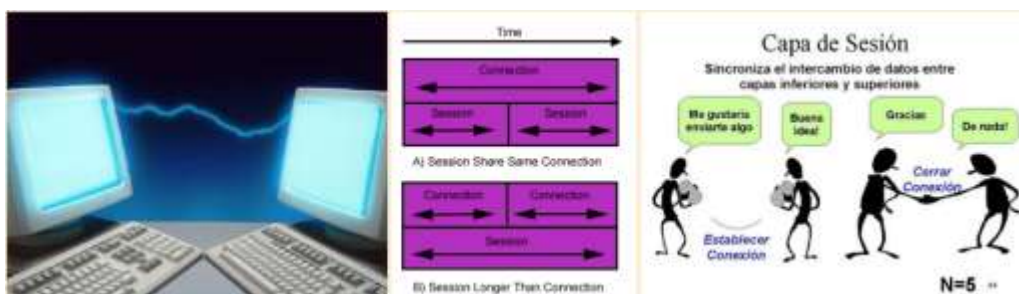


Ilustración 48 Capa sesión

3.1.2.3.6. Capa de presentación.

Esta capa se ocupa de la representación de la información, de manera que aunque distintos equipos puedan tener diferentes representaciones internas de caracteres los datos lleguen de manera reconocible, sin importar el tipo de sistema empleado. Es la primera capa que se ocupa del contenido de la transmisión, en vez del modo en que ésta se establece y se sostiene. En ella se tratan aspectos tales como la semántica y la sintaxis de los datos transmitidos, ya que distintas computadoras pueden tener diferentes formas de manejarlas. La capa de presentación es la encargada de:

- Definir el formato de los datos que se van a intercambiar entre las aplicaciones y ofrecer un conjunto de servicios de transformación de datos.
- Definir la sintaxis utilizada entre entidades de aplicación y proporcionar los medios para la selección y modificación de la representación utilizada.
- Codificar los datos en modo estándar (enteros, reales, caracteres, etc.) y realizar funciones de compresión y cifrado de datos.

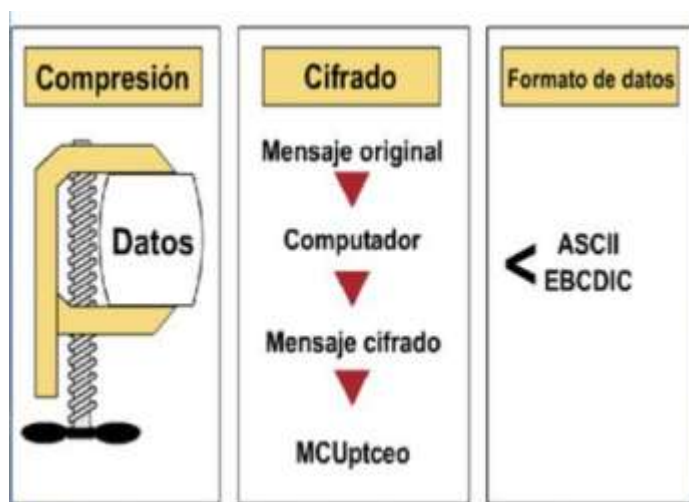


Ilustración 49 Función de la capa presentación.

3.1.2.3.7. Capa de aplicación.

Es la capa más cercana al usuario, y está relacionada con las funciones de más alto nivel que proporcionan soporte a las aplicaciones o actividades del sistema, suministrando servicios de red a las aplicaciones del usuario y definiendo los protocolos usados por las aplicaciones individuales. Su función principal es proporcionar los procedimientos precisos que permitan a los usuarios ejecutar los comandos relativos a sus propias aplicaciones. Los procesos de las aplicaciones se comunican entre si por medio de las entidades de aplicación asociadas, estando estas controladas por protocolos de aplicación, y utilizando los servicios de nivel de presentación.

La capa de aplicación establece la disponibilidad de los diversos elementos que deben participar en la comunicación sincroniza las aplicaciones que cooperan entre si y establece acuerdos sobre los procedimientos de recuperación de errores y control de la integridad de los datos.

Algunos ejemplos de proceso de aplicaciones son:

- Programas de hojas de cálculo.
- Programas de procesamiento de texto.
- Transferencia de archivos (ftp).
- Login remoto (rlogin, telnet).
- Correo electrónico (mail – smtp).
- Páginas web (http).

La capa de aplicación es donde ocurre toda la interacción del usuario con la computadora



Ilustración 50 Capa de aplicación

3.1.3. Modelo de arquitectura del protocolo TCP/IP

El modelo OSI describe las comunicaciones de red ideales con una familia de protocolos, TCP/IP no se corresponde directamente con este modelo TCP/IP combina varias capas OSI en una única capa, o no utiliza determinadas capas. La tabla enumera las capas desde la capa superior (aplicación) hasta la capa inferior (red física).

Ref. OSI N° de capa	Equivalente de capa OSI	Capa TCP/IP	Ejemplos de protocolos TCP/IP
5,6,7	Aplicación, sesión, presentación	Aplicación	NFS, NIS, DNS, LDAP, telnet, ftp, rlogin, rsh, rcp, RIP, RD/SC, SNMP y otros.
4	Transporte	Transporte	TCP, UDP, SCTP
3	Red	Internet	IPv4, IPv6, ARP, ICMP
2	Vínculo de datos	Vínculo de datos	PPP, IEEE 802.2
1	Física	Red física	Ethernet (IEEE 802.3), Token Ring, RS-232, FDDI y otros.

Tabla 1 Pila de protocolo TCP / IP

La tabla muestra las capas de protocolo TCP/IP y los equivalentes del modelo OSI. También se muestran ejemplos de los protocolos disponibles en cada nivel de la pila del protocolo TCP/IP. Cada sistema que participa en una transacción de comunicación ejecuta una única implementación de la pila del protocolo.

3.1.3.1. Capa de red física

La capa de red física especifica las características del hardware que se utilizará para la red. Por ejemplo, la capa de red física especifica las características físicas del medio de comunicaciones. La capa física de TCP/IP describe los estándares de hardware como IEEE 802.3, la especificación del medio de red Ethernet, y RS-232, la especificación para los conectores estándar.

3.1.3.2. Capa de vínculo de datos

La capa de vínculo de datos identifica el tipo de protocolo de red del paquete, en este caso TCP/IP. La capa de vínculo de datos proporciona también control de errores y estructuras. Algunos ejemplos de protocolos de capa de vínculo de datos son las estructuras Ethernet IEEE 802.2 y Protocolo punto a punto (PPP).

3.1.3.3. Capa de Internet

La capa de Internet, también conocida como capa de red o capa IP, acepta y transfiere paquetes para la red. Esta capa incluye el potente Protocolo de Internet (IP), el protocolo de resolución de direcciones (ARP) y el protocolo de mensajes de control de Internet (ICMP).

3.1.3.3.1. Protocolo IP

El protocolo IP y sus protocolos de enrutamiento asociados son posiblemente la parte más significativa del conjunto TCP/IP. El protocolo IP se encarga de:

- **Direcciones IP:** Las convenciones de direcciones IP forman parte del protocolo IP.
- **Comunicaciones de host a host:** El protocolo IP determina la ruta que debe utilizar un paquete, basándose en la dirección IP del sistema receptor.
- **Formato de paquetes:** el protocolo IP agrupa paquetes en unidades conocidas como datagramas.

- **Fragmentación:** Si un paquete es demasiado grande para su transmisión a través del medio de red, el protocolo IP del sistema de envío divide el paquete en fragmentos de menor tamaño. el protocolo IP del sistema receptor reconstruye los fragmentos y crea el paquete original.

3.1.3.3.2. Protocolo ARP

El protocolo de resolución de direcciones (ARP) se encuentra conceptualmente entre el vínculo de datos y las capas de Internet. ARP ayuda al protocolo IP a dirigir los datagramas al sistema receptor adecuado asignando direcciones Ethernet (de 48 bits de longitud) a direcciones IP conocidas (de 32 bits de longitud).

3.1.3.3.3. Protocolo ICMP

El protocolo de mensajes de control de Internet (ICMP) detecta y registra las condiciones de error de la red. ICMP registra:

- **Paquetes soltados:** Paquetes que llegan demasiado rápido para poder procesarse.
- **Fallo de conectividad:** No se puede alcanzar un sistema de destino.
- **Redirección:** Redirige un sistema de envío para utilizar otro enrutador.

3.1.3.4. Capa de transporte

La capa de transporte TCP/IP garantiza que los paquetes lleguen en secuencia y sin errores, al intercambiar la confirmación de la recepción de los datos y retransmitir los paquetes perdidos. Este tipo de comunicación se conoce como transmisión de punto a punto.

Los protocolos de capa de transporte de este nivel son el Protocolo de control de transmisión (TCP), el Protocolo de datagramas de usuario (UDP) y el Protocolo de transmisión para el control de flujo (SCTP). Los protocolos TCP y SCTP proporcionan un servicio completo y fiable. UDP proporciona un servicio de datagrama poco fiable.

3.1.3.4.1. Protocolo TCP

TCP permite a las aplicaciones comunicarse entre sí como si estuvieran conectadas físicamente. TCP envía los datos en un formato que se transmite carácter por carácter, en lugar de transmitirse por paquetes discretos. Esta transmisión consiste en lo siguiente:

- Punto de partida, que abre la conexión.
- Transmisión completa en orden de bytes.
- Punto de fin, que cierra la conexión.

TCP conecta un encabezado a los datos transmitidos. Este encabezado contiene múltiples parámetros que ayudan a los procesos del sistema transmisor a conectarse a sus procesos correspondientes en el sistema receptor.

TCP confirma que un paquete ha alcanzado su destino estableciendo una conexión de punto a punto entre los hosts de envío y recepción. Por tanto, el protocolo TCP se considera un protocolo fiable orientado a la conexión.

3.1.3.4.2. Protocolo SCTP

SCTP es un protocolo de capa de transporte fiable orientado a la conexión que ofrece los mismos servicios a las aplicaciones que TCP. Además, SCTP admite conexiones entre sistema que tienen más de una dirección, o de host múltiple. La conexión SCTP entre el sistema transmisor y receptor se denomina asociación. Los datos de la asociación se organizan en bloques. Dado que el protocolo SCTP admite varios hosts, determinadas aplicaciones, en especial las que se utilizan en el sector de las telecomunicaciones, necesitan ejecutar SCTP en lugar de TCP.

3.1.3.4.3. Protocolo UDP

UDP proporciona un servicio de entrega de datagramas. UDP no verifica las conexiones entre los hosts transmisores y receptores. Dado que el protocolo UDP elimina los procesos de

establecimiento y verificación de las conexiones, resulta ideal para las aplicaciones que envían pequeñas cantidades de datos.

3.1.3.5. Capa de aplicación

La capa de aplicación define las aplicaciones de red y los servicios de Internet estándar que puede utilizar un usuario. Estos servicios utilizan la capa de transporte para enviar y recibir datos. Existen varios protocolos de capa de aplicación. En la lista siguiente se incluyen ejemplos de protocolos de capa de aplicación:

- Servicios TCP/IP estándar como los comandos ftp, tftp y telnet.
- Comandos UNIX "r", como rlogin o rsh.
- Servicios de nombres, como NIS o el sistema de nombre de dominio (DNS).
- Servicios de directorio (LDAP).
- Servicios de archivos, como el servicio NFS.
- Protocolo simple de administración de red (SNMP), que permite administrar la red.
- Protocolo RDISC (Router Discovery Server) y protocolos RIP (Routing Information Protocol).

3.1.3.5.1. Servicios TCP/IP estándar

- **FTP y FTP anónimo:** El Protocolo de transferencia de archivos (FTP) transfiere archivos a una red remota y desde ella. El protocolo incluye el comando ftp y el daemon in.ftpd. FTP permite a un usuario especificar el nombre del host remoto y las opciones de comandos de transferencia de archivos en la línea de comandos del host local.
- **Telnet:** El protocolo Telnet permite la comunicación entre los terminales y los procesos orientados a los terminales de una red que ejecuta TCP/IP.

- **TFTP:** el protocolo de transferencia de archivos trivial (tftp) ofrece funciones similares a ftp, pero no establece la conexión interactiva de ftp.

3.1.3.5.2. Comandos UNIX "r"

Los comandos UNIX "r" permiten a los usuarios ejecutar comandos en sus equipos locales que se ejecutan en el host remoto. Estos comandos incluyen:

- rcp
- rlogin
- rsh

3.1.3.5.3. Servicios de nombres

- **DNS:** El sistema de nombre de dominio (DNS) es el servicio de nombres que proporciona Internet para las redes TCP/IP. DNS proporciona nombres de host al servicio de direcciones IP.
- **Archivos /etc :** El sistema de nombres UNIX basado en host se desarrolló para equipos UNIX autónomos y posteriormente se adaptó para el uso en red.
- **NIS:** El Servicio de información de la red (NIS) se desarrolló independientemente de DNS y tiene un enfoque ligeramente distinto. Mientras que DNS trata de facilitar la comunicación con el uso de nombres de equipos en lugar de direcciones IP numéricas, NIS se centra en facilitar la administración de la red al proporcionar control centralizado sobre distintos tipos de información de red.

3.1.3.5.4. Servicio de directorios

LDAP (Protocolo ligero de acceso a directorios) junto con el servidor de directorios Sun ONE (Sun Open Net Environment), así como otros servidores de directorios LDAP. La diferencia entre un servicio de nombres y un servicio de directorios radica en la extensión de las funciones.

3.1.3.5.5. Servicios de archivos

El protocolo de capa de aplicación NFS proporciona servicios de archivos.

3.1.3.5.6. Administración de la red

El Protocolo simple de administración de red (SNMP) permite ver la distribución de la red y el estado de los equipos clave. SNMP también permite obtener estadísticas de red complejas del software basado en una interfaz gráfica de usuario (GUI). Muchas compañías ofrecen paquetes de administración de red que implementan SNMP.

3.1.3.5.7. Protocolos de enrutamiento

Los protocolos RIP y RDISC son dos protocolos de enrutamiento disponibles para las redes TCP/IP.

3.1.4. TCP/IP Modbus

El servicio de mensajería Modbus proporciona una comunicación cliente/servidor entre los dispositivos conectados a una red Ethernet TCP/IP.

El modelo cliente/servidor está basado en cuatro tipos de mensajes:

- **Solicitud Modbus**, el mensaje enviado en la red por el cliente para iniciar una transacción.
- **Indicación Modbus**, el mensaje de solicitud recibido en el lado del servidor.
- **Respuesta Modbus**, el mensaje de respuesta enviado por el servidor.
- **Confirmación Modbus**, el mensaje de respuesta recibido en el lado del cliente.

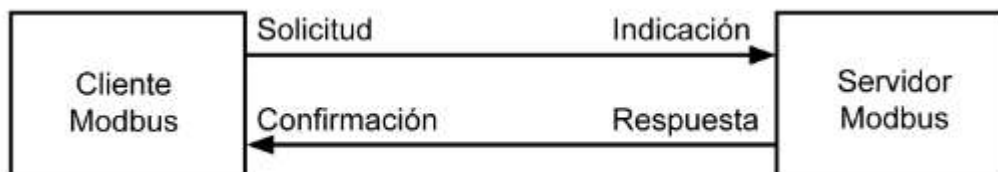


Ilustración 51 Mensaje Modbus.

Los servicios de mensajería Modbus (modelo cliente/servidor) se utilizan para el intercambio de información en tiempo real entre:

- Dos aplicaciones de dispositivos.
- Una aplicación de dispositivo y otro dispositivo.
- Dispositivos y aplicaciones HMI/SCADA.

Un PC y un programa de dispositivo que proporcione servicios en línea.

Se utiliza una cabecera especializada en TCP/IP para identificar la Unidad de datos de aplicación Modbus. Este dispositivo recibe el nombre de cabecera MBAP (cabecera Protocolo de aplicación Modbus).

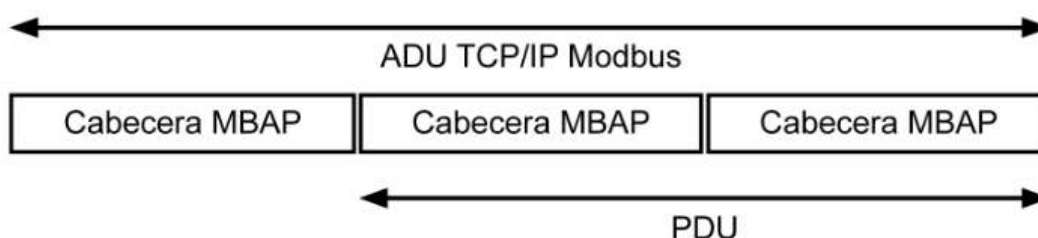


Ilustración 52 Cabecera protocolo de aplicación Modbus

La cabecera MBAP contiene los siguientes campos:

Campos	Longitud	Descripción	Cliente	Servidor
Identificador de transacción	2 bytes	Identificación de una transacción de solicitud/respuesta Modbus	Inicializado por el cliente	Copiado nuevamente por el servidor a partir de la solicitud recibida
Identificador de protocolo	2 bytes	0 = Protocolo Modbus	Inicializado por el cliente	Copiado nuevamente por el servidor a partir de la solicitud recibida
Longitud	2 bytes	Número de bytes siguientes	Inicializado por el cliente (solicitud)	Inicializado por el servidor (respuesta)
Identificador de unidad	1 byte	Identificación de un esclavo a distancia conectado a una línea serie u otros buses	Inicializado por el cliente	Copiado nuevamente por el servidor a partir de la solicitud recibida

Tabla 2 Cabecera MB

3.1.4.1. Modbus de control con protocolo TCP

Muchas aplicaciones disponibles en Internet utilizan ampliamente TCP, incluidas www, correo electrónico, protocolo de transferencia de archivos, Secure Shell, uso compartido de archivos punto a punto y aplicaciones de transmisión de medios.

Puede ver un diagrama TCP simple que muestra una aplicación www donde dos computadoras se comunican en Internet. Este diagrama contiene los tipos más comunes de comunicación física, pero la tecnología Wi-Fi también es una forma de comunicación ampliamente aceptada entre el enrutador y la PC.

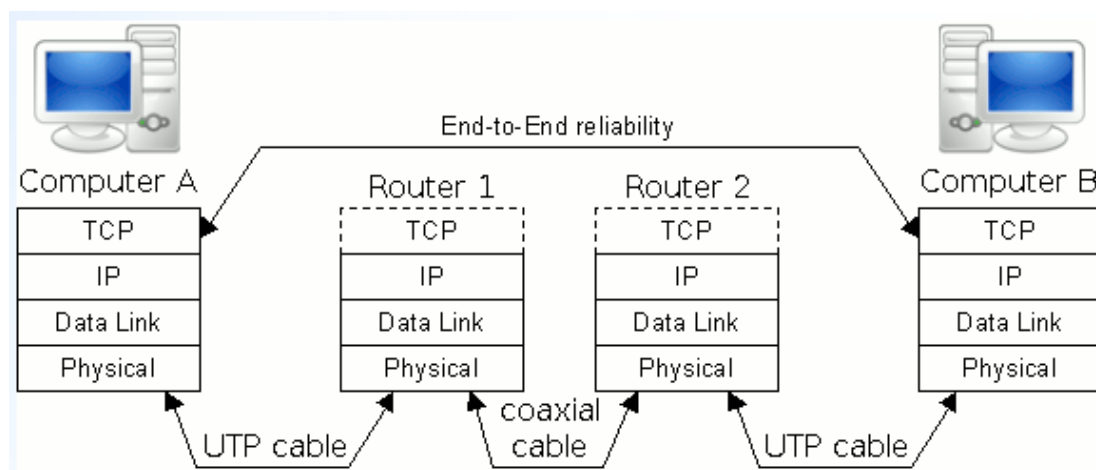


Ilustración 53 Flujo de trabajo de conexión TCP.

Cada capa física está conectada a la otra con algún tipo de método de transferencia como cable UTP o Wi-Fi. La capa física también se considera el dominio de muchos problemas de diseño de red relacionados con hardware, como topología LAN y WAN.

El par trenzado sin blindaje (UTP) se usa principalmente en la capa física. Un subconjunto típico de estos colores (blanco/azul, azul/blanco, blanco/naranja, naranja/blanco) aparece en la mayoría de los cables UTP.



Ilustración 54 Conector RJ45 y cable UTP

Los cables generalmente están hechos con alambres de cobre medidos en calibre de alambre estadounidense (AWG) 22 o 24, con el aislamiento de color generalmente hecho de un aislante

como polietileno o FEP y el paquete total cubierto con una cubierta de polietileno. UTP utiliza pares de hilos trenzados.

El cableado de par trenzado es un tipo de cableado en el que dos conductores de un solo circuito se trenzan entre sí con el fin de mejorar la compatibilidad electromagnética. El cableado de par trenzado se usa a menudo en redes de datos para conexiones de longitud corta y media debido a sus costos relativamente más bajos en comparación con la fibra óptica y el cable coaxial.

Wi-Fi es otro tipo de capa física. Funciona sobre la base del estándar IEEE 802.11. La arquitectura del protocolo consiste en una capa física con dos subcapas PLCP y PMD y la capa de enlace de datos consta de una subcapa MAC y LLC. La red Wi-Fi consta de un cliente Wi-Fi, un punto de acceso y una LAN. La LAN puede consistir en un conmutador Ethernet, un servidor de autenticación RADIUS, etc. Básicamente, Wi-Fi funciona en modo semidúplex y el estándar de Wi-Fi consiste solo en la capa física y la capa de enlace de datos.

La capa física puede transferir paquetes TCP o UDP, aunque esta página describe solo TCP, así que continuemos examinando el enlace TCP entre dos estaciones (por ejemplo, una computadora). También verá cómo se ve el encabezado de un paquete TCP y cómo se envía un paquete TCP entre estaciones.

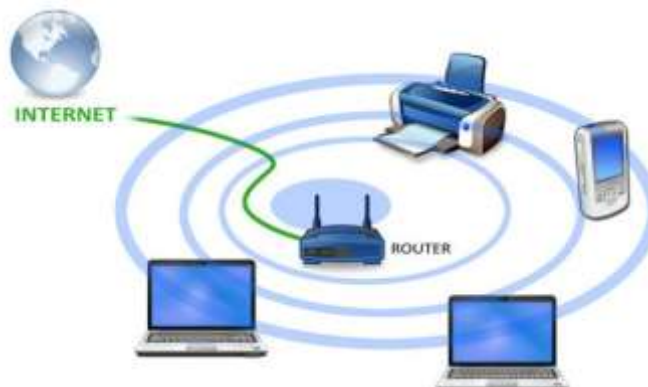


Ilustración 55 Red wi-fi en una oficina formal

La capa física puede transferir paquetes TCP o UDP, aunque esta página describe solo TCP, así que continuemos examinando el enlace TCP entre dos estaciones (por ejemplo, una computadora). También verá cómo se ve el encabezado de un paquete TCP y cómo se envía un paquete TCP entre estaciones.

TCP agrega una gran cantidad de funcionalidad al servicio IP

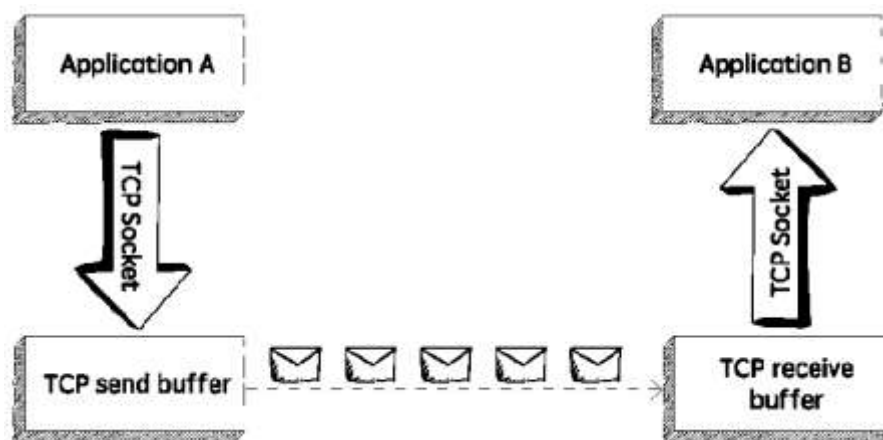


Ilustración 56 Control de flujo.

Flujos: los datos TCP se organizan como un flujo de bytes, como un archivo. La naturaleza del datagrama de la red está oculta. Existe un mecanismo (el puntero urgente) para permitir que los datos fuera de banda se marquen especialmente.

Entrega confiable: los números de secuencia se utilizan para coordinar qué datos se han

transmitido y recibido. TCP se encargará de la retransmisión si determina que se han perdido los datos.

Adaptación de la red: TCP aprenderá dinámicamente las características de retraso de una red y ajustará su funcionamiento para maximizar el rendimiento sin sobrecargar la red.

Control de flujo: TCP administra los búferes de datos y coordina el tráfico para que sus búferes nunca se desborden. Los remitentes rápidos se detendrán periódicamente para mantenerse al día con los receptores más lentos.

3.1.4.2. Operación dúplex completo

No importa cuál sea la aplicación en particular, TCP casi siempre opera en dúplex completo. A veces es útil pensar en una sesión TCP como dos flujos de bytes independientes que viajan en direcciones opuestas. No existe ningún mecanismo TCP para asociar datos en los flujos de bytes directos e inversos. Solo durante las secuencias de inicio y cierre de la conexión, TCP puede exhibir un comportamiento asimétrico (es decir, la transferencia de datos en la dirección de avance, pero no en la inversa, o viceversa).



Ilustración 57 Conexión dúplex completo

3.1.4.3. Encabezado TCP

El encabezado se incluyen un número de secuencia de 32 bits y un número de reconocimiento, así como varios bits de marca. Dado que el encabezado TCP puede tener una longitud variable según el contenido del campo de opciones, se ha proporcionado un campo de compensación de datos para determinar el comienzo real de los datos, esto normalmente se hace con todos los encabezados dentro de la pila TCP/IP. Esta función se utiliza para el control de flujo, que intenta

evitar el desbordamiento de búfer en el receptor, los datos siguen al encabezado TCP, el encabezado puede tener hasta 40 bytes de longitud y el encabezado con sus datos adjuntos se denomina segmento.

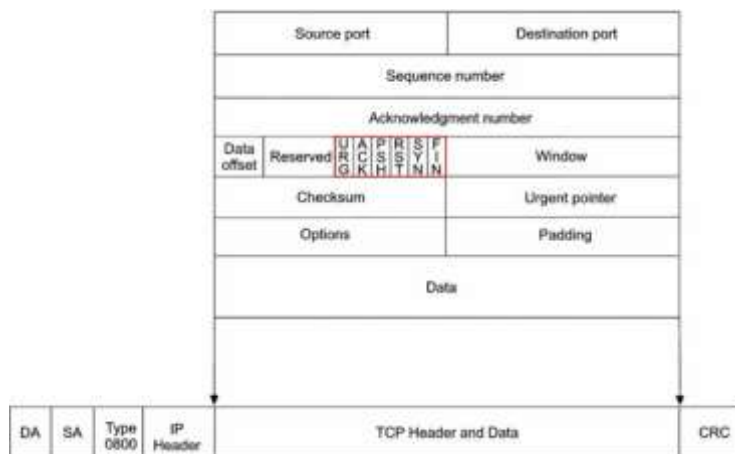


Ilustración 58 Encabezado TCP y datos encapsulados en el campo de datos IP.

El encabezado TCP contiene varios campos booleanos de un bit conocidos como indicadores para influir en el flujo de datos:

URG: un segmento es urgente y debe priorizarse

ACK - Reconoce los datos recibidos

PSH: informa al host receptor que los datos deben enviarse a la aplicación receptora

RST: cancela una conexión en respuesta a un error

SYN - Inicia una conexión

FIN - Cierra una conexión

3.1.4.4. Uso de conexiones

Cuando se utiliza el protocolo TCP con procesos, primero se debe establecer y mantener una conexión para permitir el flujo de datos. Al establecer una conexión entre dos estaciones, se proporciona el área de búfer adecuada para los datos inminentes.

Cuando la estación A quiere comunicarse con la estación B, primero debe establecer una conexión que permita la sincronización de números de secuencia y reconocimientos. Una de las

banderas dentro del encabezado TCP es el bit SYN, que se usa para indicar el número de secuencia inicial cuando se establece una conexión. Esto informa al receptor para que sincronice sus medios de comprobación de errores con este número de secuencia. Por lo tanto, la estación A envía un segmento TCP con SYN configurado y su número de secuencia, que en este caso es 10, a la estación B.

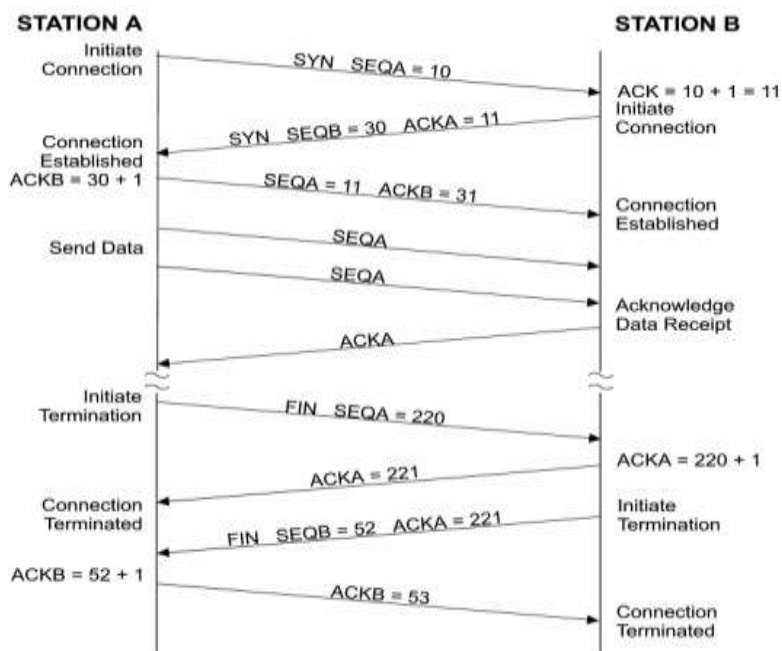


Ilustración 59 Conexión TCP desde el protocolo de enlace hasta la finalización

La estación B responde enviando un reconocimiento con el valor 11 para indicar que ese es el valor del número de secuencia que espera. La estación B tiene su propio número de secuencia que envía a la estación A. En este caso es 30 que la estación A reconoce enviando un 31. Esto establece dos conexiones. Una vez establecidas las conexiones, los datos se envían desde la estación A a la estación B con la estación A incrementando los números de secuencia y la estación B reconociéndolos. Esto es lo que se llama una conexión full-duplex ya que se establecieron dos conexiones; uno de A a B y otro de B a A.

Una vez que se haya completado la transferencia de datos requerida, las dos conexiones deben terminarse para liberar espacio de búfer en las dos estaciones. Hay una bandera llamada FIN, que

se usa para terminar la conexión. La estación A envía un segmento TCP con el indicador FIN enviado junto con un número de secuencia. La estación B reconoce la solicitud y una vez que se recibe el reconocimiento en la estación A, se termina la conexión de A a B. Esto no significa que se termine la conexión de B a A. Esto debe hacerse de la misma manera para terminar la conexión full-dúplex.

3.2 MODELO DE DATOS DE MODBUS.

En el protocolo Modbus existen bloques o rangos de direcciones que se usan para almacenar los datos y son, entradas discretas (Discretes input), salidas digitales, salidas discretas o bobinas (Coils o discrete outputs), registros de entrada (input registers) y registros de retención, registros internos o registros de salida (Holding registers o output register). Estos bloques definen el tipo y los permisos de accesos a los datos. Los esclavos tienen el acceso a estos datos puesto que la comunicación con el dispositivo de control es directo, y son alojados localmente. Estos datos por lo general residen en una parte de la memoria principal del esclavo, los maestros entonces, solicitan el acceso a estos datos a través de los códigos de función. Estos datos al momento de enviarse del esclavo al maestro forman parte de los datos asociados a la función dentro de Modbus PDU. El modelo de bloques solamente es conceptual, es decir cada esclavo define la forma de organización y acceso de los datos dentro de su memoria principal.

Tablas primarias	Tipo de objeto	Tipo	Descripción
Discretes input	bit	lectura	Este tipo de datos es dado por un sistema de entrada/salida (I/O).
Coils	bit	lectura - escritura	Este tipo de datos pueden ser modificados por un programa.
Input registers	16 bit (una palabra)	lectura	Este tipo de datos es dado por un sistema de entrada/salida (I/O).
Holding registers	16 bit (una palabra)	lectura - escritura	Este tipo de datos pueden ser modificados por un programa.

Tabla 3 Modelo de bloques de almacenamiento de datos en Modbus.

3.2.1. Funciones de Modbus.

Las funciones en Modbus permiten el acceso por parte del maestro a los bloques de datos del

clavo, ya sean funciones de lectura o escritura, una forma de llevar a cabo esto es que a través de las funciones el maestro tenga acceso a las ubicaciones estáticas de la memoria, aunque esta forma puede variar dependiendo de la implementación del protocolo Modbus. Existen tres categorías de códigos de función:

- **Públicos:** son códigos de función definidos por la organización Modbus.
- **Definidos por el usuario:** existen dos rangos de códigos de funciones, del 65 al 72 y del 100 al 110. Cada usuario puede diseñar su propia implementación de funciones.
- **Reservados:** estos códigos de función son utilizados en productos que ya no reciben soporte de algunas compañías y no estas disponibles para su uso.

En el siguiente tabal se listan algunos de los códigos de función públicos definidos por la organización Modbus, se especifica el tipo de acceso de los datos, ya sea que se acceda a un solo bit o a registros de 16 bits, además se muestran algunos códigos de diagnóstico, los cuales se utilizan para recabar datos diagnóstico o información de control del dispositivo esclavo.

Tipo			Función	Códigos de función	
				Código	Subcódigo
Acceso a los datos	Acceso por bit	Entradas físicas discretas	Read discrete inputs	02	
		Bits internos o salidas físicas discretas	Read coils	01	
			Write single coil	05	
			Write multiple coils	15	
	Acceso por 16 bits	Registros de entrada físicos	Read input register	04	
		Registros internos o registros físicos de salida	Read holding registers	03	
			Write single register	06	
			Write multiple registers	16	
			Read/write multiple registers	23	
			Mask write register	22	
			Read FIFO queue	24	
			Diagnóstico		Read exception status
Diagnostic	08	00-18,20			
Report server ID	17				
Read device identification	43	14			

Tabla 4 Extracto de los códigos de funciones públicos.

Se describen dos de las funciones de Modbus que son la función lectura de registros de salida (Read Holding Register), y la función Escritura de multiples entradas discretas (Write Multiple Coils).

3.2.1.1. Funcion Read Holding Register.

Esta función permite leer el contenido de bloques de datos contiguos de los registros de salida o Holding registers, de un dispositivo esclavo. El paquete Request especifica el inicio de la dirección del registro y el numero de registros de salida. Los registros empiezan en cero, y son direccionados de 0 a 15, no 1 a 16. En el paquete responde, los datos de respuesta están ordenados en 2 bytes por registro, para cada registro el primer byte contiene los bits más significativos y el segundo byte los bits menos significativos. A continuación, se muestra la forma en que están conformadas la PDU de los paquetes Request, Response y error o Excepción como respuesta.

Código de función	1 byte	0x03
Dirección inicial	2 bytes	0x0000 a 0xFFFF
Cantidad de salidas	2 bytes	0x0001 a 0x7D

Tabla 5 PDU del Request para el código de función 3

Código de función	1 byte	0x03
Contador de bytes	1 bytes	$2 \times N^*$
Valor del registro	$N^* \times 2$ bytes	

Tabla 6 PDU del Response para el código de función 3

Código de error	1 byte	0x83
Código de excepción	1 byte	01, 02, 03 o 04

Tabla 7 PDU de Excepción para el código de función 3

3.2.1.2. Funcion Write Multiple Coils.

Esta función permite forzar a cada salida discreta o Coil, dentro de una secuencia de Coils, a tener el valor de 1 o 0 (on u off) en un dispositivo esclavo. La PDU del Request indica la referencia del Coil que será forzado a tener un valor. Los Coils son direcciones empezando desde cero. El paquete Response regresará además del mismo código de función, la dirección de inicio, y la cantidad de Coils que fueron forzados a tener un valor. A continuación, se muestra la forma en que está conformadas la PDU de los paquetes Request, Response y error o Excepción como respuesta.

Código de función	1 byte	0x0F
Dirección inicial	2 bytes	0x0000 a 0xFFFF
Cantidad de salidas	2 bytes	0x0001 a 0x07B0
Contador de bytes	1 byte	N*
Valores de los datos	N * ×1 byte	

Tabla 8 PDU del Request para el código de función 15

Código de función	1 byte	0x0F
Dirección inicial	2 bytes	0x0000 a 0xFFFF
Cantidad de salidas	2 bytes	0x0001 a 07B0

Tabla 9 PDU del Response para el código de función 15

Código de error	1 byte	0x8F
Código de excepción	1 byte	01, 02, 03 o 04

Tabla 10 PDU del Response para el código de función 15

3.2.2. La trama Modbus.

Una trama, es una unidad de datos enviada entre dispositivos y cuyo conjunto conforma un mensaje. Esto es, los bits de un mensaje.

Su formato es variable, influyendo en este tanto el propio protocolo con su capa física de trabajo. Por ello, las tramas de mensajes de, por ejemplo, Modbus RTU y Modbus TCP son diferentes, aunque sí parecidas.

3.2.2.1. Campos de una trama Modbus

Las tramas Modbus se identifican por una serie de campos, que son los siguientes:

- **Device Address:** Dirección del dispositivo hacia donde se dirige el emisor.
- **Register Address:** Función que necesita realizar el emisor.
- **Function Code:** Dirección a la que queremos acceder.
- **Number of Registers:** Número de registros sobre los que queremos realizar la función.

Estos se envían siguiendo la secuencia.

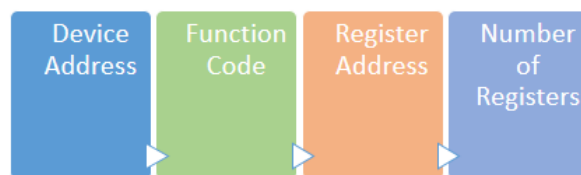


Ilustración 60 Secuencia de la trama

Además, se pueden añadir otros campos a continuación como es el Error Check.

- **Error Check:** El CRC es una verificación del mensaje, para comprobar que éste no tiene malformaciones. Esta verificación se corresponde con el checksum de la trama.

Estos son los campos que nos encontramos en un Modbus cualquiera. Después, dependiendo de si trabajamos con TCP, RTU, etc.

3.2.2.2. Trama Modbus TCP

Las tramas completas dependerán del medio que usemos para enviar nuestros mensajes Modbus. Por ejemplo, en Modbus TCP la trama Modbus viene dentro de una trama TCP, por lo que ésta es un poco más larga. Modbus TCP esta especificación permite a Modbus poder trabajar sobre arquitecturas modernas y rápidas, no limitando su uso a sistemas de control sino incluso haciendo Modbus valido para enviar tramas sobre la red empresarial.

CAPÍTULO 4 GUIAS DE LABORATORIO.

4.1 FORMATO DE GUÍAS DE LABORATORIO.

Las guías elaboradas están diseñadas para poder usar Raspberry Pi, como dispositivo de enlace de comunicación con el protocolo Modbus, que ejecuta programas con las instrucciones necesarias en lenguaje Python, para que el dispositivo logoV8 pueda controlar las entradas y salidas y en la interface gráfica para poder ejecutar aplicaciones en el laboratorio de electrónica.

En el marco de la investigación del proyecto de tesis, se aborda un contenido diseñado para fortalecer los conocimientos necesarios sobre el protocolo Modbus y su aplicación práctica. Cada guía práctica del proyecto se ajusta al establecido por la universidad para las guías de laboratorio.

A continuación, se muestran los ítems presentados para los laboratorios:



**UNIVERSIDAD TECNICA LATINOAMERICANA
FACULTAD DE INGENIERIA.**

PRACTICA DE LABORATORIO N°.

**TEMA:
LUGAR A IMPARTIR:**

- I. OBJETIVOS.
- II. INTRODUCCIÓN.
- III. MATERIALES Y EQUIPO.
- IV. PROCEDIMIENTO.
- V. PRUEBA DE CONOCIMIENTOS.
- VI. CONCLUSIONES DE LA PRÁCTICA.

4.2 GUÍA OPERACIÓN BASICA DE RASPBERRY PI.



**UNIVERSIDAD TECNICA LATINOAMERICANA
FACULTAD DE INGENIERIA.**

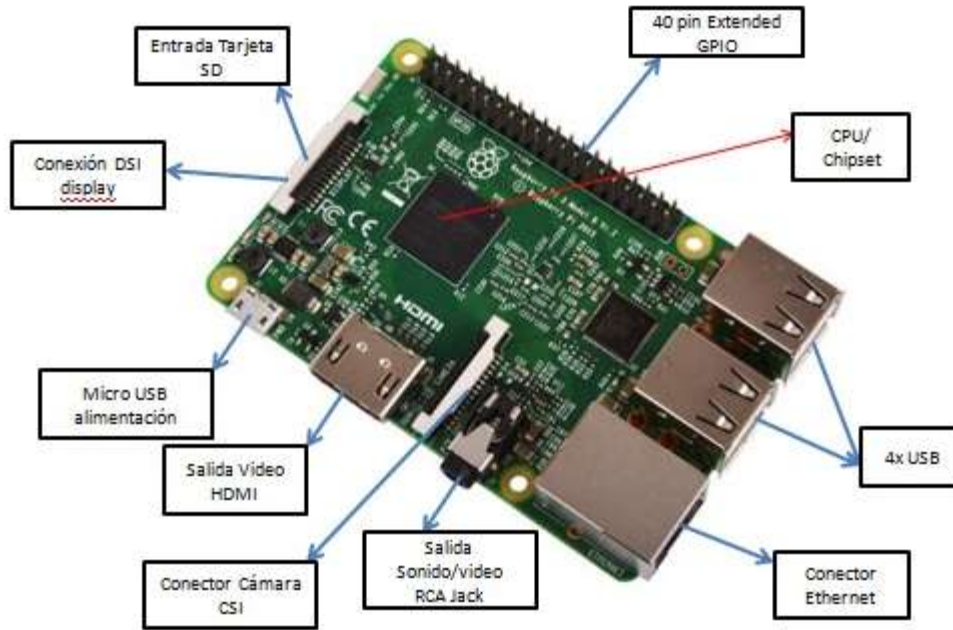
**PRACTICA DE LABORATORIO 1.
TEMA: CONOCIENDO LA RASPBERRY PI.
LUGAR A IMPARTIR: LABORATORIO DE ELECTRONICA.**

I. OBJETIVOS.

1. Conocer los componentes de la Raspberry Pi su tablero y sus ubicaciones.
2. Identificar los componentes y funciones principales de la Raspberry Pi
3. Enumerar las especificaciones de la Raspberry Pi
4. Mostrar el entorno grafico de una Raspberry Pi con sus comandos.

II. INTRODUCCIÓN.

Raspberry Pi es una pequeña computadora que tiene un tamaño similar a una placa Arduino, creada por la Fundación Raspberry Pi, que viene con CPU, GPU, puertos USB y pines de E/S. Es capaz de realizar algunas funciones simples como una computadora normal también para desarrollar proyectos de electrónica. Raspberry Pi (Modelo B de primera generación) entró en juego en febrero del 2012 y pronto obtuvo una reputación reconocida en términos de facilidad de uso y disponibilidad. De manera similar, Raspberry Pi (segunda generación) se presentó en febrero de 2015 y ofrecerá pocas mejoras en el diseño con RAM adicional que su versión anterior. Raspberry Pi 3 Modelo B se presentó en 2016, viene con un procesador de cuatro núcleos que muestra un rendimiento sólido que es 10 veces mayor que el de Raspberry Pi 1. Y la velocidad exhibida por Raspberry Pi 3 es un 80% mayor que la de Raspberry Pi 2.

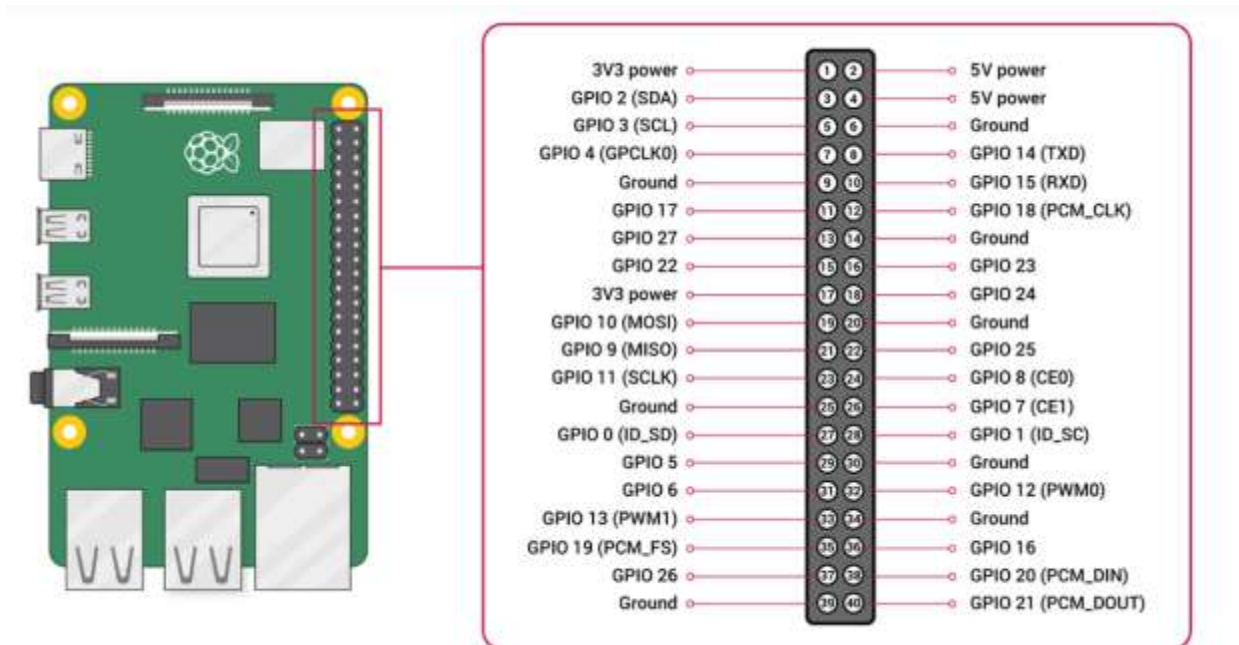


El hardware de Raspberry ha pasado por una serie de variaciones en términos de compatibilidad con dispositivos periféricos y capacidad de memoria. Cada nueva incorporación viene con una pequeña mejora en términos de diseño, donde se agregan funciones avanzadas en el dispositivo para que pueda realizar tantas funciones como sea posible como una computadora normal.

Configuración de pines de Raspberry Pi

La configuración de los pines de la placa Raspberry Pi, están preparadas para trabajar con componentes electrónicos, mediante sus pines. General Purpose Input Output (GPIO) es un sistema de entrada y salida de propósito general, es decir, consta de una serie de pines o conexiones que se pueden usar como entradas o salidas para múltiples usos.

Estos pines se pueden ver en dos filas de 20 pines. En la siguiente imagen podemos ver más claramente a que corresponde cada pin.



Estos son los 40 pines GPIO que se utilizan para la conexión con otros dispositivos. Se puede observar en la figura anterior que los pines UART (TXD y RXD) son los pines dedicados a la entrada y salida en serie que se utilizan para la comunicación en serie para datos y para la conversión de código de depuración.

Especificaciones de hardware

Raspberry Pi 3 Modelo B viene con un procesador de cuatro núcleos de 64 bits, funciones WiFi y Bluetooth y USB integradas, tiene una velocidad de procesamiento que va desde los 700 MHz hasta los 1,4 GHz donde la memoria RAM va desde los 256 hasta los 1GB.

La CPU de este dispositivo se considera como el cerebro del dispositivo que es responsable de ejecutar una cantidad de instrucciones basadas en operaciones matemáticas y lógicas. La GPU es otro chip avanzado incorporado en la placa que realiza la función de cálculo de imágenes. La placa está equipada con un cable central de video Broadcam que se utiliza principalmente para jugar videojuegos a través del dispositivo.

La Raspberry Pi 3 viene con pines GPIO (entrada y salida de uso general) que son esenciales para mantener la conexión con otros dispositivos electrónicos. Estos pines de entrada y salida reciben comandos y funcionan en función de la programación del dispositivo.

El puerto Ethernet está incorporado en este dispositivo y establece una vía para comunicarse con otros dispositivos. Puede conectar el puerto Ethernet al enrutador para mantener una conexión a Internet. La placa tiene cuatro puertos USB que se utilizan para la comunicación y se agrega una tarjeta SD para almacenar el sistema operativo.

El conector de fuente de alimentación es una parte básica de la placa que se utiliza para proporcionar alimentación de 5 V a la placa. Puede utilizar cualquier fuente para configurar la alimentación de la placa; sin embargo, es preferible conectar el cable de alimentación a través del puerto USB de la computadora portátil para proporcionar 5 V.

La Raspberry Pi 3 admite dos opciones de conexión, incluida HDMI y compuesta. El conector HDMI se utiliza para conectar LCD o TV, que pueden admitir cables de versión 1.3 y 1.4. La conexión de vídeo compuesto se utiliza para conectar la versión anterior del televisor con el dispositivo que utiliza el conector jack de 3,5 mm para la producción de audio.

El nuevo dispositivo viene con un núcleo de vídeo multimedia con gráficos 3D que es capaz de reproducir vídeo de 1080 MP. Esta característica sitúa este consejo por delante de sus predecesores, donde la calidad del vídeo no se mejoró tanto.

El disco duro USB incorporado en la placa se usa para iniciar el dispositivo, similar al disco duro de la PC donde se usa Windows para iniciar el disco duro de la computadora.

Sistema operativo

El sistema operativo oficial Raspbian se ejecuta en Pi 3 y es una distribución del sistema operativo GNU/Linux basada en Debian. Otros sistemas operativos de terceros que pueden funcionar en este dispositivo son RISC, Kodi Media Center, Windows 10 IoT core, Ubuntu Mate y administración de aulas. Hay otros sistemas no basados en Linux que pueden ejecutarse en este dispositivo y que puede elegir según sus necesidades y requisitos.



Puntos de consideración

Si se requiere permanecer en contacto constante con este dispositivo, es mejor guardarlo en un estuche, esto ayudará a prolongar la vida útil del dispositivo y también lo protegerá del calor y otros factores externos.

Puede ejecutar varios sistemas operativos, incluidos Linux o sistemas no basados en Linux, pero si está muy preocupado por el rendimiento y la estabilidad del dispositivo, ejecutar el sistema operativo oficial Raspbian sería una opción efectiva, que admite una cantidad decente de programación y software ofimática. Raspberry Pi puede admitir tarjetas SD SanDisk de hasta 64

GB. Pero si tiene intención de ejecutar el sistema operativo oficial Raspbian, una tarjeta SD 8GG sería suficiente para satisfacer sus necesidades.

Nota: Se recomienda usar Rasbian ya que es un sistema operativo desarrollado para Raspberry Pi y el cual ejecuta los comando a la perfección.

Características	
Procesador:	Broadcom BCM2837B0, SoC Cortex-A53 de 64 bits a 1,4 GHz
Memoria:	1GB
Conectividad:	• LAN inalámbrica IEEE 802.11b/g/n/ac de 2,4 GHz y 5 GHz, Bluetooth 4.2, BLE. • Gigabit Ethernet sobre USB 2.0 (rendimiento máximo 300Mbps) • 4 × interfaz USB 2.0
Vídeo y sonido:	• 1 x HDMI de tamaño completo • Puerto de visualización MIPI DSI • Puerto de cámara MIPI CSI • Salida estéreo de 4 polos y puerto de vídeo compuesto
Multimedia:	H.264, decodificación MPEG-4 (1080p30); Codificación H.264 (1080p30); Gráficos OpenGL ES 1.1, 2.0
Soporte de tarjeta SD:	Formato Micro SD para cargar sistema operativo y datos almacenamiento.
Alimentación de entrada:	• 5 V/2,5 A CC mediante conector micro USB • 5 VCC a través del encabezado GPIO • Habilitado para alimentación a través de Ethernet (PoE) (requiere PoE separado) (SOMBRERO)

Temperatura de 0-50°C

funcionamiento:

Vida útil de producción: Raspberry Pi 3 Modelo B+ permanecerá en producción hasta al menos enero de 2028

Cumplimiento: Para obtener una lista completa de las aprobaciones de productos locales y regionales, por favor visita pip.raspberrypi.com



III. MATERIALES Y EQUIPO.

1 Raspberry Pi.

1 teclado y mouse

1 Guía de Raspberry Pi

1 pantalla HDMI

IV. PROCEDIMIENTO.

1. Con la información de la guía Raspberry Pi responder las siguientes preguntas:

a. ¿Quién creó el Raspberry Pi?

b. ¿Qué es Raspberry Pi y para qué sirve?

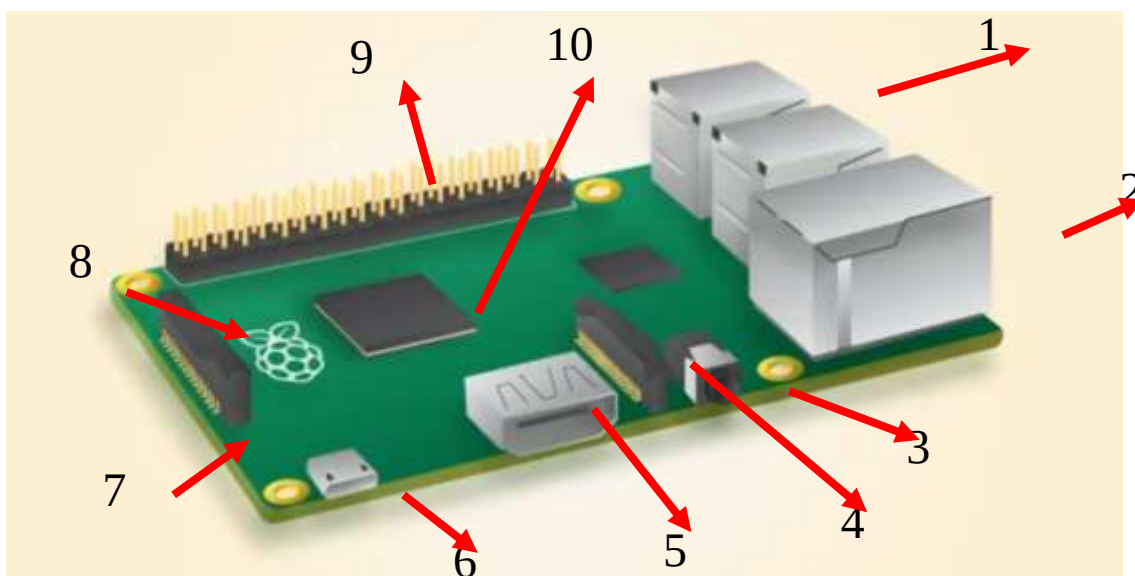
c. ¿Qué significado tiene los pines GPIO y para que se utilizan?

d. Escribir 3 especificaciones Hardware de la Raspberry Pi

e. ¿Qué software usa Raspberry Pi?

f. Por qué se recomienda utilizar el sistema operativo Raspbian.

2. Identificar los componentes que tiene la Raspberry Pi:



1. _____

6. _____

2. _____

7. _____

3. _____

8. _____

4. _____

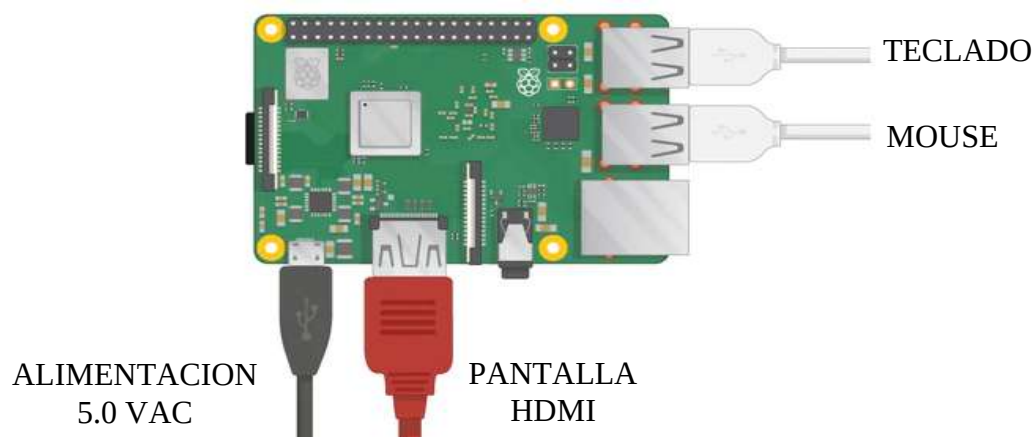
9. _____

5. _____

10. _____

3. Conexión de las Raspberry Pi.

La Raspberry Pi se encuentra conectada de la siguiente manera:



Realizar la conexión eléctrica del módulo, conectando el toma que se encuentra a un costado (figura a), en la parte de enfrente de cada uno de los módulos se encuentran dos pulsadores (figura a.1, uno verde y otro rojo), al presionar el botón verde se energiza el módulo correspondiente, y se indicara con el encendido del piloto del botón



Figura a.

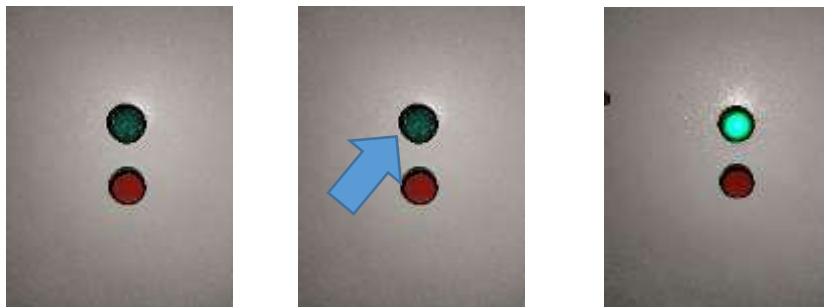
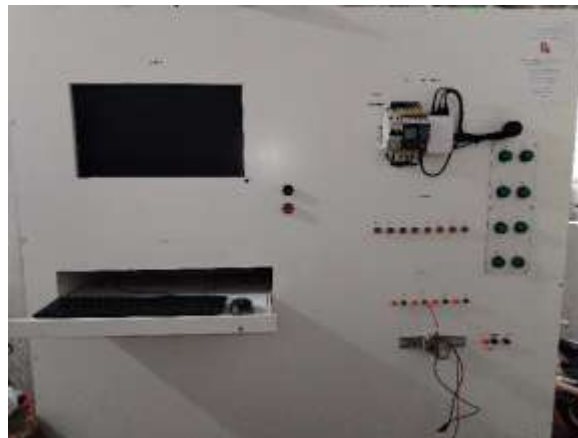


Figura b

Al energizar el módulo se comenzará a cargar el sistema operativo Raspbian en la Raspberry Pi, cuando se termine de cargar, en el monitor se visualizará el entorno grafico del sistema operativo que se muestra a continuación:



4. Explorar el entorno grafico del sistema operativo de Raspberry Pi, y completar con el nombre de cada uno de los iconos que se visualizan en el monitor:



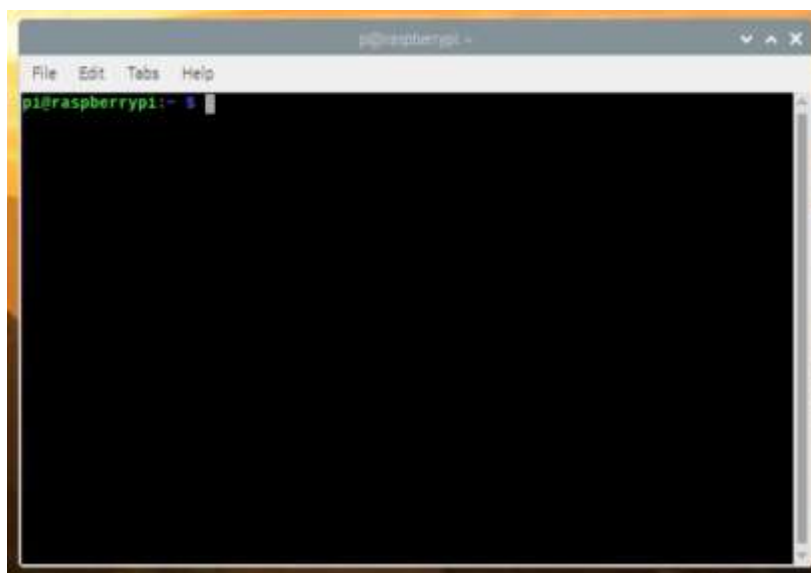






5. Terminal.

Para comenzar en el entorno del sistema operativo se debe inicial con la terminal de la Raspberry Pi e introducir las siguientes instrucciones y buscar su significado de cada una de las instrucciones:



➤ cd: _____

➤ sudo: _____

➤ pwd: _____

➤ df: _____

➤ ls: _____

➤ tree: _____

➤ nano: _____

➤ wget: _____

➤ ssh: _____

➤ fibd: _____

➤ scp: _____

➤ ping: _____

V. PRUEBA DE CONOCIMIENTOS

1. Investigar qué tipos de Raspberry Pi existen y realizar una lista con su descripción.
2. Investigar otras instrucciones que se pueden usar en la terminal de la Raspberry Pi.

VI. CONCLUSIONES DE LA PRÁCTICA

Escribir 3 conclusiones de la práctica realizada.

1. _____

2. _____

3. _____

4.3 GUÍA BÁSICA DE LOGO V8 SIEMENS.



**UNIVERSIDAD TÉCNICA LATINOAMERICANA
FACULTAD DE INGENIERÍA.**

**PRACTICA DE LABORATORIO 2.
TEMA: CONOCIENDO LOGO! V8.
LUGAR A IMPARTIR: LABORATORIO DE ELECTRONICA.**

I. OBJETIVOS.

1. Conocer las principales características de LOGO V8 siemens.
2. Identificar el entorno del programa de logo Soft comfort.

II. INTRODUCCIÓN.

LOGO! es el módulo lógico universal de Siemens.

Es un módulo lógico inteligente destinado a pequeños proyectos de automatización en entornos industriales (control de compresores, cintas transportadoras, control de puertas, etc.), en aplicaciones para espacios de oficina y/o locales comerciales y para uso doméstico (control de la iluminación, tareas de control relacionadas con las piscinas, control de acceso, etc.).

Este mini PLC está implantado en todo el mundo y puede controlarse a distancia. Tiene tareas de automatización muy sencillas de programar mediante un software intuitivo (LOGO! Soft Comfort) de fácil manejo.

En sus inicios LOGO! de Siemens se utilizó para reemplazar los relés cableados; solo permitía la programación de 30 bloques y disponía 8 funciones básicas y 8 funciones adicionales. Pero con el tiempo y, debido a sus nuevas características se fue ampliando su campo de aplicación con cada actualización y sus capacidades de comunicación. Ahora permite programar 400 bloques y tiene más de 40 funciones.

Además, es compatible con el protocolo Ethernet Industrial y dispone de un servidor incorporado de gran utilidad en aplicaciones de IIoT. Gracias a la interfaz Ethernet integrada, puede conectar los módulos lógicos con otros componentes de la serie LOGO! de Siemens y formar redes de comunicación avanzadas.

En su última versión (LOGO! 8.3) permite conexión a la nube procesando grandes volúmenes de datos, sin la dependencia de la tarjeta SD integrada en anteriores versiones de LOGO! que se van a poder almacenar en la nube.

LOGO! incluye un controlador con unidad de manejo y visualización. Con esa unidad se pueden crearse y editarse programas, así como manejarse funciones del sistema. En los módulos lógicos LOGO! ya están integradas las funciones básicas habituales en la práctica listas para su uso, dependiendo del tipo de dispositivo.

LOGO! Permite realizar tareas:

- ✓ En las instalaciones en viviendas y edificios
- ✓ En la construcción de armarios eléctricos y el montaje de máquinas y aparatos.

Además, LOGO! se puede utilizar para controladores especiales para el procesamiento de señales. Gracias a la conexión a AS-Interface, es posible el uso como periferia descentralizada con inteligencia local propia para el control de máquinas y procesos. De este modo se pueden ejecutar tareas de control en el módulo lógico LOGO! y descargar así al controlador maestro.

Para las aplicaciones en serie en la construcción de máquinas pequeñas, aparatos y armarios eléctricos, así como en el sector de instalaciones, existen variantes especiales sin unidad de mando. Estas deben cargarse a continuación mediante un módulo de programa o mediante el software para PC LOGO!Soft Comfort.

En la figura siguiente se muestra el LOGO!8 y la ubicación de cada una de sus partes:



Programar LOGO!

Por programar se entiende la creación de un programa. Básicamente, un programa de LOGO! no es más que un esquema eléctrico representado de una forma diferente. Hemos adaptado la representación al display de LOGO! para que se puede convertir sus aplicaciones en programas de LOGO!.

Para poder convertir un diagrama eléctrico en su representación de LOGO existe una herramienta que permite hacerlo es el software de programación LOGO!Soft Comfort, que permite crear, probar, simular, modificar, guardar e imprimir los programas cómodamente. Debemos comprender dos conceptos fundamentales que son **bornes** y **bloque** que se relaciona con lo mismo.

Bornes

Como bornes identificamos todas las conexiones y estados que se pueden utilizar en LOGO!. Las entradas y salidas pueden tener el estado '0' o el estado '1'. El estado '0' significa que no hay tensión en la entrada. El estado '1' significa que sí hay tensión. Los bornes 'hi', 'lo' y 'x' los hemos introducido para facilitarle la elaboración del programa:

- 'Hi' (high) tiene asignado el estado fijo '1',
- 'Lo' (low) tiene asignado el estado fijo '0'.

No debe utilizar todas las conexiones de un bloque. Para las conexiones que no se utilizan, el programa adopta automáticamente el estado que garantiza el funcionamiento del bloque en cuestión. Dado el caso, es posible identificar las conexiones no utilizadas de forma especial con el borne 'x'.

Entradas Digitales

Las entradas digitales empiezan por la letra **I**. El número de entradas digitales (I1, I2,...) se corresponde con el número de conectores de entrada del módulo base LOGO! y de los módulos digitales conectados en el orden de montaje. Las entradas digitales rápidas I3, I4, I5 e I6 de algunos modelos de LOGO! que tenga las versiones LOGO! 24/12 RCE, ¡LOGO! 24/12 RCEO, LOGO! 24 CE y LOGO! 24 CEo como contadores rápidos.

Entradas Analógicas

Las versiones LOGO! 24 CE, ¡LOGO! 24 CEO, LOGO! 24/12 RCE y LOGO! 12/24 RCEO dispone de las entradas I1, I2, I7 e I8, que también pueden programarse para ser utilizadas como entradas AI3, AI4, AI1 y AI2. Ajustar el número de entradas analógicas del LOGO!, estos módulos pueden configurarse de manera que utilicen dos entradas analógicas (AI1 y AI2), o todas las entradas (cuatro). ¡LOGO! interpreta las señales de las entradas I1, I2, I7 e I8 como valores digitales, mientras que las de las entradas AI3, AI4, AI1 y AI2 se interpretan como valores analógicos.

Salidas Digitales

Las salidas digitales empiezan por la letra **Q**. Los números de las salidas (Q1, Q2,... Q20) se corresponden con los números de los conectores de salida del módulo base LOGO! y de los módulos de ampliación conectados en el orden de montaje.

¡LOGO! 0BA8 también ofrece 64 salidas libres, identificadas con la letra x. No es posible reutilizar las salidas libres en un programa. En eso, las salidas libres difieren de las marcas, que sí pueden reutilizarse. Una salida libre resulta útil p.ej. para la función especial "Textos de mensajes" si solo el texto de mensaje es relevante en el programa.

Salidas Analógicas

Las salidas analógicas empiezan por las letras AQ. Hay ocho salidas analógicas disponibles: AQ1, AQ2 a AQ8. Una salida analógica solo puede conectarse con la entrada analógica de una función, una marca analógica (AM) o un conector de salida analógica.

.

Bloques y su numeración

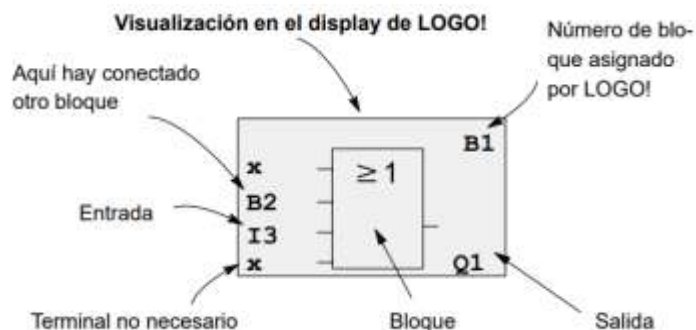
Un bloque es una función que convierte información de entrada en información de salida. Antes era necesario cablear los distintos elementos en el armario eléctrico o en la caja de conexiones. Al elaborar el programa debe conectar bornes con bloques. A tal efecto, basta con elegir la conexión deseada en el menú Co. El menú Co debe su nombre al término inglés

- **“Connector” (borne).**

Representación de un bloque en el display.

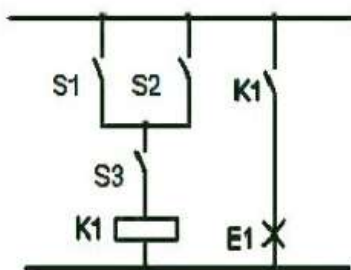
En la figura se muestra un display de logo, no es posible mostrar más de un bloque al mismo tiempo por tal razón se ha utilizado la numeración de los bloques para interpretar bien el programa final

La figura siguiente muestra una configuración de ejemplo de LOGO! y la numeración de las entradas y salidas del programa



Esquema de circuito a LOGO!

La forma de representar los circuitos de un esquema puede ser como la siguiente:



El indicador luminoso E1 se activa y desactiva a través de los interruptores (S1 o S2) y S3 (O = OR; Y=AND)

Se activa el rele K1 al cerrarse S1 o S2 y además S3.

Realización del circuito mediante LOGO!

En LOGO se creará un circuito en el que se conectarán los bloques y bornes entre ellos:



LOGO! Soft Comfort V8.0

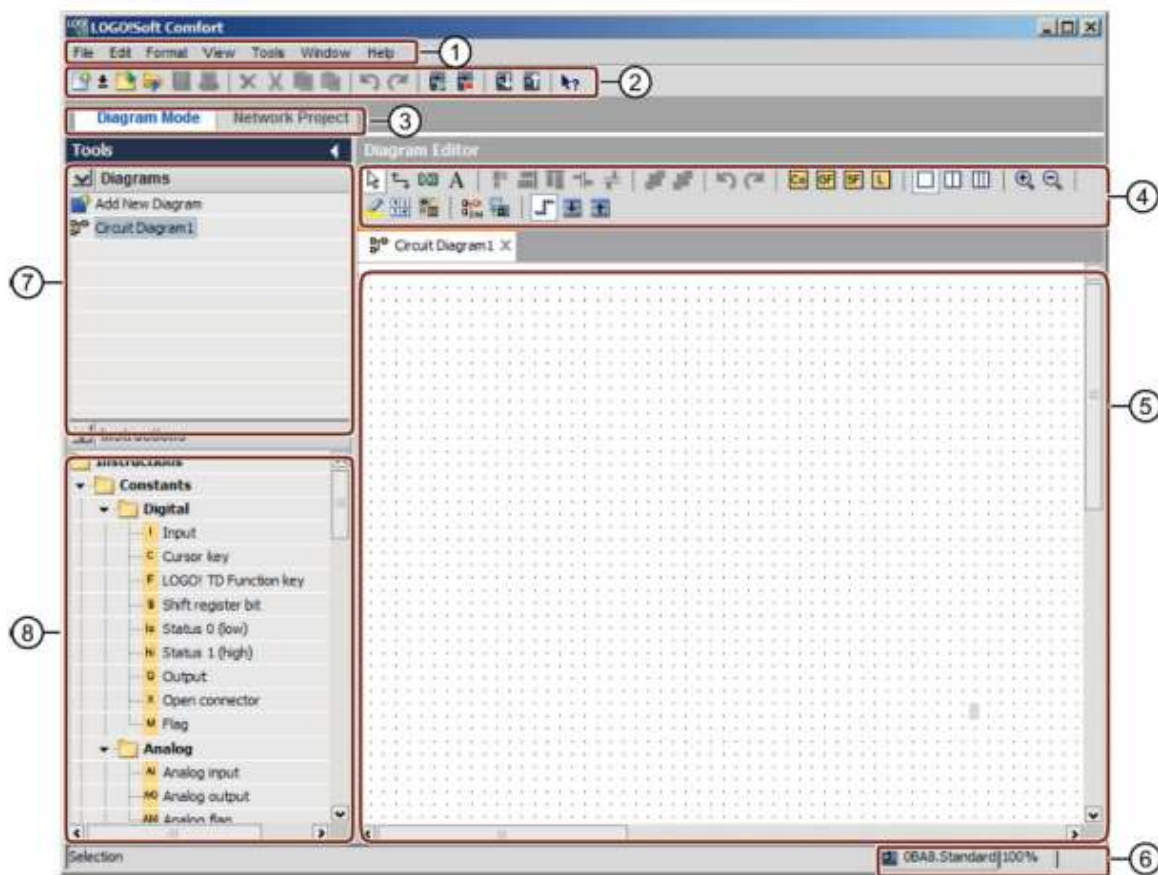
Este software constituye una interfaz de usuario completamente nueva, que ofrece las siguientes funciones:

- ✓ Visualización coherente de los menús de aplicaciones
- ✓ Nuevo sistema de trabajo basado en proyectos de red

- ✓ Visualización dividida para el modo de diagrama y el modo de red
- ✓ Visualización dividida para la barra de herramientas "Estándar" en la interfaz general del software; en el modo de diagrama, aparece la barra de herramientas "Herramientas", y en el modo de proyecto, la barra de herramientas "Red"
- ✓ Visualización en ventanas divididas con cambio de foco y función Arrastrar y soltar
- ✓ Al trabajar en un proyecto de red se puede guardar, cargar, crear y cerrar dicho proyecto
- ✓ Nuevos ajustes para el control de acceso online, con distintas modalidades de acceso
- ✓ Posibilidad de crear conexiones mediante la configuración de bloques de función NI y NQ
- ✓ Nueva referencia gráfica para el bloque de función en el campo de parámetros de esquemas de conexiones FBD
- ✓ Posibilidad de configurar la indicación en pantalla para avisos, pantalla inicial y marcas con 4 líneas para dispositivos LOGO! anteriores a 0BA8 y 6 líneas para dispositivos LOGO! a partir de 0BA8
- ✓ Mayor seguridad del sistema gracias a la creación de contraseñas de usuario y niveles de acceso mediante la configuración del control de acceso

Interfaz de programación

El modo de programación en LOGO!Soft Comfort se inicia con un diagrama vacío. La mayor parte de la pantalla la ocupa entonces el área dedicada a la creación de esquemas de conexiones. Esta área se denomina interfaz de programación. En esta interfaz de programación se disponen los símbolos y enlaces del programa. Para no perder la vista de conjunto, especialmente en el caso de programas grandes, en los extremos inferior y derecho de la interfaz de programación se dispone de barras de desplazamiento que permiten mover el programa en sentido horizontal y vertical.



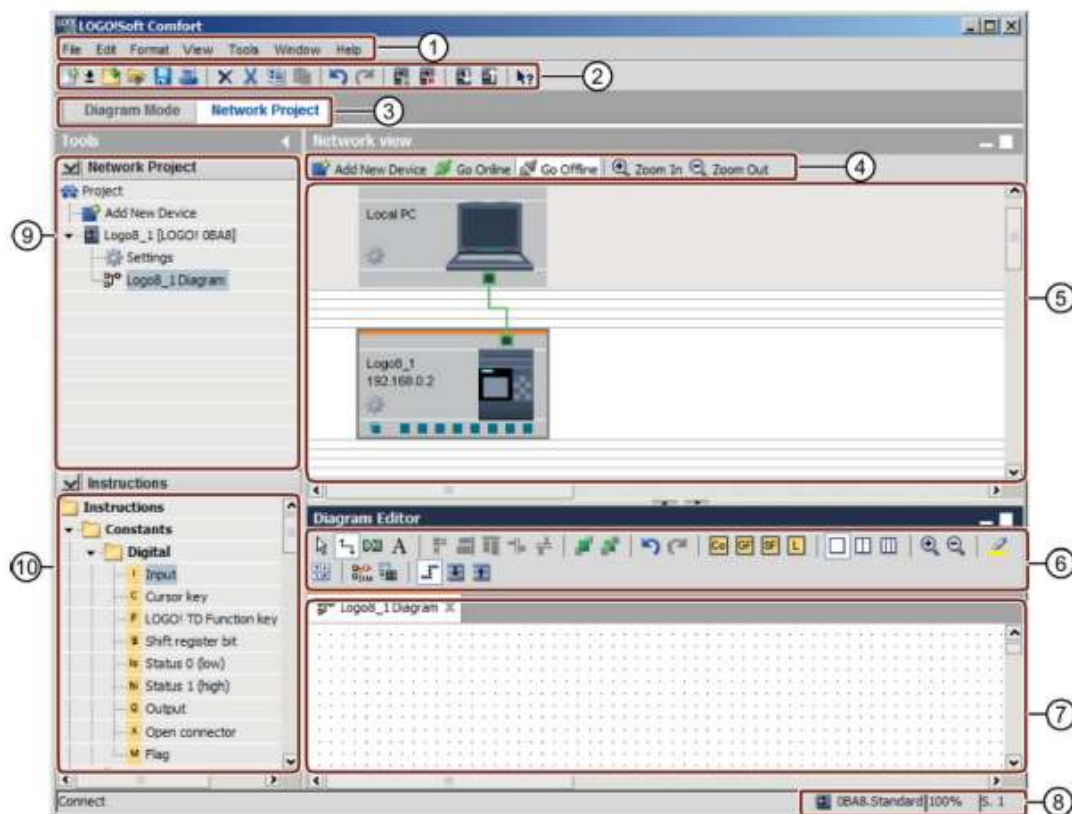
- | | |
|---|----------------------------|
| ① Barra de menús | ⑤ Interfaz de programación |
| ② Barra de herramientas "Estándar" | ⑥ Barra de estado |
| ③ Barra de modo | ⑦ Árbol de diagramas |
| ④ Barra de herramientas
"Herramientas" | ⑧ Árbol de operaciones |

Interfaz de proyecto

En la interfaz de proyecto, LOGO! Soft Comfort muestra una vista de red con los dispositivos y sus conexiones de red. Después de seleccionar "Agregar dispositivo", aparece la ventana del editor de diagramas.

Ahora, en un proyecto de red solo pueden programarse dispositivos LOGO! a partir de 0BA7. El editor de diagramas muestra los bloques de programa y los vínculos del programa. En principio, el programa está vacío.

Para no perder la vista de conjunto, especialmente en el caso de proyectos y programas grandes, en los extremos inferior y derecho de la vista de red interfaz de programación se dispone de barras de desplazamiento que permiten mover el programa en sentido horizontal y vertical.



- | | |
|------------------------------------|--|
| ① Barra de menús | ⑥ Barra de herramientas "Herramientas" |
| ② Barra de herramientas "Estándar" | ⑦ Interfaz de programación |
| ③ Barra de modo | ⑧ Barra de estado |
| ④ Barra de herramientas "Red" | ⑨ Árbol de diagramas |
| ⑤ Vista de red | ⑩ Árbol de operaciones |

III. MATERIALES Y EQUIPO.

1 LOGO V8.

1 computadora.

8 interruptor o pulsador.

1 cable de red.

4 piloto luminoso.

1 manual LOGO V8

IV. PROCEDIMIENTO.

1. Con la información de la guía responder las siguientes preguntas:

a) ¿Qué es LOGO!8?

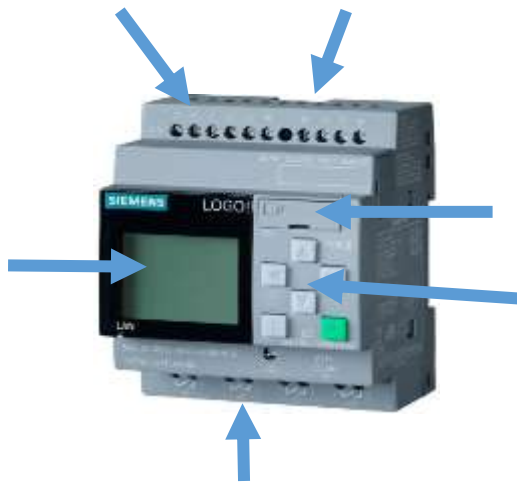
b) ¿Cuál era la capacidad de los primeros LOGO!8?

c) Mencionar 3 características del LOGO!8?

d) ¿Cuál es el nombre del software que se utiliza para programar el LOGO!8?

e) Mencionar 4 funciones del software

2. Colocar el nombre de cada parte que está formado el LOGO!



3. Realizar la conexión eléctrica del módulo, conectando el toma que se encuentra a un costado (figura a), en la parte de enfrente de cada uno de los módulos se encuentran dos pulsadores (figura b, uno verde y otro rojo), al presionar el botón verde se energiza el módulo correspondiente, y se indicara con el encendido del piloto del botón



Figura a.

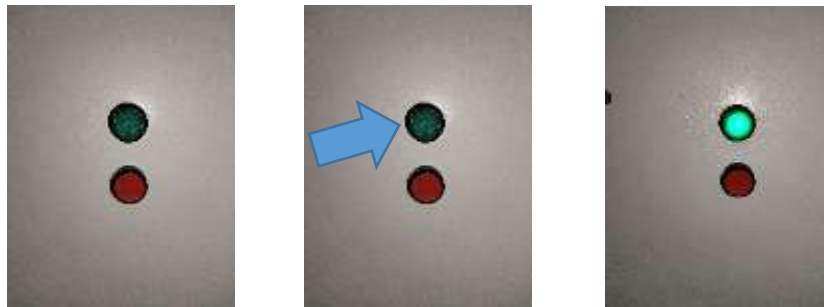
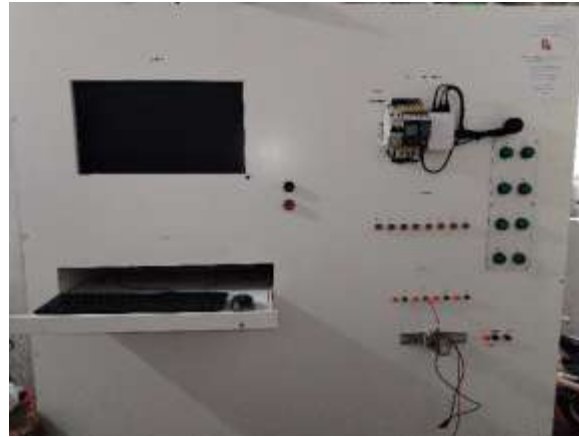


Figura b

4. Verificar que el autómeta que se encuentra a la par del logo se encuentra activo, en caso contrario habilitar el autómeta para encender el LOGO! ver figura c.

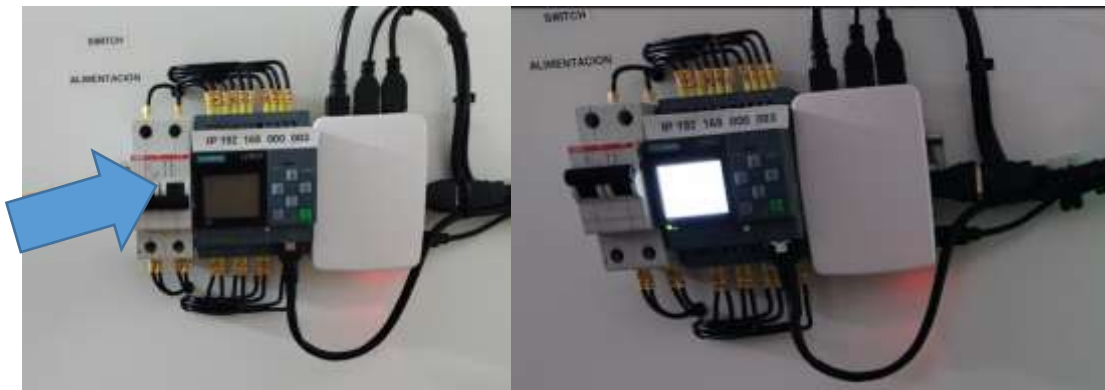


Figura c

5. En el interior del módulo se encuentra el switch, hay que conectar un cable de Ethernet, para poder utilizar el software que deberían de tener instalado en las computadoras.
Ver figura d

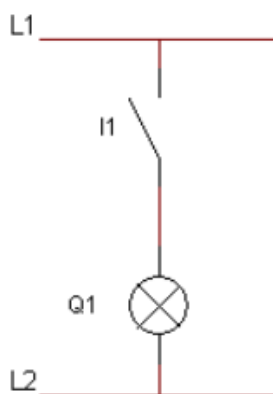


Figura d

6. Para elaborar un programa en el logo, se realiza por medio de bloques, a continuación, se presentan algunos bloques de funciones, escribir el nombre de cada función presentada:

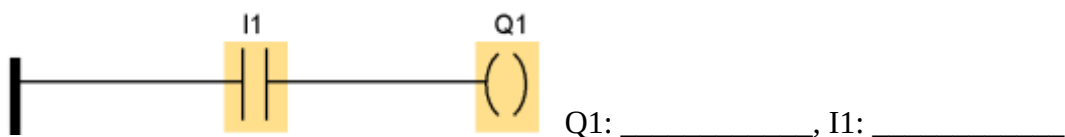


7. Programas de PLC en escalera, En el siguiente circuito de mando, ¿Cuánto vale el estado de la lámpara (Q1)? _____

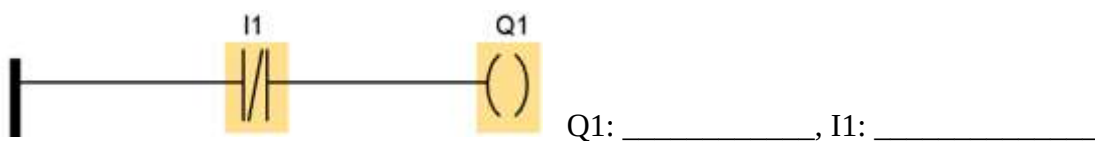


Si la respuesta es apagada o estado cero, se está pensando solo en términos de un circuito eléctrico. En términos de PLC para dar respuesta hay que ver el programa, para determinar cuál es el estado de cada uno de los dispositivos conectados, en programación se pueden tener dos opciones. ¿Cómo se encuentra el estado de salida Q1 y I1 en los dos programas?

Programa A:



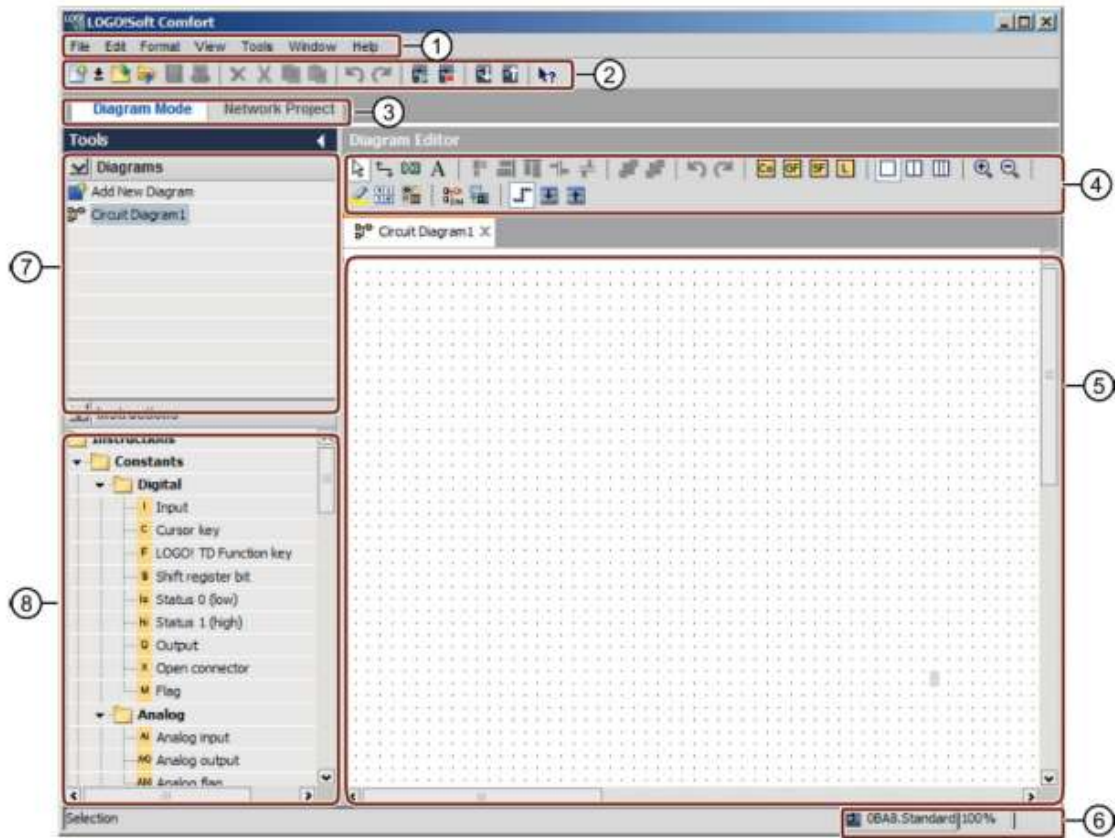
Programa B:



Como se dominan estos tipos de contactos y cuál es su función:



8. Interface del programa en logo Softcomfort, identificar las siguientes partes del programa.



- 1) _____; 2) _____; 3) _____;
 4) _____; 5) _____; 6) _____;
 7) _____; 8) _____

9. Instrucciones de uso LOGO!Soft Comfort y Funciones lógicas

Para ilustrar el uso del software se hará el siguiente programa de ejemplo:

Realizar un programa para un sistema como el que se muestra en la **Figura e** donde se necesita almacenar agua en el tanque entre el nivel inferior y el superior para que la bomba no trabaje continuamente enviando agua a la casa.

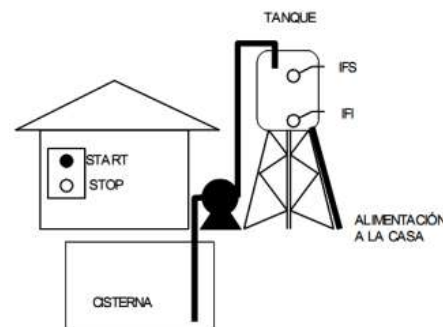


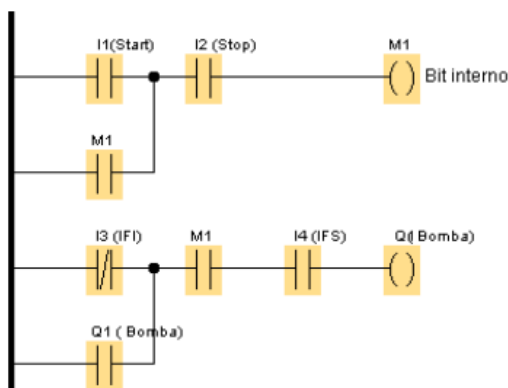
Figura e

START y STOP son pulsadores normalmente abierto y cerrado respectivamente, el interruptor flotador inferior IFI está abierto cuando está desactivado y cerrado cuando se activa; mientras que el interruptor flotador superior IFS está cerrado si está desactivado y abierto si se activa.

A continuación, se muestran la tabla de elementos de entrada y salida que necesitará conectar al módulo lógico LOGO! para llevar a cabo el programa antes descrito. (En el módulo el LOGO! ya se encuentran conectado las entradas y las salida)

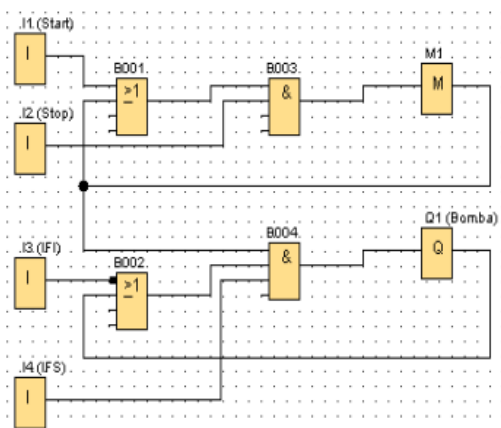
Nº	Denominación	Nombre	Modo de acción	Dirección I/O
1	Arranque de sistema automático de llenado	Start	Cierra = 1	I1
2	Paro del sistema automático de llenado	Stop	Abre = 1	I2
3	Interruptor Flotador Inferior	IFI	Cierra = 1	I3
4	Interruptor Flotador Superior	IFS	Abre = 1	I4
5	Salida hacia bomba de llenado	Bomba		Q1

El esquema de contactos o lenguaje de escalera de este problema es el siguiente:



El LOGO! por defecto se programa en lenguaje de diagramas de bloques de función, pues es la única forma en que puede ser programado directamente con las teclas del panel de mando, sin embargo el LOGO! SoftComfort ofrece la posibilidad de poder hacer el programa con esquemas de contactos cambiando las opciones del menú herramientas, aunque al visualizarlo en la pantalla integrada en el LOGO! se observe siempre como bloques de función.

Por ello el programa se realizará utilizando el lenguaje por defecto, en la siguiente imagen se encuentra el diagrama de funciones lógicas del LOGO!.



La marca M1 es un almacenamiento de resultados intermedios, ya que la recursión (o realimentación) sólo se permite vía salidas (Q) y marcas (M).

10. En esta guía se realizará las indicaciones para poder introducir el circuito lógico, siga los pasos que se muestran a continuación para simular y luego descargar la solución al módulo

lógico LOGO!.

- a) Ejecutar el programa LOGO!Soft Comfort V8.3 que se encuentra en Inicio, Todos los programas, Siemens LOGO!Soft.
- b) Dar clic en el botón Nuevo para generar un nuevo archivo.

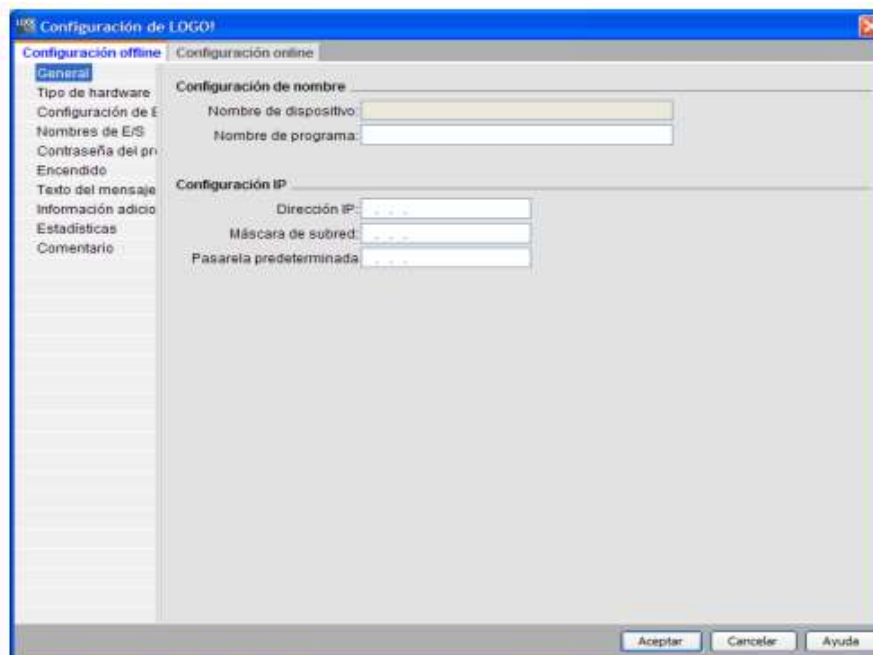


- c) Aparecerá la ventana de Configuración de LOGO! (sino de clic en el menú Archivo y luego en Propiedades), en la primera pestaña Configuración Offline, seleccionar la opción General y colocar un nombre al programa e ingresar la dirección IP, mascara de subred y pasarela predeterminada del LOGO! al que se va a conectar. Que en este caso serán las siguientes:

Dirección IP: 192.168.0.5

Mascara de subred: 255.255.255.0

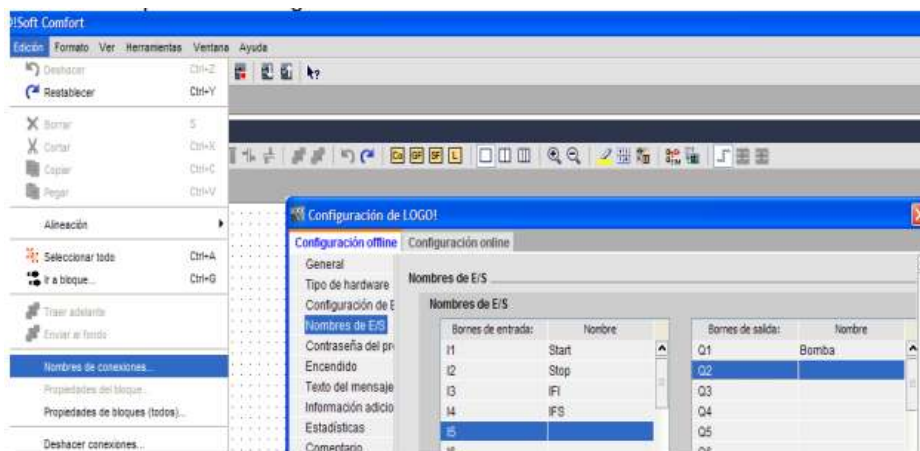
Pasarela predeterminada: 192.168.0.1



- d) Ubicar la opción de Tipo de Hardware y asegúrese de que esté seleccionado como tipo de hardware el LOGO! OBA7.ES4



- e) En la opción de Información adicional puede ingresar datos como: Autor, Nombre del proyecto, Nombre de la instalación, etc. Existen más opciones a configurar si se requiere, pero por el momento deje las demás como están y presione el botón Aceptar.
- f) Lo primero a realizar será poner los nombres de los elementos que se conectarán al LOGO! para esto selecciona la opción Nombres de conexiones que está en el menú Edición. Y colocar los nombres simbólicos de la columna “nombre” de la Tabla presentada anteriormente. Deberá quedar de la siguiente manera:

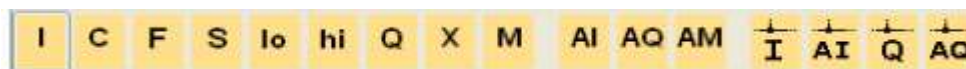


- g) Presionar “Aceptar”, a continuación, se va insertar la primera entrada I1, para ello dar clic en el icono Co (Constantes y conectores F6) del Editor de diagramas.

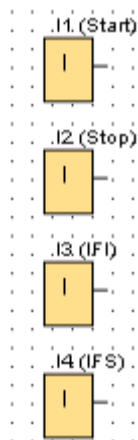


1. Selección
2. Conectar F5
3. Deshacer/Unir conexión F11
4. Insertar comentario F9
5. Constantes y conectores F6
6. Funciones básicas F7
7. Funciones especiales F8
8. Simulación F3
9. Test online

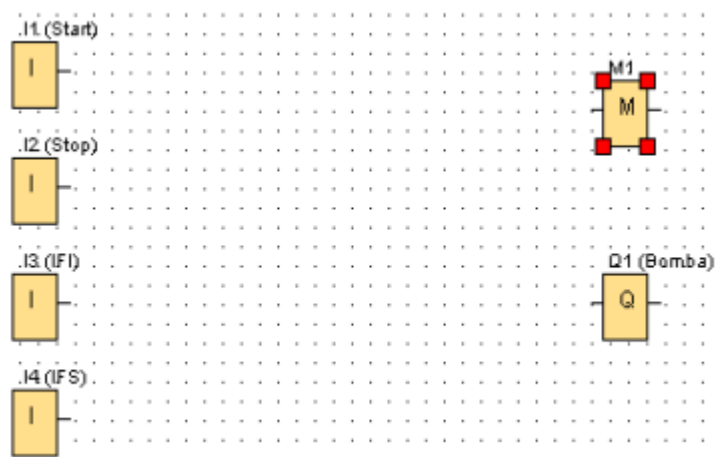
- h) Notara que en la parte inferior de la ventana de programación se presentan las opciones que se pueden seleccionar.



- i) Dar clic sobre la opción I (entrada) y luego dar clic en donde quiera colocar ese bloque en la ventana de trabajo.
- j) Mientras no desactive la opción podrá seguir insertando más entradas, insertar las demás entradas. El resultado a obtener será el siguiente:



- k) De la misma manera que con las entradas se pueden insertar la salida y la marca.
- De las opciones de Co seleccione Q y luego M. el resultado hasta ahora será como el siguiente:

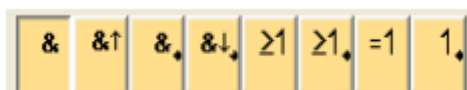


- l) Ahora procederá a colocar las dos compuertas OR y las dos AND, para ello dar clic en el icono de las funciones básicas GF del editor de diagrama.

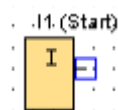


1. Selección
2. Conectar F5
3. Deshacer/Unir conexión F11
4. Insertar comentario F9
5. Constantes y conectores F6
6. Funciones básicas F7
7. Funciones especiales F8
8. Simulación F3
9. Test online

m) Aparecerán las opciones mostradas a continuación, estas funciones básicas representan en ese orden a AND, AND con detección de flanco, NAND, NADN con detección de flanco, OR, NOR, EXOR y not.



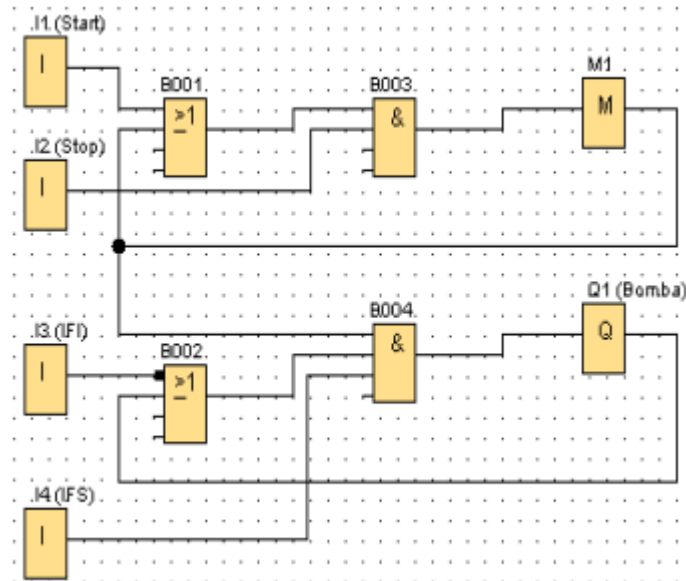
n) Ahora para realizar la conexión de los elementos que están en el espacio de trabajo del Editor de diagramas seleccionar  conectar (F5), al activarlo el cursor servirá únicamente para realizar alambrados en los bloques que están en el espacio de trabajo. Al acercar el cursor del mouse a un terminal de un componente como la I1 el terminal presenta un rectángulo azul, esto indica que puede conectar ese terminal a otro, dar clic izquierdo en este terminal de origen y arrastrar (sin soltar el botón del mouse) hasta la primera entrada del bloque OR superior donde aparecerá nuevamente el rectángulo azul seguido por un número 1, soltar el botón del mouse para finalizar la conexión.



o) Antes de hacer todas las demás conexiones por su cuenta observe el circuito lógico de funciones, el bloque OR inferior, verá que tiene un pequeño cuadrado negro en su entrada superior, esto indica que es una entrada negada, para negar esta entrada dar clic derecho sobre ella y del menú contextual que aparece seleccionar la opción **Negar Conector**.

p) Realizar todas las conexiones, teniendo presente que si requiere el movimiento de algún bloque o conexión podrá usar la  herramienta selección del Editor

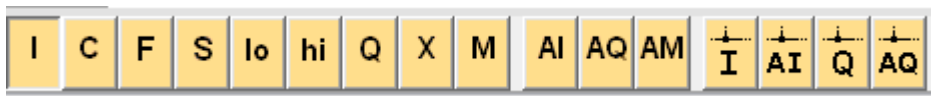
de diagramas. El circuito final deberá verse como el siguiente.



V. PRUEBA DE CONOCIMIENTOS

Seleccionar la respuesta correcta.

- ¿Cuántas entradas y salida tienen el LOGO!?
 - E:16 y S:4
 - E:8 y S:4
 - E:4 y S:2
 - E:4 y S:8
- Para realizar un programa en LOGO! se hace por medio de
 - Bornes
 - Contactos
 - Bloques
 - Comandos
- En qué área del software se crean los programas del LOGO!
 - Barra Menú
 - Barra Modo
 - Árbol de diagrama
 - interfaces de programa
- A que selección de herramientas de programación pertenece el siguiente menú:



- Funciones Básicas
- funciones especiales
- Constante/Conectores

5. llenar el siguiente cuadro con el nombre de la función y para qué sirve:

SIMBOLO	NOMBRE	FUNCION
		
		
		
		
		
		
		
		
		
		
		
		

VI. CONCLUSIONES DE LA PRÁCTICA

Escribir 2 conclusiones de la práctica realizada.

a) _____

b) _____

4.4 GUÍA INICIANDO CON MODBUS.



**UNIVERSIDAD TECNICA LATINOAMERICANA
FACULTAD DE INGENIERIA.**

**PRACTICA DE LABORATORIO 3.
TEMA: ESTADO DE LAS ENTRADAS Y SALIDAS CON
RASPBERRY PI.**

I. OBJETIVOS.

1. Comprender el funcionamiento de la aplicación SCADA para la comunicación por medio de Modbus TCP.
2. Conocer el entorno de programación Python, e integrando la Raspberry Pi con el LOGO! para la visualización del estado de entradas como las salidas.

II. INTRODUCCIÓN.

Se usara la aplicación que se ha realizado para la comunicación entre la Raspberry Pi con el LOGO! utilizando el SCADA que se ha creado para verificar las entradas y salida del LOGO!.

III. MATERIALES Y EQUIPO.

- 1 LOGO V8.
- 8 interruptor o pulsador.
- 4 piloto luminoso.
- 1 computadora.
- 1 cable de red.
- 1 manual LOGO V8

IV. PROCEDIMIENTO.

1. Realizar la conexión eléctrica del módulo, conectando el toma que se encuentra a un costado (figura a), en la parte de enfrente de cada uno de los módulos se encuentran dos pulsadores (figura b, uno verde y otro rojo), al presionar el botón verde se energiza el módulo correspondiente, y se indicara con el encendido del piloto del botón



Figura a.

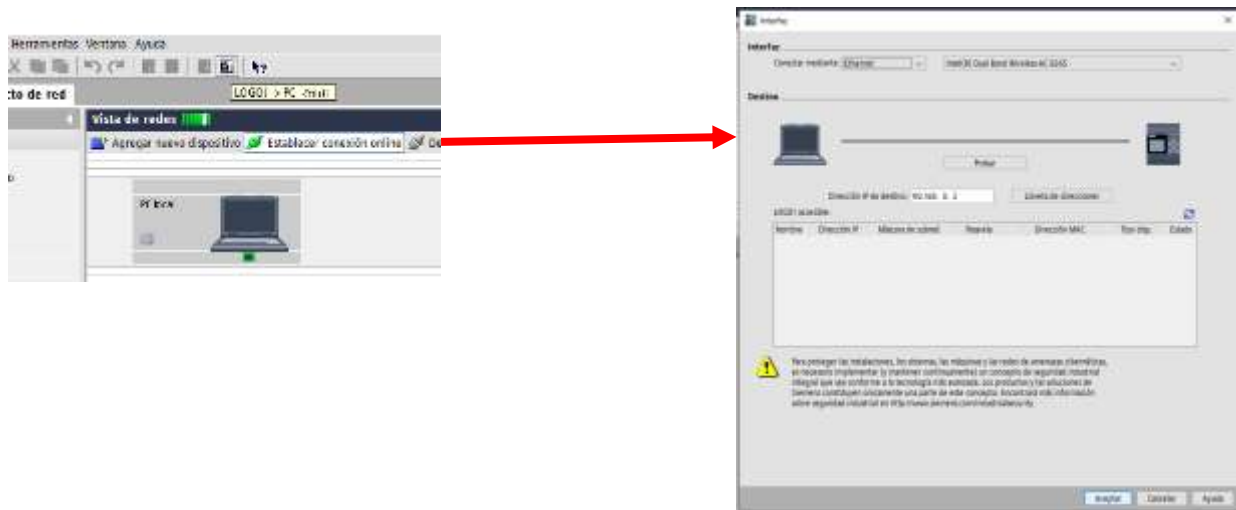


Figura b

2. Abrir LOGO SoftConfort, y abrir un proyecto de red nuevo.



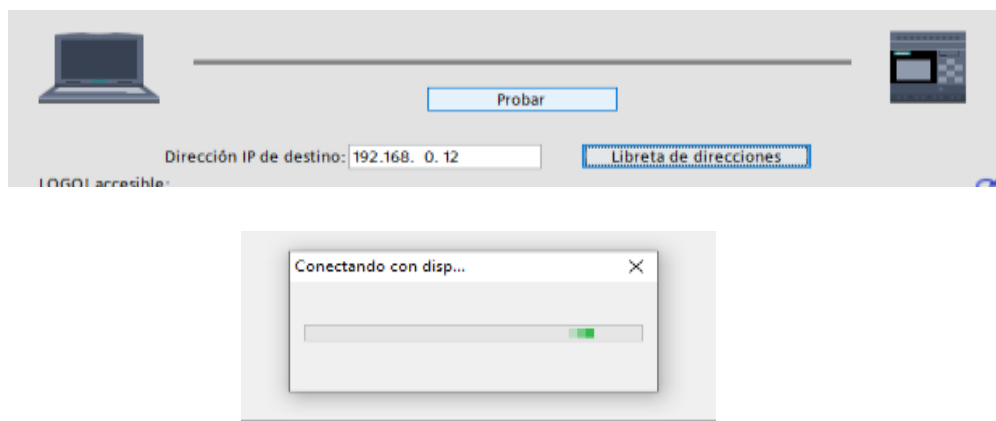
3. Establecer conexión online y conectar con la IP del logo

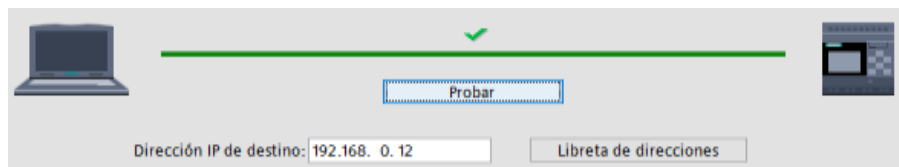


4. Colocar dirección IP de destino del logo, buscar en el logo la IP configurada.

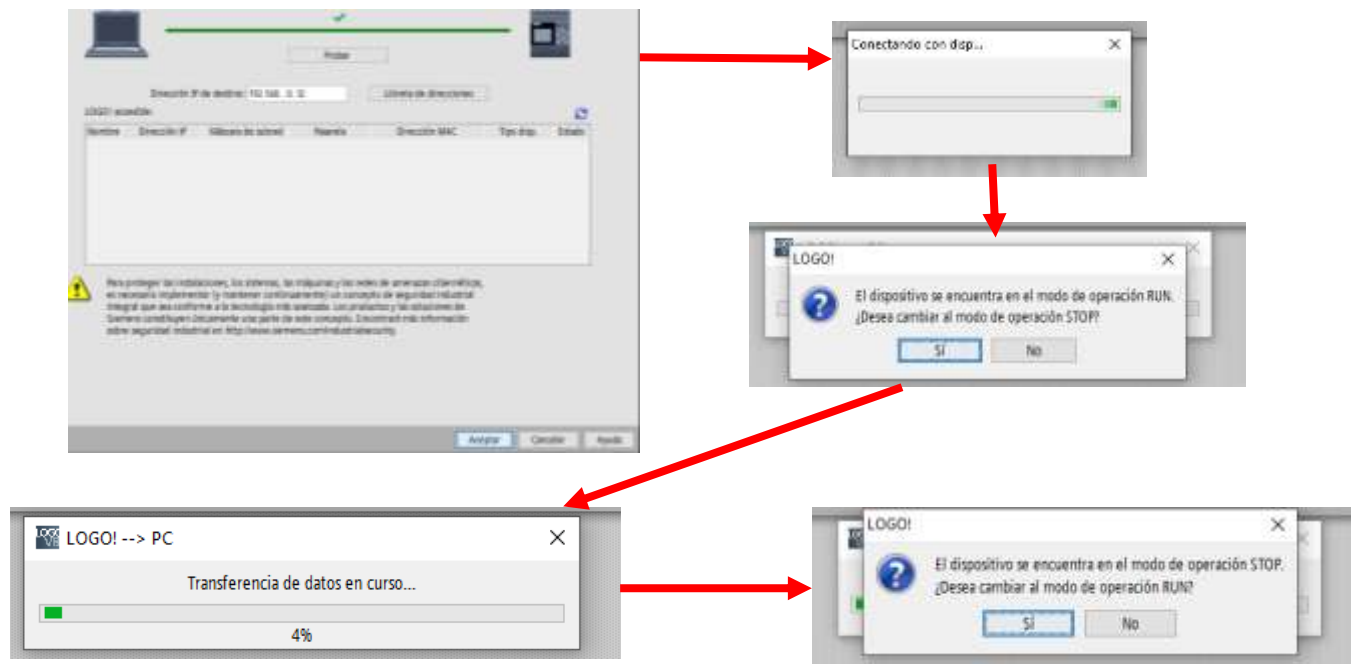


5. Cuando se conoce la IP se debe probar la conexión entre la laptop y el LOGO! dar click en botón probar.



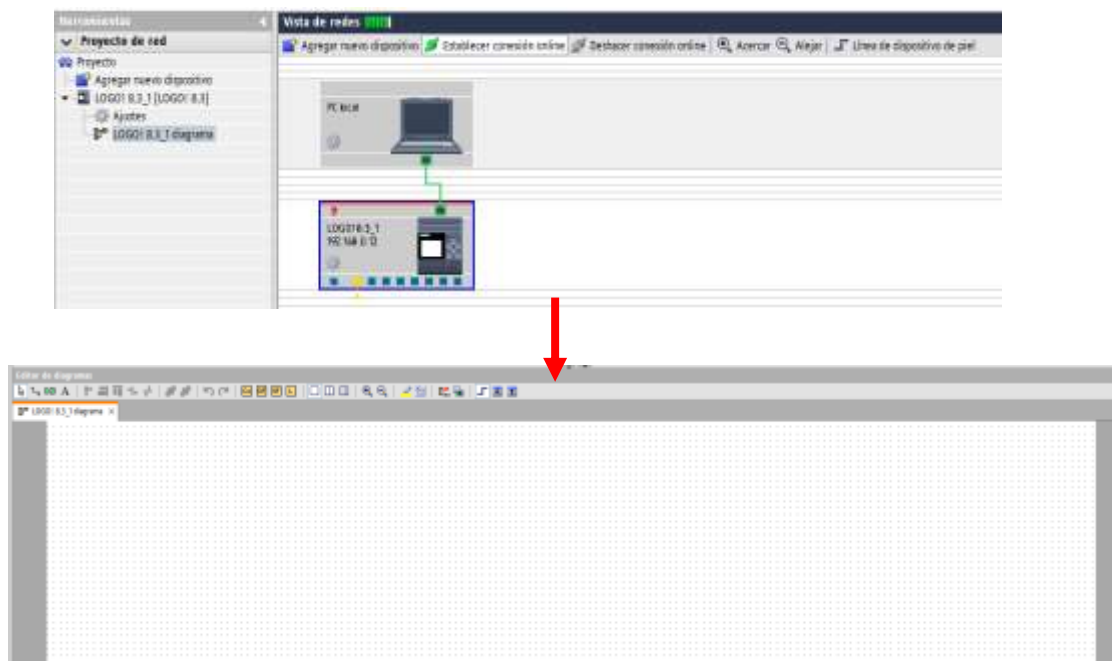


6. Click en aceptar, se cargará la conexión y la transferencia se completará.

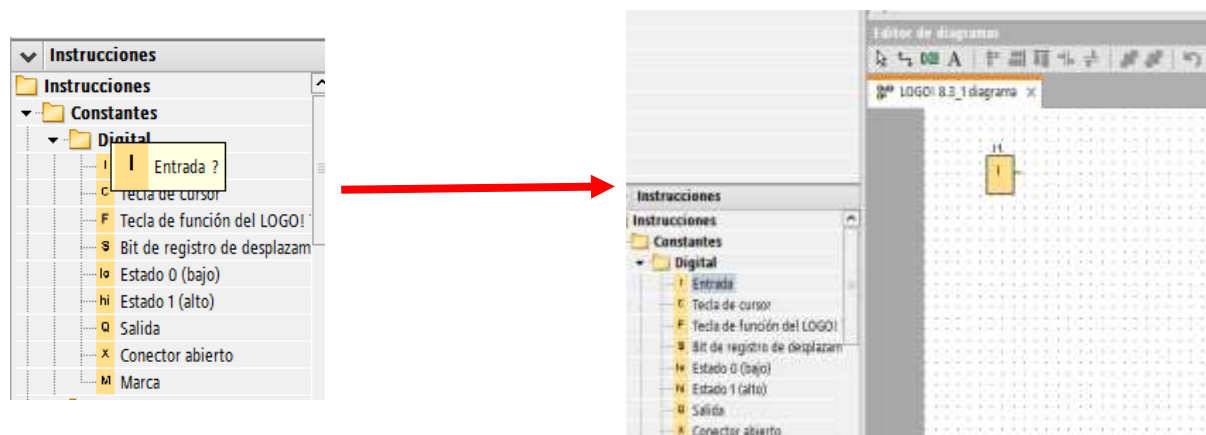


7. Cuando se establece la conexión se visualizará la siguiente ventana con la laptop y el logo

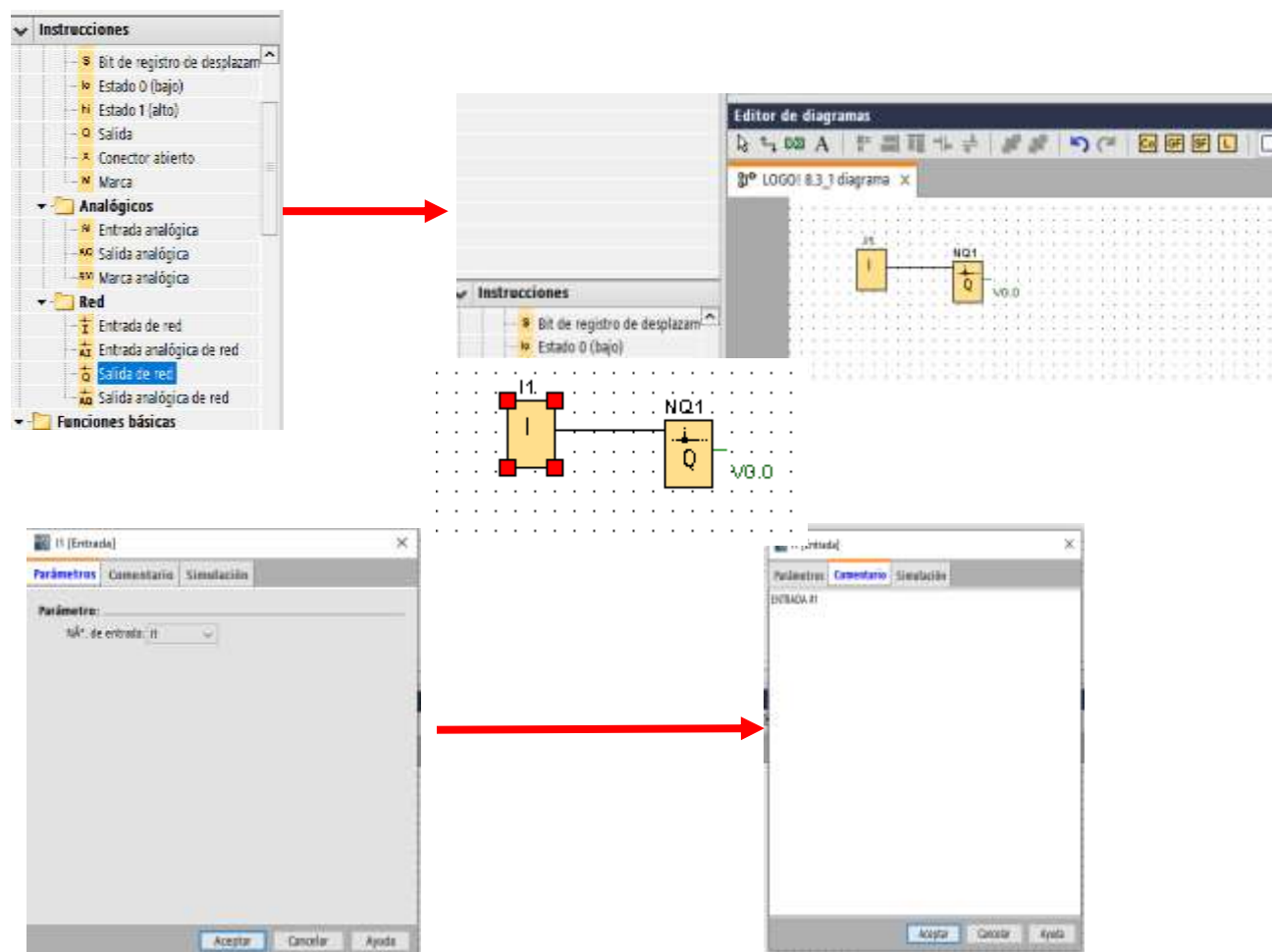
indicando la IP y el entorno de diagrama.

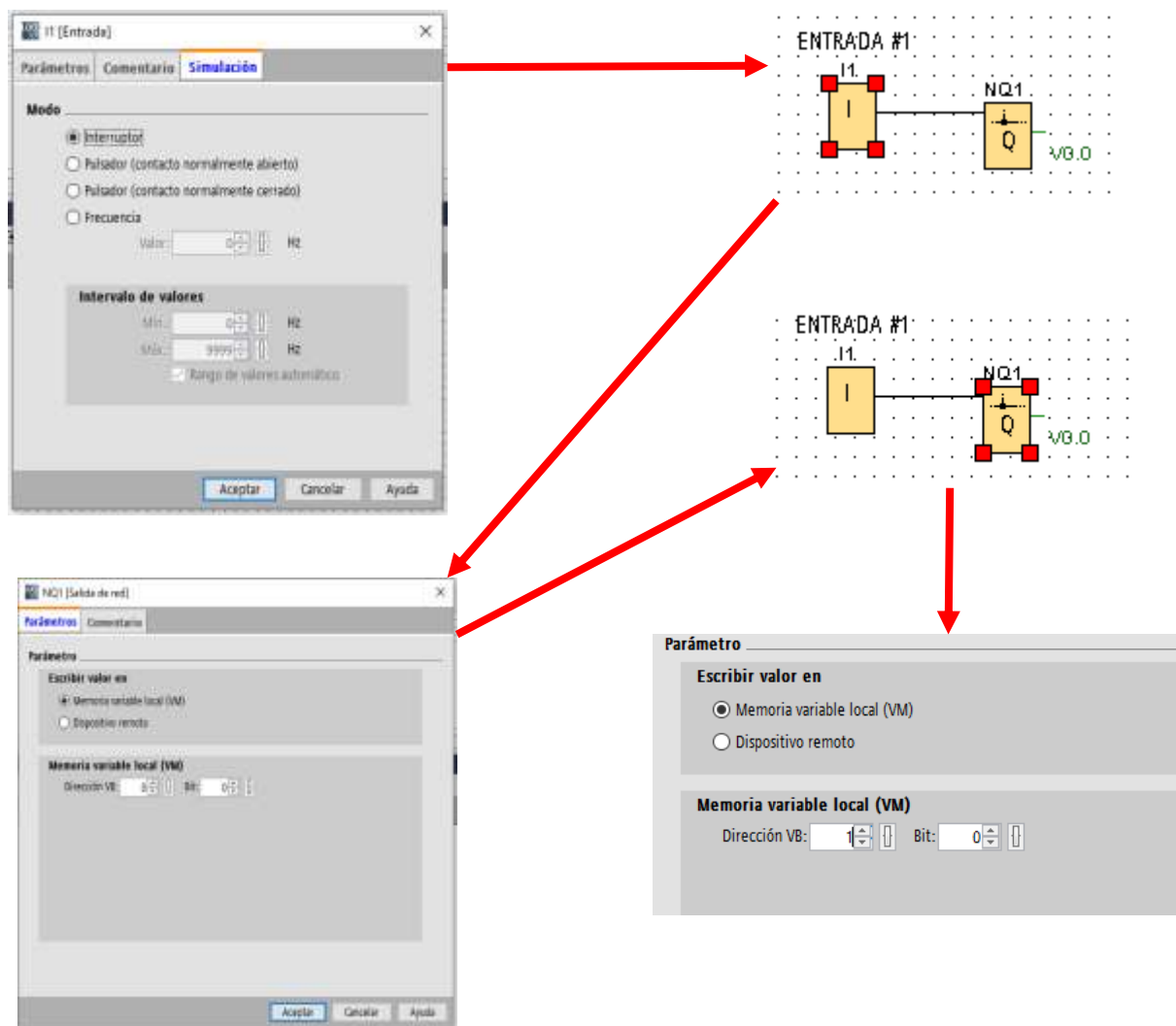


8. Insertar un bloque de constantes digitales una entrada.

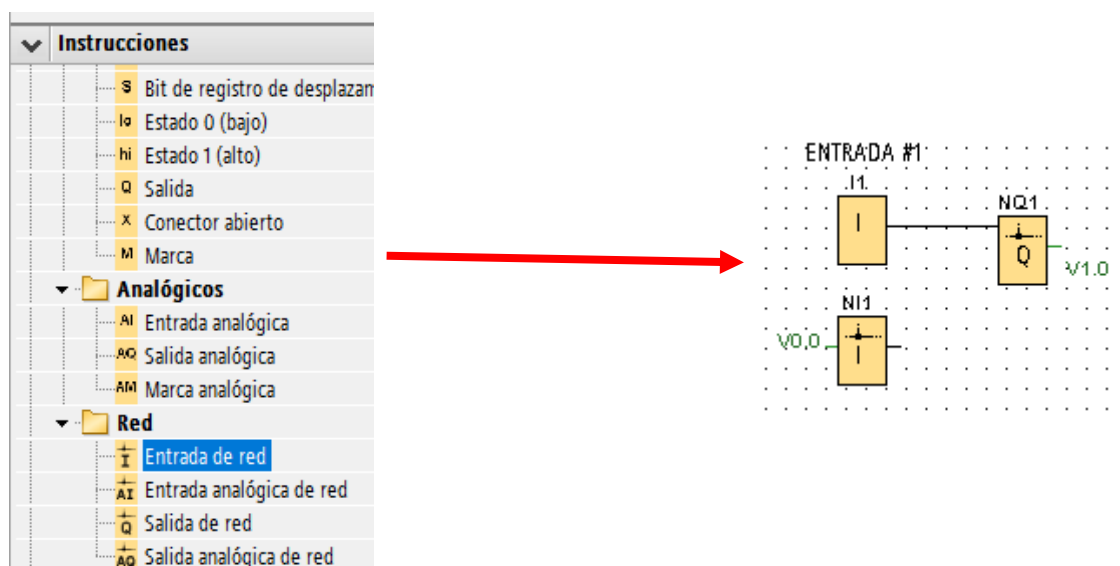


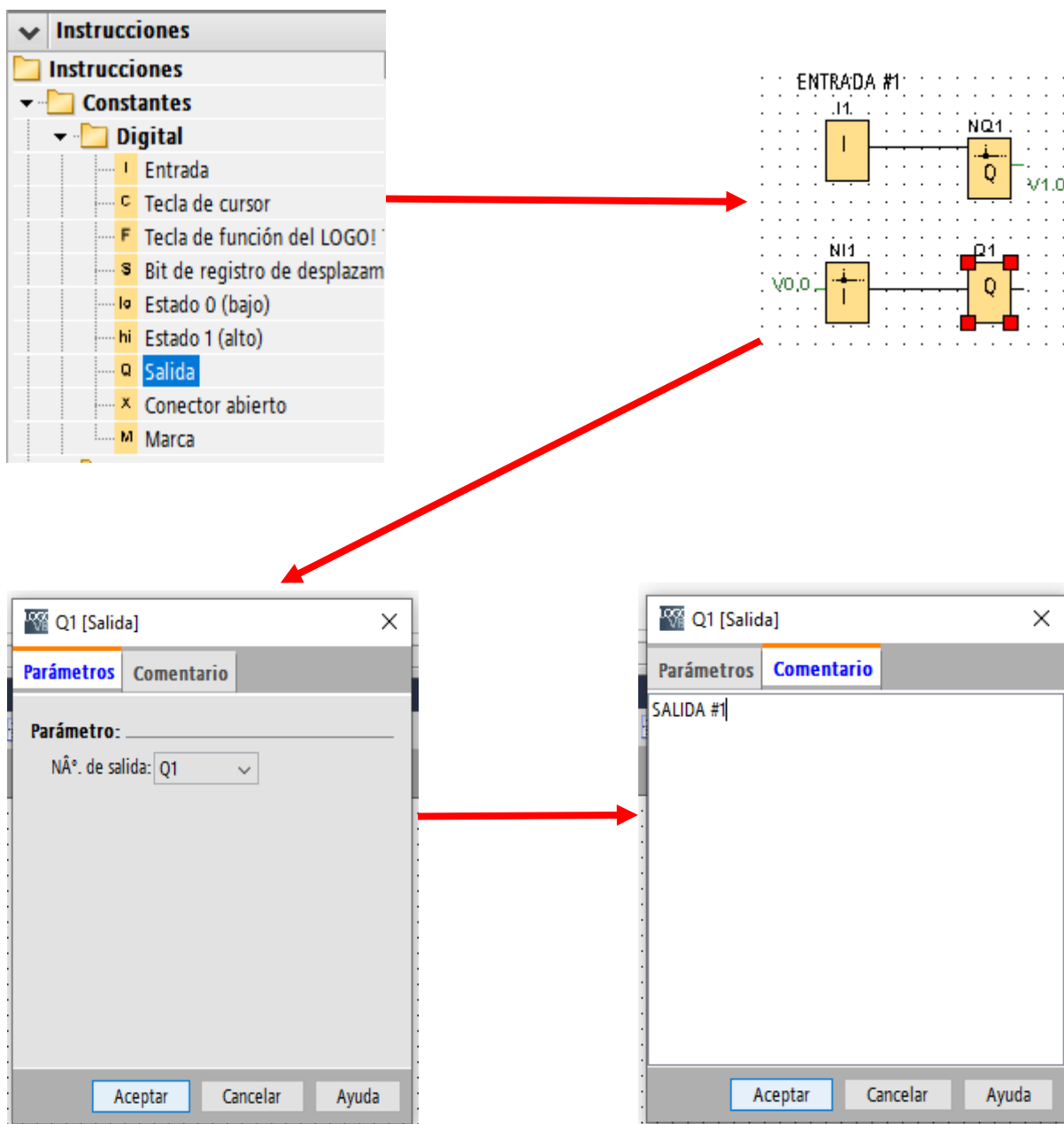
9. Insertar un bloque de constante digital de salida.





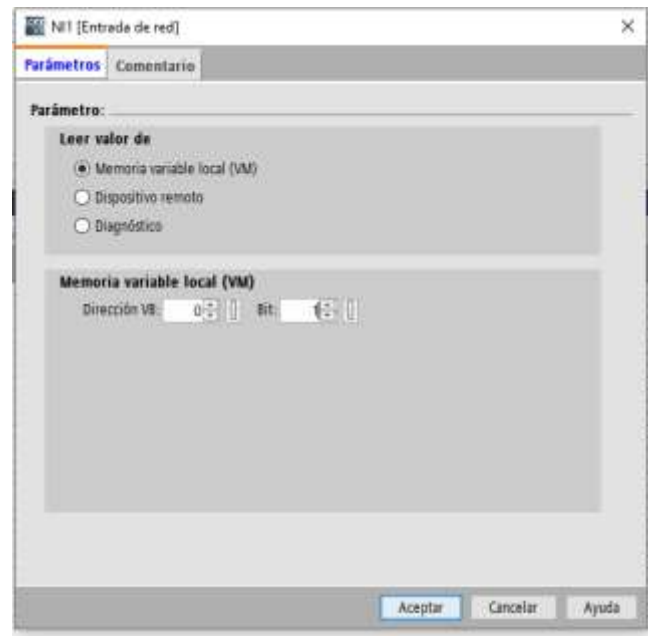
10. Insertar una entrada de red y una salida física del logo



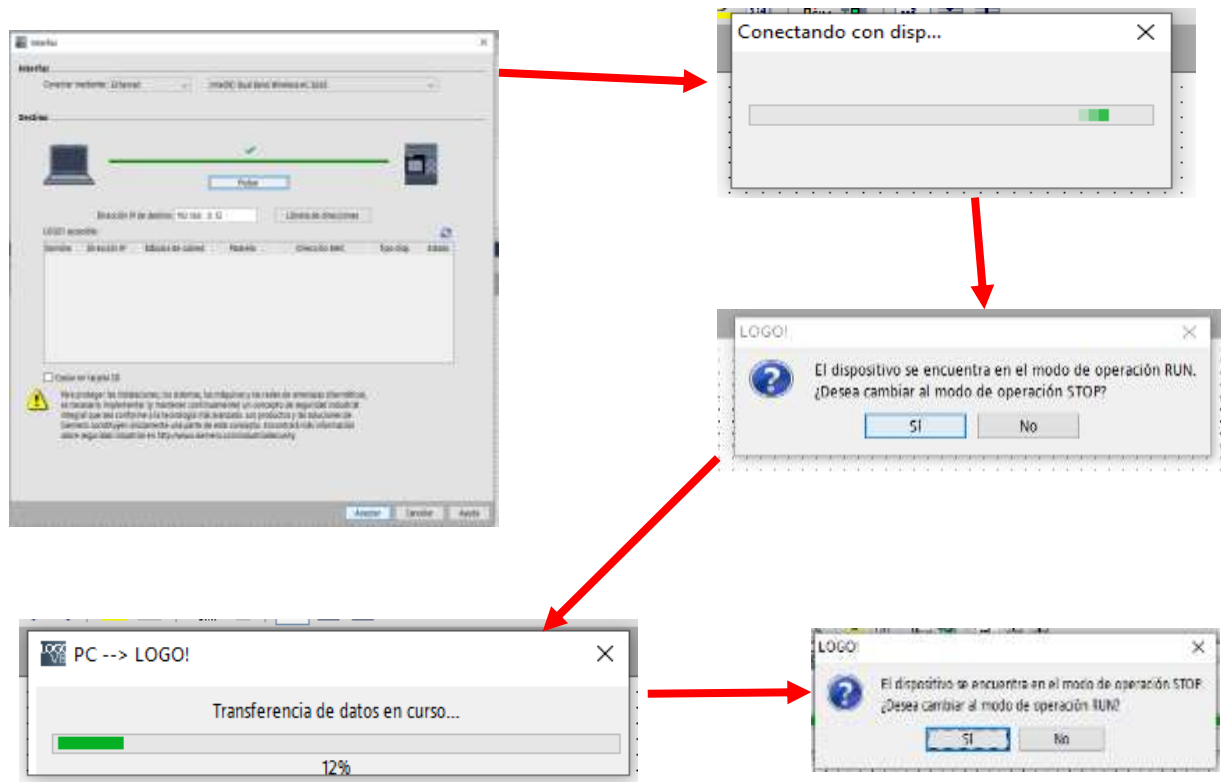


11. Las conexiones se pueden ver de esta manera además se debe de digitar la memoria interna del cada bloque de red tanto de salida como, de entrada.





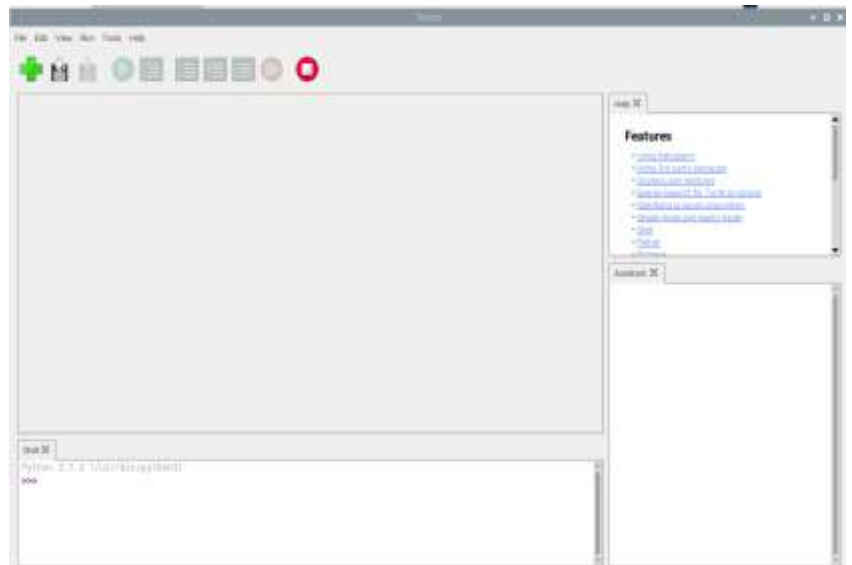
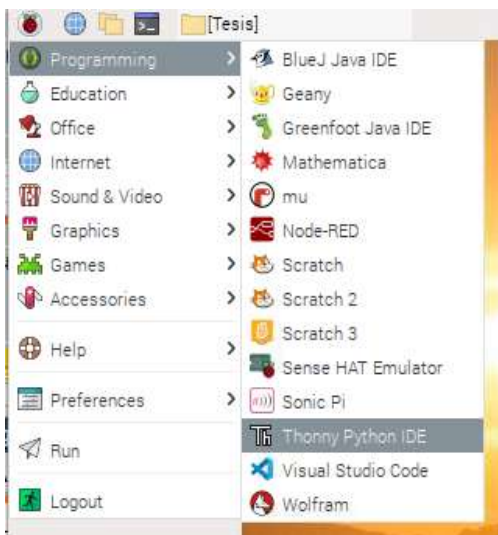
12. Cargar el programa en el icono del logo con la flecha hacia abajo



13. Abrir la Raspberry Pi para poder cargar el programa en el lenguaje Python para establecer la comunicación entre la Raspberry Pi y el LOGO!.

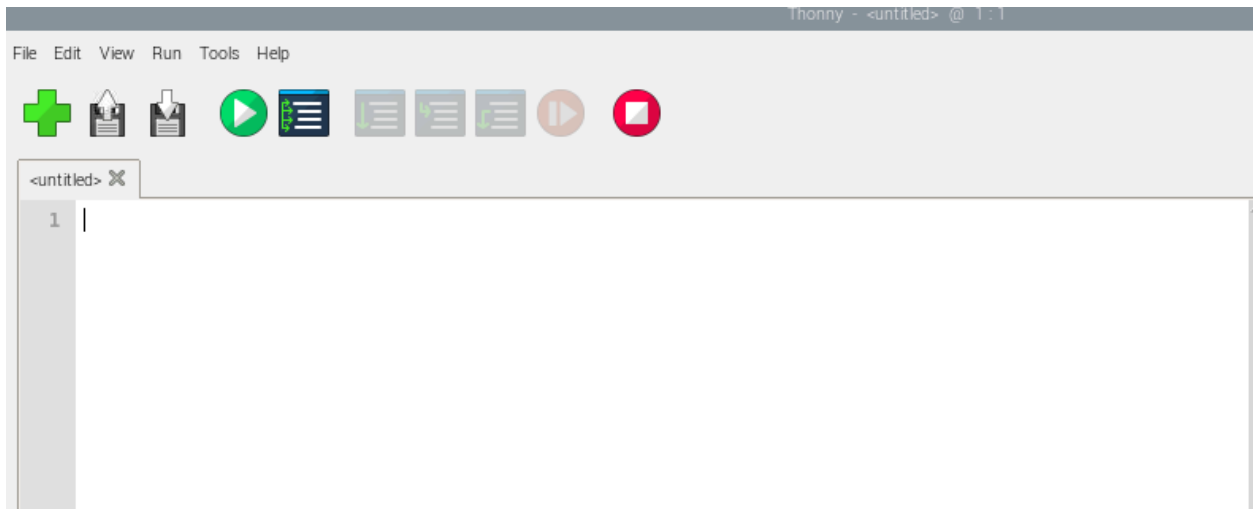


14. Buscar el programa Thonny Python IDE en el menú y se visualizara la página principal.



15. Abrir una nueva hoja de trabajo donde aparece el entorno de programación.





16. Digitar el siguiente código.

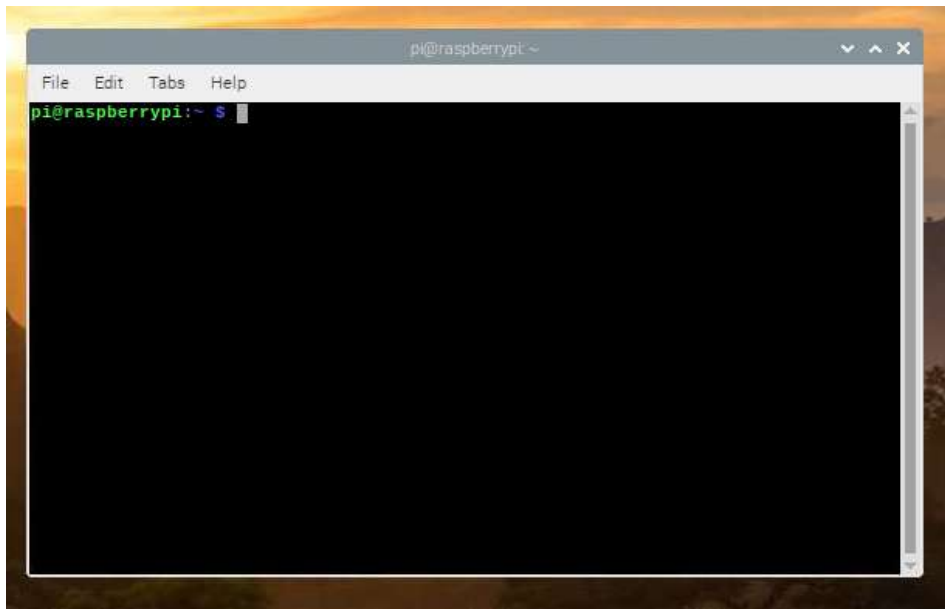
```

1  import time
2  from pyModbusTCP.client import ModbusClient
3
4  # init modbus client
5  c = ModbusClient(host="192.168.0.12", auto_open=True, auto_close=True)
6
7  # main read loop
8  while True:
9      # read 8 registers at address 0, store result in regs list
10     regs_l = c.read_holding_registers(0,8)
11
12     # if success display registers
13     if regs_l:
14         print('reg ad #1 to 8: %s' % regs_l)
15     else:
16         print('unable to read registers')
17
18     # sleep 2s before next polling
19     time.sleep(2)

```

17. Usar la terminal de la Raspberry Pi para ejecutar el programa.





18. Digitar los siguientes códigos en la terminal para ubicar el programa y ejecutarlo.

```
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
```

```
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $
```

```
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ cd New
```

```
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ cd New
pi@raspberrypi:~/Desktop/New $ ls
```

```
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ cd New
pi@raspberrypi:~/Desktop/New $ ls
12345.py
pi@raspberrypi:~/Desktop/New $ python3 ./12345.py
```

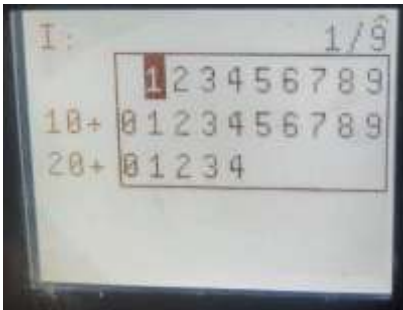
```
pi@raspberrypi: ~/Desktop/New
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ cd New
pi@raspberrypi:~/Desktop/New $ ls
12345.py
pi@raspberrypi:~/Desktop/New $ python3 ./12345.py
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
```

```

File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ cd New
pi@raspberrypi:~/Desktop/New $ ls
12345.py
pi@raspberrypi:~/Desktop/New $ python3 ./12345.py
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [0, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [1, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [1, 0, 0, 0, 0, 0, 0, 0]
reg ad #1 to 8: [1, 0, 0, 0, 0, 0, 0, 0]

```

19. Las señales del logo



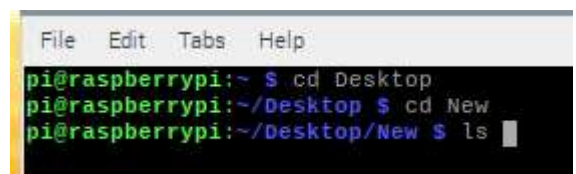
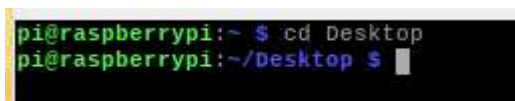
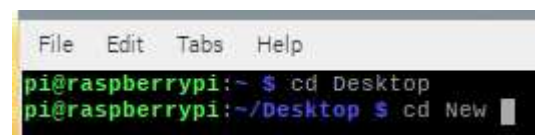
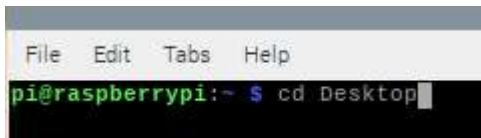
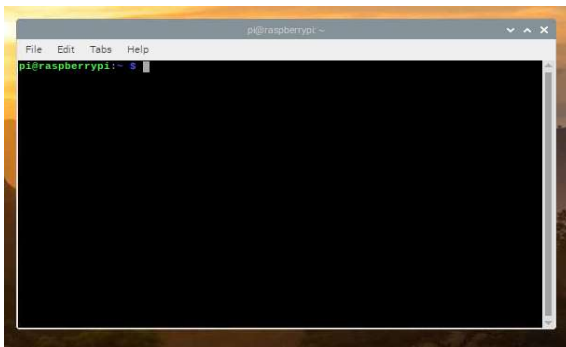
En el programa anterior solo se pueden visualizar las entradas del logo.



20. Para poder ver los estados de las entradas y las salidas se digitara el siguiente código.

```
entradas_y_salidas.py
1 from pyModbusTCP.client import ModbusClient
2 c = ModbusClient(host="192.168.0.12", auto_open=True, auto_close=True)
3
4 regs = c.read_discrete_inputs(0,8)
5 if regs:
6     print("entradas")
7
8     print(regs)
9 else:
10    print("read error")
11    time.sleep(2)
12
13 regs = c.read_coils(0,4)
14 if regs:
15     print("salidas")
16     print(regs)
17 else:
18     print("read error")
19
```

Se guarda para poder utilizarlo de nuevo en la terminal de la Raspberry Pi



```
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ cd New
pi@raspberrypi:~/Desktop/New $ ls
entradas.py  entradas_y_salidas.py
pi@raspberrypi:~/Desktop/New $
```

```
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ cd New
pi@raspberrypi:~/Desktop/New $ ls
entradas.py  entradas_y_salidas.py
pi@raspberrypi:~/Desktop/New $ python3 ./entradas_y_salidas.py
```

```
File Edit Tabs Help
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ cd New
pi@raspberrypi:~/Desktop/New $ ls
entradas.py  entradas_y_salidas.py
pi@raspberrypi:~/Desktop/New $ python3 ./entradas_y_salidas.py
entradas
[False, False, False, False, False, False, False, False]
salidas
[False, False, False, False]
pi@raspberrypi:~/Desktop/New $ python3 ./entradas_y_salidas.py
entradas
[True, False, False, False, False, False, False, False]
salidas
[False, False, False, False]
pi@raspberrypi:~/Desktop/New $
```

V. PRUEBA DE CONOCIMIENTOS

1. Investigar las instrucciones realizadas en lenguaje Python
2. Cuáles son las aplicaciones que se pueden realizar con la Raspberry Pi.

VI. CONCLUSIONES DE LA PRÁCTICA

Escribir 3 conclusiones de la práctica realizada.

1. _____

2. _____

3. _____

CAPITULO 5 CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES:

- Desarrollo de código en Python con Modbus TCP/IP: Se logró desarrollar con éxito un código en Python que permite la implementación del protocolo Modbus TCP/IP, habilitando la comunicación con dispositivos PLC. Este avance facilita la integración con sistemas SCADA, lo que abre la posibilidad de su implementación en la materia de Diseño Electrónico. La utilización de Python para este propósito se demostró eficaz, permitiendo a los estudiantes realizar simulaciones y pruebas con un enfoque práctico en el entorno industrial.
- Evaluación de librerías Modbus TCP/IP: La investigación realizada sobre las diversas librerías de comunicación Modbus TCP/IP reveló que existen opciones variadas con diferentes grados de compatibilidad y facilidad de uso. Las librerías más destacadas demostraron ser efectivas en su interacción con sistemas industriales, destacándose por su flexibilidad, aunque se identificaron áreas de mejora en cuanto a su documentación y soporte en entornos más complejos. Esto sugiere la necesidad de continuar explorando y mejorando las herramientas disponibles.
- Ventajas y limitaciones de Modbus en entornos SCADA: A lo largo del análisis de las ventajas y limitaciones de la comunicación Modbus en sistemas SCADA, se concluyó que, a pesar de ser un protocolo robusto y ampliamente utilizado, presenta algunas restricciones en cuanto a la velocidad de comunicación y la capacidad de manejo de grandes volúmenes de datos. Sin embargo, su simplicidad y facilidad de implementación continúan siendo aspectos clave que favorecen su uso en entornos de automatización industrial, especialmente en aplicaciones de pequeña y mediana escala.

- **Diseño de una interfaz de usuario en Python:** Se diseñó e implementó una interfaz gráfica en Python que permite la visualización y control de dispositivos industriales a través de la red Modbus, demostrando con éxito su integración con un sistema SCADA. La interfaz fue pensada para ser intuitiva, facilitando la interacción del usuario con los dispositivos conectados y permitiendo un monitoreo efectivo de los procesos industriales. Este diseño refuerza la importancia de la usabilidad en sistemas de automatización.
- **Elaboración de guías prácticas para la enseñanza:** Las guías prácticas desarrolladas proporcionan las competencias necesarias y detalladas para comprender y aplicar el protocolo Modbus en redes de comunicación industrial dentro de los sistemas SCADA. Estas guías están diseñadas específicamente para las materias de Autómatas Programables y Procesamiento Digital de Señales, lo que permitirá a los estudiantes tener una comprensión más profunda y aplicar el conocimiento adquirido en situaciones reales de automatización industrial.

5.2. RECOMENDACIONES:

- **Continuar la investigación** acerca de La tecnología industrial y las prácticas de comunicación por que estas están en constante evolución.
- **Explorar otras tecnologías:** Aunque el protocolo Modbus es ampliamente utilizado, existen otras tecnologías de comunicación industrial que pueden ofrecer diferentes ventajas en términos de rendimiento, seguridad y funcionalidad. Explorar y comparar estas tecnologías puede ampliar la comprensión y ayudar a tomar decisiones informadas en futuros proyectos.

- Investigar detalladamente Python: Continuar mejorando las habilidades en Python, explorando bibliotecas y frameworks específicos para la comunicación industrial, como pyModbus, pymodbus3, o Modbus-Tk. Aprender a utilizar estas herramientas permitirá desarrollar soluciones más eficientes y sofisticadas.
- Explorar la integración con otras tecnologías: Además del protocolo Modbus, investigar cómo Python puede integrarse con otras tecnologías relevantes en el campo de la comunicación industrial, como MQTT para la mensajería de máquina a máquina (M2M) o OPC-UA para la interoperabilidad entre sistemas de control.

CAPITULO 6 REFERENCIA BIBLIOGRAFICA.

6.1 REFERENCIAS BIBLIOGRÁFICAS

[1]: Proceso de conexión de interfaz entre dispositivos de entrada y salida a los plc's

Proceso de conexión de interfaz entre dispositivos de entrada y salida a los plc's. (2022). Retrieved 9 December 2022, from <https://prezi.com/p/hkgszethynj2/proceso-de-conexion-de-interfaz-entre-dispositivos-de-entrada-y-salida-a-los-plcs/>

[2]: Contactores: ¿Qué son y para qué sirven? | NIVIHE S.A.

Contactores: ¿Qué son y para qué sirven? | NIVIHE S.A. (2021). Retrieved 9 December 2022, from <https://motores-electricos.com.ar/contactores-que-son-y-para-que-sirven/>

[3]: Quero - los 5 niveles de la automatización industrial

QUERO - LOS 5 NIVELES DE LA AUTOMATIZACIÓN INDUSTRIAL. (2022). Retrieved 9 December 2022, from <https://www.queroautomation.com/los-5-niveles-de-la-automatizacion-industrial/>

[4]: Pirámide De Automatización: Qué Es Y Sus 5 Niveles | Sdi

Pirámide De Automatización: Qué Es Y Sus 5 Niveles | Sdi. (2022). Retrieved 9 December 2022, from <https://sdindustrial.com.mx/blog/piramide-de-automatizacion-que-es/#:~:text=%C3%8Dndice-,%C2%BFQu%C>

[5]: Contactor Funcionamiento Monofasico y Trifasico

"Contactor Funcionamiento Monofasico Y Trifasico". *Areatecnologia.Com*, 2022, <https://www.areatecnologia.com/electricidad/contactor.html>. Accessed 9 Dec 2022.

[6]: Anon

(2022). Retrieved 9 December 2022, from https://support.databoom.com/hc/en-us/article_attachments/360007003740/LOG

[7]: Anon

(2022). Retrieved 9 December 2022, from https://support.databoom.com/hc/en-us/article_attachments/360007014659/siem

[8]: Introducción al conjunto de protocolos TCP/IP (Guía de administración del sistema: servicios IP)

Introducción al conjunto de protocolos TCP/IP (Guía de administración del sistema: servicios IP) . (2022). Retrieved 9 December 2022, from <https://docs.oracle.com/cd/E19957-01/820-2981/6nei0r0r9/index.html#:~:text=El%20m>

[9]: Estándares del modelo OSI | KeepCoding Tech School

Estándares del modelo OSI | KeepCoding Tech School. (2022). Retrieved 9 December 2022, from <https://keepcoding.io/blog/estandares-del-modelo-osi/#:~:text=Los%20est%C3%A1ndares%20>

[10]: Modelo OSI - Concepto, cómo funciona, para qué sirve y capas

Modelo OSI - Concepto, cómo funciona, para qué sirve y capas. (2022). Retrieved 9 December 2022, from <https://concepto.de/modelo-osi/>

[11]: MODELO OSI CAPA FISICA Y CAPA ENLACE - ppt descargar

MODELO OSI CAPA FISICA Y CAPA ENLACE - ppt descargar. (2022). Retrieved 9 December 2022, from <https://slideplayer.es/slide/1547123/>

[12]: 3.11.1 Documentation

3.11.1 Documentation. (2022). Retrieved 9 December 2022, from <https://docs.python.org/3/>

[13]: BeginnersGuide - Python Wiki

BeginnersGuide - Python Wiki. (2022). Retrieved 9 December 2022, from <https://wiki.python.org/moin/BeginnersGuid>

[14]: Learn Python - Free Interactive Python Tutorial

Learn Python - Free Interactive Python Tutorial. (2022). Retrieved 9 December 2022, from <https://www.learnpython.org/>

[15]: Learn Python the Hard Way

Learn Python the Hard Way. (2022). Retrieved 9 December 2022, from <https://learnpythonthehardway.org/>

[16]: Raspberry Pi Documentation

Raspberry Pi Documentation. (2022). Retrieved 9 December 2022, from <https://www.raspberrypi.com/documentation>

[17]: Capa de sesión del modelo OSI: Conceptos básicos de los protocolos, autenticación y autorización - ComoFriki

Capa de sesión del modelo OSI: Conceptos básicos de los protocolos, autenticación y autorización - ComoFriki. (2019). Retrieved 9 December 2022, from <https://comofriki.com/capa-de-sesion-del-modelo-osi-conceptos-basicos-de-los-protocolos-autenticacion-y-autorizacion/>

[18]: Protocolos de comunicación: tramas Modbus | TCP & RTU

Protocolos de comunicación: tramas Modbus | TCP & RTU. (2018). Retrieved 9 December 2022, from <https://vestertraining.com/blog/tramas-modbus/#:~:text=La%20trama%20Modbus,>

[19]: Think Python – Green Tea Press

Think Python – Green Tea Press. (2022). Retrieved 9 December 2022, from <https://greenteapress.com/wp/think-python/>

CAPITULO 7 ANEXOS.

7.1 DISEÑO Y CONSTRUCCION DEL MÓDULO DEL PROYECTO.

7.1.1 Diseño del Módulo.

El objetivo del proyecto es el diseño y construcción de un módulo, para que los estudiantes lo puedan utilizar y manejar en las prácticas de laboratorio de la carrera de ingeniería en electrónica, favoreciendo al estudiante con una herramienta de aprendizaje, capacitación y así poder consolidar los conceptos teóricos mediante la utilización de este módulo.



Ilustración 61 Diseño del modulo

7.1.2. Descripción del modulo

El módulo de control consiste en una aplicación SCADA para el control de dispositivos industriales mediante el protocolo modbus. El dispositivo de control principal es una Raspberry Pi que se utiliza como interface, donde se ejecuta un programa desarrollado con el lenguaje de programación Python, este sistema SCADA se comunica con el controlador LOGO V8, realizando la inspección del estado de las entradas y donde se podrán activar las salidas.

El sistema SCADA se visualizara desde el monitor instalado en el módulo y se conecta un teclado y un mouse para poder controlar las operaciones realizada por la Raspberry Pi, también se han instalado 8 interruptores que serán conectados a la entrada del LOGO V8, con su respectiva conexión a un piloto led para indicar la activación de cada uno de las entradas.

7.1.3 Caracterización del modulo

Para tener una correcta identificación y familiarización con los dispositivos del modulo se ha dividido en cuatro partes:

1. Estructura física.
2. Equipo de control.
3. Diagrama eléctrico y conexión.
4. Programa de aplicación

7.1.3.1 Estructura física.

El modulo está fabricado de madera melanina que tiene como medidas 180cm x 150cm x 50 cm. El modulo es de color blanco cuenta con dos partes móviles que sobre salen del módulo para colocar el teclado y el mouse tiene solo un acceso al interior del módulo donde se observaran las conexiones eléctricas.

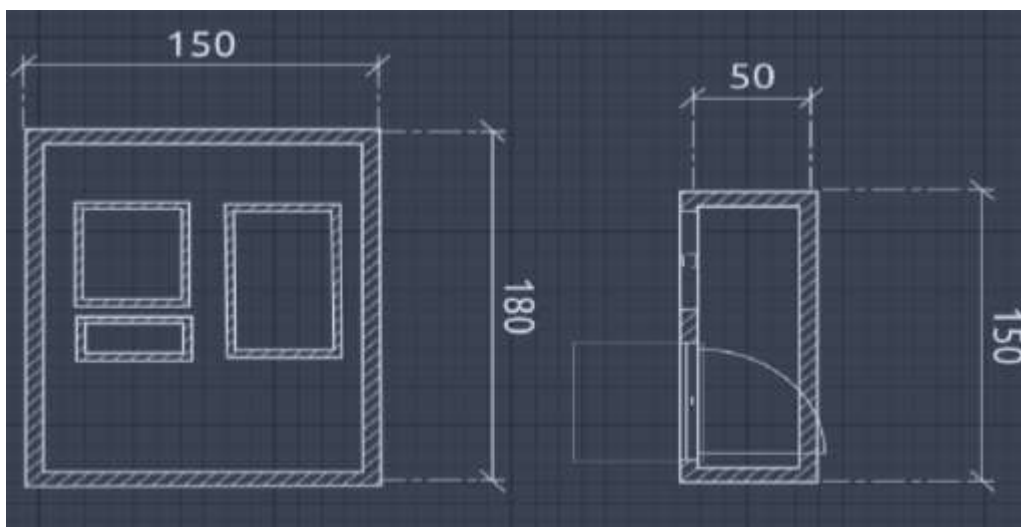


Ilustración 62 Medidas del módulo



Ilustración 63 elección de madera



Ilustración 64 edición y corte de las partes de modulo



Ilustración 65 Módulo completamente armado

7.1.3.2 Equipo de control.

Equipo de control es el nombre que recibe todo el conjunto de elementos eléctricos y electrónicos, que están conectados entre sí y que controlan la energía eléctrica que pasa por un equipo. Los equipos de control eléctrico y electrónico tienen diferentes modalidades que varían en función de su funcionamiento.

La primera de ellas se basa en el accionamiento manual que se produce cuando una persona controla la conexión del aparato. Para llevarla a cabo recurre a elementos externos como pulsadores o interruptores.

El funcionamiento automático, formado por una serie de componentes que solo se activan con una señal eléctrica. El ejemplo más habitual es el de las puertas que tienen acoplados sensores de movimiento y que se abren y se cierran cuando perciben una presencia.

7.1.3.2.1. Componentes de los equipos de control

Un aspecto importante del equipo de control eléctrico son sus componentes. Entre ellos se encuentran los interruptores que permiten el paso de la corriente o lo interrumpen. Entre otros elementos importantes se pueden ver a los pulsadores que actúan igual que los interruptores.

En este módulo se tienen 8 interruptores de control en cada uno de los lados del módulo

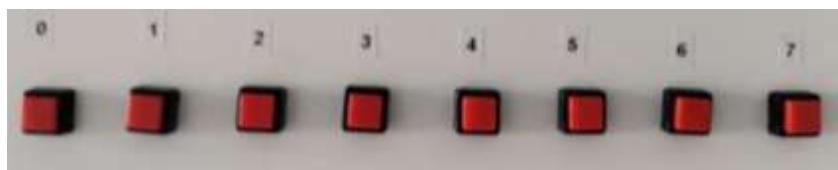


Ilustración 66 Interruptores

El piloto luminoso son indicadores de visualización que proporcionan alguna señal luminosa para el funcionamiento de un circuito en concreto. Por ello, su función es iluminar y señalar los estados del proceso que se va a llevar a cabo.

Asimismo, estos se componen de un cuerpo que adquiere una lámpara de señalización y los contactos de conexión para su instalación. Además, tiene una tapa de color (como verde o roja) para desprender una luz envolvente.

En el módulo se han instalado 8 indicadores luminoso de color verde, para indicar la activación de cada uno de los interruptores que se conectan al LOGO!



Ilustración 67 indicadores luminosos

Los terminales eléctricos se refieren al extremo de un conductor, red o dispositivo. Incluso un terminal puede ser una referencia a un conector que está conectado a este punto final, que puede usarse como un punto para conectar circuitos externos. Pueden ser de varios tipos, dependiendo de su uso.



Ilustración 68 borneras de conexión, salida del LOGO!

Los bornes de conexión eléctrica son los contactos que se utilizan para derivar la energía, que conectan cables de alimentación eléctrica a los terminales para derivar la señal hacia otro componente eléctrico.



Ilustración 69 borneras de conexión de control

El interruptor magneto térmico, permite el paso de energía para poder energizar los componentes de control, además también son protecciones eléctricas por cualquier fallo que se genera en el sistema eléctrico.



Ilustración 70 Protección eléctrica, automático 2 Amperios.

El LOGO! de Siemens es un módulo lógico inteligente destinado a pequeños proyectos de automatización en entornos industriales, en aplicaciones para espacios de oficina y/o locales comerciales y para uso doméstico.

Este mini PLC está implantado en todo el mundo y puede controlarse a distancia. Tiene tareas de automatización muy sencillas de programar mediante un software intuitivo de fácil manejo.



Ilustración 71 LOGO!

La Raspberry Pi es una computadora de bajo costo y con un tamaño compacto, del porte de una tarjeta de crédito, puede ser conectada a un monitor de computador o un TV, y usarse con un mouse y teclado estándar. Es un pequeño computador que corre un sistema operativo Linux capaz de permitirle a las personas de todas las edades explorar la computación y aprender a programar lenguajes como Scratch y Python. Es capaz de hacer la mayoría de las tareas típicas de un computador de escritorio, desde navegar en internet, reproducir videos en alta resolución, manipular documentos de ofimática, hasta reproducir juegos.

Además la Raspberry Pi tiene la habilidad de interactuar con el mundo exterior, puede ser usada en una amplia variedad de proyectos digitales, desde reproductores de música y video, detectores de padres, estaciones meteorológicas hasta cajas de aves con cámaras infrarrojas. Queremos que veas que la Raspberry Pi puede ser usada por niños y adultos por todas partes del mundo, para aprender a programar y entender cómo funcionan las computadoras.



Ilustración 72 Raspberry Pi

En este módulo es necesario tener una pantalla para visualizar el entorno del sistema operativo de la Raspberry Pi, también se conecta u teclado y un mouse para poder interactuar con el software de la Raspberry Pi.



Ilustración 73 Monitor



Ilustración 74 Teclado y Mouse

7.1.3.3. Diagrama eléctrico y conexión

Se muestra el diagrama eléctrico para el encendido y apagado de cada lado del módulo.

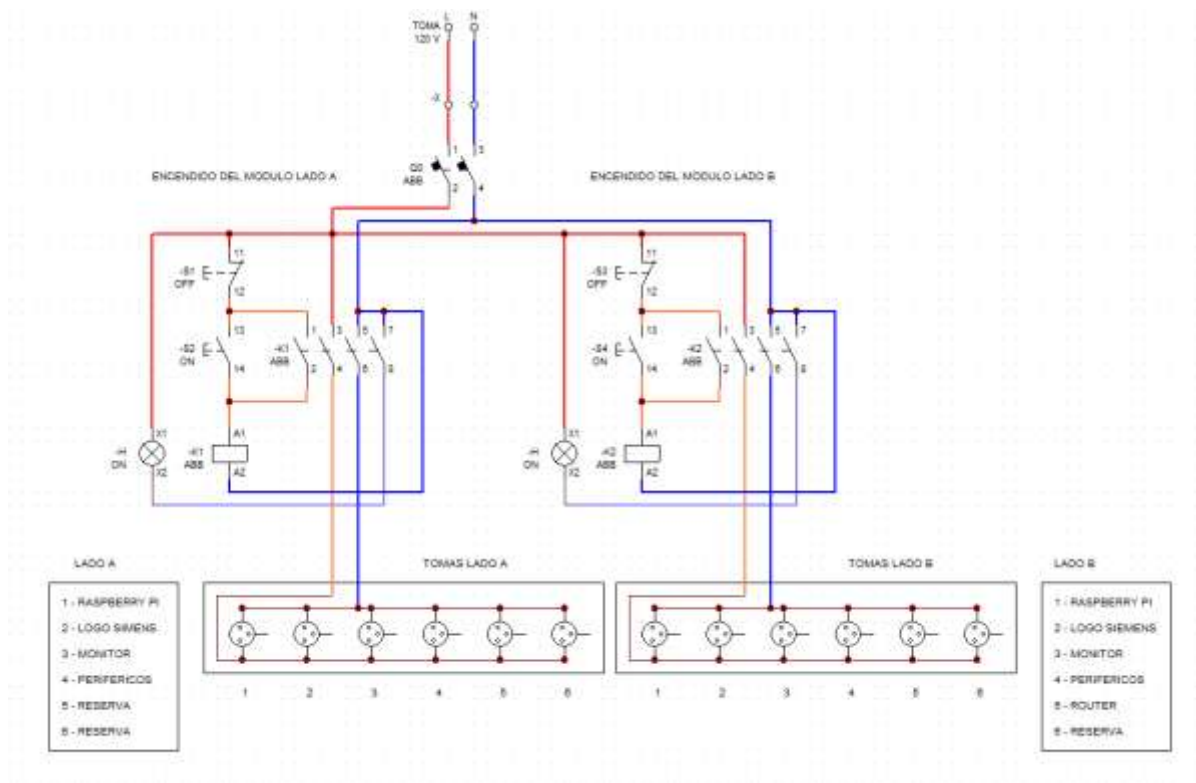


Ilustración 75 Conexión de alimentación eléctrica

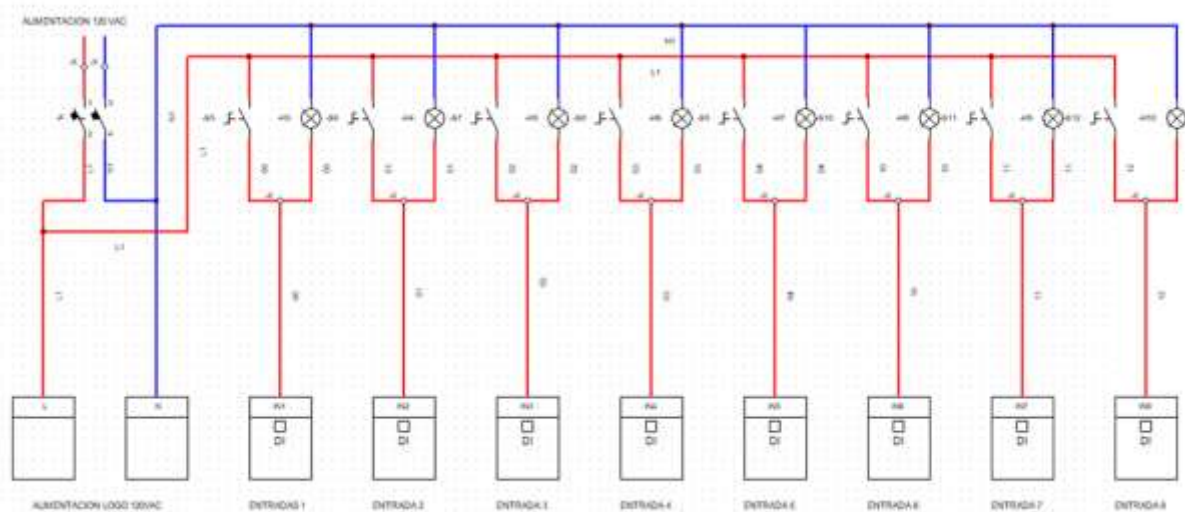


Ilustración 76 Conexión eléctrica de las entradas del LOGO!

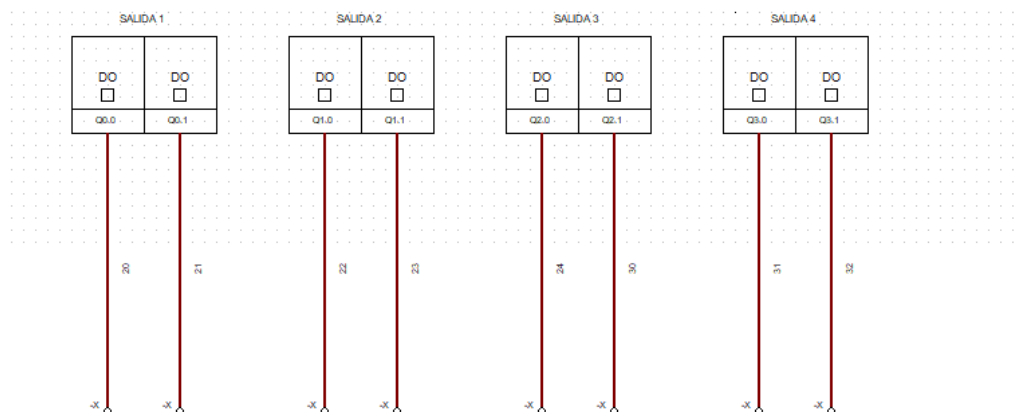


Ilustración 77 Conexión de las salidas del logo

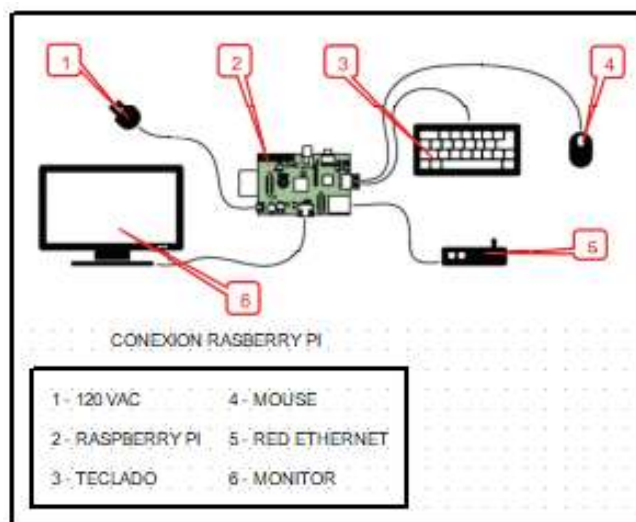


Ilustración 78 Conexión eléctrica de Raspberry Pi

Las conexiones eléctricas de todos los equipos utilizados se pueden ver a continuación.



Ilustración 79 Modulo conectado y funcionando cada uno de sus componente.

Conexiones eléctricas realizadas en el módulo.



Ilustración 80 Conexión interna del módulo de los pilotos



Ilustración 81 Conexión eléctrica de los componentes del modulo

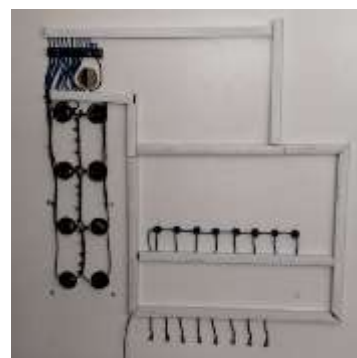


Ilustración 82 Canaleta utilizada para organizar el cable



Ilustración 83 Conexión de señales del Raspberry Pi, monitor, teclado y mouse.



Ilustración 84 Switch hub para la conexión de los cables de red

Las conexiones eléctricas se modificaron en el interior del módulo a continuación se pueden observar:



Ilustración 85 Conexiones eléctricas modificadas

Se instaló una nueva caja de registro eléctrico y se cambió la canaleta eléctrica para una mejor presentación.

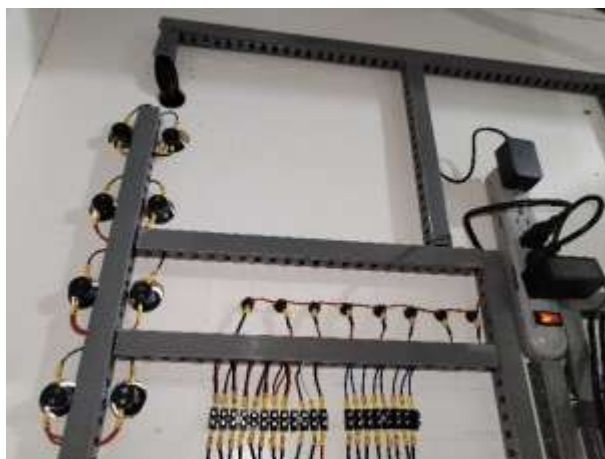


Ilustración 86 Canaleta y conexiones de control

El cableado de los componentes se encuentra organizado y ordenado y no se tienen los cables expuestos.



Ilustración 87 Switch de red conectado ordenado los cables ya no están expuestos

7.1.3.4. Programa de aplicación

A continuación, se muestra el programa de la interface grafica para la comunicación ModBus TCP, con un LOGO Siemens en esta interface grafica se muestran los estados y salidas de logo de una manera visual para ver el estado de cada uno de ellos donde también se pueden activar las salidas del logo.



Ilustración 88 Interface grafica

7.1.3.4.1 Estructura de logo

Todos los módulos LOGO! Básicos disponen de las siguientes conexiones para crear el programa, independientemente del número de módulos que se conecten:

- Entradas digitales I1 hasta I24
- Entradas analógicas AI1 hasta AI8
- Salidas digitales Q1 hasta Q16
- Salidas analógicas AQ1 y AQ12
- Marcas digitales M1 hasta M24, M8: marcas de arranque
- Marcas analógicas AM1 hasta AM6
- Bits de registro de desplazamiento S1 hasta S8
- 4 teclas de cursor
- 16 salidas no conectadas X1 hasta X16.

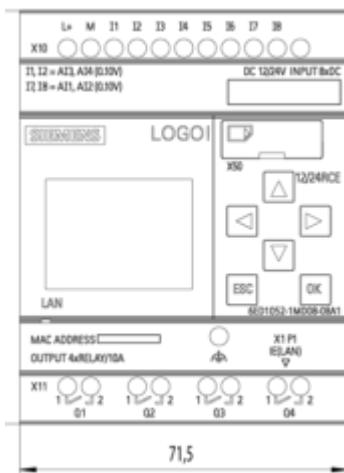


Ilustración 89 estructura del LOGO!

7.1.3.4.2. Entradas de logo

En la interface grafica del programa se puede observar unos indicadores luminosos para ver el estado actual de los mismos, cada vez que se accione una de las entradas esos indicadores luminosos se encenderán.



Ilustración 90 Indicadores luminosos desactivado en interface grafica

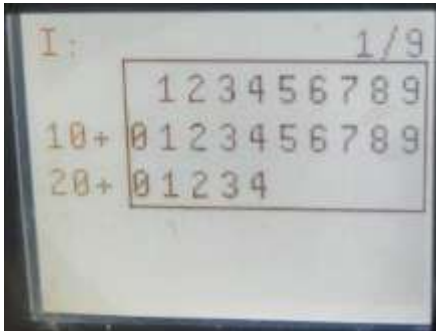


Ilustración 91 Visualización de las entradas en, pantalla del LOGO!



Ilustración 92 Indicador luminoso activada en interface gráfica

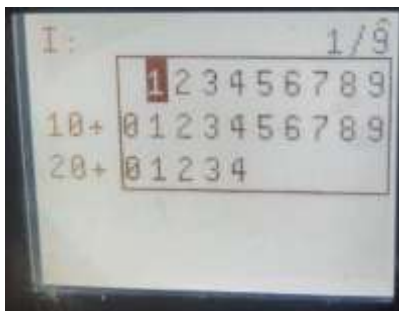


Ilustración 93 Entrada activa en pantalla del LOGO!

7.1.3.4.3. Salidas de logo

Podemos observar también 4 indicadores luminosos que indican los estados de las salidas de logo, también hay un estado de apagado y otro de encendido.

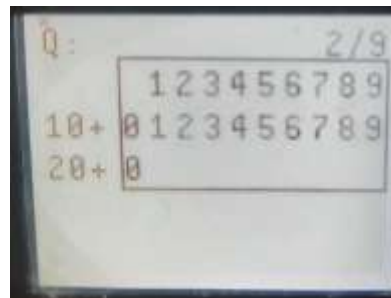


Ilustración 94 Indicador luminoso desactivado en interface gráfica y en pantalla del LOGO!

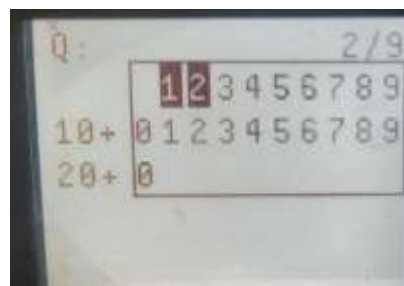


Ilustración 95 Indicador luminoso activado en interface gráfica y pantalla del LOGO!

De la interface grafica también se pueden activar cada uno de las salidas, por medio de 4 botones que se agregaron a la interface también esos botones cambian de estado de apagado a encendidos (o activador).



Ilustración 96 Interruptores e indicadores luminosos en interface gráfica desactivados y activados

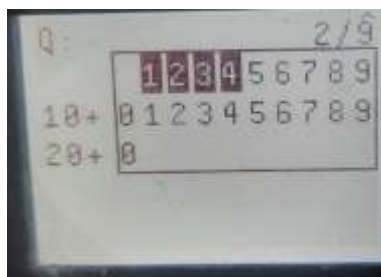


Ilustración 97 Salidas activas desde la interface gráfica

En la interface grafica también se pueden observar unos botones de información necesaria para el programa.

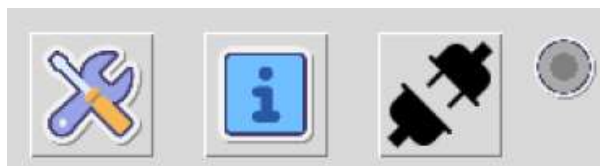


Ilustración 98 Iconos de ajustes



Configuración



Información



Información de conexión



Indicador luminoso estado de conexión

7.1.3.4.4 Configuración



En este botón se configura la dirección IP para la conexión de la red Modbus TCP, que se está utilizando en la red.



Ilustración 99 Configuración de IP

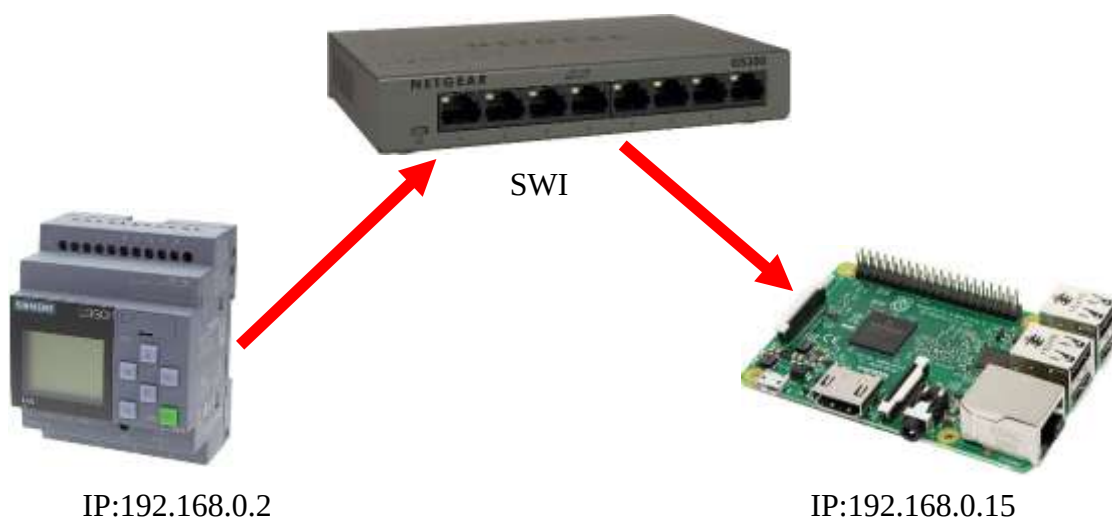


Ilustración 100 Configuración de red

7.1.3.4.5. Información



En este botón hay información de los integrantes del grupo de trabajo.



Ilustración 101 Información del grupo

7.1.3.4.6. Información de conexión



En este botón se establece la conexión entre la Raspberry Pi y el LOGO V8, donde hay dos estados de conexión que son los siguientes:

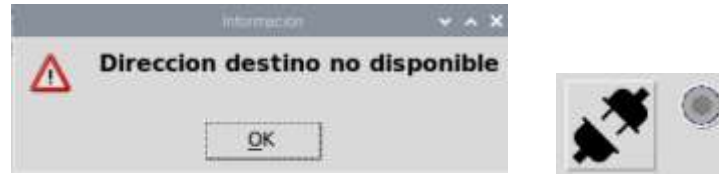


Ilustración 102 Mensaje cuando no está conectado al dispositivo



Ilustración 103 Mensaje conectada

7.1.4 Programa de la aplicación SCADA en PYTHON.

```
import tkinter
from tkinter import messagebox
from pyModbusTCP.client import ModbusClient
import tkinter as tk
from tkinter import *
from PIL import Image, ImageTk
import array as arr
import numpy as np
import os

SERVER_HOST = '192.168.0.2'
SERVER_PORT = 502
LOCATION = 'Resources/'
LOCATION_OUT_ON = (' ')
LOCATION_OUT_OFF = (' ')
LOCATION_IN_ON = (' ')
LOCATION_IN_OFF = (' ')
PHYSICAL_INPUTS = []
PHYSICAL_OUTPUTS = []
TOGGLE_OUTPUT = arr.array('b', [False, False, False, False]) #Variables para escribir
modbus
ICON_OUTPUT = [False, False, False, False]
ICON_INPUT = [False, False, False, False, False, False, False, False]
CONNECT_CLIENT = False #Variable para conectar cliente
MODBUS_READY = False

mbClient = ModbusClient(host= SERVER_HOST, port= SERVER_PORT)
#Creando formulario
ventanaPrincipal = tk.Tk()
#ventanaPrincipal.eval('tk::PlaceWindow . center')
ventanaPrincipal.title('Modbus App')
ventanaPrincipal.geometry('580x540')
```

```

#Metodo para toggle de botones
#Imagen encendido - apagado
objOFF_Img = Image.open(LOCATION + 'icons8-toggle-off-100.png')
ImgOff_sized = objOFF_Img.resize((50, 50))
IMAGE_OFF = ImageTk.PhotoImage(ImgOff_sized)

objON_Img = Image.open(LOCATION + 'icons8-toggle-on-100.png')
ImgOn_sized = objON_Img.resize((50, 50))
IMAGE_ON = ImageTk.PhotoImage(ImgOn_sized)

def toggle0():
    if TOGGLE_OUTPUT[0]:
        btnOutput0.config(image = IMAGE_OFF)
        TOGGLE_OUTPUT[0] = False
    else:
        btnOutput0.config(image = IMAGE_ON)
        TOGGLE_OUTPUT[0] = True

def toggle1():
    if TOGGLE_OUTPUT[1]:
        btnOutput1.config(image = IMAGE_OFF)
        TOGGLE_OUTPUT[1] = False
    else:
        btnOutput1.config(image = IMAGE_ON)
        TOGGLE_OUTPUT[1] = True

def toggle2():
    if TOGGLE_OUTPUT[2]:
        btnOutput2.config(image = IMAGE_OFF)
        TOGGLE_OUTPUT[2] = False
    else:
        btnOutput2.config(image = IMAGE_ON)
        TOGGLE_OUTPUT[2] = True

def toggle3():
    if TOGGLE_OUTPUT[3]:
        btnOutput3.config(image = IMAGE_OFF)
        TOGGLE_OUTPUT[3] = False
    else:
        btnOutput3.config(image = IMAGE_ON)
        TOGGLE_OUTPUT[3] = True

#Imagen de boton conectar
objConnect_Img = Image.open(LOCATION + 'icons8-connected-50.png')
ImgConnect_sized = objConnect_Img.resize((50, 50))

```

```

IMAGE_CONNECT = ImageTk.PhotoImage(ImgConnect_sized)

objDis_Img = Image.open(LOCATION + 'icons8-disconnected-50.png')
ImgDis_sized = objDis_Img.resize((50, 50))
IMAGE_DISCONNECT = ImageTk.PhotoImage(ImgDis_sized)

#Imagen de boton indicadores de conexion entradas y salidas
objIconConnect_Img = Image.open(LOCATION + 'icons8-connection-status-on-100.png')
ImgIconConnect_sized = objIconConnect_Img.resize((50, 50))
LOCATION_OUT_ON = ImageTk.PhotoImage(ImgIconConnect_sized)
LOCATION_IN_ON = ImageTk.PhotoImage(ImgIconConnect_sized)

objIconDis_Img = Image.open(LOCATION + 'icons8-connection-status-off-100.png')
ImgIconDis_sized = objIconDis_Img.resize((50, 50))
LOCATION_OUT_OFF = ImageTk.PhotoImage(ImgIconDis_sized)
LOCATION_IN_OFF = ImageTk.PhotoImage(ImgIconDis_sized)

def toggle_connect():
    global CONNECT_CLIENT, MODBUS_READY
    response = os.system("ping -c 1 " + SERVER_HOST)
    if response == 0:
        if CONNECT_CLIENT:
            btnConnect.config(image = IMAGE_DISCONNECT)
            CONNECT_CLIENT = False
            MODBUS_READY = False
            mbClient.close()
        else:
            btnConnect.config(image = IMAGE_CONNECT)
            CONNECT_CLIENT = True
            mbClient.open()
            if mbClient.open():
                MODBUS_READY = True
                messagebox.showinfo("Información", "Cliente conectado")
            else:
                MODBUS_READY = False
                messagebox.showwarning("Información", "Error conectando cliente")
        else:
            messagebox.showwarning("Información", "Direccion destino no disponible")
            btnConnect.config(image = IMAGE_DISCONNECT)

    print("Modbus listo? ", MODBUS_READY)

#Metodo para lectura
def mbRead():
    global MODBUS_READY, PHYSICAL_OUTPUTS, PHYSICAL_INPUTS,
    ICON_OUTPUT, ICON_INPUT

```

```

#Evaluando si cliente conectado
if MODBUS_READY == True:
    rango_salidas = 8192
    PHYSICAL_OUTPUTS = mbClient.read_coils(rango_salidas, 4)
    ICON_OUTPUT = np.array(PHYSICAL_OUTPUTS)
    rango_entradas = 0
    PHYSICAL_INPUTS = mbClient.read_discrete_inputs(rango_entradas, 8)
    ICON_INPUT = np.array(PHYSICAL_INPUTS)
else:
    for i in range(0, 4):
        ICON_OUTPUT[i] = False
    for y in range(0, 4):
        ICON_INPUT[y] = False
ventanaPrincipal.after(1000, mbRead)

def mbWrite():
    global MODBUS_READY, TOGGLE_OUTPUT
    if MODBUS_READY == True:
        if TOGGLE_OUTPUT[0]:
            wr0 = mbClient.write_single_coil(0, True)
        else:
            wr0 = mbClient.write_single_coil(0, False)
        if TOGGLE_OUTPUT[1]:
            wr1 = mbClient.write_single_coil(1, True)
        else:
            wr1 = mbClient.write_single_coil(1, False)
        if TOGGLE_OUTPUT[2]:
            wr2 = mbClient.write_single_coil(2, True)
        else:
            wr2 = mbClient.write_single_coil(2, False)
        if TOGGLE_OUTPUT[3]:
            wr3 = mbClient.write_single_coil(3, True)
        else:
            wr3 = mbClient.write_single_coil(3, False)
    ventanaPrincipal.after(1000, mbWrite)

#####
#####
#Definicion de frames
#####
#####
#Encabezado
frm = tk.Frame(ventanaPrincipal)
frm.grid(column=0, row=0, padx=0, pady=0, sticky='sw')
#Entradas
lblGroup = LabelFrame(ventanaPrincipal, text="Entradas")

```

```

lblGroup.grid(column=0, row=1, padx=10, pady=10)
#Salidas
lblGroup1 = LabelFrame(ventanaPrincipal, text="Salidas")
lblGroup1.grid(column=0, row=2, padx=5, pady=10)
#Botones de comando
frmConfig = tk.Frame(ventanaPrincipal)
frmConfig.grid(column=0, row=3, padx=5, pady=10)

#Metodo para cambio de ruta en los iconos de estado de entradas y salidas
def change_status():
    #salidas
    if ICON_OUTPUT[0] == True:
        lblOutput0.config(image=LOCATION_OUT_ON)
    else:
        lblOutput0.config(image=LOCATION_OUT_OFF)

    if ICON_OUTPUT[1] == True:
        lblOutput1.config(image = LOCATION_OUT_ON)
    else:
        lblOutput1.config(image = LOCATION_OUT_OFF)

    if ICON_OUTPUT[2] == True:
        lblOutput2.config(image = LOCATION_OUT_ON)
    else:
        lblOutput2.config(image = LOCATION_OUT_OFF)

    if ICON_OUTPUT[3] == True:
        lblOutput3.config(image = LOCATION_OUT_ON)
    else:
        lblOutput3.config(image = LOCATION_OUT_OFF)

    #Entradas
    if ICON_INPUT[0] == 1:
        lblInput0.config(image = LOCATION_IN_ON)
    else:
        lblInput0.config(image = LOCATION_IN_OFF)

    if ICON_INPUT[1] == 1:
        lblInput1.config(image = LOCATION_IN_ON)
    else:
        lblInput1.config(image = LOCATION_IN_OFF)

    if ICON_INPUT[2] == 1:
        lblInput2.config(image = LOCATION_IN_ON)
    else:
        lblInput2.config(image = LOCATION_IN_OFF)

```

```

if ICON_INPUT[3] == 1:
    lblInput3.config(image = LOCATION_IN_ON)
else:
    lblInput3.config(image = LOCATION_IN_OFF)

if ICON_INPUT[4] == 1:
    lblInput4.config(image = LOCATION_IN_ON)
else:
    lblInput4.config(image = LOCATION_IN_OFF)

if ICON_INPUT[5] == 1:
    lblInput5.config(image = LOCATION_IN_ON)
else:
    lblInput5.config(image = LOCATION_IN_OFF)

if ICON_INPUT[6] == 1:
    lblInput6.config(image = LOCATION_IN_ON)
else:
    lblInput6.config(image = LOCATION_IN_OFF)

if ICON_INPUT[7] == 1:
    lblInput7.config(image = LOCATION_IN_ON)
else:
    lblInput7.config(image = LOCATION_IN_OFF)

lblGroup1.after(500, change_status)

#Colocando icono enmedio
im = Image.open(LOCATION + 'icons8-hub-100.png')
mainPic = ImageTk.PhotoImage(im)
lbliconoCentral = tkinter.Label(frm, image=mainPic)
lbliconoCentral.image = mainPic
lbliconoCentral.grid(column=0, row=0, padx=220, pady=5)

lblTitle = Label(
    frm,
    text='Modbus TCP',
    font=("Times", "14")
).grid(column=0, row=1, padx=220, pady=5)

lblDevice = Label(
    frm,
    text='SIEMENS - Logo 8.3',
    font=("Times", "14")
).grid(column=0, row=2, padx=5, pady=5, sticky='sw')

```

```
#####
#####
#Grupo para entradas
#####
#####

#Entrada 0
lblInput0 = tkinter.Label(lblGroup)
lblInput0.image = lblInput0
lblInput0.grid(column=0, row=0, padx=5, pady=0)

left0 = Label(lblGroup, text="Entrada 0")
left0.grid(column=0, row=1, padx=5, pady=0)

#Entrada 1
lblInput1 = tkinter.Label(lblGroup)
lblInput1.image = lblInput1
lblInput1.grid(column=1, row=0, padx=5, pady=0)

left1 = Label(lblGroup, text="Entrada 1")
left1.grid(column=1, row=1, padx=5, pady=0)

#Entrada 2
lblInput2 = tkinter.Label(lblGroup)
lblInput2.image = lblInput2
lblInput2.grid(column=2, row=0, padx=5, pady=0)

left2 = Label(lblGroup, text="Entrada 2")
left2.grid(column=2, row=1, padx=5, pady=0)

#Entrada 3
lblInput3 = tkinter.Label(lblGroup)
lblInput3.image = lblInput3
lblInput3.grid(column=3, row=0, padx=5, pady=0)

left3 = Label(lblGroup, text="Entrada 3")
left3.grid(column=3, row=1, padx=5, pady=0)

#Entrada 4
lblInput4 = tkinter.Label(lblGroup)
lblInput4.image = lblInput4
lblInput4.grid(column=4, row=0, padx=5, pady=0)

left4 = Label(lblGroup, text="Entrada 4")
left4.grid(column=4, row=1, padx=5, pady=0)
```

```

#Entrada 5
lblInput5 = tkinter.Label(lblGroup)
lblInput5.image = lblInput5
lblInput5.grid(column=5, row=0, padx=5, pady=0)

left5 = Label(lblGroup, text="Entrada 5")
left5.grid(column=5, row=1, padx=5, pady=0)

#Entrada 6
lblInput6 = tkinter.Label(lblGroup)
lblInput6.image = lblInput6
lblInput6.grid(column=6, row=0, padx=5, pady=0)

left6 = Label(lblGroup, text="Entrada 6")
left6.grid(column=6, row=1, padx=5, pady=0)

#Entrada 7
lblInput7 = tkinter.Label(lblGroup)
lblInput7.image = lblInput7
lblInput7.grid(column=7, row=0, padx=5, pady=0)

left7 = Label(lblGroup, text="Entrada 7")
left7.grid(column=7, row=1, padx=5, pady=0)

#####
#####
#Grupo para salidas
#####
#####

#Salida 0
lblOutput0 = tkinter.Label(lblGroup1)
lblOutput0.image = lblOutput0
lblOutput0.grid(column=0, row=0, padx=5, pady=0)

left_0 = Label(lblGroup1, text="Salida 0")
left_0.grid(column=0, row=1, padx=5, pady=0)

btnOutput0 = Button(lblGroup1, image=IMAGE_OFF, command=toggle0)
btnOutput0.grid(column=0, row=2, padx=5, pady=0)

#Salida 1
lblOutput1 = tkinter.Label(lblGroup1)
lblOutput1.image = lblOutput1
lblOutput1.grid(column=1, row=0, padx=5, pady=0)

```

```

left_1 = Label(lblGroup1, text="Salida 1")
left_1.grid(column=1, row=1, padx=5, pady=0)

btnOutput1 = Button(lblGroup1, image=IMAGE_OFF, command=toggle1)
btnOutput1.grid(column=1, row=2, padx=5, pady=0)

#Salida 2
lblOutput2 = tkinter.Label(lblGroup1)
lblOutput2.image = lblOutput2
lblOutput2.grid(column=2, row=0, padx=5, pady=0)

left_2 = Label(lblGroup1, text="Salida 2")
left_2.grid(column=2, row=1, padx=5, pady=0)

btnOutput2 = Button(lblGroup1, image=IMAGE_OFF, command=toggle2)
btnOutput2.grid(column=2, row=2, padx=5, pady=0)

#Salida 3
lblOutput3 = tkinter.Label(lblGroup1)
lblOutput3.image = lblOutput3
lblOutput3.grid(column=3, row=0, padx=5, pady=0)

left_3 = Label(lblGroup1, text="Salida 3")
left_3.grid(column=3, row=1, padx=5, pady=0)

btnOutput3 = Button(lblGroup1, image=IMAGE_OFF, command=toggle3)
btnOutput3.grid(column=3, row=2, padx=5, pady=0)

#####
#####
#Ventana de configuracion
#####
#####
def ConfigWindow():
    global SERVER_HOST, SERVER_PORT
    top = Toplevel()
    top.geometry("280x440")
    top.title("Configuracion")
    top.grab_set()
    formC = tk.Frame(top)
    formC.pack()

    objConfig_Icon = Image.open(LOCATION + 'icons8-tools-100.png')
    ImgConfig_Icon_sized = objConfig_Icon.resize((100, 100))
    objConfig_Icon_sized = ImageTk.PhotoImage(ImgConfig_Icon_sized)

```

```

lblIcon = tkinter.Label(formC, image=objConfig_Icon_sized)
lblIcon.image = objConfig_Icon_sized
lblIcon.pack(pady=20)

l2Config = Label(formC, text = "Ventana de configuración", font=("Times", "14"))
l2Config.pack()

l2Config_Ip = Label(formC, text = "Dirección IP de servidor: ", font=("Times", "12"))
l2Config_Ip.pack(ipadx=10, ipady=10, fill='x')

txtServer = tk.Text(formC, height = 1, width = 1)
txtServer.pack(ipadx=10, ipady=10, fill='x')

l2Config_Port = Label(formC, text = "Puerto de servidor: ", font=("Times", "12"))
l2Config_Port.pack(ipadx=10, ipady=10, fill='x')

txtServer_Port = tk.Text(formC, height = 1, width = 1)
txtServer_Port.pack(ipadx=10, ipady=10, fill='x')

#Entrada para metodo guardar
def Guardar():
    global SERVER_HOST, SERVER_PORT
    txtServer_In = txtServer.get(1.0, "end-1c")
    SERVER_HOST = txtServer_In
    txtPort_In = txtServer_Port.get(1.0, "end-1c")
    SERVER_PORT = txtPort_In
    top.destroy()

#Entrada de metodo para text hint de cajas de texto
def setText_OnStart():
    txtServer.insert('1.0', SERVER_HOST)
    txtServer_Port.insert('1.0', SERVER_PORT)

btnGuardar = Button(formC, text='GUARDAR', command=Guardar)
btnGuardar.pack(ipadx=10, ipady=10, fill='x')

setText_OnStart()

btnQuitConfig = Button(formC, text='CERRAR', command=top.destroy)
btnQuitConfig.pack(ipadx=10, ipady=10, fill='x')

#####
#####
#Ventana de informacion
#####
#####

```

```

def InfoWindow():
    #Creando ventana informacion
    top_info = Toplevel()
    top_info.geometry("300x420")
    top_info.title("Informacion")
    top_info.grab_set()
    formI = tk.Frame(top_info)
    formI.pack()
    #Colocando icono enmedio
    imgLogo = Image.open(LOCATION + 'utla.png')
    imgLogo_sized = imgLogo.resize((100,120))
    ObjimgLogo = ImageTk.PhotoImage(imgLogo_sized)
    lblLogoUtla = tkinter.Label(formI, image=ObjimgLogo)
    lblLogoUtla.image = ObjimgLogo
    lblLogoUtla.pack(pady=20)

    lblTitle_Info = Label(formI, text='Universidad Técnica Latinoamericana', font=("Times",
"14"))
    lblTitle_Info.pack()

    lblTitle_Integrantes = Label(formI, text='Integrantes:', font=("Times", "14"))
    lblTitle_Integrantes.pack(padx=0, pady=20)

    lblIntegrante1 = Label(formI, text='John Vanega', font=("Times", "14"))
    lblIntegrante1.pack(ipadx=10, ipady=5, fill='x')
    lblIntegrante2 = Label(formI, text='Carlos Aquino', font=("Times", "14"))
    lblIntegrante2.pack(ipadx=10, ipady=5, fill='x')
    lblIntegrante3 = Label(formI, text='Ricardo Monge', font=("Times", "14"))
    lblIntegrante3.pack(ipadx=10, ipady=5, fill='x')

    btnQuit = Button(formI, text='CERRAR', command=top_info.destroy)
    btnQuit.pack(pady=20)

    #####
    #####
    #Botones de comando
    #####
    #####

    #Cambio de pantalla a Config
    objConfig_Img = Image.open(LOCATION + 'icons8-tools-100.png')
    ImgConfig_sized = objConfig_Img.resize((50, 50))
    IMAGE_CONFIG = ImageTk.PhotoImage(ImgConfig_sized)

    btnConfig = Button(frmConfig, image= IMAGE_CONFIG, command = ConfigWindow)
    btnConfig.grid(column=0, row=0, padx=10, pady=0)

```

```

#Boton de informacion
objInfo_Img = Image.open(LOCATION + 'icons8-info-squared-100.png')
ImgInfo_sized = objInfo_Img.resize((50, 50))
IMAGE_INFO = ImageTk.PhotoImage(ImgInfo_sized)

btnConfig = Button(frmConfig, image= IMAGE_INFO, command = InfoWindow)
btnConfig.grid(column=1, row=0, padx=10, pady=0)

#Boton de conectar
btnConnect = Button(frmConfig, image= IMAGE_DISCONNECT, command =
toggle_connect)
btnConnect.grid(column=2, row=0, padx=10, pady=0)

#Indicador de conectado
def con():
    if MODBUS_READY == True:
        icon_conexion = Image.open(LOCATION + 'icons8-connection-status-on-100.png')
    else:
        icon_conexion = Image.open(LOCATION + 'icons8-connection-status-off-100.png')
    icon_connected_sized = icon_conexion.resize((30, 30))
    obj_icon_connected = ImageTk.PhotoImage(icon_connected_sized)
    obj_icon_conn = tkinter.Label(frmConfig, image=obj_icon_connected)
    obj_icon_conn.image = obj_icon_connected
    obj_icon_conn.grid(column=3, row=0, padx=0, pady=0, sticky=tk.NW)
    frmConfig.after(1500, con)

#####
#####
#Funciones lazo
#####
#####

con()
mbRead()
mbWrite()
change_status()

ventanaPrincipal.mainloop()

```

7.1.4.1 Imágenes para aplicación SCADA.

Para poder realizar el programa se tiene que tener los recursos para la visualización del sistema SCADA se presenta las imágenes utilizadas.



Ilustración 104 iconos ocupadas en la interface gráfica.

7.1.4.2. Mapa de Modbus del LOGO!

Logo 8 Modbus-Implementierungstabelle										Modbus-Startadresse 0/1			0		
Coil	F01 Read Coils F05 Write Single Coil F15 Write Multiple Coils			F01 Read Coils F05 Write Single Coil F15 Write Multiple Coils			F02 Read Discrete Input			F03 Read Holding Register F06 Write Single Register F16 Write Multiple Register			F04 Read Input Register		
	Logo	Coil	Logo	Logo	Coil	Logo	Discrete Input	Logo	Register	Register	Logo	Input Register	Input Register	Input Register	
0	V0.0	8192	Q1	8256	M1	0	I1	0	VW0	0	VW0	0	AI1		
1	V0.1	8193	Q2	8257	M2	1	I2	1	VW2	1	VW2	1	AI2		
2	V0.2	8194	Q3	8258	M3	2	I3	2	VW4	2	VW4	2	AI3		
3	V0.3	8195	Q4	8259	M4	3	I4	...	...	...	...	...	AI4		
4	V0.4	8196	Q5	8260	M5	4	I5	422	VW844	4	VW844	4	AI5		
5	V0.5	8197	Q6	8261	M6	5	I6	423	VW846	5	VW846	5	AI6		
6	V0.6	8198	Q7	8262	M7	6	I7	424	VW848	6	VW848	6	AI7		
7	V0.7	8199	Q8	8263	M8	7	I8	425	???	7	???	7	AI8		
...	...	8200	Q9	...	...	8	I9	...	...	8	???	8	???		
6800	V850.0	8201	Q10	8312	M57	9	I10	511	???	...	???	...	???		
6801	V850.1	8202	Q11	8313	M58	10	I11	512	AQ1	15	AQ1	15	???		
6802	V850.2	8203	Q12	8314	M59	11	I12	513	AQ2		AQ2		???		
6803	V850.3	8204	Q13	8315	M60	12	I13	514	AQ3		AQ3		???		
6804	V850.4	8205	Q14	8316	M61	13	I14	515	AQ4		AQ4		???		
6805	V850.5	8206	Q15	8317	M62	14	I15	516	AQ5		AQ5		???		
6806	V850.6	8207	Q16	8318	M63	15	I16	517	AQ6		AQ6		???		
6807	V850.7	8208	Q17	8319	M64	16	I17	518	AQ7		AQ7		???		
6808	???	8209	Q18	8320	???	17	I18	519	AQ8		AQ8		???		
...	...	8210	Q19	...	...	18	I19	520	???		???		???		
8191	???	8211	Q20	8367	???	19	I20	...	...		???		???		
		8212	???			20	I21	527	???		???		???		
		...	...			21	I22	528	AM1		AM1		???		
		8255	???			22	I23	529	AM2		AM2		???		
						23	I24	530	AM3		AM3		???		
						24	???	...	...		...		???		
						...	...	...	...		...		???		
						63	???	591	AM62		AM62		???		
									AM63		AM63		???		
									AM64		AM64		???		