

Sistemas de Control Automático

Práctica 3. Control de un PLC mediante tramas Host-Link generadas por un PC



Jorge Pomares Baeza

Carlos Alberto Jara Bravo

Grupo de **Innovación Educativa en Automática**



Universitat d'Alacant
Universidad de Alicante

Introducción

En esta práctica se pretende controlar un autómata OMRON mediante tramas del protocolo industrial Host-Link generadas por un PC. Este protocolo es utilizado por esta marca comercial de autómatas y normalmente se transfiere bajo el medio físico soportado por el puerto serie (RS-232). Previamente, se programará una rutina dentro del autómata mediante lenguaje de contactos para el control del funcionamiento de un semáforo. El PC se encargará de establecer una comunicación tipo *maestro-esclavo* con el autómata y proporcionará una interfaz para controlar diversas funcionalidades del dispositivo, tales como activar y desactivar el autómata, y escritura y lectura de áreas de memoria del mismo.

Objetivos

- Dar a conocer el autómata como sistema de control.
- Describir conceptos fundamentales para la programación de autómatas en lenguaje de contactos.
- Dar a conocer el protocolo Host-Link para la comunicación en una red industrial.
- Dar a conocer la interfaz de programación C#.

1. Conceptos básicos de control por PLC

1.1 Sistemas de control industrial

Se podría definir **control** como la manipulación indirecta de las magnitudes de un sistema llamado planta a través de un **sistema de control** (Figura 1).

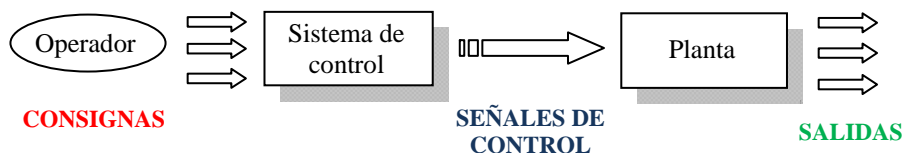


Figura 1. Esquema general de un sistema de control

El objetivo de un sistema de control es el de gobernar la respuesta de una planta sin que el operador intervenga directamente sobre sus elementos de salida. El operador manipula únicamente las magnitudes de consigna y el sistema de control se encarga de gobernar dicha salida a través de los **accionamientos**. Este planteamiento supone que el sistema de control opera con unas magnitudes de baja potencia, llamadas normalmente **señales** y gobierna unos accionamientos que son los que realmente modulan la potencia entregada a la planta (Figuras 2 y 3).

Como se ha comentado en teoría, existen dos tipos de topología dentro de los sistemas de control:

- Sistemas en *lazo abierto*: el sistema de control no recibe ningún tipo de información del comportamiento de la planta (Figura 2).
- Sistemas en *lazo cerrado*: existe una realimentación a través de los sensores desde la planta hacia el sistema de control, que le permite conocer si las acciones ordenadas a los actuadores se han realizado correctamente sobre la planta. Para ello, se requiere la existencia de unos **sensores** que detecten el

comportamiento de dicha planta y de unas **interfaces** para adaptar las señales de los sensores a las entradas del sistema de control (Figura 3).

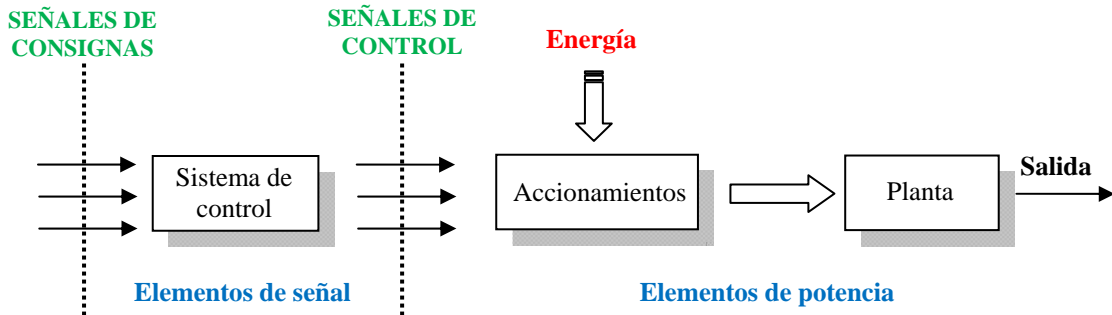


Figura 2. Sistema de control en la configuración de lazo abierto

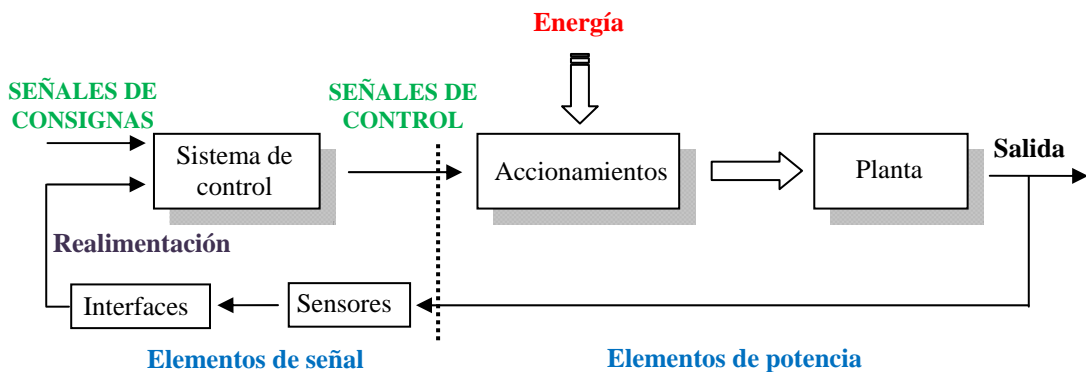


Figura 3. Sistema de control en la configuración de lazo cerrado

1.2 Definición y concepto del autómat

El **autómata** es un sistema de control programable que incluye total o parcialmente las interfaces con las señales de la planta (Figura 3). Además, físicamente se trata de un hardware estándar, con capacidad de conexión directa a las señales de la planta (niveles de tensión e intensidad industriales, transductores y periféricos electrónicos) y fácilmente programable por el usuario. De forma general, se puede definir el **autómata** o **PLC** (Programming Logic Controller) como un equipo electrónico, basado en un microprocesador o micro-controlador, que tiene generalmente una configuración modular, que puede programarse en lenguaje no informático y que está diseñado para controlar procesos en tiempo real y en ambientes hostiles (ambiente industrial).

El esquema de un proceso controlado por una autómat está compuesto por el proceso en el cuál se miden una serie de variables mediante unos sensores que se introducen como entradas (digitales o analógicas) al autómat. El autómat, en función de su programa de control, determina las acciones a tomar sobre sus salidas (digitales o analógicas) que se transmite al proceso o a la planta a través de los actuadores (Figura 4).

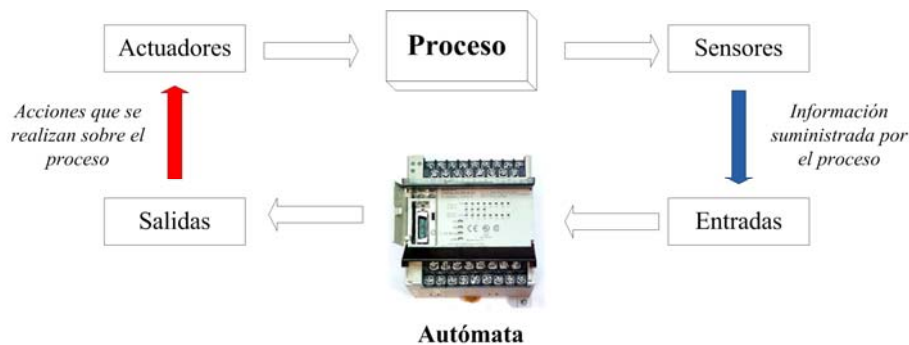


Figura 4. El autómata como sistema de control

1.2 Bloques funcionales del autómata programable

Como se puede observar en la Figura 5, un autómata se compone fundamentalmente de tres bloques principales:

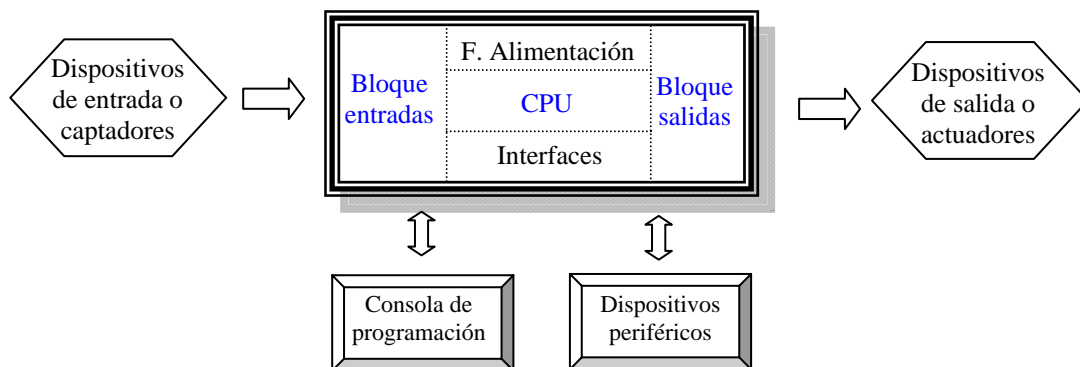


Figura 5. Bloques funcionales del autómata

- **Bloque de entradas:** adapta y codifica para la CPU las señales procedentes de los dispositivos de entrada o captadores, como por ejemplo, pulsadores, finales de carrera, sensores, etc.
- **Bloque de salidas:** decodifica las señales procedentes de la CPU, las amplifica y las envía a los dispositivos de salida o actuadores, como relés, contactores, arrancadores, electroválvulas, etc.
- **Unidad central de proceso (CPU):** este bloque es el cerebro del autómata, ya que su función es la interpretación de las instrucciones del programa de usuario y en función de las entradas, activa las salidas deseadas.

Además de los bloques principales descritos, existen otros bloques necesarios para el funcionamiento del autómata:

- **Fuente de alimentación:** a partir de una tensión exterior, la fuente de alimentación proporciona las tensiones necesarias para el funcionamiento de los distintos circuitos electrónicos del autómata.
- **Consola de programación:** la consola de programación es la que permite cargar el programa de control del autómata. Principalmente, existen dos formas de cargar el programa de control: mediante un PC el cual tiene un software de edición y carga del programa del autómata a través del puerto serie, o bien mediante las consolas de programación que se componen de un pequeño display y un teclado que permiten editar y cargar el programa de control.

- Periféricos: elementos auxiliares que se unen al mismo para realizar una función específica y que amplían su campo de aplicación.
- Interfaces: circuitos o dispositivos electrónicos que permiten la conexión a la CPU de los elementos periféricos descritos.

Un concepto importante que cabe destacar de un autómatas es su **modo de funcionamiento**. Todos los PLC poseen dos modos de funcionamiento:

- Modo programación (**PROGRAM**): permite programar y transferir el programa al autómatas.
- Modo ejecución (**RUN**): el autómatas ejecuta el programa de forma continua leyendo y escribiendo los datos mediante lo que se denomina el **ciclo de programa** (Figura 6).

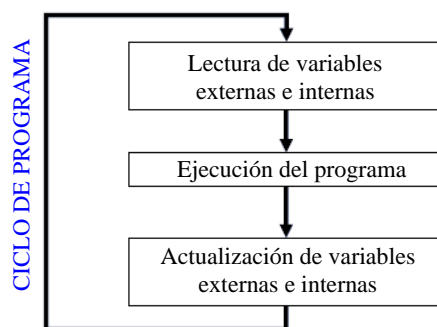


Figura 6. Ciclo de programa de un PLC

2. Programación en lenguaje de contactos de un PLC

Los PLC presentan diversas formas de codificación de las instrucciones, por lo que poseen diversos tipos de lenguajes de programación. Entre los más importantes cabe destacar los lenguajes gráficos como son el diagrama de contactos y el *grafcet*.

El *grafcet* es un diagrama funcional que describe la evolución de un proceso. En él se indican las acciones que hay que realizar y la información generada. En muchas ocasiones se utiliza como punto de partida para la posterior realización del diagrama de contactos. En la siguiente imagen se muestra una secuencia de programa realizada en un diagrama *grafcet*.

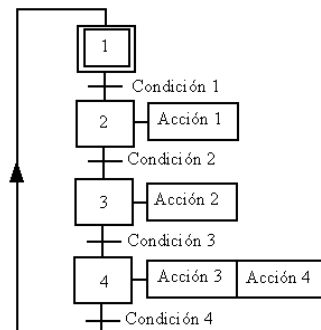


Figura 7. Secuencia de un programa en modo *grafcet*

Por otro lado, el diagrama de contactos es un lenguaje gráfico procedente del lenguaje de relés basado en símbolos que representan contactos, bloques funcionales, etc. que codifican la secuencia de control. Para las funciones lógicas más complejas se utilizan bloques de programación como temporizadores, contadores y registros de desplazamiento. En la Figura 8 se puede observar un ejemplo comentado de un programa en lenguaje de contactos.

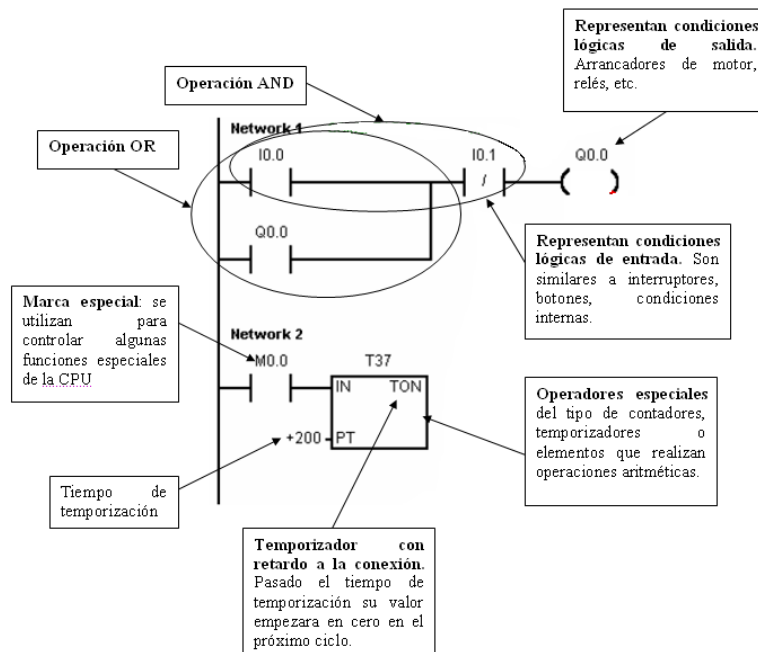
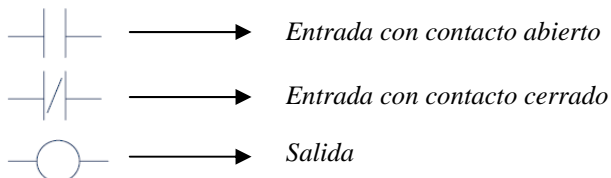


Figura 8. Ejemplo de programación en lenguaje de contactos

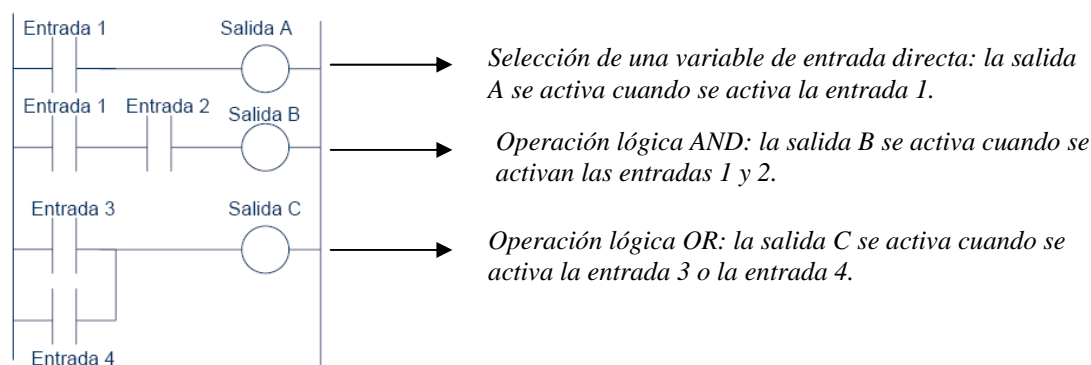
2.1 Instrucciones del lenguaje esquema de contactos

La representación del lenguaje de programación gráfico esquema de contactos o también llamado KOP (Kontakte Programm), es similar a la de los esquemas de circuitos. Los elementos de un esquema de circuitos, tales como los contactos normalmente cerrados y/o normalmente abiertos, se agrupan en **segmentos de programa**. Uno o varios segmentos constituyen las instrucciones de un programa de control. A continuación se muestran los símbolos y operaciones utilizadas con este lenguaje:

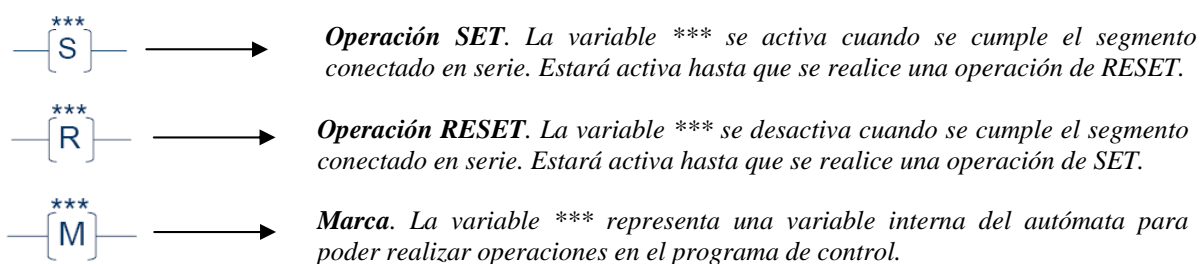
- Símbolos básicos.



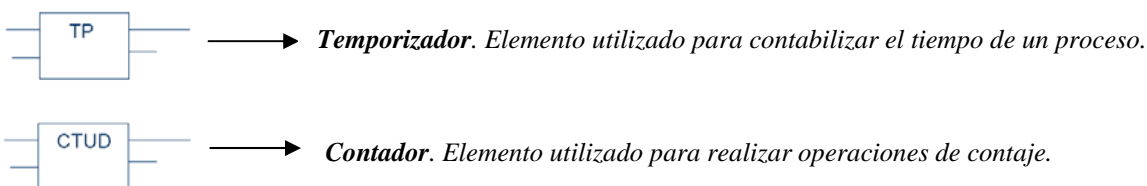
- Operaciones lógicas. Los elementos o símbolos se pueden agrupar para realizar operaciones de AND o de OR dentro del mismo segmento o instrucción.



Operaciones de memoria.



Bloques funcionales.



2.2 Configuración del autómata CQM1 en el entorno de programación Cx-Programmer

Para esta práctica, se va a utilizar el autómata CQM1 de la marca OMRON. Para introducir el programa de control se va a usar el software *Cx-Programmer*. A continuación se enumeran los pasos para la configuración del PLC dentro de este entorno de programación:

- **Inicialización de un nuevo dispositivo:** ejecuta el software y selecciona como tipo de dispositivo el CQM1 y como CPU la 41. El tipo de red es el SIMAC WAY, que es específico para un solo PLC (Figura 9).

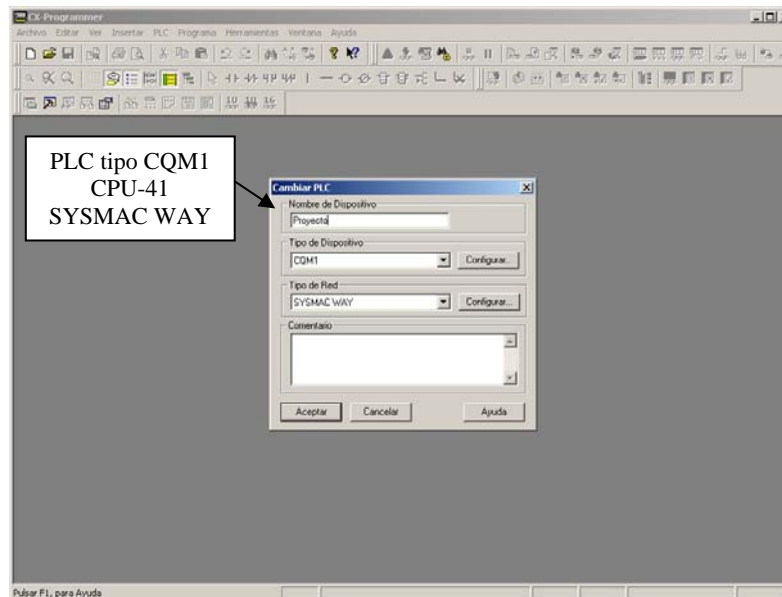


Figura 9. Pantalla inicial del entorno de programación

- **Determinación de las variables del área de trabajo del proyecto.** En este paso se realiza la configuración de las variables que se van a utilizar en el lenguaje de contactos, la tabla de entradas, las salidas, etc. El área de proyecto que existe dentro del *Cx-Programmer* tiene las siguientes opciones:

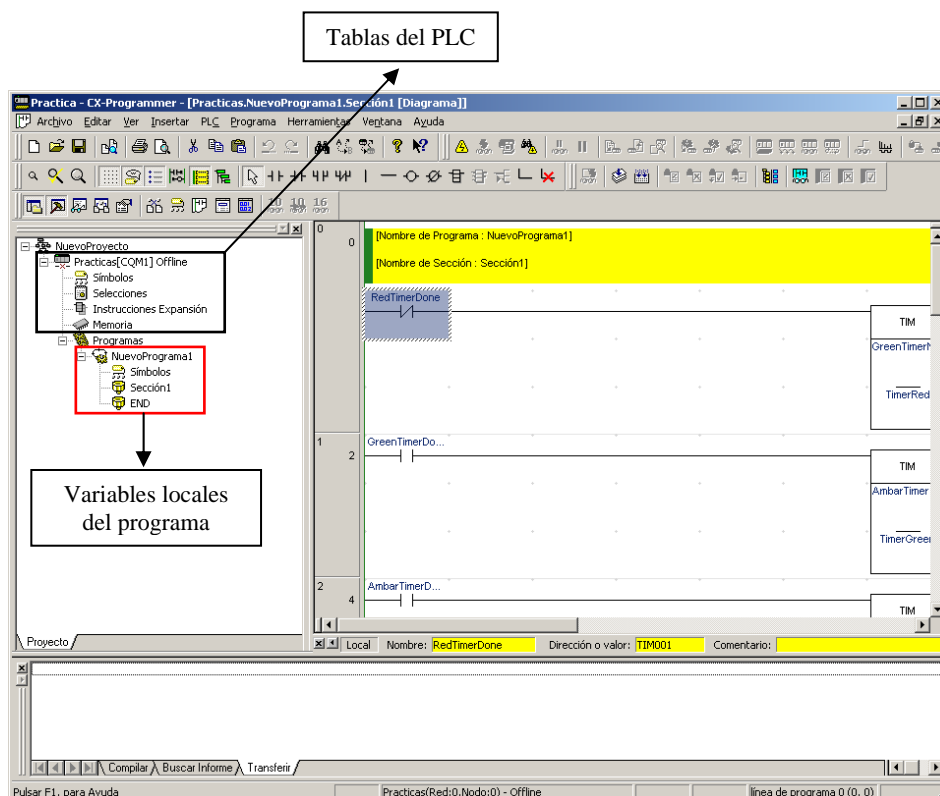


Figura 10. Entorno de trabajo del Cx-Programmer

- ✓ **Tabla de símbolos:** Contiene los símbolos definidos para un determinado PLC o programa. Estos símbolos pueden ser locales o globales. Los símbolos locales son característicos de un solo tipo de programa del PLC mientras que los globales son útiles para distintos programas. Los símbolos globales están definidos en la siguiente tabla que se muestra a continuación:

Dirección de memoria de la variable

Nombre	Tipo de datos	Dirección / Valor	Ubicación de...	Uso	Comentario
P_0_02s	BOOL	254.01	TIME#0		Tiempo Luz Verde
P_0_1s	BOOL	255.00			Bit de pulso de reloj de 0,02 segund
P_0_2s	BOOL	255.01			Bit de pulso de reloj de 0,2 segund
P_1min	BOOL	254.00			Bit de pulso de reloj de 1 minuto
P_1s	BOOL	255.02			Bit de pulso de reloj de 1,0 segund
P_CY	BOOL	255.04			Indicador de acarreo (CY)
P_Cycle_Time_Error	BOOL	253.09			Indicador de error de tiempo de ci
P_Cycle_Time_Value	UINT_BCD	AR27			Tiempo de exploración actual
P_EQ	BOOL	255.06			Indicador de igual que (EQ)
P_ER	BOOL	255.03			Indicador de error de ejecución de
P_First_Cycle	BOOL	253.15			Indicador de primer ciclo
P_GT	BOOL	255.05			Indicador de mayor que (GT)
P_Hour_Date	UINT_BCD	AR19			Horas (00-07) y fecha (08-15)
P_Low_Battery	BOOL	253.08			Indicador de batería baja
P_LT	BOOL	255.07			Indicador de menor que (LT)
P_Max_Cycle_Time	UINT_BCD	AR26			Tiempo de ciclo máximo
P_Month_Year	UINT_BCD	AR20			Mes (00-07) y año (08-15)
P_OFF	BOOL	254.04			Indicador de desbordamiento (OF)
P_Off	BOOL	253.14			Indicador de siempre OFF
P_On	BOOL	253.13			Indicador de siempre ON
P_Output_Off_Bit	BOOL	252.15			Bit de salida OFF
P_Sec_Min	UINT_BCD	AR18			Segundos (00-07) y minutos (08-1
P_Step	BOOL	254.07			Indicador de paso
P_UF	BOOL	254.05			Indicador de subdesbordamiento (

Comentarios

Nombre del símbolo

Tipo de dato: se utiliza el tipo **BOOL** para direcciones de bit y el tipo **CHANNEL** para otro tipo de direcciones

Figura 11. Tabla de símbolos del PLC

- ✓ **Selecciones:** permiten incluir el nombre físico del PLC, detalles de su configuración, y tipo de red de comunicaciones para su conexión con el ordenador. Estas especificaciones definen la relación del PLC dentro de la red .La pantalla de selección se muestra en la Figura 12.

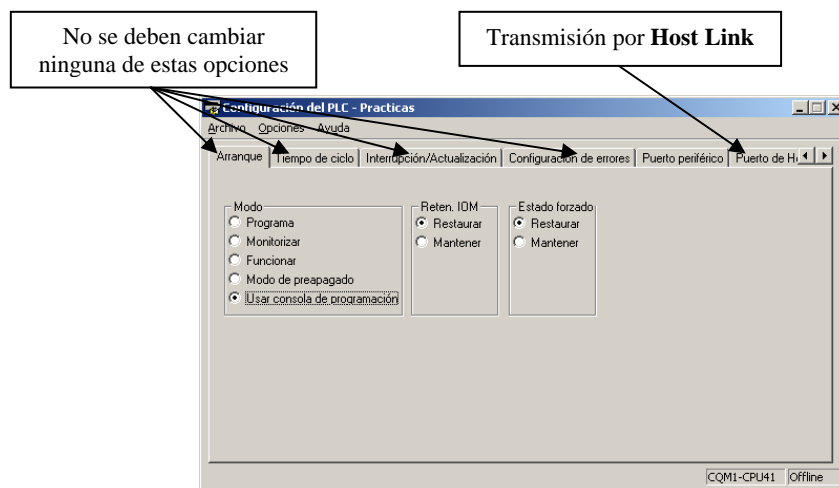


Figura 12. Opciones de la tabla de Selecciones

- ✓ **Memoria:** esta opción permite al alumno mostrar valores de la memoria del PLC en formato tabular (similar a una hoja de cálculo) de direcciones individuales de memoria o en un rango completo de un área de memoria. Los datos en cada celda de la tabla muestran como el programa ejecuta las diferentes condiciones. La memoria debe ser configurada mediante la interfaz que se muestra en la Figura 13.

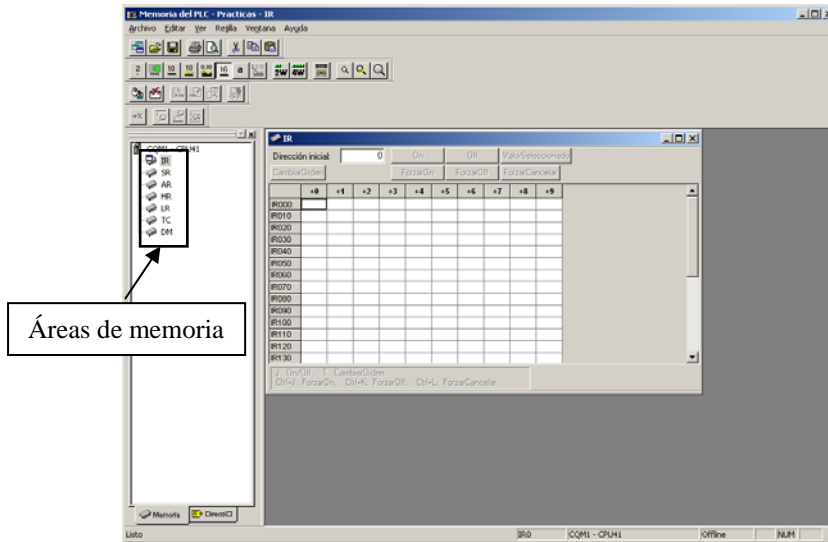


Figura 13. Áreas de memoria del PLC

2.3 Implementación del programa de control el software Cx-Programmer

Para poder introducir el programa control en lenguaje de contactos, se debe configurar las siguientes opciones del software:

- **Configuración de la tabla de símbolos.** Se debe pulsar sobre la pestaña símbolos situada en el interior del desplegable *Programas*. En el siguiente paso, se debe situarse sobre alguna de las casillas y pulsar el botón derecho, para insertar cada una de las variables del programa.

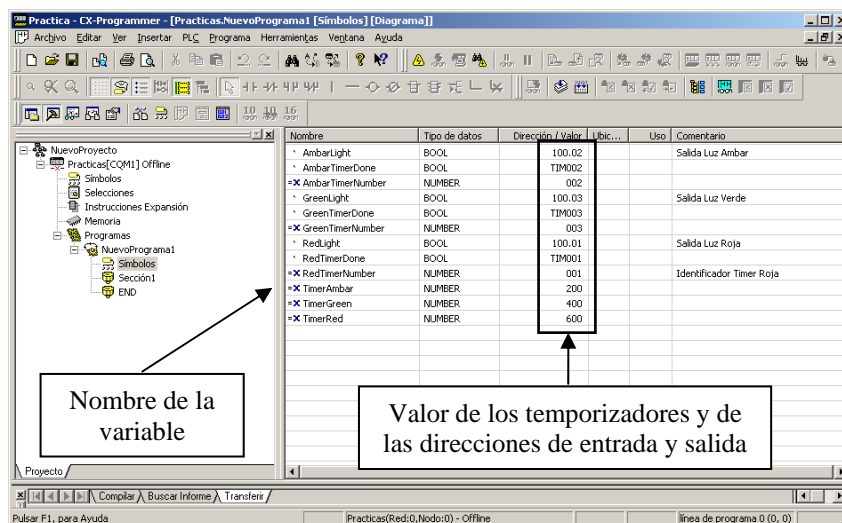


Figura 14. Tabla de símbolos I

Los tipos de datos con los que se pueden trabajar en esta aplicación pueden ser BOOL y NUMBER. Los del tipo BOOL están destinados para las entradas y las salidas, mientras que los NUMBER son para los temporizadores. Estos elementos utilizan las direcciones TIM004, TIM001, TIM002, TIM003.

- **Traducción del programa en lenguaje de contactos y su compilación.** En este segundo paso, se debe traducir el programa en lenguaje de contactos al software de programación, según lo establecido en la Figura 15.

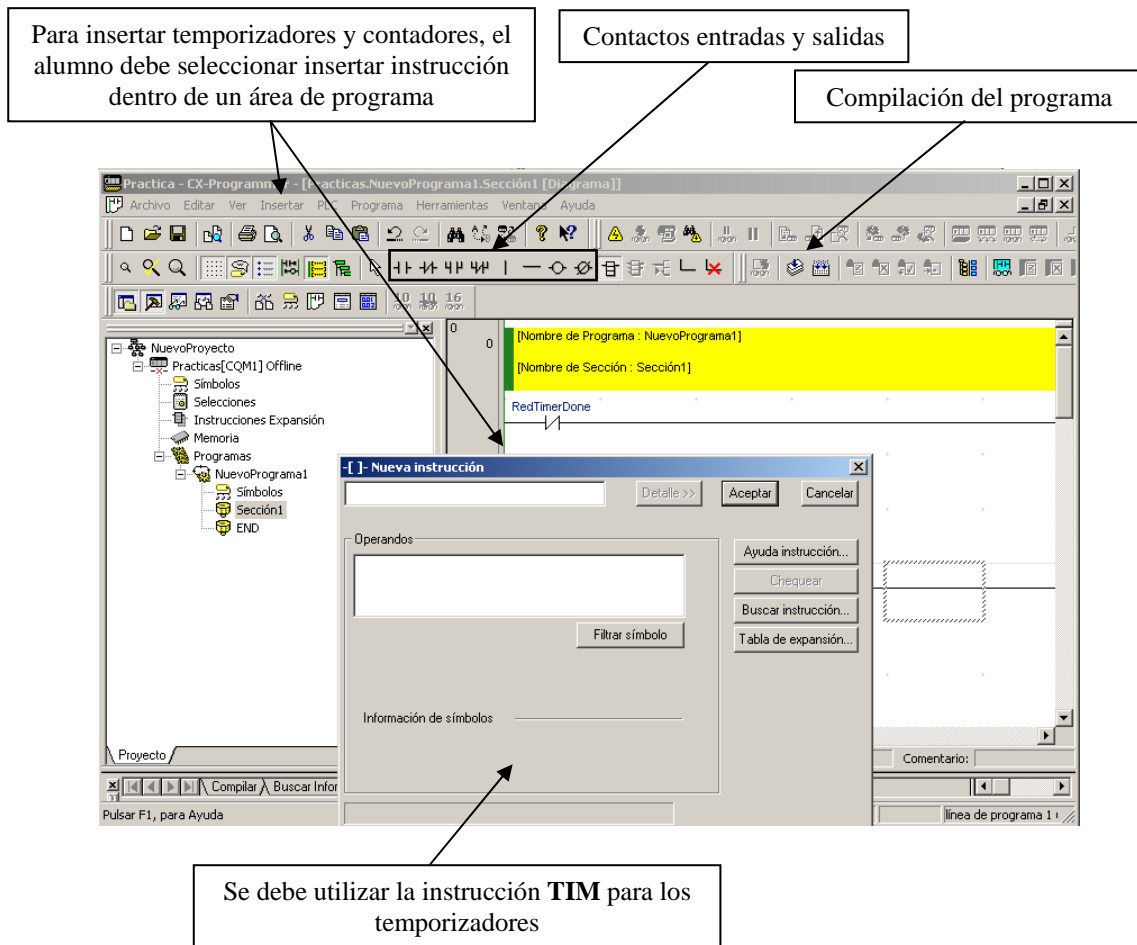


Figura 15. Tabla de símbolos II

- **Transferencia al PLC.** Después de realizar la implementación por software del programa en lenguaje de contactos, se debe configurar la comunicación con el PLC según lo establecido en lo que se muestra en la Figura 16.

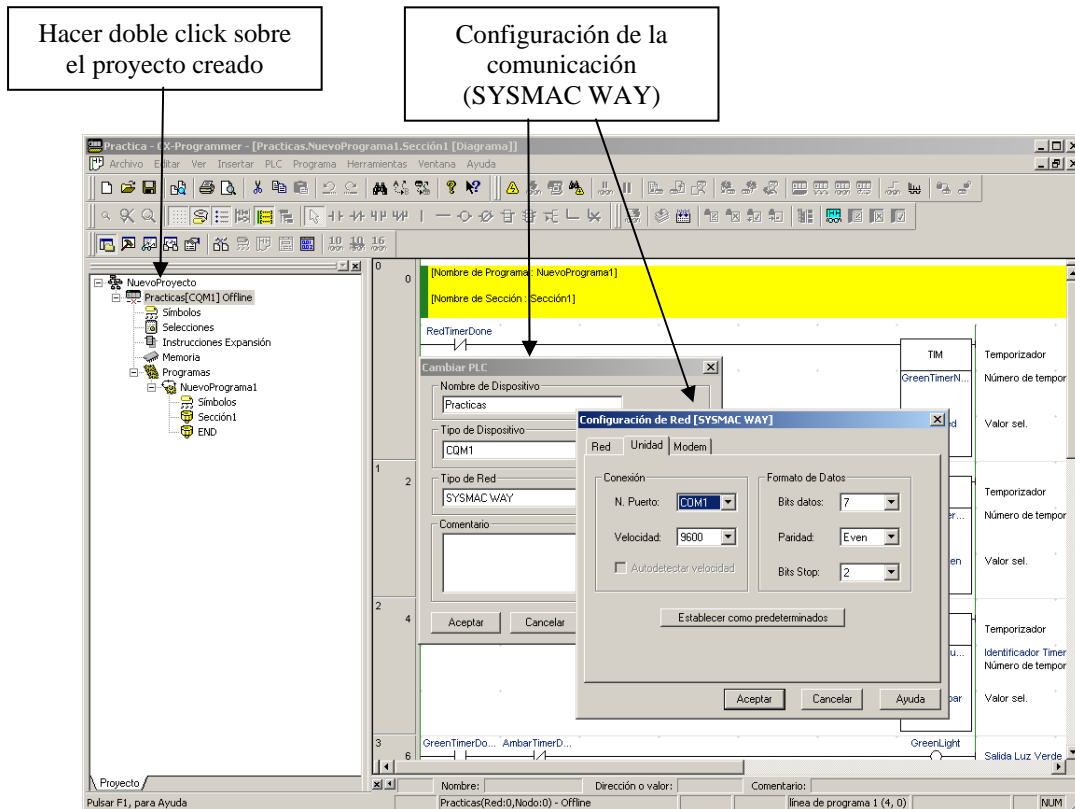


Figura 16. Configuración de las comunicaciones PC-PLC

Una vez realizada la configuración de la comunicación entre PC-PLC mediante el software de programación, solo queda establecer la conexión física mediante el cable que se puede observar en la siguiente imagen:

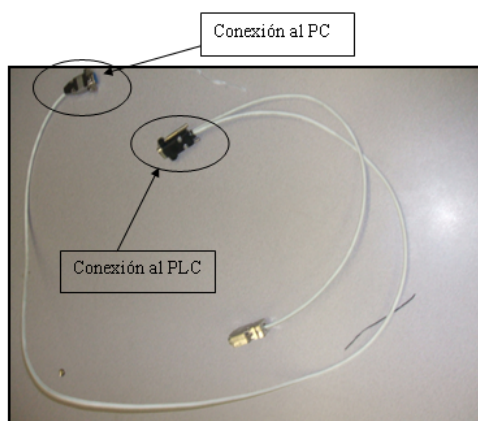


Figura 17. Cable de comunicaciones PC-PLC

Después de haber realizado la conexión con el PLC mediante las instrucciones descritas en la Figura 16, se puede observar que existen 3 formas de trabajo para el PLC:

- **Modo RUN.** Permite que el PLC ejecute el programa o programas y no permite escribir/forzar áreas de memoria desde un ordenador. Existe la posibilidad de monitorizar las áreas de memoria del PLC.
- **Modo Monitorización.** En este modo de trabajo se ejecutan los programas del PLC y las operaciones de E/S están activas. A diferencia del caso anterior el programador puede escribir en la memoria del PLC desde un ordenador.
- **Modo Stop/Programa.** Se utiliza para la transferencia de programas al PLC, durante este estado no se puede ejecutar ningún programa. Por el contrario existe la posibilidad de establecer bits (instrucciones de bobina/ contacto) en ON/OFF y forzar ON/OFF.

En la Figura 18 se muestran las explicaciones para poder realizar la transferencia del programa de control al PLC.

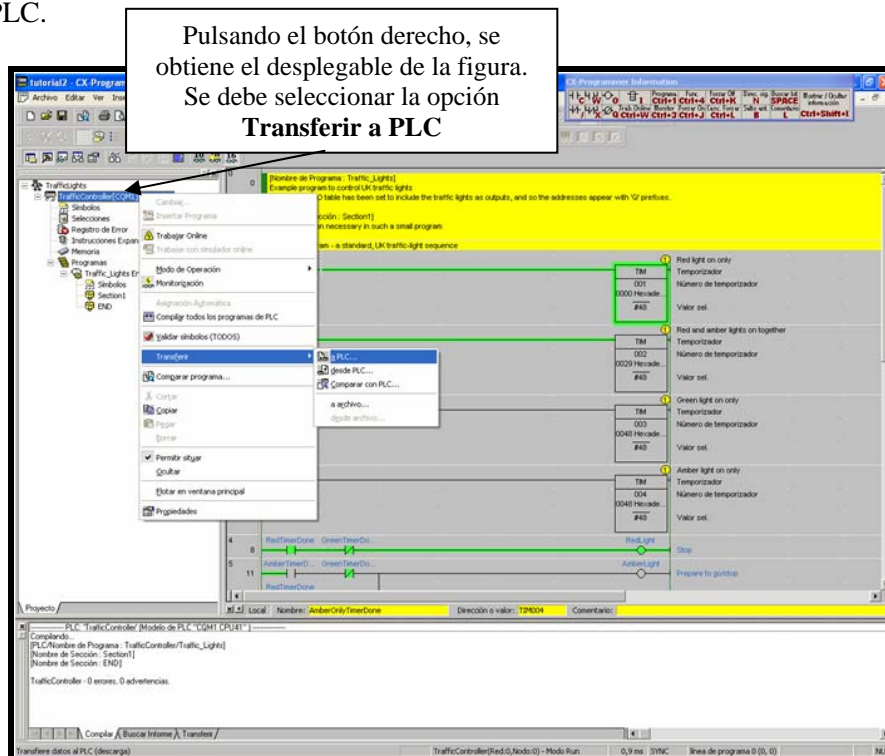


Figura 18. Opciones para realizar la transferencia del programa al PLC

Tras realizar la selección de la opción transferir a PLC, se puede seleccionar qué partes del programa se deben transferir. En esta práctica, se debe tener activados las opciones de **Programa**, **Funciones de expansión** y **Selecciones**. Estas características se pueden observar en la Figura 19.

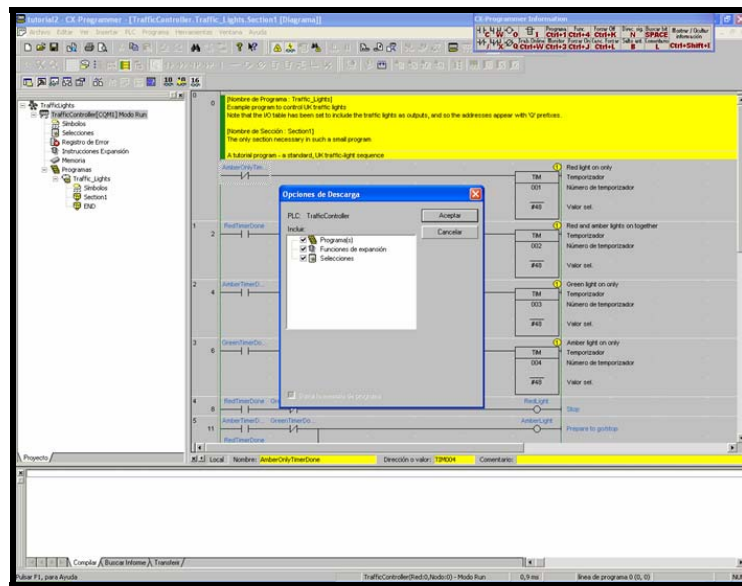


Figura 19. Opciones a transferir al PLC

3. Descripción del protocolo Host-Link

Los autómatas de la serie OMRON están preparados para la comunicación con otros periféricos mediante el uso de puertos RS-232. Para esta comunicación tienen implementado un protocolo llamado Host Link que permite el envío de comandos de programa.

El protocolo Host-Link introduce en una trama comandos modo-C que pueden ser enviados desde un *host* para leer o escribir áreas de memoria de E/S de los PLCs ó para controlar los modos de operación de los PLCs. El formato de una trama general Host Link se puede ver en la Figura 20. Cada uno de los caracteres que forman una trama ocupa 1 byte ya que son enviados en código ASCII. El formato sigue el siguiente patrón:

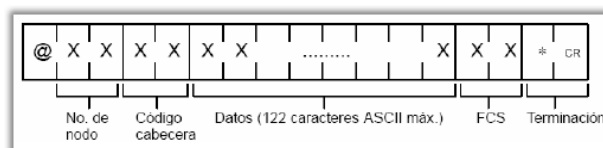


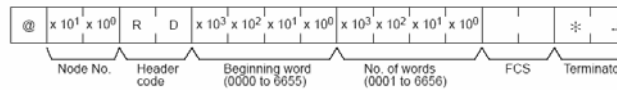
Figura 20. Formato de la trama Host-Link

El número de nodo es el código identificador del nodo receptor dentro de la red RS-422 en la que se establece la comunicación, ocupa 2 bytes que representa cada uno un carácter ASCII correspondientes a un número decimal (para nuestro caso este campo valdrá **00**). El código de cabecera vuelven a ser 2 bytes, que representan el código que identifica el comando Host Link que queremos ejecutar. En las Figuras 21 y 22, se pueden ver los códigos de cabecera para los comandos de lectura y escritura dentro de una sección de memoria DM del autómata, éstos son RD y WD respectivamente. Los datos de la trama dependen directamente del comando que se envíe así como de si es un envío o una respuesta. Por ultimo, se tiene el FCS y el código de terminación. El FCS no es más que un código de detección de errores basado en hacer una XOR con los bytes pares e impares de la trama enviada y mandar el resultado en este campo formado por dos bytes. Por último, el código de terminación siempre está formado por un asterisco seguido de un carácter de retorno de carro.

6-5-6 DM AREA READ — RD

Reads the contents of the specified number of DM words, starting from the specified word.

Command Format



Response Format

An end code of 00 indicates normal completion.

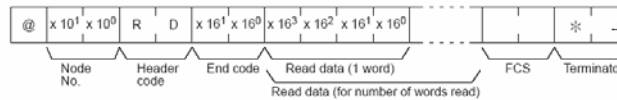
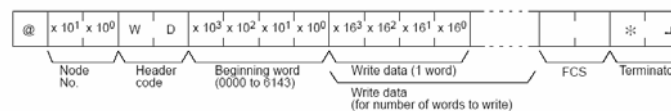


Figura 21. Descripción del comando RD para leer una serie de palabras en el área DM de memoria del PLC

6-5-14 DM AREA WRITE — WD

Writes data to the DM area, starting from the specified word. Writing is done word by word.

Command Format



Note Divide the command when writing more than 29 words of data.

Response Format

An end code of 00 indicates normal completion.

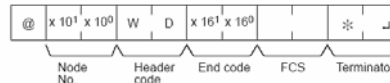


Figura 22. Descripción del comando WD para escribir una serie de palabras en el área DM de memoria del PLC

Como se puede observar en las Figuras, las palabras donde queremos escribir o leer han de expresarse en decimal y además, dado que estas sólo pueden ocupar 4 bytes, esto limita a que sólo se puede escribir entre las palabras de memoria 0000 y 9999. Es más, el protocolo Host Link está limitado a trabajar como mucho hasta la posición 6143 en el caso de la escritura y en la 6656 en la lectura, esto se puede ampliar con el uso de comandos FINS encapsulados dentro del protocolo Host Link (véase trabajos optativos).

Especificaciones para Host Link		Descripción
Modo de comunicación	Half-duplex (Full-duplex para inicializar comunicaciones con esclavos)	
Modo de sincronización	Sincronización Start-stop (modo asincrónico)	
Velocidad	Puertos RS-232C y RS-422/485: 1200/2400/4800/9600/19200/38400/57600/115200 bps Configuración por defecto: 9600	
Distancia de comunicaciones	Puerto RS-232C: 15 m. máx. Puerto RS-422/485: 500 m. máx.	
Configuración de conexiones	Puerto RS-232C: 1:1 (1:N (N=32 nodos máx.) es posible usando un convertidor) Puerto RS-422/485: 1:N (N=32 nodos máx.)	
Número de unidades conectadas	32 unidades máx. (números de unidades de '0' a '31'; para conexiones 1:1 en el número de unidad será '0')	
Estructura de la trama	Comandos modo-C	<Cabecera><Dirección><Datos><CRC><Terminador> - Cabecera: @ - Dirección: número de unidad Host Link de 0 a 31 (BCD) - Datos: comando Host Link + texto - Código de chequeo de errores (CRC): FCS - Terminador: '*' + <CR>
	Comando FINS	<Cabecera><Dirección><Datos><CRC><Terminador> - Cabecera: @ - Dirección: número de unidad Host Link de 0 a 31 (BCD) - Datos: código cabecera (siempre 'FA') + cabecera FINS + comando FINS + texto - Código de chequeo de errores (CRC): FCS - Terminador: '*' + <CR>

Figura 23. Especificaciones para el protocolo Host Link I

Código de chequeo de errores	Paridad vertical: par, impar o ninguna. Paridad horizontal: FCS		
Flujo de comandos	Flujo	Comando	Comentario
	Ordenador a PLC	Comando modo-C	Comunicaciones 1:1 ó 1:N con PLCs conectados directamente (La trama de debe de crear en el ordenador)
		Comando FINS (con protocolo Host Link)	Comunicaciones 1:1 ó 1:N con PLCs conectados directamente.
	PLC a ordenador	Comando FINS (con protocolo Host Link)	Comunicaciones usando las instrucciones del PLC SEND(090), RECV(098) y CMND(490). El ordenador de interpretar el comando y devolver una respuesta con el formato correcto. Las conexiones entre el ordenador y el PLC deben de ser 1:1.

Figura 24. Especificaciones para el protocolo Host Link II

Para poder realizar las tareas de programación del protocolo Host Link, se ha dejado como documento adjunto a la práctica, un manual de programación del autómatas CQM1 donde se explica de forma detallada cómo están compuestas las tramas de este protocolo.

4. Lenguaje de programación C#

C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA e ISO. Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma.NET el cual es similar al de Java aunque incluye mejoras derivadas de otros lenguajes (entre ellos Delphi). La creación del nombre del lenguaje, C#, proviene de dibujar dos signos positivos encima de los dos signos positivos de "C++", queriendo dar una imagen de salto evolutivo del mismo modo que ocurrió con el paso de C a C++.

5. Tareas a realizar

- Programación en lenguaje de contactos del funcionamiento de un semáforo (3 sesiones).

Programa de control

El alumno debe realizar un programa en lenguaje de contactos que permite controlar el encendido y apagado de los semáforos en una ciudad.

*El semáforo dispone de tres tonalidades: verde, rojo y ámbar, que tiene asignados una duración de tiempo y regula el paso de vehículos. El objetivo principal es evitar que el peatón sea atropellado. Inicialmente el semáforo se encuentra en rojo (los peatones pueden pasar). Transcurridos 30 segundos el semáforo se pondrá en verde. Si el peatón activa desde el semáforo el **paso de peatones**, el semáforo pasará a ambar (5 segundos) y después a rojo. A partir de este momento dispone de 30 segundos para pasar la calle. Pasados 30 segundos, el semáforo volverá a pasar a color verde. El ciclo se volverá a activar, bien una vez pasado otros 60 segundos, o bien cuando el peatón vuelva a pulsar el paso de peatones.*

NOTA: Las entradas se deben asignar dentro del intervalo 1.00-1.15 mientras que las salidas están en el intervalo 100.00-100.08.

- Programación en C# la generación de tramas Host-Link. Mediante lenguaje C#, el alumno tiene que realizar una serie de funciones para generar las tramas Host-Link y enviarlas a través del puerto serie. Por lo tanto, es necesario crear un proyecto C# con el software Visual Studio que incluya un controlador para el envío y recepción de tramas a través del puerto serie (control *SerialPort*). Este

controlador software se tendrá que configurar de manera similar a la comunicación del PC-PLC (2 sesiones).

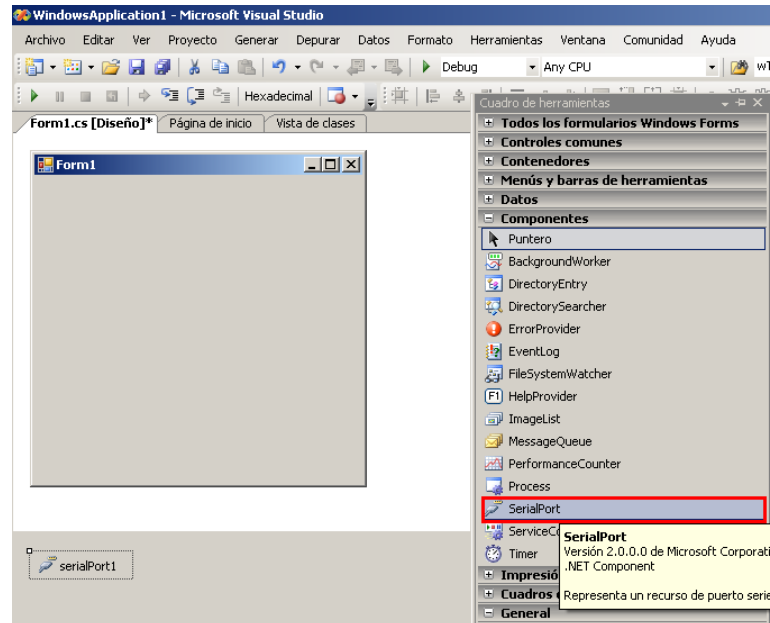


Figura 25. Interfaz SerialPort de C# para establecer la comunicación con el PLC

La generación de las tramas Host-Link consistirá en una serie de funciones que generen una variable tipo *String* que incluya la trama con su respectivo FCS y finalización de bloque. Lo más conveniente es definir una serie de constantes que faciliten la creación de las tramas:

```
/**BLOQUES INIT/FIN**/  
const String Init_Trama = "@00";  
const String End_Trama = "*\r";  
  
/**COMANDOS HOST-LINK**/  
const String Read_IR = "RR"; //Lectura canal IR (0000-0255)  
const String Write_IR = "WR"; //Escritura canal IR (0000-0252)  
const String Abort = "XZ"; //Abortar la comunicación  
. . .
```

Una estructura posible para el algoritmo que genere las tramas es el siguiente:

```
private void Send_Trama(String orden, String datos)  
{  
    String trama = Init_Trama + orden + datos;  
    String FCS = ComputeFCS(trama); //Calcula el FCS  
    trama = trama + FCS.ToUpper() + End_Trama;  
    serialPort.Write(trama); //Se envía a través del puerto serie  
    String lectura = serialPort.ReadExisting();  
}  
  
private String ComputeFCS(String trama)  
{  
    //Generación del FCS  
}
```

La función *Send_Trama* tomaría el valor de la trama y los datos a escribir, calcularía el FCS para incluirlo en la trama y finalmente, la enviaría a través del puerto serie. La función *ComputeFCS* calcularía el código FCS a partir de la trama que se desea enviar.

- Realizar una interfaz usuario con C# para poder controlar el PLC. El alumno debe realizar una interfaz de usuario utilizando diversos controles (botones, campos de texto, etc.) para poder controlar el programa de control del PLC, tal y como se muestra en la Figura 26 (1 sesión).

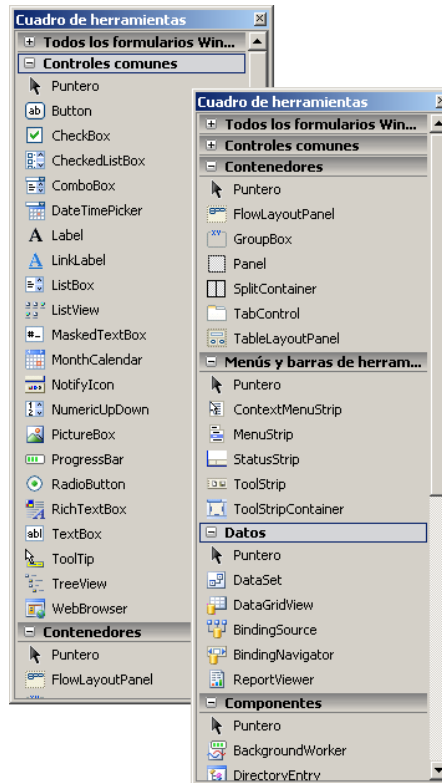


Figura 26. Cuadro de controles de C#

6. Trabajos optativos

Se podrá obtener hasta un máximo de dos puntos adicionales a sumar a la nota final de la asignatura realizando algunos de los siguientes trabajos optativos:

- Desarrollar una librería en lenguaje C# para la generación de tramas Host-Link básicas (modo-C).
- Desarrollar una librería en lenguaje C# para la generación de tramas Host-Link que puedan encapsular comandos FINS para poder controlar el autómatas OMRON de la clase CJ.
- Transcripción del programa realizado en C# a lenguaje de programación de la placa Arduino para poder controlar el PLC con esta placa.

Anexo: Puerto serie mediante las señales de salida de una placa Arduino

Los que deseen realizar el trabajo optativo de la placa Arduino, tendrán que implementar el siguiente circuito electrónico para poder enviar señales a través del puerto serie. La placa Arduino posee dos pines digitales (0-1) que corresponden a las señales de transmisión RX/TX. Sin embargo, las señales de este puerto son con niveles TTL (0-5v), por lo que si se desea realizar un puerto estándar RS-232, cuyos niveles de tensión de los pines están comprendidos entre +15v y -15v, es necesario utilizar un circuito electrónico de acondicionamiento de la señal. En la Figura 27, se muestra el esquema de este circuito, donde el chip *transceiver* MAX-232, adapta los niveles RS232 y TTL.

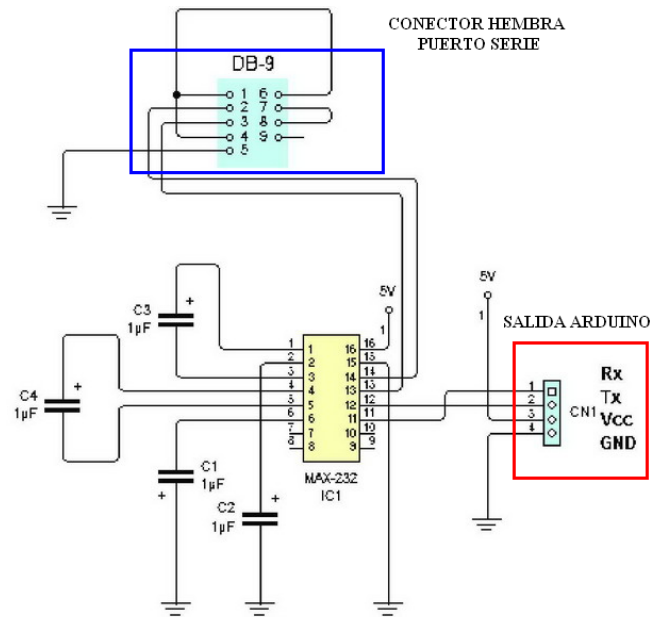


Figura 27. Transformación de señales TTL-RS232.