

REDES DE COMUNICACIÓN INDUSTRIAL. PRÁCTICAS DE LABORATORIO

**OSCAR MAURICIO RAMIREZ RONDON
JOSÉ IGNACIO SÁNCHEZ ARENAS**

**UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FISICO MECANICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y
TELECOMUNICACIONES
BUCARAMANGA
2011**

REDES DE COMUNICACIÓN INDUSTRIAL. PRÁCTICAS DE LABORATORIO

**OSCAR MAURICIO RAMIREZ RONDON
JOSE IGNACIO SANCHEZ ARENAS**

**TRABAJO DE INVESTIGACIÓN:
PARA OPTAR AL TÍTULO DE INGENIERO ELECTRÓNICO**

**DIRECTOR:
MSC. JULIO AUGUSTO GELVEZ FIGUEREDO**

**UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FISICO MECANICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y
TELECOMUNICACIONES
BUCARAMANGA
2011**

DEDICATORIA

A Dios, por haberme ayudado a acertar cuando escogí universidad, carrera y proyecto de grado: estas tres decisiones me dotaron de una perseverancia completamente desconocida en mi vida “pre-UIS”... de esas que no para hasta lograr el objetivo y que habría sido imposible de lograr si alguna de esas tres decisiones hubiera sido diferente.

A mis abuelos por haberme regalado los excelentes padres que tengo y por transferirme parte de su sabiduría a través de mis padres.

A mi padre por haber tenido hacia mí la autoridad que evito que me convirtiera en la vergüenza de la familia (delincuente o vago, etc). A mi madre por haber tenido la inteligencia emocional que evito que me convirtiera en el “témpano de hielo” carente de cualquier gesto de humanidad que habría podido ser. En general a los dos, por haberme convertido en lo mejor de cada uno de ellos, por los esfuerzos que hicieron para que yo pudiera dedicarme exclusivamente a estudiar y por la paciencia formidable que tuvieron con la espera de mí grado.

A mis hermanas, que me han ayudado a entender la sabiduría de las decisiones que mis padres tomaron en todo lo que me afectaba directa o indirectamente y que cuando tenía la edad que ahora ellas tienen calificaba de injustas por no dimensionar la totalidad de las consecuencias de que ellos hubiesen obrado como yo, en ese entonces, hubiera querido.

A mi bella hija del sol (Barrancabermeja) porque la quiero mucho más que a la ciudad en que nací –hasta el punto contestar que soy de Barranca cuando me preguntan de dónde soy-. A ti mi Barranquita del alma por las oportunidades que mis padres, mis hermanas y yo no habríamos tenido en otra parte y porque muchas de las razones que tengo para sentirme orgulloso de mi mismo te las debo a vos.

A mis amigos por aceptarme como soy, por la ayuda que me han brindado cuando la he necesitado, pero sobre todo por el ánimo a alcanzar mis metas (algunos con valiosas críticas constructivas y otros con su igualmente valioso ejemplo).

A mis profesores de bachillerato (Silvia Martínez, Félix García, Jaime Rodríguez, Mercedes de Palmera, Alexander Aguirre, los hermanos Heriberto y Antonio Ortegón, Ricardo Maffiold) porque en equipo con mis padres contribuyeron positivamente a la formación de mi personalidad y al logro de un primer requisito para ser profesional: ser bachiller.

A mis profesores de Universidad (Alberto Higuera, Juan Arteaga, Rosa Claudia, Alfredo Acevedo, Francisco Rueda, Jaime Barrero, Julio Gelvez y Herman Vargas) por hacer de mí un buen ingeniero, y más importante aun: con todo el potencial para convertirme en un excelente ingeniero a medida que la experiencia lo permita.

Y en general a todas aquellas personas que quise, a las que quiero y, hasta donde la memoria me alcanza, he querido las mismas a las que el cada vez creciente tiempo que la carrera me exigía me impidió demostrarles ese afecto como siempre he querido hacerlo.

OSCAR MAURICIO RAMIREZ RONDON

DEDICATORIA

Este trabajo es dedicado a la vida que me permite seguir en pie para afrontar nuevos retos y a toda mi familia. Este logro es un acto de fe de mi madre con raíces ancestrales donde se tejieron las ilusiones para mi vida. Dedico este triunfo especialmente. A mi padre quien me mostró que a través del trabajo y la honestidad se puede conseguir las metas, a mi hermana Dora Clemencia quien hizo realidad los deseos de mi madre aceptando con agrado la labor de ayudar incondicionalmente a sus hermanos, creyendo siempre en un mejor futuro para ellos, a mi hermana Adriana Patricia y a José Antonio Cárdenas que estuvieron cerca en todo momento para seguir creyendo que si se podía, de igual manera dedico este éxito a mi hermano Mauricio que estuvo en los momentos finales de mi carrera; agradezco inmensamente la ayuda desinteresada y cariño que me brindó para culminar esta meta.

Finalmente dedico este triunfo a mi novia y a mis amigos de universidad, que en los momentos donde el sol no se veía estuvieron para darme el apoyo necesario para afrontar las dificultades propias de nuestra vida universitaria.

JOSÉ IGNACIO SÁNCHEZ ARENAS

AGRADECIMIENTOS

Los autores expresan sus agradecimientos a:

El director (Julio A. Gélvez F.) de este proyecto de grado, por sus valiosos aportes y por estar siempre dispuesto a ayudar a los alumnos que quieren aprender y a no ayudar a los que no quieren aprender.

José Rúgeles por haber sacado a uno de los autores de este proyecto de grado de un proyecto anterior codirigido por él, lo que le permitió a dicho autor sumergirse en el fascinante mundo de la automatización industrial. Se le agradece por haber brindado la oportunidad de convertir un problema en una excelente oportunidad: no se alcanza a imaginar el favor tan grande que hizo.

Rexnik Galeano, por contactarnos con el soporte técnico de Schneider Electric para Latinoamérica.

Diego Sánchez de Schneider Electric por su valiosa ayuda en lo que respecta a los protocolos modbus serial y CANopen.

A Julián Arzayus de Smart ingeniero, por sus oportunas sugerencias y su disposición a ayudar.

A José Antonio Cárdenas, Roger Delgado y María Carolina Serrato, por la ayuda y el apoyo brindado en la ciudad de Bogotá con el fin de orientar nuestro proyecto de grado.

CONTENIDO

	Pág
INTRODUCCIÓN	28
1 COCEPTOS BASICOS	29
1.1 DEFINICION DE PROTOCOLO DE COMUNICACIÓN INDUSTRIAL.	30
1.2. RED	30
1.2.1. Tipos de redes	30
1.2.1.1. Red de área local (Local Area Network, LAN):	30
1.2.1.2. Red de área campus (Campus Area Network, CAN	31
1.2.1.3. Red de área metropolitana (Metropolitan Area Network, MAN):	31
1.2.1.4. Red de área extensa (Wide Area Network WAN):	31
1.3. REDES DE COMUNICACIÓN INDUSTRIAL	31
1.3.1. Bus de campo	32
1.3.2. Topología de las redes de comunicación industrial	32
1.3.2.1. Red en bus	33
1.3.2.2. Red en anillo:	33
1.3.2.3. Red en estrella	34
1.3.3. Dispositivos de redes de comunicación industrial:	34
1.3.3.1. Repetidor:	34
1.3.3.2. Hub (concentrador	35
1.3.3.3. Bridge (Puente):	35
1.3.3.4. Switch (Conmutador):	35
1.3.3.5. Router (Encaminador o enrutador):	36
1.4. PROCESO DE TRANSMISIÓN DE DATOS.	36
1.4.1 Elementos en la transmisión de datos	36
1.4.2. Técnicas de transmisión de datos	37

1.4.2.1. Transmisión serie	37
1.4.2.2. Transmisión en paralelo	37
1.4.3. Tipos de transmisión de datos.	38
1.4.3.1. Transmisión asíncrona	38
1.4.3.2. Transmisión síncrona:	38
1.4.3.3. Definición de baudio	39
1.4.4. Modos de transmisión de datos:	39
1.4.4.1. Transmisión simple (SIMPLEX):	39
1.4.4.2. Transmisión semidoble (HALF-DUPLEX):	39
1.4.4.3. Transmisión doble (FULL-DUPLEX):	40
1.5 MODELOS DE REFERENCIA OSI:	41
1.5.1. Características del modelo OSI	42
1.5.2. Capas del modelo OSI	42
1.5.2.1. Capa física:	42
1.5.2.2. Capa de enlace	43
1.5.2.3. Capa de red	43
1.5.2.4. Capa de transporte	43
1.5.2.5. Capa de sesión	43
1.5.2.6. Capa de presentación	44
1.5.2.7. Capa de aplicación	45
1.6. MODELO TCP/IP.	45
1.6 1. Nivel de aplicación:	45
1.6.2. Nivel de transporte:	46
1.6.3. Nivel de Internet	46
1.6.4. Nivel de interfaz de red:	46
1.7. PIRAMIDE CIM (COMPUTER INTEGRATED MANUFACTURING)	47
1.7.1. Nivel de Proceso	48
1.7.2. Nivel de Campo:	48
1.7.3. Nivel de Célula:	49
1.7.4. Nivel de Planta	49
1.7.5. Nivel de Factoría	49
2. PROCOLO DE COMUNICACIÓN AS-Interface (AS-i)	50

2.1. HISTORIA DE AS-i	50
2.1.1. Versión 2.04 (1994):	50
2.1.2. Versión 2.1 (2002):	51
2.1.2.1. Estándar (perfil MO):	51
2.1.2.2. Ampliado (perfil M1):	51
2.1.3. Versión 3.0 (2004):	51
2.2. CARACTERISTICAS DE LA RED AS-i	52
2.3. CODIFICACION DE LA SEÑAL	53
2.4. ESTRUCTURA DE UNA TRAMA AS-i	54
2.5 TIPOS DE TRANSACCIONES	55
2.5.1 Transacciones simples:	55
2.5.2 Transacciones Combinadas	56
2.5.2.1 Tipo I	56
2.5.2.2 Tipo II	57
2.5.2.3 Tipo III:	58
2.5.2.4 Tipo IV:	58
2.5.2.5 Tipo V	58
2.6 COMPONENTES DE LA RED AS-I	58
2.6.1 Maestro AS-I con Referencia TWDNOI10M3	59
2.6.2 Esclavos	61
2.6.2.1 Esclavo convencional:	62
2.6.2.2 Esclavo AS-i:	63
2.6.2.3 Perfil de los esclavos AS-i	64
2.6.3 Cable AS-i	65
2.6.4 Fuente de alimentación AS-i.	67
2.7 COMPONENTES ADICIONALES	68
2.7.1 Repetidores	68

2.7.2 Pasarelas (Gateway):	69
2.7.3 Dispositivo direccionador	70
2.8 FUNCIONAMIENTO DE LA RED AS-INTERFACE	71
2.9 PASOS PARA LA TRANSMISIÓN DE LOS BITS	71
2.10 HERRAMIENTAS DE DIAGNOSTICO	72
2.11 ARQUITECTURA DE LAS REDES AS-I:	72
2.11.1 Módulos de ampliación:	73
2.11.2 Cantidad de esclavos	73
2.12 PROGRAMACIÓN REDES AS-I	74
2.12.1 Descripción del módulo maestro AS-Interface TWDNOI10M3	74
2.12.1.1 Componentes del maestro AS-i	74
2.12.1.2 panel de visualización	76
2.12.1.3 modos de funcionamiento del maestro AS-interface	77
2.13 PROGRAMACIÓN DE LAS REDES AS-I CON TWIDOSUITE	79
2.13.1 Estructura del maestro AS-Interfaces:	79
2.13.2 Estructura de los equipos esclavos	80
2.13.3 Pasos para instalar AS-Interface	82
2.13.4 Descripción de la ventana AS-interface en modo offline	83
2.13.5 Descripción de la ventana AS-interface en modo online	84
2.13.6 Descripción de errores:	86
2.13.6.1 Error de perfil:	86
2.13.6.2 Esclavo no proyectado	86
2.13.6.3 Esclavo no detectado:	86
2.13.6.4 Direccionamiento de las entradas y salidas de los esclavos:	87
3. PROTOCOLO DE COMUNICACIÓN MODBUS SERIAL	88
3.1 CARACTERISTICAS DE MODBUS SERIAL	89

3.1.1 Topología:	89
3.1.2 Modo de transmisión	90
3.1.3 Tipo de comunicación empleada	90
3.2 ACCESO AL MEDIO	90
3.2.1 Consulta/Respuesta	90
3.2.2 Difusión	91
3.3 CAPA DE APLICACIÓN MODBUS	91
3.4 CAPA FISICA	92
3.4.1 Configuración de dos hilos	93
3.5 MODOS DE TRANSMISIÓN	95
3.5.1 Modo RTU	95
3.5.2 Modo ASCII	98
3.6 TRANSACCIONES EN MODBUS:	101
3.6.1 Consulta	102
3.6.2 Respuesta:	102
3.7 FUNCIONES MODBUS:	102
3.7.1 Funciones de acceso de bits	104
3.7.2 Funciones de acceso de palabra	104
3.7.3 Funciones de diagnostico y configuración	104
3.8 COMUNICACIÓN ENTRE EL SOFTWARE TWIDOSUITE Y PLC TWIDO	104
3.9 CONFIGURACIÓN CAPA FÍSICA	105
3.9.1 Conexión entre el maestro Twido y los esclavos de la E3T	106
3.9.1.1 Conexión Twido y Twido	106
3.9.1.2 Conexión TwidoTWDLCAE40DRF y Modicon M340	108
3.9.1.3 Conexión TwidoTWDLCAE40DRF y variador de velocidad Altivar31	109
3.9.1.4 Conexión Twido TWDLCAE40DRF y PLC Zelio:	110
3.9.2 Conexión Modicon M340 y variador de velocidad Altivar 31:	111
3.9.3 Conexión Modicon M340 y zelio	111
3.10 COMUNICACIÓN MODBUS CON EL PLC TWIDO TWDLCAE40DRF	112

3.10.1 Configuración del puerto principal con Twidosuite:	112
3.10.2 Crear en el programa de twidosuite una tabla de envío/recepción por cada instrucción EXCHx utilizada:	113
3.10.2.1 Protocolo Modbus en el maestro	113
3.10.2.2 Intercambio de mensajes en twidosuite:	116
3.10.2.3 Errores y condiciones de funcionamiento	117
3.10.2.4 Solicitudes o funciones estándar Modbus	117
3.11 COMUNICACIÓN CON EL PLC M340	125
3.11.1 Descripción modulo maestro M340	125
3.11.2 Programación del PLC M340 mediante unity pro, con modbus	126
3.11.2.1 Definir el hardware en el software, para modbus	126
3.11.2.2 Configurar el puerto modbus en el software unity pro	127
3.11.2.3 programar los bloques de comunicación	127
3.11.2.4 elaborar un programa para el control de la red	127
3.12. COMUNICACIÓN CON EL PLC ESCLAVO ZELIO “REF SR3 B101BD”:	127
3.12.1 Descripción del relé programable zelio	127
3.12.2 Configuraciones del PLC zelio	128
3.13. COMUNICACIÓN CON EL VARIADOR DE VELOCIDAD ALTIVAR 31	129
3.13.1 descripción del variador de velocidad altivar 31	130
3.13.2 configuración del variador de velocidad altivar 31	130
4. PROTOCOLO DE COMUNICACIÓN MODBUS TCP/IP	130
4.1 PROTOCOLO TCP/IP	131
4.1.1 TCP (Transmission Control Protocol):	131
4.1.2 IP (Internet Protocol):	132
4.1.2.1. Dirección de Internet (IP Address):	135
4.1.2.2 Máscaras de subred y subdireccionamiento	137

4.2	MODELO CLIENTE-SERVIDOR DE MODBUS TCP/IP	139
4.2.1	Solicitud modbus:	139
4.2.2	Indicación modbus:	139
4.2.3	Respuesta modbus	139
4.2.4	Confirmación modbus	140
4.3.	ESTRUCTURA MENSAJE MODBUS:	140
4.4	TRAMA ADU EN MODBUS TCP/IP	141
4.4.1	Cabecera MBAP	142
4.4.1.1	transaction identifier (2 bytes):	142
4.4.1.2	Protocol Identifier (2 bytes):	142
4.4.1.3	Length (2 bytes):	142
4.4.1.4	Unit Identifier (1 byte):	143
4.5	PROTOCOLO ETHERNET	143
4.6	COMUNICACIÓN ENTRE EL SOFTWARE TWIDOSUITE Y PLC TWIDO	145
4.7.1	Configuración del puerto Ethernet con el software Twidosuite:	146
4.7.2	Crear en la aplicación una tabla de envío/recepción que será utilizada por la instrucción EXCHx:	147
4.7.2.1	Protocolo Modbus TCP/IP en el cliente	147
4.7.2.2	Intercambio de mensajes modbus TCP/IP en Twidosuite	148
4.7.2.3	Errores y condiciones de funcionamiento para modbus TCP/IP	148
4.8	COMUNICACIÓN MODBUS TCP/IP CON EL PLC M340	149
4.8.1	Descripción puerto Ethernet del CPU Modicon M340 BMX P34 2020:	149
4.8.2	programación del PLC M340 mediante unity pro, con modbus TCP/IP	150
4.8.2.1	Definir el hardware en el software, para modbus TCP/IP:	150
4.8.2.3	Configurar el puerto Ethernet en el software unity pro	151
4.8.2.4	configurar los bloques ADDM, READ_VAR, WRITE_VAR, en TCP/IP	151
4.8.2.5	Elaborar un programa para el control de la red:	155
5	CANopen	156

5.1 QUE ES CANopen:	156
5.2 CARACTERÍSTICAS GENERALES DE CANopen	156
5.3 DESCRIPCION DE CAN	157
5.4 ARQUITECTURA DEL PROTOCOLO CAN	158
5.5 CAPA FISICA:	159
5.6 CAPA DE ENLACE	161
5.6.1 Principios de arbitraje del bus CAN:	162
5.6.2 Tipo De Tramas:	162
5.6.2.1 Trama De Datos	162
5.6.2.2 Trama Remota:	164
5.6.2.3 Trama de Error	165
5.6.2.4 Trama De Sobrecarga	165
5.6.2.5 Espacio Inter-Tramas	166
5.6.3. Codificación de los bits	167
5.6.4 Tipos de errores	168
5.6.4.1 Bit error:	168
5.6.4.2 Stuff error:	168
5.6.4.3 CRC error:	168
5.6.4.4 Form error:	168
5.6.4.5 Acknowledgement error:	169
5.6.5 Confinamiento de fallas	169
5.6.5.1 Error Active:	169
5.6.5.2 Error Passive:	169
5.6.5.3 Bus Off	170
5.7 CAPA DE APLICACIÓN DE CANopen:	170
5.8 MODELO DE OBJETO:	171
5.9 ESTRUCTURA DE LA COMUNICACION:	174
5.9.1 Especificación de mensajes CAN (CMS):	174
5.9.2 Administración de red (NMT):	175
5.9.3 Administración de capas (LMT):	176

5.9.4 Asignación de identificadores de mensaje (DBT):	176
5.9.5. Identificadores de mensaje “COB-ID”	176
5.10 MODELO DE COMUNICACION:	178
5.10.1 Modelo Maestro/Esclavo:	179
5.10.2 Modelo Cliente/Servidor:	180
5.10.3 Modelo productor/consumidor	181
5.11 OBJETO DE PROCESO DE DATOS (PDO):	181
5.11.1 Transmisión de PDOs	184
5.11.1.1 Transmisión asíncrona:	184
5.11.2 Transmisión sincrónica	185
5.12 OBJETO DE SERVICIO DE DATOS (SDO):	186
5.12.1 Transferencia expédita:	187
5.12.2 Transferencia en segmentos	187
5.12.3 Abort Domain Transfer:	187
5.12.4 Estructura de los SDOs	187
5.13 MENSAJES ADICIONALES:	189
5.13.1 Mensaje de sincronización (SYNC	189
5.13.2 Mensaje de etiqueta (TIME-STAMP):	189
5.13.3 Mensajes de Emergencia (EMCY):	190
5.14 COMUNICACIÓN CON EL PLC TWIDO:	190
5.14.2 Conexión Módulo maestro TWDNCO1M con la red CANopen	192
5.14.3 Administración y monitoreo de la red CANopen:	193
5.14.3.1 Restablecer la aplicación:	195
5.14.3.2 Restablecer la comunicación:	195
5.14.3.3 Inicio	195
5.14.3.4 Pre-Operativo:	196
5.14.3.5 Detenido	196
5.14.3.6 Operacional:	196
5.15 ELEMENTOS UTILIZADOS POR EL MÓDULO MAESTRO TWDNCO1M	196
5.15.1 Archivo EDS (Hoja de datos electrónica	196
5.15.2 Perfiles CANopen	197

5.15.3	Envío de PDOs a través del módulo maestro TWDNCO1M	197
5.15.4	Envío de SDOs a través del módulo maestro TWDNCO1M	199
5.15.5	Monitoreo del dispositivo	199
5.15.5.1	NMT Node Guarding	199
5.15.5.2	Heartbeat	201
5.16	CONFIGURACION DE CANopen CON EL SOFTWARE TWIDOSUITE.	202
5.16.1	Descripción general	202
5.16.1.1	Configuración:	202
5.16.1.2	Declaración del maestro CANopen	203
5.16.2	Herramientas de configuración Twidosuite para CANopen:	204
5.17.1	Importación de perfiles de esclavos:	207
5.17.2	Creación de la red CANopen:	207
5.17.3	Configuración de los parámetros de administración de red:	207
	CONCLUSIONES	209
	RECOMENDACIONES	214
	BIBLIOGRAFIA	218
	ANEXOS	222

LISTA DE FIGURAS

	Pág
Figura 1. Red en bus.	32
Figura 2. Red en anillo.	33
Figura 3. Red en estrella.	34
Figura 4. Elementos de un sistema de comunicación.	37
Figura 5. Transmisión simple.	40
Figura 6. Transmisión semidoble	41
Figura 7. Transmisión doble.	41
Figura 8. Capas del modelo OSI.	44
Figura 9. Cabeceras de tramas de datos en TCP/IP.	46
Figura 10. Modelos TCP/IP y OSI.	47
Figura 11. Pirámide CIM	48
Figura 12: Formas de onda de la señal AS-i	53
Figura 13: Estructura de los mensajes AS-i	54
Figura 14: Módulo maestro TWDNOI10M3	59
Figura 15: Botonera XAL-S2002E.	62
Figura 16: Relé ASI20MT414OR	62
Figura 17: Cable as-i	65
Figura 18: Fuente AS-i	68
Figura 19: Gateway ASI-MBG-SS-BW1641 Turck	69
Figura 20: Direccionador AS-Interface	70
Figura 21: Componentes maestro AS-i	75
Figura 22: Panel de visualización	76
Figura 23: Visualización de los modos de funcionamiento del maestro AS-i	78
Figura 24: Arquitectura del módulo TWDNOI10M3.	80
Figura 25: Estructura de un Esclavo de direccionamiento ampliado:	81
Figura 26: ventana AS-i en modo offline	83
Figura 27: Ventana AS-interface en modo online	85
Figura 28: Como se direccional las entradas y salidas de los esclavos AS-i	87

Figura 29: Capa de aplicación de modbus	91
Figura 30: Tipos de conexión maestro y esclavos en la red modbus	92
Figura 31: Topología a dos hilos	94
Figura 32: formato de Secuencia de bits en modo RTU.	95
Figura 33: Descripción de la trama modbus en modo RTU	96
Figura 34: Trama con principio y final en modo RTU:	97
Figura 35: Tiempos entre tramas	97
Figura 36: Tiempo entre secuencias de bits	98
Figura 37: Formato de secuencia de bits en modo ASCII	100
Figura 38: Trama modbus en modo ASCII	101
Figura 39: Conexión con cable USB	105
Figura 40: Modicon M340	105
Figura 41: PLC Zelio	105
Figura 42: Altivar 31	106
Figura 43: Twido TWDLCAE40	106
Figura 44: Conector mini DIN	107
Figura 46: Conexión maestro Twido y esclavos Twido	107
Figura 47: Descripción puerto serie del PLC Modicon M340	108
Figura 48: Conexión PLC Twido y M340	109
Figura 49: Pines conector RJ 45 del variador de velocidad Altivar 31	109
Figura 50: Conexión PLC Twido TWDLCAE40DRF y variador Altivar 31	110
Figura 51: Conexión Twido y Zelio	110
Figura 52: Conexión Modicon M340 y variador de velocidad Altivar 31	111
Figura 53: Conexión Modicon M340 y Zelio	111
Figura 54: BIT %MSGx.D	116
Figura 55: CPU Modicon M340 BMX P340 2020	126
Figura 56: Descripción PLC zelio:	128
Figura 57: Descripción del Altivar 31	129
Figura 58: Estructura del mensaje IP	132
Figura 59: Formato cabecera IP (versión 4)	135
Figura 60: Clases de direcciones IP	136

Figura 61: Modelo cliente/servidor:	139
Figura 62: Trama modbus	140
Figura 63: Trama ADU para modbus TCP/IP	141
Figura 64: Cabecera MBAP	142
Figura 65: Trama IEEE 802.3	145
Figura 66: Puerto Ethernet:	145
Figura 67: BIT %MSGx.D	148
Figura 68: Puerto Ethernet:	150
Figura 69: Arquitectura del protocolo CAN	158
Figura 70: Niveles de tensión:	159
Figura 71: velocidad de transmisión vs longitud de la red	160
Figura 72: Topología del bus en la capa física	161
Figura 73: Estados lógicos de la red CAN	161
Figura 74: Trama De Datos:	164
Figura 75: Formato De un espacio intertrama	167
Figura 76: Codificación de bits en CAN	167
Figura 77: Modelo de un dispositivo CANopen	173
Figura 78: Estructura del identificador de mensajes CANopen:	177
Figura 79: Modelo Maestro esclavo con comunicación no confirmada	179
Figura 80: Modelo Maestro esclavo con comunicación confirmada	179
Figura 81: Modelo cliente/servidor	180
Figura 82: Modelo push:	180
Figura 83: Modelo Pull:	181
Figura 84: mensaje de sincronización (SYNC)	190
Figura 85: Módulo maestro CANopen TWDNCO1M.	191
Figura 86: Conexión modulo maestro TWDNCOM con esclavo altivar 31	193
Figura 87: Máquina de estados para dispositivo NMT esclavo	194
Figura 88: Intercambio de PDOs con el modulo maestro TWDNCOM	198
Figura 89: Trama RTR que el NMT maestro envía a los NMT esclavos:	200
Figura 90: Mensaje que los NMT esclavos envían al NMT maestro	200
Figura 91: Estructura de un mensaje de Heartbeat	201

LISTA DE TABLAS

	Pág
Tabla 1 Características red AS-i	52
Tabla 2: Significado de cada bit	55
Tabla 3: Tipos de transacciones AS-i simples	56
Tabla 4: Transacciones combinadas:	57
Tabla 5: Perfiles del maestro	60
Tabla 6: Códigos I/O	64
Tabla 7: especificaciones cable AS-i:	66
Tabla 8: Elementos del el modulo AS-i	75
Tabla 9: Descripción de los indicadores luminosos de estado	77
Tabla 10: Descripción y configuración de los elementos y parámetros	79
Tabla 11: Descripción y configuración de los elementos y parámetros	81
Tabla 12: Resumen de pasos para instalar AS-interface	82
Tabla 13: Características de la ventana AS-i en modo OFFLINE	84
Tabla 14: características de la ventana AS-i en modo online	85
Tabla 15: Direccionamiento de las entradas y salidas de los esclavos AS-i	87
Tabla 16: configuración a dos hilos	94
Tabla 17 caracteres AS-I	99
Tabla 18: principales funciones de modbus	103
Tabla 19: Tabla de control, envío y recepción.	114
Tabla 20: Descripción palabras de sistema y código de error EXCHx	117
Tabla 21: Lectura de N bits	118
Tabla 22: Lectura de N palabras	119
Tabla 23: Escritura de un bit	120
Tabla 24: Escritura de una palabra	120
Tabla 26: Escritura de N palabras	122
Tabla 27: Descripción de la CPU Modicon M340 BMX P340 2020	125
Tabla 28: Direcciones IP	137
Tabla 29: Código error EXCHx para modbus TCP/IP:	149

Tabla 30: Descripción de los parámetros de la función ADDM	152
Tabla 31 Descripción de los parámetros de la función READ_VAR	152
Tabla 32: Descripción de los parámetros de la función WRITE_VAR	154
Tabla 34: Ejemplo de variable compleja:	173
Tabla 35: Distribución de los COB-IDs en CAL	175
Tabla 36: Distribución general de los identificadores en CANopen	177
Tabla 37: Estructura de los PDOs communication parameter	183
Tabla 38 Estructura de los PDOs Mapping Parameter:	184
Tabla 39: tipo de transmisión de PDOs:	185
Tabla 40: Estructura de parámetros Client / Server SDO:	188
Tabla 41: Velocidad en baudios frente a longitud del cable:	192
Tabla 43: Estado de un dispositivo esclavo NTM	195
Tabla 44: Valor del campo state en un mensaje de NMT Node Guarding:	201
Tabla 45: Valor del campo state en un mensaje de Heartbeat	202
Tabla 46: Pasos con Twidosuite para configurar CANopen:	203
Tabla 47: Pasos para la declaración del maestro CANopen en Twidosuite.	204
Tabla 48: Herramientas de configuración CANopen:	204
Tabla 49: Importación de archivos EDS	208

LISTA DE ANEXOS

	Pág
ANEXO 1: AS-i	222
ANEXO 2: MODBUS SERIAL	279
ANEXO 3: MODBUS TCP/IP	335
ANEXO 4: CANopen	415
ANEXO 5: PROCESO INDUSTRIAL	452

RESUMEN

TITULO: REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO*.

AUTORES: OSCAR M. RAMIREZ RONDON Y JOSE I. SANCHEZ ARENAS**.

PALABRAS CLAVE: protocolos, comunicación, industrial, AS-Interface, Modbus, TCP/IP, CANopen.

DESCRIPCION DEL CONTENIDO:

Este trabajo de grado ayuda a asimilar la teoría de cuatro protocolos de comunicación industrial (AS-Interface, Modbus, Modbus TCP/IP y CANopen) y su implementación en los equipos Telemecanique que tiene la E3T-UIS, a través de cuatro materiales didácticos que son:

Este libro que consta de cinco capítulos teóricos (el primero es de conceptos básicos y cada uno de los otros cuatro capítulos para un protocolo de comunicación industrial de los ya mencionados).

Una guía práctica o manual por cada uno de los cuatro protocolos de comunicación industrial, que explica los diferentes pasos y requisitos para la configuración de dicho protocolo de comunicación según las características de los equipos que compro la E3T. Cada uno de estos manuales constituye uno de los cuatro primeros anexos de este libro.

Tres tableros (o mímicos) implementados en la HMI Magelis mediante el software Vijeo Designer, para simular 3 procesos automatizados típicos en una planta de embotellado, donde se implementan los protocolos de comunicación con sus respectivas redes usando los PLCs de la E3T. Estos tableros se incluyen en el anexo 5 de este libro.

Cinco prácticas de laboratorio: las cuatro primeras son cada una aplicación de uno de los cuatro protocolos de comunicación industrial y parte de los procesos de la planta de embotellado; la última práctica consiste en la interconexión de las redes sugeridas en las cuatro primeras prácticas. Estas prácticas de laboratorio forman parte del quinto y último anexo de este libro.

Trabajo de Grado.

Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones Director: Julio Augusto Gelvez Figueredo.

ABSTRACT

TITLE: INDUSTRIAL COMMUNICATION NETWORKS. LABORATORY PRACTICES*.

AUTHORS: OSCAR M. RAMIREZ RONDON Y JOSE I. SANCHEZ ARENAS**

KEY WORDS: protocols, communication, industrial, AS-Interface, Modbus, TCP/IP, CANopen.

CONTENT DESCRIPTION:

This degree work helps to learn the four industrial communication protocols (AS-Interface, Modbus, Modbus TCP/IP y CANopen) theory and its implementation in the Telemecanique hardware which E3T-UIS owns, through four didactic material; they are:

This book which consists of five theoretical chapters (the first one is about basic concepts and each of the four remaining chapters for one of the industrial communication protocols aforementioned).

A practice guide or handbook for each of the four industrial communication protocols, to explain the different this communication protocol configuration steps and requirements according to the hardware purchased by the E3T. Each of this handbooks is this book's one of the first four annexes.

Three panels (or mimics) implemented in the HMI Magelis through Vijeo Designer software, to simulate three typical automated processes in a bottling plant, where the communication protocols with its respective networks are implemented using the E3T's PLC. These panels are included in annex five.

Five laboratory practices: each of the first four is a application of one of the four communication protocols and part of the bottling plant processes; the last practice is about the interconnection of the networks suggested in the first four practices. These laboratory practices are part of this book's fifth and last annex.

Degree Work

Physical-Mechanical Engineering Faculty. Electrical, Electronic and Telecommunications Engineering School. Director: Julio Augusto Gelvez Figueredo.

INTRODUCCIÓN

Telemecanique (marca del grupo Schneider Electric) es un fabricante de soluciones de automatización industrial que ofrece una completa línea de productos, que van desde variadores de velocidad para motores hasta interfaces hombre-máquina (HMI = Human Machine Interface), pasando desde luego por autómatas programables.

La gama de PLCs inicia en el que Telemecanique denomina “Rele Programable Zelio”, que es el menos potente de los que ofrecen pero el más económico; este autómata tiene un número reducido de entradas-salidas (10 = 6 entradas [de las cuales 4 son analógicas] + 4 salidas), aunque pueden expandirse mediante un modulo a 14 entradas-salidas mas (8 entradas + 6 salidas), para un total de 24 entradas-salidas; solo puede funcionar como esclavo modbus o como servidor modbus TCP/IP; puede programarse manualmente a través de una pantalla y unos botones que trae en su parte delantera o usando un programa gratuito que corre en Windows XP denominado “Zelio soft”. El fabricante provee un cable para transferir programas de usuario entre el PC y el PLC y viceversa.

El siguiente miembro de la familia de PLCs de Telemecanique es el Twido, más poderoso que el Zelio toda vez que cuenta con 40 entradas-salidas (24 entradas y 16 salidas), expandibles a través de 7 módulos a 224 salidas mas (32 por modulo) o a 224 entradas mas (32 por modulo), para un total de 248 entradas (24 + 224), 240 salidas (16 + 224) o 264 entradas-salidas (40 + 224). A diferencia del Zelio, el Twido no solo puede ser esclavo modbus sino también maestro modbus; trae un puerto RJ45 hembra mediante el cual puede actuar como cliente y/o servidor Modbus TCP/IP. Además trae un bus interno mediante el cual puede brindar comunicación y alimentación a módulos maestros de protocolos AS-interface y CANopen. Su programación se efectúa mediante un software gratuito llamado “TwidoSuite”. El fabricante suministra un cable para transferir programas entre el PLC y el autómata o entre el autómata y el PLC.

La E3T cuenta con 4 equipos Zelio, 4 PLC Twido y 2 PLC M340; este último es el más poderoso de los autómatas de Telemecanique que posee la E3T. El PLC M340 implementa los protocolos modbus serial (como maestro o como esclavo) y modbus TCP/IP (como cliente y/o servidor); puede manejar hasta 1024 entradas-salidas y su programación se efectúa a través de un software licenciado conocido como “Unity Pro”. Este autómata también cuenta con su propio cable para hacer transferencia de programas entre el PLC y el PC e igualmente en sentido contrario. Los PLC que le siguen al M340 tanto en capacidad como en precio (el Atrium, el Premium y el Quantum) se programan con el mismo software que el PLC M340.

La interface hombre-máquina (Magelis) que tiene la E3T puede comunicarse por los protocolos modbus serial y modbus TCP/IP, mientras que los variadores de velocidad (4 ATV31) con que cuenta la E3T pueden comunicarse por los protocolos modbus (sobre línea serial) y CANopen.

El nuevo plan de estudios de la carrera de ingeniería electrónica ofrece como asignatura electiva el curso “redes de comunicación industrial” que tiene asignados 3 créditos, 4 horas teóricas semanales y carece de horas de laboratorio. Este trabajo de grado presenta una propuesta de laboratorio (prácticas o guías) para dicha asignatura aprovechando los equipos descritos en los párrafos anteriores. Para ello se ha elegido un proceso industrial y se subdividió en cuatro partes con el fin de ilustrar una aplicación de cada uno de los cuatro protocolos de comunicación industrial que pueden implementarse con los PLC Telemecanique que tiene la E3T; al final se implementan estos cuatro protocolos trabajando en red para ofrecer al estudiante un panorama global de la utilidad del aprendizaje logrado con el laboratorio propuesto.

1. CONCEPTOS BÁSICOS.

1.1 DEFINICION DE PROTOCOLO DE COMUNICACIÓN INDUSTRIAL.

Un protocolo de comunicación industrial es un conjunto de normas o mecanismos de arbitraje que engloba todas las reglas y convenciones que deben seguir un conjunto de equipos cualesquiera para poder intercambiar información, por ejemplo, un protocolo de comunicación industrial determina cual de los terminales está autorizado en cada momento para transmitir información por el bus y cual o cuales terminales están autorizados para recoger información del bus.

1.2. RED

Una red es un conjunto de dispositivos interconectados físicamente (ya sea vía alámbrica o vía inalámbrica) para compartir recursos e intercambiar información, respetando las reglas y protocolos de comunicación.

1.2.1. Tipos de redes: existen varios tipos de redes, las cuales se clasifican de acuerdo a su tamaño.

1.2.1.1. Red de área local (Local Area Network, LAN): permite transportar información a alta velocidad a través de computadoras personales a uno o varios dispositivos que se encuentren a una distancia de máximo 2 Km de radio.

1.2.1.2. Red de área campus (Campus Area Network, CAN): es una colección de LANs que están distribuidas geográficamente dentro de una misma entidad en un área llamada campus; los campus pueden ser de oficinas, universidades, industrias, etc. Una CAN utiliza tecnologías FDDI¹ y Gigabit Ethernet² para su proceso de comunicación.

1.2.1.3. Red de área metropolitana (Metropolitan Area Network, MAN): las redes de área metropolitana cubren extensiones mayores que las redes LAN como ciudades o distritos. Las redes MAN utilizan la interconexión de redes LAN para distribuir información a los diferentes puntos de la ciudad o distrito.

1.2.1.4. Red de área extensa (Wide Area Network WAN): las redes de área extensa cubren grandes regiones geográficas como un país, un continente o incluso el mundo; estas redes utilizan los cables transoceánicos y los satélites para interconectar puntos de información que están a grandes distancias.

1.3. REDES DE COMUNICACIÓN INDUSTRIAL.

Deben su origen a la fundación FieldBus (Buses de campo). Una red de comunicación industrial es la interconexión de islas automatizadas³; esta interconexión se ejecuta mediante buses de campo.

¹ FDDI (Fiber Distributed Data Interface): es un conjunto de estándares para la transmisión de datos en redes de computadoras de área extendida (WAN) o local (LAN) mediante cable de fibra óptica

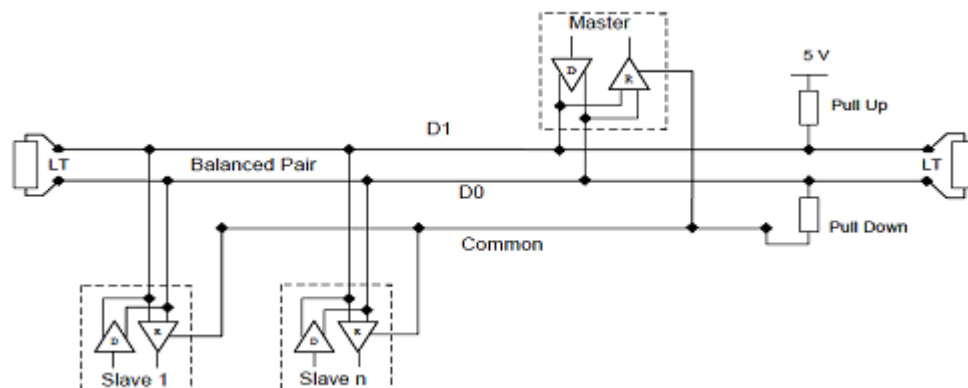
² Gigabit Ethernet: aplicación del estándar Ethernet que transmite datos a 1 Gigabit por segundo.

³ Islas automatizadas: grupos de equipos (PLCs, CNC, variadores de velocidad, PC) aislados entre sí y dedicados cada uno al control de una máquina o de una parte de un proceso.

1.3.1. Bus de campo: es un sistema de transmisión de información (datos) que simplifica la instalación y operación de máquinas y equipos industriales; su objetivo es sustituir las conexiones punto a punto entre los elementos de campo y el equipo de control a través del tradicional lazo de corriente de (4 -20mA) o (0 a 10V DC); los buses de campo generalmente son redes digitales, bidireccionales, multipunto, con gran velocidad en la transmisión de datos que están montadas sobre un bus serie⁴, y conectan dispositivos de campo como PLC's, transductores, actuadores, sensores y equipos de supervisión.

1.3.2. Topología de las redes de comunicación industrial: la topología de redes describe el modo en que varios dispositivos en una red son interconectados. Existen tres topologías básicas: bus, anillo y estrella.

Figura 1. Red en bus.



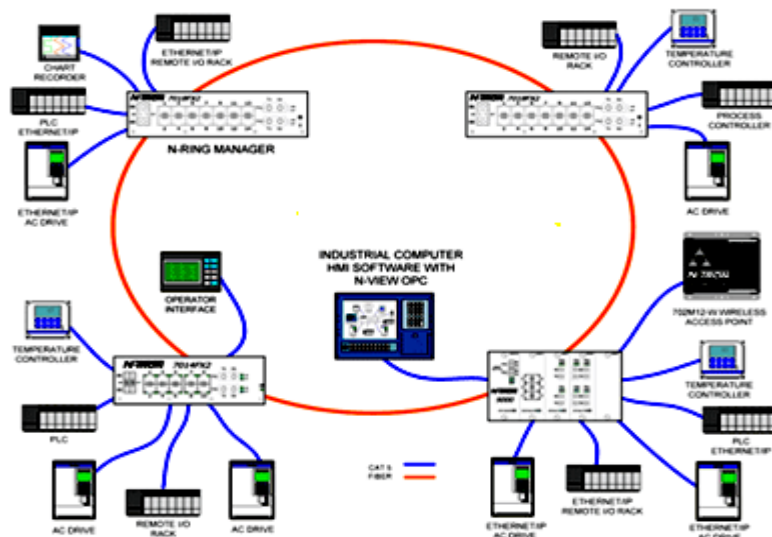
Fuente: MODBUS over serial line specification and implementation. V1.02, pg 22.

⁴ Bus serie: los datos son enviados, bit a bit y se reconstruyen por medio de registros o rutinas de software.

1.3.2.1. Red en bus: esta topología está conformada por una red lineal sobre la que van conectados todos los elementos o dispositivos, cada uno de los cuales verifica el campo de dirección del mensaje para saber si la información contenida en la red es para él o no; si lo es la recibe y actúa conforme a ella. Ver figura 1.

1.3.2.2. Red en anillo: en esta estructura las estaciones están unidas unas con otras formando un círculo por medio de un cable común. Las señales circulan en un solo sentido regenerándose en cada nodo, cada nodo examina la información y supervisa si la información está dirigida a él, si no es de tal manera la pasa al siguiente nodo. Esta topología tiene como gran desventaja que cuando se rompe una conexión, se cae la red completa. Ver figura 2

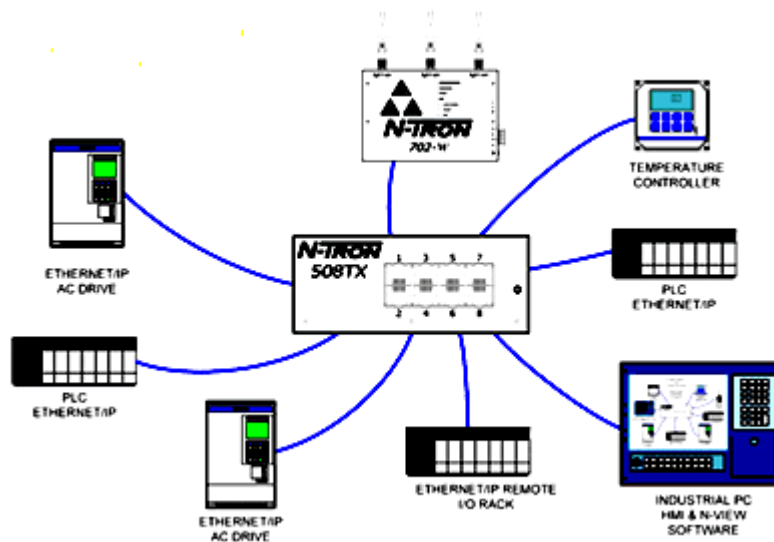
Figura 2. Red en anillo.



Fuente: <http://www.instrumentacionycontrol.net/es/curso-supervision-procesos-por-computadora/243-las-redes-industriales>. Editada por los autores.

1.3.2.3. Red en estrella: la red se une en un único punto central, el cual dirige los bloques de información hacia sus destinos, este punto central también monitorea el tráfico de información y evita las colisiones en la red. En esta topología no se afecta el resto de la red cuando se interrumpe alguna conexión de algún dispositivo que esté conectado al punto central. Ver figura 3

Figura 3. Red en estrella.



Fuente: <http://www.instrumentacionycontrol.net/es/curso-supervision-procesos-por-computadora/243-las-redes-industriales-principales.html>. Editada por los autores.

1.3.3. Dispositivos de redes de comunicación industrial: estos dispositivos son utilizados para procesar las señales de los equipos en los procesos industriales

1.3.3.1. Repetidor: equipo que actúa a nivel físico, prolongando la longitud de la red mediante la unión de dos segmentos de red y amplificando la señal. Junto con

la señal se amplifica el ruido; por esta razón el número de estaciones que pueden compartir el medio es limitado. Opera en la capa física del modelo OSI (Open System Interconnection), que se explica en la página 34 de este texto.

1.3.3.2. Hub (concentrador): consiste en una serie de puntos de conexión (puertos, 8 o 12 generalmente) que son bidireccionales y cuya función es la de reenvío de los paquetes de información que llegan por un puerto cualquiera hacia el resto de puertos⁵, de forma tal que todos los equipos que estén conectados a este dispositivo reciban los paquetes de datos. Los hubs se utilizan para crear un punto de conexión central para los medios de cableado (red en estrella). Los Hubs pueden ser activos (amplifican la señal [realizan la función de repetidor], permitiendo conectar equipos a distancias de hasta 609 metros) o pasivos (reparten la señal conectando equipos a distancias de hasta 30 metros). Opera en la capa física del modelo OSI.

1.3.3.3. Bridge (Puente): equipo que interconecta dos segmentos de red, utilizando el mismo protocolo de comunicación para pasar datos de un segmento hacia el otro segmento con base en la dirección física de destino de cada paquete de datos. Un bridge solo pasa las tramas pertenecientes a cada segmento de la red; por esta característica se mejora el rendimiento de las redes al disminuir el tráfico de datos inútiles. Opera en la capa 2 (enlace) del modelo OSI.

1.3.3.4. Switch (Conmutador): “es un bridge con múltiples puertos. Analiza las tramas que le llegan y genera unas tablas de tráfico que le permiten dirigir cada trama únicamente al puerto de salida donde se encuentre el equipo destinatario”⁶,

⁵ Definición tomada de la página 163 del libro “Comunicaciones Industriales” y modificada por los autores

⁶ Definición tomada de la página 165 del libro “Comunicaciones Industriales”.

optimizando así el ancho de banda de la red y evitando las colisiones. Opera en la capa 2 (enlace) del modelo OSI.

1.3.3.5. Router (Encaminador o enrutador): se basa en el funcionamiento de los bridges; es un equipo de interconexión de redes que actúa a nivel de los protocolos de red (capa 3) del modelo de referencia OSI, determinando las rutas más eficientes que deberá seguir la información. “Cuando llega un paquete a un router, este lee la dirección del destinatario y envía el paquete al siguiente nodo más próximo a la dirección de destino (por ejemplo, otro router que realizara el mismo proceso)”⁷. Existen router que son también switch con 4 puertos y punto de acceso WIFI.

1.3.3.6. Gateway (puerta de enlace): son equipos para interconectar redes con protocolos de comunicación y arquitecturas diferentes. Su propósito es traducir la información del protocolo utilizado en la red de origen al protocolo usado en la red de destino. Opera en la capa 7 (aplicación) del modelo OSI.

1.4. PROCESO DE TRANSMISIÓN DE DATOS.

1.4.1 Elementos en la transmisión de datos: la transmisión de datos se refiere al envío de información en forma confiable desde un punto de origen hasta su destino final, ya sea a pocos metros o a muchos kilómetros de distancia. Se requieren tres elementos para todo proceso de comunicación, estos son: transmisor, medio y receptor.

⁷ Tomado de la página 166 del libro “Comunicaciones Industriales”.

Figura 4. Elementos de un sistema de comunicación.



Fuente: Autores.

- ❖ **Transmisor o emisor:** dispositivo que transmite los datos
- ❖ **Medio:** canal o forma de conexión que permite la transferencia de datos entre el transmisor y el receptor.
- ❖ **Receptor o destino:** dispositivo de destino de los datos

1.4.2. Técnicas de transmisión de datos: básicamente son dos; transmisión serie y transmisión en paralelo.

1.4.2.1. Transmisión serie: los bits se transmiten de uno en uno sobre una línea de transmisión única; por ejemplo, para enviar un byte de datos (8 bits) cada bit es enviado después de otro a través de una línea agregando además un bit para designar el inicio y uno o dos bits mas para indicar el final del byte⁸. Esta técnica es muy útil cuando las transmisiones son a larga distancia.

1.4.2.2. Transmisión en paralelo: Esta técnica requiere de una línea de transmisión por cada bit enviado; los bits se transmiten en grupo sobre varias

⁸ Al final del byte se agrega un bit de paridad (este concepto se profundiza en la pagina) mas el bit de final de byte o se agregan dos bits de final de byte si no se trabaja con bit de paridad.

líneas al mismo tiempo. La transmisión en paralelo es más rápida pero se encarece por necesitar de muchos conductores y porque aumenta la complejidad de los transmisores y los receptores de la línea.

1.4.3. Tipos de transmisión de datos.

1.4.3.1. Transmisión asíncrona: corresponde a la capa física del modelo OSI; es la más usada para proveer conectividad entre computadoras personales, módems, faxes, PLC's etc. La característica más importante de este tipo de transmisión es que los relojes del transmisor y del receptor son independientes y no sincronizados. Para sincronizar la información, las cadenas de bits de datos se envían precedidas por un bit de arranque y seguidas de un bit de parada, o seguidas por dos bits de parada o de un bit de paridad (que ayuda a determinar los errores de transmisión) más un bit de parada. Este tipo de transmisión de datos requiere muchos bits de comprobación y de control (generalmente 2 o 3 por cada byte transmitido) en comparación con la transmisión síncrona; en la transmisión asíncrona el uso de relojes independientes en el transmisor y el receptor limita la transmisión de las cadenas de bits de datos a cadenas relativamente cortas (≤ 8 bits); también limita la velocidad hasta 64 Kbps.

1.4.3.2. Transmisión síncrona: en este tipo de transmisión se puede transmitir cadenas de bits de datos que incluyan gran número de bits, debido a que no se necesitan bits de arranque y de parada para sincronizar los datos de información; la sincronización entre transmisor y receptor se realiza mediante canales separados de reloj, los cuales manejan la transmisión y recepción de la información mediante la exactitud del tiempo

1.4.3.3. Definición de baudio: el número de veces que puede cambiar el nivel (valor – por ejemplo en volts–) de una señal digital en un segundo se mide en baudios; si dichos cambios pueden darse cada Δt segundos, entonces

$$\text{Baudios} = 1s / \Delta t.$$

El numero de baudios se relaciona con el numero de bits por segundo (bps) mediante la fórmula

$$\text{bps} = \text{baudios} \times \log_2 (n)$$

Donde n es el numero de niveles que puede tomar la señal digital; si una señal es binaria ($n=2$), el numero de baudios será igual al número de bps; si $n=4$, el numero de bps será el doble del numero de baudios; lo anterior puede constatarse reemplazando $n=2$ y posteriormente $n=4$ en la formula anterior.

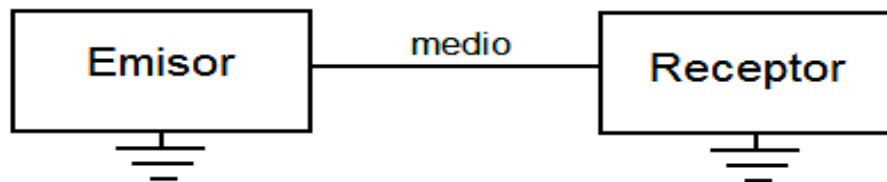
1.4.4. Modos de transmisión de datos:

1.4.4.1. Transmisión simple (SIMPLEX): Se presenta cuando el medio de transmisión se usa en una sola dirección. Un ejemplo de transmisión simple es la radio, la cual envía información al usuario sin que este último efectúe una transmisión de respuesta. Ver figura 5

1.4.4.2. Transmisión semidoble (HALF-DUPLEX): En este modo de transmisión ambos extremos del sistema de comunicación cumplen funciones de transmisor y receptor y los datos se desplazan en ambos sentidos pero no simultáneamente.

Un ejemplo de transmisión semidoble son los transceptores⁹ portátiles que usan el método push to talk¹⁰ (como el walkie talkie). Otro ejemplo de transmisión semidoble es el bus de campo modbus serial, donde el medio usualmente se implementa con dos hilos (alambres) que transportan una señal diferencial (tensión de un hilo con respecto a la referencia menos la tensión del otro hilo con respecto a esa misma referencia) para reducir el efecto de las interferencias. Ver figura 6

Figura 5. Transmisión simple.



Fuente: Autores

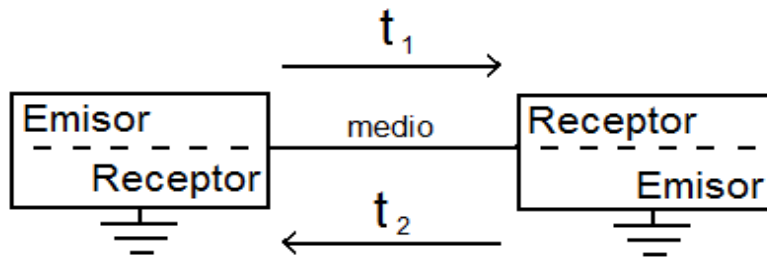
1.4.4.3. Transmisión doble (FULL-DUPLEX): el sistema es similar al Half dúplex, pero los datos se desplazan por el medio de transmisión en ambos sentidos simultáneamente. Para ello ambos transmisores poseen diferentes frecuencias de transmisión (medio inalámbrico como la telefonía móvil o alámbrico de dos hilos como la telefonía fija) o dos caminos de comunicación separados (medio habitualmente implementado con cuatro hilos [dos hilos llevando una señal

⁹ Dispositivo que realiza funciones tanto de transmisión como de receptor.

¹⁰ Push to talk traduce “Pulsar para hablar”: consiste en apretar un botón para transmitir y en liberarlo para recibir, permitiendo implementar comunicaciones uno a uno o uno a varios; Esta comunicación half dúplex puede ser más útil que una full dúplex dependiendo del contexto, por ejemplo cuando se habla en grupo, ya que la conversación es menos ruidosa porque no se filtra el ruido de todos los transceptores del grupo (lo que hace muy popular en comunicaciones empresariales). Cuando uno de los miembros del grupo tiene el botón pulsado y habla, los demás miembros del grupo lo escuchan y no pueden interrumpirlo hasta que suelte el botón (después de decir “cambio”).

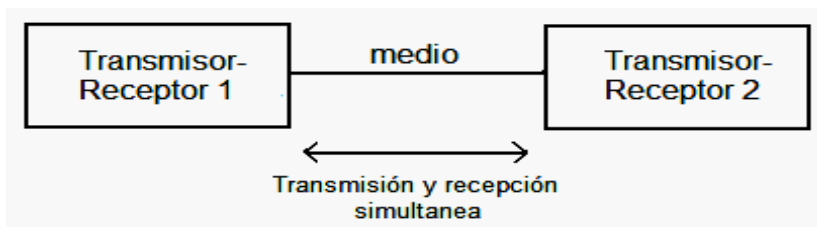
diferencial en una dirección y otros dos hilos llevando otra señal diferencial en dirección opuesta]) Ver figura 7

Figura 6. Transmisión semidoble



Fuente: Autores

Figura 7. Transmisión doble.



Fuente: Autores.

1.5 MODELOS DE REFERENCIA OSI:

El modelo OSI (Open Systems Interconnection) de ISO (International Standards Organization), tiene como propósito estandarizar la comunicación entre equipos de diferentes fabricantes con sus propias características, mediante la división de la

comunicación en niveles y la normalización de cada protocolo perteneciente a un nivel.

1.5.1. Características del modelo OSI: el modelo OSI presenta las siguientes características:

Tiene una estructura multinivel: diseñada para que cada nivel se dedique a resolver su parte de la comunicación.

Interacción entre niveles: Cada equipo se comunica con otros equipos que están a su mismo nivel utilizando los niveles inferiores de su nivel en el mismo equipo. En otras palabras el nivel N utiliza los servicios del nivel N-1 para comunicarse con el nivel N de otros equipos y el nivel N+1 utiliza los servicios de N para comunicarse con el nivel N+1 de otros equipos

Encabezados: En cada nivel se agrega al mensaje un formato de control; la función de este formato es permitir que un nivel en el receptor de un equipo se entere de que el mismo nivel en el emisor de otro equipo le envía información.

1.5.2. Capas del modelo OSI: el modelo OSI esta compuesto por 7 capas que a continuación se presentan. la figura 8 presenta este modelo

1.5.2.1. Capa física: Se encarga de especificar la conexión física (ej.: conectores RJ45, DB9, Mini DIN, etc.) y el medio de comunicación (ej.: cable de par trenzado, cable coaxial, fibra óptica, etc.) entre el dispositivo y la red con sus características físicas, mecánicas y eléctricas.

1.5.2.2. Capa de enlace: Esta capa maneja bloques de bits y les añade los mecanismos necesarios para poder detectar y corregir los errores que puedan aparecer (ej.: CRC, Cyclic Redundance Code). También se encarga del control de acceso al medio (ej.: maestro-esclavo, CSMA/CD, token, etc.).

1.5.2.3. Capa de red: se ocupa del direccionamiento mediante técnicas de encaminamiento (routing). “Aquí se aplican algoritmos de control de tráfico y optimización, de manera que un mensaje pueda estar fragmentado en varios paquetes que no tiene por qué seguir el mismo camino. Cada paquete seguirá el camino óptimo determinado por los controladores de la red atendiendo a variables tales como la disponibilidad de equipos y el estado del tráfico de la red en cada momento”¹¹. Esta capa es muy útil, principalmente en la comunicación entre redes.

1.5.2.4. Capa de transporte: “tiene la misión de garantizar un enlace fiable entre terminales (garantiza que el conjunto de paquetes que conforman el mensaje estén formados en la secuencia correcta, sin errores ni duplicaciones). Divide la información (mensajes) en paquetes manejables por el sistema de transmisión; estos paquetes son entregados a los mecanismos del nivel 3, para que los transporte a su destino y una vez allí los devuelva al nivel 4, que los unirá en su secuencia correcta, reconstruyendo así el mensaje original (gestiona la apertura y cierre de las conexiones de envío y recepción de datos)”¹².

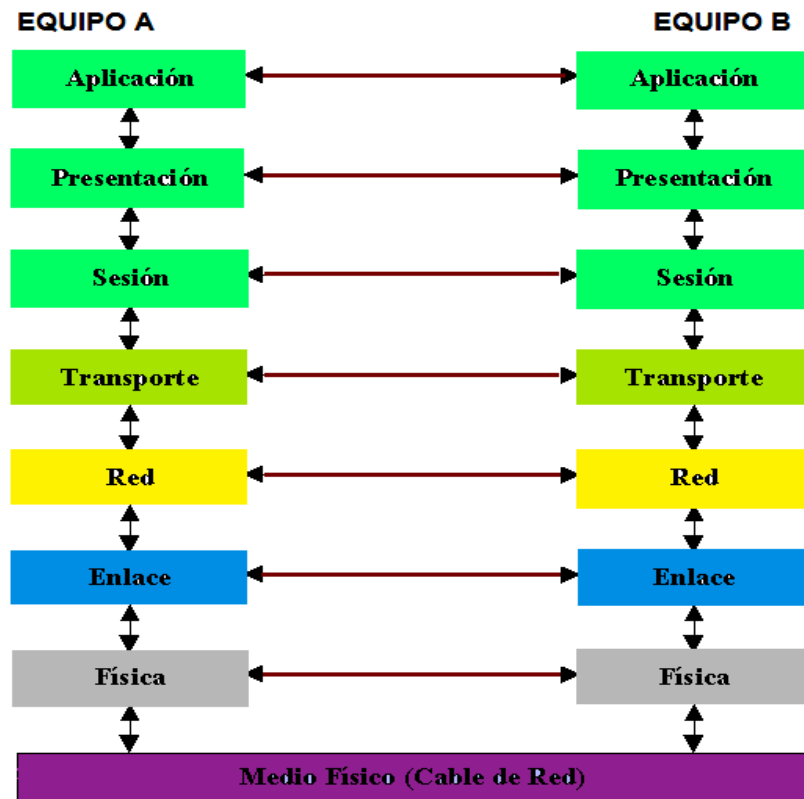
1.5.2.5. Capa de sesión: Es la responsable de administrar la comunicación entre los programas de los equipos de una red, estableciendo el inicio de la comunicación, su mantenimiento y posterior finalización; controla el diálogo

¹¹ Tomado del libro “Comunicaciones Industriales”, pg 28.

¹² Tomado de las páginas 29 y 178 del libro “Comunicaciones Industriales”.

estableciendo el orden en que los mensajes deben fluir entre los usuarios finales y permite escribir programas que correrán en cualquier instalación de red.

Figura 8. Capas del modelo OSI.



http://www.pchardware.org/redes/redes_osi.php

1.5.2.6. Capa de presentación: Codifica y decodifica la información que se transmite para evitar que sea utilizada por usuarios no autorizados, o dicho de otra forma, “adecua la información para transmitirla y que pueda entenderla el receptor”¹³.

¹³ Tomado del libro “Telecomunicaciones y telemática”, pg 157.

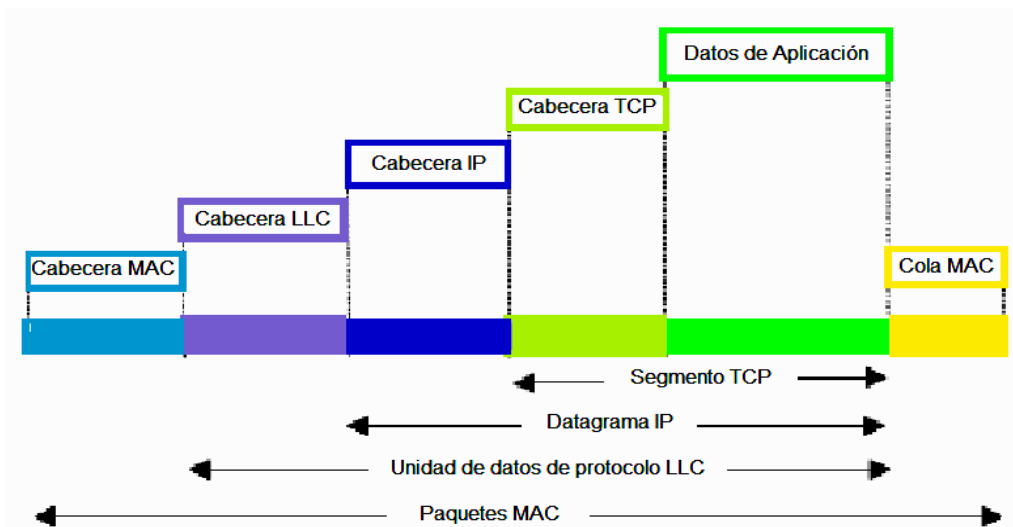
1.5.2.7. Capa de aplicación: esta capa se entiende en forma directa con el usuario a través de aplicaciones (ej.: Procesador de palabras, presentaciones graficas, base de datos etc.) y es un campo de libre utilización para fabricantes y usuarios; “maneja los protocolos que las aplicaciones de usuario invocan para comunicarse con otros equipos de la red, utilizando los servicios de las capas inferiores. El usuario de una aplicación simplemente especifica la operación que desea hacer con otro(s) equipo(s) y la capa de aplicación se encarga de traducirla en las instrucciones reales que hay que ejecutar en la red para lograrlo”.

1.6. MODELO TCP/IP.

Está basado en un modelo de referencia de cuatro niveles, y es la base de Internet que sirve para enlazar computadoras incluso utilizando diferentes sistemas operativos. La figura 9 presenta la forma en que se empaquetan las tramas de datos en el modelo TCP/IP y la figura 10 presenta como cada nivel del modelo TCP/IP corresponde a uno o más niveles del modelo de referencia OSI. A continuación se explica cada nivel

1.6 1. Nivel de aplicación: pertenece al nivel más alto del modelo TCP/IP y corresponde a las capas de aplicación, presentación y sesión del modelo OSI, representa las aplicaciones disponibles para el usuario. Algunos de los protocolos pertenecientes a esta capa son el protocolo SMTP (Simple Mail Transfer Protocol o protocolo simple de transferencia de correo), el protocolo FTP (File Transfer Protocol o protocolo de transferencia de archivos) y el HTTP (Hypertext Transfer Protocol o protocolo de transferencia de hipertexto).

Figura 9. Cabeceras de tramas de datos en TCP/IP.



Fuente: <http://www.textoscientificos.com/redes/tcp-ip/comparacion-modelo-osi>

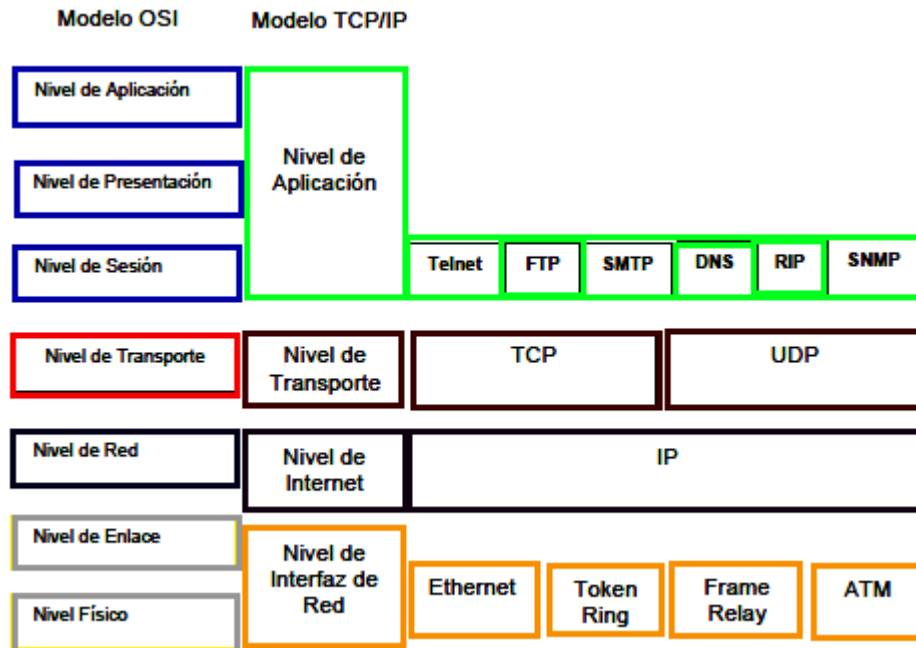
1.6.2. Nivel de transporte: Este nivel es equivalente a lo definido para la capa de transporte del modelo OSI y tiene como finalidad manejar los datos y proporcionar la fiabilidad necesaria en el transporte de los mismos.

1.6.3. Nivel de Internet: Este nivel equivale a lo definido para la capa de red del modelo OSI. Aquí se controla la comunicación de una máquina a otra, aceptando una solicitud para enviar un paquete desde la capa de transporte, junto con una identificación de la máquina, hacia la capa destino.

1.6.4. Nivel de interfaz de red: el nivel inferior del modelo TCP/IP consta de una capa de interfaz de red responsable de aceptar los datagramas IP y transmitirlos hacia una red específica. Una interfaz de red puede consistir en un dispositivo

controlador o un complejo subsistema que utiliza un protocolo de enlace de datos propios.

Figura 10. Modelos TCP/IP y OSI.

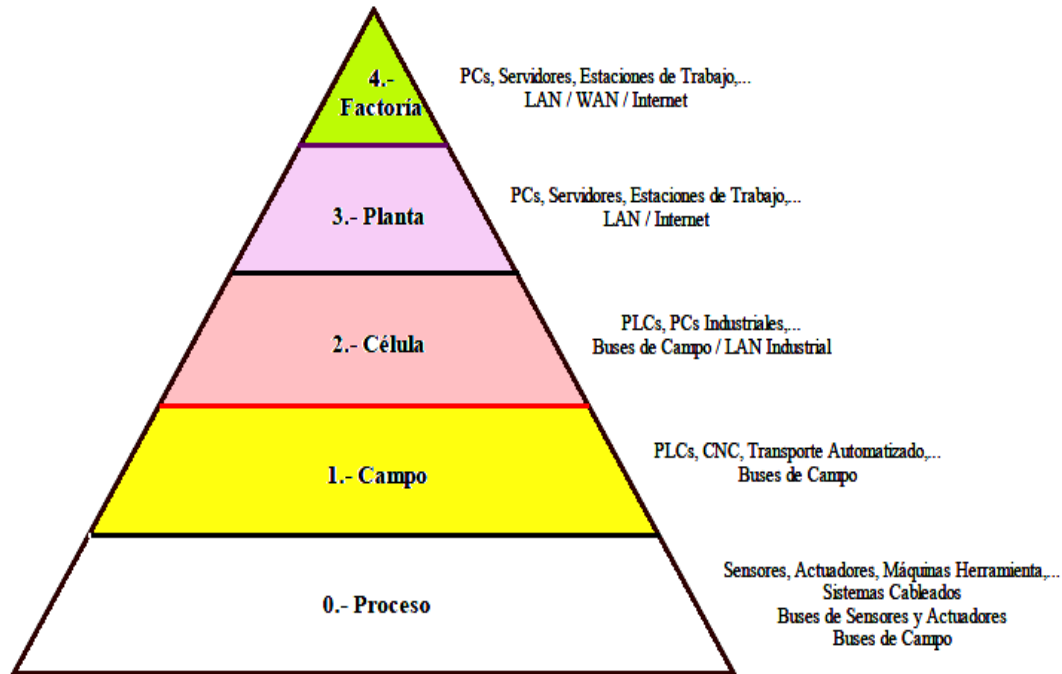


Fuente: <http://www.textoscientificos.com/redes/tcp-ip/comparacion-modelo-osi>

1.7. PIRAMIDE CIM (COMPUTER INTEGRATED MANUFACTURING)

En una red industrial las comunicaciones se realizan agrupándose en subsistemas que pertenecen a niveles de información tratada. Cada subsistema debe tener comunicación directa con los subsistemas del mismo nivel y con los niveles inmediatamente superior e inferior. Así aparecen cinco niveles (ver figura 11).

Figura 11. Pirámide CIM



Fuente: http://tv.uvigo.es/uploads/material/Video/1466/ISAD_Tema5.pdf

1.7.1. Nivel de Proceso: en este nivel se realiza el control directo de las máquinas y sistemas de producción. Los dispositivos conectados son sensores, actuadores, instrumentos de medida, máquinas de control numérico, etc. Se utiliza cableado tradicional o el bus de campo AS-Interface (AS-i).

1.7.2. Nivel de Campo: los dispositivos que pertenecen a este nivel son autómatas de gama baja y media, sistemas de control numérico, transporte automatizado, en este nivel se utilizan las medidas proporcionadas por el nivel 0 y se les asignan las consignas a los dispositivos de dicho nivel. Generalmente se usan buses de campo del tipo: AS-i, modbus, Device Net, CANopen

Los niveles de proceso y campo utilizan paquetes de información del orden de los bits o bytes

1.7.3. Nivel de Célula: en este nivel se encuentran sistemas que controlan la secuencia de fabricación y/o producción, los cuales asignan las consignas al nivel de campo. Se emplean autómatas de gama media y alta, así como equipos de visualización, redes LAN y buses de campo como, Profibus FMS, Profibus PA, Ethernet y Modbus.

1.7.4. Nivel de Planta: corresponde al sistema de diseño y gestión en el que se estudian las órdenes de fabricación, producción y supervisión que seguirán los niveles inferiores de la pirámide CIM. Se emplean estaciones de trabajo, servidores de bases de datos, redes LAN del tipo Ethernet TCP/IP.

Los niveles de célula y planta utilizan paquetes de información del orden de los Kbytes

1.7.5. Nivel de Factoría: gestiona la producción completa de la empresa, comunica las distintas plantas, mantiene las relaciones con los proveedores y clientes y proporciona las consignas básicas para el diseño y la producción de la empresa. Se emplean computadores personales (PC), estaciones de trabajo y servidores de distinta índole. Se usan redes LAN y/o WAN utilizando Ethernet TCP/IP. etc.

2. PROCOLO DE COMUNICACIÓN AS-Interface (AS-i)

2.1. HISTORIA DE AS-i

AS-interface, es el resultado de un proyecto iniciado en 1990 por un consorcio compuesto por 11 empresas fabricantes de sensores y actuadores; este bus se exhibió en 1994 con un protocolo de comunicación industrial del mismo nombre que tiene la ventaja de transmitir datos a través de un cable bifilar a distancias de hasta 100 metros sin repetidores y 300 metros con repetidores; este cable también sirve como medio de energización¹⁴¹⁵. El protocolo está diseñado para trabajar en el nivel más bajo de la pirámide de automatización, donde existe gran número de elementos de tipo digital binario; entre los que se pueden destacar pulsadores, finales de carreras, sensores inductivos y capacitivos, fotodetectores y electroválvulas.

La association for promoting interfaces with us capability for binary actuators and sensors, llamada también AS-i association; tiene como una de sus funciones la promoción y difusión del protocolo de comunicación AS-i; el cual está regulado por los estándares (EN 50295, IEC 947)

2.1.1. Versión 2.04 (1994): en su forma original la red era capaz de direccionar hasta 31 dispositivos (esclavos) de tipo binario, donde cada dispositivo podía intercambiar 4 bits de datos de entrada y 4 bits de datos de salida, dando como resultado un total de 124 entradas y 124 salidas en una sola red.

¹⁴ Energización o alimentación: quiere decir que el cable es usado para entregar a los esclavos AS-i la tensión y la corriente que necesitan.

¹⁵ Por los dos hilos del cable AS-i se transmiten simultáneamente alimentación (potencia) y comunicaciones (datos).

2.1.2. Versión 2.1 (2002): esta versión, mejora las prestaciones del bus, entre las que se destacan el aumento del número de direcciones de esclavos al doble, la mejora de la capacidad de diagnóstico; permite la gestión de señales analógicas manteniendo la compatibilidad con elementos que se conectaban en la versión anterior. En esta versión, el protocolo soporta dos tipos de perfil para los esclavos:

2.1.2.1. Estándar (perfil M0): en su nivel básico es capaz de direccionar hasta 31 esclavos, cada uno de los cuales controla hasta 4 entrada y 4 salidas para un total de 248 señales digitales.

2.1.2.2. Ampliado (perfil M1): también conocidos como esclavos A/B. que se direccionan de la siguiente forma: 1A, 1B, 2A, 2B, hasta el ordinal 31.

De lo anterior se puede deducir que la capacidad de direccionamiento de esclavos se duplica y el número total de entrada/salidas se incrementa de la siguiente manera:

$62 \text{ Esclavos} \times 4 \text{ entradas} = 248 \text{ señales digitales de entrada}$

$62 \text{ Esclavos} \times 3 \text{ salidas} = 186 \text{ señales digitales de salidas}$

En este perfil el tiempo de una transacción es máximo de 10 ms; mientras que en el perfil de M0 es justo la mitad; se entiende por transacción la acción de pregunta del maestro y respuesta del esclavo.

2.1.3. Versión 3.0 (2004): Esta versión no cambia la estructura básica (perfil M0), adicionando las siguientes características respecto a la versión anterior:

Permite el intercambio de mayor cantidad de datos utilizando la técnica de transacciones combinadas (las cuales se definirán posteriormente en el

numeral...2.5.2) pudiéndose accesar hasta 8 entradas y 8 salidas con direccionamiento ampliado o extendido (direccionamiento A/B).

Esclavos de tipo A/B con entradas y salidas analógicas

Compatible con Esclavos de versiones anteriores

2.2. CARACTERISTICAS DE LA RED AS-i

Las principales características de este protocolo de comunicación se presentan en la siguiente tabla

Tabla 1 Características red AS-i

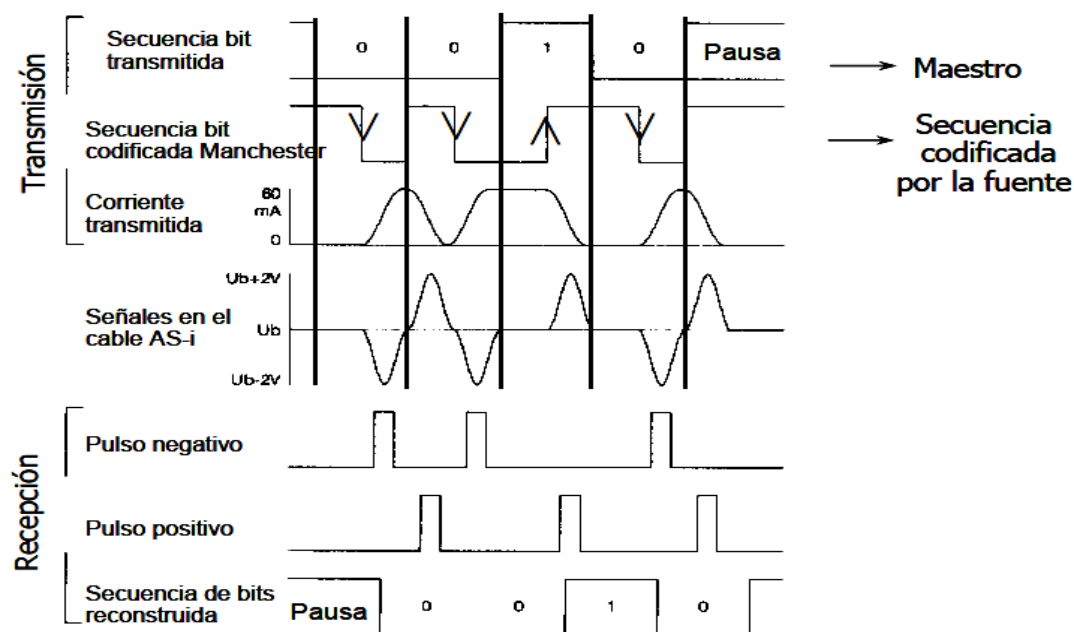
Características	Descripción
Velocidad de transmisión	166,67 Kbps
Tiempo de bit(TB)	6 μ s
Numero de esclavos	62
Tiempo máximo de ciclo en modo normal	5 ms
Detección de error bit de paridad	bit de paridad par
Corrección de error	Reenvió de datos por parte del maestro
Acceso al medio	Sondeo cíclico por parte del maestro
Tipo de bus	Maestro-esclavo, monomaestro

Fuente: Trabajo de grado “protocolo AS- interface, analizador de red”, pg. 24

2.3. CODIFICACION DE LA SEÑAL

En el bus de comunicación, cada mensaje está formado por una petición del maestro y su respectiva respuesta; las tramas de la señal mensaje son codificadas en formato Manchester donde "0" lógico genera un flanco negativo de señal, un "1" lógico genera un flanco positivo de señal y el periodo sin señal se toma como un nivel lógico alto. Después de codificar la señal es modulada en pulsos alternativos (APM); esta señal es representada por pulsos de corriente con amplitudes entre 55mA y 68mA los cuales generan pulsos de voltaje tipo $(\sin)^2$ con amplitudes de $\pm 2V$ sobre el nivel de DC, obteniendo de esta manera la señal presentada en la siguiente figura.

Figura 12: Formas de onda de la señal AS-i



Fuente: Trabajo de grado "protocolo AS- interface, analizador de red, pg. 25"
 .Modificado por los autores

Los pulsos de voltaje resultantes pueden ser modelados por la siguiente ecuación

Ecuación 1

$$u(t) \approx \pm U_{send} \sin^2 \left(\frac{2\pi}{6\mu s} t \right) \quad \text{con} \quad U_{send} \approx 2 [V]$$

2.4. ESTRUCTURA DE UNA TRAMA AS-i

La estructura de una trama para la petición del maestro y la respuesta del esclavo, se muestran en la fig. 13 y en la tabla 2 respectivamente, en esta tabla y esta figura se observa, las peticiones de los Maestros y las respuestas de los esclavos que tienen un bit de inicio fijo a “0” y un bit de fin fijo a “1”; además, esta estructura de mensajes utiliza un bit de paridad para la detección de errores, lo cual permite mayor seguridad en el intercambio de datos entre el maestro y el esclavo. Es importante destacar que el protocolo maneja paridad par.

Figura 13: Estructura de los mensajes AS-i



Fuente: trabajo de grado “protocolo AS-interface. Implementación de un nodo maestro en PC, pg. 20”

Tabla 2: Significado de cada bit

Maestro		Esclavo	
Bit	Descripción	ST	Bit de comienzo, siempre "0"
ST	<i>Bit</i> de inicio de datagrama. Fijo a "0".	I3...I0	Bits de datos netos del esclavo
CB	<i>Bit</i> de Control.	PB	Bit de paridad
	0 = Transmisión de Datos/Parámetros. 1 = Transmisión de Comandos.	EB	Bit de finalización, siempre "1"
A4 - A0	Dirección del Esclavo.		
I4	<i>Bit</i> de Información.		
	0 = Petición de Datos. 1 = Petición de Parámetros.		
I3 - I0	<i>Bits</i> de Información.		
PB	<i>Bit</i> de Paridad. Paridad Par.		
EB	<i>Bit</i> de fin de datagrama. Fijo a "1".		

Fuente: trabajo de grado "protocolo AS-interface. Implementación de un nodo maestro en PC, pg. 21" modificado por los autores

2.5 TIPOS DE TRANSACCIONES

Existen dos tipos de transacciones, las cuales son simples y combinadas:

2.5.1 Transacciones simples: estas transacciones definen las funciones de intercambio de datos, parametrización, diagnóstico y administración de la red.

2.5.2.2 Tipo II: Este tipo de transacción usa el mecanismo de transferencia de datos de AS-i para construir un canal de comunicación serial full dúplex para la transferencia de bits entre el maestro y el esclavo.

Esta transacción puede ser usada para en el manejo de sensores y actuadores analógicos, dispositivos de campo con parámetros variables y el reemplazo de interfaces de 4 – 20mA.

Tabla 4: Transacciones combinadas:

Tipo	PERFIL Esclavo	PERFIL Maestro	Descripción
I	S-7.3	M.3	Entradas - Salidas de 16- <i>Bits</i> .
I	S-7.4	M.3	Dispositivos de campo complejos.
II	S-7.5.5	M.4	Combinación de dispositivos de campo.
II	S-B.A.5	M.4	Comunicación serial en dispositivo de campo.
III	S-7.A.7	M.4	4I/4O en modo de direccionamiento extendido.
III	S-7.A.A	M.4	8I/8O en modo de direccionamiento extendido.
IV	S-7.A.8	M.4	Entradas de 16- <i>Bits</i> en direccionamiento extendido.
IV	S-7.A.9	M.4	Entradas duales de 16- <i>Bits</i> en modo extendido.
V	S-6.0	M.4	Entradas - Salidas de 16- <i>Bits</i> de alta velocidad.
<i>Safety</i>	S-7.B	–	Cualquier esclavo de entrada para trabajo seguro.

Fuente: trabajo de grado “protocolo AS-interface. Implementación de un nodo maestro en PC, pg. 22” modificado por los autores.

En la tabla se puede observar el tipo de transacción, el perfil del maestro las cuales pueden ser: si la transición es de tipo I tiene un perfil M.3, si la transición es de un orden superior al Tipo I el perfil del maestro es M4. También se observa el perfil del esclavo que está relacionado con el tipo de transición y las entradas digitales binarias del dispositivo esclavo. Esta referencia es la versión 3.0 de AS-interface. Ver esclavos sección 2.6.2 de este documento de grado.

2.5.2.3 Tipo III: Este tipo de transacción usa el mecanismo de transferencia de datos de AS-i para construir canales de comunicación half dúplex de 4-Bits u 8-Bits entre el maestro y el esclavo, en el modo de direccionamiento extendido. Esta transacción es usada específicamente en el manejo de teclados, torres visualizadoras de señal, válvulas terminales y sensores/actuadores de 8-Bits.

2.5.2.4 Tipo IV: Este tipo de transacción usa el mecanismo de transferencia de datos de AS-i para construir un canal de comunicación semi o full dúplex donde se transmiten 16-Bits de datos desde el esclavo al maestro en el modo de direccionamiento extendido. Este tipo de transacciones es para el uso de sensores de 16 bits los cuales reemplaza la interfaz de 4 – 20mA

2.5.2.5 Tipo V: Estas transacción usa el mecanismo de transferencia de datos de AS-i para construir canales de comunicación full dúplex rápida, donde se transmiten 8-Bits, 12-Bits o 16-Bits de datos desde el maestro al esclavo usando 2, 3 o 4 direcciones estándar. Este tipo de transacciones es para el uso de sensores/actuadores de 16 bits los cuales reemplaza la interfaz de 4 – 20mA por lazos de control

2.6 COMPONENTES DE LA RED AS-I

Los componentes básicos de la red AS-i son: maestro AS-i, esclavos AS-i, Cable AS-i y fuente de alimentación

2.6.1 Maestro AS-I con Referencia TWDNOI10M3

Figura 14: Módulo maestro TWDNOI10M3



Fuente: [http://www.global-download.schneider-electric.com/852575770039EC5E/all/657A35633D45CE56852576400063BF01/\\$File/as-i_bus-07.pdf](http://www.global-download.schneider-electric.com/852575770039EC5E/all/657A35633D45CE56852576400063BF01/$File/as-i_bus-07.pdf)

Cada Maestro AS-i controla una red AS-i; encargándose de recoger los datos de la red y enviarlos al PLC correspondiente, y viceversa, también se ocupa de la comunicación con 1 o un máximo de 62 esclavos utilizando la técnica de muestreo polling (servicio de pregunta/respuesta).

El maestro también realiza estas otras funciones:

Administra y organiza el tráfico de datos en la red AS-Interface y en caso necesario pone los datos de los sensores y actuadores a disposición del PLC o de un sistema de bus superior; (por ejemplo, modbus, Ethernet, CANopen), a través de las denominadas pasarelas DP/AS-Interface.

Se encarga del diagnóstico de los esclavos, principalmente errores de comunicación y fallas.

Realiza Control de intercambio de datos.

La característica más sobresaliente es la autoconfiguración, de esta manera el maestro AS-i realiza todo el proceso de configuración y monitoreo del sistema; entre las que se destacan: velocidad de transmisión de datos, cantidad de datos de los mensajes y asignación de dirección.

El maestro AS-i se caracteriza por sus perfiles, los cuales están relacionados con cada versión del protocolo AS-i y con el tipo de transición simples o combinada ya presentadas en 2.5. A continuación se presenta la tabla de perfiles del maestro para la versión 3.0 del protocolo AS-i

Tabla 5: Perfiles del maestro

Perfil	Funcionalidad
M0	intercambio cíclico de datos, sólo datos binarios, sin modo de direccionamiento extendido
M1	igual que M0, se añaden datos analógicos y parámetros AS-i acíclicos
M2	intercambio cíclico de datos y parámetros AS-i acíclicos
M3	igual que M1, se completa con el modo de direccionamiento extendido (62 esclavos), y el protocolo analógico S-7.3.x y S-7.4.x
M4	igual que M3, se completa con el protocolo de datos de la transmisión de bytes

Fuente: http://www.ifm.com/obj/ifm_AS-Interface_Catalogue_ES_08.pdf

El maestro AS-i tiene la ventaja de acceder a todos los parámetros del bus como son velocidad de transmisión, número de esclavos, etc.; por esta razón es posible tener el control permanente de:

Esclavos conectados

Esclavos activos

Esclavos “desaparecido de la configuración nominal”.

Alimentación del bus

Estado del maestro

Usualmente la puesta en tensión del maestro conlleva los siguientes pasos:

Inicialización: el maestro inicia su buffer de memoria, tablas de datos, base de datos de esclavos detectados, base de datos de los esclavos activos, base de datos de la configuración nominal de la red.

Detección y activación de esclavos: el maestro busca en el bus todos los esclavos conectados y compara esta lista con la lista de configuración nominal, para intercambiar datos con los esclavos activos.

Intercambio cíclico de datos: el maestro se comunica con los esclavos y comienza a intercambiar datos.

2.6.2 Esclavos: los esclavos del protocolo AS-i reconocen los bits de datos enviados por el maestro y les devuelven sus propios datos. Hay dos tipos diferentes de esclavos: esclavos convencionales (estos módulos no traen el chip AS-i) como: módulos digitales, módulos analógicos, módulos neumáticos, etc. y esclavos AS-i como arrancadores de motores, relés, botoneras, etc.

A continuación se explicará los dos tipos de esclavos:

Figura 15: Botonera XAL-S2002E.



Figura 16: Relé ASI20MT414OR



Fuente: [http://www.global-download.schneider electric.com/85257578007e5c8a/all/07670f5195aab95f88257578003a0f88/\\$file/48361-en%20\(web\).PDF](http://www.global-download.schneider electric.com/85257578007e5c8a/all/07670f5195aab95f88257578003a0f88/$file/48361-en%20(web).PDF)

2.6.2.1 Esclavo convencional: Son elementos digitales binarios o actuadores estándar, como finales de carrera, electroválvulas, sensores etc. que necesitan de un módulo de interface que contenga el chip AS-i (módulo activo) para poder conectarse al bus AS-i, estos sensores y actuadores ocuparan una dirección del bus; los módulos disponen de tomas de señales estandarizadas, lo cual permite la conexión de sensores y actuadores. Algunos de estos esclavos se describen a continuación:

Módulo AS-i pasivo: no contiene electrónica integrada, están diseñados para conectar sensores y actuadores con chip AS-i integrado

Esclavos analógicos: son esclavos que intercambian valores analógicos con el maestro AS-i. A continuación se presenta esclavos analógicos con los siguientes perfiles:

Esclavos analógicos según perfil S7.1/S7.2: estos esclavos necesitan de un programa controlador y/o bloques de funciones para la transferencia secuencial de los datos analógicos.

Esclavos analógicos según perfil S7.3/S7.4: están diseñados para trabajar con maestros AS-i extendidos, los cuales desarrollan automáticamente la transferencia de datos con estos esclavos.

2.6.2.2 Esclavo AS-i: Estos esclavos ya incluyen desde la fábrica el chip AS-i, por lo cual se puede conectar directamente al bus mediante un distribuidor o toma pasivos. Ocupan una dirección completa del bus; se pueden colocar 31 Esclavos básicos o 62 ampliados, estos Esclavos AS-i se identifican mediante un código de entradas-salidas y uno o más identificadores (ID)

Módulos AS-i: son esclavos AS-i a los que se pueden conectar hasta 4 sensores convencionales y hasta 4 actuadores binarios convencionales.

Sensores/actuadores con conexión AS-i integrada: estos sensores se conectan directamente al bus.

Módulo AS-i activo: son usados para conectar sensores y actuadores convencionales.

2.6.2.3 Perfil de los esclavos AS-i: cada esclavo AS-i que se conecta al bus está definido por su perfil, el cual se determina por el código de entradas y salidas (código I/O) y el código de identificación (código ID).

El código I/O permite identificar la configuración de entradas y salidas de un esclavo AS-i como: número de entradas, número de entradas/salidas, número de salidas y tres estados.

Este código I/O utiliza la información de D0, D1, D2 y D3 para hacer las diferentes combinaciones que serán representadas en código hexadecimal de 0 a F y ocuparán la primera posición del perfil.

Tabla 6: Códigos I/O

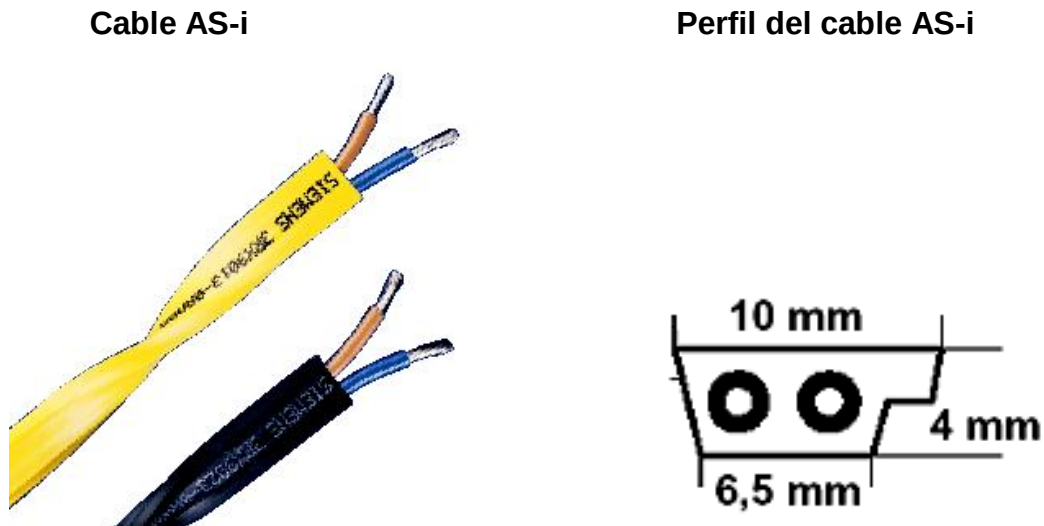
	D0	D1	D2	D3
0	IN	IN	IN	IN
1	IN	IN	IN	OUT
2	IN	IN	IN	IN/OUT
3	IN	IN	OUT	OUT
4	IN	IN	IN/OUT	IN/OUT
5	IN	OUT	OUT	OUT
6	IN	IN/OUT	IN/OUT	IN/OUT
7	IN/OUT	IN/OUT	IN/OUT	IN/OUT
8	OUT	OUT	OUT	OUT
9	OUT	OUT	OUT	IN
A	OUT	OUT	OUT	IN/OUT
B	OUT	OUT	IN	IN
C	OUT	OUT	IN/OUT	IN/OUT
D	OUT	IN	IN	IN
E	OUT	IN/OUT	IN/OUT	IN/OUT
F	IN/OUT/NUL	IN/OUT/NUL	IN/OUT/NUL	IN/OUT/NUL
IN : Entrada OUT : Salida IN/OUT : Entrada/Salida IN/OUT/NUL : Tres estados				

Fuente: “Bus AS-i tecnología de control tema 4, Pg 28”

Código de identificación (código ID): permite diferenciar esclavos con el mismo código I/O y se representa en código hexadecimal de 0 a F, ocupan la segunda posición del perfil. A partir de la versión 2.1 de AS-i este código se extendió proporcionando otros dos códigos ID1 y ID2 que también tiene codificación hexadecimal, ocuparan las posiciones tres y cuatro del perfil de los esclavos AS-i. ID1 representa las funcionalidades del esclavo e ID2 proporciona información adicional del esclavo, por ejemplo: en los esclavos analógicos este código indica el número de canales

2.6.3 Cable AS-i

Figura 17: Cable as-i



Fuente: <http://homepage.cem.itesm.mx/vlopez/as-interface.htm>

Tabla 7: Especificaciones cable AS-i:

Sección de los conductores	2x0,75/2x1,75 mm²
Norma	DIN VDE 0295, clase 6
Terminal AS-I +	Marron
Terminal AS-I -	Azul
Color del cable de datos 30 VDC	Amarillo
Color cable alimentación auxiliar 24 VDC	Negro
Color cable alimentación Auxiliar 230 VAC	Rojo
Tensión de servicio	300V

Fuente: “comunicaciones industriales guía práctica Pg 66”

El cable AS-i se ha diseñado como cable bifilar engomado, plano y flexible, sin trenzar ni apantallar, de 2X1.5mm². El cable está envuelto por una cubierta mecánicamente diseñada para que solo encaje en una posición evitando que se puedan conectar estaciones con la polaridad incorrecta.

El cable plano amarillo es el estándar, su geometría es fija y asimétrica; se encarga de transmitir los datos de toda la red y la alimentación a los dispositivos de tipo digital binario conectados en la misma.

Para los actuadores se necesita una alimentación auxiliar (tensión auxiliar de 24 V DC o 230 V AC), para el cable de alimentación auxiliar a 24 V DC se utiliza un cable de color negro, y para el cable de alimentación auxiliar a 230 V AC se utiliza el mismo cable pero en color rojo.

El cable AS-i es autocicatrizante y está diseñado para que unas tomas especiales (llamadas tipo vampiro) perforen el recubrimiento del cable y la cubierta aislante de los dos conductores, realizando una conexión eléctrica segura. Los agujeros producidos por las cuchillas de contacto en el revestimiento de goma del cable se cierran por sí mismos (autocicatrizante), restableciendo el grado de protección IP67,

En el cable AS-i debemos tener en cuenta las siguientes características básicas
Hasta 100m de longitud máxima entre la alimentación y el esclavo más alejado ,si
no hay repetidores (máximo permitido:3v de caída de tensión en 100m)
31 o 62 Esclavos, según versión, conectados a un mismo segmento de cable.

2.6.4 Fuente de alimentación AS-i: La fuente de alimentación AS-i, permite transmitir datos y suministrar energía a los sensores conectados en la red. Para ello, los datos transmitidos en la red AS-Interface se envían en forma de pulsos, también se encarga de modular la tensión continua en la red y no necesita conexión a tierra debido a que el bus está comprendido dentro de las tensiones de seguridad muy bajas (IEC 742-1).

La fuente de alimentación AS-i distribuye una tensión de 30V DC y hasta 8A. Tiene capacidad de suministrar una alimentación adicional de 24V DC, si se requiere un mayor consumo de energía, para este caso se debe utilizar un cable AS-i de color negro.

La fuente de alimentación AS-i se puede dividir en dos partes:

Alimentación: 30V DC, marcados como AS-i+ y AS-i-, aislada a tierra, con alimentación auxiliar de 24V por separado.

Filtro: El sistema de filtrado de la fuente permite la transmisión de datos y el suministro de corriente al bus.

Las fuentes de alimentación están protegidas contra sobretensiones, cortocircuitos y sobrecargas, pudiéndose colocar en cualquier punto de la red, pero se recomienda instalarla cerca de los Esclavos de mayor consumo; para Esclavos con consumo elevado (mayor que 100mA) se debe alimentar de forma independiente con el cable AS-i de color negro.

Figura 18: Fuente AS-i



Fuente:<http://www.construnario.com/catalogo/schneider-electric-espana-sa/>

2.7 COMPONENTES ADICIONALES

Existen algunos componentes adicionales en la red AS-i tales como: Repetidores Pasarelas y Dispositivo direccionador, que a continuación se presentan

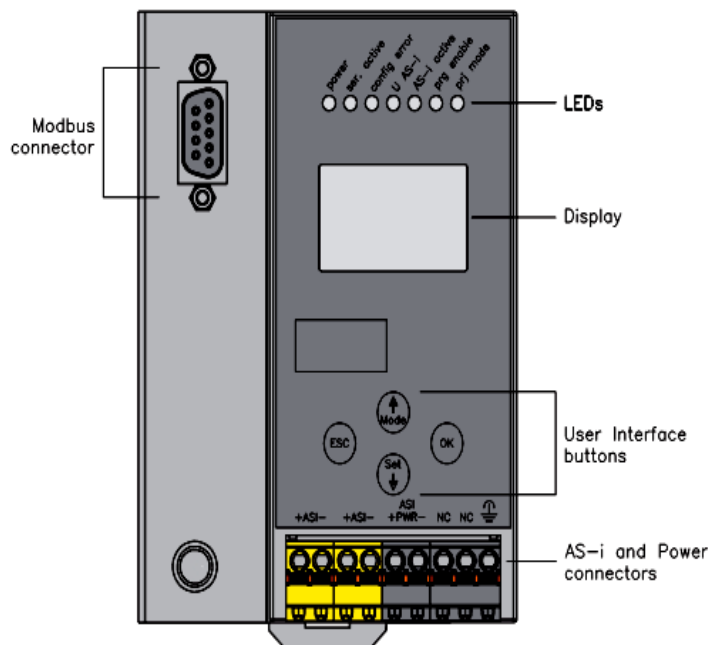
2.7.1 Repetidores: La red AS-Interface funciona sin problemas hasta una longitud de 100 metros, distancia limitada por las especificaciones eléctricas. En caso de que la instalación necesite más de 100 metros se necesita un módulo que regenere la señal del bus, a este módulo se le llama repetidor. Al añadir un repetidor este aísla eléctricamente los dos segmentos de la red, por este motivo

es necesario colocar una fuente de alimentación AS-i por cada segmento de la red.

Al colocar cualquier elemento activo de transmisión en una red AS-i, se añade un retardo de propagación de la señal. En el caso de los repetidores estos añaden al tiempo de respuesta de los esclavos 13.2 (microsegundos), por este efecto un sistema con más de dos repetidores en serie provocara un fallo de time-out por parte del maestro al tardar la respuesta del esclavo más de lo permitido.

2.7.2 Pasarelas (Gateway): En general para la automatización de procesos complejos, la red AS-Interface se puede conectar a un sistema de bus superior (por ejemplo, modbus, ethernet, CANopen). Ver figura 19

Figura 19: Gateway ASI-MBG-SS-BW1641 Turck



Fuente: <http://www.clrwtr.com/pdf/turck/turck-as-interface-masters-gateways.pdf>

Los gateways también pueden conectar redes del mismo protocolo de comunicación (ejemplo, dos redes AS-i, dos redes Modbus), consiguiendo así tantas redes del mismo protocolo como número de pasarelas nos permita el bus. En la figura 19 se presenta un Gateway ASI-MBG-SS-BW1641 que sirve para conectar al bus superior Modbus

2.7.3 Dispositivo direccionador: Antes de poner en marcha la red AS-Interface hay que asignar todas las direcciones de los elementos. Esto se puede realizar en modo OFFLINE con la ayuda de un dispositivo de programación, o en modo ONLINE por medio del software de programación.

Figura 20: Direccionador AS-Interface



Fuente: <http://homepage.cem.itesm.mx/vlopez/as-interface.htm>

En algunos esclavos, también se puede realizar después de su montaje en la red, por medio del dispositivo de direccionamiento.

Las direcciones de los esclavos pueden tener un valor entre 1 y 31 o entre 1A y 31A, y 1B y 31B, en caso de utilizar el perfil ampliado de la versión ASi 2.1. Cualquier esclavo nuevo, tiene por defecto la dirección 0. La asignación de las direcciones es totalmente libre debido a que el programa no exige colocar los esclavos en una dirección determinada, se puede programar para que un esclavo tenga la dirección 21 o 28 y su funcionamiento es igual. También es indiferente el orden de los esclavos en la red, por ejemplo al esclavo con dirección 21 le puede seguir el esclavo con dirección 22 o con dirección 30. Ver figura 20

2.8 FUNCIONAMIENTO DE LA RED AS-INTERFACE

AS-I es un sistema de transmisión de información excepcionalmente fiable debido a la implementación de diferentes mecanismos de detección de errores que actúan antes de la reconstrucción del mensaje. En AS-i los datos son modulados sobre una señal continua, utilizando métodos manchester y no retorno a cero (NRZ) que ofrecen inmunidad a las interferencias electromagnéticas. En el bus AS-i las tramas llevan poca información pero el uso de técnicas como la codificación Manchester y las señales de tipo $(\sin)^2$ hacen que la transmisión sea muy eficiente. La velocidad de transmisión del sistema es de 53.3Kbits.

2.9 PASOS PARA LA TRANSMISIÓN DE LOS BITS

Codificación sin retorno a cero (NRZ)

Conversión a código manchester, se transmiten mediante modulación de pulsos Alternados (APM). El ancho de banda es reducido y la señal es de tipo $(\sin)^2$

2.10 HERRAMIENTAS DE DIAGNOSTICO

Las herramientas de diagnóstico sirven para suministrar información referente a:

Esclavos con fallas de funcionamiento

Transitorios provocados por causas externas como: sobrecargas, ruidos, interferencias.

No detención de Esclavos

Falta de alimentación del bus

Errores de programación de la red.

La red AS-i funciona óptimamente en condiciones difíciles debido a:

La fuente de alimentación AS-i tiene protecciones contra sobretensiones, cortocircuitos y caídas de tensión. Si algún esclavo presenta un fallo, este no afecta los otros elementos de la red.

Un Esclavo deja de funcionar si su alimentación está por debajo de 14V.

Un Esclavo puede soportar caídas de tensión en tiempos muy cortos, menores a 1 milisegundo. El cambio de polaridad no destruye los esclavos, ya que estos tienen protección contra esta condición.

2.11 ARQUITECTURA DE LAS REDES AS-I:

La arquitectura, depende de dos elementos que se describen a continuación:

Módulos de ampliación

Número de esclavos

2.11.1 Módulos de ampliación: en una red AS-i, si la diseñamos para trabajar con sensores y/o actuadores debemos saber qué tipo de actuador o sensor utilizamos para colocar módulos de ampliación activos o pasivos. Si son sensores y/o actuadores convencionales como finales de carrera, electrovalvulas, que necesitan de un módulo de interface entonces tenemos que colocar módulos activos, los cuales pueden ser:

Unidireccionales: permiten 4 señales de entrada o 4 señales de salida.

Bidireccionales: tienen 4 señales de entrada y 4 señales de salida para un solo elemento, requiriendo una sola dirección del bus.

Si utilizamos sensores o actuadores con capacidad AS-i, tendremos que colocar módulos pasivos, en este caso se puede acceder a utilizar las características propias de este tipo de sensores, como posiciones de reposo, diagnósticos, tiempos de filtrado, etc.

2.11.2 Cantidad de esclavos: Desde la versión 2.1 de AS-i el número máximo de esclavos es de 62, a estos esclavos también se les llamo Esclavos A/B, Algunas características para la implementación de esclavos son:

Si los Esclavos tienen capacidad de comunicación, podremos colocar 62 como máximo, ocupando cada uno una dirección

Si los Esclavos son convencionales tendremos que instalarlos a módulos activos. En función del número de elementos a conectar, tendremos que colocar uno o más segmentos de red

2.12 PROGRAMACIÓN REDES AS-I

Para el funcionamiento de las redes AS-i es necesario tener maestros, esclavos, software de programación, fuentes, etc.

A continuación se describirá la comunicación con el maestro **TWDNOL10M3** de telemecanique utilizando el software de programación twido suite.

2.12.1 Descripción del módulo maestro AS-Interface TWDNOI10M3: el módulo maestro TWDNOI10M3 es un procesador de comunicaciones para el protocolo AS-i que puede ser instalado en los autómatas de la serie twido de telemecanique; trabaja con los software de programación twido de versiones mayores o iguales a la versión 2.0 y con la versión 2.1 del protocolo AS-i, en este trabajo de grado se utilizara el software versión 2.0 de twidosuite

2.12.1.1 Componentes del maestro AS-i: el módulo master AS-Interface TWDNOI10M3 integra estas funciones

Perfil M3: este perfil tiene las funcionalidades definidas por el estándar AS-Interface V2.1, pero no acepta los perfiles analógicos S7-4.

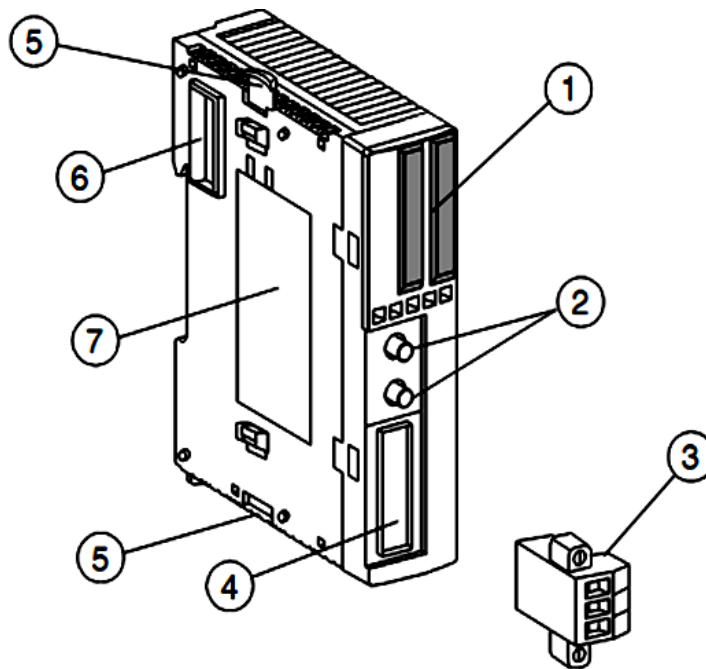
Acepta un canal AS-Interface por módulo (un solo cable)

Gestión de perfiles y parámetros

protección contra la inversión de polaridad en las entradas del bus

El módulo tiene en su parte frontal unos indicadores que informan del estado operativo de los Esclavos y estado operativo del mismo. Ver figura 21 y tabla 8.

Figura 21: Componentes maestro AS-i



Fuente: “Master modbus Controladores programables módulos de comunicación, Pg 28”

Tabla 8: Elementos del el modulo AS-i

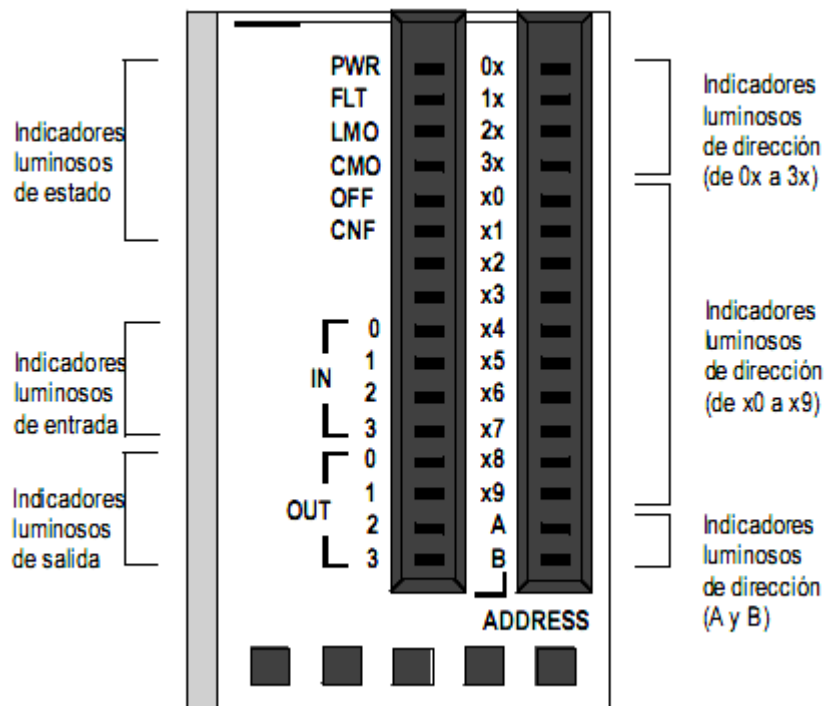
N.º	Componente	Descripción
1	Pantalla	● Indicadores luminosos de estado: indican el estado del bus AS-Interface. ● Indicadores luminosos de E/S: indican el estado de las E/S de un slave especificado por los indicadores de dirección. ● Indicadores luminosos de dirección: indican la dirección de los slaves.
2	Botones	Permiten seleccionar una dirección de un slave, así como cambiar el modo.
3	Terminal de usuario	Se conecta al cable AS-Interface.
4	Conector del cable AS-Interface	Permite instalar el terminal.
5	Botón de retención	Retiene/libera el módulo de un autómata.
6	Conector de ampliación	Permite llevar a cabo la conexión al módulo Twido así como la conexión de otro módulo de E/S.
7	Etiqueta del producto	Indica la referencia y las especificaciones del módulo.

Fuente: “Master modbus Controladores programables módulos de comunicación pg 28,”

2.12.1.2 panel de visualización

El panel de visualización es el encargado de indicar las direcciones de los esclavos con sus respectivas entradas y salidas, también se encarga de mostrar el modo de funcionamiento de la red AS-interface

Figura 22: Panel de visualización



Fuente: “master de modbus controladores programables módulos de comunicación Pg 35”

Tabla 9: Descripción de los indicadores luminosos de estado

Indicador luminoso	Estado	Descripción
PWR	●	Indica si el módulo AS-Interface está conectado.
	○	Indica que el módulo AS-Interface no recibe suficiente alimentación.
FLT	●	Indica que la configuración cargada en el master AS-Interface no es correcta o que se ha producido un error en el bus AS-Interface.
	○	Funcionamiento correcto del módulo.
LMO	○	Indica que el módulo no se encuentra en modo offline (el módulo permanece en modo online desde el arranque). Nota: Parpadea durante el arranque.
CMO	●	Indica que el módulo se encuentra en modo online.
OFF	●	Indica que el módulo se encuentra en el modo normal offline.
	○	Indica que el módulo se encuentra en otro modo de funcionamiento.
CNF	○	Este indicador ya no se utiliza. Nota: Parpadea durante el arranque.
○ Apagado ● Iluminado		

Fuente: “master de modbus controladores programables módulos de comunicación Pg 35”

2.12.1.3 modos de funcionamiento del maestro AS-interface: Los modos de funcionamiento del módulo AS-Interface pueden modificarse con los Botones PB1 y PB2 o con el software twidosuite. Los indicadores luminosos indican el estado en el que se encuentra el módulo AS-i. Ver figura 23

Modo protegido normal: No hay errores. Hay intercambio de información con los esclavos.

Modo protegido normal-offline: si se pulsa más de tres segundos el botón PB2 la comunicación con los esclavos se interrumpe, entonces se enciende OFF. Si se desea salir de este estado se debe pulsar más de tres segundos el botón PB2

Modo protegido normal-intercambio de datos desactivado: la entrada y salida en este modo solo puede realizarse con el software de programación twidosuite. En este modo no hay comunicación con los esclavos

Direccionamiento de los esclavos: las direcciones de los esclavos se indican mediante los diodos LED fijos: **x0, x1...x9 y 0x, 1x, 2x, 3x**, y el indicador de tipo **A/B**; cuando alguna dirección no se está usando con algún esclavo se indica con el LED intermitente. Ejemplo: si queremos observar un esclavo que está configurado en la dirección 16 A y tiene activadas las entradas 2 y 3 y las salidas 1y4 en el panel de control debemos observar el led 1x, led x6 y el led de A iluminados esto indica que está en la dirección 16A, de igual manera observamos que se iluminan el led 2 y 3 de IN indicando las entradas y el led 1 y 4 de OUT indicando las salidas del esclavo. Si tenemos conectados varios esclavos al módulo, estos se pueden visualizar en el panel oprimiendo los botones PB1 y PB2. Al oprimir los botones en el panel se observa la dirección de cada esclavo con sus diferentes entradas y salidas.

Figura 23: Visualización de los modos de funcionamiento del maestro AS-i

Modos de funcionamiento	PWR	FLT	LMO	CMO	OFF	CNF
Modo protegido normal	●	○	○	●	○	○
Modo protegido normal (offline)	●	●	○	●	●	○
Modo protegido normal (intercambio de datos desactivado)	●	●	○	●	○	○
○ Apagado ● Iluminado						

Fuente: “master de modbus controladores programables módulos de comunicación Pg. 36,

2.13 PROGRAMACIÓN DE LAS REDES AS-I CON TWIDOSUITE

El Software Twidosuite permite:

Configurar el bus en modo offline. Esto se realiza declarando los Esclavos y distribuyendo las diferentes direcciones en el bus

Configurar el bus para una red AS-i distinta a la que se estaba trabajando

Tener en cuenta los parámetros de los Esclavos

Controlar el estado del bus mediante objetos específicos llamados palabras y bits, para almacenar la información procedente o enviada al maestro AS-i

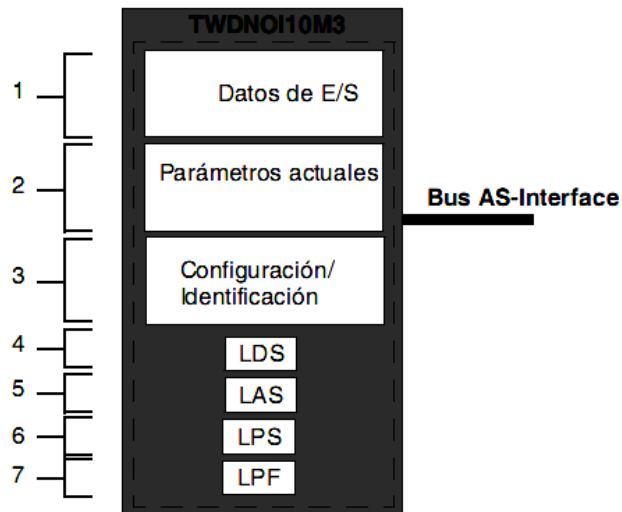
2.13.1 Estructura del maestro AS-Interfaces: El módulo AS-Interface con referencia TWDNOI10M3. Incluye esquemas de datos que sirven para gestionar listas de Esclavos e imágenes de datos de entrada/salida, almacenándolos en una memoria. Ver figura 24 y tabla 10

Tabla 10: Descripción y configuración de los elementos y parámetros

Dirección	Elemento	Descripción
1	Datos de E/S (IDI, ODI)	Imágenes de las 248 entradas y de las 186 salidas del bus AS-Interface V2.
2	Parámetros actuales (PI, PP)	Imagen de los parámetros de todos los slaves.
3	Configuración/Identificación (CDI, PCD)	Este campo contiene todos los códigos de E/S y los códigos de identificación de todos los slaves detectados.
4	LDS	Lista de todos los slaves detectados en el bus.
5	LAS	Lista de los slaves activados en el bus.
6	LPS	Lista de los slaves previstos en el bus y configurados mediante TwidoSuite.
7	LPF	Lista de los slaves que tienen un fallo de dispositivo periférico.

Fuente: “twidosuite V2.0, Pg 232”

Figura 24: Arquitectura del módulo TWDNOI10M3.



Fuente: "twidosuite V2.0, pg. 232"

2.13.2 Estructura de los equipos esclavos: muestran los elementos que sirven para gestionar la comunicación en el protocolo AS-i

Cada esclavo de direccionamiento estándar cuenta con:

4 bits de entrada/salida

4 bits de parametrización

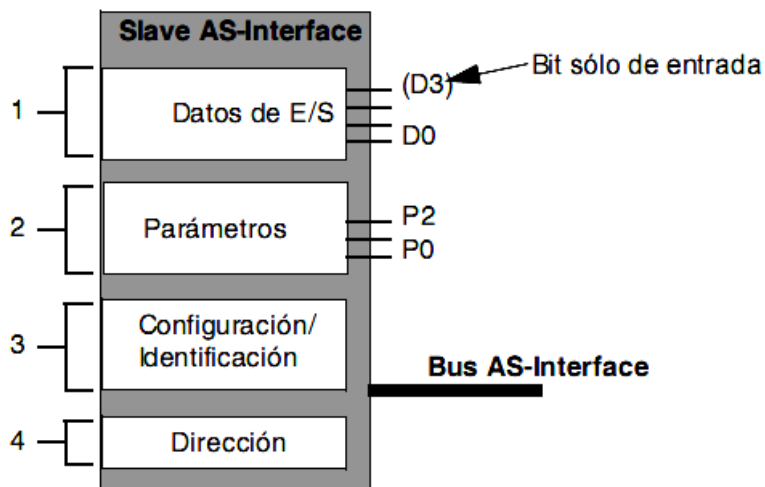
Cada esclavo de direccionamiento ampliado cuenta con:

4 bits de entrada/salida (el último bit se utiliza solo para las entradas)

3 bits de parametrización

Cada esclavo tiene una dirección, un perfil y un subperfil; este último es el encargado del intercambio de variables

Figura 25: Estructura de un Esclavo de direccionamiento ampliado:



Fuente: “twidosuite V2.0, pg 233”

Tabla 11: Descripción y configuración de los elementos y parámetros

Dirección	Elemento	Descripción
1	Datos de entrada/salida	El slave almacena los datos de entrada y los pone a disposición del master AS-Interface. El módulo master actualiza los datos de salida.
2	Parámetros	Los parámetros permiten controlar y conmutar los modos de funcionamiento internos del sensor
3	Configuración/Identificación	Este campo contiene: ● El código correspondiente a la configuración de las entradas/salidas (I/O) ● El código de identificación del slave (ID) ● Los subcódigos de identificación del slave (ID1 y ID2)
4	Dirección	Dirección física del slave.
Nota: Los parámetros de funcionamiento, dirección, datos de configuración y de identificación se almacenan en una memoria no volátil.		

”Fuente: “twidosuite V2.0, Pg 233”

“Tomado de Twidosuite V2.0, guía de programación: Ref. 35013228.02, 09/2007.
Pg. 231,232 y 233. Reformado por autores”

2.13.3 Pasos para instalar AS-Interface

Tabla 12: Resumen de pasos para instalar AS-interface

Modo	Fase	Descripción
Local	Declaración del módulo	Selección del emplazamiento del módulo master AS-Interface TWDNOI10M3 en el bus de ampliación.
	Configuración del canal del módulo	Selección de los modos "master".
	Declaración de los equipos slaves	Selección, para cada equipo, de: ● El número de emplazamiento en el bus. ● El tipo de slave de direccionamiento estándar o direccionamiento ampliado.
	Validación de los parámetros de configuración	Validación en el slave.
	Validación global de la aplicación	Validación de aplicación.
Local o conectado	Simbolización (opcional)	Simbolización de las variables asociadas a los equipos slaves.
	Programación	Programación de la función AS-Interface V2.
Conectado	Transferir	Transferencia de la aplicación en el PLC.
	Depuración	Depuración de la aplicación mediante: ● La ventana de AS-Interface que permite visualizar los slaves (dirección, parámetros) y asignarles las direcciones deseadas. ● Las pantallas de diagnóstico que permiten identificar los fallos.

Fuente: “Twidosuite V2.0, Pg 234”

Para que el protocolo de comunicación AS-interface funcione correctamente se debe tener en cuenta que:

Un módulo master AS-Interface en el bus se puede eliminar y configurar sin ningún problema, sin embargo una vez que se han declarado dos módulos master AS-Interface en el bus de expansión, TwidoSuite no permite declarar otro.

Antes de conectar el PC mediante el software twido al autómata, debemos asegurarnos con el direccionador de que no haya ningún esclavo presente físicamente en el bus en la dirección cero y que no haya dos esclavos presentes físicamente en la misma dirección.

2.13.4 Descripción de la ventana AS-interface en modo offline: Esta ventana permite visualizar y modificar los parámetros asociados al módulo maestro y a los esclavos AS-i en modo offline

Figura 26: ventana AS-i en modo offline

Esclavos std/A	Esclavos /B
	00
XAL-S2002E	01
	02
ASI20MT4I4OR	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14
	15
	16
	17
	18
	19
	20

Aplicar **Cancelar**

Esclavo 1/A Nombre: XAL-S2002E

Características permanentes

Perfil: IO B ID A ID1 7 ID2 E

Comentario: Caja c/puls. I Verde-0 Rojo

Parámetros permanentes

☒ Bits 0 ☒ Inutilizado 2 ☒ Inutilizado
☐ Decimal 1 ☒ Inutilizado

Entradas/Salidas

Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo
1	%IA1.1A.2		1	%QA1.1A.0	
2	%IA1.1A.3		2	%QA1.1A.1	

Modo Master

Intercambio de datos activado ☐ Parada red ☐ Direccionamiento automático ☐

Fuente: “software Twidosuite”

Tabla 13: Características de la ventana AS-i en modo OFFLINE

Bloques	Descripción
Configuración de AS-Interface	Imagen del bus deseada por el usuario: visualización de slaves con configuración de direcciones estándar y ampliadas previstos en el bus. Mueva el cursor en sentido descendente por la barra vertical para acceder a las direcciones siguientes. Las direcciones inaccesibles corresponden a las direcciones que no se pueden utilizar para configurar un slave. Si, por ejemplo, se declara un nuevo slave con configuración de dirección estándar con la dirección 1A, la dirección 1B se volverá inaccesible automáticamente.
Slave xxA/B	Configuración del slave seleccionado: ● Características: código E/S, código ID, códigos ID1 e ID2 (perfiles) y comentarios sobre el slave. ● Parámetros: lista de parámetros (modificable), en forma binaria (cuatro casillas de verificación) o decimal (una casilla) a elección del usuario. ● Entradas/salidas: lista de las E/S disponibles y sus direcciones respectivas.

Fuente: “twidosuite V2.0, Pg 235”

2.13.5 Descripción de la ventana AS-interface en modo online: La ventana de AS-Interface en modo online proporciona una imagen del bus que incluye:

lista de esclavos introducidos durante la configuración con su nombre y la lista de los esclavos desconocidos, en una misma lista aparecen los esclavos conocidos y los esclavos desconocidos

El estado del módulo AS-Interface y de los esclavos.

La imagen del perfil, los parámetros y los valores de las salidas y entradas de los esclavos reconocidos.

obtener un diagnóstico de los esclavos erróneos

asignarles a los esclavos las direcciones deseadas durante la primera depuración.

Figura 27: Ventana AS-interface en modo online

Slaves std/

Slaves /B

Slave 1A

00

01

02

03

04

05

06

07

08

09

10

11

12

13

14

15

16

17

18

19

20

ASI20MT4IE

INOUT24/12

Desconocido

WXA36

Aplicar

Cancelar

Slave 1/A

Nombre: Slave 1A

Características permanentes

Perfil

E/S

F

ID

F

ID1

F

ID2

F

Comentario

Comentario

Parámetros permanentes

Bits

0

✓

Parámetro 1

2

✓

Parámetro 3

Decimal

1

✓

Parámetro 2

3

✓

Parámetro 4

Entradas/salidas

Entrada

Dirección

Símbolo

Salida

Dirección

Símbolo

Modo Master

Bus AS-Interface

Configuración OK

OFF

Direccionamiento automático posible

OFF

Slaves direc 0 detectado

OFF

Corte de alimentación

OFF

Slaves OK

ON

Modo protegido

OFF

Direccionamiento automático activo

ON

Parada red

OFF

Fuente: “twidosuite V2.0, guía de programación 09/2007. Pg 244”

La siguiente tabla resume las características de esta ventana en tres bloques de información en modo online

Tabla 14: características de la ventana AS-i en modo online

Bloques	Descripción
Configuración de AS-Interface	Imagen del bus deseada por el usuario: visualización de slaves con configuración de direcciones estándar y ampliadas previstos en el bus. Mueva el cursor en sentido descendente por la barra vertical para acceder a las direcciones siguientes. Las direcciones inaccesibles corresponden a las direcciones que no se pueden utilizar para configurar un slave. Si, por ejemplo, se declara un nuevo slave con configuración de dirección estándar con la dirección 1A, la dirección 1B se volverá inaccesible automáticamente.

Tabla 14: continuación

Slave xxA/B	Configuración del slave seleccionado: ● Características: código I/O , código ID, códigos ID1 e ID2 (perfiles) y comentarios sobre el slave. ● Parámetros: lista de parámetros (modificable), en forma binaria (cuatro casillas de verificación) o decimal (una casilla) a elección del usuario. ● Entradas/salidas: lista de las I/O disponibles y sus direcciones respectivas.
Modo Master	Activación o desactivación posible de las dos funciones disponibles para el módulo AS-Interface (direccionamiento automático, por ejemplo). El modo "Direccionamiento automático" está seleccionado por defecto. Nota: La función de activación de intercambio de datos aún no está disponible.

Fuente: "twidosuite V2.0, guía de programación 09/2007. Pg 245"

2.13.6 Descripción de errores:

2.13.6.1 Error de perfil: cuando el perfil previsto por el usuario al configurar una dirección determinada no se corresponde con el perfil real detectado en dicha dirección del bus

2.13.6.2 Esclavo no proyectado: cuando el bus detecta un nuevo esclavo no previsto en la configuración aparece un indicador rojo en esta dirección y el nombre del esclavo aparece como "Desconocido"

2.13.6.3 Esclavo no detectado: cuando a un esclavo se le asigna un perfil pero el bus no detecta ningún esclavo.

2.13.6.4 Direccionamiento de las entradas y salidas de los esclavos:

El direccionamiento empleado en esta sección es el que se utiliza para programar en ladder empleando en el software de twidosuite

Figura 28: Como se direccional las entradas y salidas de los esclavos AS-i

%	IA, QA, IWA, QWA	x	.	n	.	i
Símbolo	Tipo de objeto	Ampliación dirección del módulo		dirección del slave		Canal n.º

Fuente: “twidosuite V2.0, Pg 255”

La siguiente tabla especifica los valores de los objetos esclavo AS-interface

Tabla 15: Direccionamiento de las entradas y salidas de los esclavos AS-i

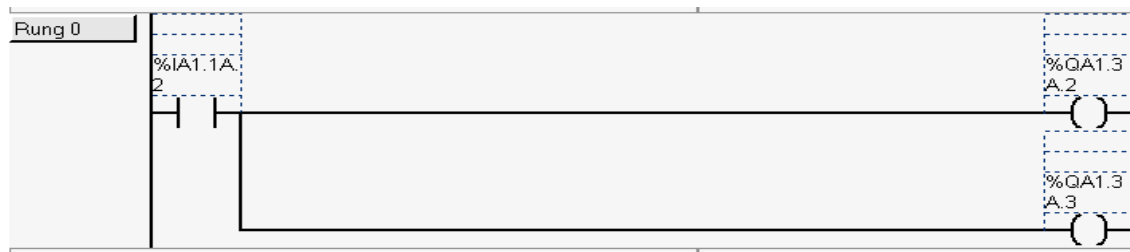
Componente	Valores	Comentario
IA	-	Imagen de la entrada binaria física del slave.
QA	-	Imagen de la salida binaria física del slave.
IWA	-	Imagen de la entrada física analógica del slave.
QWA	-	Imagen de la salida física analógica del slave.
x	De 1 a 7	Dirección del módulo AS-Interface en el bus de ampliación.
n	De 0A a 31B	El slot 0 no se puede configurar.
i	De 0 a 3	-

Fuente: “twidosuite V2.0, Pg 255”

EJEMPLO:

La siguiente dirección %IA1.1A.2 es la empleada para usar en el programa hecho en TwidoSuite, la cual es una entrada de una botonera XAL-S2002E en una red AS-i. El %IA1 representa una entrada de un esclavo conectado a la red AS-i del modulo maestro AS-i ubicado en la posición 1 del bus de ampliación, 1A representa que al esclavo le asignaron esta dirección en la ventana de configuración y el número 2 denota una de las dos entradas o contactos de la botonera, en este caso es un contacto normalmente abierto que se representa físicamente por un botón de color verde. Ver tabla 15

Diagrama ladder:



Fuente: "software Twidosuite"

3. PROTOCOLO DE COMUNICACIÓN MODBUS SERIAL

Fue diseñado en 1979 por Modicon para su gama de controladores lógicos programables (PLCs). Modbus es un protocolo de comunicación industrial de tipo maestro/esclavo, que permite a un único maestro solicitar respuestas a un esclavo de máximo 32 esclavos sin repetidores o de 247 esclavos con repetidores, este protocolo también permite realizar acciones dependiendo de las respuestas de los esclavos. El maestro puede enviar una solicitud a un esclavo o a todos los esclavos simultáneamente. Los esclavos devuelven una respuesta a las solicitudes que se les envían individualmente. Se debe aclarar que el maestro sólo puede iniciar una solicitud Modbus al tiempo mientras la respuesta a dicha solicitud no sea enviada al maestro por parte del esclavo o en su defecto haya expirado el “time out”. Existen tres tipos de implementación modbus conocidas:

Modbus con transmisión serial asíncrona.

Modbus sobreTCP/IP

Modbus plus

En este aparte del documento se presenta el protocolo modbus serial.

Modbus está aceptado por la IEC como una especificación publica disponible bajo la designación IEC PAS 6203.

3.1 CARACTERISTICAS DE MODBUS SERIAL

3.1.1 Topología: La topología que implementa el protocolo de comunicación modbus es en bus “ver 1.3.2.1 de este trabajo de grado”.

3.1.2 Modo de transmisión: el protocolo modbus se transmite en banda base, es decir las señales son transmitidas en su frecuencia original sin modular, y se codifican en códigos banda base como: no retorno a cero (NRZ), retorno a cero (RZ), codificación diferencial, códigos Manchester

3.1.3 Tipo de comunicación empleada: se emplea comunicación asíncrona “ver sección 1.4.3.1”. La velocidad de transmisión de datos para el protocolo modbus empleando comunicación asíncrona puede ir desde los 75 baudios a 64000 baudios, pero el estándar del protocolo modbus exige que todos los dispositivos que emplee este protocolo tengan una velocidad de transmisión de 19200 baudios. Con este tipo de comunicación se puede alcanzar una distancia máxima de 1200 m sin repetidores.

3.2 ACCESO AL MEDIO

Para acceder al medio se emplea la técnica Maestro-Esclavo, en la cual se emplea un dispositivo maestro y uno o varios dispositivos esclavos; el dispositivo maestro será el único que podrá iniciar las transacciones¹⁶ para que los esclavos respondan con alguna información o realice la acción requerida a la petición del maestro. Existen dos tipos de transacciones:

3.2.1 Consulta/Respuesta (Query-response): Es un intercambio punto a punto entre el maestro y un esclavo, esto puede ser un simple reconocimiento o acknowledge

¹⁶ Transacción: es una solicitud enviada por el maestro al esclavo y su respectiva respuesta del esclavo al maestro cuando el mensaje no es de difusión

3.2.2 Difusión (broadcast/no response): El maestro envía un mensaje a todos los esclavos de la red, este tipo de mensaje no tiene respuesta por parte de los esclavos y se emplea para mandar datos comunes de configuración.

3.3 CAPA DE APLICACIÓN MODBUS

Figura 29: Capa de aplicación de modbus



Fuente: http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf

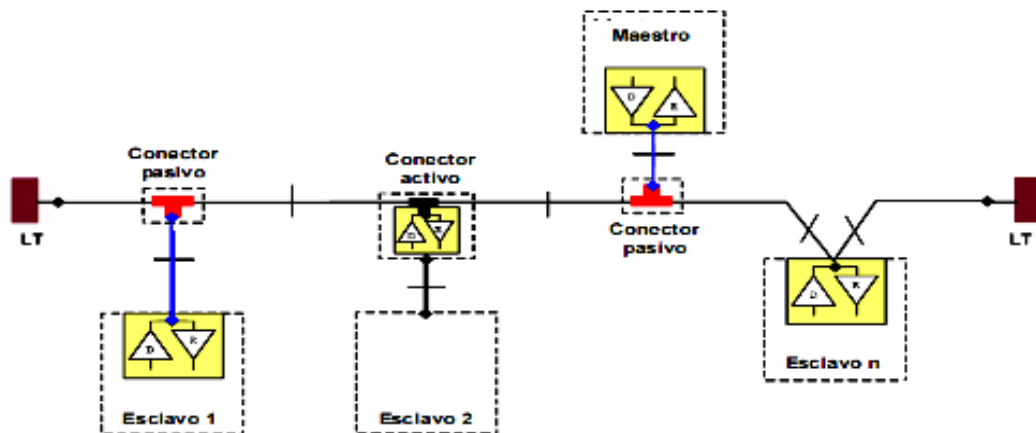
Modbus es un protocolo ubicado en la capa de usuario correspondiente al nivel 7 del modelo de OSI (*Open Sistem Interconnection*).

En la figura 29 se aprecia la capa de protocolo de aplicación modbus con sus respectivos capas de protocolo adyacentes dependiendo de la capa física usada como: modbus sobre TCP/IP, (Ethernet) y Modbus estándar (*EIA/TIA 232 o 485*).

3.4 CAPA FISICA:

El protocolo de comunicación Modbus utiliza el estándar EIA/TIA 485, también conocido como RS485 con el propósito de establecer enlaces halfduplex y multipunto en una configuración de dos hilos. Asimismo este protocolo cuenta con la posibilidad de implementar una configuración de cuatro hilos por medio de la interfaz RS422, o una comunicación serial común con RS232.

Figura 30: Tipos de conexión maestro y esclavos en la red modbus



Fuente: Trabajo de grado: "diseño e implementación de una solución de compatibilidad entre dispositivos interconectados bajo protocolo modbus Pg 36"

En la red Modbus el maestro y el (los) esclavo(s) se comunican siempre por medio de una línea serial pasiva, es decir por una línea que solo lleva señales eléctricas, de esta forma todos los dispositivos se encuentran conectados a un cable principal constituido por 3 conductores de los cuales 2 forman un par trenzado

(configuración de 2 hilos) por el cual son transmitidos los datos en ambas direcciones. La función del tercer cable se explica en la siguiente pagina (sección 3.4.1)

Cada equipo se puede conectar de tres formas distintas Ver figura 30:

Directamente, con un grupo de conductores de entrada y otro grupo de salida.

Con conector pasivo, se enlaza en un punto del cable principal y forma una derivación al esclavo o maestro.

Con conector activo, se enlaza en un punto del cable, ejecuta las labores de un transceiver y deriva hacia el equipo las señales ya procesadas.

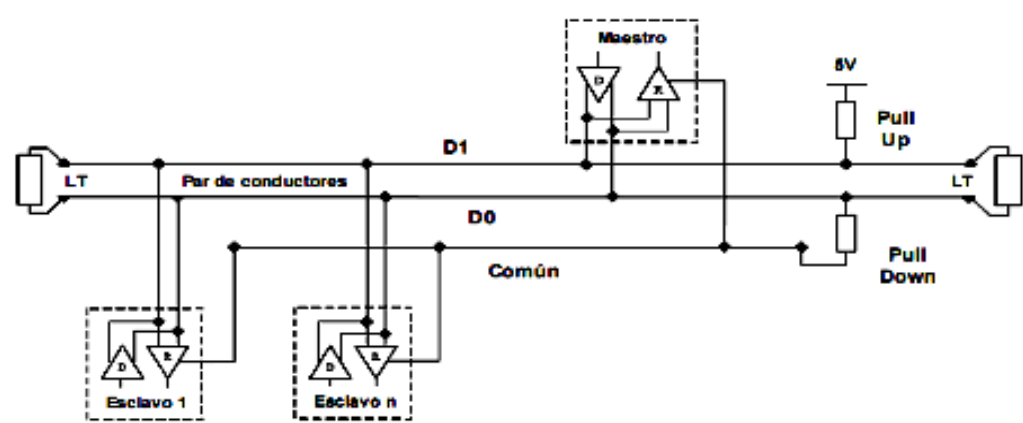
NOTA: los maestros y los esclavos modbus con que cuenta la E3T incluyen el transceiver, por lo que todos los conectores modbus usados en este proyecto de grado son pasivos

En la figura 30 se observa en cada extremo del cable principal las resistencias de terminación de línea (LT), que son de carácter obligatorio y tienen un valor de resistencia de 120 ohms y 0.5 wattios. Estas resistencias tienen como finalidad alcanzar la máxima transmisión de potencia de y hacia cada equipo conectado al bus y minimizar el efecto de reflexión en una línea de transmisión mejorando la continuidad de la impedancia en la línea de transmisión.

3.4.1 Configuración de dos hilos: La configuración de dos hilos se hace siguiendo el estándar EIA/TIA 485 también conocido como RS 485. En un bus 2 hilos (D1 y D0) de este tipo existe siempre un tercer conductor que interconecta todos los equipos y establece una referencia eléctrica, de allí su nombre de común.

En el bus sólo un dispositivo a la vez tiene derecho de transmitir y los demás esperan su turno.

Figura 31: Topología a dos hilos



Fuente trabajo de grado “diseño e implementación de una solución de compatibilidad entre dispositivos interconectados bajo protocolo modbus Pg 36”

Tabla 16: configuración a dos hilos

Señales necesarias		Uso	Nombre EIA/TIA 485	Descripción
Cable principal	Cable de derivación			
D1	D1	In/Out	B/B'	Terminal 1 del transceiver, Voltaje V1 (V1 > V0 para estado binario 1 [OFF])
D0	D0	In/Out	A/A'	Terminal 0 del transceiver, Voltaje V0 (V0 > V1 para estado binario 0 [ON])
Común	Común	-	C/C'	Señal común y opcional para alimentación

Fuente: Trabajo de grado “diseño e implementación de una solución de compatibilidad entre dispositivos interconectados bajo protocolo modbus Pg 36”

3.5 MODOS DE TRANSMISIÒN

Modbus utiliza dos tipos de transmisión para definir el mensaje transmitido por la red y determinar el contenido de las tramas que serán transmitidas a través de los conductores. Estos modos de transmisión son: ASCII y RTU.

3.5.1 Modo RTU: es el modo más comercial implementado en ingeniería debido a que tiene una mayor densidad de caracteres permitiendo tener un mayor rendimiento que el modo ASCII utilizando la misma tasa de transmisión;

En modo RTU cada 8 bits (1byte) de datos de información transmitida en el campo de un mensaje contiene dos caracteres hexadecimal de 4 bits, a este byte se le agrega 3 bits más para ser empaquetado en un formato de 11 bits (formato de secuencia de bits). El formato secuencia de bits en modo RTU es:

1 bit de inicio,

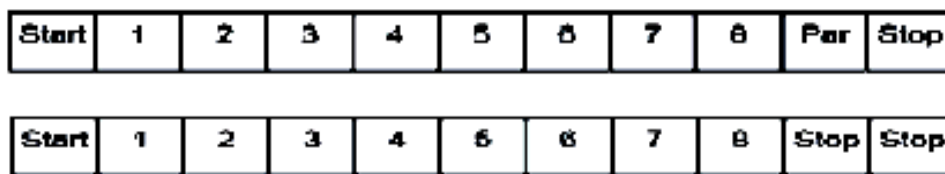
8 bits de datos (el bit menos significativo se envía primero),

1 bit de paridad

1 bit de parada.

Para el bit de paridad existen dos opciones: impar o par, si se utiliza la no paridad no se utiliza el bit de paridad y se utiliza dos bits de parada. Ver la siguiente figura.

Figura 32: formato de Secuencia de bits en modo RTU.



Fuente: "MODBUS over Serial Line, Pg. 12"

El protocolo de comunicación modbus exige que los equipos que se comuniquen con este protocolo por defecto tengan configurado la opción par. Los fabricantes y desarrolladores de equipos con protocolo modbus recomiendan disponer de la opción de no paridad con el fin de asegurar mayor compatibilidad entre sus marcas, de modo tal se tienen dos bits de parada.

Es importante resaltar que cada byte de datos necesita exactamente de un formato de secuencia de bits para ser transmitido (11bits) y estos a su vez conforman el campo de un mensaje el cual se empaqueta en un formato llamado trama. Una trama está definida por la dirección de esclavo (1byte), el código de función (1byte), los datos del mensaje (0-252 bytes) y el chequeo de redundancia cíclica CRC (2 bytes), ver figura 33

Figura 33: Descripción de la trama modbus en modo RTU

Slave Address	Function Code	Data	CRC
1 byte	1 byte	0 up to 252 byte(s)	2 bytes CRC Low CRC Hi

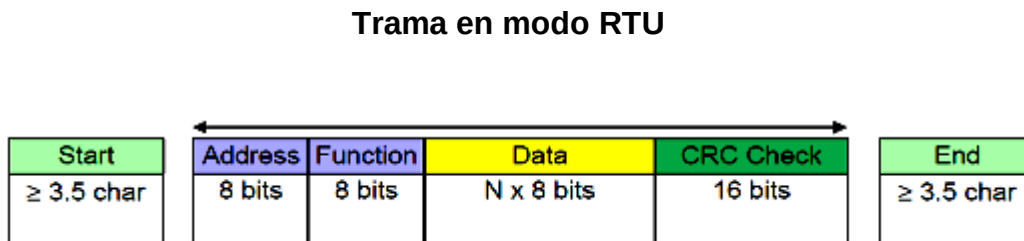
Fuente: “MODBUS over Serial Line, pg 13”

Una trama comienza y termina con un intervalo de silencio de al menos 3.5 carácter de un byte¹⁷, esto permite a los dispositivos que reciben una nueva trama comenzar el inicio del mensaje y saber cuando el mensaje se completa. El envío parcial de mensajes deben ser detectados y los errores se deben establecer como resultados.

¹⁷ Carácter de byte: es el tiempo que gastan 8 bits para ser transmitidos, ejemplo si la velocidad de transmisión es de 19200 bits/segundos el tiempo de transmisión de un carácter de byte será 8/19200 (sg)

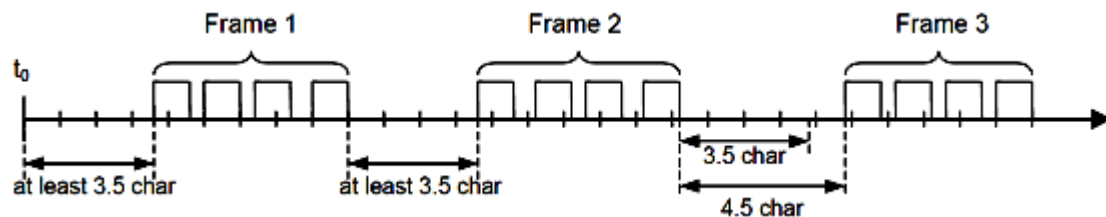
En el modo RTU, las tramas están separadas por un intervalo de silencio de al menos 3,5 el carácter de un byte.

Figura 34: Trama con principio y final en modo RTU:



Fuente: “MODBUS over Serial Line, pg 13”

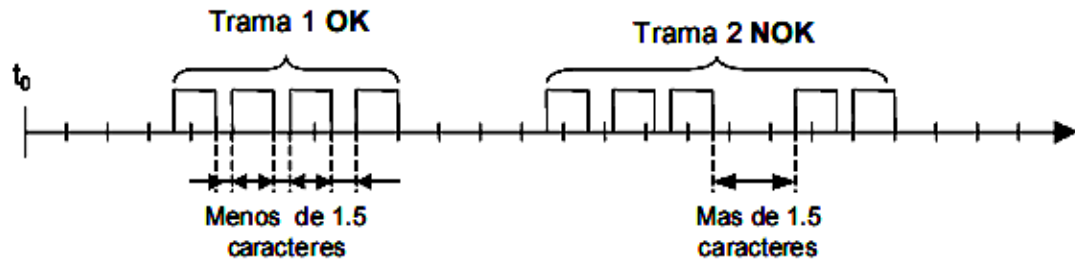
Figura 35: Tiempos entre tramas



Fuente: “MODBUS over Serial Line, pg 13”

Cada mensaje dentro del protocolo modbus en modo RTU debe ser enviado como una cadena ininterrumpida de secuencias de bits separados a lo sumo por un tiempo equivalente al empleado en la transmisión de 1.5 carácter de un byte, cuando se supera este límite el mensaje es declarado incompleto y tiene que ser descartado por el receptor. Ver la siguiente figura

Figura 36: Tiempo entre secuencias de bits

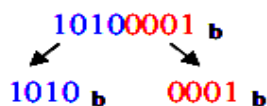


Fuente: Trabajo de grado “diseño e implementación de una solución de compatibilidad entre dispositivos interconectados bajo protocolo modbus Pg 30”

3.5.2 Modo ASCII: En el modo ASCII cada byte de información es transmitido como dos caracteres ASCII, la gran ventaja de ASCII es el tiempo que permite entre caracteres hexadecimales es mayor al tiempo que permite el modo RTU, sin que existan errores de transmisión.

Este modo se basa en el código estándar americano para Intercambio de Información “ASCII”, el cual es un código alfanumérico universal, usado por las computadoras y equipos electrónicos.

La forma de codificar cada byte de información en ASCII se explica a continuación: Cada byte de información del mensaje se divide en dos nibbles (cuatro bits).



Cada nibble es convertido en un carácter en código ASCII sumándole

30 h si el nibble está entre 0h y 9h,

37h si el nibble está entre Ah y Fh,

$1010 = A_h \xrightarrow{+37_h} 41_h$ En Código ASCII **A**
 $0001 = 1_h \xrightarrow{+30_h} 31_h$ En Código ASCII **1**

Un código ASCII requiere 7 bits de datos para ser codificado en código binario. En la tabla de la figura 17 se pueden observar las siguientes equivalencias:

A (hexadecimal) = 10 (decimal) =LF (código ASCII)
 1 (hexadecimal)= 1 (decimal) =SOH (código ASCII)
 37 (hexadecimal) = 55 (decimal) = 7 (código ASCII)
 30(hexadecimal) = 48 (decimal) =0 (código ASCII)

Tabla 17 caracteres AS-I

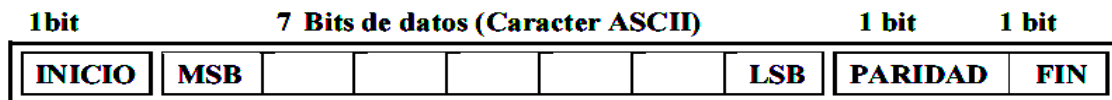
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENO	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL
	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127

Fuente: Trabajo de grado "Modbus. Monitoreo de la red empleando labview, pg165"

En la tabla 17 el valor hexadecimal se obtiene interceptando la fila con la columna. Por ejemplo, el carácter "B" del código ASCII está en fila 4 columna 2, luego su valor hexadecimal es 42=42h. El valor decimal se localiza debajo de cada carácter, ejemplo para "B" es 66 d.

Después de codificar la información en ASCII se empaqueta en un formato de 10 bits llamado formato de secuencia de bits para ser enviado por el puerto serie. El carácter "serie ASCII" consiste en 1 bit de inicio, 7 bits de datos (el bit menos significativo se envía primero), 1 bit de paridad y 1 bit de parada. En este modo también hay dos opciones para el bit de paridad: par o impar y otra opción de no paridad, en este caso se reemplazara el bit de paridad por un bit de parada o fin adicional, quedando el formato de secuencia de bits con dos bits de parada o fin. En la figura 37 se presenta el formato de secuencia de bits en modo ASCII

Figura 37: Formato de secuencia de bits en modo ASCII

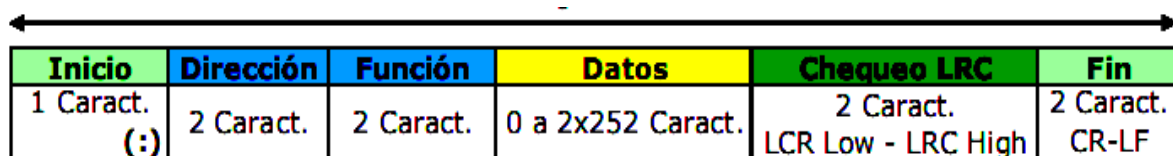


Fuente: Trabajo de grado "Modbus. Monitoreo de la red empleando labview Pg 37"

En el modo ASCII cada byte de datos necesita de 2 caracteres para ser transmitido (14 bits, más los de inicio, paridad y parada), debido a la codificación (hexadecimal –ASCII). Por esta razón es menos eficiente en tasa de bits transmitidos que el modo RTU.

La trama de información del modo ASCII está compuesta por un campo de inicio (1 carácter), la dirección de esclavo (2 caracteres), el código de función (2 caracteres), los datos (2x252=504 caracteres), el chequeo de redundancia longitudinal (2 caracteres), un campo de finalización (2 caracteres).

Figura 38: Trama modbus en modo ASCII



Fuente: Trabajo de grado “diseño e implementación de una solución de compatibilidad entre dispositivos interconectados bajo protocolo modbus, Pg 33”

En la figura anterior se observa que la trama inicia con el carácter dos puntos (“:”) y que su final está compuesto por los caracteres CR y LF

3.6 TRANSACCIONES EN MODBUS:

Las transacciones en modbus se realizan a través de las consultas que hace el dispositivo maestro a sus esclavos, los cuales responden suministrando la información o realizando la acción suministrada por el maestro.

Un maestro puede leer el estado de un grupo de salidas y entradas o puede leer o escribir los datos contenidos en los registros.

Cuando el esclavo responde al maestro, este utiliza el código de la función para indicar una respuesta normal libre de errores o alguna respuesta de error llamada respuesta de excepción.

Las transacciones están basadas en consultas y respuestas, las cuales se detallan a continuación:

3.6.1 Consulta: El código de función le indica al esclavo que tipo de operación debe realizar. Por ejemplo el código de operación 04 solita al esclavo la lectura de registros de palabras de entrada y este responde con su contenido.

La información debe indicarle al esclavo en que registro comienza la petición y cuantos registros se quieren leer. Para garantizar la integridad del contenido del mensaje en el esclavo existe un método de verificación de errores.

3.6.2 Respuesta: Cuando el esclavo envía una respuesta normal, el código de función de respuesta es una réplica del código de función de la consulta, y la información de los esclavos es enviada a través de los bytes de datos que contiene valores o el estado de registro, si se detecta un error el código de función se modifica para indicar que la respuesta es un mensaje de error y que el código de la información que describe el error viene en los bytes de información o de datos. La verificación del campo de error permite que el maestro pueda confirmar que el contenido del mensaje es válido.

3.7 FUNCIONES MODBUS:

La programación de las principales funciones para la implementación del protocolo modbus están descritas en los documentos disponibles al público por la organización Modbus-IDA. Las funciones se clasifican en tres principales grupos:

Funciones de acceso de bits

Funciones de acceso a palabras

Funciones de diagnóstico y configuración

Tabla 18: principales funciones de modbus

				Código de función		Hex
				Código	Sub-código	
Acceso a datos	Acceso a bits	Entradas físicas discretas, Bits internos o Bobinas físicas	Leer entradas discretas	02		02
			Leer múltiples bobinas	01		01
			Escribir bobina sencilla	05		05
			Escribir múltiples bobinas	15		0F
	Acceso a 16 bits	Entradas físicas de registros, registros internos o salidas físicas de registros	Leer registro de entrada	04		04
			Leer registros en memoria	03		03
			Escribir registro sencillo	06		06
			Escribir múltiples registros	16		10
			Leer/Escribir múltiples registros	23		17
			Enmascarar registros escritos	22		16
			Leer cola FIFO (First input First output)	24		18
	Acceso a archivos guardados		Leer archivo guardado	20	6	14
			Escribir archivo guardado	21	6	15
	Diagnóstico		Leer estado de error	07		07
			Diagnóstico	08	00-18 20	08
			Obtener contador de eventos de comunicación	11		0B
			Obtener registros de eventos de comunicación	12		0C
			Reportar identificación de esclavo	17		11
			Reportar identificación de dispositivo	43	14	2B
			Otras		Interfase de transporte encapsulada	43
	Referencia general CANopen	43			13	2B

Fuente: Trabajo de grado “diseño e implementación de una solución de compatibilidad entre dispositivos bajo protocolo modbus PG 26”

3.7.1 Funciones de acceso de bits: para la lectura de entradas discretas y lectura o escrituras de salidas discretas

3.7.2 Funciones de acceso de palabra: para la lectura de registros de 16 bits de entradas y la lectura y la escritura de registros de salidas.

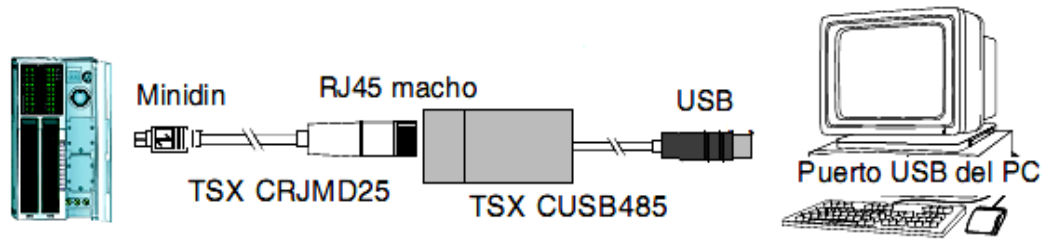
3.7.3 Funciones de diagnostico y configuración: se utiliza para configurar los parámetros del procesador y obtener información acerca de los eventos y errores ocurridos en el proceso de comunicación.

A continuación se presenta la tabla con las funciones más importantes de modbus

3.8 COMUNICACIÓN ENTRE EL SOFTWARE TWIDOSUITE Y PLC TWIDO

El autómatas Twido con Ref. TWDLCAE40DRF tiene un puerto terminal EIA RS485 integrado con un conector miniDIN hembra de 8 pines, con su propia fuente de alimentación interna de 5 Vdc, denominado por el fabricante como1. Este puerto está diseñado para la comunicación con el puerto USB del PC usando el software de programación TwidoSuite y el protocolo modbus. A través del puerto 1 y utilizando el protocolo modbus, el PLC Twido puede comunicarse con otros Twidos, con PLCs M340, PLCs Zelios y variadores de velocidad Altivar31 entre otros. La conexión entre el PC y el puerto 1 RS485 del autómatas Twido se realiza empleando los cables TSXCUSB485 y TSXCRJMD25 o a través de un módem de línea telefónica.

Figura 39: Conexión con cable USB



Fuente: "Twidosuite V2.0, Pg 78" Modificado por los autores"

3.9 CONFIGURACIÓN CAPA FÍSICA

Para comunicar el maestro Twido con referencia TWDLCAE40DRF y los cuatro esclavos de propiedad de la escuela E3T fue necesario implementar conectores y cables bajo el estándar de la capa física EIA/TIA 485.

Estos son los cuatro esclavos utilizados: PLC Twido TWDLCAE40DRF, PLC Modicon M340, variador de velocidad Altivar 31 y PLC zelio

Figura 40: Modicon M340



Fuente: "Manual unity pro, Pg 6

Figura 41: PLC Zelio



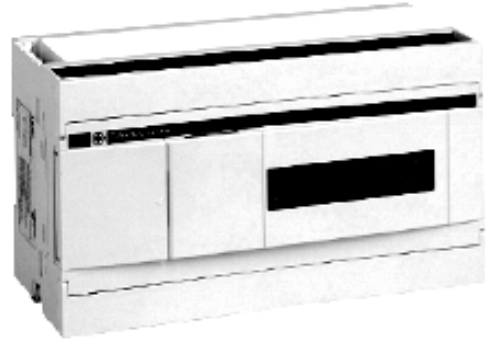
Fuente: "Relés programables Zelio logic Pg 1"

Figura 42: Altivar 31



Fuente: "Altivar 31 Guía de Programación, Pg 1"

Figura 43: Twido TWDLCAE40



Fuente: "Autómata programables Catálogo 2005, Pg 1"

3.9.1 Conexión entre el maestro Twido y los esclavos de la E3T

Conexión Twido y Twido

Conexión Twido y Modicon M340

Conexión Twido y variador de velocidad

Conexión Twido y zelio

3.9.1.1 Conexión Twido y Twido: para realizar esta conexión es necesario instalar el conector mini DIN de el puerto principal (puerto 1) con el cable AWG 26 siguiendo la conexión de pines indicada en la figura 45, la cual utiliza el estándar EIA RS485 de la capa física del modelo OSI.

Figura 44: Conector mini DIN

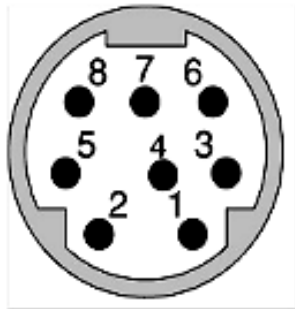


Figura 45: Diagrama de pines mini DIN

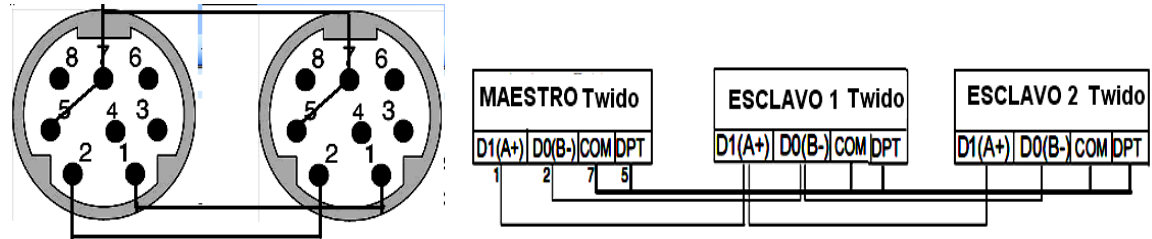
Clavijas	Base RS485	Opción RS485	RS232-C
1	D1 (A+)	D1 (A+)	RTS
2	D0 (B-)	D0 (B-)	DTR
3	NC	NC	TXD
4	/DE	NC	RXD
5	/DPT	NC	DSR
6	NC	NC	GND
7	0 V	0 V	GND
8	5 V	5 V	5 V

Nota: Consumo total máximo a 5 V (pin 8): 180 mA

Fuente TwidoSuite V2.0Guía de programación, 09/2007, pg. 79

El estándar EIA-RS485 indica que la conexión de dos hilos se realiza con las señales D0, D1 y el común ver el aparte 3.4.1 de este documento. En la tabla de pines se observa que las señales para la conexión son 1,2 y 7, pero el autómata Twido exige que la señal DPT presente en el pin 5 tiene que ir conectada al común para indicar que la comunicación se va a realizar por modbus. De tal manera la conexión entre dos autómatas Twido es la siguiente.

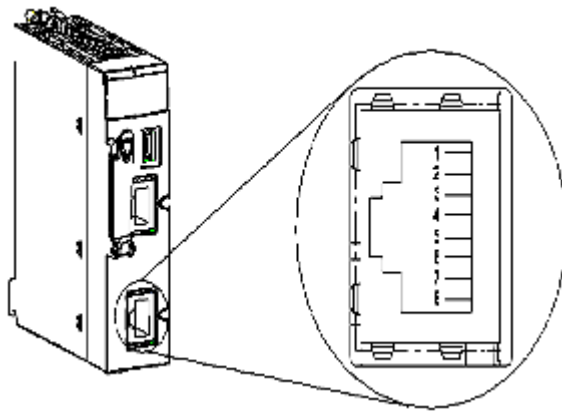
Figura 46: Conexión maestro Twido y esclavos Twido



Fuente: “TwidoSuite V2.0Guía de programación, pg. 98” Modificado por autores

3.9.1.2 Conexión TwidoTWDLCAE40DRF y Modicon M340: El PLC Modicon M340 trae un módulo procesador, el cual tiene un conector RJ 45 para configurar el protocolo de comunicación modbus serial, las características de este puerto son las siguientes:

Figura 47: Descripción puerto serie del PLC Modicon M340



Fuente: “Modicon M340 con Unity Pro, Pg.20”

Los pines utilizados por la conexión serie son los siguientes:

Pin 4: señal D1

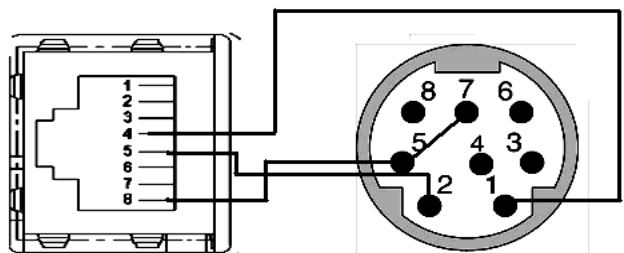
Pin 5: señal D2

Pin 8: puesta a tierra de la conexión serie(0 V)

Pin 7: se utiliza solamente cuando el M340 actúa como maestro

El puerto serial del autómata Twido TWDLCAE40DRF ya se explico. La conexión entre el PLC Twido con referencia **TWDLCAE40DRF** y el PLC M340 por el puerto serial de cada uno, como se muestra a continuación:

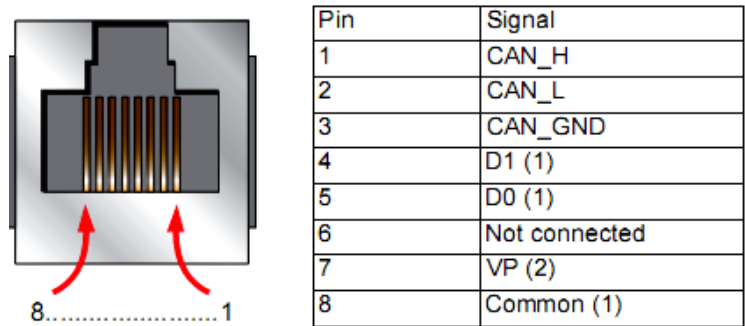
Figura 48: Conexión PLC Twido y M340



Fuente: “Modicon M340 con Unity Pro, Pg.21”

3.9.1.3 Conexión TwidoTWDLCAE40DRF y variador de velocidad Altivar31: el variador de velocidad Altivar 31 tiene un puerto para comunicarse, el cual está provisto de un conector RJ 45 hembra que tiene el siguiente diagrama de pines

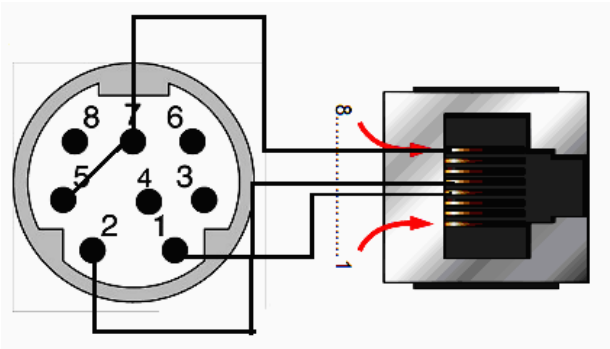
Figura 49: Pines conector RJ 45 del variador de velocidad Altivar 31



Fuente: “Altivar 31Modbus User’s manual Pg 4”

Para la conexión del PLC Twido TWDLCAE40DRF con el variador de velocidad es necesario conocer el diagrama de pines del Twido y del variador de velocidad Altivar 31 y hacer la siguiente conexión de la figura 50, en base al protocolo de la capa física EIA RS 485

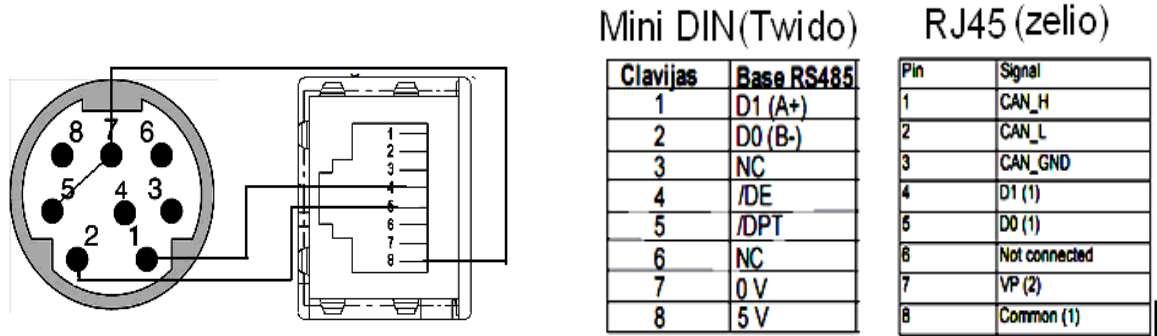
Figura 50: Conexión PLC Twido TWDLCAE40DRF y variador Altivar 31



Fuente: “Altivar 31Modbus User’s manual, Pg 4” Modificado por los autores”

3.9.1.4 Conexión Twido TWDLCAE40DRF y PLC Zelio: La conexión de sus puertos para utilizar el protocolo de comunicación modbus es la siguiente:

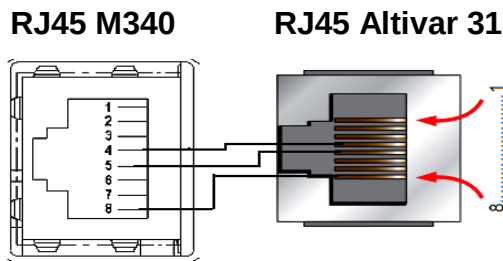
Figura 51: Conexión Twido y Zelio



Fuente: “TwidoSuite V2.0Guía de programación, pg. 79” Modificado por autores

3.9.2 Conexión Modicon M340 y variador de velocidad Altivar 31: La conexión del PLC Modicon M340 y el variador de velocidad Altivar 31 para implementar el protocolo de comunicación modbus es la mostrada a continuación:

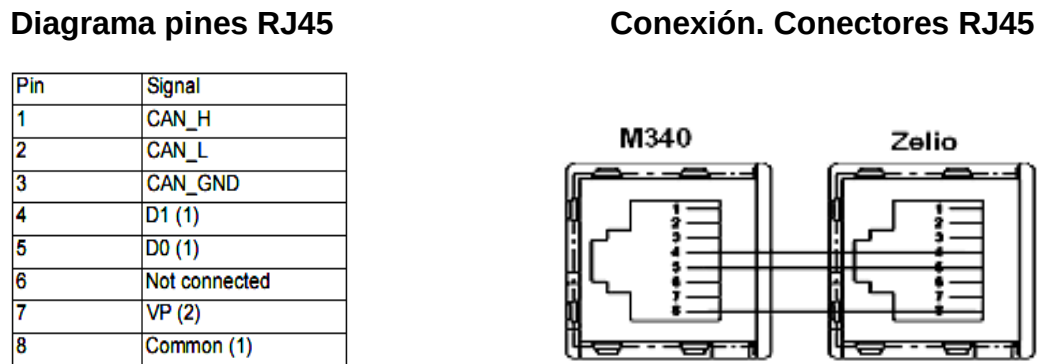
Figura 52: Conexión Modicon M340 y variador de velocidad Altivar 31



Fuente: “Altivar 31Modbus User’s manual, Pg 4” Modificado por los autores

3.9.3 Conexión Modicon M340 y zelio: La conexión del PLC modicon M340 y el PLC Zelio, con sus conectores RJ45 para configurar el protocolo de comunicación modbus utilizando el estándar EIA 485 es la indicada en la figura

Figura 53: Conexión Modicon M340 y Zelio



Fuente: “Modicon M340 con Unity Pro, Pg.20” Modificado por los autores

3.10 COMUNICACIÓN MODBUS CON EL PLC TWIDO TWDLCAE40DRF

Para configurar el autómata twido con el fin de utilizar una conexión serie para enviar y recibir caracteres de datos mediante el protocolo Modbus, se debe seguir los siguientes pasos:

Configurar el puerto serie para el protocolo Modbus mediante el software TwidoSuite.

Crear en el programa hecho en TwidoSuite una tabla de envío/recepción por cada instrucción EXCHx utilizada

3.10.1 Configuración del puerto principal con Twidosuite: Parte de este trabajo consiste en implementar el protocolo modbus en el autómata Twido con referencia TWDLCAE40DRF el cual está provisto de un puerto principal además tiene un segundo puerto que necesita el adaptador serie TWDNAC485.

El puerto 1 es el que se utilizo en los programas para el presente trabajo.

Para configurar el puerto 1 se debe seguir estos pasos:

Seleccionar Puerto 1.

configurar en la ventana del puerto en tipo de protocolo, configurar Modbus y en dirección asignar la dirección del maestro o esclavo que se quiera configurar configurar en los demás elementos de la red la dirección y el tipo de protocolo, este proceso es muy similar al paso de configuración del puerto, la diferencia es que se le da clic en el elemento que se quiere configurar ya sea maestro o esclavo.

Acceder a la ventana de configuración de red asociado a los parámetros de hardware de conexión Modbus.

Configurar la ventana de configuración de red. Esto se realiza configurando la velocidad del bus, el modo de transmisión, la paridad, tiempo de respuesta del mensaje y tiempo de espera entre tramas. cuando se configura la paridad esta puede ser par o impar o sin paridad, si elige sin paridad se deben seleccionar dos bits de parada.

3.10.2 Crear en el programa de twidosuite una tabla de envío/recepción por cada instrucción EXCHx utilizada: Para esta aplicación hay que tener en cuenta varios factores entre los cuales están:

3.10.2.1 Protocolo Modbus en el maestro: El modo de funcionamiento Modbus maestro permite al autómatas enviar una solicitud Modbus a un esclavo y esperar una respuesta, para ello se puede utilizar la instrucción EXCHx.

Este modo de funcionamiento admite Modbus ASCII y Modbus RTU, con un tamaño máximo de las tramas enviadas o recibidas de 256 bytes. Existe una tabla de palabras denominada tabla de control, envío y recepción asociada a cada instrucción EXCHx utilizada la cual está formada por tablas de control, de envío y de recepción. Ver tabla 19

Es bueno resaltar que además de las solicitudes a los esclavos individuales el autómatas maestro de Modbus puede iniciar la difusión de una solicitud a todos los esclavos. En este caso el byte de comando debe establecerse en 00, mientras que la dirección de esclavo debe establecerse en 0.

Tabla 19: Tabla de control, envío y recepción.

	Byte más significativo	Byte menos significativo
Tabla de control	Comando	Longitud (envío/recepción)
	Offset de recepción	Offset de envío
Tabla de envío	Byte 1 enviado	Byte 2 enviado
	...	...
	...	Byte n enviado
	Byte n+1 enviado	
Tabla de recepción	Byte 1 recibido	Byte 2 recibido
	...	...
	...	Byte p recibido
	Byte p+1 recibido	

Fuente: “Twidosuite V2.0 guía de programación, Pg 127”

A continuación se presenta cada tabla que componen a la tabla de envío y recepción:

Tabla de control: esta tabla contiene los siguientes parámetros.

Byte de longitud: este parámetro contiene la longitud de la tabla de envío en bytes y puede contener como máximo 250 bytes. Si el parámetro de offset de envío es igual a 0, será igual a la longitud de la tabla de envío. Si el parámetro de offset de envío no es igual a 0 no se enviará un byte de la tabla de envío (indicado por el valor de offset, ver cada uno de los casos específicos para las funciones modbus “tablas de envío/recepción”) y este parámetro será igual a la longitud de la propia trama más uno, debido al byte desplazado por el byte offset.

Byte de comando: puede tomar valores de 0 para difusión, 1 para transmisión en modo RTU y 2.

Byte de offset de emisión: Se utiliza para gestionar los envíos asociados a los valores de byte/palabra del protocolo Modbus. Por ejemplo, si este byte es igual a 2, el segundo byte se ignorará, haciendo que el tercer byte de la tabla sea el segundo en enviarse.

Byte de offset de recepción: contiene el rango de bytes (1 para el primer byte, 2 para el segundo byte, etc.) dentro de la tabla de recepción¹⁸ que se agregará cuando se envíe el paquete. Se utiliza para gestionar los envíos asociados a los valores de byte/palabra del protocolo Modbus. Por ejemplo, si este byte es igual a 4, el cuarto byte de la tabla se representará con un cero y el cuarto byte recibido se introducirá en la quinta ubicación de la tabla.

Tablas de envío y recepción

Cuando se ejecuta el programa, el autómata determina cuál es la capa de enlace de datos y realiza todas las conversiones necesarias para procesar el envío y la respuesta. Una vez enviados todos los bytes, el autómata cambia a modo de recepción y espera a recibir los bytes. La recepción finaliza de una de las formas siguientes:

Se detecta un timeout en un carácter o en una trama¹⁹

Si se recibe los caracteres que indica el fin de trama cuando se está trabajando en modo ASCII

La tabla de recepción está llena.

La capa de enlace de datos de Modbus tiene las siguientes limitaciones:

Dirección: de 1 a 247 direcciones posibles

Palabras: 125 palabras de 16 bits cada palabra

¹⁸ La tabla de recepción está formada por palabras de memoria que se dejan sin usar y que se reservan mediante la instrucción EXCHx, para ser utilizadas por el esclavo en su respuesta

¹⁹ una vez transcurrido 1.5 veces el tiempo de carácter del byte en Modbus RTU o una vez pase el timeout configurado mediante el software en Modbus ASCII. El timeout es el tiempo que se programa para que un esclavo le responda la solicitud del maestro.

3.10.2.2 Intercambio de mensajes en twidosuite: Para que el intercambio de mensaje en twidosuite tenga éxito, pueden incorporarse en su programa dos servicios de comunicación como son:

Instrucción EXCHx: para enviar/recibir mensajes.

El bit %MSGx.D: para garantizar que una instrucción EXCHx comience a ejecutarse si y solo si se ha finalizado la ejecución de otra instrucción EXCHx

Instrucción EXCHx²⁰

La instrucción EXCHx permite al autómeta Twido enviar o recibir información dirigida o procedente de los dispositivos Modbus²¹.

El usuario define una tabla de palabras (%MWi: L) que contiene información de control y los datos que se van a enviar y recibir (hasta 250 bytes en el envío o recepción).

Sintaxis: [EXCHx %MWi: L]

Donde: x = número de puerto (1 ó 2)

L = número de palabras de la tabla de envío/recepción

Figura 54: BIT %MSGx.D

Entrada/salida	Definición	Descripción
%MSGx.D	Comunicación completa	0: solicitud en curso. 1: comunicación finalizada si se produce el final del envío, se recibe el carácter final, se produce un error o se restablece el bloque.

Fuente: “TwidoSuite V2.0 Guía de programación, Pg 132” Modificada por autores

²⁰ La información de la instrucción EXCHx fue tomada tal cual de “twidosuite V2.0 guía de programación, Pg 131”

²¹ los envíos se pueden hacer por turnos o simultáneamente a todos los esclavos de la red, mientras que las recepciones solo se pueden hacer por turnos; los envíos simultáneos a todos los esclavos no tienen respuesta

3.10.2.3 Errores y condiciones de funcionamiento: Si se desea saber qué tipo de error sucedió, es necesario consultar la palabra de sistema %SW63 que contiene el código de error EXCHx del puerto 1 y la palabra de sistema %SW64 que contiene el código de error EXCHx del puerto 2. La tabla 20 presenta estas palabras de sistema

3.10.2.4 Solicitudes o funciones estándar Modbus: estas solicitudes se utilizan para intercambiar palabras de memoria (%MWi) o bits (%Mi) entre el maestro y los esclavos. Cada solicitud se puede representar con una tabla de envío/recepción escrita o definida en el programa hecho con el software TwidoSuite, en la cual se describe el funcionamiento de esta solicitud.

Tabla 20: Descripción palabras de sistema y código de error EXCHx

<i>Palabras de sistema</i>	<i>Uso</i>
%SW63	Código de error EXCH1: 0: operación correcta 1: número excesivo de bytes para enviar (> 250) 2: tabla de envío demasiado pequeña 3: tabla de palabras demasiado pequeña 4: tabla de recepción desbordada 5: temporización transcurrida 6: envío 7: comando incorrecto en la tabla 8: puerto seleccionado no configurado/disponible 9: error de recepción 10: no se puede utilizar %KW si se está recibiendo 11: offset de envío mayor que la tabla de envío 12: offset de recepción mayor que la tabla de recepción 13: procesamiento EXCH detenido por el autómata
%SW64	Código de error EXCH2: consulte %SW63.

Fuente: “TwidoSuite V2.0 Guía de programación, Pg 133”

A continuación se presentan las solicitudes modbus representadas en las tablas de envío y recepción.

Tabla 21: Lectura de N bits, funciones 01 y 02 de modbus

	Tabla Índice	Byte más significativo	Byte menos significativo
Tabla de control	0	01 (emisión/recepción)	06 (longitud de emisión) (*)
	1	03 (offset de recepción)	00 (offset de emisión)
Tabla de envío	2	Slave a (de 1 a 247)	01 ó 02 (código de solicitud)
	3	Dirección del primer bit que se va a leer	
	4	N_1 = Número de bits que se van a leer	
Tabla de recepción (después de la respuesta)	5	Slave a (de 1 a 247)	01 ó 02 (código de respuesta)
	6	00 (byte añadido por la acción offset Rx)	N_2 = Número de bytes de datos que se van a leer = $[1+(N_1-1)/8]$, donde $[]$ significa parte integral
	7	Valor del 1.º byte (valor = 00 ó 01)	Valor del 2.º byte (si $N_1 > 1$)
	8	Valor del 3.º byte (si $N_1 > 1$)	
	...		
	$(N_2/2)+6$ (si N_2 es par) $(N_2/2+1)+6$ (si N_2 es impar)	Valor del byte N_2^o (si $N_1 > 1$)	

(*) Este byte recibe también la longitud de la cadena emitida después de la respuesta

Fuente: “TwidoSuite V2.0 Guía de programación, PG 141”.

Lectura de N palabras, funciones 03 y 04 de modbus

Tabla 22: Lectura de N palabras

	Tabla Índice	Byte más significativo	Byte menos significativo
Tabla de control	0	01 (emisión/recepción)	06 (longitud de emisión) (*)
	1	03 (offset de recepción)	00 (offset de emisión)
Tabla de envío	2	Slave a (de 1 a 247)	03 ó 04 (código de solicitud)
	3	Dirección de la primera palabra que se va a leer	
	4	N = Número de palabras de lectura (1)	
Tabla de recepción (después de la respuesta)	5	Slave a (de 1 a 247)	03 ó 04 (código de respuesta)
	6	00 (byte añadido por la acción offset Rx)	2*N (número de bytes leídos)
	7	Primera palabra leída	
	8	Segunda palabra leída (si N>1)	
	...		
	N+6	Palabra N leída (si N>2)	

(*) Este byte recibe también la longitud de la cadena emitida después de la respuesta

Fuente: “TwidoSuite V2.0 Guía de programación, Pg 142”

El offset de recepción (Rx) en la tabla 22 cuando es igual a = 3 agregará un byte de valor cero en la tercera posición de la tabla de recepción. Esto permite un buen posicionamiento en esta tabla del número de bytes leídos y de los valores de las palabras leídas. Las tablas de envío/recepción trabajan con palabras de 2 bytes, cada byte con dos caracteres. No es lo mismo ubicar un par de caracteres en el byte menos significativo, que en el byte más significativo. Ejemplo suponiendo que 00 es el par de caracteres en hexadecimales del byte no utilizado en la palabra; los caracteres hexadecimales 02 serán equivalentes a $0 \cdot 16^4 + 0 \cdot 16^3 + 0 \cdot 16^1 + 2 \cdot 16^0 = 2$ (en decimal) si dicho par de caracteres se ubican en el byte menos significativo de una palabra; por el contrario si el mismo par de caracteres se ubica en el byte más significativo de la palabra equivaldría a $0 \cdot 16^4 + 2 \cdot 16^3 + 0 \cdot 16^1 + 0 \cdot 16^0 = 512$ (en decimal).

Tabla 23: Escritura de un bit, función 05 de modbus

	Tabla Índice	Byte más significativo	Byte menos significativo
Tabla de control	0	01 (emisión/recepción)	06 (longitud de emisión) (*)
	1	00 (offset de recepción)	00 (offset de emisión)
Tabla de envío	2	Slave a (de 1 a 247)	05 (código de solicitud)
	3	Dirección del bit que se va a escribir	
	4	Valor del bit que se va a escribir	
Tabla de recepción (después de la respuesta)	5	Slave a (de 1 a 247)	05 (código de respuesta)
	6	Dirección del bit escrito	
	7	Valor escrito	

(*) Este byte recibe también la longitud de la cadena emitida después de la respuesta.

Fuente: “TwidoSuite V2.0 Guía de programación, Pg 143”

Esta solicitud no necesita utilizar un offset.

Para asignar el valor 1 a un bit, la palabra que contiene el valor del bit que se va a escribir debe contener el valor FF00h. Si se desea asignar el valor de 0 a un bit, la palabra que contiene el valor del bit que se va a escribir debe contener el valor de 0.

Tabla 24: Escritura de una palabra

	Tabla Índice	Byte más significativo	Byte menos significativo
Tabla de control	0	01 (emisión/recepción)	06 (longitud de emisión) (*)
	1	00 (offset de recepción)	00 (offset de emisión)
Tabla de envío	2	Slave a (de 1 a 247)	06 (código de solicitud)
	3	Dirección de la palabra que se va a escribir	
	4	Valor de la palabra que se va a escribir	
Tabla de recepción (después de la respuesta)	5	Slave a (de 1 a 247)	06 (código de respuesta)
	6	Dirección de la palabra escrita	
	7	Valor escrito	

Fuente: “TwidoSuite V2.0 Guía de programación, Pg 144”

EL byte de emisión (*) recibe también el número de la longitud de la cadena de bytes emitida después de la respuesta.

Esta solicitud no necesita utilizar un offset.

La trama de respuesta es la misma que la de esta solicitud, en caso contrario enviará un código de error

Tabla 25: Escritura de N bits, Función 15 de modbus

	Tabla Índice	Byte más significativo	Byte menos significativo
Tabla de control	0	01 (emisión/recepción)	8 + número de bytes (emisión)
	1	00 (offset de recepción)	07 (offset de emisión)
Tabla de envío	2	Slave a (de 1 a 247)	15 (código de solicitud)
	3	Número del primer bit que se va a escribir	
	4	N_1 = Número de bits que se van a escribir	
	5	00 (byte no enviado, efecto de offset)	N_2 = Número de bytes de los datos que se van a escribir = $[1+(N_1-1)/8]$, donde $[]$ significa parte integral
	6	Valor del 1.º byte	Valor del 2.º byte
	7	Valor del 3.º byte	Valor del 4.º byte
	...		
	$(N_2/2)+5$ (si N_2 es par) $(N_2/2+1)+5$ (si N_2 es impar)	Valor del byte N_2	
Tabla de recepción (después de la respuesta)		Slave a (de 1 a 247)	15 (código de respuesta)
		Dirección del 1.º bit escrito	
		Dirección de los bits escritos (= N_1)	

Fuente: “TwidoSuite V2.0 Guía de programación, Pg 145”

El offset de emisión (07); asigna 00.al séptimo byte de la trama enviada

Tabla 26: Escritura de N palabras, función 16 modbus

	Tabla Índice	Byte más significativo	Byte menos significativo
Tabla de control	0	01 (emisión/recepción)	8 + (2*N) (longitud de emisión)
	1	00 (offset de recepción)	07 (offset de emisión)
Tabla de envío	2	Slave a (de 1 a 247)	16 (código de solicitud)
	3	Dirección de la primera palabra que se va a escribir	
	4	N = Número de palabras que se van a escribir	
	5	00 (byte no enviado, efecto de offset)	2*N = N.º de bytes que se van a escribir
	6	Primer valor de la palabra que se va a escribir	
	7	Segundo valor que se va a escribir	
	...		
	N+5	Valores N que se van a escribir	
Tabla de recepción (después de la respuesta)	N+6	Slave a (de 1 a 247)	16 (código de respuesta)
	N+7	Dirección de la primera palabra escrita	
	N+8	Dirección de las palabras escritas (= N)	

Fuente: “TwidoSuite V2.0 Guía de programación, PG 146”

Las tablas de envío y recepción, fueron tomadas de TwidoSuite V2.0 Guía de programación. 35013228.02, 09/2007. Pg 142 a 146²².

Explicación: tabla escritura de N palabras

Esta tabla se utiliza para gestionar la comunicación cuando el maestro quiere escribir en algún esclavo. Cuando se escribe en un esclavo se puede hacer con palabras de memoria %Mwx²³. Para la comunicación modbus usando esta tabla el programa debe incluir la instrucción EXCHx, que se encarga de ejecutar la función

²² Este trabajo tiene algunas modificaciones comparado con la Fuente.

²³ En él % Mwx, x representa un numero entero que puede ser mayor o igual a 0 y menor o igual que (y-1) donde (y) es el número máximo de palabras de memoria del dispositivo, ejemplo: (y) es 3000 para el PLC Twido

modbus contenida en esta tabla (envió de solicitud y su respectiva respuesta) con los parámetros e información contenida en la misma.

A continuación se describe paso a paso la tabla de escritura de N palabras

La tabla de escritura de N palabras es una tabla de envío/recepción y está conformada por tres tablas, la tabla de control, la tabla de envío y la tabla de recepción. La tabla de control es la encargada de dimensionar la tabla de envío, la tabla de recepción está formada por unas palabras de memoria reservadas por la orden EXCHx para que el esclavo responda en ellas la petición del maestro, es decir, en la tabla de recepción el maestro guarda lo que el esclavo le responde, además esta tabla muestra la dirección del esclavo con el que se está comunicando, o sea el esclavo que responde a la solicitud del maestro. Cada palabra está formada por dos bytes, byte más significativo y byte menos significativo, de esta manera cada fila de la tabla escritura de N palabras es una palabra²⁴.

Tabla de control: La primera palabra (Tabla Índice 0) de esta tabla está formada por el byte más significativo que toma valor de 01 cuando se produce una solicitud RTU y 0 cuando es una difusión (00), y por el byte menos significativo que indica cual va a ser el numero de palabras de la tabla de envío, esta longitud de palabras está compuesta por 8 bytes fijos (4 palabras) más (2*N); donde (2*N) indica el número de bytes que se quieren escribir en el esclavo o en los esclavos para el caso de una difusión.

La segunda palabra (Tabla Índice 1), está compuesta por offset de envío (00) y el offset de emisión (07), este par de parámetros (00 y 07) indican el número del byte donde el offset va a tener efecto tanto en la tabla de recepción como en la envío

²⁴ El parámetro 'Tabla Índice' de la tabla de envío/recepción informa el número de palabras que conforman la dicha tabla

respectivamente. Un ejemplo del efecto del byte de emisión se observa en el byte 07 de la tabla de envío el cual no se transmite.

Tabla de envío: Su primera palabra (Tabla Índice 2) está compuesta por el byte más significativo que indica la dirección de/los esclavo(s) con quien(es) se va a establecer comunicación más el byte menos significativo que es el que indica el código de la solicitud modbus que se va a realizar, en este caso escritura de N palabras (16 decimal o 10 hexadecimal).

Tabla índice (3), es una palabra que indica la dirección de la primera palabra que se va a escribir en el esclavo o los esclavos.

Tabla índice (4), es la palabra que indica el número de palabras que se van a escribir en el programa teniendo como punto de partida la dirección de la primera palabra "(Tabla Índice 3)". Este grupo de palabras tendrá direcciones consecutivas, es decir, si la primera palabra del maestro se va a escribir en la dirección 3 (%MW3) del esclavo y se quiere escribir 5 palabras en el/los esclavo(s), estas tendrán las direcciones 3 a 7 (%MW3, %MW4, %MW5, %MW6 y %MW7) "palabras del esclavo"

Esta palabra (Tabla Índice 5), está conformada por el byte más significativo el cual no se envía por efecto del offset y el byte menos significativo que es el número de bytes que se van a escribir en el esclavo. Por ejemplo si se van a escribir 5 palabras el número de bytes tiene que ser 10.

Tabla índice (6). Será el valor de la primera palabra que se va a escribir

Tabla índice (7). El segundo valor de la segunda palabra que se va a escribir.

Tabla índice (N+5). Contiene el valor de la palabra N que se va a escribir.

Tabla de recepción: el byte más significativo de Tabla Índice (N+6) contendrá la dirección del esclavo que responderá, mientras su byte menos significativo indicará el código de la respuesta modbus (escritura de N palabras)

Tabla índice (N+7). Mostrará la dirección de la primera palabra escrita.

Tabla índice (N+8). Contiene la dirección de las palabras escritas (=N)

3.11 COMUNICACIÓN CON EL PLC M340

3.11.1 Descripción modulo maestro M340: La comunicación por modbus con el PLC Modicon M340 es posible gracias a la CPU MODICON M340 BMX P34 2020²⁵ la cual tiene incorporado el puerto serial para transmitir los datos del protocolo de comunicación modbus a través de un conector RJ 45 hembra que incorpora el estándar de la capa física EIA RS 485. La CPU Modicon M340 BMX P34 2020 se presenta en la figura 55 y su descripción en la tabla 27.

Tabla 27: Descripción de la CPU Modicon M340 BMX P340 2020

Dirección	Descripción
1	LED de estado de los procesadores de la parte frontal
2	Canal integrado (canal 0) dedicado a la conexión serie.
3	Anillo de identificación del puerto serie (negro)

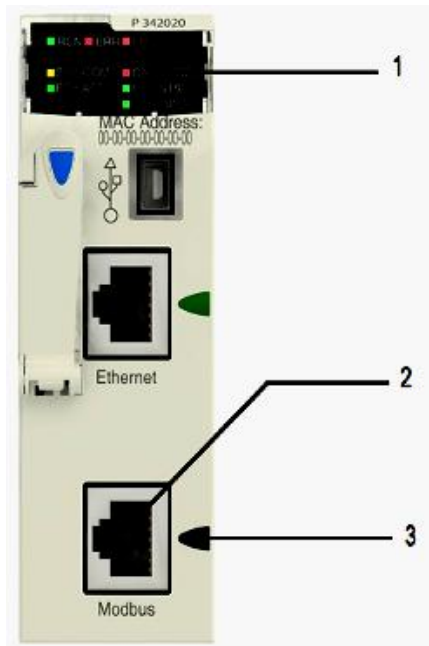
Fuente: “Modicon M340 con Unity Pro, Conexión serie, pg. 21”

El estado de la comunicación serie se indica mediante un indicador led “de color Amarillo”

Si el led es intermitente la comunicación está en curso, si el led se apaga la comunicación no está en curso:

²⁵ Modulo que usualmente se conecta en el rack inmediatamente después de la fuente de alimentación y que tiene dos puertos provistos de conectores RJ45 hembra (el superior para Modbus TCP/IP y el inferior para Modbus RTU y Modbus ASCII)”

Figura 55: CPU Modicon M340 BMX P340 2020



Fuente: “Modicon M340 con Unity Pro, pg. 20”

3.11.2 Programación del PLC M340 mediante unity pro, con modbus: para establecer una comunicación correcta entre el PLC M340 y el software unity es necesario configurar el PLC mediante el software unitypro de la siguiente forma:

3.11.2.1 Definir el hardware en el software, para modbus: consiste en describir el PLC Modicon M340 en el software unity pro tal como está conformado físicamente, es decir se tiene que definir el tipo de rack, “entendiéndose por tipo de rack cuantos módulos pueden conectarse a dicho rack”, que clase de módulos y en qué orden están ubicados en el rack; estos módulos pueden ser módulos de comunicaciones, módulos entradas digitales, módulos entradas y salidas analógicas, la fuente etc.

3.11.2.2 Configurar el puerto modbus en el software unity pro: el puerto se configura en una ventana que tiene dos opciones, la primera si el PLC va a ser maestro y la segunda si el PLC va a ser esclavo “similar al software twidosuite”, además en esta misma ventana se tiene que configurar los parámetros típicos de la red como: la velocidad de transmisión de datos, el modo de transmisión de datos (RTU o ASCII), bit de paridad o no paridad, tiempo entre tramas, tiempo de respuesta del mensaje y bits de parada.

3.11.2.3 programar los bloques de comunicación : existen tres bloques “ADDM, READ_VAR AND WRITE_VAR” para la comunicación modbus empleando el software Unity Pro, estos bloques conforman una parte del programa que gestiona la comunicación y sirven para intercambiar palabras de memoria (%MWi) o bits (%Mi) entre el maestro y los esclavos.

3.11.2.4 elaborar un programa para el control de la red: este programa controla todas las comunicaciones entre el maestro y los esclavos, además determina lo que se va a realizar con los datos recibidos.

El software unity pro tiene 5 clases de lenguajes para programar, en este documento se empleo el lenguaje ladder.

3.12. COMUNICACIÓN CON EL PLC ESCLAVO ZELIO “REF SR3 B101BD”:

3.12.1 Descripción del relé programable zelio: el autómata esclavo zelio tiene un modulo de extensión con referencia SR3 MBU01BD para la comunicación usando modbus, esta comunicación se realiza a través de un puerto serie

mediante un conector RJ 45 hembra que incorpora el estándar de la capa física EIA RS 485.

Figura 56: Descripción PLC zelio:

PLC Zelio



Modulo de extensión



Fuente: “Relés programables Zelio, catálogo- 2004. Pg 32”

3.12.2 Configuraciones del PLC zelio: para configurar el zelio de modo tal que exista comunicación por modbus se sugiere el siguiente procedimiento:

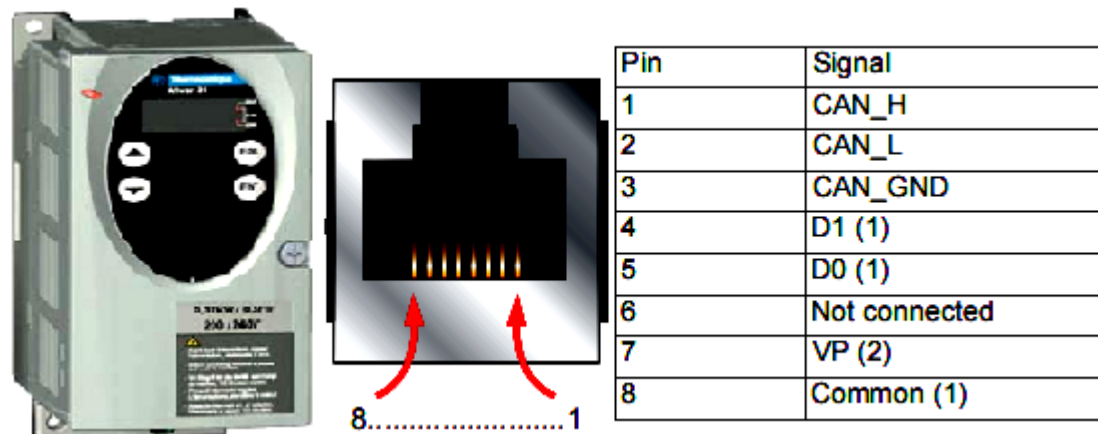
Primero: definir el hardware en el software, esto se realiza cuando se crea un nuevo programa en el software; el software le preguntará la referencia del autómatas zelio la del módulo expansión de comunicación por modbus y la del módulo de expansión entradas salidas, si lo requiere.

El zelio no necesita configurarse como maestro o como esclavo ni hay que elegir el protocolo de comunicación debido a que él únicamente puede funcionar como esclavo y solo tiene comunicación por modbus

Segundo: configurar los parámetros de la red modbus, esto se hace empleando él en software Zelio soft, o se puede configurar con el panel de control²⁶. Para La comunicación con el zelio existe el lenguaje ladder y lenguaje fbd (diagrama de bloques funcionales), dependiendo de lo que se quiere hacer con el zelio se deberá escoger si se programa en ladder o en fbd. Si solo se quiere leer las entradas y salidas del zelio es decir si se pretende activar variables en otros PLCs o dispositivos de automatización como por ejemplo variadores de velocidad o Interfaces hombre máquina se podrá usar ladder o fbd. Si se está interesado en escribir palabras en el zelio y activar con esas palabras algunas de las variables de Zelio por ejemplo sus salidas entonces se necesitará usar el lenguaje fbd. Los dos lenguajes no se pueden usar al mismo tiempo.

3.13. COMUNICACIÓN CON EL VARIADOR DE VELOCIDAD ALTIVAR 31

Figura 57: Descripción del Altivar 31



Fuente: “Altivar 31Modbus User’s manual, Pg 4” Modificado por los autores”

²⁶ hace referencia a una interfaz hombre-máquina que viene de fabrica integrada a la parte delantera del PLC Zelio

Para realizar la comunicación con el variador de velocidad altivar 31 se sugiere seguir estos pasos:

3.13.1 descripción del variador de velocidad altivar 31: La comunicación por modbus con el altivar 31 es posible gracias a que este tiene incorporado un puerto serial para transmitir los datos del protocolo de comunicación a través de un conector RJ 45 hembra que incorpora el estándar de la capa física EIA RS 485. En la figura 57 se presenta el variador de velocidad altivar 31

3.13.2 configuración del variador de velocidad altivar 31.El variador de velocidad es un esclavo que sirve para comunicarse usando los protocolos de comunicación industrial modbus y CANopen; dependiendo con que protocolo se comunique su configuración se modifica.

Para comunicarse por modbus es necesario ir al menú de comunicaciones y configurar la dirección de red y la velocidad de transmisión.

En el menú de control (Ctl) se debe configurar el medio por el que van a recibir las consignas de control y de velocidad, que puede ser el terminal, el bornero o el bus de comunicaciones.

En el menú de control del motor (drC) se introducen los parámetros de la placa del motor como es la tensión nominal, la corriente nominal, la velocidad nominal y el factor de potencia.

Por último en el menú entrada/salidas (I-O), hay un parámetro que permite escoger entre control de dos o tres hilos. El control vía bus solo es posible si el control de dos hilos esta seleccionado.

4. PROTOCOLO DE COMUNICACIÓN MODBUS TCP/IP

Modbus TCP/IP también conocido como modbus/TCP es simplemente el protocolo modbus RTU con una interfaz TCP que se ejecuta en la capa física Ethernet.

El modelo de los mensajes en la capa de aplicación (nivel 7 del modelo OSI) del protocolo de comunicación modbus define las normas para la organización e interpretación de los datos independientemente del medio de transmisión de estos. TCP/IP se adapta al modelo OSI y se refiere al Protocolo de la capa de transporte TP y el Protocolo de Internet IP, el cual proporciona el modo de transmisión de los datos para modbus TCP/IP.

En resumen Modbus TCP/IP, utiliza la capa TCP/IP con la capa física Ethernet (IEEE 802.3) y un método estándar de representación de datos, el mismo utilizado por Modbus serial para transportar los datos que constituye el mensaje entre dispositivos compatibles. En esencia, el mensaje Modbus TCP/IP es simplemente una comunicación modbus encapsulada en una trama Ethernet TCP/IP

4.1 PROTOCOLO TCP/IP

TCP/IP es un protocolo compatible con cualquier sistema operativo y con cualquier tipo de hardware, este protocolo es utilizado por todos los ordenadores conectados a Internet para comunicarse entre sí. TCP/IP está conformado a su vez por un conjunto de protocolos, pero los más importantes son el TCP y el IP.

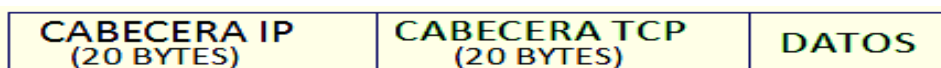
4.1.1 TCP (Transmission Control Protocol): El protocolo de control de transmisión (TCP) pertenece a la capa de transporte y es encargado de dividir el

mensaje original en datagramas de menor tamaño; los datagramas serán dirigidos a través del protocolo IP de forma individual.

El protocolo TCP se encarga además de añadir la información necesaria a cada uno de los datagramas para su correcta transmisión, la cual se inserta al inicio de los datos que componen el datagrama en forma de cabecera, esta cabecera tiene al menos 160 bits que se encuentran repartidos en varios campos con diferente significado. Cuando la información se divide en datagramas para ser enviada el TCP enumera los datagramas, de manera que sea posible volver a unirlos en el orden adecuado; a demás solicita el envío de los datagramas individuales que no hayan llegado o que contengan errores, sin que sea necesario volver a enviar el mensaje completo.

4.1.2 IP (Internet Protocol): El IP es un protocolo que pertenece al nivel de red, por lo tanto, es utilizado por los protocolos del nivel de transporte como TCP para enrutar los datos hacia su destino, para ello se utiliza una nueva cabecera que se antepone al datagrama que se está tratando. Si es el protocolo TCP el encargado de manejar el datagrama antes de pasarlo al IP, la estructura del mensaje una vez tratado la presenta la figura 58.

Figura 58: Estructura del mensaje IP



Fuente: http://elsitiodeltelecomunicaciones.iespana.es/protocolo_TCP_IP.htm

La cabecera IP “versión 4” tiene un tamaño de 160 bit y está formada por varios campos de diferente significado. En seguida se presentan estos campos:

Versión. Tamaño 4 bits: Siempre vale (0100).

Tamaño Cabecera (IHL). Tamaño 4 bits: la longitud de la cabecera de esta versión, esta conforma por 5 grupos de 32 bits. Su valor correcto es de 5, pero puede tener valores hasta de 15, para el caso de utilizar tamaño de cabeceras ampliadas.

Tipo de servicio. Tamaño 8 bits: indica la prioridad de paquetes durante el tránsito de la información en momentos de sobrecarga de la red. Estos 8 bits se agrupan de la siguiente manera; los 5 bits de menos peso son independientes e indican las características del servicio.

Bit 0: sin uso, su valor es 0.

Bit 1: 1 costo mínimo, 0 costo normal.

Bit 2: 1 máxima fiabilidad, 0 fiabilidad normal.

Bit 3: 1 máximo rendimiento, 0 rendimiento normal.

Bit 4: 1 mínima demora, 0 demora normal.

Los 3 bits restantes están relacionados con la precedencia de los mensajes, un indicador señala el nivel de urgencia basado en un sistema de precedencia. La urgencia que estos estados representan aumenta a medida que el número formado por estos 3 bits lo hace, y se representan con los siguientes nombres:

000: De rutina.

001: Inmediato.

010: Inmediato.

011: Relámpago.

100: Invalidación relámpago.

101: Procesando llamada crítica y de emergencia.

110: Control de trabajo de Internet.

111: Control de red.

Longitud total. Tamaño 16 bits: especifica la longitud total del datagrama IP expresado en bytes “cabecera + datos”.

Identificación. Tamaño 16 bits: identifica con un valor único cada paquete del emisor, para que de esta manera se pueden reconstruir los paquetes grandes fragmentados en algún punto de la red.

Indicadores (flags). Tamaño 3 bits: Se utiliza para especificar valores relativos a la fragmentación de paquetes e indican lo siguiente:

bit 0: Reservado; es igual a 0

bit 1: 0 = divisible, 1 = No divisible

bit 2: 0 = último fragmento, 1 = fragmento intermedio

Posición de Fragmento (fragmento offset) Tamaño 13 bits: indica la posición en unidades de 64 bits, que ocupa el paquete actual dentro del datagrama original

Tiempo de Vida (TTL) Tamaño 8 bits. indica el máximo número de direccionadores que un paquete puede atravesar. Cada vez que algún nodo procesa este paquete disminuye su valor como mínimo en 1, cuando llega a ser 0 el paquete no será reenviado

Protocolo. Tamaño 8 bits: indica el protocolo del siguiente nivel utilizado en la parte de datos del datagrama.

Suma de Control de Cabecera (checksum). tamaño 16 bits. Se recalcula cada vez que algún nodo cambia alguno de sus campos por ejemplo, el Tiempo de Vida. El método de cálculo consiste en sumar el complemento a 1 de cada palabra de 16 bits de la cabecera y hacer el complemento a 1 del valor resultante

Dirección de origen. Tamaño 32bits. Contiene la dirección del equipo (host) que envía el paquete.

Dirección de destino. Tamaño 32bits. Esta dirección es la del equipo que recibirá la información. Los routers o gateways intermedios deben conocerla para dirigir correctamente el paquete²⁷.

²⁷ El texto formato de cabecera IP se tomo de. < <http://f34k.files.wordpress.com/2008/01/microsoft-word-cabecera-ip.pdf>> [citado el 28 de octubre del 2010].

Figura 59: Formato cabecera IP (versión 4)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Versión				IHL				Tipo de servicio								Tamaño total															
Identificación																Flags		Fragmento Offset													
Tiempo de vida								Protocolo								checksum															
Dirección IP del emisor																															
Dirección IP del receptor																															

Fuente: <http://inquisicion.iespana.es/ip.htm>

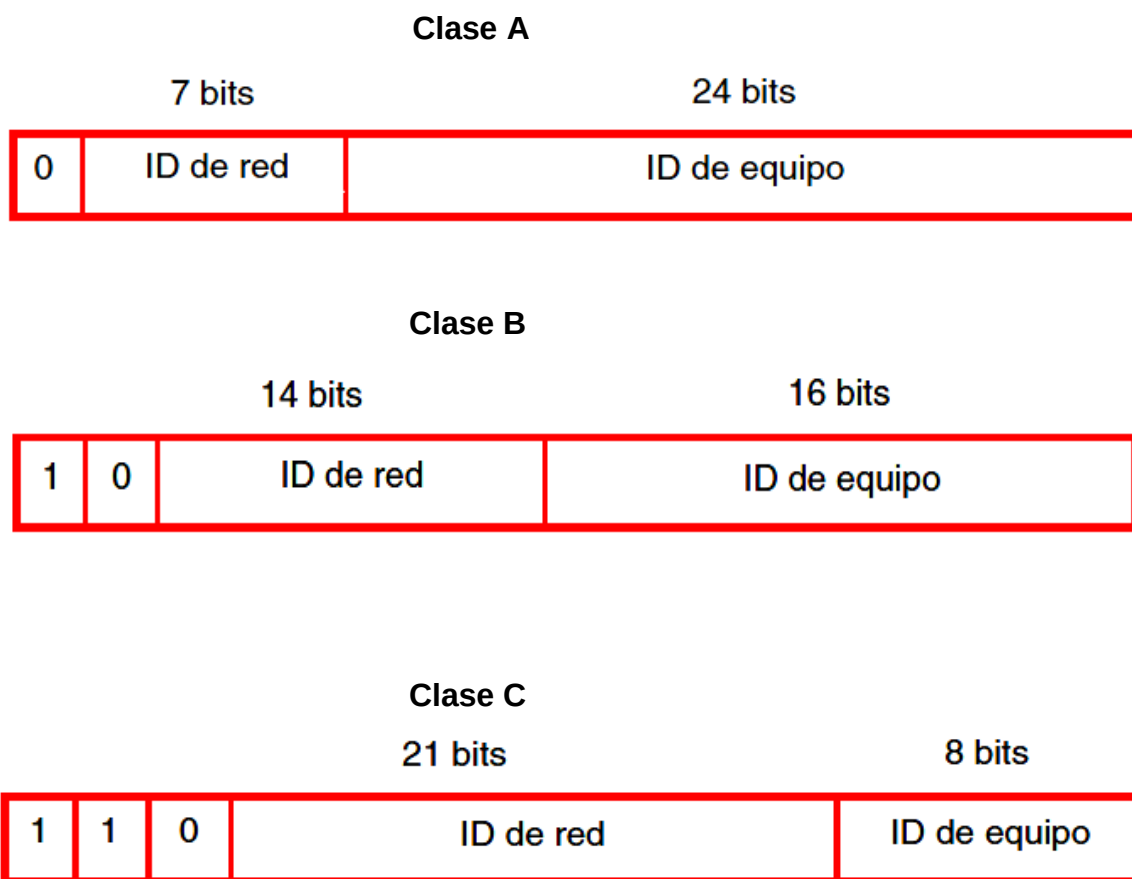
4.1.2.1. Dirección de Internet (IP Address): esta dirección utiliza 4 byte y sirve para identificar tanto al dispositivo que se encuentran en una misma red como al que se encuentra en otra red. En internet se encuentran conectadas redes de tamaños muy diversos por esta razón se establecen tres clases de direcciones, las cuales se representan mediante tres rangos de valores

Clase A: Son las que en su primer byte tienen un valor comprendido entre 1 y 126, utilizando únicamente este primer byte para identificar la red, quedando los otros tres bytes disponibles de la dirección para el identificador del equipo, permitiendo un número máximo de 16387054 de dispositivos en cada una de las redes. Estas clases de dirección se utiliza para redes a gran escala con un gran número de sitios conectados.

Clase B: las direcciones de clase B utiliza los dos primeros bytes para identificar la red, utilizando un valor comprendido entre 128 a 191 para el primer byte y entre 1 a 254 para el segundo byte. Los dos últimos bytes constituyen el identificador del equipo, permitiendo un número máximo de 64516 equipos en la misma red. Esta clase de direcciones se utiliza para redes de escala media.

Clase C: este tercer tipo de direcciones utiliza los tres primeros bytes para identificar la red, con un rango desde, 192 a 223 para el primer byte, 1 a 254 para el segundo byte y 1 a 254 para el tercer byte. El último byte es el encargado de identificar los equipos, permitiendo un máximo de 254 equipos en cada red. Estas direcciones se utiliza para redes de pequeña escala con pocos equipos conectados.

Figura 60: Clases de direcciones IP



Fuente: “iPC/XBT G/XBT GC/XBT GK/XBT GT/XBT GTW: Magelis Controlador Modbus TCP/IP, 11/2008, pg. 42”

Tabla 28: Direcciones IP

Clase	Primer byte	Identificación de red	Identificación de equipo	Número de redes	Número de redes
A	1.....126	1 Byte	3 Byte	126	16387064
B	127.....191	2 Byte	2 Byte	16256	64516
C	192.....223	3 Byte	1 Byte	2064512	254

Fuente: http://elsitiodeltelecomunicaciones.iespana.es/protocolo_TCP_IP.htm

El número 0 está reservado para los equipos que no conocen su dirección.

El número 255 se reserva para el broadcast (mensaje de difusión). El broadcast es necesario cuando se pretende hacer que un mensaje sea visible para todos los sistemas conectados a la misma red

4.1.2.2 Máscaras de subred y subdireccionamiento: La dirección IP está compuesta por dos identificadores, uno para la red y el otro para el equipo, el cual puede incluir un identificador de subred. El identificador de subred sirve para controlar redes locales sin hacer ningún cambio en la red designada por el identificador de la red.

La máscara de subred muestra el número de bits asignado respectivamente al identificador de red y al indicador de subred “los bits para estos dos identificadores toman el valor de 1” la máscara subred también asigna bits al identificador de equipo pero en esta ocasión los bits toman el valor 0

A continuación se explica una red clase B que está determinada por cuatro bytes que toman valores de 140, 186, 90 y 3 respectivamente.

Ejemplo: Red 140.186.90.3

Red clase B

16 bits	8 bits	8 bits
Identificador de red = 140.186	Identificador de subred = 90	Identificador de equipo = 3

Mascara subred

De 24 bits a 1	De 8 bits a 0

Fuente: “iPC/XBT G/XBT GC/XBT GK/XBT GT/XBT GTW, Magelis Controlador Modbus TCP/IP, pg. 43”

Esta división del identificador de equipo permite crear 254 subredes con 254 equipos por cada subred para un direccionamiento de clase B.

El valor de la máscara de la subred elegida debe coincidir con el tipo de dirección IP. El valor de la máscara de subred será el siguiente:

Para una dirección de clase A: 255.xxx.xxx.xxx,

Para una dirección de clase B: 255.255.xxx.xxx,

Para una dirección de clase C: 255.255.255.xxx,

El valor de xxx lo elige el usuario, para el caso de este proyecto “(250 o 0)”²⁸.

²⁸ Modelo cliente-servidor tomado de: IPC/XBT G/XBT GC/XBT GK/XBT GT/XBT GTW, Magelis Controlador Modbus TCP/IP, 11/2008, pg. 42

4.2 MODELO CLIENTE-SERVIDOR DE MODBUS TCP/IP

El modelo de mensajería (MSM) de la capa de aplicación de Modbus proporciona una comunicación cliente / servidor entre dispositivos conectados a una red Ethernet TCP / IP.

Este modelo cliente/servidor está basado en cuatro tipos de mensajes.

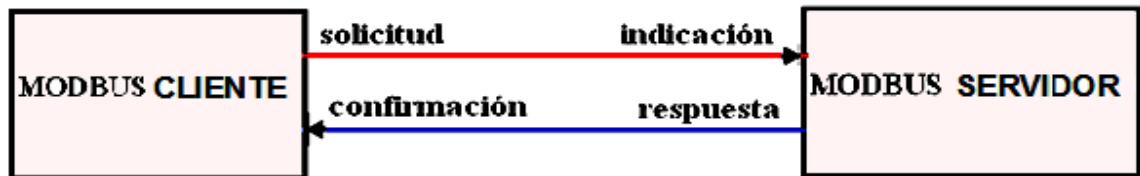
Solicitud modbus

Confirmación modbus

Indicación modbus

Respuesta modbus

Figura 61: Modelo cliente/servidor:



Fuente: “.modbus messaging on TCP/IP implementation guide v1.0b, pg 2”

4.2.1 Solicitud modbus: Es el mensaje enviado por la red, a el cliente para iniciar una transacción,

4.2.2 Indicación modbus: Es el mensaje de la petición recibida en el servidor.

4.2.3 Respuesta modbus: Es el mensaje de respuesta enviado por el servidor

4.2.4 Confirmación modbus: Es el mensaje de respuesta recibido por el cliente

El modelo cliente/servidor se utilizan en tiempo real para el intercambio de información:

entre la aplicación del dispositivo y otro dispositivo

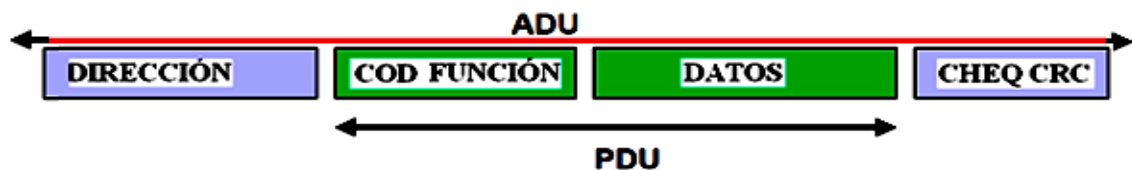
entre aplicaciones HMI / SCADA y dispositivos,

entre un PC y un programa que proporciona servicios en línea

4.3. ESTRUCTURA MENSAJE MODBUS:

La capa de aplicación del protocolo Modbus estándar define una trama para transmitir el mensaje modbus. Está trama tiene como base la unidad de datos de protocolo (PDU) independiente de las capas del modelo OSI que se encuentren por debajo del séptimo nivel. A la PDU se le adiciona una dirección y el chequeo de errores CRC para forma la tramaunidad de datos de aplicaciones (ADU) “trama de modbus serial”.

Figura 62: Trama modbus



Fuente: “Modbus messaging on TCP/IP implementation guide v1.0b, Pg 2”

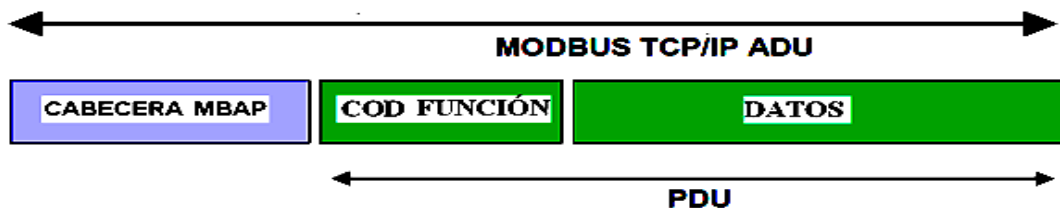
El maestro inicia una transacción modbus construyendo la trama unidad de datos de aplicación (ADU).

El código de función indica al esclavo que tipo de acción se va a realizar.

4.4 TRAMA ADU EN MODBUS TCP/IP

Modbus TCP/IP encapsula la unidad de datos de protocolo (PDU) modbus junto con la cabecera de protocolo modbus de aplicación (MBAP) para formar la trama unidad de datos de aplicación de modbus TCP/IP (ADU TCP/IP) Ver figura 63

Figura 63: Trama ADU para modbus TCP/IP

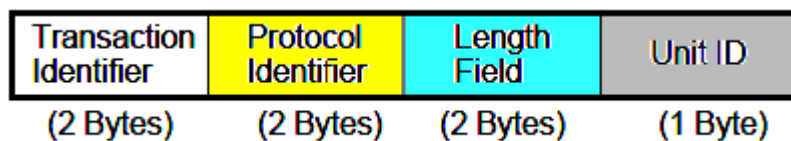


Fuente: modbus messaging on TCP/IP implementation guide v1.0b, Pg 3

La figura 63 presenta la PDU de modbus serial junto con la cabecera MBAP. La cabecera MBAP utiliza para gestionar el control de errores los métodos de control de errores de la capa de enlace del estándar Ethernet TCP/IP, a demás esta cabecera utiliza la unidad de identificación (unit.ID) para direccional la trama ADU de modbus TCP/IP

4.4.1 Cabecera MBAP: La cabecera del protocolo modbus de aplicación (MBAP), tiene una longitud de 7 bytes e incluye los siguientes campos: Unidad de identificación (unit ID), longitud de campo (Length Field), identificador del protocolo (Protocol Identifier) e identificador de la transacción (Transaction Identifier). Ver figura 64

Figura 64: Cabecera MBAP



Fuente: “Introduction to modbus TCP/IP acromag pg 20, www.acromag.com”

4.4.1.1 transaction identifier (2 bytes): Este campo es usado para sincronizar las transacciones²⁹ cuando se envían mensajes múltiples a lo largo de la misma conexión TCP por un cliente sin esperar una respuesta.

4.4.1.2 Protocol Identifier (2 bytes): Este campo es siempre 0 para servicios Modbus y esta reservando para futuras aplicaciones.

4.4.1.3 Length (2 bytes): Este campo es un recuento de bytes de los campos de la unidad de identificación (1 byte), código de función (1 byte) y el campo de los datos (0 hasta 252 bytes).

²⁹ “sincronizar las transacciones” lo que se quiere decir es que este parámetro se usa para asociar cada respuesta a su respectiva solicitud

4.4.1.4 Unit Identifier (1 byte): Este campo es usado para direccionar los datos junto con el código de función de modbus serial.

4.5 PROTOCOLO ETHERNET

Ethernet proporciona servicios correspondientes a las capas 1 y 2 del modelo de referencia OSI y la norma IEEE 802.3 especifica la capa física capa 1 del modelo OSI y el canal de acceso de datos (capa 2) del modelo OSI

La norma IEEE 802.3 es un método de establecimiento de comunicaciones físicas a través de una red de área local (LAN). Esta norma Especifica el protocolo de transporte de información del nivel físico dentro de una arquitectura de red a capas, tal como TCP/IP.

A continuación se presenta las diferentes alternativas de Ethernet basadas en normas IEEE 802.3.

10BASE2: 10 Mbit/s sobre coaxial fino. Longitud máxima del segmento 185 metros

10BASE5: 10 Mbit/s sobre coaxial grueso. Longitud máxima del segmento 500 m

10BASET: 10 Mbit/s sobre par trenzado UTP. Longitud máxima 100 metros

10BASEF: 10 Mbit/s sobre fibra óptica. Longitud máxima del segmento 1000 m

100BASETX/T4/FX: 100Mbit/s, es la Fast Ethernet

1000BASEX/T: Gigabit Ethernet.

El formato de la trama de IEEE 802.3 lo componen los siguientes elementos:

Preámbulo. Tamaño 7 bytes: se utiliza una secuencia de 7 bytes (56 bits) para sincronizar y estabilizar el medio físico antes de iniciar la transmisión de datos, estos bits se transmiten en orden de izquierda a derecha y son codificados en manchester para formar una onda periódica . El patrón del preámbulo es:

10101010 10101010 10101010 10101010 10101010 10101010 10101010

Inicio de Trama (Start Of Frame). Tamaño 1 byte: el byte de inicio de trama se utiliza para identificar cuando termina el preámbulo y cuando comienzan los datos, el byte de inicio de trama es muy similar a los bytes del preámbulo. Se diferencia en que este termina con “11” en vez de “10”.

Dirección de destino Tamaño 6 bytes: en este campo se coloca la dirección MAC³⁰ del equipo destino. Cada estación examina este campo para determinar si debe aceptar la trama (si es la estación destinataria).

Dirección de origen: tamaño 6 bytes. El sistema emisor coloca aquí su propia dirección MAC

Longitud de la trama: tamaño 2 bytes. Este campo identifica la longitud de datos, esta debe ser menor o igual a 1500 bytes

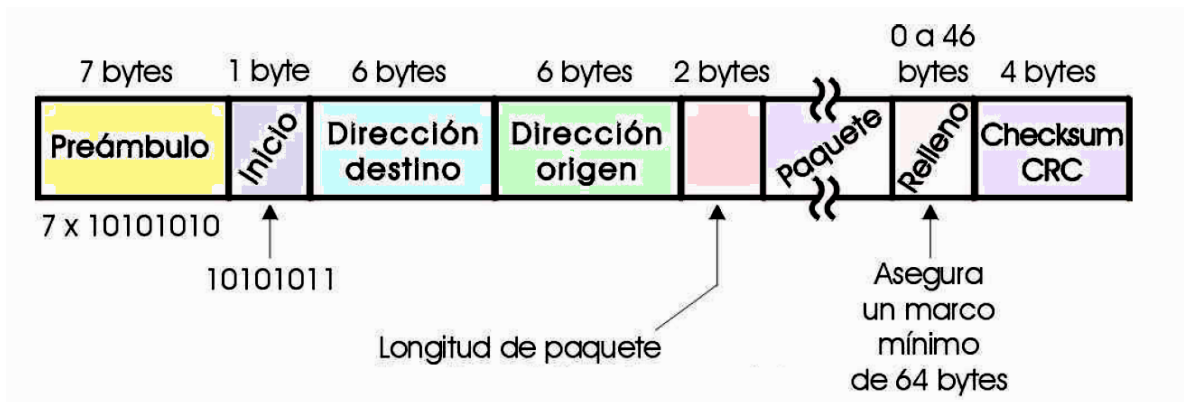
Paquete: tamaño de 0 a 1500 bytes. Esta campo lo constituye los datos + la cabecera de transporte + la cabecera de red.

Relleno: tamaño de 0 a 46 bytes. Este campo se utiliza cuando la trama Ethernet no alcanza los 64 bytes mínimos para que no se presenten problemas de detección de colisiones.

Secuencia de verificación de trama (FCS): tamaño 4 bytes. Contiene un valor de verificación CRC (Control de Redundancia Cíclica), creado por el dispositivo emisor y recalculado por el dispositivo receptor para verificar la existencia de tramas dañadas.

³⁰ Dirección MAC: es un identificador de 6 bloques hexadecimales (48 bits) que sirve para identificar de forma única los dispositivos en una red ethernet. Se conoce también como la dirección física de cada dispositivo.

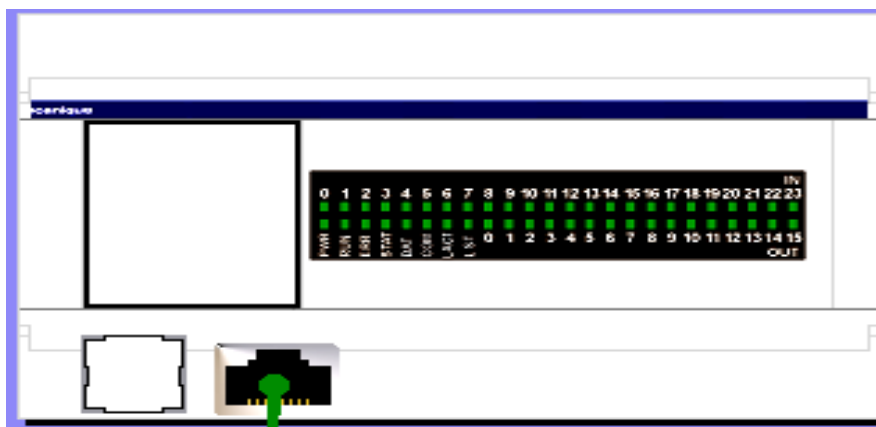
Figura 65: Trama IEEE 802.3



Fuente: <http://www.uruguayeduca.edu.uy/Userfiles/P0001%5CFile%5CIEEE802.3>

4.6 COMUNICACIÓN ENTRE EL SOFTWARE TWIDOSUITE Y PLC TWIDO

Figura 66: Puerto Ethernet:



Fuente: "software twidosuite"

El autómata Twido con Ref TWDLCAE40DRF tiene un puerto terminal RJ 45 que sigue las especificaciones 10BASET de la norma IEEE 802.3 del protocolo Ethernet, el cual se conoce como el puerto 3. Este puerto está diseñado para la comunicación entre el software de programación TwidoSuite (en el PC) y el autómata (o PLC) Twido usando el protocolo modbus TCP/IP. Ver figura 66

4.7 CONFIGURACIÓN MODBUS TCP/IP EN EL PLC TWIDO TWDLCAE40DRF

Para configurar el autómata Twido con el fin de utilizar una conexión Ethernet para enviar y recibir caracteres de datos mediante el protocolo modbus TCP/IP, se debe seguir los siguientes pasos:

Configurar el puerto Ethernet para el protocolo modbus TCP/IP mediante el software TwidoSuite.

Crear en la aplicación una tabla de envío/recepción que será utilizada por la instrucción EXCHx o utilizar la opción de los macros comm que presenta el software twidosuite para este mismo propósito.

4.7.1 Configuración del puerto Ethernet con el software Twidosuite: el puerto Ethernet, se configura como puerto 3 en el software Twidosuite y se utiliza para la comunicación modbus TCP/IP. Si se desea configurar el puerto ethernet para una aplicación modbus TCP/IP se sugiere seguir estos pasos:

Construir la red Modbus TCP/IP con el/los cliente/clientes y el/los servidor/servidores en la ventana describir del software TwidoSuite

Seleccionar Puerto 3.

Acceder a la ventana **configuración**, asociado a los parámetros de hardware de conexión Modbus TCP/IP, para ello se debe dar clic en el puerto Ethernet y configurar los siguientes parámetros de esta ventana:

Configuración de la dirección IP, donde se encuentra los siguientes parámetros: dirección IP, mascara subred y dirección Gateway.

Tiempo de inactividad para la conexión modbus TCP/IP: este tiempo es el tiempo máximo que espera el cliente/clientes para comunicarse con el servidor/servidores. Configurar los dispositivos remotos. Para configurar los dispositivos remotos es necesario configurar por cada dispositivo remoto en el índice que presente la tabla en la ventana de configuración del software de twidosuite los siguientes parámetros: dirección IP de los servidores, identificador de la unidad "unit ID" y tiempo de espera de conexión usualmente "100ms"

4.7.2 Crear en la aplicación una tabla de envío/recepción que será utilizada por la instrucción EXCHx: Para esta aplicación hay que tener en cuenta varios factores entre los cuales están:

4.7.2.1 Protocolo Modbus TCP/IP en el cliente: El modo de funcionamiento ModbusTCP/IP en el cliente permite al autómatas enviar una solicitud Modbus a un servidor y esperar una respuesta. Para ello es necesaria la instrucción EXCHx. Existe una tabla de palabras (tabla de envío y recepción) asociada a la instrucción EXCHx la cual está formada por tablas de control, de envío y de recepción. Ver sección 3.10.2 de este documento.

4.7.2.2 Intercambio de mensajes modbus TCP/IP en Twidosuite: Para que el intercambio de mensaje en twidosuite tenga éxito puede incorporarse en su programa dos servicios de comunicación:

Instrucción EXCHx: para enviar/recibir mensajes.

El bit %MSGx.D: para garantizar que una instrucción EXCHx comience a ejecutarse si y solo si se ha finalizado la ejecución de otra instrucción EXCHx

Instrucción EXCHx

La instrucción EXCHx permite al autómeta Twido enviar o recibir información dirigida o procedente de los dispositivos Modbus.

El usuario programa una tabla de palabras (%MWi: L) “tablas de envió/recepción” que contiene información de control y los datos que se van a enviar y recibir.

Sintaxis: [EXCHx %MWi: L].

Donde: x = número de puerto (3)

L = número de palabras de la tabla de envío y de recepción

Figura 67: BIT %MSGx.D

Entrada/salida	Definición	Descripción
%MSGx.D	Comunicación completa	0: solicitud en curso. 1: comunicación finalizada si se produce el final del envío, se recibe el carácter final, se produce un error o se restablece el bloque.

Fuente: “TwidoSuite V2.0 Guía de programación 09/2007, Pg 132”

4.7.2.3 Errores y condiciones de funcionamiento para modbus TCP/IP: Si se desea saber qué tipo de error sucedió, es necesario consultar la palabra de sistema %SW65 que contiene el código de error EXCHx del puerto 3.

A continuación se describe la palabra de sistema con su código de error EXCHx.

Tabla 29: Código error EXCHx para modbus TCP/IP:

Palabra de sistema	Función	Descripción	Control
%SW65	Código de error del bloque EXCH3	El código de error EXCH3 sólo se aplica en autómatas Twido TWDLCAE40DRF habilitados para Ethernet. 1-4, 6-13: consulte %SW63. (Tenga en cuenta que el código de error 5 no es válido y se sustituye con los códigos de error 109 y 122 específicos de Ethernet que se describen a continuación.) A continuación, se indican los códigos de error específicos de Ethernet: 101: la dirección IP no existe. 102: se ha perdido la conexión TCP. 103: no hay ranuras disponibles (todos los canales de conexión están ocupados). 104: no hay red. 105: no se puede alcanzar la red. 106: la red perdió la conexión durante el reinicio. 107: conexión cancelada por el dispositivo peer. 108: conexión restablecida por el dispositivo peer. 109: temporización de conexión transcurrida. 110: intento de conexión rechazado. 111: el ordenador principal no funciona. 120: índice desconocido (el dispositivo remoto no está indexado en la tabla de configuración). 121: grave (MAC, chip, IP duplicada) 122: recibiendo temporización transcurrida tras enviar los datos. 123: inicio de Ethernet en progreso (¹) Esta palabra de sistema no está disponible en el autómata Twido Extreme TWDLEDCK1	S

Nota: “S” significa controlado por el sistema

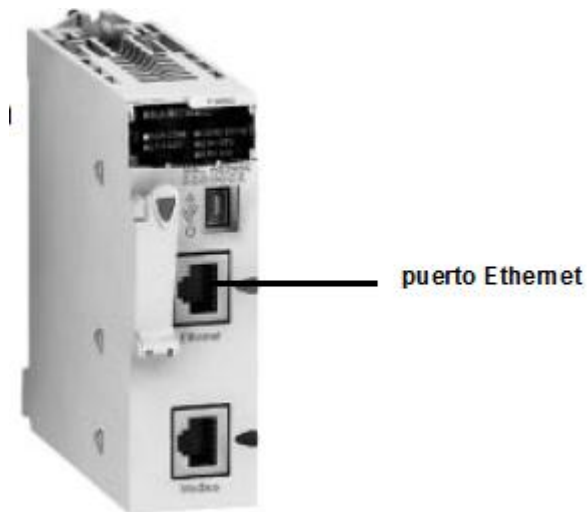
Fuente: “TwidoSuite V2.0 Guía de programación, Pg 702” Modificado por autores

4.8 COMUNICACIÓN MODBUS TCP/IP CON EL PLC M340

4.8.1 Descripción puerto Ethernet del CPU Modicon M340 BMX P34 2020: La comunicación por modbus TCP/IP en el PLC Modicon M340 es posible gracias a la CPU BMX P34 2020 la cual tiene incorporado el puerto Ethernet para transmitir

los datos del protocolo de comunicación modbus TCP/IP a través de un conector RJ 45 hembra que sigue las especificaciones 10BASET de la norma IEEE 802.3.

Figura 68: Puerto Ethernet:



Fuente:http://instrumentacionycontrol.net/Descargas/Descargas/SCHNEIDER/Automatas_PLC/IyCnet_Hardware_Modicon_M340.pdf

4.8.2 programación del PLC M340 mediante unity pro, con modbus TCP/IP:

para establecer una comunicación modbus TCP/IP entre el PLC M340 y el software unity pro es necesario configurar el software unity pro de la siguiente manera:

4.8.2.1 Definir el hardware en el software, para modbus TCP/IP: consiste en describir el PLC Modicon M340 en el software unity pro tal como está conformado físicamente, es decir se tiene que definir el tipo de rack; se entiende por tipo de rack cuantos módulos, puede albergar el mismo; también hay que especificar el

tipo o clase de cada uno de los módulos (módulo de comunicaciones, módulo de entradas digitales, módulos entradas y salidas analógicas, la fuente, etc.) conectados al rack y su orden o lugar en dicho rack

4.8.2.3 Configurar el puerto Ethernet en el software unity pro: el puerto se configura en una ventana que tiene dos opciones, la primera si el PLC va a ser maestro (cliente) y la segunda si el PLC va a ser esclavo (servidor) similar al software twidosuite; además en esta misma ventana se tienen que configurar los parámetros típicos de modbus TCP/IP para una red Ethernet como: la velocidad de transmisión de datos, el modo de transmisión de datos (RTU o ASCII), bit de paridad o no paridad, tiempo entre tramas del mensaje, timeout y bits de parada.

4.8.2.4 configurar los bloques ADDM, READ_VAR, WRITE_VAR, en TCP/IP: estos bloques funcionales se utilizan para el programa correspondiente a la parte de gestión de comunicación y sirve para intercambiar palabras de memoria (%MWi) o bits (%Mi) entre el maestro y los esclavos.

Cada bloque tiene una tarea específica en el intercambio de palabras de memorias o bits, estas tareas son:

BLOQUE ADDM: dirección del servidor a quien se le envía la información

BLOQUE READ_VAR: para lectura de variables.

BLOQUE WRITE_VAR: para escritura de variables.

Cada función conlleva una serie de parámetros encargados de configurar los bloques que sirven en la comunicación modbus. Estos parámetros se presentan en las tablas 30, 31 y 31.

Tabla 30: Descripción de los parámetros de la función ADDM

PARÁMETROS	TIPO	DESCRIPCIÓN
IN	STRING	Dirección de dispositivo en una conexión bus o serie. La sintaxis de la dirección tiene la siguiente estructura donde: r : Número de bastidor del procesador. Siempre = 0. m : Número de ranura del procesador del bastidor, siempre = 0 c : Número de canal, siempre = 0. nodo : Número de esclavo al que se envía la solicitud.
OUT	ARRAY [0...7] “arreglo de 8 palabras tipo entero”	Matriz que representa la dirección de un dispositivo en un nuevo formato. Este parámetro se puede utilizar como parámetro de entrada para varias funciones de comunicación.

Fuente: “Modicon M340 con Unity Pro. Conexión serie. Manual del usuario, pg.93”

Tabla 31 Descripción de los parámetros de la función READ_VAR

PARÁMETRO	TIPO	DESCRIPCIÓN
ADR	ARRAY [0...7] OF INT	Dirección de la entidad de destino proporcionada por el parámetro OUT de la función ADDM.
OBJ	STRING	Tipo de objeto que se va a leer. Los tipos disponibles son: %M : bit interno %MW : palabra interna %I : bit de entrada externa

		%IW: palabra de entrada externa
NUM	DINT	Dirección del primer objeto que se va a leer.
NB	INT	Número de objetos consecutivos que se van a leer.
GEST	ARRAY [0...3] OF INT	Tabla de gestión de intercambios que se compone de las siguientes palabras: Primera palabra: Palabra administrada por el sistema que se compone de dos bytes: ☐ Byte de mayor valor: Número de función ☐ Byte de menor valor: Bit de actividad (rang 0) y bit de cancelación (rang 1). Segunda palabra: Palabra administrada por el sistema que se compone de dos bytes: ☐ Byte de mayor valor: Informe de operación ☐ Byte de menor valor: Informe de comunicación Tercera palabra: Palabra gestionada por el usuario, que define el tiempo de respuesta Máximo con una base de tiempo de 100 ms. Cuarta palabra: Palabra gestionada por el sistema, que define la duración del intercambio.
RECP	ARRAY [n..m] OF INT	Tabla de palabras que contiene el valor de los objetos leídos.

Fuente: "Modicon M340 con Unity Pro. Conexión serie. Manual del usuario, pg.94"

Tabla 32: Descripción de los parámetros de la función WRITE_VAR

PARÁMETRO	TIPO	DESCRIPCIÓN
ADR	ARRAY [0...7] OF INT	Dirección de la entidad de destino proporcionada por el parámetro OUT de la función ADDM.
OBJ	STRING	Tipo de objeto que se va a escribir. Los tipos disponibles son: %M: bit interno %MW: palabra interna Nota: WRITE_VAR no puede utilizarse para las variables %I y %IW
NUM	DINT	Dirección del primer objeto que se va a escribir.
NB	INT	Número de objetos consecutivos que se van a escribir
EMIS	ARRAY [n..m] OF INT	Tabla de palabras que contienen el valor de los objetos que se van a escribirse.
GEST	ARRAY [0...3] OF INT	Tabla de gestión de intercambios que se compone de las siguientes palabras: Primera palabra: Palabra administrada por el sistema que se compone de dos bytes: ☐ Byte de mayor valor: Número de función ☐ Byte de menor valor: Bit de actividad (rang 0) y bit de cancelación (rang 1). Segunda palabra: Palabra administrada por el sistema que se compone de dos bytes: ☐ Byte de mayor valor: Informe de operación ☐ Byte de menor valor: Informe de comunicación Tercera palabra: Palabra gestionada por el usuario, que define el tiempo de respuesta Máximo con una base de tiempo de 100 ms. Cuarta palabra: Palabra gestionada por el sistema, que define la duración del intercambio

Fuente: “Modicon M340 con Unity Pro. Conexión serie. Manual del usuario, pg.95”

4.8.2.5 Elaborar un programa para el control de la red: este programa controla todas las comunicaciones entre el maestro y los esclavos, además determina que funciones y tareas se van a realizar para determinada actividad.

El software unity pro tiene 5 clases de lenguajes, se recomienda el lenguaje ladder por ser uno de los lenguajes más utilizados en la automatización.

5. PROTOCOLO DE COMUNICACIÓN CANopen

5.1 QUE ES CANopen:

CANopen es el estándar internacional (EN 50.325-4) protocolo de alto nivel basado en CAN (Controller Area Network). CAN nació en la década de los 80 con el objetivo de gestionar el bus de campo del mismo nombre para reducir el cableado excesivo en los vehículos; no obstante su implementación generalizada se dio en la década de los 90 por fabricantes como Mercedes-Benz, Volvo, Renault y Citroen entre otros. CAN comprende la capa 1 (física), la capa 2 (enlace de datos) y la capa de aplicación (CAL) del modelo de referencia OSI. La capa de aplicación (CAL) define un conjunto de servicios y protocolos pero no define su contenido por lo cual necesita un protocolo de comunicación que complemente esta tarea, es aquí donde aparece CANopen, quien define el contenido de los protocolos y servicios de CAL, como el perfil de comunicación, dispositivos estandarizados, perfil de interfaz y el perfil de la aplicación, CANopen es usado en sectores tan diversos como: dispositivos médicos, vehículos, vehículos ferroviarios y marítimos, en la automatización de edificios, equipos en el control de energía etc.

5.2 CARACTERÍSTICAS GENERALES DE CANopen

CANopen es un bus de tipo serie, compuesto por CAN y una serie de servicios de comunicaciones. Trabaja en modo broadcast y es muy utilizado en transmisiones de corto alcance debido a su efectividad.

Sus principales características son:

Velocidad de hasta 1 Mbit/s a 40 metros

CANopen carece de un elemento central de control como un maestro, por esta razón cualquier equipo puede enviar y recibir datos por cuenta propia.

El acceso al bus CAN se realiza mediante un sistema de prioridades que elimina los retrasos en la transmisión. Este sistema de prioridades viene establecido por un elemento identificador denominado COB-ID³¹.

El control de errores desconecta cualquier dispositivo defectuoso para mantener la comunicación al 100% entre el resto de los dispositivos.

Los mensajes en CANopen se reconocen mediante un identificador de mensaje con el cual se determina la prioridad asignada de dicho mensaje.

La longitud de datos está limitada en 8 bytes.

Los componentes electrónicos tienen una elevada inmunidad a interferencias electromagnéticas y al deterioro físico, debido a que fueron diseñados para aplicaciones en los vehículos.

Los datos del bus son reconocidos o rechazados por todos los dispositivos.

Los dispositivos con defecto se aíslan de forma automática y se pueden agregar dispositivos sin necesidad de tener que hacer cambios para el reconocimiento de estos.

Un dispositivo tiene la habilidad de pedir información de otros dispositivos.

5.3 DESCRIPCION DE CAN

CAN significa Controllor Area Network (Red de área de control). El bus CAN fue desarrollado por Bosch GmbH en Alemania y se publicó en 1991. Ha sido utilizado principalmente para la comunicación en automoción, pero debido a su gran utilización se ha introducido en la industria como un estándar. El bus accede al medio con una técnica denominada como CSMA/CD+CR o “Carrier Sense, Multiple Access/Collision Detection + Collision Resolution” (Acceso múltiple con

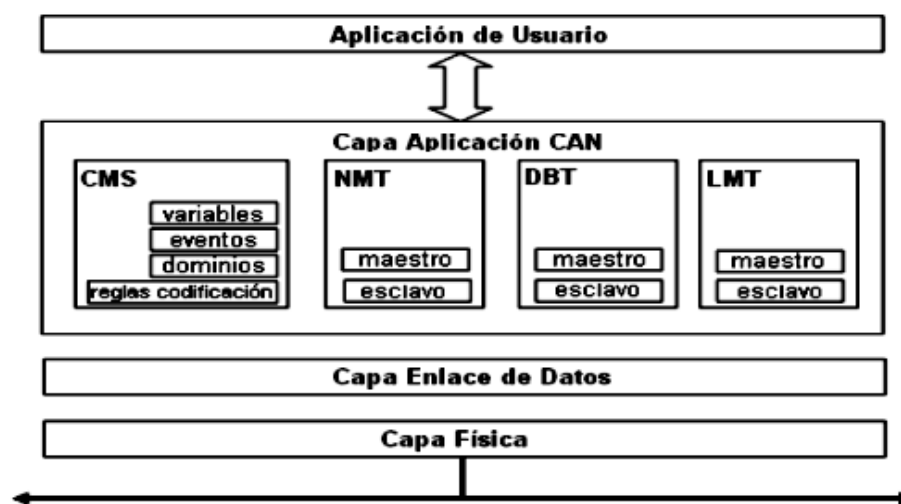
³¹ Identificador de objeto de comunicación

detección de portadora, detección de colisión más resolución de colisión). En términos generales el acceso al medio de CAN consiste en que todos los dispositivos verifican si el bus está ocupado antes de transmitir su información. Cuando ocurre una colisión de mensajes en el bus, empieza el arbitraje para la recepción del mensaje; el mensaje de prioridad más alto se recibe primero y así sucesivamente, hasta que todos los mensajes hayan sido recibidos.

El protocolo CAN tiene un bit inteligente no destructivo, esto significa que cada bit de un mensaje es comparado con cada bit de otro mensaje en un momento determinado, pero el mensaje con la prioridad más alta no se destruye y se transmite; el mensaje que no gana el arbitraje de bus se detiene y se retransmite. Éste es un punto importante que ayuda a minimizar el tiempo de fuera de servicio del bus y aumentan al máximo el uso eficaz del ancho de banda disponible.

5.4 ARQUITECTURA DEL PROTOCOLO CAN

Figura 69: Arquitectura del protocolo CAN

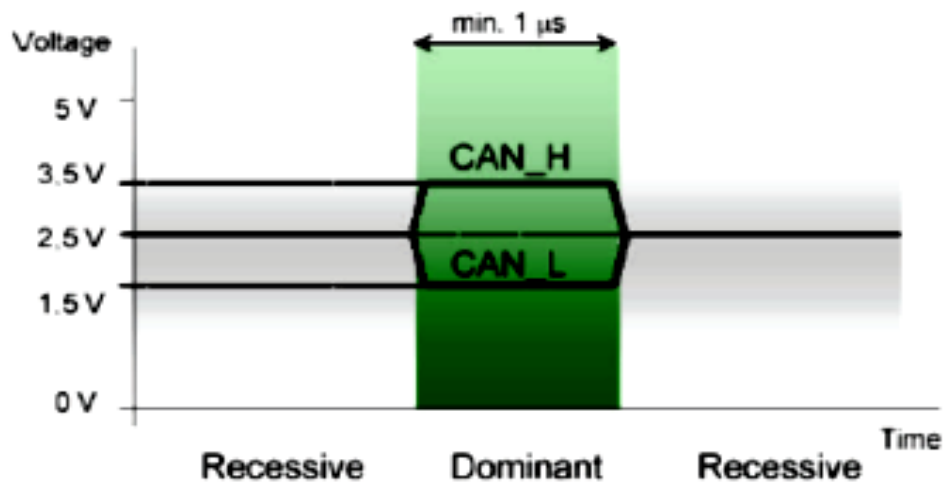


Fuente: "Capítulo 3 protocolos de capas superiores, Pg 126"

La arquitectura basada en CAN corresponde a una serie de protocolos que conforman las capas 1, 2 y 7 de acuerdo al modelo de capas ISO/OSI. Las capas 1 y 2 representan el estándar ISO99-1 e ISO99-2, y la capa 7 el estándar CAN CIA-200.

5.5 CAPA FÍSICA:

Figura 70: Niveles de tensión:



Fuente: <http://iaci.unq.edu.ar/materias/laboratorio2/transparencias%5Ccan.pdf>

La ISO 11898 define la capa física de CAN. Esta capa física es una interfaz de 2 hilos en modo diferencial con dos estados lógicos: dominante bajo y recesivo alto. Estado lógico "0" dominante, se produce cuando "CAN_H - CAN_L es igual a 2"

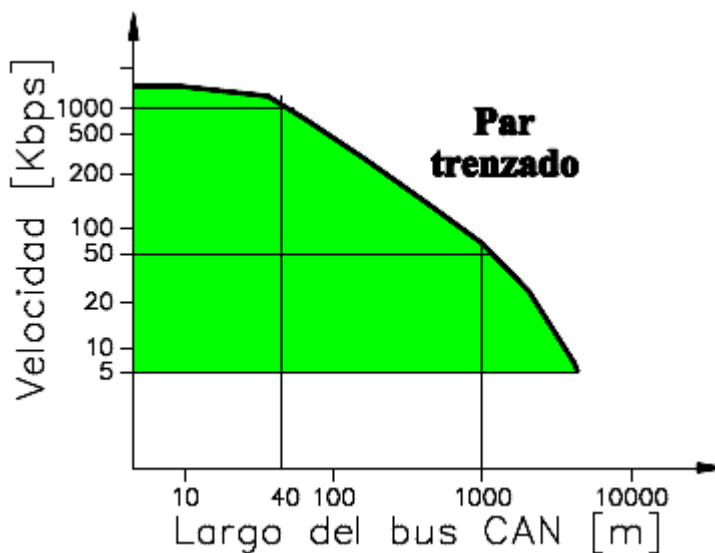
con valores domíniales, es decir, $CAN\_H = 3.5$ y $CAN\_L = 1.5$. También se produce cuando $2 \text{ volts} > CAN\_H - CAN\_L > 0.9 \text{ Volts}$.

El estado lógico "1" recesivo, se produce cuando $CAN\_H - CAN\_L$ es igual a 0" con $CAN\_H = 2.5$ y $CAN\_L = 2.5$ Volts. También se produce cuando el voltaje de $0 \text{ volts} < CAN\_H - CAN\_L < 0.5 \text{ volts}$. Ver figura 70

La señal se transmite por un cable par trenzado apantallado (STP) o un cable par trenzado no apantallado (UTP) o un cable plano (cinta). Cada nodo usualmente utiliza un conector de 9-pin subD "DB9".

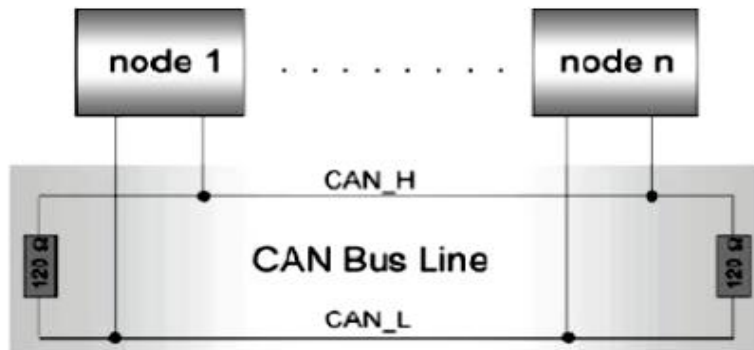
La velocidad de transmisión depende de la longitud del cable a alta velocidad hasta 1 Mbit/s sobre distancias de 40 m y a baja velocidad 50 kbits/s sobre distancias de 1Km. Debido a los retardos, impedancias y reflexión en la línea es necesario colocar en los extremos del bus impedancias de carga de 120 ohms para solucionar estos inconvenientes y estabilizar la red. Ver figura 71

Figura 71: velocidad de transmisión vs longitud de la red



Fuente: <http://iaci.unq.edu.ar/materias/laboratorio2/transparencias%5Ccan.pdf>

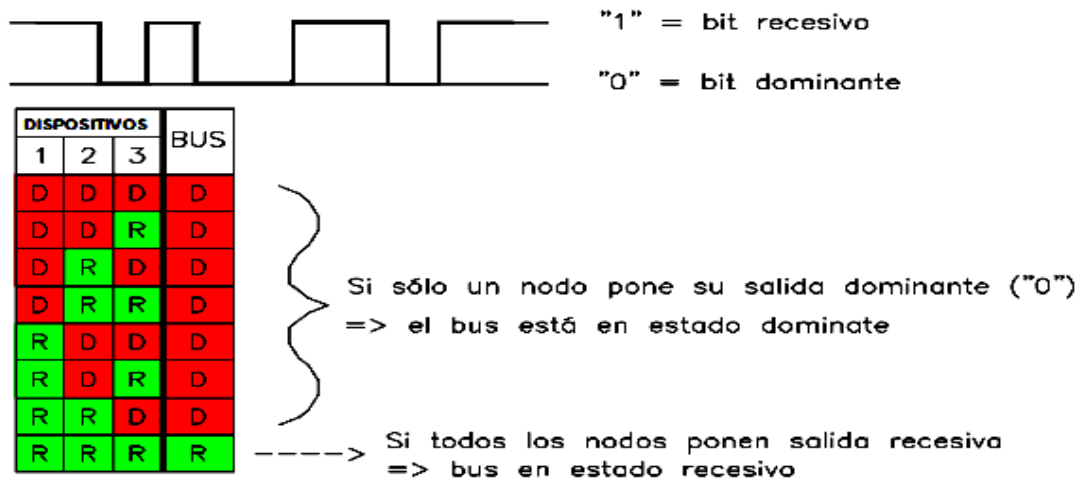
Figura 72: Topología del bus en la capa física



Fuente: <http://ocw.um.es/ingenierias/sistemas-embebidos/material-de-clase-1/s.pdf>

5.6 CAPA DE ENLACE:

Figura 73: Estados lógicos de la red CAN



Fuente: <http://iaci.unq.edu.ar/materias/laboratorio2/transparencias%5Ccan.pdf>

5.6.1 Principios de arbitraje del bus CAN: El principio de arbitraje está basado en el método “CSMA/CD+CR” Este principio consiste en una comparación del nivel de bit transmitido por un dispositivo con el nivel monitoreado en el bus realizado por cualquiera de los dispositivos transmisores. Durante la transmisión del campo arbitraje, el transmisor comprueba en cada bit si todavía está autorizado para transmitir o si está transmitiendo otra estación de mayor prioridad, los dispositivos transmisores con mensajes que todavía no están autorizados para transmitir se convierten inmediatamente en receptores y posteriormente repetirán su intento de transmisión. Ver figura 73

5.6.2 Tipo De Tramas: Existen cuatro tipos de tramas y una intertrama:

5.6.2.1 Trama De Datos: Se utiliza para enviar datos de un dispositivo a otro(s). Es el tipo de trama que más frecuentemente se utiliza en una red CAN. Una trama de datos contiene los siguientes campos de bit: Inicio de trama (SOF), Campo de arbitraje, Campo de Control, Campo de Datos, Campo CRC, Campo de Acuse de Recibo (ACK) y Campo de Fin de Trama (EOF).

Bit de Inicio de Trama (SOF): Este bit marca el inicio de una trama de datos o de una trama remota y es representado por un bit dominante. Un dispositivo puede comenzar a transmitir solo cuando el bus esta libre. Todos los dispositivos se sincronizan con el flanco ascendente del bit SOF (Start of frames) de la estación que transmite primero. El fin de la trama es indicado por una sucesión mínima de 11 bits recesivos

Campo de Arbitraje: El campo de arbitraje está constituido por el campo identificador y el bit de Solicitud Remota de Transmisión (RTR). Una trama es identificada y su prioridad definida por el campo identificador. En el formato básico del sistema CAN la longitud de su campo identificador es de 11 bits, lo que significa que pueden existir 2048 (2^{11}) tramas diferentes. Los bits del campo identificador son transmitidos en orden desde el bit 10 al bit

0, transmitiéndose primero el bit más significativo (bit 10). El bit RTR identifica a la trama de datos y a la trama de solicitud remota. Cuando se transmite simultáneamente estas dos tramas este bit hace dominante la trama de datos y recesiva la trama de solicitud remota (Trama RTR).

Campo de control: este campo lo conforman 6 bits. El primer bit, Bit Identificador de Extensión IDE, distingue entre tramas de formato básico³² y tramas de formato extendido³³. Para tramas de formato básico este bit es transmitido dominantemente (bit con valor 0). Por tanto las tramas del formato básico prevalece sobre la extendida. El segundo bit (0) es reservado para futuras extensiones del protocolo CAN y es transmitido dominantemente hasta que su función sea definida. Los cuatro bits restantes se utilizan para indicar el número de bytes de datos transmitidos. Un valor de 0 corresponde a 0 bytes de datos, un valor de 8 corresponde a 8 bytes de datos. en CAN los valores mayores a 8 bytes se utilizan para propósitos especiales.

Campo De Datos: Este campo contiene los datos a ser transferidos dentro de una trama CAN y puede contener de 0 a 8 bytes. El bit más significativo es transmitido primero.

Campo CRC: consiste en una secuencia CRC de chequeo de 15 bits y un bit delimitador transmitido recesivamente. La secuencia CRC indica si la trama recibida fue afectada o no por perturbaciones del medio. La secuencia de chequeo de trama es derivada de un Código de Redundancia Cíclica que funciona eficientemente en tramas con menos de 127 bits.

Campo ACK: está formado por dos bits, el ACK SLOT y el ACK DELIMITER. El dispositivo transmisor envía en este campo dos bits recesivos y espera el ACK de recepción correcta de al menos un dispositivo receptor, de esta manera cualquier dispositivo receptor que ha recibido correctamente una trama lo reporta al dispositivo transmisor transformado el bit recesivo slot ACK por un bit dominante. El ACK DELIMITER siempre es un bit recesivo.

³² Las tramas del formato básico tienen un campo identificador de 11 bits

³³ Las tramas del formato extendido tiene un campo identificador de 29 bits con lo que pueden existir 512 millones de tramas distintas.

5.6.2.3 Trama de Error: durante la transmisión o recepción la detección de cualquier error de una trama de datos o remota es señalado por una trama de error que intencionalmente causa que el transmisor de la trama repita la transmisión de la misma. La detección de un error durante la transmisión o recepción de una trama de error también causa la transmisión de una nueva trama de error. Una trama de error contiene dos campos de bits. El primer campo (ERROR FLAG) está dado por la superposición de banderas de error transmitidas por uno o varios nodos. El segundo campo (ERROR DELIMITER) es el delimitador de error, que es una secuencia de 8 bits recesivos e indica el fin de la trama. El campo ERROR FLAG puede tener dos formas:

ACTIVE ERROR FLAG: consiste en 6 bits dominantes consecutivos y es transmitido por dispositivos cuyo estado es Error Activado³⁴.

PASSIVE ERROR FLAG: consiste en 6 bits recesivos consecutivos a. Esta bandera es transmitida por nodos cuyo estado es Error Pasivo³⁵.

La existencia de 6 bits consecutivos de igual polaridad viola la regla de relleno de bit³⁶ usada en la codificación del bus CAN, por esto las otras estaciones también detectan el error y comienzan a transmitir un ERROR FLAG por su cuenta.

5.6.2.4 Trama De Sobrecarga: Esta trama se utiliza para solicitar un retraso de la próxima trama de datos o trama remota por el dispositivo receptor (Trama de Sobrecarga Solicitada) o para señalar ciertas condiciones de error (Trama de

³⁴ Un dispositivo está en estado de error activo cuando su tasa promedio de detección de error está por debajo de un límite específico

³⁵ Un dispositivo está en estado de error pasivo cuando algunos o todos sus bits recesivos pueden ser sobre-escritos por bits dominantes de otros nodos

³⁶ El Principio el Relleno de Bits consiste en insertar un bit adicional dominante después de 5 bits consecutivos del mismo valor.

Sobrecarga Reactiva). las tramas de sobrecarga reactiva son transmitidas después de la detección de las siguientes condiciones de error:

Detección de un bit dominante durante los primeros dos bits del campo de interrupción³⁷.

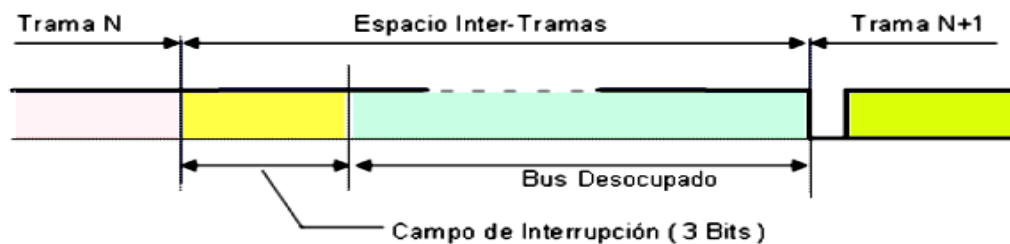
Detección de un bit dominante “0” en el último bit del campo EOF

Una trama de sobrecarga puede ser considerada como una forma especial de trama de error pues su bandera de sobrecarga (OVERLOAD FLAG) tiene la misma forma que el campo ACTIVE ERROR FLAG de la trama de error; lo que hace que los otros dispositivos detecten la condición de sobrecarga y empiecen a mandar banderas de sobrecargas. Su campo delimitador de sobrecarga consiste en 8 bits recesivos igual al ERROR DELIMITER. La trama de sobrecarga no causa la re-transmisión de una nueva trama de sobrecarga, como lo hace una trama de error. La generación de una trama de sobrecarga es limitada a condiciones muy específicas relacionadas al campo de interrupción.

5.6.2.5 Espacio Inter-Tramas: las Tramas de Datos y remotas son separadas de todas las tramas precedentes como tramas de datos, tramas remotas, tramas de error, tramas de sobrecarga por un espacio denominado inter-trama. Las tramas de error y de sobrecarga son transmitidas sucesivamente sin un espacio inter-tramas entre ellas. El espacio inter-tramas es un campo de bits recesivos que representa una distancia mínima de tres bits entre tramas llamado Campo de Interrupción, seguido por un campo de longitud aleatoria de nivel recesivo (bus desocupado).

³⁷ Un campo de Interrupción es un campo de bits recesivos que representa una distancia mínima de tres bits entre tramas,

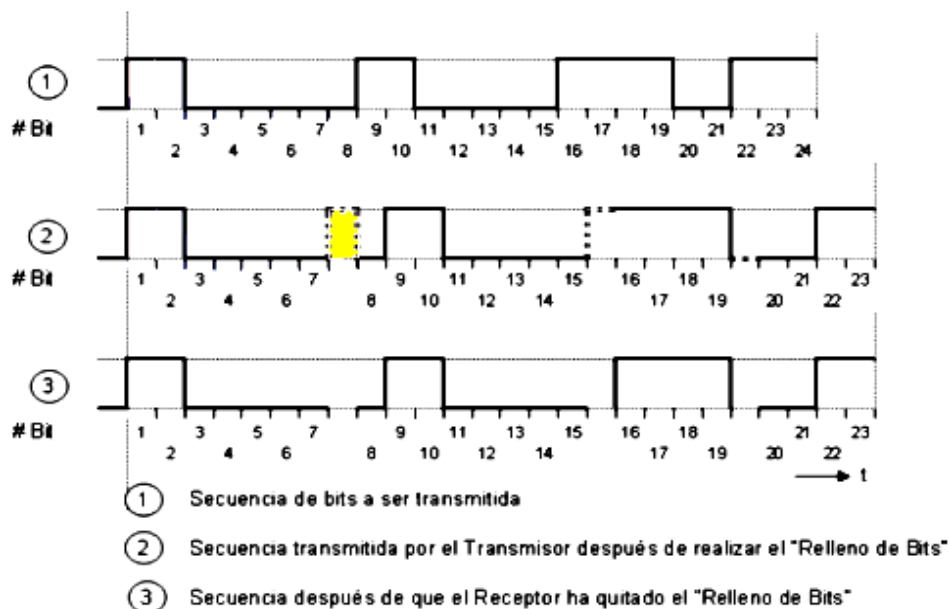
Figura 75: Formato De un espacio intertrama



Fuente: "capítulo 2, protocolo CAN, Pg 53"

5.6.3. Codificación de los bits: En el protocolo CAN los bits de una trama utilizan el código No Retorno a Cero (NRZ); lo que significa que en un intervalo de tiempo de bit, puede haber un nivel dominante o un nivel recesivo.

Figura 76: Codificación de bits en CAN



Fuente: "capítulo 2, protocolo CAN, Pg 56"

Para transmitir largas secuencias de bits de igual nivel es necesario insertar un bit adicional de nivel complementario³⁸ (Bit de Relleno) después de 5 bits, dentro del arreglo de bits, el cual servirá para obtener un número suficiente de transiciones para re-sincronizar los dispositivos receptores.

En las tramas remotas y de datos, los segmentos de trama: SOF, campos de arbitraje, de control, de datos, y de secuencia CRC son codificados con el método de relleno de bits; el resto de campos de bits (delimitador CRC, campos ACK y EOF) son de forma fija y no utilizan relleno de bits. Las tramas de sobrecarga y error son de forma fija y no utiliza el método de relleno de bits.

5.6.4 Tipos de errores: En CAN pueden darse 5 tipos de errores, estos son:

5.6.4.1 Bit error: un dispositivo que envía un bit por el bus también monitorea el valor de este en el bus. Cuando el valor monitoreado es distinto al valor enviado se produce esta clase de error.

5.6.4.2 Stuff error: este error se produce cuando aparece un sexto bit consecutivo con igual polaridad que los cinco anteriores.

5.6.4.3 CRC error: este error se produce cuando el cálculo del valor del campo con el código de redundancia cíclica realizado por el dispositivo transmisor es diferente al valor calculado por el dispositivo receptor.

5.6.4.4 Form error: se produce cuando algún campo de formato fijo como las tramas de sobrecarga y las tramas de error presenta uno o más bits no permitidos.

³⁸ Bit de nivel opuesto a los 5 bits consecutivos precedentes

5.6.4.5 Acknowledgement error: se da cuando el dispositivo receptor modifica el slot ACK para no confirmar la llegada de una trama. El transmisor pone “11” o sea, dos bits recesivos, si algún nodo no puede recibir la trama, no indica el ACK poniendo “01” en ese slot. Así, todos los nodos lo detectan y descartan la trama. Esto es justamente lo que permite que el broadcast y el multicast sean confiables.

5.6.5 Confinamiento de fallas: CAN utiliza un mecanismo de confinamiento de fallas que se implementa mediante una máquina de estados, y sirve para minimizar el efecto de los errores. Cada dispositivo del bus puede encontrarse en alguno de los siguientes estados

5.6.5.1 Error Active: un dispositivo en este estado puede tomar parte en cualquier comunicación, pero frente a un error, debe transmitir un “Active Error Flag”. Un flag de este tipo de error contiene 6 bits dominantes consecutivos, por ser todos los bits dominantes el dispositivo que transmite este tipo de error tiene la mayor prioridad en la comunicación, ganando siempre la oportunidad de transmitir cuando existe alguna comunicación en curso de otros dispositivos. En este estado los dispositivos receptores detectan un “Stuff Error”

5.6.5.2 Error Passive: un dispositivo en este estado también puede tomar parte en cualquier comunicación pero frente a un error, debe transmitir un “Passive Error Flag”. Un flag “bandera” de este tipo contiene 6 bits recesivos, por ser todos los bits recesivos el dispositivo que transmite este tipo de error tiene la menor prioridad en la comunicación, perdiendo siempre la oportunidad de transmisión cuando existe alguna comunicación en curso de otros dispositivos.

Un dispositivo se encuentra en este estado porque sospecha que él es el causante de los problemas que se están dando en el bus.

5.6.5.3 Bus Off: un dispositivo en este estado no puede participar en ninguna comunicación debido a que él se encuentra desconectado del bus. Un dispositivo se encuentra en este estado porque asume que él es el causante de los problemas en la comunicación.

5.7 CAPA DE APLICACIÓN DE CANopen:

CANopen utiliza un subconjunto de servicios y protocolos de CAL³⁹ (capa de aplicación CAN) y define su contenido, además CANopen utiliza otros tipos de servicios que no pertenecen a CAN, como el diccionario de objetos que es la herramienta principal de este protocolo. CAL fue una de las primeras especificaciones producidas por CiA (CAN in Automation), basada en un protocolo existente desarrollado originalmente por Philips Medical Systems. CAL a través de tipos de datos estándar, reglas de codificación de bits, objetos⁴⁰ de comunicación estandarizados y la definición de los correspondientes protocolos para el intercambio de estos objetos, permite la descripción de aplicaciones distribuidas de cualquier tipo basadas en CAN. CAL también especifica servicios y protocolos para la administración de la red y se basa en los siguientes servicios: CMS (Especificación de mensaje CAN), NMT (Administración de la red), DBT (Distribuidor identificador), LMT (Capa de Administración)⁴¹

³⁹ CAL define los servicios y los protocolos que utiliza en desarrollo de comunicación pero no define su contenido, por lo cual necesita de otro protocolo de comunicación superior, como CANopen

⁴⁰ Los objetos de comunicación estandarizados pueden ser datos, funciones o parámetros, con los que se pueden definir la funcionalidad de un dispositivo

⁴¹ Estos cuatro servicios se explicaran en detalle en la sección modelo de comunicaciones de este capítulo

5.8 MODELO DE OBJETO:

CANopen utiliza una metodología con la que describe los dispositivos basada en la definición de los perfiles. Cada perfil define un dispositivo estándar el cual, especifica la funcionalidad requerida para todo dispositivo que pertenezca a dicho perfil. Estos perfiles incluyen un mecanismo por el cual los fabricantes pueden incluir funcionalidad propia, extendiendo el perfil en cuestión, con lo que se garantiza que los perfiles estándar evolucionen y no se vuelvan obsoletos. La parte más importante del perfil de un dispositivo es la descripción del diccionario de objetos, que constituye la verdadera interface entre las aplicaciones y el bus CAN, desde el punto de vista de datos como de la configuración de los dispositivos.

La descripción de los perfiles consiste en un agrupamiento de objetos, único para cada perfil del dispositivo, accesible a través del bus CAN en una forma ordenada y predefinida. Cada objeto se direcciona utilizando un índice de 16 bits, con lo que se permite el acceso a elementos individuales de las estructuras de datos.

El rango relevante de objetos va desde el índice 1000 al 9FFF. Para cada dispositivo de la red existe un OD (diccionario de objetos), que contiene todo los parámetros que describen el dispositivo y su comportamiento en la red. Si la entrada corresponde a una variable estática el direccionamiento de las entradas del diccionario se realiza mediante el índice de 4 dígitos Hexadecimal, como en la tabla 33, si en vez de una entrada de variable estática se tiene una entrada de variable compleja es necesario un subíndice de 2 dígitos Hexadecimales para acceder a un miembro en particular. La tabla 34 presenta un ejemplo donde se presentan los parámetros de comunicación de una interface RS232 que corresponden a una variable compleja

Tabla 33: Estructura del diccionario de objetos:

INDICE ⁴²	OBJETO	DESCRIPCIÓN
0000	No usado	
0001 - 001F	Tipos de dato estáticos	Contiene definiciones de tipos de dato estándar como boolean, enteros, string, punto flotante, etc. Se incluyen como referencia, no pueden ser leídas ni escritas.
0020 - 003F	Tipos de dato complejos	Contiene definiciones de estructuras predefinidas, compuestas de tipos estáticos, comunes a todos los dispositivos.
0040 - 005F	Tipos de dato específicos del fabricante	Contiene definiciones de estructuras predefinidas, compuestas de tipos estáticos, específicas de un dispositivo en particular
0060 - 007F	Tipos de dato estáticos específicos del perfil del dispositivo	Definiciones de tipos de dato básicos específicos para el perfil del dispositivo.
0080 - 009F	Tipos de dato complejos específicos del perfil del dispositivo	Tipos de dato complejos específicos del perfil del dispositivo
00A0 – 0FFF	Reservado	
1000 – 1FFF	Rango para el perfil de comunicaciones	Contiene parámetros de configuración del bus CAN. Estas entradas del diccionario son comunes a todos los dispositivos
2000 – 5FFF	Rango para el perfil específico del fabricante	Contiene las extensiones al perfil estándar, realizadas por el fabricante.
6000 – 9FFF	Rango para perfiles de dispositivo estandarizados	Contiene todos los objetos de datos comunes a un tipo de perfil que pueden ser leídos o escritos desde la red. Algunas de las entradas son obligatorias mientras que otras son opcionales.
A000 – FFFF	Reservado	

Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, pg 8”

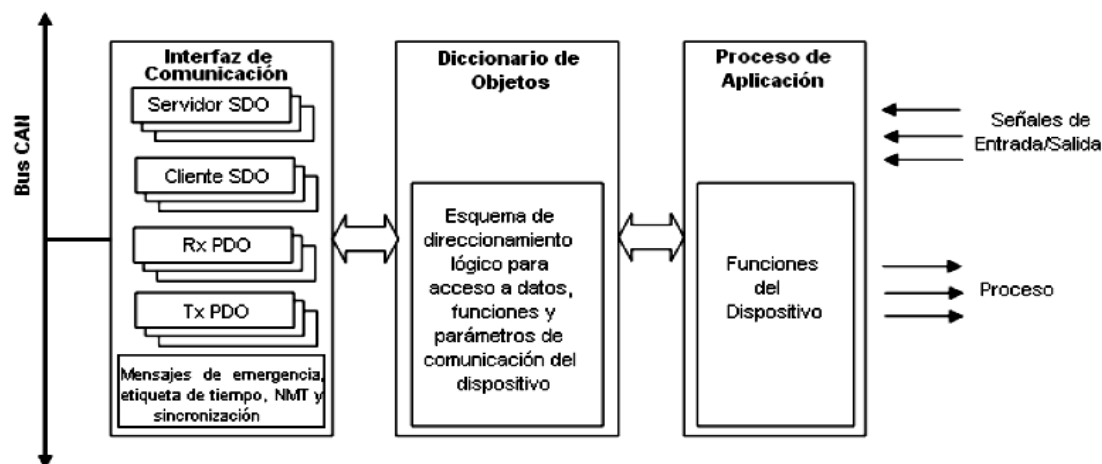
⁴² El índice se utiliza para que cada objeto se direcciona, permitiendo el acceso a elementos individuales de la estructura de datos. El índice de la tabla 33 se presenta de forma Hexadecimal

Tabla 34: Ejemplo de variable compleja:

ÍNDICE	SUBÍNDICE	VARIABLE	TIPO DE DATO
6092h	00h	Número de miembros	Unsigned ⁴³ 8
	01h	Baud Rate	Unsigned16
	02h	Número de Data Bits	Unsigned8
	03h	Número de Stop Bits	Unsigned8
	04h	Paridad	Unsigned8

Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, pg. 8”

Figura 77: Modelo de un dispositivo CANopen



Fuente: “capítulo 3 protocolos de capas superiores, pg. 133”

Para leer o escribir el número de bit de parada es necesario acceder a la entrada con índice 6092 hexadecimal y luego, al subíndice 03 hexadecimal.

⁴³ Unsigned es un tipo de dato básico que representa los datos a través de secuencia de bits de longitud n, donde n es el número de bits. Ejemplo un unsigned8 representa una secuencia de 8 bits(1 byte)

Un perfil de dispositivo, aparte de la descripción del diccionario de objetos, debe incluir: Los parámetros para la transmisión de datos del proceso controlado, un mapeo de los objetos de aplicación (presentes en el diccionario de objetos) de los mensajes involucrados en la transmisión. La Figura 77 muestra el modelo básico de un dispositivo CANopen con la interfaz de comunicación para conectar el dispositivo a la red, el proceso de aplicación y el diccionario de objetos que actúa como interfaz entre la red y el proceso de aplicación.

5.9 ESTRUCTURA DE LA COMUNICACION:

CANopen ofrece una estructura para la comunicación de dispositivos orientada a objetos de comunicación. Se entiende por objeto de comunicación o communication object (COB) una unidad de transporte en un bus CAN. Estos objetos se identifican mediante un COB-ID único que se corresponde con el valor del Campo de Arbitraje de la trama CAN. Ver la sección 5.6.2.1.

En CANopen los perfiles de los dispositivos constituyen la interface entre las aplicaciones del usuario y el bus CAN. Estos perfiles acceden al bus a través de un subconjunto de CAL (capa de aplicación de CAN). Esta capa está compuesta por los siguientes cuatro servicios:

5.9.1 Especificación de mensajes CAN (CMS): CMS define objetos de comunicación básicos que se implementa directamente a través de los servicios de la capa de enlace de datos y objetos de comunicación mejorados, estos requieren de la introducción de protocolos y servicios adicionales “los objetos CMS” Entre los objetos básicos se tienen: variables solo de lectura, variables solo de escritura y eventos no controlados, mientras que entre los mejorados: variables multiplexadas, variables de lectura-escritura, dominios y eventos controlados.

CMS define 8 niveles de prioridad en sus mensajes, cada uno con 220 COB-IDs⁴⁴, ocupando desde el 1 al 1760. Los identificadores restantes (0, 1761-2031) se reservan para NMT, DBT y LMT, como muestra la Tabla 35. En una red CAN el mensaje con el COB-IDs más pequeño es el de mayor prioridad.

Tabla 35: Distribución de los COB-IDs en CAL

COB-ID ⁴⁵	UTILIZACIÓN	CANTIDAD
0	NMT Start/Stop	1
1 - 220	Objetos CMS Prioridad 0	220
221 - 440	Objetos CMS Prioridad 1	220
441 - 660	Objetos CMS Prioridad 2	220
661 - 880	Objetos CMS Prioridad 3	220
881 - 1100	Objetos CMS Prioridad 4	220
1101 - 1320	Objetos CMS Prioridad 5	220
1321 - 1540	Objetos CMS Prioridad 6	220
1541 - 1760	Objetos CMS Prioridad 7	220
1761 - 2015	Monitoreo dispositivos NMT	255
2016 - 2031	Servicios de NMT, LMT y DBT	16

Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, pg. 5”

5.9.2 Administración de red (NMT): los servicios y protocolos NMT proporciona la configuración, administración y monitoreo de los dispositivos, de tal forma que puede realizar las tareas de inicializar, arrancar, parar o detectar fallos en los dispositivos. Se basa en el concepto de maestro-esclavo, donde un dispositivo en particular, llamado “Maestro NMT” toma el papel de coordinador mientras que el

⁴⁴ Identificador de objetos de comunicación

⁴⁵ El identificador COB-ID es transmitido en el campo identificador de trama de datos CAN. la tabla 35 está representada en decimal

resto actúa como “Esclavos NMT”. Con los servicios de control de módulo el maestro NMT inicializa a los esclavos NMT permitiendo la comunicación entre ellos. A través de los servicios de control de errores, el maestro NMT detecta fallas remotas en la red CAN. Por medio de los servicios de configuración un maestro NMT puede cargar y descargar datos de configuración como parámetros, datos o código ejecutable desde o hacia un dispositivo.

5.9.3 Administración de capas (LMT): Permite leer, cambiar, y configurar ciertos parámetros de las capas de CAN; por ejemplo: un dispositivo (Maestro NMT) puede establece parámetros específicos de una capa y transmitirlos a otros dispositivos. En CANopen, este servicio recibe el nombre de LSS (Layer Setting Services). Permite modificar Dispositivo-ID⁴⁶ (objeto con índice 100B), parámetros de temporización de la capa física y la dirección NMT de un esclavo NMT (objeto con índice 1018).

5.9.4 Asignación de identificadores de mensaje (DBT): su responsabilidad es asignar en forma dinámica identificadores CAN de 11 bits, también llamados COB-IDs (*Communication Object Identifier*), a los COBs usados por el servicio CMS. Como los COB-IDs deben ser únicos, este servicio debe trabajar sincronizado con los DBT de otros dispositivos conectados a la red CANopen. También se basa en el modelo maestro-esclavo, existiendo sólo un DBT maestro.

5.9.5. Identificadores de mensaje “COB-ID”: CANopen define un conjunto de identificadores para direccionar hasta 128 dispositivos en una misma red

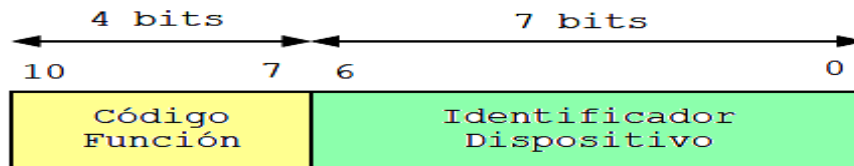
El identificador de mensaje de 11 bits se divide en dos partes:

4 bits para el código de función

⁴⁶ Identificador del dispositivo

7 bits para el identificador de nodo (Dispositivo-ID)

Figura 78: Estructura del identificador de mensajes CANopen:



Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, Pg. 9, Modificado por los Autores”

La prioridad del mensaje la determina su funcionalidad, por este motivo se coloca el código de función antes del identificador de dispositivo. Para el código de función se utiliza 4 bits, de esta forma se distinguen 16 funciones básicas para la recepción y transmisión de los datos de proceso, para la comunicación peer to peer, control y monitoreo de dispositivos, notificación de emergencias y recepción de los mensajes de sincronización y time-stamp.

Tabla 36: Distribución general de los identificadores en CANopen

Broadcast objects of the CANopen Predefined Connection Set			
Object	Function code (ID-bits 10-7)	COB-ID	Communication parameters at OD index
NMT Module Control	0000	000h	–
SYNC	0001	080h	1005h, 1006h, 1007h
TIME STAMP	0010	100h	1012h, 1013h

Peer-to-Peer objects of the CANopen Predefined Connection Set			
Object	Function code (ID-bits 10-7)	COB-ID *	Communication parameters at OD index
EMERGENCY	0001	081h - 0FFh	1024h, 1015h
PDO 1 (transmit)	0011	181h - 1FFh	1800h
PDO 1 (receive)	0100	201h - 27Fh	1400h
PDO 2 (transmit)	0101	281h - 2FFh	1801h
PDO 2 (receive)	0110	301h - 37Fh	1401h
PDO 3 (transmit)	0111	381h - 3FFh	1802h
PDO 3 (receive)	1000	401h - 47Fh	1402h
PDO 4 (transmit)	1001	481h - 4FFh	1803h
PDO 4 (receive)	1010	501h - 57Fh	1403h
SDO (transmit/server)	1011	581h - 5FFh	1200h
SDO (receive/client)	1100	601h - 67Fh	1200h
NMT Error Control	1110	701h - 77Fh	1016h, 1017h

Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, Pg. 9”

5.10 MODELO DE COMUNICACION:

El modelo de comunicación especifica los diferentes objetos de comunicación y servicios de los modos disponible de transmisión de mensaje de activación.

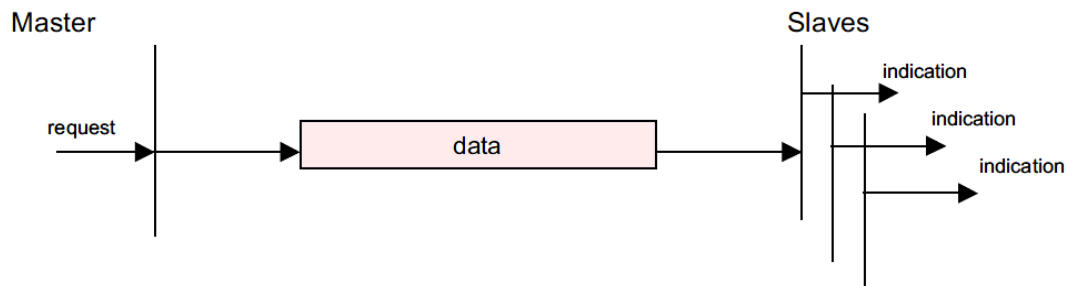
El modelo de comunicación soporta la transmisión de mensaje síncronos y asíncronos. La transmisión de mensajes síncronos se realiza por medio de objetos de comunicación pre-definidos⁴⁷ y son transmitidos con respecto a un mensaje de sincronización predefinido. Los mensajes asíncronos se pueden transmitir en cualquier instante de tiempo. CANopen soporta los modelos de comunicación

⁴⁷ Los mensajes de sincronización, de emergencia y time stamp conforma los mensajes predefinidos y son utilizados para la sincronización de los dispositivos (objetos SYNC) y para generar notificaciones de emergencia en forma opcional.

cliente/servidor, maestro-esclavo y productor-consumidor en sus variantes push y pull.

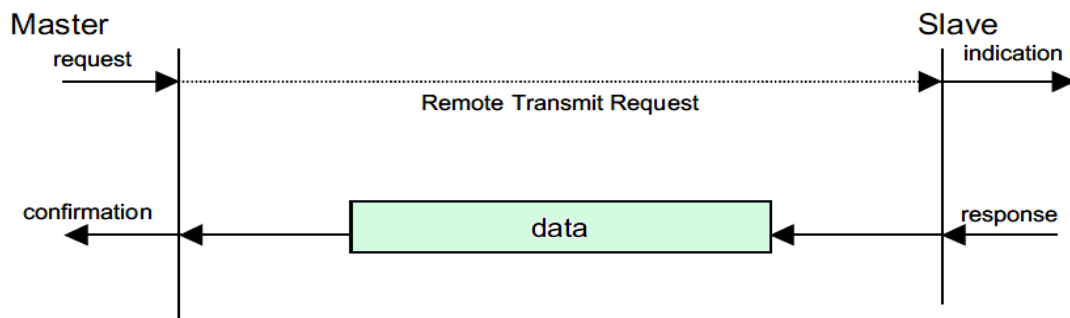
5.10.1 Modelo Maestro/Esclavo: En todo momento hay exactamente un dispositivo en la red que actúa como maestro y los otros dispositivos actúan como esclavos. El maestro hace una solicitud y el esclavo de destino o los esclavos de destino responden si el protocolo requiere este comportamiento. Ver figuras 79 y 80

Figura 79: Modelo Maestro esclavo con comunicación no confirmada



Fuente: "CANopen, Application Layer and Communication Profile, Pg 18"

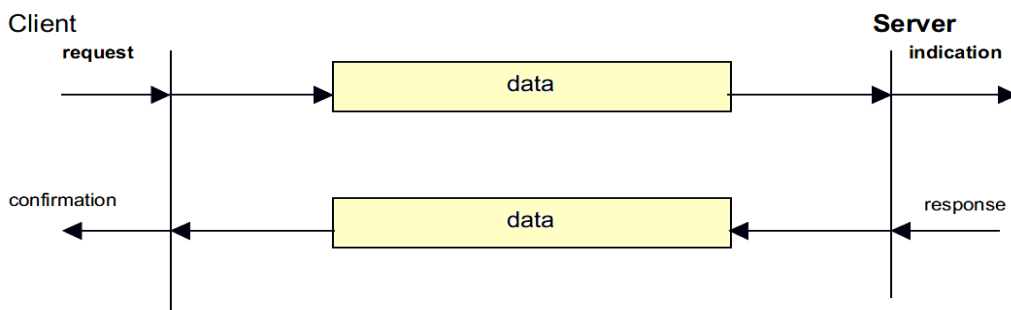
Figura 80: Modelo Maestro esclavo con comunicación confirmada



Fuente: "CANopen, Application Layer and Communication Profile, Pg 18"

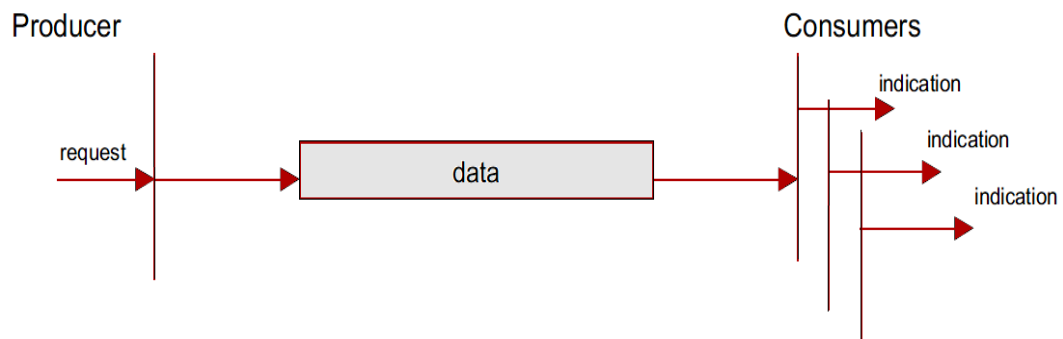
5.10.2 Modelo Cliente/Servidor: Esta es una relación entre un cliente y un servidor. Un cliente emite una solicitud para que el servidor desarrolle una determinada tarea, cuando el servidor termina la tarea manda un mensaje de respuesta. El cliente confirma el mensaje de respuesta del servidor. Ver figuras 81

Figura 81: Modelo cliente/servidor:



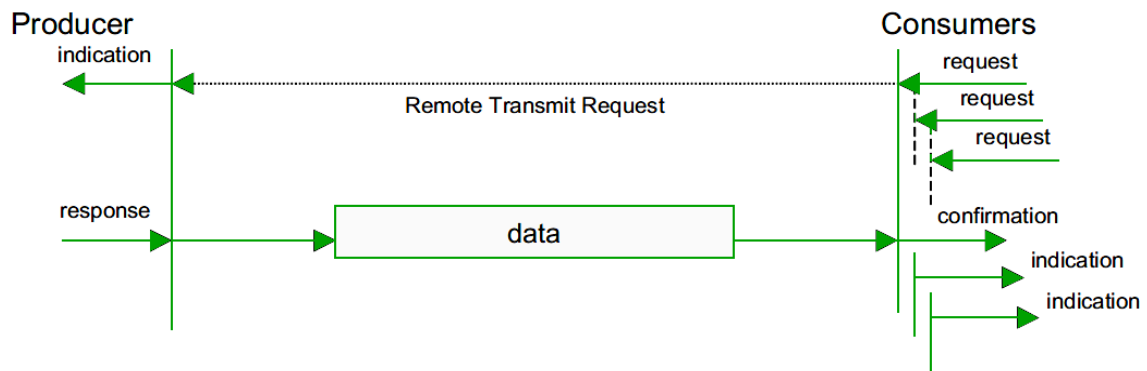
Fuente: “CANopen, Application Layer and Communication Profile, Pg 19”

Figura 82: Modelo push:



Fuente: “CANopen, Application Layer and Communication Profile, Pg 19”

Figura 83: Modelo Pull:



Fuente: “CANopen, Application Layer and Communication Profile, Pg 19”

5.10.3 Modelo productor/consumidor: El modelo productor / consumidor implica una relación de productores y cero o más consumidores. El modelo push se caracteriza por un servicio sin confirmar, solicitada por el productor. El modelo pull se caracteriza por un servicio confirmado solicitada por el consumidor. Ver figuras 82 y 83

5.11 OBJETO DE PROCESO DE DATOS (PDO):

Este tipo de objetos permite intercambiar datos del proceso en tiempo real. Implementa el modelo de comunicaciones productor/consumidor donde un dispositivo (productor) transmite los datos del proceso a uno o más dispositivos (consumidores). Los PDOs proveen únicamente dos servicios, escribir datos (Write-PDO) asociado a un PDO de transmisión de datos “T_PDO” y leer datos Read-PDO asociado a un PDO de recepción de datos “R_PDO”. Los T_PDO son mensajes con información del proceso que el dispositivo transmite, por ejemplo la escritura de un valor en un dispositivo para activar una variable, estos T_PDO son implementados mediante una trama de datos. Los R_PDO son los mensajes con

información del proceso que un dispositivo detecta, por ejemplo un dispositivo que controle la apertura de una bomba detectara la orden en el bus, estos objetos de proceso se implementa con una trama remota de CAN, en ambos casos se utiliza un campo de datos de 8 bytes. La cantidad de PDOs y la longitud de cada uno dependen de cada dispositivo y de la aplicación, por lo cual esta información debe incluirse en el perfil del dispositivo correspondiente. De este modo, cada PDO tendrá una estructura en el diccionario de objetos. La estructura para el primer R_PDO tiene un índice de 1400 h y el primer T_PDO, se ubica en la entrada con un índice 1800 h. Los PDOs se describen por los PDOs communication parameters y los PDOs parameter mapping. Los PDOs communication parameters describe la capacidad de comunicación de los PDOs y los PDOs parameter mapping contiene información de el contenido de los PDOs (variables de los dispositivos). Los índices de las correspondientes entradas de diccionario de objetos se comienzan en el siguiente orden.

RPDO communication parameter index, comienza en 1400h

TPDO communication parameter index, comienza en 1800h

RPDO mapping parameter index, comienza en 1600h

TPDO mapping parameter index, comienza en 1A00h

La tabla 37 presenta la estructura de los PDOs Communication Parameters ubicada en el diccionario de objetos. Un PDO de transmisión de datos “TPDO” o un PDO de recepción de datos “RPDO” es un objeto de 64 bytes que puede contener hasta ocho objetos de aplicación de 8 bytes o cuatro objetos de aplicación de 16 bytes o cualquier combinación de estos dos tipos de objetos de aplicación, sin que se supere el límite global de 64 bytes del PDO, por esta razón se requiere definir la posición de estos objetos de aplicación dentro del campo de datos de un PDO. Esta especificación es designada como “Mapeo PDO”. Cada PDO puede mapear un máximo de 64 bytes, de esta forma se define el contenido de la trama del campo de datos de CAN. Cada objeto de aplicación está definido por un índice y subíndice de su entrada correspondiente en el diccionario de

objetos, por tal motivo el mapeo de estos objetos está dado por una lista que contiene el orden en que debe mapearse los objetos de aplicación de acuerdo al tipo de PDO. En la posición 0021 se define la estructura para el mapeo de objetos, llamada PDO Mapping Parameter. Ver tabla 38

Tabla 37: Estructura de los PDOs communication parameter:

ÍNDICE	SUBÍNDICE	VARIABLE	TIPO DE DATO	DESCRIPCIÓN
0020h	00h	Número de miembros	Unsigned8	Cantidad de miembros en la estructura. Siempre en el subíndice 00h se encuentra este dato para cualquier tipo de dato complejo.
	01h	COB-ID	Unsigned32	Identificador del COB. Es el valor que va en el Campo de arbitraje de la trama de datos de CAN.
	02h	Transmission type	Unsigned8	Indica si los PDOs deben enviarse en forma sincrónica o asincrónica, cíclica o acíclica, etc.
	03h	Inhibit time	Unsigned16	Para permitir que los objetos de comunicación (COB) con baja prioridad sean transmitidos de vez en cuando, se define un tiempo de inhibición para los PDOs. Este tiempo de inhibición es el tiempo mínimo que debe pasar entre dos transmisiones consecutivas del mismo PDO.
	04h	reservado	Unsigned8	No se utiliza en la especificación actual de CANopen.
	05h	Event timer	Unsigned16	Timer para activar la transmisión del PDO correspondiente

Fuente: “CANopen, Application Layer and Communication Profile, Pg 83, modificado por los autores”

Tabla 38 Estructura de los PDOs Mapping Parameter:

ÍNDICE	SUBÍNDICE	VARIABLE	TIPO DE DATO	DESCRIPCIÓN
0021h	00h	Número de mapeos	Unsigned8	Cantidad de mapeos en la estructura
	01h	Objeto # 1	Unsigned 32	Índice y subíndice del objeto mapeado en el diccionario del dispositivo.
	02h	Objeto # 2	Unsigned 32	Índice y subíndice del objeto mapeado en el diccionario del dispositivo.
.....				
	40h	Objeto # 64	Unsigned 32	Índice y subíndice del objeto mapeado en el diccionario del dispositivo.

Fuente: “CANopen, Application Layer and Communication Profile, Pg 83, modificado por los autores”

5.11.1 Transmisión de PDOs: CANopen define varios modos de comunicación para activar la transmisión de PDOs:

5.11.1.1 Transmisión asíncrona: en este tipo de transmisión se puede clasificar los siguientes modos para transmitir los PDOs

Eventos o timer: la transmisión de un mensaje es causada por la ocurrencia de un evento específico definido en el perfil del dispositivo

Solicitud remota: la transmisión de mensajes PDO puede comenzar, cuando se recibe una solicitud remota (trama RTR) enviada por otro dispositivo

Temporizador: se utiliza un temporizador para transmitir PDOs cada determinado tiempo

5.11.2 Transmisión sincrónica: la transmisión sincrónica de mensajes PDO se realiza mediante la recepción de objetos SYNC, cada vez que llega un mensaje SYNC se abre una ventana de transmisión sincrónica para que los PDOs sincrónicos se envíen. Se distinguen dos modos dentro de este tipo de transmisión:

Modo cíclico: son mensajes que se transmiten de forma periódica dentro de las ventanas que sincroniza los objetos SYNC, pero no se transmite en todas las ventanas de tiempo sino en cierto tiempo especificado por el campo Transmission Type of Communication Parameter correspondiente.

Modo acíclico: son mensajes que se transmiten a partir de un evento de la aplicación. Se transmiten dentro de las ventanas que sincroniza los objetos SYNC pero no de forma periódica. En la tabla 39 presenta los distintos tipos de transmisión de PDOs, definidos por el Transmission Type of Communication Parameter.

Tabla 39: tipo de transmisión de PDOs:

Trans-Mission Type	Condition to trigger PDO (B=both needed, O=one or both)			PDO Transmission
	SYNC*	RTR*	Event*	
0	B	-	B	Sync, acyclic
1-240	O	-	-	Sync, cyclic
241-251	-	-	-	<i>reserved</i>
252	B	B	-	Sync, after RTR
253	-	O	-	Async, after RTR
254	-	O	O	Async, manufacturer specific event
255	-	O	O	Async, device profile specific event

Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, Pg. 16”

SYNC = objeto SYNC recibido

RTR = recibida trama RTR

Event = cambio de valor de un dato, temporizador etc.

5.12 OBJETO DE SERVICIO DE DATOS (SDO):

Los objetos de servicio de datos o SDOs requieren implementar los modelos de comunicación cliente / servidor o peer-to-peer, para acceder a los diccionarios de objetos de los dispositivos. Normalmente este tipo de objetos es usado para la configuración de dispositivos y transferencia de grandes cantidades de datos no relevantes (mensajes de baja prioridad) en forma directa, para el control del proceso. Este tipo de objetos ofrece transferencia de datos sin conexión con confirmación, por tal motivo cada SDO involucra el intercambio de dos tramas CAN con valores distintos en el campo de arbitraje.

Un SDO es representado en CMS “Ver 5.9.1 de este documento” como un objeto de tipo Multiplexed Domain (multiplexado de dominio). Para la transferencia de los SDOs se definen una serie de protocolos petición-respuesta, estos protocolos son: Initiate Domain Download, Initiate Domain Upload, Download Domain Segment, Upload Domain Segment y Abort Domain Transfer

Download significa escribir en el diccionario de objetos y Upload leer del diccionario de objetos.

Al ser Multiplexed Domains los SDOs pueden transferir datos de cualquier longitud, sin embargo CANopen define dos tipos de transferencia para los SDO basándose en el tamaño de los datos a transferir, estos tipos de transferencia son: transferencia expédita y transferencia en segmentos.

Los mensajes tanto del cliente como del servidor siempre tienen una longitud de 8 bytes aunque no todos contengan información significativa.

5.12.1 Transferencia expédita: usada para transmitir mensajes con longitud de datos menor o igual a 4 bytes, para lo cual se usan los protocolos Initiate Domain Download o Initiate Domain Upload. En la transferencia expédita no se aplica fragmentación, se envía un único mensaje CAN y se recibe la confirmación del servidor.

5.12.2 Transferencia en segmentos: se usa para transmitir mensajes con longitud de datos mayor o igual a 5 bytes. Se aplica fragmentación en segmentos dividiendo los datos en múltiples mensajes CAN. En este tipo de transferencia el cliente espera la confirmación del servidor por cada segmento.

Para el primer mensaje, se usan los protocolos Initiate Domain Download o Initiate Domain Upload según corresponda. Para los mensajes subsiguientes, se usan los protocolos Download Domain Segment o Upload Domain Segment según se quiera leer o escribir una entrada en el diccionario.

5.12.3 Abort Domain Transfer: este servicio se usa para notificar los errores tanto para clientes como para servidores, que se producen al acceder a las entradas del diccionario de objetos del servidor

5.12.4 Estructura de los SDOs: los SDOs se definen en el diccionario de objetos mediante una estructura que contiene parámetros relacionados con la transmisión de los mismos como se puede observar en la tabla 35. En una red CANopen, se pueden definir hasta 128 SDOs para clientes y 128 SDOs para servidores. Se utiliza la misma estructura de parámetros para dispositivos cliente

como para dispositivos servidores llamada Client SDO Parameter o Server SDO Parameter en uno u otro caso.

Tabla 40: Estructura de parámetros Client / Server SDO:

ÍNDICE	SUBÍNDICE	VARIABLE	TIPO DE DATO	DESCRIPCIÓN
0022h	00h	Número de miembros	Unsigned8	Cantidad de miembros en la estructura. Siempre en el subíndice 00h se encuentra este dato para cualquier tipo de dato complejo.
	01h	COB-ID Cliente → Servidor	Unsigned32	Identificador del COB correspondiente al SDO. Es el valor que va en el campo de arbitraje de la trama CAN enviada por el cliente al servidor.
	02h	COB-ID Servidor → Cliente	Unsigned32	Identificador del COB correspondiente al SDO. Es el valor que va en el campo de arbitraje de la trama CAN enviado por el servidor al cliente
	03h	Dispositivo-ID	Unsigned8	Identificador del dispositivo. Es configurable mediante LMT (LSS).

Fuente: “CANopen, Application Layer and Communication Profile, Pg 83, modificado por los autores”

5.13 MENSAJES ADICIONALES:

Los mensajes adicionales son mensajes u objetos predefinidos de CANopen que permiten la sincronización de los dispositivos y la generación de notificaciones de emergencia. Estos objetos o mensajes son:

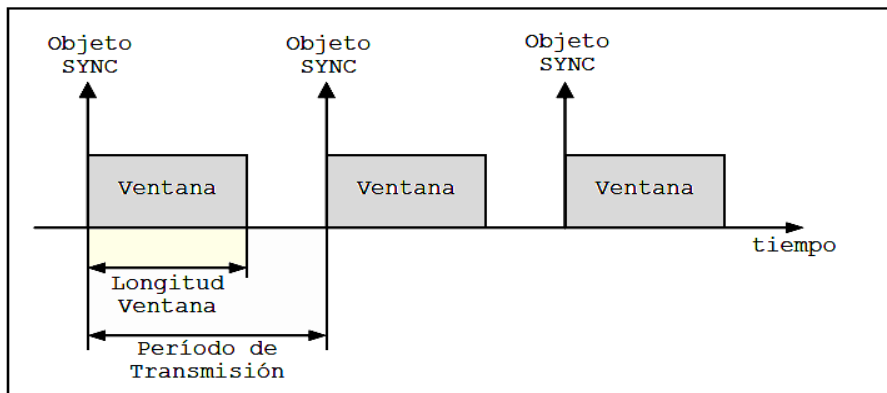
5.13.1 Mensaje de sincronización (SYNC): En una red CANopen, hay un dispositivo que es el productor de objetos SYNC y una serie de dispositivos consumidores de objetos SYNC. Cuando los consumidores reciben el mensaje del productor, abren su ventana de sincronismo y pueden ejecutar sus tareas sincrónicas⁴⁸. El COB-ID usado por este objeto de comunicación puede encontrarse en la entrada 1005h del diccionario de objetos. El campo de datos de la trama CAN de este objeto de comunicación se envía vacío y su comportamiento es determinado por dos parámetros: la longitud de la ventana o synchronous window Length, que se ubica en la entrada 1007h del diccionario de objetos y el parámetro periodo de transmisión o communication cycle period, que se encuentra en la posición 1006h del diccionario de objetos. La figura 84 presenta el mensaje de sincronización (SYNC)

5.13.2 Mensaje de etiqueta (TIME-STAMP): CANopen define un objeto denominado “time stamp” de prioridad relativamente alta, usado por el dispositivo productor para designar una referencia de tiempo común a los otros dispositivos. Este tipo de objetos representa una cantidad absoluta de tiempo en milisegundos y se implementa como una secuencia de 48 bits. Puede ser accedido mapeando el objeto en 1013h a un PDO.

⁴⁸ Son las tareas que permite coherencia temporal y coordinación entre los dispositivos.

5.13.3 Mensajes de Emergencia (EMCY): Los mensajes de emergencia son mensajes de error de alta prioridad y son enviados para señalar estados de error internos en los dispositivos. Los objetos de este tipo se transmiten del dispositivo en el que se presentó la falla a otros dispositivos y sirven como interrupciones o notificaciones de alerta.

Figura 84: mensaje de sincronización (SYNC)



Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, Pg. 17”

5.14 COMUNICACIÓN CON EL PLC TWIDO:

5.14.1 Módulo maestro CANopen TWDNCO1M.: la figura 85 presenta el módulo master CANopen TWDNCO1M y a continuación se presenta sus elementos.

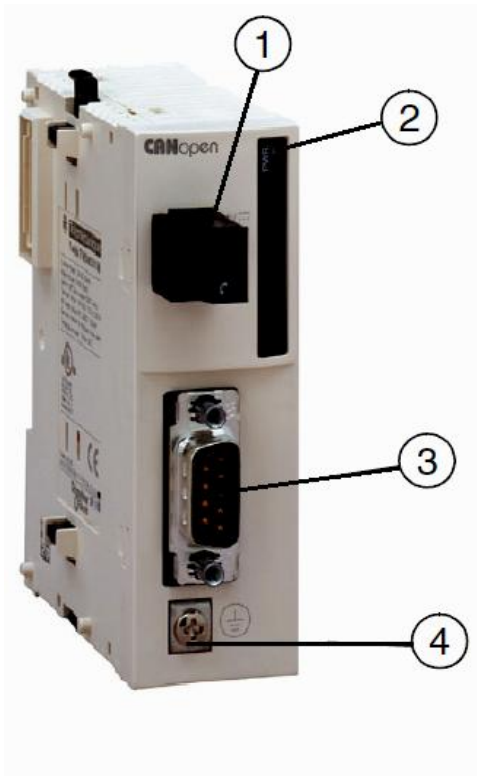
1 Conector para la fuente de alimentación: Conector de 3 puntos utilizado para conectar una fuente de 24 VCC.

2 Indicador luminoso: Muestra el estado del bus CANopen.

3 Terminal DB9: Sirve para conectar el cable de interfase CANopen.

4 Puesta a tierra de protección

Figura 85: Módulo maestro CANopen TWDNCO1M.



Fuente: “Master de modbus controladores programables módulos de comunicación Pg 44”

El maestro CANopen TWDNCO1M admite en la red hasta 16 dispositivos esclavos sin repetidores. La velocidad de transmisión depende estrictamente de la longitud de bus y el tipo de cable utilizado. La tabla 41 describe la relación entre la velocidad de transmisión máxima y la longitud del bus en un segmento CAN sin repetidores.

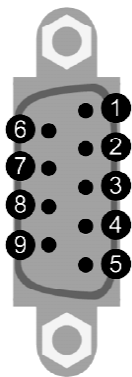
Tabla 41: Velocidad en baudios frente a longitud del cable:

Velocidad máxima de transmisión en baudios	Longitud del bus
500 Kbit/s	100 m
250 Kbit/s	250 m
125 Kbit/s	500 m

Fuente: “Master de modbus controladores programables módulos de comunicación Pg 42”

5.14.2 Conexión Módulo maestro TWDNCO1M con la red CANopen: El módulo maestro se conecta a la red mediante un conector hembra D-SUB de 9 pins. La conexión debe ajustarse a la siguiente asignación de pins, ver tabla 42:

Tabla 42: Asignación de pines del conector D-SUB de 9 pins



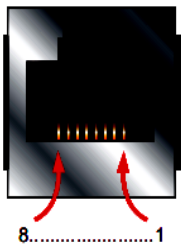
Contactos	Señal	Descripción
1	No utilizado	Reservado
2	CAN_L	Línea del bus de CAN-L (bajo dominante)
3	CAN_GND	Puesta a tierra CAN
4	No utilizado	Reservado
5	CAN_SHLD	Blindaje opcional CAN
6	GND	Puesta a tierra opcional
7	CAN_H	Línea del bus CAN-H (alto dominante)
8	No utilizado	Reservado
9	CAN_V+	NC (no conectado)

Nota 1: La disposición de los contactos corresponde a la leyenda de la ilustración anterior.
Nota 2: Los contactos reservados se utilizarán en una futura especificación.

Fuente: “Master de modbus controladores programables módulos de comunicación Pg 50”

El cable que conecta el bus de campo CANopen y el módulo maestro debe tener un conector hembra que cumpla con el diagrama de pins de contactos según se describe en la tala 42. El cable de red CANopen es un cable trenzado de doble par con blindaje conforme con el estándar de CANopen CiA DR-303-1. Para que el módulo maestro TWDNCO1M se comuniquen con el esclavo “variador de velocidad altivar de propiedad de la E3T” es necesario seguir la siguiente conexión:

Figura 86: Conexión modulo maestro TWDNCOM con esclavo altivar 31

Conector RJ 45	Pin	Señal	Conector D-SUB de 9 pins	pin	Señal
	1	CAN_H		1	No utilizado
	2	CAN_L		2	CAN_L
	3	CAN_GND		3	CAN_GND
	4	D1		4	No utilizado
	5	D0		5	CAN_SHDL
	6	No utilizado		6	GND
	7	VP		7	CAN_H
	8	Común		8	No utilizado
				9	CAN_V+

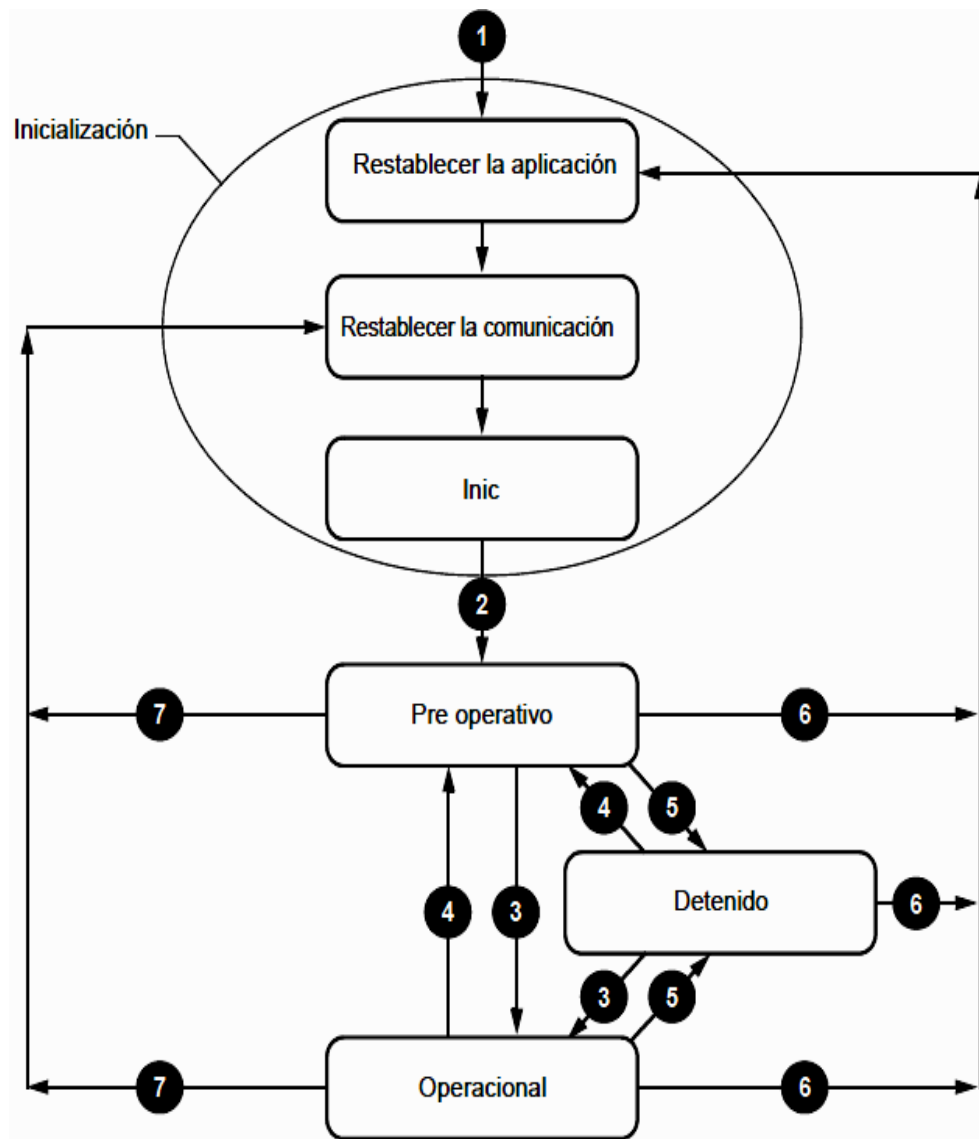
Fuente: “Altivar 31Modbus User’s manual Pg 4, modificado por los autores”

La conexión se realiza uniendo el pin 2 del conector DB-9 con el pin 2 del conector RJ 45, el pin 3 del conector DB-9 con el pin 8 del conector RJ 45 y el pin 7 del conector DB-9 con el pin 1 del conector RJ 45. Nota: los demás pins no son utilizados en CANopen

5.14.3 Administración y monitoreo de la red CANopen: CANopen incluye una serie de mensajes para la administración y monitoreo de los dispositivos en la red llamados Network Management (NMT). Esta serie de mensajes emplea un modelo

de comunicaciones maestro / esclavo en el cual, un dispositivo cumple con el rol de NMT maestro y el resto son NMT esclavos. Un dispositivo esclavo NMT puede encontrarse en alguno de los siguientes estados:

Figura 87: Máquina de estados para dispositivo NMT esclavo:



Fuente: “twidosuite V2.0, guía de programación 09/2007. Pg 267”

Tabla 43: Estado de un dispositivo esclavo NTM

Número	Descripción
1	Arranque del módulo
2	Después de la inicialización, el módulo pasa automáticamente al estado PRE-OPERATIONAL.
3	Indicación de servicio NMT: START REMOTE NODE
4	Indicación de servicio NMT: PRE-OPERATIONAL
5	Indicación de servicio NMT: STOP REMOTE NODE
6	Indicación de servicio NMT: RESET NODE
7	Indicación de servicio NMT: RESET COMMUNICATION

Fuente: "twidosuite V2.0, guía de programación 09/2007. Pg 267"

5.14.3.1 Restablecer la aplicación: después del arranque, el dispositivo pasa a este estado para que el perfil del dispositivo se inicie y toda la información del dispositivo se restablezca con valores predeterminados. Se utiliza el servicio de gestión de la red (NMT) "RESET NODE"

5.14.3.2 Restablecer la comunicación: después del estado "Restablecer la Aplicación" el dispositivo pasa automáticamente a este estado en donde todos los parámetros de los objetos de comunicación compatibles (objetos desde 1000h a 1FFFh) se guardan en el directorio de objetos. Se Utiliza el servicio de gestión de la red (NMT) "RESET COMMUNICATION".

5.14.3.3 Inic: después de permanecer en el estado "Restablecer" el dispositivo pasa a estado init. Este estado le permite definir los objetos de comunicación necesarios (SDOs, PDOs y Emergency), instalar los servicios CAL correspondientes y Configurar el CAN-Controller.

5.14.3.4 Pre-Operativo: después de permanecer en estado "Inic" el dispositivo pasa a este estado, donde se puede modificar la configuración. Sin embargo, sólo se pueden utilizar los objetos SDO para leer o escribir datos relacionados con el dispositivo. En este estado se utiliza el servicio de gestión de la red (NMT) "PRE-OPERATIONAL".

5.14.3.5 Detenido: El dispositivo pasa al estado "Detenido" al recibir la indicación del servicio NMT "STOP REMOTE NODE", si se encontraba en el estado "Pre-Operativo" o "Operacional". En este estado sólo permanece activa la función de supervisión del esclavo ("Node guarding").

5.14.3.6 Operacional: El dispositivo pasa al estado "Operational" si se encontraba en el estado "Pre-Operativo" al recibir la indicación "START REMOTE NODE". En este estado se puede utilizar todas las funciones del dispositivo, la comunicación puede utilizar objetos PDO o SDO.

5.15 ELEMENTOS UTILIZADOS POR EL MÓDULO MAESTRO TWDNCO1M:

Los elementos básicos para que exista comunicación entre el Módulo maestro CANopen TWDNCO1M y los esclavos en una red CANopen, se describen a continuación:

5.15.1 Archivo EDS (Hoja de datos electrónica): los archivos EDS describen las propiedades de comunicación de un dispositivo en la red CAN como: velocidad en baudios, tipos de transmisión, funciones de entradas y salidas, etc. El archivo

EDS se utiliza para configurar los dispositivos y es proporcionado por los fabricantes de dichos dispositivos.

5.15.2 Perfiles CANopen: La familia de perfiles CANopen se basa en un “perfil de comunicación”. Este perfil especifica los mecanismos de comunicación y su descripción a través del estándar (DS301). En la automatización industrial los tipos más importantes de dispositivos se describen en los “perfiles de dispositivos”. A continuación se presentan los principales estándares de los perfiles de los dispositivos:

Módulos de entrada/salida analógicos y binarios (DS401).

Motores (DS402).

Dispositivos de control (DSP403).

Autómatas de bucle cerrado (DSP404).

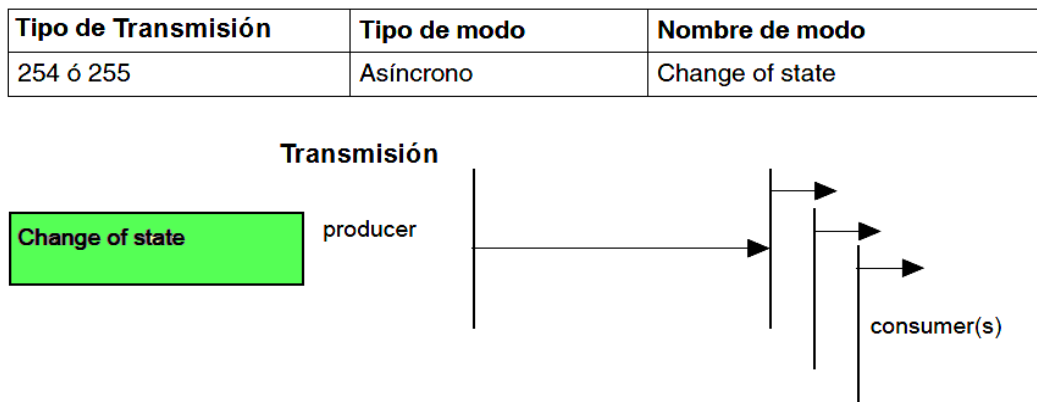
Autómatas (DS405).

Codificadores (DS406).

5.15.3 Envío de PDOs a través del módulo maestro TWDNCO1M: existen dos tipos de PDOs: los PDOs que contiene los datos del dispositivo que envía los PDOs, denominados TPDOs y los PDOs que contiene los datos de otro dispositivo que el dispositivo recibirá, denominados RTPOs. Los PDOS están basados en un modelo "Productor Consumidor", el dispositivo que envía un PDO se denomina "productor" y el dispositivo que lo recibe se denomina "consumidor". Ejemplo para que el módulo maestro TWDNCO1M escriba el valor de una de sus salidas en alguna entrada de un esclavo para que este active una de sus salidas, se envía un TPDO asociado con PDO productor que contiene el valor de la salida que se va a escribir en la entrada del dispositivo esclavo; en este caso el maestro TWDNCO1M es el productor y el esclavo es el consumidor. Si ahora se quiere que el módulo maestro lea alguna salida de un esclavo para guardar el valor en alguna

de sus entradas y de esta manera activar una de sus salidas, se recibe un RPDO que contiene el valor de la salida del esclavo, este valor se va a guardar en alguna entrada del maestro; en este caso el esclavo es el productor de PDOs y el maestro es el consumidor de PDOs. El módulo maestro TWDNCO1M puede intercambiar los PDOs de la siguiente forma. Ver figura 88

Figura 88: Intercambio de PDOs con el modulo maestro TWDNCOM



Fuente: "twidosuite V2.0, guía de programación 09/2007. Pg 271"

"Change of state" corresponde a la modificación de un valor de entrada (control de eventos). El control de eventos permite hacer un uso óptimo del ancho de banda del bus, ya que se envía sólo la modificación en vez de toda la imagen del proceso. De esta manera se obtiene un tiempo de respuesta muy corto.

Cuando se selecciona el envío de PDO change of state", es probable que ocurran varios eventos al mismo tiempo, lo que generará retrasos mientras espera a que se envíen los PDO de menor prioridad al bus. Como norma general, sólo se debe seleccionar el envío de PDOs con módulos de entrada analógicos si el tiempo de inhibición se ha configurado para evitar una sobrecarga de bus.

5.15.4 Envío de SDOs a través del módulo maestro TWDNCO1M: el maestro utiliza los SDOs cuando el dispositivo esclavo se encuentra en el estado operacional o pre-operacional. Existen dos clases de SDOs en el funcionamiento de la red CANopen, los SDOs de lectura o también llamados descarga y los SDOs de escritura también llamados de carga, estos SDOs se envían utilizando modelo cliente/servidor. Por ejemplo para un SDO de descarga, el cliente envía una solicitud indicando el objeto que se va a leer y el servidor devuelve los datos contenidos en el objeto. Si el SDO es de carga el cliente envía una solicitud indicando el objeto que se va a escribir y el valor deseado, para que el servidor le responda con un mensaje de confirmación.

5.15.5 Monitoreo del dispositivo: Se utiliza para saber si un dispositivo está operando correctamente. La comunicación utiliza el modelo maestro-esclavo, en donde el NMT maestro monitorea el estado de los dispositivos. A esto se le llama node guarding. Opcionalmente los dispositivos pueden monitorear el estado del NMT maestro. A esto se le llama life guarding. Estos mensajes sirven para detectar errores en las interfaces de red de los dispositivos y se puede implementar de dos maneras distintas, pudiendo utilizarse sólo una de ellas a la vez: NMT Node Guarding o Heartbeat.

5.15.5.1 NMT Node Guarding: el dispositivo maestro NMT consulta a cada dispositivo esclavo NMT, cada cierto tiempo el estado de comunicación. El dispositivo esclavo transmite su estado de comunicación con un mensaje de respuesta. En caso de que el maestro no reciba respuesta dentro de un intervalo

de tiempo definido⁴⁹ o la respuesta no sea la esperada por el maestro, este indicará un evento denominado node guarding en su aplicación. De igual manera el esclavo indicará un evento “life guarding” a su aplicación en caso de que su estado no haya sido consultado por el maestro dentro de un intervalo de tiempo definido. El maestro interroga a los esclavos mediante una trama remota (RTR) con la siguiente estructura:

Figura 89: Trama RTR que el NMT maestro envía a los NMT esclavos:

COB-ID
700h + Node_ID

Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, Pg. 19

Los NMT esclavos responden con el siguiente mensaje:

Figura 90: Mensaje que los NMT esclavos envían al NMT maestro

COB-ID	Byte 0
700h + Node_ID	Bit 7: toggle, bit 6-0: state

Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, Pg. 19”

El toggle bit (bit 7) es un bit que va alternando su valor en cada mensaje de Node

⁴⁹ Este tiempo se conoce como life time y se calcula a sí: $\text{Life Time} = \text{Guard Time} \times \text{Life Time Factor}$. Los valores de guard time oscilan entre 250 milisegundos y 2 segundos. El valor de life time factor para garantizar un funcionamiento fiable debe ser de 2.

El objeto 100Ch contiene el parámetro "Guard Time" expresado en milisegundos.

El objeto 100Dh contiene el parámetro "Life Time Factor".

Guarding. El valor de "Toggle Bit" enviado en el primer mensaje Guarding es "0", a continuación, el bit cambia en cada siguiente mensaje Guarding, lo que permite indicar si se ha perdido un mensaje. Los bits del 0 al 6 indican el estado del dispositivo. Su valor se presenta en la siguiente tabla:

Tabla 44: Valor del campo state en un mensaje de NMT Node Guarding:

VALUE	STATE
0h	Initialising
1h	Disconnected
2h	Connecting
3h	Preparing
4h o 84h	stopped
5h o 85h	Operational
7Fh o FFh	Pre-operational

Fuente: "Departamento de Informática y Automática, Facultad de Ciencias, Pg. 20"

5.15.5.2 Heartbeat: Cada dispositivo manda un mensaje de Heartbeat cada cierto tiempo para informar que está en estado operativo. En este caso el mensaje de Boot-up⁵⁰ se considera que es el primer mensaje de Heartbeat. Un dispositivo se reporta como caído cuando el NMT maestro deja de recibir estos mensajes en un tiempo determinado.

Figura 91: Estructura de un mensaje de Heartbeat

COB-ID	Byte 0
700h + Node_ID	State

Fuente: "Departamento de Informática y Automática, Facultad de Ciencias, Pg. 20"

⁵⁰ Mensaje Boot-up: por medio de este mensaje un dispositivo indica al maestro NMT que ha alcanzado el estado pre-operacional después de que la inicialización se ha completado.

Tabla 45: Valor del campo state en un mensaje de Heartbeat

STATE	MEANING
0h	Boot-up
4h	Stopped
5h	Operational
7Fh	Pre-operational

Fuente: “Departamento de Informática y Automática, Facultad de Ciencias, Pg. 20”

5.16 CONFIGURACION DE CANopen CON EL SOFTWARE TWIDOSUITE.

5.16.1 Descripción general: La configuración CANopen se realiza mediante las herramientas⁵¹ de configuración a las que se accede desde la ventana Describir en TwidoSuite teniendo presente los siguientes ítems.

La configuración de esclavos, maestro y red CANopen, así como la asignación de los parámetros de configuración se realizan únicamente en modo offline.

No se permite ningún cambio a la configuración de CANopen en el modo online.

En el modo online y en el estado operativo sólo se pueden ajustar determinados parámetros como los parámetros PDOs de direccionamiento %IWC y %QWC

5.16.1.1 Configuración: a continuación se describen los distintos pasos que se deben realizar empleando el software Twidosuite para configurar el bus CANopen:

⁵¹ La configuración de CANopen con el software Twidosuite se tomo de Twidosuite V2.0 guía de programación, paginas: 280, 282, 283, 284, 285 y 288

Tabla 46: Pasos con Twidosuite para configurar CANopen:

Modo	Fase	Descripción
Local	Declaración de master CANopen	En la ventana Describir, añada un módulo master CANopen a la aplicación Twido TWDNCO1M
	Configuración de la red CANopen	Configure la red CANopen mediante las operaciones siguientes: ● Importe archivos EDS a todos los dispositivos slave del catálogo de red. ● Agregue los dispositivos slave del catálogo a la red CANopen.
	Asignación de PDO	Realice la asignación de los objetos TPDO y RPDO de cada dispositivo slave declarado en la red.
	Conexión de PDO	Conecte cada PDO de slave al PDO correspondiente del módulo master.
Local o conectado	Simbolización (opcional)	Simbolización de las variables asociadas a los equipos slave.
	Programación	Programación de la función CANopen.
Conectado	Transferir	Transferencia de la aplicación al PLC.
	Depuración	Depuración de la aplicación mediante las pantallas de animación y supervisión

Fuente: “twidosuite V2.0, pg. 280, Modificado por los autores”

El programa de interfase Twidosuite solo permite un modulo maestro TWDNCO1M CANopen en el bus de ampliación Twido, este se declara y se elimina como cualquier otro módulo de ampliación. La dirección 127 está reservada para el módulo maestro TWDNCO1M por lo tanto no debe haber ningún esclavo presente físicamente con esta dirección.

5.16.1.2 Declaración del maestro CANopen: en esta parte del trabajo se explica cómo declarar un módulo maestro CANopen en el bus interno del autómatas. La siguiente tabla muestra los pasos necesarios para la declaración del maestro CANopen.

Tabla 47: Pasos para la declaración del maestro CANopen en Twidosuite.

Paso	Acción	Comentario
1	Seleccionar el paso Describir de la interfase de TwidoSuite.	Consulte (Ventana de aplicación Describir, Twido, Software de programación TwidoSuite, Ayuda online).
2	Ver el catálogo de productos y elegir un módulo TWDNCO1M para agregarlo a la descripción de sistema.	Consulte (Colocación de un módulo, Twido, Software de programación TwidoSuite, Ayuda online). Nota: Un módulo master TWDNCO1M se puede insertar en cualquier posición de ampliación disponible. Se puede mover a continuación a la derecha o a la izquierda de otros módulos de ampliación. Es posible seguir añadiendo otros módulos de ampliación (hasta 7 entre los que se incluye el módulo TWDNCO1M). Nota: Sólo se admite un módulo master CANopen TWDNCO1M. Sólo se admiten los autómatas TWDC•A24DRF, TWDC••40DRF, TWDLMDA20*** y TWDLMDA40***.

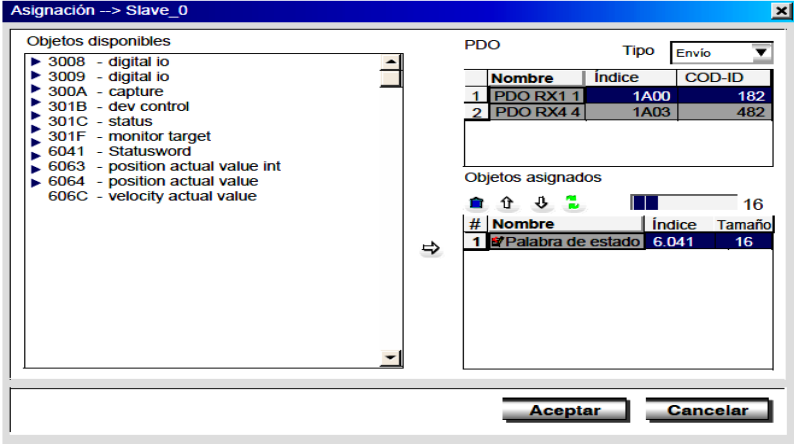
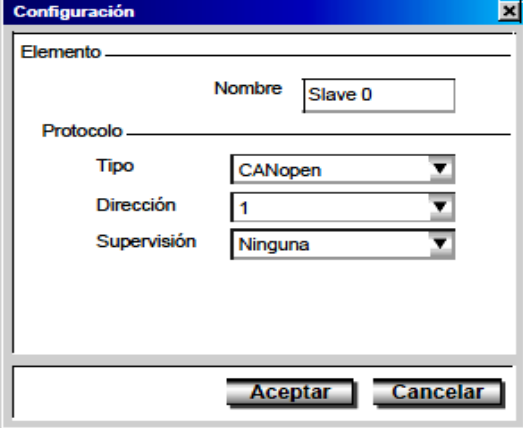
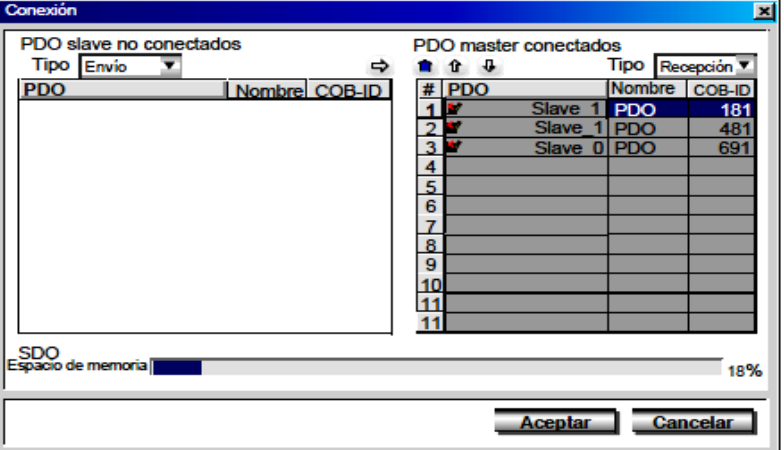
Fuente: “twidosuite V2.0, pg. 282”

5.16.2 Herramientas de configuración Twidosuite para CANopen: para acceder a la herramienta de configuración CANopen es necesario ir a la ventana describir y hacer doble clic en el elemento adecuado, tal y como se indica en la tabla 48:

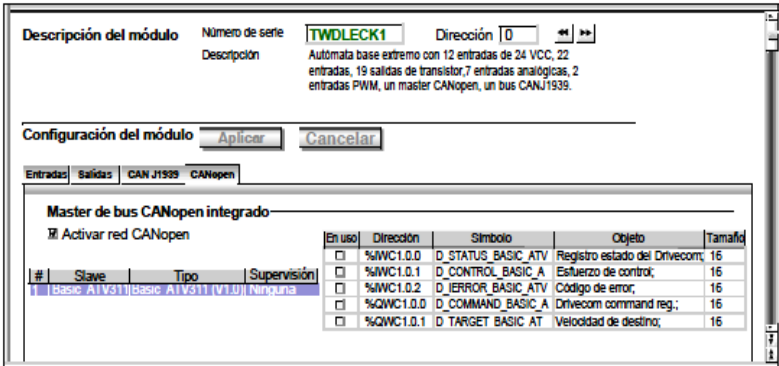
Tabla 48: Herramientas de configuración CANopen:

Elemento CANopen	Acción	Resultado																																						
Master CANopen TWDNCOM1 (sólo lectura)	Hacer doble clic (o hacer clic con el botón derecho del ratón y seleccionar Configuración) en el master.	Descripción del módulo Número de serie Descripción TWDNCO1M Módulo master del bus CANopen (50 mA) Dirección 1 ◀ ▶ Configuración del módulo <table><thead><tr><th>#</th><th>Slave</th><th>Tipo</th><th>Supervisión</th></tr></thead><tbody><tr><td>1</td><td>Slave_0</td><td>Basic ATV31 (V1.0)</td><td>Ninguno</td></tr></tbody></table><table><thead><tr><th>En uso</th><th>Dirección</th><th>Símbolo</th><th>Objeto</th><th>Tamaño</th></tr></thead><tbody><tr><td>☐</td><td>%IWC1.0.0</td><td>D_STATUS_BASIC_ATV</td><td>Registro estado del Drivcom;</td><td>16</td></tr><tr><td>☐</td><td>%IWC1.0.1</td><td>D_CONTROL_BASIC_A</td><td>Esfuerzo de control;</td><td>16</td></tr><tr><td>☐</td><td>%IWC1.0.2</td><td>D_IERROR_BASIC_ATV</td><td>Código de error;</td><td>16</td></tr><tr><td>☐</td><td>%QWC1.0.0</td><td>D_COMMAND_BASIC_A</td><td>Drivcom command reg.;</td><td>16</td></tr><tr><td>☐</td><td>%QWC1.0.1</td><td>D_TARGET_BASIC_AT</td><td>Velocidad de destino;</td><td>16</td></tr></tbody></table>	#	Slave	Tipo	Supervisión	1	Slave_0	Basic ATV31 (V1.0)	Ninguno	En uso	Dirección	Símbolo	Objeto	Tamaño	☐	%IWC1.0.0	D_STATUS_BASIC_ATV	Registro estado del Drivcom;	16	☐	%IWC1.0.1	D_CONTROL_BASIC_A	Esfuerzo de control;	16	☐	%IWC1.0.2	D_IERROR_BASIC_ATV	Código de error;	16	☐	%QWC1.0.0	D_COMMAND_BASIC_A	Drivcom command reg.;	16	☐	%QWC1.0.1	D_TARGET_BASIC_AT	Velocidad de destino;	16
#	Slave	Tipo	Supervisión																																					
1	Slave_0	Basic ATV31 (V1.0)	Ninguno																																					
En uso	Dirección	Símbolo	Objeto	Tamaño																																				
☐	%IWC1.0.0	D_STATUS_BASIC_ATV	Registro estado del Drivcom;	16																																				
☐	%IWC1.0.1	D_CONTROL_BASIC_A	Esfuerzo de control;	16																																				
☐	%IWC1.0.2	D_IERROR_BASIC_ATV	Código de error;	16																																				
☐	%QWC1.0.0	D_COMMAND_BASIC_A	Drivcom command reg.;	16																																				
☐	%QWC1.0.1	D_TARGET_BASIC_AT	Velocidad de destino;	16																																				

Continuación tabla 48

Elemento CANopen	Acción	Resultado
Slave CANopen conectado al master CANopen	Hacer doble clic (o hacer clic con el botón derecho del ratón y seleccionar Configuración CANopen) en el slave.	
Slave CANopen (conectado o no conectado)	Hacer clic con el botón derecho del ratón en el slave y seleccionar Configuración .	
Puerto CANopen del master	Hacer doble clic (o hacer clic con el botón derecho del ratón y seleccionar Configuración CANopen) en el puerto CANopen morado.	

Continuación tabla 48

Elemento CANOpen	Acción	Resultado
Extreme TWDLEDCK1 (sólo lectura)	Hacer doble clic (o hacer clic con el botón derecho del ratón y seleccionar Configuración) en la base y, a continuación, seleccionar la pestaña CANOpen en el panel que aparece.	
Red CANOpen	Hacer doble clic o hacer clic con el botón derecho del ratón y seleccionar Configuración .	

Fuente: “twidosuite V2.0, paginas 283,284 y. 285”

5.17 DECLARACIÓN DE LOS ESCLAVOS EN LA RED CANOpen

La declaración de los esclavos en la red CANOpen es un proceso de tres etapas que consiste en lo siguiente:



importación de los archivos EDS de los dispositivos esclavos de CANopen en el catálogo Describir Twido.



creación de la red CANopen, agregando como máximo 16 dispositivos esclavos del catálogo a la red.



configuración de los parámetros de administración de la red (velocidad de red y parámetros de protocolo de control de errores).

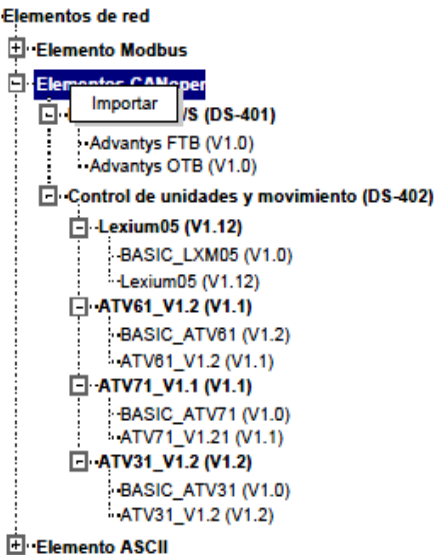
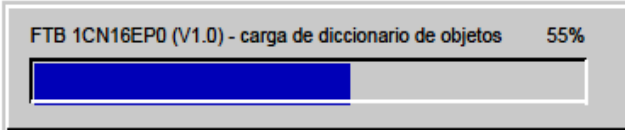
5.17.1 Importación de perfiles de esclavos: En la tabla siguiente se describe cómo importar perfiles de esclavos de CANopen nuevos (archivos .EDS) en el catálogo de la herramienta de configuración de CANopen. Ver tabla 49

5.17.2 Creación de la red CANopen: en una red CANopen solo se puede declarar esclavos cuyo perfiles EDS ya hayan sido agregados o ya estén almacenados en el catálogo del software. Los pasos para crear una red CANopen se presentan en el Anexo 4 de este documento.

5.17.3 Configuración de los parámetros de administración de red: este procedimiento se presenta en el Anexo 4 de este documento

Nota: se recomienda al estudiante leer las páginas 296 a la 312 del documento twidosuite V2.0 guía de programación, para que termine de comprender toda la parte teórica del protocolo de comunicación CANopen.

Tabla 49: Importación de archivos EDS

Paso	Acción
1	Hacer clic con el botón derecho del ratón en Elementos CANopen en el Catálogo de la ventana Describir y seleccionar Importar como se muestra a continuación.  Resultado: aparece el cuadro de diálogo Abrir del sistema operativo.
2	Ir a la carpeta que contiene los archivos EDS de los dispositivos slave de CANopen que desea agregar al catálogo. Resultado: el nombre de los archivos EDS disponibles aparece en el cuadro de diálogo Abrir:
3	Elegir un archivo EDS ("nombredearchivo".EDS) de la lista y hacer clic en Abrir. Resultado: la herramienta de configuración de CANopen carga el diccionario de objetos del dispositivo seleccionado. Nota: Este proceso puede durar unos minutos en función del tamaño del archivo EDS seleccionado. Una barra de progreso indica el estado del proceso de carga, tal y como se muestra en el ejemplo siguiente: 
4	Esperar hasta que termine el proceso de carga. A continuación, repetir los pasos 2 a 3 para cualquier perfil de slave que se desee agregar al catálogo. Nota: Sólo es necesario realizar este proceso una vez, ya que TwidoSuite guarda todos los perfiles de dispositivos y diccionarios de objetos que se muestran en el catálogo cargado.

Fuente: “twidosuite V2.0, pg. 288”

6. CONCLUSIONES

* Aprender la teoría de los protocolos de comunicación industrial es complicado porque se encuentra demasiada información con frecuencia contradictoria, debido a que las traducciones de los estándares de cada protocolo son adaptadas a la terminología propia del país o región del traductor, lo que complica la comprensión de los mismos, ya que cuando no se traduce mal, se dan varios nombres a un elemento; por tal motivo en este proyecto de grado se trato al máximo de obtener los estándares de los protocolos de comunicación industrial de fuentes oficiales y en idioma ingles.

*Si se quiere aprender protocolos de comunicación industrial, so pena de gastar en vano tiempo y esfuerzo, se debe empezar de protocolos de menos complejidad e ir avanzando hacia los de mayor complejidad; por ejemplo en el caso de los protocolos de comunicación industrial estudiados en este trabajo de grado, se recomienda empezar por AS-i, después Modbus sobre línea serial, continuar con Modbus TCP/IP y terminar con CANopen. Es mejor comezar por AS-i por que la programacion de la comunicación es transparente al usuario (es decir, el usuario no tiene que programarla, a diferencia de los otros protocolos).

*Si bien es cierto que la implementación de protocolo Modbus sobre línea serial (RS485, bus de dos hilos) tiene la ventaja de ser bastante extendida y por consiguiente la de ser una frecuente candidata para comunicar dispositivos industriales de diferentes fabricantes, es igualmente cierto que su debilidad radica en la importancia excesiva que tiene el maestro en la red; si el maestro falla toda la red también fallará; por consiguiente si se usa Modbus sobre línea serial en un proceso critico es importante montar una segunda red que permanezca operativa en caso de fallar la principal, para mejorar la confiabilidad. Otro inconveniente de Modbus sobre línea serial es que a medida que la red crece (ingresan más

esclavos) el programa del maestro se va haciendo más largo y muy probablemente más complejo, debido a que el maestro administra todo el tráfico de la red. Esto dificulta su diseño y el posterior hallazgo de cualquier error en el mismo.

*Modbus TCP/IP permite contar con una red muy confiable, al tener todos los equipos de una red la posibilidad de ser servidores, clientes o clientes y servidores simultáneamente, cada equipo estará programado para administrar (enviar o recibir) los datos que le interesan, de modo tal que la falla en un equipo de la red solo afectará los servicios de los que ese equipo es cliente o servidor, o dicho de otra manera, solo afectará a los equipos a los que les enviaba o les recibía datos, es decir, perjudicará a una parte de la red, pero no a la red entera. Lo anterior también contribuye a que los programas sean más pequeños y por lo tanto más fáciles de diseñar y de encontrarse la falla en ellos si se presenta.

*Los fabricantes del sector (previa solicitud) ofrecen partes de programas y en muy pocas ocasiones programas completos, pero hechos para redes en las que intervienen equipos a los que no se tiene acceso (en este caso porque no los tiene la E3T-UIS), luego entender un programa del que se desconoce una parte o parte del cual está programado en función de equipos que no conocemos y de los que por consiguiente no entendemos su funcionamiento y/o programación, es muy complicada. Por esta razón, todos los programas descritos en los anexos de este trabajo de grado se incluyen completos y fueron hechos para utilizar los equipos que posee la E3T-UIS, de tal forma que si no se entienden se pueden implementar para comprenderlos .

*Los macros Drive y los macros Comm son herramientas que Telemecanique incluye en los PLC Twido para facilitar la programación de la comunicación via Modbus, Modbus TCP/IP y CANopen (recomendable para personas sin experiencia en la programación o uso del software del PLC Twido o para cuando

se crean varios programas de PLCs que se comunican entre si o programas muy largos); no obstante para los intereses de este trabajo de grado impedían apreciar la aplicación del protocolo de comunicación industrial, lo cual motivo a los autores a programar en ocasiones solo con macros, en otras sin macros, en otras algunos intercambios de mensajes con macros y otras sin macros dentro del mismo programa y otras veces se programaba una tarea con macros y posteriormente esa misma tarea sin macros para que se pueda apreciar bien la diferencia.

*Vijeo Designer es un paquete de software formado por un editor del mismo nombre (Vijeo Designer) para PC y un firmware para la HMI llamado Vijeo Designer Runtime; la instalación de este último en la HMI es controlada desde el PC con el software “Instalador de Runtime”. Si este firmware se desconfigura y por consiguiente la HMI enciende, pero no funciona (ni ejecuta el programa que tiene en memoria ni se deja transferir uno nuevo), se debe ejecutar un *procedimiento denominado “Recuperación” para habilitarla nuevamente. La recuperación también se lleva a cabo con el instalador de runtime, donde se explican claramente los 7 pasos de la recuperación; el primero de estos pasos exige un cable (con conectores DB9 hembra en ambos extremos) que vende el fabricante de la HMI (Telemecanique) y que fue hecho por los autores de este trabajo de grado y donado a la E3T-UIS.

*Una caja de derivación Telemecanique (donde se implemente el bus de campo, ya sea para CANopen o para modbus) y tres cables para conectar dispositivos a ella cuestan \$750.000 (precios de 2010). Los autores de este trabajo de grado lograron hacer eso mismo con \$15.000, usando tres uniones tipo T (tres conectores RJ45 hembra), cinco cables y dos terminaciones de línea (resistencia para el bus de campo CANopen y resistencia en serie con un condensador para modbus sobre línea serial de dos hilos) implementadas en dos conectores RJ45 macho. Para implementar el bus de campo CANopen los autores de este trabajo de grado donaron a la E3T-UIS 8 uniones tipo T, 4 cables maestro-uni3n (conector

DB9 hembra en un extremo y conector RJ45 macho en el otro extremo) y 8 resistencias de terminación de línea implementadas en 8 conectores RJ45 macho. Para implementar el protocolo modbus sobre línea serial de dos hilos, los autores de este trabajo de grado obsequiaron a la UIS-E3T 12 uniones tipo T, 4 cables Twido-union (conector miniDIN macho de 8 pines en un extremo y conector RJ45 macho en el otro extremo) y 8 terminaciones de línea implementadas en 8 conectores RJ45 macho. Los autores de este proyecto de grado también aportaron a la E3T-UIS 16 cables directos (no cruzados) UTP con conectores RJ45 macho en los extremos que sirven para conectar las uniones en ambos protocolos, para conectar los variadores a las uniones tipo T en ambos protocolos y para conectar los PLC M340 y los relees programables Zelio a las uniones modbus. Estos aportes al inventario de la E3T-UIS son importantes porque independientemente de que el docente del curso redes de comunicación industrial decida utilizar o no utilizar las prácticas de laboratorio generadas por este trabajo de grado, permitirán que cuatro grupos de estudiantes trabajen simultáneamente y por aparte en una práctica de CANopen que use un PLC Twido, un módulo maestro CANopen y un variador de velocidad ATV31 o que lo hagan también de manera simultánea y por aparte en una práctica de modbus con un maestro (PLC Twido o PLC M340 o HMI Magelis) y hasta dos esclavos (PLC Twido, M340, Zelio, ATV31 o HMI Magelis).

*Las prácticas de laboratorio se diseñaron para contribuir a optimizar el aprendizaje en el laboratorio mediante la utilización en cada práctica de la menor cantidad de equipo posible, de modo que esto permite maximizar el número de grupos de estudiantes en el laboratorio y por consiguiente minimizar el número de estudiantes en cada grupo de estudiantes. Dicho de otra forma, la E3T-UIS tiene 4 PLC Twido, 4 módulos maestro CANopen, 4 módulos maestro AS-i, 4 variadores de velocidad ATV31, 4 interfaces IP20 compactas, 4 cajas de pulsadores y 4 fotodetectores; este trabajo de grado diseñó prácticas de laboratorio que permiten

dividir un curso de estudiantes de laboratorio en 4 grupos, asegurando que cada grupo tenga acceso al equipo necesario para desarrollar la misma practica.

*Parte de la complejidad del protocolo CANopen radica en que no utiliza siempre el mismo modelo de comunicación (o método de acceso al medio), como si lo hacen AS-i (maestro-esclavo), Modbus sobre línea serial (maestro-esclavo) y Modbus TCP/IP (cliente-servidor). CANopen utiliza un modelo de comunicación diferente en cada uno de sus servicios: utiliza maestro-esclavo para los servicios NMT; cliente-servidor para sus servicios SDO y productor-consumidor para sus servicios PDO.

7. RECOMENDACIONES

La automatización de procesos industriales, y más específicamente la implementación de redes de comunicación industrial es un campo tan apasionante como supremamente útil y de gran aplicación en la industria Colombiana; por esta razón la E3T-UIS debería brindar a sus egresados la posibilidad de trabajar con los PLC, switches, fuentes, variadores de velocidad, motores y en general con los equipos de su propiedad que sean necesarios para implementar redes de comunicación industrial y simular procesos industriales.

*Telemecanique actualiza la documentación de sus productos con frecuencia, ya que el lapso entre una y otra versión de un mismo documento de los incluidos en la bibliografía de este trabajo de grado es, en promedio, de dos años, por lo que antes de empezar el estudio de cualquier documento de dicho fabricante deberá observarse la fecha de publicación del mismo con el fin de inferir si puede haber una versión más reciente para proceder a su búsqueda.

*Aprender a implementar redes de comunicación industrial es complicado porque hay muy poca bibliografía al respecto, de manera que es mejor, en lugar de perder días, semanas y hasta meses enteros buscando en internet o en bibliotecas de su ciudad o incluso de otra ciudad como en el caso de los autores de este proyecto de grado, invertir ese tiempo en leer las veces que sea necesario (hasta que se entiendan) los estándares del protocolo de interés y la documentación (manuales) del fabricante de los equipos disponibles para el montaje de la red.

Si hubiese una próxima inversión de la E3T-UIS en equipos Telemecanique sería muy bueno que parte de ella se dedicara a adquirir las siguientes referencias: para el Zelio un SR3NET01BD (extensión de comunicación para servidor Modbus

TCP/IP) y un SR3XT141BD (extensión de 8 entradas todo o nada y 6 salidas a rele); para los esclavos AS-i que tiene la E3T-UIS un terminal de direccionamiento (ya sea el XZMC11 o el ASITERV2) y su respectivo cable; para el M340 un BMX P34 2030 (procesador con un puerto para Modbus TCP/IP y otro para CANopen), un BMXEIA0100 (modulo Maestro AS-Interface) y un TCSEGPA23F14F (gateway Modbus TCP/IP a Profibus DP).

Tan pronto como el presupuesto lo permita, seria espectacular que la E3T-UIS empezara a robustecer su línea Siemens que hoy cuenta con 5 PLC S7-200, para que un día no muy lejano esta ultima tenga el mismo buen nivel con el que actualmente cuenta su línea Telemecanique; un buen comienzo es adquirir un PLC S7-300, un modulo lógico LOGO, una HMI Siemens de características similares a la HMI Magelis XBT GT 2330 de Telemecanique que posee la E3T-UIS y un variador de velocidad de características similares a los ATV31 (ATV31HU22M2) que también tiene la E3T-UIS.

Telemecanique vende las uniones y cables (RS485 EIA) necesarios para implementar el protocolo Modbus sobre línea serial (bus); no obstante estos accesorios, a pesar de su sencillo diseño, son bastante costosos (alrededor de \$100.000 el cable y \$500.000 la unión de tres terminales o T), por lo que podría procederse a construirlos: de ser así hay que tener mucho cuidado con la primera nota de la pagina 124 de la guía de programación TwidoSuite V2.0 del 2007 que permanece sin modificaciones en la pg 126 de la V2.2 (2009) del mismo documento, que fue la versión más reciente que se encontró; la citada nota es seguida por una grafica rotulada como "Cable RS485 EIA" que da la impresión de que en una red Modbus el pin 5 (DPT) solo debe conectarse al pin 7 (común "COM") en el maestro Twido y no en los esclavos Twido. Esta errónea impresión puede causar pérdidas considerables de tiempo, sobre todo en novatos en redes de comunicación industrial que ante la ausencia de comunicación modbus buscan insistentemente errores que probablemente no existan en los programas del

maestro y los esclavos. Lo correcto es que, sea en el conector del cable o en la unión que se construya, se cortocircuiten el pin 5 y el pin 7 tanto del maestro Twido modbus como del esclavo Twido modbus.

Con el “cable” (BMX FCW 303) para el modulo de entradas (BMX DDI 3202K) del PLC M340 se debe poner especial atención en su interpretación para evitar resultados inesperados e indeseados. Esto se debe a que dicho “cable” realmente es un conector del que salen dos grupos con 20 cables cada uno; uno de estos grupos de cables esta etiquetado como “A” y el otro como “B”, pero estas “A” y “B” no son las mismas “A” y “B” utilizadas en las paginas 46 (2/18) y 49 (2/21) del pdf “Modicon M340 plataforma de automatización catalogo ‘09” y en la página 120 del pdf “Modicon M340 Using Unity Pro Discrete Input/Output Modules User Manual 05/2010”, es decir, las primeras 16 entradas (%I0.0.0 a %I0.0.15) del modulo de entradas estarán en el grupo de cables “A” y las demás entradas (%I0.0.16 a %I0.0.31) en el grupo de cables “B”.

Saber es importante, tanto como certificar lo aprendido. Smart-ing que es la empresa autorizada por Schneider-Electric para brindar su soporte técnico en Colombia, ofrece un diplomado bastante flexible, ya que dura 7 meses y lo único que exige es desplazarse a la ciudad de Bogotá un fin de semana (3 días) al mes. Tiene un costo de \$3.500.000 (precio del 2010) e incluye, además de la capacitación, una certificación de Schneider-Electric como instalador de redes de comunicación industrial con equipos de sus marcas (entre esas Telemecanique).

La E3T-UIS cuenta con una sola HMI, lo que obliga a que la quinta y última práctica de laboratorio diseñada por este proyecto de grado tenga que desarrollarse asinando a todos los estudiantes del curso de laboratorio alrededor de la HMI, o dividiendo al curso en grupos que tendrían que hacer dicha práctica por turnos. Para evitar este problema se sugiere hacer otro proyecto de grado cuyo objetivo sea implementar dicha práctica con mímicos hechos en labview y

con mímicos físicos (tableros); esto sería un excelente complemento para este trabajo de grado, puesto que permitiría dividir al curso de laboratorio en dos cursos que entonces podrían realizar la quinta practica de laboratorio simultáneamente.

*Telemecanique no ofrece la opción de cambiar desde el software TwidoSuite la dirección que trae de fabrica un esclavo AS-i, lo que constituye una desventaja porque algunos de los esclavos AS-i que posee la E3T-UIS tienen la misma dirección y por consiguiente no podrán usarse en una misma red AS-i. Por esta razón dentro de las compras recomendadas por este trabajo de grado se incluye un direccionador AS-i.

BIBLIOGRAFIA

- a) RODRIGUEZ PENIN, Aquilino. Comunicaciones Industriales. Guía práctica. Primera Edición. Barcelona: Marcombo, 2008. 286p.
- b) PUERTO, Jhon Carlos. GOMEZ, José David. Redes industriales de comunicación basadas en profibus. Trabajo de grado Ingenieros Mecánicos. Bucaramanga: Universidad Industrial de Santander. Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería Mecánica, 1999. 427p.
- c) RONCANCIO REYES, Israel. Diseño e implementación de una solución de compatibilidad entre dispositivos interconectados bajo protocolo modbus. Trabajo de grado Ingeniero Electrónico. Bucaramanga: Universidad Industrial de Santander. Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, 2007. 167p..
- d) DUQUE PARDO, Jorge Eliecer. Procesador de comunicación Modbus. Implementación con microcontrolador. Trabajo de Investigación Magister en Ingeniería Electrónica. Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, 2006. 183p.
- e) CARREÑO MARTINEZ, Sinle Marcela. ARDILA ALBARRACIN, Pedro. Modbus. Monitoreo de la red empleando Labview. Trabajo de grado Ingenieros Electrónicos. Bucaramanga: Universidad Industrial de Santander. Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, 2005. 178p.
- f) MARRIAGA CABRALES, Arnol Rafael. ABREU SILVA, Oscar Augusto. Protocolo AS-Interface: implementación de un nodo esclavo en un

microcontrolador. Trabajo de grado Ingenieros Electrónicos. Bucaramanga: Universidad Industrial de Santander. Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, 2006. 71p.

g) ORDUZ TIBADUIZA, Diego Andrés. ORTIZ BALLESTEROS, Rigoberto. Protocolo AS-Interface, analizador de red. Trabajo de grado Ingenieros Electrónicos. Bucaramanga: Universidad Industrial de Santander. Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, 2006. 85p.

h) GUIZA GARCIA, Danny Alberto. CONTRERAS CORONEL, José Rodrigo. Protocolo AS-Interface. Implementación de un nodo maestro en PC. Trabajo de grado Ingenieros Electrónicos. Bucaramanga: Universidad Industrial de Santander. Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, 2007. 95p.

i) Modbus application protocol specification V1.1b [PDF]. 2006. http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf.

j) MODBUS over Serial Line Specification and Implementation Guide V1.02 [PDF].2006.http://www.modbus.org/docs/Modbus_over_serial_line_V1_02.pdf

k) Modbus Messaging on TCP/IP Implementation Guide V1.0b [PDF]. 2006. http://www.modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf

l) Can in Automation. CANopen Application Layer and Communication Profile CiADraftStandard301 [PDF]. 2002. <http://www.can-cia.org/index.php?id=440>

- m) Schneider Electric. Zelio Logic 2 Manual de Usuario SR2MAN01 [PDF]. 2007.[http://www.globaldownload.schneiderelectric.com/85257578007E5C8A/all/BE61F0C72321DB478825757700134C9C/\\$File/35007146.pdf](http://www.globaldownload.schneiderelectric.com/85257578007E5C8A/all/BE61F0C72321DB478825757700134C9C/$File/35007146.pdf)
- n) Telemecanique. Reles programables zelio logic. La nueva generación[PDF].2004.<http://www.xtec.cat/~jnogues%20documents/Zelio/Cat%E1logo-Tarifa%20ZelioII.pdf>
- o) Telemecanique. TwidoSuite V2.2 Guía de programación [PDF].2009.[http://www.globaldownload.schneiderelectric.com/85257578007E5C8A/all/1FD7B0C140308E4F8825757800476557/\\$File/35013228k01000.pdf](http://www.globaldownload.schneiderelectric.com/85257578007E5C8A/all/1FD7B0C140308E4F8825757800476557/$File/35013228k01000.pdf)
- p) Telemecanique. Twido Controladores Programables Modulos de Comunicación Guia de hardware [PDF]. 2009. [http://www.global-download.schneiderelectric.com/85257578007E5C8A/all/5DEAE1D6876596F388257578007ED722/\\$File/35013269k01000.pdf](http://www.global-download.schneiderelectric.com/85257578007E5C8A/all/5DEAE1D6876596F388257578007ED722/$File/35013269k01000.pdf)
- q) Telemecanique. Modicon M340 Using Unity Pro S Architectures and Communication Services Reference Manual. [PDF]. 2010. [http://www.globaldownload.schneiderelectric.com/852575A6007E5FD3/all/BAF48F81F446A6DA852576C00070A630/\\$File/35013341_k01_000_04.pdf](http://www.globaldownload.schneiderelectric.com/852575A6007E5FD3/all/BAF48F81F446A6DA852576C00070A630/$File/35013341_k01_000_04.pdf)
- r) Telemecanique. Modicon M340 with Unity Pro Serial Link User Manual[PDF].2010.[http://www.globaldownload.schneiderelectric.com/852575A6007E5FD3/all/C8119F196DAFA4E6852576C0006F9B67/\\$File/35012430_k01_000_04.pdf](http://www.globaldownload.schneiderelectric.com/852575A6007E5FD3/all/C8119F196DAFA4E6852576C0006F9B67/$File/35012430_k01_000_04.pdf)
- s) Telemecanique. Vijeo Designer Tutorial [PDF]. 2006. [http://www.global-download.schneiderelectric.com/85257563005C5CC2/all/C1256EDE0038AFB2C1256FB90042D3F1/\\$File/35007041_k01_000_04.pdf](http://www.global-download.schneiderelectric.com/85257563005C5CC2/all/C1256EDE0038AFB2C1256FB90042D3F1/$File/35007041_k01_000_04.pdf)

- t) Telemecanique. Manual del usuario de Magelis XBT GT y XBT GK [PDF]. 2007. [http://www.globaldownload.schneiderelectric.com/852575770039EC5E/all/301791EFB1F317B6852575830051D446/\\$File/manual%20de%20usuario%20magelis%20xbtgk%202007%20espa%F1ol.pdf](http://www.globaldownload.schneiderelectric.com/852575770039EC5E/all/301791EFB1F317B6852575830051D446/$File/manual%20de%20usuario%20magelis%20xbtgk%202007%20espa%F1ol.pdf)
- u) Telemecanique. iPC/XBT G/XBT GC/XBT GK/XBT GT/XBT GTW Magelis Controlador Modbus (RTU). [PDF]. 2008. Disponible en el CD del software Vijeo Designer V4.7 y en la ayuda de dicho software.
- v) Telemecanique. XBT G/XBT GC/XBT GK/XBT GT Magelis Controlador del dispositivo esclavo Modbus [PDF]. 2008. Disponible en el CD del software Vijeo Designer V4.7 y en la ayuda de dicho software..
- w) Telemecanique. Guía de programación Altivar 31 Variadores de velocidad para motores asíncronos V1.7. 2004. Disponible en el CD que el fabricante incluye por la compra del variador.
- x) Telemecanique. Altivar 31 User's manual Communication variables V1.7. 2005. Disponible en el CD que el fabricante incluye por la compra del variador.
- y) Telemecanique. Altivar 31 User's manual Modbus. 2005. Disponible en el CD que el fabricante incluye por la compra del variador.
- z) Telemecanique. Altivar 31 User's manual CANopen. 2005. Disponible en el CD que el fabricante incluye por la compra del variador.

ANEXO 1: MANUAL AS-i

CONTENIDO

	Pág.
1 MANUAL AS-i	227
1.1 PASOS PARA CREAR UNA APLICACIÓN AS-INTERFACE CON TWIDOSUITE	227
1.1.1 ABRA EL PROGRAMA TWIDOSUITE y CREE UN PROYECTO NUEVO DE LA SIGUIENTE FORMA:	227
1.1.2 ADICIONE EL MODULO AS-I AL AUTÓMATA.	229
1.1.3 AGREGUE LOS ESCLAVOS AS-I A LA RED	229
1.1.4 CONFIGURACIÓN BOTONERA XAL-S2002E	230
1.1.5 PROGRAMA LADDER CONFIGURACION DE LA BOTONERA XAL-2002E	235
1.1.5.1 TRANSFERIR EL PROGRAMA DEL PC AL PLC	238
1.1.6 CONFIGURACIÓN RELE-ASI20MT414OR	243
1.2 CONFIGURACIÓN DE ESCLAVOS QUE NO PERTENECEN A LA LISTA DEL GRUPO DE PERFILES DE AS-INTERFACE Y DE ESCLAVOS QUE YA TIENEN UNA DIRECCION DEFINIDA.	247
1.3 PRIMER PROGRAMA DE AS-interface	250
1.3.1 PROGRAMA EN LADDER	252
1.3.1.1 PRIMERA PARTE:	252
1.3.1.2 SEGUNDA PARTE:	254
1.3.1.3 TERCERA PARTE DEL PROGRAMA	256
1.3.1.4 CUARTA PARTE:	259
1.4 SEGUNDO PROGRAMA DE AS-interface	263
1.4.1 MONTAJE DE DOS MAESTROS Y CINCO ESCLAVOS	264
1.4.2 DIRECCIONES DE LOS ESCLAVOS AS-i	265
1.4.3 PROGRAMA EN LADDER	267
1.4.3.1 PRIMERA PARTE	267
1.4.3.2 PARTE 2	272
1.4.3.3 TERCERA PARTE	274

LISTA DE FIGURAS

	Pág.
Figura 1: CREAR UN PROYECTO	227
Figura 2: PLC TWIDO CON MÓDULO MAESTRO AS-i	228
Figura 3: VENTANA CON LAS DIRECCIONES PARA LOS ESCLAVOS AS-i	229
Figura 4: OPCIÓN IMPORTAR	230
Figura 5: GRUPO DE PERFILES DE AS-i	230
Figura 6: CONFIGURACIÓN BOTONERA XAL-2002E	231
Figura 7: PERFIL Y DIRECCIÓN DE LA BOTONERA XAL-200E	232
Figura 8: DIRECCIÓN DE LOS ESCLAVOS	232
Figura 9: NOMBRE TÉCNICO DE ESCLAVO	233
Figura 10: PERFIL	233
Figura 11: COMENTARIO	233
Figura 12: PARÁMETROS PERMANENTES	234
Figura 13: ENTRADAS Y SALIDAS	234
Figura 14: DIRECCIONAMIENTO EMPLEADO EN AS-i	235
Figura 15: PROGRAMA EN LADDER	236
Figura 16: PRIMERA PARTE DEL PROGRAMA EN LADDER	236
Figura 17: Segunda parte del programa (Rung 1)	237
Figura 18: Tercera parte del programa (Rung 2)	238
Figura 19: ARRANQUE AUTOMATICO	239
Figura 20: RUN/STOP	239
Figura 21: VENTANA DEPURAR	240
Figura 22: VENTANA TRANSFERENCIA PC → AUTÓMATA	241
Figura 23: PROGRAMA SIN ERRORES	241
Figura 24: CLIC EN MÓDULO TWDNOI10M3	242
Figura 25: BOTONERA XAL-S200E BIEN CONFIGURADA	242
Figura 26: RUN Y ACEPTAR	243
Figura 27: NUEVA DIRECCIÓN	243

Figura 28: INTERFACES IP20 COMPACTAS	244
Figura 29: DIRECCIÓN RELÉ ASI20MT4I4OR	245
Figura 30: PROGRAMA LADDER	245
Figura 31: RUN Y ACEPTAR	246
Figura 32: VENTANA ONLINE	246
Figura 33: PLC CON MODULO MAESTRO AS-I Y DOS ESCLAVOS AS-i	246
Figura 34: BOTONERA XAL-S2002 CON PERFIL 3FFF Y RELE ASI20MT4I4OR	247
Figura 35: ESCLAVO DESCONOCIDO CON PERFIL B A 7 E	248
Figura 36: ARREGLO DEL PERFIL PARA LA BOTONERA XAL-S200E	249
Figura 37: PERFIL CORRESPONDIENTA AL RELÉ ASI20MT4I4OR	249
Figura 38: ESCLAVOS BIEN CONFIGURADOS	250
Figura 39: MONTAJE DE UN MAESTRO Y DOS ESCLAVOS	251
Figura 40: DIRECCION DEL ESCLAVO XAL-S2002E	251
Figura 41: DIRECCION DEL ESCLAVO ASI20MT4I4OR	252
Figura 42: SECCIÓN 1	253
Figura 43: SECCIÓN 2	254
Figura 44: SECCIÓN 3	254
Figura 45: SECCIÓN 4	255
Figura 46: SECCIÓN 5	255
Figura 47: PALABRAS DE MEMORIA %MW0 Y %MW1	256
Figura 48: PALABRAS DE MEMORIA %MW1 Y %MW2	257
Figura 49: PALABRAS DE MEMORIA %MW3 Y %MW4	257
Figura 50: PALABRAS DE MEMORIA %MW5 Y %MW6	258
Figura 51: INSTRUCCIÓN ASI-CMD	258
Figura 52: TABLA DE PALABRAS DE MEMORIA	260
Figura 53: Modos de diagnosticos	261
Figura 54: Tabla para supervisar el estado del bus	262
Figura 55: RED AS-i	264
Figura 56: BOTONERA CON DIRECCIÓN 01A “RED 1”	265
Figura 57: BOTONERA CON DIRECCIÓN 02A “RED 1”	265
Figura 58: RELÉ CON DIRECCIÓN 03A “RED 1”	266

Figura 59: RELÉ CON DIRECCIÓN 01A “RED 2”	266
Figura 60: BOTONERA XAL-S200E CON DIRECCIÓN 02A “RED 2”	267
Figura 61: SECCIÓN 1	268
Figura 62: SECCIÓN 2	269
Figura 63: SECCIÓN 3	270
Figura 64: SECCIÓN 4	271
Figura 65: SECCIÓN 5	272
Figura 66: SECCIÓN 6	273
Figura 67: BLOQUE DE FUNCION	274
Figura 68: SALIDAS QUE ACTIVA EL BLOQUE EN CADA PASO	274
Figura 69: ORDENES PARA SUPERVISAR EL ESTADO DE LOS ESCLAVOS Y DEL BUS DE LA RED AS-I 1	275
Figura 70: ORDENES PARA SUPERVISAR EL ESTADO DE LOS ESCLAVOS Y DEL BUS DE LA RED AS-I 2	275
Figura 71: BLOQUE GENERADOR DE PASOS PARA LA SUPERVISIÓN	276
Figura 72: PASOS DE LA SUPERVISIÓN	276
Figura 73: TABLA DE ANIMACIÓN CON LOS ESCLAVOS 03A DE LA RED1 Y 02A DE LA RED 2	277

LISTA DE TABLAS

	Pág.
Tabla 1: Bits utilizados en el diagnóstico del bua AS-i	262
Tabla 2: Palabras de sistema en AS-i	263

1 MANUAL AS-i

1.1 PASOS PARA CREAR UNA APLICACIÓN AS-INTERFACE CON TWIDOSUITE

1.1.1 ABRA EL PROGRAMA TWIDOSUITE y CREE UN PROYECTO NUEVO DE LA SIGUIENTE FORMA:

Figura 1: CREAR UN PROYECTO

The screenshot shows the TWIDOSUITE software interface. On the left, there is a sidebar with a 'Gestión de proyectos' (Project Management) section. This section includes a list of actions: 'Actualizar la información del proyecto' (Update project information), 'Crear un proyecto nuevo' (Create a new project), 'Abrir un proyecto.' (Open a project), 'Abrir un proyecto reciente.' (Open a recent project), 'Copiar el proyecto actual' (Copy the current project), and 'Cerrar el proyecto actual.' (Close the current project). Below this list is a 'Crear un proyecto nuevo' (Create a new project) section with instructions: 'En esta pantalla puede: - seleccionar el nombre del proyecto e introducir la información del proyecto mediante un formulario; - adjuntar una foto en la ficha "Imagen".' (On this screen you can: - select the project name and enter the project information through a form; - attach a photo in the 'Image' tab). It also says 'Haga clic en "Crear" para crear el proyecto.' (Click on 'Create' to create the project.) and a 'Crear' button at the bottom.

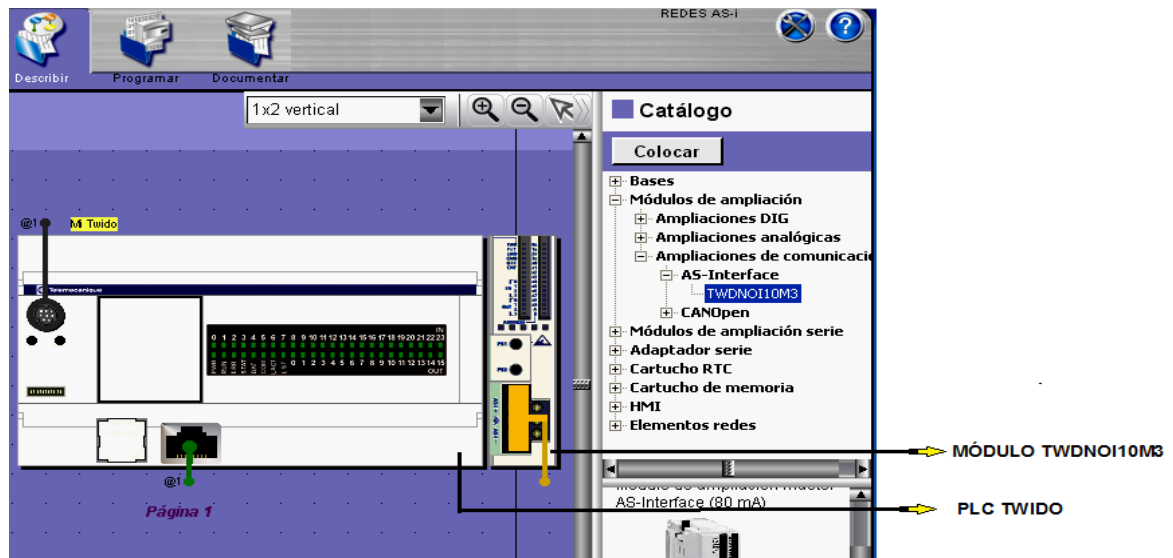
The main area of the interface is titled 'Información del proyecto' (Project Information). It contains two sections: 'Información del archivo' (File Information) and 'Información del proyecto' (Project Information). The 'Información del archivo' section has fields for 'Proyecto' (Project) with the value 'REDES AS-i' and 'Directorio' (Directory) with the value 'H:\PLAN DE PROYECTO DE GRADO\PROGRAMAS TWIDO SUITE\Gra'. The 'Información del proyecto' section has fields for 'Autor' (Author) with the value 'Jose manuel', 'Departamento' (Department) with the value 'xxxxx', 'Índice' (Index) with the value 'xxxxx', and 'Propiedad industrial' (Industrial Property) with the value 'UIS'. Below these fields is a table with three columns: 'Comentario' (Comment), 'Descripción' (Description), and 'Imagen' (Image). The table is currently empty.

Fuente: software twidosuite⁵²

⁵² La fuente de todas las figuras de los cuatro manuales es el software twidosuite

En la parte izquierda de su pantalla elija crear un proyecto nuevo de las siguientes opciones, crear un proyecto nuevo, abrir un proyecto, abrir un proyecto reciente. Cuando elije crear un proyecto nuevo, aparece a la derecha de estas opciones la información del proyecto, con los siguientes ítems: proyecto, directorio (elije donde quiere guardar el proyecto, como mis documentos, una memoria USB, etc.), autor, departamento, índice y propiedad industrial. Al concluir estas tareas haga clic en crear. Ver figura 1

Figura 2: PLC TWIDO CON MÓDULO MAESTRO AS-i



Ya creado el proyecto vaya a describir para que pueda visualizar el PLC y el catalogo. Elimine el PLC que aparece en la pantalla y dentro de la opción catalogo elija el PLC TWIDOSUITE TWDLCDE40DRF, para esta tarea, elija bases, en bases elija compactas y en compactas elija el autómata, seguidamente ubique el autómata en la pantalla.

1.1.2 ADICIONE EL MODULO AS-I AL AUTÓMATA.

Vaya al catalogo, en el catalogo elija módulos de ampliación, en módulos de ampliación elija ampliaciones, en ampliaciones de comunicaciones elija AS-interface y en AS-interface elija el módulo TWDNOI10M3. Nota: el esquema tiene que corresponder con el autómata físico (PLC real), en el número de módulos y el orden entre ellos. Ver figura 2

1.1.3 AGREGUE LOS ESCLAVOS AS-I A LA RED

Al Hacer doble clic en el modulo maestro AS-i se debe desplegar la siguiente ventana. Ver figura 3

Figura 3: VENTANA CON LAS DIRECCIONES PARA LOS ESCLAVOS AS-i

The screenshot shows a software window for configuring an AS-i system. At the top, there's a header section with the following information:

Descripción del módulo	Referencia	Dirección
	TWDNOI10M3	1
Descripción: Módulo de ampliación master AS-Interface (80 mA)		

Below this is the 'Configuración del módulo' section. On the left, there's a list of slave addresses (Esclavos std/A and Esclavos /B) ranging from 00 to 20. The right side of the window contains configuration options for a specific slave (Esclavo 1/A, Nombre: Slave 1A).

Características permanentes:

Perfil: IO ☐ F ID ☐ F ID1 ☐ F ID2 ☐ F

Comentario:

Parámetros permanentes:

☒ Bits 0 ☒ Ajuste 1 2 ☒ Ajuste 3

☐ Decimal 1 ☒ Ajuste 2 3 ☒ Ajuste 4

Entradas/Salidas:

Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo

Modo Master:

Intercambio de datos activado ☐ Parada red ☐ Direccionamiento automático ☒

Dé clic en la opción importar:

Figura 4: OPCIÓN IMPORTAR



Se debe desplegar la siguiente tabla

Figura 5: GRUPO DE PERFILES DE AS-i

Grupo de perfiles de AS Interface:

16 : Productos de seguridad

Catálogo AS-Interface: Productos de seguridad

Perfil	Nombre AS-Int...	@	Comentario
0.B.F.F	ASISSLB4	std	Interfaz de segur. para XB4
0.B.F.F	ASISSLB5	std	Interfaz de segur. para XB5
0.B.F.F	ASISSLC1	std	Inter de seguridad 2E de 1xM12
0.B.F.F	ASISSLC2	std	Inte. de seguridad 2E de 2xM12
0.B.F.F	ASISLLS	std	Interf. seguridad 2E de M16
7.F	ASISAFEMON1	std	Monitor de seguridad 1 canal
7.F	ASISAFEMON1B	std	Monitor de seguridad 1 canal
7.F	ASISAFEMON2	std	Monitor de seguridad 2 canales
7.F	ASISAFEMON2B	std	Monitor de seguridad 2 canales

La tabla de la figura 5 contiene la mayoría de los esclavos AS-i fabricados por schneider y utilizados en la industria. Si el esclavo que se busca no está en esta lista se debe configurarlo con el perfil y la referencia del dispositivo.

1.1.4 CONFIGURACIÓN BOTONERA XAL-S2002E

Buscamos en grupo de perfiles de AS-interface la caja de pulsadores y en ella elegimos XAL-S2002, como lo presenta la figura 6.

Figura 7: PERFIL Y DIRECCIÓN DE LA BOTONERA XAL-200E

The screenshot shows a configuration window for the XAL-S2002E device. On the left, a list of slave addresses (00-20) is shown, with 'XAL-S2002E' selected at address 01. The main area contains the following fields and tables:

Aplicar **Cancelar**

Esclavo 1/A Nombre:

Características permanentes

Perfil: IO ID ID1 ID2

Comentario

Parámetros permanentes

☒ Bits 0 ☒ Inutilizado 2 ☒ Inutilizado

☐ Decimal 1 ☒ Inutilizado

Entradas/Salidas

Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo
1	%IA1.1A.2		1	%QA1.1A.0	
2	%IA1.1A.3		2	%QA1.1A.1	

Modo Master

Intercambio de datos activado ☐ Parada red ☐ Direccionamiento automático ☐

La ventana de la figura 7 está conformada por los siguientes elementos:

Figura 8: DIRECCIÓN DE LOS ESCLAVOS

The screenshot shows a window for configuring slave directions. It features a list of slave addresses (00-08) on the left, with 'XAL-S2002E' selected at address 01. The window is titled 'Esclavos std/A' and 'Esclavos /B'.

La figura 8 muestra la dirección de los esclavos. Existen direcciones A y B y cada clase de dirección tiene 31 posibles lugares, de esta manera se puede colocar en una red AS-interface 62 Esclavos

Figura 9: NOMBRE TÉCNICO DE ESCLAVO

Esclavo 1/A	Nombre:	XAL-S2002E

La figura 9 presenta la referencia del esclavo o el inventario del producto, ejemplo cuando se compra el esclavo caja de pulsadores se conoce como botonera XAL-S2002E

Figura 10: PERFIL

Características permanentes								
Perfil:	IO	B	ID	A	ID1	7	ID2	E

La figura 10 presenta el perfil que es la matrícula que tiene el software del producto para su correcta configuración

Figura 11: COMENTARIO

Comentario	Caja c/puls. I Verde-0 Rojo

La figura 11 muestra el comentario, que describe las características del esclavo configurado

Figura 12: PARÁMETROS PERMANENTES

Parámetros permanentes

☒ Bits

0

☒

Inutilizado

☐ Decimal

1

☒

Inutilizado

2

☒

Inutilizado

Los parámetros permanentes son parámetros que trae por defecto el software twidosuite y son utilizados por algunos dispositivos esclavos. Los esclavos que pertenencia a la E3T no utilizan estos parámetros

Figura 13: ENTRADAS Y SALIDAS

Entradas/Salidas					
Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo
1	%IA1.1A.2		1	%QA1.1A.0	
2	%IA1.1A.3		2	%QA1.1A.1	

Las entradas y salidas que presentan la figura 13, son las entradas y salidas del esclavo que está conectado al modulo maestro AS-i por medio de la red. Estas entradas y salidas se direccionan como lo indica la figura. la figura 14, esta figura muestra el orden que se debe seguir para direccionar un dispositivo esclavo en la red AS-i. Ejemplo %IA1.1A.2, representa una entrada binaria de un elemento que está en la posición 2 de un esclavo que tiene una dirección 1A, conectado a un módulo AS-Interface que está en la posición 1 del bus de ampliación.

Figura 14: DIRECCIONAMIENTO EMPLEADO EN AS-i

%	IA, QA, IWA, QWA	x	.	n	.	i
Símbolo	Tipo de objeto	Ampliación dirección del módulo		dirección del slave		Canal n.º

En la siguiente tabla se especifican los valores de los objetos slave AS-Interface V2:

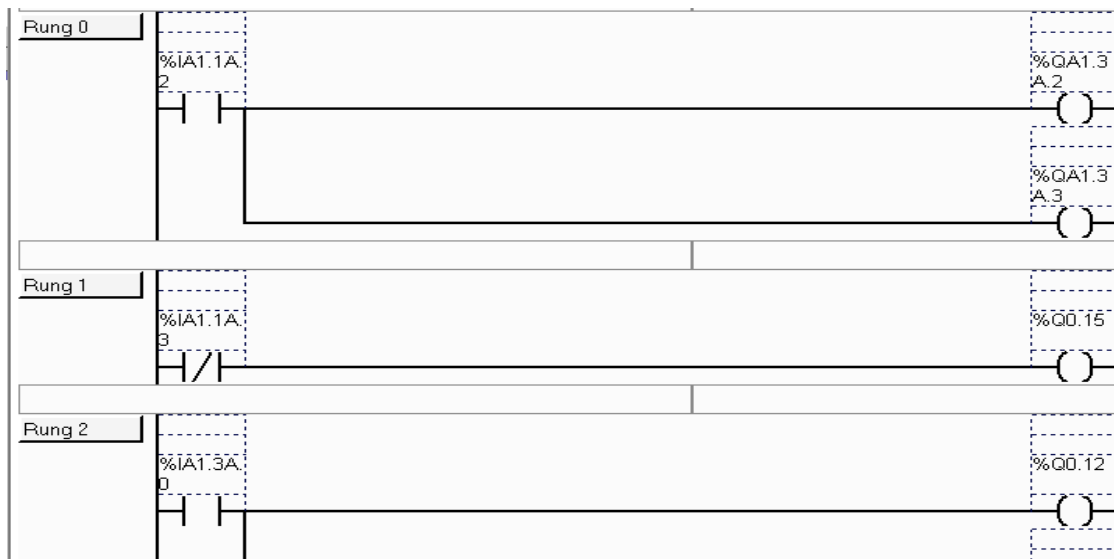
Componente	Valores	Comentario
IA	-	Imagen de la entrada binaria física del slave.
QA	-	Imagen de la salida binaria física del slave.
IWA	-	Imagen de la entrada física analógica del slave.
QWA	-	Imagen de la salida física analógica del slave.
x	De 1 a 7	Dirección del módulo AS-Interface en el bus de ampliación.
n	De 0A a 31B	El slot 0 no se puede configurar.
i	De 0 a 3	-

Fuente: “twidosuite V2.0, guía de programación 09/2007. Pg 255”

1.1.5 PROGRAMA LADDER CONFIGURACION DE LA BOTONERA XAL-2002E

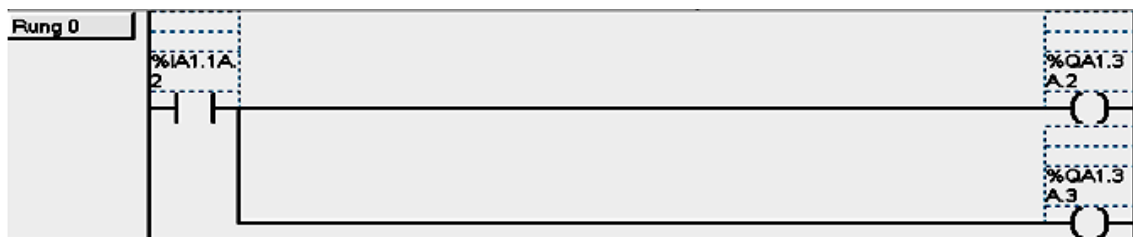
Si se desea tener acceso a las variables de la botonera, se debe elaborar un programa que se integre al PLC twido, en este programa se debe utilizar las variables de la botonera para poder tener supervisión del elemento físico. A continuación se presenta este programa el cual fue diseñado para el control de una botonera XAL-S2002E y un relé ASI20MT414OR que tiene conectado un sensor óptico a la entrada cero. Ver figura 15

Figura 15: PROGRAMA EN LADDER



Este programa corresponde al control de una red AS-i que tiene dos esclavos. El primer esclavo es una botonera XAL-2002E con dirección 1A y con dos entradas una normalmente abierta, otra normalmente cerrada. El segundo esclavo presenta un relé ASI20MT414OR, que está configurado en la dirección 3A y cuenta con cuatro entradas y cuatro salidas. En el rung (0) del programa en ladder presenta el control de las salidas dos y tres del relé, mediante la entrada normalmente abierta de la botonera. Ver figura 16

Figura 16: PRIMERA PARTE DEL PROGRAMA EN LADDER



Para su mejor entendimiento se explica cada entrada y salida de la red.

%I1.1A.2: representa una entrada binaria del botón normalmente abierto que está en la posición 2 del esclavo botonera (XAL-2002E) con una dirección 1A, conectado a un módulo AS-Interface que está en la posición 1 del bus de ampliación.

%QA1.3A.2: representa el estado binaria de la tercera salida (Q2) del relé ASI20MT414OR con dirección 3A, conectado a un módulo AS-Interface que está en la posición 1 del bus de ampliación. Nota Q2 representa la tercera salida del relé debido a que la primera salida del relé es Q0.

%QA1.3A.3

Tiene el mismo significado de %QA1.3A.2 excepto que el (3) corresponde a la cuarta salida del rele

La primera parte del programa (Rung 0) funciona de la siguiente manera, cuando se pulsa el botón normalmente abierto del esclavo XAL-S2002E, se activan las salidas 3 y 4 del relé

Figura 17: Segunda parte del programa (Rung 1)



%I1.1A.3 corresponde al botón normalmente cerrado del esclavo XAL-2002E, el cual cuando se oprime activa la salida 15 del autómeta twido.

Figura 18: Tercera parte del programa (Rung 2)



%I1.3A.0 corresponde a la entrada uno del esclavo ASI20MT414OR y %Q0.12 corresponde a la salida 12 del PLC. Cuando se activa la entrada uno del relé activa la entrada 12 del PLC

El siguiente paso que se debe hacer es transferir el programa al PLC twido, para ello se hace lo siguiente:

1.1.5.1 TRANSFERIR EL PROGRAMA DEL PC AL PLC

Para transferir el programa del PC al PLC se debe colocar el PLC en run. Para ello se tiene dos opciones:

1) a través del software:

Se da clic en configurar, después se selecciona (configurar el comportamiento) y seguidamente se marca la opción de (arranque automático en Run). Como se presenta en la figura 19.

Figura 19: ARRANQUE AUTOMATICO

2) **con hardware y software.** Se da clic en configurar, después se selecciona (configurar el material), enseguida se selecciona una de las entradas con RUN/STOP. Ver figura 20

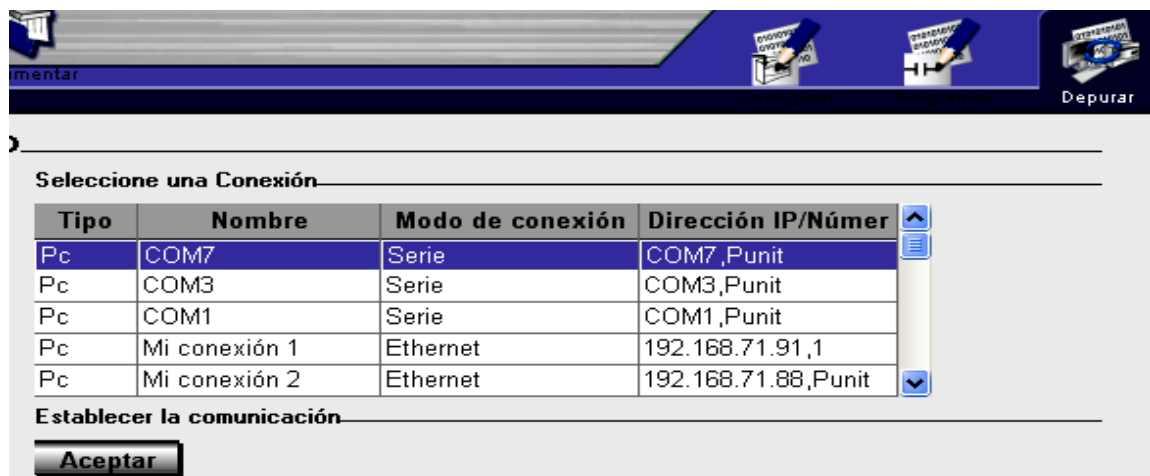
Figura 20: RUN/STOP

Duración del módulo								
Aplicar Cancelar								
Entradas/Salidas								
Tabla de entradas								
Pos	Dirección	Símbolo	Utilizado por	Filtrado	Guardado	¿Run/Stop?	Evento	Alta prioridad
1	%I0.0			3 ms	☐	☒		
2	%I0.1			3 ms	☐	☐		
3	%I0.2			3 ms	☐	☐	No utilizado	☐
4	%I0.3			3 ms	☐	☐	No utilizado	☐
5	%I0.4			3 ms	☐	☐	No utilizado	☐
6	%I0.5			3 ms	☐	☐	No utilizado	☐
7	%I0.6			3 ms	☐	☐		
8	%I0.7			3 ms	☐	☐		
9	%I0.8			3 ms	☐	☐		

Por último se une en el PLC el terminar de 24 V a la entrada seleccionada en el software y el terminar de cero volts al terminal común.

El siguiente paso que se debe hacer para transferir el programa del PC al PLC es ir a la ventana Depurar, la cual se presenta la figura 21

Figura 21: VENTANA DEPURAR



En la venta Depurar de la figura 21 se hace clic en Aceptar para que se despliegue la ventana de la figura 22 en la cual se elije la opción Transferencia PC → autómatas y clic en aceptar

Al lado derecho del programa está la opción “vigilar la configuración material”, se hace clic en esta opción para ver el programa que se le transmitió al PLC, la línea de color verde que presenta el programa indica que el programa no tiene errores. Ver figura 23

Figura 22: VENTANA TRANSFERENCIA PC → AUTÓMATA

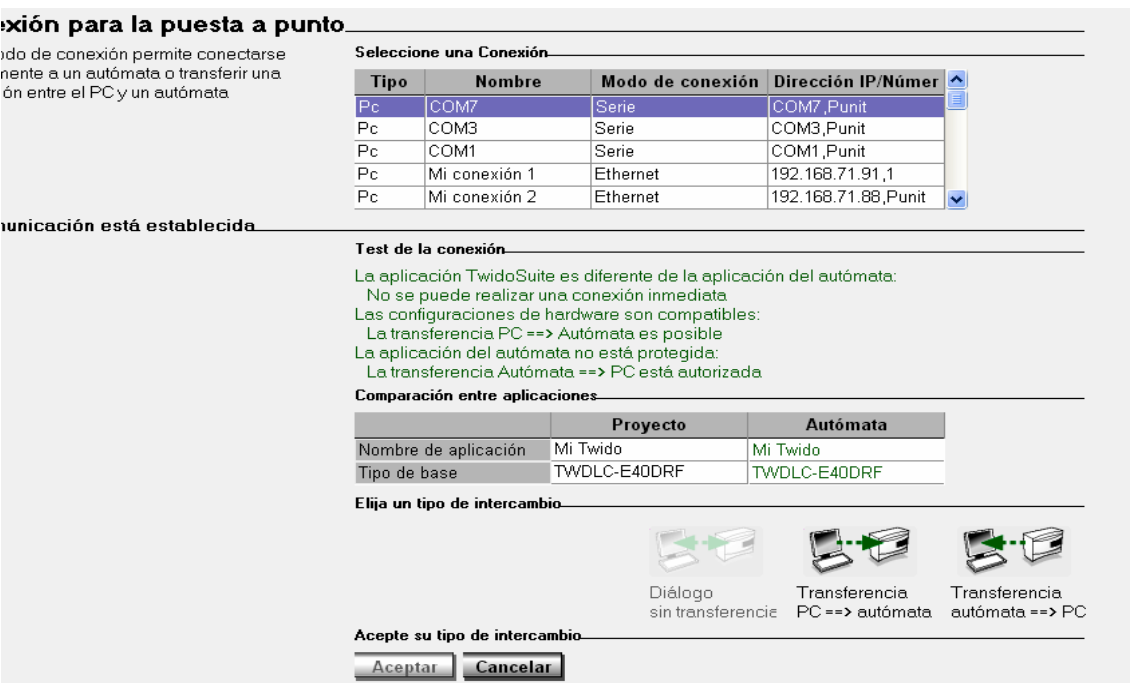
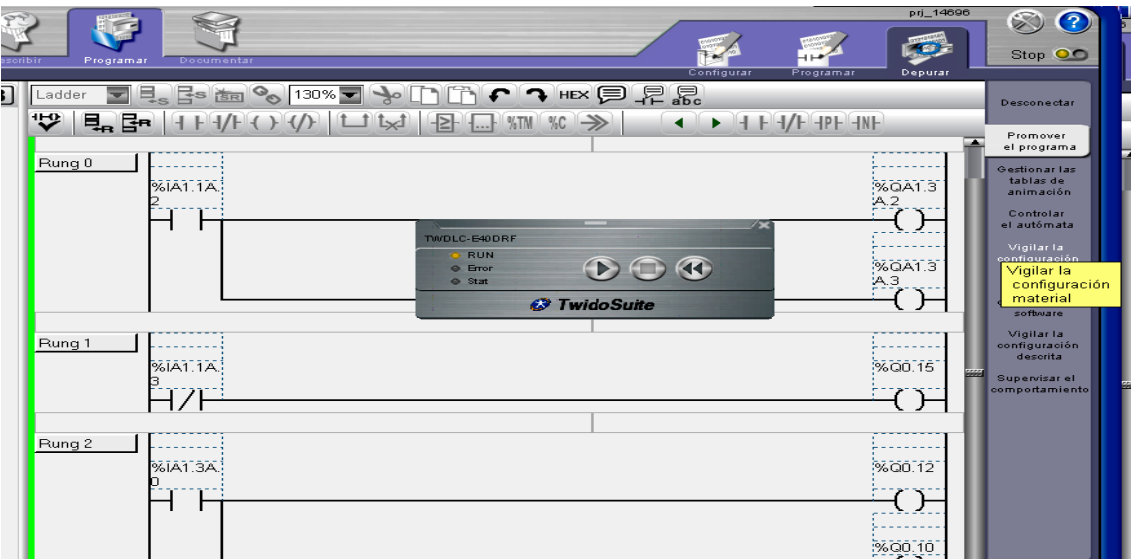
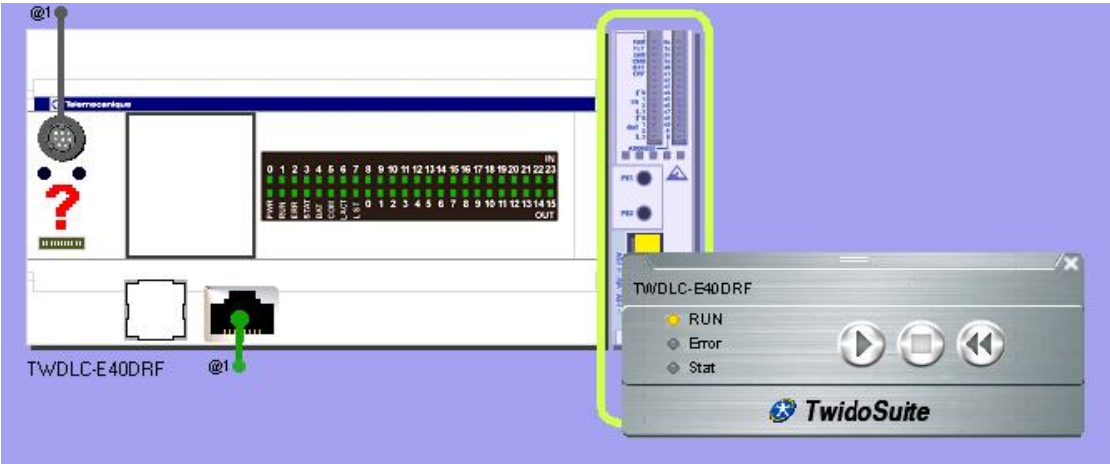


Figura 23: PROGRAMA SIN ERRORES



Para ver el esclavo botonera configurada se da clic al modulo TWDNOI10M3 en la ventana describir. Ver figura 24

Figura 24: CLIC EN MÓDULO TWDNOI10M3



Cuando se da clic en el modulo maestro AS-i se puede observar el esclavo. El color verde nos indica que están bien configurados en el software. Ver figura 25

Figura 25: BOTONERA XAL-S200E BIEN CONFIGURADA



Por último se da run y aceptar para que el PLC reconozca el esclavo. Ver figura 26

Figura 26: RUN Y ACEPTAR



1.1.6 CONFIGURACIÓN RELE-ASI20MT414OR

Para agregar otro esclavo es necesario seleccionar la dirección donde se quiere colocar el esclavo, para este ejemplo la dirección 03A. Ver figura 27

Figura 27: NUEVA DIRECCIÓN



Se Busca en grupo de perfiles de AS-interface la opción interfaces IP20 compactas, dentro de esta opción se encuentra la referencia del esclavo (ASI20MT414OR) y su descripción. Ver figura 28

Figura 28: INTERFACES IP20 COMPACTAS

Grupo de perfiles de AS Interface:

18 : Interfaces IP20 compactas

Catálogo AS-Interface: Interfaces IP20 compactas

Introducir

Perfil	Nombre AS-Int...	@	Comentario
0.A.7.0	ASI20MT4IE	etd	Interfaz tipo A/B 4E
3.A.7.0	ASI20MT211OTE	etd	Interf 2E/15 triac A/B
7.0.F.E	ASI20MT414OR	std	Interfaz est 4E/4S relés
7.0.F.E	ASI20MT414OS	std	Interfaz est 4E/4S estáticas
7.0.F.E	ASI20MT414OSA	std	Interfaz est 4E/4S estáticas
7.3.F.D	ASI20MA2VI	std	Interfaz 2 entr. anal. 4-20mA
7.3.F.D	ASI20MA2VU	std	Interfaz 2 entradas ana 0-10V
7.A.7.0	ASI20MT4I30RE	etd	Interfaz A/B 4E/3S relés
7.A.7.0	ASI20MT4I30SAE	etd	Interfaz A/B 4E/3S estáticas
7.A.7.0	ASI20MT4I30SE	etd	Interfaz A/B 4E/3S estáticas

Referencia: ASI20MT414OR

Interfaz IP20 compacta estándar para conectar 4 sensores y 4 accionadores convencionales al bus AS-i. entr. PNP 24Vcc alimentadas por el bus. Salidas tipo relés 230V

Versión:

Proveedor: Telemecanique

Perfil AS-i: 7.0.F.E

Configuración E/S:

Entradas:

Tipo PNP 24Vcc alimentadas por el bus 200 mA máx

D0: I-1 Señal sensor 1

D1: I-2 Señal sensor 2

D2: I-3 Señal sensor 3

D3: I-4 Señal sensor 4

Salidas:

Tipo relés 230V AC. Desactivado en 0 por watchdog

D0: O-1 Salida accionador 1

D1: O-2 Salida accionador 2

D2: O-3 Salida accionador 3

D3: O-4 Salida accionador 4

Clic en introducir y se despliega una ventana con las características (nombre, perfil, comentario, parámetros permanentes, entradas y salidas del rele. Ver figura 29

El siguiente paso es elaborar un programa en el PLC twido que sirva para tener acceso a las variables del relé ASI20MT414OR. Ver figura 30

Figura 29: DIRECCIÓN RELÉ ASI20MT4I4OR

Esclavos std/A

Esclavos /B

00

01

02

03

04

05

06

07

08

09

10

11

12

13

14

15

16

XAL-S2002E

ASI20MT4I4OR

Aplicar

Cancelar

Esclavo 2/A

Nombre: ASI20MT4I4OR

Características permanentes:

Perfil: IO 7 ID 0 ID1 F ID2 E

Comentario Interfaz est 4E/4S relés

Parámetros permanentes:

Bits

0

Inutilizado

2

Inutilizado

Decimal

1

Inutilizado

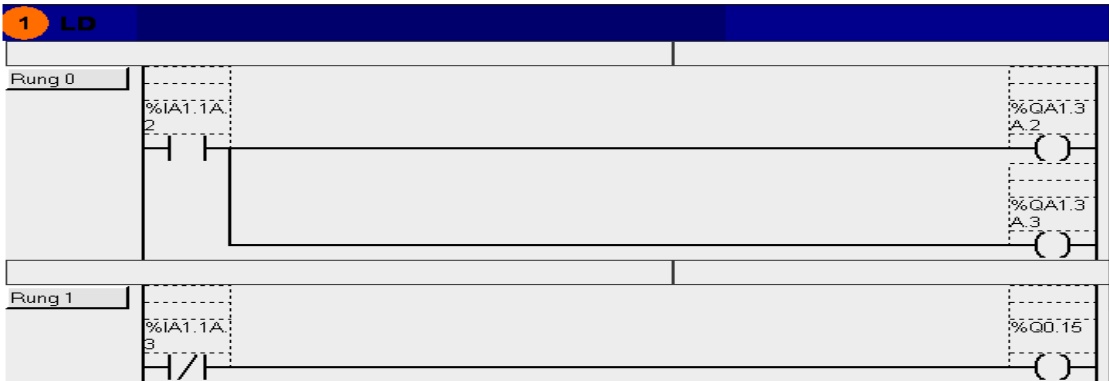
3

Inutilizado

Entradas/Salidas:

Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo
1	%IA1.2A.0		1	%QA1.2A.0	
2	%IA1.2A.1		2	%QA1.2A.1	
3	%IA1.2A.2		3	%QA1.2A.2	
4	%IA1.2A.3		4	%QA1.2A.3	

Figura 30: PROGRAMA LADDER



El siguiente paso que se debe realizar es transferir el programa del PC al PLC, similar al proceso que se empleo con la botonera. El último paso que se realiza es hacer clic en RUN Y Aceptar. Ver figura 31

La ventana de figura 32 describe los esclavos cuando son reconocidos por el maestro. El color verde indica que existe comunicación entre el maestro y los esclavos

Figura 31: RUN Y ACEPTAR



Figura 32: VENTANA ONLINE

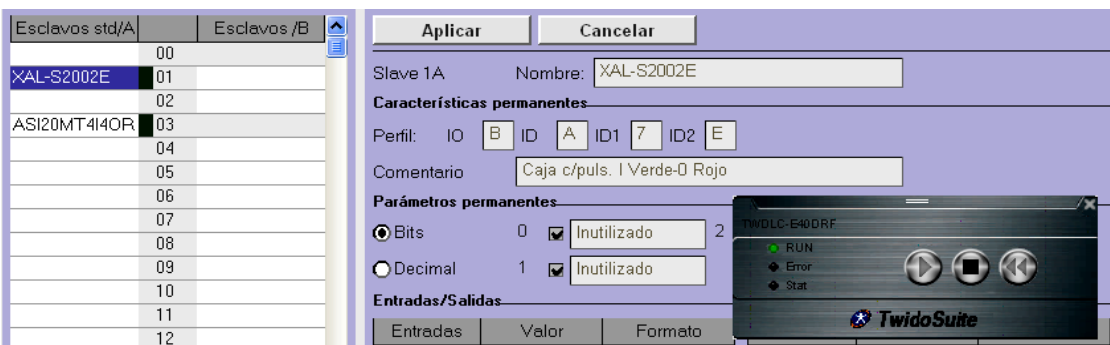
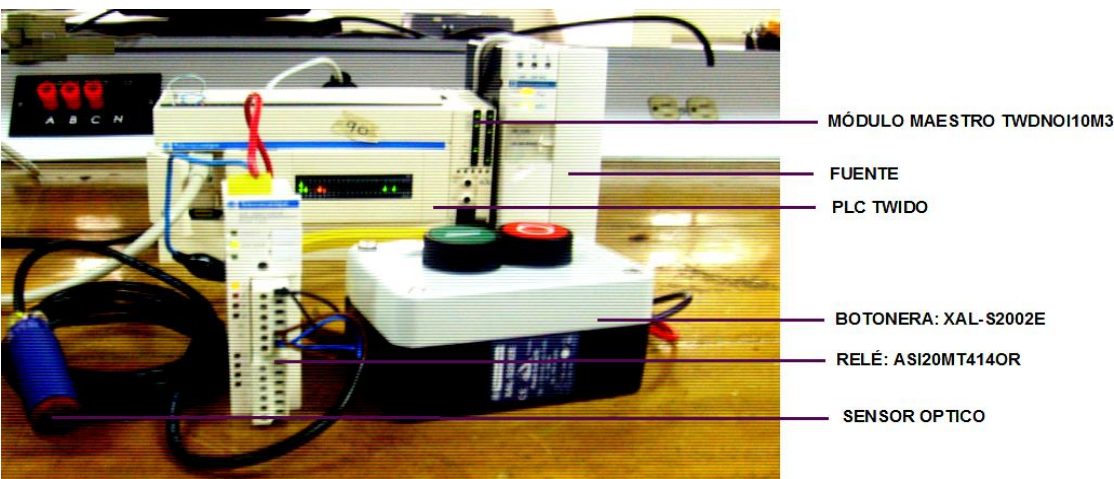


Figura 33: PLC CON MODULO MAESTRO AS-I Y DOS ESCLAVOS AS-i



El montaje físico de los esclavos y PLC se puede ver en la figura 33

1.2 CONFIGURACIÓN DE ESCLAVOS QUE NO PERTENECEN A LA LISTA DEL GRUPO DE PERFILES DE AS-INTERFACE Y DE ESCLAVOS QUE YA TIENEN UNA DIRECCION DEFINIDA.

Los esclavos traen una dirección, un perfil y una matrícula de fábrica para que sean configurados en software de programación. Cuando el software twidosuite no reconoce el dispositivo seguramente es porque no aparece en su lista de perfiles de AS-interface, este es el caso del ejemplo que se presenta a continuación, en el cual se ilustra la manera de configurar un dispositivo cuando no lo reconoce el software twidosuite y de configurar otro dispositivo cuando es asigna en una dirección que no coincide con la que trae por defecto.

Figura 34: BOTONERA XAL-S2002 CON PERFIL 3FFF Y RELE ASI20MT4I4OR

Esclavos std/A	Esclavos /B
	00
XAL-S2002	01
ASI20MT4I4OR	02
	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14

Aplicar		Cancelar			
Esclavo 1/A	Nombre:	XAL-S2002			
Características permanentes:					
Perfil:	ID	ID1	ID2		
	3	F	F		
Comentario: Caja c/puls. I Verde-0 Rojo					
Parámetros permanentes:					
☒ Bits	0	☒ Inutilizado	2 ☒ Inutilizado		
☐ Decimal	1	☒ Inutilizado	3 ☒ Inutilizado		
Entradas/Salidas:					
Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo
1	%I1.1A.0		1	%QA1.1A.2	
2	%I1.1A.1		2	%QA1.1A.3	

Ejemplo: se programó el esclavo correspondiente a la botonera XAL-S2002 con perfil: 3 F F F, que trae por defecto la librería de grupo de perfiles de AS- interface del software twido, en la dirección 01A y el esclavo relé ASI20MT4I40R con perfil: 7 0 F E, en la dirección 2A. Ver figura 34

Cuando se depura el programa, el esclavo no es reconocido y se presenta otro esclavo como desconocido con otro perfil diferente al asignado. Ver figura 35

Figura 35: ESCLAVO DESCONOCIDO CON PERFIL B A 7 E

Esclavos std/A		Esclavos /B	
	00		
XAL-S2002	01		
ASI20MT4I40R	02		
Desconocido	03		
	04		
	05		
	06		
	07		
	08		
	09		
	10		
	11		
	12		
	13		
	14		
	15		

Aplicar

Cancelar

Slave 1A

Nombre: XAL-S2002

Características permanentes

Perfil:

IO B ID A ID1 7 ID2 E

Comentario

Caja c/puls. I Verde-0 Rojo

Parámetros permanentes

☒ Bits

0 ☒ Inutilizado

2 ☒ Inutilizado

☐ Decimal

1 ☒ Inutilizado

Entradas/Salidas

Entradas	Valor	Formato	Salidas	Valor	Formato
%IA1.1A.2	0	Dec.	%QA1.1A.0	0	Dec.
%IA1.1A.3	0	Dec.	%QA1.1A.1	0	Dec.

Para solucionar este inconveniente es necesario cambiar el perfil que trae la lista de esclavos por el perfil que muestra el software. La referencia del esclavo 1 (botonera) también se modifica por la referencia que trae la botonera en su manual, en este caso XAL-S2002 por XAL-S2002E. Ver figura 36

La figura 36 nos presenta la botonera configurada correctamente, pero presenta otro esclavo desconocido en la dirección 03A, este esclavo tiene el perfil del relé ASI20MT4I40R, lo que hace pensar que la dirección correcta del rele es la 03A y no la 02A. Ver figura 37

Figura 36: ARREGLO DEL PERFIL PARA LA BOTONERA XAL-S200E

Esclavos std/A	Esclavos /B
00	
XAL-S2002E	01
ASI20MT4I4O	02
Desconocido	03
04	
05	
06	
07	
08	
09	
10	
11	
12	
13	
14	
15	

Aplicar **Cancelar**

Slave 1A Nombre: XAL-S2002E

Características permanentes

Perfil: IO **B** ID **A** ID1 **7** ID2 **E**

Comentario: Caja c/puls. I Verde-0 Rojo

Parámetros permanentes

☒ Bits 0 ☒ Inutilizado 2 ☒ Inutilizado

☐ Decimal 1 ☒ Inutilizado

Entradas/Salidas

Entradas	Valor	Formato	Salidas	Valor	Formato
%IA1.1A.2	0	Dec.	%QA1.1A.0	0	Dec.
%IA1.1A.3	0	Dec.	%QA1.1A.1	0	Dec.

Figura 37: PERFIL CORRESPONDIENTA AL RELÉ ASI20MT4I4OR

Esclavos std/A	Esclavos /B
00	
XAL-S2002E	01
ASI20MT4I4O	02
Desconocido	03
04	
05	
06	
07	
08	
09	
10	
11	
12	
13	
14	
15	
16	

Aplicar **Cancelar**

Slave 3A Nombre: Desconocido

Características permanentes

Perfil: IO **7** ID **0** ID1 **F** ID2 **E**

Comentario:

Parámetros permanentes

☐ Bits 0 ☐ 2 ☐

☐ Decimal 1 ☐ 3 ☐

Entradas/Salidas

Entradas	Valor	Formato	Salidas	Valor	Formato
%IA1.3A.0	0	Dec.	%QA1.3A.0	0	Dec.
%IA1.3A.1	0	Dec.	%QA1.3A.1	0	Dec.
%IA1.3A.2	0	Dec.	%QA1.3A.2	0	Dec.
%IA1.3A.3	0	Dec.	%QA1.3A.3	0	Dec.

Se Hace los cambios correspondientes y se depura para obtener los esclavos bien configurados, el color verde nos indica el éxito de la configuración. Ver figura 38.

Figura 38: ESCLAVOS BIEN CONFIGURADOS

1.3 PRIMER PROGRAMA DE AS-interface

- 1) Mostrar la sintaxis de las entradas y salidas de los esclavos AS-interfase así como su interacción entre ellas y las entradas y salidas del PLC Twido.
- 2) Mostrar el uso de la instrucción ASI_CMD para que el maestro AS-interface cambie el modo de funcionamiento.
- 3) Mostrar el uso de la instrucción ASI_CMD para obtener la lista de esclavos detectados (LDS), la lista de esclavos activos (LAS), la lista de esclavos erróneos (LPF) y el estado del bus.

250

Figura 39: MONTAJE DE UN MAESTRO Y DOS ESCLAVOS

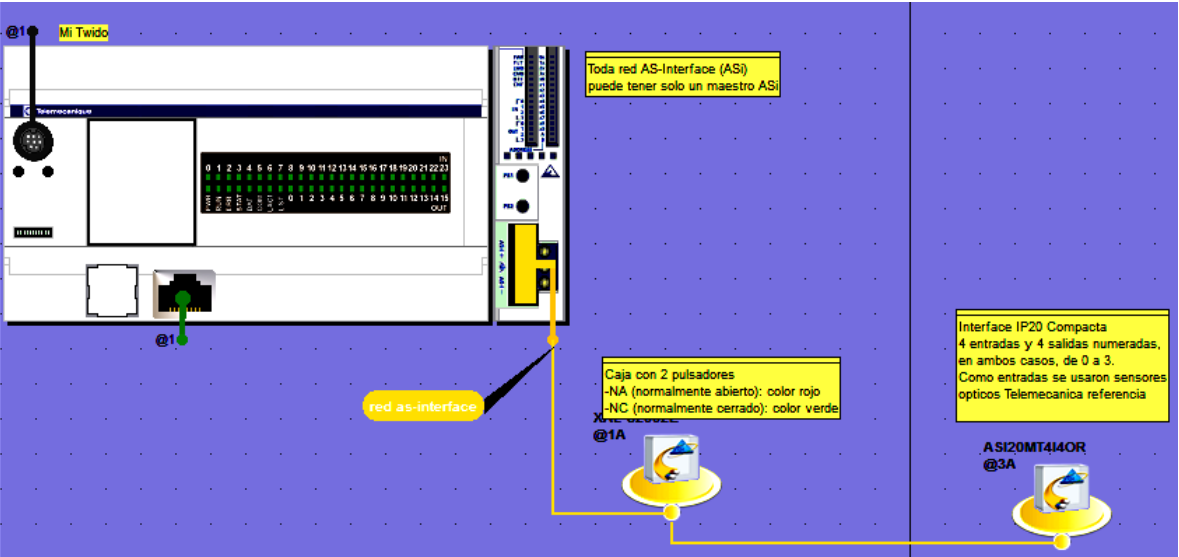


Figura 40: DIRECCION DEL ESCLAVO XAL-S2002E

Esclavos std/A	Esclavos /B
	00
XAL-S2002E	01
	02
ASI20MT4140	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14
	15
	16

Aplicar

Cancelar

Esclavo 1/A Nombre: XAL-S2002E

Características permanentes

Perfil: IO B ID A ID1 7 ID2 E

Comentario Caja c/puls. I Verde-0 Rojo

Parámetros permanentes

☒ Bits

0 ☒ Inutilizado 2 ☒ Inutilizado

☐ Decimal

1 ☒ Inutilizado

Entradas/Salidas

Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo
1	%IA1.1A.2	RED1_ESC1_E	1	%QA1.1A.0	
2	%IA1.1A.3	RED1_ESC1_E	2	%QA1.1A.1	

La figura 40 presenta la dirección donde está configurada la botonera, perfil y el direccionamiento de sus entradas “%IA1.1A.2 y %IA1.1A.3” utilizadas en el programa ladder

Figura 41: DIRECCION DEL ESCLAVO ASI20MT4I4OR

The screenshot displays the configuration interface for a Siemens PLC slave. On the left, a list of slaves is shown, with 'ASI20MT4I4O' selected. The right pane shows the configuration for 'Esclavo 3/A'.

Esclavo 3/A Nombre: ASI20MT4I4OR

Características permanentes:

Perfil: IO 7 ID 0 ID1 F ID2 E

Comentario: Interfaz est 4E/4S relés

Parámetros permanentes:

☒ Bits 0 ☒ Inutilizado 2 ☒ Inutilizado

☐ Decimal 1 ☒ Inutilizado 3 ☒ Inutilizado

Entradas/Salidas:

Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo
1	%IA1.3A.0	RED1_ESC3_E	1	%QA1.3A.0	RED1_ESC3_S
2	%IA1.3A.1	RED1_ESC3_E	2	%QA1.3A.1	RED1_ESC3_S
3	%IA1.3A.2		3	%QA1.3A.2	RED1_ESC3_S
4	%IA1.3A.3		4	%QA1.3A.3	RED1_ESC3_S

Modo Master:

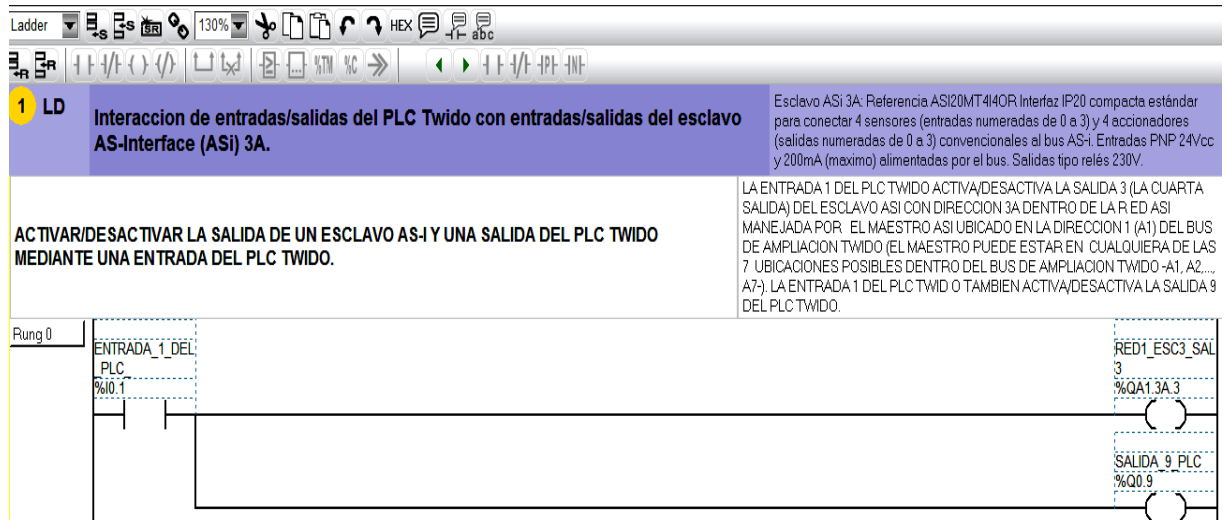
Intercambio de datos activado ☐ Parada red ☒ Direccionamiento automático

La figura 41 muestra la dirección donde está configurado el relé ASI20MT4140R, perfil y el direccionamiento de sus entradas y salidas. (Las entradas y salidas utilizadas en el programa son las que tienen un comentario en símbolo)

1.3.1 PROGRAMA EN LADDER

1.3.1.1 PRIMERA PARTE: Presenta la interacción de entradas / salidas del PLC Twido con las entradas/salidas del esclavo AS-interface con referencia ASI20MT4I4OR configurada en la dirección 3A

Figura 42: SECCIÓN 1



Sección 1: la entrada 1 (I0.1) del PLC twido activa/desactiva la salida 4 (Q.3) del esclavo AS-i con dirección 3A dentro de la red AS-i manejada por el maestro AS-i ubicado en la dirección 1 (A1) del bus de ampliación. La entrada 1 (I0.1) del PLC twido también activa/desactiva la salida 9 (Q0.9) del PLC twido. Ver figura 42

Sección 2: la primera entrada (I 0) del esclavo setea la salida 3 (Q0.3) del PLC twido. “setear una salida quiere decir que esta permanece activada aunque se desactive la entrada que activo dicha salida; la salida seteada solo se desactiva con una orden reset. Ver figura 43

Sección 3: la entrada 1 (I.1) del esclavo AS-i desactiva/activa la salida 3 (Q2) del mismo esclavo AS-i y desactiva/activa la salida 6 (Q0.6) del PLC twido. Ver figura 44

Figura 43: SECCIÓN 2

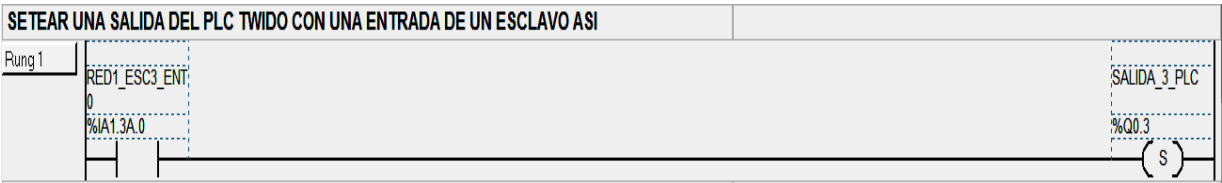
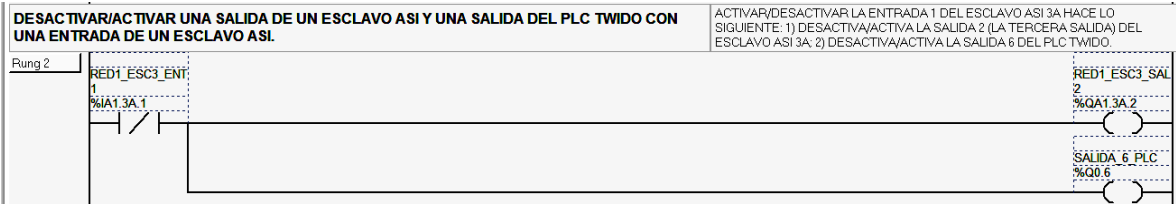


Figura 44: SECCIÓN 3

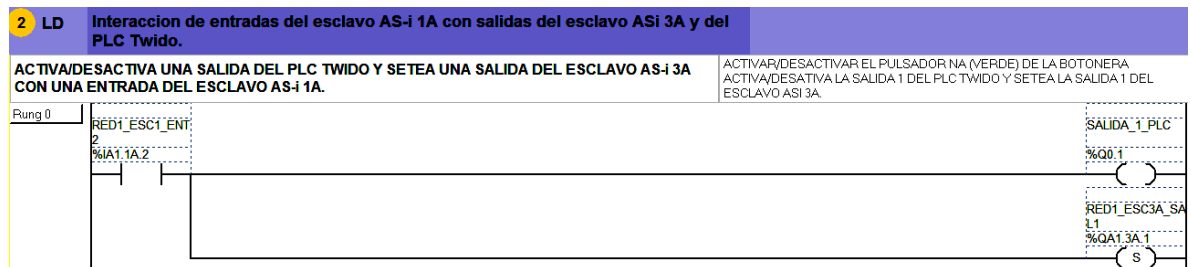


1.3.1.2 SEGUNDA PARTE: Presenta la interacción de entradas del esclavo AS-i configurado en la dirección 1A con salidas del esclavo AS-i configurado en la dirección 3A y con las salidas del PLC Twido.

El esclavo AS-i 1A tiene las siguientes características: Referencia XAL-S2002E. Caja con 2 pulsadores. Pulsador "I" (contacto normalmente abierto "NA") color verde. Pulsador "O" (contacto normalmente cerrado "NC") color rojo. Por estar asignada la entrada del pulsador rojo (NC) a un contacto normalmente cerrado (%IA1.1A.3), dicho pulsador funcionara como un pulsador NA; como la entrada del pulsador NA esta asignada a un contacto NA (%IA1.1A.2) este funcionara como pulsador NA, lo que permite considerar al esclavo AS-i 1A como una caja de dos pulsadores NA.

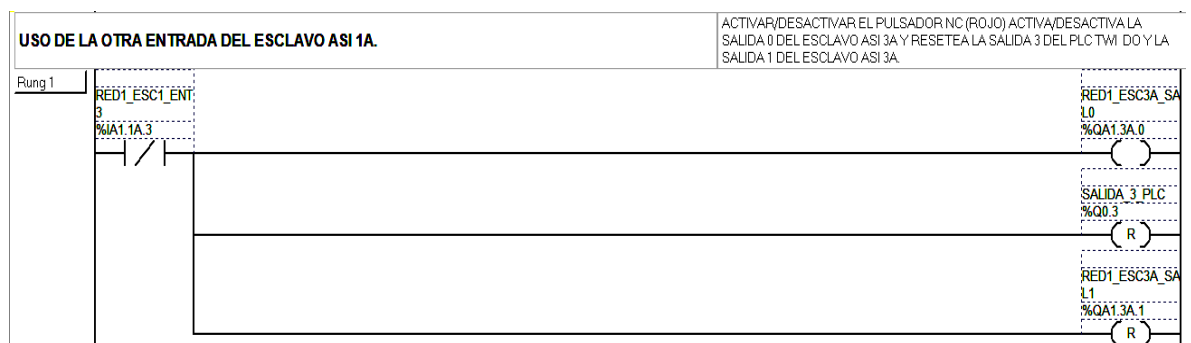
Sección 4: El pulsador NA (verde) de la botonera esclavo AS-i 3A activa/desactiva la salida 1 (Q0.1) del PLC twido y setea la salida 2 (Q1) del esclavo AS-i 3A. Ver figura 45

Figura 45: SECCIÓN 4



Sección 5: El pulsador NC (rojo) activa/desactiva la salida 1 (Q0) del esclavo AS-i 3A, resetea la salida 3 (Q0.3) del PLC twido y la salida 2 (Q1) del esclavo AS-i 3A. Ver figura 46

Figura 46: SECCIÓN 5



PROGRAMACIÓN Y DIAGNÓSTICO DEL BUS V2 AS-INTERFACE

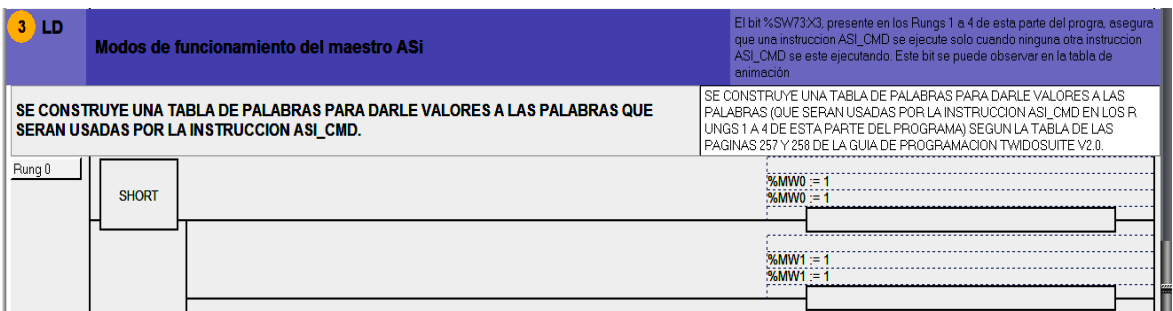
Las palabras de memorias y bits asociadas al bus AS-i aportan información del funcionamiento del bus, estado de los esclavos. El bus de expansión intercambia estos objetos explícitamente entre el controlador Twido y el maestro AS-Interface de la siguiente manera:

- ❖ A petición del programa de usuario mediante la instrucción: ASI_CMD
- ❖ Mediante la ventana de AS-Interface o la tabla de animación

1.3.1.3 TERCERA PARTE DEL PROGRAMA: Presenta la tabla de palabras utilizadas por la instrucción ASI_CMD y el uso de las palabras (%MW) de la tabla de palabras mediante instrucción ASI_CMD para gestionar los modos de funcionamiento del maestro AS-i

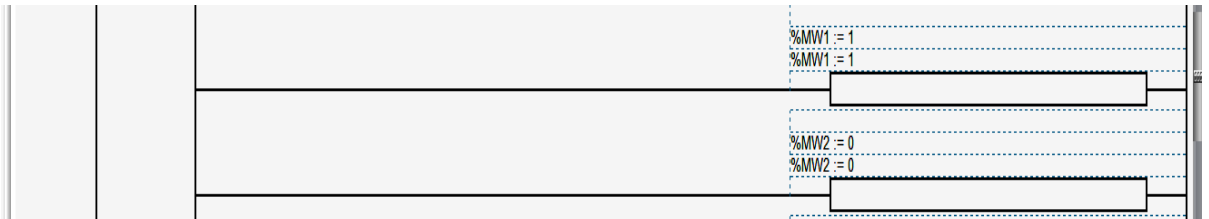
Sección 6: construcción de una tabla de palabras que servirán para darle valores a las palabras que serán usadas por la instrucción ASI_CMD e indicar de esta forma en qué modo de funcionamiento esta el maestro AS-i. “se recomienda consultar las páginas 257 y 258 de la guía de programación twidosuite V2.0”. Ver figuras 47, 48, 49 y 50

Figura 47: PALABRAS DE MEMORIA %MW0 Y %MW1



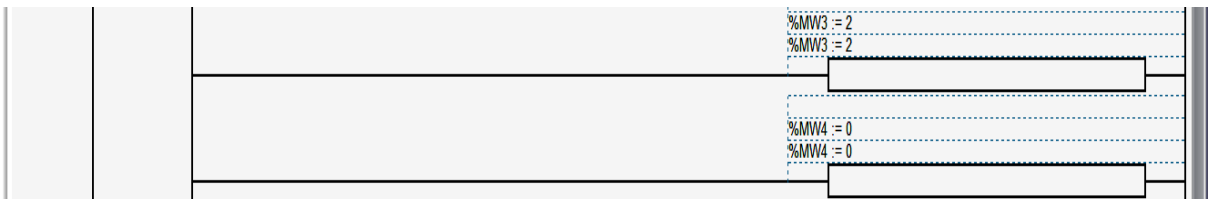
La figura 47 presenta los valores que deben tomar las palabras de memorias %MW0 y %MW1 para indicar que se va a solicitar el paso a modo offline

Figura 48: PALABRAS DE MEMORIA %MW1 Y %MW2



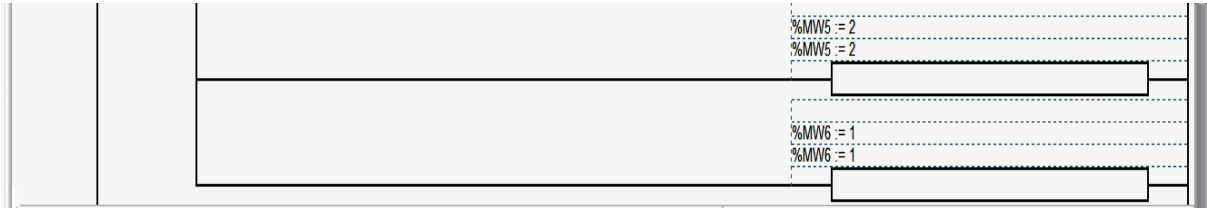
La figura 48 presenta los valores que deben tomar las palabras de memorias %MW1 y %MW2 para indicar que se va a solicitar la salida del modo offline. Observe que la palabra de memoria %MW1 también se utilizó para pasar a modo offline, gracias a que la tabla de palabras son palabras consecutivas

Figura 49: PALABRAS DE MEMORIA %MW3 Y %MW4



La figura 49 presenta los valores que deben tomar las palabras de memorias %MW3 y %MW4 para indicar que se va a solicitar que se prohíba el intercambio de datos entre el maestro y sus esclavos.

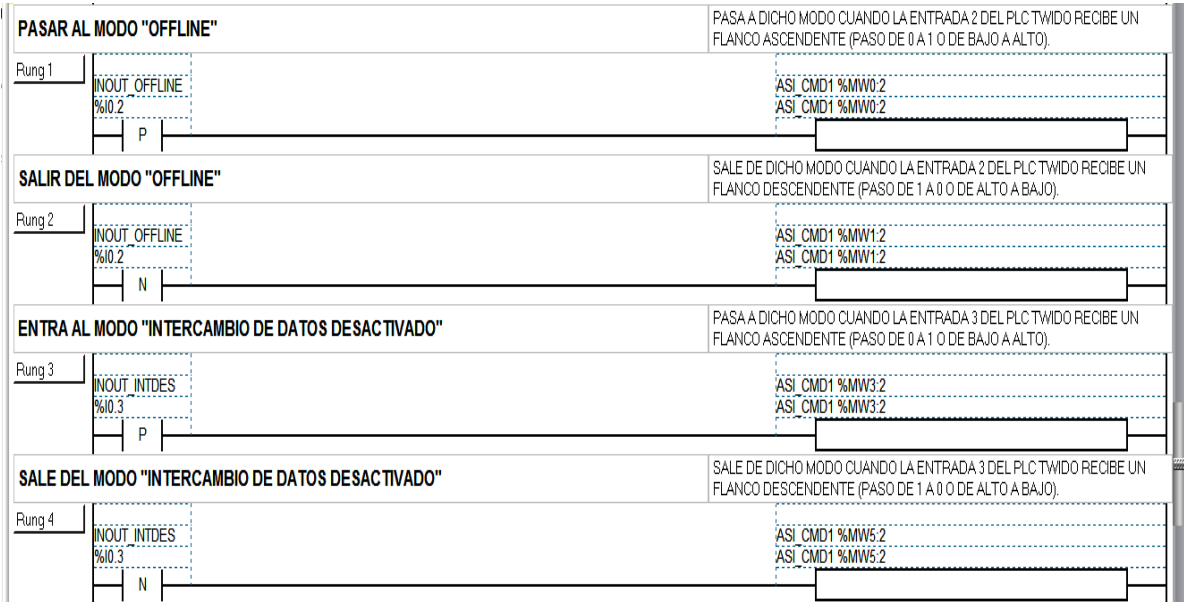
Figura 50: PALABRAS DE MEMORIA %MW5 Y %MW6



La figura 50 presenta los valores que deben tomar las palabras de memorias %MW5 y %MW6 para indicar que se va a solicitar el intercambio de datos entre el maestro y sus esclavos

Sección 7: presenta como se debe utilizar la instrucción ASI_CMD para gestionar los modos de operación del maestro AS-i

Figura 51: INSTRUCCIÓN ASI-CMD



La figura 51 muestra cómo la instrucción ASI_CMD1 gestiona el intercambio de los diferentes modos del maestro a medida que las entradas %I0.2 y %I0.3 se activan y desactivan. La sintaxis para configurar la instrucción ASI_CMD es la siguiente.

Esta se define de la siguiente forma: **ASI_CMDn %MWx: l.**

n es la dirección del modulo de ampliación AS-interface. Toma valores de 1 a 7.

x es el número de la primera palabra de memoria transmitida.

l es la longitud de la instrucción en palabras de memoria y puede ser máximo 2 (dos palabras de memoria).

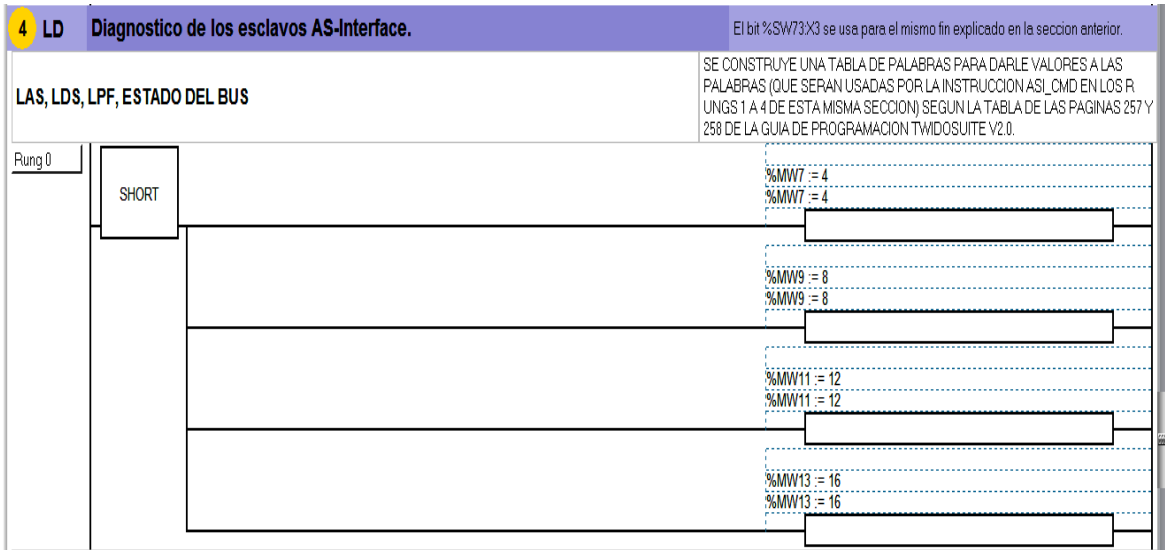
1.3.1.4 CUARTA PARTE: Presenta la tabla de palabras utilizadas por la instrucción ASI_CMD y el uso de las palabras de memorias de la tabla de palabras mediante instrucción ASI_CMD para gestionar el diagnóstico de los esclavos

Sección 8: presenta la construcción de una tabla de palabras que servirán para darle valores a las palabras que serán usadas por la instrucción ASI_CMD e indicar de esta forma el diagnostico de los esclavos de la red AS-i. Ver figura 52

La figura 52 presenta los valores que deben tomar las palabras de memorias %MW7, %MW9, %MW11 y %MW13 para indicar que solicitud del diagnostico de los esclavos se va a realizar, de esta manera cuando %MW7 toma el valor de cuatro se indica que se va a solicitar que el maestro lea la lista de esclavos activos (tabla LAS) de la dirección 0A a 15A (1 bit por esclavo), cuando %MW9 toma el valor de 8 se indica que se va a realizar la solicitud para la lectura de la lista de esclavos detectados (tabla LDS) de la dirección 0A a 15A (1 bit por esclavo), Cuando %MW11 toma el valor de 12 se indica que se va a realizar la lectura de la

lista de los dispositivos erróneos de los esclavos (tabla LPF) de la dirección 0A a 15A (1 bit por esclavo) y cuando %MW13 toma el valor de 16 se indica que se va a realizar la lectura del estado del bus.

Figura 52: TABLA DE PALABRAS DE MEMORIA

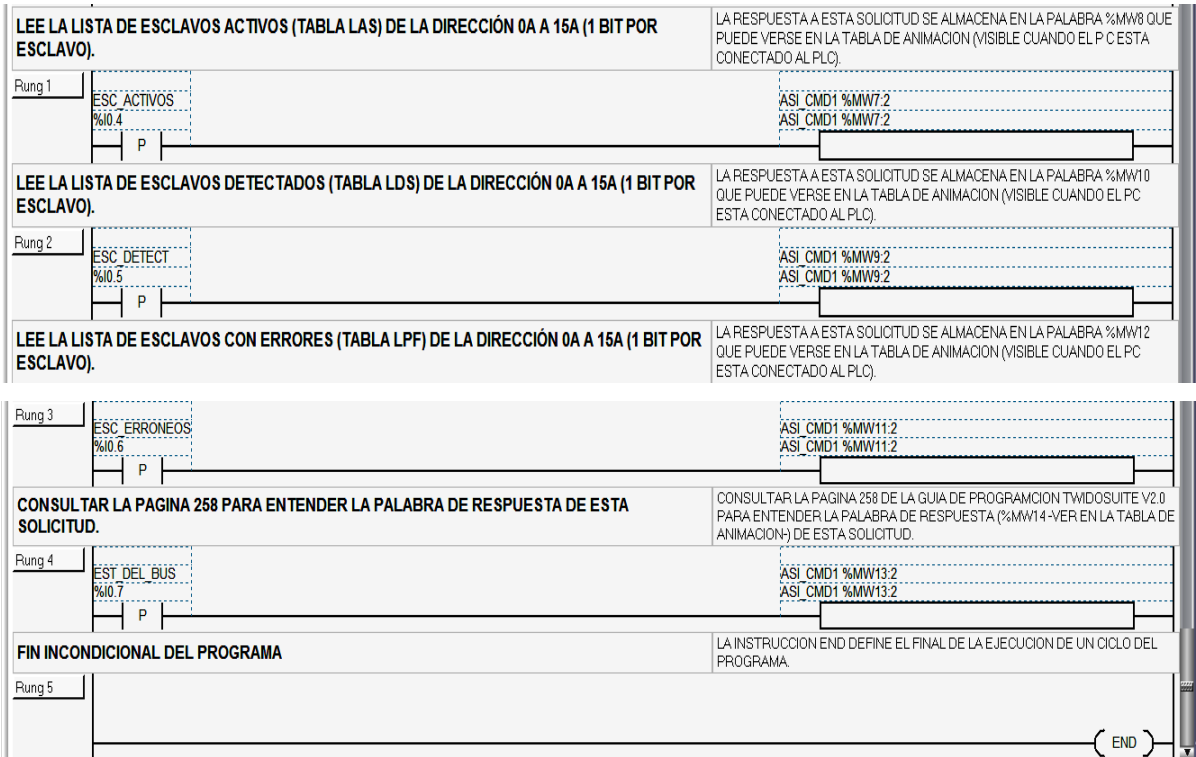


Sección 9: presenta como se debe utilizar la instrucción ASI_CMD para gestionar el diagnostico de los esclavos AS-Interface. Ver figura 53

La figura 53 presenta cómo la instrucción ASI_CMD1 gestiona el intercambio de las diferentes modos de diagnóstico de los esclavos a medida que las entradas %I0.4, %I0.5, %I0.6 y %I0 se activan y desactivan. La respuesta a cada solicitud se puede consultar en la tabla de animación en la palabra de memoria que le sigue a la palabra de memoria utilizada para la solicitud. Ejemplo se utilizó la palabra de memoria %MW7 para solicitar la lectura de la lista de esclavos activos de la dirección 0A a 15A, la respuesta a esta solicitud se puede observar en la palabra de memoria %MW8, cada bit de la palabra de memoria de respuesta representa un esclavo, cuando este bit vale uno significa que el esclavo esta activo, si por lo contrario el bit vale 0 significa

que el esclavo no está activo. En el caso de la solicitud de lectura del estado del bus utilizando una palabra de memoria % MWx, el formato del resultado a dicha solicitud se expresa en la palabra de memoria %MWx+1, representada en la tabla 1.

Figura 53: Modos de diagnósticos de los esclavos



En la figura 54 presenta el resultado de las solicitudes de diagnostico de los esclavos, el estado de las entradas y salidas, forzadas o no forzadas “forzar significa activar la entrada o salida a través de la tabla de animación” y la palabra de sistema %SW73 utilizada para conocer el estado de la red. Esta palabra utiliza la siguiente tabla para supervisar el estado del bus. Ver tabla 2

Figura 54: Tabla para supervisar el estado del bus

	Us	Dirección	Símbolo	Actual	Guardado	Formato
1	☐	%I0.0	RUN_STOP	F 1	0	Decimal
2	☒	%I0.1	Q09_Y_QA13A3	1	0	Decimal
3	☒	%I0.2	INOUT_OFFLINE	F 0	0	Decimal
4	☒	%I0.3	INOUT_INTDES	0	0	Decimal
5	☒	%I0.4	ESC_ACTIVOS	F 1	0	Decimal
6	☒	%I0.5	ESC_DETECT	F 1	0	Decimal
7	☒	%I0.6	ESC_ERRONEOS	F 0	0	Decimal
8	☒	%I0.7	EST_DEL_BUS	F 1	0	Decimal
9	☒	%Q0.3	SALIDA 3 PLC	F 1	0	Decimal
10	☒	%Q0.6	SALIDA 6 PLC	0	0	Decimal
11	☒	%Q0.9	SALIDA 9 PLC	1	0	Decimal
12	☒	%MW8	ESCLAVOS_ACTI	0000000000001010	0000000000000000	Binario
13	☒	%MW10	ESCLAVOS_DET	0000000000001010	0000000000000000	Binario
14	☒	%MW12	ESCLAVOS_ERR	0	0	Decimal
15	☒	%MW14	ESTADO_DEL_B	0010101100100101	0000000000000000	Binario
16	☐	%SW73	PMER_MAESTR	0000000000001011	0000000000000000	Binario
17	☒	%IA1.3A.0	RED1_ESC3_ENT	F 0	0	Decimal
18	☒	%IA1.3A.1	RED1_ESC3_ENT	F 1	0	Decimal
19	☒	%QA1.3A.0	RED1_ESC3_SAL	F 1	0	Decimal
20	☒	%QA1.3A.1	RED1_ESC3_SAL	F 0	0	Decimal
21	☒	%QA1.3A.2	RED1_ESC3_SAL	0	0	Decimal
22	☒	%QA1.3A.3	RED1_ESC3_SAL	1	0	Decimal
23	☒	%IA1.1A.2	RED1_ESC1_ENT	0	0	Decimal
24	☒	%IA1.1A.3	RED1_ESC1_ENT	1	0	Decimal
25						

Tabla 1: Bits utilizados en el diagnóstico del bus

%MWx+1		Designación (1 = OK, 0 = NOK)
Menos significativo	Bit 0	Configuración OK
	Bit 1	LDS.0 (slave presente en la dirección 0)
	Bit 2	Direccionamiento automático activo
	Bit 3	Direccionamiento automático disponible
	Bit 4	Modo de configuración activo
	Bit 5	Operación normal activa
	Bit 6	APF (problema de alimentación)
	Bit 7	Modo offline preparado
Más significativo	Bit 0	Periférico correcto
	Bit 1	Intercambio de datos activo
	Bit 2	Modo offline
	Bit 3	Modo normal (1)
	Bit 4	Fallo de comunicación con el master AS-Interface
	Bit 5	Instrucción ASI_CMD en curso
	Bit 6	Instrucción ASI_CMD errónea

Fuente: “twido guía de programación, Pg 258”

Tabla 2: Palabras de sistemas en AS-i

Palabras de sistema	Bit	Descripción
%SW73 y %SW74	0	Estado del sistema (= 1 si la configuración es correcta, 0 si no lo es)
	1	Intercambio de datos (= 1 si el intercambio de datos está activado, 0 si está en modo Intercambio de datos desactivado (véase p. 260))
	2	Sistema en stop (= 1 si el modo Local (véase p. 260) está activado, 0 si no lo está)
	3	Instrucción ASI_CMD finalizada (= 1 si finalizada, 0 si en curso)
	4	Error de instrucción ASI_CMD (= 1 si lo hay, 0 si no lo hay)

Fuente: “twido guía de programación, Pg 256”

1.4 SEGUNDO PROGRAMA DE AS-interface

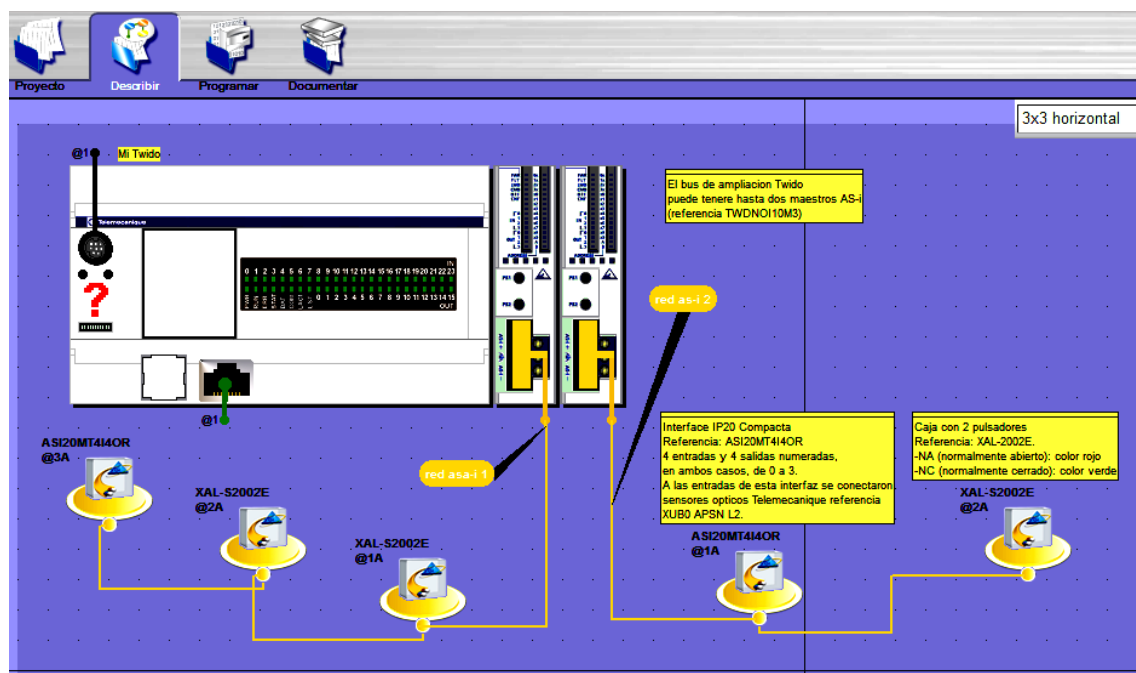
Este programa presenta tres objetivos:

- 1) trabajar con dos redes AS-i a través de dos maestros AS-i, que es el máximo de maestros que el PLC twido admite en su bus de ampliación. Activar salidas en las dos redes AS-i con entradas del PLC Twido y activar salidas del PLC Twido con entradas de las dos redes AS-i. Controlar salidas de la red AS-i 2 con entradas de la red AS-i 1 y salidas de la red AS-i 1 con entradas de la red AS-i 2.
- 2) Mostrar la gestión de las salidas AS-i y de la lectura de los estados y del bus AS-i que puede hacerse mediante el bloque de función del controlador del conmutador de tambor.
- 3) supervisar periódicamente el estado de los esclavos y el estado del bus.

1.4.1 MONTAJE DE DOS MAESTROS Y CINCO ESCLAVOS

La figura 55 presenta dos redes AS-i conformada por: un módulo maestro TWDN0I10M3, dos botoneras XAL S2002E y un relé ASI20MT4140R para la red1. Un módulo maestro TWDN0I10M3, Una botera XAL S2002E y un relé ASI20MT4140R para la red 2.

Figura 55: RED AS-i



La figura 55 presenta dos redes AS-i conformada por: un módulo maestro TWDN0I10M3, dos botoneras XAL S2002E y un relé ASI20MT4140R para la red1. Un módulo maestro TWDN0I10M3, Una botera XAL S2002E y un relé ASI20MT4140R para la red 2.

1.4.2 DIRECCIONES DE LOS ESCLAVOS AS-i: Las direcciones de los esclavos AS-i de la red 1 se presentan en las figuras 56, 57 y 58.

Figura 56: BOTONERA CON DIRECCIÓN 01A “RED 1”

Esclavos std/A	Esclavos /B
	00
XAL-S2002E	01
XAL-S2002E	02
ASI20MT4I4O	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14

Aplicar
Cancelar

Esclavo 1/A Nombre: XAL-S2002E

Características permanentes
Perfil: IO ☐ B ID ☐ A ID1 7 ID2 E

Comentario: Caja c/puls. I Verde-0 Rojo

Parámetros permanentes
☒ Bits 0 ☒ Inutilizado 2 ☒ Inutilizado
☐ Decimal 1 ☒ Inutilizado

Entradas/Salidas

Entradas	Dirección	Símbolo	Salidas	Dirección
1	%IA1.1A.2		1	%QA1.1A.0
2	%IA1.1A.3		2	%QA1.1A.1

La figura 56 presenta la botonera XAL-S2002E configurada en la dirección 01A

Figura 57: BOTONERA CON DIRECCIÓN 02A “RED 1”

Esclavos std/A	Esclavos /B
	00
XAL-S2002E	01
XAL-S2002E	02
ASI20MT4I4O	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14

Aplicar
Cancelar

Esclavo 2/A Nombre: XAL-S2002E

Características permanentes
Perfil: IO ☐ B ID ☐ A ID1 7 ID2 E

Comentario: Caja c/puls. I Verde-0 Rojo

Parámetros permanentes
☒ Bits 0 ☒ Inutilizado 2 ☒ Inutilizado
☐ Decimal 1 ☒ Inutilizado

Entradas/Salidas

Entradas	Dirección	Símbolo	Salidas	Dirección
1	%IA1.2A.2		1	%QA1.2A.0
2	%IA1.2A.3		2	%QA1.2A.1

En La figura 57 se observa la botonera XAL-S2002E configurada en la dirección 02A, la dirección del relé ASI20MT4I40 es 03A se puede observar en la figura 58,, las direcciones de los esclavos AS-i de la red 2 se presentan en las figuras 59 y 60

Figura 58: RELÉ CON DIRECCIÓN 03A “RED 1”

Esclavos std/A	Esclavos /B
	00
XAL-S2002E	01
XAL-S2002E	02
ASI20MT4I40	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14
	15
	16

Aplicar		Cancelar		
Esclavo 3/A		Nombre: ASI20MT4I4OR		
Características permanentes				
Perfil:	IO	ID	ID1 ID2	
	7	0	F E	
Comentario: Interfaz est 4E/4S relés				
Parámetros permanentes				
☒ Bits	0	☒ Inutilizado	2 ☒ Inutilizado	
☐ Decimal	1	☒ Inutilizado	3 ☒ Inutilizado	
Entradas/Salidas				
Entradas	Dirección	Símbolo	Salidas	Dirección
1	%IA1.3A.0		1	%QA1.3A.0
2	%IA1.3A.1		2	%QA1.3A.1
3	%IA1.3A.2		3	%QA1.3A.2
4	%IA1.3A.3		4	%QA1.3A.3

Figura 59: RELÉ CON DIRECCIÓN 01A “RED 2”

Esclavos std/A	Esclavos /B
	00
ASI20MT4I40	01
XAL-S2002E	02
	03
	04
	05
	06
	07
	08
	09
	10
	11
	12
	13
	14
	15
	16

Aplicar		Cancelar		
Esclavo 1/A		Nombre: ASI20MT4I4OR		
Características permanentes				
Perfil:	IO	ID	ID1 ID2	
	7	0	F E	
Comentario: Interfaz est 4E/4S relés				
Parámetros permanentes				
☒ Bits	0	☒ Inutilizado	2 ☒ Inutilizado	
☐ Decimal	1	☒ Inutilizado	3 ☒ Inutilizado	
Entradas/Salidas				
Entradas	Dirección	Símbolo	Salidas	Dirección
1	%IA2.1A.0		1	%QA2.1A.0
2	%IA2.1A.1		2	%QA2.1A.1
3	%IA2.1A.2		3	%QA2.1A.2
4	%IA2.1A.3		4	%QA2.1A.3

La figura 59 presenta el esclavo AS-i relé ASI20MT4I4O configurado en la dirección 01A

Figura 60: BOTONERA XAL-S200E CON DIRECCIÓN 02A “RED 2”

The screenshot shows a configuration window for the XAL-S200E device. On the left, a list of slaves is shown with 'XAL-S2002E' selected at address 02. The main configuration area on the right includes fields for 'Nombre' (XAL-S2002E), 'Perfil' (IO B, ID A, ID1 7, ID2 E), and 'Comentario' (Caja c/puls. I Verde-0 Rojo). Under 'Parámetros permanentes', 'Bits' and 'Decimal' are both set to 'Inutilizado'. The 'Entradas/Salidas' table is as follows:

Entradas	Dirección	Símbolo	Salidas	Dirección	Símbolo
1	%IA2.2A.2		1	%QA2.2A.0	
2	%IA2.2A.3		2	%QA2.2A.1	

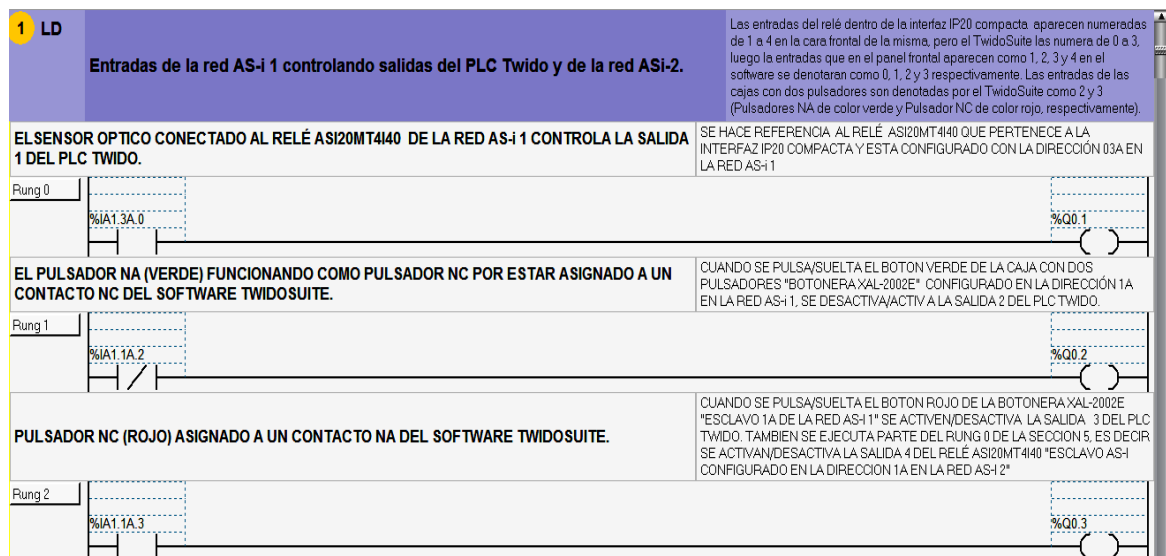
La figura 60 presenta la dirección 02A donde se configuro el esclavo XAL-S2002E

1.4.3 PROGRAMA EN LADDER

1.4.3.1 PRIMERA PARTE: Esta parte del programa consiste en activar/desactivar salidas en las dos redes AS-i con entradas del PLC Twido y activar/desactivar salidas del PLC Twido con entradas de las dos redes AS-i. También presenta como activar/desactivar salidas de la red AS-i 2 con entradas de la red AS-i 1 y activar/desactivar salidas de la red AS-i 1 con entradas de la red AS-i 2.

Sección 1: presenta en el rung 0 la activación/desactivación de la salida 1 (Qo.1) del PLC twido cuando el sensor óptico conectado en la entrada 1 del relé ASI20MT4I40 configurado en la dirección 03A de la red AS-i 1 se activa/desactiva, Seguidamente en el rung 1 se muestra como la salida 2 del PLC twido se desactiva/activa cuando el botón verde de caja de dos pulsadores XAL-2002E configurado en la dirección 1A de la red AS-i se pulsa/suelta. Por último en le rung 2 se presenta la activación/desactivación de la salida 3 (Q0.3) del PLC twido cuando se pulsa/suelta el botón rojo de la botonera XAL-2002E configurada en dirección 1A de la red AS-i 1 a demás con este mismo botón rojo se ejecuta parte del rung 0 de la sección 5, es decir se activan/desactiva la salida 4 del relé ASI20MT4I40 "esclavo AS-i configurado en la dirección 1A en la red AS-i 2". Ver figura 61

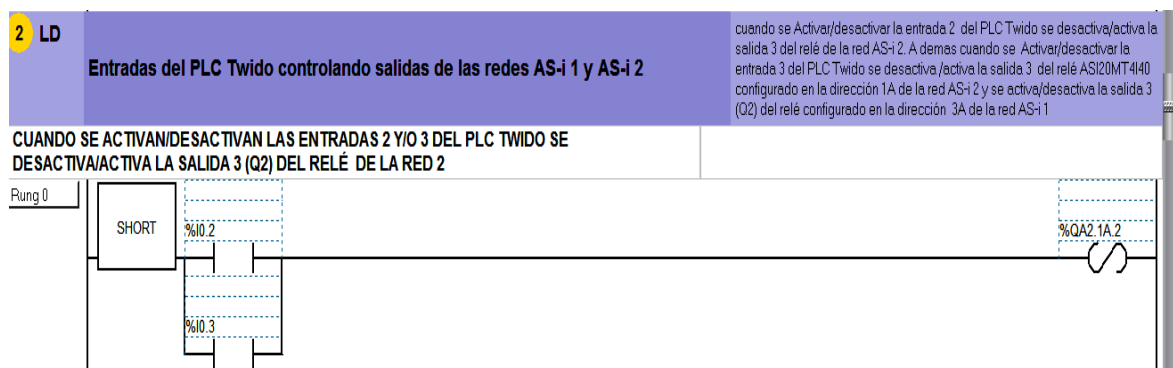
Figura 61: SECCIÓN 1



Sección 2: esta sección presenta como mediante las entradas del PLC Twido se controla las salidas de los esclavos de las redes AS-i 1 Y AS-i 2.

Cuando se activa/desactiva la entrada 2 del PLC twido se desactiva/activa la salida 3 (Q2) del relé ASI20MT4I40 configurado en la dirección 01A de la red AS-i 2. A demás cuando se Activar/desactivar la entrada 3 del PLC Twido se desactiva /activa la salida 3 del relé ASI20MT4I40 configurado en la dirección 01A de la red AS-i 2 y se activa/desactiva la salida 3 (Q2) del relé configurado en la dirección 3A de la red AS-i 1. Ver figura 62

Figura 62: SECCIÓN 2

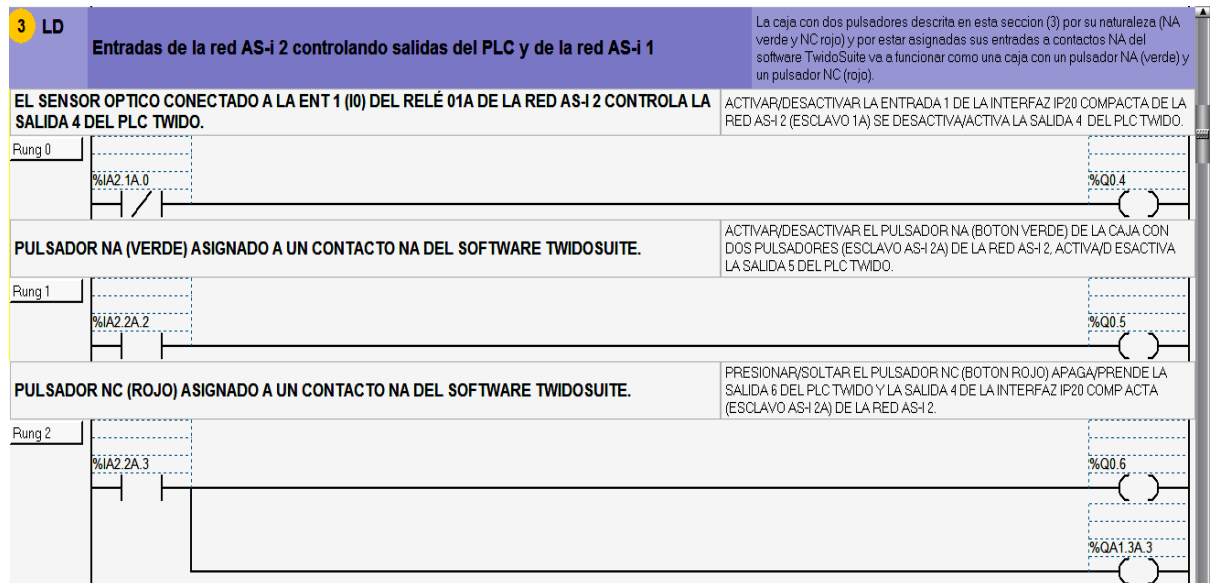


Sección 3: presenta las entradas de la red AS-i 2 controlando salidas del PLC Twido y de la red AS-i 1.

Esta sección presenta de qué forma en el rung 0, el sensor óptico conectado a la entrada 1 (I0) activa/desactiva la entrada 1 de la interfaz IP20 compacta “relé ASI20MT4I40” configurado en la dirección 01A de la red AS-i 2 la cual desactiva/activa la salida 4 (Q0.4) del PLC twido. Seguidamente el rung 1 presenta como se activa/desactiva la salida 5 (Q0.5) cuando el pulsador NA (botón verde) de la caja con dos pulsadores configurado en la dirección 02A de la red AS-i 2 se activa/desactiva y por último el rung 2 presenta la activación/desactivación de la salida 6 (Q0.6) del PLC Twido y la activación/desactivación de la salida 4 (Q.3) del relé ASI20MT4I40 configurado en la dirección 03A de la red AS-i 1 cuando se

activa/desactiva el pulsador NC (botón rojo) de la caja de dos pulsadores “botonera XAL- 2002E” configurada en la dirección 02A de la red AS-i 2. Ver figura 63

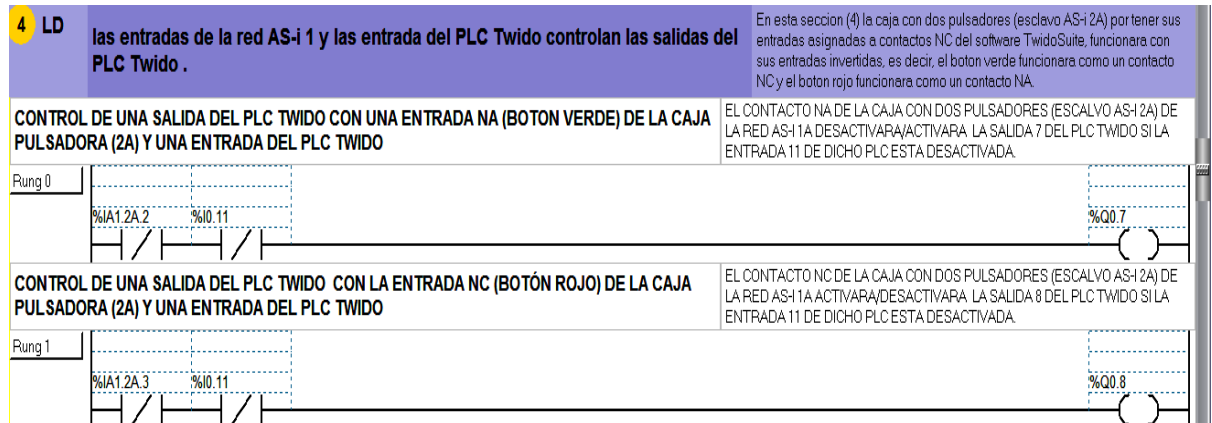
Figura 63: SECCIÓN 3



Sección 4: en esta sección las entradas de la red AS-i 1 con las entradas del PLC Twido controlan las salidas del PLC Twido.

Esta sección presenta en el rung 0 la activación/desactivación de la salida 7 (Q0.7) del PLC Twido cuando se desactiva/activa el botón verde (NA) de la caja de pulsadores “botonera XAL-2002E” configurada en la dirección 02A de la red AS-i 1 Y no se activa la entrada 11 (I0.11) del PLC Twido. A demás en esta sección se presenta en rung 1, la activación/desactivación de la salida 8 (Q0.8) del PLC Twido cuando se desactiva/activa el botón rojo (NC) de la caja de pulsadores “botonera XAL-2002E” configurada en la dirección 02A de la red AS-i 1 y no se activa la entrada 11 (I0.11) del PLC Twido. Ver figura 64

Figura 64: SECCIÓN 4

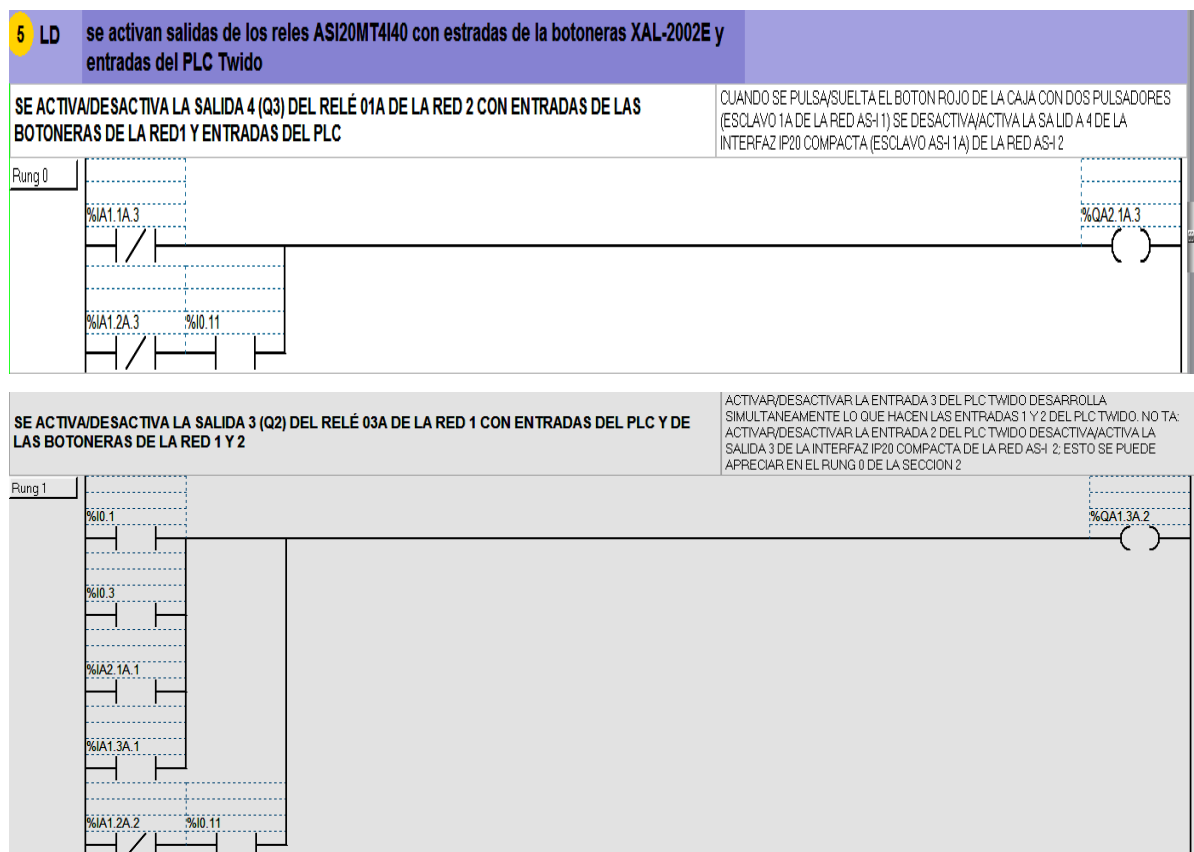


Sección 5: en esta sección se activan salidas de los relés ASI20MT4I40 con estradas de las botoneras XAL-2002E y entradas del PLC Twido.

En esta sección en el rung 0 se activa/desactiva la salida 4 (Q3) del relé ASI20MT4I40 configurado en la dirección 01A de la red AS-i 2, cuando se desactiva/activa el botón rojo (NC) de la caja con dos pulsadores “botonera XAL-2002E” configurada en la dirección 1A de la red AS-i 1 y/o cuando se desactiva/activa el botón rojo (NC) de la botonera XAL-2002E configurada en la dirección 2A de la red AS-i con la entrada 11 (I0.11) del PLC Twido activada. El rung 1 de la sección presenta la activación/desactivación de la salida 3 (Q2) del relé configurado en la dirección 03A de la red AS-i 1 cuando la entrada 1 (I0.1) del PLC se activa/desactiva y/o cuando la entrada 3 (I0.3) del PLC se activa/desactiva y/o cuando la entrada 2 (I1) del relé configurado en la dirección 01A de la red AS-2 se activa/desactiva y/o cuando la entrada 2 (I1) del relé configurado en la dirección 03A de la red AS-1 se activa/desactiva y/o cuando el botón verde (NA) de la botonera configurada en la dirección 02A de la red AS-1 se desactiva/activa con la entrada 11 (I0.11) del PLC Twido activada.

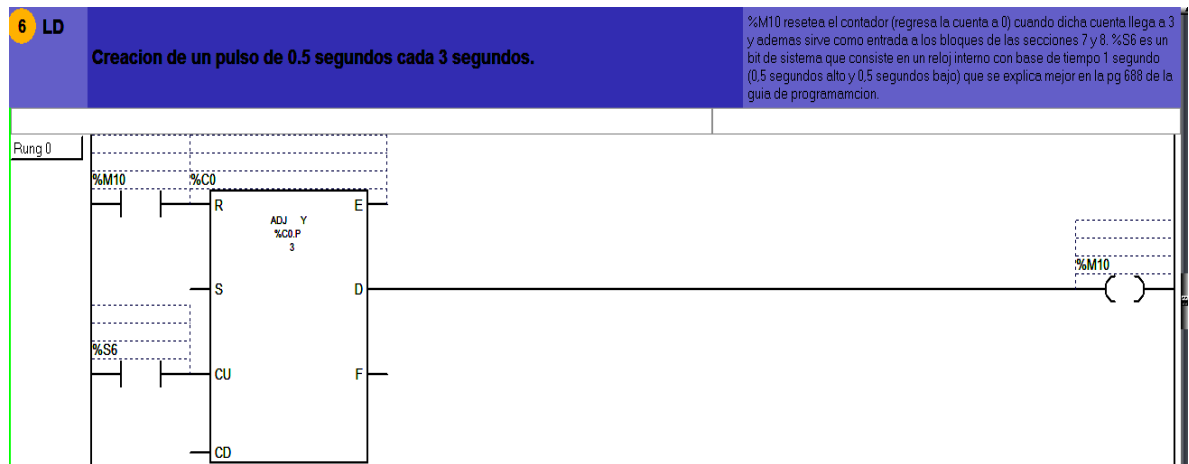
Nota: en el rung 1 cuando se activa/desactiva la entrada 3 (I0.3) del PLC Twido se activa/desactiva simultáneamente las entradas 1 (I0.1) y 2 (I0.2) del PLC Twido. Por lo tanto al activar/desactivar la entrada 2 (I0.2) del PLC twido se desactiva/activa la salida 3 (Q2) del relé configurado en la dirección 01A de la red AS-i 2; esto se puede apreciar en el rung 0 de la sección 2. Ver figura 65

Figura 65: SECCIÓN 5



1.4.3.2 PARTE 2: el propósito de esta parte del programa es presentar el bloque de función del controlador del conmutador de tambor, que servirá para gestionar las salidas AS-i, la lectura de los estados de los esclavos y la lectura del bus AS-i.

Figura 66: SECCIÓN 6



Sección 6: presenta el bit de sistema %S6, que es un reloj interno que controla los cambios de estados con una base de tiempo de 1 segundo (0,5 segundos alto y 0,5 segundos bajo). También se presenta en esta sección el bit %M10 utilizado para que recete el bloque contador (regresa la cuenta a 0) cuando dicha cuenta llega a 3 y además para que sirva como entrada a los bloques de las secciones 7 y 8. Ver figura 66

Sección 7: presenta el bloque de función del controlador del conmutador de tambor, y muestra las salidas del programa que va a manejar. La figura 67 presenta el bloque de función del controlador del conmutador de tambor.

Para ver las salidas del PLC Twido, bits de memoria y salidas AS-i, se debe dar doble clic sobre el bloque %DR0. Estas salidas el bloque las activa en cada uno de sus pasos. Ver figura 68

Figura 67: BLOQUE DE FUNCION

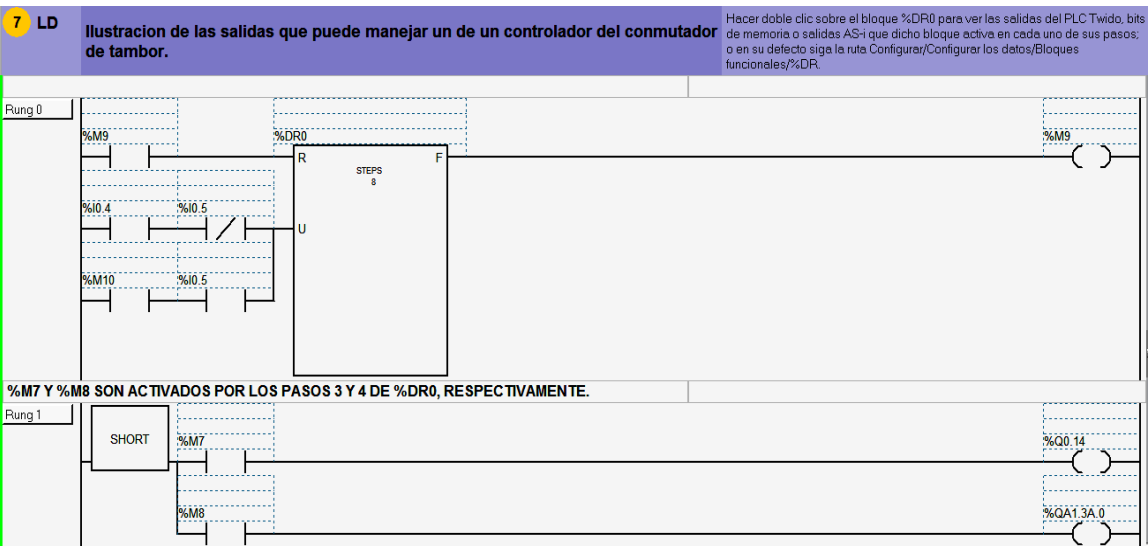


Figura 68: SALIDAS QUE ACTIVA EL BLOQUE EN CADA PASO

Todos

Uso	%DR	Símbolo
☒	%DR0	
☐	%DR1	
☐	%DR2	
☐	%DR3	
☐	%DR4	
☐	%DR5	
☐	%DR6	
☐	%DR7	

Todos

Uso	%DR	Símbolo
☒	%DR0	
☒	%DR1	
☐	%DR2	
☐	%DR3	
☐	%DR4	
☐	%DR5	
☐	%DR6	
☐	%DR7	

Aplicar Cancelar

Cantidad de pasos: 8

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Paso 0:	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 1:	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 2:	☐	☒	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 3:	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 4:	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 5:	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 6:	☐	☐	☐	☐	☐	☐	☒	☒	☐	☐	☐	☐	☐	☐	☐	☐
Paso 7:	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Salidas: %Q??.?, %QA??.? o %M?

Bit 0:	%Q0.11	Bit 6:	%QA1.3A.1	Bit 12:	
Bit 1:	%Q0.12	Bit 7:	%QA2.1A.1	Bit 13:	
Bit 2:	%Q0.13	Bit 8:		Bit 14:	
Bit 3:	%M7	Bit 9:		Bit 15:	
Bit 4:	%M8	Bit 10:			
Bit 5:	%QA2.1A.0	Bit 11:			

Aplicar Cancelar

Cantidad de pasos: 7

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Paso 0:	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 1:	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 2:	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 3:	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 4:	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 5:	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 6:	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐	☐	☐	☐	☐
Paso 7:	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Salidas: %Q??.?, %QA??.? o %M?

Bit 0:	%M0	Bit 6:		Bit 12:	
Bit 1:	%M1	Bit 7:		Bit 13:	
Bit 2:	%M2	Bit 8:		Bit 14:	
Bit 3:	%M3	Bit 9:		Bit 15:	
Bit 4:	%M4	Bit 10:			
Bit 5:	%M5	Bit 11:			

Aplicar Cancelar

1.4.3.3 TERCERA PARTE: esta parte del programa presenta la supervisión del estado de los esclavos y del bus.

Sección 8: En esta sección se gestiona la ejecución de las ordenes ASI_CMD con un bloque de función del controlador del conmutador de tambor (%DR). La ejecución de las ordenes ASI_CMD también se puede hacer a través de entradas del PLC Twido como se hizo en el primer programa del manual de AS-Interface, la diferencia entre ese primer programa y este programa, es que en este programa la supervisión realizada mediante las ordenes ASI_CMD se hace periódicamente (cada 2 segundos), es decir, haya o no actividad de las entradas. Ver figuras 69, 70, 71 Y 72.

Figura 69: ORDENES PARA SUPERVISAR EL ESTADO DE LOS ESCLAVOS Y DEL BUS DE LA RED AS-I 1

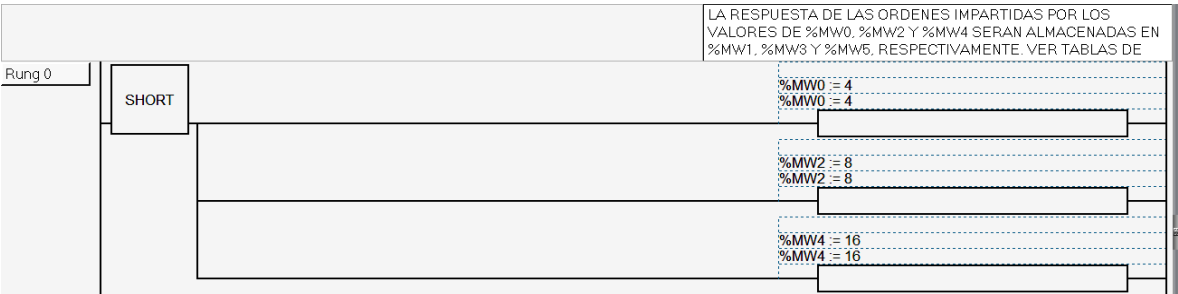


Figura 70: ORDENES PARA SUPERVISAR EL ESTADO DE LOS ESCLAVOS Y DEL BUS DE LA RED AS-I 2

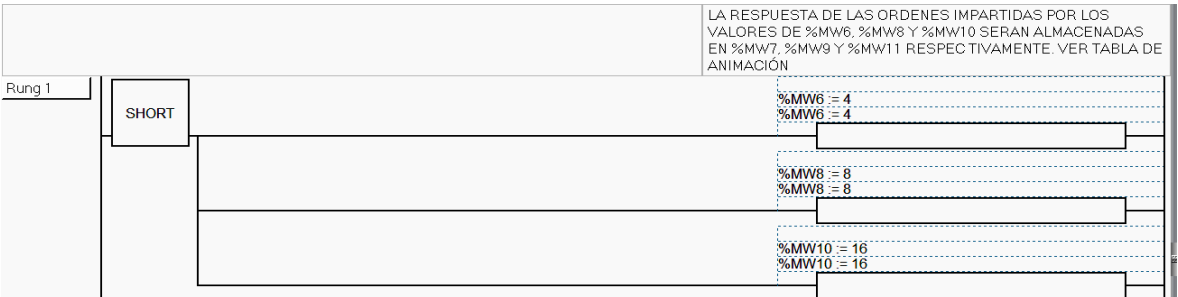


Figura 71: BLOQUE GENERADOR DE PASOS PARA LA SUPERVISIÓN

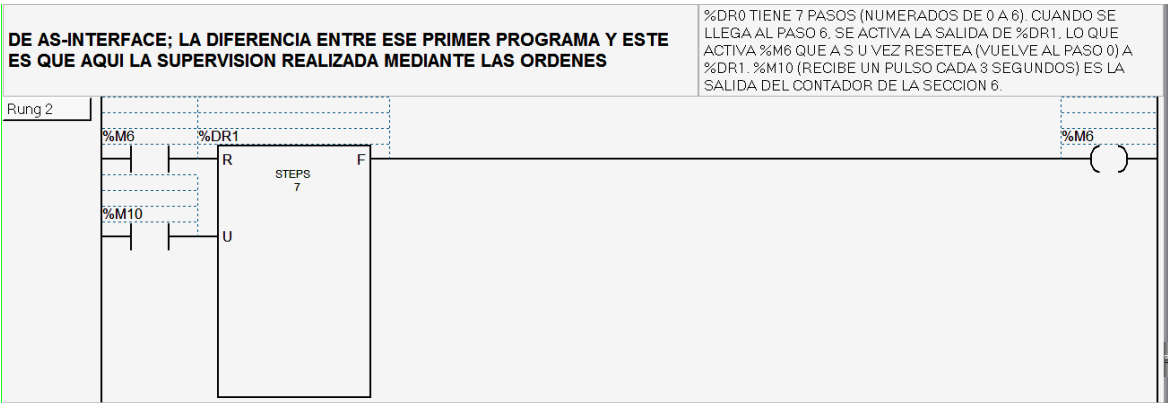
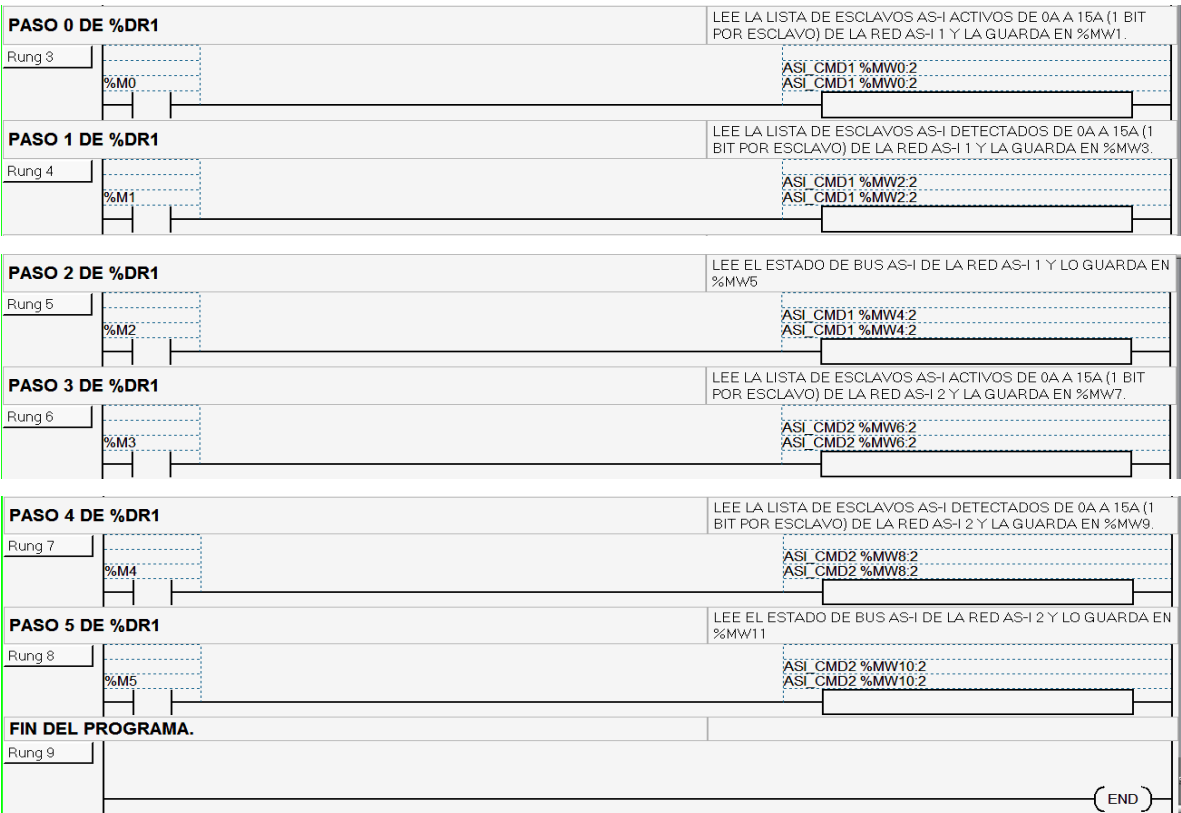


Figura 72: PASOS DE LA SUPERVISIÓN



A continuación se presenta un ejemplo con las tablas de animación donde se desconecta el esclavo con dirección 03A “relé” en la red 1 y el esclavo 02A “botonera” de la red 2. Ver figuras 73 y 74

Figura 73: TABLA DE ANIMACIÓN CON LOS ESCLAVOS 03A DE LA RED1 Y 02A DE LA RED 2

		Us	Dirección	Símbolo	Actual	Guardado	Formato
1		☐	%I0.0		F 1	0	Decimal
2		☒	%I0.1		0	0	Decimal
3		☒	%I0.2		0	0	Decimal
4		☒	%I0.3		0	0	Decimal
5		☒	%I0.4		0	0	Decimal
6		☒	%I0.5		F 1	0	Decimal
7		☒	%I0.11		0	0	Decimal
8		☒	%DR0.S		2	0	Decimal
9		☒	%DR1.S		0	0	Decimal
10		☐	%SW73		0000000000001011	0000000000000000	Binario
11		☐	%SW74		0000000000001011	0000000000000000	Binario
12		☒	%MW1		0000000000001110	0000000000000000	Binario
13		☒	%MW3		0000000000001110	0000000000000000	Binario
14		☒	%MW5		0010101100100101	0000000000000000	Binario
15		☒	%MW7		0000000000001110	0000000000000000	Binario
16		☒	%MW9		0000000000001110	0000000000000000	Binario
17		☒	%MW11		0010101100100101	0000000000000000	Binario
18		☒	%S6		0	0	Decimal
19							

En la figura 73 se puede observar en la palabra de memorias %MW1 la cual representa los esclavos activos de la red 1, que existen tres esclavos en la red 1, debido a que cada bit cuando está en uno representa la presencia de un esclavo activo con dirección A de una lista de 0 a 15. Estos esclavos tienen dirección 01A, 02A y 03A respectivamente, cada bit representa un esclavo “0000000000001110”.

La palabra de memoria % MW7 representa los esclavos de la red 2, y en ella se pueden observar la presencia de los esclavos activos 01A y 02A. La figura 74 representa la ausencia de los esclavos 03A de la red 1 y 02A de la red 2, debido a que el bit que representaba la activación de estos esclavos esta en 0 “esclavo desactivado”

Figura 74

		Us	Dirección	Símbolo	Actual	Guardado	Formato
1		☐	%I0.0		F 1	0	Decimal
2		☒	%I0.1		0	0	Decimal
3		☒	%I0.2		0	0	Decimal
4		☒	%I0.3		0	0	Decimal
5		☒	%I0.4		0	0	Decimal
6		☒	%I0.5		F 1	0	Decimal
7		☒	%I0.11		0	0	Decimal
8		☒	%DR0.S		0	0	Decimal
9		☒	%DR1.S		5	0	Decimal
10		☐	%SW73		0000000000001010	0000000000000000	Binario
11		☐	%SW74		0000000000001010	0000000000000000	Binario
12		☒	%MW1		0000000000000110	0000000000000000	Binario
13		☒	%MW3		0000000000000110	0000000000000000	Binario
14		☒	%MW5		0010101100101100	0000000000000000	Binario
15		☒	%MW7		0000000000000010	0000000000000000	Binario
16		☒	%MW9		0000000000000010	0000000000000000	Binario
17		☒	%MW11		0010101100101100	0000000000000000	Binario
18		☒	%S6		1	0	Decimal
19							

ANEXO 2: MANUAL MODBUS

CONTENIDO

	Pág.
2 PASOS PARA CREAR UNA APLICACIÓN MODBUS CON TWIDOSUITE	283
2.1 CREE UN PROYECTO NUEVO DE LA SIGUIENTE FORMA:	283
2.1.2 CONSTRUYA LA RED MODBUS:	285
2.1.3 CONFIGURACIÓN DEL PUERTO 1 MODBUS.	285
2.1.4 CONFIGURACIÓN DE LOS ESCLAVOS.	286
2.1.5 CONFIGURACION DE LA RED MODBUS	288
2.2 PRIMER PROGAMA: COMUNICACIÓN ENTRE UN MAESTRO TWIDO Y UN ESCLAVO TWIDO.	290
2.2.1 RED MODBUS	291
2.2.2 PROGRAMA EN LADDER	291
2.2.2.1 PRIMERA PARTE	291
2.2.2.2 SEGUNDA PARTE:	295
2.2.2.3 TERCERA PARTE: GESTION DE LA COMUNICACIÓN:	298
2.3 SEGUNDO PROGAMA: COMUNICACIÓN ENTRE UN MAESTRO TWIDO, UN ESCLAVO TWIDO, UN ESCLAVO M340, Y UN ESCLAVO ZELIO.	299
2.3.1 PRIMERA PARTE: COMUNICACIÓN ENTRE EL MAESTRO PLC TWIDO CON EL ESCLAVO 1 PLC TWIDO	300
2.3.1.1 PROGRAMA DEL MAESTRO	300
2.3.1.2 PROGRAMA EN LADDER	302
2.3.2 SEGUNDA PARTE: COMUNICACIÓN ENTRE EL PLC TWIDO MAESTRO Y EL PLC M340 ESCLAVO	313
2.3.2.1 PROGRAMA MAESTRO	313
2.3.2.2 PROGRAMA DEL ESCLAVO 2	314
2.3.2.3 PROGRAMA LADDER	314
2.3.3 TERCERA PARTE: COMUNICACIÓN ENTRE EL MAESTRO PLC TWIDO CON EL ESCLAVO 3 PLC ZELIO	322
2.3.3.1 CONFIGURACIÓN DEL ESCLAVO ZELIO Y DEL MAESTRO TWIDO	323
2.3.3.2 ENTRADAS Y SALIDAS DEL MAESTRO TWIDO Y DEL ESCLAVO ZELIO	324
2.3.3.3 PROGRAMA LADDER DEL MAESTRO TWIDO	325

LISTA DE FIGURAS

	Pág.
FIGURA 1: CREAR UN PROYECTO	283
FIGURA 2: TWIDO TWDLCD40DRF	284
FIGURA 3: PUERTO 1	285
FIGURA 4: CONFIGURACIÓN DEL PUERTO 1	286
FIGURA 5: CATALOGO DE LOS ESCLAVOS MODBUS	287
FIGURA 6: ESCLAVOS MODBUS DE LA E3T	287
FIGURA 7 CONFIGURACION DE UN ESCLAVO	288
FIGURA 8: RED MODBUS CON DOS ESCLAVOS	289
FIGURA 9: PARAMETROS MODBUS	290
FIGURA 10: RED MODBUS CON UN MAESTRO Y UN ECLAVO, TWIDO	291
FIGURA 11: PROGRAMA DEL MAESTRO	292
FIGURA 12: PROGRAMA DEL ESCLAVO:	293
FIGURA 13: TABLA DE ESCRITURA DE DOS PALABRAS	293
FIGURA 14: PROGRAMA DEL ESCLAVO	295
FIGURA 15: PROGRAMA DEL MAESTRO	296
FIGURA 16: TABLA DE LECTURA DE 2 PALABRAS	296
FIGURA 17: GESTION DE LA COMUNICACION	298
FIGURA 18: RED MODBUS	300
FIGURA 19: CONFIGURACIÓN PUERTO PLC TWIDO MAESTRO	301
FIGURA 20: CONFIGURACION DEL ESCLAVO 1	301
FIGURA 21: LECTURA DE UN BIT	303
FIGURA 22: PROGRAMA DEL ESCLAVO 1, SECCION 2:	304
FIGURA 23: PROGRAMA DEL MAESTRO, SECCION 2:	305
FIGURA 24: ESCRITURA DE UN BIT	305
FIGURA 25: PROGRAMA DEL MAESTRO, SECCION 4	306
FIGURA 26: PROGRAMA DEL ESCLAVO 1, SECCION 4	306
FIGURA 27: PROGRAMA DEL ESCLAVO, SECCION 5	307
FIGURA 28: PROGRAMA DEL MAESTRO, SECCION 5	308

FIGURA 29: PROGRAMA DEL MAESTRO, SECCION 6	308
FIGURA 30: BLOQUE DE FUNCION DEL REGISTRO DE BITS DE DESPLAZAMIENTO	309
FIGURA 31: FIJACION A 1 DEL BIT MENOS SIGNIFICATIVO DEL REGISTRO DE DESPLAZAMIENTO	309
FIGURA 32: GESTION DE LOS 5 BITS MENOS SIGNIFICATIVOS DEL REGISTRO DE DESPLAZAMIENTO %SBR0	310
FIGURA 33: BITS (6 A 10) DEL REGISTRO DE DESPLAZAMIENTO %SBR0	310
FIGURA 34: BITS (10 A 15) DEL REGISTRO DE DESPLAZAMIENTO %SBR0	311
FIGURA 35: BLOQUE CONTADOR PROGRESIVO	311
FIGURA 36: UTILIZACION DE LA INSTRUCCIÓN EXCHx	312
FIGURA 37: CONFIGURACION DEL ESCLAVO	313
FIGURA 38: CONFIGURACIÓN DEL ESCLAVO 2	314
FIGURA 39 LECTURA DE DOS BITS	316
FIGURA 40: ACTIVACIÓN DE %M0 Y %M1	317
FIGURA 41: ESCRITURA DEL VALOR DE LOS BITS 8 Y 9 DE %MW46	318
FIGURA 42: ESCRITURA EN LOS BITS %M2 Y M3	319
FIGURA 43: FUNCIÓN C_RD1W DE LA MACRO COM 2	319
FIGURA 44: LECTURA DE LOS BITS 0 Y 1 DE LA PALABRA %MW0	320
FIGURA 45: FUNCIÓN C_WRNW DE LA MACRO COM 2	320
FIGURA 46: ESCRITURA EN LOS BITS 0 Y 1 DE LA PALABRA %MW1	321
FIGURA 47: GESTIÓN DE LA COMUNICACIÓN, SEGUNDA PARTE	322
FIGURA 48: CONFIGURACIÓN DEL MAESTRO Y EL ESCLAVO	324
FIGURA 49: LECTURA DE LA PALABRA DE MEMORIA %MW20	325
FIGURA 50: VALOR LEÍDO DE LA PALABRA %MW20.	326
FIGURA 51: CODIGO DE SOLICITUD 10 H PARA ESCRIBIR EN %MW17	327
FIGURA 52: VALORES DE %MW150 QUE SE VAN A ESCRIBIR	327
FIGURA 53: ESCRITURA DE LOS VALORES DE %MW140 EN %MW16	328
FIGURA 54: PROGRAMA DEL ESCLAVO ZELIO	329
FIGURA 55: CONFIGURACIÓN DEL MAESTRO Y EL ESCLAVO	330
FIGURA 56: ESCLAVO ALTIVAR CON MACROS DRIVE	333

LISTA DE TABLAS

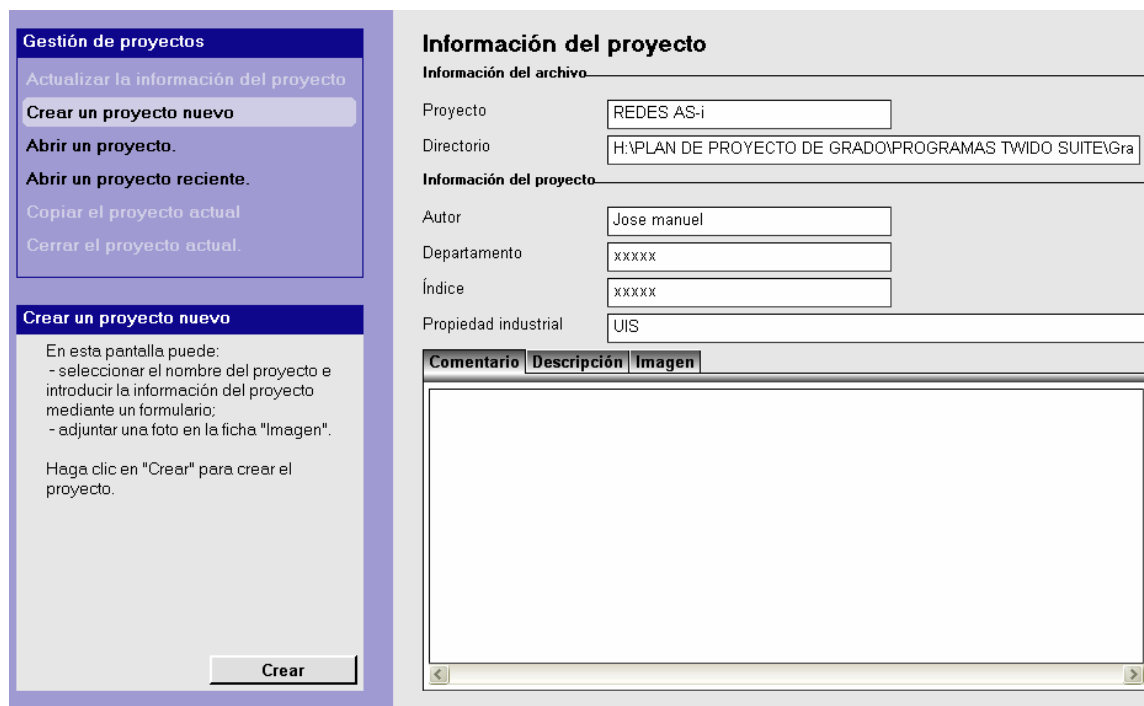
	Pág.
Tabla 1: VARIABLES DEL MAESTRO TWIDO Y DEL ESCLAVO TWIDO	302
Tabla 2: ENTRADAS DEL MAESTRO TWIDO QUE ACTIVAN/DESACTIVAN SALIDAS DEL ESCLAVO M340	315
Tabla 3: ENTRADAS DEL ESCLAVO M340 QUE ACTIVAN/DESACTIVAN SALIDAS DEL MAESTRO TWIDO	316
Tabla 4: ENTRADAS DEL MAESTRO TWIDO QUE CONTROLAN SALIDAS DEL ESCLAVO 3 ZELIO	324
Tabla 5: ENTRADAS DEL ESCLAVO ZELIO QUE CONTROLAN SALIDAS DEL MAESTRO TWIDO	325
Tabla 6: ENTRADAS DEL MAESTRO TWIDO QUE CONTROLAN ACCIONES DEL ESCLAVO 4 ALTIVAR 31	331
Tabla 7: FUNCIÓN D_SELECT_SPEED, MODO DE COSIGNA	333

2 PASOS PARA CREAR UNA APLICACIÓN MODBUS CON TWIDOSUITE

2.1 CREE UN PROYECTO NUEVO DE LA SIGUIENTE FORMA:

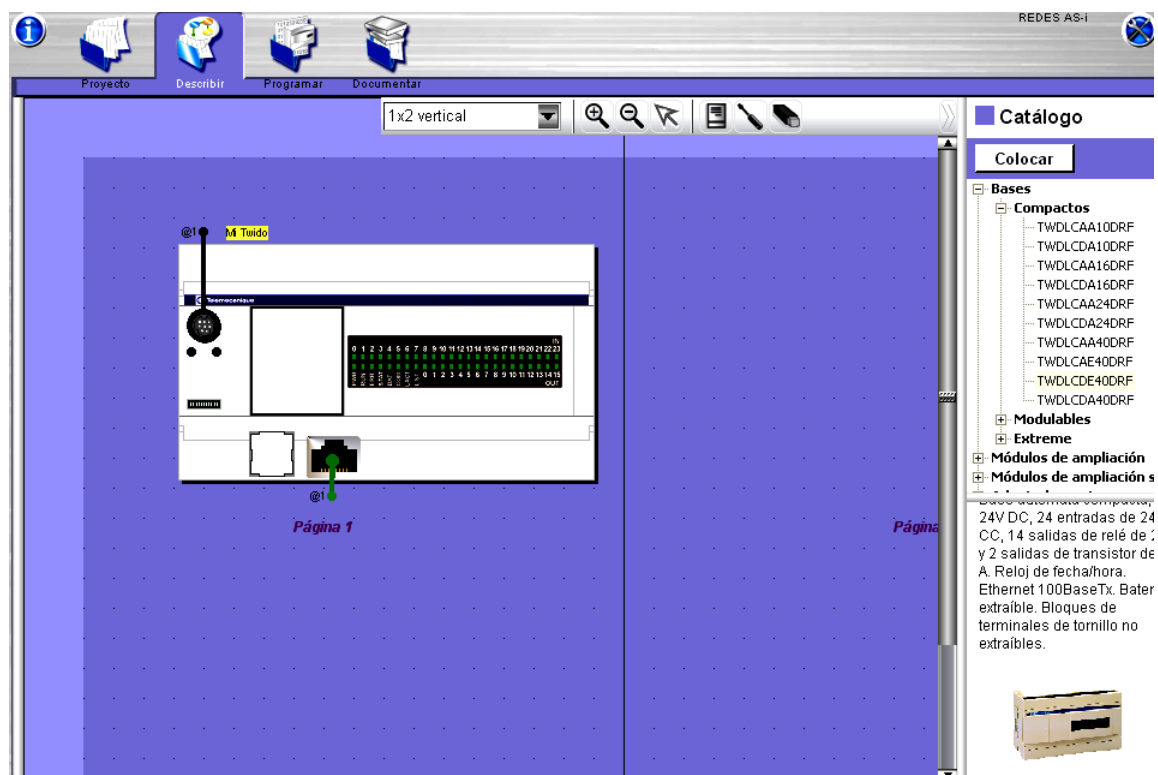
En la parte izquierda de su pantalla elija crear un proyecto nuevo de las siguientes opciones, crear un proyecto nuevo, abrir un proyecto, abrir un proyecto reciente. Cuando elije crear un proyecto nuevo, aparece a la derecha de estas opciones la información del proyecto, con los siguientes ítems: proyecto, directorio (elige donde quiere guardar el proyecto, como mis documentos, una memoria USB, etc.), autor, departamento, índice y propiedad industrial. Al concluir estas tareas haga clic en crear. Ver figura 1

FIGURA 1: CREAR UN PROYECTO



Ya creado el proyecto vaya a describir. Elimine el PLC que aparece en la pantalla y dentro de la opción catalogo elija el PLC TWIDO TWDLCADE40DRF, de la siguiente manera. Elija bases, en bases elija compactas y en compactas elija el autómata ya mencionado. Seguidamente ubique el autómata en la pantalla. Ver figura 2

FIGURA 2: TWIDO TWDLCADE40DRF



Nota: modbus TCP-IP no tiene módulos de ampliación debido a que se configura por el puerto 3 del autómata TWDLCAE40DRF.

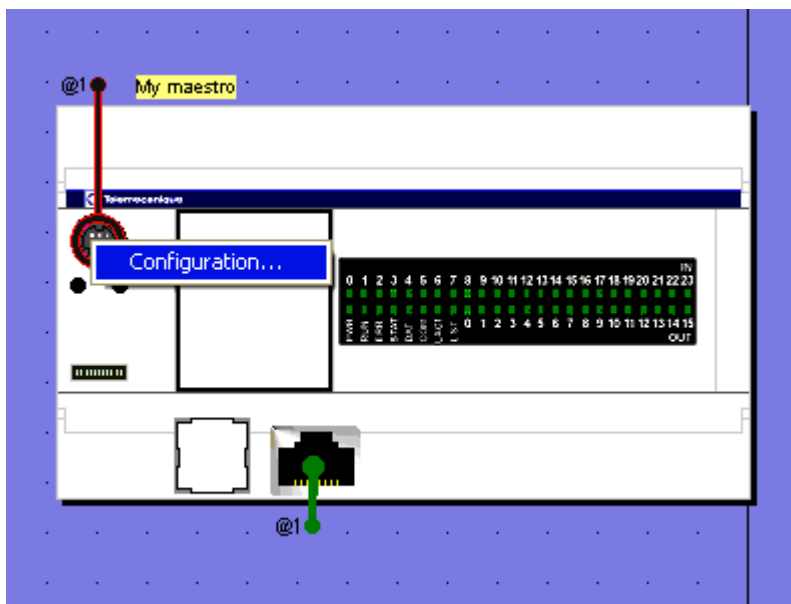
2.1.2 CONSTRUYA LA RED MODBUS:

Para construir la red modbus se necesita seguir tres pasos:

- ❖ Configuración de el puerto 1 modbus
- ❖ Configuración de los esclavos
- ❖ Configuración de la red modbus

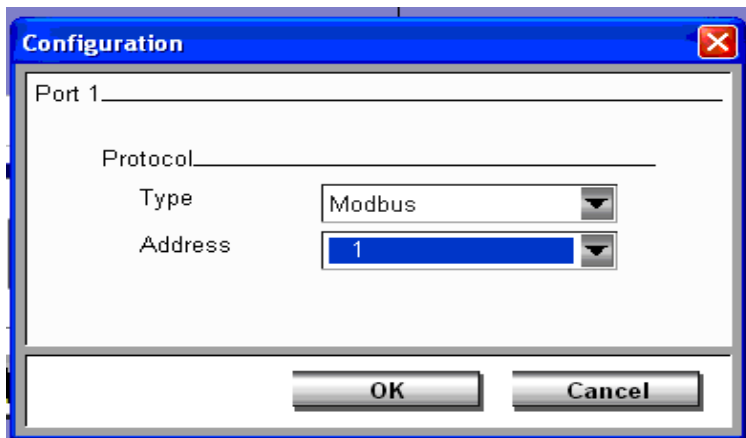
2.1.3 CONFIGURACIÓN DEL PUERTO 1 MODBUS: Se realiza de la siguiente forma: en el PLC se ubica el puerto 1, en el se da clic derecho y se elige configurar. Ver figura 3

FIGURA 3: PUERTO 1



Cuando se elije configurar se despliega la tabla de la figura 4, se elije modbus en Type y se asigna la dirección que se quiere utilizar para el maestro modbus en Address. Después OK

FIGURA 4: CONFIGURACIÓN DEL PUERTO 1



2.1.4 CONFIGURACIÓN DE LOS ESCLAVOS: Para configurar los esclavos es necesario ir a catalogo, en catalogo elegir elemento de red (network) y en elementos de red elegir elementos modbus (modbus elementos), en donde se encontraran todos los esclavos. Ver figura 5

En la figura 5 se elije el tipo de esclavo que se quiere configurar en la red. Ejemplo si se quiere configurar una red modbus con un esclavo PLC Twido y un esclavo variador de velocidad Altivar 31. Ver figura 6.

Ya elegido el tipo de esclavo, se procede a configurarlo de la siguiente manera: se da clic derecho en el esclavo y se elije configurar. Se desplegará la siguiente tabla. Ver figura 7

FIGURA 5: CATALOGO DE LOS ESCLAVOS MODBUS

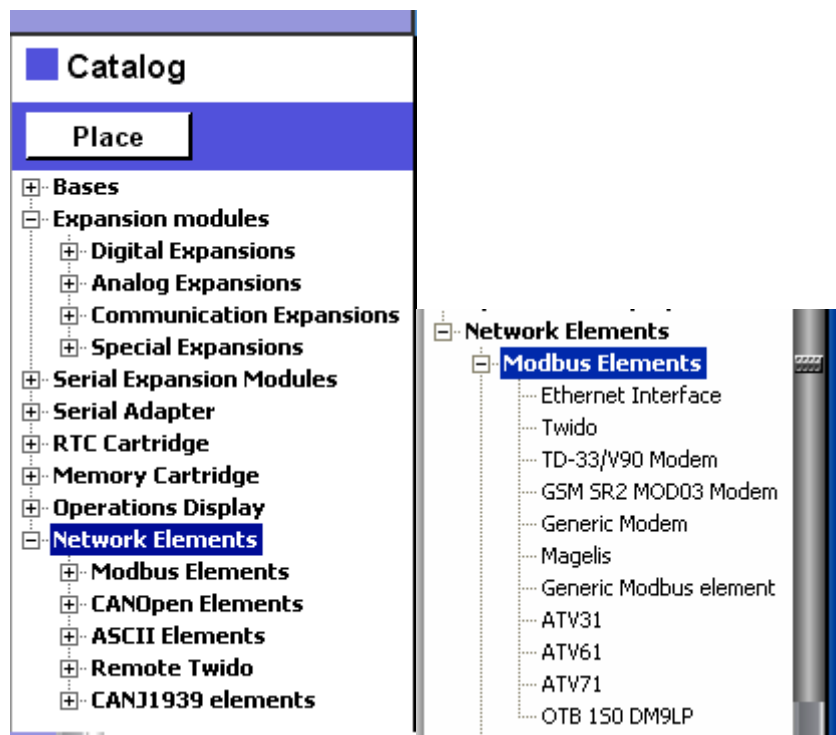


FIGURA 6: ESCLAVOS MODBUS DE LA E3T

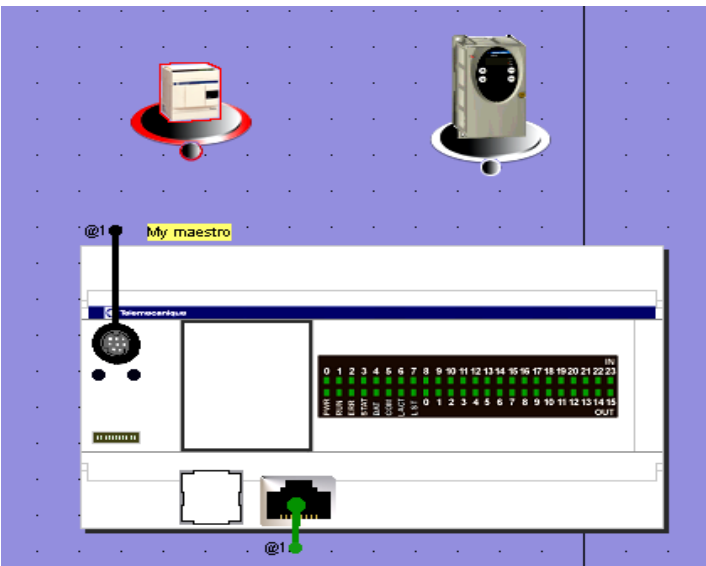
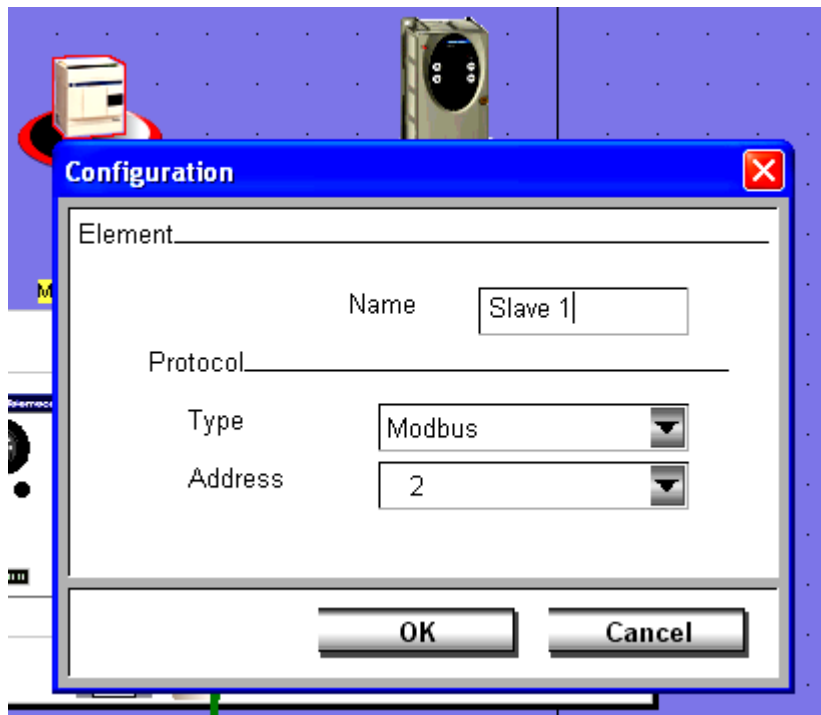


FIGURA 7 CONFIGURACION DE UN ESCLAVO



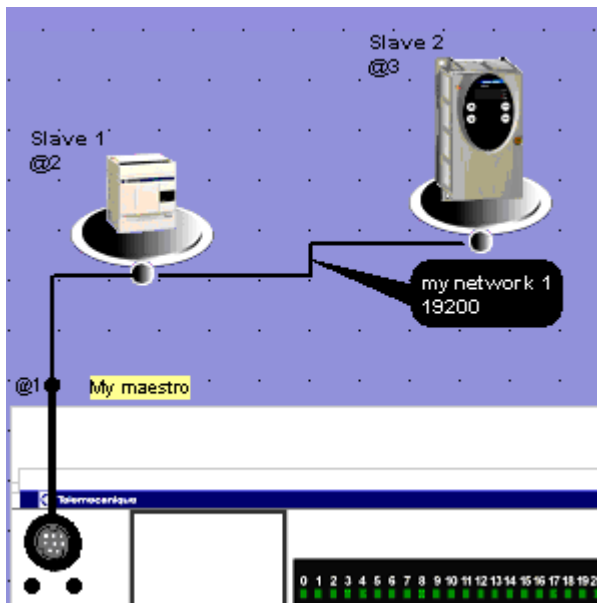
En la tabla de la figura 7 se asigna la dirección del esclavo, ejemplo 2, se asigna modbus en Type y se da el nombre del esclavo. Este procedimiento se realiza con todos los esclavos que conforma la red. Es importante señalar que cada elemento de red (esclavos, maestro) debe tener una dirección diferente.

2.1.5 CONFIGURACION DE LA RED MODBUS: Para configurar la red modbus es necesario primero construirla en el software twidosuite, lo cual se realiza colocando la flecha que se puede manipular con el mouse encima del punto de la plataforma que sostiene al esclavo; cuando se realiza esta acción aparece un signo en forma de asterisco, el cual indica que el esclavo está listo para unirse a

través de la red con los otros elementos de la red, se une los puntos tanto el del maestro como los puntos del esclavo y se obtiene la red modbus. Ver figura 8.

La configuración de la red se elabora de la siguiente manera, al hacer clic sobre la red modbus y después clic sobre configurar, se despliega una tabla la cual contiene los elementos de la red a configurar. Ver figura 9

FIGURA 8: RED MODBUS CON DOS ESCLAVOS



La tabla de la figura 9 contiene los parámetros propios del protocolo modbus, como el modo en el cual se quiere enviar los bits de datos (RTU o ASCII), velocidad de la red (19200), parida, bit de parada, tiempo de respuesta y tiempo de espera entre tramas. El estudiante debe tener los conocimientos necesarios para la correcta configuración de esta tabla.

Nota: el protocolo modbus maneja el modelo de comunicación Maestro esclavo

FIGURA 9: PARAMETROS MODBUS

Configuración

Red

Tipo: Modbus Nombre: mi red 1

Parámetros

Caudal: 19200 Bit de parada: 1

Bits de datos: 8 (RTU) Tiempo de respuesta: 10 x 100 ms

Paridad: par Tiempo de espera entre tramas: 10 ms

Aceptar Cancelar

El siguiente paso es elaborar un programa en el PLC Twido que sirva para tener acceso a las variables de los dispositivos que conforman la red modbus.

El programa del PLC Twido se puede elaborar en varios lenguajes de programación pero en este manual se utilizará el lenguaje ladder.

A continuación se presentan dos programas en este lenguaje.

2.2 PRIMER PROGRAMA: COMUNICACIÓN ENTRE UN MAESTRO TWIDO Y UN ESCLAVO TWIDO.

Este programa activa la salida número 1 (Q0.1) del esclavo PLC twido al activar la entrada 1 (I0.1) del maestro PLC Twido y la salida 1 (Q0.1) del maestro al activar la entrada número 1 (I0.1) del esclavo, a demás acciona la salida 10 (Q0.10) del esclavo cuando el potenciómetro 0 (IW0.0.0) del maestro supera la mitad de su

valor y la salida 10 (Q0.10) del maestro cuando el potenciómetro 0 (IW0.0.0) del esclavo supera la mitad de su valor.

La programación del programa antes mencionado se divide en dos, un programa que se transfiere al maestro y otro programa que se le transfiere al esclavo. A continuación se presenta este programa.

2.2.1 RED MODBUS

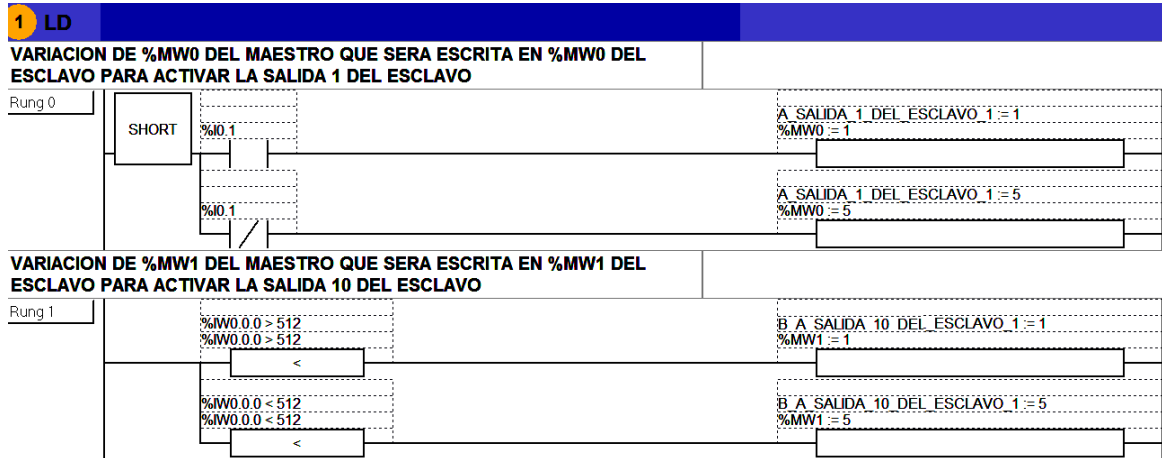
FIGURA 10: RED MODBUS CON UN MAESTRO Y UN ECLAVO, TWIDO



2.2.2 PROGRAMA EN LADDER

2.2.2.1 PRIMERA PARTE: esta primera parte presenta las variables y los elementos de configuración necesarios empleados para que al activar/desactivar entradas del maestro Twido se activa/desactiva salidas del esclavo Twido.

FIGURA 11: PROGRAMA DEL MAESTRO



Las figuras 11 y 12 presentan como la entrada I0.1 del maestro utiliza la palabra de memoria %MW0 para después activar la salida Q0.1 en el programa del esclavo y como el potenciómetro número (%IW0.0.0) cero del maestro cuando su valor es mayor de 512 utiliza la palabra de memoria %MW1 para activar la salida número 10 (Q0.10) del esclavo. Para que las palabras de memoria %MW0 y %MW1 cumplan con su objetivo de activar salidas en el esclavo es necesario utilizar la estructura de la tabla de escritura de N palabras, presentada en el documento “redes de comunicación industrial. Prácticas de laboratorio” para construir en el programa ladder una tabla de escritura de dos palabras que será utilizada por la instrucción EXCHx. Se recuerda que en el programa del maestro se desarrollan las tablas de escrituras y de lecturas utilizadas para la comunicación en modbus y que el maestro escribe y lee en el esclavo. Cuando el maestro activa salidas en el esclavo se utiliza la tabla de escritura, si el esclavo activa salidas en el maestro se utiliza tabla de lectura. La figura 13 presenta la tabla de escritura necesaria para la gestionar la comunicación de esta primera parte del programa.

FIGURA 12: PROGRAMA DEL ESCLAVO:

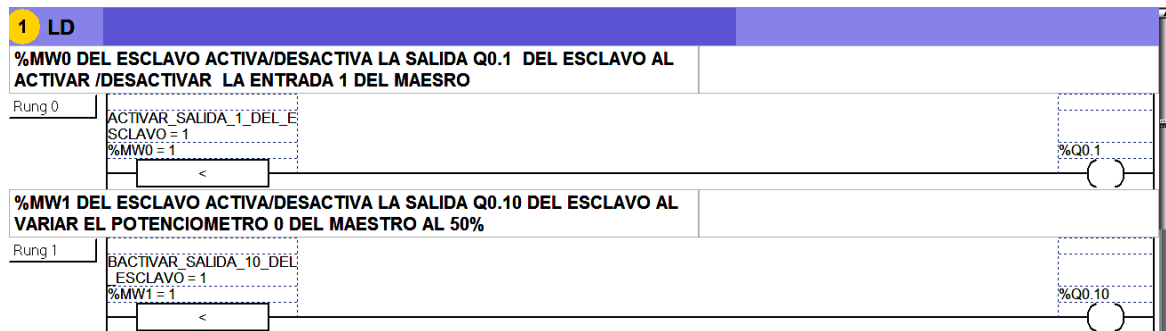
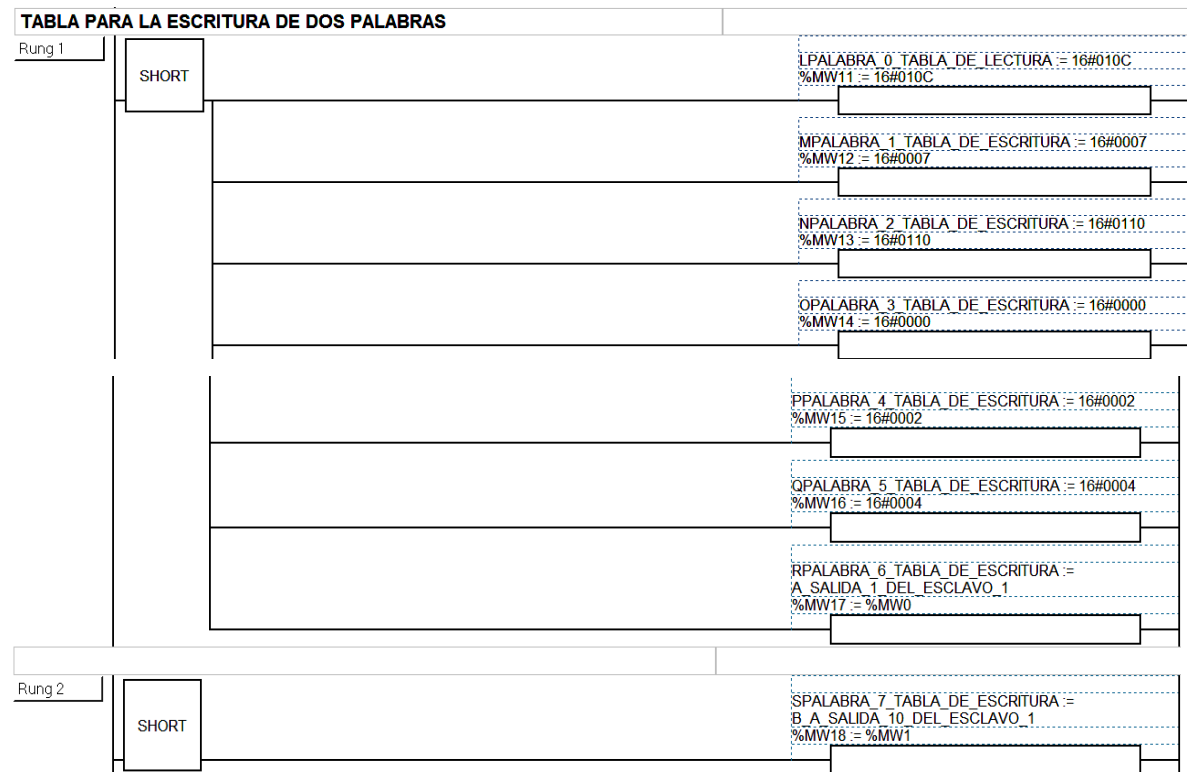


FIGURA 13: TABLA DE ESCRITURA DE DOS PALABRAS



La tabla de escritura de dos palabras de la figura 13 está conformada por las siguientes palabras de memoria.

Palabra 0: %MW11= 16#010C. Esta palabra corresponde a la subtabla de control dentro de la tabla de escritura de N palabras y es utilizada para controlar la subtabla de envío y recepción dentro de la tabla de escritura. Esta palabra está compuesta por dos bytes, el byte más significativo corresponde al código de emisión/recepción = 01 y el byte menos significativo corresponde a la longitud de la tabla de envío y es igual a $8 + (2*N)$ donde N es el numero de palabras que se quieren escribir. En este caso este byte es igual 0C en hexadecimal

Palabra 1: %MW12 = 16#0007. Esta palabra corresponde a la subtabla de control. La palabra tiene dos bytes, el más significativo se utiliza para el offset de recepción = 00 y el menos significativo para el offset de emisión =07

Palabra 2: %MW13 = 16#0110. Esta palabra corresponde a la subtabla de envío y tiene dos bytes, el más significativo se utiliza para la dirección del esclavo en este caso 01 y el byte menos significativo para el código de solicitud de la tabla de escritura. Para la tabla de escritura es igual a 10 en hexadecimal.

Palabra 3: %MW14 = 16#0000. Esta palabra corresponde a la subtabla de envío y contiene la dirección de la primera palabra de memoria correspondiente al programa del esclavo en la cual se va a escribir. Para este ejemplo 16#0000 “%MW0”

Palabra 4: %MW15 = 16#0002. Corresponde a la subtabla de envío y contiene el número de palabras que se van a escribir, para este caso dos palabras 16#0002 “dos palabras”

Palabra 5: %MW16 = 16#0004. Corresponde a la subtabla de envío y contiene en el byte más significativo el byte no enviado por efecto del offset = 00 y en el byte menos significativo el numero de bytes que se van a escribir. Para este caso 04 “4 bytes = 2 palabras”.

Palabra 6: %MW17 = %MW0. Corresponde a la subtabla de envío de la tabla de escritura y contiene el valor de la primera palabra que se va a escribir. En este ejemplo se va a escribir el valor 1 de la palabra del maestro %MW0 en la primera palabra “%MW0” del esclavo. **Nota:** en este caso, la palabra de memoria %MW17 trae el valor de la palabra de memoria %MW0 del maestro, cuando %MW17 se hace igual a %MW0 (% MW17 = %MW0) se escribe este valor en la palabra de memoria %MW0 del esclavo

Palabra 7: % MW18 = %MW1 Corresponde a la subtabla de envío y contiene el segundo valor que se va a escribir.

Los valores de las palabras de memoria 6 y 7 de la tabla de escritura de dos palabras son los que se van a escribir en las palabras %MW0 y %MW1 del esclavo

2.2.2.2 SEGUNDA PARTE: presenta la tabla de lectura de N palabras empleada para leer palabras de memoria del esclavo. Las palabras leídas del esclavo sirven para activa/desactiva salidas del maestro twido.

FIGURA 14: PROGRAMA DEL ESCLAVO

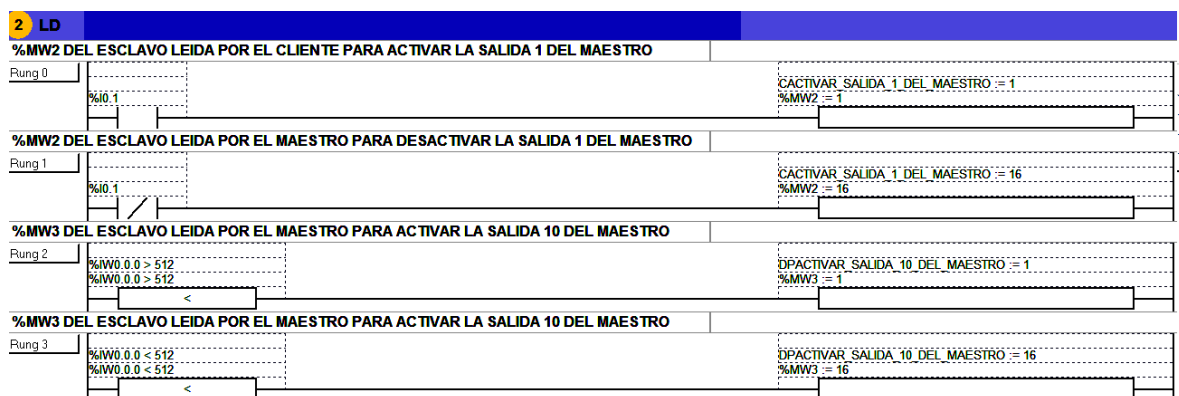
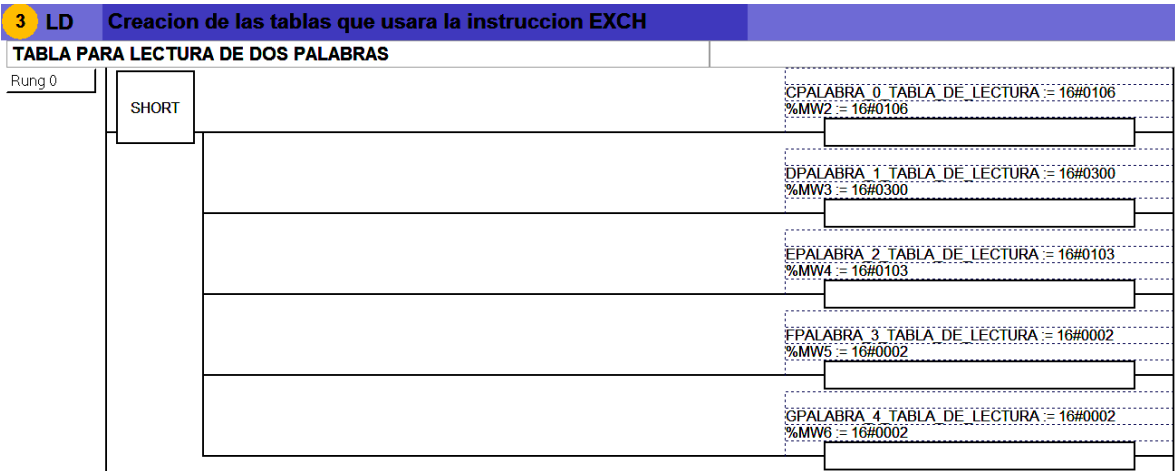


FIGURA 15: PROGRAMA DEL MAESTRO



En esta parte del programa el maestro lee del esclavo. Se observa que en el programa se activa las salidas Q0.1 y Q0.10 cuando se activa las palabras de memoria %MW9 y %MW10 de la tabla de lectura. Enseguida se describe la tabla de lectura de N palabras para leer dos palabras.

FIGURA 16: TABLA DE LECTURA DE 2 PALABRAS



Descripción de la tabla de lectura para leer dos palabras

Palabra 0: %MW2 = 16#0106. Esta palabra corresponde a la subtabla de control utilizada para controlar la tabla de envío y recepción dentro de la tabla de lectura. Esta palabra está compuesta por dos bytes, el byte más significativo corresponde al código de emisión/recepción = 01 y el byte menos significativo corresponde a la longitud de la tabla de envío y es igual a 06

Palabra 1: %MW3 = 16#0300. Corresponde a la subtabla de control y contiene el offset de recepción en el byte más significativo = 03 y el offset de envío en el byte menos significativo = 00.

Palabra 2: %MW4 = 16#0103 corresponde a la subtabla de envío en la tabla de lectura y contiene la dirección del esclavo y el código de solicitud de la tabla de lectura respectivamente.

Palabra 3: %MW5 = 16#0002. Corresponde a la subtabla de envío y contiene la dirección de la primera palabra que se va a leer para este caso 0002.

Palabra 4: %MW6 = 16#0002. Corresponde a la subtabla de envío y contiene el número de palabras de lectura. En este ejercicio se utilizaron 2

Nota: la subtabla de recepción de la tabla de lectura la genera el programa. Ver Pg 142 de la guía de programación. En esta subtabla responde el programa, por lo tanto se asignan palabras para que el programa responda en ellas. Estas palabras deben ser continuas después de la última palabra de la tabla de envío, ósea para este caso debe ser **%MW7, %MW8, %MW9 y %MW10**. Las palabras **%MW9 y %MW10** contiene el valor que va leer el maestro. Cuando ellas sean igual a uno el maestro acciona las salidas **Q0.1 y Q0.10** respetivamente. Ver figura 15

2.2.2.3 TERCERA PARTE: GESTION DE LA COMUNICACIÓN: La gestión de la comunicación se hace a través de la instrucción EXCHx y el bloque de función %MSGx. Ver guía de programación twidosuite Pg 132.

La instrucción EXCHx permite al autómatas Twido enviar o recibir información de los dispositivos Modbus para lo cual el usuario define una tabla de palabras (%MWi: L) que contiene información de control y los datos que se van a enviar o recibir. Los intercambios de mensajes se realizan mediante la instrucción EXCHx. El PLC Twido debe finalizar el intercambio de la primera instrucción EXCHx antes de ejecutarse una segunda, para ello se utiliza el bit de entrada %MSG1.D que indica cuando esta en 1 que el mensaje de la instrucción EXCHx a finalizado. Si %MSG1.D está en 0 indica que el mensaje no ha finalizado.

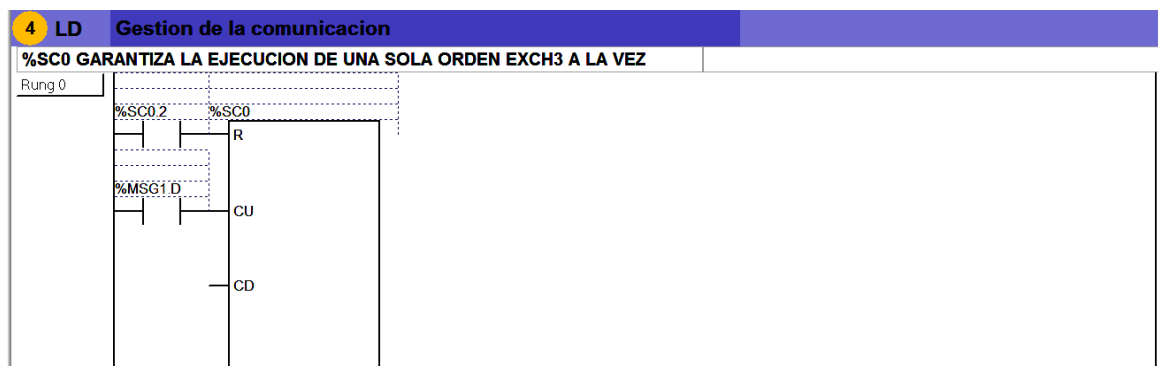
La instrucción EXCHx se utiliza con la tabla de palabras seleccionadas de la siguiente forma:

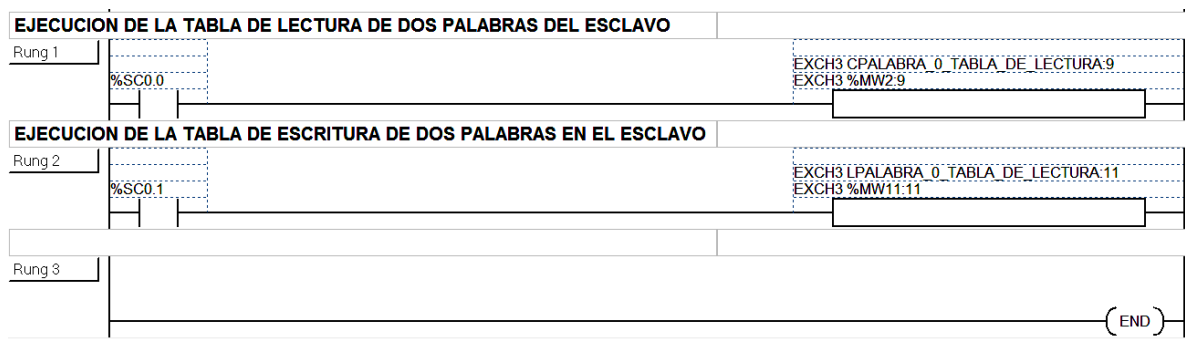
[EXCHx %MWi:L]

Donde: x = número de puerto (1 ó 2). Para este ejemplo.

L = número de palabras de la tabla de palabras de control, de envío y de recepción. Para este ejemplo L es igual a 9 para la tabla de lectura, comenzando con la palabra %MW2 y para el caso de la tabla de escritura L es igual a 11 comenzando en la palabra %MW11. Ver figura 17

FIGURA 17: GESTION DE LA COMUNICACION





La figura 17 presenta en el rung 0, un contador de pasos que tiene en su entrada el bit %MSG1.D, el cual sirve para indicar cuando a finalizado una instrucción EXCHx y por este motivo el contador pase al siguiente paso. Este contador de pasos se resetea al tercer paso o sea en %SCO.2 debido a que solo tiene dos instrucciones EXCHx que son la que facilita el intercambio de mensajes de las tablas de lectura y escritura respectivamente.

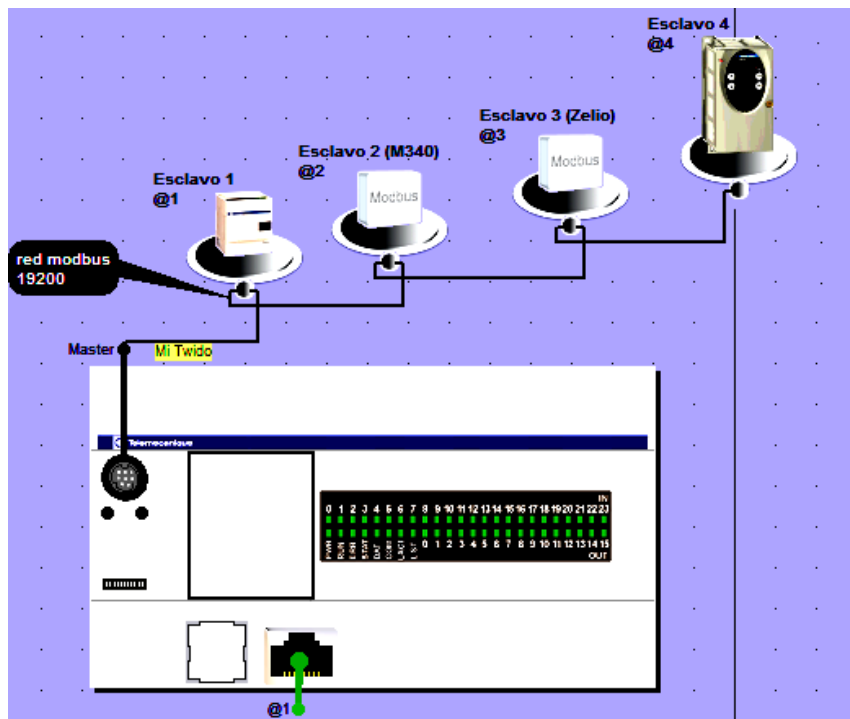
Los rungs 1,2 y 3 presentan los pasos para cada instrucción EXCHx y la finalización del programa.

Elaborado el programa lo transferimos al autómeta Twido, esto se realiza dando clic en depurar y clic en OK .Después transferir programa al PLC y clic en OK Por ultimo seleccionamos RUN

2.3 SEGUNDO PROGAMA: COMUNICACIÓN ENTRE UN MAESTRO TWIDO, UN ESCLAVO TWIDO, UN ESCLAVO M340, Y UN ESCLAVO ZELIO.

La figura 18 presenta la conexión de un maestro Twido, un esclavo twido, un esclavo M340, un esclavo zelio. Vista desde el programa del Maestro

FIGURA 18: RED MODBUS



2.3.1 PRIMERA PARTE: COMUNICACIÓN ENTRE EL MAESTRO PLC TWIDO CON EL ESCLAVO 1 PLC TWIDO

2.3.1.1 PROGRAMA DEL MAESTRO:

La figura 19 muestra la configuración del puerto del Twido maestro y la figura 20 presenta como se configura el protocolo modbus para el esclavo 1 en el programa del maestro. **Nota:** la configuración del protocolo modbus en el programa del esclavo debe tener la misma configuración que el programa del maestro, pero con la diferencia que en la ventana **Puerto 1** se configura con los parámetros del esclavo y la ventana **Elemento** se configura con los parámetros del maestro

FIGURA 19: CONFIGURACIÓN PUERTO PLC TWIDO MAESTRO

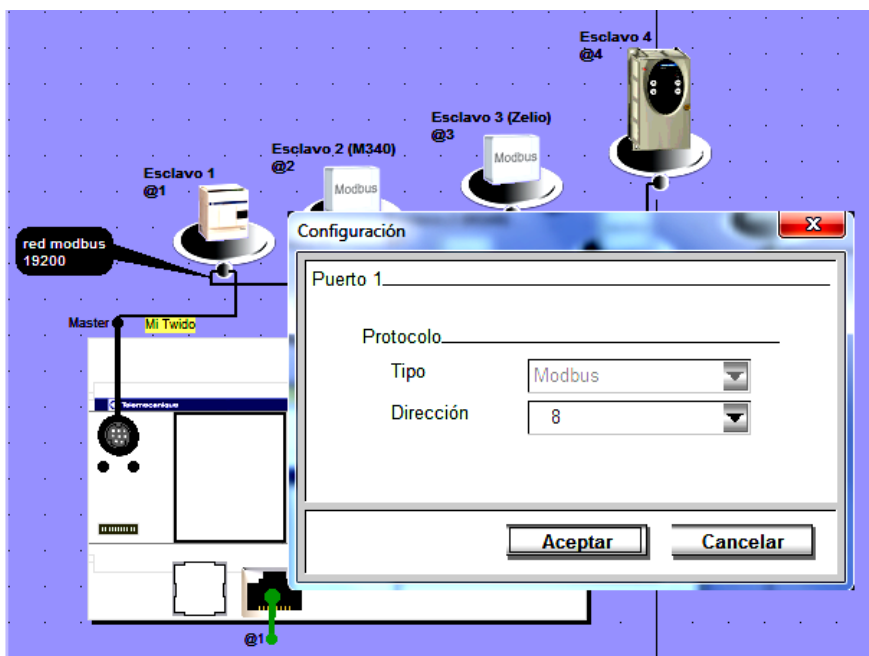
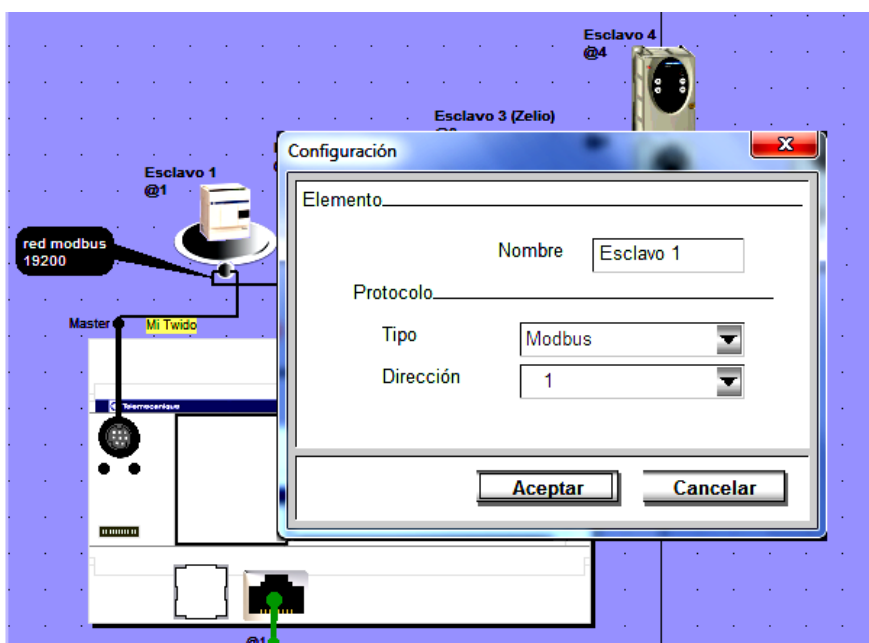


FIGURA 20: CONFIGURACION DEL ESCLAVO 1



2.3.1.2 PROGRAMA EN LADDER

PRIMERA PARTE DEL PROGRAMA LADDER: COMUNICACIÓN ENTRE EL MAESTRO PLC TWIDO Y EL ESCLAVO 1 PLC M340

Esta parte del programa ladder tiene como finalidad presentar la escritura y lectura de un bit con macros. Comm, con tablas de lectura y escritura de un bit y de N bits. Los bits escritos y leídos se emplearan para activar/desactivar salidas del PLC maestro y del PLC esclavo, a demás se presentan el funcionamiento de los bloques temporalizadores Tof, Ton, Tp y el bloque de función de registro de bits de desplazamiento. La tabla 1 presenta las variables utilizadas por el PLC maestro y PLC esclavo

Tabla 1: VARIABLES DEL MAETRO TWIDO Y DEL ESCLAVO TWIDO

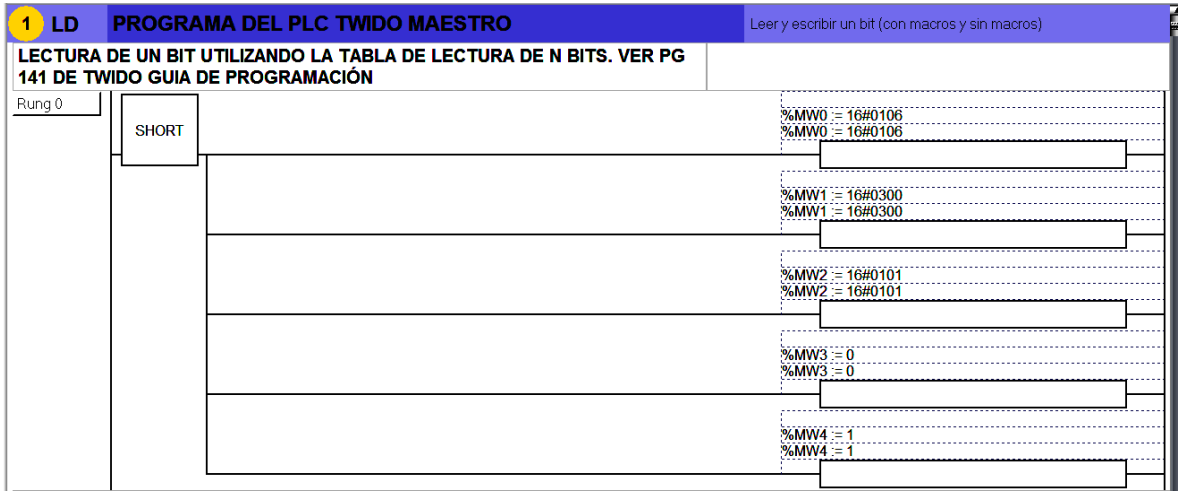
MAESTRO		MAESTRO		ESCLAVO		ESCLAVO
%I0.1	→	%MW12	→	%M1	→	→%Q0.0
%I0.2	→	%MW28	→	%M3	→	→%Q0.1 a %Q0.15
%Q0.0	←	← %MW7:X8		← %M0		← %I0.1
%Q0.1	←	← %MW23:X8		← %M2		← %I0.2

La tabla 1 presenta todas las variables utilizadas en la primera parte del programa en ladder. Cuando se activa la entrada %I0.1 del maestro esta activa la palabra de memoria %MW12, que a su vez escribe el valor de dicha palabra en el bit %M1 del esclavo para que este bit active la salida %Q0.0 del esclavo. Si se activa la entrada %I0.2 del maestro, esta activa la palabra de memoria %MW28 del

maestro para que active el bit %M3 del esclavo el cual activará/desactivará la salidas de la QO.1 a la Q0.15 secuencialmente a través del bloque de función del registro de bits de desplazamiento. Al activar la entrada %I0.1 del esclavo se activa el bit %M0 para que su valor sea leído por el bit 8 de la palabra de memoria %MW7 para activar la salida %Q0.0 del maestro. Por último cuando se activa la entrada %I0.2 del esclavo se activa el bit %M2 del maestro que a su vez activa el bit 8 de la palabra de memoria %MW23 para que dicha palabra active la salida %Q0.1 del maestro.

PROGRAMA DEL MAESTRO, sección 1: presenta la programación de la tabla de lectura de N bits para leer el valor del bit %M0 del esclavo. Ver figura 21.

FIGURA 21: LECTURA DE UN BIT

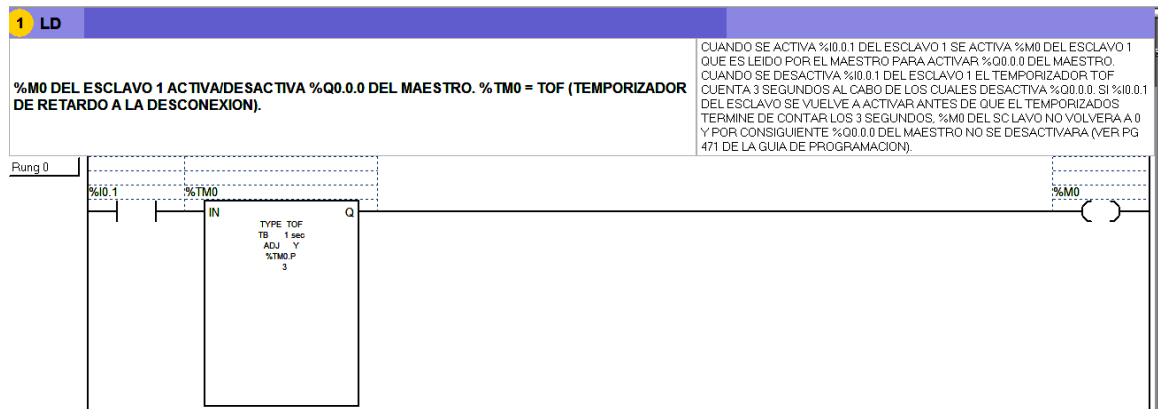


La figura 23 presenta la programación de la tabla de palabras para leer un bit. La palabra de memoria %MW3 lleva la dirección del bit que se va leer en este caso

%M0 “MW3 = 0” y %MW4 lleva el número de bits que se van a leer, para este caso 1 “%MW4 = 1”

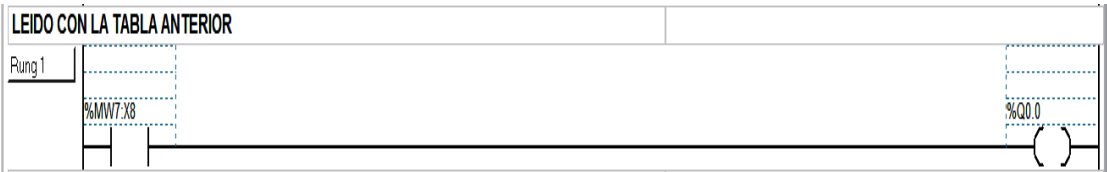
PROGRAMA DEL ESCLAVO 1, Sección 2: presenta como se puede activar/desactivar la salida Q0.0 del maestro con la lectura del bit %M0 y la programación del temporizador TOF del esclavo 1.

FIGURA 22: PROGRAMA DEL ESCLAVO 1, SECCION 2:



En la figura 22. Cuando se activa %I0.1 del esclavo 1 se activa %M0 del esclavo 1 que es leído por el maestro para activar %Q0.0 del maestro. Cuando se desactiva %I0.1 del esclavo 1 el temporizador TOF cuenta 3 segundos al cabo de los cuales desactiva %Q0.0. Si %I0.1 del esclavo se vuelve a activar antes de que el temporizador termine de contar los 3 segundos, %M0 del esclavo no volverá a 0 y por consiguiente %Q0.0 del maestro no se desactivara (ver pág. 471 de la Twido guía de programación).

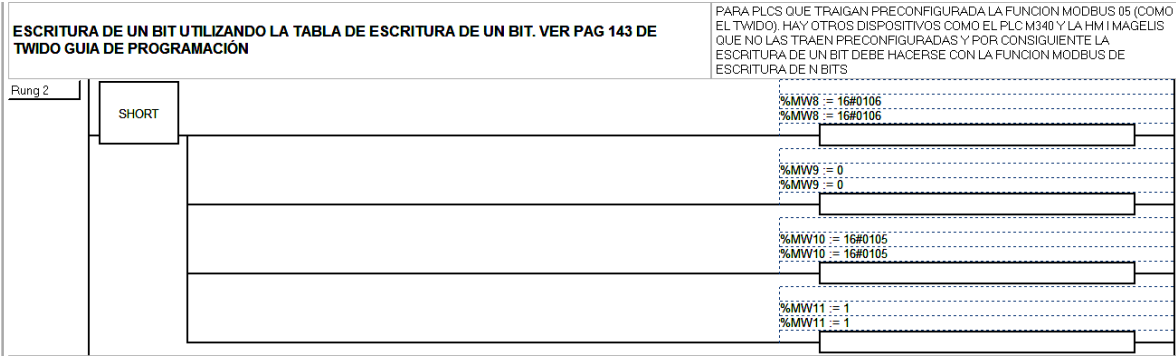
FIGURA 23: PROGRAMA DEL MAESTRO, SECCION 2:



En la figura 23 el bit 8 de la palabra de memoria %MW7 activa la salida %Q0.0

PROGRAMA DEL MAESTRO, Sección 3: presenta la programación de la tabla de palabras para escribir en el bit %M1 del esclavo.

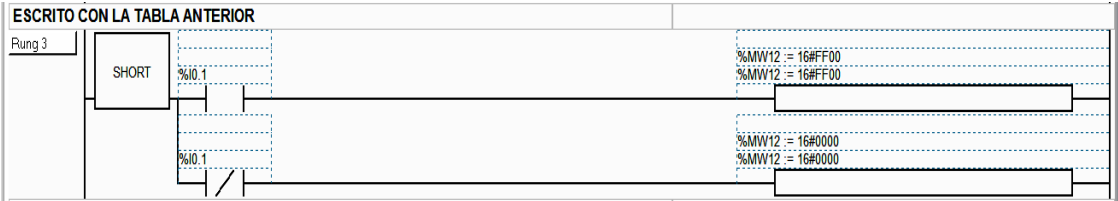
FIGURA 24: ESCRITURA DE UN BIT



La palabra de memoria %MW11 lleva la dirección del bit que va hacer escrito. En este caso %M1 “%MW11 = 1”

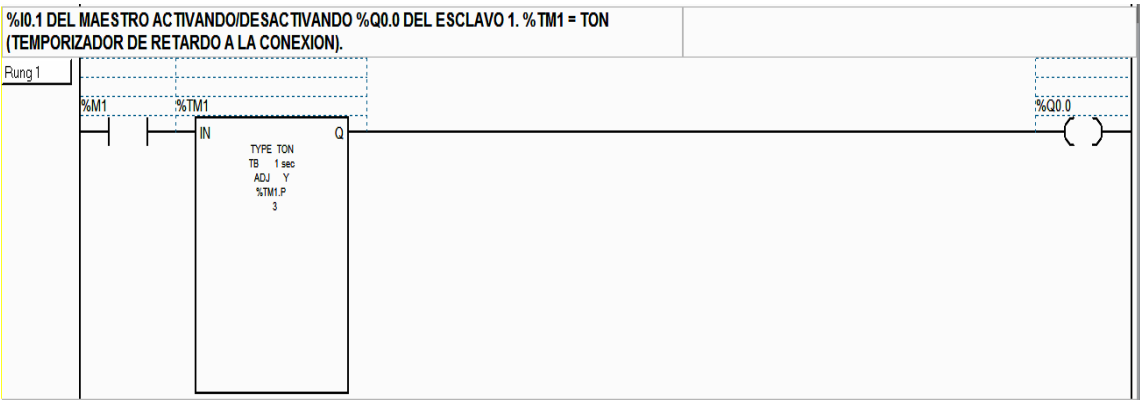
PROGRAMA DEL MAESTRO, Sección 4: presenta como se puede activar/desactivar la salida Q0.0 del esclavo 1 utilizando la programación del temporizador TON y la escritura de un bit para activar el bit %M1 del esclavo 1

FIGURA 25: PROGRAMA DEL MAESTRO, SECCION 4



La entrada %I0.1 activa la palabra de memoria %MW12

FIGURA 26: PROGRAMA DEL ESCLAVO 1, SECCION 4

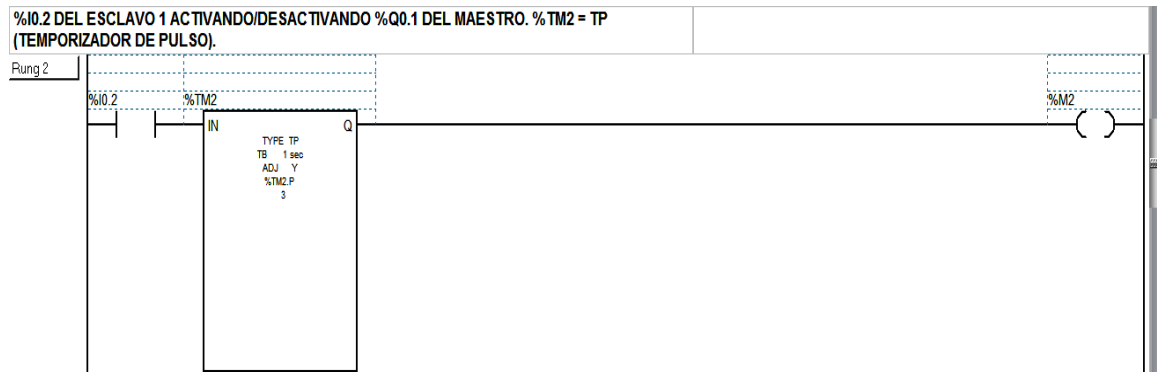


En la figura 26 %I0.1 el Maestro activa %M1 del esclavo 1, lo que el bloque TON inicie una cuenta de 3 segundos, al cabo de los cuales se activara %Q 0.0 del esclavo 1. Si %I0.1 del Maestro se desactiva antes de que el temporizador TON

termine de contar los 3 segundos, Q0.0 no se activara (ver pág. 472 de Twido guía de programación).

PROGRAMA DEL ESCLAVO 1, Sección 5: presenta como se puede activar/desactivar la salida Q0.1 del maestro con la lectura del bit %M2 utilizando la función C_RD1B_VAL_1 de la macros com1 y el temporizador TP del esclavo

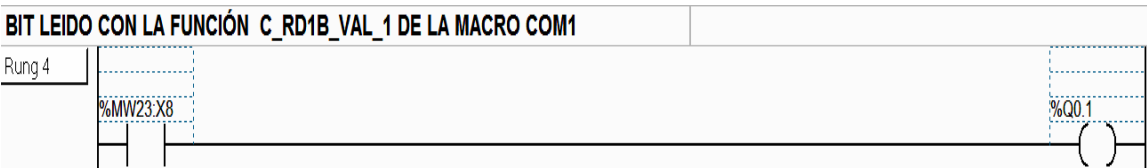
FIGURA 27: PROGRAMA DEL ESCLAVO, SECCION 5



En la figura 27, la entrada %I0.2 del esclavo 1 activa el bit %M2 del esclavo 1 y este a su vez activa %Q0.1 del maestro; simultáneamente empieza la cuenta de 3s del temporizador al cabo de los cuales se desactivara %M2 del esclavo 1 y por consiguiente %Q0.1 del maestro. La activación/desactivación reactivación %I0.2 antes de la finalización de la cuenta del temporizador no tiene ningún efecto en su funcionamiento (ver pág. 473 de Twido guía de programación)

La figura 28 presenta la función C_RDB_VAL_1 de la macro com1 para leer el valor del bit %M2 del esclavo y almacenarlo en el bit 8 de la palabra de memoria %MW23 del maestro y activar por medio de esta la salida %Q0.1 del maestro.

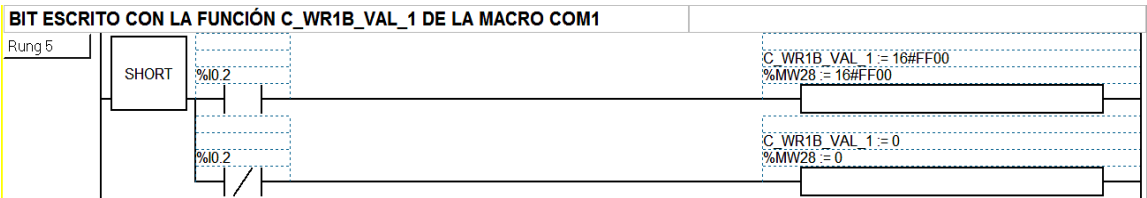
FIGURA 28: PROGRAMA DEL MAESTRO, SECCION 5



PROGRAMA DEL MAESTRO, Sección 6: presenta como se puede activar/desactivar la salida Q0.1 hasta la Q0.15 del esclavo secuencialmente “cuando se activa una salida se desactiva la anterior a esta” cuando se escribe para activar el bit %M3 utilizando la función C_WR1B_VAL_1 de la macros com1 y la función del registro de bits de desplazamiento

La figura 29 presenta la entrada %I0.2 para activar la palabra de memoria %MW28 gestionada por la función C_WR1B_VAL_1 DE LA MACRO COM1

FIGURA 29: PROGRAMA DEL MAESTRO, SECCION 6



Un flanco ascendente en I0.3 del esclavo 1 desplaza los bits del registro hacia la derecha mientras que un flanco ascendente en %I0.2 del maestro activa %M3 del esclavo1, lo que a su vez desplaza los bits del registro hacia la izquierda. El primer

flanco ascendente de %I0.2 del maestro activa %Q0.1 del esclavo 1; el segundo flanco ascendente de %I0.2 del maestro desactiva %Q0.1 del esclavo1 y activa %Q0.0.2 del esclavo1 y así sucesivamente hasta desactivar %Q0.14 del esclavo1 para activar %Q0.0.15 del esclavo1 .Ver figura 30.

FIGURA 30: BLOQUE DE FUNCION DEL REGISTRO DE BITS DE DESPLAZAMIENTO

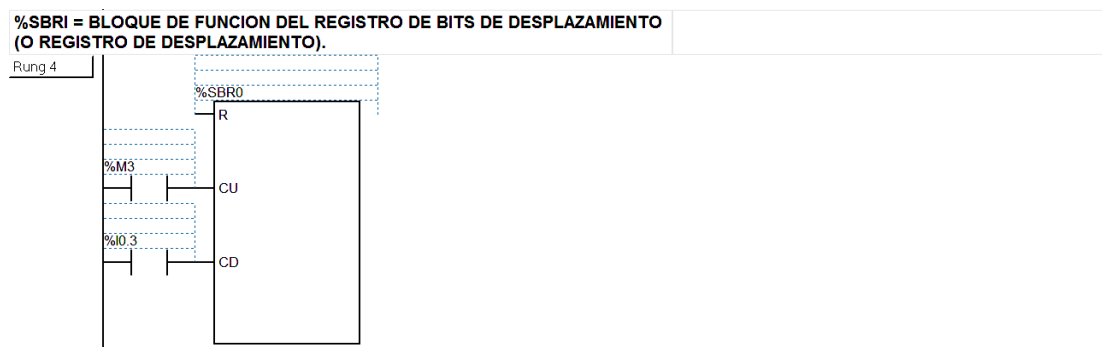


FIGURA 31: FIJACION A 1 DEL BIT MENOS SIGNIFICATIVO DEL REGISTRO DE DESPLAZAMIENTO



Esta parte del programa evita que se pierda el bit 15 cuando el registro de desplazamiento se desplaza hacia la izquierda. El valor del bit 15 pasa al bit 0.

FIGURA 32: GESTION DE LOS 5 BITS MENOS SIGNIFICATIVOS DEL REGISTRO DE DESPLAZAMIENTO %SBR0

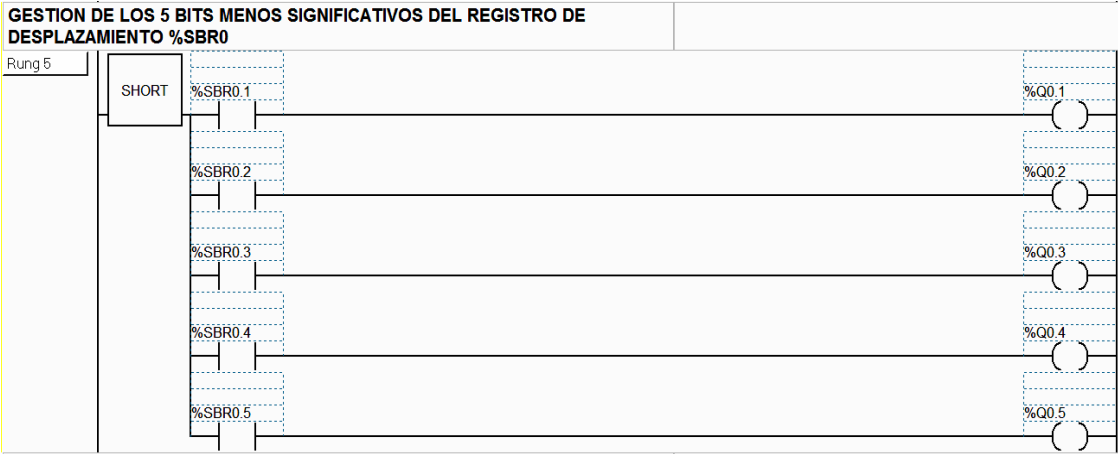
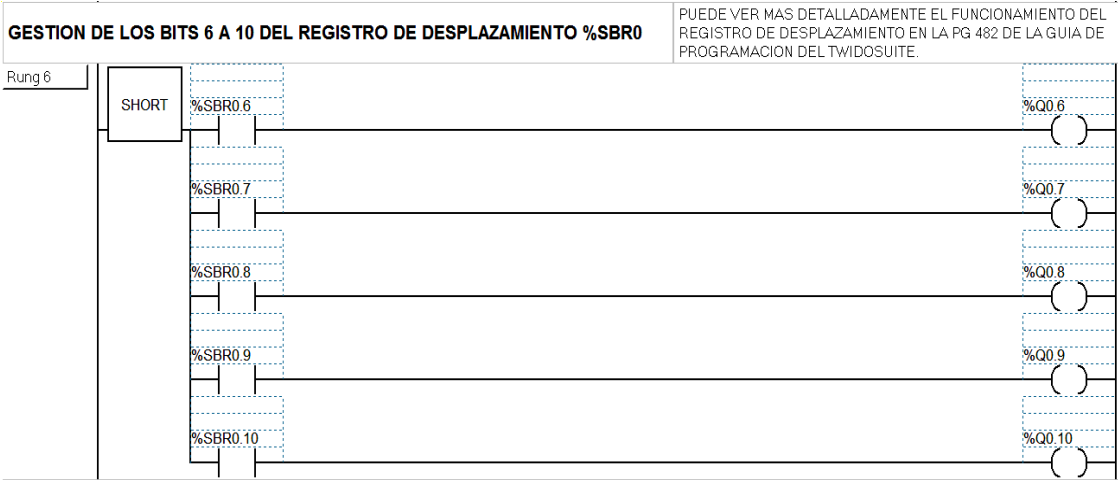


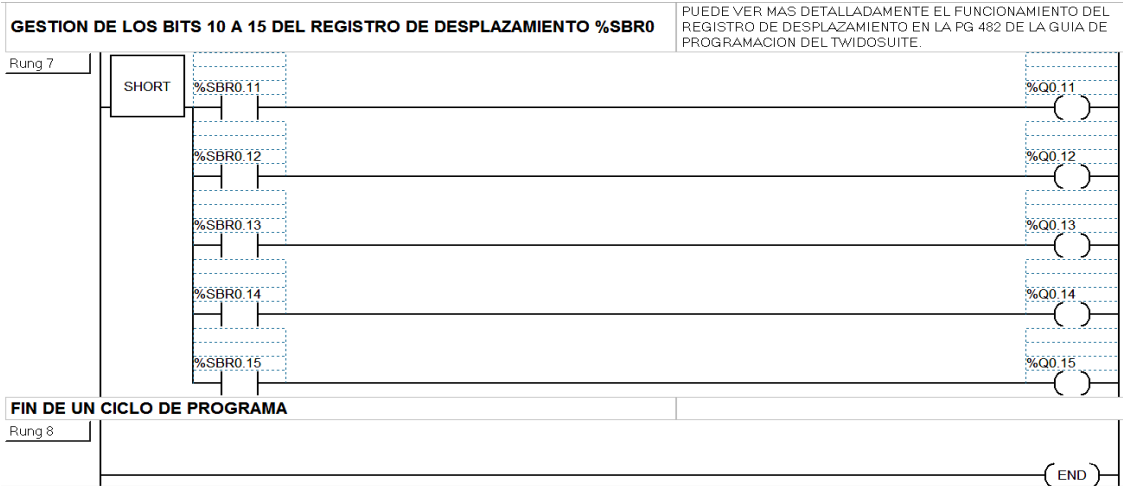
FIGURA 33: BITS (6 A 10) DEL REGISTRO DE DESPLAZAMIENTO %SBR0



En la figuras 32, 33 y 34, cada bit %SBRO.x activa/desactiva la salida %Q0.x, es decir: %SBR0.1 activa/desactiva %Q0.0.1 del esclavo 1, SBR0.2 activa/desactiva %Q0.0.2 del esclavo 1 y así sucesivamente hasta que %SBR0.15 activa/desactiva

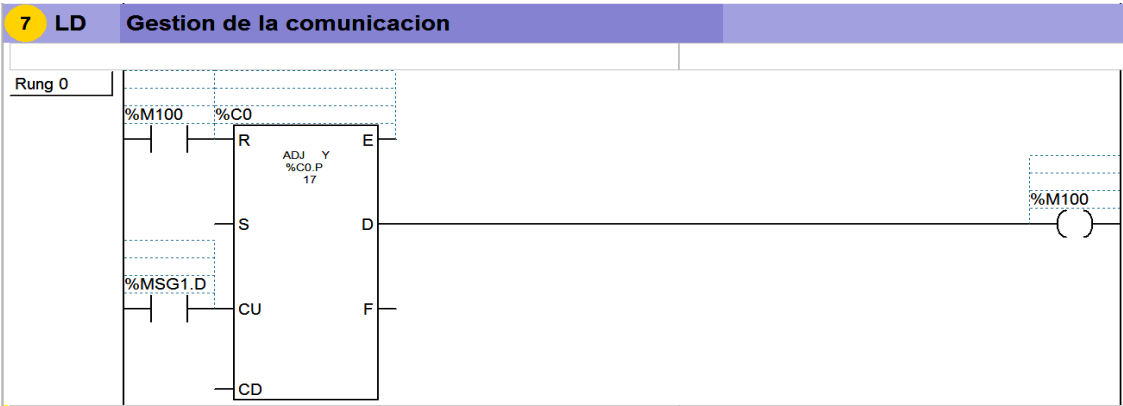
%Q0.0.15 del esclavo 1. **NOTA:** se recomienda ver el funcionamiento del registro de desplazamiento en la Pg 482 de la guía de programación del Twidosuite.

FIGURA 34: BITS (10 A 15) DEL REGISTRO DE DESPLAZAMIENTO %SBR0



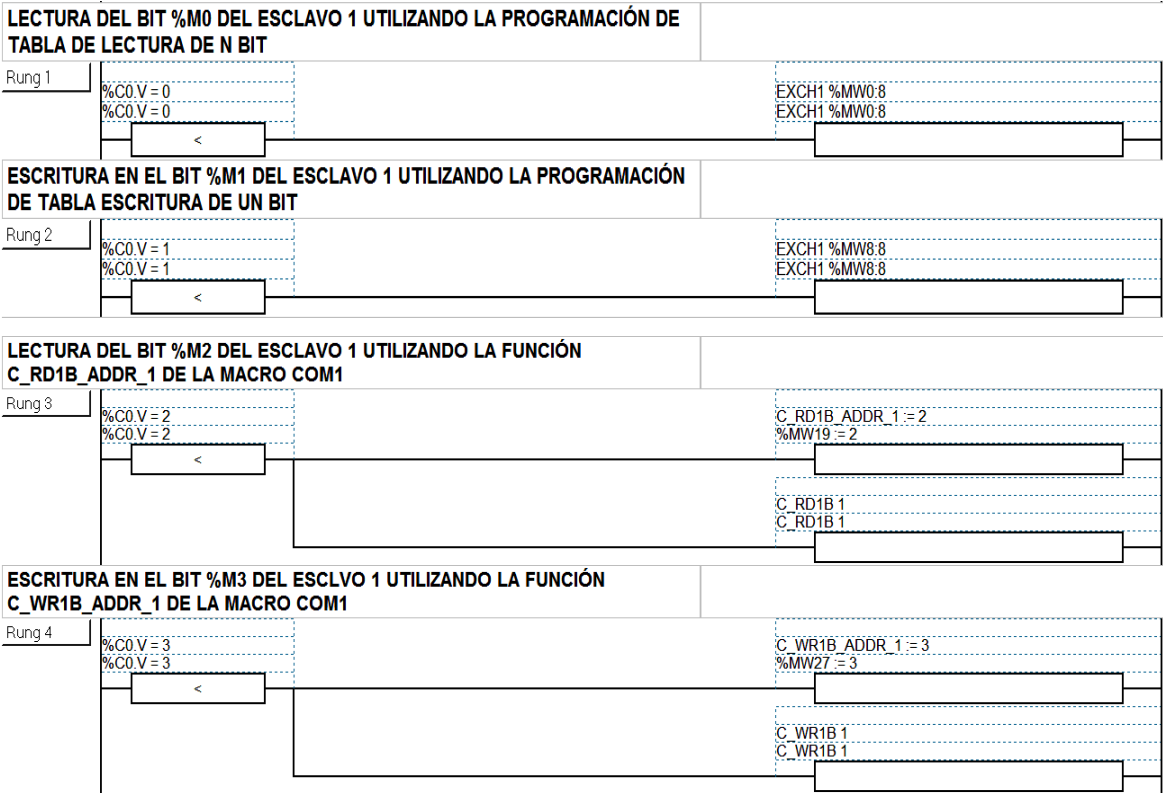
PROGRAMA DEL MAESTRO, GESTIÓN DE LA COMUNICACIÓN

FIGURA 35: BLOQUE CONTADOR PROGRESIVO



El bloque de la figura 35 es utilizado con el bit %MSG1.D para informar cuando una instrucción EXCHx ha terminado.

FIGURA 36: UTILIZACION DE LA INSTRUCCIÓN EXCHx

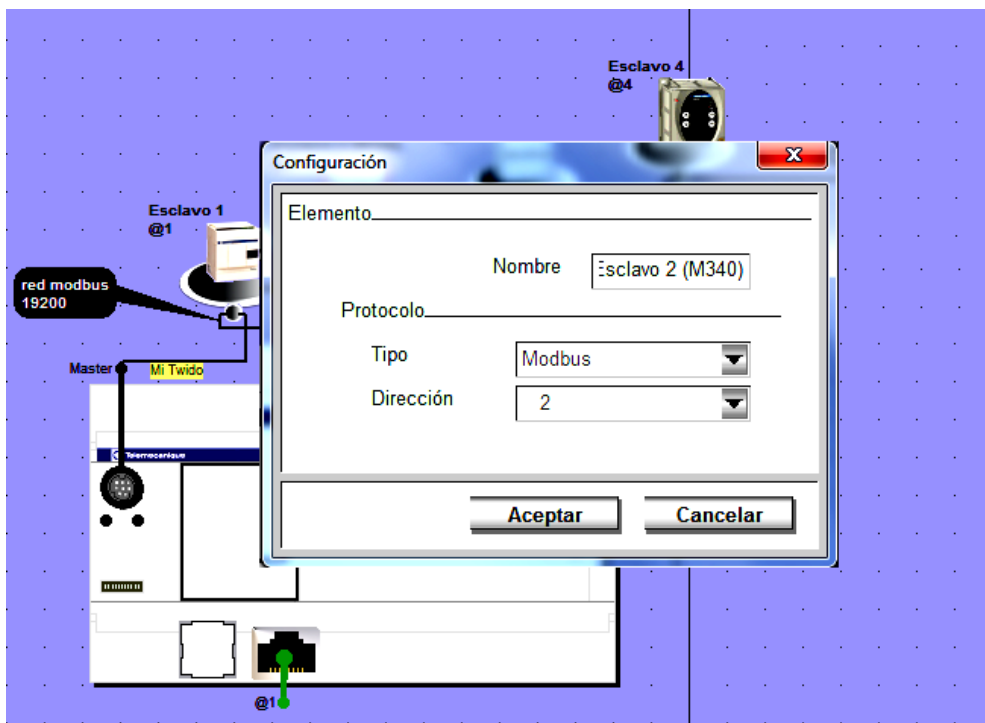


La figura 36 presenta la EXCHx utilizada para gestionar el intercambio de mensajes. Esta instrucción ya fue presentada en la tercera parte del primer programa de este documento.

2.3.2 SEGUNDA PARTE: COMUNICACIÓN ENTRE EL PLC TWIDO MAESTRO Y EL PLC M340 ESCLAVO

2.3.2.1 PROGRAMA MAESTRO

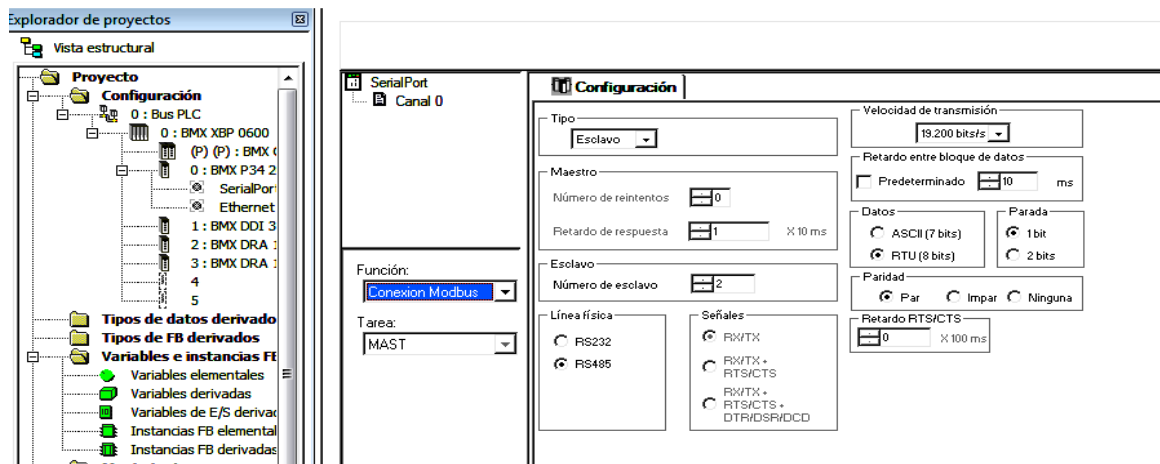
FIGURA 37: CONFIGURACION DEL ESCLAVO



La figura 37 presenta como se configura el protocolo modbus para el esclavo 2 en el programa del maestro. **Nota:** la configuración del protocolo modbus para el maestro Twido en el programa del maestro ya se hizo en la figura 19, por lo tanto no se presenta en esta parte del programa.

2.3.2.2 PROGRAMA DEL ESCLAVO 2

FIGURA 38: CONFIGURACIÓN DEL ESCLAVO 2



La figura 38 presenta como se configura el protocolo modbus para el esclavo 2 en el programa del esclavo. Para acceder a esta de configuración previamente debe ir a la ventana explorador de proyectos para abrir la carpeta configuración y en ella elegir “0: BUS PLC” para seleccionar SerialPort. **Nota:** Se recomienda al usuario ver el quinto programa del documento de modbus TCP-IP, donde se presenta el manejo del software Unity pro para crear un nuevo programa.

2.3.2.3 SEGUNDA PARTE DEL PROGRAMA LADDER. COMUNICACIÓN ENTRE EL MAESTRO PLC TWIDO Y EL ESCLAVO 1 PLC M340:

La segunda parte del programa ladder ejecuta las siguientes acciones:

%I0.1.1 del esclavo 2 (M340) activa/desactiva el bit %M0 del esclavo 2 para que sea leído su valor y almacenado en el bit 8 del palabra MW39 del maestro (Twido) para activar/desactivar %Q0.2 del maestro. Un flanco descendente en %I0.1.2 del

esclavo 2 ("E2") inicia el conteo de 3 segundos en el temporizador TOF, al final de los cuales se activara el bit %M1 que será leído por el maestro y guardado su valor en el bit 9 de la palabra %MW39 del maestro para activar %Q0.3 del maestro ("M"). En el bit %M2 del esclavo 2 se escribe el valor del bit 8 de la palabra %MW46 del maestro que es activada por la entrada %I0.3 del Maestro para activar/desactivar %Q0.2.1 del esclavo 2. El bit %M3 del esclavo2 esta activado mientras %I0.4 del maestro también lo esté; cuando %I0.4 del maestro se activa el bit 9 de %MW46 se activa y el temporizador TON en el esclavo inicia una cuenta de 3 segundos al cabo de los cuales activara %Q0.2.2 del esclavo2. %I0.1.3 del esclavo 2 activa/desactiva el bit 0 de la palabra %MW0 que activara /desactivara el bit 0 de la palabra %M57 del maestro y esta a su vez activar/desactivar %Q0.4 del Maestro. Un flanco ascendente en %I0.1.4 del Esclavo2 activara el bit 1 de %MW0 durante 3 segundos del esclavo 2, al cabo de los cuales se desactivara el bit 1 de %MW0; el bit 1 de %MW0.1 del Esclavo2 que activa/desactiva el bit 1 de %MW57 del maestro, activara/desactivara %Q0.5 del Maestro. %I0.5 y %I0.6 del Maestro activan/desactivan el bit 0 de %MW64 y el bit 1 %MW64 del maestro para que activen/desactiven el bit 0 de %MW1 y el bit 1 de %MW1 del esclavo2 respectivamente, que a su vez activan/desactivan %Q0.2.3 y %Q0.2.4 del esclavo2 respectivamente. Si se activa %I0.5 o %I0.6 del maestro se activara %Q0.2.5 del esclavo 2, pero si se activa %I0.5 y %I0.6 del maestro no se activara %Q0.2.5 del esclavo 2.

Tabla 2: ENTRADAS DEL MAESTRO TWIDO QUE ACTIVAN/DESACTIVAN SALIDAS DEL ESCLAVO M340

MAESTRO TWIDO	ESCLAVO M340
%I0.3	%Q0.2.1
%I0.4	%Q0.2.2
%I0.5	%Q0.2.3
%I0.6	%Q0.2.4

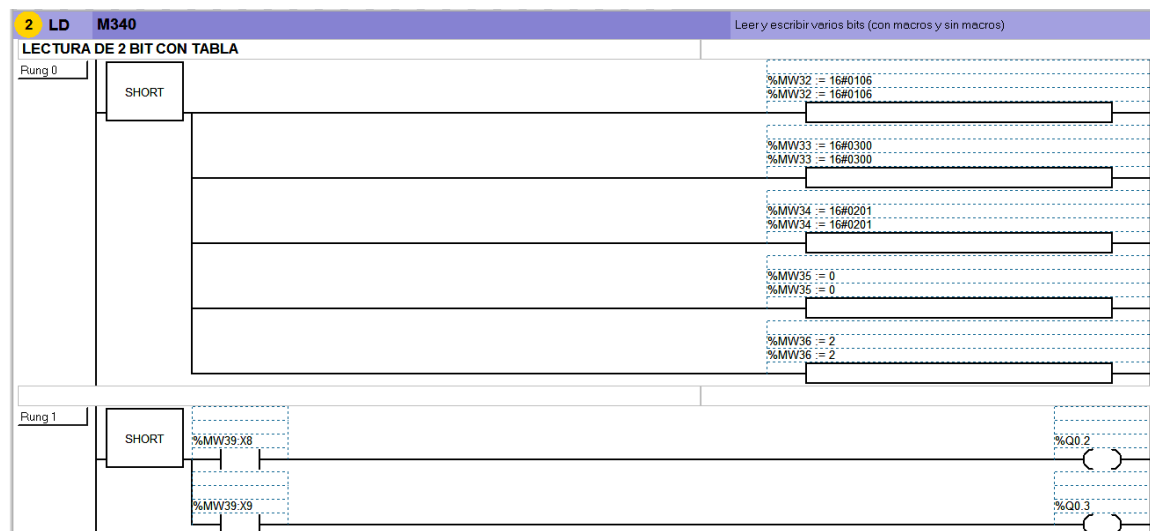
En la tabla 2, cuando se activa/desactiva cada entrada de maestro se activa y desactiva cada salida del esclavo.

Tabla 3: ENTRADAS DEL ESCLAVO M340 QUE ACTIVAN/DEACTIVAN SALIDAS DEL MAESTRO TWIDO

ESCLAVO M340	MAESTRO TWIDO
%I0.1.1	%Q0.2
%I0.1.2	%Q0.3
%I0.1.3	%Q0.4
%I0.1.4	%Q0.5

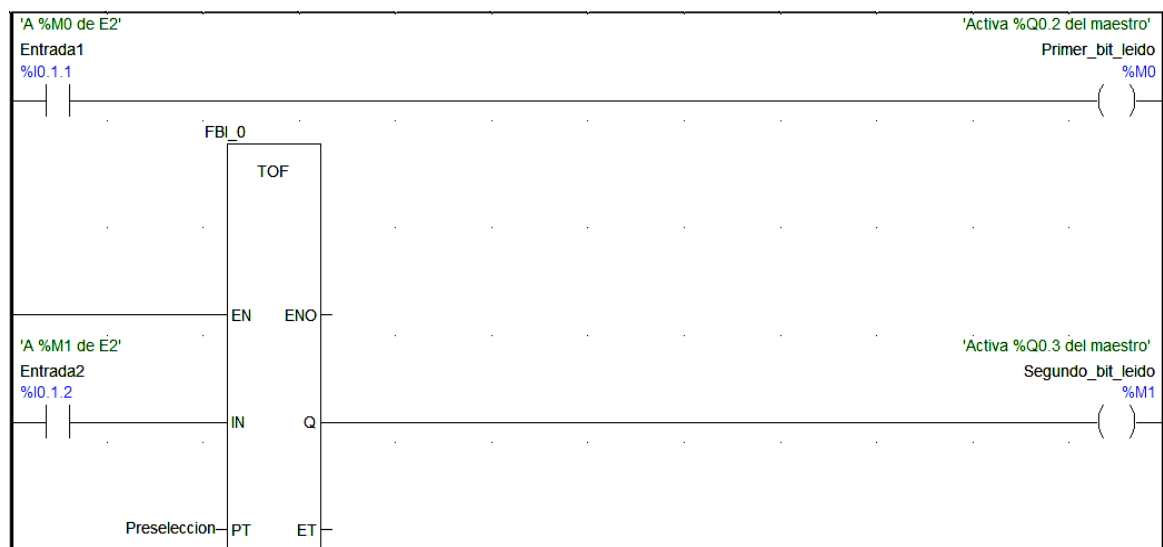
PROGRAMA LADDER DEL MAESTRO, Sección 7: presenta la programación de la tabla de lectura de N bits, para que los bits 8 y 9 de la palabra de memoria %MW39 del maestro lea el valor de los bits %M0 y % M1 del esclavo2 para activar/desactivar las salidas %Q0.2 y %Q0.3 del maestro. Ver figura 39

FIGURA 39 LECTURA DE DOS BITS



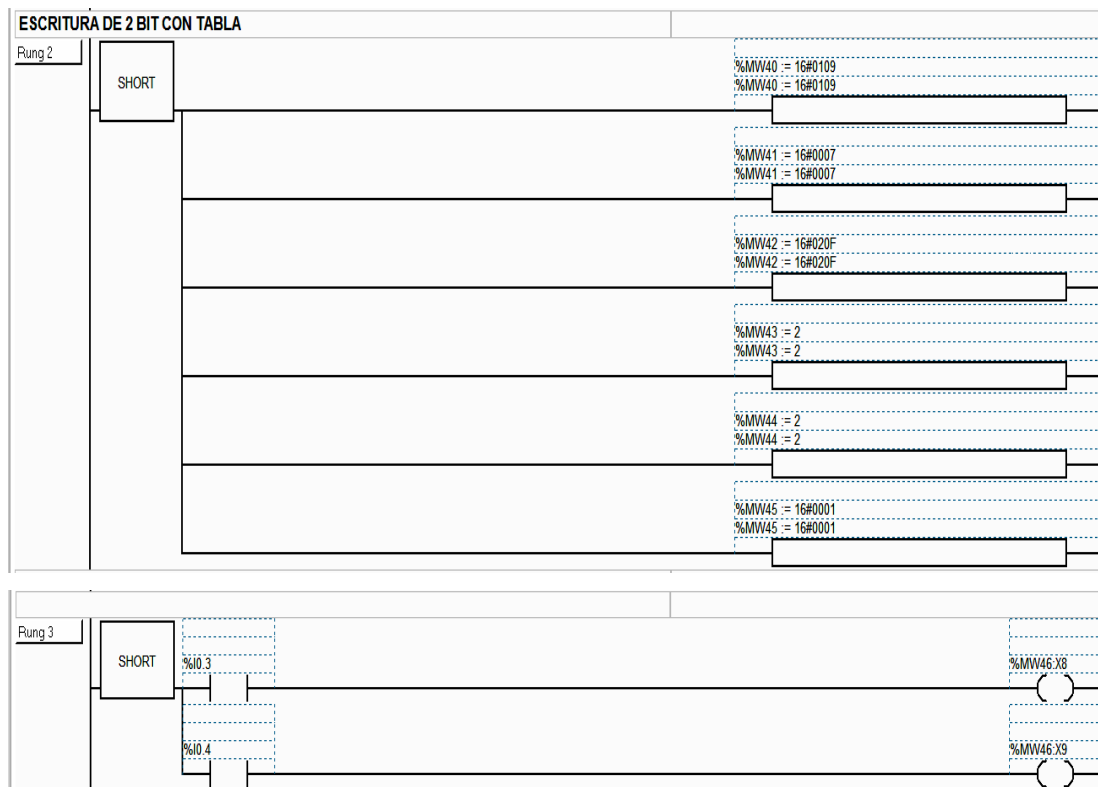
PROGRAMA LADDER DEL ESCLAVO 2, Sección 7: presenta el siguiente funcionamiento: cuando se activa/desactiva la entrada %I0.1.1 del esclavo 2 (M340) se activa/desactiva el bit %M0 del esclavo 2 para que sea leído por el maestro (Twido). Un flanco descendente en %I0.1.2 del esclavo 2 "E2" inicia el conteo de 3 segundos en el temporizador TOF, al final de los cuales se activara el bit %M1 que será leído por el maestro "M". Ver figura 40

FIGURA 40: ACTIVACIÓN DE %M0 Y %M1



PROGRAMA DEL MAESTRO, sección 8: presenta la programación de la tabla de escritura de N bits para escribir el valor de los bits 8 y 9 de la palabra de memoria %MW46 del maestro en los bits %M2 y %M3 del esclavo 2 para activar/desactivar las salidas %Q0.2.1 Y %Q0.2.2 del esclavo 2. Ver figura 41

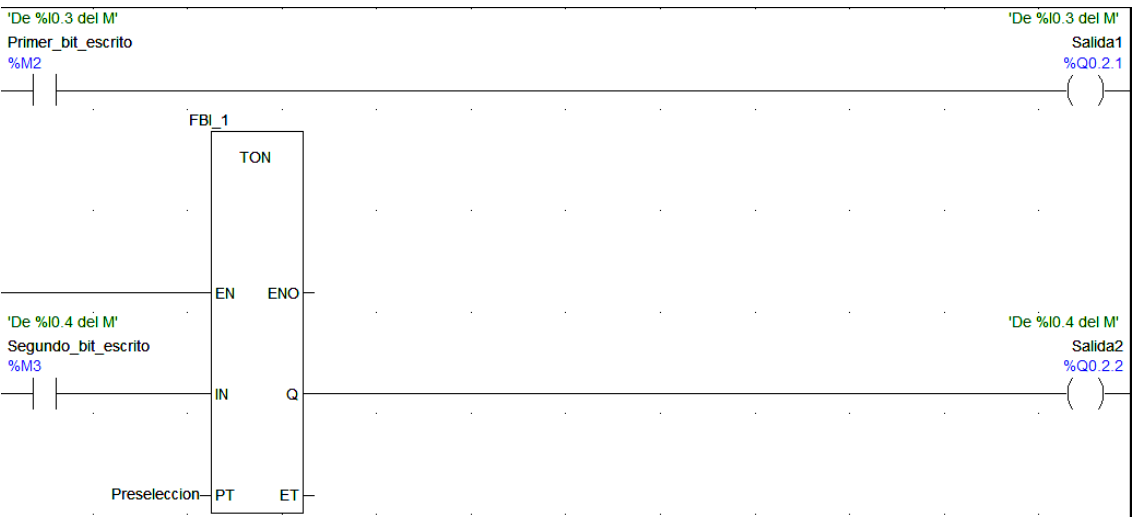
FIGURA 41: ESCRITURA DEL VALOR DE LOS BITS 8 Y 9 DE %MW46



La figura 41 presenta la programación de la tabla de escritura de N bits para escribir el valor de los bits 8 y 9 de la palabra de memoria %MW46 en los bits %M2 y %M3 del esclavo. En la tabla programada para escribir dos bits, a la palabra %MW43 se le asigna el número del primer bit que se va a escribir “M2” y la palabra %MW44 se le asigna el número de bits que se van a escribir “2”

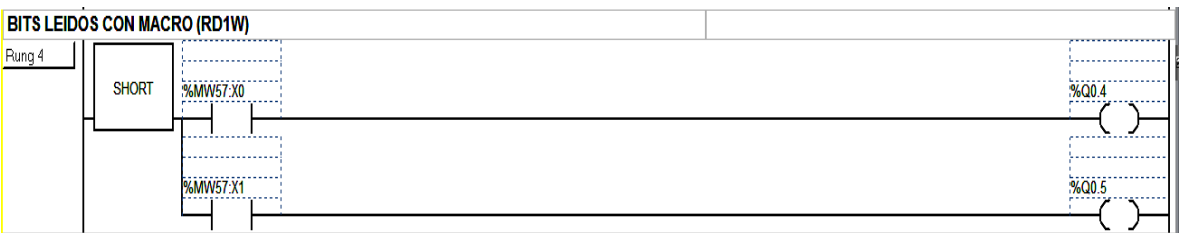
PROGRAMA DEL ESCLAVO 2, sección 8: presenta como se activa la salida %Q0.2.1 del esclavo 2 cuando %M2 del esclavo 2 se activa y como se activa %Q0.2.2 del esclavo 2 después de que el TON cuente 3 segundos de haberse activado %M3 del esclavo 2. Ver figura 43

FIGURA 42: ESCRITURA EN LOS BITS %M2 Y M3



PROGRAMA DEL MAESRO, sección 9: presenta la utilización de la función C_RD1W de la macro com 2 para que los bits 0 y 1 de la palabra de memoria %MW57 del maestro lea los bits 0 y 1 de la palabra de memoria %MW0 del esclavo 2 para activar/desactivar las salidas %Q0.4 y %Q0.5 del maestro

FIGURA 43: FUNCIÓN C_RD1W DE LA MACRO COM 2



PROGRAMA DEL ESCLAVO 2, sección 9: presenta como se activa el bit 0 de la palabra de memoria %MW0 del esclavo 2 cuando la entrada I0.1.3 del esclavo 2 se activa y como se activa el bit 1 de la palabra %MW0 del esclavo 2 después de que el TP cuente 3 segundos de haberse activado I0.1.4 del esclavo 2. Ver figura 44

FIGURA 44: LECTURA DE LOS BITS 0 Y 1 DE LA PALABRA %MW0

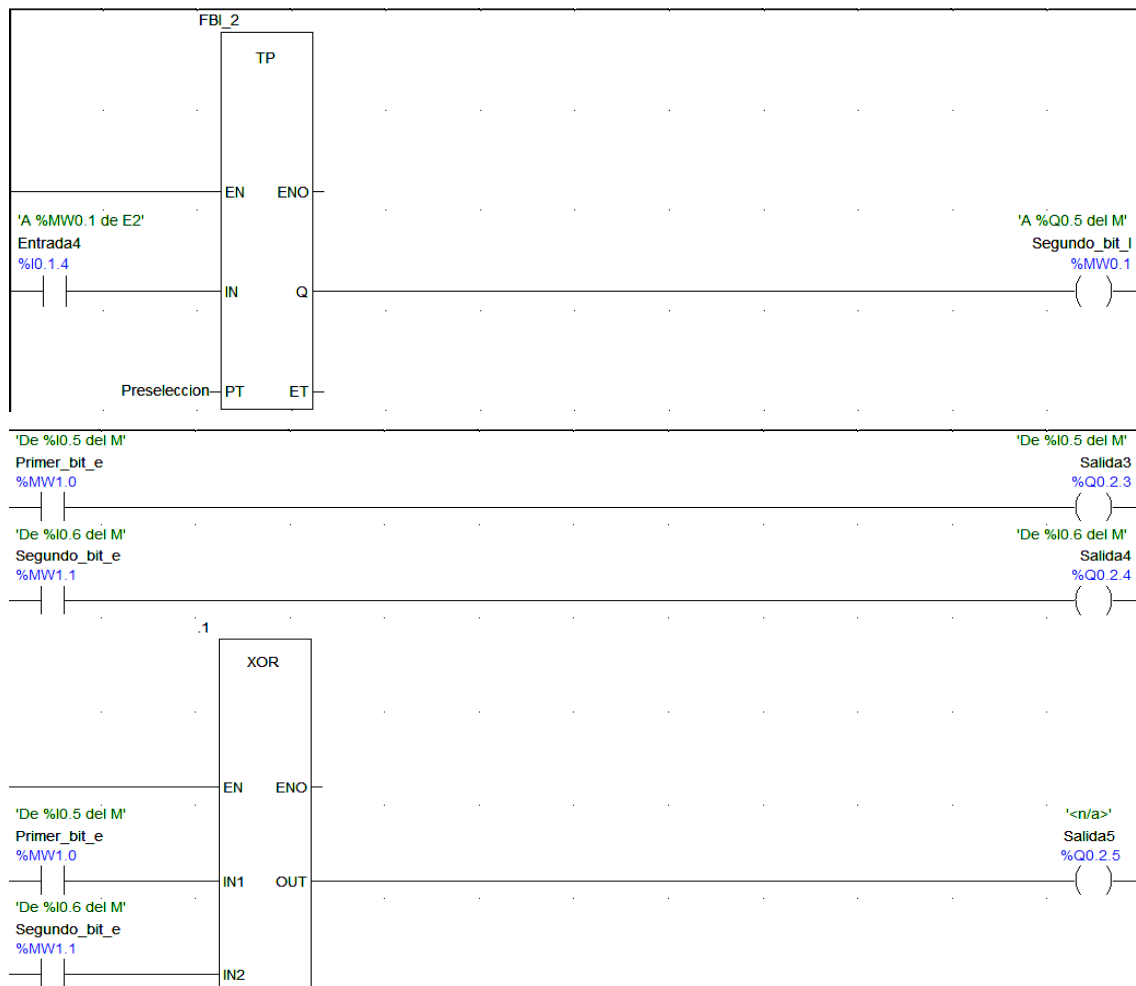


PROGRAMA DEL MAESRO, sección 10: presenta la utilización de la función C_WRNW de la macro com 2 para que los bits 0 y 1 de la palabra de memoria %MW64 del maestro escriba sus valores en el esclavo. Ver figura 45

FIGURA 45: FUNCIÓN C_WRNW DE LA MACRO COM 2



FIGURA 46: ESCRITURA EN LOS BITS 0 Y 1 DE LA PALABRA %MW1

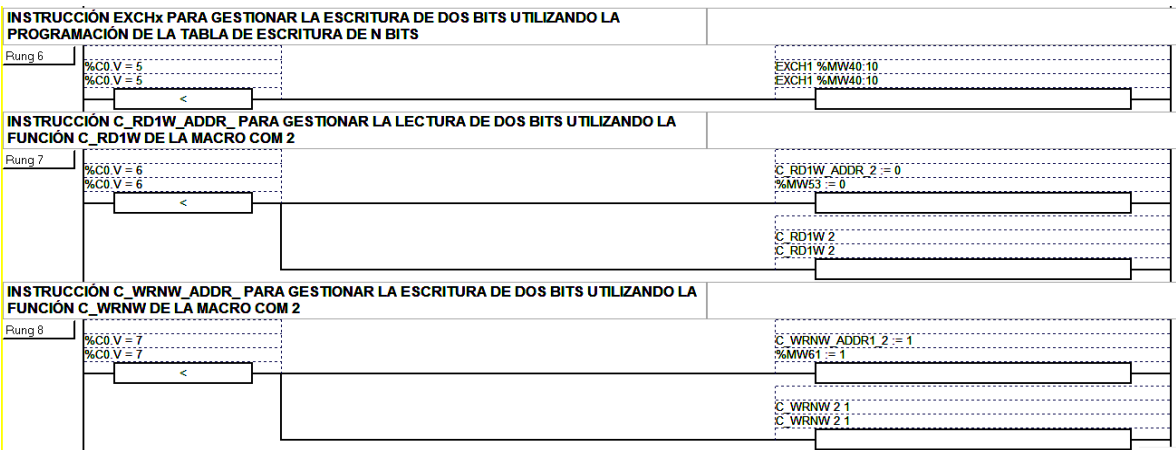


PROGRAMA DEL ESCLAVO 2, sección 10: presenta como se activa la salida %Q0.2.3 del esclavo 2 cuando el bit 0 de la palabra %MW1 del esclavo 2 se activa y como se activa %Q0.2.4 del esclavo 2 cuando el bit 1 de la palabra %MW1 del esclavo 2 se activa, a demás la salida % Q0.2.5 se activa si y solo si una de la entras %I0.5 o %I0.6 se activa, cuando se activan esta dos entradas no se activara % Q0.2.5. Ver figura 46

GESTIÓN DE LA COMUNICACIÓN, SEGUNDA PARTE

La gestión de la comunicación se hace con el bloque contador progresivo, con el bit %MSG1.D y la instrucción EXCHx. El bloque contador progresivo ya se presenta en la figura 35 y es el encargado junto con el bit %MSG1 de informar cuando una instrucción EXCHx ha terminado. A continuación se presenta las instrucciones EXCHx que se usara para gestionar el intercambio de mensajes de la segunda parte del programa. Ver figura 47

FIGURA 47: GESTIÓN DE LA COMUNICACIÓN, SEGUNDA PARTE



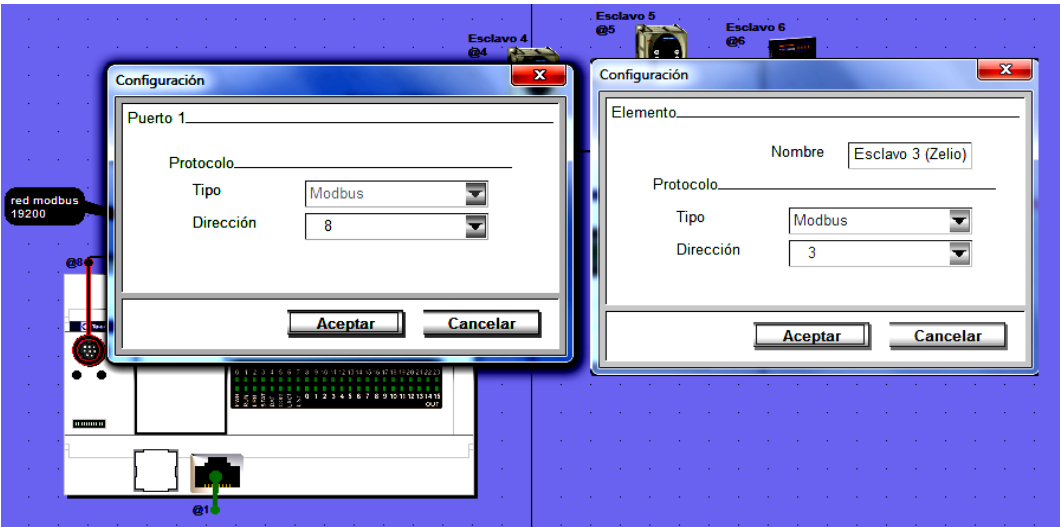
En la figura 47, $\%CO.v = x$ indica el paso que ejecuta el bloque contador progresivo. En cada paso de este bloque se ejecuta una instrucción EXCHx

2.3.3 TERCERA PARTE: COMUNICACIÓN ENTRE EL MAESTRO PLC TWIDO CON EL ESCLAVO 3 PLC ZELIO: La tercera parte del programa ejecuta las siguientes acciones:

I1 del esclavo 3 (Zelio) activa la palabra de memoria %MW20 guardada en O1 XT1 del esclavo 3 para que sea leído su valor y almacenado en la palabra %MW135 del maestro (Twido) y cuando esta palabra valga 1 active %Q0.6 del maestro. I2 del esclavo 3 (Zelio) activa la palabra de memoria %MW20 guardada en O1 XT1 del esclavo 3 para que sea leído su valor y almacenado en la palabra %MW135 del maestro (Twido) y cuando esta palabra valga 2 active %Q0.7 del maestro. Cuando se activa I1 y I2 del esclavo 3 (Zelio) activa la palabra de memoria %MW20 guardada en O1 XT1 del esclavo 3 para que sea leído su valor y almacenado en la palabra %MW135 del maestro (Twido) para que cuando esta palabra valga 3 active %Q0.6 y %Q0.7 del maestro. En la palabra de memoria %MW16 almacenada en J1 XT1 del esclavo 3 se escribe el valor de la palabra de memoria %MW140 del maestro, cuando esta palabra vale 1, es porque las entradas %I0.7 y %I0.8 del Maestro y %Q1 del esclavo 3 están activadas, cuando vale 2 es porque %I0.9 y %I0.10 del maestro están activadas y por ende se activara la salida %Q2 del esclavo 3, cuando vale 3 es porque las entradas %I0.11 y %I0.12 fueron activadas para activar las salidas Q1 y Q2. Por último los valores de la palabra de memoria %MW150 del maestro serán escritos en la palabra de memoria %MW17 del esclavo 3 para activar la salida Q3 del esclavo 3 cuando la entrada I0.21 del maestro se activa, para activar la salida Q4 del esclavo 3 cuando la entrada I0.22 se activa y para activar las salidas Q3 y Q4 del esclavo 3 cuando las entradas I0.21 y I0.22 del maestro se activan.

2.3.3.1 CONFIGURACIÓN DEL ESCLAVO ZELIO Y DEL MAESTRO TWIDO: La configuración del maestro y el esclavo se hace en el programa del maestro, en el cual se configura el puerto 1 del PLC maestro y la dirección y el nombre del esclavo. Ver figura 48

FIGURA 48: CONFIGURACIÓN DEL MAESTRO Y EL ESCLAVO



2.3.3.2 ENTRADAS Y SALIDAS DEL MAESTRO TWIDO Y DEL ESCLAVO ZELIO

Tabla 4: ENTRADAS DEL MAESTRO TWIDO QUE CONTROLAN SALIDAS DEL ESCLAVO 3 ZELIO

ENTRADAS DEL TWIDO	SALIDAS ZELIO
%I0.7 Y %I0.8	Q1
%I0.9 Y %I0.10	Q2
%I0.11 Y %I0.12	Q1 Y Q2
%I0.21	Q3
%I0.22	Q4
%I0.21 Y %I0.22	Q3 Y Q4

La tabla 4 presenta la activación de las salidas del esclavo 3 Zelio cada vez que se activan las entradas correspondientes del maestro twido. Nota cuando no se activa %I0.21 y %I0.22 no se activa ninguna salida.

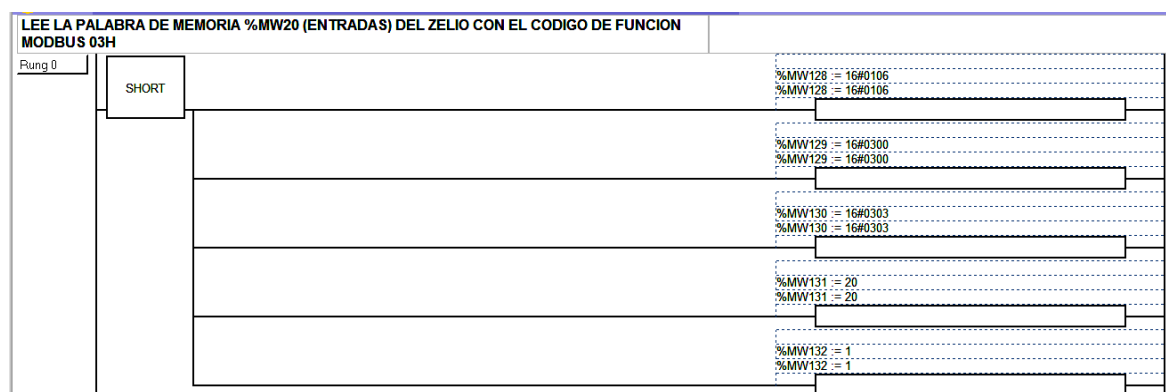
Tabla 5: ENTRADAS DEL ESCLAVO ZELIO QUE CONTROLAN SALIDAS DEL MAESTRO TWIDO

ENTRAS DEL ZELIO	SALIDAS DEL TWIDO
I1	%Q0.6
I2	%Q0.7
I1 Y I2	%Q0.6 Y %Q0.7

La tabla 5 presenta la activación de las salidas del Maestro Twido cada vez que se activan las entradas correspondientes del esclavo 3 “zelio”

2.3.3.3 PROGRAMA LADDER DEL MAESTRO TWIDO, sección 11: esta sección presenta la lectura de los valores de la palabra de memoria %MW20 del esclavo zelio con el código de solicitud 03H “lectura de N palabras”, estos valores serán almacenados en la palabra de memoria %MW135 del maestro para activar las salidas del maestro Twido. Ver figura 49 y 50

FIGURA 49: LECTURA DE LA PALABRA DE MEMORIA %MW20



En la figura 49 la palabra %MW131 indica la dirección a leer del esclavo Zelio “palabra que se va a leer” y la palabra %MW132 indica el número de palabras que se van a leer

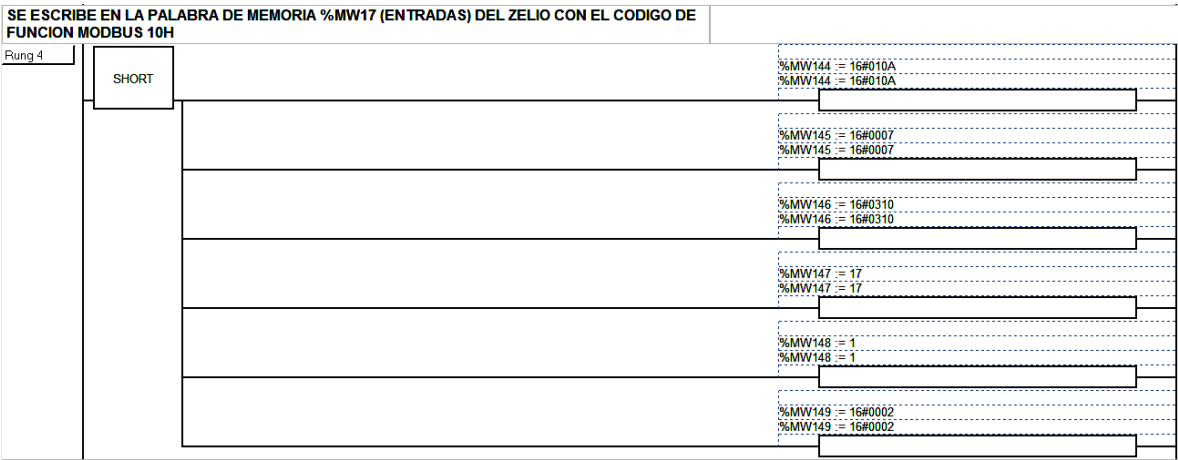
FIGURA 50: VALOR LEÍDO DE LA PALABRA %MW20.



La figura 50 muestra los valores leídos de la palabra de memoria %MW20 del esclavo zelio en la palabra de memoria %MW135 del maestro para activar las salidas del maestro zelio

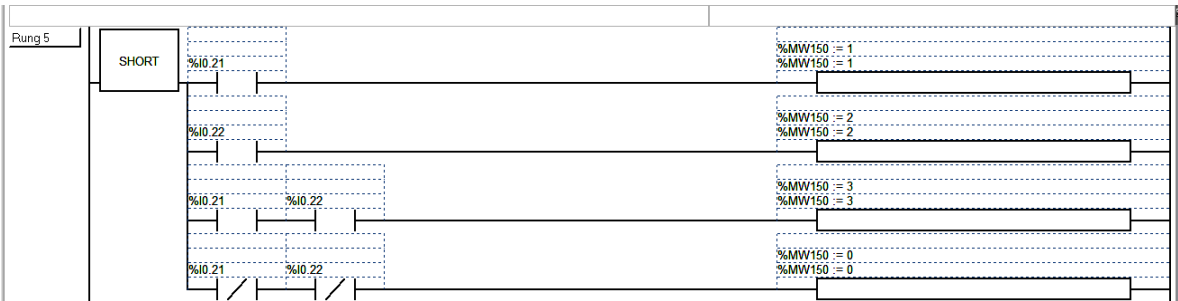
PROGRAMA LADDER DEL MAESTRO TWIDO, sección 12: esta sección presenta la escritura de los valores de la palabra de memoria %MW150 del maestro twido con el código de solicitud 10H “escritura de N palabras”, en la palabra de memoria %MW17 del esclavo Zelio para activar las salidas del esclavo Zelio. Ver figura 51 y 52

FIGURA 51: CODIGO DE SOLICITUD 10 H PARA ESCRIBIR EN %MW17



La figura 51 presenta en la palabra de memoria %MW147 la dirección donde se va a escribir los valores de la palabra %MW150 del maestro “palabra escrita”. Además presenta en la palabra de memoria %MW148 el número de palabras que se van a escribir

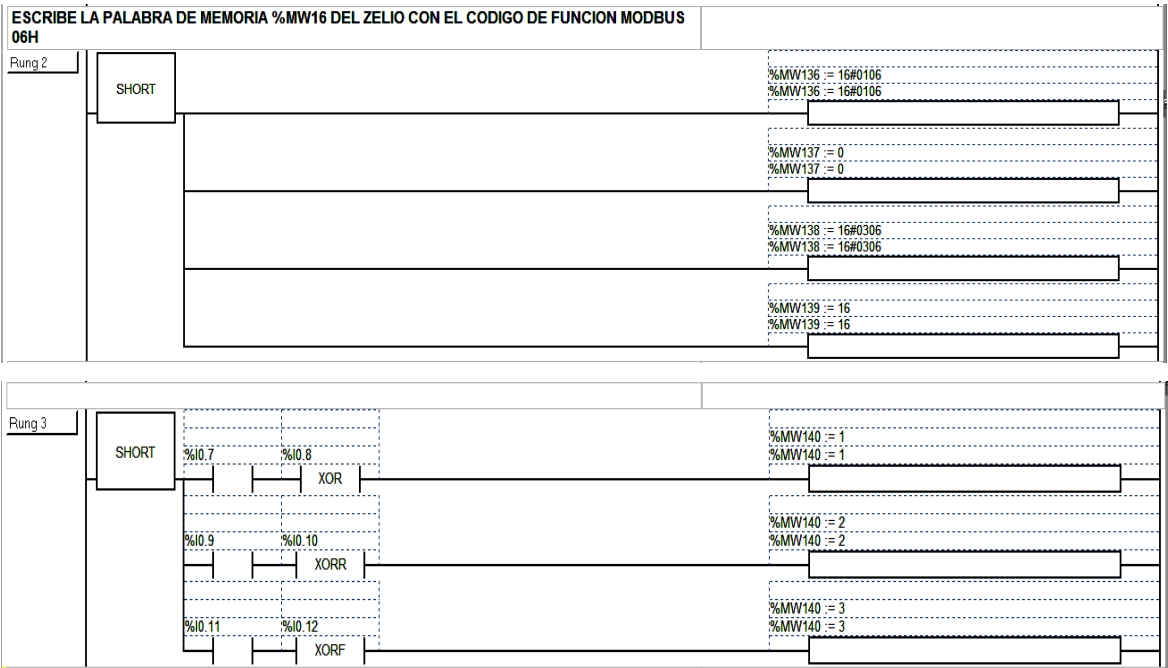
FIGURA 52: VALORES DE %MW150 QUE SE VAN A ESCRIBIR



La figura 52 muestra los valores que se van a escribir de la palabra de memoria %MW150 del maestro Twido en la palabra de memoria %MW17 del esclavo 3 para activar las salidas del esclavo 3

PROGRAMA LADDER DEL MAESTRO TWIDO, sección 13: esta sección presenta la escritura de los valores de la palabra de memoria %MW140 del maestro twido con el código de solicitud 06H “escritura de una palabras”, en la palabra de memoria %MW16 del esclavo Zelio para activar las salidas del esclavo Zelio. Ver figura 53

FIGURA 53: ESCRITURA DE LOS VALORES DE %MW140 EN %MW16

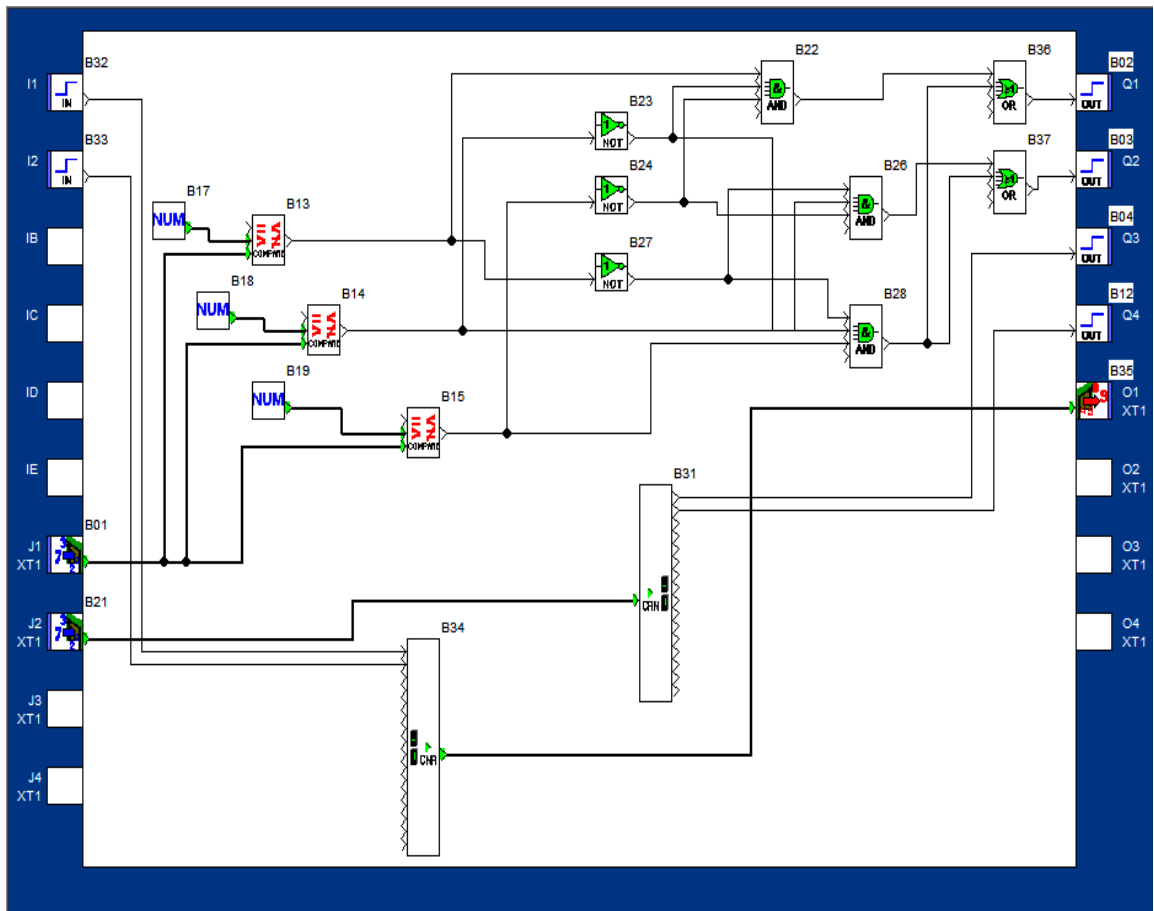


La figura 53 presenta el desarrollo de la escritura de los valores de la palabra %MW140 del maestro en la palabra %MW16 del esclavo Zelio para activar las salidas del zelio con las entradas del PLC Twido.

PROGRAMA DEL ESCLAVO ZELIO

En la tercera parte del segundo programa se presentó el contenido de esta parte del programa, de tal modo que ahora se presenta la descripción del programa del Zelio. Ver figura 54.

FIGURA 54: PROGRAMA DEL ESCLAVO ZELIO

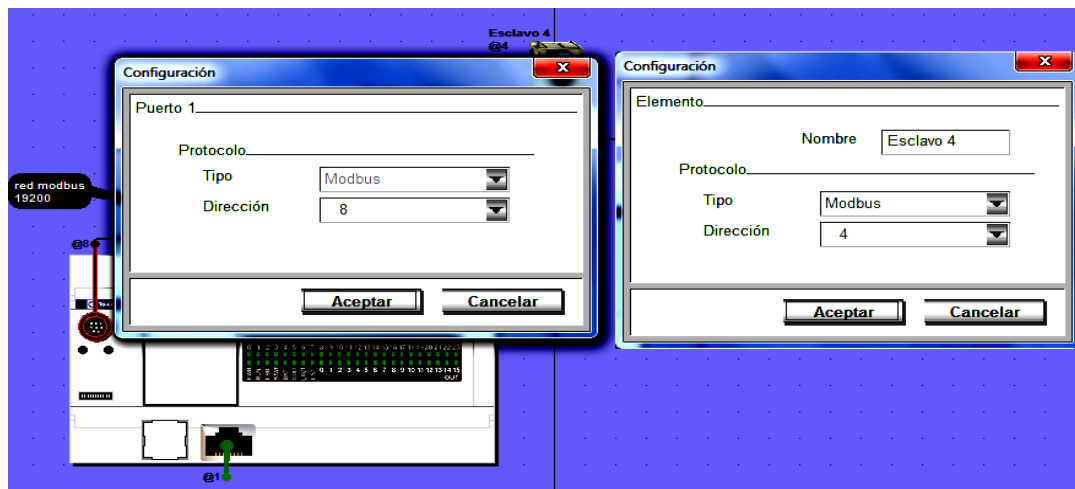


CUARTA PARTE: COMUNICACIÓN ENTRE EL MAESTRO PLC TWIDO CON EL ESCLAVO ALTIVAR 31 UTILIZANDO MACROS DRIVE

CONFIGURACIÓN DEL ESCLAVO ATV 31 Y DEL MAESTRO

La configuración del maestro y el esclavo se hace en el programa del maestro, en el cual se configura el puerto 1 del PLC maestro y la dirección y el nombre del esclavo. Ver figura 55

FIGURA 55: CONFIGURACIÓN DEL MAESTRO Y EL ESCLAVO



Para configurar el maestro y el esclavo de la figura 55 primero se da clic en el puerto 1 del Twido para configurar la ventana con el nombre “configuración puerto 1” en donde se configura el tipo de protocolo y la dirección del esclavo. Segundo se da clic en el Altivar 31 para que se despliegue la ventana con el nombre de

“configuración de elemento” en donde se configura el nombre del esclavo el tipo de protocolo y la dirección del esclavo

CUARTA PARTE DEL PROGRAMA LADDER: COMUNICACIÓN ENTRE EL MAESTRO PLC TWIDO Y EL ESCLAVO ALTIVAR 31

En esta cuarta parte del programa ladder el variador de velocidad ATV31 (esclavo 4) que controla un motor trifásico se controla con la Macro Drive 4 a través de sus funciones: D_RUN_FWD, D_RUN_REV, D_STOP, D_SELECT_SPEED y D_CLEAR_ERR. El motor puede realizar las siguientes acciones: girar en cualquiera de los dos posibles sentidos de giro, parar, trabajar, alcanzar y mantener alguna de estas tres velocidades (400 rpm, 800 rpm, 1200rpm y 1600 rpm).

Tabla 6: ENTRADAS DEL MAESTRO TWIDO QUE CONTROLAN ACCIONES DEL ESCLAVO 4 ALTIVAR 31

ENTRADA DEL MAESTRO	MOTOR
I0.13	gira a la derecha
I0.14	gira a la izquierda
I0.15	para
I0.13 Y I0.16	gira a la derecha con una velocidad de 400 rpm
I0.13 Y I0.17	gira a la derecha con una velocidad de 800 rpm
I0.13 Y I0.18	gira a la derecha con una velocidad de 1200 rpm
I0.13 Y I0.19	gira a la derecha con una velocidad de 1600 rpm
I0.14 Y I0.16	gira a la izquierda con una velocidad de 400 rpm
I0.14 Y I0.17	gira a la izquierda con una velocidad de 800 rpm
I0.14 Y I0.18	gira a la izquierda con una velocidad de 1200 rpm
I0.14 Y I0.19	gira a la izquierda con una velocidad de 1600 rpm

MACROS DRIVE

El proceso para configurar las macros drive se describe en el anexo de CANopen en las figuras 23, 24 y 25. Las funciones que componen las macros drive se presentan a continuación

Función D_MANAGER

La función controla la unidad de velocidad altivar 31 y se encarga de que el dispositivo siga funcionando. Esta función debe activarse al principio de cada ciclo de funcionamiento del twido y antes de activar cualquier otra función de la macro drive.

D_MANAGER parameter0: Este parámetro indica la instancia de esclavo a la que debe aplicarse la macro.

Función D_RUN_FWD

Esta función guía a la unidad de velocidad ATV en la ejecución del proceso Ejecutar siguiente “.giro a la derecha”.

D_RUN_FWD parameter0: Este parámetro indica la instancia de esclavo a la que debe aplicarse la macro.

Función D_RUN_REV

Esta función guía a la unidad de velocidad ATV en la ejecución del proceso Ejecutar anterior. “giro a la izquierda”.

D_RUN_REV parameter0: Este parámetro indica la instancia de esclavo a la que debe aplicarse la macro.

Función D_STOP

Esta función guía a la unidad de velocidad ATV en la aplicación de la secuencia Detener, para el proceso.

D_STOP parameter0: Este parámetro indica la instancia de esclavo a la que debe aplicarse la macro.

Función D_SELECT_SPEED

Esta función define la velocidad con el valor de la consigna o el valor preestablecido en función del modo de funcionamiento de la ATV seleccionado.

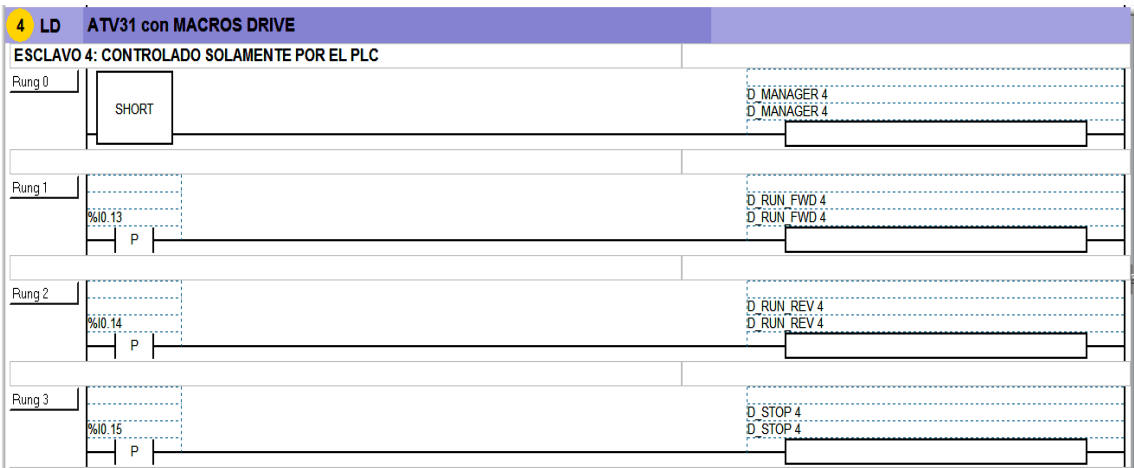
Para poder utilizar esta macro, deben introducirse los símbolos que aparecen a continuación: En modo de consigna, debe introducir D_SETPOINT_MODE y D_SETPOINT. Tenga en cuenta que la función de macros introduce D_SELECT_SPEED_VAL automáticamente. Ver tabla 7

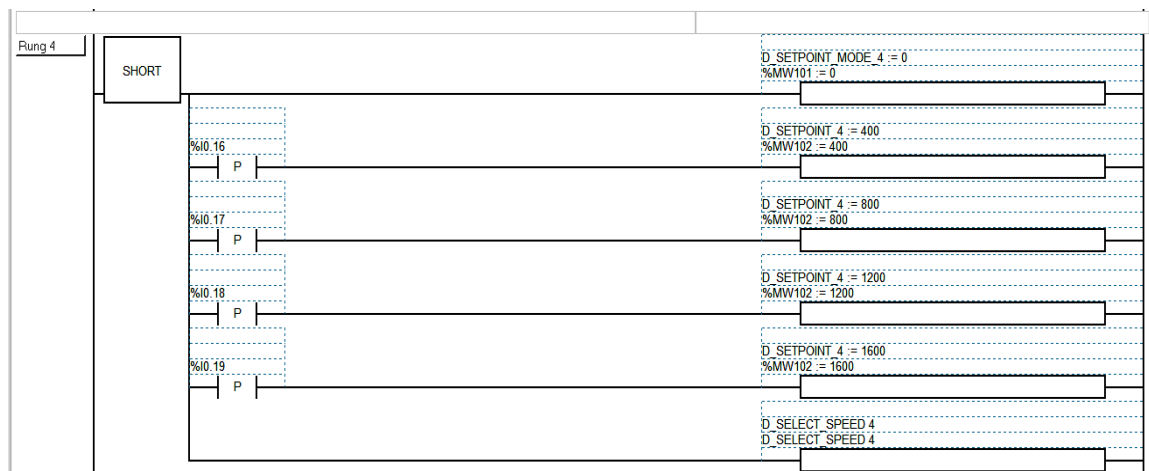
Tabla 7: FUNCIÓN D_SELECT_SPEED, MODO DE COSIGNA

Símbolo	Valor	Introducción
D_SETPOINT_MODE	0	Este símbolo guía a la ATV para que funcione en el modo de consigna.
D_SETPOINT	Velocidad	Este símbolo debe introducirse con el valor correspondiente a la velocidad tras haber activado el modo de consigna.
D_SELECT_SPEED_VAL	D_SETPOINT_MODE	La macro introduce este símbolo con el valor de bit leído tras la realización.

Fuente “software twidosuite, ayuda macros drive”

FIGURA 56: ESCLAVO ALTIVAR CON MACROS DRIVE





La figura 56 presenta las entradas del PLC Twido que servirán para activar las funciones de la macro drive 4 que controlaran el variador de velocidad AT

ANEXO 3: MANUAL MODBUS TCP/IP

CONTENIDO

3 GUIA PARA CREAR UNA APLICACIÓN MODBUS TCP-IP CON TWIDOSUITE	340
3.1 CONFIGURACIÓN DEL PUERTO 3	340
3.2 PRIMER PROGRAMA	342
3.2.1 CONFIGURACIÓN DE LA DIRECCIÓN IP	342
3.2.1.1 DIRECCIÓN IP DEL PLC 88.	342
3.2.1.2 DIRECCIÓN IP DEL PLC 89	343
3.2.1.3 DIRECCIÓN IP DEL PLC 90	343
3.2.2 PROGRAMA LADDER: CLIENTE, CLIENTE-SERVIDOR Y SERVIDOR	345
3.2.2.1 PRIMERA PARTE: ACCIONES QUE REALIZA EL CLIENTE (TWIDO 88) EN EL CLIENTE-SERVIDOR (TWIDO 89).	345
3.2.2.2 SEGUNDA PARTE DEL PROGRAMA: ACCIONES QUE REALIZA EL CLIENTE-SERVIDOR EN EL CLIENTE.	348
3.2.2.3 TERCERA PARTE DEL PROGRAMA: ACCIONES QUE REALIZA EL CLIENTE-SERVIDOR (TWIDO 89) EN EL SERVIDOR (TWIDO 90)	351
3.2.2.4 CUARTA PARTE DEL PROGRAMA: ACCIONES QUE REALIZA EL SERVIDOR (TWIDO 90) EN EL CLIENTE-SERVIDOR (TWIDO 89)	355
3.2.2.5 GESTION DE LA COMUNICACIÓN PARA LA PARTE TRES Y CUATRO DEL PROGRAMA UTILIZANDO MACROS COM	357
3.2.2.6 GESTION DE LA COMUNICACIÓN LECTURA Y ESCRITURA DE DOS BITS, PARTE UNO Y DOS DEL PROGRAMA CLIENTE, CLIENTE-SERVIDOR Y SERVIDOR.	358
3.3 SEGUNDO PROGRAMA	359
3.3.1 CONFIGURACIÓN DE LA DIRECCIÓN IP:	360
3.3.1.1 DIRECCIÓN IP DEL PLC CLIENTE 1 (PLC 89).	360
3.3.1.2 DIRECCIÓN IP DEL PLC CLIENTE 2 (PLC 88).	361
3.3.2 MANEJO DEL SOFTWARE VIJEO DESIGNER	361
3.3.2.1 CREACIÓN DE UN PROYECTO EMPLEANDO EL SOFTWARE VIJEO	362
3.3.3 DESCRIPCION PROGRAMA LADDER CLIENTE 1 TWIDO 89	368
3.3.3.1 TABLAS DE LECTURA Y ESCRITURAS DE N PALABRAS	370
3.3.3.2 TABLA DE LECTURA N PALABRAS PARA LEER LA PALABRA %MW2 DEL SERVIDOR HMI	370
3.3.3.3 TABLA DE ESCRITURA DE N PALABRAS UTILIZADA PARA ESCRIBIR EN LAS PALABRAS %MW0 Y %MW1 DEL SERVIDOR (HMI MAGELIS)	371
3.3.3.4 GESTION DE LA COMUNICACIÓN PARA LA LECTURA DE UNA PALABRA Y ESCRITURA DE DOS PALABRAS.	372
3.3.4 DESCRIPCION PROGRAMA LADDER CLIENTE 2 TWIDO 88	373
3.3.4.1 TABLA DE ESCRITURA DE N PALABRAS	374
3.3.4.2 GESTIÓN DE LA COMUNICACIÓN CON LA INSTRUCCION EXCH3	374
3.4 TERCER PROGRAMA	375
3.4.1 CONFIGURACIÓN DE LA DIRECCIÓN IP:	376

3.4.1.1 DIRECCIÓN IP DEL PLC SERVIDOR 1 (TWIDO 88	376
3.4.1.2 DIRECCIÓN IP DEL PLC SERVIDOR 2 (PLC 89).	376
3.4.2 CONFIGURACION DEL CLIENTE MAGELIS CON SOFTWARE VIJEO DESIGNER	377
3.4.3 DESCRIPCION PROGRAMA LADDER SERVIDOR 1 TWIDO 88	383
3.4.4 DESCRIPCION PROGRAMA LADDER SERVIDOR 2 TWIDO 89	384
3.5 CUARTO PROGRAMA	385
3.5.1 CONFIGURACION DE LA RED TCP/IP PARA EL CLIENTE M340	385
3.5.1.1 CREE UN PROYECTO EN EL SOFTWARE UNITY PRO	385
3.5.1.2 CONFIGURE LA RED ETHERNET	391
3.5.1.3 PROGRAMA DEL CLIENTE M340:	394
3.5.1.4 PROGRAMA DEL CLIENTE-SERVIDOR HMI (Magelis)	405
3.5.1.5 PROGRAMA SERVIDOR:	409

LISTA DE FIGURAS

	Pág.
Figura 1: PUERTO 3	340
Figura 2: CONFIGURACIÓN DEL CLIENTE	341
Figura 3: DIRECCION IP PLC 88 CLIENTE	343
Figura 4: DIRECCION IP DEL PLC 89 CLIENTE-SERVIDOR	344
Figura 5: DIRECCION IP PLC 90 SERVIDOR	344
Figura 6: BITS 8 Y 9 DE LA PALABRA %MW18 QUE SE VAN A ESCRIBIR	346
Figura 7: BITS %M3 Y %M4 ESCRITOS POR EL CLIENTE	346
Figura 8: TABLA ESCRITURA DE N BITS PARA ESCRIBIR DOS BITS	347
Figura 9: BITS %M1 Y %M2 QUE SE VAN A LEER	349
Figura 10 BITS LEIDOS Y ALMACENADOS EN LOS BITS 8 Y 9 DE %MW17	349
Figura 11: TABLA DE LECTURA DE DOS BITS	350
Figura 12: BITS 0 Y 1 DE LA PALABRA %MW16 QUE SE VAN A ESCRIBIR	352
Figura 13: %MW1 ESCRITA CON EL VALOR DE LOS BITS 0 Y 1 DE %MW16	352
Figura 14: VENTANA DE LAS MACROS COMM	353
Figura 15: DESCRIPCION DE LA MACRO COMM	354
Figura 16: PALABRAS UTILIZADAS POR LA MACRO COMM	355
Figura 17: BITS 0 Y 1 DE LA PALABRA %MWO QUE SE VAN A LEER	356
Figura 18: VALORE LEIDOS Y ALMACENADOS EN LOS BITS DE %MW7	356
Figura 19: GESTION DE LA COMUNICACIÓN PRIMER PROGRAMA 3 Y 4	357
Figura 20: GESTION DE LA COMUNICACIÓN PRIMER PROGRAMA 1 Y 2	358
Figura 21: DIRECCIÓN IP DEL CLIENTE 1 (PLC 89)	360
Figura 22: DIRECCIÓN IP DEL PLC CLIENTE 2 (PLC 88)	361
Figura 23: VENTANA CREAR UN NUEVO PROYECTO	362
Figura 24: NOMBRE DEL PROYECTO	363
Figura 25: CARACTERÍSTICAS DE LA MAGELIS	363
Figura 26: DIRECCIÓN IP DE LA MAGELIS	364
Figura 27: CONFIGURACIÓN DE LA MAGELIS COMO ESCLAVO	365
Figura 28: VENTANA PRINCIPAL	366

Figura 29: PANELES DEL PROYECTO	367
Figura 30: VARIABLES DEL PROYECTO	367
Figura 31: CARACTERÍSTICAS DE LA CONEXION	368
Figura 32: PALABRA %MW0 QUE SE VA A ESCRIBIR	369
Figura 33: VALORES LEIDOS Y ALMACENADOS EN %MW9	370
Figura 34: PROGRAMACIÓN DE LA TABLA PARA LEER LA %MW2	371
Figura 35: TABLA PARA ESCRIBIR EN LAS PALABRAS %MW0 Y %MW1	371
Figura 36: CONTADOR DE PASOS	372
Figura 37: GESTION DE LA COMUNICACIÓN SEGUNDO PROGRAMA 1	373
Figura 38: PALABRA DE MEMORIA %MWO QUE SE VA A ESCRIBIR	373
Figura 39: TABLA PARA ESCRIBIR %MW0 EN %MW3	374
Figura 40: GESTION DE LA COMUNICACIÓN SEGUNDO PROGRAMA 2	374
Figura 41: DIRECCIÓN IP DEL SERVIDOR 1 (PLC 88)	376
Figura 42: DIRECCIÓN IP SERVIDOR 2 (PLC 89)	377
Figura 43: DESCRIPCION DEL UN PROYECTO	378
Figura 44: TIPO DE MAGELIS	378
Figura 45: DIRECCION IP DE LA HMI	379
Figura 46: MAGELIS COMO CLIENTE TCP/IP	380
Figura 47: VENTANA PRINCIPAL CARACTERÍSTICAS DE LA MAGELIS	380
Figura 48: PANELES DEL CLIENTE MAGELIS	381
Figura 49: VARIABLES DE LOS SERVIDORES	381
Figura 50: CARACTERÍSTICAS DE CONEXIÓN DEL MAESTRO HMI TCP/IP	382
Figura 51: PALABRA %MW0 DEL SERVIDOR LEÍDA POR LA MAGELIS	383
Figura 52: PALABRA %MW1 DEL SERVIDOR ESCLITA POR LA MAGELIS	384
Figura 53: VARIABLES DEL SERVIDOR LEIDAS POR EL CLIENTE	384
Figura 54: UNIDAD CENTRAL DE PROCESO BMX P342020	385
Figura 55: VENTANA EXPLORADOR DE PROYECTOS	386
Figura 56: VENTANA CON EL PLC M340	387
Figura 57: BASTIDOR DE 6 SLOTS CON REFERECIA BMX XBP 0600	387
Figura 58: FUENTE DE ALIMENTACIÓN CON RERENCIA BMXCPS 3500	388
Figura 59: MÓDULO BMX DDI3202K	389

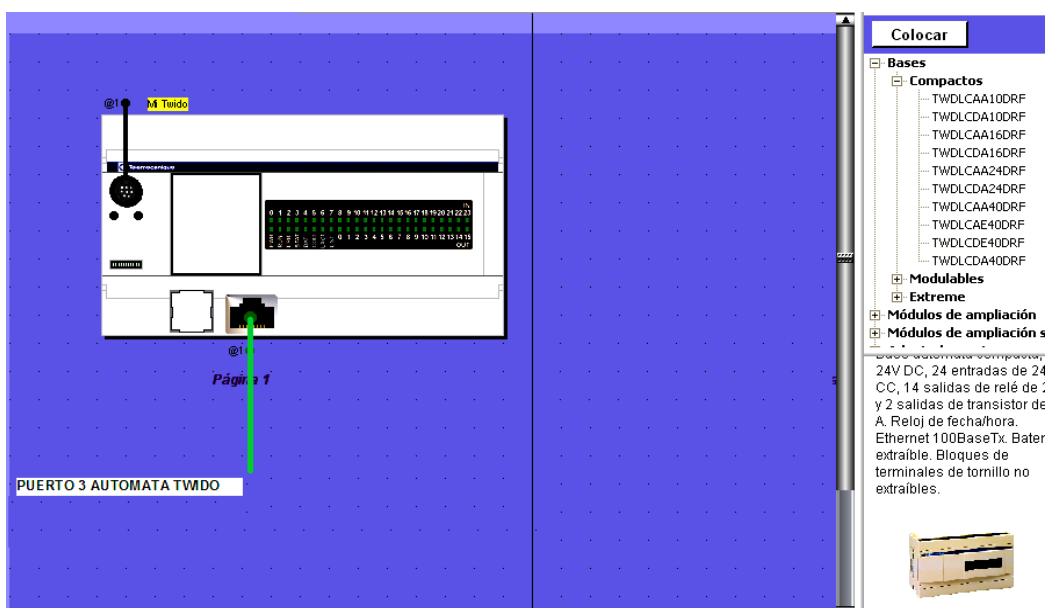
Figura 60: MÓDULO BMX DRA 1605	390
Figura 61: MÓDULO 2 BMX DRA 1650	390
Figura 62: MÓDULO BMX AMM 0600	391
Figura 63: CONFIGURACIÓN DE UNA RED ETHERNET	392
Figura 64: PARAMETROS A CONFIGULAR EN LA RED ETHERNET	393
Figura 65: PARÁMETROS CONFIGURADOS DE LA RED ETHERNET	393
Figura 66: CONFIGURACIÓN PROGRAMA DEL CLIENTE M340	394
Figura 67: CONFIGURACIÓN DEL BLOQUE WRITE_VAR	396
Figura 68: VALORES DE %MW1 QUE SE VAN A ESCRIBIR	399
Figura 69: BLOQUE READ_VAR	400
Figura 70: VALORES LEIDOS Y ALMACENADOS EN %MW0 DEL CLIENTE M340	402
Figura 71: BLOQUE WRITE_VAR PARA LA ESCRITURA DE %MW2 EN %MW0	403
Figura 72: VALORES DE LA PALABRA %MW2 DEL CLIENTE	404
Figura 73: CONFIGURACIÓN DEL MAGELIS COMO CLIENTE-SERVIDORA	406
Figura 74: REPRESENTACIÓN GRAFICA DEL PROCESO	406
Figura 75: VARIABLES DEL CLIENTE M340 Y DEL SERVIDOR M340	407
Figura 76: CONFIGURACIÓN MAGELIS COMO SERVIDORA	408
Figura 77: CONFIGURACIÓN MAGELIS COMO CLIENTE	408
Figura 78: VALORES ESCRITOS EN LA PALABRA %MW1 DEL SERVIDOR M340	410
Figura 79: VALORES %MW0 DEL SERVIDOR M340 QUE SE VAN A LEER	411
Figura 80: VALORES DE LA PALABRA %MW4 DEL SERVIDO M340	412
Figura 81: BITS %M1 Y %M2 DEL SERVIDOR M340	413
Figura 82: VARIABLES DEL PROGRAMA DEL CLIENTE M340	414
Figura 83: VARIABLES DEL PROGRAMA DEL SERVIDOR M340	414

3 GUIA PARA CREAR UNA APLICACIÓN MODBUS TCP-IP CON TWIDOSUITE

3.1 CONFIGURACIÓN DEL PUERTO 3

El puerto 3, tiene un conector RJ 45 hembra. Ver figura 1 de este documento. Por el cual se va a configurar modbus TCP-IP de la manera siguiente: dar doble clic en el puerto 3 para que se despliegue la ventana de la figura.2

Figura 1: PUERTO 3



En la figura 2 se configura la dirección IP. Las redes TCP/IP pueden ser de clase A, clase B y clase c. El autómata twido utiliza una dirección IP clase C. Por tal motivo los tres primeros números de IP Address son los utilizados para identificar la red y el ultimo es el encargado de identificar el equipo.

Figura 2: CONFIGURACIÓN DEL CLIENTE

Ethernet networks

IP address configuration

☐ From a server IP address: 192 . 168 . 71 . 88
☒ Configured Subnet mask: 255 . 255 . 255 . 0
 Gateway address: 192 . 168 . 71 . 1

Marked IP

☐ Specify a marked IP address Specify an IP address for the marked connection. . . .

Idle time

NOTE: The controller will detect the active and passive TCP connections and will close those that are inactive for the time indicated here. If the maximum idle time is set to 0, the controller will not run the detection.

Enter time out for the TCP connection.
 10 min(s) Default

Remote Devices

Index	Slave IP address	Unit ID	Connection time out (100 ms)
1	192.168.71.89	255	100
2			
3			
4			

OK Cancel

La figura 2 presenta la configuración de la dirección del PLC 88 (cliente). La dirección IP clase C tiene una máscara de subred con el valor 255, 255, 255, 0. El valor de la dirección IP del Gateway Address no se configura en este documento debido a que no se empleo ningún Gateway, si existía un Gateway se colocaría la dirección IP de este. En la figura 2, se debe configurar el tiempo de espera (Enter time out for the connection), que es el tiempo dado para que el equipo responda, este tiempo generalmente es de 10 ms, a demás se debe configurar la dirección IP del equipo servidor "Slave IP Address" con el cual se va a comunicar el equipo maestro. Si se desea configurar el autómatas como servidor no se configura la dirección IP en dispositivos remotos "Remote Devices".

Para que el protocolo de comunicación modbus TCP-IP funcione adecuadamente en el autómata Twido se debe emplear un lenguaje de programación para que el PLC lo interprete y desarrolle las tareas programadas por el usuario. En este documento se empleara el lenguaje de programación ladder. Se debe aclarar que el protocolo modbus TCP/IP emplea el modelo de comunicación tipo Cliente / Servidor

3.2 PRIMER PROGRAMA

PROGRAMA QUE EJECUTA LA COMUNICACIÓN ENTRE UN PLC CLIENTE TWIDO, UN PLC TWIDO CLIENTE-SERVIDOR Y UN PLC TWIDO SERVIDOR

El programa Ladder está compuesto por tres programas, uno para cada PLC Twido: programa cliente del PLC 88 empleado para la comunicación con el programa servidor del PLC 89. Programa cliente-servidor del PLC 89 elaborado para la comunicación con el programa cliente PLC 88 y el programa servidor del PLC 90. Programa servidor del PLC 90 elaborado para que exista comunicación con el programa cliente del PLC 89. A

3.2.1 CONFIGURACIÓN DE LA DIRECCIÓN IP:

3.2.1.1 DIRECCIÓN IP DEL PLC 88. El PLC 88 tiene una dirección IP (192.168.71.88) y va a ser el cliente del PLC 89 con dirección IP (192.168.71.89), En la figura 3 se observa esta configuración.

3.2.1.2 DIRECCIÓN IP DEL PLC 89. El PLC 89 con dirección IP (192.168.71.89) va a ser el cliente del PLC 90 con dirección IP (192.168.71.90) y a su vez servidor del PLC 88.con dirección IP (192.168.71.88) . Ver figura 4.

3.2.1.3 DIRECCIÓN IP DEL PLC 90. El PLC 90 va a ser el servidor del PLC 89. En la figura 5 se observa la configuración de su dirección IP.

Figura 3: DIRECCION IP PLC 88 CLIENTE

Ethernet networks

IP address configuration

☐ From a server IP address: 192 . 168 . 71 . 88
☒ Configured Subnet mask: 255 . 255 . 255 . 0
Gateway address: 192 . 168 . 71 . 1

Marked IP

☐ Specify a marked IP address Specify an IP address for the marked connection. . . .

Idle time

NOTE: The controller will detect the active and passive TCP connections and will close those that are inactive for the time indicated here. If the maximum idle time is set to 0, the controller will not run the detection.

Enter time out for the TCP connection.

10 min(s) Default

Remote Devices

Index	Slave IP address	Unit ID	Connection time out (100 ms)
1	192.168.71.89	255	100
2			
3			
4			

OK Cancel

Figura 4: DIRECCION IP DEL PLC 89 CLIENTE-SERVIDOR

Ethernet networks

IP address configuration

☐ From a server IP address: 192 . 168 . 71 . 89
☒ Configured Subnet mask: 255 . 255 . 255 . 0
Gateway address: 192 . 168 . 71 . 1

Marked IP

☐ Specify a marked IP address Specify an IP address for the marked connection. . . .

Idle time

NOTE: The controller will detect the active and passive TCP connections and will close those that are inactive for the time indicated here. If the maximum idle time is set to 0, the controller will not run the detection.

Enter time out for the TCP connection.
10 min(s) Default

Remote Devices

Index	Slave IP address	Unit ID	Connection time out (100 ms)
1	192.168.71.90	255	100
2			
3			
4			

OK Cancel

Figura 5: DIRECCION IP PLC 90 SERVIDOR

Configuration

Ethernet networks

IP address configuration

☐ From a server IP address: 192 . 168 . 71 . 90
☒ Configured Subnet mask: 255 . 255 . 255 . 0
Gateway address: 192 . 168 . 71 . 90

Marked IP

☐ Specify a marked IP address Specify an IP address for the marked connection. . . .

Idle time

NOTE: The controller will detect the active and passive TCP connections and will close those that are inactive for the time indicated here. If the maximum idle time is set to 0, the controller will not run the detection.

Enter time out for the TCP connection.
10 min(s) Default

Remote Devices

Index	Slave IP address	Unit ID	Connection time out (100 ms)
1			
2			
3			
4			

OK Cancel

3.2.2 PROGRAMA LADDER: CLIENTE, CLIENTE-SERVIDOR Y SERVIDOR

Este programa hace lo siguiente: Las entradas I0.1 y I0.2 del cliente (Cliente 88) ponen en 0 o 1 los valores de los bits 3 y 4 (%M3 y %M4) del cliente-servidor (PLC 89) que a su vez activan las salidas 3 y 4 (%Q0.3 y %Q0.4) del cliente-servidor. Las entradas 1 y 2 (%I0.1 y %I0.2) del cliente-servidor activan/desactivan los bits 1 y 2 (%M1 y %M2) del cliente-servidor que a su vez activan/desactivan las salidas Q0.1 y Q0.2 del cliente. Dicho de otra forma este programa hace que el cliente lea los bits %M1 y %M2 del cliente-servidor y escriba en los bits %M3 y %M4 del cliente-servidor. Las entradas I0.1 y I0.2 del servidor (PLC 90) activan/desactivan los bits 0 y 1 de %MW0 del servidor (palabra leída por el cliente-servidor) que a su vez activan/desactivan las salidas Q0.5 y Q0.6 del cliente.

Las entradas I0.3 y I0.4 del cliente-servidor (Cliente-Servidor 89) activan/desactivan los bits 0 y 1 de %MW1 del servidor (palabra escrita por el cliente-servidor) para activar la salida Q0.2 del servidor y desactivar la salida Q0.4 del servidor. O dicho de otra forma, cuando se activa/desactiva la entrada I0.3 del cliente-servidor, se activa/desactiva la salida Q0.2 del servidor; cuando se activa/desactiva la entrada I0.4 del cliente-servidor, se desactiva/activa la salida Q0.4 del servidor. En resumen, el cliente-servidor lee %MW0 del servidor y escribe en %MW1 del servidor. Este programa se divide en cuatro partes:

3.2.2.1 PRIMERA PARTE: ACCIONES QUE REALIZA EL CLIENTE (TWIDO 88) EN EL CLIENTE-SERVIDOR (TWIDO 89):

En esta primera parte las entradas 1 y 2 del cliente (PLC 88) ponen en 0 o 1 los valores de los bits %M3 y %M4 del cliente-servidor (PLC 89) para que estos bits activen las salidas %Q0.3 y %Q0.4 del cliente-servidor. Dicho de otra forma, la palabra de memoria %MW18 del cliente escribe el valor de sus bits 8 y 9 en los bits %M3 y %M4 del cliente-servidor utilizando la tabla de escritura de N bits. Ver figuras 6, 7 y 8.

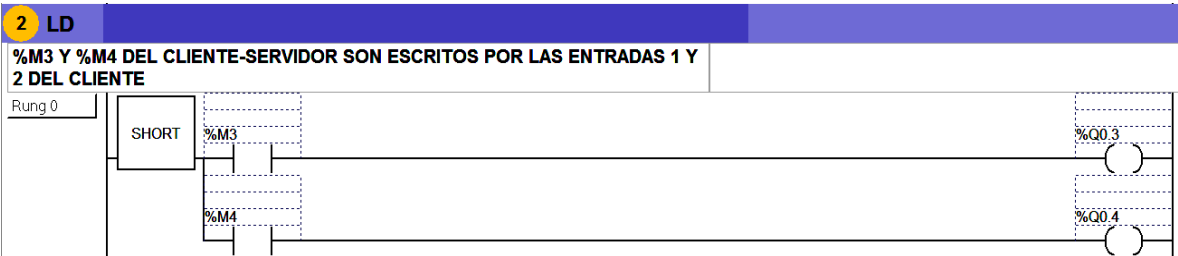
PROGRAMA DEL CLIENTE

Figura 6: BITS 8 Y 9 DE LA PALABRA %MW18 QUE SE VAN A ESCRIBIR



PROGRAMA DEL CLIENTE-SERVIDOR:

Figura 7: BITS %M3 Y %M4 ESCRITOS POR EL CLIENTE



PROGRAMA DEL CLIENTE

La figura 8 presenta la programación de la tabla de N bits para escribir dos bits, a continuación se explica el significado de cada palabra de memoria %MWx

Figura 8: TABLA ESCRITURA DE N BITS PARA ESCRIBIR DOS BITS

ESCRITURA DE 2 BITS	
Rung 1	SHORT
	%MW8 := 16#0109 %MW8 := 16#0109
	%MW9 := 16#0007 %MW9 := 16#0007
	%MW10 := 16#010F %MW10 := 16#010F
	%MW11 := 16#0003 %MW11 := 16#0003
	%MW12 := 16#0002 %MW12 := 16#0002
	%MW13 := 16#0001 %MW13 := 16#0001
	%MW14 := %MW18 %MW14 := %MW18

Palabra 0: %MW8= 16#0109. Esta palabra corresponde a la subtabla de control en la tabla de escritura de N bits. Esta palabra está compuesta por dos bytes, el byte más significativo corresponde al código de emisión/recepción = 01 y el byte menos significativo corresponde a: 8 + número de bytes (emisión). Para este caso es igual a 9 por que se envía solo un byte.

Palabra 1:%MW9 = 16#0007. Esta palabra corresponde a la subtabla de control. La palabra tiene dos bytes, el más significativo se utiliza para el offset de recepción = 00 y el menos significativo para el offset de emisión =07

Palabra 2: %MW10 = 16#010F. Esta palabra corresponde a la subtabla de envío y tiene dos bytes, el más significativo se utiliza para la dirección del esclavo en este caso 01 y el byte menos significativo para el código de solicitud de la tabla de escritura de N bits que es igual a 0F hexadecimal.

Palabra 3: %MW11 = 16#0003. Esta palabra corresponde a la subtabla de envío y contiene la dirección del primer bit que se va a escribir en el servidor. En este ejercicio es igual 0003, o sea se va a escribir en el bit %M3 del servidor

Palabra 4: %MW12 = 16#0002. Corresponde a la subtabla de envío y contiene el número de bits que se van a escribir. En este ejercicio se van a escribir dos bits (0002)

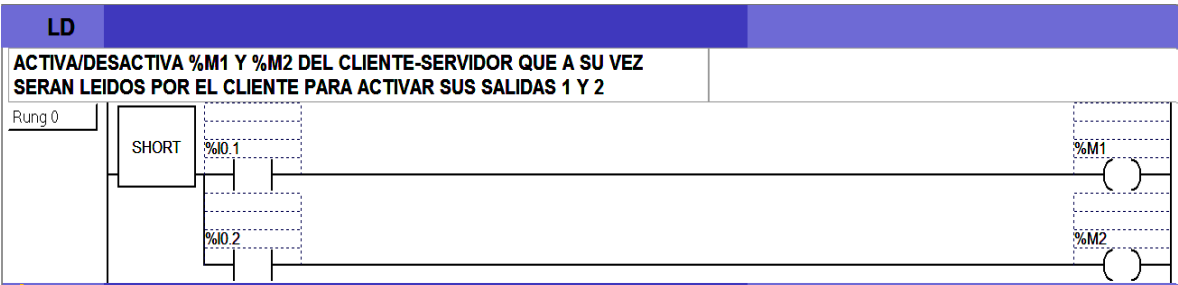
Palabra 5: %MW13 = 16#0001. Corresponde a la subtabla de envío y contiene en el byte más significativo el byte no enviado por efecto del offset = 00 y en el byte menos significativo el numero de bytes que se van a escribir. Para este ejemplo 01 byte (8 bits).de los cuales solo se utilizan 2 bits.

Palabra 6: %MW14 = %MW18. Corresponde a la subtabla de envío de la tabla de escritura de N bits y contiene el valor del primer byte que se va a escribir en los bits del programa del servidor.

3.2.2.2 SEGUNDA PARTE DEL PROGRAMA: ACCIONES QUE REALIZA EL CLIENTE-SERVIDOR EN EL CLIENTE: En esta segunda parte las entradas %I0.1 y %I0.2 del cliente-servidor (PLC 89) activan/desactivan los bits %M1 y %M2 del cliente-servidor (PLC 89) que a su vez activan/desactivan las salidas %Q0.1 y %Q0.2 del cliente (PLC 88). Dicho de otra forma el cliente lee los bits %M1 y %M2 del cliente-servidor con el código de solicitud 01 “lectura de N bits” para activar sus salida %Q0.1 y %Q0.2. Ver figuras 9, 10 y 11.

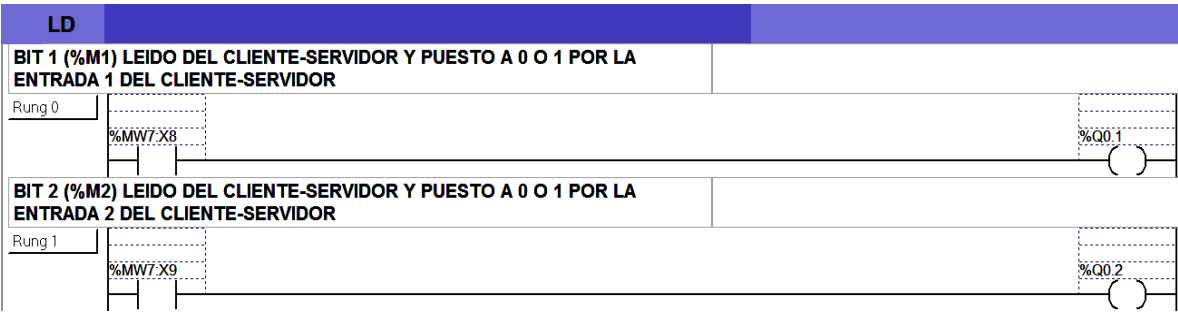
PROGRAMA DEL CLIENTE-SERVIDOR (TWIDO 89)

Figura 9: BITS %M1 Y %M2 QUE SE VAN A LEER



PROGRAMA DEL CLIENTE (TWIDO 88)

Figura 10 BITS LEIDOS Y ALMACENADOS EN LOS BITS 8 Y 9 DE %MW17



PROGRAMA CLIENTE

Figura 11: TABLA DE LECTURA DE DOS BITS

3 LD Tablas de comunicacion	
LECTURA DE 2 BITS	
Rung 0	SHORT %MW0 = 16#0106 %MW0 = 16#0106 %MW1 = 16#0300 %MW1 = 16#0300 %MW2 = 16#0101 %MW2 = 16#0101 %MW3 = 16#0001 %MW3 = 16#0001 %MW4 = 16#0002 %MW4 = 16#0002

DESCRIPCIÓN DE LA TABLA DE LECTURA N BITS PARA LEER DOS BITS

Palabra 0: %MW0 = 16#0106. Esta palabra corresponde a la subtabla de control utilizada para controlar la tabla de envío y recepción dentro de la tabla de lectura para N bits. Esta palabra la conforma dos bytes, el byte más significativo corresponde al código de emisión/recepción = 01 y el byte menos significativo corresponde a la longitud de la tabla de envío y es igual a 06

Palabra 1: %MW1 = 16#0300. Corresponde a la subtabla de control y contiene el offset de recepción en el byte más significativo = 03 y el offset de envío en el byte menos significativo = 00.

Palabra 2: %MW2 = 16#0101 corresponde a la subtabla de envío en la tabla de lectura y contiene la dirección del esclavo y el código de solicitud de la tabla de lectura de N bits respectivamente.

Palabra 3: %MW3 = 16#0001. Corresponde a la subtabla de envío y contiene la dirección del primer bit que se va a leer. 0001. en este caso

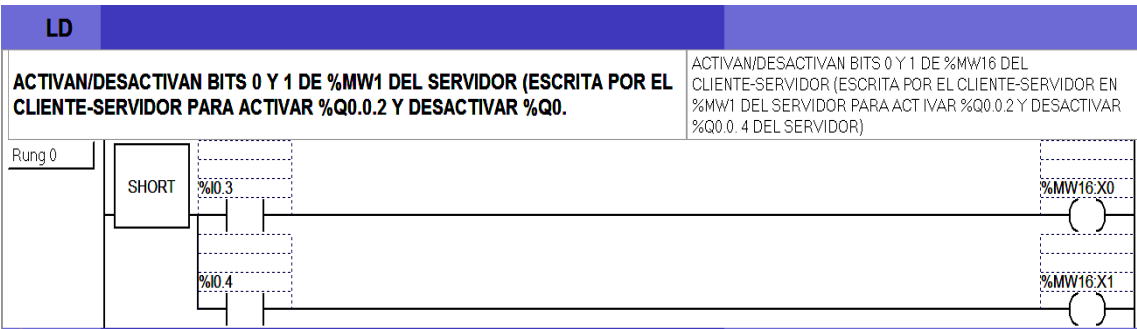
Palabra 4: %MW4 = 16#0002. Corresponde a la subtabla de envío y contiene el número de bits que se van a leer. En este ejercicio se utilizaron 2 bits

Nota: la subtabla de recepción de la tabla de lectura de N bits la genera el programa. Ver la sección 3.10.2.4 del documento redes de comunicación industrial prácticas de laboratorio, por este motivo se asignan palabras para que el programa responda en la tabla de recepción. Estas palabras deben ser continuas después de la última palabra de la tabla de envío, ósea para este caso debe ser **%MW5, %MW6 y %MW7**. La palabra **%MW7** contiene el byte donde están los dos bits que va leer el cliente para activar o desactivar las salidas **%Q0.1 y %Q0.2**.

3.2.2.3 TERCERA PARTE DEL PROGRAMA: ACCIONES QUE REALIZA EL CLIENTE-SERVIDOR (TWIDO 89) EN EL SERVIDOR (TWIDO 90): En esta parte del programa las entradas **%I0.3 y %I0.4** del cliente-servidor (PLC 89) activan/desactivan los bits 0 y 1 de la palabra de memoria **%MW16** del servidor (palabra escrita por el cliente-servidor en **% MW1** del servidor) para activar la salida **%Q0.2** del servidor y desactivar la salida **%Q0.4** del servidor.

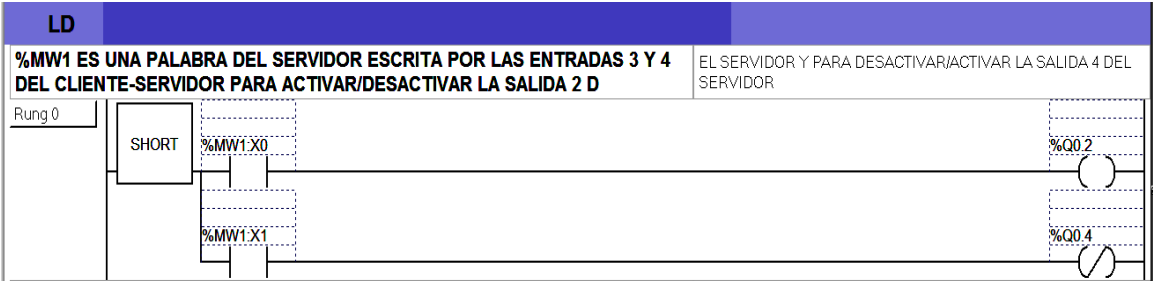
PROGRAMA DEL CLIENTE-SERVIDOR

Figura 12: BITS 0 Y 1 DE LA PALABRA %MW16 QUE SE VAN A ESCRIBIR



PROGRAMA DEL SERVIDOR

Figura 13: %MW1 ESCRITA CON EL VALOR DE LOS BITS 0 Y 1 DE %MW16

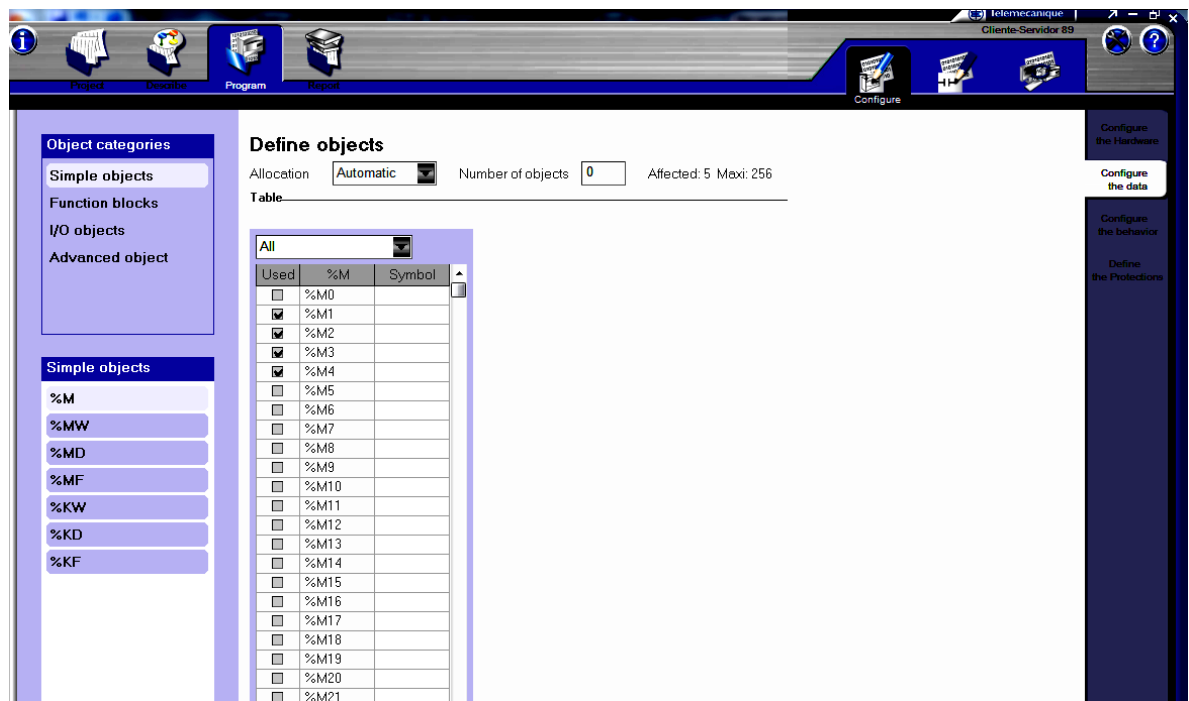


Para que el cliente pueda escribir los bits 0 y 1 de la palabra de memoria %MW16 en los bits 0 y 1 de la palabra %MW1 es necesario programar una tabla de escritura como ya se detallo en la primera parte de este programa o configurar en el programa Twidosuite las macros comm que implícitamente elaboran las tablas de escritura y lectura para palabras y/o bits. En esta parte del programa se utilizaran las macros comm

ESCRITURA DE DOS BITS UTILIZANDO LOS MACROS COM

Si se desea configurar las macros comm se debe seguir los siguientes pasos: se debe ir a programar en la barra principal del Twidosuite para localizar la subventana donde se elije configurar y en la columna que está al lado de la subventana se debe elegir configurar los datos. Se desplegara la siguiente ventana:

Figura 14: VENTANA DE LAS MACROS COMM



En categoría de objetos, elegimos objetos avanzados y en objetos avanzados elegimos macros comm, se desplegará una lista de macros com donde se debe

seleccionar alguna macro, para este ejemplo se eligió la macro comm 1 Ver figura 14.

Figura 15: DESCRIPCION DE LA MACRO COMM

Object categories

Simple objects

Function blocks

I/O objects

Advanced object

Advanced object

%SCH

PID

Macros Comm

Macros Drive

Define objects

AllocationAutomaticNumber of objects0Affected: 2 Maxi: 32

Table

All

Used	Macro	Conf.
☐	Comm 0	☐
☒	Comm 1	☒
☐	Comm 2	☐
☐	Comm 3	☐
☐	Comm 4	☐
☐	Comm 5	☐
☐	Comm 6	☐
☐	Comm 7	☐
☐	Comm 8	☐
☐	Comm 9	☐
☐	Comm 10	☐
☐	Comm 11	☐
☐	Comm 12	☐
☐	Comm 13	☐
☐	Comm 14	☐
☐	Comm 15	☐

ApplyCancel

Help with Comm macros

Configured

NetworkEthernet

Network address1

Functions

☐PRE Function☐POST Function

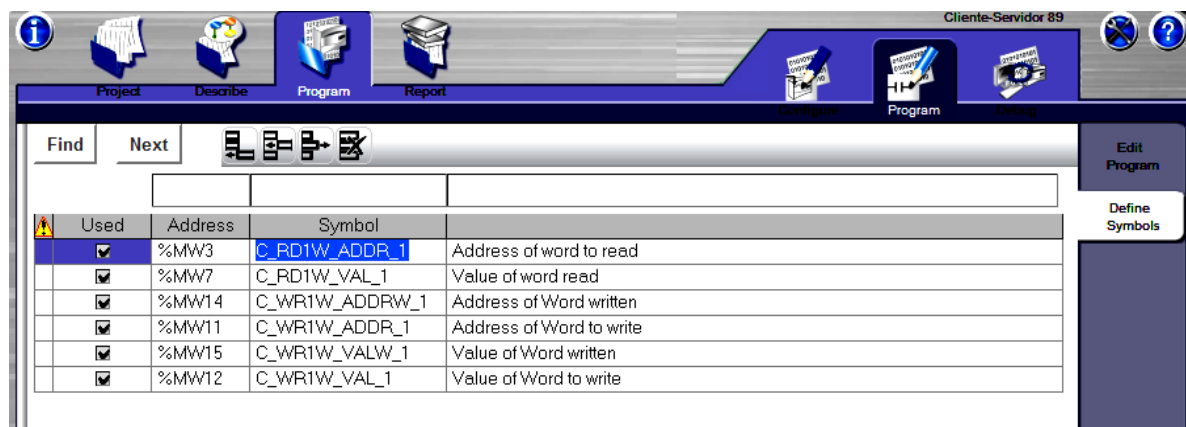
Function name	Start address	Number of words	Symbols
C_RD1B	0	8	☐
C_RD1W	0	8	☒
C_WR1B	0	8	☐
C_WR1W	8	8	☒
C_RDNW	0	N+7	☐
C_WRNW	0	N+9	☐

La macro que se seleccionó se tiene que configurar dependiendo que tarea se desee realizar con ella. En este caso se seleccionó la función del macro comm C_RD1W, que significa lectura de una palabra y se utilizará en la cuarta parte de este programa para que el cliente-servidor (PLC 89) lea los dos bits 0 y 1 de la palabra %MW0 del servidor (PLC 90) que a su vez activan/desactivan las salidas Q0.5 y Q0.6 del cliente-servidor. También se selecciono C_WR1W que se utilizará para la escritura de una palabra que será la empleada en desarrollar la comunicación para escribir los bits 0 y 1 de la palabra %MW16 del cliente-servidor en la palabra % MW1 del servidor en tercera parte de este programa. La figura 15 presenta este tipo de configuración, observe que en la tabla de funciones de esta figura aparecen las siguientes funciones: C_RD1B “lectura de un bit”, C_RD1W “lectura de una palabra”, C_WR1B “escritura de un bit”, C_WR1W “escritura de una palabra”, C_RDNW “lectura de N palabras”, C_WRNW “escritura de N

palabras”. Cada función se le debe reservar 8 palabras consecutivas indicando la dirección de la primera palabra “Start address” excepto a las funciones C_RDNW y C_WRNW que se le asignan una cantidad fija de palabras “7 para CRDNW y 9 para C_WRNW” + N que es el numero de palabras que se van a escribir o a leer. “a C_WRNW y C_RDNW también se le debe indicar la dirección de la primera palabra reservada”

Si se desea ver que palabras de memoria %MWx utiliza esta macro comm para gestionar el proceso de escritura y lectura se va a programar, programar y definir símbolos. Ver figura 16 **NOTA:** a la función C_RD1W se le reservaron 8 palabras consecutivas en el programa del cliente-servidor, comenzando en la palabra de memoria %MW0 y a la función C_WR1W se le reservaron 8 palabras consecutivas en el programa del cliente-servidor, comenzando en la palabra de memoria %MW8.

Figura 16: PALABRAS UTILIZADAS POR LA MACRO COMM



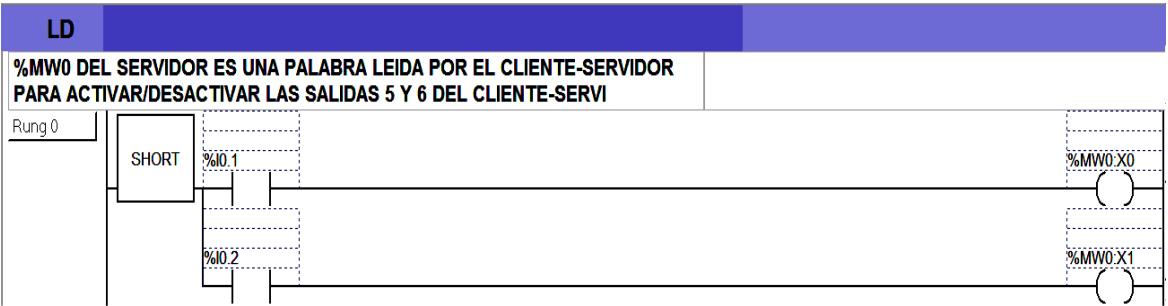
Used	Address	Symbol	
☒	%MW3	C_RD1W_ADDR_1	Address of word to read
☒	%MW7	C_RD1W_VAL_1	Value of word read
☒	%MW14	C_WR1W_ADDRW_1	Address of Word written
☒	%MW11	C_WR1W_ADDR_1	Address of Word to write
☒	%MW15	C_WR1W_VALW_1	Value of Word written
☒	%MW12	C_WR1W_VAL_1	Value of Word to write

3.2.2.4 CUARTA PARTE DEL PROGRAMA: ACCIONES QUE REALIZA EL SERVIDOR (TWIDO 90) EN EL CLIENTE-SERVIDOR (TWIDO 89): Esta parte del programa ejecuta las siguientes acciones: cuando se activan las entradas I0.1 y I0.2 del servidor (Servidor 90) se activan los bits 0 y 1 de la palabra de memoria

%MW0 del servidor que serán leídos en la palabra de memoria %MW7 por el cliente-servidor para activar/desactivar las salidas Q0.5 y Q0.6 del cliente-servidor.

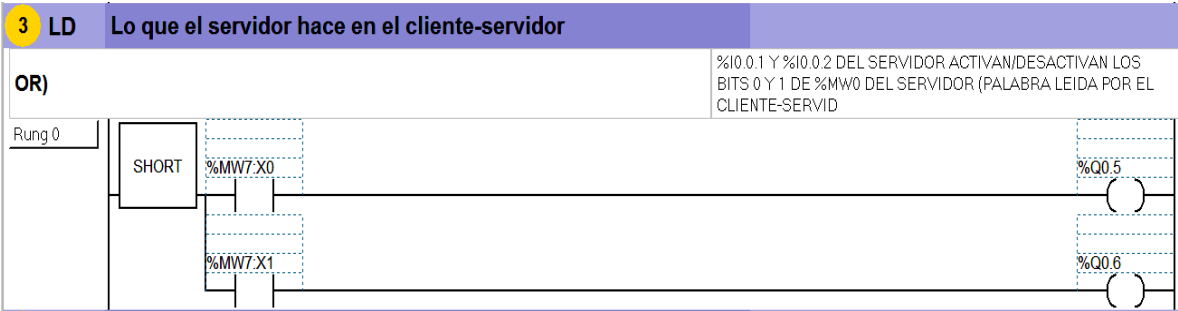
PROGRAMA DEL SERVIDOR

Figura 17: BITS 0 Y 1 DE LA PALABRA %MWO QUE SE VAN A LEER



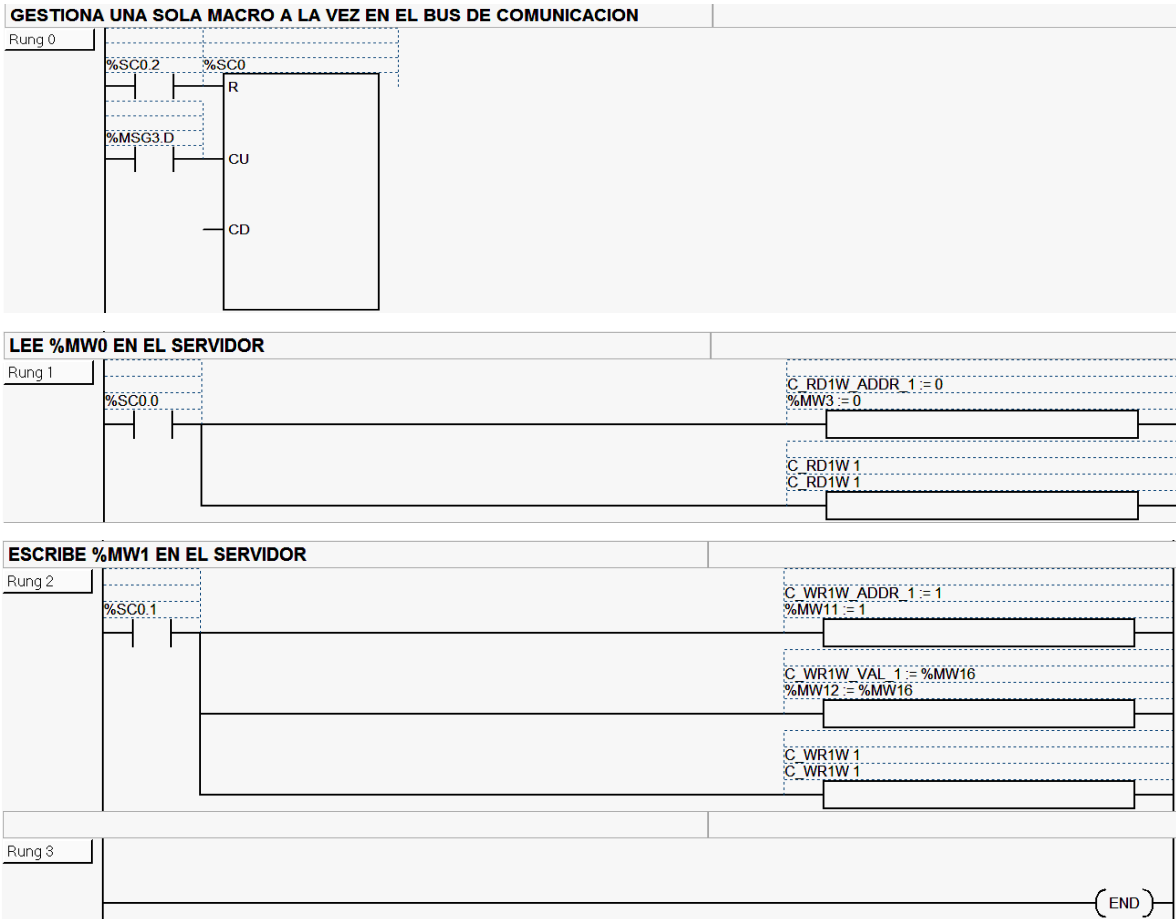
PROGRAMA DEL CLIENTE-SERVIDOR

Figura 18: VALORE LEIDOS Y ALMACENADOS EN LOS BITS DE %MW7



3.2.2.5 GESTION DE LA COMUNICACIÓN PARA LA PARTE TRES Y CUATRO DEL PROGRAMA UTILIZANDO MACROS COM

Figura 19: GESTION DE LA COMUNICACIÓN PRIMER PROGRAMA 3 Y 4

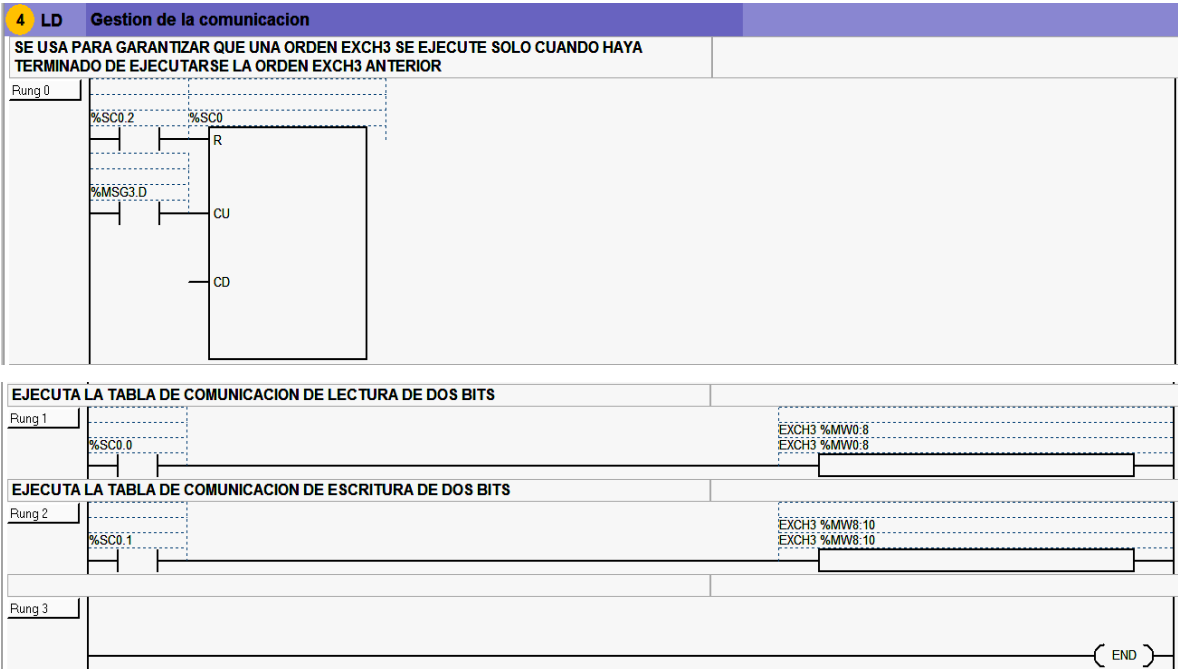


En la figura 19 se observa el contador de pasos con el bit de entrada `%MSG3.D` utilizado para indicar que ha finalizado el intercambio de las primer grupo de funciones de la macro com, antes de ejecutarse una segunda grupo de funciones de la macro comm. El contador de pasos ejecuta dos pasos y se resetea en el tercero “SC0.2”, en el paso SC0.0 ejecuta el grupo de funciones de la macro para

la lectura de una palabra y en el paso SC0.1 ejecuta el grupo de funciones de la macro para la escritura de una palabra.

3.2.2.6 GESTION DE LA COMUNICACIÓN LECTURA Y ESCRITURA DE DOS BITS, PARTE UNO Y DOS DEL PROGRAMA CLIENTE, CLIENTE-SERVIDOR Y SERVIDOR.

Figura 20: GESTION DE LA COMUNICACIÓN PRIMER PROGRAMA 1 Y 2



La gestión de la comunicación se hace a través de la instrucción **EXCHx** y el bit %MSGx.D “x= 1 modbus serial y x= 2 modbus TCP/IP” que se implementa a la entrada del contador de pasos y sirve para controlar el intercambios de las

instrucción EXCHx. El intercambio de instrucciones EXCHx consiste en que una instrucción EXCHx debe finalizar antes que comience otra instrucción EXCHx.

La instrucción EXCHx se utiliza para gestionar los mensajes de la tabla de palabras de lectura y escritura y se selecciona de la siguiente forma:

[EXCHx %MWi: L]

Donde: x = número de puerto (3). Para la comunicación por Ethernet

L = número de palabras de la tabla de palabras de control, de envío y de recepción. Para este caso 8 comenzando con la palabra %MW0 para la tabla de lectura de N bits y 10 comenzando en la palabra %MW8 para la tabla de escritura de N bits. Ver figura 33

3.3 SEGUNDO PROGRAMA

PROGRAMA PARA LA COMUNICACIÓN ENTRE DOS PLCs CLIENTES TWIDO Y UN SERVIDOR HMI MAGELIS

Este programa en twidosuite lo conforma dos programas en lenguaje ladder que son los encargados de comunicar los dos PLCs clientes Twido con el programa servidor que maneja la HMI Magelis. En general el programa hace lo siguiente:

El programa escribe %MW0 del cliente 1 (PLC 89) en %MW0 del servidor (HMI Magelis) para indicar si alguna de las entradas %Q0.1, %Q0.2 o %Q0.3 del cliente 1 (PLC 89) está activada o si ninguna lo está. Esto se puede observar en el panel 1 de la Magelis, luego escribe %IW0.0.0 del cliente 1 en %MW1 del servidor para mostrarla en un tacómetro (Panel 2) del servidor. También lee %MW2 del servidor para indicarle al cliente 1 (Twido 89) cuál de sus salidas % Q0.1, Q0.2 o Q0.3 está activada. Estas salidas del cliente 1 (PLC 89) se pueden observar en el

panel 3 de la HMI. Por último la entrada IO.1 del cliente 2 (PLC 88) activa/desactiva el primer bit de la palabra de memoria %MW0 en el cliente, para que esta palabra sea escrita en la palabra de memoria %MW3 del servidor (HMI Magelis) para activar/desactivar una sirena y alternar entre dos mensajes (uno de "estado normal" y otro de "temperatura critica") en el panel 3 del servidor (Magelis)

3.3.1 CONFIGURACIÓN DE LA DIRECCIÓN IP:

3.3.1.1 DIRECCIÓN IP DEL PLC CLIENTE 1 (PLC 89). El PLC Twido 89 va a ser el cliente 1 de la HMI (Magelis). En la figura 21 se observa la dirección IP clase C del cliente configurada como 192.168.71.89 y la dirección IP clase C 192.168.71.84 del servidor (Magelis).

Figura 21: DIRECCIÓN IP DEL CLIENTE 1 (PLC 89)

The screenshot shows a 'Configuration' window with the following sections:

- Ethernet networks:**
 - IP address configuration:**
 - ☐ From a server
 - ☒ Configured
 - IP address: 192 . 168 . 71 . 89
 - Subnet mask: 255 . 255 . 255 . 0
 - Gateway address: 192 . 168 . 71 . 89
 - Marked IP:**
 - ☐ Specify a marked IP address
 - Specify an IP address for the marked connection: . . .
 - Idle time:**
 - NOTE: The controller will detect the active and passive TCP connections and will close those that are inactive for the time indicated here. If the maximum idle time is set to 0, the controller will not run the detection.
 - Enter time out for the TCP connection: 10 min(s) [Default]
 - Remote Devices:**

Index	Slave IP address	Unit ID	Connection time out (100 ms)
1	192.168.71.84	255	100
2			
3			
4			

At the bottom are 'OK' and 'Cancel' buttons.

3.3.1.2 DIRECCIÓN IP DEL PLC CLIENTE 2 (PLC 88). El PLC Twido 88 va a ser el cliente 2 de la HMI (Magelis). En la figura 22 se observa la dirección IP “192.168.71.88” del cliente (PLC 88) y la dirección IP “192.168.71.84” del servidor (Magelis).

Figura 22: DIRECCIÓN IP DEL PLC CLIENTE 2 (PLC 88)

Ethernet networks _____

IP address configuration _____

☐ From a server IP address:

☒ Configured Subnet mask:

 Gateway address:

Marked IP _____

☐ Specify a marked IP address Specify an IP address for the marked connection.

Idle time _____

NOTE: The controller will detect the active and passive TCP connections and will close those that are inactive for the time indicated here. If the maximum idle time is set to 0, the controller will not run the detection.

Enter time out for the TCP connection.

min(s)

Remote Devices _____

Index	Slave IP address	Unit ID	Connection time out (100 ms)
1	192.168.71.84	255	100
2			
3			
4			

3.3.2 MANEJO DEL SOFTWARE VIJEO DESIGNER

Para configurar la HMI (Magelis) es necesario tener conocimiento en el manejo del software VIJEO DESIGNER. A continuación se ilustra como utilizar el software VIJEO para crear una aplicación cuando la Magelis actúe como servidora

3.3.2.1 CREACIÓN DE UN PROYECTO EMPLEANDO EL SOFTWARE VIJEI

Para crear un proyecto en Vijeo Designer se debe abrir el software en el PC y configurar la ventana que se despliega de la siguiente manera: se elija que acción crear un nuevo proyecto. Ver figura 23

Figura 23: VENTANA CREAR UN NUEVO PROYECTO



Dar clic en siguiente para configurar el nombre del proyecto, descripción o comentarios, tipo de proyecto y contraseña del proyecto. Ver la siguiente figura 24

Figura 24: NOMBRE DEL PROYECTO



Crear un nuevo proyecto

Introduzca el nombre del proyecto a crear

Nombre del proyecto prueba con Twido

Descripción o comentario

proyecto que comunica dos Twidos clientes con una magelis servidora

Tipo

☐ Proyecto con un sólo destino

☒ Proyecto con 1 destinos

☒ Contraseña del proyecto

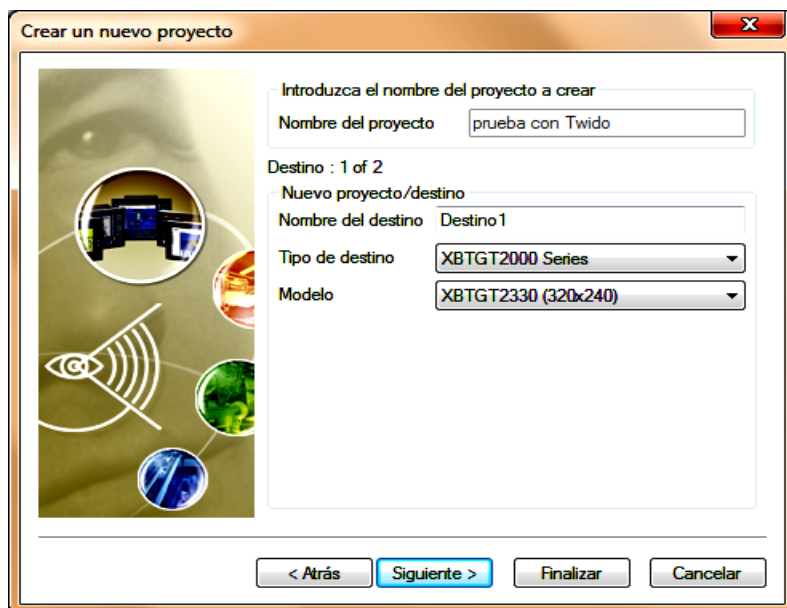
Ingrese contraseña

Confirme contraseña

Recordar (opcional)

< Atrás Siguiente > Finalizar Cancelar

Figura 25: CARACTERÍSTICAS DE LA MAGELIS



Crear un nuevo proyecto

Introduzca el nombre del proyecto a crear

Nombre del proyecto prueba con Twido

Destino : 1 of 2

Nuevo proyecto/destino

Nombre del destino Destino1

Tipo de destino XBTGT2000 Series

Modelo XBTGT2330 (320x240)

< Atrás Siguiente > Finalizar Cancelar

En la figura 24 en opción Tipo, se puede elegir con un destino o con más de un destino, el destino hace referencia al número de Magelis empleadas en una red. En esta misma ventana es a voluntad del usuario configurar la contraseña. Siguiendo con la creación de un nuevo proyecto se hace clic en siguiente para configurar en la ventana de la figura 25. El nombre del destino, uno si hay una sola Magelis. Tipo de destino, es la referencia de la Magelis y el modelo “modelo de la Magelis”.

Clic en siguiente para configurar en la ventana de la figura 26 la dirección IP de la Magelis. La Magelis que tiene el almacén de E3T tiene una dirección IP clase C igual a 192.168.71.84

Figura 26: DIRECCIÓN IP DE LA MAGELIS

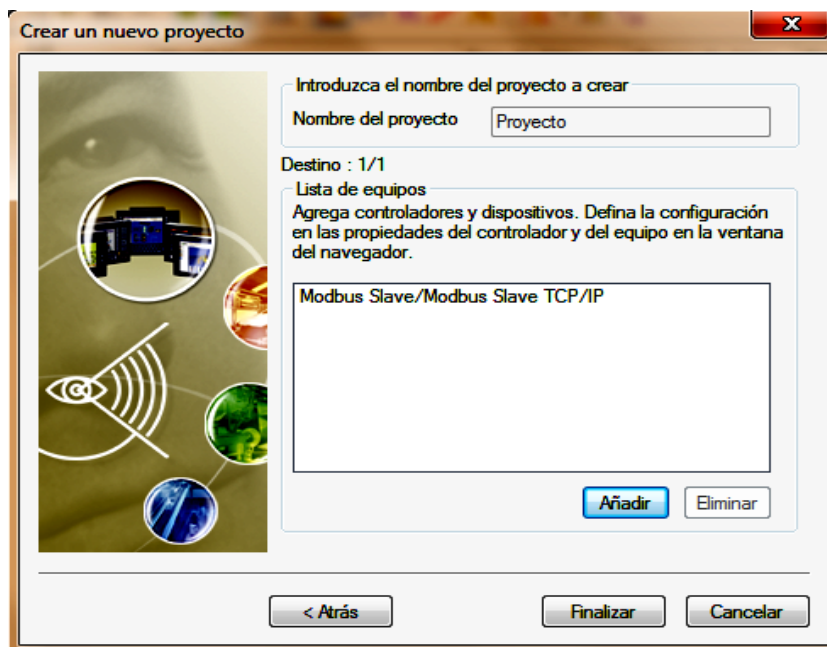


The screenshot shows a window titled "Crear un nuevo proyecto" with a close button (X) in the top right corner. On the left side, there is a vertical panel with a background image of a person's face and several circular icons representing different network or system components. The main area of the window contains the following fields and controls:

- A label "Introduzca el nombre del proyecto a crear" above a text input field containing "jose ignacio".
- A label "Destino : 1 of 2" above a section titled "Configuración de destino".
- Inside the "Configuración de destino" section, there is a checked checkbox labeled "Asignar la siguiente dirección IP".
- Below the checkbox, there are three rows of IP configuration fields:
 - "Dirección IP" with the value "192 . 168 . 71 . 84".
 - "Máscara subred" with the value "255 . 255 . 255 . 0".
 - "Puerta enlace predeterminada" with the value "0 . 0 . 0 . 0".
- At the bottom of the window, there are four buttons: "< Atrás", "Siguiente >" (highlighted in blue), "Finalizar", and "Cancelar".

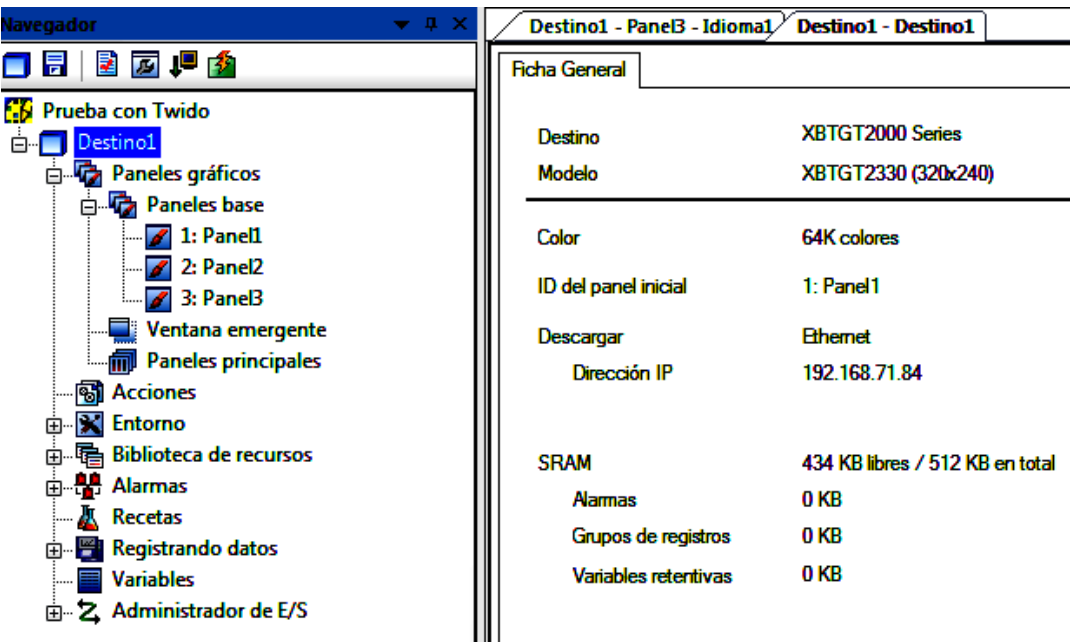
Clic en siguiente para elegir que protocolo de comunicación va a emplear la Magelis. Para este ejercicio se utilizara la Magelis como esclavo TCP/IP, ver figura 27

Figura 27: CONFIGURACIÓN DE LA MAGELIS COMO ESCLAVO



Por último, clic en finalizar para terminar de construir el proyecto. El siguiente paso es construir la representación grafica del proceso (paneles) a supervisar y las variables que se van a utilizar en el proceso de la comunicación. Para este paso se recomienda leer el tutorial Vijeo-Designar, versión 4.6 (fecha 08/2007). Ya elaborado los paneles y configuradas sus variables se observa los aspectos más importantes en la ventana principal, la cual tiene una serie de ítems que describe estos aspectos. Ver figura 28

Figura 28: VENTANA PRINCIPAL



En Destino se observa el modelo, la dirección IP, el protocolo de comunicación, etc. En paneles gráficos se observa los paneles elaborados por el usuario para gestionar la comunicación, para este ejemplo se construyeron los siguientes tres paneles.

Los paneles de la figura 29 son los mencionados al comienzo de este programa y son los encargados de visualizar las acciones que realiza los programas en ladder para los clientes twido en el proceso de la comunicación.

Las variables utilizadas en el programa dos clientes Twido y una Magelis servidora, se presentarán en la figura 30

Figura 29: PANELES DEL PROYECTO

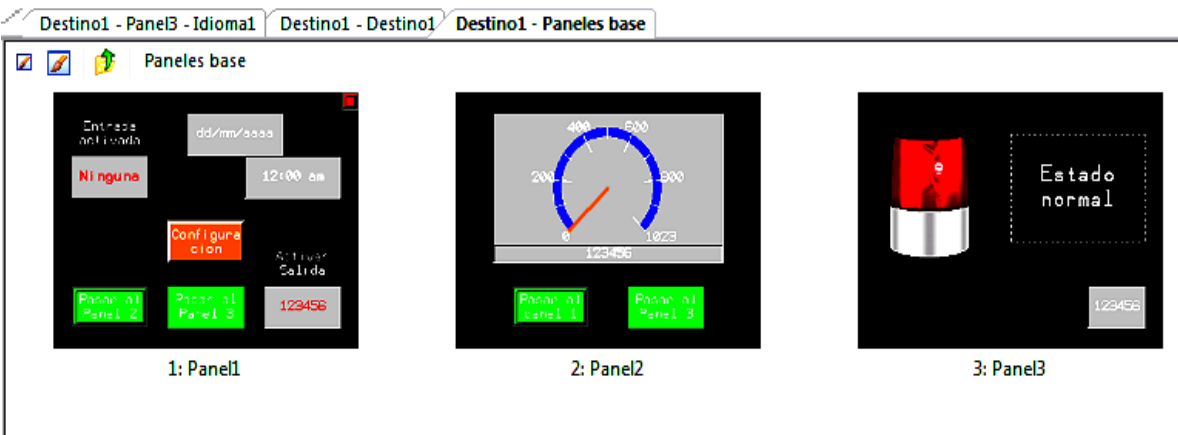


Figura 30: VARIABLES DEL PROYECTO

Navegador

Destino1 - GrupoDeAlarmas1 - Idioma1 Destino1 - Texto Recursos - Idioma1 Destino1 - Seguridad Destino1 - Panel3 - Idioma1 Destino1 - Destino1 - Idioma1

	Nombre	Tipo de datos	Origen de datos	Grupo de escan...	Dirección de di...	Grupo de alarm...	Grupo de regist...
1	Discreto01	Discreto	Interno			Desactivado	Ninguno
2	Entero01	Entero	Externo	ModbusSlaveTCP	%MW0	Desactivado	Ninguno
3	Entero02	Entero	Externo	ModbusSlaveTCP	%MW1	Desactivado	Ninguno
4	Entero03	Entero	Externo	ModbusSlaveTCP	%MW2	Desactivado	Ninguno
5	Sirena	Discreto	Externo	ModbusSlaveTCP	%MW3:X0	Desactivado	Ninguno

Prueba con Twido

Destino1

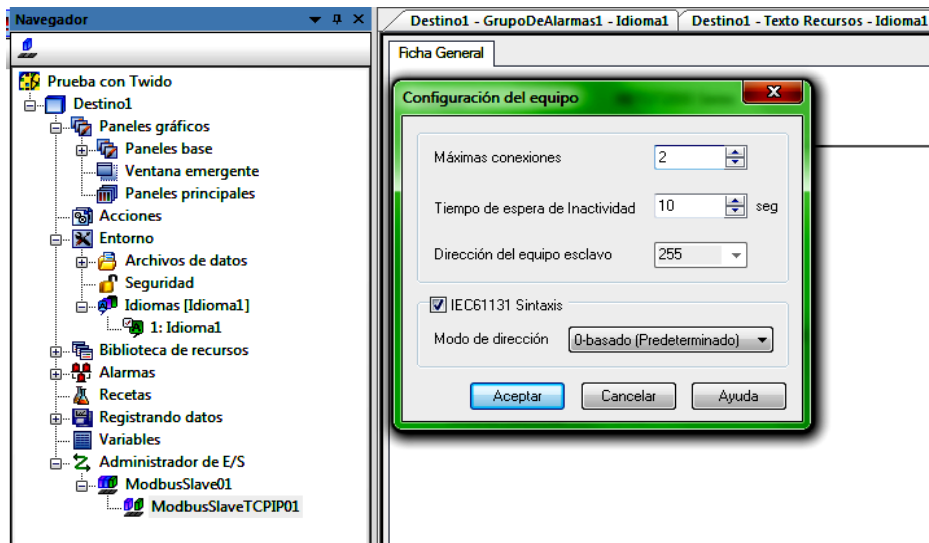
- Paneles gráficos
 - Paneles base
 - 1: Panel1
 - 2: Panel2
 - 3: Panel3
 - Ventana emergente
 - Paneles principales
- Acciones
- Entorno
 - Archivos de datos
 - Seguridad
 - Idiomas [Idioma1]
 - 1: Idioma1
- Biblioteca de recursos
 - Alarmas
 - Recetas
 - Registrando datos
 - Variables
- Administrador de E/S

En la figura 30 se observa las variables tipo entero utilizadas en la comunicación con el PLC twido 89 y las palabras de memoria (%MW0, %MW1 y %MW2) de la HMI usadas en el programa ladder para que el cliente twido 89 escriba y lea de ellas. Se observa también en grupo de escaneo que la Magelis esta configurada en modo Modbus Slave TCP. Por último la variable Sirena es de tipo discreto y se

usa en la comunicación con el PLC cliente twido 88 para activar una alarma en el panel 3 de la Magelis, utilizando la palabra de memoria %MW3 y el bit 0 (X0) de esta palabra.

Es importante observar en la ventana principal en Administrador de E/S, Modbus slave01 y Modbus Slave TCP/IP01 las características de conexión del esclavo TCP/IP (servidor Magelis). Ver figura 31

Figura 31: CARACTERÍSTICAS DE LA CONEXION

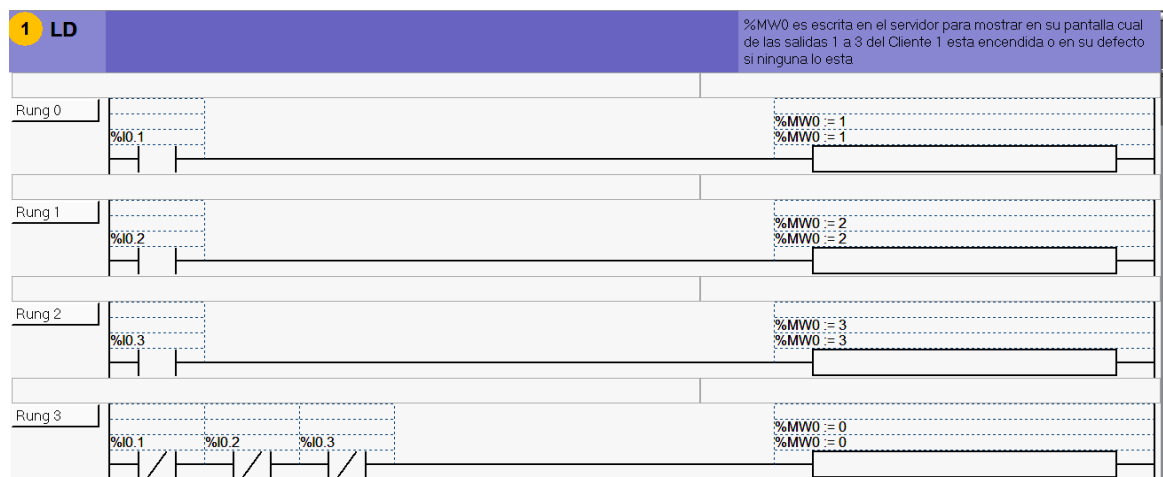


3.3.3 DESCRIPCION PROGRAMA LADDER CLIENTE 1 TWIDO 89

El programa cliente del twido 89 se divide en dos partes, la primera parte el cliente twido 89 activa/desactiva las entradas (%I0.1, %I0.2 o %I0.3) o desactiva todas las entradas que a su vez activan o desactiva la palabra de memoria %MW0 que a su vez se escribe en la palabra de memoria %MW0 del servidor (HMI Magelis) para

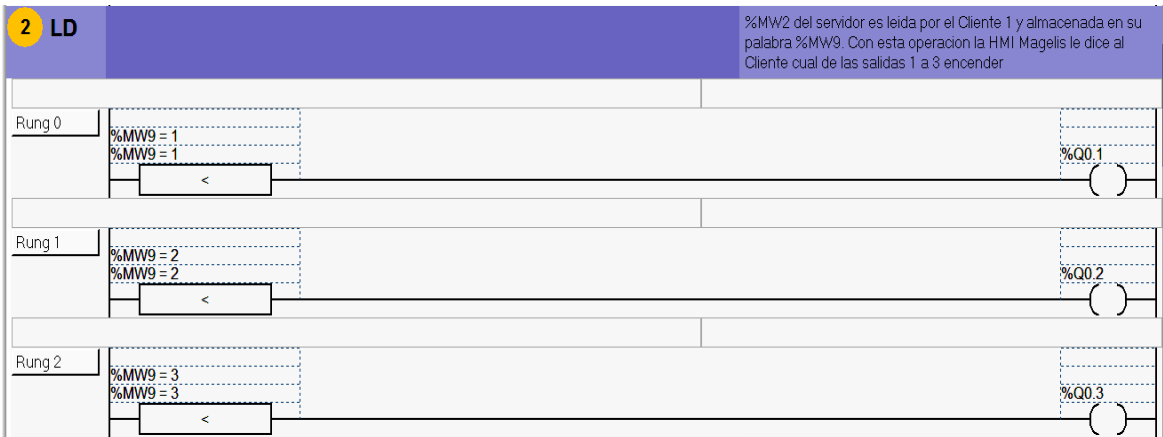
indicar en el (panel 1) si alguna de las entradas 1 a 3 del cliente 1 twido 89 está activada o si ninguna lo está. También escribe %IW0.0.0 del cliente 1 en %MW1 del servidor para mostrarla en un tacómetro (Panel 2) del servidor Magelis. La figura 32 presenta como a la palabra de memoria %MW0 se le asignan valores 0,1, 2 o 3 cada vez que se activa una entrada “I0.1, I0.2 o I0.3” o se desactivan todas. La segunda parte del programa presentan los valores de la palabra de memoria %MW2 del servidor Magelis que son leídos por el Cliente 1 twido 89 y almacenada en su palabra de memoria %MW9 para que la HMI Magelis active mediante el panel 2 las salidas Q0.1,Q0.2 o Q0.3 del cliente 1 twido 89. Ver la figura 33

Figura 32: PALABRA %MW0 QUE SE VA A ESCRIBIR



En la figura 33 se presenta los valores 1,2 y 3 leídos y almacenados en la palabra de memoria % MW9 para activar cuando esta palabra se hace igual a uno de estos las salidas %Q0.1, %Q0.2 o Q0.3.

Figura 33: VALORES LEIDOS Y ALMACENADOS EN %MW9



3.3.3.1 TABLAS DE LECTURA Y ESCRITURAS DE N PALABRAS

Como ya se explico en el primer programa las tablas de lectura y escrituras de N palabras sirven para desarrollar la comunicación entre dos dispositivos que tengan configurado el protocolo de comunicación modbus TCP/IP. A continuación se presentan la programación de estas tablas en ladder.

3.3.3.2 TABLA DE LECTURA N PALABRAS PARA LEER LA PALABRA %MW2 DEL SERVIDOR HMI

La figura 34 presenta la programación de la tabla para leer la palabra de memoria %MW2 con el código de solicitud 03 de modbus. La palabra de memoria %MW5 lleva La dirección de la palabra leída “palabra %MW2”

Figura 34: PROGRAMACIÓN DE LA TABLA PARA LEER LA %MW2

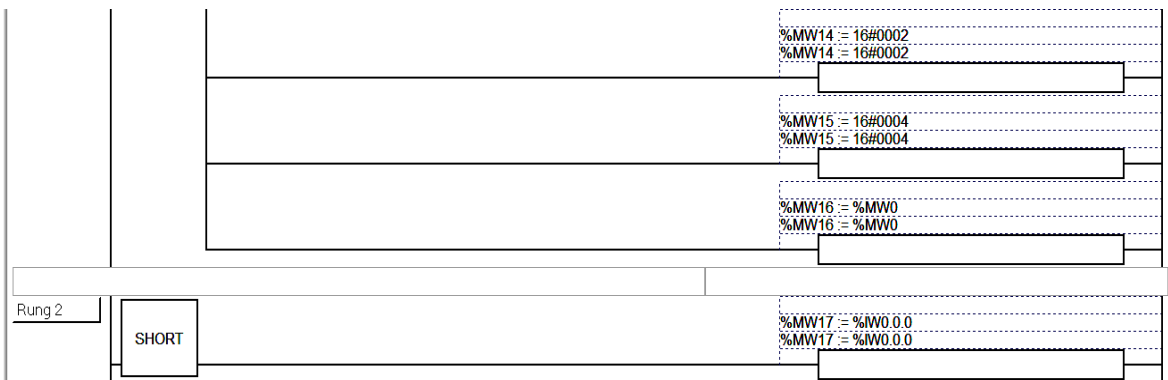
3 LD Tablas de envio/recepcion para leer y escribir palabras		
LEE %MW2 DEL SERVIDOR (HMI MAGELIS)		
Rung 0	SHORT	%MW2 = 16#0106 %MW2 = 16#0106 %MW3 = 16#0300 %MW3 = 16#0300 %MW4 = 16#0103 %MW4 = 16#0103 %MW5 = 16#0002 %MW5 = 16#0002 %MW6 = 16#0001 %MW6 = 16#0001

3.3.3.3 TABLA DE ESCRITURA DE N PALABRAS UTILIZADA PARA ESCRIBIR EN LAS PALABRAS %MW0 Y %MW1 DEL SERVIDOR (HMI MAGELIS)

Figura 35: TABLA PARA ESCRIR EN LAS PALABRAS %MW0 Y %MW1

ESCRIBE EN %MW0 Y %MW1 DEL SERVIDOR (HMI MAGELIS)		
Rung 1	SHORT	%MW10 = 16#010C %MW10 = 16#010C %MW11 = 16#0007 %MW11 = 16#0007 %MW12 = 16#0110 %MW12 = 16#0110 %MW13 = 16#0000 %MW13 = 16#0000

Figura 35 (continuación)



La figura 35 presenta la programación de la tabla para escribir en la palabra de memoria %MW0 y %MW1 del servidor con el código de solicitud 10 H de modbus. Las palabras de memoria %MW16 y %MW17 lleva La dirección de las palabras que se van a escribir del cliente “palabra %MW0, % IW0.0.0”.

3.3.3.4 GESTION DE LA COMUNICACIÓN PARA LA LECTURA DE UNA PALABRA Y ESCRITURA DE DOS PALABRAS.

Figura 36: CONTADOR DE PASOS

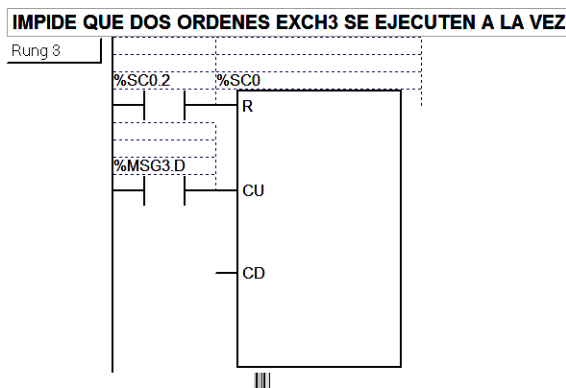
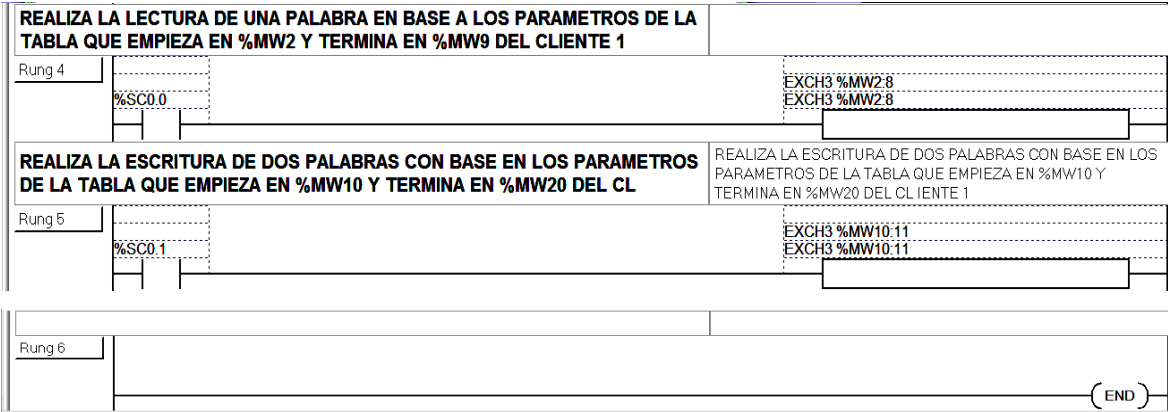


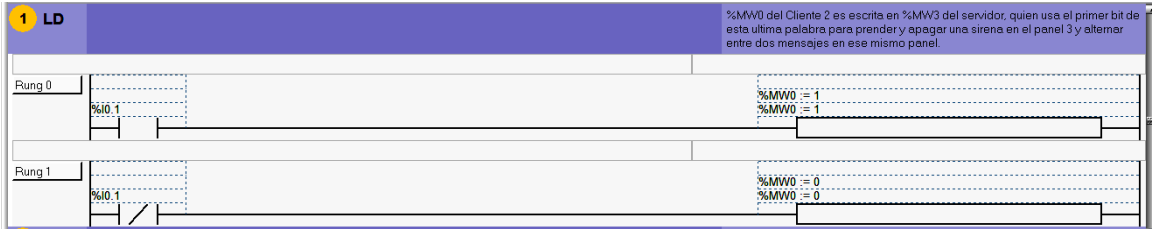
Figura 37: GESTION DE LA COMUNICACIÓN SEGUNDO PROGRAMA 1



3.3.4 DESCRIPCION PROGRAMA LADDER CLIENTE 2 TWIDO 88

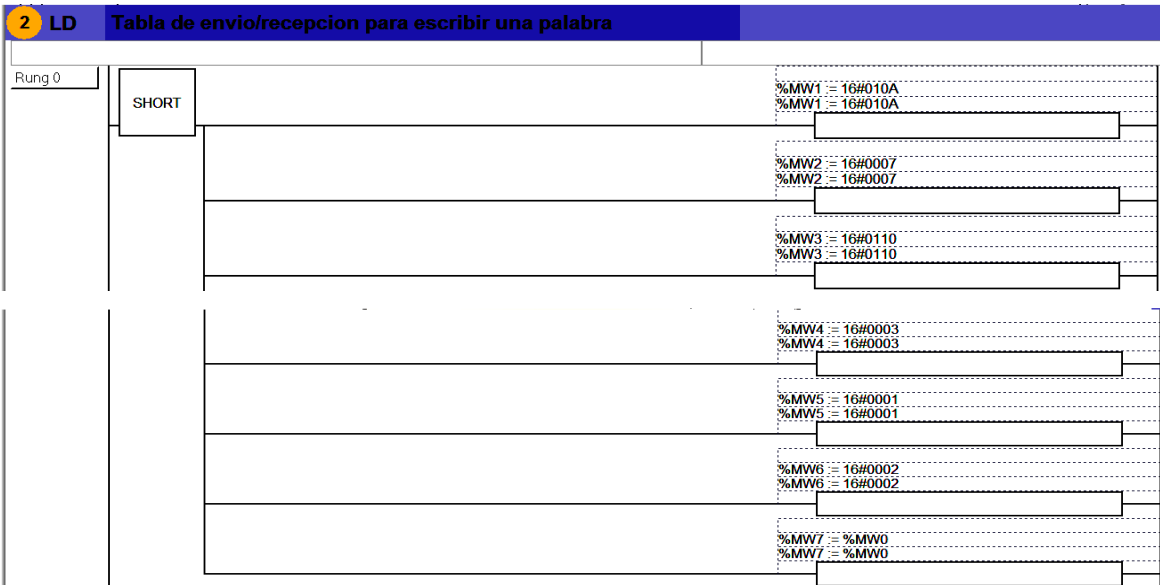
Este programa usa la entrada 1 del cliente 2 para activar/desactivar el primer bit de la palabra de memoria `%MW0`, la cual se escribe en la palabra de memoria `%MW3` del servidor (HMI Magelis) para activar/desactivar una sirena y alternar entre dos mensajes (uno de "estado normal" y otro de "temperatura crítica") en el panel 3 del servidor. Ver figura 38

Figura 38: PALABRA DE MEMORIA `%MWO` QUE SE VA A ESCRIBIR



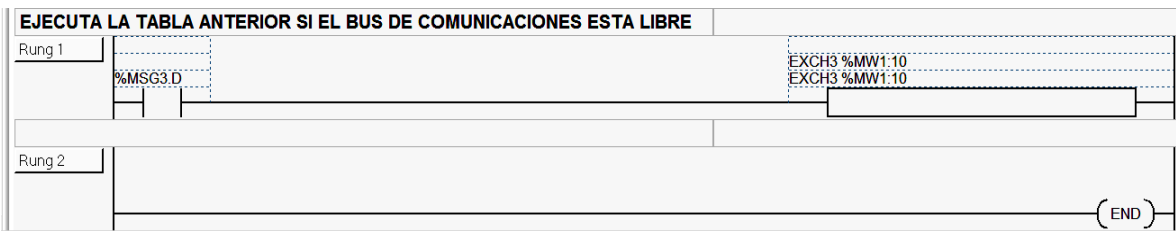
3.3.4.1 TABLA DE ESCRITURA DE N PALABRAS

Figura 39: TABLA PARA ESCRIBIR %MW0 EN %MW3



3.3.4.2 GESTIÓN DE LA COMUNICACIÓN CON LA INSTRUCION EXCH3

Figura 40: GESTION DE LA COMUNICACIÓN SEGUNDO PROGRAMA 2



En la figura 54 no se utiliza el bloque contador de pasos debido a que solo hay una instruccion EXCH3.

3.4 TERCER PROGRAMA

PROGRAMA PARA LA COMUNICACIÓN ENTRE UN CLIENTE HMI MAGELIS Y DOS SERVIDORES TWIDOS.

Este programa se complementa con el programa del cliente (HMI Magelis) denominado "Cliente Modbus TCP/IP". La función del programa es cambiar el valor de la palabra de memoria %MW0 en el programa del servidor 1 PLC Twido 88 mediante la acción de flancos ascendentes activados por las entradas 1 a 6 (sensores ópticos), para que esta palabra de memoria sea leída por el cliente Magelis y así cambiar la posición horizontal de un objeto que se simula a través del movimiento de una banda transportadora en el panel 1 del programa del cliente. El cliente Magelis también escribe valores del 0 al 3 en la palabra de memoria %MW1 del servidor 1 Twido 88 para activar una de sus salidas 1 a 3 o ninguna de ellas. Este programa guarda el valor del potenciómetro 1 del servidor 2 Twido 89 en la palabra de memoria %MW1 del servidor 2 Twido 89 para que esta palabra sea leída por el cliente (HMI Magelis) y así incrementar/decrementar en el panel 2 del cliente Magelis un gráfico de barras vertical con etiqueta fija. El potenciómetro 1 simula una entrada analógica, como por ejemplo una termocupla. También se activan/desactivan los bits %M0 y %M1 del servidor 2 con las entradas 1 y 2 del servidor 2 Twido 89 para que sean leídos por el cliente y de esta forma activar una alarma de presión y una alarma de humedad que se podrán observar en el panel 1 del cliente Magelis

3.4.1 CONFIGURACIÓN DE LA DIRECCIÓN IP:

Figura 41: DIRECCIÓN IP DEL SERVIDOR 1 (PLC 88)

IP address configuration

☐ From a server IP address: 192 . 168 . 71 . 88

☒ Configured Subnet mask: 255 . 255 . 255 . 0

Gateway address: 192 . 168 . 71 . 88

Marked IP

☐ Specify a marked IP address Specify an IP address for the marked connection. . . .

Idle time

NOTE: The controller will detect the active and passive TCP connections and will close those that are inactive for the time indicated here. If the maximum idle time is set to 0, the controller will not run the detection.

Enter time out for the TCP connection.

10 min(s) Default

Remote Devices

Index	Slave IP address	Unit ID	Connection time out (100 ms)
1			
2			
3			
4			

3.4.1.1 DIRECCIÓN IP DEL PLC SERVIDOR 1 (TWIDO 88). El PLC Twido 88 va a ser el servidor 1 del cliente HMI (Magelis). En la figura 41 se observa la dirección IP clase C del servidor 1 configurada como 192.168.71.88. NOTA: no se configura otra dirección IP en “Remote Devices” debido a que el PLC Twido88 cumple la función de servidor únicamente. Ver figura 41

3.4.1.2 DIRECCIÓN IP DEL PLC SERVIDOR 2 (PLC 89). El PLC Twido 89 va a ser el servidor 2 de la HMI Magelis (cliente). En la figura 42 se observa la dirección IP clase C del servidor 2 (Twido 89). Configurada como 192.168.71.89

Figura 42: DIRECCIÓN IP SERVIDOR 2 (PLC 89)

Ethernet networks

IP address configuration

☐ From a server IP address: 192 . 168 . 71 . 89

☒ Configured Subnet mask: 255 . 255 . 255 . 0

Gateway address: 192 . 168 . 71 . 89

Marked IP

☐ Specify a marked IP address Specify an IP address for the marked connection. . . .

Idle time

NOTE: The controller will detect the active and passive TCP connections and will close those that are inactive for the time indicated here. If the maximum idle time is set to 0, the controller will not run the detection.

Enter time out for the TCP connection.

10 min(s) Default

Remote Devices

Index	Slave IP address	Unit ID	Connection time out (100 ms)
1			
2			
3			
4			

3.4.2 CONFIGURACION DEL CLIENTE MAGELIS CON SOFTWARE VIJEO DESIGNER

Configure en el proyecto: nombre del proyecto, descripción o comentarios, tipo de proyecto y contraseña del proyecto. Ver figura 43

Configure en la ventana de la figura 44, el nombre del destino, tipo de destino y modelo

Configure la dirección IP de la Magelis. La Magelis que tiene el almacén de E3T tiene una dirección IP clase C configurada como 192.168.71.84. Ver figura.45

Figura 43: DESCRIPCION DEL UN PROYECTO

The dialog box is titled "Crear un nuevo proyecto" and features a sidebar with a background image of a person's face and several circular icons representing different project types (e.g., server, globe, eye). The main area contains the following fields:

- Introduzca el nombre del proyecto a crear:** A text box with "prueba con Twido" entered.
- Descripción o comentario:** A text area with "proyecto que comunica dos Twidos clientes con una magelis servidora" entered.
- Tipo:** Two radio buttons. The first is "Proyecto con un sólo destino". The second is "Proyecto con" followed by a spinner box set to "1" and the word "destinos". The second option is selected.
- Contraseña del proyecto:** A checked checkbox. Below it are three text boxes: "Ingrese contraseña" (filled with dots), "Confirme contraseña" (filled with dots), and "Recordar (opcional)".

At the bottom are four buttons: "< Atrás", "Siguiente >" (highlighted in blue), "Finalizar", and "Cancelar".

Figura 44: TIPO DE MAGELIS

The dialog box is titled "Crear un nuevo proyecto" and features the same sidebar as Figure 43. The main area contains the following fields:

- Introduzca el nombre del proyecto a crear:** A text box with "prueba con Twido" entered.
- Destino : 1 of 2:** A section header.
- Nuevo proyecto/destino:** A section header.
- Nombre del destino:** A text box with "Destino1" entered.
- Tipo de destino:** A dropdown menu with "XBTGT2000 Series" selected.
- Modelo:** A dropdown menu with "XBTGT2330 (320x240)" selected.

At the bottom are four buttons: "< Atrás", "Siguiente >" (highlighted in blue), "Finalizar", and "Cancelar".

Figura 45: DIRECCION IP DE LA HMI



Elija que protocolo de comunicación va a emplear la Magelis. Para este ejercicio se utilizara la Magelis como maestro TCP/IP, ver figura 46

Por último, clic en finalizar para terminar de construir el proyecto. El siguiente paso es construir los paneles y las variables que se van a utilizar en el proceso de la comunicación. Para este paso se recomienda leer el tutorial Vijeo-Designar, versión 4.6 (fecha 08/2007). Ya elaborado los paneles y configuradas sus variables observamos los aspectos más importantes en la ventana principal la cual tiene una serie de ítems que describe estos aspectos. Ver figura 47

Figura 46: MAGELIS COMO CLIENTE TCP/IP

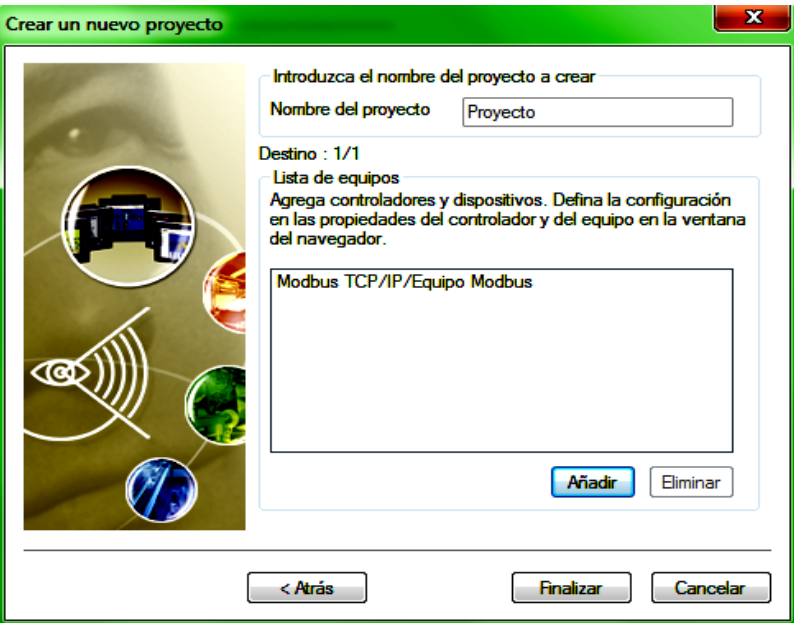
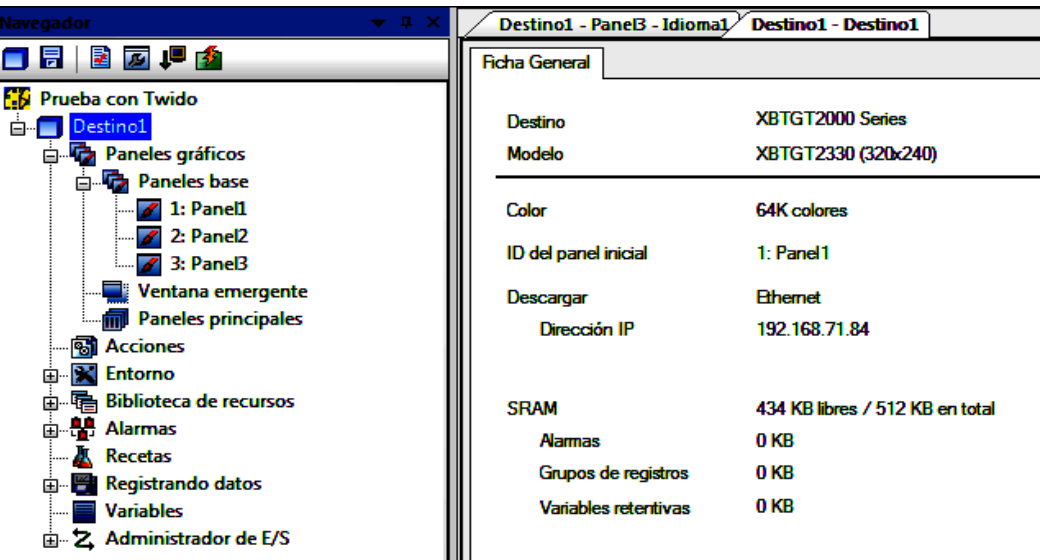


Figura 47: VENTANA PRINCIPAL CARACTERÍSTICAS DE LA MAGELIS



En Destino se observa el modelo, la dirección IP, el protocolo de comunicación, etc. En paneles gráficos se observa los paneles elaborados por el usuario para gestionar la comunicación, para este ejemplo se construyeron los siguientes dos paneles.

Figura 48: PANELES DEL CLIENTE MAGELIS

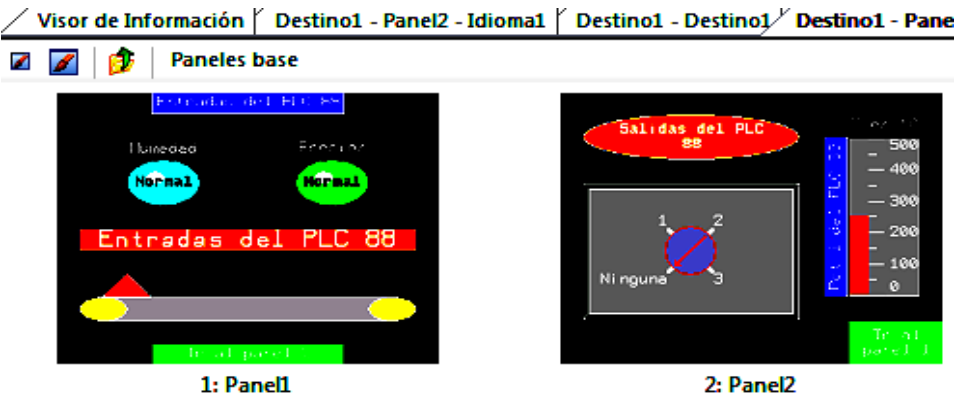


Figura 49: VARIABLES DE LOS SERVIDORES

	Nombre	Tipo de datos	Origen de datos	Grupo de escan...	Dirección de di...	Grupo de alarm...	Grupo de regist...
1	Humedad	Discreto	Externo	EquipoModbus02	%M1	Desactivado	Ninguno
2	Posicion	Entero	Externo	EquipoModbus01	%MW0	Desactivado	Ninguno
3	Potenciometro1Servidor2	Entero	Externo	EquipoModbus02	%MW0	Desactivado	Ninguno
4	Presion_Alta	Discreto	Externo	EquipoModbus02	%M0	Desactivado	Ninguno
5	Salidas	Entero	Externo	EquipoModbus01	%MW1	Desactivado	Ninguno

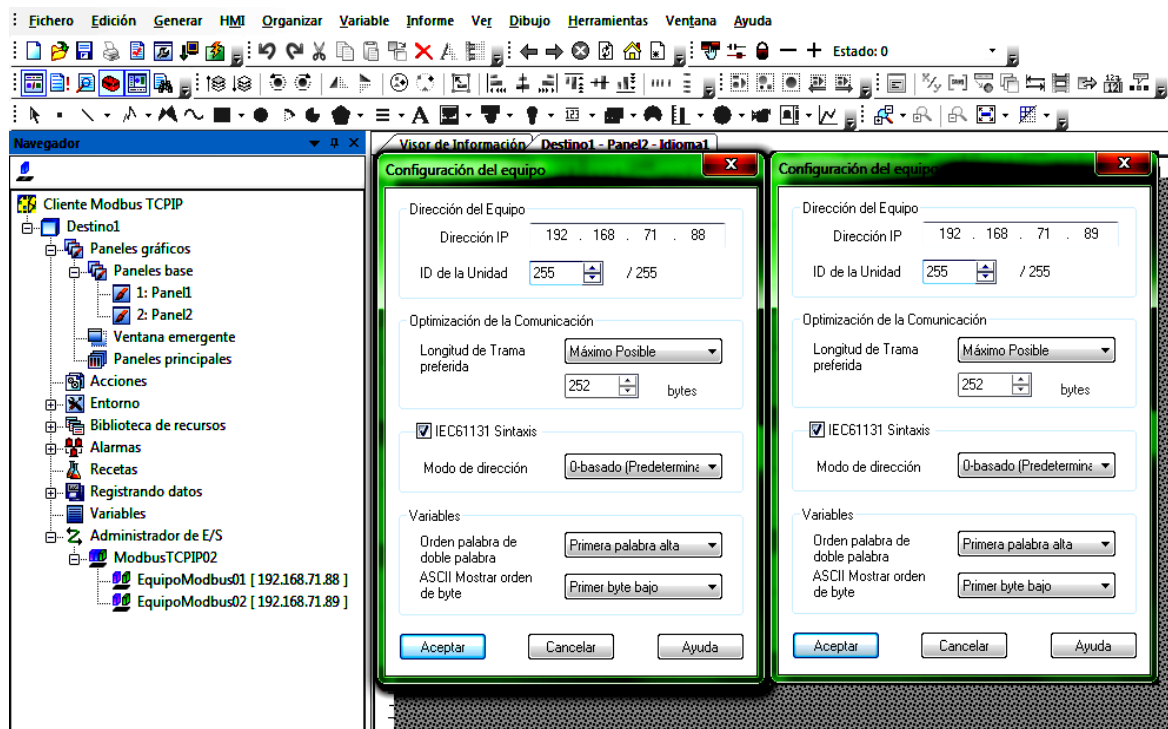
Los paneles de la figura 48 son los mencionados al comienzo de este programa y son los encargados de visualizar las acciones que realiza los programas en ladder para los servidores twido 88 y twido 89.

Las variables utilizadas por la Magelis para la comunicación con los dos servidores PLCs Twido, se presentan en la figura 49

En la figura 49 se observa las variables enteras y discretas que utiliza la Magelis en la comunicación con el PLC twido 88 y el PLC twido 89. Las palabras de memoria (%MW0, %MW1) son las utilizadas por la Magelis para comunicarse con el PLC 88, y las variables de la Magelis para comunicarse con el PLC 89 son la palabra %MW0 y las variables humedad y Presión_Alta que es de tipo discreto y se usa para activar una alarma en el panel 1 de la Magelis, utilizando los bits %M0 y % M1.

Por último se observa en la ventana principal, en Administrador de E/S y en Modbus TCP/IP02 las características de conexión del maestro TCP/IP con los dos servidores twido.88 y twido 89 y sus respectivas conexiones IP, Ver figura 50

Figura 50: CARACTERÍSTICAS DE CONEXIÓN DEL MAESTRO HMI TCP/IP



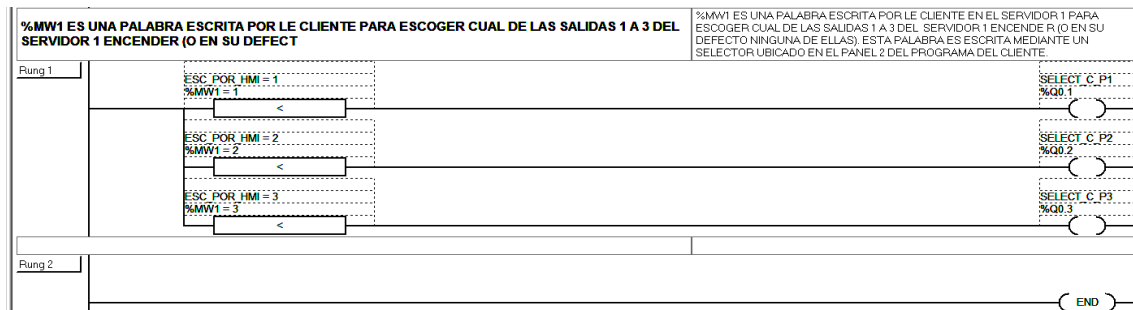
3.4.3 DESCRIPCION PROGRAMA LADDER SERVIDOR 1 TWIDO 88: en el programa servidor 1 del twido 88 se divide en dos partes, la primera parte el cliente Magelis lee la palabra de memoria %MW0 del servidor 1 twido 88 para cambiar la posición horizontal de un objeto simulado a través de una banda transportadora en el panel 1 del cliente mediante la acción de flancos ascendentes activados por las entradas 1 a 6 (sensores ópticos) del servidor. Ver el siguiente programa.

Figura 51: PALABRA %MW0 DEL SERVIDOR LEÍDA POR LA MAGELIS



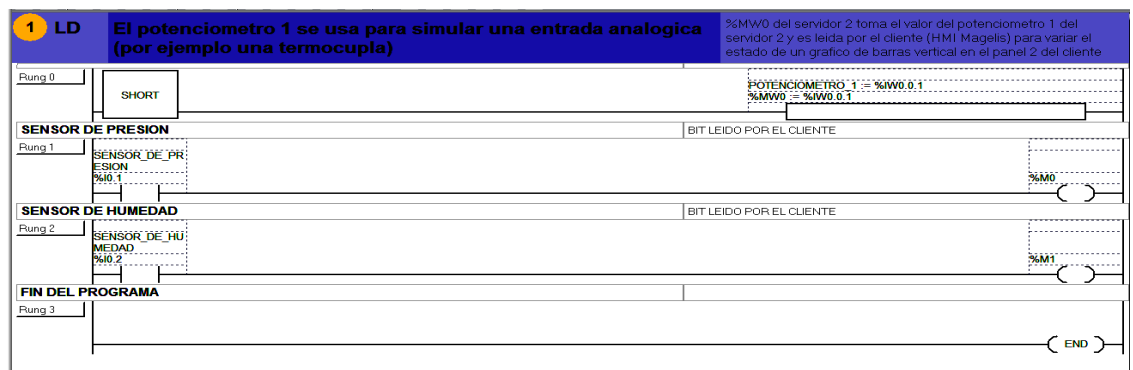
La segunda parte del programa el cliente escribe en la palabra de memoria %MW1 del servidor 1 twido 88 para activar alguna de las tres salidas o no activar salidas. Esta palabra es escrita mediante un selector ubicado en el panel 2 del programa del cliente. El programa del servidor 1 muestra las tres salidas y la palabra de memoria %MW1.ver figura 52

Figura 52: PALABRA %MW1 DEL SERVIDOR ESCLITA POR LA MAGELIS



3.4.4 DESCRIPCION PROGRAMA LADDER SERVIDOR 2 TWIDO 89: este programa guarda el valor del potenciómetro 1 del servidor 2 twido 89 en la palabra de memoria %MW1 del servidor 2 twido 89 para que sea leída por el cliente (HMI Magelis) y así incrementar/decrementar la visualización de un grafico de barras vertical con etiqueta fija en el panel 2 del cliente. El potenciómetro 1 simula una entrada analógica, como por ejemplo una termocupla. También se activan/desactivan los bits %M0 y %M1 del servidor 2 con las entradas 1 y 2 del servidor 2 Twido 89 para que sean leídos por el cliente y de esta forma activar una alarma de presión y una alarma de humedad que se puede observar en el panel 1 del cliente Magelis. Ver figura 53

Figura 53: VARIABLES DEL SERVIDOR LEIDAS POR EL CLIENTE



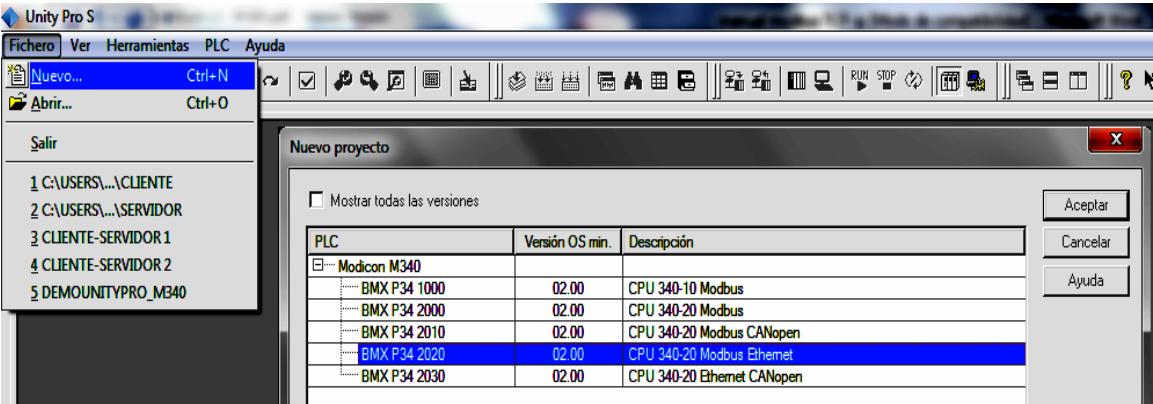
3.5 CUARTO PROGRAMA

PROGRAMA QUE GESTIONA LA COMUNICACIÓN ENTRE UN CLIENTE M340, UN CLIENTE-SERVIDOR HMI MAGELIS Y UN SERVIDOR M340.

3.5.1 CONFIGURACION DE LA RED TCP/IP PARA EL CLIENTE M340: La configuración de la red TCP/IP para un PLC M340 se construye a través del software Unity pro. A continuación se presentan una serie de pasos principales para una aplicación unity pro que servirá para configurar la red TCP/IP

3.5.1.1 CREE UN PROYECTO EN EL SOFTWARE UNITY PRO: Abra el proyecto de la siguiente manera: vaya a todos los programas en su PC y dé doble clic en el icono de unity pro, se desplegara una la ventana de la figura 54.

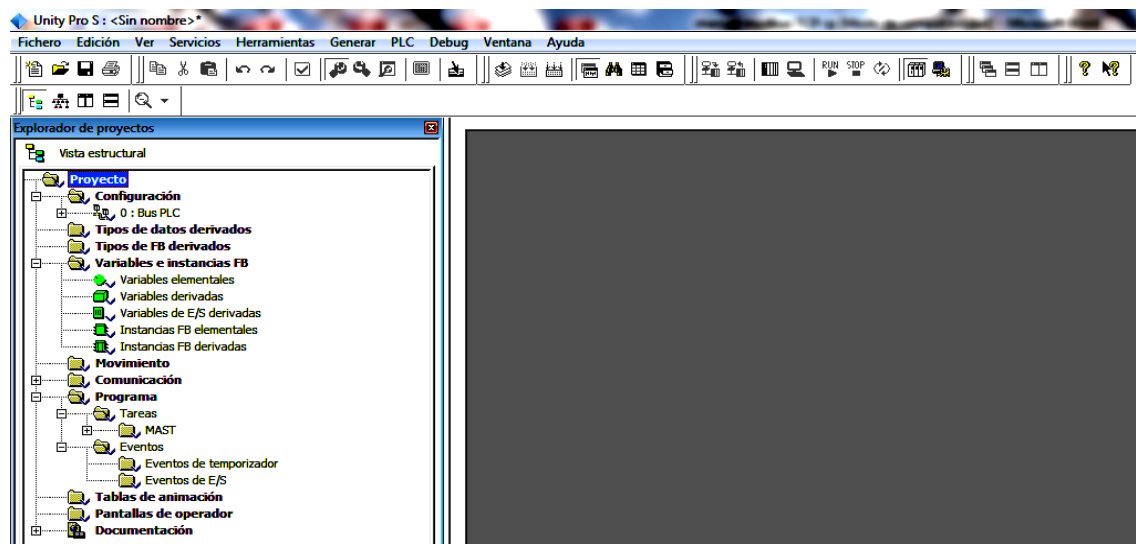
Figura 54: UNIDAD CENTRAL DE PROCESO BMX P342020



en la ventana de la figura 54 vaya a fichero y elija nuevo, se desplegara un cuadro donde están todas las unidades centrales de proceso que utiliza el PLC M340. Para esta aplicación se utilizara la unidad central de proceso con referencia BMX P342020.

Cuando se elije el tipo de unidad central se despliega una ventana, en donde se presenta los ítems necesarios para configurar una aplicación unity pro. Ver figura 55

Figura 55: VENTANA EXPLORADOR DE PROYECTOS



ARME EL PLC M340 EN EL SOFTWARE UNITY PRO

Vaya a la ventana explorador de proyectos y dé doble clic en la carpeta configurar o dé clic en configurar y doble clic en Bus PLC, se desplegara una ventana donde está el PLC M340. Ver figura 56

Figura 56: VENTANA CON EL PLC M340

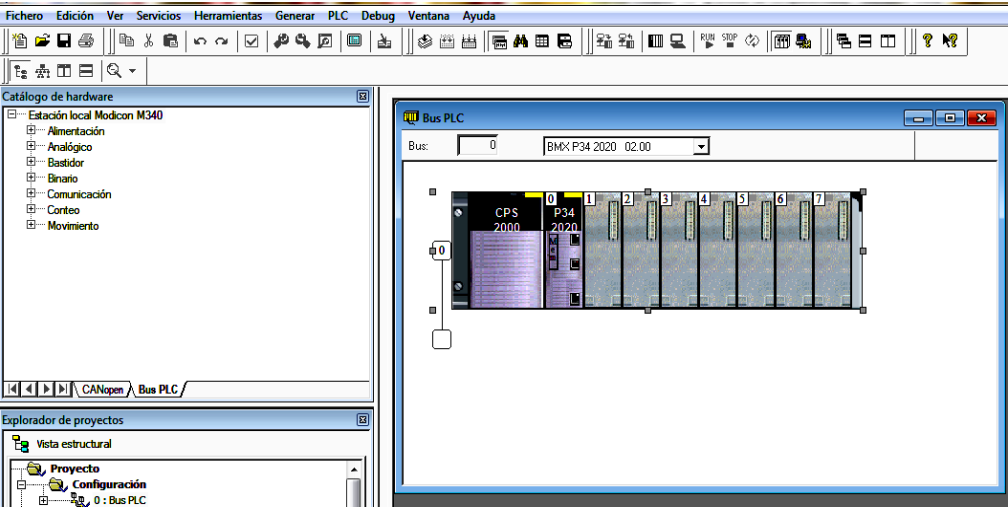
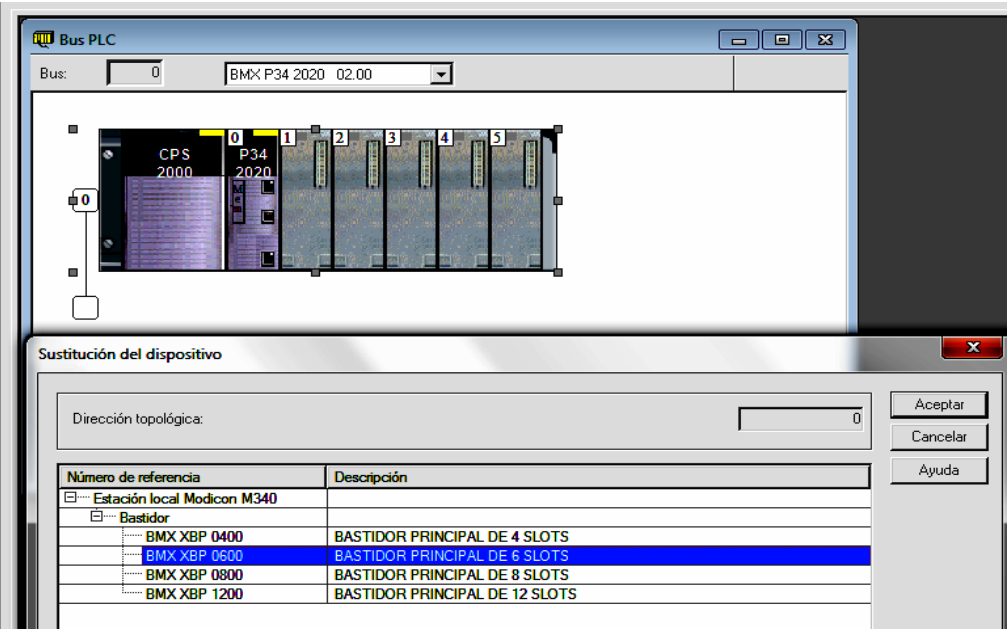


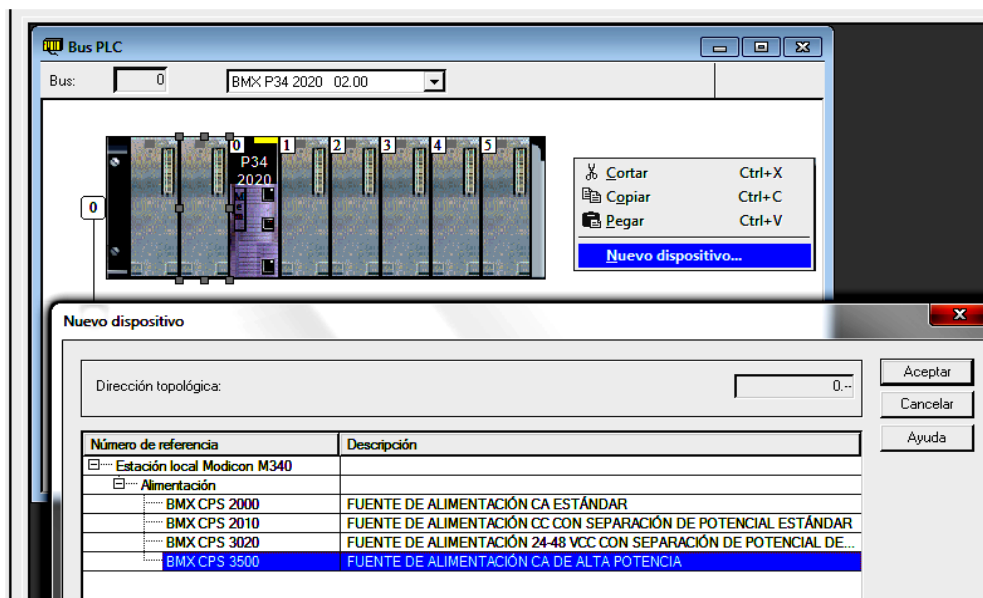
Figura 57: BASTIDOR DE 6 SLOTS CON REFERENCIA BMX XBP 0600



La figura 56 presenta un PLC M340 con referencia BMXCPS 2020 versión 02.00 con características diferentes al que tiene la E3T, por tal motivo se debe sustituir y armar el PLC con las características que tiene el PLC M340 de la E3T. Primero se debe sustituir el bastidor de 8 slots por uno de 6 slots, para esta tarea se debe dar clic derecho sobre el bastidor y elegir la opción sustitución del bastidor en la ventana que se despliega, cuando se elige sustitución del bastidor se despliega otra ventana donde se debe elegir el bastidor de 6 slots con referencia BMX XBP 0600. Ver figura 57

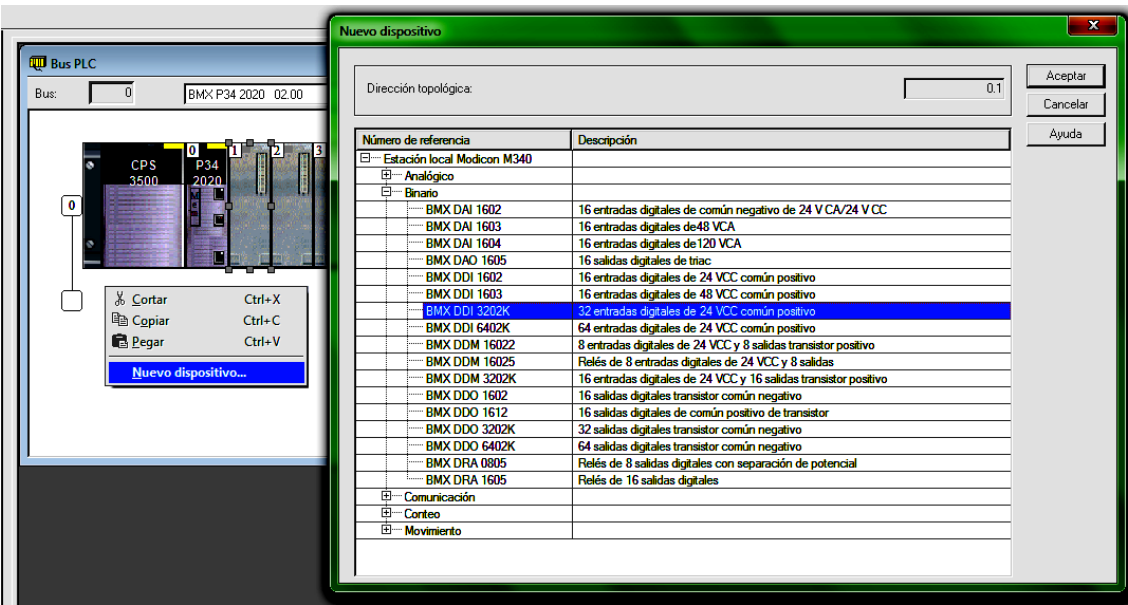
Ya elegido el bastidor es necesario sustituir la fuente de alimentación por la fuente que tiene la E3T de la siguiente manera: clic derecho sobre la fuente que esta por defecto en el PLC para eliminarla. Ya eliminada esta fuente se da clic en donde estaba la fuente y clic en la opción nuevo dispositivo para que se despliegue una ventana donde se elige la fuente de alimentación con referencia BMXCPS 3500. Ver figura 58

Figura 58: FUENTE DE ALIMENTACIÓN CON REFERENCIA BMXCPS 3500



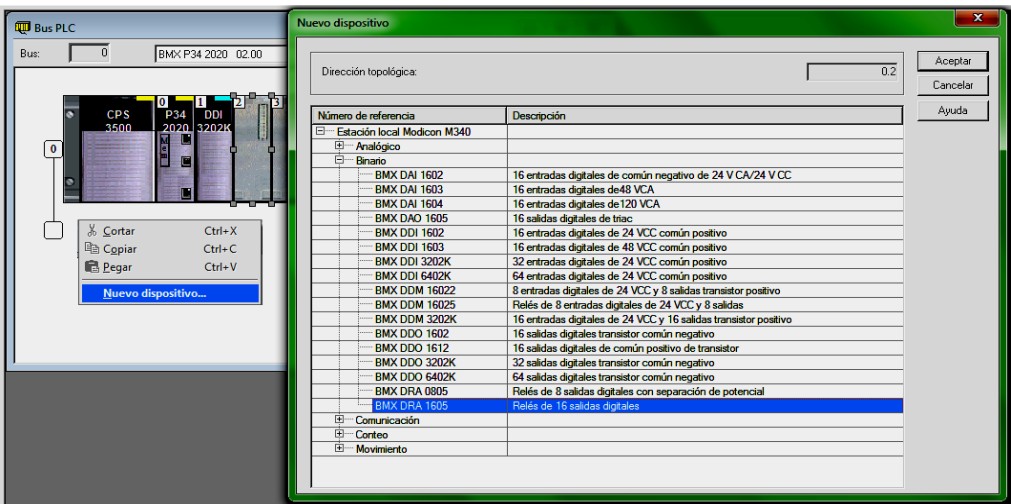
El siguiente paso es colocar los módulos binarios de entradas y salidas en el bastidor del M340 igual como esta en el PLC Modicon M340 de la E3T. Para desarrollar este paso se debe primero: instalar el módulo BMX DDI3202K (32 entradas digitales) en la posición uno del bastidor de la siguiente manera: clic derecho, nuevo dispositivo y en la ventana que se despliega se elije BMX DDI320K. Ver figura 59

Figura 59: MÓDULO BMX DDI3202K



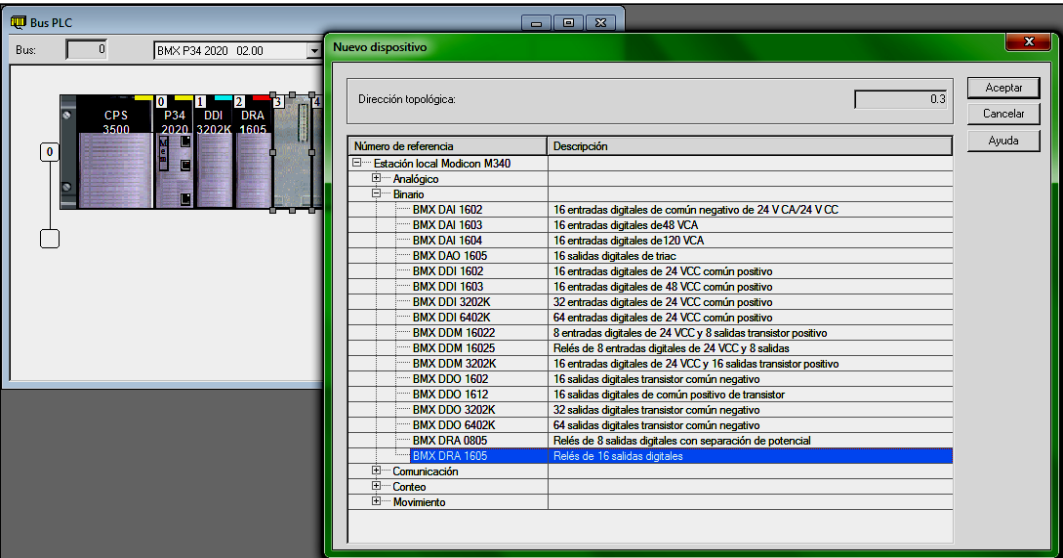
Segundo. Instalar el módulo BMX DRA 1605 (16 salidas digitales) en la posición dos del bastidor de la siguiente manera: clic derecho, nuevo dispositivo y en la ventana que se despliega se elije BMX DRA 1605. Ver figura 60.

Figura 60: MÓDULO BMX DRA 1605



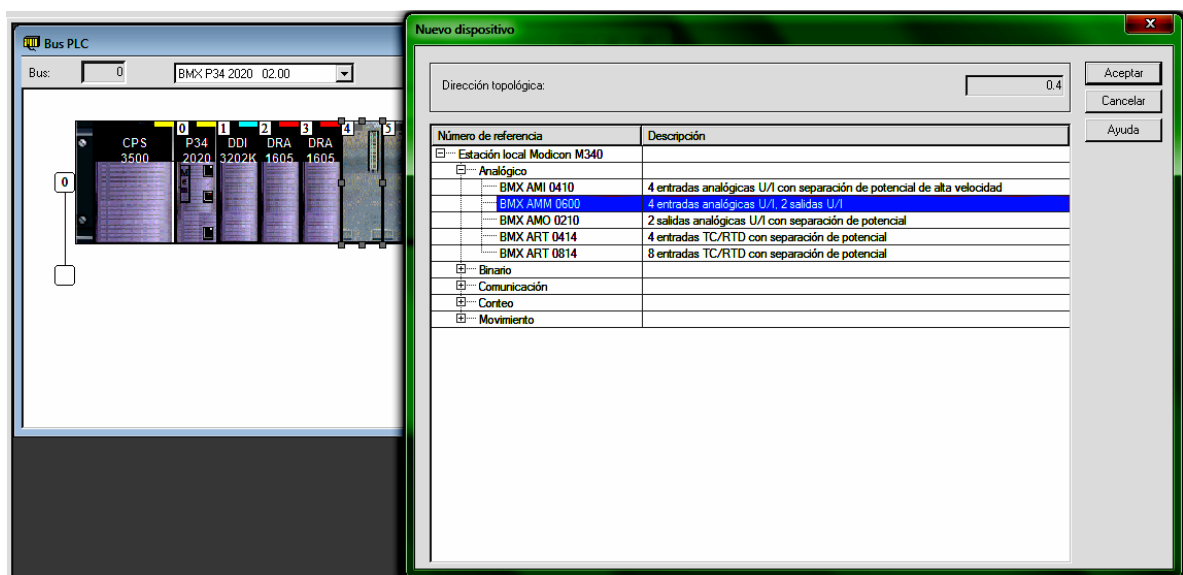
Tercero, Instalar otro módulo BMX DRA 1605 (16 salidas digitales) en la posición tres del bastidor de la siguiente manera: clic derecho, nuevo dispositivo y en la ventana que se despliega se elije BMX DRA 1605. Ver figura 61.

Figura 61: MÓDULO 2 BMX DRA 1650



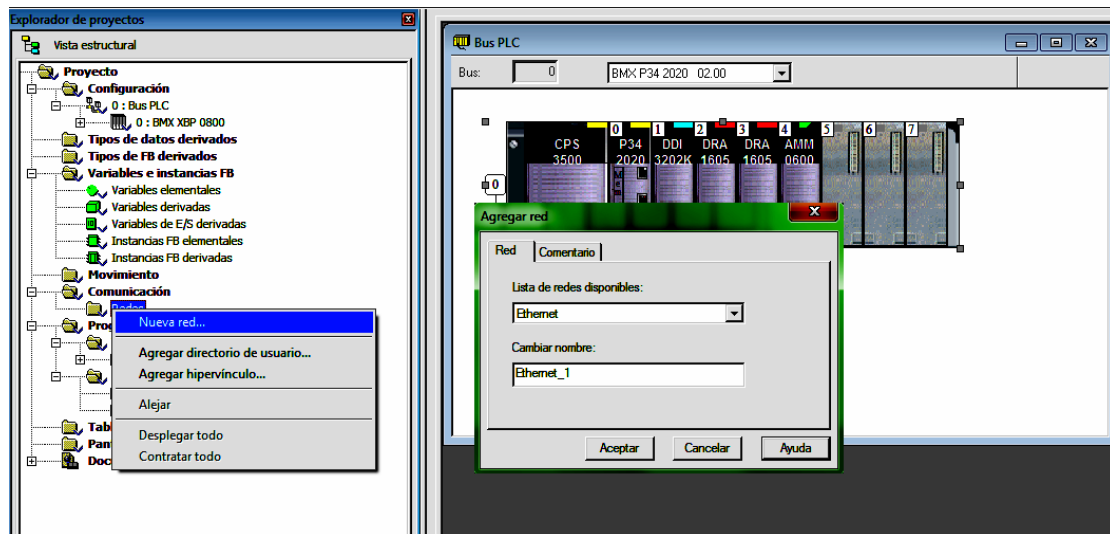
Cuarto. Instalar el módulo BMX AMM 0600 (4 entradas analógicas y 2 salidas analógicas) en la posición cuatro del bastidor de la siguiente manera: clic derecho, nuevo dispositivo y en la ventana que se despliega se elije BMX AMM 0600. Ver figura 62

Figura 62: MÓDULO BMX AMM 0600



3.5.1.2 CONFIGURE LA RED ETHERNET: la configuración de una red ETHERNET en el software Unity pro se elabora de la siguiente manera: vaya a la ventana explorador de proyectos, abra la carpeta de comunicaciones, de clic derecho en la carpeta Redes y elija la opción Nueva Red, configure la ventana que se desplegó eligiendo ethernet en lista de redes disponibles. Ver figura 63

Figura 63: CONFIGURACIÓN DE UNA RED ETHERNET



Cuando se termina de desarrollar estos pasos, la red ethernet aparece en la carpeta de redes pero con una x en color rojo que señala la falta de configuración de otros parámetros de esta red. Para configurar estos parámetros se da clic derecho sobre el nombre de la red (Ethernet_1) y en la ventana que se despliega elegir la opción abrir, se despliega otra ventana con los siguientes parámetros a configurar. Ver figura 64

Los siguientes parámetros son los que se necesitan para configurar la red ethernet.

En familia de modelo elegir: CPU 2020 02.00, CPU 2030 02.00

En dirección del módulo elegir: batidor=0, modulo = 0 y canal = 3

En dirección IP del módulo escribir: dirección IP= 192.168.71.87, máscara de subred = 255.255.255.0 y dirección de la pasarela = 0.0.0.0. Ver figura 65

Figura 64: PARAMETROS A CONFIGULAR EN LA RED ETHERNET

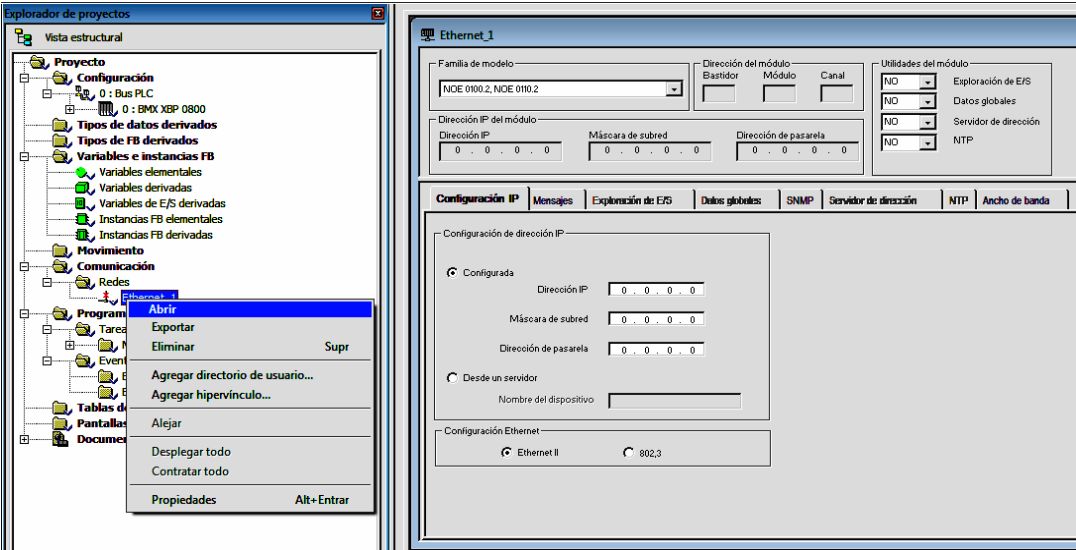
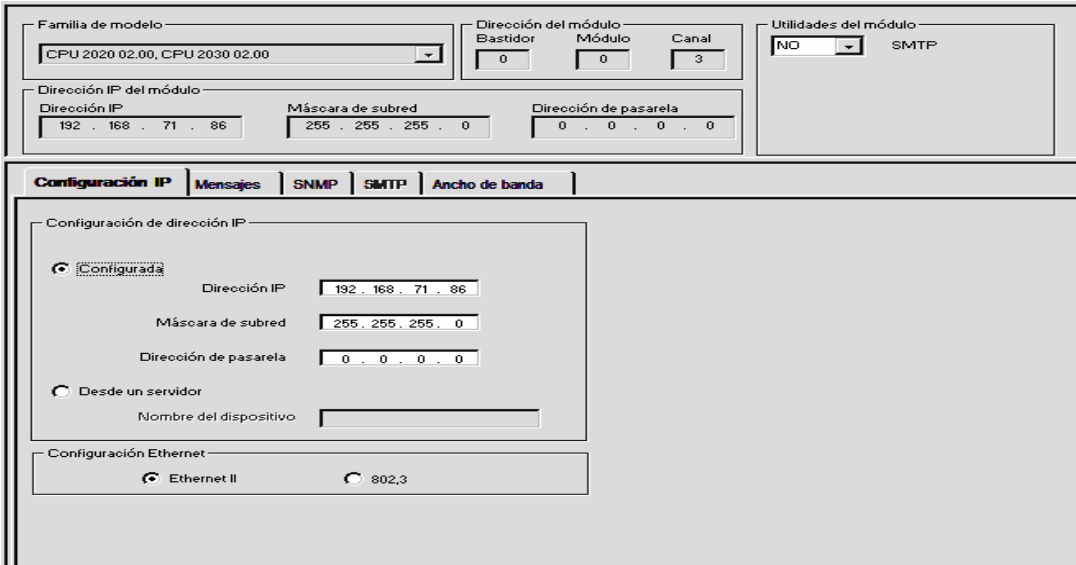


Figura 65: PARÁMETROS CONFIGURADOS DE LA RED ETHERNET

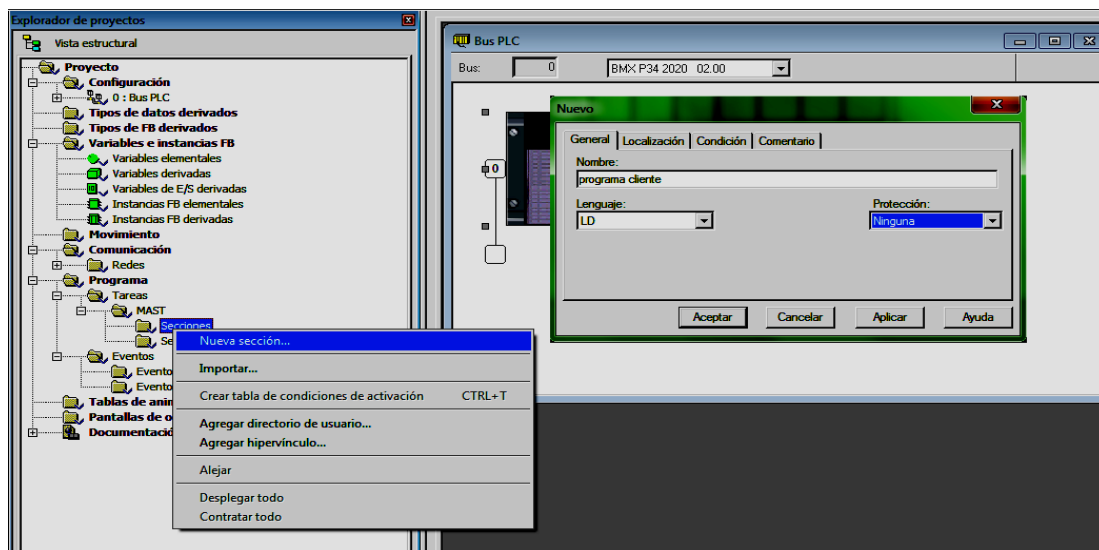


3.5.1.3 PROGRAMA DEL CLIENTE M340:

Para construir el programa del cliente se debe seguir los siguientes ítems:

Ir a la ventana de proyectos y dentro de esta ventana ir a Programa, Tareas, Mast, clic derecho en sección, clic en nueva sección y la ventana que se despliega escribir el nombre del programa y elegir el lenguaje en el que se desea trabajar. Ver figura 66.

Figura 66: CONFIGURACIÓN PROGRAMA DEL CLIENTE M340



DESCRIPCION DEL PROGRAMA CLIENTE:

Este programa es el programa del cliente (M340 con dirección IP: 192.168.71.86) que en conjunto con el programa del servidor (M340 con dirección IP: 192.168.71.87) y con el programa del cliente-servidor (HMI Magelis con IP: 192.168.71.84) hace lo siguiente: Si la entrada 1 del cliente es activada se activa

la salida 1 del servidor. Si la entrada 2 del cliente es activada se activa la salida 2 del servidor. Si la entrada 1 y la entrada 2 del cliente son activadas se activa la salida 3 del servidor. Si la entrada 1 del servidor es activada se activa la salida 1 del cliente y permanece activada aunque la entrada 1 del servidor se desactive. Si la entrada 2 del servidor se activa entonces la salida 2 del cliente se activa y permanece así aun si la entrada 2 del servidor se desactiva. Si la entrada 3 del servidor se activa las salidas 1 y 2 del cliente se desactivan. La siguiente explicación se escribe en función de lo que se visualiza en el cliente-servidor. Cuando el nivel del tanque está por debajo del 20% de su capacidad total, %I0.1.4 esta desactivada. Si el nivel del tanque está entre el 20% y el 40% de su capacidad total, %I0.1.4 estará activada. Si el nivel del tanque está entre el 40% y el 60% de su capacidad total, %I0.1.5 se activara. Para un nivel del tanque entre el 60% y el 80% de su capacidad total, %I0.1.6 estará encendida. Para un nivel del tanque entre el 80% y el 100% de su capacidad total, %I0.1.7 estará activada. Cuando el tanque está completamente lleno, %I0.1.8 se activara. Para probar esa última sección del programa tenga en cuenta que las entradas deben ser activadas en orden ascendente (4, 5, 6 7 y 8) y apagadas en orden descendente (8, 7, 6 5 y 4) como naturalmente se desplaza el nivel de un tanque cuando este se llena o se desocupa (en un proceso continuo, es decir, sin saltos o discontinuidades).

PRIMERA PARTE: USO DE LA PALABRA DE MEMORIA %MW1 DEL CLIENTE.

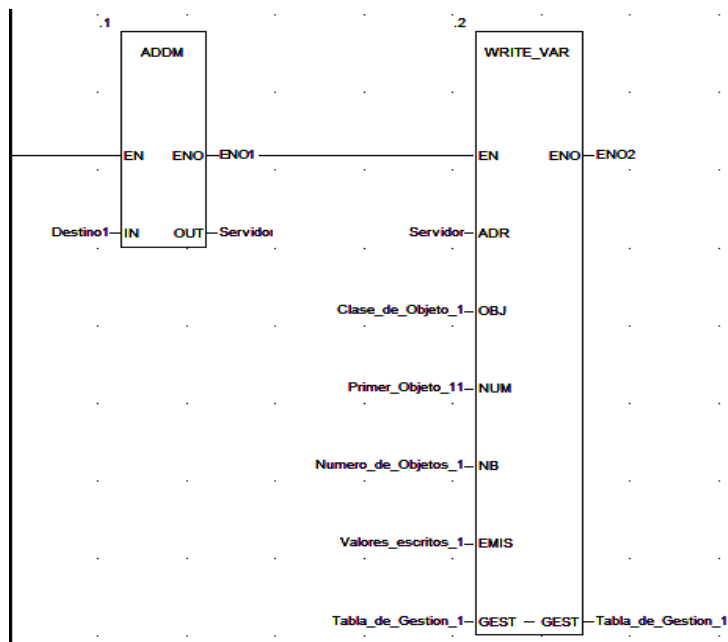
Esta palabra de memoria %MW1 del cliente será escrita en la palabra de memoria %MW1 del servidor para que con ella se pueda desarrollar las siguientes acciones. Si la entrada 1 del cliente se activa (mientras su entrada 2 esta desactivada), la palabra %MW1 del servidor valdrá 1 y por consiguiente se

activara la salida 1 del servidor. Si la entrada 2 del cliente se activa (sin que este activada su entrada 1), la palabra %MW1 del servidor valdrá 2, motivo por el cual se activara la salida 2 del servidor. Si las entradas 1 y 2 del cliente están activas la palabra %MW1 del servidor valdrá 3 y por lo tanto se activara la salida 3. Si ni la entrada 1 ni la entrada 2 del cliente están activas entonces %MW1 valdrá 0, por este motivo no se activara ninguna salida en el servidor.

PROGRAMA EN LADDER

Sección 1: Configuración del bloque WRITE_VAR utilizado para la escritura de la palabra de memoria %MW1 del cliente en la palabra de memoria %MW1 del servidor. A continuación se presenta el bloque WRITE_VAR. Ver figura 64

Figura 67: CONFIGURACIÓN DEL BLOQUE WRITE_VAR



El bloque **ADDM** de la figura 81 tiene una entrada **IN** para configurarla con un dato tipo string que tiene la dirección del dispositivo en una conexión bus o serie. Este tipo de dato se configura siguiendo este orden: **r.m.c.nodo**.

r: es el número de bastidor del procesador y debe ser siempre cero.

m: es el número de ranura del procesador del bastidor, debe ser siempre cero

C: es número de canal, es siempre cero ya que la conexión serie de un procesador es siempre el canal cero.

nodo: es el número del dispositivo al cual se le envía la solicitud. Para este programa es la dirección IP del servidor igual a 192.168.71.87

La salida **OUT** del bloque ADDM se configura con un tipo de dato ARRAY [0...7] OF INT, que tiene una matriz que representa la dirección de un dispositivo. El parámetro se puede utilizar como parámetro de entrada para varias funciones de comunicación. Esta salida se configura como servidor.

La salida **ENO1** se utiliza para la ejecución del bloque uno, y toma valores de uno cuando la ejecución a finalizado y de cero cuando la ejecución esta en proceso o no se a realizado

Bloque **WRITE_VAR** utiliza para su configuración:

Una entrada **ADR** de tipo "ARRAY [0..7] OF INT" para la dirección de la entidad de destino proporcionada por el parámetro OUT de la función ADDM

Una entrada **OBJ de tipo "STRING"** para indicar que se va a escribir palabras de memorias %MW y/o bis internos %M

Una entrada **NUM** de tipo "DINT" que indica la dirección del primer objeto que se va a escribir en el servidor, en este caso será el 1, es decir, %MW1

Una entrada **NB** de tipo "INT" con la que se indica el número de objetos consecutivos que se van a escribir, en este programa se escribe un objeto que es la palabra de memoria %MW1 del cliente.

Una entrada **EMIS** de tipo "ARRAY [n...m] OF INT", conformada por una tabla de palabras que contienen el valor de los objetos que van a escribirse, en este

ejemplo el valor que se va a escribir es la palabra %MW1 del cliente, que se escribe en la palabra de memoria % MW1 del servidor

Una entrada GEST de tipo “ARRAY [0 3] OF INT”, conformada por una tabla de gestión de intercambios que se compone de las siguientes palabras:

Palabra 1: palabra administrada por el sistema que se compone de dos bytes:

Byte más significativo: se utiliza para el número de intercambios

Byte menos significativo: este byte utiliza dos bits, el bit de actividad (rango 0) y el bit de cancelación (rango 1).

Palabra 2: palabra administrada por el sistema que se compone de dos bytes:

Byte más significativo: este byte reporta el informe de la operación

Byte menos significativo: este byte reporta el Informe de la comunicación

Palabra 3: palabra gestionada por el usuario, que define el tiempo de respuesta máximo con una base de tiempo de 100 ms.

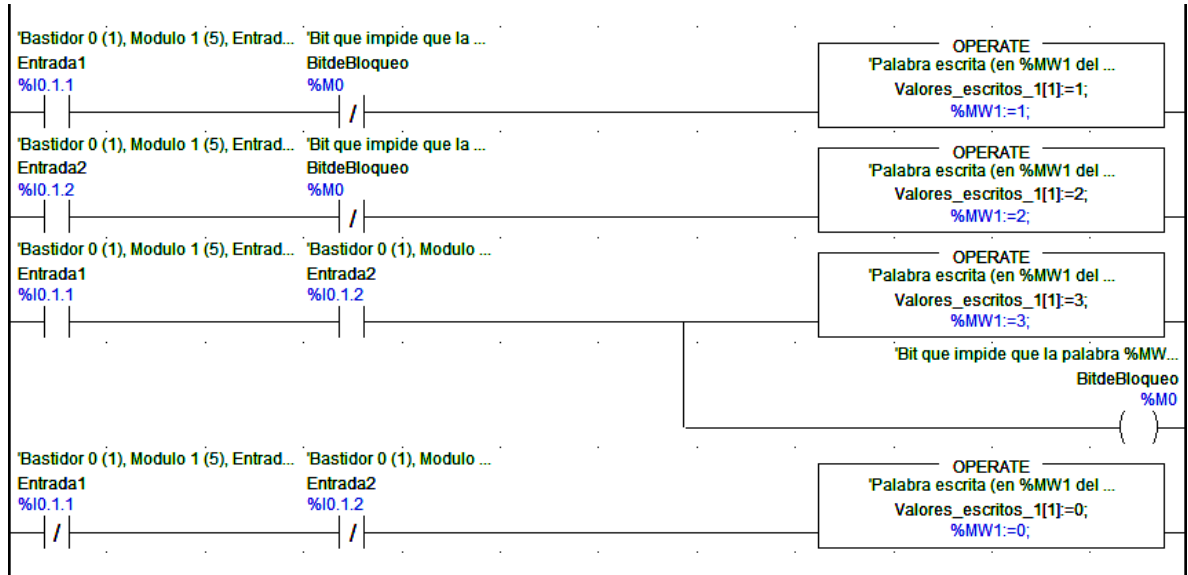
Palabra de rango 4: Palabra gestionada por el sistema, que define la duración del intercambio.

ENO2 se utiliza para la ejecución del bloque dos, y toma valores de uno cuando la ejecución a finalizado y de cero cuando la ejecución está en proceso o no sé a realizado

Sección 2: esta sección muestra las diferentes entradas que se utilizan para activar los valores de la palabra de memoria %MW1 del cliente. Ver figura 68

La figura 68 muestra los diferentes valores que toma la palabra %MW1 cuando las entradas I0.1.1 y/o I0.1.2 son activadas o desactivadas. Si se activa la entrada %I0.1.1, la palabra de memoria %MW1 toma el valor de 1. Si se activa la entrada %I0.1.2, la palabra de memoria %MW1 toma el valor de 2. Si se activan ambas entradas, se activa la salida %M0 (bit de bloqueo). Si no se activan ninguna entrada, la palabra de memoria toma el valor de 0. **Nota:** el bit de bloqueo impide que la palabra %MW1 tome los valores 1 y 2 a la vez.

Figura 68: VALORES DE %MW1 QUE SE VAN A ESCRIBIR

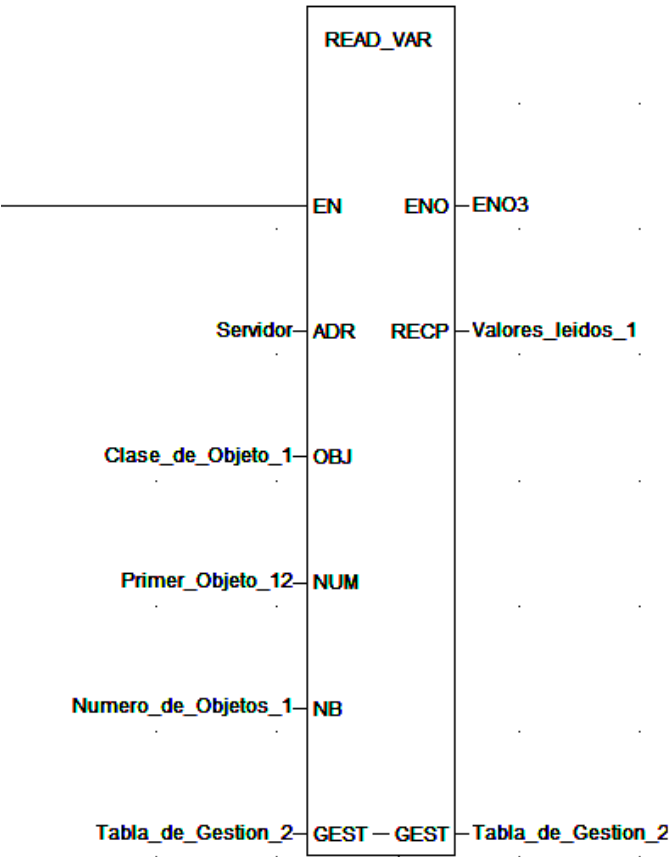


SEGUNDA PARTE: USO DE LA PALABRA DE MEMORIA %MW0 DEL SERVIDOR

La palabra de memoria %MW0 será leída por el cliente y almacenada en su palabra de memoria %MW0. Si se activa la entrada 1 del servidor, %MW0 del cliente valdrá 1 y por consiguiente la salida 1 del cliente se activará y permanecerá activada después de desactivarse la entrada 1 del servidor. Si se activa la entrada 2 del servidor entonces %MW0 del cliente valdrá 2, por lo tanto la salida 2 del cliente se activará y permanecerá así aunque la entrada 2 del servidor se desactive. Si la entrada 3 del servidor se activa, %MW0 del cliente valdrá 3, por lo tanto las salidas 1 y 2 del cliente se desactiven. Observe que la orden reset tiene prioridad sobre la orden set (las entradas 1 y 2 del servidor no tendrán efecto si la entrada 3 del servidor esta activada).

Sección 3: Configuración del bloque READ_VAR utilizado para leer la palabra de memoria %MW0 del servidor. La palabra de memoria %MW0 se almacena en la palabra de memoria %MW0 del cliente. A continuación se presenta el bloque READ_VAR. Ver figura 69

Figura 69: BLOQUE READ_VAR



La figura 69 muestra el bloque READ_VAR sin un bloque ADDM que lo preceda, debido a que el bloque READ_VAR lee en la misma dirección (servidor) en la que

escribe el bloque **WRITE_VAR** de la sección anterior, entonces la salida del bloque **ADDM** de la sección anterior también le sirve al bloque **READ_VAR** de esta sección.

Parámetros del bloque **READ_VAR**:

Entrada **ADR**: es la misma entrada **ADR** del bloque **WRITE_VAR**. Ya explicado.

Entrada **OBJ de tipo "STRING"**. Explicada en el bloque **WRITE_VAR**.

Entrada **NUM** de tipo "DINT" que indica la dirección del primer objeto que se va a leer del servidor, en este caso será el 0, es decir, %MW0

Una entrada **NB** de tipo "INT" con la que se indica el número de objetos consecutivos que se van a leer, en este programa se va a leer un objeto (%MW0).

Salida **RECP** de tipo "ARRAY [n...m] OF INT", conformada por una tabla de palabras que contienen el valor de los objetos que van a leer, en este ejemplo el valor que el cliente va a leer es la palabra %MW0 del servidor.

Entrada **GEST**. Explicada en el bloque **WRITE_VAR**

Entrada **EN**: utilizada para enlazar la salida del bloque **ADDM**

Salida **ENO3**: se utiliza para la ejecución del bloque tres, y toma valores de uno cuando la ejecución a finalizado y de cero cuando la ejecución está en proceso o no sé a realizado

Sección 4: esta sección muestra los diferentes valores de la palabra de memoria %MW0 del servidor guardados en la palabra de memoria %MW0 del cliente para activar las salidas del cliente. Ver figura 70

La figura 70 muestra. Si %MW0 del cliente vale 1, la salida 1(%Q0.2.1) del cliente se activara. Si %MW0 del cliente vale 2, la salida 2(%Q0.2.2) del cliente se activara. Si %MW0 del cliente vale 3, las salidas 1 y 2 del cliente se desactivan.

Figura 70: VALORES LEIDOS Y ALMACENADOS EN %MW0 DEL CLIENTE M340

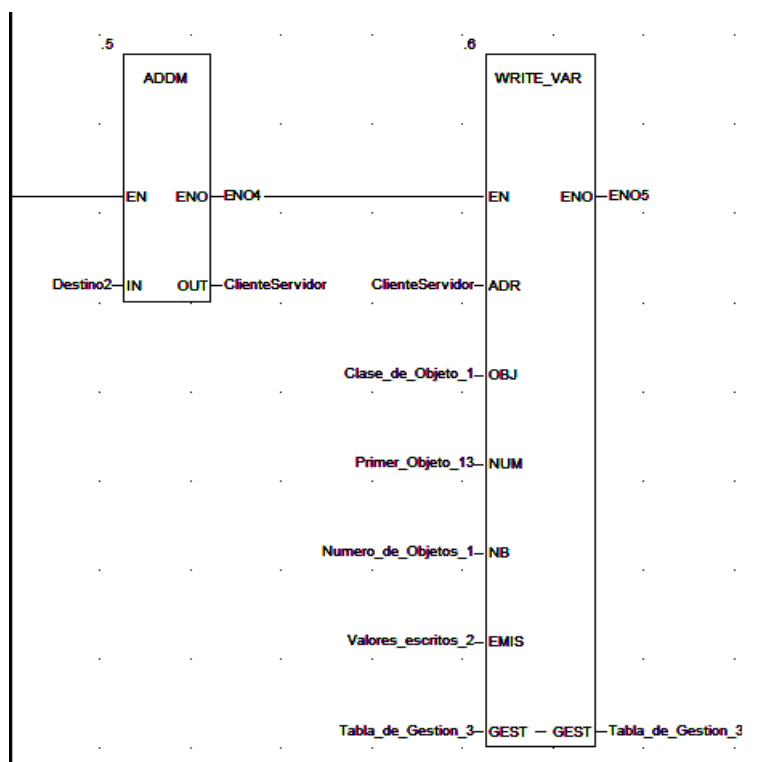


PARTE 3 ESCRITURA DE LA PALABRA DE MEMORIA %MW2 DEL CLIENTE EN LA PALABRA DE MEMORIA %MW0 DEL CLIENTE-SERVIDOR

En esta parte del programa se escribe una palabra de memoria en el cliente-servidor. Se escribe %MW2 del cliente en %MW0 del cliente-servidor para mostrar 6 de los posibles estados del nivel de un tanque. Cuando el nivel del tanque está por debajo del 20% de su capacidad total, %I0.1.4 se desactiva. Si el nivel del tanque está entre el 20% y el 40% de su capacidad total, %I0.1.4 se activa. Si el nivel del tanque está entre el 40% y el 60% de su capacidad total, %I0.1.5 se activa. Para un nivel del tanque entre el 60% y el 80% de su capacidad total, %I0.1.6 se activa. Para un nivel del tanque entre el 80% y el 100% de su capacidad total, %I0.1.7 se activa y cuando el tanque está completamente lleno, %I0.1.8 se activa. Para probar esa última sección del programa tenga en cuenta que las entradas se activan en orden ascendente (4, 5, 6 7 y 8) y se desactivan en orden descendente (8, 7, 6 5 y 4) como naturalmente se desplaza el nivel de un tanque cuando este se llena o se desocupa.

Sección 5: Configuración del bloque WRITE_VAR utilizado para la escritura de la palabra de memoria %MW2 del cliente en la palabra de memoria %MW0 del cliente-servidor. A continuación se presenta el bloque WRITE_VAR. Ver figura 71

Figura 71: BLOQUE WRITE_VAR PARA LA ESCRITURA DE %MW2 EN %MW0



El bloque **ADDM** de la figura 71 es igual al bloque ADDM de la figura 67, excepto en la dirección IP de destino, que ahora es igual a 192.168.71.84 (dirección IP cliente-servidor) y en la salida **ENO4** que es utilizada para la ejecución del bloque 5, y toma valores de uno cuando la ejecución a finalizado y de cero cuando la ejecución está en proceso o no sé a realizado

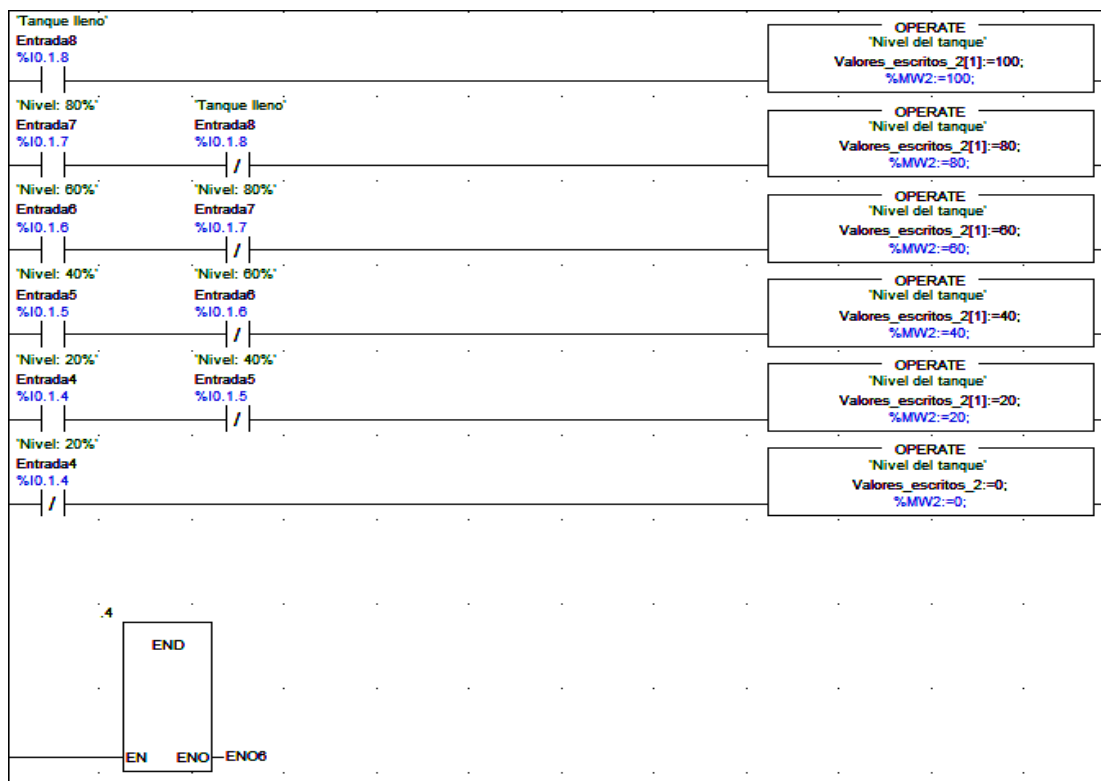
Bloque **WRITE_VAR** de la figura 71 es el mismo utiliza en la figura 67 pero con estas diferencias:

Entrada **NUM** de tipo “DINT” que indica la dirección del primer objeto que se va a escribir en el cliente-servidor, en este caso será el 2, es decir, la palabra de memoria %MW2 del cliente.

Entrada **NB** de tipo “INT” con la que se indica el número de objetos consecutivos que se van a escribir, en este programa se va a escribir un objeto (%MW2) del cliente en la palabra de memoria %MW0 del cliente-servidor.

ENO5 se utiliza para la ejecución del bloque seis, y toma valores de uno cuando la ejecución a finalizado y de cero cuando la ejecución está en proceso o no sé a realizado

Figura 72: VALORES DE LA PALABRA %MW2 DEL CLIENTE



Sección 6: muestra los valores que toma la palabra de memoria %MW2 del cliente cada vez que se varia una entrada del cliente. Si se desactiva %I0.1.4, %MW2 vale cero. Si se activa %I0.1.4, %MW2 vale 20. Si se activa %I0.1.5, %MW2 vale 40. Si se activa %I0.1.6, %MW2 vale 60. Si se activa %I0.1.7, %MW2 vale 80 y para un valor de %MW2 igual 100, %I0.1.8 debe estar activada. Ver figura 72

3.5.1.4 PROGRAMA DEL CLIENTE-SERVIDOR HMI (Magelis)

La Magelis cliente-servidor no necesita la elaboración de un programa en alguna clase de lenguaje para establecer comunicación con otros dispositivos, para esta labor se debe construir la representación gráfica del proceso (paneles) a supervisar. Para configurar la Magelis como cliente-servidor se debe seguir una serie de ítems, que a continuación se presentan.

Abra el software Vijeo designer y configure: en la primera ventana que se despliega: nombre del proyecto, descripción o comentarios, tipo de proyecto y contraseña del proyecto. En la segunda ventana que se despliega: nombre del destino, tipo de destino y modelo. En la tercera ventana que se despliega: configure la dirección IP de la Magelis, la Magelis que tiene el almacén de E3T tiene una dirección IP clase C igual a 192.168.71.84. Configure en la siguiente ventana que protocolo de comunicación va a emplearse. Para este ejercicio se utilizo la Magelis como maestro TCP/IP y esclavo TCP/IP. Ver figura 73.

Por último, clic en finalizar para terminar de construir el proyecto.

El siguiente paso es construir la representación grafica del proceso “paneles” a supervisarlos y las variables que se van a utilizar. En este ejemplo se elaboró un

panel para controlar y visualizar los procesos descritos en los programas del cliente y servidor. Ver figura 74

Figura 73: CONFIGURACIÓN DEL MAGELIS COMO CLIENTE-SERVIDORA

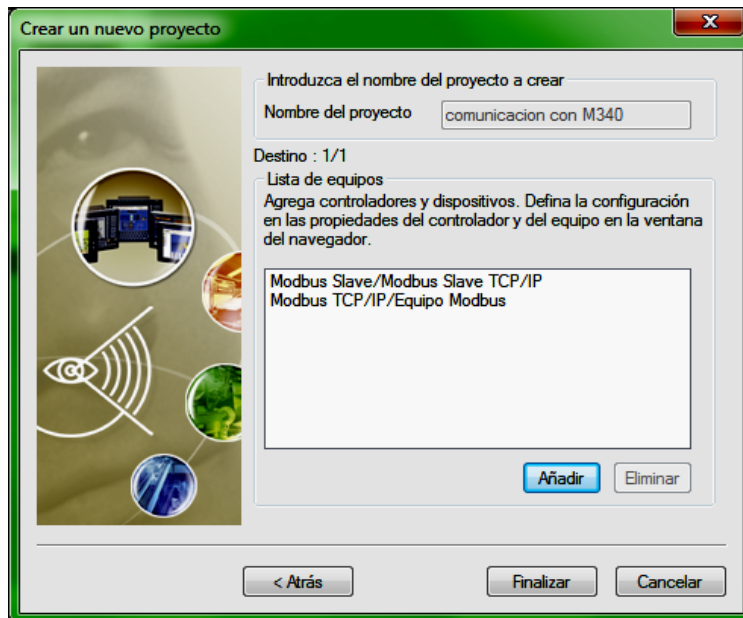
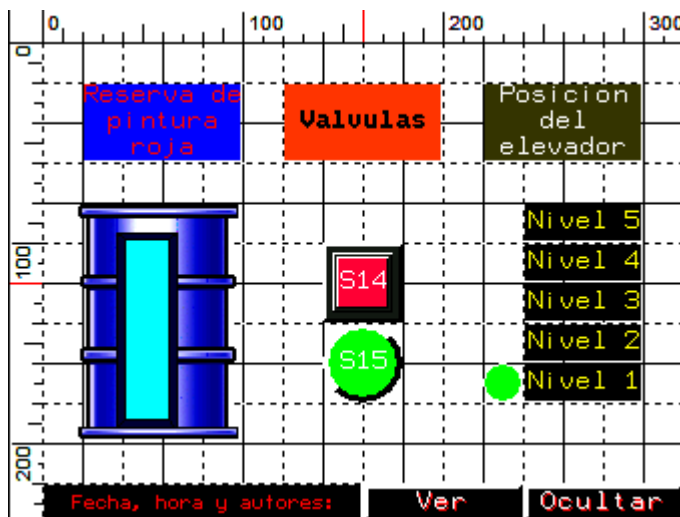
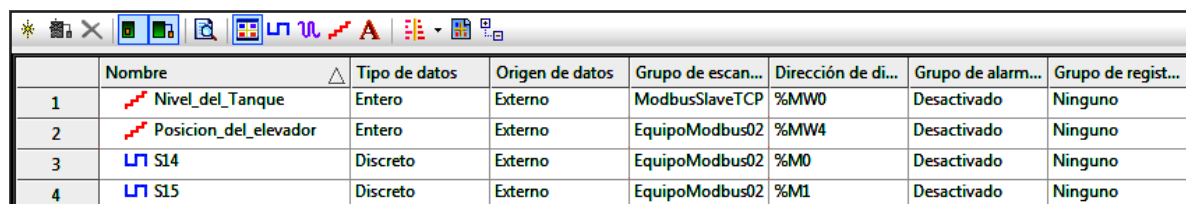


Figura 74: REPRESENTACIÓN GRAFICA DEL PROCESO



La figura 74 muestra tres procesos: primero, llenado de tanque (reserva de pintura roja), el cual se describe en el programa del cliente. Este proceso utiliza la palabra de memoria %MW0 del cliente para escribir los valores en el cliente-servidor (Magelis). Segundo, activar y desactivar las válvulas 1 y 2 del servidor. Para este proceso se escribe los valores del cliente-servidor en los bits %M0 y %M1 del servidor. Tercero, supervisión de la posición de un elevador utilizando la lectura de la palabra de memoria %MW4 del programa del servidor. Estas variables se presentan en la figura 75.

Figura 75: VARIABLES DEL CLIENTE M340 Y DEL SERVIDOR M340



	Nombre	Tipo de datos	Origen de datos	Grupo de escan...	Dirección de di...	Grupo de alarm...	Grupo de regist...
1	Nivel_del_Tanque	Entero	Externo	ModbusSlaveTCP	%MW0	Desactivado	Ninguno
2	Posicion_del_elevador	Entero	Externo	EquipoModbus02	%MW4	Desactivado	Ninguno
3	S14	Discreto	Externo	EquipoModbus02	%M0	Desactivado	Ninguno
4	S15	Discreto	Externo	EquipoModbus02	%M1	Desactivado	Ninguno

Por último vamos a la ventana del navegador para configurar la Magelis como cliente y servidor. Primero se configura la Magelis como servidor de la siguiente manera: se hace clic en Administrador de E/S, clic en ModbusSlave01, doble clic en ModbusSlave01TCPIP01 para que se despliegue la ventana de la figura 76 en la cual se debe configurar: Máximas conexiones, en este caso uno. Tiempo de espera de Inactividad. Dirección del equipo esclavo y por último se elige IEC61131 para que la Magelis trabaje con sintaxis estándar que hace referencia a la utilización de palabras %MW y bits %M.

Figura 76: CONFIGURACIÓN MAGELIS COMO SERVIDORA

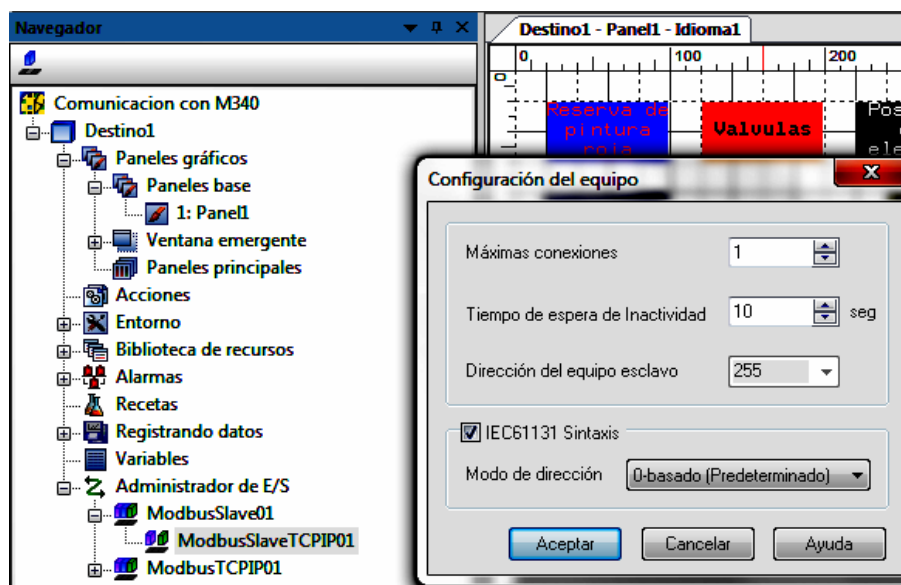
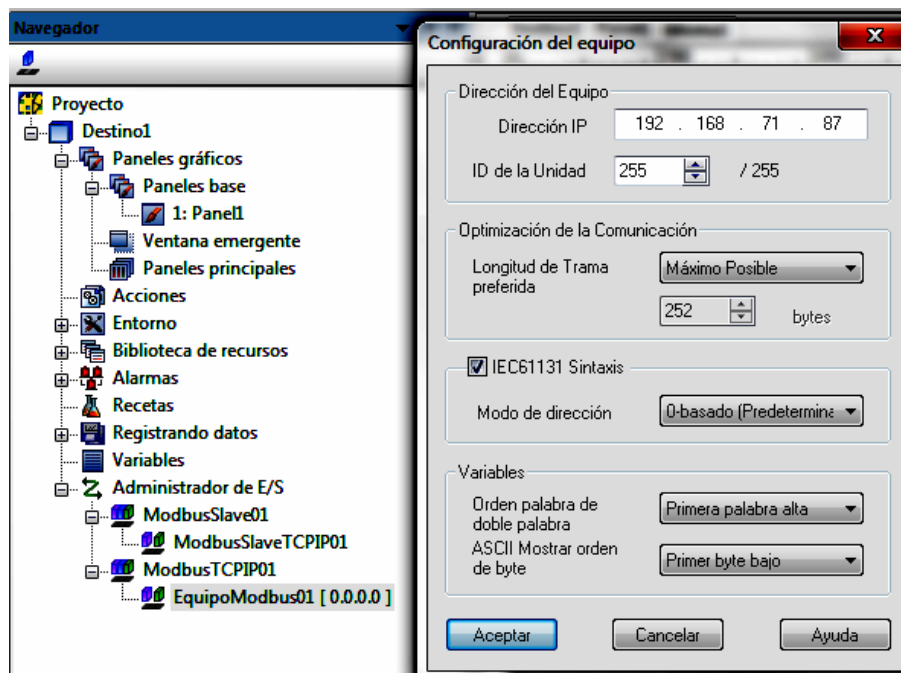


Figura 77: CONFIGURACIÓN MAGELIS COMO CLIENTE



Segundo configuración de la Magelis como cliente, para ello de clic en Administrador de E/S, clic en Modbus TCPIP y doble clic en Equipo Modbus01 [0.0.0.0] para configurar la ventana de la figura 77

La dirección IP de la figura 77 es la dirección IP del servidor

3.5.1.5 PROGRAMA SERVIDOR:

RESUMEN DEL PROGRAMA. Este programa es el programa del servidor (M340 con IP: 192.168.71.87), que en conjunto con el programa del cliente (M340 con dirección IP: 192.168.71.86) y con el programa del cliente-servidor (HMI Magelis con dirección IP: 192.168.71.84) hace lo siguiente: Si la entrada 1 del cliente está activada se activa la salida 1 del servidor. Si la entrada 2 del cliente está activa se activa la salida 2 del servidor; si la entrada 1 y la entrada 2 del cliente están activas se activa la salida 3 del servidor. Si la entrada 1 del servidor está activa se activa la salida 1 del cliente y permanece activada aunque la entrada 1 del servidor se desactive. Si la entrada 2 del servidor se activa entonces la salida 2 del cliente se activa y permanece así aun si la entrada 2 del servidor se desactiva. Si la entrada 3 del servidor se activa, las salidas 1 y 2 del cliente se desactivan. A continuación se explica la relación entre el servidor y lo que se ve en el cliente-servidor: si la entrada 5 esta activada, el ascensor esta en el nivel 1. Si se activa la entrada 6, el elevador esta en el nivel 2 y así sucesivamente hasta llegar al quinto y último nivel donde la entrada 9 se activa. Lo último que hace el programa es activar las salidas 14 y 15 en función del estado de 2 bits (%M0 y %M1) que escribe el cliente-servidor en el servidor.

PRIMERA PARTE: USO DE LA PALABRA DE MEMORIA %MW1 DEL CLIENTE ESCRITA EN LA PALABRA DE MEMORIA %MW1 DEL SERVIDOR:

La palabra de memoria %MW1 del cliente será escrita en la palabra de memoria %MW1 del servidor para poder desarrollar las siguientes acciones. Si la entrada 1 del cliente se activa (mientras su entrada 2 esta desactivada), la palabra %MW1 del servidor valdrá 1 y por consiguiente se activa la salida 1 (%Q0.2.1) del servidor. Si la entrada 2 del cliente se activa (sin que este activada su entrada 1), la palabra %MW1 del servidor valdrá 2, motivo por el cual se activa la salida 2 (%Q0.2.2) del servidor. Si las entradas 1 y 2 del cliente son activadas la palabra %MW1 del servidor valdrá 3 y por lo tanto se activa la salida 3 (%Q0.2.3) del servidor. Si ni la entrada ni la entrada 2 del cliente están activadas %MW1 vale 0, luego no se activará ninguna salida en el servidor. Ver figura 78

Figura 78: VALORES ESCRITOS EN LA PALABRA %MW1 DEL SERVIDOR
M340

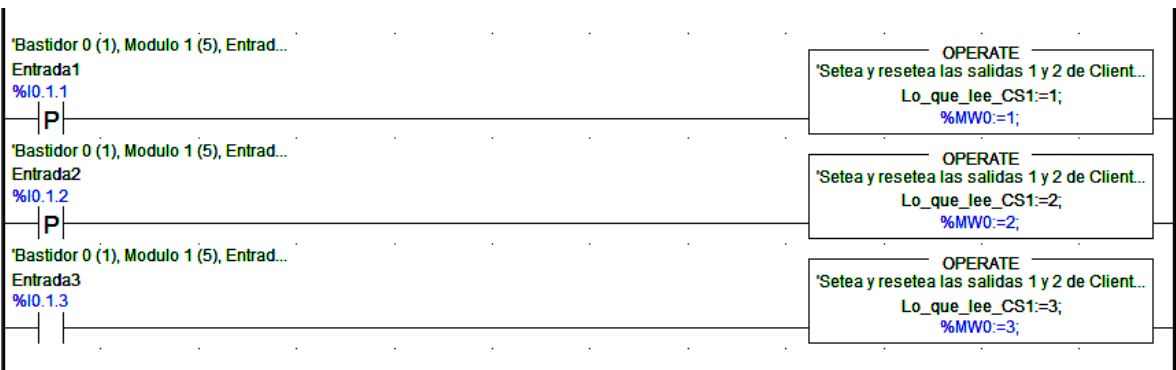


La figura 78 muestra los valores escritos en la palabra de memoria %MW1 del servidor para activar las diferentes salidas.

SEGUNDA PARTE: PALABRA DE MEMORIA %MW0 DEL SERVIDOR LEIDA POR EL CLIENTE:

En esta parte del programa la palabra de memoria %MW0 del servidor será leída por el cliente y almacenada en su palabra de memoria %MW0 para realizar las siguientes acciones Si se activa la entrada 1 del servidor, %MW0 del cliente valdrá 1 y por consiguiente la salida 1 del cliente se activará y permanecerá activada después de desactivarse la entrada 1 del servidor. Si se activa la entrada 2 del servidor, %MW0 del cliente valdrá 2, por lo tanto la salida 2 del cliente se activara y permanecerá así aunque la entrada 2 del servidor se desactive. Si la entrada 3 del servidor se activa, %MW0 del servidor valdrá 3, lo que causara las salidas 1 y 2 del cliente se desactiven. Observe que la orden reset tiene prioridad sobre la orden set (las entradas 1 y 2 del servidor no tendrán efecto si la entrada 3 del servidor esta activada). Ver figura 79.

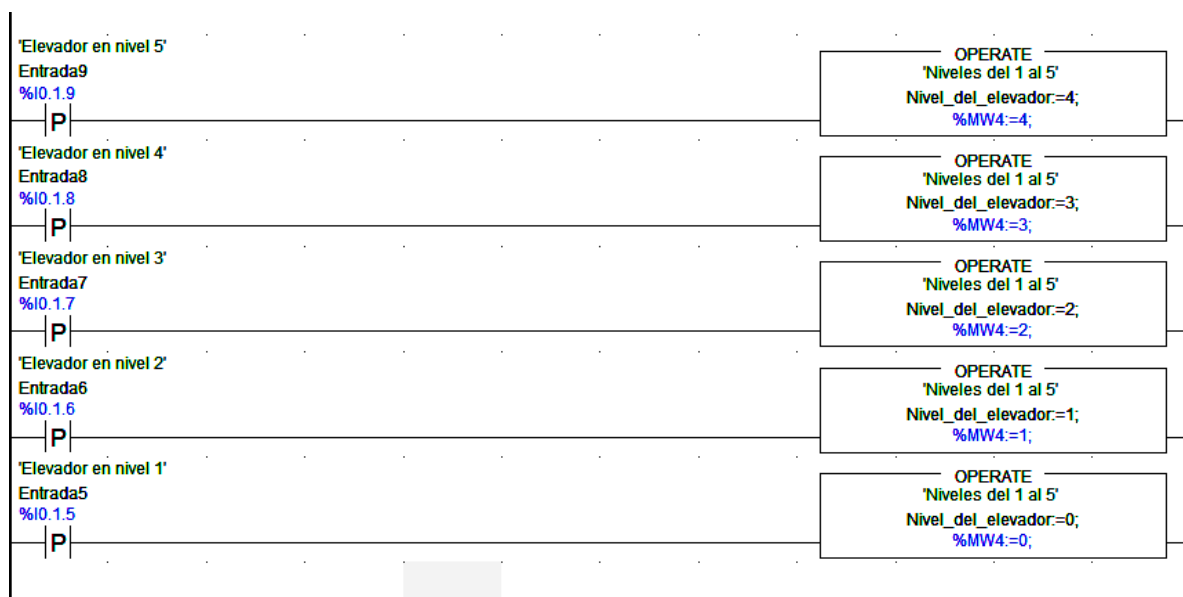
Figura 79: VALORES %MW0 DEL SERVIDOR M340 QUE SE VAN A LEER



TERCERA PARTE: PALABRA DE MEMORIA %MW4 DEL SERVIDOR LEIDA POR EL CLIENTE:

La palabra de memoria %MW4 será una palabra que lee el cliente-servidor para supervisar la posición de un elevador o ascensor. Las entradas de esta sección (de la 5 a la 9) representan sensores (por ejemplo ópticos) en cada piso de una edificación. Si la entrada 5 esta activada, el ascensor esta en el nivel 1; si se activa la entrada 6, el elevador esta en el nivel 2 y así sucesivamente hasta llegar al quinto y último nivel cuando la entrada 9 se activa. Ver figura 80

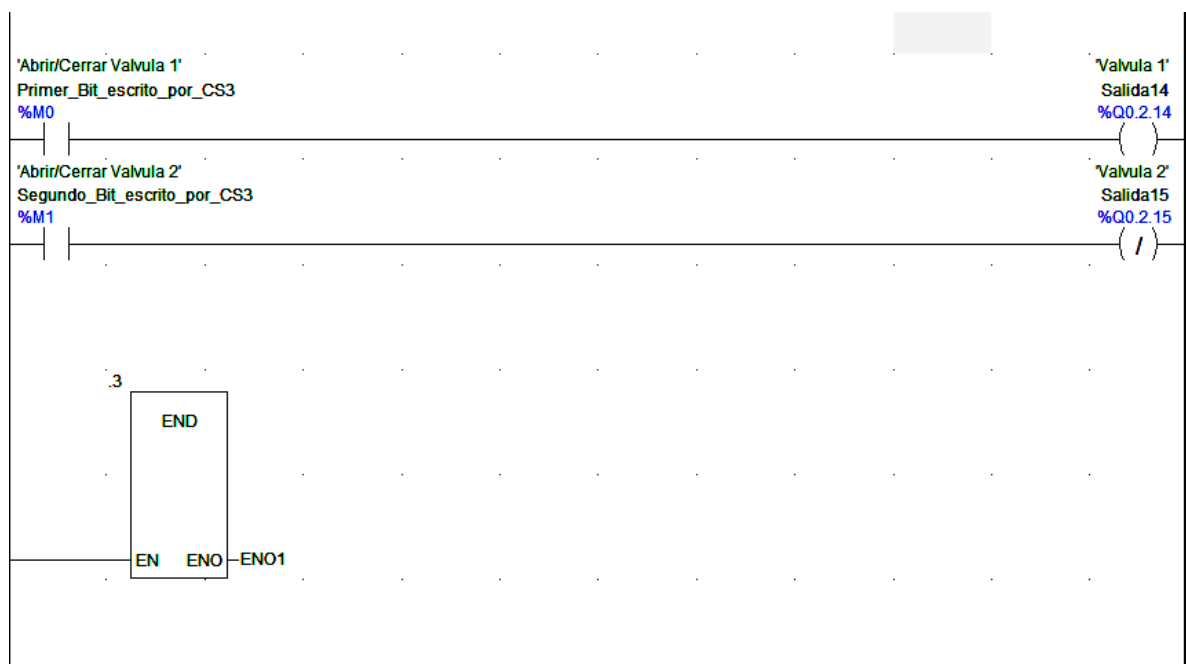
Figura 80: VALORES DE LA PALABRA %MW4 DEL SERVIDO M340



La figura 80 muestra los diferente valores que toma la palabra de memoria %MW4 para que la Magelis lea estos valores y supervise el proceso de elevación a medida que se vayan activando las diferentes entradas (pulsos) del servidor

CUARTA PARTE: BITS %M0 Y %M1 ESCRITOS POR EL CLIENTE-SERVIDOR EN EL SERVIDOR.

Figura 81: BITS %M1 Y %M2 DEL SERVIDOR M340



En la figura 81, el cliente-servidor escribe en los Bits del servidor (%M0 y %M1) para activar/desactivar la válvula 1 (Salidas 14) y desactivar/activar la válvula 2 (salida 15).

Por último se presenta en este documento las variables utilizadas por el cliente y el servidor, estas variables se encuentra en editor de datos de cada programa.

Figura 82: VARIABLES DEL PROGRAMA DEL CLIENTE M340

Nombre	Tipo	Dirección	Valor	Comentario
BitdeBloqueo	EBOOL	%M0		Bit que impide que la palabra %MW1 tome los valores 1 y 2
Clase_de_Obj...	STRING		%MW	Indica a los bloques WRITE_VAR y READ_VAR que se van a escribir/leer palabras de memoria
ClienteServidor	ARRAY[0..7] O...			Salida del bloque 5 (ADDM): tipo de dato requerido por la entrada ADR del bloque 6
Destino1	string[32]		0.0.3(192.1...	Entrada del bloque 1 (ADDM)
Destino2	string[32]		0.0.3(192.1...	Entrada del bloque 5 (ADDM)
ENO1	EBOOL			Ejecucion del bloque 1 finalizada (1); en proceso o no realizada (0)
ENO2	EBOOL			Ejecucion del bloque 2 finalizada (1); en proceso o no realizada (0)
ENO3	EBOOL			Ejecucion del bloque 3 finalizada (1); en proceso o no realizada (0)
ENO4	EBOOL			Ejecucion del bloque 4 finalizada (1); en proceso o no realizada (0)
ENO5	EBOOL			Ejecucion del bloque 5 finalizada (1); en proceso o no realizada (0)
ENO6	EBOOL			Ejecucion del bloque 6 finalizada (1); en proceso o no realizada (0)
Entrada0	EBOOL	%I0.1.0		Run/Stop
Entrada1	EBOOL	%I0.1.1		Bastidor 0 (1), Modulo 1 (5), Entrada 1 (32)
Entrada2	EBOOL	%I0.1.2		Bastidor 0 (1), Modulo 1 (5), Entrada 2 (32)
Entrada3	EBOOL	%I0.1.3		No utilizada (Reservada)
Entrada4	EBOOL	%I0.1.4		Nivel: 20%
Entrada5	EBOOL	%I0.1.5		Nivel: 40%
Entrada6	EBOOL	%I0.1.6		Nivel: 60%
Entrada7	EBOOL	%I0.1.7		Nivel: 80%
Entrada8	EBOOL	%I0.1.8		Tanque lleno
Numero_de_O...	INT		1	Indica la cantidad de objetos escritos/leídos por WRITE_VAR y READ_VAR
Primer_Objeto...	DINT		1	Indica que el primer objeto MW escrito en el servidor sera el 1, es decir, %MW1
Primer_Objeto...	DINT		0	Indica que el primer objeto MW leído en el servidor sera el 0, es decir, %MW0
Primer_Objeto...	DINT		0	Indica que el primer objeto MW escrito en el cliente-servidor sera el 0, es decir, %MW0
Salida1	EBOOL	%Q0.2.1		Bastidor 0 (1), Modulo 2 (5), Salida 1 (16)
Salida2	EBOOL	%Q0.2.2		Bastidor 0 (1), Modulo 2 (5), Salida 2 (16)
Salida3	EBOOL	%Q0.2.3		Bastidor 0 (1), Modulo 2 (5), Salida 3 (16)
Servidor	ARRAY[0..7] O...			Salida del bloque 1 (ADDM)
Tabla_de_Ges...	ARRAY[1..4] O...		%MW3	Palabras gestionadas por el sistema
Tabla_de_Ges...	ARRAY[1..4] O...		%MW8	Palabras gestionadas por el sistema
Tabla_de_Ges...	ARRAY[1..4] O...		%MW12	Palabras gestionadas por el sistema
Valores_escrit...	ARRAY[1..1] O...		%MW1	Palabra escrita (en %MW1 del Servidor)
Valores_escrit...	ARRAY[1..1] O...		%MW2	Nivel del tanque
Valores_leidos...	ARRAY[1..1] O...		%MW0	Palabra leída (%MW0 del Servidor)

Figura 83: VARIABLES DEL PROGRAMA DEL SERVIDOR M340

Nombre	Tipo	Dirección	Valor	Comentario
ENO1	BOOL			Salida de Bloque "END"
Entrada0	EBOOL	%I0.1.0		Run/Stop
Entrada1	EBOOL	%I0.1.1		Bastidor 0 (1), Modulo 1 (5), Entrada 1 (32)
Entrada2	EBOOL	%I0.1.2		Bastidor 0 (1), Modulo 1 (5), Entrada 2 (32)
Entrada3	EBOOL	%I0.1.3		Bastidor 0 (1), Modulo 1 (5), Entrada 3 (32)
Entrada4	EBOOL	%I0.1.4		No utilizada (reservada)
Entrada5	EBOOL	%I0.1.5		Elevador en nivel 1
Entrada6	EBOOL	%I0.1.6		Elevador en nivel 2
Entrada7	EBOOL	%I0.1.7		Elevador en nivel 3
Entrada8	EBOOL	%I0.1.8		Elevador en nivel 4
Entrada9	EBOOL	%I0.1.9		Elevador en nivel 5
Lo_que_escib...	INT	%MW1		Activa salidas 1,2 y 3 de servidor en funcion de las entradas de Cliente
Lo_que_lee_C...	INT	%MW0		Setea y resetea las salidas 1 y 2 de Cliente en funcion de las entradas 1,2 y 3 del servidor
Nivel_del_elev...	INT	%MW4		Niveles del 1 al 5
Primer_Bit_esc...	EBOOL	%M0		Abrir/Cerrar Valvula 1
Salida1	EBOOL	%Q0.2.1		Bastidor 0 (1), Modulo 2 (5), Salida 1 (32)
Salida2	EBOOL	%Q0.2.2		Bastidor 0 (1), Modulo 2 (5), Salida 2 (32)
Salida3	EBOOL	%Q0.2.3		Bastidor 0 (1), Modulo 2 (5), Salida 3 (32)
Salida4	EBOOL	%Q0.2.4		Bastidor 0 (1), Modulo 2 (5), Salida 3 (32)
Salida14	EBOOL	%Q0.2.14		Valvula 1
Salida15	EBOOL	%Q0.2.15		Valvula 2
Segundo_Bit_...	EBOOL	%M1		Abrir/Cerrar Valvula 2

ANEXO 4: MANUAL CANopen

CONTENIDO

	Pág.
4 PASOS PARA CREAR UNA APLICACIÓN CANOpen CON TWIDOSUITE	418
4.1 CREEACIÓN DE UN PROYECTO NUEVO	418
4.1.1 ADICIÓN DEL MÓDULO CANopen EN EL AUTÓMATA TWIDO	419
4.1.2 CONSTRUCCIÓN DE LA RED CANopen	420
4.2 PRIMER PROGRAMA: PROGRAMA LADDER SIN MACROS DRIVE QUE EJECUTA LA COMUNICACIÓN ENTRE UN PLC MAESTRO TWIDO Y UN ESCLAVO VARIADOR DE VELOCIDAD ALTIVAR 31.	424
4.2.1 DESCRIPCION DEL PROGRAMA	424
4.2.1.1 PRIMERA PARTE: ESTADOS DEL ESCLAVO ATV 31	424
4.2.1.2 SEGUNDA PARTE: Estados Pre-operacional y Operacional (NTM)	425
4.2.1.3 TERCERA PARTE : LECTURA Y ESCRITURA DE SDOs	428
4.2.1.4 CUARTA PARTE: LECTURA Y ESCRITURA DE PDOs	432
4.3 SEGUNDO PROGRAMA: PROGRAMA LADDER CON MACROS DRIVE QUE EJECUTA LA COMUNICACIÓN ENTRE UN PLC MAESTRO TWIDO Y UN ESCLAVO VARIADOR DE VELOCIDAD ALTIVAR 31.	433
4.3.1 RED CANopen	434
4.3.2 CONFIGURACIÓN DE LAS MACRODRIVE	437
4.3.3 PRIMERA PARTE: ACCIONES QUE REALIZA EL MOTOR # 1	439
4.3.4 SEGUNDA PARTE: ACCIONES QUE REALIZA EL MOTOR # 2	443
4.4 TABLAS DE ANIMACIÓN	448

LISTA DE FIGURAS

	Pág.
FIGURA 1: CREACIÓN DE UN PROYECTO NUEVO	418
FIGURA 2: PLC TWIDO TWDLCDE40DRF	419
FIGURA 3: PLC TWIDO CON MÓDULO TWDNCO1M	420
FIGURA 4: SELECCIÓN DEL DISPOSITIVO ESCLAVO	421
FIGURA 5: PARÁMETROS DE LA RED	422
FIGURA 6: CONFIGURACIÓN DEL ESCLAVO	422
FIGURA 7: PDOs DEL ESCLAVO BASIC_ATV31 (V1.1)	423
FIGURA 8: SECUENCIA DE ESTADOS DEL ATV31	425
FIGURA 9: ESTADOS PRE-OPERACIONAL Y OPERACIONAL	426
FIGURA 10: TABLA MÁQUINA DE ESTADO DE CANopen	426
FIGURA 11: INSTRUCCIÓN CAN_CMD	427
FIGURA 12: ESCRITURA DE UN SDO	429
FIGURA 13: LECTURA DE UN SDO	430
FIGURA 14: GESTIÓN DE LA COMUNICACIÓN	431
FIGURA 15: ESCRITURA Y LECTURA DEL PDO % QWC2.0.0	432
FIGURA 16: ESCRITURA Y LECTURA DEL PDO %QWC2.0.1	433
FIGURA 17: RED CANOPEN	434
FIGURA 18: PARÁMETROS DE LA RED CANopen	435
FIGURA 19: CONFIGURACIÓN DEL ESCLAVO 1	435
FIGURA 20: CONFIGURACIÓN DEL ESCLAVO 2	436
FIGURA 21: PDOs DEL ESCLAVO 1	436
FIGURA 22: PDOs DEL ESCLAVO 2	437
FIGURA 23: VENTANA DONDE SE ENCUENTRA LOS MACROS DRIVE	438
FIGURA 24: MACROS DRIVE PARA EL ESCLAVO 1	438
FIGURA 25: MACROS DRIVE PARA EL ESCLAVO 2	439
FIGURA 26: ACTIVACIÓN DE LAS MACROS DRIVE DEL MOTOR 1	440

FIGURA 27: FUNCIÓN D_RUN_FWD 1	440
FIGURA 28: FUNCIÓN D_RUN_REV 1	441
FIGURA 29: FUNCIÓN D_STOP 1	441
FIGURA 30	442
FIGURA 31 ACTIVACIÓN DE LA MACRO DRIVE DEL MOTOR 2	443
FIGURA 32: FUNCIÓN D_RUN_FWD 2	443
FIGURA 33: FUNCIÓN D_RUN_REV 2	444
FIGURA 34: FUNCIÓN D_STOP 2	444
FIGURA 35: VELOCIDADES 1,2 Y 3 DEL MOTOR 2	445
FIGURA 36: VELOCIDADES 4,5 Y 6 DEL MOTOR 2	446
FIGURA 37: VELOCIDADES 7,8 Y 9	446
FIGURA 38: VELOCIDADES 10,11 Y 12	447
FIGURA 39: VELOCIDADES 13,14 Y 15 DEL MOTOR 2	447
FIGURA 40	448
FIGURA 41: TABLA DE ANIMACIÓN PRIMER PROGRAMA	449
FIGURA 42: TABLA DE ANIMACIÓN SEGUNDO PROGRAMA	449
FIGURA 43: TABLA CONFIGURACIÓN DEL ALTIVAR 31	450

4 PASOS PARA CREAR UNA APLICACIÓN CANOpen CON TWIDOSUITE

4.1 CREEACIÓN DE UN PROYECTO NUEVO:

En la parte izquierda de su pantalla se elije crear un proyecto nuevo de las siguientes opciones, crear un proyecto nuevo, abrir un proyecto, abrir un proyecto reciente. Cuando se elije crear un proyecto nuevo, se despliega a la derecha de estas opciones la información del proyecto, con los siguientes ítems: proyecto, directorio (elije donde quiere guardar el proyecto, como mis documentos, una memoria USB, etc.), autor, departamento, índice y propiedad industrial. Al concluir estas tareas se hace clic en crear

FIGURA 1: CREACIÓN DE UN PROYECTO NUEVO

Gestión de proyectos

Actualizar la información del proyecto

Crear un proyecto nuevo

Abrir un proyecto.

Abrir un proyecto reciente.

Copiar el proyecto actual

Cerrar el proyecto actual.

Crear un proyecto nuevo

En esta pantalla puede:

- seleccionar el nombre del proyecto e introducir la información del proyecto mediante un formulario;
- adjuntar una foto en la ficha "Imagen".

Haga clic en "Crear" para crear el proyecto.

Crear

Información del proyecto

Información del archivo

Proyecto: REDES AS-i

Directorio: H:\PLAN DE PROYECTO DE GRADO\PROGRAMAS TWIDO SUITE\Gra

Información del proyecto

Autor: Jose manuel

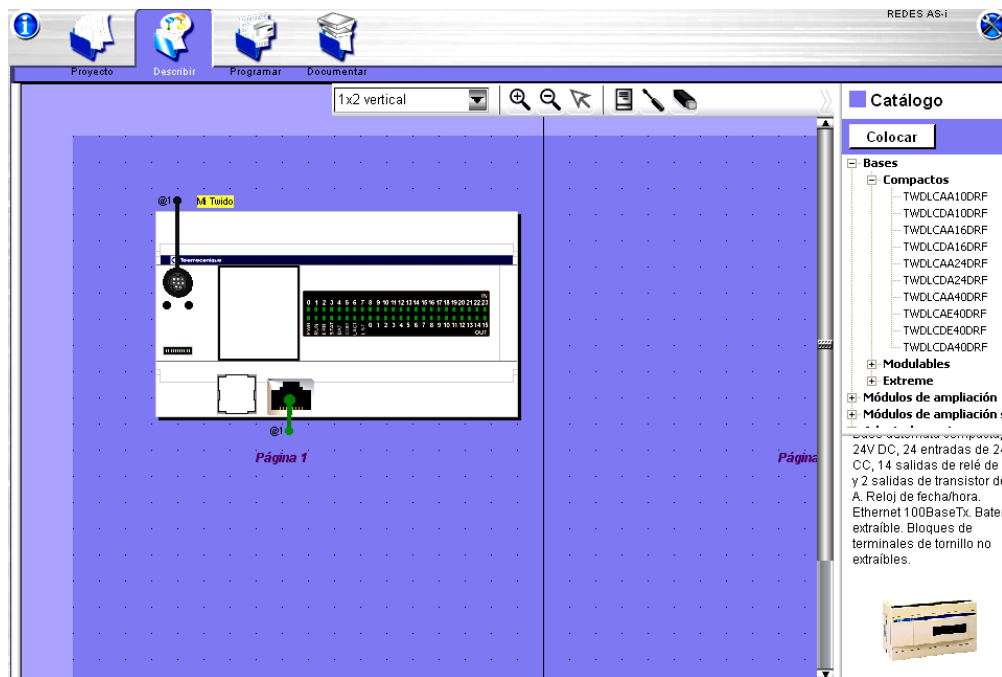
Departamento: xxxxx

Índice: xxxxx

Propiedad industrial: UIS

Comentario | **Descripción** | **Imagen**

FIGURA 2: PLC TWIDO TWDLCD40DRF



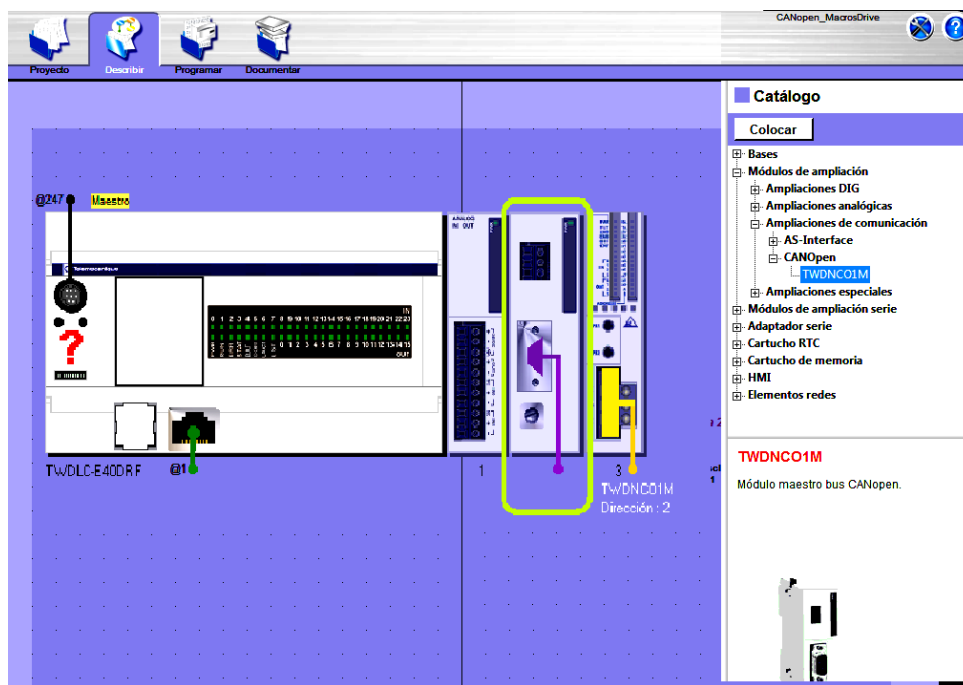
Ya creado el proyecto se va a describir. Se elimina el PLC que aparece en la pantalla y dentro de la opción catalogo se elije el PLC TWIDO TWDLCD40DRF, de la siguiente manera. Se elije bases, en bases se elije compactas y en compactas se elije el autómata ya mencionado. Seguidamente se ubica el autómata en la pantalla. Ver figura 2

4.1.1 ADICIÓN DEL MÓDULO CANopen EN EL AUTÓMATA TWIDO: Esto se hace de la siguiente manera:

Se Va al catalogo y se elije módulos de ampliación, en módulos de ampliación se elije ampliaciones de comunicación, en ampliaciones de comunicaciones se elije CANopen y en CANopen se elije el módulo TWDNCO1M.

Nota: el esquema tiene que corresponder con el autómata físico (PLC real), en número de módulos y el orden entre ellos. La figura 3 muestra el módulo maestro de expansión CANopen en la segunda posición, después del módulo de salidas y antes del módulo AS-i.

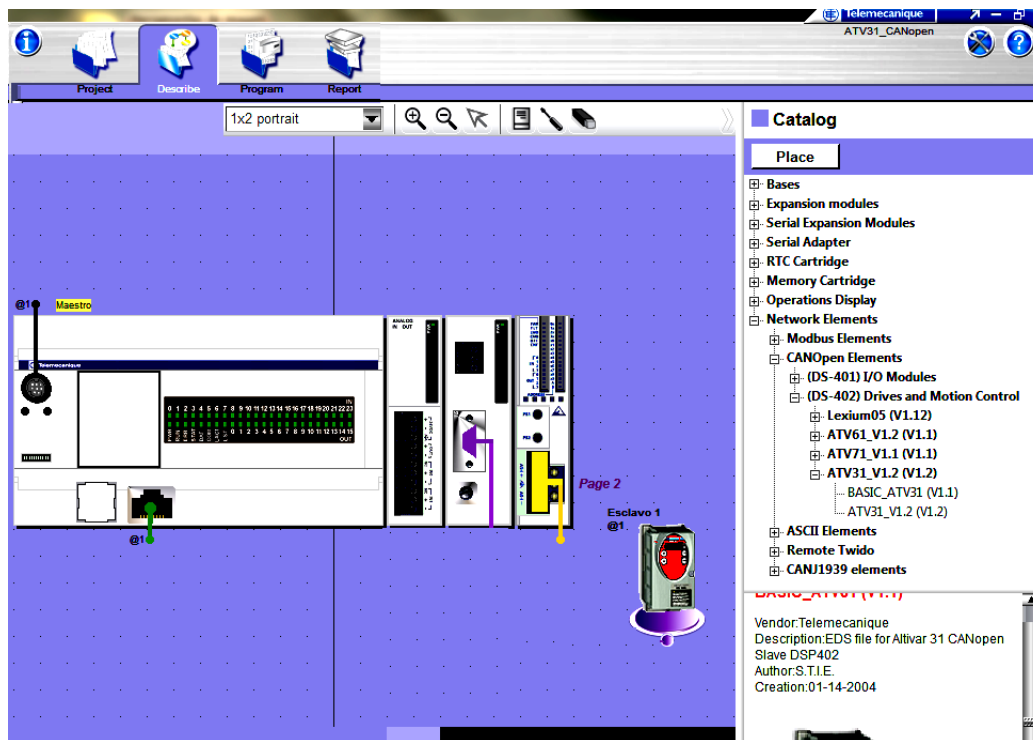
FIGURA 3: PLC TWIDO CON MÓDULO TWDNCO1M



4.1.2 CONSTRUCCIÓN DE LA RED CANopen : Para construir la red CANopen se debe hacer lo siguiente:

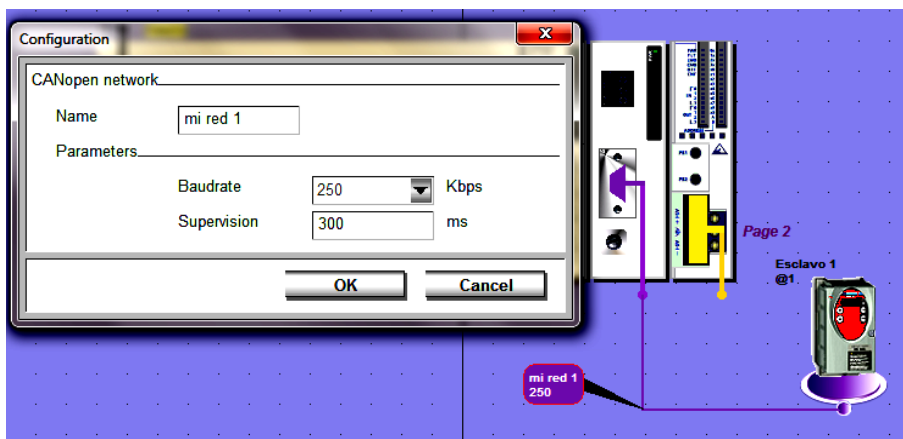
- ❖ Seleccionar el dispositivo esclavo que se desea utilizar. Para este manual se seleccionó el dispositivo esclavo BASIC_ATV31 (V1.1) de la siguiente forma: se seleccionó catálogo, elementos redes, elementos CANopen y BASIC_ATV31 (V1.1). Ver figura 4

FIGURA 4: SELECCIÓN DEL DISPOSITIVO ESCLAVO



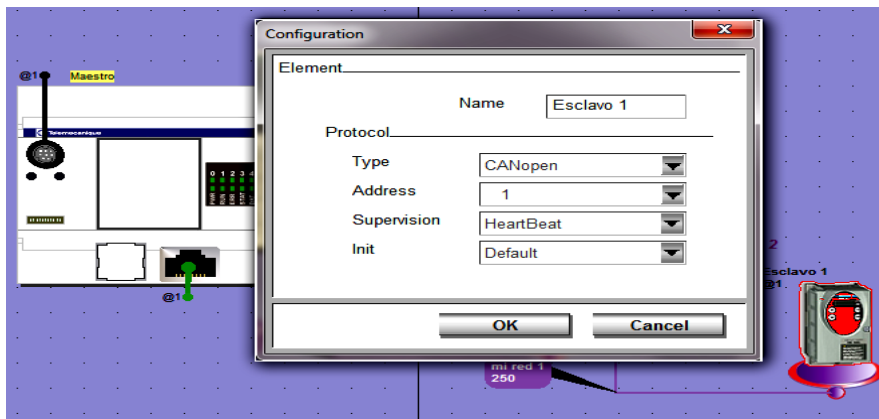
- ❖ unir los nodos. Esto se realiza haciendo clic en uno de los nodos y con el botón del ratón presionado unir el otro nodo.
- ❖ Configurar el nombre de la red, la velocidad de transmisión de datos y supervisión. Estos parámetros se configuran al hacer doble clic en la red. Ver figura 5. Es importante señalar que la velocidad de transmisión de datos debe ser configurado igual en el o los esclavos que en la red, para el dispositivo BASIC_ATV31 (V1.1) se seleccionó una velocidad de 250 Kbps

FIGURA 5: PARÁMETROS DE LA RED



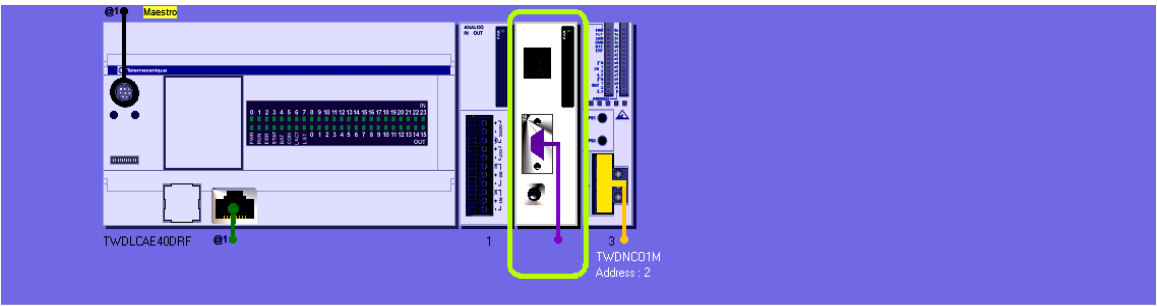
- ❖ Configurar los esclavos. En este manual se va a configurar el esclavo ATV31, para hacer esta configuración se hace clic con el botón derecho sobre el ATV31 para seleccionar Configuración. Cuando se selecciona configuración se despliega una ventana con los parámetros CANopen que necesita el ATV31 para ser configurado como: nombre del dispositivo, tipo, dirección y supervisión. Ver figura 6.

FIGURA 6: CONFIGURACIÓN DEL ESCLAVO



El altivar 31 trae por defecto PDOs para su configuración, los cuales se pueden observar al hacer doble clic en el módulo maestro TWDNCO1M CANopen en la pestaña Programar, Configurar. Ver figura 7.

FIGURA 7: PDOs DEL ESCLAVO BASIC_ATV31 (V1.1)



Description of the module

Reference number: Address:

Description: CANopen bus master module.

Module configuration.

#	Slave	Type	Supervision	Init	Used	Address	Symbol	Object	Size
1	Esclavo_1	BASIC_ATV31 (V1.1)	None	Default	☒	%IWC2.0.0	D_STATUS_ESCLAVO_	Drivecom status register;	16
					☐	%IWC2.0.1	D_CONTROL_ESCLAV	Control effort;	16
					☐	%IWC2.0.2	D_ERROR_ESCLAVO_	Error code;	16
					☐	%IWC2.0.3	D_MCURRENT_ESCLA	Motor Current;	16
					☒	%QWC2.0.0	D_COMMAND_ESCLAV	Drivecom command reg.	16
					☒	%QWC2.0.1	D_TARGET_ESCLAVO_	Target velocity;	16

Vaya a la ventana programar, programar y elabore un programa en el lenguaje que desee “lista o ladder”. Para este manual se eligió el lenguaje de programación llamado ladder. A continuación se presentan dos programas en ladder con el propósito de que el estudiante se familiarice con este tipo de programación.

4.2 PRIMER PROGRAMA: PROGRAMA LADDER SIN MACROS DRIVE QUE EJECUTA LA COMUNICACIÓN ENTRE UN PLC MAESTRO TWIDO Y UN ESCLAVO VARIADOR DE VELOCIDAD ALTIVAR 31.

Este programa es hecho para mostrar que en una red CANopen la programación sin macros drive es posible. El programa ejecuta intercambios de datos a través de PDOs y SDOs, también se habilita la opción de pasar del estado operacional al estado pre-operacional de la red CANopen y viceversa para que el usuario verifique que en estado pre-operacional solo es posible escribir o leer parámetros de configuración a través de SDOs, mientras que no es posible escribir o leer datos de proceso (entradas y salidas) a través de PDOs, en cambio en estado operacional es posibles leer y escribir SDOs y PDOs.

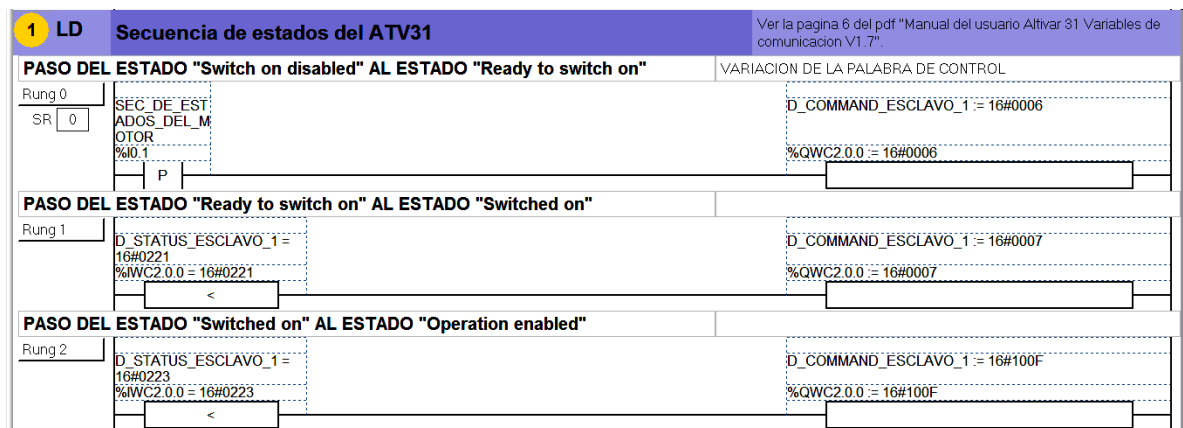
4.2.1 DESCRIPCION DEL PROGRAMA

4.2.1.1 PRIMERA PARTE: ESTADOS DEL ESCLAVO ATV 31: El esclavo variador de velocidad para su correcto funcionamiento debe pasar por una máquina de estados hasta llegar al estado “operation enabled” donde es posible controlarlo con un programa de comunicación. Para el mejor entendimiento se recomienda ver la página 6 del pdf "Manual del usuario Altivar 31 Variables de comunicacion V1.7", donde se encuentra los comandos para cada estado. En la figura 8 se presenta en lenguaje ladder la elaboración de los estados requeridos por el variador de velocidad Altivar para su correcto funcionamiento.

La figura 8 muestra los cambios de estados activados mediante los PDOs %IWC2.0.0 y %QWC2.0.0 que se observan en la figura 7 pertenecen a los PDOs básicos que trae por defecto el variador de velocidad Altivar 31. El PDO %IWC

2.0.0 representa el comando de un estado del esclavo mientras que el PDO %QWC2.0.0 representa el comando antes de entrar a un estado, de este modo cuando %QWC2.0.0 toma el valor 16#0006 el variador entra en parada (shutdown) o cuando el PDO %IWC 2.0.0 toma el valor 16#0221 el variador entra al estado listo para arrancar hasta llegar al estado operation enable pasando por el estado encendido (Switched on) y las transiciones, encender (Switch on) y operación permitida (Enable operation).

FIGURA 8: SECUENCIA DE ESTADOS DEL ATV31



4.2.1.2 SEGUNDA PARTE: Estados Pre-operacional y Operacional (NTM): La segunda parte del programa describe como las palabras de memorias %MW0 y %MW1 determinan el estado pre-operacional de la red mientras que las palabras de memorias %MW2 y %MW3 determina el estado operacional de la red CANopen. Ver figura 9

Las palabras de memoria descritas en la figura 9 llevan los parámetros de comando de la máquina de estados del protocolo CANopen que serán utilizados por la instrucción CAN_CMD para programar, gestionar la red CANopen y realizar

comprobaciones de diagnóstico en dispositivos esclavos individuales. La tabla presentada en la figura 10 describe que valores deben tomar las palabras de memorias para que lleven los parámetros de comando de la máquina de estado operacional y pre-operacional.

FIGURA 9: ESTADOS PRE-OPERACIONAL Y OPERACIONAL

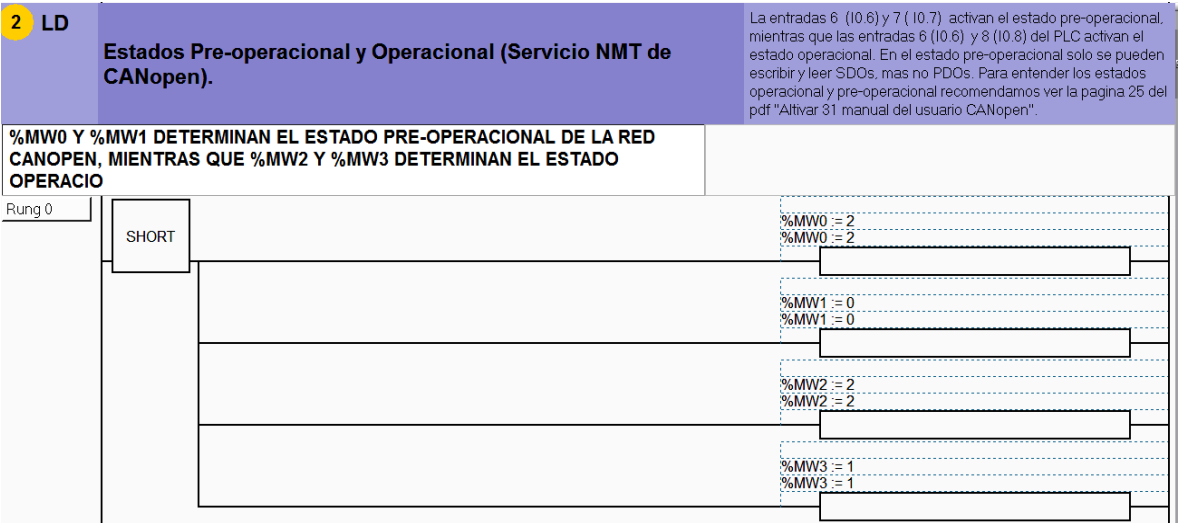


FIGURA 10: TABLA MÁQUINA DE ESTADO DE CANopen

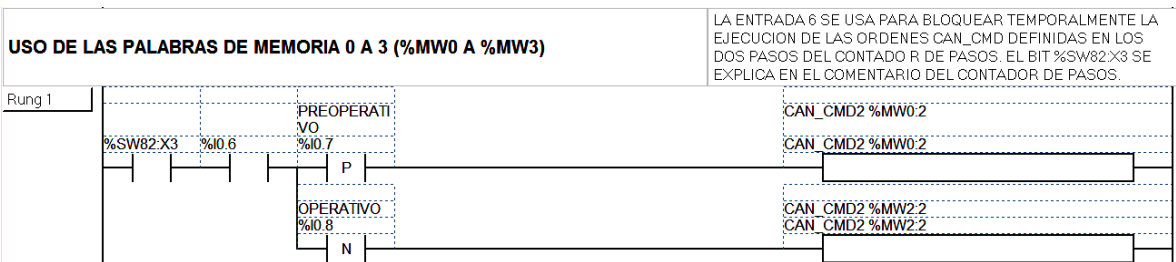
%MWx	%MWx+1		%MWx+2		%MWx+3		%MWx+4		%MWx+5		Acción
	Bit [15-8]	Bit [7-0]	Bit [15-8]	Bit [7-0]	Bit [15-8]	Bit [7-0]	Bit [15-8]	Bit [7-0]	Bit [15-8]	Bit [7-0]	
1	0		—								Restablecer la comunicación CANopen.
1	1										Restablecer los nodos CANopen.
2	0										Pasar del modo operativo al modo preoperativo.
2	1										Pasar al modo de funcionamiento.

Fuente: “twidosuite V2.0, Pg 308, modificado por los autores”

La tabla de la figura 10 describe que valores debe tomar las palabras de memoria %MWx y %MWx +1 para que lleven los parámetros de comando de la máquina de estados, ejemplo para el caso de la palabras de memorias utilizadas en este manual es necesario que tomen estos valores: %MW0= 2 y %MW1=0, para que el programa pase de estado operacional a pre-operacional. %MW2= 2 y %MW3=1 para que el programa pase de estado pre-operacional a operacional.

A continuación se utiliza la instrucción CAN_CMD. Ver figura 11.

FIGURA 11: INSTRUCCIÓN CAN_CMD



En la figura 11 se describe el uso de las palabras de memorias (%MW0 a %MW3) a través de la instrucción CAN_CMD. En este programa cuando se activan las entradas %I0.6 y %I0.7 el programa se pone en estado pre-operativo a través de la instrucción CAN_CMD2 %MW0:2. La descripción de la instrucción CAN_CMD2 %MW0:2 es la siguiente: el 2 que acompaña a la instrucción CAN_CMD es para indicar la dirección de ampliación del módulo master CANopen en el bus Twido. El 0 que acompaña a la palabra memoria %MW indica el número de la primera palabra de memoria transmitida, en este caso %MW0 y el 2 que esta después de los dos puntos indica la longitud de la instrucción en número de palabras, para este ejemplo dos palabras. En conclusión la instrucción CAN_CMD que se utiliza

para poner el sistema en pre-operativo utiliza dos palabras de memoria comenzando por la palabra de memoria %MW0 y la instrucción CAN_CMD utilizada para que el sistema entre en estado operacional utiliza las palabras de memoria %MW2 y %MW3 cuando la entradas %I0.6 y %I0.8 se activan. El bit %SW82 X3 se utiliza para ejecutar solo una instrucción CAN_CMD al tiempo Para completar el conocimiento de dicha red se recomienda leer la presentación de la instrucción CAN_CMD en la página 305 y 308 del pdf, “twidosuite V2.0 guía de programación”.

4.2.1.3 TERCERA PARTE : LECTURA Y ESCRITURA DE SDOs

ESCRITURA DE UN SDO: En esta parte del programa se escribirá en el esclavo variador de velocidad el SDO HSP que limita la velocidad máxima de dicho esclavo. Se recomienda ver la página 24 del PDF manual del usuario altivar 31 variables de comunicación V1.7.

la figura 12 describe la tabla de palabras de memorias utilizadas para la escritura del SDO=HSP. Se recomienda al usuario ver la tabla de las páginas 308 y 309 de twidosuite V2.0 guía de programación que indican cómo se deben elaborar las tablas de lectura y escritura.

En la figura 12 la palabra %MW8 lleva el valor del HSP. Para escribir el valor de este SDO en el esclavo se debe construir la tabla de escritura de la figura 12, que la compone las siguientes palabras:

%MW4= 16#0004, el valor de 4 en hexadecimal de la palabra de memoria %MW4 indica que va a iniciar la escritura del comando SDO.

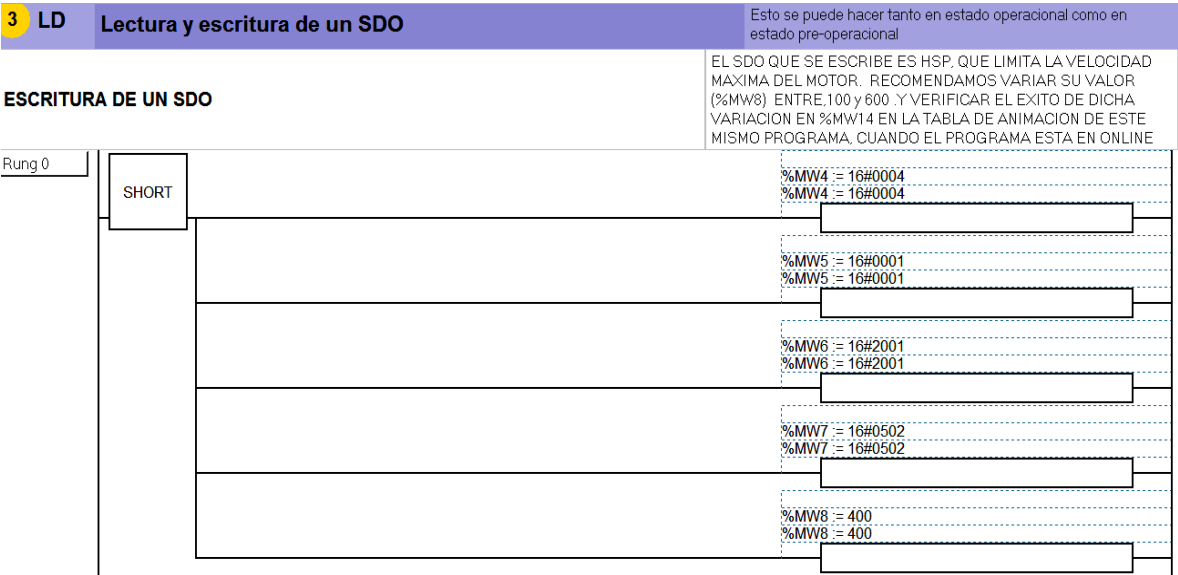
%MW5= 16#0001, el valor de esta palabra indica la dirección del esclavo en el que se va a escribir

%MW6= 16#2001, el valor de esta palabra de memoria indica la dirección del SDO que se va a escribir.

%MW7= 16#0502, el valor de la palabra de memoria %MW7 está compuesta por dos bytes, el byte más significativo indica el subíndice de la dirección del SDO, que tiene un valor de 05 para este ejemplo y el byte menos significativo indica la longitud de datos en bytes que se van a escribir, para este ejemplo se va a escribir una palabra que tiene 2 bytes.

%MW8= 400, el valor de esta palabra indica el valor del SDO que se va a escribir en el esclavo

FIGURA 12: ESCRITURA DE UN SDO



LECTURA DE UN SDO: En esta parte del programa se leerá del esclavo variador de velocidad altivar 31 el SDO HSP que limita la velocidad máxima de dicho esclavo. En la figura 13 se programa la tabla de palabras de memorias utilizadas para la lectura del SDO=HSP.

FIGURA 13: LECTURA DE UN SDO

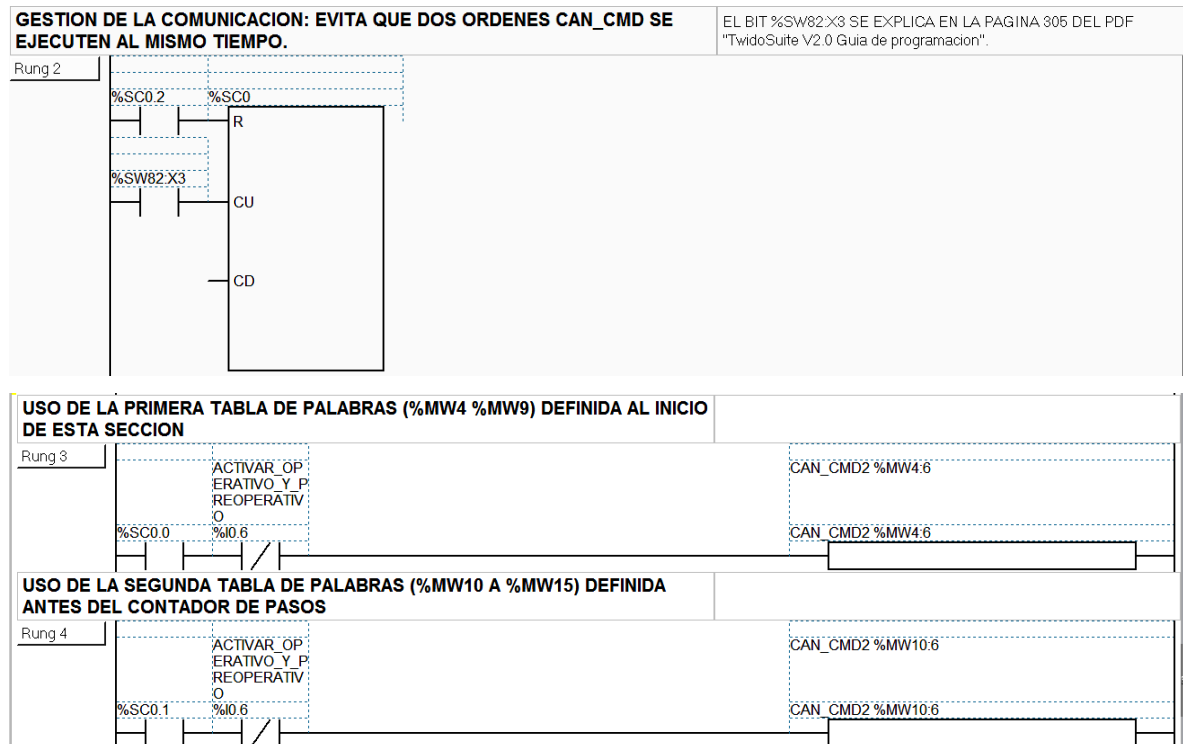


La figura 13 muestra la programación de la tabla de palabras para la lectura de un SDO, La programación de la tabla de palabras está compuesta por cinco palabras de memorias consecutivas desde la palabra de memoria %MW10 hasta la palabra de memoria %MW14, esta última palabra se reserva para que el esclavo responda el valor del SDO en ella. El valor de la palabra de memoria %MW14 se pueda ver en la tabla de animación cuando el programa está en modo online. Las otras cuatro palabras de memorias tienen el mismo significado que las cuatro primeras palabras de memoria para la escritura de un SDO, excepto la primera palabra de memoria (%MW10) que inicia la lectura del comando SDO

GESTIÓN DE LA COMUNICACIÓN: la comunicación se gestiona a través de la instrucción CAN_CMD y el bit %SW82:X3, este bit sirve para ejecutar solo una instrucción CAN_CMD al tiempo, dicho bit vale 0 cuando una orden CAN_CMD se está ejecutando y vale 1 cuando dicha orden se ha terminado de ejecutar. La instrucción CAN_CMD cumple la misma función que la instrucción EXCHx usada en los programas para los protocolos modbus y modbus TCP/IP, es decir sirve para ejecutar la lectura o escritura de datos. La figura 14 presenta en el programa

ladder la ejecución y el control de la instrucción CAN_CMD empleando el bit %SW82: X3.

FIGURA 14: GESTIÓN DE LA COMUNICACIÓN



En la figura 14 el contador de pasos se Resetea cuando llega al paso %SC0.2 después de contar los dos pasos %SC0.0 y %SC0.1. El bit %SW82: X3 garantiza que toda una instrucción CAN_CMD termine en un paso antes de comenzar otra instrucción CAN_CMD.

El Rung 3 de la figura 14 presenta la instrucción CAN_CMD que sirve para gestionar el intercambio de mensajes SDOS que se transmiten por medio de palabras de memoria. Esta instrucción gestiona la escritura del SDO= HSP

utilizando las palabras de memorias %MW4 al %MW9 por lo que se emplea la siguiente sintaxis: CAN_CMD2 %MW4:6

El Rung 4 de la figura 14 presenta la gestión de la lectura del SDO=HSP utilizando la instrucción CAN_CMD. Esta instrucción gestiona la lectura del SDO utiliza las palabras de memorias %MW10 al %MW15, por lo que se emplea la siguiente sintaxis: CAN_CMD2 %MW10:6

4.2.1.4 CUARTA PARTE: LECTURA Y ESCRITURA DE PDOS

En esta parte del programa se escribe y se leen dos PDOS: %QWC2.0.1 (D_TARGET_ESCLAVO_1), para el control de la velocidad y %QWC2.0.0 (D_COMMAND_ESCLAVO_1) para el control del sentido de giro y parada del motor. Ver figura 15 y figura 16. Se recomienda ver pdf, Altivar 31 Communication variables, V1.7, página 12

FIGURA 15: ESCRITURA Y LECTURA DEL PDO % QWC2.0.0

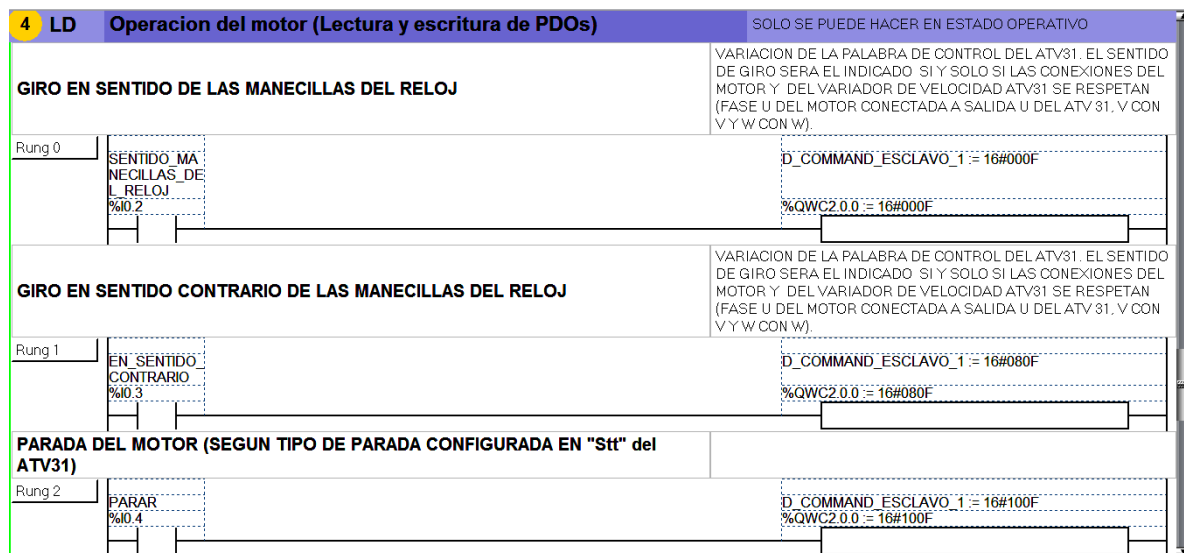
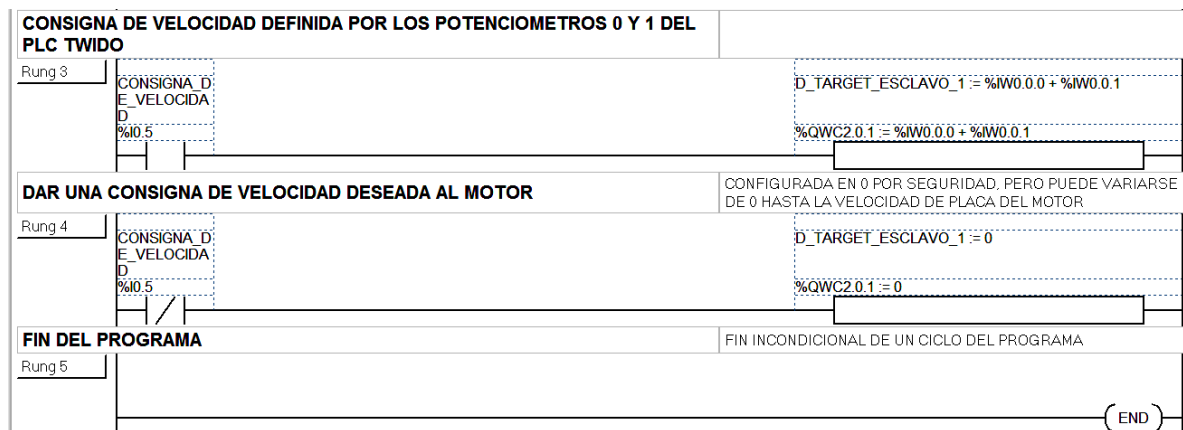


FIGURA 16: ESCRITURA Y LECTURA DEL PDO %QWC2.0.1



La figura 15 muestra el control de giro y parada del motor a través del PDO %QWC2.0.0 (D_COMMAND_ESCLAVO_1), este PDO cuando toma el valor de 16#000F el motor gira en sentido de las manecillas del reloj y cuando toma el valor 16#080F el motor gira en sentido contrario de las manecillas del reloj. Para que el motor se detenga el PDO debe tomar el valor de 16#100F. Este PDO es llamado palabra de control y se describe en “Altivar 31 Communication variables, V1.7, página 6”. La figura 16 muestra como se controla la velocidad del motor a través del PDO %QWC2.0.1 (D_TARGET_ESCLAVO_1). La consigna de velocidad está definida por los potenciómetros 0 (%IW0.0.0) y 1 (%IW0.0.1) del PLC Twido y su valor está dada por la suma de estos dos cuando $\%QWC2.0.1 = \%IW0.0.0 + \%IW0.0.1$.

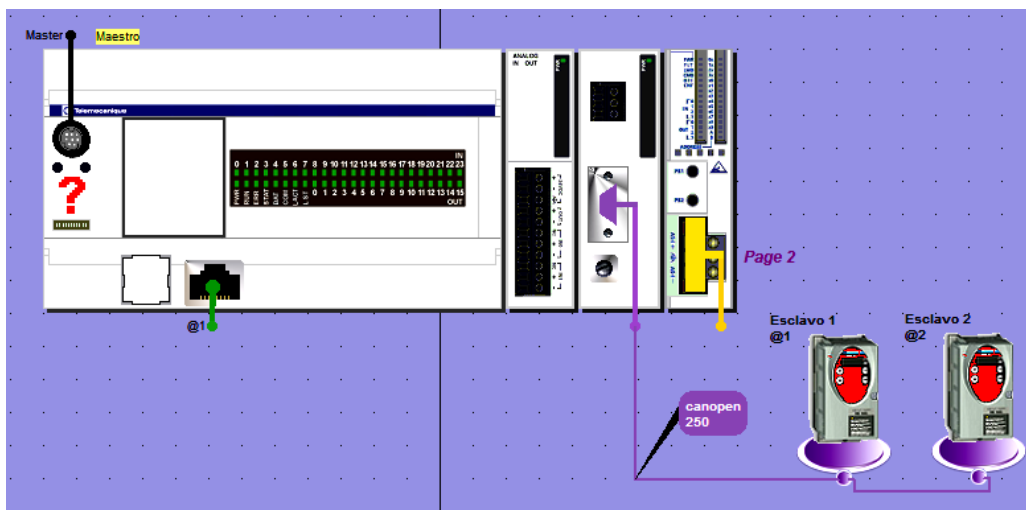
4.3 SEGUNDO PROGRAMA. PROGRAMA LADDER CON MACROS DRIVE QUE EJECUTA LA COMUNICACIÓN ENTRE UN PLC MAESTRO TWIDO Y UN ESCLAVO VARIADOR DE VELOCIDAD ALTIVAR 31: .este programa maneja 2 motores usando macros drive y el protocolo CANopen. El motor 1 puede girar a la derecha, girar a la izquierda, parar, tomar una velocidad baja (300 rpm), media

(1000 rpm) o alta (1700 rpm) y copiar la velocidad y/o el sentido de giro del motor 2. El motor 2 también puede girar en ambos sentidos, parar y tomar una de 15 velocidades posibles (100, 200, 300,..., 1500 rpm); además el motor 2 tiene la posibilidad de copiar la velocidad y/o el sentido de giro del motor 1

4.3.1 RED CANopen

Esta red está constituida por dos variadores de velocidad Altivar 31 y un PLC Twido, como se presenta en la figura 17

FIGURA 17: RED CANOPEN



Los elementos que componen la red CANopen (esclavos Altivar 31, la red) deben ser configurados como lo indican las figuras 18, 19 y 20.

Cuando ya está configurada la red se pueden ver en la ventana (programar, configurar) los PDOs que trae por defecto el variador de velocidad Basic_Altivar 31

(V1.1) y que van a ser utilizados para la elaboración del programa en lenguaje ladder. Ver figuras 21 y 22.

FIGURA 18: PARÁMETROS DE LA RED CANopen

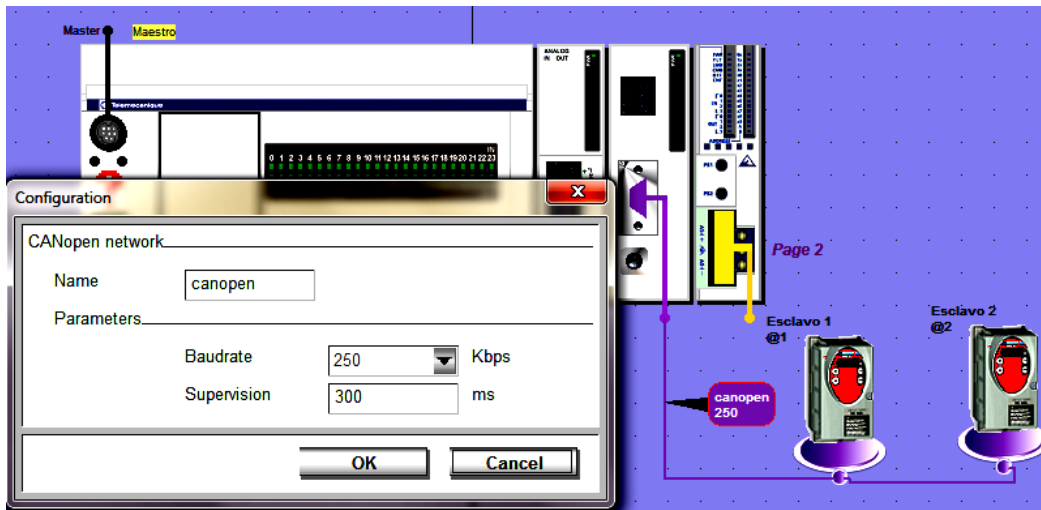


FIGURA 19: CONFIGURACIÓN DEL ESCLAVO 1

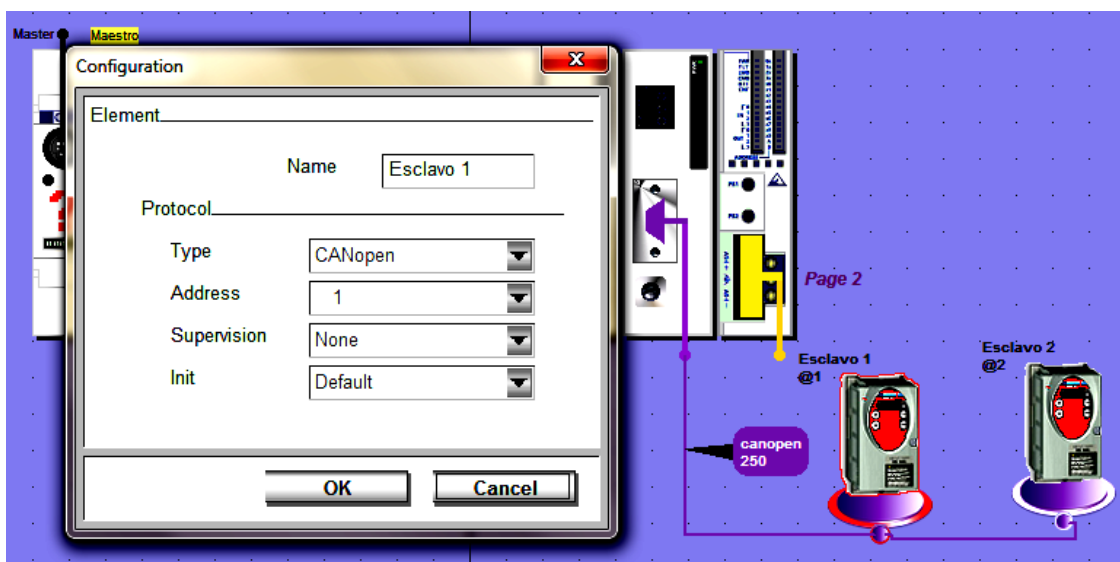


FIGURA 20: CONFIGURACIÓN DEL ESCLAVO 2

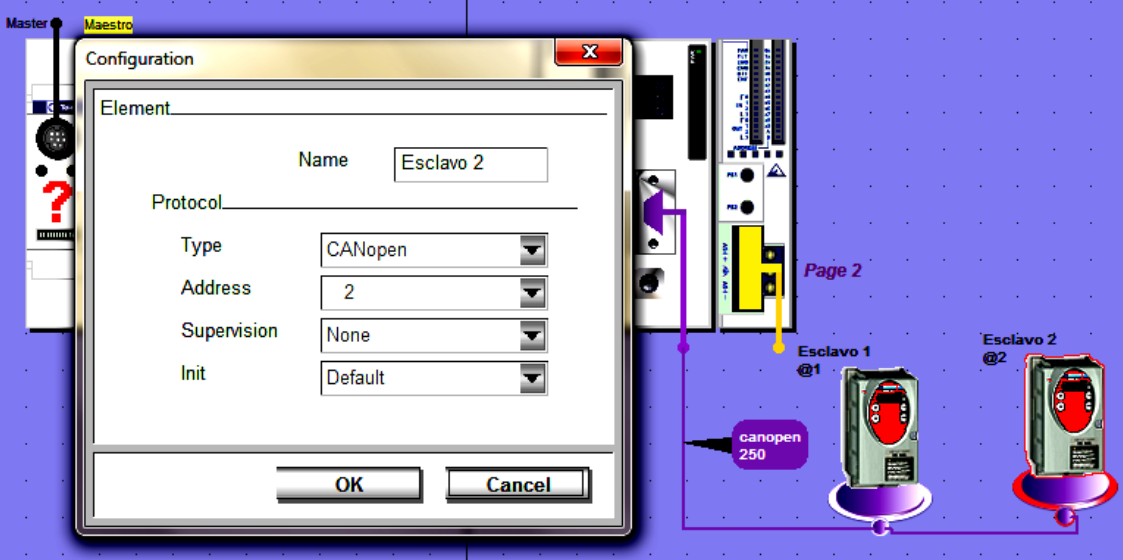


FIGURA 21: PDOs DEL ESCLAVO 1

Project Describe Program Report

247 Maestro

Description of the module Reference number TWDNCO1M Address 2

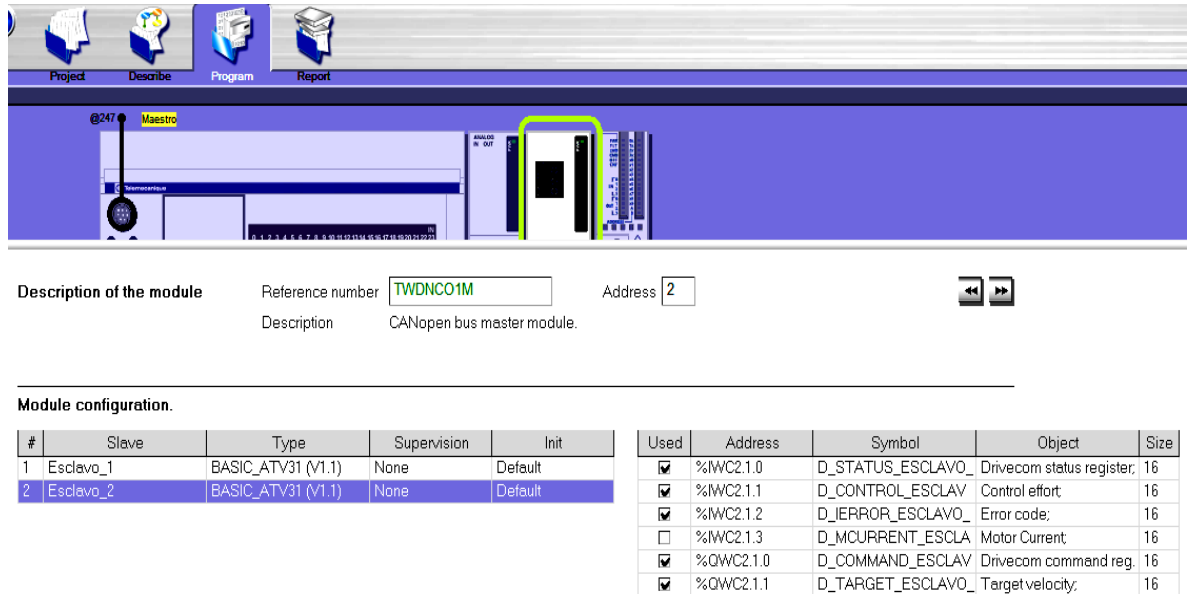
Description CANopen bus master module.

Module configuration.

#	Slave	Type	Supervision	Init
1	Esclavo_1	BASIC_ATV31 (V1.1)	None	Default
2	Esclavo_2	BASIC_ATV31 (V1.1)	None	Default

Used	Address	Symbol	Object	Size
☒	%IWC2.0.0	D_STATUS_ESCLAVO_	Drivecom status register;	16
☒	%IWC2.0.1	D_CONTROL_ESCLAV	Control effort	16
☒	%IWC2.0.2	D_IERROR_ESCLAVO_	Error code;	16
☐	%IWC2.0.3	D_MCURRENT_ESCLA	Motor Current	16
☒	%QWC2.0.0	D_COMMAND_ESCLAV	Drivecom command reg.	16
☒	%QWC2.0.1	D_TARGET_ESCLAVO_	Target velocity;	16

FIGURA 22: PDOs DEL ESCLAVO 2



4.3.2 CONFIGURACIÓN DE LAS MACRODRIVE:

Una vez configurado todos los parámetros de red, es posible configurar los Macros Drive que serán utilizados en la elaboración del programa. Los Macros Drive se configuran de forma sencilla y simplifica el programa de control, por este motivo el programador no tiene que disponer de conocimientos avanzados del protocolo de comunicación CANopen debido a que la gestión y la creación de la tramas de comunicación se realizan internamente en los Macros.

Para configurar los Macros Drive se va a la ventana “programar, configurar” para elegir (en la parte lateral derecha) la opción de “Configurar los datos”. Cuando se hace estos pasos aparecerá una ventana con las diferentes posibilidades de configuración de datos que ofrece el Twido. Los Macros Drive se encuentran dentro de “Objetos avanzados” en la parte lateral izquierda. Ver figura 23

FIGURA 23: VENTANA DONDE SE ENCUENTRA LOS MACROS DRIVE

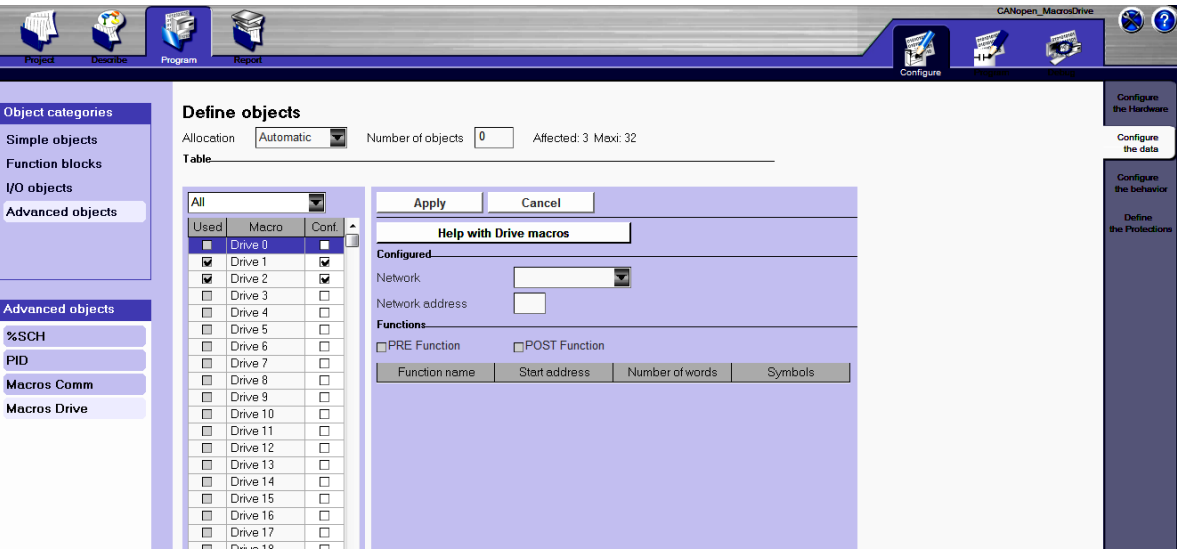
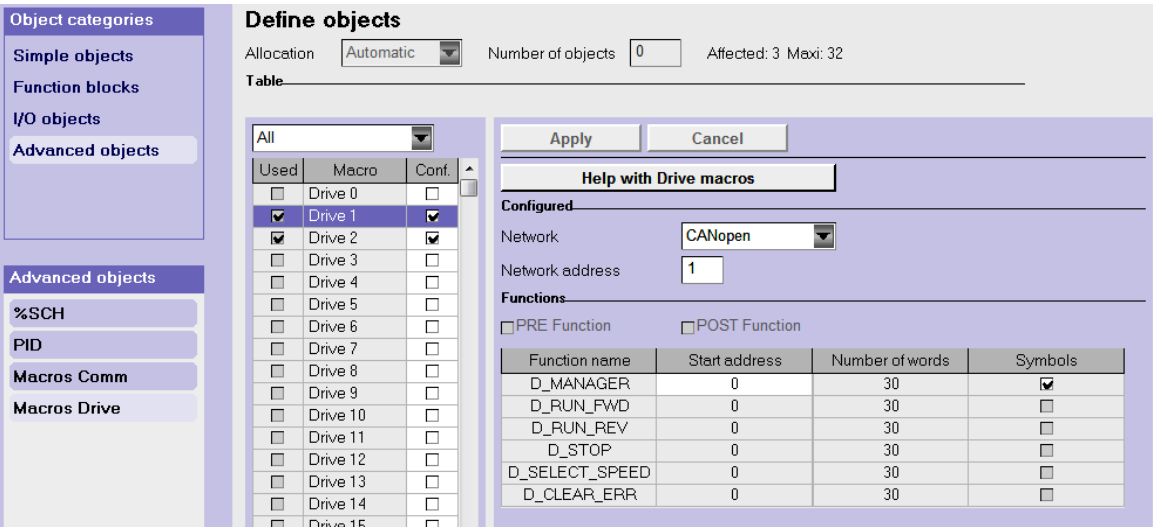


FIGURA 24: MACROS DRIVE PARA EL ESCLAVO 1



En la lista de 32 Macros (“Drive 0” hasta “Drive 31”) que presenta la figura 23 se elije el número de macros que se van a configurar, los Macros seleccionados se

activan marcando el cuadro “Conf” de la misma lista y se configuran seleccionando el tipo de red de comunicación; CANopen para este manual. Después de haber seleccionado el tipo de Red, en la opción “**Dirección de Red**” se tiene que asignar la dirección del esclavo al que vamos a controlar con la Macro Drive. En este manual se va a controlar dos esclavos Twido con direcciones 1y2. Ver figuras 24 y 25.

FIGURA 25: MACROS DRIVE PARA EL ESCLAVO 2

Definir los objetos

Asignación: Automático Número de objetos 0 Asignadas: 3 Máx: 32

Tabla

Uso	Macro	Conf.
☐	Drive 0	☐
☒	Drive 1	☒
☒	Drive 2	☒
☐	Drive 3	☐
☐	Drive 4	☐
☐	Drive 5	☐
☐	Drive 6	☐
☐	Drive 7	☐
☐	Drive 8	☐
☐	Drive 9	☐
☐	Drive 10	☐
☐	Drive 11	☐
☐	Drive 12	☐
☐	Drive 13	☐
☐	Drive 14	☐

Aplicar Cancelar

Ayuda sobre las macros Drive

General

Red: CANopen

Dirección en la red: 2

Funciones

☐ Función PRE ☐ Función POST

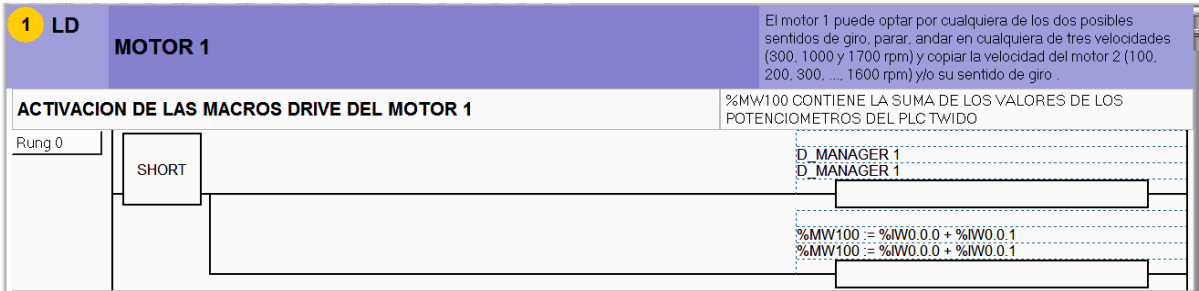
Nombre de función	Dirección de inicio	Número de palabras	Símbolos
D_MANAGER	30	30	☒
D_RUN_FWD	30	30	☐
D_RUN_REV	30	30	☐
D_STOP	30	30	☐
D_SELECT_SPEED	30	30	☐
D_CLEAR_ERR	30	30	☐

La figura 24 y figura 25 muestra cómo se activa las Macro Drive cuando se marca el cuadro en symbols de D_MANAGER. A continuación se muestra el programa en ladder presentado al inicio de esta sección

4.3.3 PRIMERA PARTE: ACCIONES QUE REALIZA EL MOTOR # 1: En esta primera parte el BASIC_ATV31 (esclavo 1) que controla el motor #1 se controla con la Macro Drive 1 a través de sus funciones: D_RUN_FWD, D_RUN_REV, D_STOP, D_SELECT_SPEED y D_CLEAR_ERR. El motor 1 puede realizar las

siguientes acciones: girar en cualquiera de los dos posibles sentidos de giro, parar, trabajar, alcanzar y mantener alguna de estas tres velocidades (300, 1000 y 1700 rpm) y copiar cualquier velocidad que tenga el motor 2 y/o su sentido de giro. La figura 26 muestra la activación de la macros drive # 1 y la suma de los valores de los potenciómetros del PLC Twido utilizando la palabra de memoria %MW100.

FIGURA 26: ACTIVACIÓN DE LAS MACROS DRIVE DEL MOTOR 1



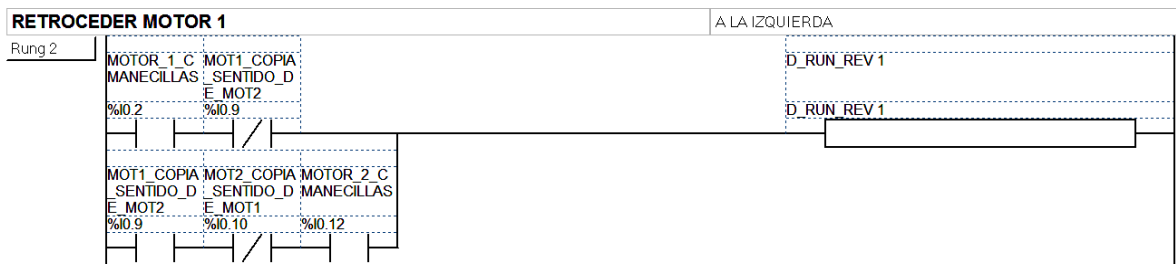
La figura 27 muestra como se debe utilizar la función D_RUN_FWD 1 del Macro Drive # 1 para que el motor # 1 gire en sentido de las manecillas del reloj y para que el motor # 1 copie el giro del motor 2 cuando este gira en sentido de las manecillas del reloj.

FIGURA 27: FUNCIÓN D_RUN_FWD 1



La figura 28 muestra el uso de la función D_RUN_REV del Macro Drive # 1 utilizada para que el motor 1 gire en sentido contrario de las manecillas del reloj y para que el motor 1 copie el giro del motor 2 cuando este gira en sentido contrario de las manecillas de reloj.

FIGURA 28: FUNCIÓN D_RUN_REV 1



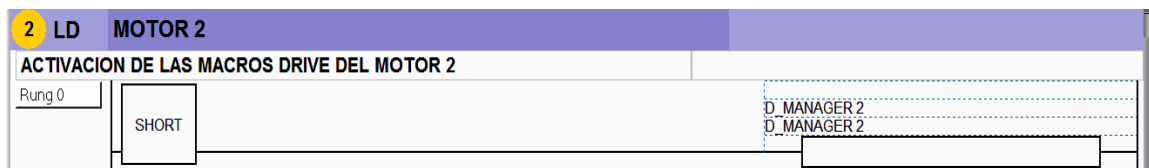
La figura 29 muestra el uso de la función D_STOP 1 utilizada para que el motor 1 pare y para que pare ambos motores

FIGURA 29: FUNCIÓN D_STOP 1



4.3.4 SEGUNDA PARTE: ACCIONES QUE REALIZA EL MOTOR # 2: en esta segunda parte el BASIC_ATV31 (esclavo 2) que controla el motor #2 se controla con la Macro Drive 2 a través de sus funciones: D_RUN_FWD 2, D_RUN_REV 2, D_STOP 2, D_SELECT_SPEED_2 y D_CLEAR_ERR 2. El motor 2 puede realizar las siguientes acciones: girar en cualquiera de los dos posibles sentidos de giro, parar, tomar una de entre 15 opciones de velocidades (100, 200, 300,..., 1500 rpm) y copiar la velocidad del motor 1 (300, 1000 o 1700 rpm) y/o su sentido de giro. La figura 31 muestra la activación de la macros drive # 2

FIGURA 31 ACTIVACIÓN DE LA MACRO DRIVE DEL MOTOR 2



La figura 32 muestra como se debe utilizar la función D_RUN_FWD 2 para que el motor 2 gire en sentido de las manecillas del reloj y para que el motor 2 copie el giro del motor 1 cuando este gira en sentido de las manecillas del reloj

FIGURA 32: FUNCIÓN D_RUN_FWD 2



La figura 33 muestra el uso de la función D_RUN_REV 2 utilizada para que el motor 2 gire en sentido contrario de las manecillas del reloj y para que el motor 2 copie el giro del motor 1 cuando este gira en sentido contrario a las manecillas del reloj.

FIGURA 33: FUNCIÓN D_RUN_REV 2



La figura 34 muestra el uso de la función D_STOP 2 utilizada para que el motor 2 pare y para que los dos motores paren

FIGURA 34: FUNCIÓN D_STOP 2



La siguiente parte del programa utiliza la palabra de memoria %MW100 para comparar y aproximar el valor de velocidad a un número entero para después guardar este valor en la palabra de memoria %MW200 e indicar esta velocidad en alguna de las salidas %Q0.x, la entrada I0.8 normalmente abierta cuando no se acciona desactiva la salidas %Q0.x para indicar que el motor 2 a copiado la velocidad del motor 1. Ejemplo: cuando la velocidad está entre 500RPM y 600RPM, el valor guardado en la palabra de memoria %MW200 es 600 y este se indica activando la salida %Q0.6 del PLC Maestro Twido. Las figuras 35, 36,37, 38 y 39 muestran como se elabora esta parte del programa.

FIGURA 35: VELOCIDADES 1,2 Y 3 DEL MOTOR 2

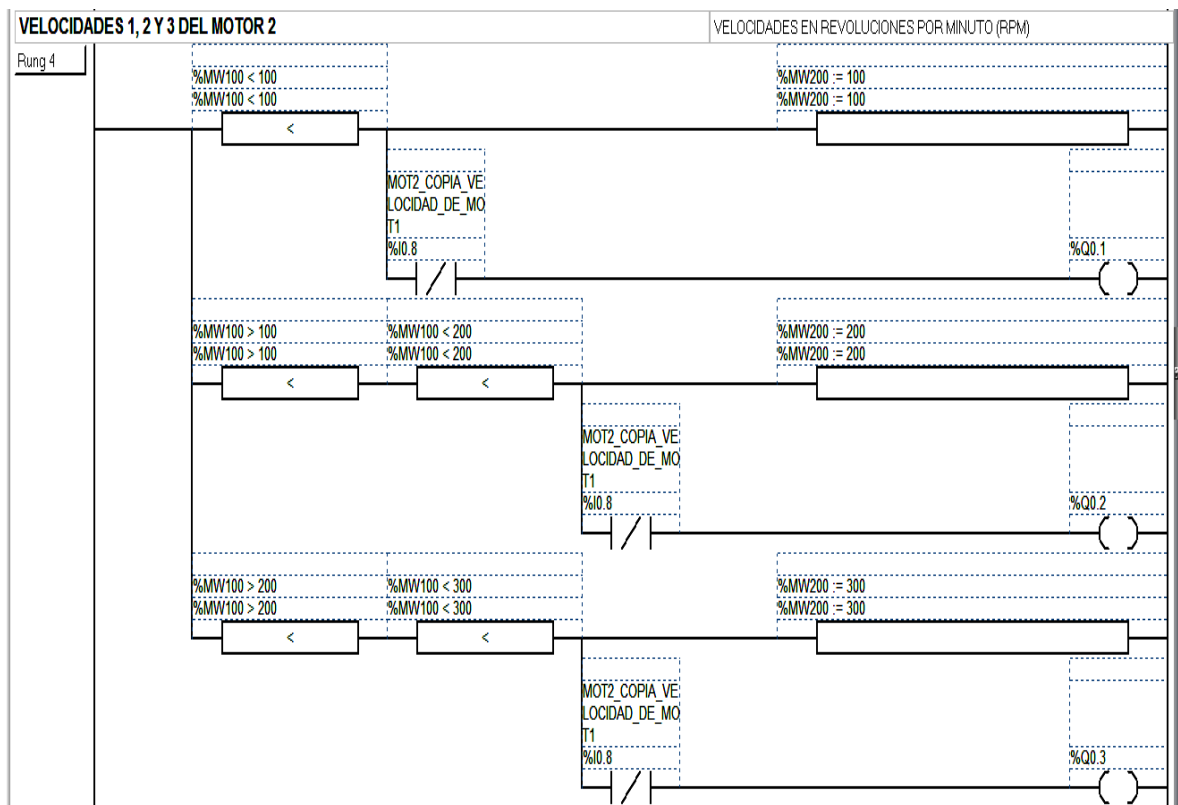


FIGURA 36: VELOCIDADES 4,5 Y 6 DEL MOTOR 2

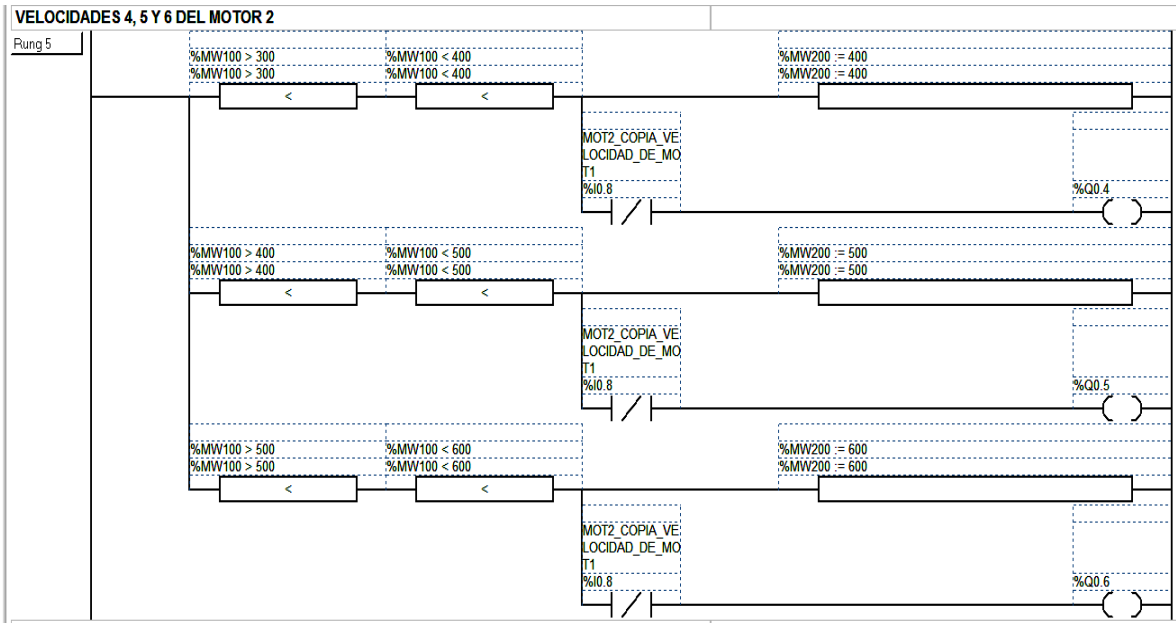


FIGURA 37: VELOCIDADES 7,8 Y 9

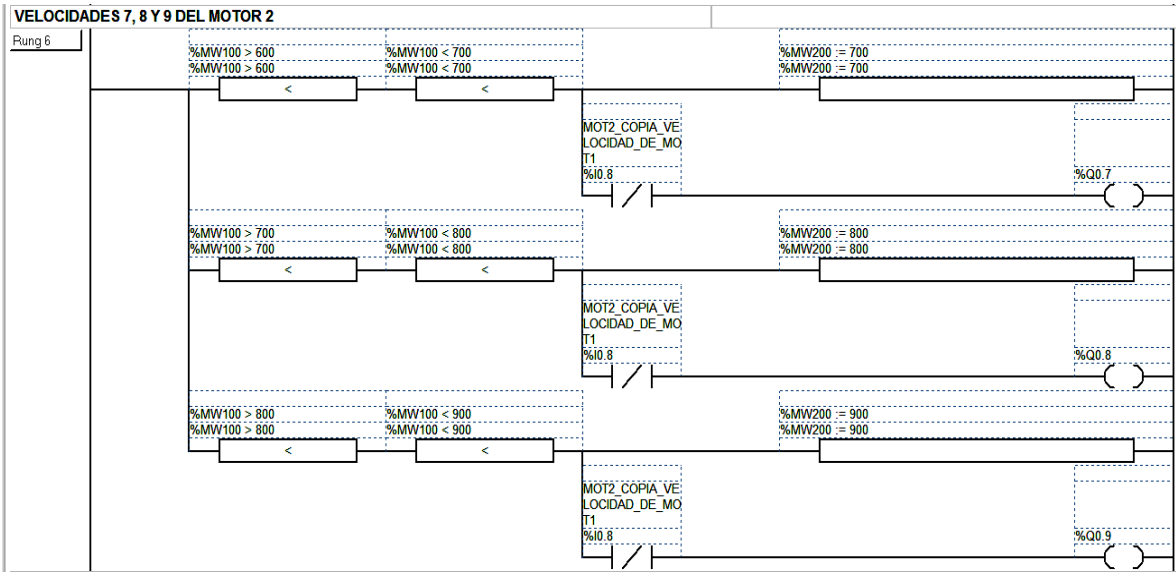


FIGURA 38: VELOCIDADES 10,11 Y 12 DEL MOTOR 2

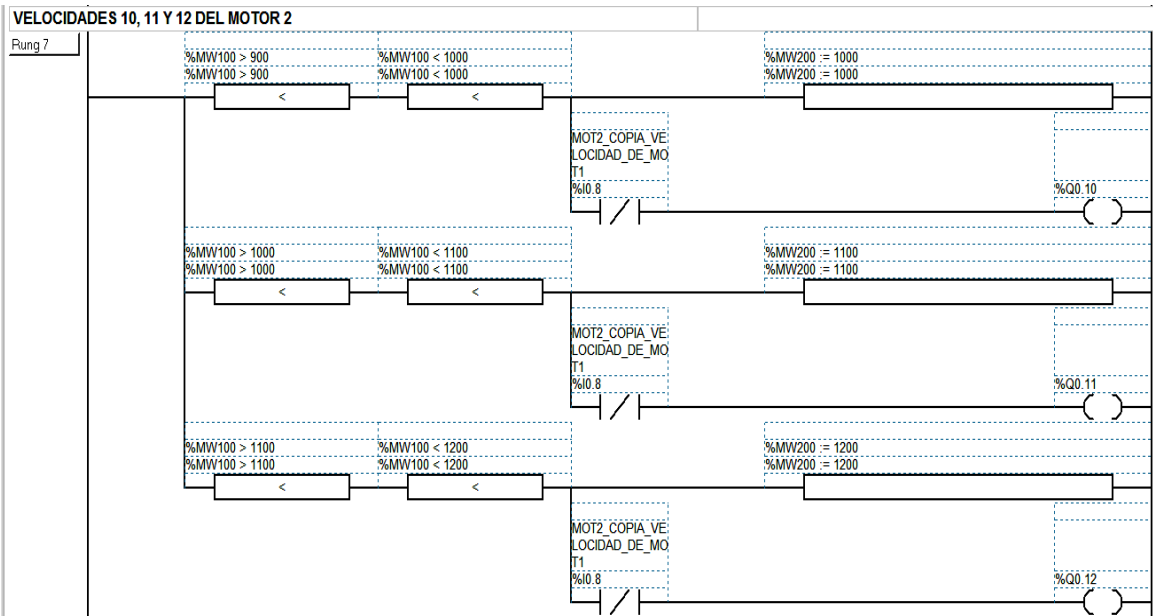
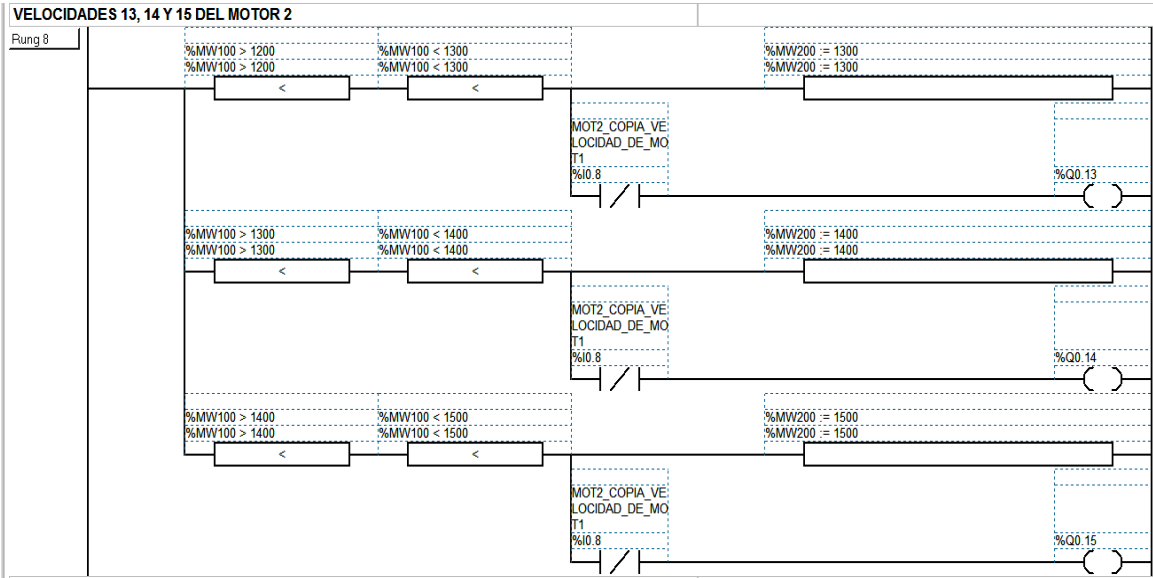
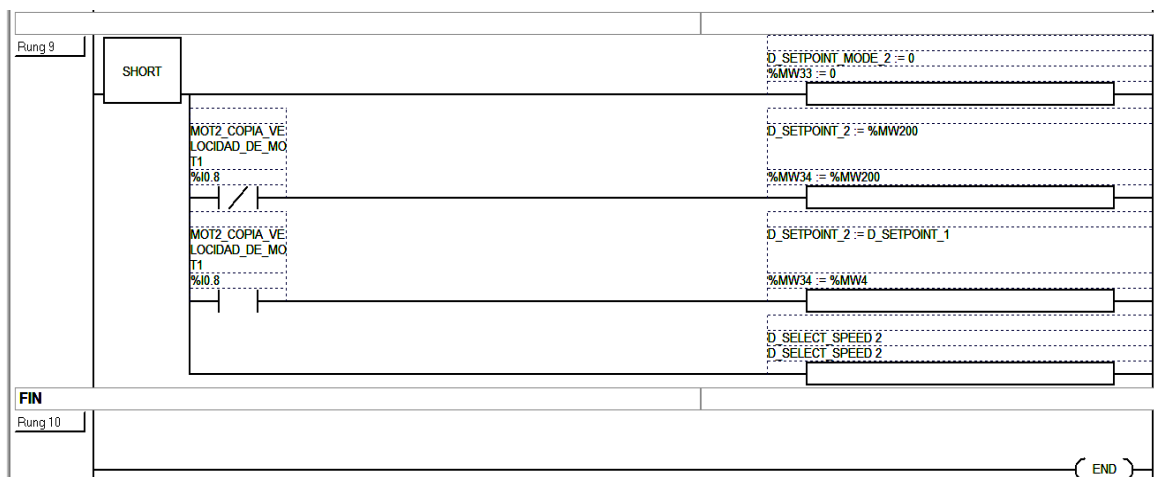


FIGURA 39: VELOCIDADES 13,14 Y 15 DEL MOTOR 2



En la última parte del este programa se utilizan la función D_SELECT_SPEED 2 con sus símbolos: D_SETPOINT_ MODE_2 que utiliza la palabra de memoria %MW33 y sirve para indicar al altivar 31 “motor 1” que funcione en modo de consigna. También se utiliza el símbolo D_SETPOINT_2, que se debe introducir con el valor correspondiente a la velocidad tras haber activado el modo de consigna, para este caso se utiliza la palabra de memoria %MW34 para introducir las velocidades al motor 2 de 100 RPM, 200 RPM, 300 RPM hasta 1500 RPM. Cuando D_SETPOINT_2 es igual a D_SETPOINT_1 el motor 2 copia la velocidad del motor 1. Por último la macro introduce el símbolo D_SELECT_SPEED_2 con el valor de bit leído. Ver figura 40

FIGURA 40



4.4 TABLAS DE ANIMACIÓN: las tablas de animación presentan las variables utilizadas en el programa y el estado actual de estas variables, por tal motivo es posible supervisar el funcionamiento del programa. Las variables se pueden forzar (F 1) desde la tabla de animación o sea se puede activar las variables desde la tabla de animación

FIGURA 41: TABLA DE ANIMACIÓN PRIMER PROGRAMA

	Us	Dirección	Símbolo	Actual	Guardado	Formato
1	☐	%SW20		2	0	Decimal
2	☒	%SW82		0000000000000011	0000000000000000	Binario
3	☒	%QWC2.0.0	D_COMMAND_E	080F	0000	Hexadecimal
4	☒	%QWC2.0.1	D_TARGET_ESCL	324	0	Decimal
5	☒	%IWC2.0.0	D_STATUS_ESCL	8627	0000	Hexadecimal
6	☒	%MW0		2	0	Decimal
7	☒	%MW1		0	0	Decimal
8	☒	%MW2		2	0	Decimal
9	☒	%MW3		1	0	Decimal
10	☒	%MW14		400	0	Decimal
11	☐	%I0.0	RUN_STOP_DEL	F 1	0	Decimal
12	☒	%I0.1	SEC_DE_ESTAD	F 1	0	Decimal
13	☒	%I0.2	SENTIDO_MANE	0	0	Decimal
14	☒	%I0.3	EN_SENTIDO_CO	F 1	0	Decimal
15	☒	%I0.4	PARAR	0	0	Decimal
16	☒	%I0.5	CONSIGNA_DE	F 1	0	Decimal
17	☒	%I0.6	ACTIVAR_OPERA	0	0	Decimal
18	☒	%I0.7	PREOPERATIVO	0	0	Decimal
19						

FIGURA 42: TABLA DE ANIMACIÓN SEGUNDO PROGRAMA

	Us	Dirección	Símbolo	Actual	Guardado	Formato
1	☒	%SW82		00000000000001011	0000000000000000	Binario
2	☐	%SW20		0000001000000010	0000000000000000	Binario
3	☒	%MW3	D_SETPOINT_MO	0	0	Decimal
4	☒	%MW4	D_SETPOINT_1	300	0	Decimal
5	☒	%MW33	D_SETPOINT_MO	0	0	Decimal
6	☒	%MW34	D_SETPOINT_2	400	0	Decimal
7	☒	%MW2	D_ERROR_1	3310	0000	Hexadecimal
8	☒	%MW32	D_ERROR_2	0000	0000	Hexadecimal
9	☒	%IWC2.0.2	D_IERROR_ESCL	3310	0000	Hexadecimal
10	☒	%IWC2.1.2	D_IERROR_ESCL	0000	0000	Hexadecimal
11	☒	%IWC2.0.0	D_STATUS_ESCL	0000011000100111	0000000000000000	Binario
12	☒	%IWC2.1.0	D_STATUS_ESCL	1000011000100111	0000000000000000	Binario
13	☐	%I0.0	RUN_STOP_DEL	F 1	0	Decimal
14	☒	%I0.1	MOTOR_1_MANE	F 1	0	Decimal
15	☒	%I0.2	MOTOR_1_CMAR	0	0	Decimal
16	☒	%I0.3	PARAR_EL_MOT	0	0	Decimal
17	☒	%I0.4	MOTOR_1_A_300	F 1	0	Decimal
18	☒	%I0.5	MOTOR_1_A_100	0	0	Decimal
19	☒	%I0.6	MOTOR_1_A_170	0	0	Decimal
20	☒	%I0.7	MOT1_COPIA_VE	0	0	Decimal
21	☒	%I0.8	MOT2_COPIA_VE	0	0	Decimal
22	☒	%I0.9	MOT1_COPIA_SE	0	0	Decimal
23	☒	%I0.10	MOT2_COPIA_SE	0	0	Decimal
24	☒	%I0.11	MOTOR_2_MANE	0	0	Decimal
25	☒	%I0.12	MOTOR_2_CMAR	F 1	0	Decimal
26	☒	%I0.13	PARAR_EL_MOT	0	0	Decimal
27	☒	%I0.14	PARAR_LOS_2	0	0	Decimal
28	☒	%MW100	PARAR_LOS_2 MOTORES	325	0	Decimal

FIGURA 43: TABLA CONFIGURACIÓN DEL ALTIVAR 31

Menú	Cod	Pg en Program	Pg en Variables	Valor	Obligatorio	Nota	Convención
drC-	bFr	23	26	60	Si		*drC- = Menu de control del motor
drC-	UnS	23	26	220	Si		*I-O- = Menu de entradas/s alidas
drC-	FrS	23	26	60	No		*Ctl- = Menu de control
drC-	nCr	23	26	6,9	Si		*Fun- = Menu de funciones de aplicaciones
drC-	nSP	23	27	1700	Si		*CON- = Menu de comunicación
drC-	CCS	23	27	0,77	Si		*SUP- = Menu de supervision
I-O-	tCC	27	30	2C	No		
Ctl-	LAC	38	32	L3	Si		
Ctl-	Fr1	38	32	Ndb	Si		
Ctl-	Fr2	38	32	UpdH	No		
Ctl-	rFC	39	33	C113	Si		
Ctl-	CHCF	39	33	SEP	Si		
Ctl-	Cd1	39	33	Ndb	Si		
Ctl-	Cd2	39	33	tEr	No		
Ctl-	CCS	40	34	C114	Si		
Fun-	rPt	42	35	LIn	No		
Fun-	Inr	43	35	0.1	No		
Fun-	ACC	43	35	0.5	No		
Fun-	dEC	43	35	0.5	No		
Fun-	Stt	45	36	Fst	Si		
Fun-	dCF	45	37	0	No		
Fun-	tdC	46	37	0.5	No		
Fun-	tdC1	47	38	0.5	No		
CON-	Add	70	51	1	Si		
CON-	tbr	70	51	19200	Si		
CON-	tF0	70	51	8n2	Si		
CON-	tt0	70	51	1	Si		
CON-	AdC0	70	51	0	Si		
CON-	bdC0	70	51	250	No		
CON-	ErC0	70	21	0	No	Solo lectura	
SUP-	LIt	72	20		No	Solo lectura	
	LFRD		12				
	RFRD		13			Solo lectura	
	CMD		15				
	Lfr		15				
	CMI		16				
	ETA		17			Solo lectura	
	rFr		17			Solo lectura	

Finalmente se presenta una tabla para facilitarle al usuario la programación manual o a través de los protocolos modbus y CANopen. La tabla resume los documentos “Altivar 31, User's manual, communication variables V1.7”y “Altivar 31 Guía de programación, variadores de velocidad para motores asíncronos V1.7” para comprender la tabla se debe entender lo siguiente:

Menú: es el menú del altivar 31

C0d.: son los tipos de códigos que maneja el Altivar 31

Pg en program: es la página que corresponde al documento Altivar 31 guía de programación, variadores de velocidad para motores asíncronos V1.7

Pg en variables: es la página que corresponde al documento Altivar 31, User's manual, communication variables V1.7

Valor: corresponde al valor de los parámetros del motor de la E3T que se asigna al Altivar 31

Obligatorio: cuando es **si** el altivar 31 se tiene que configurar con los parámetros presentes en la tabla. **No** es opcional. Ver figura 43

ANEXO 5: PRACTICAS DE LABORATORIO

CONTENIDO

	Pg
PRACTICA 1: AS-INTERFACE (AS-i)	454
PRACTICA 2: MODBUS (SOBRE LINEA SERIAL)	460
PRACTICA 3: CANopen	466
PRACTICA 4: MODBUS TCP/IP	474
PRACTICA 5: PLANTA EMBOTELLADORA	481

LISTA DE GRAFICAS

	Pg
Figura 1: sistema envasado de producto simplificado	459
Figura 2: sistema taponado de botellas simplificado	478
Figura 3: sistema envasado de producto interconectado con sistema taponado de botellas	480
Figura 4 (panel 1): introducción (resumen del programa)	485
Figura 5 (panel 5): Información del trabajo de grado, su director y autores	486
Figura 6 (panel 2): sistema llenado de tanques real	487
Figura 7 (panel 6): Lista de variables del panel 2	488
Figura 8 (panel 3): sistema envasado de producto real	490
Figura 9 (panel 7): Lista de variables del panel 3	491
Figura 10 (panel 4): sistema taponado de botellas real	494
Figura 11 (panel 8): lista de variables del panel 4	495

PRACTICA 1: AS-INTERFACE (AS-i)

REQUISITO:

Para facilitar el desarrollo de esta práctica se recomienda leer el capítulo 2 y el anexo 1 (anexo AS-i) del proyecto de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”, o por lo menos el anexo 1.

EQUIPO NECESARIO

1. Un PLC Twido referencia TWDLCAE40DRF, marca Telemecanique.
2. Un cable de programación (USB a MiniDin de 8 pines) marca Telemecanique, para el PLC Twido.
3. Una regleta de pulsadores (o en su defecto de interruptores).
4. Un módulo de ampliación (Gateway) maestro AS-Interface, referencia TWDNOI10M3, marca Telemecanique.
5. Una fuente AS-i referencia ABLB3002, marca Telemecanique.
6. Dos esclavos AS-i: una Interface IP20 compacta referencia ASI20MT4I4OR y una caja con dos pulsadores XAL-S2002E, ambos marca Telemecanique.
7. Un sensor óptico referencia XUB0 APSN L2, marca Telemecanique.
8. Un bus de campo AS-i, marca Telemecanique.

La E3T-UIS cuenta con todo el equipo necesario para esta práctica, que es prestado a sus estudiantes en el Almacén E3T ubicado en el edificio de ingeniería civil, para ser utilizado en los laboratorios de la E3T-UIS (que están en el edificio de ingeniería civil) en el horario de 7:00 am a 12:00 pm y de 1:00 p.m a 8:00 p.m.

OBJETIVO

Aplicar los conocimientos adquiridos mediante la lectura de los documentos citados en el ítem “Requisito” a un subproceso de una planta de embotellado que es el embasado de un producto.

RECOMENDACIONES

- Utilice un temporizador de retardo a la conexión (TON) para controlar el tiempo de llenado de cada botella.
- El temporizador TON guarda el tiempo de preselección (final del conteo) en la palabra %T*Mi*.P (donde *i* es el numero de temporizador y será un valor entre 0 y 127, ya que pueden usarse hasta 128 temporizadores) que puede modificarse en el programa mediante una orden de asignación. Este temporizador también guarda el valor actual de su cuenta en la palabra %T*Mi*.V.
- Utilice bobinas set y reset.
- Tanto a las secciones como a los escalones (rungs) se les pueden introducir comentarios que permiten organizar mejor el programa y hacerlo más fácil de entender para cualquier persona diferente del autor.

- Llegue al laboratorio con una primera versión del programa completo, pues depurar (eliminarle los errores) esa primera versión puede requerir las dos horas que dura el laboratorio y si llega al laboratorio a hacerla no alcanzara a depurarla.
- Si no cuenta con la regleta de pulsadores puede usar la de interruptores para simular los pulsadores; esta simulación es exitosa si las entradas del PLC Twido conectadas a dicha regleta de interruptores se asignan a contactos de flanco ascendente en el programa.
- Simule el motor mediante una salida del PLC Twido.

VOCABULARIO

HMI (Human-Machine Interface): Interface Hombre-Máquina.

Dosificador: Tanque de forma similar a una tolva (cilíndrico en su parte superior y con forma de cono recto invertido en su parte inferior) en cuya parte más baja se encuentra una válvula.

DESARROLLO

1. Cree una red AS-i (en la opción describir del software TwidoSuite) con los dos esclavos AS-i citados en el numeral cuatro del ítem “equipo necesario”.
2. Para la creación del programa sugerido en el próximo numeral (3) deberán hacerse las siguientes suposiciones (que dejaran de serlo en la quinta practica puesto que serán otros procesos de la planta de embotellado que también se implementaran):

- Se llenaran las botellas con una sola clase de líquido para evitar tener que lavar el dosificador.
- Siempre habrá líquido en el dosificador.
- Siempre habrá botellas en la banda transportadora antes del dosificador.
- Otro sistema (cuya programación no forma parte del objetivo de esta práctica) se encargara de retirar las botellas llenas al final de la banda transportadora para llevarlas al sistema de taponado.
- Diseñe un programa que cumpla lo siguiente:
- La entrada 0 será el Run/Stop del PLC.
- El arranque del sistema envasado de producto se hace localmente (pulsador conectado a una entrada del PLC).
- La parada del sistema envasado de producto podrá hacerse localmente (pulsador conectado a una entrada del PLC) o remotamente (botón verde de la caja con dos pulsadores). La parada del sistema detendrá el motor de la banda transportadora.
- Habrá dos paradas de emergencia (la local en una entrada del PLC y la remota en el botón rojo de la caja con dos pulsadores), que bloquean el arranque; dicho arranque quedara nuevamente operativo cuando se desbloquee mediante otra entrada del PLC (esta entrada normalmente se encuentra vinculada a otro PLC o a una HMI, pero aquí no se hará así para no incrementar la complejidad de la práctica, puesto que es la primera; no obstante esto se aplicara en la quinta practica).

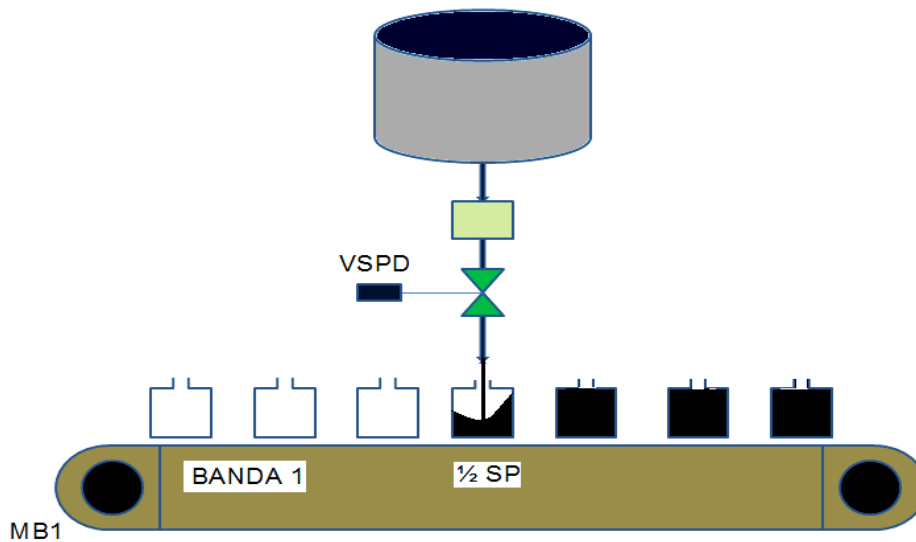
- Cuando la botella pase por el sensor de posición (simulado en esta práctica mediante un fotodetector) deberá detenerse el motor de la banda transportadora y abrirse el dosificador de líquido para empezar el llenado de la botella que durará 10 segundos, después de los cuales el dosificador se cerrará y el motor de la banda transportadora reiniciará su movimiento.
- Si la parada normal o la de emergencia se produce durante el llenado de una botella el sistema deberá almacenar el tiempo de llenado transcurrido entre el inicio del llenado y la parada normal o de emergencia, para cumplir con el tiempo faltante una vez se arranque nuevamente el sistema, evitando de esta manera que se derrame líquido en la banda transportadora.
- Tanto la parada normal como la parada de emergencia del sistema envasado de producto durante el llenado de una botella cerrarán la válvula del dosificador.

En resumen el sistema usará cuatro entradas del PLC (arranque, parada local, parada de emergencia local y desbloqueo), las dos entradas AS-i de la caja con dos pulsadores (parada remota y parada de emergencia remota), una entrada AS-i de la interface IP20 compacta para el sensor de posición (fotodetector) ubicado debajo del dosificador que detecta la botella en posición de llenado y dos salidas del PLC (una para la válvula del dosificador y otra para el arranque/parada del motor de la banda transportadora).

1. Implemente la red AS-i descrita en el numeral 1 (con el equipo necesario 4 al 8).
2. Transfiera el programa del PC al PLC usando el cable de programación (USB a MiniDin de 8 pines) del PLC Twido.
3. Active la entrada run/stop del PLC.

4. Usando la regleta de interruptores o de pulsadores y el sensor óptico, verifique que su programa cumpla todos los requisitos agrupados en el paso 3.

Figura 1: sistema de envasado de producto simplificado.



Fuente: Tesis de grado “Redes de comunicaciones industriales basadas en profibus”, pg 314. Modificada por los autores.

Nota: En la grafica 1 se observan las variables (válvula salida producto del dosificador), MB1 (motor banda 1) y SP (sensor de posición); este sensor funcionara como un contacto normalmente abierto, es decir, se cierra (envía un nivel alto -1-) cuando una botella pasa frente a él.

PRACTICA 2: MODBUS (SOBRE LINEA SERIAL)

REQUISITO

Para facilitar el desarrollo de esta práctica se recomienda leer el capítulo 3 y el anexo 2 (anexo modbus) del proyecto de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”, o por lo menos el anexo 2.

EQUIPO NECESARIO

1. Un PLC Twido referencia TWDLCAE40DRF, marca Telemecanique.
2. Un cable de programación (USB a MiniDin de 8 pines) marca Telemecanique, para el PLC Twido.
3. Un modulo de ampliación (Gateway) maestro AS-Interface, referencia TWDNOI10M3, marca Telemecanique.
4. Una fuente AS-i referencia ABLB3002, marca Telemecanique.
5. Dos esclavos AS-i: una interface IP20 compacta referencia ASI20MT4I4OR y una caja con dos pulsadores XAL-S2002E, ambos marca Telemecanique.
6. Un sensor óptico referencia XUB0 APSN L2, marca Telemecanique.
7. Una regleta de pulsadores (o en su defecto de interruptores).
8. Un bus de campo AS-i.

9. Un variador de velocidad Altivar 31 (ATV31) referencia ATV31HU22M2, marca Telemecanique.
10. Un motor trifásico referencia referencia DL 1021N, marca De Lorenzo.
11. Nueve cables con conector tipo banana en cada extremo (cables banana-banana).
12. Dos cables Ethernet (Cable UTP con conectores RJ45 macho en cada extremo). Ambos directos (no cruzados).
13. Un cable modbus para Twido (conector MiniDIN de 8 pines en un extremo y conector RJ45 macho en el otro extremo).
14. Dos uniones tipo T (tres puertos RJ45 hembra) para protocolo modbus (una para maestro y otra para esclavo).
15. Dos terminaciones de línea (bus) para las uniones tipo T del numeral anterior. Estas terminaciones de línea son conectores RJ45 macho con una resistencia de $120\ \Omega$ y un condensador de $1\ \text{nF}$ conectados en serie entre los terminales 4 y 5 de cada conector.

La E3T-UIS cuenta con todo el equipo necesario para esta práctica, que es prestado a sus estudiantes en el Almacén E3T ubicado en el edificio de ingeniería civil, para ser utilizado en los laboratorios de la E3T-UIS (que están en el edificio de ingeniería civil) en el horario de 7:00 a.m a 12:00 p.m y de 1:00 p.m a 8:00 p.m.

OBJETIVO

Aplicar los conocimientos adquiridos a través de la lectura de los documentos citados en el ítem “Requisito” a un subproceso de una planta de embotellado que es el embasado de un producto. A diferencia de la practica anterior donde el funcionamiento del motor de la banda transportadora se simulo mediante una salida del PLC Twido, esta práctica integrara al montaje un motor controlado por el PLC Twido (maestro modbus) a través de un variador de velocidad (esclavo modbus).

RECOMENDACIONES

Para esta práctica también son validas las recomendaciones de la práctica anterior; además:

- Esta práctica fue diseñada para lograr el mismo objetivo de la práctica de AS-i, pero con un montaje un poco diferente; por esta razón los programas de esta y de la practica anterior serán muy parecidos, y por consiguiente será más fácil y rápido lograr el programa de esta práctica a partir de modificaciones del programa de la practica anterior que empezando a programar desde cero. Cuando se crea un programa en TwidoSuite este software crea una carpeta y un archivo ejecutable con el mismo nombre, por defecto, en la carpeta mis proyectos que puede accederse siguiendo la ruta C:\Program Files\Schneider Electric\TwidoSuite\Mis proyectos (no obstante dicha ubicación es modificable); vaya a esta carpeta o a donde haya guardado el programa, copie la carpeta y el archivo ejecutable del programa de la práctica de AS-i y péguelos en una ubicación diferente (por ejemplo la carpeta mis documentos o el escritorio), cámbieles el nombre por uno apropiado (por ejemplo “practica modbus”); puede dejarlos en esta nueva ubicación o cortarlos y pegarlos en la ubicación original (donde estaban antes de copiarlos y cambiarles el nombre). Luego del procedimiento anterior podrá abrir el

programa con el nuevo nombre y modificarlo para lograr el objetivo de esta práctica, sin que esto afecte el programa de la práctica de AS-i.

- Si no le queda claro el procedimiento de la recomendación anterior saque pantallazos del programa de la práctica de AS-i y vaya copiando el programa de esos pantallazos en un programa nuevo en TwidoSuite; cuando termine de copiarlo podrá empezar a hacer las modificaciones necesarias para lograr el objetivo de esta práctica. Este procedimiento es un poco más demorado que el anterior, pero aun así es mucho más rápido que empezar a hacer el programa desde el principio.
- En el software TwidoSuite podrá encontrar ayuda sobre los macros drive siguiendo la ruta “configurar\configurar los datos\objetos avanzados\macros drive\ayuda sobre las macros drive”. Tenga en cuenta que en dicha ayuda el término “instancia” hace referencia al número del conjunto de macros que se está usando (Drive 0, Drive 1,..., Drive 31) que no necesariamente tiene que coincidir con la dirección en la red del dispositivo cuya comunicación con el PLC/modulo maestro está administrando dicho conjunto de macros.

VOCABULARIO

- TwidoSuite: Software gratuito de Telemecanique creado exclusivamente para programar los autómatas de la gama Twido.
- Macros Drive: ordenes que evitan tener que programar el diagrama de estados del variador de velocidad y la gestión de las comunicaciones (instrucciones de envío o recepción de datos ejecutadas según tablas de palabras).

DESARROLLO

Ejecute los pasos 1 a 4 del ítem “Desarrollo” de la práctica de AS-i y después los siguientes pasos:

5. Cree una red modbus (en la opción describir del software TwidoSuite) con el maestro modbus (PLC Twido) y el esclavo modbus (variador de velocidad ATV31).
6. Programe los macros Drive necesarios para reemplazar la simulación del motor mediante una salida del PLC por ordenes al variador vía modbus que produzcan efectos sobre el motor descrito en el numeral 10 del ítem equipo necesario.
7. Conecte el motor en estrella (Y) usando los cables banana-banana y las salidas del variador con las correspondientes entradas del motor (U2 del motor con U/T1 del variador, V2 del motor con V/T2 del variador y W2 del motor con W/T3 del variador). No olvide conectar el neutro del motor con el neutro del variador.
8. Conecte las dos fases de entrada del variador y su neutro con dos fases y el neutro de la fuente trifásica del laboratorio.
9. Introduzca al variador de velocidad ATV31 los datos de placa del motor que se va a utilizar; esto se puede hacer en el menú de control “drC”, donde aparecen las siguientes opciones: “bFr” es la frecuencia estándar del motor, “UnS” la tensión nominal del motor, “nCr” la corriente nominal del motor, “nSP” la velocidad nominal del motor y “COS” es el coseno del ángulo de desfase.

10. Configure en el variador los mismos parámetros modbus (velocidad, paridad, bits de parada, modo RTU o ASCII, time out y tiempo entre tramas) que en el PLC Twido. En el variador estos parámetros están en el menú de comunicaciones “COM”, mientras que en el PLC Twido estos parámetros se configuran en su programa en TwidoSuite haciendo doble clic en el bus de campo de la red modbus creada en paso 5; no olvide configurar la dirección de esclavo modbus al variador (también en el menú de comunicaciones “COM”).
11. Implemente el bus de campo con el equipo necesario 12 al 15 y úselo para crear la red modbus del paso 5.
12. Transfiera el programa del PC al PLC usando el cable de programación (USB a MiniDin de 8 pines) del PLC Twido.
13. Active la entrada run/stop del PLC.
14. Use la regleta de interruptores o de pulsadores y el fotodetector para verificar que su programa cumpla todos los requisitos agrupados en el paso 3 del ítem “Desarrollo” de la práctica de AS-i, haciendo énfasis en que el comportamiento del motor controlado mediante el resultado de los pasos 8 a 11 del ítem “Desarrollo” de esta práctica este de acuerdo dichos requisitos. Nota: esta práctica no incluye grafica porque es válida la misma grafica de la practica 1 (grafica 1, pg 6).

PRACTICA 3: CANopen

REQUISITO

Para facilitar el desarrollo de esta práctica se recomienda leer el capítulo 5 y el anexo 4 (anexo CANopen) del proyecto de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”, o por lo menos el anexo 4.

EQUIPO NECESARIO

1. Un PLC Twido referencia TWDLCAE40DRF, marca Telemecanique.
2. Un cable de programación (USB a MiniDin de 8 pines) marca Telemecanique, para el PLC Twido.
3. Una regleta de pulsadores (o en su defecto de interruptores).
4. Un módulo de ampliación (Gateway) maestro AS-Interface, referencia TWDNOI10M3, marca Telemecanique.
5. Una fuente AS-i referencia ABLB3002, marca Telemecanique.
6. Dos esclavos AS-i: una interface IP20 compacta referencia ASI20MT4I4OR y una caja con dos pulsadores XAL-S2002E, ambos marca Telemecanique.
7. Un sensor óptico referencia XUB0 APSN L2, marca Telemecanique.

8. Un bus de campo AS-i.
9. Un modulo de ampliación (Gateway) maestro CANopen, referencia TWDNCO1M, marca Telemecanique.
10. Una fuente DC de 24V.
11. Un variador de velocidad Altivar 31 (ATV31) referencia ATV31HU22M2, marca Telemecanique.
12. Un motor trifásico referencia DL 1021N, marca De Lorenzo.
13. Nueve cables con conector tipo banana en cada extremo (cables banana-banana).
14. Dos cables Ethernet (Cable UTP con conectores RJ45 macho en cada extremo). Ambos directos (no cruzados).
15. Un cable CANopen para Twido (conector DB9 hembra en un extremo y conector RJ45 macho en el otro extremo).
16. Dos uniones tipo T (tres puertos RJ45 hembra) para protocolo CANopen.
17. Dos terminaciones de línea para las uniones tipo T del numeral anterior. Estas terminaciones de línea son conectores RJ45 macho con una resistencia de $120\ \Omega$ y $\frac{1}{4}\ W$ conectada entre los terminales 1 y 2 de cada conector.

La E3T-UIS cuenta con todo el equipo necesario para esta práctica, que es prestado a sus estudiantes en el Almacén E3T ubicado en el edificio de ingeniería civil, para ser utilizado en los laboratorios de la E3T-UIS (que están en el edificio de ingeniería civil) en el horario de 7:00 a.m a 12:00 p.m y de 1:00 p.m a 8:00 p.m.

OBJETIVO

Aplicar los conocimientos adquiridos con la ayuda de los documentos citados en el ítem “Requisito” a un subproceso de una planta de embotellado, que es el taponado de botellas.

RECOMENDACIONES

Si usted decide consultar documentación del fabricante (Telemecanique), debe tener en cuenta que en el pdf “Twido Controladores programables Módulos de comunicación Guía de hardware” de mayo del 2009, así como en su versión de septiembre de 2007 aparece un error en la pagina 88 en la tabla en la que se explica el funcionamiento del led de estado del maestro CANopen (numeral 9 del ítem “equipo necesario” de esta práctica); el bus esta activado (sin errores) cuando dicho led está iluminado, no cuando este led está apagado como lo afirma el pdf en cuestión.

VOCABULARIO

Taponador: sistema con un cilindro neumático dotado con un sensor de presión y otro sensor de posición inicial, que sirve para tapar botellas.

DESARROLLO

1. Cree una red AS-i (en la opción “describir” del software TwidoSuite) con el maestro y los dos esclavos citados en los numerales cuatro y seis, respectivamente, del ítem “equipo necesario”.
.
2. Cree una red CANopen (en la opción “describir” del software TwidoSuite) con el maestro y el esclavo citados en los numerales 9 y 11, respectivamente, del ítem “equipo necesario”.
3. Para la creación del programa sugerido en el próximo paso (4) deberán hacerse las siguientes suposiciones (que dejaran de serlo en la quinta práctica puesto que serán otros procesos de la planta de embotellado que también se implementaran).
 - Siempre habrá botellas llenas en la banda transportadora antes del taponador.
 - Otro sistema (cuya programación no forma parte del objetivo de esta práctica) se encargara de retirar las botellas llenas y tapadas al final de la banda transportadora para llevarlas al sistema de empacado.
4. Diseñe un programa que cumpla lo siguiente:
 - La entrada 0 será el Run/Stop del PLC.
 - El arranque del sistema taponado de botellas se hace localmente (pulsador conectado a una entrada del PLC).

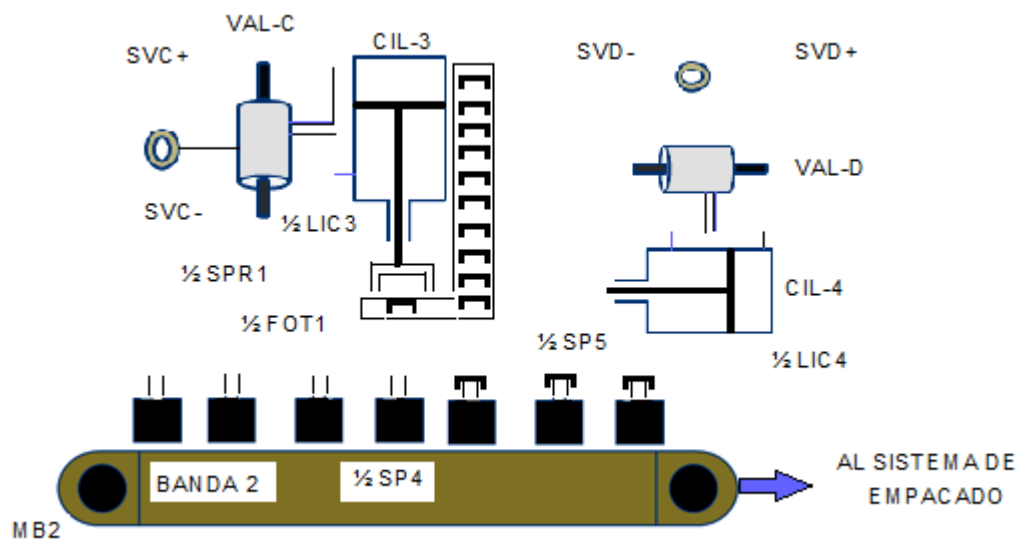
- La parada del sistema taponado de botellas podrá hacerse localmente (pulsador conectado a una entrada del PLC) o remotamente (botón verde de la caja con dos pulsadores). La parada del sistema detendrá el motor de la banda transportadora.
- Habrá dos paradas de emergencia (la local en una entrada del PLC y la remota en el botón rojo de la caja con dos pulsadores), que bloquean el arranque; dicho arranque quedara nuevamente operativo cuando se desbloquee mediante otra entrada del PLC (esta entrada normalmente se encuentra vinculada a otro PLC o a una HMI, pero aquí no se hará así para no incrementar la complejidad de la practica; no obstante esto se aplicara en la quinta practica).
- Un sensor de posición se encargara de detener la banda transportadora mientras se produzca el taponado de una botella.
- El taponador tendrá un cilindro auxiliar que será el encargado de llevar cada tapa del sistema alimentador de tapas al punto x (ver grafica 2 en la página 15) en la trayectoria del taponador.
- Un sensor de posición indicara que hay una tapa lista en el alimentador de tapas para ser llevada al punto x (en la trayectoria del taponador).
- Un fotodetector indicara que hay una tapa en el punto x y dará la señal al cilindro auxiliar para que vuelva a su posición inicial.
- Si el taponador está en su posición inicial, el cilindro auxiliar también y hay una tapa en el punto x sobre la trayectoria del taponador, el taponador llevara la tapa del punto x al sitio de taponado (sobre la botella).
- Un sensor de presión se encargara de regresar el taponador a su posición inicial para poner fin al taponado de una botella.

- Finalizado el proceso de taponado de una botella el motor de la banda transportadora debe ponerse nuevamente en movimiento para llevar la siguiente botella llena a la posición de taponado.
 - Cada vez que el sistema pare (parada normal o de emergencia) deberá memorizar su estado (el estado de sus variables) para retomarlo cuando vuelva arrancarse.
5. Programe los macros Drive necesarios para enviar órdenes al variador vía CANopen que paren y arranquen el motor descrito en el numeral 12 del ítem equipo necesario.
 6. Conecte el motor en estrella (Y) usando los cables banana-banana y las salidas del variador con las correspondientes entradas del motor (U2 del motor con U/T1 del variador, V2 del motor con V/T2 del variador y W2 del motor con W/T3 del variador). No olvide conectar el neutro del motor con el neutro del variador.
 7. Conecte las dos fases de entrada del variador y su neutro con dos fases y el neutro de la fuente trifásica del laboratorio.
 8. Introduzca al variador de velocidad ATV31 los datos de placa del motor que se va a utilizar; esto se puede hacer en el menú de control “drC”, donde aparecen las siguientes opciones: “bFr” es la frecuencia estándar del motor, “UnS” la tensión nominal del motor, “nCr” la corriente nominal del motor, “nSP” la velocidad nominal del motor y “COS” es el coseno del ángulo de desfase.
 9. Configure la dirección del variador ATV31 y la velocidad de transmisión en la red CANopen en el menú de comunicaciones “COM” del variador. Esta velocidad debe coincidir el dato de velocidad disponible al hacer doble clic sobre el bus de campo de la red CANopen creada en el paso 2.

10. Implemente la red AS-i descrita en el numeral 1 (con el equipo necesario 4 al 8).

Figura 2: sistema taponado de botellas simplificado

Fuente: Tesis de grado “Redes de comunicaciones industriales basadas en profibus”, pg 314. Modificada por los autores



Fuente: Tesis de grado “Redes de comunicaciones industriales basadas en profibus”, pg 314. Modificada por los autores.

11. Implemente el bus de campo con el equipo necesario 14 al 17 y úselo para crear la red CANopen del numeral 2.

12. Transfiera el programa del PC al PLC usando el cable de programación (USB a MiniDin de 8 pines) del PLC Twido.
13. Active la entrada run/stop del PLC.
14. Use la regleta de interruptores o de pulsadores y el fotodetector para verificar que su programa cumpla todos los requisitos agrupados en el paso 4 del ítem “Desarrollo” de esta práctica.
15. Realice la programación necesaria para hacer sin macros Drive lo que se hace en el numeral 5. Inhabilite temporalmente lo que hizo en el paso 5 usando, las veces que necesite, la instrucción “open”.
16. Vuelva a ejecutar los pasos 10 al 12 para verificar que la programación sugerida en el paso anterior (13) funcione.

Nota: En la grafica 2 se observan las salidas MB2 (motor banda 2), SVC+ (solenoides válvula C elongando el taponador), SVC- (solenoides válvula C retrayendo el taponador), SVD+ (solenoides válvula D elongando el cilindro auxiliar) y SVD- (solenoides válvula D retrayendo el cilindro auxiliar). También aparecen las entradas SPR1 (sensor de presión 1), Fot1 (tapa en el punto x), LIC3 (límite inferior del taponador), SP4 (botella en posición de taponado), SP5 (tapa en el alimentador de tapas lista para ser llevada al punto x) y LIC4 (límite inferior cilindro auxiliar). El centro de la elipse roja es el “punto x” y el centro de la elipse amarilla es el “sitio de taponado”.

PRACTICA 4: MODBUS TCP/IP

REQUISITO

Haber leído y entendido el capítulo 4 y el anexo 3 (anexo modus TCP/IP) del trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO” o por lo menos el anexo 3. También es necesario haber desarrollado las practicas anteriores (AS-i, Modbus sobre línea serial y CANopen).

EQUIPO NECESARIO

1. Dos PLC Twido referencia TWDLCAE40DRF, marca Telemecanique.
2. Un cable de programación (USB a MiniDin de 8 pines) del PLC Twido.
3. Dos regletas de pulsadores (o en su defecto de interruptores).
4. Dos módulos de ampliación (Gateway) maestro AS-Interface, referencia TWDNOI10M3, marca Telemecanique.
5. Dos fuentes AS-i referencia ABLB3002, marca Telemecanique.
6. Cuatro Esclavos AS-i: dos Interfaces IP20 compactas referencia ASI20MT4I4OR y dos cajas con dos pulsadores XAL-S2002E, todos marca Telemecanique.
7. Dos sensores ópticos referencia XUB0 APSN L2, marca Telemecanique.

8. Dos buses de campo AS-i.
9. Un modulo de ampliación (Gateway) maestro CANopen, referencia TWDNCO1M, marca Telemecanique.
10. Una fuente DC de 24V.
11. Dos variadores de velocidad Altivar 31 (ATV31) referencia ATV31HU22M2, marca Telemecanique.
12. Dos motores trifásicos referencia DL 1021N, marca De Lorenzo.
13. Dieciocho cables con conector tipo banana en cada extremo (cables banana-banana).
14. Un cable CANopen para Twido (conector DB9 hembra en un extremo y conector RJ45 macho en el otro extremo).
15. Dos uniones tipo T (tres puertos RJ45 hembra) para protocolo CANopen.
16. Dos terminaciones de línea para las uniones tipo T del numeral anterior.
17. Un cable modbus para Twido (conector MiniDIN de 8 pines en un extremo y conector RJ45 macho en el otro extremo).
18. Dos uniones tipo T (tres puertos RJ45 hembra) para protocolo modbus (una para maestro y otra para esclavo).
19. Dos terminaciones de línea para las uniones tipo T del numeral anterior.

20. Siete cables Ethernet (Cable UTP con conectores RJ45 macho en cada extremo). Todos directos (no cruzados).

21. Un switch.

La E3T-UIS cuenta con todo el equipo necesario para esta práctica, que es prestado a sus estudiantes en el Almacén E3T ubicado en el edificio de ingeniería civil, para ser utilizado en los laboratorios de la E3T-UIS (que están en el edificio de ingeniería civil) en el horario de 7:00 a.m a 12:00 p.m y de 1:00 p.m a 8:00 p.m.

OBJETIVO

Esta práctica pretende interconectar los resultados de las tres prácticas anteriores, es decir, busca interconectar dos procesos de una planta de embotellado (el envasado de producto y el taponado de botellas), de forma que el estudiante tome mayor conciencia de la utilidad de los protocolos de comunicación industrial y al mismo tiempo aumente su experiencia para que adquiera la confianza en sí mismo que necesitara para la quinta y última práctica, en donde se agregara un tercer proceso de la planta de embotellado a los dos procesos implementados mediante el desarrollo de esta práctica.

RECOMENDACIONES

- Verifique que la red tenga acceso a los dos PLC conectados a ella antes de hacer cualquier prueba; lo anterior se logra abriendo la herramienta “símbolo de sistema” disponible a través de la ruta inicio\todos los programas\accesorios\símbolo de

sistema y haciendo “ping” por turnos a los dos PLC. Hacer ping a un PLC quiere decir digitar (en la herramienta símbolo de sistema) la palabra “ping” seguida por la dirección IP asignada al PLC y pulsar la tecla “intro” (o enter); la orden ping enviara 4 bytes de datos al PLC con la dirección IP que precede y mostrara en la herramienta símbolo de sistema si estos bytes fueron recibidos o no recibidos (perdidos). Si fueron recibidos podrán hacerse las pruebas sugeridas en el desarrollo de esta práctica, de lo contrario debe revisarse que los cables Ethernet estén bien conectados, que las pestañas de estos cables no estén partidas y que el switch este funcionando y registre actividad en sus puertos usados, además revisar en el software TwidoSuite que el programa del PLC muestre la misma dirección IP que aquella con la que se le está haciendo ping.

VOCABULARIO

- Switch: equipo que interconecta dos o más segmentos de red. Analiza las tramas que le llegan y genera unas tablas de tráfico que le permiten dirigir cada trama únicamente al puerto de salida donde se encuentre el equipo destinatario, eliminando así el tráfico de datos inútiles para optimizar el ancho de banda de la red y evitando las colisiones (entre tramas).

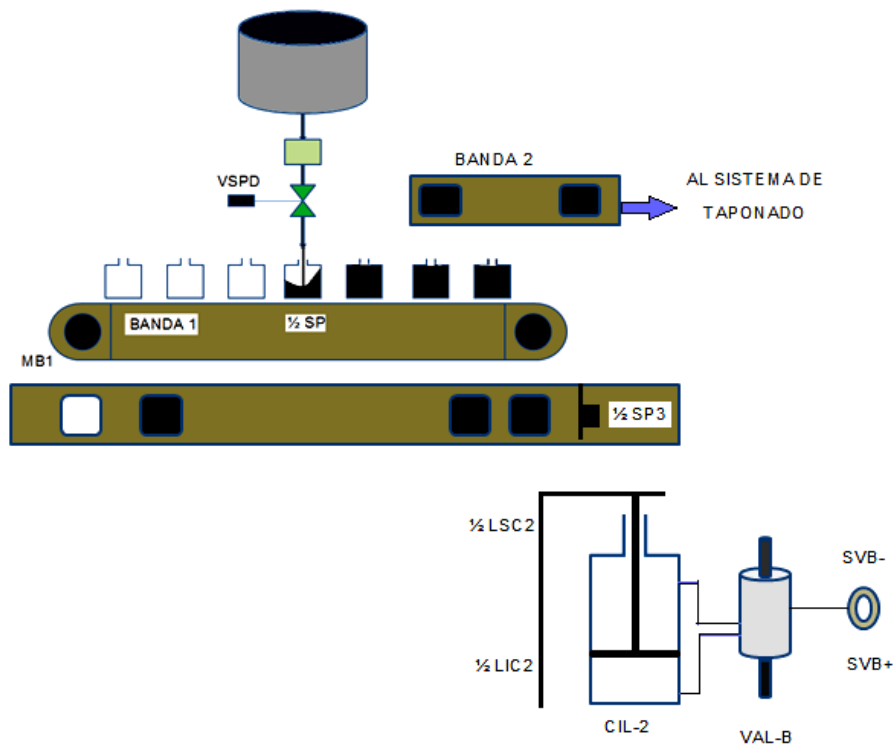
DESARROLLO

1. Para la creación del programa sugerido en el paso 3 deberán hacerse las siguientes suposiciones (que dejaran de serlo en la quinta practica puesto que serán otros procesos de la planta de embotellado que también se implementaran):
 - Se llenaran las botellas con una sola clase de líquido para evitar tener que lavar el dosificador.

- Siempre habrá líquido en el dosificador.
 - Siempre habrá botellas en la banda transportadora 1 antes del dosificador.
 - Otro sistema (cuya programación esta por fuera del alcance de esta práctica) se encargara de retirar las botellas llenas y tapadas al final de la banda transportadora 2 para llevarlas al sistema de empacado.
2. Observe la grafica al final de esta práctica para que entienda como el cilindro al final de la banda transportadora 1 obliga a introducir algunas modificaciones en los programas de las prácticas de modbus sobre línea serial y de CANopen para que este par de programas (en PLCs diferentes) interactúen. Dichas modificaciones deben satisfacer los siguientes requerimientos:
- Las paradas normales y las de emergencia locales y remotas de cualquiera de los dos PLC Twido deben detener completamente las redes modbus y CANopen, es decir, las paradas de un PLC Twido deben detener los procesos (desactivar salidas) de ese PLC Twido y del otro PLC Twido.
 - Cuando una botella llena llegue al final de la banda transportadora 1 el sensor de posición 3 lo detectara y detendrá dicha banda mientras el cilindro 2 lleva esa botella al inicio de la banda transportadora 2 y regresa a su posición inicial (limite inferior); esta banda también se detendrá cuando una botella este debajo del dosificador (posición de llenado) y reactivara su marcha 10 segundos después (tiempo de llenado).
 - La banda transportadora 2 estará en movimiento cuando se termine de poner la tapa a una botella o cuando el cilindro 2 haya puesto la botella en el inicio de dicha banda (cilindro dos en su límite superior) y se detendrá cuando una botella llegue al sitio de taponado o cuando el cilindro 2 regrese a su posición inicial (limite inferior).

3. Asigne una dirección IP a cada PLC (en cada programa) para que puedan comunicarse entre ellos; procure que la dirección IP que asigne a cada PLC Twido sea la misma que tiene escrita en su cara frontal. Si con el procedimiento descrito en el ítem “recomendaciones” comprueba que la dirección IP que tiene el PLC Twido coincide con la que usted le asigne en el programa (software TwidoSuite), entonces podrá transferir el programa usando la red TCP/IP (switch), pero si no coincide tendrá que transferir el programa al PLC Twido usando el cable descrito en el numeral 2 del ítem “equipo necesario”.
4. Implemente dos redes AS-i (una en cada PLC Twido) con asignando a cada una de ellas la mitad de los equipos descritos en los numerales 1 y del 3 al 8 del ítem “equipo necesario”.
5. Implemente una red modbus sobre línea igual a la de la practica 2.
6. Implemente una red CANopen igual a la de la practica 3.
7. Interconecte los dos PLCs usando el switch (que debe alimentarse con una fuente DC de 24 V).
8. Transfiera el programa de la práctica de modbus sobre línea serial modificado (según el paso 1) a un PLC Twido y el programa de la práctica de CANopen modificado (según el paso 1) a otro PLC Twido.
9. Active la entrada run/stop de cada PLC Twido.
10. Use la regleta de interruptores o de pulsadores y el sensor óptico para verificar que su programa cumpla todos los requisitos agrupados en el numeral 4 del ítem “Desarrollo” de esta práctica.

Figura 3: sistema envasado de producto interconectado con sistema taponado de botellas.



Fuente: Tesis de grado “Redes de comunicaciones industriales basadas en profibus”, pg 314. Modificada por los autores.

PRACTICA 5: PLANTA EMBOTELLADORA

REQUISITO

Haber desarrollado las cuatro practicas anteriores (AS-i, Modbus sobre línea serial, CANopen y Modbus TCP/IP).

EQUIPO NECESARIO

1. Dos PLC Twido referencia TWDLCAE40DRF, marca Telemecanique.
2. Un cable de programación (USB a MiniDIN de 8 pines) marca Telemecanique, para el PLC Twido.
3. Tres regletas de pulsadores (o en su defecto de interruptores).
4. Dos módulos de ampliación (Gateway) maestro AS-Interface, referencia TWDNOI10M3, marca Telemecanique.
5. Dos fuentes AS-i referencia ABLB3002, marca Telemecanique.
6. Dos esclavos AS-i: una interface IP20 compacta referencia ASI20MT4I4OR y una caja con dos pulsadores referencia XAL-S2002E, ambos marca Telemecanique.
7. Dos sensores ópticos referencia XUB0 APSN L2, marca Telemecanique.

8. Dos buses de campo AS-i.
9. Un modulo de ampliación (Gateway) maestro CANopen, referencia TWDNCO1M, marca Telemecanique.
10. Una fuente DC de 24V.
11. Un variador de velocidad Altivar 31 (ATV31) referencia ATV31HU22M2, marca Telemecanique.
12. Dos motores trifásicos referencia DL 1021N, marca De Lorenzo.
13. Dieciocho cables con conector tipo banana en cada extremo (cables banana-banana).
14. Un cable CANopen para Twido (conector DB9 hembra en un extremo y conector RJ45 macho en el otro extremo).
15. Dos uniones tipo T (tres puertos RJ45 hembra) para protocolo CANopen.
16. Dos terminaciones de línea para las uniones tipo T del numeral anterior.
17. Un cable modbus para el PLC Twido (conector MiniDIN de 8 pines macho en un extremo y conector RJ45 macho en el otro extremo).
18. Dos uniones tipo T (tres puertos RJ45 hembra) para protocolo modbus (una para maestro y otra para esclavo).
19. Dos terminaciones de línea para las uniones tipo T del numeral anterior.

20. Nueve cables Ethernet (Cable UTP con conectores RJ45 macho en cada extremo). Todos directos (no cruzados).

21. Un switch.

22. Un PLC M340.

23. Un cable de programación, marca Telemecanique y referencia BMXXCAUSBH018, para el PLC M340.

24. Una interface hombre-máquina (HMI) Magelis XBT-GT 2330.

25. Un cable de programación, marca Telemecanique y referencia XBT ZG935, para la HMI Magelis (terminales USB macho en ambos extremos).

La E3T-UIS cuenta con todo el equipo necesario para esta práctica, que es prestado a sus estudiantes en el Almacén E3T ubicado en el edificio de ingeniería civil, para ser utilizado en los laboratorios de la E3T-UIS (que están en el edificio de ingeniería civil) en el horario de 7:00 a.m a 12:00 p.m y de 1:00 p.m a 8:00 p.m.

OBJETIVOS

- Integrar el PLC M340 al montaje de la práctica anterior; este PLC se encargará de controlar un tercer proceso (llenado de tanques) de la planta de embotellado, que precederá al de emvasado de producto.

- Implementar en la HMI Magelis los mímicos contruidos en Vijeo-Designer (graficas 4, 6 y 8) por los autores del trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”

RECOMENDACIONES

Fabrique dos o tres cables ethernet directos, ya que a veces los que presta el almacén E3T (por ser de dominio público) tienen partidas las pestañas de los conectores RJ45 macho, lo que llevara a desconexiones constantes de los conectores RJ45 macho y hembra. Cables de un metro de longitud suelen ser suficientes: un metro de cable UTP tiene un precio aproximado en el mercado de \$1.000 y cada ponchada (como se le conoce popularmente al servicio de unir de forma permanente un conector RJ45 macho a un cable UTP, prestado por ferreterías y almacenes de dispositivos electrónicos) vale \$1.200, luego el cable completo valdría \$3.400. Otra opción es agregar una ponchadora (herramienta que hace las “ponchadas”) a su caja de herramientas, lo que reduciría el precio del cable a completo a \$1.500 (\$1.000 de un metro de cable UTP y \$250 de cada conector RJ45 macho); esta reducción de precio es relativa, ya que la ponchadora tiene un costo promedio de \$30.000.

VOCABULARIO

- Unity Pro: software proporcionado por el fabricante (Telemecanique) del PLC M340 para programar este y otros PLC de gama alta del mismo fabricante. Una ventaja del Unity Pro con respecto al TwidoSuite es que el primero, a diferencia del segundo, permite abrir dos o más programas simultáneamente; el Unity-Pro coordina la transferencia de un programa entre el PC y el PLC m340, hecha a través cable USB o Ethernet (numerales 23 y 20 respectivamente del ítem “equipo necesario”).

- Vijeo Designer: software proporcionado por el fabricante (Telemecanique) de la HMI Magelis XBT GT 2330 para hacer en el PC los programas de la HMI; cada programa es un conjunto de paneles o pantallazos que podrán observarse en la HMI. Este software también coordina la transferencia de un programa a la HMI (la HMI no puede ejecutar dos programas simultáneamente), realizada mediante un cable USB o vía ethernet (numerales 25 y 20 respectivamente del ítem “equipo necesario”).
- Embolo: parte móvil de forma cilíndrica (con diámetro de la base de mayor longitud que su altura) que se mueve dentro de un cilindro (neumático o hidráulico) mediante la presión ejercida por el fluido y en cuyo centro de una de sus caras va conectado el vástago (barra que entra o sale del cilindro).

DESARROLLO

Figura 4 (panel 1): introducción (resumen del programa).



Fuente: trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”.

1. Abra el programa de Vijeo Designer titulado “mimicos TG” y observe que los paneles 2, 3 y 4 (graficas 6, 8 y 10) corresponden a los sistemas llenado de tanques, envasado de producto y taponado de botellas que pretende hacer interactuar esta práctica. Como lo indica el panel 1 (figura 4), cada uno de estos paneles contiene una lista que explica el significado de las variables que usa (figuras 7, 9 y 11).
2. En el programa “mimicos TG” observe en el editor de variables (accesible haciendo clic en la opción “variables” de la ventana “navegador”) la distribución de las variables dentro de las palabras de la HMI y observe al mismo tiempo la gestión de las variables sugerida por el archivo en Excel “variables PE” (HMI servidora {modbus TCP/IP} de dos PLC: un M340 y un Twido; al mismo tiempo la HMI como esclava {modbus serial} de un Twido).

Figura 5 (panel 5): Información del trabajo de grado, su director y autores. Disponible al pulsar el interruptor “ver” del panel 1 (Figura 4).



Fuente: trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”.

3. Los paneles 5, 6, 7 y 8 (figuras 5, 7, 9 y 11) son ejemplos de paneles emergentes y en el programas “mímicos TG” se puede acceder o salir de ellos mediante los interruptores “ver” y “ocultar” de los paneles 1, 2, 3 y 4 (figuras 4, 6, 8 y 10) respectivamente. Verifique mediante una simulación.
4. Use el software Unity Pro para hacer un programa que controle el sistema de llenado de tanques que se presenta en la grafica 6 y que debe cumplir los siguientes requisitos:
 - Habrá arranque, parada y parada de emergencia local, pero no parada ni parada de emergencia remota. Dichas funciones trabajaran igual que las de las prácticas anteriores, pero ya no serán implementadas mediante entradas del PLC sino a través de botones en el panel 2 (mímico 1) de la HMI Magelis.

Figura 6 (panel 2): sistema llenado de tanques real.



Fuente: trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”.

- El funcionamiento del tanque 1 (pulsador FT1) controlara la entrada de producto (VEPT1) y la salida de producto al dosificador (VSPT1). La entrada de producto podrá iniciarse si el tanque 1 no está siendo lavado (VEAT1, VSAT1 cerradas y AT1 apagado), si hay producto en la tubería de alimentación (APT1), si el sistema ha sido puesto en funcionamiento mediante EST, no está en parada ni en parada de emergencia (no se han pulsado PN ni PE) y si el nivel de producto en el tanque 1 está por debajo de su nivel inferior (NIT1 en bajo). La salida de producto al dosificador podrá darse si el dosificador no está en proceso de lavado, si los sistemas llenado de tanques y envasado de producto han sido puestos en funcionamiento mediante EST y ESD respectivamente, no está en parada ni en parada de emergencia (no se han pulsado PN ni PE), si el nivel de producto en el tanque 1 está por encima de su nivel de fondo (NFT1 en alto) y si

Figura 7 (panel 6): Lista de variables del panel 2

Lista de variables del panel 2											
EST: Energiza (arranca) el sistema llenado de tanques (SLT).											
Piloto(LED) EST: Rojo fijo si el SLT esta energizado y rojo intermitente si se encuentra en parada de emergencia.											
PN: Parada normal del SLT.											
PE: Parada de emergencia de los 3 procesos.											
APT1 Y APT2: Alimentacion producto tanques 1 y 2.											
VEAT1 y VEAT2: Valvula entrada de agua tanques 1 y 2.											
VSAT1 y VSAT2: Valvula salida de agua tanques 1 y 2.											
AT1 y AT2: Ásperson tanques 1 y 2.											
FT1: Funcionamiento de Tanque 1 (VEPT1 y VSPT1).											
FT2: Funcionamiento de Tanque 2 (VEPT2 y VSPT2).											
LT1 y LT2: Lavado del Tanques 1 y 2.											
NFT1 y NFT2: Nivel de Fondo tanques 1 y 2.											
NIT1 y NIT2: Nivel inferior tanques 1 y 2.											
NST1 y NST2: Nivel superior tanques 1 y 2.											

Fuente: trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”.

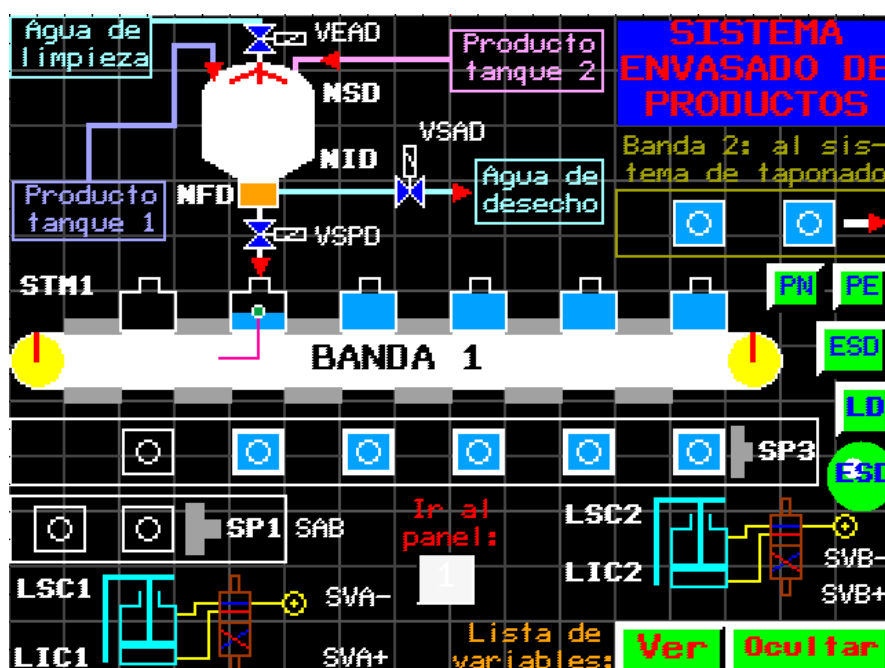
el dosificador esta vacio y limpio (o cuando el nivel del dosificador cae por debajo del nivel inferior, si ya se le ha trasferido producto).

- Si se pulsa FT2 mientras se está llenando el tanque 1 o está saliendo producto de este al dosificador, el tanque 2 podrá iniciar su proceso de llenado si el tanque 1 supera su nivel inferior, pero deberá esperar a que se acabe el producto del tanque 1 en el dosificador y a que este último se lave para proceder a enviar producto al dosificador.
- El proceso de lavado de tanque se inicia si, y solo si, no se está llenando el tanque (VEPT1 cerrada), no se está transfiriendo producto al dosificador (VSPT1 cerrada), si el tanque 1 esta vacio (NFT1 en bajo), no está activado el pulsador FT1 y el sistema de llenado de tanques ha sido activado (se ha pulsado EST sin que posteriormente se hallan pulsado PN ni PE). Una vez iniciado el lavado se abrirán las válvulas de entrada y salida de agua (VEAT1 y VSAT1 en alto) y se encenderá el aspensor (AT1 en alto); así se lavara el tanque y se sacara el agua sucia al mismo tiempo. El lavado tendrá un tiempo de duración de 10 segundos; cuando finalice este tiempo se cerrara la válvula de entrada de agua (VEAT1 en bajo) y se apagara el aspensor (AT1 en bajo), pero la válvula de salida de agua permanecerá abierta hasta que no exista nivel de fondo en el tanque. Si se está lavando el tanque y se alcanza su máximo nivel, la válvula de entrada de agua se cerrara y se apagara el aspensor, manteniéndose abierta la válvula de salida agua (sucia) hasta cuando se encuentre vacio el tanque. Es posible lavar los dos tanques simultáneamente, incluso al mismo tiempo que el dosificador.
- Debe quedar una señal que indique que el proceso de lavado de tanque ha finalizado, para autorizar el envío de producto desde el tanque 1 o desde el tanque 2.
- Si durante el funcionamiento de un tanque o el proceso de lavado ocurre una parada o una parada de emergencia el sistema debe almacenar el estado de sus variables

antes de cerrar todas las válvulas y apagar el aspersor, para reanudar su funcionamiento en las mismas condiciones en que estaba cuando se dio la parada.

- El funcionamiento del tanque 2 es igual al del tanque 1.

Figura 8 (panel 3): sistema envasado de producto real



Fuente: trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”.

Nota: cuando se dice que una válvula está cerrada implica que la variable a la que se encuentra asociada tendrá un nivel bajo (0 lógico), mientras que si está abierta su variable tendrá un nivel alto (1 lógico); esto también será válido para el aspersor y el sensor de alimentación de producto al tanque 1, donde el nivel alto en sus variables (1 lógico) significara que el aspersor esta prendido y que hay producto en la línea de alimentación, mientras un nivel bajo (0 lógico) significa el aspersor está apagado y

que no hay producto en la línea de alimentación, respectivamente. Las variables de entrada se simbolizan con letra más gruesa que las variables de salida; una válvula (entendida como el conjunto de su imagen y su variable) cerrada se simbolizara en el panel de la HMI Magelis mediante su imagen fija, mientras que cuando esté abierta aparecerá en el panel de la HMI Magelis de forma intermitente; cuando el aspensor este prendido también aparecerá intermitente, mientras que si está apagado aparecerá de forma fija. Las variables de los sensores de nivel así como las de producto en la línea de alimentación cambiaran de color cuando se encuentren en nivel alto.

5. En el software TwidoSuite programe el sistema envasado de producto con las variables de la grafica 8 (panel 3). Este programa deberá cumplir los siguientes requisitos:

Figura 9 (panel 7): Lista de variables del panel 3.

Lista de variables del panel 3									
MSD:	Nivel	superior	del	dosificador.					
NID:	Nivel	inferior	del	dosificador.					
NFD:	Nivel	de	Fondo	del	dosificador.				
VEAD:	Valvula	entrada	de	agua	al	dosificador.			
VSAD:	Valvula	salida	de	agua	del	dosificador.			
VSPD:	Valvula	salida	de	producto	del	dosificador.			
LIC1 y LSC1:	Limites	inferior	y	superior	cilindro	1.			
SVA+ y SVA-:	sacar	y	guardar	vastago	del	cilindro	1.		
LIC2 y LSC2:	Limites	inferior	y	superior	cilindro	2.			
SVB+ y SVB-:	sacar	y	guardar	vastago	del	cilindro	2.		
PN:	Parada	normal.	PE:	Parada	de	emergencia.			
ESD:	Energizado	(arranque)	del	sistema	envasado	de			
	producto	(dosificador).							
LD:	Lavado	del	dosificador.						
Piloto(LED) ESD:	Encendido	si	el	SEP	esta	funcionando			
	e	intermitente	durante	PE.					
SP1:	botella	lista	para	ser	llevada	a	la	banda	1.
SP2:	botella	en	posicion	de	llenado.				
SP3:	botella	lista	para	ser	llevada	a	la	banda	2.
SAB:	sistema	alimentador	de	botellas.					

Fuente: trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”.

El lavado del dosificador se podrá llevar a cabo cuando se pulse LD, si el dosificador esta vacío, las válvulas de entrada de producto de los tanques 1 y 2 están cerradas, la válvula de salida de producto del dosificador está cerrada, y debe haberse arrancado el sistema de envasado de producto (ESD) sin que posteriormente se haya entrado en una condición de parada normal o parada de emergencia. Si se cumplen las estas condiciones se iniciara el proceso de lavado

Abriendo las válvulas de entrada y salida de agua del dosificador y prendiendo el aspersor; el proceso de lavado finalizara a los diez segundos, cerrando la válvula de entrada de agua al dosificador y apagando el aspersor; la válvula de salida de agua (sucia) permanecerá abierta hasta cuando haya salido toda el agua del dosificador (NFD en bajo). Si el agua alcanza el nivel superior del dosificador, la válvula de entrada de agua al dosificador se cerrara, se apagara el aspersor y se mantendrá abierta la válvula de salida de agua hasta cuando el dosificador este vacío.

- El sistema alimentador de botellas (SAB) se encenderá si el sistema de envasado de producto está funcionando (se ha pulsado ESD y posteriormente no se ha pulsado ni PN ni PE), si existe producto en el dosificador y si no se está lavando el dosificador. El SAB se apagara si el nivel de producto en el dosificador cae por debajo de su nivel inferior; esto implica que entre el nivel inferior y el nivel de fondo del dosificador quedara el liquido requerido para llenar dos botellas.
- Para activar la válvula solenoide A+ que sacara el vástago del cilindro 1 (llevara su embolo hacia LSC1), el sistema envasado de producto debe estar funcionando, no debe existir una botella en la posición final del vástago (embolo en LSC1), el alimentador de botellas debe estar funcionando, el sensor de posición 1 debe estar detectando una botella (en posición para ser llevada a la banda 1), el vástago guardado (embolo en el LIC1), la válvula solenoide A- debe estar desenergizada y el motor de la banda 1 debe estar apagado (banda 1 detenida). Para desenergizar la válvula solenoide A+ el embolo debe estar en el LSC1. Para activar la válvula

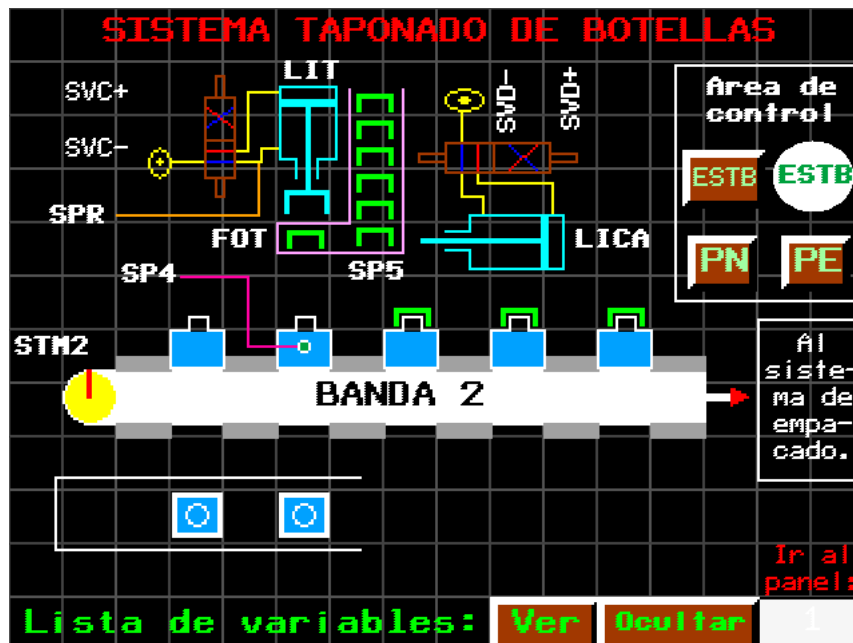
solenoides A- que guardara el vástago (llevara el embolo a LIC1) el embolo debe estar en LSC1, la válvula solenoide A+ debe estar desenergizada y el motor de la banda 1 estar apagado.

- La banda 1 operara si el sistema de envasado de producto está funcionando, no hay señal de alarma por parte del sensor térmico del motor 1, hay una botella para ser transportada o al sitio de llenado o para trasportarla al sistema de taponado. Esta banda se detendrá cuando la botella llegue a su sitio de llenado (SP2 en alto) o cuando llegue una botella al final de la banda 1 para ser transportada a la banda 2 (SP3 en alto) y cuando se hayan llenado todas las botellas que debían llenarse con el producto del dosificador y se haya terminado de enviar todas las botellas al sistema de taponado.
- La válvula solenoide B+ (que llevara la botella llena del final de la banda 1 al inicio de la banda 2) se activara si los sistemas envasado de envasado y taponado de botellas están funcionando, si el embolo del cilindro 2 está en su límite inferior y el sensor de posición 3 se activa, si la válvula solenoide B- esta desenergizada y si no hay botella en la posición final del vástago del cilindro 2 (embolo en LSC2). La válvula solenoide B+ se desactivara cuando el embolo llegue a LSC2. La válvula solenoide B- se energizara si el embolo del cilindro 2 está en su límite superior, el sensor de posición 3 esta desactivado y la válvula solenoide B+ esta desenergizada. La válvula solenoide B- se desenergiza cuando el embolo del cilindro 2 llega al LIC2.
- El envasado de producto (llenado de una botella) se iniciara si el sistema de envasado de producto está funcionando (arrancado y sin posterior parada normal o de emergencia), si las válvulas de entrada y salida de agua del dosificador están cerradas y el aspersor está apagado, el motor de la banda 1 está detenido y la botella esta posicionada debajo de la válvula salida de producto del dosificador (SP2 en alto). El llenado de cada botella dura diez segundos, después de los cuales la válvula salida de producto del dosificador se cierra y se reinicia el movimiento del motor de la banda 1.

- El comportamiento de arranque después de una parada normal o de emergencia será el mismo que el descrito para el sistema llenado de tanques.

Nota: para este proceso (envasado de producto real) sigue siendo válida la nota que precede a la grafica 8.

Figura 10 (panel 4): sistema taponado de botellas real.



Fuente: trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”.

6. Programe en TwidoSuite el sistema taponado de botellas real (grafica 10), que deberá cumplir los siguientes requisitos:

- El motor de la banda 2 girara si los sistemas taponado de botellas y envasado de producto están funcionando, su sensor térmico no registra alarma, si el cilindro 2 ha

terminado de poner la botella llena en la banda 2 y si se ha terminado de tapar una botella. El motor 2 parara cuando la botella llena llegue al sitio de taponado (SP4 en alto) y treinta segundos después de haber taponado la ultima botella (para permitir el paso del tiempo que requiere enviar todas las botellas al sistema de empacado).

Figura 11: lista de variables del panel 4 (sistema taponado de botellas real).

Lista de variables del panel 4															
ESTB: energizado (arranque) del sistema taponado de botellas.															
PM: parada normal. PE: parada de emergencia.															
Piloto(LED) ESTB: encendido cuando el STB esta funcionando e intermitente cuando el STB esta en PE.															
FOT: Fotodetector. SPR: sensor de presion.															
SP4: sensor de posición 4 (botella en posición de taponado).															
SP5: sensor de posición 5 (tapa lista para ser llevada del alimentador de tapas al punto x (en la trayectoria del taponador).															
LICf: limite inferior del cilindro auxiliar.															
LIT: limite inferior del taponador.															
STM2: sensor de temperatura del motor 2 (banda 2).															
SVC+ Y SVC-: sacar y guardar, respectivamente, el vástago del taponador.															
SVD+ Y SVD-: sacar y guardar, respectivamente, el vástago del cilindro auxiliar.															

Fuente: trabajo de grado “REDES DE COMUNICACIÓN INDUSTRIAL. PRACTICAS DE LABORATORIO”.

- El taponado de una botella exige que el sistema taponado de botellas este funcionando, que el alimentador de tapas este cargado, que el sensor de posición 5 indique que hay una tapa lista para ser llevada al punto x (en el centro de la elipse roja de la grafica 2), de modo que si el embolo del cilindro auxiliar está en su posición inicial (límite inferior) y el del taponador también, se active la válvula solenoide D+ que lleve la tapa al punto x, donde será detectada por el fotodetector, que a su vez desactivara la válvula solenoide D+

y activara la válvula solenoide D-; cuando el embolo del cilindro haya llegado al límite inferior del cilindro auxiliar se energizara la válvula solenoide C+ cuyo efecto es que el taponador lleve la tapa del punto x al sitio de taponado (en el centro de la elipse amarilla de la grafica 2). La válvula solenoide C+ será desactivada por el sensor de presión mientras se activa la válvula solenoide C- ; la válvula solenoide C- se desactiva cuando el embolo del taponador llegue nuevamente a su posición inicial (límite inferior), poniendo fin al taponado de una botella (condición para que arranque nuevamente el motor de la banda 2).

- El comportamiento de arranque después de una parada normal o de emergencia será el mismo que el descrito para el sistema llenado de tanques.
7. Cree una red modbus TCP/IP conectando al switch el PLC M340, la HMI Magelis y el PLC Twido que maneja el sistema de envasado de producto. El PLC Twido que maneja el sistema taponado de botellas conéctelo al PLC M340 a través de una red modbus sobre línea serial.
 8. En el PLC Twido al que piensa transferirle el programa del sistema de envasado de producto real, implemente una red AS-i y una red modbus sobre línea serial iguales a las que implemento en la práctica de modbus sobre línea serial.
 9. En el PLC Twido al que piensa transferirle el programa del sistema de taponado de botella real, implemente una red AS-i y una red CANopen iguales a las que implemento en la práctica de CANopen.
 10. Instale los pulsadores o interruptores de las regletas a las entradas de los tres PLC.

11. Transfiera los 4 programas (3 de los PLCs y el de la HMI Magelis), vía cable USB la primera vez si no tienen asignada una dirección IP (compruebe haciéndoles “ping” con el procedimiento descrito en el ítem “recomendaciones” de la práctica de modbus TCP/IP) y vía ethernet si ya la tienen o a partir de la segunda vez en adelante, si no la tenían.
12. Active la entrada Run/Stop de los 3 PLC.
13. A través de la HMI Magelis y de las regletas de pulsadores o interruptores, verifique que cada uno de los programas descritos en los pasos 4 a 6 cumplan con dichos requisitos y que la activación y desactivación de las entradas de los PLC mediante las regletas, así como la activación de los pulsadores de la HMI causan los cambios descritos en las notas de esta práctica en los paneles 2, 3 y 4 de la HMI Magelis.