UNIVERSIDAD NACIONAL AUTÓNOMA

DE MÉXICO

FACULTAD DE INGENIERÍA

DESARROLLO DE UN AMBIENTE DE PROGRAMACIÓN DE TIPO DE DIAGRAMA DE ESCALERA PARA EL PROGRAMADOR LÓGICO MODULAR

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A N:

DANIEL DEL ANGEL GARCÍA

ALEJANDRO MILLÁN GARCÍA

DIRECTOR: M. EN I. ANTONIO SALVA CALLEJA

MEXICO, DF. 2008

AGRADECIMIENTOS

(Daniel Del Angel García)

Agradezco a mi esposa, por todo su amor, apoyo, paciencia pero sobre todo su confianza en mi durante toda mi carrera, siempre has sido ejemplo y motivación para seguir adelante sobre todo en los tiempos difíciles, siempre estuviste ahí ayudándome en cada proyecto que hacía, siempre estuviste conmigo para celebrar cuando me iba bien ó para levantarme cuando no, has sido siempre mi mejor amiga, Gracias por todo TE AMO

A mis chamacas Kyara y Rebeca por enseñarme que lo único que te hace realmente feliz no es el dinero, sino estar con bien con tu familia. Por enseñarme a perdonar y por creer que soy un ser sumamente especial.

A mis Padres por todo el apoyo que me han dado y por los sacrificios que hicieron para que hoy en día pueda estar culminando este proyecto, gracias por confiar en mí, pero sobre todo gracias por todo el amor que me han dado.

A mis hermanos (incluidos Pepe y Paty), sobrinos, amigos, y compañeros de trabajo, que con su cariño y buenos deseos siempre me han alentado a seguir adelante y a ser una mejor persona. No los menciono a todos pero no olvido a ninguno.

A Alejandro Millán le agradezco no solo el haber sido un excelente compañero de tesis, sino un gran amigo que me apoyo en todo momento, Valió la pena el esfuerzo.

Al M. en I. Antonio Salva Calleja. Por su paciencia, interés, disposición y sus sabios consejos, de corazón lo digo, no pude haber tenido mejor director de Tesis, de vedad muchas Gracias.

A la Lic. Ivonne Ramírez, por su amistad, su gran apoyo y confianza en el trabajo, su constante esfuerzo en todo lo que hace, ha sido ejemplo y motivación para culminar mi tesis. Gracias.

Al Doctor Isidro Ávila Martínez, por su confianza y apoyo, usted ha sido para mí un gran ejemplo de trabajo duro, honestidad y compromiso. Gracias Doctor.

A la UNAM y a la Facultad de Ingeniería, porque en sus aulas aprendí mucho más que circuitos, ecuaciones y a programar; aprendí que el verdadero valor del conocimiento no radica en la cantidad de dinero que obtenga de él, sino en el poder usarlo para ayudar a la gente, a mi país y al mundo en general. Agradezco infinitamente todo lo aprendido y me lleno de orgullo en saber que estudie en la mejor universidad de mi país, pero al mismo tiempo me queda un compromiso de honrar y retribuirle a la universidad y a México todo lo que me han dado.

Dedico esta tesis a mis Amados mamá y papá. A mis queridas hermanas. Para ustedes con todo mi cariño.

AGRADECIMIENTOS

(Alejandro Millán García)

A Dios. Por darme la vida y permitirme llegar a este punto, por todas mis capacidades, dones y defectos. Por darme la fortaleza y ecuanimidad en los momentos más difíciles.

A mi mamá. Por ser la mejor madre del mundo. Por demostrarme que con cariño y amor a Dios cualquier cosa es posible. Agradezco tu dedicación y cuidado incondicional.

A mi papá. Por enseñarme que un par de sonrisas son mejores que todo el dinero del planeta. Por mostrarme el camino de la rectitud, honestidad y responsabilidad. Eres el mejor.

A ambos. Por enseñarme los más profundos valores de la vida. Por sus desvelos, regaños y demás obligaciones, ahora entiendo que a partir de ello se forja la mejor educación. Muy afortunado soy al tenerlos como padres, les debo todo lo que soy y lo que he logrado. Muchas Gracias. Los quiero mucho

A mis hermanas, Marisol y Marifer. Por apoyarme en todo momento. Han enriquecido mi existencia con cada experiencia que hemos vivido. Agradezco su compañía y cariño. Espero que este sea un sueño compartido y les sirva como guía para lograr los suyos. Las quiero mucho

A mi Abuelita Tere. Por encomendarme a Dios en los momentos más complicados. Nunca te olvidaremos…

A mi tío Rosalio. Por apoyarme siempre que lo he necesitado. Esta tesis también la dedico a tus Hijos, por lo mucho que te hubieran dado.

A mi tío Pepe. Por enseñarme que la imparcialidad es la mejor arma en cualquier conflicto.

A mi tía Chabe. Por el cariño mostrado a cada uno de los miembros de mi familia.

A mis primos Fer, Oscar, Jaime y Luis. Por compartir los mejores momentos de mi niñez. Por ser fuente de alegrías y travesuras.

A mis amigos, Emmanuel, Hugo, Marisol, Roberto, César, Alin, Cándido, Alejandra, Miguel, Everardo, Israel, Héctor, Brody, José Alberto, Nayeli, Diana, Jaime, Ariadna, Edgar, Rodrigo, Lidia, Daniel, Gaby, José Luis, Telma, Alex, Clau, Maggie, Diego, Dario, Danny,

Beto, George y todas aquellas personas que han dejado huella en mi vida. Agradezco a Dios por haberlos puesto en mi camino.

A Pily Camarillo. Gracias a ti, se que aún puedo encontrar a personas de gran corazón, sinceras y con nobleza inigualable. Gracias por tu amistad.

A Daniel. Mejor amigo y compañero de tesis. Gracias por enseñarme que cualquier cosa es alcanzable y posible. Ambos sabemos que todo el tiempo dedicado a este proyecto ha valido la pena.

Al Ingeniero José de Jesús Ruiz Guillén, mentor en mis Prácticas Profesionales. Gracias por enseñarme que la resolución de problemas no radica en la complejidad de la solución sino en la facilidad de su implementación.

A la Licenciada Ivonne Ramírez. Gracias por enseñarme que la confianza permite establecer lazos de cordialidad, compañerismo, familiaridad y sobre todo de amistad.

A Jorge Fernández. Gracias por creer en mí y en mis capacidades. Gracias por otorgarme una gran oportunidad de trabajo.

A Armando, Pily y George. Amigos de la UANL. Su gestión me ha permitido crecer en muchos sentidos. Gracias por enseñarme que la amistad y la confianza es la mejor herramienta para lograr resultados en conjunto.

Al M.I. Antonio Salvá Calleja. Tutor de tesis. Gracias por creer en nosotros, por otorgarnos este gran proyecto. Gracias por compartir sus conocimientos e impulsar el desarrollo de nuestra formación profesional.

A la Universidad Nacional Autónoma de México, Facultad de Ingeniería. Gracias por estos años de conocimientos, crecimiento y logros. Me enseñaste que en verdad, la grandeza radica en el espíritu.

“Mantén siempre los pies en la Tierra… pero la mirada en las estrellas”

CONTENIDO

ÍNDICE DE IMÁGENES

ÍNDICE DE TABLAS

INTRODUCCIÓN

Los procesos industriales, para fines de su control y/o monitoreo, pueden subdividirse en varios subprocesos individuales que tienen asociados a ellos variables físicas, tales como: temperatura, presión, nivel, etc. Cada subproceso individual es típicamente controlado mediante un sistema de lazo cerrado, analógico o digital.

En la práctica los diversos subprocesos requieren de un arbitraje lógico que regule secuencias de eventos entre ellos, un ejemplo de esta clase de secuencias es una bomba que suministre un reactivo, que ha de combinarse químicamente con otra sustancia en una autoclave. Esta bomba, deberá funcionar sólo por un tiempo determinado y a condición de que el otro reactivo ya se encuentre presente a una determinada temperatura; tanto la temperatura como el nivel requeridos son controladas individualmente por sendos lazos.

Se intuye la necesidad de una instancia de control que podría ser una compuerta lógica AND combinada con un temporizador; las entradas de la compuerta serían variables booleanas cuyo estado testificaría simplemente si la temperatura es adecuada o no y si el nivel de la segunda sustancia en la autoclave es conveniente o no; la salida de la compuerta podría disparar a un sistema de temporización que mantuviera en operación la bomba de suministro del primer reactivo el tiempo requerido.

El arbitraje lógico mencionado en el párrafo anterior es denominado control analógico o secuencial y para llevarlo a cabo se requiere lo siguiente:

1. **Sensores booleanos** de diversas condiciones de proceso, que testifiquen el que las variables asociadas con los diversos subprocesos locales se encuentren o no en un determinado rango.
2. **Sistema lógico**, conformado típicamente por compuertas lógicas, flip-flops, temporizadores, etc.; el cual procesa las señales proporcionadas por los sensores booleanos.
3. **Actuadores lógicos** cuyas entradas son a su vez salidas del sistema lógico mencionado en el párrafo anterior.

Por lo anterior, en la industria es muy frecuente la necesidad de llevar a cabo el control secuencial, de eventos relacionados con bloques funcionales asociados a un determinado proceso productivo, ejemplos de esto pueden apreciarse en la industria automotriz, de alimentos y petroquímica, sólo por mencionar algunas.

Hasta la década de los setentas el sistema lógico asociado, con un sistema de control secuencial, era realizado por elementos físicos fijos, que realizaban las compuertas requeridas de acuerdo con lo que un determinado proceso requiriera en un momento dado. En caso de que hubiera que hacer modificaciones al sistema lógico había que alambrar de nuevo o incluso rehacer completamente el hardware requerido, esto consumía mucho tiempo y dinero. En la figura I.1 se muestra un esquema posible de alambrado para la situación de control lógico mencionada en el primer párrafo.

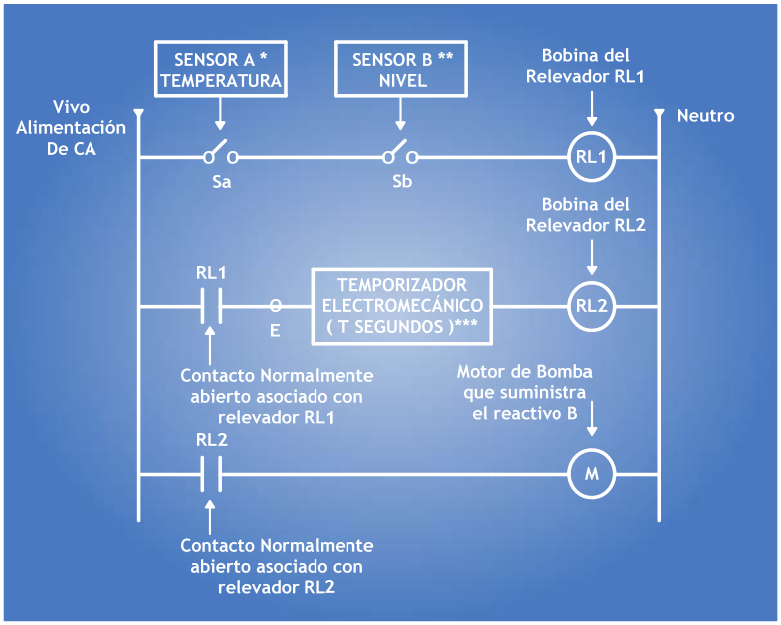


Figura I.1 Esquema de alambrado para el control lógico de un proceso Industrial.

En la figura I.1 se puede observar lo siguiente:

1. El sensor A cierra el interruptor Sa cuando la temperatura del reactivo A, presente en la autoclave, está comprendida entre T_1 y T_2. Se supone que la temperatura requerida está comprendida en ese rango.
2. El sensor B cierra el interruptor Sb cuando el reactivo A se encuentre presente en la autoclave con el volumen requerido.
3. El temporizador electromecánico energiza la bobina del relevador RL2, por un tiempo T, cuando en su entrada (E) pasa el potencial del vivo del suministro eléctrico.

En la industria, a un esquema como el mostrado en la figura anterior, se le llama diagrama de escalera dada la obvia similitud visual.

Otra manera de resolver nuestro problema sería empleando componentes electrónicos integrados. La lógica requerida sería efectuada a un nivel de baja potencia, sin embargo, se sigue teniendo el problema de realambrado cuando hubiera que realizar cambios en la lógica de control.

La solución sería sustituir la electrónica TTL por una computadora monotablilla que tuviera un puerto de entrada y un puerto de salida, la ventaja de este procedimiento implica que si hubiera la necesidad de realizar cambios en la lógica de control, sólo habría que cambiar el programa que se ejecuta en la computadora monotablilla, sin que sea necesario hacer modificaciones al hardware.

Todavía hay que considerar que requiere además de un sistema programador experto que domine aspectos tanto de hardware como de software relacionados con el microcontrolador o microprocesador que sea el núcleo de la computadora monotablilla implicada.

En consecuencia, es deseable contar con un lenguaje de programación que genere código para la computadora monotablilla, de modo que el usuario final no se encuentre con detalles técnicos de la arquitectura y funcionamiento del microcontrolador correspondiente, sino sólo con la manera en que se debe declarar los módulos lógicos que su aplicación requiera en un momento dado.

Un sistema que conjunta el hardware descrito como sistema genérico de control lógico aunado con software que facilite al usuario final el desarrollo de aplicaciones es conocido en la industria como CONTROLADOR LÓGICO PROGRAMABLE, (PLC por sus siglas en inglés).

El objetivo del proyecto de tesis, reportado en esta presentación escrita, fue el de diseñar y desarrollar un programa denominado **PUMA ESCALERA**, que realiza bloques funcionales denominados módulos lógicos construidos por el usuario en una interfaz gráfica, los cuales son usualmente requeridos en el control lógico de procesos; tales módulos manejarán entradas y salidas binarias que se implantan en la interfaz mediante cuadros de diálogo. A partir de estos módulos lógicos se generarán tramos de código ejecutable en **LENGUAJE SIIL1**, obteniéndose de esta manera un proceso de traducción. Estos tramos de código se encontrarán en un archivo de texto que se creará después de que la traducción se haya llevado a cabo.

Así la programación del PLM implicará en un principio la construcción de los módulos lógicos **PUMA ESCALERA**, continuará con la traducción de estos al Software de interpretación de Instrucciones Lógicas (SIIL1), cuyos tramos de código ejecutable generados serán implantados en una computadora monotablilla basada en el microcontrolador 68HC11F1.

CAPÍTULO 1

ESTRUCTURA Y FUNCIONAMIENTO BÁSICO DEL PROGRAMADOR LÓGICO MODULAR

En este capítulo se describe de una manera general la estructura a bloques del Programador Lógico Modular (PLM); la organización y nomenclatura asociada con las variables booleanas que maneja; las características a nivel de "caja negra", de los módulos lógicos que puede realizar el dispositivo y el formato sintáctico de las instrucciones para declararlos en SIIL1 (lenguaje propio del PLM para la utilización por parte del usuario final).

Se presenta una descripción del formato que un programa en SIIL1 debe tener de manera que el mismo pueda ser procesado en una PC para obtener código objeto, listo para ser cargado y ejecutado en el PLM.

El capítulo concluye con la metodología a seguir para la realización de una aplicación de control lógico empleando el PLM.

1.1 Estructura Básica Del Programador Lógico Modular

El PLM, es un dispositivo orientado a la realización de diversos bloques funcionales típicos de aplicaciones de control lógico, como podrían ser: compuertas lógicas, temporizadores, contadores de eventos y secuenciadores de estados; en la nomenclatura del PLM se le llama módulo lógico a un bloque de los mencionados anteriormente, realizado virtualmente por software ejecutable en el microcontrolador 68HC11F1, que gobierna el funcionamiento del dispositivo.

Los módulos lógicos que el PLM puede realizar son:

a) Compuertas AND de dos, tres y cuatro entradas.
b) Compuertas OR de dos, tres y cuatro entradas.
c) Compuertas NAND de dos, tres y cuatro entradas.
d) Compuertas NOR de dos, tres y cuatro entradas.
e) Compuertas XOR de dos, tres y cuatro entradas.
f) Compuertas XOR negada de dos, tres y cuatro entradas.
g) Inversores y seguidores lógicos.
h) Cinco tipos diferentes de temporizador.
i) Dos tipos de contadores de eventos.
j) Secuenciadores de estados de uno a ocho bits.
k) Flip-Flops asíncronos.

Cabe señalar aquí que las compuertas XOR y XOR negada se denominan respectivamente como EOR y EORN en la terminología del PLM, además de que para todas las compuertas existe la posibilidad de negación para cualquiera de las entradas, contribuyendo esto a disminuir el número de módulos requeridos en una determinada aplicación.

El PLM puede operar de estos modos denominados respectivamente autónomo y esclavo. Al operar de manera autónoma el PLM puede realizar un sistema de control lógico, ejecutando el código correspondiente que se encontrará residente como firmware en una EPROM contenida en el mismo, esta idea se ilustra en la figura 1.1; cuando el PLM opera en modo esclavo el mismo se encontrará ligado vía serie con una computadora de tipo PC donde puede correrse software que permitirá probar y depurar los programas que requieran el PLM para realizar una determinada aplicación de control lógico, véase la figura 1.2.

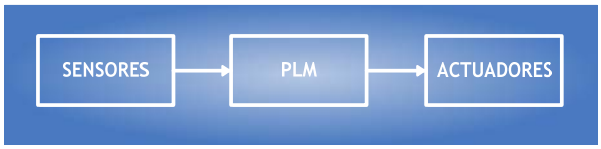


Figura 1.1 PLM operando de forma autónoma

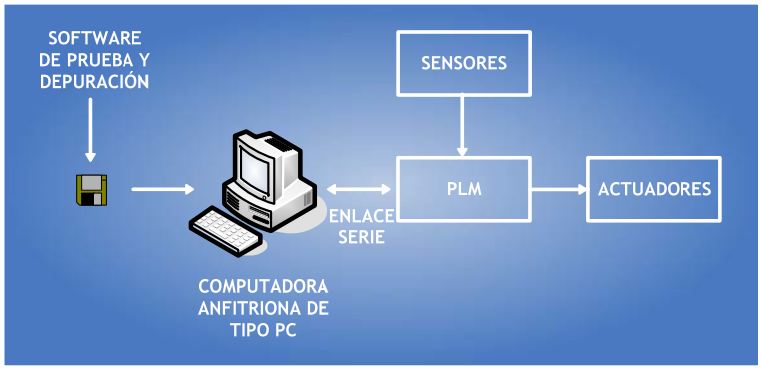


Figura 1.2 PLM operando en modo esclavo

La estructura a bloques del PLM se muestra en la figura 1.3, como se aprecia en la figura el dispositivo cuenta con 32 entradas y 16 salidas booleanas y está conformado por los siguientes cinco bloques funcionales:

1) Computadora Central (CC)
2) Bloque de Entradas (BE)
3) Bloque de Salidas (BS)
4) Bloque de Comando Local y Despliegue (BCLD)
5) Fuente de Alimentación (FA)

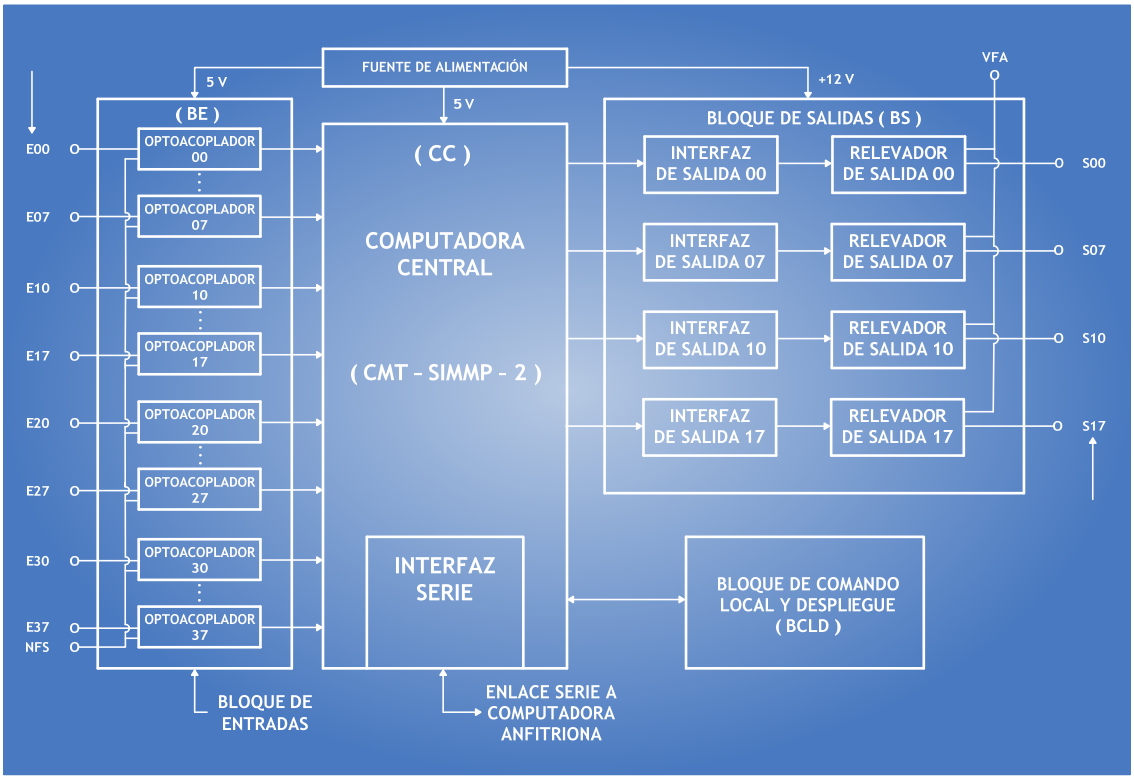


Figura 1.3 Estructura a Bloques del Programador Lógico Modular

A continuación se describe, en lo general, el funcionamiento de cada uno de los bloques del PLM.

1.1.1 Computadora Central (CC)

La computadora central del PLM está realizada por la computadora monotablilla (CMT) SIMMP-2, cuya CPU es el microcontrolador 68HC11F1 fabricado por la compañía Motorola, la CMT SIMMP-2 puede operar en cualquiera de los cuatro modos en los que puede funcionar el 68HC11 y cuenta con facilidades que permiten que la misma opere de manera autónoma o bien controlada vía serie por una computadora anfitriona de tipo PC. La CMT SIMMP-2 fue desarrollada por el M.I. Antonio Salvá Calleja y tiene las siguientes características principales:

1) Capacidad para operar en cualquiera de los cuatro modos asociados con el 68HC11.
2) Firmware interlocutor que permite enlazarla vía serie a una computadora PC, donde se ejecuta un manejador hexadecimal (programa pumma.exe) mediante el cual se puede entre otras cuestiones realizar lo siguiente:

    a) Cargar desde la PC, programas en lenguaje de máquina del microcontrolador para su ejecución en el mismo.
    b) Lectura desde la PC, de la memoria contenida en la tarjeta.
    c) Compatibilidad con herramientas de software asociadas con el 68HC11, permitiendo esto la ejecución en la tarjeta de programas originalmente

escritos en lenguaje C o ensamblador, lográndose esto mediante la importación del archivo S19 correspondiente que haya sido generado por el software de ensamble o compilación respectivo.

d) Capacidad para configurar diversos mapas de memoria al operar en modo extendido.
e) Programador integrado de memorias EPROM, mediante el cual pueden programarse EPROM's usando el propio manejador hexadecimal y hardware contenido en la tarjeta.

La CMT SIMMP-2 como computadora central del PLM opera en el modo expandido del 68HC11, contándose en este caso con seis puertos, cuatro de entradas y dos de salidas, con los que se realizan a nivel de la CC las entradas y salidas con que cuenta el PLM.

1.1.2 Bloque de Entradas (BE)

Esta parte está conformada por 32 entradas optoacopladas, el PLM reconoce un nivel de uno lógico, para una determinada entrada, cuando nominalmente se presente un voltaje de 24 volts medido entre la Terminal correspondiente y el punto NFS (neutro de la fuente de sensores), en otro caso el nivel tomado será cero lógico. En realidad para los niveles de uno y cero lógico en las entradas corresponden sendos intervalos de voltaje.

Como se observa en la figura 1.3, las 32 entradas están agrupadas en cuatro grupos de ocho entradas cada uno, esto se debe a que la información en el microcontrolador empleado está organizada en bytes.

Las entradas son denotadas empleando tres caracteres, el primero puede ser la letra "e" como mayúscula o minúscula, el segundo es un número comprendido en un rango de cero a 3 que indica el grupo, y finalmente el tercer carácter puede ser un número comprendido entre cero y siete que indica el bit de entrada correspondiente; así por ejemplo, la entrada correspondiente al bit 3 del grupo 2 puede ser indicada como "E23", para cada grupo de entradas corresponde un puerto físico con una determinada dirección en el mapa de puertos de la CC.

1.1.3 Bloque de Salidas (BS)

Este bloque está realizado por dos puertos de salida de la CC, de modo que para cada unos de los bits se cuenta con una interfaz a un relevador de baja potencia de contactos normalmente abiertos.

Todas las terminales comunes de contactos de los 16 relevadores están conectadas al punto VFA (vivo de la fuente de actuadores) en tanto que para cada relevador el otro contacto está conectado con su correspondiente Terminal de salida asociada.

La salidas se denotan con tres caracteres, el primero es la letra “s” como mayúscula o minúscula, el segundo es un número que puede ser cero o uno indicando esto el grupo al que pertenece la salida en cuestión, finalmente el tercer carácter es un número comprendido entre cero y siete que denota el número de bit de salida correspondiente; así por ejemplo, la salida 4 del grupo uno puede indicarse como “S14”.

Al verificarse el nivel de uno lógico para una determinada salida habrá continuidad eléctrica entre las terminales VFA y la propia correspondiente con la salida, en otro caso no habrá continuidad eléctrica. La máxima corriente permisible, para disparo del actuador correspondiente, es 500mA.

1.1.4 Bloque de Comando Local y Despliegue (BCLD)

Desde el punto de vista del usuario final este bloque está constituido por tres componentes, uno de ellos de la Unidad Desplegadora (UD) que maneja dos renglones de 16 caracteres, otro es un panel que contiene cinco postes que habilitan sendas entradas binarias auxiliares, cuatro botones para comando local y dos pares de postes donde se podrían colocar puentes (jumpers) que configurarían la manera en que el PLM respondería a una reinicialización del programa del usuario; el tercer componente del BCLD es un reloj de tiempo real, que puede servir simplemente como testigo de la hora o como base de tiempo para una función especial del dispositivo; que permite generar disparos a otros módulos lógicos para horarios predeterminados por el usuario en el programa SIIL1, hecho de acuerdo con las necesidades de una determinada aplicación.

Para la implantación de la unidad desplegadora se utilizó el desplegado alfanumérico inteligente AND491 fabricado por la corporación electrónica PURDY; en lo que toca al reloj de tiempo real se empleó el chip MM58274N fabricado por NATIONAL que es un componente pensado para interconectarse con un microcontrolador o microprocesador.

Los botones y postes para entradas auxiliares o colocación de puentes, están validados por dos puertos de entrada (denotados como PAUXA y PAUXB) adicionales a los que forman parte de la arquitectura de la CMT SIMMP-2. Los cuatro botones están denotados como BAXA, BAXB, BAXC y BAXD; los tres primeros son usados para poner el reloj de tiempo real a una hora determinada, el último es usado como auxiliar en una de las instrucciones que maneja la UD; los postes que presentan las cinco entradas auxiliares están denotados como EA1, EA2, EA3 y EA4 y EA5; los puentes sirven para configurar diferentes tipos de respuesta al restablecimiento y se denominan Ja y Jb. En la figura 1.2 se muestra un esquema a bloques del BCLD y en la tabla 1.1 se resumen, en lo general, las funciones de los botones de comando local, los puertos y las entradas binarias auxiliares.

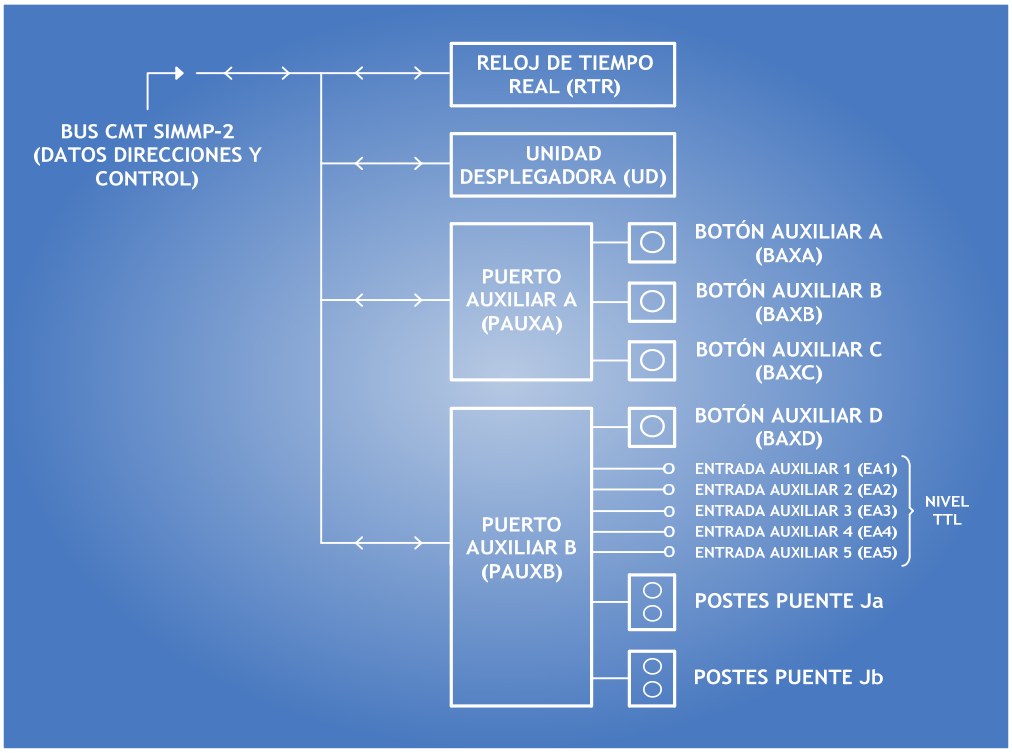


Figura 1.4 Esquema a bloques del Bloque de Comando Local y Despliegue del PLM

INSTANCIA DEL BLOQUE DE COMANDO LOCAL Y DESPLIEGUE (BCLD)	USO EN EL PLM DESDE EL PUNTO DE VISTA DEL USUARIO FINAL.
Botones BAXA, BAXB y BAXC	Ajuste y puesta a tiempo del reloj de tiempo real (RTR)
Botón BAXD	Este botón se emplea para desplegar secuencialmente mensajes priorizados en la UD
Entradas auxiliares EA1 a EA5	Reservadas para funciones futuras que pudieran requerir botones o puentes
Puente Ja	Con Ja no colocado, al reinicializar el programa del usuario el reloj de tiempo real se pone en ceros (00:00:00), en otro caso el RTR conserva la hora al reinicializar el programa del usuario.
Puente Jb	Con Jb no colocado, al reinicializar el programa del usuario, se ponen en cero todas las variables booleanas que use la aplicación, en otro caso las variables conservan el valor que tenían antes de la reinicialización

Tabla 1.1 Resumen de funciones asociadas con instancias del BCLD, del PLM

1.1.5 Fuente de Alimentación (FA)

El PLM requiere para su funcionamiento de dos fuentes de voltaje, una de 12 volts y otra de 5 volts, la primera polariza únicamente a los relevadores del bloque de salidas y requiere de una capacidad de corriente de un Ampere, el requerimiento de corriente de la segunda fuente mencionada es de 500mA; cabe señalar aquí que para el primer prototipo del PLM, reportado en esta tesis, la fuente de alimentación se implantó empleando una fuente comercial para laboratorio de electrónica.

1.2 Variables Booleanas En El PLM

Las entradas y salidas de los módulos lógicos que pueden ser realizados con el PLM son variables booleanas que son clasificadas como: variables booleanas de entrada (VBE), variables booleanas de salida (VBS) y variables booleanas intermediarias (VBI).

Dado que la información en el microcontrolador 68HC11 está organizada en bytes, las variables mencionadas aquí están aglutinadas en conjuntos (grupos) de ocho variables de un mismo tipo; esto es, hay cuatro grupos de variables booleanas de entrada, dos grupos de variables booleanas de salida y 21 grupos de variables booleanas intermediarias; a continuación se describen conceptos asociados con cada unos de los tipos de variables del dispositivo.

1.2.1 Variables Booleanas de Entrada (VBE)

Este tipo de variables están asociadas con sendas terminales de entrada siendo cada una de ellas optoacopladas a la CC, cada Terminal de entrada puede recibir una señal lógica de voltaje (0-24 volts) proveniente de algún sensor, que sea parte del sistema de control lógico, que se requiera implantar en un momento dado.

El primer prototipo del PLM está pensado para manejar 32 VBE’s. Cada VBE se especifica con tres caracteres, el primero es una letra “e” mayúscula o minúscula, el segundo es un número del cero al tres que denota el grupo al que pertenece la VBE y el tercero es un número del cero al siete que define el bit asociado del puerto de entrada relacionado con el grupo de entradas de que se trate; así por ejemplo, la quinta variable del grupo de entradas dos se podría denotar como E25.

1.2.2 Variables Booleanas de Salida (VBS)

Existen para el PLM 16 variables booleanas de salida, cada una de ellas está asociada con un relevador de baja potencia cuyos contactos se cierran al presentar la variable de salida correspondiente el nivel de uno lógico, al ser cero lógico el valor en cuestión tales contactos permanecen abiertos; las VBS están aglutinadas en dos grupos de ocho salidas cada uno; de esta forma, se emplean tres caracteres para denotar a una VBS, el primero es la letra “s” mayúscula o minúscula, el segundo es un número que puede ser cero o uno que denota el número de grupo de salida y el tercero es un dígito del cero al siete que define el bit asociado con el puerto de salida físico de la CC relacionado; así por ejemplo, la salida dos del grupo de salidas cero se podría definir como S02.

1.2.3 Variables Booleanas Intermediarias (VBI)

Este tipo de variables son manejadas internamente y no tienen entradas o salidas físicas asociadas, su función consiste en servir de enlace entre módulos lógicos cuando esto sea necesario; por ejemplo, supóngase que se tiene una situación de control lógico que requiere de varias compuertas lógicas, puede suceder que las salidas de algunas de ellas sean variables requeridas como entrada de otras, el emplear variables físicas de salida para habilitar esta circunstancia no sería conveniente dado que su número es limitado, de ahí la necesidad de contar con las VBI; desde luego que una variable booleana, que sea entrada de un módulo y salida de otro, puede ser una salida física si esto es necesario, lo que obviamente no es permitido es el hecho de que una variable sea simultáneamente salida de más de un módulo.

Las VBI están agrupadas en 21 grupos de ocho variables cado uno, de esta forma se puede contar dentro del PLM con 168 variables de este tipo; la notación empleada para designarlas emplea cuatro caracteres, el primero es la letra “i” mayúscula o minúscula, el segundo y tercero representan a un número comprendido entre cero y veinte que denota el número de grupo al que pertenece la VBI y el cuarto es un dígito del cero al siete que define el número de VBI dentro del grupo; así por ejemplo, la VBI cuatro del grupo doce de VBI’s se denominaría I124.

1.3 Descripción General De Los Módulos Lógicos

Los módulos lógicos (ML) que puede realizar el PLM constituyen los bloques funcionales elementales para la realización de aplicaciones de control lógico y pueden ser representados a nivel de “caja negra” como se muestra en la figura 1.5, donde se muestra un ML que presenta “m” entradas y “n” salidas; m y n varían de acuerdo con el tipo de función que un determinado ML realice; así por ejemplo, para una compuerta AND de tres entradas m y n

serían tres y uno respectivamente; en cambio, un secuenciador de estados con palabras de cuatro bits requerirá tres entradas y cinco salidas.

Figura 1.5 Representación de bloque de un módulo lógico de m entradas y n salidas

En la figura 1.5 XEm representaría a un carácter que podría ser cualquiera de las letras e, i o s mayúsculas o minúsculas dependiendo esto del tipo de variable (VBE, VBI o VBS) asociada con la entrada con la entrada m-esima del ML; IEm y JEm serían respectivamente los números asociados con el grupo y el número de bit correspondientes con la variable m-esima de entrada; XSm representaría a un carácter que podría ser cualquiera de las letras i o s mayúsculas o minúsculas dependiendo del tipo de variable (VBI o VBS) asociada con la salida m-esima del ML; ISm y JSm serían respectivamente los números asociados con el grupo y el número de bit correspondientes con la variable m-esima de salida. Por ejemplo, una compuerta NAND de tres entradas con negación en una de ellas se muestra en la figura 1.6, las entradas son respectivamente las variables E01, I45 (entrada negada) y E13, la salida es la variable S02, nótese que las entradas pueden ser de grupos diferentes.

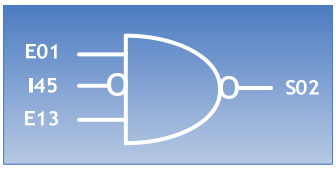


Figura 1.6 Representación de una compuerta NAND.

Para todos los ML que validan compuertas lógicas se tiene que los mismos responden al nivel que presenten sus entradas; sin embargo, algunos de los ML que nos son compuertas están diseñados de modo que responden a flancos que se presenten en una o varias de sus entradas; en la figura 1.7 se muestra un ML que se encuentra en este caso, se trata de un temporizador de tipo monodisparo (one-shot) que presentará en su salida (variable S14) un pulso verificado alto de una duración determinada, cada vez que en la entrada de disparo (variable E12) se manifieste un flanco de bajada, en lo que toca a la otra entrada (variable E02) el ML responde el nivel, cuando el mismo es alto el temporizador está habilitado, en caso de que el nivel sea bajo se retorna la salida a su nivel no verificado no respondiendo el módulo a los disparos hasta que la entrada mencionada retorne a el nivel alto.

A nivel de los esquemas de bloques asociados con los ML la sensibilidad a flancos de una entrada es denotada mediante una flecha vertical cuyo sentido indica el tipo de flanco asociado, véase la figura 1.7.

Figura 1.7 a) Representación de un temporizador monodisparo (one-shot).

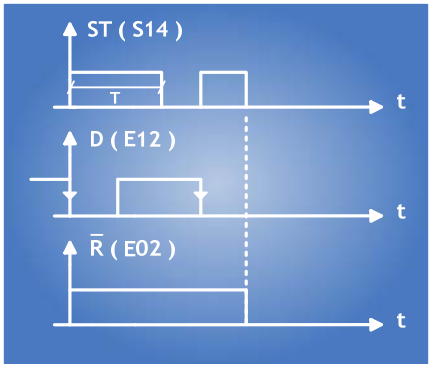


Figura 1.7 b) Diagrama de tiempos correspondiente al temporizador mostrado

1.4 Formas Sintácticas Asociadas Con Los Módulos Lógicos

Cada uno de los módulos lógicos, requeridos en una determinada aplicación, deben haber sido declarados secuencialmente en un archivo de texto, para que el mismo sea procesado por una computadora anfitriona empleando un programa denominado SIIL1 (Software de Interpretación de Instrucciones Lógicas), que genera el código objeto que ha de ejecutar la CC del PLM de modo que los ML requeridos queden realizados; además de las declaraciones asociadas con los módulos, en el archivo mencionado se requieren colocar otras instrucciones, no relacionadas directamente con algún ML, pero necesarias para la ejecución adecuada del programa que ha de ejecutarse en el PLM, de esto se hablará más adelante.

Al conjunto de instrucciones, mencionadas anteriormente, escritas en secuencia en un archivo de texto, se le denomina programa fuente en lenguaje SIIL1 asociado con la aplicación que ha de realizar el PLM. A excepción de los módulos que requieren datos adicionales que ha de proporcionar el usuario, la forma sintáctica de las declaraciones asociadas con los mismos requiere de un solo renglón en el archivo de texto a procesar, esta forma se ilustra a continuación:

CODM#N ENT_1, ENT_2, …, ENT_m, SAL_1, SAL_2, …, SAL_n, DA_1, …DA_q,…, CADBI;

Donde:

- CODM es una cadena de caracteres que simboliza la función efectuada por el módulo.
- N es el número asociado con el módulo, ya que todos los ML de un mismo tipo que use una aplicación, deben ser numerados.

- ENT_1 a ENT_m son las designaciones asociadas con las m variables de entrada que el módulo requiera.
- SAL_1 a SAL_n son las designaciones asociadas con las n salidas que pudiera tener el módulo.
- DA_1 a DA_q son datos auxiliares que pudieran ser requeridos por algunos módulos, estos podrían ser entre otros: el tiempo asociado con la duración de un pulso generado por un temporizador o bien el número de estados que ha de presentar un secuenciador, etc. hay módulos que no requieren de estas especificaciones, tal es el caso de las compuertas lógicas. Para los módulos que si requieren de estos datos, **q** es un número que está comprendido entre cero y tres.

- CADBI es una cadena formada por unos y ceros que especifica diversas características de funcionamiento como podrían ser: que entradas a una compuerta van a tener negación implícita, a que tipo de flanco responde una entrada de algún otro tipo de módulo, etc. En el Anexo A "Manual de Usuario" de esta tesis se trata en detalle para cada módulo las características que en cada caso define cada carácter (1 ó 0), que integra la cadena CADBI que corresponda.

Cabe señalar que el primer carácter de la instrucción nunca deberá estar en la primera columna y que al final de la misma siempre ha de colocarse el caracter "**;**".

Como ejemplo de estructura sintáctica, a continuación se muestra la instrucción asociada con la declaración de la compuerta NAND de tres entradas mostrada en la figura 1.6.

NAND3#1 E01, I45, E13, S02, 101;

En la instrucción anterior CODM es la palabra NAND3 que denota el hecho de que se trata de una compuerta NAND de tres entradas; por ejemplo, si se hubiera tratado de una compuerta NAND de cuatro entradas CODM hubiera sido NAND4.

Nótese además que la cadena binaria asociada (CADBI) consta de tres bits, ya que la compuerta es de tres entradas, indicándose que la entrada I45 deberá tener negación implícita colocando un cero en la posición que corresponde a tal entrada; así, si se requiriera que tuvieran negación implícita las dos primeras entradas (E01 e I45) CADBI sería 001.

Se aprecia también en la declaración anterior que se le ha asignado el número uno a la compuerta NAND en cuestión.

1.5 Formato De Un Programa En SIIL1

1.5.1 Características generales de la ejecución de un programa en SIIL1 en la CC del PLM.

Al correr un programa en SIIL1 en la CC del PLM, el código asociado con cada ML es ejecutado cíclicamente siguiendo la siguiente secuencia:

1. Se copian en un buffer de entrada (BE) en RAM el estado que guardan los cuatro puertos asociados con las 32 entradas físicas VBE.
2. Se ejecuta uno a uno el código asociado con cada uno de los ML que el usuario haya declarado en el programa fuente correspondiente, tomándose del BE las entradas que cada módulo requiera, las salidas que se fueran generando son colocadas en un buffer de salida (BS) en RAM; si el ML emplea una o varias VBI como entradas los valores asociados con las mismas son tomados de un buffer intermediario (BI) en RAM, en caso de que haya en el ML una o varias salidas de tipo VBI los valores correspondientes son escritos en el BI.
3. Se copia el estado del BS en los puertos físicos asociados con las VBS.
4. Se regresa al paso uno.

De lo anterior se aprecia que el código asociado con cada módulo es ejecutado repetitivamente, variando el intervalo de repetición de acuerdo con el número de módulos que contenga el programa; esto es, a mayor número de módulos crece el periodo de repetición.

Existen módulos que requieren que el periodo de repetición de la ejecución de su código asociado sea constante (10ms), tal es el caso por ejemplo de los temporizadores, para hacer esto posible el código asociado es colocado en una rutina de servicio de interrupción que es invocada con una periodicidad de 10 ms, empleándose para ello facilidades de temporización con que cuenta la CC del PLM.

En consecuencia, el código asociado con un programa en SIIL1 está dividido en dos partes, una de ellas es la que ejecuta de acuerdo con los cuatro pasos descritos en párrafos anteriores, a esta parte se le llama subprograma principal, la otra parte está constituida por el código cuya ejecución es temporizada y se denomina subprograma temporizado. En la figura 1.8 se ilustra esta idea. Cabe señalar aquí que todo programa en SIIL1 debe tener un subprograma principal; sin embargo, puede haber programas que no contengan un subprograma temporizado.

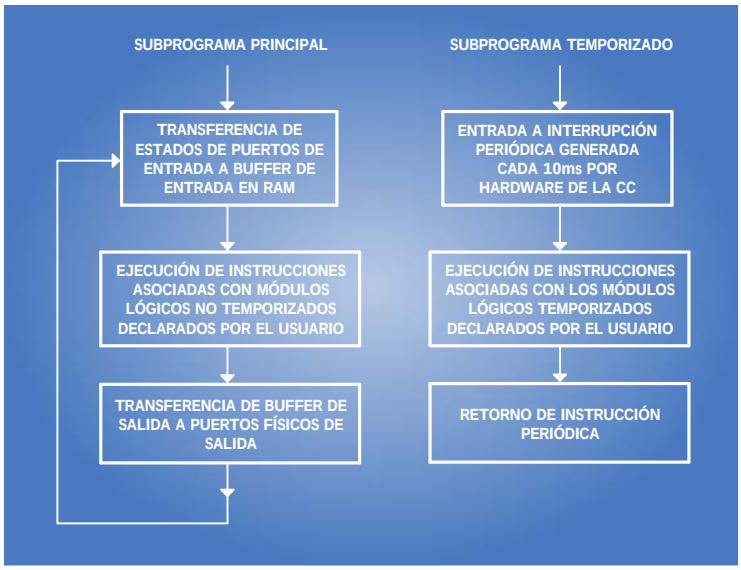


Figura 1.8 Ejecución, en la CC del PLM, de los dos subprogramas que integran un programa en SIIL1

1.5.2 Forma de un programa fuente en SIIL1

El programa fuente en SIIL asociado con una aplicación está constituido por declaraciones, que pueden ser comandos o instrucciones; los comandos son indicaciones tales como inicio o fin del subprograma principal, tipo de mapa de memoria empleado en la CC, inicio o fin de instrucciones asociadas con el subprograma temporizado; las instrucciones son declaraciones asociadas con características que han de tener los módulos empleados por la aplicación y deben respetar la sintaxis descrita en párrafos anteriores. En lo general un programa fuente en SIIL1 está integrado por la siguiente secuencia de declaraciones.

1. Comando que indica el tipo de configuración de funcionamiento, la sintaxis asociada es CONFIGN, donde N es un número entero que puede ser uno, dos ó tres; de esta manera, existen a la fecha de elaboración de este trabajo de tesis, tres posibles configuraciones de funcionamiento para el PLM. En la tabla 1.2 se resumen las características principales de funcionamiento asociadas con cada configuración.
2. Comando que marca el inicio del subprograma principal, la sintaxis correspondiente en este caso es INPROG.
3. Instrucciones asociadas con los módulos que se desea integren al subprograma principal.
4. Comando que marca el fin del subprograma principal, la sintaxis en este caso es FINPP.
5. Comando que marca el inicio del subprograma temporizado, la sintaxis asociada es INMODI.
6. Instrucciones asociadas con los módulos que han de integrar al subprograma temporizado.
7. Comando que indica el fin del subprograma temporizado, la sintaxis en este caso es FINMODI.

Los siete componentes del programa fuente han de ser colocados respetando el orden anterior; al igual que en el caso de las instrucciones asociadas con los módulos, los comandos deberán ser concluidos con el caracter "**;**". Todo texto colocado a la derecha del caracter "**;**" no es tomado en cuenta por el programa que genera el código objeto, de esta manera pueden adicionarse comentarios al programa fuente.

CONFIGURACIÓN	ENTRADAS	SALIDAS	MÁXIMO TAMAÑO DEL PROGRAMA (KB)
1	32	16	7.5
2 *	8	8	7.5
3	32	16	24

Tabla 1.2 Resumen de características de funcionamiento del PLM asociadas con las diferentes configuraciones.

* Esta configuración se empleó para la prueba de los módulos con lógica nivel TTL.

1.6 Metodología Para Estructurar Una Aplicación De Control Lógico Empleando El PLM

Toda aplicación de control lógico puede integrarse empleando tres conjuntos de elementos funcionales denominados como:

1. Elementos sensores, los cuales son dispositivos que presentan en su salida un nivel lógico que testifica un determinado evento como podrían ser por ejemplo el paso de un producto por una banda transportadora, el fin del recorrido de un embolo, el que un operador oprima un botón, etc. Los sensores pueden estar constituidos desde simples interruptores hasta por bloques que basan su funcionamiento en componentes electrónicos. Frecuentemente en la industria los niveles lógicos empleados son cero y 24 volts.
2. Elementos lógicos, los cuales son dispositivos que realizan funciones booleanas cuyas entradas son las variables presentadas por los sensores, siendo sus salidas variables lógicas que comandan a elementos actuadores, de modo que físicamente se realicen los eventos que la aplicación requiera.
3. Elementos actuadores, que son dispositivos que actúan directamente sobre el proceso y son comandados por las salidas que generan los elementos lógicos mencionados en el, ejemplos de actuadores podrían ser resistencias eléctricas que suministren calor, motores eléctricos que muevan elementos mecánicos de diversa índole, etc.

El papel del PLM en la realización de un sistema de control lógico es el de implantar los elementos lógicos que la aplicación requiera, empleando para ello a los módulos lógicos que el mismo puede realizar virtualmente.

A continuación se describe el proceso a seguir para que el conjunto de módulos lógicos necesarios en un control lógico tomen forma en el PLM, se supone que el PLM debe operar en modo esclavo y debe estar convenientemente enlazado con una computadora anfitriona

de tipo PC; desde luego que el desarrollo completo debe contemplar lo relacionado con los sensores y actuadores. En síntesis los pasos a seguir son los siguientes:

1. Definir los módulos lógicos que la aplicación requiera, especificando para cada uno las variables de entrada y salida empleadas respetando el hecho de que una variable no puede ser salida de más de un módulo; de lo anterior escribir el programa fuente asociado siguiendo los lineamientos dados en el tema 1.5, empleando para ello a un editor de texto convencional que se ejecute en la computadora anfitriona.

2. Guardar el texto generado en el paso anterior en un archivo con un nombre seleccionado por el usuario y con la extensión SIL.

3. Ejecutar en la computadora anfitriona el Software de Interpretación de Instrucciones Lógicas (programa SIIL1.EXE) tomando como archivo de entrada el generado en el paso anterior, en caso de haber errores de sintaxis en las declaraciones mencionadas en el paso uno se mostrarán en pantalla los mismos.
En caso que no haya errores se indicará esto, generándose además un archivo binario con el mismo nombre dado por el usuario en el paso dos y con la extensión BLM; este último archivo contendrá el código ejecutable por el PLM correspondiente a la aplicación que se esté desarrollando.

4. Si se detectaron errores en el paso anterior corregirlos en el editor de texto y regresar al paso dos. Si no hubo errores de sintaxis proceder al paso cinco.

5. Transferir para ejecución, a la memoria RAM del PLM, el código contenido en el archivo BLM generado en el paso tres, esto puede hacerse empleando el manejador hexadecimal PUMMA propio de la tarjeta SIMMP-2.

6. En caso de que el programa no opere correctamente en el PLM hacer los cambios necesarios, a nivel de los módulos lógicos empleados por la aplicación, y regresar al paso uno. Si el programa opera correctamente proceder al siguiente paso.

7. Desenergizar el PLM, colocar una EPROM borrada en la base correspondiente, así como los puentes J4 y J5 en la CC (tarjeta SIMMP-2), energizar el PLM, oprimir y soltar el botón de restablecimiento del mismo.

8. Programar la EPROM con el código objeto contenido en el archivo BLM generado en el paso tres. Para efectuar esto se puede emplear el manejador hexadecimal PUMMA, propio de la tarjeta SIMMP-2.

9. Quitar los puentes J4 y J5 y validar el modo autónomo de operación, esto último se hace colocando el puente J11 en la CC (CMT SIMMP-2).

10. Oprimir y soltar el botón de restablecimiento del PLM, al hacer esto deberá ejecutarse en forma autónoma el programa del usuario, de esta manera el sistema de control lógico diseñado funcionará siendo realizado por el PLM.

1.7 Ejemplo

Para aclarar algunos de los conceptos descritos en este capítulo, se muestra a continuación como se programaría el PLM para realizar la situación de control lógico mencionada en la introducción de esta tesis. En la figura 1.9 se muestra un esquema a bloques que emplea módulos propios del PLM, para realizar la situación de control lógico mostrada en la figura I.1, se supone que la bomba que suministra el reactivo B debe operar durante treinta segundos.

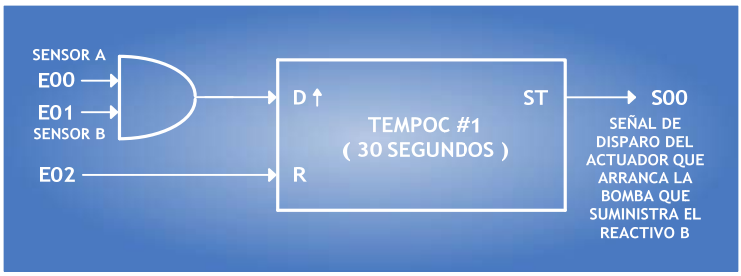


Figura 1.9 Esquema a bloques de la situación de control lógico, mostrada en la figura I.1

El temporizador requerido en este ejemplo es de tipo monodisparo (one shot) como el ilustrado en la figura 1.7, para claridad en este ejemplo, la sintaxis genérica asociada con un ML que realice un temporizador como el requerido se muestra a continuación:

TEMPOC #N DISPARO, HABILITACIÓN, SALIDA, DURACIÓN, ABC;

Donde:

- N representa el número de temporizador.
- DISPARO denota la variable booleana que dispara el temporizador.
- HABILITACIÓN denota a la variable booleana asociada con la habilitación y restablecimiento del temporizador.
- SALIDA denota a la variable booleana asociada con la habilitación y restablecimiento del temporizador.
- DURACIÓN denota tiempo, que ha de especificarse en horas, minutos, segundos y centésimas de segundo de acuerdo con el formato 00:00:00.00; por ejemplo, si se desea que el pulse dure dos horas cuarenta y cinco minutos con diez segundos, esto se especificará como 02:45:10.00, en caso de que el tiempo dure ya sea menos de una hora o de un minuto se deberá denotar con ceros los espacios correspondientes a las horas y/o minutos según sea el caso.
- A es un caracter que podrá ser cero si se desea que el disparo sea por flanco de bajada o uno si se desea que el disparo sea por flanco de subida.
- B es una caracter que podrá ser cero si se desea que la habilitación sea por nivel alto y el restablecimiento sea por nivel bajo; en caso de que se requiera que la habilitación sea por nivel bajo y el restablecimiento por nivel alto, B deberá ser cero. Para que el temporizador responda a los disparos la habilitación del mismo deberá estar verificada, en caso de verificarse el restablecimiento mientras se verifica el pulso de salida, el mismo regresará a su nivel no verificado, véase la figura 1.7b.

- C es un caracter que podrá ser cero si se desea que el pulso de salida tenga verificación en bajo y uno en caso de que desee que tal verificación sea en alto.

Nótese en la figura 1.9 el empleo de dos variables booleanas intermediarias, aclarándose aquí que el valor por defecto de una variable booleana del PLM es cero, siendo esta la causa de que el nivel requerido para la señal de habilitación sea bajo, apreciándose el empleo, como delimitadora, de la variable intermediaria I01.

El programa correspondiente en SIIL1 es el siguiente:

CONFIG1; declaración de configuración de funcionamiento.
INPPROG; declaración de inicio de subprograma principal

La siguiente línea corresponde a la instrucción asociada con el módulo que realiza la compuerta lógica requerida.

AND2#1 E00, E01, I00, 111; compuerta and número 1
FINPP; declaración de fin de subprograma principal
INMODI; declaración de inicio de subprograma temporizado
TEMPOC#1 I00, I01, S00, 00:00:30.00, 101; temporizador
FINMODI; declaración de fin de subprograma temporizado

En la figura 1.10 se muestra el conexionado al PLM de los sensores y actuadores asociados con el ejemplo aquí descrito. Cuando la misma es realizada por el PLM, de acuerdo al diagrama de bloques mostrado en la figura 1.9

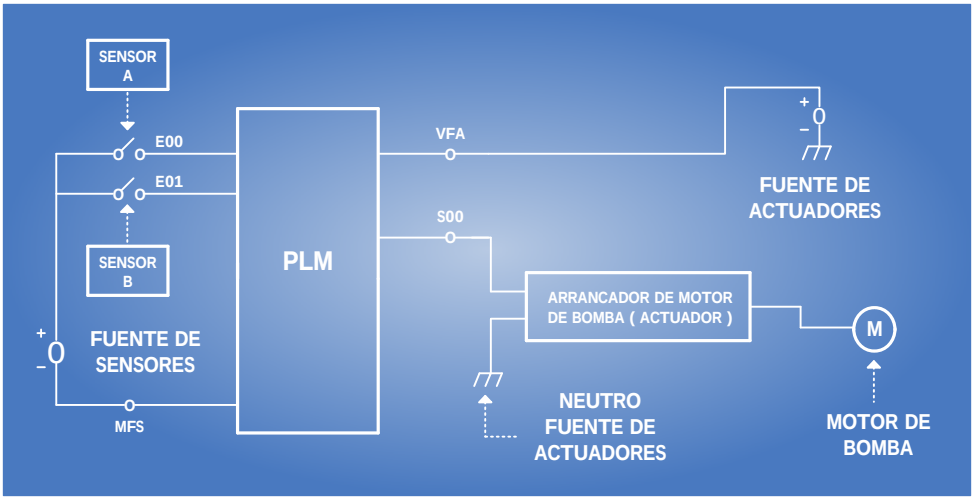


Figura 1.10 Conexión de los sensores y actuadores, asociados con la situación de control lógico mostrada en la figura I.1,

CAPITULO 2

LENGUAJE DE ESCALERA

2.1 Lenguajes De Programación De Un PLC

Para programar un PLC existen ya muchos lenguajes de programación. Estos lenguajes se agrupan en dos grandes categorías: Lenguajes Gráficos Y Lenguajes Textuales. Los lenguajes Gráficos a su vez se subdividen en lenguaje de Carta de Funciones Secuénciales o Grafcet**,** Plano de Funciones y lenguaje Ladder ó de escalera (ver figura 2.1).

Figura 2.1 Clasificación de los lenguajes

2.1.1 Lenguajes Gráficos

Se denomina lenguaje gráfico a la representación basada en símbolos gráficos, de tal forma que según la disposición en que se encuentran cada uno de estos símbolos Y en conformidad a su sintaxis que lo gobierna, expresa una lógica de mando y control. Dentro de ellos tenemos:

2.1.1.1 Carta de Funciones Secuénciales o Grafcet

El Grafcet es una representación de análisis gráfico donde se establecen las funciones de un sistema secuencial. Este lenguaje consiste en una secuencia de etapas y transiciones, asociadas respectivamente con acciones y condiciones.

Las etapas representan las acciones a realizar y las transiciones las condiciones que deben cumplirse para ir desarrollando acciones. La Etapa - Transición es un conjunto indisociable. Un ejemplo es el mostrado en la figura 2.2

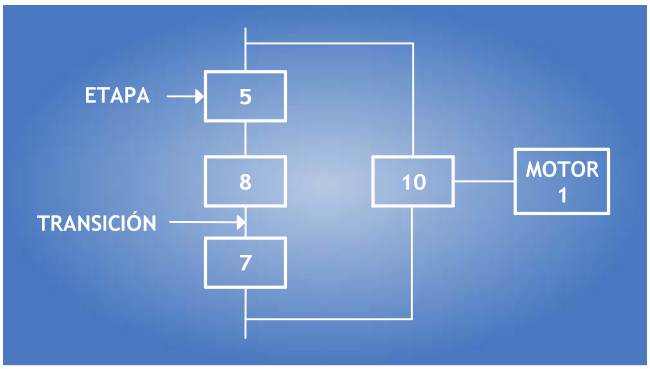


Figura 2.2 – Etapas y Transiciones la representación Grafcet

2.1.1.2 Plano de Funciones

Es una representación gráfica orientada a las puertas lógicas AND, OR y sus combinaciones. Las funciones individuales se representan con un símbolo, donde su lado izquierdo se ubica las entradas y en el derecho las salidas (figura 2.3). Los símbolos usados son iguales o semejantes a los que se utilizan en los esquemas de bloques en electrónica digital.

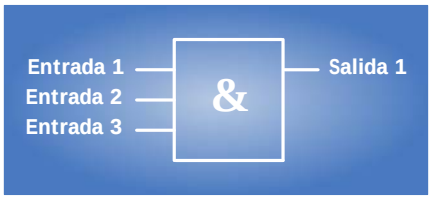


Figura 2.3 Ejemplo plano de funciones

2.1.1.3 Diagrama de Contactos o LADDER

Es la representación gráfica que tiene cierta analogía a los esquemas de contactos. Su estructura obedece a la semejanza que existe con los circuitos de control con lógica cableada, es decir, utiliza la misma representación de los contactos normalmente abiertos y normalmente cerrados, con la diferencia que su interpretación es totalmente diferente.

Además de los simples contactos que dispone, existen otros elementos que permiten realizar cálculos aritméticos, operaciones de comparación, implementar algoritmos de regulación, etc. Su gran difusión se debe por facilitar el trabajo a los usuarios. La figura 2.4 es un ejemplo de esta representación gráfica.

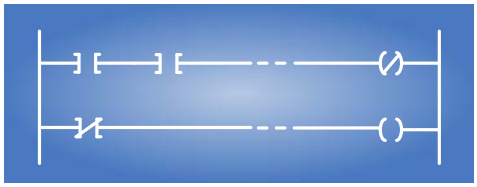


Figura 2.4 Ejemplo Ladder

2.1.2 Lenguajes Textuales

Este tipo de lenguaje se refiere básicamente al conjunto de instrucciones compuesto de letras, códigos y números de acuerdo a una sintaxis establecida.

Se considera un lenguaje de menor nivel que los gráficos y por lo general se utilizan para programar pequeños PLCs cuyos programas no son muy complejos, o para programar instrucciones no programables en modo gráfico.

Existen dos lenguajes diferentes en nivel y tipo de aplicación, ellos son:

2.1.2.1 Lista de Instrucciones

Son instrucciones del tipo Booleanas, utilizando para su representación letras y números.

Dado que se usan abreviaturas nemotécnicas, no se requiere gran memoria para tareas de automatización.

La desventaja radica en la magnitud del trabajo que es necesario para su programación, especialmente si el programa consta de unos cientos de instrucciones.

La Tabla 2.1 es una representación de un programa en lista de instrucciones para diferentes marcas de PLC's.

SIEMENS (SIMATIC)	TELEMECANIQUE	GENERAL ELECTRIC
U E0.1	L I0.01	LD %I0001
U E0.2	A I0.02	AND %I0002
O E0.3	O I0.03	OR %I0003
= A3.1	= O3.01	OUT %Q0031

Tabla 2.1 Representación de un programa en lista de instrucciones.

2.1.2.2 Texto Estructurado

Es un lenguaje del tipo booleano de alto nivel y estructurado, incluye las típicas sentencias de selección (IF-THEN-ELSE) y de interacción (FOR, WHILE Y REPEAT), además de otras funciones específicas para aplicaciones de control.

Su uso es ideal para aplicaciones en las que se requiere realizar cálculos matemáticos, comparaciones, emular protocolos, etc.

El siguiente es un ejemplo de programa en texto estructurado para un PLC marca Telemecanique TSX-07.

```
LD    [%MW10>100]
ST    %Q0.3
AND   [%MW20<%MW35]
ST    %Q0.5
LD    %I0.2
OR    [%MW30>=%MW40]
ST    %Q0.4
```

Para el desarrollo del software PUMA ESCALERA se utilizará el lenguaje de escalera, por lo que en lo sucesivo solo se ahondará en el mismo.

2.2 Lenguaje De Escalera

El **LADDER**, también denominado lenguaje de contactos o en escalera, es un lenguaje de programación gráfico muy popular dentro de los autómatas programables debido a que está basado en los esquemas eléctricos de control clásicos. De este modo, con los conocimientos que todo técnico eléctrico posee, es muy fácil adaptarse a la programación en este tipo de lenguaje.

2.2.1 Elementos de programación

Para programar un autómata con lenguaje de escalera, además de estar familiarizado con las reglas de los circuitos de conmutación, es necesario conocer cada uno de los elementos de que consta este lenguaje.

2.2.1.1 Elementos básicos

En la tabla 2.2 se pueden observar los símbolos de los elementos básicos y sus descripciones.

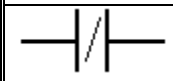
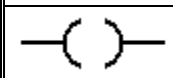
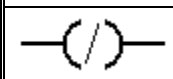
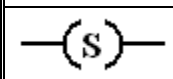
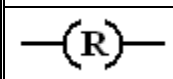
ELEMENTOS BÁSICOS EN LADDER		
Símbolo	**Nombre**	**Descripción**
	Contacto NA	Se activa cuando hay un uno lógico en el elemento que representa, esto es, una entrada (para captar información del proceso a controlar), una variable interna o un bit de sistema.
	Contacto NC	Su función es similar al contacto NA anterior, pero en este caso se activa cuando hay un cero lógico, cosa que deberá de tenerse muy en cuenta a la hora de su utilización.
	Bobina NA	Se activa cuando la combinación que hay a su entrada (izquierda) da un uno lógico. Su activación equivale a decir que tiene un uno lógico. Suele representar elementos de salida, aunque a veces puede hacer el papel de variable interna.
	Bobina NC	Se activa cuando la combinación que hay a su entrada (izquierda) da un cero lógico. Su activación equivale a decir que tiene un cero lógico. Su comportamiento es complementario al de la bobina NA.
	Bobina SET	Una vez activa (puesta a 1) no se puede desactivar (puesta a 0) si no es por su correspondiente bobina en RESET. Sirve para memorizar bits y usada junto con la bobina RESET dan una enorme potencia en la programación.
	Bobina RESET	Permite desactivar una bobina SET previamente activada.
	Bobina JUMP	Permite saltarse instrucciones del programa e ir directamente a la etiqueta que se desee. Sirve para realizar subprogramas.

Tabla 2.2 Elementos Básicos en Ladder

2.2.1.2 Variables internas y bits de sistema

Se suele indicar mediante los caracteres B ó M (en este desarrollo se utilizaron las caracteres E y S) y tienen tanto bobinas como contactos asociados a las mismas de los tipos vistos en el punto anterior. Su número de identificación suele oscilar, en general, entre 0 y 255. Su utilidad fundamental es la de almacenar información intermedia para simplificar esquemas y programación.

Los bits de sistema son contactos que el propio autómata activa cuando conviene o cuando se dan unas circunstancias determinadas. Existe una gran variedad, siendo los más importantes los de arranque y los de reloj, que permiten que empiece la ejecución desde un sitio en concreto y formar una base de tiempos respectivamente. Su nomenclatura es muy diversa, dependiendo siempre del tipo de autómata y fabricante.

2.2.1.3 Temporizadores

El temporizador es un elemento que permite cuentas de tiempo con el fin de activar bobinas pasado un cierto tiempo desde la activación. El esquema básico de un temporizador varía de un autómata a otro, pero siempre se puede encontrar una serie de señales fundamentales, no obstante, con nomenclaturas totalmente distintas. La figura 2.5 es el símbolo mediante el que se denota un temporizador.

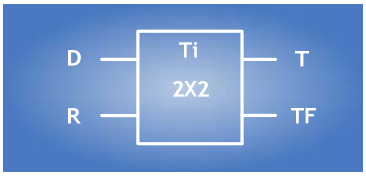


Figura 2.5 Temporizador

En lo sucesivo se manejarán seis tipos de Temporizadores: TempOA, TempOB, TempOC, TempOD, TempOE, TempOG. Los cuales se Explican ampliamente en el Anexo A.

2.2.1.4 Contadores

El contador es un elemento capaz de llevar el cómputo de las activaciones de sus entradas, por lo que nos es útil para memorizar sucesos que no tengan que ver con el tiempo pero que se necesiten realizar un determinado número de veces. La figura 2.6 es el símbolo mediante el que se denota un contador.

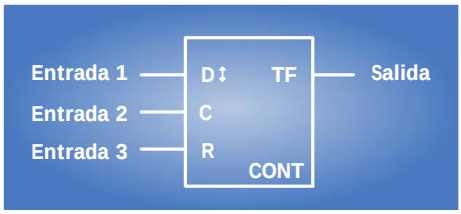


Figura 2.6 Contador

2.2.2 Distribución de un programa

En cuanto a su equivalencia eléctrica, se puede imaginar que la línea vertical de la izquierda representa el terminal de alimentación, mientras que la línea vertical de la derecha representa el terminal de tierra.

El orden de ejecución es generalmente de arriba a abajo y de izquierda a derecha, primero los contactos y luego las bobinas, de manera que al llegar a éstas ya se conoce el valor de los contactos y se activan si procede. El orden de ejecución puede variar de un autómata a otro, pero siempre se respetará el orden de introducción del programa, de manera que se ejecuta primero lo que primero se introduce.

2.2.2.1 Sistemas Combinacionales

Aunque en los sistemas industriales la programación se centra en procesos secuenciales, no teniendo demasiado interés los procesos combinacionales, es necesario conocer la lógica combinacionál ya que en muchas ocasiones es necesaria en la programación secuencial.

Una vez obtenida la función lógica de un problema combinacional, el paso a esquema de contactos es muy sencillo. De acuerdo con el álgebra de Boole aplicada a la conmutación, las sumas serán contactos en paralelo, los productos contactos en serie y las negaciones contactos normalmente cerrados. En la figura 2.7 se muestra un ejemplo de esquema de escalera para una determinada ecuación.

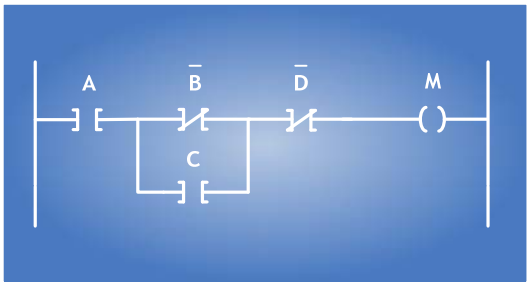


Figura 2.7. Lenguaje escalera para la función M = A(B'+C)D'

2.2.2.2 Sistemas secuenciales

Aunque es posible programar sistemas secuenciales en lenguaje de escalera, sólo se suele utilizar para el control de sistemas sencillos. En aquellos más complejos se utiliza la programación modular o el GRAFCET.

CAPITULO 3
AUTÓMATA IDENTIFICADOR DE ERRORES

En cualquier sistema, es importante contar con algún proceso que pueda identificar los posibles errores que se puedan presentar a largo de su funcionamiento. Las tareas de reconocimiento de errores son realizadas generalmente a través de un proceso ya establecido, lo cual implica un conocimiento total de todas variables permisibles, así como el correcto discernimiento de todas las soluciones disponibles.

En la industria, la mayoría de estos procesos son tareas automatizadas. La automatización es un sistema diseñado para llevar a cabo determinadas tareas anteriormente efectuadas por los seres humanos. Algunos de estos procesos automatizados pueden funcionar de manera independiente o semi-independiente del control humano.

En ingeniería, los sistemas automatizados han permitido diseñar una infinidad de procesos programados, principalmente para el control y guía de múltiples tareas, entre ellas, trabajos cuyo fin es la identificación de errores. Un elemento fundamental en estos sistemas es la creación de entidades llamadas autómatas. Un autómata es un sistema secuencial, como un equipo electrónico programable en lenguaje no informático y diseñado para controlar, en tiempo real y en ambiente industrial, procesos secuenciales.

En este proyecto fue necesaria la creación de un **autómata finito deterministico**. Un autómata finito o máquina de estado finito es un modelo matemático de un sistema que recibe una **cadena** constituida por símbolos de un alfabeto y determina si esa cadena pertenece al **lenguaje** que el autómata reconoce.

El PUMA ESCALERA utiliza un autómata identificador de errores para detectar posibles inconsistencias en la construcción de los módulos lógicos dentro del diagrama de escalera elaborado durante el proceso de **COMPILACIÓN**. Estas inconsistencias implican errores en la estructura esquemática de los módulos lógicos y auxiliares. La estructura de cada módulo es vista a detalle en el Anexo A “Manual del Usuario”.

El autómata no sólo se limita a la detección de errores, sino que genera una interfaz en la que se muestra, mediante texto, el error encontrado. Los mensajes de error generados no serán intuitivos, como los expuestos en lenguajes de programación como C o Java, ya que estos indicarán puntualmente su ubicación, además de su posible solución.

3.1 Estructura del diagrama de escalera en el PUMA ESCALERA

El Diagrama de Escalera es también conocido como ZONA DE TRABAJO en el PUMA ESCALERA. Esta ZONA DE TRABAJO está conformada por una matriz de imágenes, teniendo en su estructura 50 renglones y 10 columnas, contando de esta manera con 500 imágenes. Así por ejemplo, en el renglón uno habrá diez imágenes, colocada cada una en su columna asociada.

Las asociaciones entre el número columnas y el número de renglones a utilizar dependerán del módulo lógico o auxiliar a insertar. El módulo lógico FFARS, por ejemplo, utilizará 20 imágenes de la ZONA DE TRABAJO, es decir, 2 renglones completos consecutivos. El primer renglón de este módulo es el mostrado en la figura 3.1:

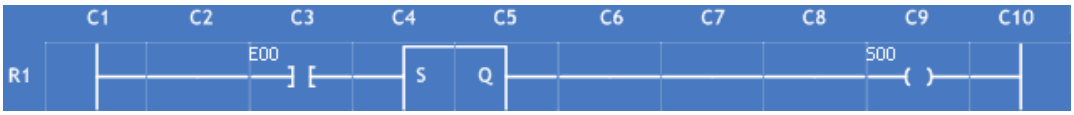


Figura 3.1.- Primer renglón de un FFARS en la ZONA DE TRABAJO

En la figura 3.1 se puede observar la estructura a detalle del primer renglón del FFARS, que se ha insertado en el renglón 1 (R1). En la **columna 1** se encuentra el inicio a partir del "vivo" del diagrama, seguido de un cable que se encuentra en la **columna 2**. La variable de entrada del módulo (incluye variable binaria de entrada) está situada en la **columna 3**, que inmediatamente se conecta con el Set (S) del Flip-Flop en la **columna 4**. Continúa con la salida Q en la **columna 5**, siguiendo con un arreglo de cables que va desde la **columna 6** hasta la **columna 8**. Terminando con la variable de salida en la **columna 9**; esta salida se conecta con el "neutro" del diagrama, ubicado en la **columna 10**.

De la misma manera se puede realizar un análisis del segundo renglón del FFARS:

Figura 3.2.- Segundo renglón de un FFARS en la ZONA DE TRABAJO

Como en el primer renglón, en la **columna 1** se encuentra el inicio a partir del "vivo" del diagrama, seguido de un cable que se encuentra en la **columna 2**. La segunda variable de entrada del módulo (incluye variable binaria de entrada) está situada en la **columna 3**, que inmediatamente se conecta con el Reset (R) del Flip-Flop en la **columna 4**. Continúa con información del módulo en la **columna 5**, siguiendo con un arreglo de imágenes vacías que van desde la **columna 6** hasta la **columna 9**. Conectándose con el "neutro" del diagrama, ubicado en la **columna 10**.

Este análisis se puede realizar con cada uno de los módulos lógicos y auxiliares. Sin embargo sería un proceso muy engorroso y repetitivo. Lo verdaderamente importante a destacar es la manera en que se encuentran desplegadas las imágenes para formar un determinado módulo. La disposición de cada imagen la establece el software, dependiendo de la estructura que se esté configurando, pero que pasaría si por algún motivo, ajeno al software, alguna de estas imágenes no concuerda con el módulo que se esta formando. Esta situación puede darse por diversos motivos, todos ellos relacionados con el usuario, entre los cuales se puede mencionar la sobre-posición de módulos a la hora de incrustarlos en la ZONA DE TRABAJO, la acción de dar clic sobre los renglones que ya tienen incrustados módulos lógicos o auxiliares para insertar un módulo lógico individual como entradas o salidas.

La solución a este tipo de problemas es la inclusión de un proceso durante la **COMPILACIÓN**, en el cual se puedan detectar errores en la estructura de cada módulo lógico insertado. Este proceso debe ser un proceso secuencial o autómata, pues debe

recorrer cada renglón de la ZONA DE TRABAJO, iniciando desde el renglón 1 (R1) hasta el renglón 50 (R50). Para este propósito fue creado el "**AUTÓMATA DETECTOR DE ERRORES**", analizado a continuación.

3.2 Creación del Autómata Detector de Errores

3.2.1 Análisis del Módulo Lógico Flip-Flop Asíncrono FFARS

La creación de un proceso secuencial o autómata implica la concepción de instancias o entidades que sigan una secuencia predeterminada. Estas entidades pueden ser un conjunto de estados, que mediante una cadena de caracteres, se puede situar en ellos, así como "saltar" de uno a otro o establecernos definitivamente en uno, siendo este un estado final.

La cadena de caracteres deberá tener una longitud finita, formada a partir de un alfabeto o conjunto de caracteres que también deberá ser finito. Para este caso, el alfabeto lo formarán las diferentes imágenes que conformen a los módulos lógicos o auxiliares. Para el ejemplo del FFARS, el alfabeto lo formarán (por orden de aparición) las siguientes imágenes:

1.  ,
2.  ,
3.  ,
4. 

,
5. 

,
6.  ,
7.  ,
8.  ,
9.  ,
10.  ,
11.  ,
12.  ,
13.  ,
14. 

,
15. 

,
16.  ,
17.  ,
18.  ,
19.  ,
20. 

De esta manera tenemos un alfabeto de 20 imágenes. El autómata asociado a este alfabeto estaría formado por 21 estados, numerados consecutivamente. El estado inicial (1) "pasará" al estado siguiente (2) mediante la imagen (que fungirá como cadena de caracteres) numerada como 1 de la lista anterior. De esta manera el **estado 2** "pasará" al **estado 3** mediante la imagen numerada como 2 de la lista anterior. Esta secuencia seguirá hasta llegar al estado final (21).

Esta secuencia fue establecida siguiendo la construcción de la estructura del FFARS (Flip-Flop Asíncrono RS), vista en el punto anterior. De esta manera, la columna 1 (C1) del renglón 1 (R1) es la imagen numerada como 1 de la lista anterior, la cual será la cadena de caracteres que nos permitirá ir del **estado 1** al **estado 2**. Enseguida se tomará la columna 2 (C2) del renglón 1 (R1) que es la imagen numerada como 2 de la lista anterior, esta se tomará como la cadena de caracteres que nos permitirá ir del **estado 2** al **estado 3**.

Cuando se llegue a la columna 10, la cadena de caracteres que nos permitirá ir del **estado 11** al **estado 12** es la imagen numerada como 11 de la lista anterior. Esta acción permitirá ir del final del renglón 1 (R1) al inicio del renglón 2 (R2). Este encadenamiento seguirá hasta recorrer toda la estructura del FFARS. Lo cual significa llegar al estado 21 del autómata.

El haber llegado al **estado 21** indica que se ha tenido éxito en el reconocimiento de la estructura de un FFARS. Pero, que pasaría si en alguno de los estados de este autómata no se reconoce una cadena de caracteres, es decir, que sucedería si alguna imagen no concuerda con la estructura del FFARS en algún estado. En este momento se justificaría la existencia de este autómata identificador de errores, pues habrá que crear **estados de error** en cada uno de los 21 estados ya preestablecidos. Las cadenas de caracteres que enviarán a estos estados serán todas las imágenes del alfabeto excepto la imagen asociada con el estado analizado en ese momento. Por ejemplo, para ir del **estado 1** al **estado 2** se necesita únicamente la imagen 1 de la lista anterior, cualquier otra imagen del alfabeto nos llevará a un **estado de error**.

Lo explicado en los cuatro párrafos anteriores es mostrado la figura 4.3 que es un autómata identificador de errores:

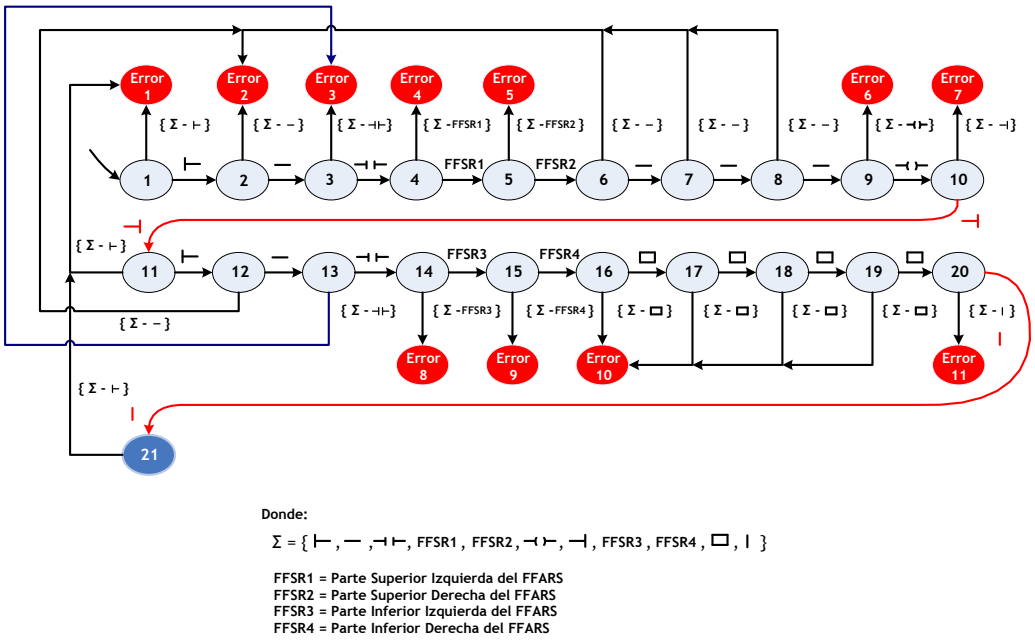


Figura 3.3 – Autómata identificador de errores para un FFARS

Como se muestra en la figura 3.3, este autómata identificador de errores sigue los lineamientos dictados por la definición de un autómata finito (S,Σ,T,s,A), el cual tiene las siguientes características:

- Σ es un alfabeto.
- S es un conjunto de estados.
- T es la función de transición $T : S \times (\Sigma \cup \{\epsilon\}) \to \mathcal{P}(S)$
- $s \in S$ es el **estado inicial**.
- $A \subseteq S$ es un conjunto de **estados** de aceptación o **finales**.

Que aplicados al autómata identificador de errores en la estructura del FFARS resultarían:

- $\Sigma = \{ \vdash , —, \dashv\vdash , \text{FFSR1} , \text{FFSR2} , \dashv\vdash , \dashv , \text{FFSR3} , \text{FFSR4} , \square , \text{I} \}$
- **S** = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, Error 1, Error 2, Error 3, Error 4, Error 5, Error 6, Error 7, Error 8, Error 9, Error 10, Error 11}
- **s** = { 1}
- **A** = { 21}

Como era de esperarse el alfabeto fue modificado de 20 imágenes a sólo 11, ya que varias de ellas se repiten. Esto conlleva a que sólo existirán **11 posibles errores** durante el recorrido de todo el autómata. Como se dijo anteriormente, estos errores se pueden presentar si se encuentran incongruencias en la estructura del FFARS. Cada estado muestra una **transición** a un estado de **error** de acuerdo a las siguientes condiciones.

El Estado 1 → Error 1, si la transición incluye $\{\Sigma - \vdash\}$ (todos los símbolos del alfabeto excepto el símbolo **inicio**).

El Estado 2 → Error 2, si la transición incluye $\{\Sigma - —\}$ (todos los símbolos del alfabeto excepto el símbolo **cable**).

El Estado 3 → Error 3, si la transición incluye $\{\Sigma - \dashv\vdash\}$ (todos los símbolos del alfabeto excepto el símbolo **entrada**).

El Estado 4 → Error 4, si la transición incluye $\{\Sigma - \text{FFSR1}\}$ (todos los símbolos del alfabeto excepto el símbolo **FFSR1**).

El Estado 5 → Error 5, si la transición incluye $\{\Sigma - \text{FFSR2}\}$ (todos los símbolos del alfabeto excepto el símbolo **FFSR2**).

El Estado 6 → Error 2, si la transición incluye $\{\Sigma - —\}$ (todos los símbolos del alfabeto excepto el símbolo **cable**).

El Estado 7 → Error 2, si la transición incluye $\{\Sigma - —\}$ (todos los símbolos del alfabeto excepto el símbolo **cable**).

El Estado 8 → Error 2, si la transición incluye $\{\Sigma - —\}$ (todos los símbolos del alfabeto excepto el símbolo **cable**).

El Estado 9 → Error 6, si la transición incluye $\{\Sigma - \dashv\vdash\}$ (todos los símbolos del alfabeto excepto el símbolo **salida**).

El Estado 10 → Error 7, si la transición incluye $\{\Sigma - \dashv\}$ (todos los símbolos del alfabeto excepto el símbolo **fin**).

El Estado 11 → Error 1, si la transición incluye $\{\Sigma - \vdash\}$ (todos los símbolos del alfabeto excepto el símbolo **inicio**).

El Estado 12 → Error 2, si la transición incluye $\{\Sigma - —\}$ (todos los símbolos del alfabeto excepto el símbolo **cable**).

El Estado 13 → Error 3, si la transición incluye $\{\Sigma - \dashv\vdash\}$ (todos los símbolos del alfabeto excepto el símbolo **entrada**).

El Estado 14 → Error 8, si la transición incluye $\{\Sigma - \text{FFSR3}\}$ (todos los símbolos del alfabeto excepto el símbolo **FFSR3**).

El Estado 15 → Error 9, si la transición incluye $\{\Sigma - \text{FFSR4}\}$ (todos los símbolos del alfabeto excepto el símbolo **FFSR4**).

El Estado 16 → Error 10, si la transición incluye $\{\Sigma - \square\}$ (todos los símbolos del alfabeto excepto el símbolo **vacío**).

El Estado 17 → Error 10, si la transición incluye $\{\Sigma - \square\}$ (todos los símbolos del alfabeto excepto el símbolo **vacío**).

El Estado 18 → Error 10, si la transición incluye $\{\Sigma - \square\}$ (todos los símbolos del alfabeto excepto el símbolo **vacío**).

El Estado 19 → Error 10, si la transición incluye $\{\Sigma - \square\}$ (todos los símbolos del alfabeto excepto el símbolo **vacío**).

El Estado 20 → Error 11, si la transición incluye $\{\Sigma - |\}$ (todos los símbolos del alfabeto excepto el símbolo **neutro**).

El Estado 21 → Error 1, si la transición incluye $\{\Sigma - \vdash\}$ (todos los símbolos del alfabeto excepto el símbolo **inicio**).

Con la transición hacia estos estados de error el autómata cumple su objetivo en la detección de errores. La llegada a estas instancias generará en la interfaz del PUMA ESCALERA una ventana que mostrará al usuario el tipo de ERROR que se ha encontrado en la estructura del FFARS.

3.2.1.1 Realimentación En El Autómata Identificador De Errores.

Ahora imagine que en la ZONA DE TRABAJO se encuentran dos módulos lógicos FFARS consecutivos. El primero se encuentra en los dos primeros renglones (R1 Y R2), el segundo de ellos esta ubicado en los dos renglones siguientes (R3 y R4). Ahora suponga que el proceso de **COMPILACIÓN** ha iniciado y ha llegado a la etapa en la actúa el autómata identificador de ERRORES.

El autómata sólo identificaría el primer módulo lógico, ya que no tendría manera de llegar al siguiente renglón, para ser más específicos al renglón 3 (R3). El **estado 21** es el estado final del autómata, una vez que se ha llegado hasta esta entidad no habría forma de trasladarse hacia otro estado.

La solución a éste problema es la realimentación del autómata. El ciclo o bucle de realimentación es un proceso que detecta una cadena en el estado, la compara con una ya establecida y realiza alguna acción preprogramada necesaria para mantener al autómata funcionando correctamente. En este caso, el ciclo comenzaría en el **estado 21** (que es el estado final) e iría hacia el **estado 2** mediante el símbolo " ⊢". Esta instancia es donde comienza el análisis de la estructura del FFARS, lógicamente, después de haber detectado el símbolo " ⊢". Con el ciclo de realimentación el autómata detector de errores quedaría como se muestra en la figura 3.4:

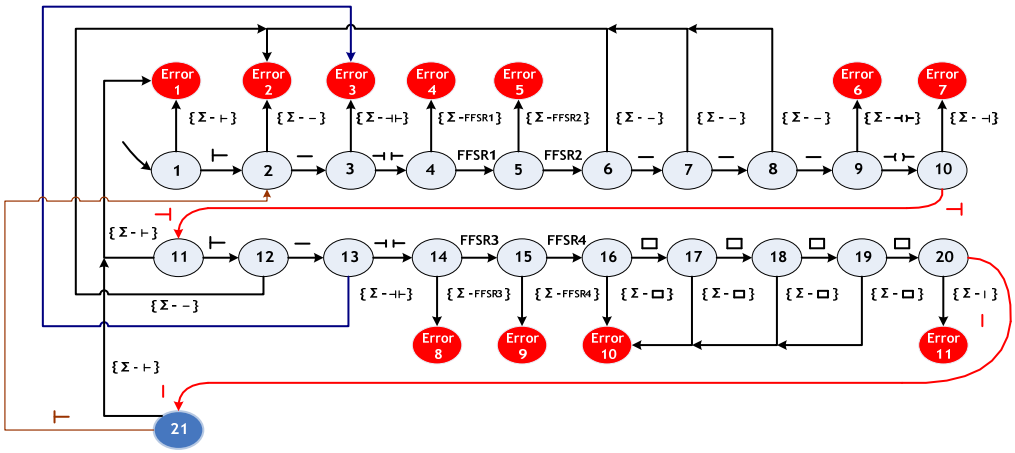


Figura 3.4 – Autómata identificador de errores para un FFARS con realimentación

Con la colocación del ciclo de realimentación cada estado del autómata puede trasladarse hacia otro de acuerdo a su correspondiente cadena de caracteres. Incluso, si se ha llegado a un estado de ERROR se generaría una interfaz en la cual el usuario identificaría el tipo de error encontrado, que una vez resuelto, se comenzaría nuevamente el proceso de **COMPILACIÓN** iniciando una vez más el recorrido del autómata a través del estado 1.

Sería conveniente entonces, con todos los datos recaudados realizar una tabla de transiciones, donde se identifique claramente la función de transición de cada estado. De esta manera se completaría la quíntupla que define a un autómata finito.

SÍMBOLO→ ESTADO	⊢	—	⊣ ⊢	FFSR1	FFSR2	⊣ ⊢	⊣	FFSR3	FFSR4	□	|
1	**2**	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1
2	Error 2	**3**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
3	Error 3	Error 3	**4**	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3
4	Error 4	Error 4	Error 4	**5**	Error 4	Error 4	Error 4	Error 4	Error 4	Error 4	Error 4
5	Error 5	Error 5	Error 5	Error 5	**6**	Error 5	Error 5	Error 5	Error 5	Error 5	Error 5
6	Error 2	**7**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
7	Error 2	**8**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
8	Error 2	**9**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
9	Error 6	Error 6	Error 6	Error 6	Error 6	**10**	Error 6	Error 6	Error 6	Error 6	Error 6
10	Error 7	Error 7	Error 7	Error 7	Error 7	Error 7	**11**	Error 7	Error 7	Error 7	Error 7
11	**12**	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1
12	Error 2	**13**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
13	Error 3	Error 3	**14**	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3
14	Error 8	Error 8	Error 8	**15**	Error 8	Error 8	Error 8	Error 8	Error 8	Error 8	Error 8
15	Error 9	Error 9	Error 9	Error 9	**16**	Error 9	Error 9	Error 9	Error 9	Error 9	Error 9
16	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	**17**	Error10
17	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	**18**	Error10
18	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	**19**	Error10
19	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	**20**	Error10
20	Error11	Error11	Error11	Error11	Error11	Error11	Error11	Error11	Error11	Error11	**21**
21	**1**	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1
Error 1	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 2	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 3	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 4	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 5	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 6	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 7	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 8	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 9	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 10	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										
Error 11	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1										

Tabla 3.1 Tabla de Transiciones del autómata identificador de errores del FFARS

Con la tabla 4.1 de transiciones se completa el análisis del autómata identificador de errores en la estructura esquemática del módulo lógico Flip-Flop Asíncrono SR (FFARS). Este tipo de estudio metódico es aplicable a otros módulos lógicos que permite realizar el PUMA ESCALERA. Sería adecuado entonces analizar otro módulo lógico cuya estructura fundamental sea distinta a la del FFARS.

3.2.2 Análisis De Una Composición De Compuertas Lógicas

El módulo lógico seleccionado es una composición de tres compuertas lógicas. Las dos primeras serán compuertas lógicas OR, de dos entradas cada una. Las salidas de las compuertas OR serán las entradas de la tercera compuerta lógica, una compuerta AND. Esta estructura ocupa dos renglones de la ZONA DE TRABAJO en el PUMA ESCALERA. El primero de ellos es el mostrado en la figura 3.5:

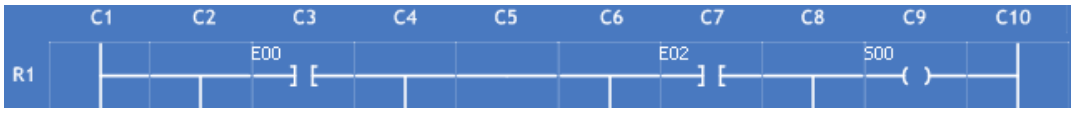


Figura 3.5 – Primer renglón de una composición de compuertas lógicas en la zona de trabajo.

En la figura 3.5 se puede observar la estructura a detalle del primer renglón de la composición de compuertas, que se ha insertado en el renglón 1 (R1). En la **columna 1** se encuentra el inicio a partir del "vivo" del diagrama, seguido de un conector perpendicular que se encuentra en la **columna 2**. La primera variable de entrada de la primera compuerta OR (incluye variable binaria de entrada) esta situada en la **columna 3**, que inmediatamente se conecta con un nuevo conector perpendicular en la **columna 4**. Continúa con un símbolo cable en la **columna 5**, siguiendo con un conector perpendicular en la **columna 6** seguido de la primera variable de entrada de la segunda compuerta OR (incluye variable binaria de entrada) situada en la **columna 7**. Continuando con un conector perpendicular en la **columna 8**. Terminando con la variable de salida en la **columna 9**; esta salida se conecta con el "neutro" del diagrama, ubicado en la **columna 10**.

De la misma manera se puede realizar un análisis del segundo renglón del FFARS(figura 3.6):

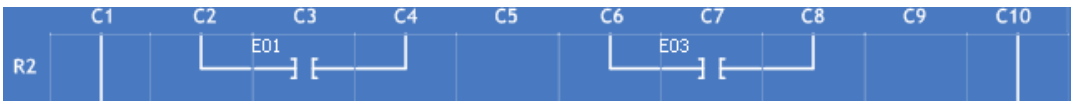
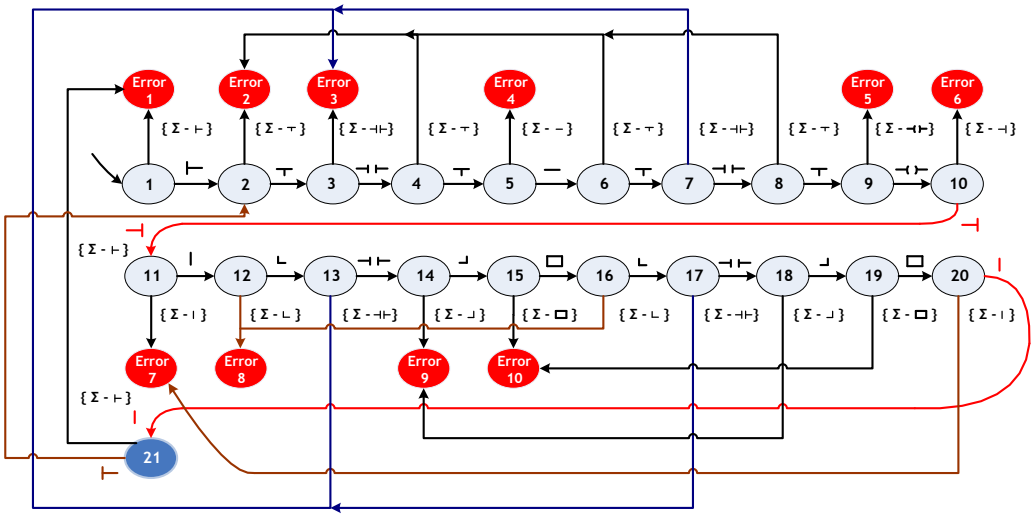


Figura 3.6 – Segundo renglón de una composición de compuertas lógicas en la zona de trabajo.

En la **columna 1** se encuentra el "vivo" del diagrama, seguido de un codo derecho que se encuentra en la **columna 2**. La segunda variable de entrada de la primera compuerta OR (incluye variable binaria de entrada) esta situada en la **columna 3**, que inmediatamente se conecta con un codo izquierdo en la **columna 4**. Continúa con un símbolo vacío en la **columna 5**, siguiendo con un codo derecho en la **columna 6** seguido de la segunda variable de entrada de la segunda compuerta OR (incluye variable binaria de entrada) situada en la **columna 7**. Continuando con un codo izquierdo en la **columna 8**. Siguiendo con un símbolo vacío en la **columna 9**; terminando con el "neutro" del diagrama, ubicado en la **columna 10**.

El autómata identificador de errores asociado es el mostrado por la figura 3.7:



$\Sigma = \{ \vdash, \top, \dashv\vdash, —, \dashv\vdash, \dashv, \lfloor, \llcorner, \lrcorner, \square \}$

Figura 3.7 - Autómata identificador de errores para una composición de compuertas lógicas con realimentación

Siguiendo los lineamientos de la definición de un autómata finito (S,Σ,T,s,A), vistos en el punto anterior, este autómata tiene las siguientes características:

- $\Sigma = \{ \vdash, \top, \dashv\vdash, —, \dashv\vdash, \dashv, \lfloor, \llcorner, \lrcorner, \square \}$
- **S** = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, Error 1, Error 2, Error 3, Error 4, Error 5, Error 6, Error 7, Error 8, Error 9, Error 10}
- **s** = { 1}
- **A** = { 21}

Este autómata, además, ya cumple con un ciclo de realimentación, por lo que la tabla de transiciones ya cuenta con todos los requerimientos necesarios:

SÍMBOLO→ *ESTADO*	⊢	⊤	⊣⊢	—	⊣⊢	⊣	|	⌞	⌟	□
1	**2**	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1
2	Error 2	**3**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
3	Error 3	Error 3	**4**	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3
4	Error 2	**5**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
5	Error 4	Error 4	Error 4	**6**	Error 4	Error 4	Error 4	Error 4	Error 4	Error 4
6	Error 2	**7**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
7	Error 3	Error 3	**8**	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3
8	Error 2	**9**	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2	Error 2
9	Error 5	Error 5	Error 5	Error 5	**10**	Error 5	Error 5	Error 5	Error 5	Error 5
10	Error 6	Error 6	Error 6	Error 6	Error 6	**11**	Error 6	Error 6	Error 6	Error 6
11	Error 7	Error 7	Error 7	Error 7	Error 7	Error 7	**12**	Error 7	Error 7	Error 7
12	Error 8	Error 8	Error 8	Error 8	Error 8	Error 8	Error 8	**13**	Error 8	Error 8
13	Error 3	Error 3	**14**	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3
14	Error 9	Error 9	Error 9	Error 9	Error 9	Error 9	Error 9	Error 9	**15**	Error 9
15	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	**16**
16	Error 8	Error 8	Error 8	Error 8	Error 8	Error 8	Error 8	**17**	Error 8	Error 8
17	Error 3	Error 3	**18**	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3	Error 3
18	Error 9	Error 9	Error 9	Error 9	Error 9	Error 9	Error 9	Error 9	**19**	Error 9
19	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	Error10	**20**
20	Error 7	Error 7	Error 7	Error 7	Error 7	Error 7	**21**	Error 7	Error 7	Error 7
21	**1**	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1	Error 1
Error 1	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 2	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 3	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 4	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 5	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 6	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 7	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 8	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 9	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									
Error 10	Genera una Interfaz con un Mensaje de Error, comienza de nuevo el recorrido del autómata → 1									

Tabla 3.2 Tabla de Transiciones del autómata identificador de errores de la composición de tres compuertas lógicas

Con esta tabla de transiciones se completa el análisis del autómata identificador de errores en la estructura esquemática de una composición de tres compuertas lógicas (2 compuertas OR y una compuerta AND).

A través del análisis del autómata identificador de errores del FFARS y del autómata identificador de errores de una composición de compuertas lógicas, se ha demostrado el uso eficiente que otorgarán a la hora de identificar errores en la estructura de módulos lógicos.

El uso de los autómatas dependerá de la estructura de cada módulo, esta elección representaría a largo plazo una pérdida de tiempo pues cada módulo lógico contará con un autómata identificador de errores y cada vez que se deseará encontrar alguna incongruencia en algún módulo, el compilador tendría que realizar un recorrido a través de todos los autómatas existentes, seleccionando una estructura que concuerde con la que se esté buscando. Esto implicaría la pérdida de varios segundos, pues seguramente en el proyecto existirán varios módulos lógicos, por consiguiente, para cada módulo se hará una búsqueda para determinar que autómata identificador de errores le correspondería.

La solución para disminuir la cantidad de búsquedas durante la etapa de reconocimiento de errores sería la construcción de un **único** AUTÓMATA IDENTIFICADOR DE ERRORES que detecte las incongruencias de todos los módulos lógicos existentes.

En consecuencia, el autómata generado para el FFARS, así como el autómata creado para la composición de COMPUERTAS se acoplarán en un único autómata identificador de errores, que detectará ambas estructuras durante su recorrido, por consiguiente encontrará también todos los posibles errores:

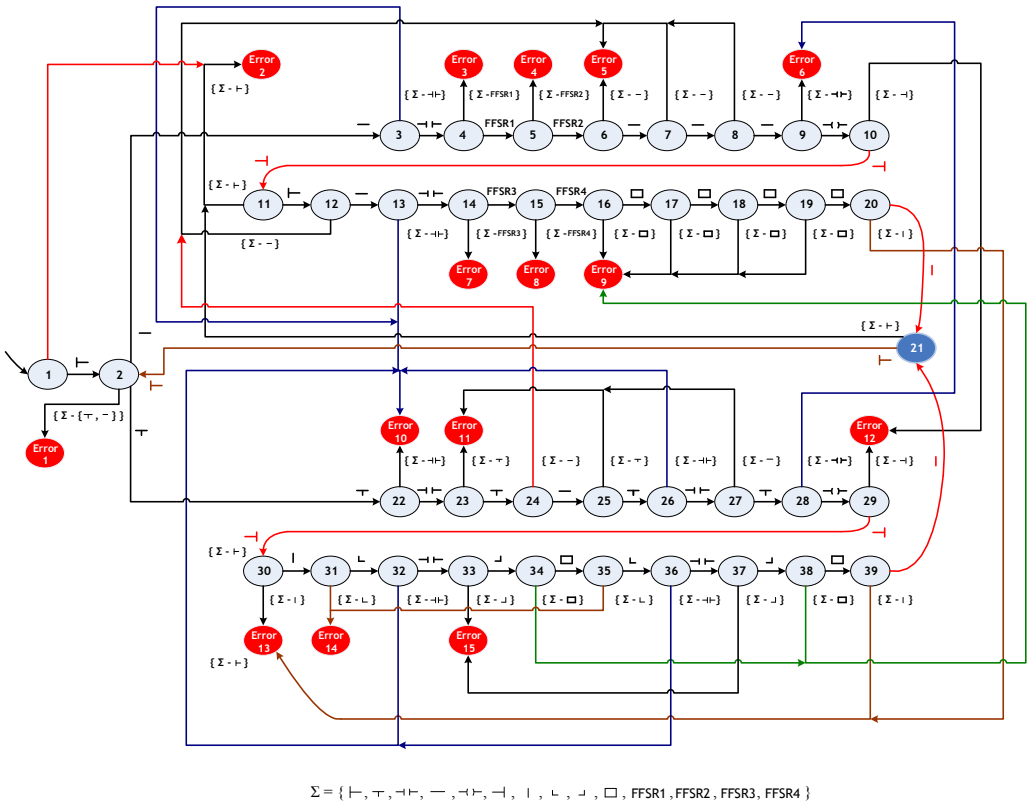


Figura 3.8 Autómata identificador de errores para un FFARS y para una composición de compuertas lógicas con realimentación.

El autómata detector de errores mostrado en la figura 3.8 tiene **39 estados**, si se toma en cuenta de que el recorrido lo realizaran módulos lógicos con estructuras correctas. De estos 39 estados destacan los **dos primeros**, ya que en estos estados se encuentra la bifurcación que decidirá el camino que se seguirá en el autómata, todo esto dependiendo de la estructura que se esté examinando, en este caso el análisis del FFARS comienza en el estado 2, siguiendo hacia el **estado 3** con el símbolo "—" y terminando en el estado 21. De manera similar para la composición de compuertas lógicas el estudio comienza en el estado 2, siguiendo hacia el **estado 22** con el símbolo "⊤" y terminando en el estado 21.

Este autómata cuenta con 15 estados finales de ERROR, los cuales se obtuvieron en principio, de la suma de los estados finales de error de los autómatas analizadores de los módulos lógicos **FFARS** y el conjunto de compuertas lógicas respectivos. El resultado real de la adición fue de 21 estados finales de error, sin embargo, se eliminaron 6 de ellos, ya que varias transiciones hacia estos eran equivalentes, lo que provocaba redundancia en la detección de errores en cada estructura.

En resumen, se tiene un autómata detector de errores con 54 estados, siendo uno de ellos inicial y 16 que son finales o de aceptación. Además de un alfabeto de **15 elementos** (mostrados en la parte inferior de la figura 3.2.6), que al igual que los estados finales de error, son una selección de símbolos de los dos autómatas antes mencionados, siguiendo un criterio de no repetición. En cuanto a la tabla de transiciones sería intrascendente desplegarla ya que su contenido es similar a las dos tablas anteriores,

De esta manera, el autómata detector de errores creado hasta ahora, crecerá con los criterios definidos hasta el momento, respecto a cada módulo lógico añadido.

3.3 Características del Autómata Detector de Errores Final

El autómata detector de errores final analiza 15 módulos, 13 lógicos y 2 auxiliares. Además de una serie de composiciones de compuertas lógicas (21 posibles combinaciones).

Por obvias razones, el autómata resultante, crece a gran escala. El número de estados aumenta de **54** (último autómata creado) a **336**. El Estado 1 es el único **estado inicial**. Existen 2 estados finales de aceptación (estado 11 y estado 21.5) y 34 estados finales de **ERROR**.

Los estados del autómata detector de errores no están numerados consecutivamente. Esta construido de tal manera que la numeración esta relacionada con el orden vertical que guardan los estados. Así por ejemplo, los dos **estados finales de aceptación** tienen el número 11 y el número 21.5 respectivamente, es decir, el primer estado de aceptación se encuentra en la columna numerada como 11, mientras que el segundo se localiza en la columna numerada como 21. Para una mejor identificación, estos estados se encuentran representados mediante el color azul, como se muestra en la figura 3.3.1

En la figura 3.3.1 se puede apreciar claramente la disposición de todos los estados. Este autómata es similar a una gran matriz, por lo menos de una manera visual, ya que tiene renglones y columnas. De esta manera podemos encontrar al **estado 1** en la columna 1, al **estado 2** en la columna 2 y así sucesivamente. El número de columnas es 33. Los elementos de cada columna están numerados de forma decimal o incluso mediante letras, así por ejemplo, en la columna 5 se pueden encontrar cadenas de caracteres como 5.2, 5.3 y/o 5.C. La notación punto ayudará en la identificación de un elemento en una cierta columna.

Sin embargo, la notación en los nombres de los estados es poco relevante en la construcción del autómata. Lo verdaderamente importante es seguir las transiciones que siguen a través de ellos. El recorrido a través de éstos determinará el reconocimiento de la estructura que se esté analizando, así como los posibles errores que contenga.

La figura 3.3.1 no muestra los estados finales de ERROR, ya que su inclusión en la misma imagen provocaría una confusión visual, pues las transiciones (flechas en la figura) hacia ellos impedirían observar de manera eficiente el recorrido que se tiene que llevar a cabo en cada estructura.

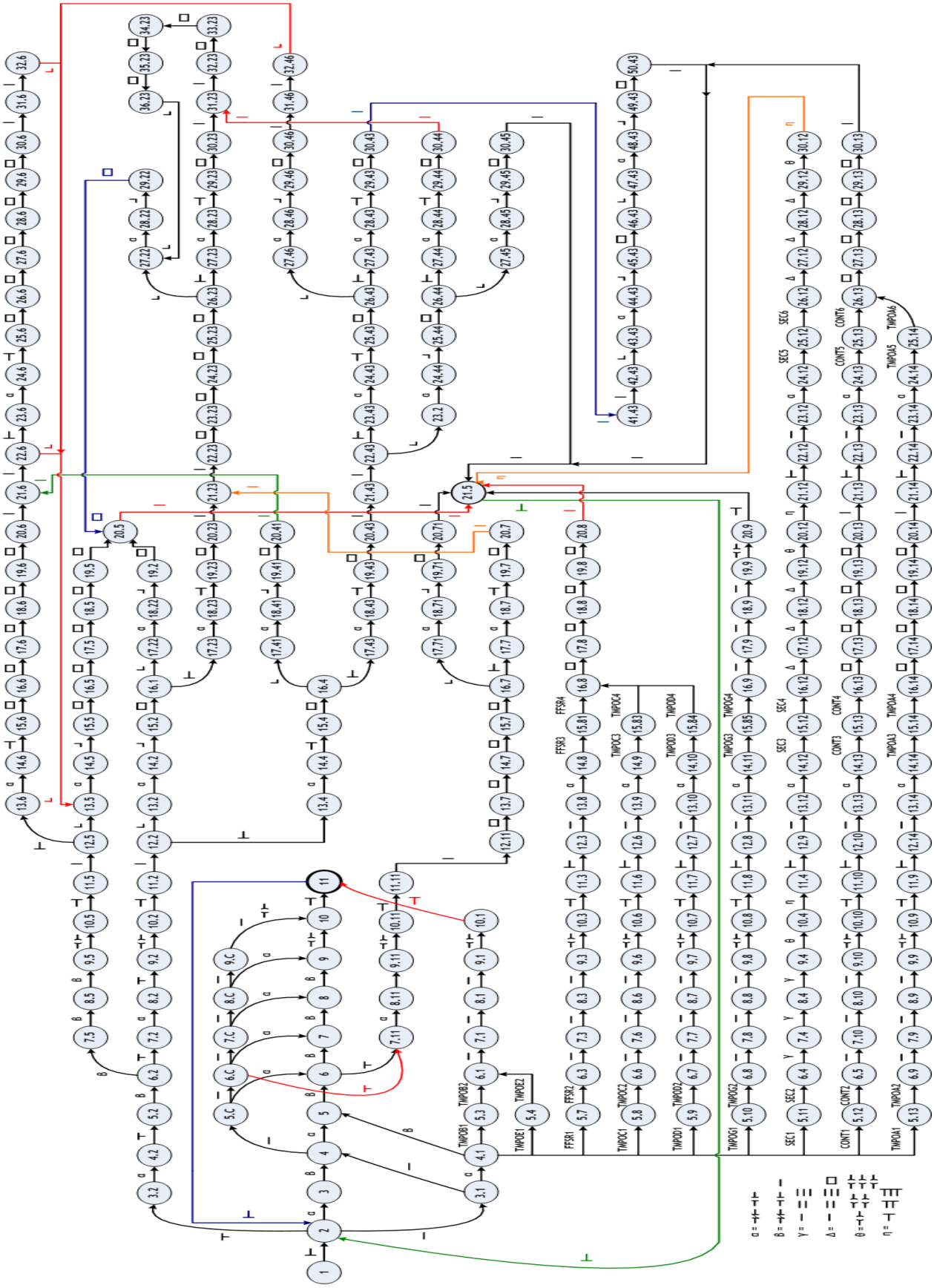


Figura 3.9 Autómata detector de errores.

3.3.1 Estados de Error del Autómata Detector de Errores Final

Para evitar **inconsistencias** en el correcto entendimiento del autómata se muestra a continuación una tabla con todos los estados, así como la transición a su correspondiente estado final de ERROR:

ESTADOS DONDE SUCEDE EL ERROR	TRANSICIÓN AL CORRESPONDIENTE ESTADO DE ERROR
Estado 1, Estado 2, Estado 3, Estado 3.2, Estado 4, Estado 5, Estado 5.c, Estado 5.2, Estado 6, Estado 6.c, Estado 6.2, Estado 7, Estado 7.c, Estado 7.5, Estado 8, Estado 8.c, Estado 8.2, Estado 8.5, Estado 8.11, Estado 13.2, Estado 13.4, Estado 13.5, Estado 13.6, Estado 13.8, Estado 13.9, Estado 13.10, Estado 13.11, Estado 13.12, Estado 13.13, Estado 13.14, Estado 21.5, Estado 21.13, Estado 21.14, Estado 23.12	**Error 1**: Símbolo Erróneo
Estado 3.1	**Error 2**: Después del cable debes insertar una entrada, un cable o el inicio de un TempOB.
Estado 4.1, Estado 4.2	**Error 3**: Debería existir un bloque un símbolo de entrada o cable.
Estado 4.3, Estado 5.3	**Error 4**: Falta parte del bloque TempOB.
Estado 5.4	**Error 5**: Falta parte del bloque TempOE.
Estado 5.7, Estado 14.8, Estado 15.81	**Error 6**: Falta parte del bloque FFARS.
Estado 5.8, Estado 14.9, Estado 15.83	**Error 7**: Falta parte del bloque TempOC.
Estado 5.9, Estado 14.10, Estado 15.84	**Error 8**: Falta parte del bloque TempOD.
Estado 5.10, Estado 14.11, Estado 15.85	**Error 9**: Falta parte del bloque TempOG.
Estado 5.11	**Error 10**: Falta la parte superior derecha del secuenciador.
Estado 5.12, Estado 14.13, Estado 15.13, Estado 24.13, Estado 25.13	**Error 11**: Falta parte del bloque contador.
Estado 6.1, Estado 6.3, Estado 6.5, Estado 6.6, Estado 6.7, Estado 6.8, Estado 6.9, Estado 7.1, Estado 7.3, Estado 7.6, Estado 7.7, Estado 7.8 Estado 7.9, Estado 7.10, Estado 8.1, Estado 8.3, Estado 8.6, Estado 8.7, Estado 8.8, Estado 8.9, Estado 8.10, Estado 12.3, Estado 12.6, Estado 12.7, Estado 12.8, Estado 12.9, Estado 12.10, Estado 12.14, Estado 16.9, Estado 17.9, Estado	**Error 12**: Debería haber un cable.

18.9, Estado 22.12, Estado 22.13, Estado 22.14	
Estado 7.2, Estado 7.11, Estado 17.22, Estado 17.23, Estado 17.41, Estado 17.43, Estado 17.7, Estado 17.71, Estado 23.2, Estado 23.43, Estado 23.6, Estado 23.13, Estado 23.14, Estado 27.22, Estado 27.23, Estado 27.43, Estado 27.46, Estado 27.44, Estado 27.45, Estado 30.13, Estado 43.43	**Error 13**: Debes ingresar una entrada, no puedes poner un cable en ese sitio.
Estado 9, Estado 9.1, Estado 9.2, Estado 9.3, Estado 9.5, Estado 9.6, Estado 9.7, Estado 9.8, Estado 9.9, Estado 9.10, Estado 9.11, Estado 19.9	**Error 14**: Debes ingresar una salida, no puedes poner un cable en ese sitio.
Estado 9.c	**Error 15**: Debes ingresar una entrada o un cable.
Estado 10, Estado 10.1, Estado 10.2, Estado 10.3, Estado 10.5, Estado 10.6, Estado 10.7, Estado 10.8, Estado 10.9, Estado 10.10, Estado 10.11, Estado 10.12, Estado 20.23, Estado 20.41, Estado 20.43, Estado 20.5, Estado 20.6, Estado 20.7, Estado 20.71, Estado 20.8, Estado 20.9, Estado 20.13, Estado 20.14, Estado 30.23, Estado 30.43, Estado 30.46, Estado 30.44, Estado 30.45, Estado 30.6, Estado 30.13	**Error 16**: Se esperaba un fin de línea.
Estado 11, Estado 11.2, Estado 11.3, Estado 11.4, Estado 11.5, Estado 11.6, Estado 11.7, Estado 11.8, Estado 11.9, Estado 11.10, Estado 11.11, Estado 21.23, Estado 21.43, Estado 21.6, Estado 21.12, Estado 31.23, Estado 31.46, Estado 31.6, Estado 31.43	**Error 17**: Se esperaba el inicio de un renglón.
Estado 12.2, Estado 12.5, Estado 14.2, Estado 14.4, Estado 14.5, Estado 14.6, Estado 16.1, Estado 16.4, Estado 16.7, Estado 18.22, Estado 18.23, Estado 18.41, Estado 18.43, Estado 18.7, Estado 18.71, Estado 22.43, Estado 22.6, Estado 24.43, Estado 24.44, Estado 24.6, Estado 26.23, Estado 26.43, Estado 26.44, Estado 28.22, Estado 28.23, Estado 28.43, Estado 28.46, Estado 28.44, Estado 28.45, Estado 32.46, Estado 32.6, Estado 42.43, Estado 44.43, Estado 36.23, Estado 46.43	**Error 18**: Se esperaba símbolo de conector.
Estado 2.1, Estado 6.10, Estado 7.12, Estado 8.12, Estado 9.12, Estado 12.11, Estado 13.7, Estado 14.7, Estado 15.2, Estado 15.4, Estado 15.5, Estado 15.6, Estado 15.7, Estado 16.5, Estado 16.6, Estado 16.8, Estado 16.13, Estado 16.14, Estado 17.5, Estado 17.6, Estado 17.8, Estado 17.13, Estado 17.14, Estado 18.5, Estado 18.6, Estado 18.8, Estado 18.13, Estado 18.14, Estado 19.2, Estado 19.23, Estado 19.41, Estado 19.43, Estado 19.5, Estado 19.6, Estado 19.7,	**Error 19**: No se esperaba ningún símbolo.

Estado 19.71, Estado 19.8, Estado 19.13, Estado 19.14, Estado 22.23, Estado 23.23, Estado 24.23, Estado 25.23, Estado 25.43, Estado 25.44, Estado 25.6, Estado 26.6, Estado 26.13, Estado 27.6, Estado 27.13, Estado 28.6, Estado 28.13, Estado 29.22, Estado 29.23, Estado 29.43, Estado 29.46, Estado 29.44, Estado 29.45, Estado 29.6, Estado 29.13, Estado 32.23, Estado 33.23, Estado 34.23, Estado 35.23, Estado 45.43	
Estado 5.13, Estado 14.14, Estado 15.14, Estado 24.14, Estado 25.14	**Error 20**: Falta parte del bloque TempOA.
Estado 3.3	**Error 21**: Falta 1era. parte del mensajero.
Estado 4.4	**Error 22**: Falta 2da. parte del mensajero.
Estado 5.15	**Error 23**: Falta 3era. parte del mensajero.
Estado 5.14	**Error 24**: Falta 2da. parte de la alarma.
Estado 6.4, Estado 7.4, Estado 8.4, Estado 16.12, Estado 17.12, Estado 18.12	**Error 25**: Falta(n) cable(s) del secuenciador.
Estado 9.4, Estado 19.12	**Error 26**: Falta(n) salida(s) del secuenciador.
Estado 10.4, Estado 20.12	**Error 27**: Falta(n) el(los) fin(es) del secuenciador.
Estado 14.12	**Error 28**: Falta la parte media izquierda del secuenciador.
Estado 15.12	**Error 29**: Falta la parte media derecha del secuenciador.
Estado 24.12	**Error 30**: Falta la parte inferior izquierda del secuenciador.
Estado 25.12	**Error 31**: Falta la parte inferior derecha del secuenciador.
Estado 26.12, Estado 27.12, Estado 28.12	**Error 32**: Falta(n) cable(s) del secuenciador o se esperaban símbolos vacíos, si su secuenciador tiene 1 salida.
Estado 29.12	**Error 33**: Falta(n) salidas(s) del secuenciador o se esperaban símbolos vacíos, si su secuenciador tiene 1 salida.
Estado 30.12	**Error 34**: Falta(n) el(los) fin(es) del secuenciador o se esperaban símbolos vacíos, si su secuenciador tiene 1 salida.

Tabla 3.C. Tabla que muestra la correspondiente transición a un determinado Estado de Error de cada estado

Nota: Para encontrar algún estado en específico en la tabla anterior, puede utilizar la herramienta de Microsoft Word: CTRL + B.
De nada sirve mostrar los estados de error si no se muestra su correspondiente solución. Una vez que aparezca el mensaje de error en la interfaz del programa, estos son los sucesivos pasos de resolución a realizar en cada caso.

Soluciones:

Error 1: Símbolo Erróneo. Este tipo de error sucede en compuertas lógicas. Pues en lugar de existir entradas (o entradas negadas) en la columna C3, pueden hallarse símbolos vacíos  o símbolos cable  . La solución es la inserción de entradas o entradas negadas en estos lugares mediante las herramientas convenientes.

Error 2: Hay símbolos diferentes a un cable, una entrada o un inicio del temporizador OB. La solución es la inserción de estos símbolos. Un cable en el caso de una compuerta lógica, una entrada en caso de un módulo ALARMA o una compuerta lógica. Pero si se trata de un símbolo de inicio en el temporizador OB, hay que ingresar de nuevo el módulo completo.

Error 3: Debería existir un bloque un símbolo de entrada o cable. La solución es la inserción de un cable en el caso de una compuerta lógica y una entrada en caso de un módulo ALARMA o una compuerta lógica.

Error 4: Falta parte del bloque TempOB. La solución es el reingreso del temporizador OB configurado anteriormente, en el renglón donde se encontró el error.

Error 5: Falta parte del bloque TempOE. La solución es el reingreso del temporizador OE configurado anteriormente, en el renglón donde se encontró el error.

Error 6: Falta parte del bloque FFARS. La solución es el reingreso del módulo FFARS configurado anteriormente, en el renglón donde se encontró el error.

Error 7: Falta parte del bloque TempOC. La solución es el reingreso del temporizador OC configurado anteriormente, en el renglón donde se encontró el error.

Error 8: Falta parte del bloque TempOD. La solución es el reingreso del temporizador OD configurado anteriormente, en el renglón donde se encontró el error.

Error 9: Falta parte del bloque TempOG. La solución es el reingreso del temporizador OG configurado anteriormente, en el renglón donde se encontró el error.

Error 10: Falta la parte superior derecha del secuenciador. La solución es el reingreso del módulo SECUENCIADOR configurado anteriormente, en el renglón donde se encontró el error.

Error 11: Falta parte del bloque contador. La solución es el reingreso del módulo CONTADOR configurado anteriormente, en el renglón donde se encontró el error.

Error 12: Debería haber un cable. Este error sucede en módulos temporizadores, secuenciadores y contadores. Este tipo de entidades tienen estructuras determinadas, por lo son sensibles a cualquier símbolo ajeno a su construcción. La solución es la inserción de un símbolo cable donde haya sucedido el error.

Error 13: Debes ingresar una entrada, no puedes poner un cable en ese sitio. Este error sucede en compuertas lógicas, específicamente en las columnas C3 y C7, ya que por diversas circunstancias pudo haber sido incrustado un símbolo cable en estas regiones. La solución es la inserción de una entrada (o entrada negada) donde haya sucedido el error.

Error 14: Debes ingresar una salida, no puedes poner un cable en ese sitio. Este tipo de error es muy común en todos los módulos lógicos y auxiliares, no hay que olvidar que todas las salidas deben obligatoriamente ser ubicadas en la columna C9, por lo tanto cualquier otro símbolo en esta región provocara un mensaje de error. La solución es la inserción de una salida donde haya sucedido el error.

Error 15: Debes ingresar una entrada o un cable. Este tipo de error sucede cuando se tiene en la columna C9 específicamente un símbolo entrada. La solución es la inserción de una salida o incluso un símbolo cable en donde haya sucedido el error.

Error 16: Se esperaba un fin de línea. Los símbolos de fin de línea le indican al compilador que un renglón ha terminado, si estos símbolos no existieran el compilador simplemente no podría seguir analizando la estructura en cuestión. La solución es el borrado de toda la estructura mediante la herramienta BORRAR. Por ende, habrá que reingresar y configurar correctamente la estructura que provocó el error.

Error 17: Se esperaba el inicio de un renglón. Los símbolos de inicio de línea le indican al compilador que un renglón ha iniciado, si estos símbolos no existieran el compilador simplemente no podría seguir analizando la estructura en cuestión. La solución es el borrado de toda la estructura mediante la herramienta BORRAR. Por ende, habrá que reingresar y configurar correctamente la estructura que provocó el error.

Error 18: Se esperaba símbolo de conector. Estos errores suceden en compuertas lógicas. Este tipo de construcciones necesitan varios símbolos de conexión que definen su estructura, si falta alguna de ellas se enviará un mensaje de error. La solución es el borrado de toda la estructura mediante la herramienta BORRAR. Por ende, habrá que reingresar y configurar correctamente la estructura que provocó el error.

Error 19: No se esperaba ningún símbolo. Los módulos que ocupan más de un reglón en la ZONA DE TRABAJO generalmente tienen símbolos vacíos predeterminados en su estructura. Si se llega a corromper esta secuencia predeterminada de construcción se enviará el mensaje de error. La solución es el borrado de toda la estructura mediante la herramienta BORRAR. Por ende, habrá que reingresar y configurar correctamente la estructura que provocó el error.

Error 20: Falta parte del bloque TempOA. La solución es el reingreso del temporizador OA configurado anteriormente, en el renglón donde se encontró el error.

Error 21: Falta 1era. parte del mensajero. La solución es el reingreso del módulo mensajero configurado anteriormente, en el renglón donde se encontró el error.

Error 22: Falta 2da. parte del mensajero. La solución es el reingreso del módulo mensajero configurado anteriormente, en el renglón donde se encontró el error.

Error 23: Falta 3era. parte del mensajero. La solución es el reingreso del módulo mensajero configurado anteriormente, en el renglón donde se encontró el error.

Error 24: Falta 2da. parte de la alarma. La solución es el reingreso del módulo alarma configurado anteriormente, en el renglón donde se encontró el error.

Error 25: Falta(n) cable(s) del secuenciador. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 26: Falta(n) salida(s) del secuenciador. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 27: Falta(n) el(los) fin(es) del secuenciador. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 28: Falta la parte media izquierda del secuenciador. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 29: Falta la parte media derecha del secuenciador. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 30: Falta la parte inferior izquierda del secuenciador. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 31: Falta la parte inferior derecha del secuenciador. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 32: Falta(n) cable(s) del secuenciador o se esperaban símbolos vacíos, si su secuenciador tiene 1 salida. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 33: Falta(n) salidas(s) del secuenciador o se esperaban símbolos vacíos, si su secuenciador tiene 1 salida. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

Error 34: Falta(n) el(los) fin(es) del secuenciador o se esperaban símbolos vacíos, si su secuenciador tiene 1 salida. La solución es el reingreso del módulo secuenciador configurado anteriormente, en el renglón donde se encontró el error.

CAPITULO 4

EJEMPLOS DE APLICACIÓN

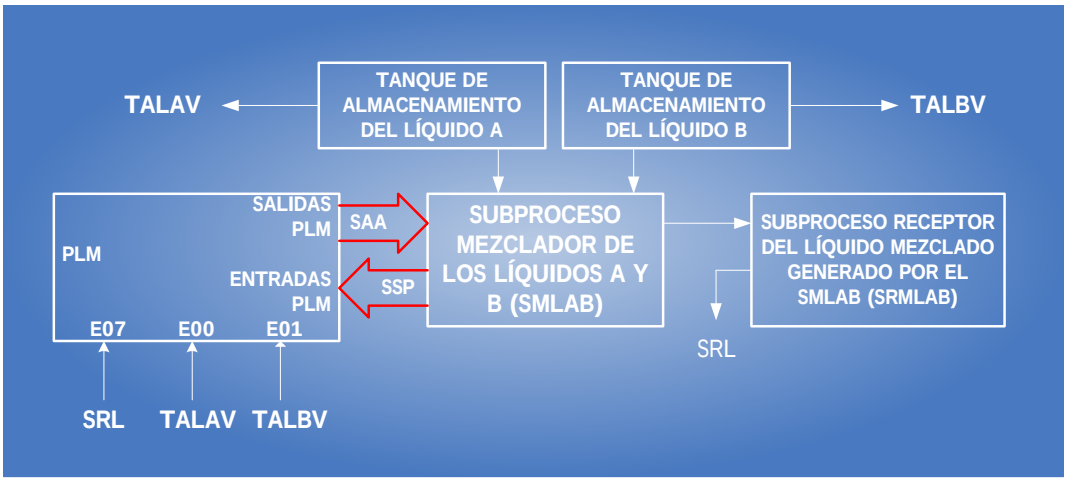
En el presente capítulo se presentarán dos ejemplos de aplicación del PLM programando éste usando diagramas de escalera virtuales, procesados por el software desarrollado en esta tesis. Los ejemplos aquí mostrados son únicamente de carácter ilustrativo.

4.1 Descripción De Un Proceso Susceptible De Ser Automatizado.

Un determinado proceso industrial, puede dividirse en una cadena de subprocesos eslabonados, uno de esos subprocesos requiere que en un tanque, aquí denotado con la letra C, se mezclen dos líquidos A y B en una proporción volumétrica de uno a dos, antes de mezclar los líquidos A y B los mismos deben ser irradiados por separado con una lámpara infrarroja, esto último hace necesario el empleo de dos tanques auxiliares donde se deben colocar las materias primas aquí mencionadas para ser irradiadas, después de lo cual se pasan al tanque de mezclado, una vez que los dos líquidos se encuentran en el tanque mencionado, se debe activar un agitador contenido en el mismo por 30 minutos, después de lo cual, se ha de descargar la mezcla mediante una bomba, de modo que el subproducto obtenido sea empleado por otro subproceso.

El subproceso mezclador objeto de este ejemplo, se le denominará de aquí en adelante como subproceso de mezclado de líquidos A y B (SMLAB), y al subproceso receptor que recibe el subproducto generado por el SMLAB se le denotará como SRMLAB, en la figura 5.1 se ilustra el eslabonamiento entre estos dos subprocesos, nótese la señal SRL (Subproceso Receptor Listo), mediante la cual se testifica que el SRMLAB está en condiciones de recibir una descarga de subproducto, además en la misma figura, se aprecia la existencia de otras dos señales de entrada al PLM (TALAV y TALBV) que testifican el hecho de que alguno de los tanques de almacenamiento que suministran la materia prima que emplea el SMLAB, se encuentre vacío.

El nivel de verificación para la señal SRL es alto y el correspondiente a las señales TALAV y TALBV es bajo, nótese que en la figura 4.1 Se muestra un esquema global del subproceso de mezclado de los líquidos A y B (SMLAB) a ser automatizado por el PLM utilizando PUMA ESCALERA, y su eslabonamiento con el subproceso receptor del subproducto generado se indican también las entradas del PLM que se emplearán para captar las tres señales aquí mencionadas.



SAA = SEÑALES DE ACTIVACIÓN DE ACTUADORES ASOCIADOS CON EL SMLAB

SSP = SEÑALES GENERADAS POR SENSORES LIGADOS AL SMLAB

SRL = TESTIGO DE SUBPROCESO RECEPTOR LISTO PARA RECIBIR PRODUCTO DEL SMLAB

TALAV = TESTIGO DE TANQUE DE ALMACENAMIENTO DE LÍQUIDO A VACÍO

TALBV = TESTIGO DE TANQUE DE ALMACENAMIENTO DE LÍQUIDO B VACÍO

Figura 4.1 Esquema global del subproceso de mezclado de los líquidos A y B (SMLAB) y su eslabonamiento con el subproceso receptor del subproducto generado.

A cada descarga de producto de la mezcla mencionada en el párrafo anterior se le denomina lote, debiendo suministrarse 20 de ellos en una jornada. Las materias primas aquí representadas por los líquidos A y B están contenidas en dos tanques de almacenamiento denominados como A y B; para cada líquido existe un tanque auxiliar con volúmenes en la proporción requerida, donde se lleva a cabo la irradiación infrarroja mencionada en párrafos anteriores; así el tanque auxiliar B tendrá un volumen igual al doble del asociado con el tanque auxiliar A, de esta forma, para lograr la proporción volumétrica requerida primero se suben los líquidos A y B a sus respectivos tanques auxiliares, para después descargarlos al tanque C donde se ha de efectuar el mezclado mediante el agitador mencionado en el párrafo anterior.

En la Figura 4.2 se muestra la disposición de los componentes del SMLAB, apreciándose en la misma las variables del PLM asociadas, las cuales serán señales de arranque para los motores de las bombas y el agitador (variables de salida del PLM) y señales suministradas por los sensores de nivel (variables de entrada al PLM). Se supone que los sensores de nivel son booleanos, proporcionando cada uno de ellos un nivel de uno lógico, cuando el nivel del líquido rebasa la posición del mismo, en otro caso el nivel lógico asociado será cero.

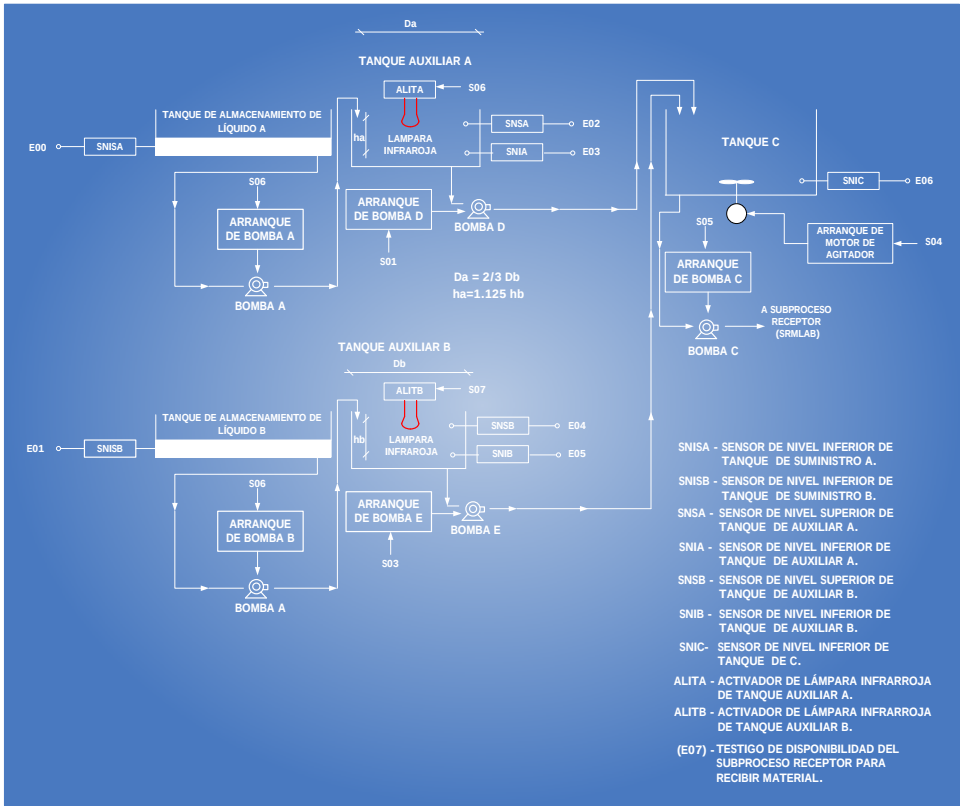


Figura 4.2 Esquema del SMLAB.

4.1.1 Descripción General Del SMLAB

Inicialmente tanto los tanques auxiliares A y B como el tanque de mezclado deben estar vacíos, se requiere que al oprimirse el botón de restablecimiento del PLM, se inicie la rutina lógica que suministrará al subproceso receptor los 20 lotes, la secuencia sería la siguiente:

1.- Se llenan los dos tanques auxiliares empleándose para ello a las bombas A y B, esto requerirá que las señales de arranque correspondientes (S00 y S02) presenten un nivel alto, en caso de que un determinado tanque de almacenamiento se encuentre vacío, se deberá poner en cero la señal de activación (S00 ó S02) que corresponda, notificando esta situación al operador mediante un mensaje móvil en la UD.

2.- Una vez que el tanque auxiliar A se ha llenado, el contenido del mismo deberá ser transferido al tanque de mezclado empleado para ello a la bomba D, lo que requerirá que la señal de activación de la misma (S01), presente un nivel de uno lógico, desde luego que el nivel de la salida S00 (activación de la bomba A) deberá pasar a cero.

Durante el curso de llenado y vaciado de los tanques auxiliares, deberán estar activadas las lámparas infrarrojas de cada uno.

3.- Una vez que el tanque auxiliar B se ha llenado, el contenido del mismo deberá ser transferido al tanque de mezclado empleando para ello a la bomba E, lo que requerirá que la señal de activación de la misma (S03), presente un nivel de uno, desde luego que el nivel de la salida S (activación de la bomba B) deberá pasar a cero.

4.- Al completarse los dos pasos anteriores se deberá verificar por 30 minutos la señal de activación del motor del agitador (S04).

5.- Al completarse el mezclado del paso anterior (flanco de bajada S04), se deberá verificar en alto la salida S05, que activa la bomba C, esto sujeto al permisivo SRL (entrada E07) del PLM, transfiriéndose al siguiente subproceso receptor la mezcla de los líquidos A y B obtenida en el paso cuatro.

6.- Si no se han despachado los veinte lotes requeridos, se deberá pasar al paso uno, para iniciar de nuevo el subproceso, en otro caso se deberá notificar al operador que se ha completado la cuota diaria de lotes. Para reiniciar el despacho de otros veinte lotes se deberá oprimir el botón restablecimiento del PLM.

La realización mediante el PLM, de la rutina lógica descrita a grandes rasgos en párrafos anteriores, puede ser hecha de diversas maneras aquí se presentará una de ellas, que emplea secuenciadores de estado, temporizadores monodisparo, diversas compuertas lógicas, módulos de tipo alarma y módulos de tipo mensajero.

El proceso de diseño del programa en SIIL1 que valida esta aplicación puede desdoblarse en los siguientes pasos:

1.- Selección de módulos y diseño de interconexión lógica de los mismos, para controlar el llenado y descarga del tanque auxiliar A.

2.- Selección de módulos y diseño de interconexión lógica de los mismos, para controlar el llenado y descarga del tanque auxiliar B.

3.- Selección de módulos y diseño de interconexión lógica de los mismos, para controlar el mezclado en el tanque C y la descarga del mismo a otro subproceso.

4.- Selección de un módulo contador de eventos que lleve la cuenta de los lotes despachados.

5.- Determinación de los distintos mensajes de alarma o información de flujo del subproceso de mezcla volumétrica arbitrado por el PLM, que podrían ser de alguna utilidad para operador.

6.- Identificación de las variables booleanas que dispararían los mensajes de alarma mencionados en el paso anterior.

7.- En base a la información asociada con los dos pasos anteriores especificar los módulos mensajeros y de alarma necesarios.

A continuación se detallan los módulos empleados para cada uno de los puntos anteriores, así como también el interconexionado entre ellos. Para una mejor comprensión es recomendable observar la figura 4.2 conforme se avance en la lectura de las secciones 4-1-2 a 4-1-6.

4.1.2 Módulos empleados para realizar la automatización del llenado y descarga del tanque auxiliar A.

Para la realización de la rutina lógica de llenado y vaciado del tanque auxiliar A, se emplearon un secuenciador de estados de 2 x 3, tres temporizadores monodisparo y tres compuertas lógicas; en la figura 4.3 se aprecia en los módulos empleados y las interconexiones entre ellos, el temporizador 3 (TEMPOC#3) tiene como función el generar la señal de restablecimiento del secuenciador implicado, esto debe acontecer, cada que se ha completado una descarga de subproducto, al subproceso receptor (flanco de bajada del sensor de nivel SNIC (entrada E06 del PLM), de modo que se puede iniciar un nuevo ciclo de mezclado siempre que no se haya completado la cuota de veinte lotes; por lo tanto, la señal de salida del temporizador 3 (I06), también será empleada por otros módulos implicados en la automatización de subproceso SMLAB.

Los otros dos temporizadores mostrados en la figura 4.3 (TEMPOC#1 Y TEMPOC#2), sirven para evitar disparos espurios del secuenciador (SEC2#1), debidos a rebotes en las señales generadas por los sensores de nivel SNIA Y SNSA.

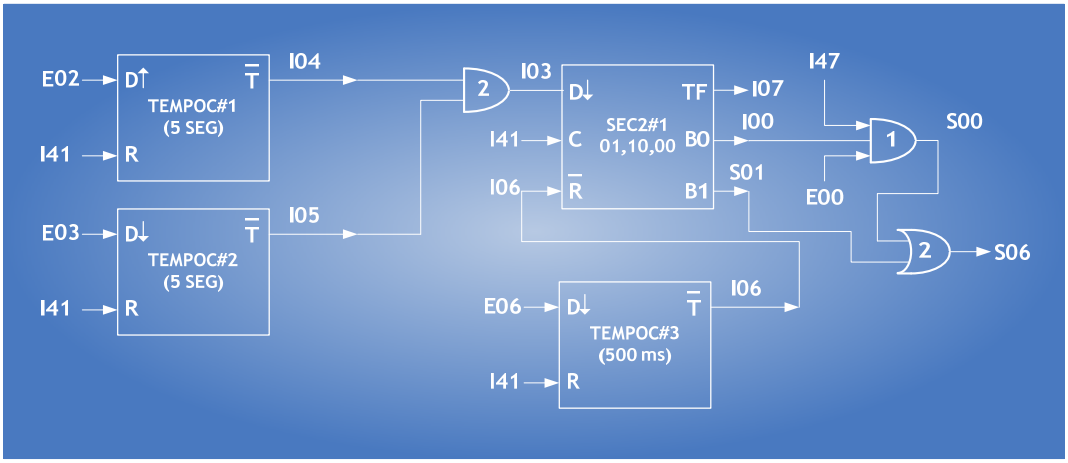


Figura 4. 3 - Módulos lógicos empleados en la realización de la rutina lógica de llenado y vaciado del tanque auxiliar A.

Las entradas de disparo, congelamiento y restablecimiento de este temporizador son las VBI I03, I41 e I06, siendo el disparo del mismo por flanco de bajada y los niveles de verificación de las entradas de congelamiento y restablecimiento son respectivamente alto y bajo. El testigo de fin de carrera es la VBI I07 y se verifica en un nivel alto.

La función de cada uno de los 2 bits de salida de este secuenciador se explica a continuación:

El bit menos significativo es la VBI I00, al presentar un nivel de uno lógico hace que se active la bomba A, siempre que haya líquido en el tanque de almacenamiento A (E00 en alto) y que la VBI I47 presente un nivel alto, la cual acontecerá siempre que no se hayan despachado los veinte lotes, véase la compuerta AND3#1 en la figura 4. 3 y más adelante, lo concerniente con el contador de eventos empleado para contar los lotes despachados.

El otro bit es la VBS S01 que activa la bomba D la cual descarga al tanque de mezclado el contenido del tanque auxiliar A, una vez que este último se ha llenado.

En la tabla 4.1 se muestra la secuencia de estados requerida para el secuenciador SEC2#1, indicándose ahí también los retroavisos asociados.

ESTADO	**B1(S01)**	**B0(I00)**	**RETROAVISO PARA AVANCE**	**RETROAVISO AL ESTADO INICIAL**
1(inicial)	0	1	Flanco de subida de E02	
2	1	0	Flanco de bajada de E03	
3	0	0		Nivel bajo en VBI I06

Tabla 4.1 - Secuencia de estados del secuenciador SEC2#1, empleado en la realización de la rutina lógica de llenado y vaciado del tanque auxiliar mayúscula inicial A.

De acuerdo con lo explicado en párrafos anteriores, la declaración en SIIL1 del secuenciador SEC2#1:

SEC2#1 I03,I41,I06,I07,S01,I00,3,0111;
B01,B10,B00;

Declaraciones en SILL1 asociados con las compuertas lógicas y temporizadores implicados en el llenado y vaciado del tanque auxiliar A.

De acuerdo con lo mostrados en la figura 4.3, las declaraciones en SILL1 para cada uno de los temporizadores y compuertas lógicas ahí mostradas, se detalla a continuación:

Para el temporizador monodisparo número uno (TEMPOC#1) la declaración es:

TEMPOC#1 E02,I41,I04, 00:00:05:00, 100;

Para el temporizador monodisparo número dos (TEMPOC#2) la declaración es:

TEMPOC#2 E03,I41,I05, 00:00:05:00, 000;

Para el temporizador monodisparo número tres (TEMPOC#3) la declaración es:

TEMPOC#3 E06,I41,I06, 00:00:05:00, 000;

Para la compuerta and tres entradas (AND3#1) la declaración es:

AND3#1 I47, E00, I00, S00, 111;

Para la compuerta and dos entradas (AND2#2) la declaración es:

AND2#1 I04, I05, I03, 11;

Para la compuerta OR de dos entradas (OR2#2) la declaración es:

OR2#2 S00, S01, S06, 11;

4.1.3 Módulos empleados para realizar la automatización del llenado y descarga del tanque auxiliar B.

Para la realización de la rutina lógica de llenado y vaciado del tanque auxiliar B, se emplearon un secuenciador de estados de 2 x 3, tres temporizadores monodisparo y tres interconexiones entre ellos, nótese que la señal de restablecimiento del secuenciador implicado, es la VBI I06 tres la salida del temporizador monodisparo número tres (TEMPOC#3) muestra la figura 4.3, esto debe acontecer, cada que se ha completado una descarga de su producto, al subproceso receptor (flanco de bajada del sensor de nivel) SNCI (entrada E06 del PLM), de modo que se pueda iniciar un nuevo ciclo de mezclado siempre que no se haya completado la cuota de veinte lotes; por lo tanto, la señal de salida del temporizador número tres (I06), también será empleada por otros módulos implicados en la automatización del subproceso SMLAB.

Los otros dos temporizadores mostrados en la figura 4.4 (TEMPOC#4 Y TEMPOC#5), sirven para evitar disparos espurios del secuenciador (SEC2#2), decididos a rebotes en las señales generadas por los sensores de nivel SNIB Y SNSB.

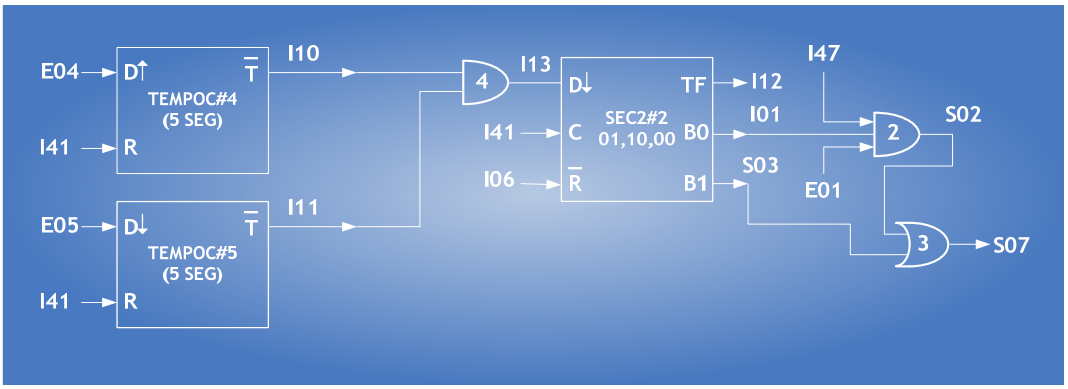


Figura 4. 4 -. Módulos lógicos empleados en la realización de la rutina lógica de llenado y vaciado del tanque auxiliar B.

5.1.3.1 Configuración del secuenciador de estados SEC2#2 mostrado en la figura 4.4

Las entradas de disparo, congelamiento y restablecimiento de este temporizador son las VBI I13, I41 e I06, siendo el disparo del mismo por flanco de bajada y los niveles de verificación de las entradas de congelamiento y restablecimiento son respectivamente alto y bajo. El testigo de fin de carrera es la VBI I12 y se verifica en nivel alto.

La función de cada uno de los 2 bits de salida de este secuenciador se explica a continuación:

El bit menos significativo es la VBI I01, que al presentar un nivel de uno lógico hace que se active la bomba B, siempre que haya líquido en el tanque de almacenamiento B (E01 en alto) y que la VBI I47 presente un nivel alto, lo cual acontecerá siempre que no se hayan despachado los veinte lotes, véase la compuerta AND3#3 en la figura 4.4 y más adelante, lo concerniente con el contador de eventos empleado para contar los lotes despachados.

El otro bit es la VBS S03 que activa la bomba D la cual descarga el tanque de mezclado el contenido del tanque auxiliar B. Una vez que este último se ha llenado.

En la tabla 4.2 se muestra la secuencia de estado requerida para el secuenciador SEC2#2, indicándose ahí también los retroavisos asociados.

ESTADO	B1(S03)	B0(I01)	RETROAVISO PARA AVANCE	RETROAVISO AL ESTADO INICIAL
1(inicial)	0	1	Flanco de subida de E04	
2	1	0	Flanco de bajada de E05	
3	0	0		Nivel bajo en VBI I06

Tabla 4.2 - Secuencia de estados del secuenciador SEC2#2, empleado en la realización de la rutina lógica de llenado y vaciado del tanque auxiliar B.

De acuerdo con lo explicado los párrafos anteriores, la declaración en SIIL1 del secuenciador SEC2#2 es:

SEC2#2 I13,I41,I06,I07,S01,I00,3,0111;
B01,B10,B00;

Declaraciones en SIIL1 asociados con las compuertas lógicas y temporizadores implicados en el llenado y vaciado del tanque auxiliar B.

De acuerdo con lo mostrado en la figura 4.4, las declaraciones en SIIL1 para cada uno de los temporizadores y compuertas lógicas ahí mostradas, se detalla continuación:

Para el temporizador monodisparo un número cuatro (TEMPOC#4) la declaración es:
TEMPOC#4 E04,I41,I10, 00:00:05:00, 100;

Para el temporizador monodisparo un número cinco (TEMPOC#5) la declaración es:

TEMPOC#5 E05,I41,I11, 00:00:05:00, 000;

Para la compuerta and de tres entradas (AND3#3) la declaración es:

AND3#3 I47,E01,I01, S02, 111;

Para la compuerta and de dos entradas (AND2#2) la declaración es:

AND2#2 I10, I11, I13, 11;

Para la compuerta OR de dos entradas (OR2#3) la declaración es:

OR2#3 S02, S03, S07, 11;

4.1.4 Módulos empleados para realizar la automatización del mezclado de los líquidos A y B en el tanque C.

Para la realización de la rutina lógica que valida el mezclado de los líquidos A y B, que se lleva a cabo en el tanque C, se emplearon un secuenciador de estados de 2 x 3, cuatro módulos empleados y las interconexiones entre ellos, nótese que la señal de restablecimiento del secuenciador implicado, es la VBI I06 que es la salida del temporizador monodisparo número tres (TEMPOC#3) mostrado en la figura 4.3, esto debe acontecer, cada que se ha completado una descarga de subproducto, al subproceso receptor (flanco de bajada del sensor de nivel SNIC entrada E06 del PLM), de modo que se pueda iniciar un nuevo ciclo de mezclado siempre que no se haya completado la cuota de veinte lotes; por lo tanto, la señal de salida del temporizador número tres (I06), también será empleada por otros módulos implicados en la automatización del subproceso SMLAB.

Los otros tres temporizadores mostrados en la figura 4.5 (TEMPOC#6, TEMPOC#7 Y TEMPOC#8), sirven, los dos primeros, para generar los dos retroavisos asociados con el secuenciador (SEC2#3), y el último delimita el tiempo que ha de estar activado el agitador para llevar a cabo el mezclado de los líquidos A y B.

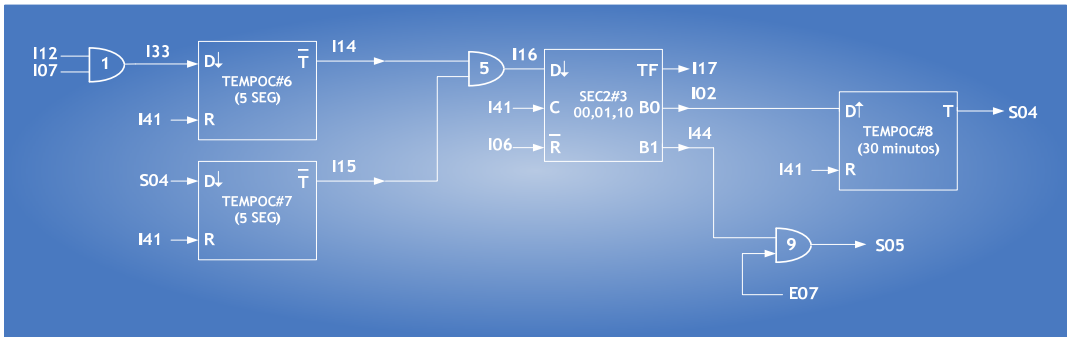


Figura 4. 5. Módulos lógicos empleados en la realización de la rutina lógica para efectuar el mezclado de los líquidos A y B en el tanque C.

4.1.4.1 Configuración del secuenciador de estados SEC2#3 mostrado en la figura 4.5

Las entradas de disparo, congelamiento y restablecimiento de este temporizador son las VBI I16, I41 e I06, siendo el disparo del mismo por flanco de bajada y los niveles de verificación de las entradas de congelamiento y restablecimiento son respectivamente alto y bajo. El testigo de fin de carrera es la VBI I17 y se verifica en nivel alto.

La función de cada uno de los dos de salida de este secuenciador se explica a continuación:

El bit menos significativo es la VBI I02, que al presentar un flanco de subida hace que el temporizador monodisparo active por 30 segundos el motor que mueve el agitador.

El otro bit es la VBI I44 que activa a la bomba C, que descarga al subproceso receptor el resultado de la mezcla de los líquidos A y B, esto último siempre y cuando el testigo de que el subproceso receptor está listo (VBE E07 en alto).

En la tabla 4.3 se muestra la secuencia de estado requerida para el secuenciador SEC2#3, empleado con la realización de la rutina lógica para efectuar el mezclado de los líquidos A y B en el tanque C. Indicándose ahí también los retroavisos asociados.

ESTADO	B1(I44)	B0(I02)	RETROAVISO PARA AVANCE	RETROAVISO AL ESTADO INICIAL
1(inicial)	0	0	Flanco de bajada de I33*	
2	0	1	Flanco de bajada de S04**	
3	1	0		Nivel bajo en VBI I06

Tabla 4. 3 - Secuencia de estados del secuenciador SEC2#3.

* La VBI I33 presentará un flanco de bajada, cada vez que se completa el vaciado de los tanques auxiliares A Y B al tanque de mezclado C, ya que la misma es la salida de una compuerta NAND (NAND2#1) cuyas dos entradas son los testigos de fin de carrera de los dos secuenciadores que arbitran el llenado y vaciado de dichos tanques auxiliares.

** El flanco de bajada de la VBS S04 marca el fin del intervalo de tiempo que el motor del agitador en el tanque C está activado.

De acuerdo con lo explicado en párrafos anteriores, la declaración en SIIL1 del secuenciador SEC2#3 es:

SEC2#3 I16, I41, I06, I17, S05, I02, 3, 0111;
B00, B01, B10;

Declaraciones en SIIL1 asociadas con las compuertas lógicas y temporizadores implicados en el mezclado, en el tanque C., de los líquidos A y B.

De acuerdo con lo mostrado en la figura 4.4, las declaraciones en SIIL1 para cada uno de los temporizadores y compuertas lógicas ahí mostradas, se detalla a continuación:

Para el temporizador monodisparo número seis (TEMPOC#6) la declaración es:

TEMPOC#6 I33, I41,I14, 00:00:00.50, 000;

Para el temporizador monodisparo número siete (TEMPOC#7) la declaración es:

TEMPOC#7 S04, I41,I15, 00:00:00.50, 000;

Para el temporizador monodisparo número ocho (TEMPOC#8) la declaración es:

TEMPOC#8 I02, I41,S04, 00:30:00.00, 000;

Para la compuerta and de dos entradas (AND2#5) la declaración es:

AND2#5 I14, I15, I16, 11;

Para la compuerta and de dos entradas (AND2#9) la declaración es:

AND2#9 I44, E07, S05, 11;

Para la compuerta NAND de dos entradas (NAND2#1) la declaración es:

NAND2#1 I12, I07, I33, 11;

4.1.5 Descripción de los módulos empleados para llevarla cuenta de lotes despachados y a la misma en la UD.

Para llevar la cuenta de lotes despachados, se emplea un módulo contador de eventos ascendente con cuenta de cero a veinte, con avance de cuenta por flanco de bajada de la VBE E06 (sensor SNIC), siendo las entradas de congelamiento y restablecimiento habilitadas por la VBI I41, con niveles de verificación alto y bajo respectivamente; para testigo de fin de carrera del módulo contador se seleccionó a la VBI I47 con verificación en bajo; la declaración para el contador de eventos es:

CONTA#1 E06, I41,I41,I47, 0, 20, 01010;

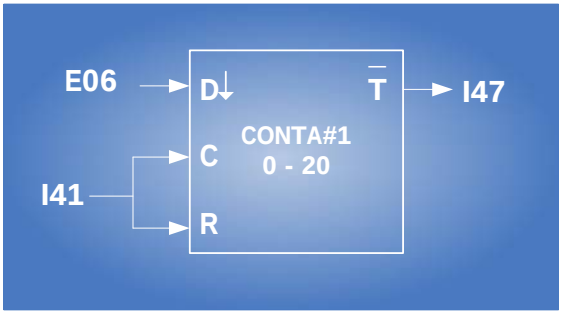


Figura 4.6 - Módulo contador de eventos empleado en la automatización del subproceso SMLAB.

4.1.6 Definición de los mensajes a desplegarse en la UD al llevarse a cabo la rutina lógica que valida la automatización del subproceso SMLAB.

Conforme se vaya llevando a cabo el subproceso de mezclado, en la UD deberán aparecer diversos mensajes que testifiquen entre otras cosas lo siguiente: condiciones de alarma, o simplemente el aviso al operador acerca de qué paso del SMLAB se está llevando a cabo en un momento dado.

En el renglón dos de la UD se desplegarán los distintos mensajes móviles, al ejecutarse el programa que valida la automatización del SMLAB, esto debido a la ocurrencia de diversos eventos, que pueden ser de alarma o simplemente informativos, todos estos mensajes carecen de texto fijo y se moverán con una cadencia de 20 centésimas de segundos entre posiciones sucesivas. En la tabla 4.4 se muestra el texto para cada caso, así como también la VB (denotada como VBD), que dispara cada mensaje y el nivel de verificación de esta última para que esto suceda; el orden de prioridad de los mensajes es el mismo en el que cada uno de ellos aparece en la tabla.

VBBD	NVVD/NM	TEXTO DEL MENSAJE MÓVIL A DESPLEGAR AL VERIFICARSE LA VBD ASOCIADA
I30	Bajo/1	Tanques de almacenamiento de líquidos Ay B vacíos.
S04	Alto/2	Proceso de mezclado en tanque C activado.
S05	Alto/3	Descarga de tanque mezclador en curso, (bomba C activada).

Tabla 4. 4 Relación de mensajes de alarma e informativos a desplegarse en el renglón dos de la UD al llevarse a cabo la automatización del SMLAB empleando el PLM.

Nota:
VBD. Denota a la variable booleana que dispara cada mensaje.
NVVD. Denota en nivel de verificación de la variable booleana para que se dispare el mensaje.
NM. Denota el número de módulo mensajero asociado con el texto móvil a desplegar en cada caso.

A continuación se describen las operaciones lógicas empleadas para obtener la VBI I30 que dispara al mensaje de prioridad 1; véase la tabla 4.4

La VBI I30 se obtiene como la salida de una compuerta OR de dos entradas, que son las VBE E00 y E01, la declaración correspondiente es:

OR2#1 E00, E01, I30, 11;

Al programa en SIIL1 que hace que el PLM realice la automatización del SMLAB se le llamó EJAP11 y el mismo está contenido en el archivo de texto EJAP11.SIL, que se muestra a continuación, ahí mismo se pueden apreciar las declaraciones asociadas con los módulos de tipo alarma y mensajero empleados.

```
INPROG;

 NAND2#1 I12, I07, I33, 11;
 AND2#1 I04,I05,I03,11;
 AND3#2 I47,I00,E00,S00,111;
 AND2#3 I10,I11,I13,11;
 AND3#4 I47,I01,E01,S02,111;
AND2#6 I14,I15,I16,11;
 AND2#7 I44,E07,S05,11;
ALARMA#1 I30,0;
 ALARMA#2 S04,1;
 ALARMA#3 S05,1;

 FINPP;

 INMODI;
 TEMPOC#1 E02,I41,I04,00:00:05.00,100;
 TEMPOC#2 E03,I41,I05,00:00:05.00,000;
 TEMPOC#3 E06,I41,I06,00:00:00.50,000;
 SEC2#1 I03,I41,I06,I07,S01,I00,3,0111;
 # b01, b10;
 ## b00;
 CONTA#1 E06,I41,I41,I47,0,20,01110;
 OR2#1 S00,S01,S06,11;
 TEMPOC#4 E04,I41,I10,00:00:05.00,100;
 TEMPOC#5 E05,I41,I11,00:00:05.00,000;
 SEC2#2 I13,I41,I06,I12,S03,I01,3,0111;
 # b01, b10;
 ## b00;
 OR2#2 S02,S03,S07,11;
 TEMPOC#6 I33,I41,I14,00:00:05.00,000;
 TEMPOC#7 S04,I41,I15,00:00:05.00,000;
 SEC2#3 I16,I41,I06,I17,I44,I02,3,0111;
 # b00, b01;
 ## b10;
 TEMPOC#8 I02,I41,S04,00:30:00.00,101;
 MENSAJERO#1 "",1,16,20,101;
```

Tanques de almacenamiento de líquidos A y B vacíos.
MENSAJERO#2 "",1,16,20,101;
Proceso de mezclado en tanque C activado.
MENSAJERO#3 "",1,16,20,101;
Descarga de tanque mezclador en curso, (bomba C activada).
OR2#3 E00,E01,I30,11;

FINMODI;

4.1.7 Modelado Del Proceso Descrito en los Puntos 4.1.1 al 4.1.6 utilizando el Puma-Escalera

A continuación se muestra el uso del programa PUMA ESCALERA para el modelado del proceso descrito anteriormente en los puntos 4.1.1 al 4.1.6. La configuración de cada modulo se ha explicado con detalle en el Anexo A, por tal motivo solo se mostrarán los módulos que deban ser insertados.

4.1.7.1 Módulos Empleados Para Realizar La Automatización Del Llenado Y Descarga Del Tanque Auxiliar A.

En primer lugar se modelará el proceso de llenado y descarga del tanque auxiliar A. La pantalla en un inicio tendrá la apariencia de la figura 4.8

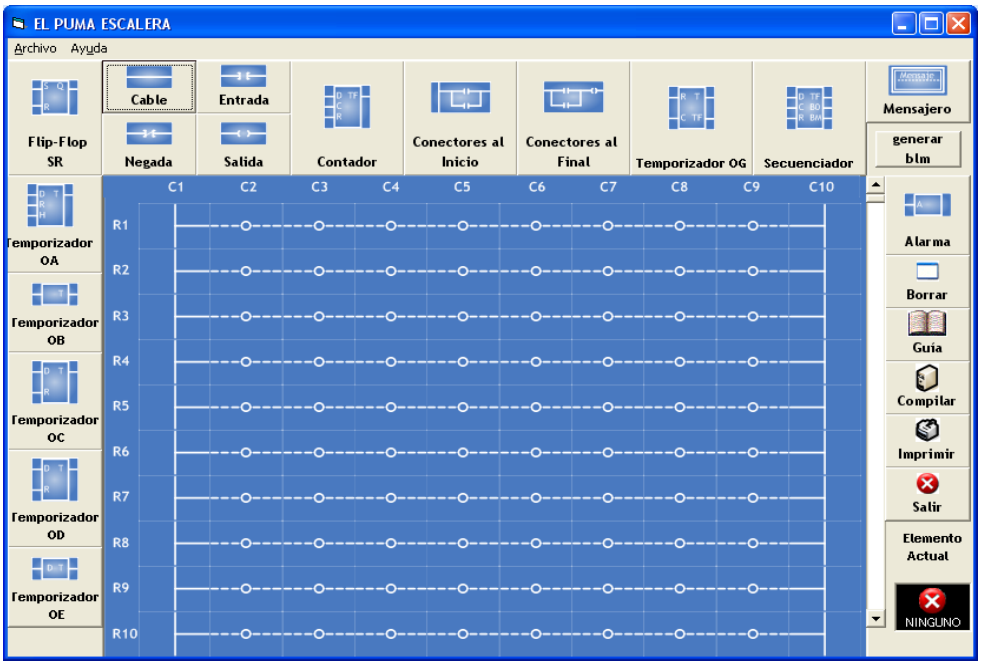


Figura 4.8 - Apariencia inicial del puma escalera.

Se debe ingresar el TEMPOC#1 con entradas E02 e I41, y salida I04 con duración del pulso de salida de 5 segundos, mostrado en la figura 4.9

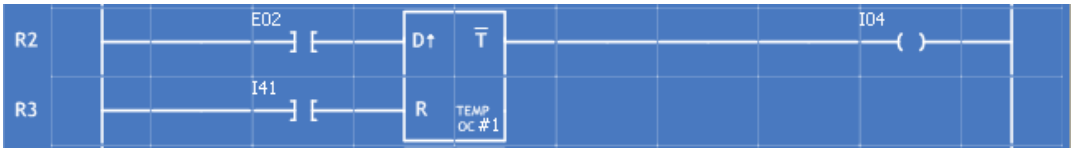


Figura 4.9 - TEMPOC#1.

A continuación se ingresa el TEMPOC#2 con entradas E03 e I41, y salida I05, con duración del pulso de salida de 5s. Mostrado en la figura 4.10.

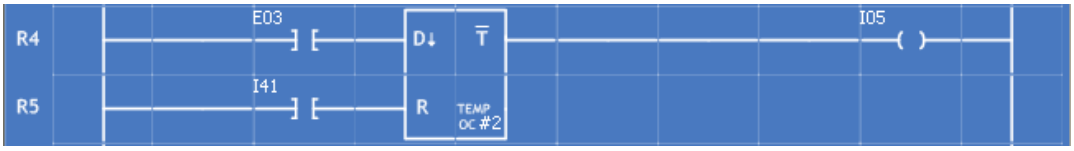


Figura 4.10 - TEMPOC#2.

Posteriormente se ingresa un AND2#2 con entradas I04 e I05, y salida I03; ver figura 4.11.

Figura 4.11 - AND2#2.

Se debe de ingresar un TEMPOC#3 con entradas E06 e I41 y salida I06, con duración del pulso de salida de 50 centésimas de segundo. Mostrado en la figura 4.12.

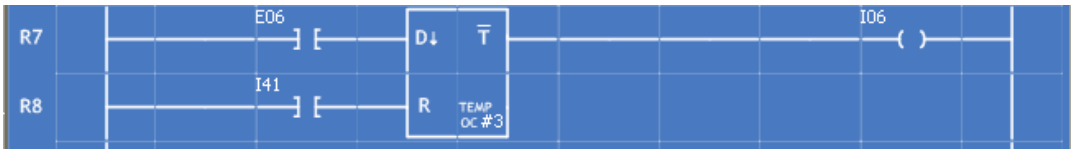


Figura 4.12 - TEMPOC#3.

A la salida de la compuerta AND2#2 se encuentra el SEC2#1 con entradas I03, I41 e I06, y salida TF = I07 y B0,B1 iguales a I00 y S01 respectivamente. El secuenciador se muestra en la figura 4.13

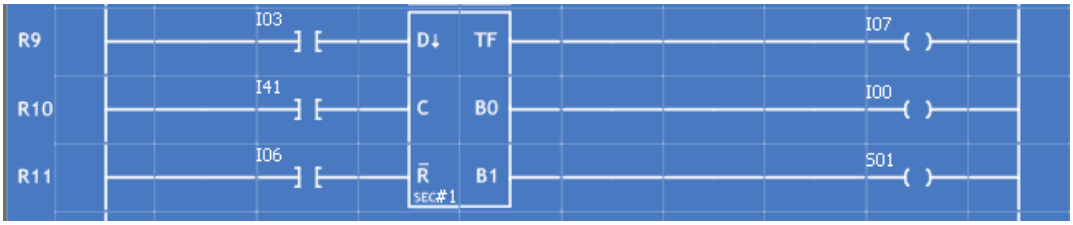


Figura 4.13 - SEC2#1.

La salida del módulo contador de eventos es la entrada de la AND3#1 por lo tanto debe ser declarado previo a esta. Las entradas del modulo contador son E06 para el disparo e I41 para C y R, la salida es T que verifica en bajo y lleva la cuenta del 0-20. La figura 4.14 representa al contador de eventos CONTA#1.

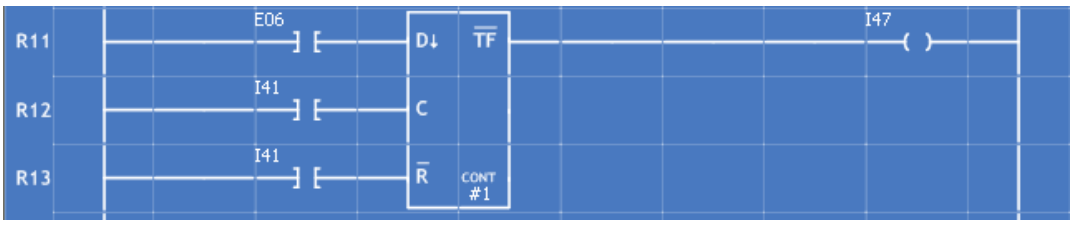


Figura 4.14 - CONTA#1.

A la salida B0 del SEC2#1 está el modulo AND3#1 con entradas I47, I00, y E00 y con salida S00 representado en la figura 4.15.

Figura 4.15 - AND3#1

Para finalizar esta sección se debe insertar una OR2#2 Con entradas S00 y S01 y salida S06; ver figura 4.16

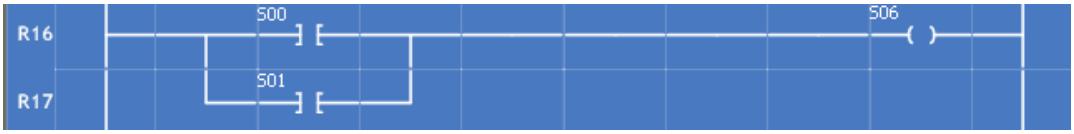


Figura 4.16 - OR2#2

El diagrama de escalera resultante es igual al mostrado en la figura 4.17.

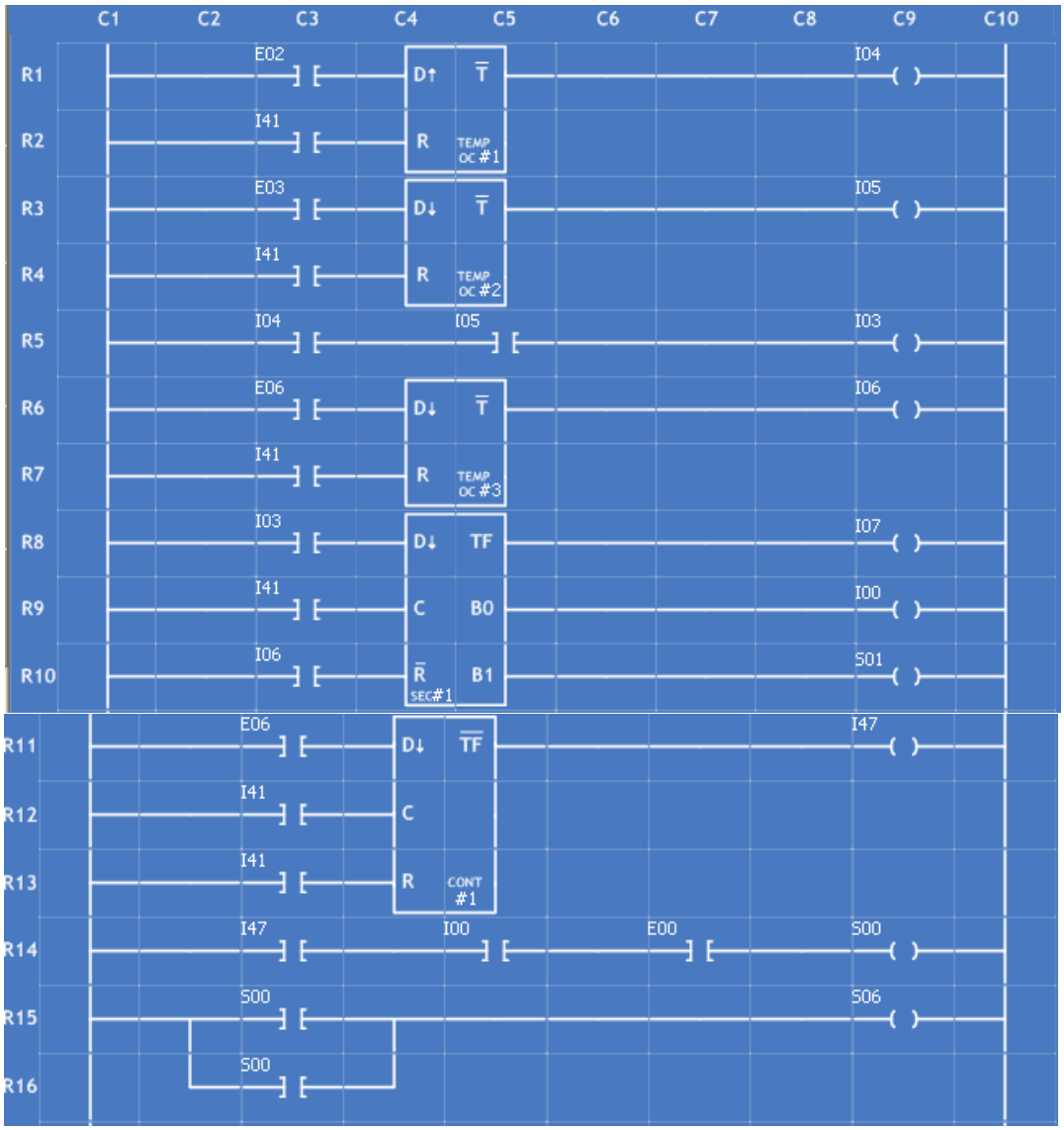


Figura 4.17 - El diagrama de escalera resultante.

4.1.7.2 Módulos Empleados Para Realizar La Automatización Del Llenado Y Descarga Del Tanque Auxiliar B.

A continuación se modelará el proceso de llenado y descarga del tanque auxiliar B. Se debe ingresar un TEMPOC#4 con entradas E04 e I41 y salida I10, con duración del pulso de salida de 5 segundos. Mostrado en la figura 4.18

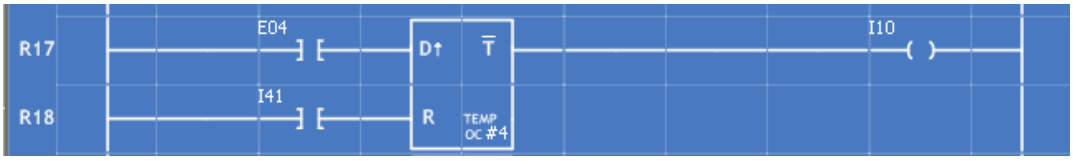


Figura 4.18 - TEMPOC#4 con entradas E04 e I41 y salida I10.

Posteriormente se ingresa un TEMPOC#5 con entradas E05 e I41 y salida I11, con duración del pulso de 5 segundos. Mostrado en la figura 4.19.

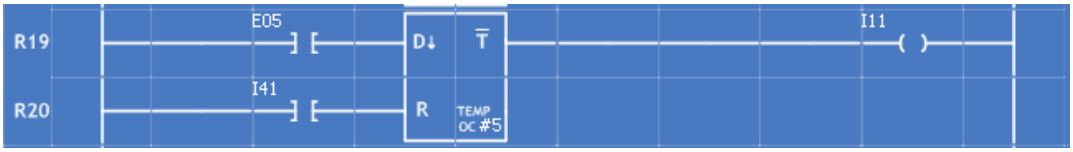


Figura 4.19 - TEMPOC#5 con entradas E05 e I41 y salida I11.

A la salida de los dos Temporizadores anteriores se encuentra la compuerta AND#4 con entradas I10 e I11 y salida I13. Mostrada en la figura 4.20

Figura 4.20 - AND#4 con entradas I10 e I11 y salida I13.

Posteriormente se debe ingresar el secuenciador SEC2#2 con entradas I13, I41 e I06 y salidas I12, I01 y S03 y configurar sus estados (01,10,00). Tal como se muestra en la figura 4.21

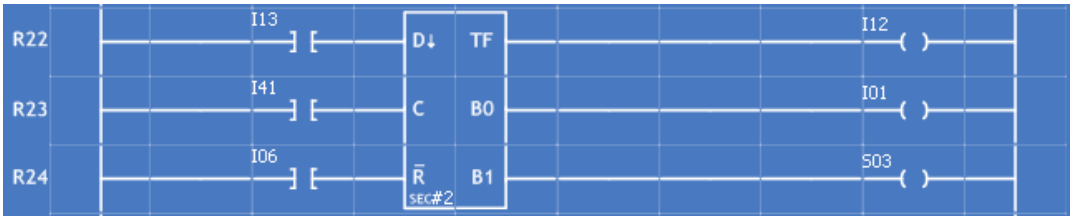


Figura 4.21 - SEC2#2 con entradas I13, I41 e I06 y salidas I12,I01 y S03.

A la salida del secuenciador SEC2#2 debe de ir una compuerta AND3#2, que llevará por entradas I47,I01,y E01 además su salida será S02. Tal como la mostrada en la figura 4.22

Figura 4.22 - AND de tres entradas (I47, I01 y E01) y una salida (S02).

Por último para este proceso se deberá introducir a una compuerta OR2#3 la salida de la compuerta AND3#2 y la salida del Secuenciador SEC2#2, y la salida de la comporta OR deberá ser S07, tal y como se muestra en la figura 4.23.

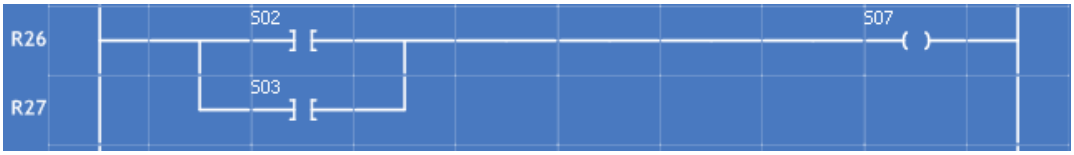


Figura 4.23 - OR2#3 con entradas S02 y S03 con salida S07.

De modo que el diagrama de escalera para la rutina lógica del llenado y vaciado del tanque auxiliar B, se muestra en la figura 4.24

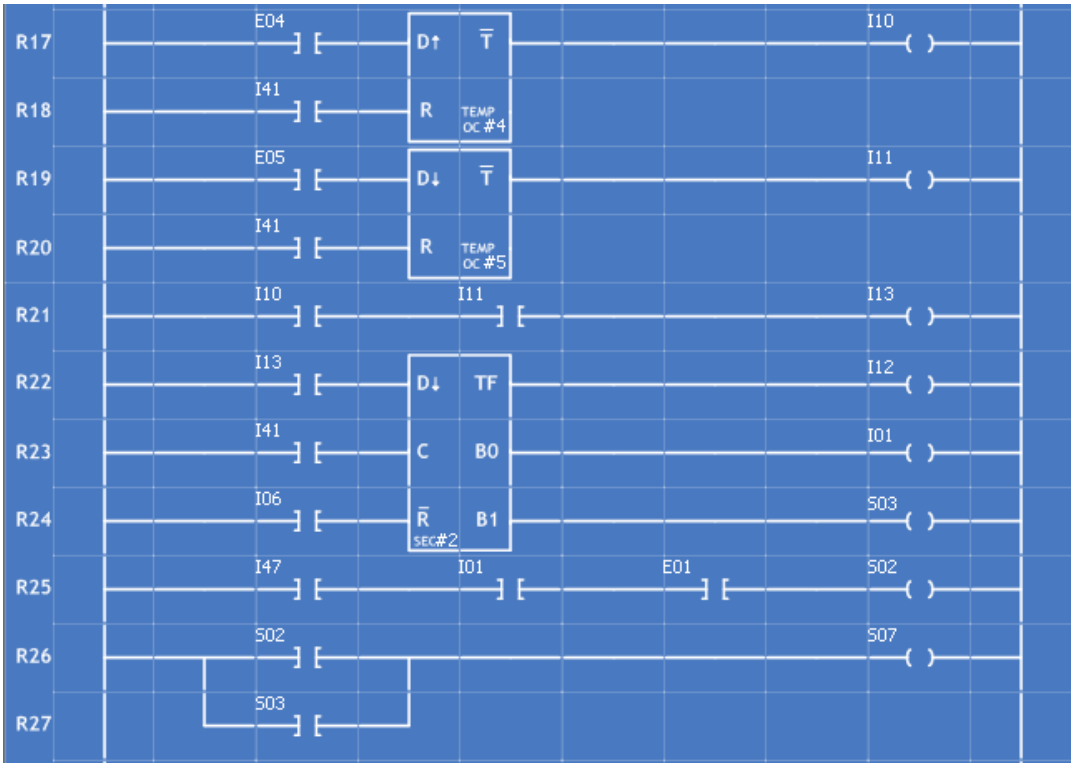


Figura 4.24 - Diagrama de escalera para la rutina lógica del llenado y vaciado del tanque auxiliar B.

4.1.7.3 Módulos Empleados Para Realizar La Automatización Del Mezclado De Los Líquidos A Y B En El Tanque C.

Para realizar este proceso, primero se debe de ingresar las VBI I12 e I07 a una compuerta NAND de dos entradas, el PUMA ESCALERA no cuenta propiamente con una compuerta NAND, sin embargo esta se puede implementar con una compuerta AND y la salida pasarla por un INVERSOR. Tal y como se muestra en la figura 4.25

Figura 4.25 – Equivalencia de una compuerta NAND en el PUMA ESCALERA.

Posteriormente se debe ingresar el TEMPOC#6 con entradas I33 e I41 y con salida I14. Así como el mostrado en la figura 4.26.

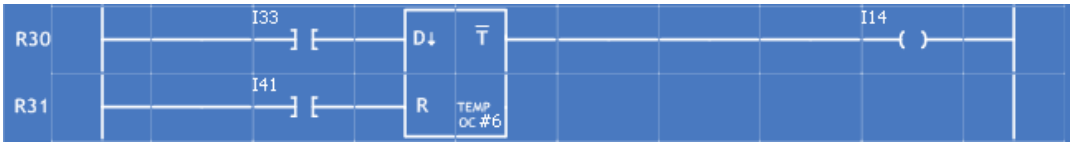


Figura 4.26 - TEMPOC#6 con entradas I33 e I41 y con salida I14.

A continuación se inserta el TEMPOC#7 con entradas S04 e I41 y salida I15. Con un tiempo de 5 segundos. Mostrado en la figura 4.27.

Figura 4.27 - TEMPOC#7 con entradas S04 e I41 y salida I15.

A la salida de los dos temporizadores anteriores se encuentra la compuerta lógica AND que tendrá por entradas I14 e I15 y llevará por salida I16. Tal como la mostrada en la figura 4.28.

Figura 4.28 - Compuerta AND con entradas I14 e I15 y salida I16.

A continuación se insertará el Secuenciador SEC2#3 con entradas I16, I41 e I06 y llevará por salidas I17, I02 e I44. Sus estados serán 00,01 y 10 respectivamente. Como lo muestra la figura 4.29

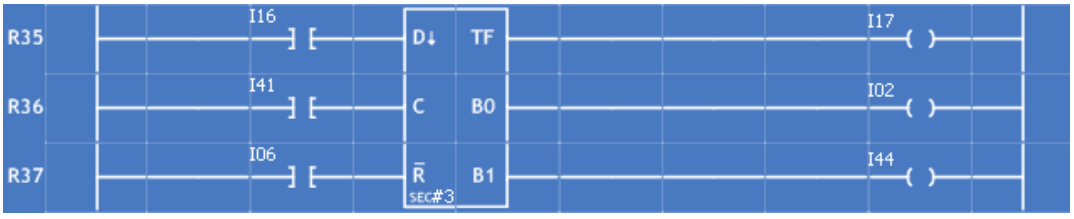


Figura 4.29 Secuenciador SEC2#3 con entradas I16, I41 e I06, y salidas I17, I02 e I44.

Casi para finalizar se debe de ingresar el TEMPOC#8 con entradas I02 e I41 y llevará por salida S04. El tiempo del pulso de salida será de 30 minutos. El temporizador se deberá ver igual al de la figura 4.30.

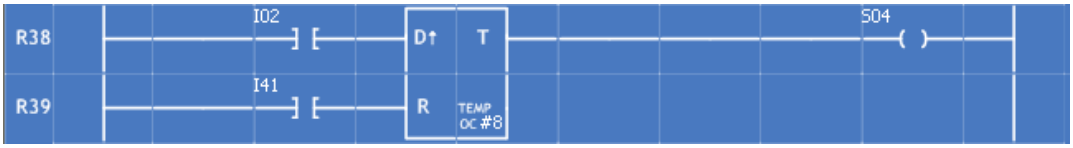


Figura 4.30 - Temporizador TEMPOC#8 con entradas I02 e I41 y llevará por salida S04.

Finalmente La salida del secuenciador SEC2#3 será la entrada de una compuerta AND, la otra entrada será E07 y la salida deberá ser S05, tal como se muestra en la figura 4.31.

Figura 4.31 - Compuerta AND con entradas I44 y E07, y salida S05.

De modo que el diagrama de escalera para la rutina lógica de mezclado de los líquidos A y B, es mostrada en la figura 4.32

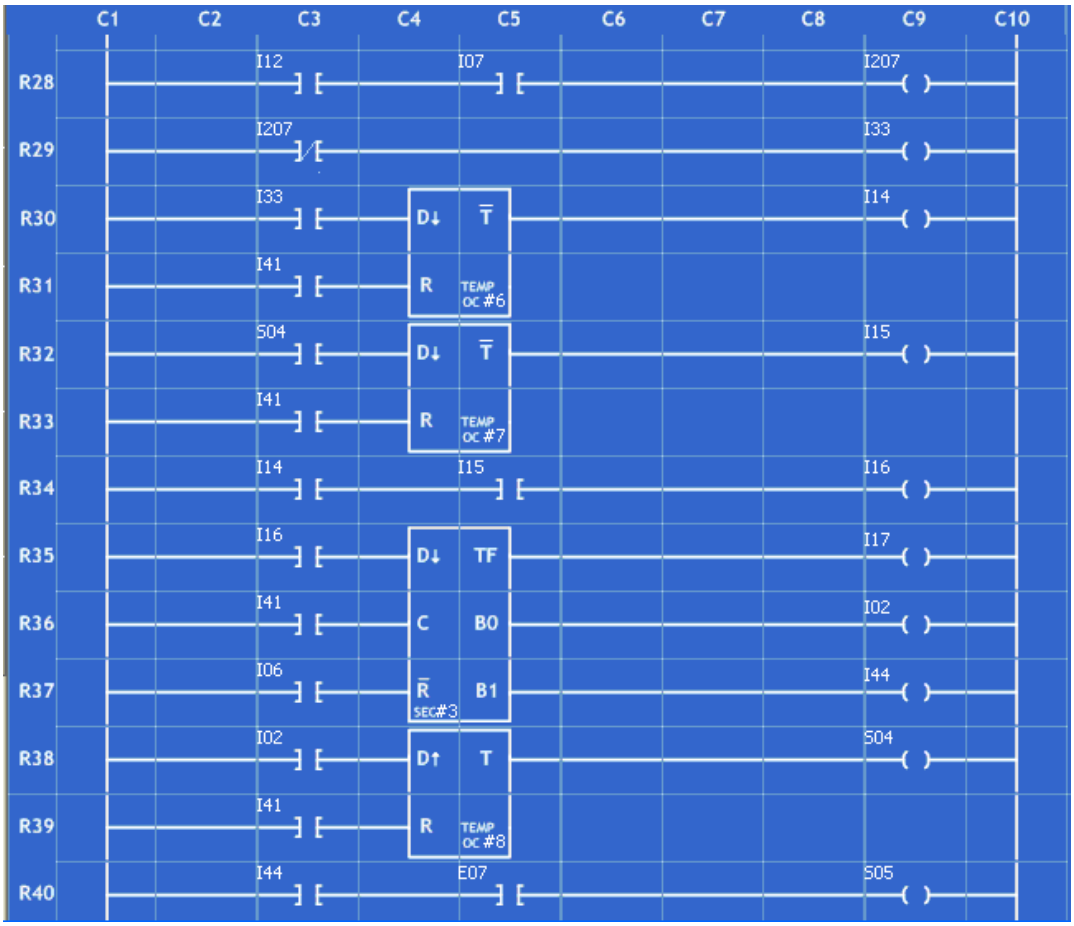


Figura 4.32 - Diagrama de escalera para la rutina lógica de mezclado de los líquidos A y B.

4.1.7.4 Declaración Del Mensajero.

Conforme se vaya llevando a cabo el subproceso de mezclado, en la UD deberán aparecer diversos mensajes que testifiquen entre otras cosas lo siguiente: condiciones de alarma, o simplemente el aviso al operador acerca de qué pasos del SMLAB se están llevando a cabo en un momento dado.

En el renglón dos de la UD se desplegarán los distintos mensajes móviles, al ejecutarse el programa que valida la automatización del SMLAB, esto debido a la ocurrencia de diversos eventos, que pueden ser de alarma o simplemente informativos, todos estos mensajes carecen de texto fijo y se moverán con una cadencia de 20 centésimas de segundos entre posiciones sucesivas. En la tabla 4. 4 se muestra el texto para cada caso, así como también la VB (denotada como VBD), que dispara cada mensaje y el nivel de verificación de esta última para que esto suceda; el orden de prioridad de los mensajes es el mismo en el que cada uno de ellos aparece en la tabla.

VBD	NVVD/NM	TEXTO DEL MENSAJE MÓVIL A DESPLEGAR AL VERIFICARSE LA VBD ASOCIADA.
I30	BAJO/1	Tanques de almacenamiento de líquidos A y B vacíos.
S04	ALTO/2	Proceso de mezclado en tanque C activado.
S05	ALTO/3	Descarga de tanque mezclador en curso, (bomba C activada).

Tabla 4.4 - Relación de mensajes de alarma e informativos a desplegarse en el renglón dos de la UD al llevarse a cabo la automatización del SMLAB empleando el PLM.

Los mensajeros una vez declarados se ven en la figura 4.33 Mensajeros.

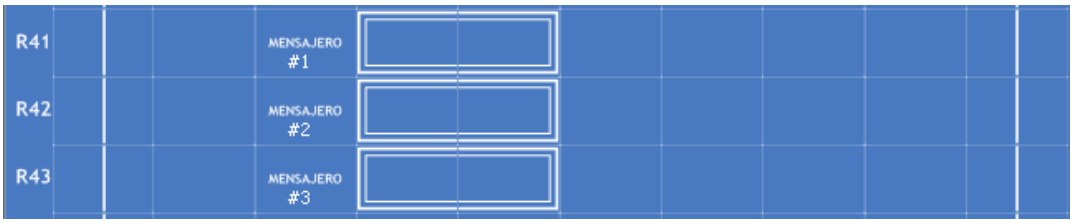


Figura 4.33 - Mensajeros que se asocian a los módulos de alarma.

A continuación se describe la operación lógica empleada para obtener la BVI I30, que dispara el mensaje de prioridad uno.

La VBI I30 se obtiene como la salida de una compuerta OR de dos entradas, que son las VBE E00 Y E01. Como se puede ver en la figura 4.34

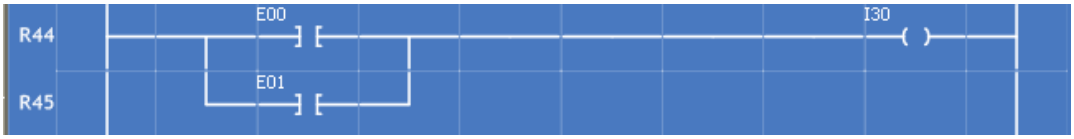


Figura 4.34 - OR de dos entradas E00 y E01 y salida I30.

La declaración de las alarmas se ve en la figura 4.35.

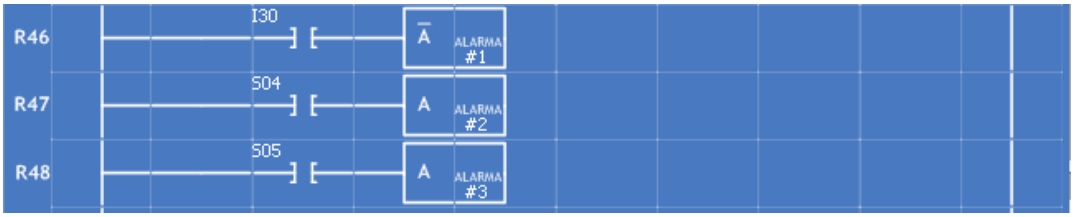


Figura 4.35 - Declaración de los módulos de alarmas.

El bloque de declaración de los módulos mensajeros y alarmas se puede ver completo en la figura 4.36.

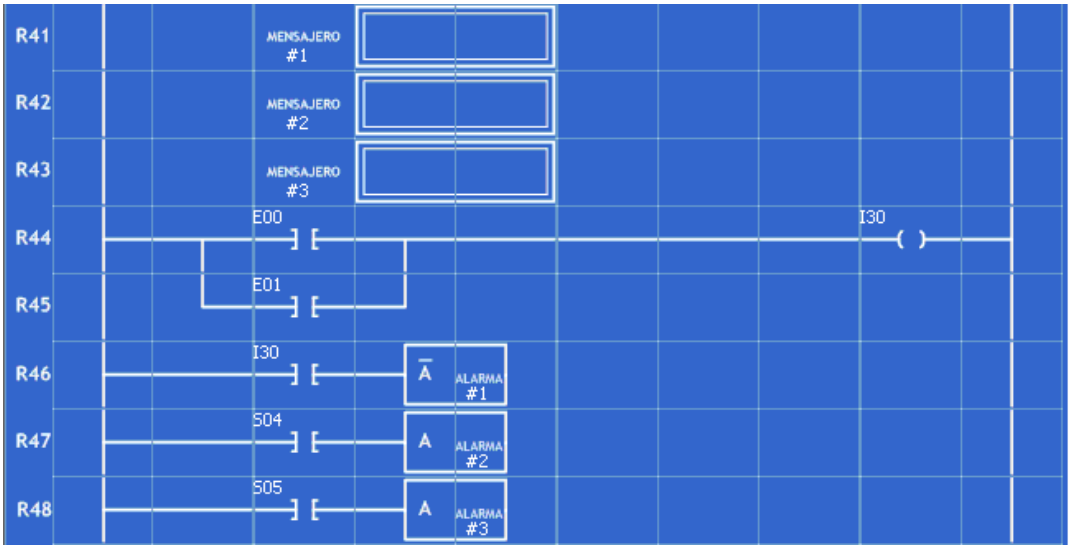


Figura 4.36 - Bloque de declaración de Módulos Mensajeros y Alarmas.

El programa en SILL1 generado por el Software PUMA-ESCALERA que hace que el PLM realice la automatización del SMLAB se muestra a continuación.

INPROG;
AND2#1 I04,I05,I03,11;
AND3#2 I47,I00,E00,S00,111;
AND2#3 I10,I11,I13,11;
AND3#4 I47,I01,E01,S02,111;
AND2#5 I12,I07,I207,11;
NOT#1 I207,I33;
AND2#6 I14,I15,I16,11;
AND2#7 I44,E07,S05,11;
ALARMA#1 I30,0;
ALARMA#2 S04,1;

ALARMA#3 S05,1;

FINPP;

INMODI;
TEMPOC#1 E02,I41,I04,00:00:05.00,100;
TEMPOC#2 E03,I41,I05,00:00:05.00,000;
TEMPOC#3 E06,I41,I06,00:00:00.50,000;
SEC2#1 I03,I41,I06,I07,S01,I00,3,0111;
 # b01, b10;
b00;
CONTA#1 E06,I41,I41,I47,0,20,01110;
OR2#1 S00,S01,S06,11;
TEMPOC#4 E04,I41,I10,00:00:05.00,100;
TEMPOC#5 E05,I41,I11,00:00:05.00,000;
SEC2#2 I13,I41,I06,I12,S03,I01,3,0111;
 # b01, b10;
b00;
OR2#2 S02,S03,S07,11;
TEMPOC#6 I33,I41,I14,00:00:05.00,000;
TEMPOC#7 S04,I41,I15,00:00:05.00,000;
SEC2#3 I16,I41,I06,I17,I44,I02,3,0111;
 # b00, b01;
b10;
TEMPOC#8 I02,I41,S04,00:30:00.00,101;
MENSAJERO#1 "",1,16,20,101;
Tanques de almacenamiento de líquidos A y B vacíos.
MENSAJERO#2 "",1,16,20,101;
Proceso de mezclado en tanque C activado.
MENSAJERO#3 "",1,16,20,101;
Descarga de tanque mezclador en curso, (bomba C activada).
OR2#3 E00,E01,I30,11;

FINMODI;

Si comparamos el código generado por el Software PUMA-ESCALERA con el código SILL1 podemos observar que es igual salvo por la declaración de la compuerta lógica NAND la cual, como ya se había explicado, no existe en el PUMA-ESCALERA sin embargo se puede implementar con la declaración de una compuerta AND y un INVERSOR.

Por lo tanto la declaración de la compuerta lógica NAND en SILL1:

NAND2#1 I12, I07, I33, 11;

Es equivalente a la declaración de la compuerta lógica AND y la inversión de la salida:

AND2#5 I12,I07,I207,11;
NOT#1 I207,I33;

4.2 Segundo Ejemplo De Aplicación

Segundo ejemplo de aplicación, que consiste en realizar con el PLM la lógica requerida por un arrancador de voltaje reducido basado en autotransformador con transición de circuito abierto, para un motor de inducción trifásico [3]; en la industria existen arrancadores de este tipo implantados con lógica alambrada [4]. En la figura 4.37 se ilustra el esquema del conexionado al motor de los elementos actuadores de potencia (contactos), se supone que la secuencia de arranque requiere que los contactos 1A y 2A se cierren al oprimirse el botón de arranque "A" y permanezcan así por tres segundos para abrirse una vez que ha transcurrido este tiempo, después de esto debe haber un tiempo de espera de medio segundo para cerrar el contacto M (marcha); como es usual en este tipo de sistemas se cuenta con un sensor de sobrecarga (OL) que abre sus contactos asociados al detectarse esta condición, además de que se debe contar con un botón de paro (P) de modo que al oprimirse el mismo, el motor sea desconectado del suministro trifásico que lo alimenta, se supone que el tap seleccionado para los autotransformadores es el adecuado para lograr el par de arranque requerido por la carga mecánica que mueve el motor.

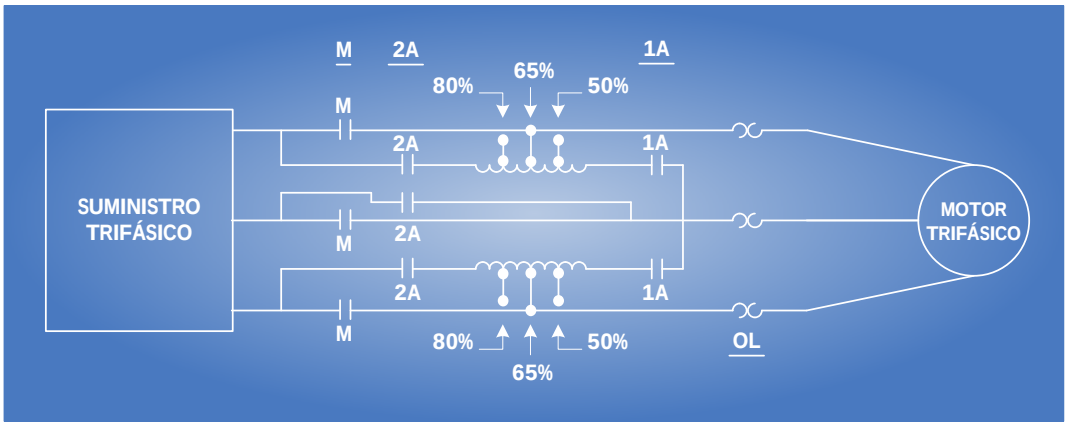


Figura 4.37- Conexionado de arrancador de voltaje reducido basado en un autotransformador cuya secuencia de arranque se ha de implantar con el PLM.

Una forma de realizar la secuencia de arranque con módulos lógicos realizables por el PLM se ilustra en la figura 4.38, apreciándose en la misma el empleo de cinco módulos los cuales son: una compuerta AND de dos entradas, un Flip-Flop RS con prioridad al RESET, un temporizador monodisparo (one-shot) con salida verificada en alto y duración de tres segundos con disparo por flanco de subida, un temporizador con retardo de activación (ON DELAY) con retardo de 3.5 segundos y un seguidor lógico; en la misma figura se muestran las variables booleanas empleadas y con que elementos físicos están relacionadas, en la figura 4.39 se ilustra el conexionado al PLM de los sensores y actuadores requeridos en esta aplicación.

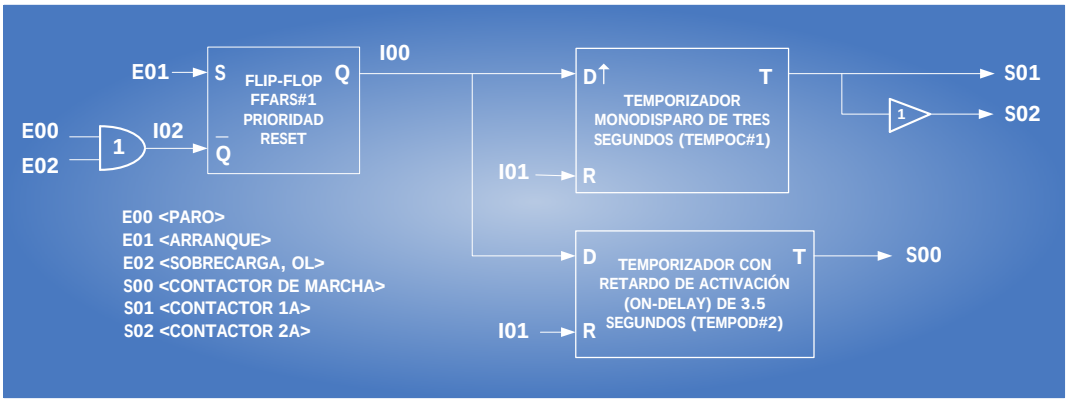


Figura 4.38 - Implantación de la secuencia de arranque requerida, empleando módulos realizables por el PLM.

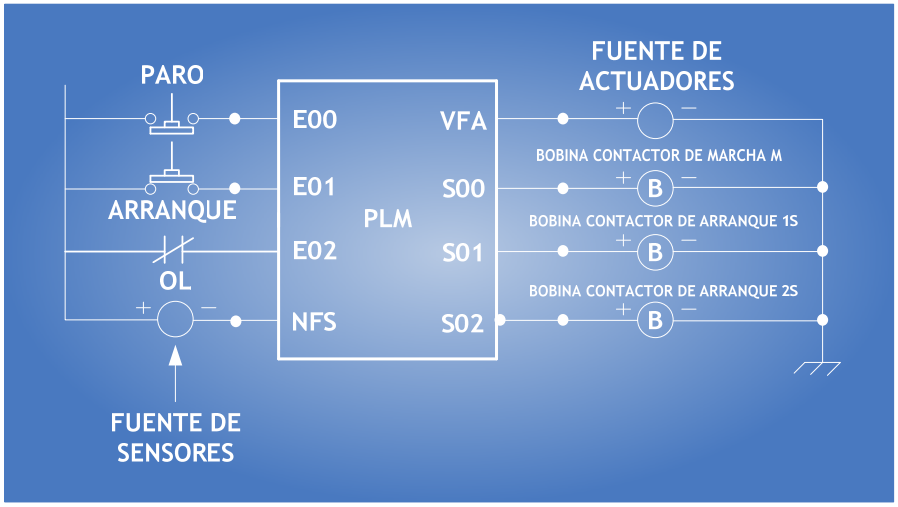


Figura 4.39 - Conexionado al PLM de los sensores y actuadores requeridos en esta aplicación.

Para un mismo tipo de módulo el lenguaje SIIL1 permite especificar entre otras cosas, los niveles de verificación de las entradas y salidas que el mismo pudiera contener; de esta manera, los niveles de verificación para las entradas “S” y “R” del Flip- Flop se especificaron como alto y bajo respectivamente, que es lo adecuado para este caso, el nivel de verificación de las entradas de RESET para los dos temporizadores se definió como alto, el nivel de verificación para las salidas de los mismos se definió como alto, nótese el empleo de la VBI I01 como señal de RESET para ambos temporizadores, cabe señalar aquí que siempre que se inicia la ejecución de un programa en el PLM, los buffers asociados con las variables booleanas son inicializados con ceros, de esta manera, una VBI que no sea empleada como salida por algún módulo tendrá siempre un nivel de cero lógico.

El programa en SILL1 necesario para este arrancador de voltaje es el siguiente:

INPROG;
AND2#1 E00, E02, I02, 11;
SEG#1 S01, S02;
FFARS#1 E01, I02, I00, 1000;
FINPP;
INMODI;
TEMPOC#1 I00, I01, S01, 00:00:03.00, 101;
TEMPOD#2 I00, I01, S00, 00:00:03.50, 10;
FINMODI;

4.2.1 MODELADO DEL PROCESO DESCRITO EN EL PUNTO 4.2 UTILIZANDO EL PUMA-ESCALERA.

Primero se inserta la compuerta lógica AND con entradas físicas E00 y E02 y la salida I02 como se muestra en la figura 4.40.

Figura 4.40 - Compuerta lógica AND de dos entradas E00 y E02 con una salida I02.

A continuación se debe insertar el FFASR con entradas E01 y la salida de la compuerta AND que es I02, u salida VBI I00. Ver Figura 4.41.

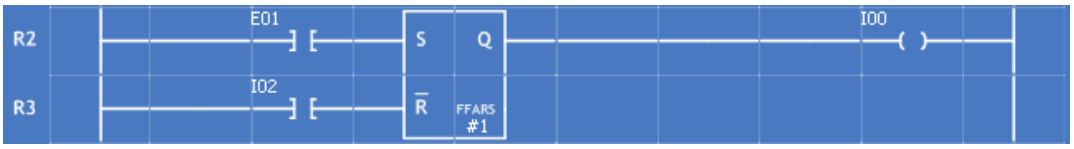


Figura 4.41 FFASR#1

La salida del FFASR#1 deberá ser entonces la entrada del TEMPOC#1 y del TEMPOD#2. En el TEMPOC#1 adicionalmente estará como entrada la VBI I01. Y tendrá como salida S01. Su implementación esta mostrada en la figura 4.42.

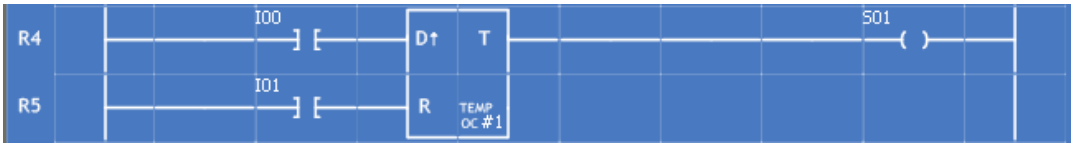


Figura 4.42 - TEMPOC#1.

El temporizador TEMPOD#2 llevará por entradas las VBI I00 e I01; su salida será S00. Lo anterior se observa en la figura 4.43

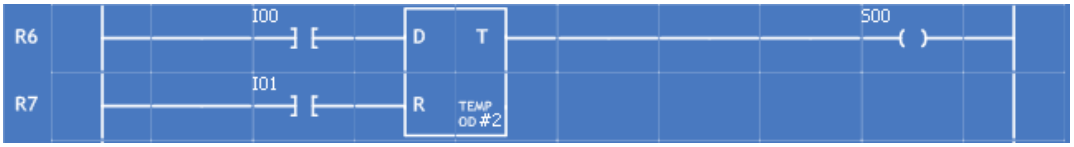


Figura 4.43 - TEMPOD#2.

La salida S01 del TEMPOC#1 se usará como entrada a un seguidor el cual se verá reflejado en al salida física S02. Ver figura 4.44.

Figura 4.44 - Seguidor de la salida física S01.

De modo que el diagrama de escalera de este ejemplo de aplicación es el mostrado en la figura 4.45.

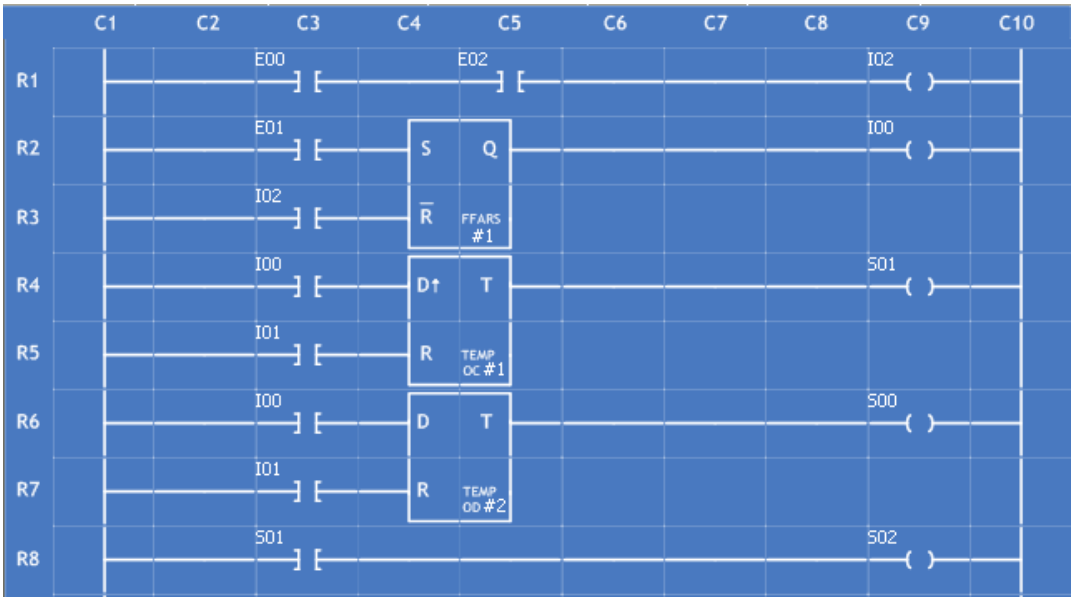


Figura 4.45 - Diagrama de escalera de todos los módulos del ejemplo.

El código generado en lenguaje SILL1 por el PUMA-ESCALERA es el siguiente.

INPROG;
AND2#1 E00,E02,I02,11;
FFARS#1 E01,I02,I00,1000;
SEG#1 S01,S02;
FINPP;

INMODI;
TEMPOC#1 I00,I01,S01,00:00:03.00,101;

TEMPOD#2 I00,I01,S00,00:00:03.50,10;
FINMODI;

Si comparamos el código generado por el Software PUMA-ESCALERA con el código SILL1 podemos observar que es igual, aunque el orden de las sentencias no es el mismo, las diferencias no infieren en la ejecución del programa.

CONCLUSIONES

Tomando como base el objetivo propuesto al inicio de este trabajo de tesis, el desarrollo del Software PUMA-ESCALERA ha finalizado exitosamente. Él software generado permite al usuario definir diagramas de escalera, y a partir de éste generar un archivo de texto con extensión: ".SIL", el cual contiene la traducción a lenguaje SIIL1 del diagrama de escalera que el usuario ha desarrollado.

Durante el diseño del software se optó por hacer la inserción de los objetos, en la pantalla creando una estructura de renglones y columnas fijos, cada renglón cuenta con 10 columnas y cada una de esas columnas es un objeto (en Visual Basic el objeto que elegimos es un PictureBox). La inserción de un modulo consta de varios objetos o columnas y/o renglones. Al elegir este diseño, la traducción, la estructura de los bloques de inserción (FFARS, tempoc, etc.), así como el diseño del compilador se simplificó de manera considerable, sin embargo, implica que cada renglón deba ser declarado previamente y por ende el número de renglones es finito. Adicionalmente otra complicación que se generó es que al ser imágenes la unidad con la que todo se genera, existen algunos bloques que tardan en cargarse y su manejo se hace un poco pesado para equipos con poca memoria RAM (menos de 256 MB de RAM).

El software cuenta con 50 renglones en el área de trabajo, para generar los diagramas de escalera. Para fines del desarrollo de esta tesis, dicho número de renglones es considerado como suficiente.

Debe tomarse en cuenta que el software debe ser instalado sobre plataformas Windows (probado en Windows Me, Windows XP y Windows Vista). Esto debido a que se uso como lenguaje de desarrollo Visual Basic 6.0 y debido a que el software genera un archivo con extensión "SIL", y una vez hecho esto se debe compilar en un programa que solo se ejecuta en plataformas Windows.

Las facilidades de programación que se tienen con Visual Basic, permitieron que el desarrollo del software fuese en un tiempo corto, esto es una de las razones por la que fue elegido este lenguaje..

Tomando en cuenta todas las circunstancias que se dieron durante el desarrollo del software se observa que si el desarrollo del software se iniciará el día de hoy con la experiencia adquirida y sin restricción alguna de tiempo, el desarrollo sería con un lenguaje de programación diferente, y no se utilizaría como unidad de diseño objetos de imagen. Cabe mencionar que una de las cosas que quedan como lección en el desarrollo de este Software, es que la elección del lenguaje de programación y la forma en la que se desarrolla un software, puede ir cambiando dependiendo de la experiencia del desarrollador y de las herramientas existentes del momento. Pues lo que es optimo el día de hoy, puede no serlo el día de mañana.

ANEXO A

MANUAL DE USUARIO

La intención de este manual es adentrar al usuario lo más pronto posible al uso adecuado del programa **"PUMA ESCALERA"**, cuyo objetivo es la traducción del Lenguaje de Escalera al lenguaje de programación SIIL1. A través de este tutorial se usarán términos sencillos, tratando de hacer más amena la utilización del software. El PUMA ESCALERA es una herramienta innovadora que sirve de enlace entre el diagrama de escalera y el Programador Lógico Modular (PLM).

A.2. Interfaz Del PUMA ESCALERA

La figura A.1 muestra la interfaz principal del programa, la cual se divide en tres secciones:

- La primera consta de botones que interactúan con el usuario constantemente, ya sea para solicitarle información o simplemente para indicarle que ha cometido algún error a la hora de ingresar los datos, los llamaremos BOTONES DE INTERACCIÓN, entre los que se encuentran MODULOS LÓGICOS, TEMPORIZADORES, CONTADOR DE EVENTOS, SECUENCIADOR DE ESTADOS, FLIP-FLOP ASÍNCRONO, etc. Más adelante en este manual se describirán a detalle cada uno de estos botones, pero lo más importante que se debe saber es que con ellos se podrá configurar cada módulo que se vaya a insertar en la zona de trabajo.

- La segunda sección y quizá la más importante es la ZONA DE TRABAJO que es donde se realizará el diagrama de ESCALERA, está constituida por RENGLONES (nombrados desde R1 hasta R50) y COLUMNAS (nombradas desde C1 hasta C10) para facilitar la localización de figuras.

  En la zona de trabajo aparecerán datos como el número de MÓDULO insertado, el nombre de las entradas o salidas (que corresponden con las variables booleanas de entrada o de salida), por sólo nombrar los más importantes, éste programa fue creado esencialmente como una herramienta de dibujo, algunos datos que se ingresen a la hora de configurar cada módulo no aparecerán en la pantalla por lo que se recomienda tener los datos a la mano si ocasionalmente surge algún olvido, como por ejemplo, cuando se ABRA un proyecto creado anteriormente.

- El tercer conjunto lo forman las HERRAMIENTAS, las cuales podemos subdividir en tres partes: las primeras son herramientas para el manejo de archivos, como lo son las ya conocidas ABRIR, GUARDAR y SALIR. Las segundas se basan principalmente en la ayuda al usuario, así, el botón BORRAR tiene la facultad de borrar renglones, conectores, módulos e incluso toda la zona de trabajo, el botón GUÍA le conducirá inmediatamente a este manual cuyo propósito, como ya se dijo anteriormente, es introducir al usuario lo más pronto

posible al manejo del software, el botón IMPRIMIR ayudará en las tareas de revisión del proyecto de una manera más práctica y analítica ya que imprimirá la zona de trabajo. Y finalmente las que se utilizan para el proceso de compilación y traducción, todo este proceso es realizado por el botón COMPILAR que es como un VTP (Viaje Todo Pagado), ya que realiza múltiples funciones, desde rellenar renglones vacíos hasta la generación del archivo SIIL1.

Todo este panorama es una explicación a grandes rasgos de lo que puede lograr el PUMA ESCALERA, como toda versión 1.0 algún usuario que ahora mismo esté leyendo este texto pudiera encontrar errores en el software. Consideramos por supuesto la existencia de Service Pack's, sin embargo no creemos superar en número a los que ya tiene Windows.

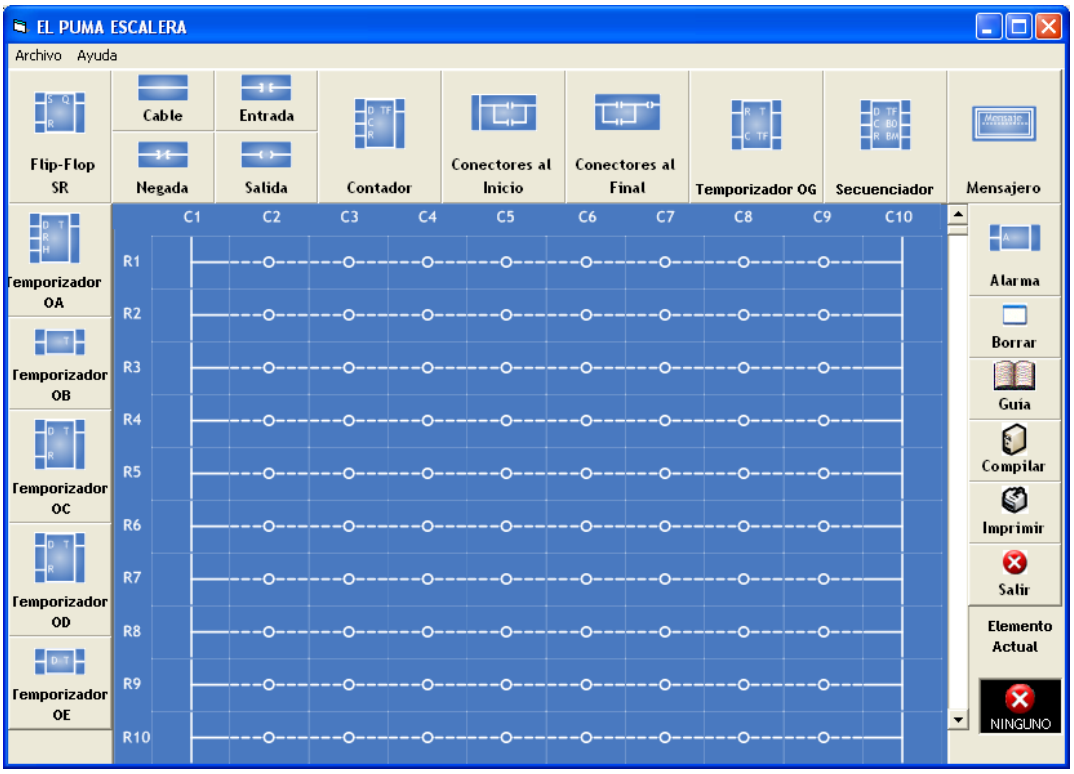


Figura A.1 – Interfaz del PUMA ESCALERA

A.3. Creación Del Diagrama De Escalera

Dado que este software esta diseñado para la traducción de un diagrama de ESCALERA a LENGUAJE SILL lo más adecuado sería comenzar dibujando ese diagrama de escalera, para éste propósito necesitamos la ayuda de los botones de interacción y la zona de trabajo.

A.3.1 *Seguidor*

Este Módulo Lógico simplemente coloca en la Variable Booleana declarada como salida, el nivel lógico que exista en la Variable Booleana declarada como entrada al mismo. Un SEGUIDOR se representa en la figura A.2:

Figura A.2 - Seguidor

Su representación mediante diagrama de escalera la podemos ver en la figura A.3:

Figura A.3 – Seguidor en el Diagrama de Escalera

El módulo más sencillo a dibujar en el diagrama de ESCALERA es el SEGUIDOR, ya que sólo cuenta con 2 elementos: una entrada y una salida, para lograrlo sólo se necesita tener un renglón disponible como lo muestra la figura A.4:

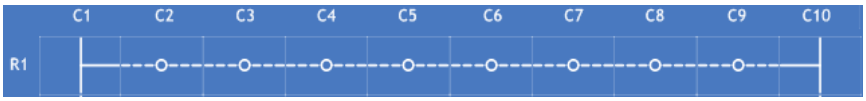


Figura A.4 – Renglón individual para el PUMA ESCALERA

Como se puede apreciar en la figura A.4 cada renglón esta constituido por 10 columnas, cada pequeño rectángulo es una columna, el SEGUIDOR podrá ser colocado en siete rectángulos centrales del renglón, es decir, en la columna C2, C3, C4, C5, C6, C7 o C8. Como ayuda para que el usuario identifique claramente los rectángulos se puede observar un pequeño círculo seguido de líneas suspensivas a sus costados en el centro de cada rectángulo.

El primer paso para comenzar el dibujo es dar clic en un BOTÓN DE INTERACCIÓN, para el caso del SEGUIDOR será el botón de ENTRADA mostrado en la figura A.5:

Figura A.5 – Botón para insertar una Entrada

Este botón se encuentra en la parte superior izquierda de la interfaz principal y como veremos más adelante es utilizado en la mayoría de los módulos del diagrama de escalera.

Una vez que hayamos dado clic aparecerá una ventana de diálogo que permitirá elegir la variable booleana asociada al SEGUIDOR que de acuerdo al PLM podrán ser Variables Booleanas de Entrada (VBE), Variables Booleanas de Salida (VBS) o Variables

Booleanas Intermediarias (VBI). Antes de continuar con nuestra explicación es pertinente manifestar una definición de cada una de ellas.

Variables Booleanas de Entrada (VBE). El primer prototipo del PLM está pensado para manejar 32 VBE's. Cada VBE se especifica con tres caracteres, el primero es una letra "e" mayúscula o minúscula, el segundo es un número del cero al tres que denota el grupo al que pertenece la VBE y el tercero es un número del cero al siete que define el bit asociado del puerto de entrada relacionado con el grupo de entradas de que se trate; así por ejemplo, la quinta variable del grupo de entradas dos se podría denotar como E25.

Variables Booleanas de Salida (VBS). Existen para el PLM 16 variables booleanas de salida, las VBS están aglutinadas en dos grupos de ocho salidas cada uno; de esta forma, se emplean tres caracteres para denotar a una VBS, el primero es la letra "s" mayúscula o minúscula, el segundo es un número que puede ser cero o uno que denota el número de grupo de salida y el tercero es un dígito del cero al siete que define el bit asociado con el puerto de salida físico; así por ejemplo, la salida dos del grupo de salidas cero se podría definir como S02.

Variables Booleanas Intermediarias (VBI). Este tipo de variables son manejadas internamente y no tienen entradas o salidas físicas asociadas, su función consiste en servir de enlace entre módulos lógicos cuando esto sea necesario; por ejemplo, supóngase que se tiene una situación de control lógico que requiere de varias compuertas lógicas, puede suceder que las salidas de algunas de ellas sean variables requeridas como entrada de otras, el emplear variables físicas de salida para habilitar esta circunstancia no sería conveniente dado que su número es limitado, de ahí la necesidad de contar con las VBI; desde luego que una variable booleana, que sea entrada de un módulo y salida de otro, puede ser una salida física si esto es necesario, lo que no es permitido es el hecho de que una variable sea simultáneamente salida de más de un módulo.

Las VBI están agrupadas en 21 grupos de ocho variables cado uno, de esta forma se puede contar dentro del PLM con 168 variables de este tipo; la notación empleada para designarlas emplea cuatro caracteres, el primero es la letra "i" mayúscula o minúscula, el segundo y tercero representan a un número comprendido entre cero y veinte que denota el número de grupo al que pertenece la VBI y el cuarto es un dígito del cero al siete que define el número de VBI dentro del grupo; así por ejemplo, la VBI cuatro del grupo doce de VBI's se denominaría I124.

Así, existen múltiples opciones para denotar al SEGUIDOR. Al elegir una variable booleana de Entrada se tendrá un gran abanico de posibilidades empezando desde E00 hasta E37. De igual manera al elegir una variable booleana de Salida la sintaxis comenzará desde S00 hasta S17, es decir, sólo habrá 14 opciones. Si se elige la variable booleana Intermediaria tendrá aún un catálogo más extenso que irá desde I00 hasta I207.

Al dar clic en el botón ENTRADA aparecerá el cuadro de diálogo mostrado en la figura A.6:

Figura A.6 – Ventana para configurar Variables de Entrada

Como lo muestra la figura A.6 hay dos opciones a escoger: “Variables de Entrada” y “Variables de Salida”. Al desplegar la lista de las Variables de Entrada aparecerán Variables Booleanas de Entrada y Variables Booleanas Intermediarias, como lo indican las siguientes figuras (figura A.7 y A.8):

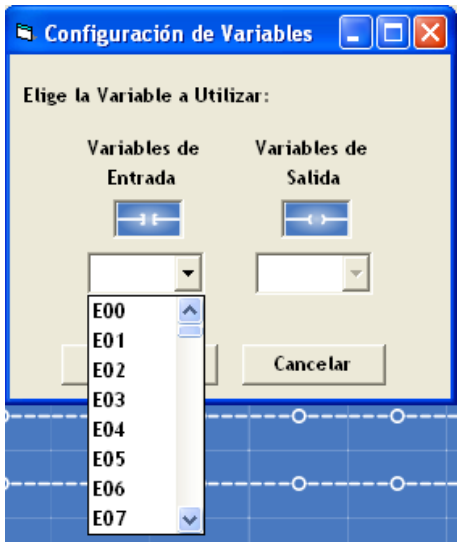


Figura A.7 – Menú de Variables Booleanas de Entrada en la ventana: “Configuración de Variables”

Figura A.8 - Menú de Variables Booleanas Intermediarias en la ventana: "Configuración de Variables"

En la figura A.7 se observa una lista desplegada de variables booleanas de Entrada que van desde E00 hasta E37. En la figura A.8 se muestran las variables booleanas Intermediarias enlistadas desde I00 hasta I207, las cuales se encuentran enseguida de las VBE, para poder seleccionar cualquier variables sólo hay que mover la barra de desplazamiento hacia arriba o abajo para poder elegirla.

Una vez configurado el SEGUIDOR, daremos clic en el botón "Insertar" como lo muestra la figura A.9:

Figura A.9 – Selección de una Variable Booleana de Entrada en la ventana: "Configuración de Variables"

Hasta ahora, se ha configurado el SEGUIDOR, a continuación: se insertará la figura en la zona de trabajo. Como ya se mencionó, la zona de trabajo consta de 50 renglones, conformados cada uno por 10 columnas. Por razones obvias, solo se muestran 10 renglones de manera simultánea.

La ZONA DE TRABAJO En el costado izquierdo de cada renglón aparece su nombre abreviado, así, el renglón 1 se encuentra abreviado como R1, el renglón 2 como R2, etc. Lo mismo sucede con las columnas, la columna 1 esta abreviada como C1, la columna 2 como C2, etc. Lo explicado se puede apreciar mejor en la Figura A.10:

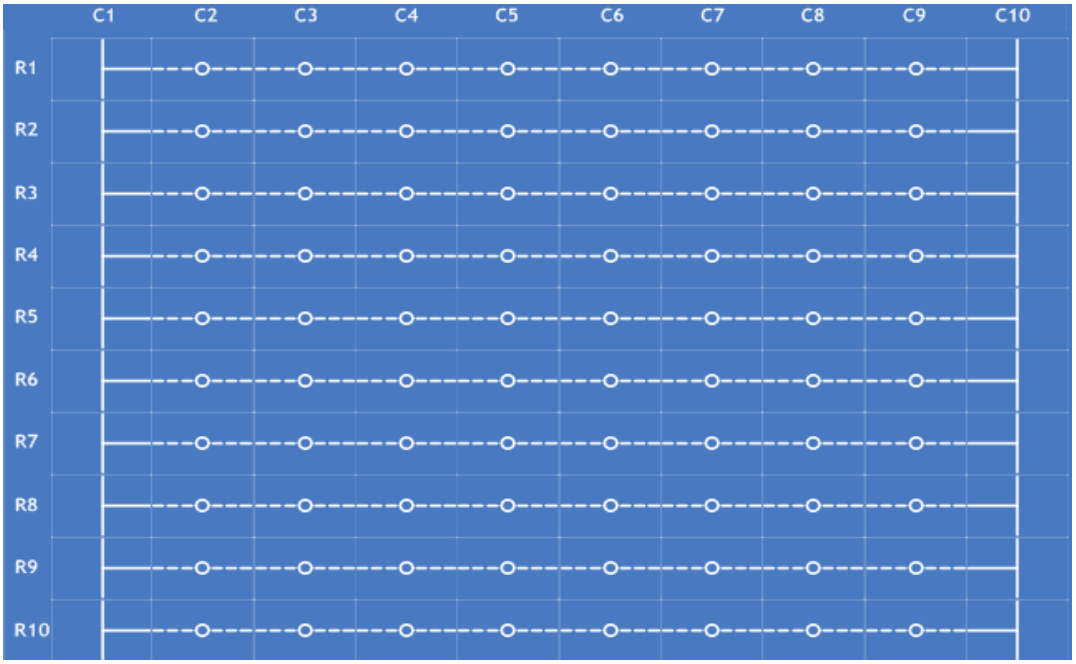


Figura A.10 – Zona de Trabajo en el PUMA ESCALERA.

En la parte inferior derecha tenemos una herramienta que se utiliza para saber que figura se va a insertar. Esta herramienta nos indica el elemento actual que se esta configurando (Figura A.11). Como estamos utilizando una ENTRADA para dibujar el SEGUIDOR el símbolo de entrada aparecerá en el elemento actual será una entrada:

Figura A.11 – Imagen que muestra el elemento actual a configurar o insertar, en este caso una Entrada.

A continuación deberá elegir el renglón donde se insertará la figura, como se puede observar en la interfaz principal existe una BARRA DE DESPLAZAMIENTO VERTICAL que permite desplazarse a través de todos los renglones y así seleccionar el renglón a utilizar. Se recomienda comenzar a insertar los elementos desde el renglón 1 (R1), esto permite tener un orden en la elaboración de los diagramas.

Siguiendo la recomendación anterior se colocará la figura en el renglón 1 y en la columna 2, es decir, Se dará UN CLIC CON EL BOTÓN IZQUIERDO DEL RATÓN EN LAS COORDENADAS (R1, C2) que es el primer lugar disponible del renglón. Se

le asignará la VBE, que en este ejemplo será E00. R1 resultará como lo muestra la figura A.12:

Figura A.12 – Inserción de la entrada E00 en la columna C2.

Como se observa en la figura anterior la columna C2 cambio su apariencia default por la de una ENTRADA, además, en su costado superior izquierdo se desplegó el nombre de la VBE que se le asignó.

Dado que un SEGUIDOR consta de una ENTRADA y una SALIDA el evento siguiente es insertar la salida, para tal efecto será necesario dar CLIC EN EL BOTÓN SALIDA (figura A.13) de la interfaz principal.

Figura A.13 – Botón para insertar una Salida.

Este botón se encuentra en la parte superior izquierda de la ventana. Al dar clic en él se desplegará la pantalla mostrada en la figura A.14:

Figura A.14 – Ventana para configurar Variables de Salida.

La figura A.14 muestra dos opciones a seleccionar: “Variables de Entrada” y “Variables de Salida”, pero Variables de Entrada aparece deshabilitada, por convención sólo se pueden utilizar variables booleanas de salida (figura A.15) y variables booleanas intermediarias (figura A.16) al configurar una salida:

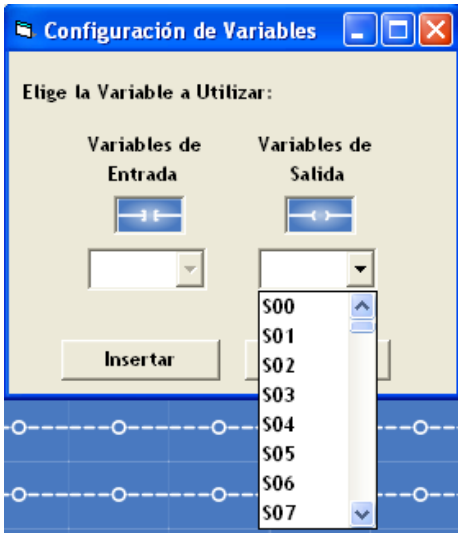


Figura A.15 – Menú de Variables Booleanas de Salida en la ventana: "Configuración de Variables".

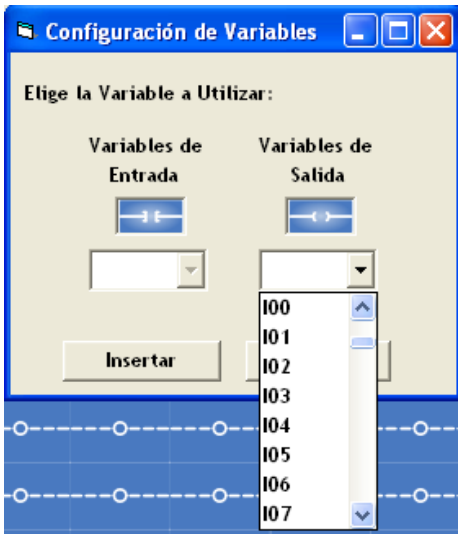


Figura A.16 – Menú de Variables Booleanas Intermediarias en la ventana: "Configuración de Variables".

En la figura A.15 se observa una lista desplegada de variables booleanas de Salida que van desde S00 hasta S17. En la figura A.16 se muestran las variables booleanas Intermediarias enlistadas desde I00 hasta I207, las cuales se encuentran enseguida de las VBS, para poder seleccionar cualquiera de las variables sólo hay que mover la barra de desplazamiento hacia arriba o abajo para poder elegirla.

Una vez seleccionada la VB correspondiente a la salida, hacer clic en el botón "Insertar" (figura A.17).

Figura A.17 – Selección de una Variable Booleana de Salida en la ventana: "Configuración de Variables".

La ventana "Configuración de variables" desaparecerá y la zona de trabajo se hará completamente visible. Podemos insertar la SALIDA en el renglón 1.

IMPORTANTE: LAS SALIDAS SÓLO PODRÁN SER INSERTADAS EN LA COLUMNA 9 (C9) DEL RENGLÓN ASOCIADO AL MÓDULO SELECCIONADO. DE NO SER ASÍ, EL PROGRAMA MARCARÁ ERROR EN TIEMPO DE COMPILACIÓN.

Teniendo en cuenta las consideraciones anteriores, la SALIDA se insertará en R1,C9 DANDO CLIC CON EL BOTÓN IZQUIERDO DEL RATÓN en la columna 9 del renglón 1, dando como resultado lo que se aprecia en la figura A.18:

Figura A.18 – Inserción de la Salida S00 en la columna C9.

La figura A.18 se muestra un renglón con la estructura de un SEGUIDOR, sin embargo, el proceso no ha terminado aún. Las columnas C3, C4, C5, C6, C7 y C8 aún conservan la apariencia default que se les asigna al iniciar el programa. El software no reconocerá este renglón como un SEGUIDOR, pues detectará que estas columnas se encuentran vacías y marcará ERRORES en tiempo de compilación. Para finalizar se deberá UNIR la ENTRADA E00 con la SALIDA S00. Esto se consigue mediante el botón CABLE (figura A.19) de la ventana principal:

Figura A.19 - Botón para insertar un cable.

Este botón se encuentra en la parte superior izquierda de la ventana. Al dar clic en él no mostrará ningún cuadro de diálogo ya que sólo es una herramienta que tiene la utilidad de unir módulos que así lo necesiten. Como indicativo de que se está manipulando éste

elemento, en la pantalla de ELEMENTO ACTUAL se mostrará la imagen del elemento cable, tal como lo muestra en la figura A.20:

Figura A.20 – Imagen que muestra el elemento actual a configurar o insertar, en este caso un Cable.

Para implementar la UNIÓN que necesitamos se deberá dar clic con el botón izquierdo del ratón en la columna C3. Repetiremos el mismo procedimiento para las columnas C4, C5, C6, C7 y C8. No será necesario dar clic en el botón CABLE cada vez que se utilice, es posible dar clic en cada columna en la que sea necesario hasta finalizar la UNIÓN.

Una vez terminado el proceso de unión, el SEGUIDOR quedará completamente dibujado, como lo muestra la figura A.21:

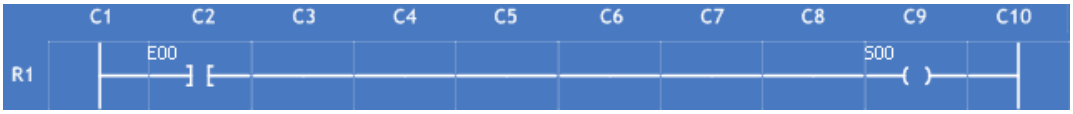


Figura A.21 – Renglón R1 después de la inserción de elementos en todas las columnas para formar un SEGUIDOR.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA: E00
SALIDA: S00

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

SEG#1 E00,S00

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.2 *Inversor*

Este Módulo Lógico simplemente pone en la Variable Booleana declarada como salida el nivel lógico opuesto al que exista en la Variable Booleana declarada como entrada al mismo. Un INVERSOR se representa en la figura A.22:

Figura A.22 - Inversor

La representación en diagrama de escalera del inversor es la mostrada en la figura A.23:

Figura A.23 – Inversor en el Diagrama de Escalera.

El proceso de inserción de un INVERSOR a la zona de trabajo es prácticamente el mismo que el utilizado para el SEGUIDOR. La única diferencia notable es que ahora se deberá insertar una ENTRADA NEGADA en lugar de una ENTRADA, el símbolo mostrado en la figura A.24 se le conoce en el diagrama de ESCALERA como ENTRADA NEGADA.

Para configurar un INVERSOR. Habrá que dar clic en el botón llamado NEGADA (Figura A.24) ubicado en la parte superior izquierda de la ventana principal.

Figura A.24 – Botón para insertar una Entrada Negada

Este botón mostrará un cuadro de diálogo similar al del botón ENTRADA y el botón SALIDA:

Figura A.25 Figura 3.14 – Ventana para configurar Variables de Entrada Negadas.

Como lo muestra la figura A.25 hay dos opciones a escoger: “Variables de Entrada” y “Variables de Salida”, se ha deshabilitado la opción “Variables de Salida” para evitar que se terminen las VBS. Al desplegar la lista de las Variables de Entrada aparecerán Variables Booleanas de Entrada y Variables Booleanas Intermediarias, como lo indican las figuras A.26 y A.27:

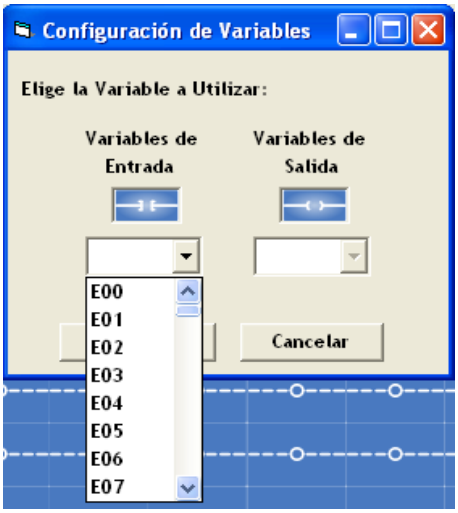


Figura A.26 – Menú de Variables Booleanas de Entrada Negadas en la ventana: "Configuración de Variables".

Figura A.27 – Menú de Variables Booleanas Intermediarias en la ventana: "Configuración de Variables".

En la figura A.26 se observa una lista desplegada de variables booleanas de Entrada que van desde E00 hasta E37. En la figura A.27 se muestran las variables booleanas Intermediarias enlistadas desde I00 hasta I207, las cuales se encuentran enseguida de las VBE, para poder seleccionar cualquiera de estas variables sólo hay que mover la barra de desplazamiento hacia arriba o abajo para poder elegirla.

Una vez que se haya elegido la VBE para el INVERSOR se tendrá que dar clic en el botón "Insertar" como lo muestra la figura A.28:

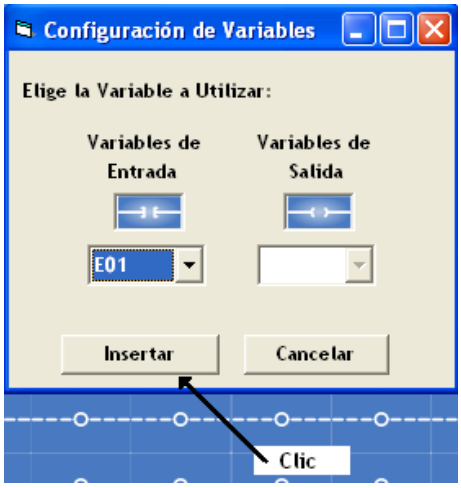


Figura A.28 – Selección de una Variable Booleana de Entrada Negada en la ventana: "Configuración de Variables".

Por ejemplo, se elegirá el renglón 2 (R2) y la columna 4 (C4) para insertar la NEGADA. DAR UN CLIC CON EL BOTÓN IZQUIERDO DEL RATÓN EN LAS COORDENADAS (R2, C4). Se asignará la VBE E01. R2 quedará como lo muestra la figura A.29

Figura A.29 – Inserción de la entrada negada E01 en la columna C4.

Como se observa en la figura A.29 la columna C4 cambio su apariencia default por la de una NEGADA, a si mismo, en su costado superior izquierdo apareció la VBE asignada.

Dado que un INVERSOR consta de una ENTRADA NEGADA y una SALIDA el evento siguiente es insertar la salida, para éste propósito simplemente se tendrá que DAR CLIC EN EL BOTÓN SALIDA (Figura A.30) de la interfaz principal.

Figura A.30 – Botón para insertar una Salida.

Como se observo previamente se desplegará una pantalla como la mostrada en la figura A.31:

Figura A.31 – Ventana para configurar Variables de Salida.

Cabe recordar que por convención sólo se pueden utilizar variables booleanas de salida (figura A.32) y variables booleanas intermediarias (figura A.33) para configurar una SALIDA:

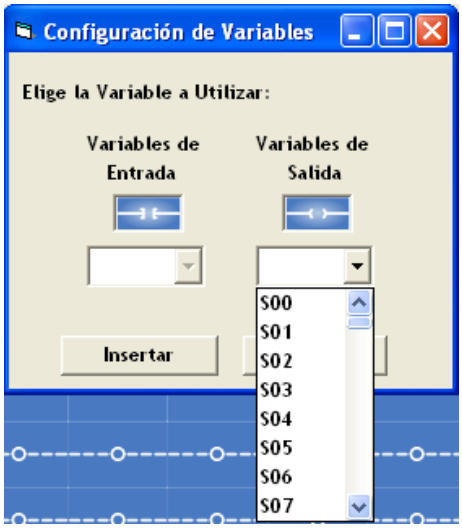


Figura A.32 - Menú de Variables Booleanas de Salida en la ventana: “Configuración de Variables”.

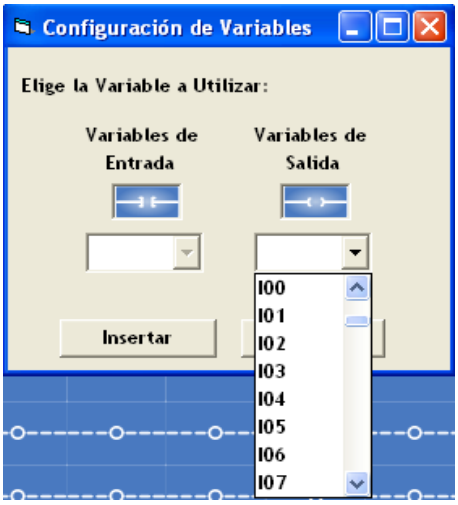


Figura A.33 – Menú de Variables Booleanas Intermediarias en la ventana: "Configuración de Variables".

En la figura A.32 se observa una lista desplegada de variables booleanas de Salida que van desde S00 hasta S17. En la figura A.33 se muestran las variables booleanas Intermediarias enlistadas desde I00 hasta I207, las cuales se encuentran enseguida de las VBS, para poder seleccionar cualquier variables sólo hay que mover la barra de desplazamiento hacia arriba o abajo para poder elegirla.

Una vez configurada la salida, se deberá dar clic en el botón "Insertar" (figura A.34).

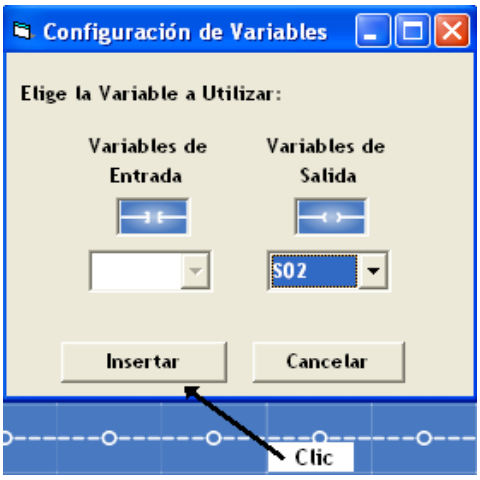


Figura A.34 – Selección de una Variable Booleana de Salida en la ventana: "Configuración de Variables".

La ventana "Configuración de variables" desaparecerá y la zona de trabajo se hará completamente visible.

IMPORTANTE: LAS SALIDAS SÓLO PODRÁN SER INSERTADAS EN LA COLUMNA 9 (C9) DEL RENGLÓN ASOCIADO AL MÓDULO SELECCIONADO. DE NO SER ASÍ, EL PROGRAMA MARCARÁ ERROR EN TIEMPO DE COMPILACIÓN.

Para insertar la SALIDA se deberá dar CLIC CON EL BOTÓN IZQUIERDO DEL RATÓN en la columna 9 del renglón 2.

Figura A.35 – Inserción de la Salida S02 en la columna C9.

La figura A.35 muestra un renglón con la estructura de un INVERSOR, sin embargo, el proceso no ha terminado. Las columnas C2, C3, C5, C6, C7 y C8 aún conservan la apariencia default que se les asigna al iniciar el programa. El software no reconocerá este renglón como un INVERSOR, pues detectará que estas columnas se encuentran vacías y marcará ERRORES en tiempo de compilación. Para concluir se deberá UNIR la ENTRADA NEGADA E01 con la SALIDA S02, así como las columnas C2 y C3. Esto se puede lograr mediante el botón CABLE (Figura A.36) de la ventana principal:

Figura A.36 – Botón para insertar un cable

Al dar clic en él no mostrará ningún cuadro de diálogo ya que sólo es una herramienta que tiene la utilidad de unir módulos que así lo necesiten.

Para implementar la UNIÓN que se requiere, se deberá dar un clic con el botón izquierdo del ratón en la columna C2. Repetir el procedimiento para las columnas C3, C5, C6, C7 y C8. No será necesario dar clic en el botón CABLE cada vez que se utilice.

El INVERSOR quedará como lo muestra la figura A.37

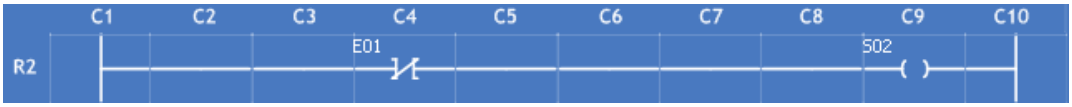


Figura A.37 – Renglón R2 después de la inserción de elementos en todas las columnas para formar un INVERSOR.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA: E01
SALIDA: S02

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

NOT#1 E01,S02

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.3 Compuerta Lógica And

La COMPUERTA LÓGICA AND se representa con el símbolo de la figura A.38 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer una compuerta lógica AND):

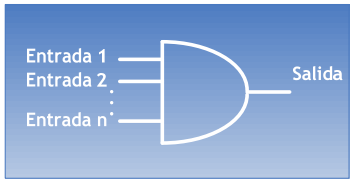


Figura A.38 – Compuerta lógica AND.

Su representación en diagrama de escalera es la mostrada por la figura A.39:

Figura A.39 - Compuerta lógica AND en el Diagrama de Escalera.

Este módulo lógico tiene interesantes aspectos en el DIAGRAMA DE ESCALERA. Como se puede observar en las figuras A.38 y A.39 las estructuras tienen cambios que a primera vista muestran cierta dificultad de entendimiento; sin embargo, guardan similitudes importantes. El orden de las Entradas en la compuerta lógica tiene un sentido vertical mientras que en DIAGRAMA DE ESCALERA guardan un sentido horizontal. En el caso de la salida sólo cambia el símbolo de representación, mientras que en la compuerta lógica es dibujada como una línea horizontal, en el diagrama de escalera es mostrada como un par de paréntesis con líneas horizontales en sus costados.

En el PUMA ESCALERA la compuerta AND puede ser representada mediante un renglón. Es así que las ENTRADAS podrán ser insertadas en las columnas C2, C3, C4, C5, C6, C7 y C8.

EL NÚMERO DE ENTRADAS MÁXIMO PERMITIDO SERÁ DE CUATRO ENTRADAS POR CADA RENGLÓN. Sin olvidar, claro su correspondiente SALIDA en la columna C9.

A continuación se obviarán en algunos aspectos ya expuestos en los anteriores apartados, como son, la inserción de VBE y VBS.

Dar clic en el botón ENTRADA de la ventana principal, una vez desplegado el cuadro de diálogo, seleccionar la VBE de la PRIMERA ENTRADA, el cual podrá ir desde E00 hasta E37 para variables booleanas de entrada e I00 hasta I207 para variables booleanas intermediarias. Ingresar dando clic con el botón izquierdo del ratón en el renglón 1 (R1) y en la Columna 4 (C4), como lo muestra la figura A.40:

Figura A.40 - Inserción de la entrada E00 en la columna C4.

Repetir este paso tantas veces como entradas se requieran, par las salidas habrá que dar clic en el botón SALIDA y una vez desplegado el cuadro de diálogo asignar la VBS, la cual podrá ir desde S00 hasta S17 para variables booleanas de salida y desde I00 hasta I207 para variables booleanas intermediarias. Ingresar la salida, dando clic con el botón izquierdo del ratón en el renglón 1 (R1) y en la columna 9 (C9) obligatoriamente. Un ejemplo seria el mostrado en la figura A.41, la cual muestra una estructura de una AND con dos ENTRADAS:

Figura A.41 - Inserción de la entrada E01 en la columna C2 y la Salida S00 en la Columna C9.

Observando la figura A.41 lo único que nos queda por hacer es realizar el proceso de UNIÓN entre las ENTRADAS y la SALIDA, así como el llenado de columnas vacías que en este caso serán C2 y C3.

La unión y el llenado de columnas vacías, se lleva a cabo con el botón CABLE de la ventana principal, y posteriormente en las columnas C2, C3, C5, C7 y C8. Finalmente el renglón 1 contiene un AND con dos entradas, como el mostrado en la figura A.42:

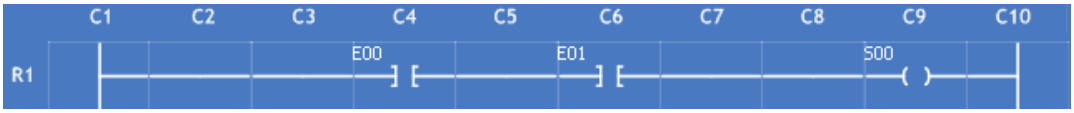


Figura A.42 – Renglón R1 después de la inserción de elementos en todas las columnas para formar una COMPUERTA LÓGICA AND.

Cabe señalar que la posición de las ENTRADAS no estará restringida, es decir, será el usuario quien decida donde ubicarlas, así la entrada E00 pudo haber sido colocada en C2 y la entrada E01 en C3. Pero siempre respetando la posición de la salida que en todos los casos se insertará en la Columna 9 (C9).

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA 1: E00
ENTRADA 2: E01
SALIDA: S00
¿PREINVERSIÓN de la entrada 2?: NO, que esta representado a través del primer dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
¿PREINVERSIÓN de la entrada 1?: NO, que esta representado a través del segundo dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

AND2#1 E00,E01,S00,11;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

¿Pero qué pasa si una de las entradas de la compuerta lógica AND tiene preinversión? La figura A.43 muestra una AND con la entrada 1 preinvertida:

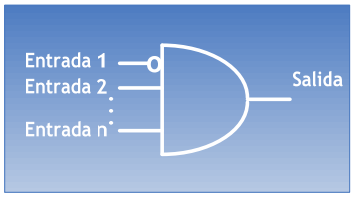


Figura A.43 – Compuerta Lógica AND con una de sus entradas preinvertida.

La respuesta es sencilla, para la Entrada 1 Se usara un símbolo ya conocido: la ENTRADA NEGADA. Así, su representación en el diagrama de escalera es la mostrada en la figura A.44:

Figura A.44 – Compuerta Lógica AND en el Diagrama de Escalera con una de sus Entradas Preinvertida.

La construcción del diagrama de escalera para una compuerta lógica AND de dos entradas siendo una de ellas preinvertida en EL PUMA ESCALERA comenzaría con la inserción de la ENTRADA NEGADA.

Se deberá dar clic en el botón NEGADA de la ventana principal, una vez desplegado el cuadro de diálogo Elegir la VBE de la PRIMERA ENTRADA, el cual podrá ir desde E00 hasta E37 para variables booleanas de entrada e I00 hasta I207 para variables booleanas intermediarias. Elegir una VBE cualquiera y la ingresamos dando clic con el botón izquierdo del ratón en el renglón 2 (R2) y en la Columna 3 (C3), como lo muestra la figura A.45

Figura A.45 – Inserción de la Entrada E02 en la columna C3.

El resto del proceso es similar al expuesto con anterioridad, se selecciona la VBE y se inserta en alguna columna valida. Lo siguiente es configurar e insertar la salida y rellenar el renglón con el botón cable. El resultado debe ser algo similar a lo mostrado en la figura A.46, donde finalmente el renglón 1 tiene una AND con dos entradas, siendo la primera de ellas preinvertida:

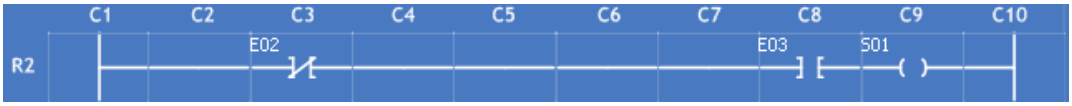


Figura A.46 – Renglón R2 después de la inserción de elementos en todas las columnas para formar una COMPUERTA LÓGICA AND con una de sus Entradas Preinvertida.

Ahora se sabe que cuando se requiera de alguna ENTRADA PREINVERTIDA en una compuerta lógica AND se usará la ENTRADA NEGADA para representarla.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA 1: E02
ENTRADA 2: E03
SALIDA: S01
¿PREINVERSIÓN de la entrada 2?: NO, que esta representado a través del primer dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
¿PREINVERSIÓN de la entrada 1?: SI, que esta representado a través del segundo dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

AND2#1 E02,E03,S01,10;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.3.1 Uso De Variables Intermediarias

Hasta ahora se ha visto el uso de variables booleanas de entrada y variables booleanas de salida en los módulos lógicos que se han insertado. Sin embargo existe la posibilidad de representar una compuerta lógica AND mediante la utilización de variables booleanas intermediarias.

Para poder ejemplificar su uso se representará una compuerta lógica AND de cuatro entradas (**MÁXIMO PERMITIDO**) utilizando el PUMA ESCALERA. Las entradas tienen por nombres E00, E01, E02, E03 respectivamente y una SALIDA llamada S00.

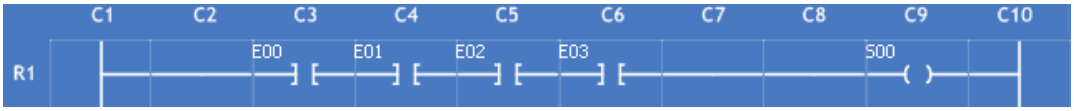


Figura A.47 – Renglón R1, después de la inserción de todas las entradas posibles (también podrían ser negadas) para formar una COMPUERTA LÓGICA AND en el mismo.

Como lo muestra la Figura A.47 comúnmente se utilizaría sólo un renglón de la zona de trabajo para poder representar una AND de cuatro ENTRADAS, no obstante existe la posibilidad de conseguir este mismo objetivo utilizando dos renglones. Y ¿cómo lograr esto?, la respuesta es: utilizando VARIABLES BOOLEANAS INTERMEDIARIAS.

La idea de fragmentar esta representación en dos renglones surge de la división misma de la compuerta AND de cuatro ENTRADAS en dos AND's. La primera AND con 2 ENTRADAS y la segunda AND con tres ENTRADAS. La figura A.48 nos ayudará a entender mejor este concepto:

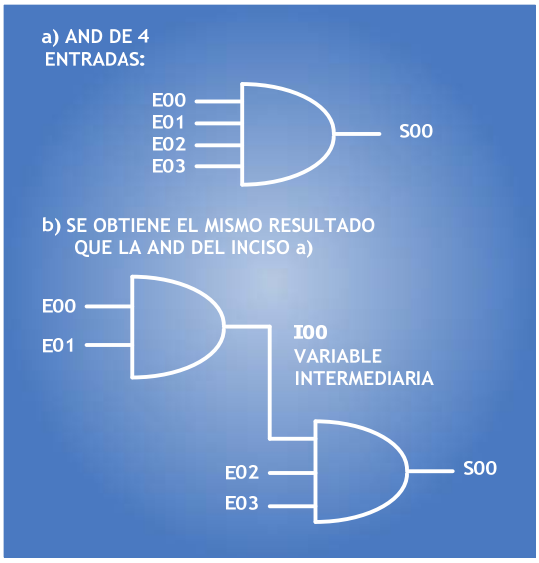


Figura A.48 – a) Compuerta Lógica AND de cuatro Entradas. b) Combinación de 2 compuertas lógicas AND.

El inciso a) de la figura A.48 muestra una compuerta lógica AND de cuatro ENTRADAS (E00, E01, E02 y E03). En el inciso b) de la misma figura podemos ver que se obtiene el mismo resultado que la AND de cuatro entradas. En la primera AND del inciso b) se encuentran las dos primeras ENTRADAS (E00 y E01) la cual tiene una salida llamada I00 que es una VARIABLE BOOLEANA INTERMEDIARIA. I00 es intermediaria porque se convierte en una de las ENTRADAS de la segunda AND, siendo E02 y E03 las otras dos ENTRADAS DE ESTA COMPUERTA, esta segunda AND tendrá como resultado la SALIDA S00.

La construcción de estas dos AND's en el PUMA ESCALERA comienza con la arquitectura del primer renglón, que se realizará en base a la primera AND del inciso b) de la figura A.49.

Al dar clic en el botón ENTRADA de la ventana principal, una vez desplegado el cuadro de diálogo se deberá seleccionar la VBE de la PRIMERA ENTRADA. Se elige E00 para esta entrada y se ingresa dando clic con el botón izquierdo del ratón en el renglón 1 (R1) y en la Columna 3 (C3). Nuevamente se deberá dar clic en el botón ENTRADA y seleccionar la SEGUNDA ENTRADA. Se elige E01 para esta entrada y se ingresa dando clic con el botón izquierdo del ratón en el renglón 1 (R1) y en la Columna 6 (C6).

A continuación se deberá dar clic en el botón SALIDA y una vez desplegado el cuadro de diálogo se elige la VBS en este caso será del tipo intermedia I00. Una vez seleccionada la salida, se procede a insertarla dando clic con el botón izquierdo del ratón en el renglón 1 (R1) y en la columna 9 (C9) obligatoriamente.

Después de rellenar los espacios con el botón cable el renglón 1 quedará como lo indica la figura A.49:

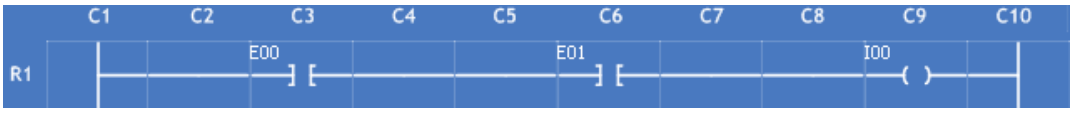


Figura A.49 – Renglón R1, después de la inserción de una compuerta AND con una variable intermediaria I00 fungiendo como salida.

Para realizar la construcción del segundo renglón, que se realizará en base a la segunda AND del inciso b) de la figura A.48.

En primer lugar, se insertará la VARIABLE INTERMEDIARIA. Se deberá dar clic en el botón ENTRADA de la ventana principal, una vez desplegado el cuadro de diálogo Elegir la variable I00 (nota esta variable aparecerá en este listado de variables de entrada tal como se muestra en la figura A.50, puesto que ya ha sido elegida como variable de salida):

Figura A.50 – Ventana para configurar una variable de Entrada Intermediaria

Para ingresar la VBI se deberá dar clic con el botón izquierdo del ratón en el renglón 2 (R2) y en la Columna 3 (C3). Nuevamente dar clic en el botón ENTRADA e ingresar la SEGUNDA ENTRADA que deberá ser E02 para esta entrada e ingresarla en el renglón 2 (R2) y en la Columna 5 (C5). El siguiente paso es ingresar la TERCERA ENTRADA la cual debe ser E03 para esta entrada e ingresarla en el renglón 2 (R2) y en la Columna 7 (C7).

Al dar clic en el botón SALIDA y una vez desplegado el cuadro de diálogo, elegir la salida S00, e ingresar la misma en el renglón 2 (R2) y en la columna 9 (C9) obligatoriamente.

Finalmente habrá que llenar las columnas vacías para obtener un renglón como el mostrado en la figura A.51:

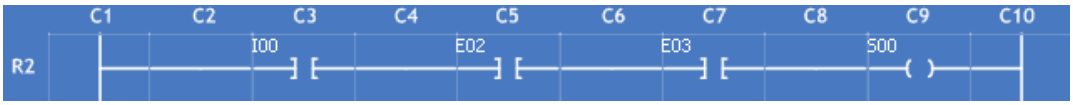


Figura A.51 – Renglón R2, después de la inserción de una compuerta AND con la variable intermediaria I00 como Entrada.

Finalmente la estructura de la compuerta lógica AND de cuatro ENTRADAS representada en dos renglones mediante dos AND's en la ZONA DE TRABAJO queda lista y representada en la figura A.52:

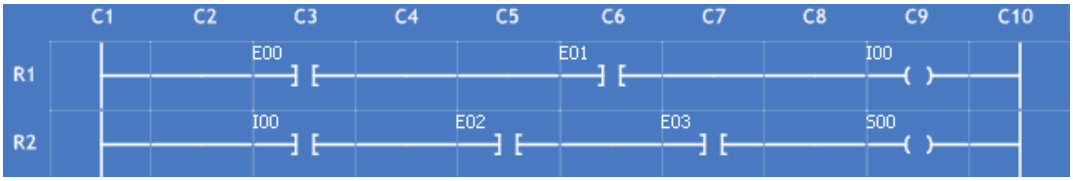


Figura A.52 – Renglones R1 y R2, después de la inserción de una compuerta AND de cuatro entradas.

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

AND2#1 E00,E01,I00,11;
AND3#2 I00,E02,E03,S00,111;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.4 Compuerta Lógica Nand

Una COMPUERTA LÓGICA NAND tiene la siguiente estructura (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer una compuerta lógica NAND):

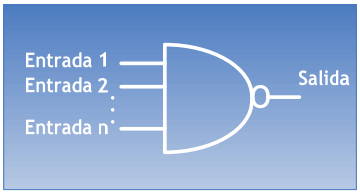


Figura A.52 – Compuerta Lógica NAND.

Su representación en diagrama de escalera esta mostrada en la figura A.53:

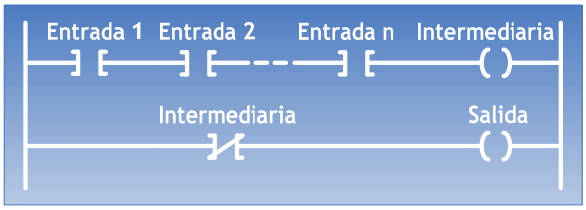


Figura A.53 – Compuerta Lógica NAND en el Diagrama Escalera.

En la figura A.53 se muestra la utilización de una VARIABLE BOOLEANA INTERMEDIARIA en la construcción de la compuerta lógica NAND en el diagrama de escalera. Las "n" ENTRADAS que se tengan formarán una compuerta lógica AND, la cual tendrá como SALIDA una VARIABLE BOOLEANA INTERMEDIARIA. Esta variable intermediaria fungirá una PREINVERSIÓN mediante una ENTRADA NEGADA que es mostrada en una nueva Salida, obteniendo así una compuerta lógica NAND.

Para la creación de la compuerta analógica NAND de dos ENTRADAS de acuerdo con la explicación anterior en el PUMA ESCALERA. Se debe de crear una compuerta lógica AND con Salida Intermedia tal como lo muestra la figura A.54:

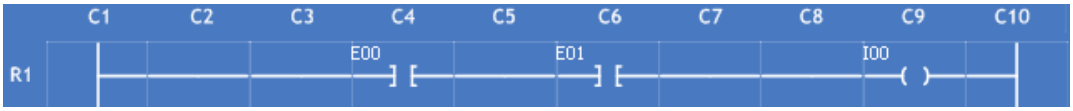


Figura A.54 – Renglón R1, después de la inserción de una compuerta AND de dos Entradas y una Salida Intermediaria.

En el renglón 2 (R2) se hará la preinversión de la VARIABLE BOOLEANA INTERMEDIARIA I00 cuyo resultado será mostrado en una SALIDA que fungirá como la salida de una compuerta lógica NAND.

El segundo renglón se creará a continuación. Comenzamos con la inserción de la VARIABLE INTERMEDIARIA. Al dar clic en el botón NEGADA de la ventana principal, una vez desplegado el cuadro de diálogo se deberá seleccionar la entrada, que en el ejemplo será I00. Se ingresa dando clic con el botón izquierdo del ratón en el renglón 2 (R2) y en la Columna 5 (C5) (figura A.55).

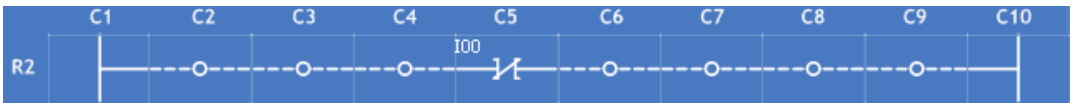


Figura A.55 – Inserción en la Columna C5 del Renglón R2 de una Entrada Negada Intermediaria.

Posteriormente habrá que dar clic en el botón SALIDA y una vez desplegado el cuadro de diálogo seleccionar la salida S00 e ingresarla en el renglón 2 (R2) y en la columna 9 (C9).

Después de haber rellenado los espacios por medio del botón CABLE, el renglón 2 deberá de quedar como lo índica la figura A.56:

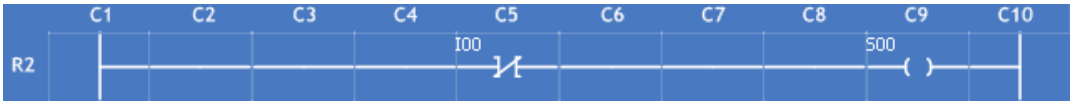


Figura A.56 – Renglón R2, con una Entrada Negada Intermediaria en C5 y una Salida S00. Juntos forma un Seguidor.

Finalmente la estructura de la compuerta lógica NAND de dos ENTRADAS en la ZONA DE TRABAJO deberá verse como se muestra a continuación (figura A.57):

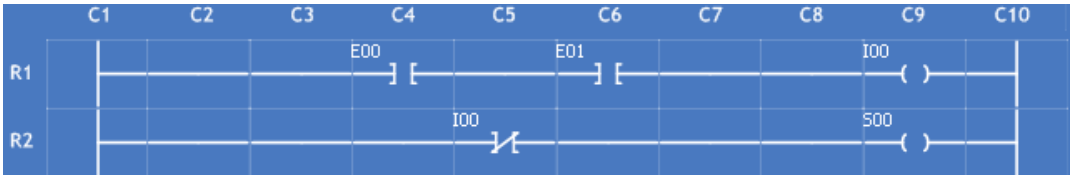


Figura A.57 - Renglones R1 y R2, después de la inserción de una compuerta NAND de dos entradas.

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

AND2#1 E00,E01,I00,11;
NOT#1 I00,S00;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.5 Compuerta Lógica Or

Una COMPUERTA LÓGICA OR tiene la siguiente estructura (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer una compuerta lógica OR):

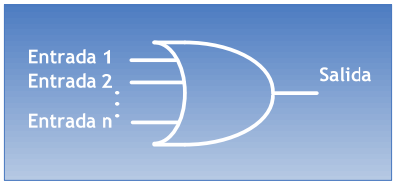


Figura A.58 – Compuerta Lógica OR

Su representación en diagrama de escalera es mostrada por la figura A.59:

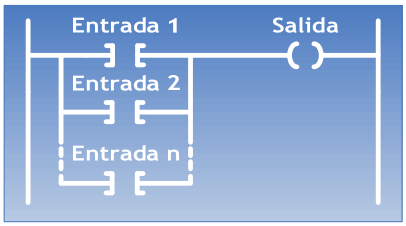


Figura A.58 – Compuerta Lógica OR en el Diagrama de Escalera.

La estructura de una compuerta lógica OR en el diagrama de ESCALERA muestra algunas variaciones con respecto al esquema comúnmente conocido, expuesto en la figura A.58. El orden de las ENTRADAS en la compuerta lógica tiene un sentido vertical mientras que en DIAGRAMA DE ESCALERA tienen un arreglo de unión en los costados, una debajo de la otra. No obstaste mantienen su sentido vertical.

Para la construcción de este tipo de compuertas lógicas se han creado estructuras especiales para facilitar al usuario la inserción de éstas. Es así, que para elaborar una compuerta lógica OR NECESARIAMENTE habrá que hacer uso de los aditamentos específicos llamados CONECTORES.

Se instauraron dos botones para lograr este objetivo, éstos se encuentran en la ventana principal del programa:

Figura A.59 – Botón Conector al Inicio

Figura A.60 – Botón Conector al Final

La figura A.59 muestra el botón “Conectores al Inicio” que se encuentra en la parte superior de la ventana principal. Tiene por finalidad será la introducción de compuertas lógicas OR’s en la columna 3 (C3) de cualquier renglón que se haya seleccionado para insertarlas. Independientemente de la COLUMA del renglón donde se dé el clic izquierdo con el ratón, toda la estructura de la compuerta se construirá forzosamente a partir de la COLUMNA 3, de ahí el nombre del botón, los módulos lógicos OR’s aparecerán al INICIO de los renglones en los que se haya decidido incrustarlas.

La figura A.60 muestra el botón “Conectores al Final” que se encuentra en la parte superior de la ventana principal. Tiene por finalidad ser la introducción de compuertas lógicas OR’s en la columna 7 (C7) de cualquier renglón que se haya seleccionado para insertarlas. Independientemente de la COLUMA del renglón donde se dé el clic izquierdo con el ratón, toda la estructura de la compuerta se construirá forzosamente a partir de la COLUMNA 7, de ahí el nombre del botón, los módulos lógicos OR’s aparecerán al FINAL de los renglones en los que se haya decidido incrustarlas.

La mejor manera de entender estas ideas es mediante un ejemplo. Para construir una compuerta lógica OR con dos entradas. Habrá que dar clic en el botón CONECTORES AL INICIO, el cuadro de diálogo que se mostrará será el siguiente (figura A.61):

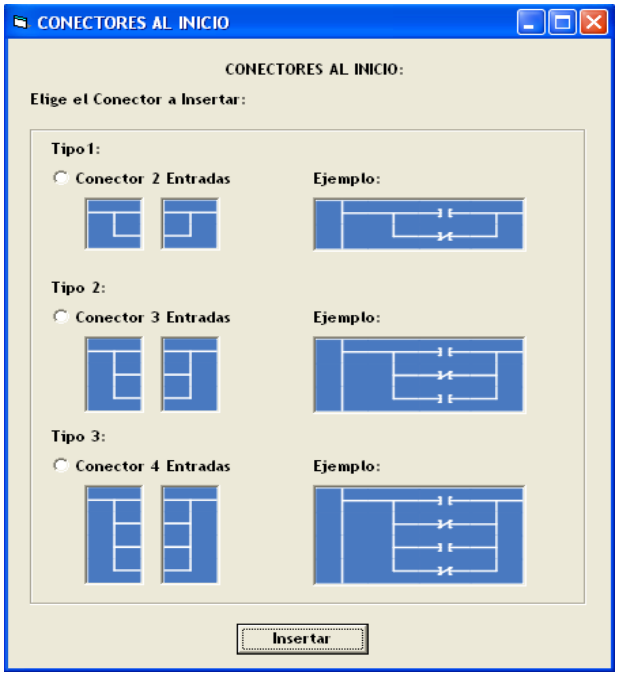


Figura A.61 – Ventana para Configurar “Conectores al Inicio”

En la figura A.61 se observa el cuadro de diálogo “CONECTORES AL INICIO”, en la parte superior de éste se encuentra la leyenda “Elige el Conector a Insertar:”, esto indica que forzosamente tenemos que elegir una de las tres opciones mostradas en la ventana. En el TIPO 1 se localiza el Conector de dos entradas, el cual tiene por finalidad introducir una estructura para construir una compuerta lógica OR de dos entradas como lo indica su ejemplo. En el TIPO 2 se localiza el Conector de tres entradas, cuyo propósito es introducir una estructura para construir una compuerta lógica OR de tres entradas como lo indica su ejemplo. En el TIPO 3 se localiza el Conector de cuatro entradas, el cual tiene por finalidad introducir una estructura para construir una compuerta lógica OR de cuatro entradas como lo indica su ejemplo.

Para este ejemplo se utilizará la opción “Tipo 1” con un clic izquierdo del ratón para insertar un Conector de dos entradas. Enseguida se deberá dar clic en el botón “Insertar” de la misma ventana.

NOTA: El botón cerrar se encuentra deshabilitado, la ventana no se podrá cerrar por este medio. Por tal motivo es imprescindible estar totalmente seguro de que se quiere insertar CONECTORES AL INICIO.

A continuación en la ZONA DE TRABAJO incrustar un CONECTOR AL INICIO, seleccionando el renglón 2 (R2) para este fin. Dar clic con el botón izquierdo del ratón en cualquier columna del renglón 2; aparecerá en la zona de trabajo la siguiente estructura (figura A.62):

Figura A.62 – Estructura insertada en los Renglones R2 y R3 por la ventana de configuración "Conectores al inicio".

Enseguida aparecerá la siguiente ventana de diálogo:

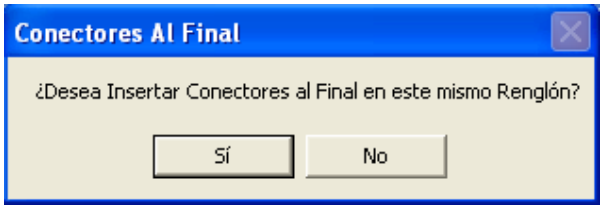


Figura A.63 – Ventana de Confirmación para insertar la estructura "Conectores al Inicio"

El cuadro de diálogo mostrado en la figura A.63 tiene por objetivo introducir más compuertas lógicas OR's inmediatamente después de la que estamos configurando. Más adelante se verá con detalle esta nueva herramienta, En este momento se elegirá NO. Una vez realizado lo anterior se deberá desplegar lo siguiente:

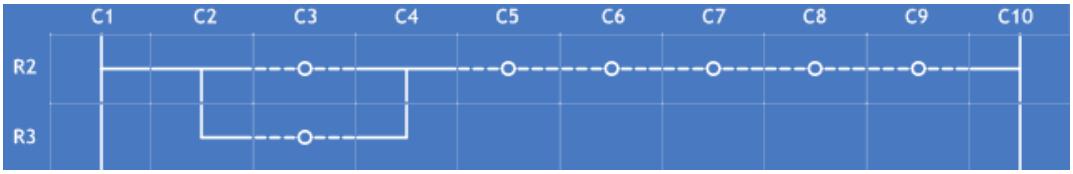


Figura A.64 – Estructura insertada en los Renglones R2 y R3 por la ventana de configuración "Conectores al inicio", después de una validación negativa en la ventana de Confirmación.

La figura A.64 muestra la estructura que ayudará a construir una compuerta lógica OR de DOS entradas, es importante destacar que aunque se dio clic en el renglón 2 para insertar esta estructura, ésta ocupó también el renglón 3, el cual quedó COMPLETAMENTE DESHABILITADO, excepto la columna 3 que es el lugar donde aparecerá una de las entradas de la compuerta, por lo que ya no se podrá utilizar para la incrustación de otros módulos lógicos.

Es relevante también indicar que la estructura en sí deshabilita todos los lugares en donde se incrustó. Así, sólo quedarán disponibles las Columnas C3, C5, C6, C7, C8 y C9 del renglón 2 y la Columna C3 del renglón 3 para la inserción de módulos lógicos.

Queda ahora seguir con la construcción de la compuerta lógica. Dar clic en el botón ENTRADA de la ventana principal, una vez desplegado el cuadro de diálogo Elegir el nombre de la PRIMERA ENTRADA. Elegir E00 para esta entrada e ingresarla en el renglón 2 (R2) y en la Columna 3 (C3), como lo muestra la figura A.65:

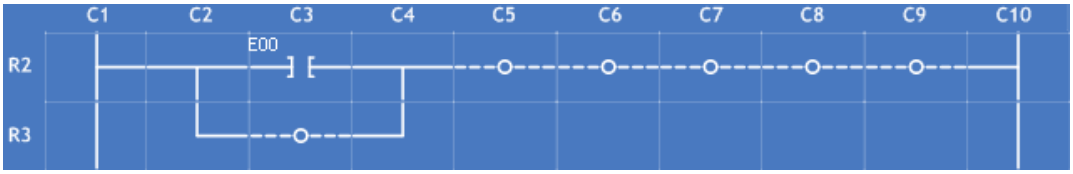


Figura A.65 – Inserción de la Entrada E00 en la Columna C3 del Renglón R2 en la estructura “Conectores el Inicio”.

Ahora ingresar la segunda ENTRADA eligiendo E01,en el renglón 3 (R1) y en la Columna 3 (C3):

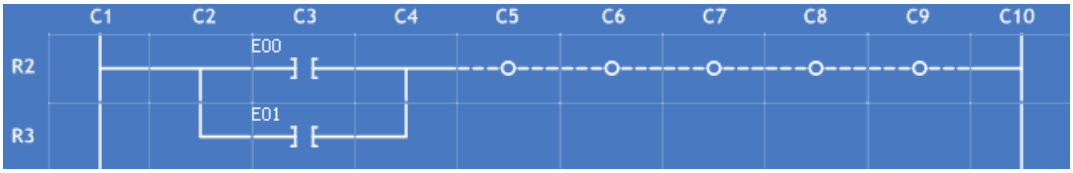


Figura A.66 – Inserción de la Entrada E01 en la Columna C3 del Renglón R3 en la estructura “Conectores el Inicio”.

Ingresar la SALIDA eligiendo, por ejemplo S00 e ingresarla en el renglón 2 (R2) y en la columna 9 (C9) obligatoriamente.

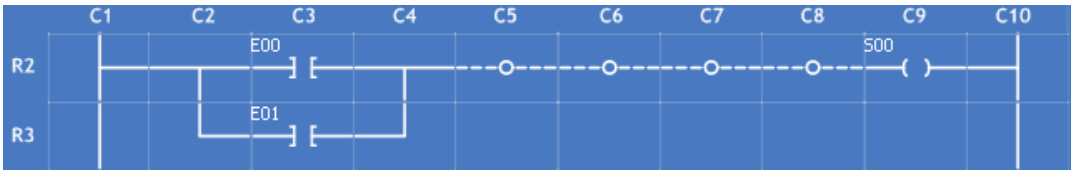


Figura A.67 – Inserción de la Salida S00 en la Columna C9 del Renglón R2 en la estructura “Conectores el Inicio”.

Realizar el proceso de UNIÓN entre las ENTRADAS y la SALIDA. Recordando que este proceso se lleva a cabo con el botón CABLE de la ventana principal, por lo tanto, al dar clic en él se deberá dar clic con el botón izquierdo del ratón en las columnas C5, C6, C7 y C8. El resultado final será una COMPUERTA LÓGICA OR de dos entradas tal y como se muestra en la figura A.68:

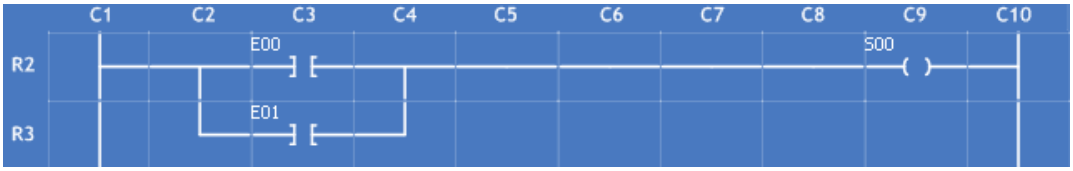


Figura A.68 – Compuerta Lógica OR creada mediante el uso de la herramienta “Conectores al Inicio”.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA 1: E00
ENTRADA 2: E01
SALIDA: S00

¿PREINVERSIÓN de la entrada 2?: NO, que esta representado a través del primer dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
¿PREINVERSIÓN de la entrada 1?: NO, que esta representado a través del segundo dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

OR2#1 E00,E01,S00,11;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

Un ejemplo más completo y didáctico es la construcción de una compuerta lógica OR con 3 entradas, siendo una de ellas preinvertida. Comenzar dando un clic en el botón CONECTORES AL FINAL de la ventana principal.

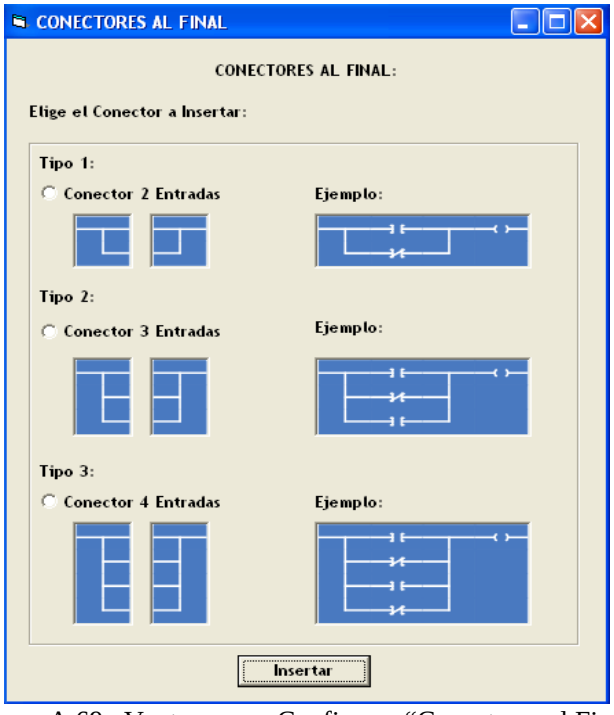


Figura A.69– Ventana para Configurar “Conectores al Final”

En la figura A.69 se observa el cuadro de diálogo “CONECTORES AL FINAL”. Al igual que en los CONECTORES AL INICIO en la parte superior de éste se encuentra la leyenda “Elige el Conector a Insertar:”, esto indica que forzosamente tenemos que elegir una de las tres opciones mostradas en la ventana.

En el TIPO 1 se localiza el Conector de dos entradas, el cual tiene por finalidad introducir una estructura para construir una compuerta lógica OR de dos entradas como lo indica su ejemplo. En el TIPO 2 se localiza el Conector de tres entradas, cuyo propósito es introducir una estructura para construir una compuerta lógica OR de tres

entradas como lo indica su ejemplo. En el TIPO 3 se localiza el Conector de cuatro entradas, el cual tiene por finalidad introducir una estructura para construir una compuerta lógica OR de cuatro entradas como lo indica su ejemplo.

El propósito de utilizar CONECTORES AL FINAL en este ejemplo es mostrarle al lector que se pueden crear las mismas compuertas lógicas OR ya sea utilizando los "Conectores al Inicio" o como en este caso implementando los "Conectores al final". Y obviamente viene al caso la pregunta: ¿Para qué crear dos estructuras para la misma tarea? La respuesta es simple: Se utilizarán para la creación de arquitecturas híbridas, es decir, COMPUERTAS AND seguidas de COMPUERTAS OR o viceversa. Estas arquitecturas se verán más adelante.

Dar clic en la opción "Tipo 2" con un clic izquierdo del ratón para insertar un Conector de tres entradas. Repetir esta acción en el botón "Insertar" de la misma ventana.

NOTA: El botón cerrar se encuentra deshabilitado, la ventana no se podrá cerrar por este medio. Por tal motivo es imprescindible estar totalmente seguro de que se quiere insertar CONECTORES AL FINAL.

Para incrustar el CONECTOR AL FINAL seleccionar el renglón 4 (R4). Aparecerá en la zona de trabajo la siguiente estructura (Figura A.70):

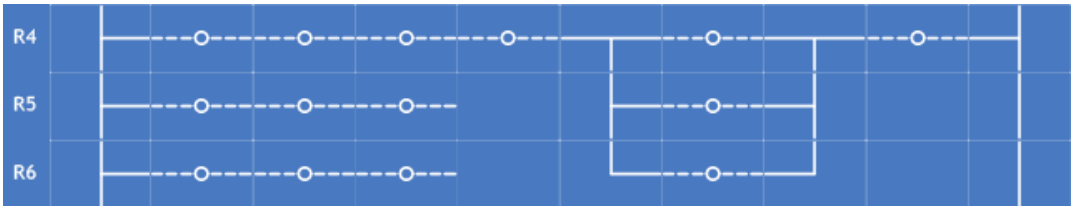


Figura A.70 – Estructura insertada en los Renglones R4, R5 y R6 por la ventana de configuración "Conectores al Final".

Enseguida aparecerá la siguiente ventana de diálogo:

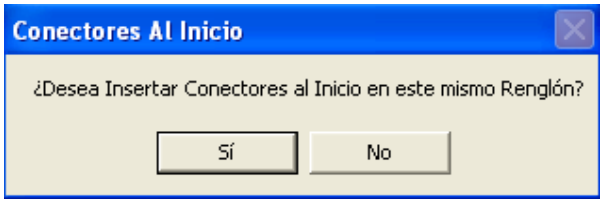


Figura A.71 – Ventana de Confirmación para insertar la estructura "Conectores al Final"

El cuadro de diálogo mostrado en la figura A.71 tiene por objetivo introducir más compuertas lógicas OR's del lado IZQUIERDO de la que estamos configurando. Más adelante se verá con detalle esta nueva herramienta, por lo pronto hacer clic en NO. Una vez realizado lo anterior la estructura será :

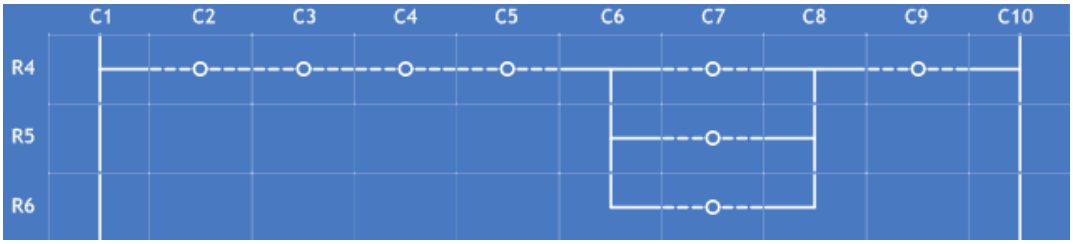


Figura A.72 – Estructura insertada en los Renglones R4, R5 y R6 por la ventana de configuración “Conectores al Final”, después de una validación negativa en la ventana de Confirmación.

La figura A.72 muestra la estructura que nos ayudará a construir una compuerta lógica OR de TRES entradas, es importante destacar que aunque se dio clic en el renglón 4 para insertar esta estructura, ésta ocupó también el renglón 5 y el renglón 6, los cuales quedaron COMPLETAMENTE DESHABILITADOS, excepto la columna 7 en cada renglón que son los lugares donde aparecerán las entradas de la compuerta, por lo que ya no se podrá utilizar para la incrustación de otros módulos lógicos.

Es relevante también indicar que la estructura en sí deshabilita todos los lugares en donde se incrustó. Así, sólo quedarán disponibles las Columnas C2, C3, C4, C5, C7 y C9 del renglón 4, la Columna C7 del renglón 5 y la Columna C7 del renglón 6 para la inserción de módulos lógicos.

Queda ahora seguir con la construcción de la compuerta lógica. Insertar la ENTRADA E02 en el renglón 4 (R4) y en la Columna 7 (C7), como lo muestra la figura A.73

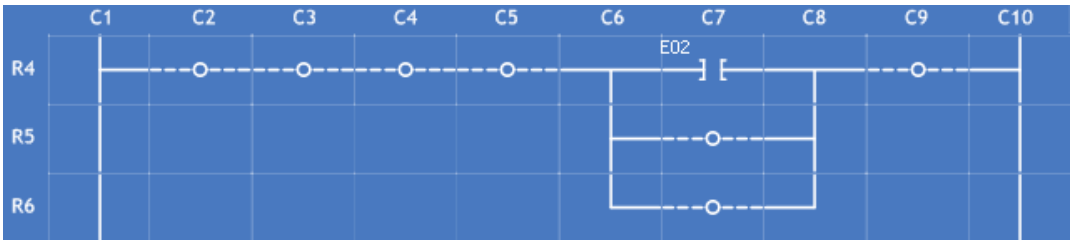


Figura A.73 – Inserción de la Entrada E02 en la Columna C7 del Renglón R4 en la estructura “Conectores el Final”.

La segunda ENTRADA será la que tenga la PREINVERSIÓN. En consecuencia hacer clic en el botón NEGADA de la ventana principal, una vez desplegado el cuadro de diálogo Elegir E03 para esta entrada y la ingresamos en el renglón 5 (R5) y en la Columna 7 (C7) (figura A.74):

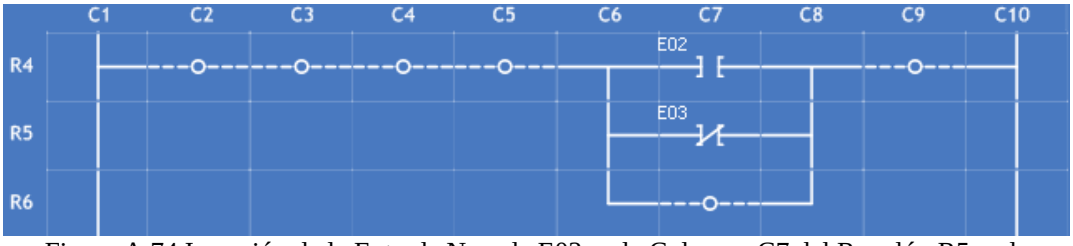


Figura A.74 Inserción de la Entrada Negada E03 en la Columna C7 del Renglón R5 en la estructura “Conectores el Final”.

Para la última ENTRADA Elegir E04 la ingresamos en el renglón 6 (R6) y en la Columna 7 (C7), como lo muestra la figura A.75

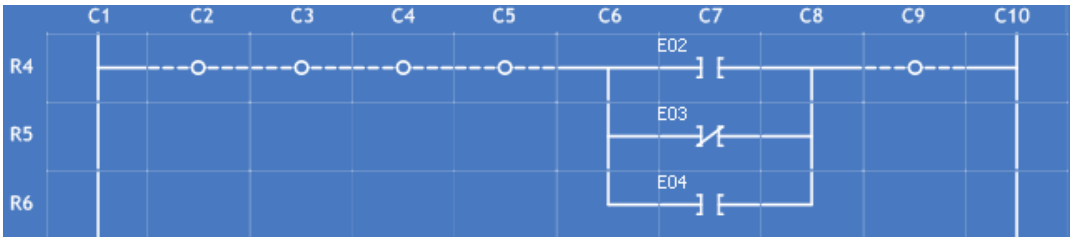


Figura A.75 – Inserción de la Entrada E04 en la Columna C7 del Renglón R6 en la estructura "Conectores el Final".

A continuación Insertar la salida S01 e ingresarla en el renglón 4 (R4) y en la columna 9 (C9).

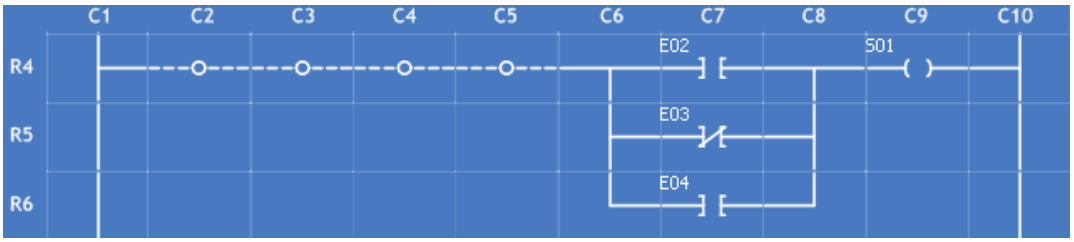


Figura A.76 – Inserción de la Salida S01 en la Columna C9 del Renglón R4 en la estructura "Conectores el Final".

Finalmente habrá que realizar el proceso de UNIÓN en las COLUMNAS VACÍAS. Recordando que este proceso se lleva a cabo con el botón CABLE de la ventana principal. El resultado final será una COMPUERTA LÓGICA OR de tres entradas, como la que se muestra en la figura A.77:

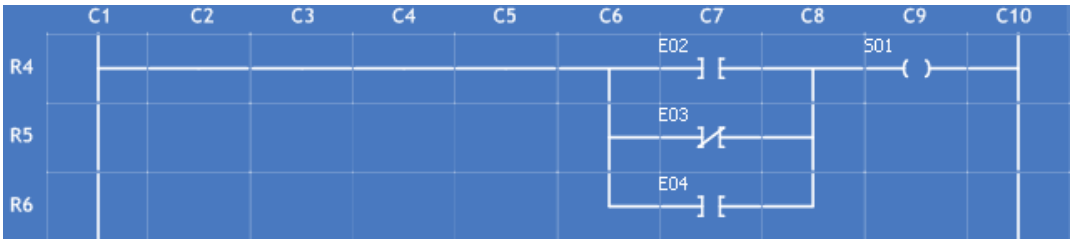


Figura A.77 – Compuerta Lógica OR de Tres Entradas creada mediante el uso de la herramienta "Conectores al Inicio".

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA 1: E02
ENTRADA 2: E03
ENTRADA 2: E04
SALIDA: S01
¿PREINVERSIÓN de la entrada 3?: NO, que esta representado a través del primer dígito (izquierda a derecha) de los 3 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1

¿PREINVERSIÓN de la entrada 2?: SI, que esta representado a través del segundo dígito (izquierda a derecha) de los 3 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
¿PREINVERSIÓN de la entrada 1?: NO, que esta representado a través del tercer dígito (izquierda a derecha) de los 3 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

OR3#1 E02,E03,E04,S01,101;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

Por lo explicado hasta el momento sería trivial realizar el ejemplo de una compuerta lógica OR de cuatro entradas (máximo número de entradas posible). La única diferencia cuya relevancia sería notable se presentaría a la hora de escoger el tipo de conector en las ventanas de configuración de "CONECTORES AL INICIO" y "CONECTORES AL FINAL" respectivamente, en cuyo caso sería 3. Los pasos subsecuentes serían superficiales dado que ya se conoce el proceso.

A.3.6 Compuerta Lógica Nor

Una COMPUERTA LÓGICA NOR se representa con el símbolo mostrado en la figura A.78 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer una compuerta lógica NOR):

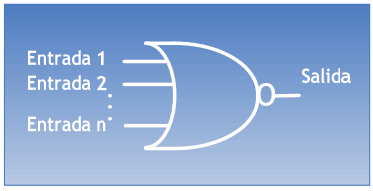


Figura A.78 – Compuerta Lógica NOR

La Figura A.79 muestra la representación en diagrama de escalera:

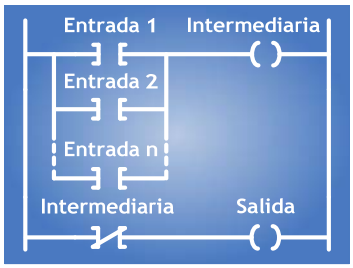


Figura A.79 – Compuerta Lógica NOR en el Diagrama de Escalera.

En la figura A.79 se muestra la utilización de una VARIABLE BOOLEANA INTERMEDIARIA en la construcción de la compuerta lógica NOR en el diagrama de escalera. Las “n” ENTRADAS que se tengan formarán una compuerta lógica OR, la cual tendrá como SALIDA una VARIABLE BOOLEANA INTERMEDIARIA. Esta variable intermediaria fungirá una PREINVERSIÓN mediante una ENTRADA NEGADA que es mostrada en una nueva Salida, obteniendo así una compuerta lógica NOR.

Dado que ya se ha explicado la creación de la compuerta lógica OR y el Inversor, Se propone como ejercicio para el lector la creación de la compuerta lógica NOR.

La estructura de la compuerta lógica NOR deberá tener una forma muy parecida a la de la figura A.80 (variará de acuerdo al número de ENTRADAS):

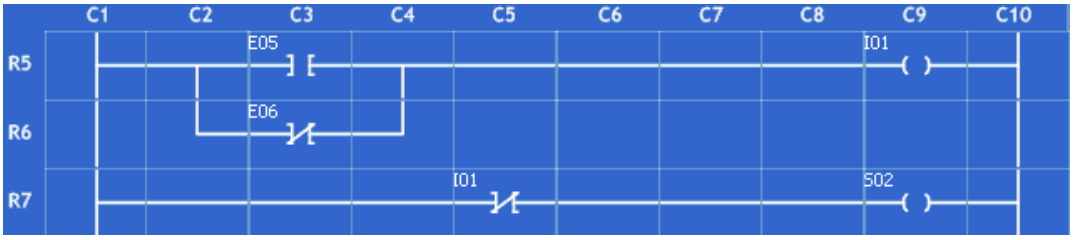


Figura A.80 – Compuerta lógica NOR creada mediante el uso de la herramienta “Conectores al Inicio” en los renglones R5 y R6. Y la preinversión en el Renglón R7 para obtener la Salida en la Columna C9.

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

OR2#1 E05,E06,I01,01;
NOT#1 I01,S02;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.7 Compuerta Lógica Xor De Dos Entradas

Una COMPUERTA LÓGICA XOR de DOS ENTRADAS esta representada con el símbolo de la figura A.81(se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer una compuerta lógica XOR de DOS ENTRADAS):

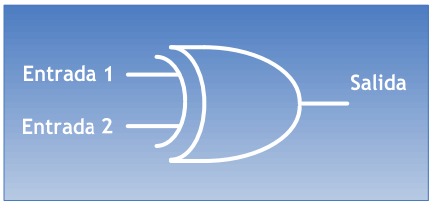


Figura A.81 – Compuerta Lógica XOR de dos Entradas.

El Circuito XOR equivalente es:

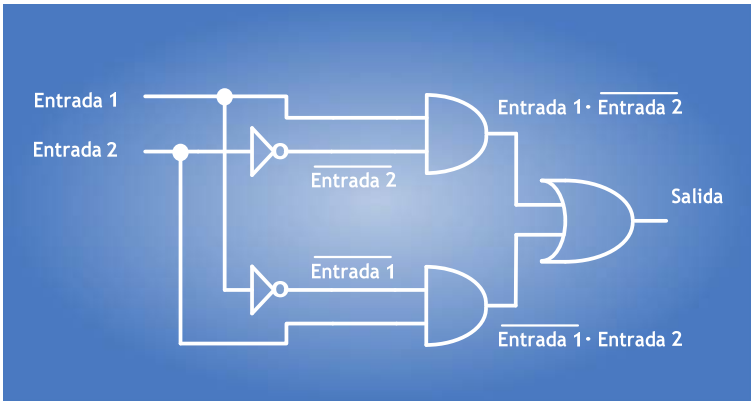


Figura A.82 – Combinación de varias compuertas lógicas para construir una Compuerta Lógica XOR de dos Entradas.

Cuya ecuación es la siguiente:

$$Salida = Entrada\,1 \cdot \overline{Entrada\,2} \quad + \quad \overline{Entrada\,1} \cdot Entrada\,2$$

La figura A.83 muestra la representación XOR mediante un diagrama de escalera:

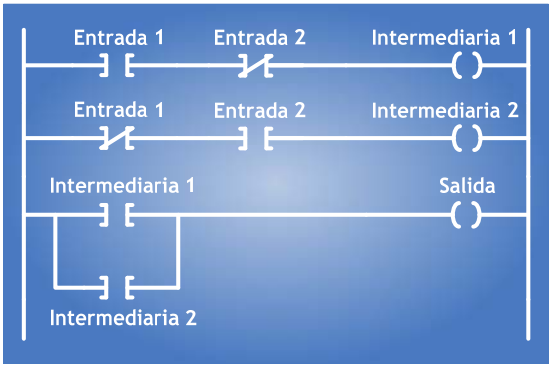


Figura A.83 – Compuerta Lógica XOR de dos Entradas en el Diagrama de Escalera.

La figura A.83 muestra la estructura de una compuerta lógica XOR de DOS entradas, la cual está constituida por dos compuertas lógicas AND y una compuerta lógica OR.

La primera compuerta AND tiene 2 entradas, la segunda de ellas tiene PREINVERSIÓN (Entrada 2) y tiene como salida una variable booleana intermediaria llamada: Intermediaria 1.

$$Intermediaria\,1 = Entrada\,1 \cdot \overline{Entrada\,2}$$

La segunda compuerta AND tiene las mismas 2 entradas, la primera de ellas tiene PREINVERSIÓN (Entrada 1) y tiene como salida una variable booleana intermediaria llamada: Intermediaria 2.

$$Intermediaria\,2 = \overline{Entrada\,1} \cdot Entrada\,2$$

Asimismo la compuerta OR tiene como entradas las dos variables intermediarias que fungieron como salidas de las compuertas AND, a su vez su salida será representada a través de una variable booleana de salida.

$$Salida = Intermediaria\ 1 \quad + \quad Intermediaria\ 2$$

Debido a que se ha visto ya la construcción de compuertas lógicas AND y OR, con entradas preinvertidas, se deja a consideración del lector el desarrollo de este ejemplo. El Lector deberá de llegar a una estructura como la mostrada en la figura A.84, la cual es una compuerta lógica XOR de DOS ENTRADAS:

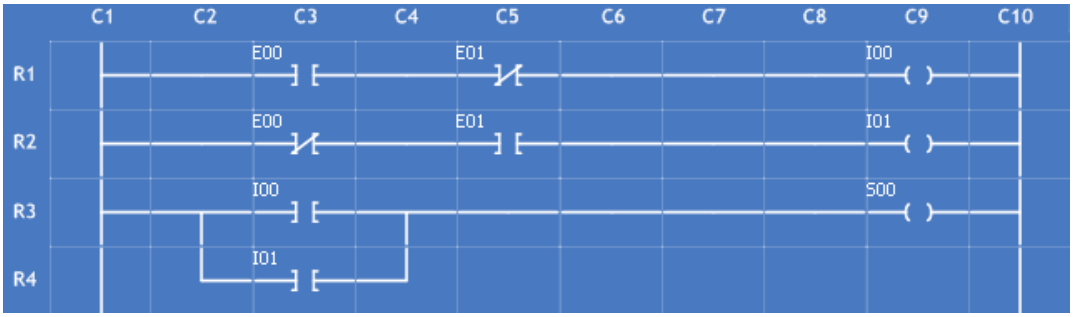


Figura A.84 – Construcción de una Compuerta Lógica XOR de dos Entradas en el PUMA ESCALERA.

En la Figura A.84 se observa la construcción de ina compuerta lógica está basada en dos compuertas Lógicas AND en los Renglones R1 y R2 respectivamente, y la inserción de una compuerta Lógica OR en los renglones R3 y R4. Estas últimas a través de la herramienta “Conectores al Inicio”.

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

AND2#1 E00,E01,I00,01;
AND2#2 E00,E01,I01,10;
OR2#1 I00,I01,S00,11;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.8 Compuerta Lógica Xor De Tres Entradas

Una COMPUERTA LÓGICA XOR de TRES ENTRADAS esta representada con el símbolo de la figura A.85((se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer una compuerta lógica XOR de TRES ENTRADAS):

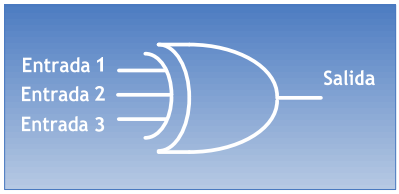


Figura A.85 – Compuerta Lógica XOR de tres Entradas.

El Circuito XOR equivalente es:

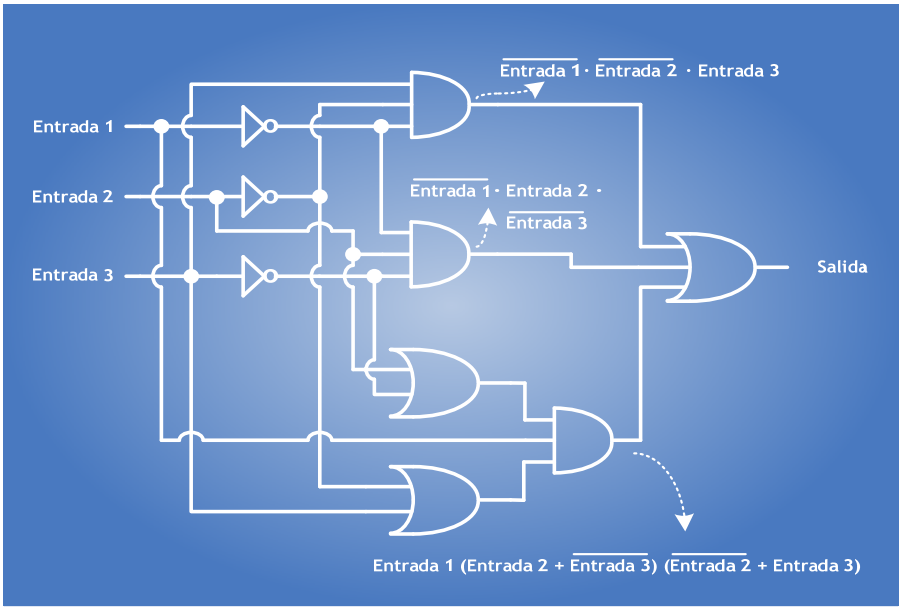


Figura A.86 – Combinación de varias compuertas lógicas para constituir una Compuerta Lógica XOR de tres Entradas.

Cuya ecuación es la siguiente:

$$Salida = \overline{Entrada\,1}\cdot\overline{Entrada\,2}\cdot Entrada\,3 \;+\; \overline{Entrada\,1}\cdot Entrada\,2\cdot\overline{Entrada\,3} \;+\; Entrada\,1\,(Entrada\,2+\overline{Entrada\,3})(\overline{Entrada\,2}+Entrada\,3) \qquad \ldots\ldots\ldots\ldots\ldots\ldots(1)$$

La figura A.87 muestra la representación XOR mediante un diagrama de escalera:

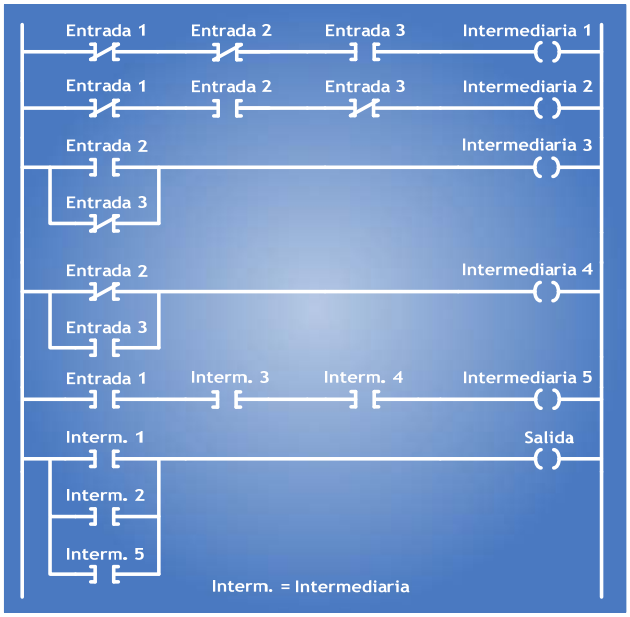


Figura A.87 – Compuerta Lógica XOR de tres Entradas en el Diagrama Escalera.

La figura A.8.3 muestra la estructura de una compuerta lógica XOR de TRES entradas, la cual está constituida por dos compuertas lógicas AND, seguidas de dos compuertas lógicas OR, a continuación nuevamente una compuerta lógica AND y finalmente otra compuerta lógica OR.

La primera compuerta AND tiene 3 entradas, teniendo PREINVERSIÓN la primera (Entrada 1) y la segunda (Entrada 2). Tiene como salida una variable booleana intermediaria llamada: Intermediaria 1. Esta compuerta es el primer sumando de la ecuación (1):

$$Intermediaria\ 1 = \overline{Entrada\ 1} \cdot \overline{Entrada\ 2} \cdot Entrada\ 3$$

La segunda compuerta AND tiene las mismas 3 entradas, teniendo PREINVERSIÓN la primera (Entrada 1) y la tercera (Entrada 3). Tiene como salida una variable booleana intermediaria llamada: Intermediaria 2. Esta compuerta es el segundo sumando de la ecuación (1):

$$Intermediaria\ 2 = \overline{Entrada\ 1} \cdot Entrada\ 2 \cdot \overline{Entrada\ 3}$$

Para formar el tercer sumando de la ecuación (1) necesitamos el uso de tres compuertas lógicas: 2 compuertas OR y una compuerta AND. La primera compuerta OR tiene 2 entradas (Entrada 2 y Entrada 3), teniendo PREINVERSIÓN la Entrada 3. Su salida será una variable booleana intermediaria llamada: Intermediaria 3. Esta compuerta es el segundo multiplicador del tercer sumando de la ecuación (1):

$$Intermediaria\ 3 = Entrada\ 2 \quad + \quad \overline{Entrada\ 3}$$

La segunda compuerta OR tiene las mismas 2 entradas (Entrada 2 y Entrada 3), teniendo PREINVERSIÓN la Entrada 2. Su salida será una variable booleana intermediaria llamada: Intermediaria 4. Esta compuerta es el tercer multiplicador del tercer sumando de la ecuación (1):

$$Intermediaria\ 4 = \overline{Entrada\ 2} \quad + \quad Entrada\ 3$$

El primer multiplicador del tercer sumando es la Entrada 1, por lo tanto, el tercer sumando será una compuerta AND cuyas entradas serán las dos variables intermediarias que fungieron como salidas de las compuertas OR (Intermediaria 3 e Intermediaria 4) y la Entrada 1, para su salida Se utilizará otra variable booleana intermediaria llamada: Intermediaria 5.

$$Intermediaria\ 5 = Entrada\ 1 \cdot Intermediaria\ 3 \cdot Intermediaria\ 4$$

Finalmente la Salida de la compuerta XOR de TRES entradas será representada con una compuerta lógica OR cuyas entradas serán la variable Intermediaria 1 (1er. Sumando de la ecuación 1), la variable Intermediaria 2 (2do. Sumando de la ecuación 1) y la variable Intermediaria 5 (3er. Sumando de la ecuación 1). Obviamente la salida será una variable booleana de salida.

Salida = *Intermediaria* 1 + *Intermediaria* 2 + *Intermediaria* 5

Debido a que se ha visto ya la construcción de compuertas lógicas AND y OR, con entradas preinvertidas, se deja a consideración del lector el desarrollo de este ejemplo. El Lector deberá de llegar a una estructura como la mostrada en la figura A.88, la cual es una compuerta lógica XOR de TRES ENTRADAS:

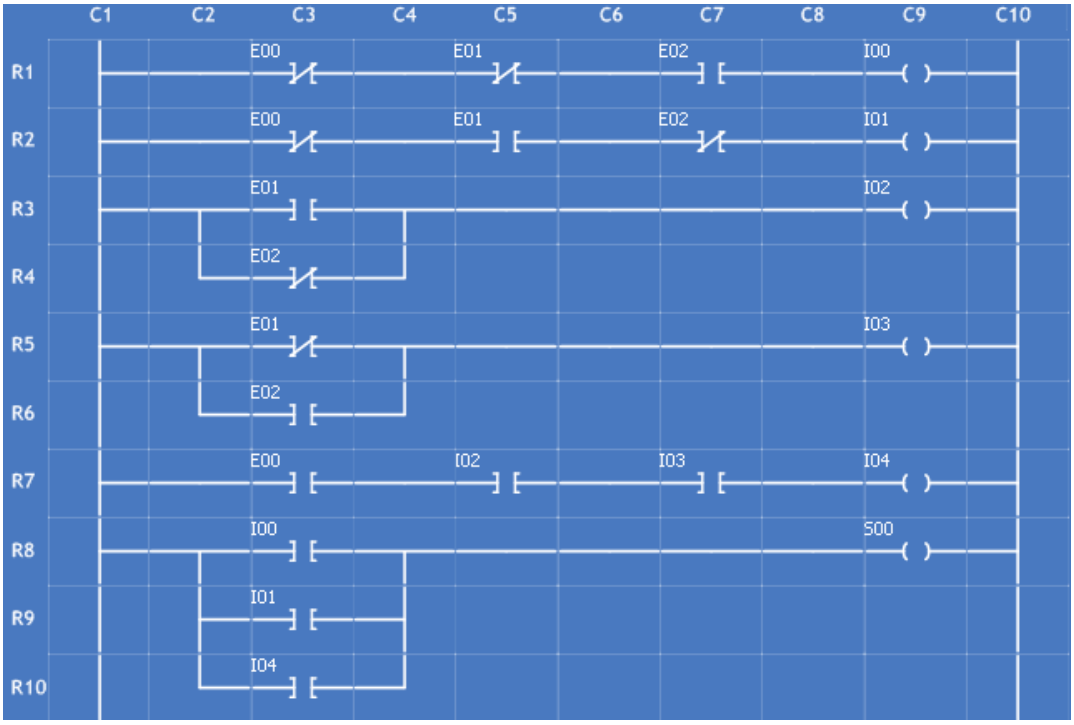


Figura A.88 – Construcción de una Compuerta Lógica XOR de tres Entradas en el PUMA ESCALERA.

En la figura A.88 se observa la construcción de una compuerta lógica AND en los Renglones R1, R2 y R7 respectivamente, y la inserción de tres compuertas Lógicas OR en los renglones R3, R5 y R8. Estas últimas a través de la herramienta “Conectores al Inicio”.

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

AND3#1 E00,E01,E02,I00,100;
AND3#2 E00,E01,E02,I01,010;
OR2#1 E01,E02,I02,01;
OR2#2 E01,E02,I03,10;
AND3#3 E00,I02,I03,I04,111;
OR3#3 I00,I01,I04,S00,111;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.9 Compuerta Lógica Xor De Cuatro Entradas

Una COMPUERTA LÓGICA XOR de CUATRO ENTRADAS esta representada con el símbolo de la figura A.89 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer una compuerta lógica XOR de CUATRO ENTRADAS):

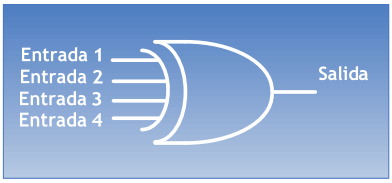


Figura A.89 – Compuerta Lógica XOR de Cuatro Entradas

El Circuito XOR equivalente es:

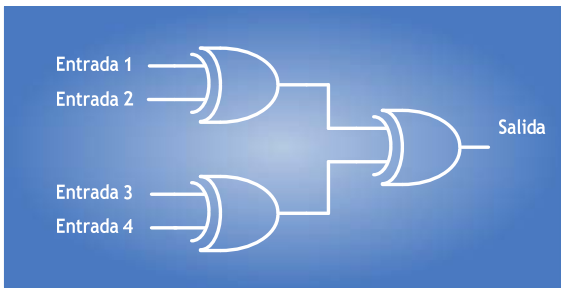


Figura A.90 – Compuerta Lógica XOR de Cuatro Entradas en el Diagrama Escalera.

Como se puede ver en la figura A.90 una compuerta XOR de CUATRO entradas es una composición de tres compuertas XOR de dos entradas. La compuerta lógica XOR de dos entradas ya ha sido examinada a detalle unas cuantas páginas atrás, por este motivo queda como ejercicio para el usuario realizar todo el proceso de dibujado del diagrama de ESCALERA de esta compuerta.

A.3.10 Compuerta Lógica Xor Negada

Una COMPUERTA LÓGICA XOR NEGADA esta representada con el símbolo de la figura A.91(se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer una compuerta lógica XOR NEGADA):

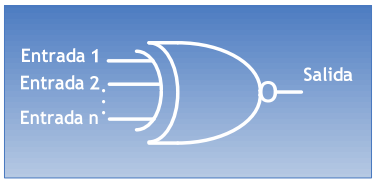


Figura A.91 – Compuerta Lógica XOR Negada

Esta compuerta puede tener varios circuitos equivalentes dependiendo del número de entradas de esta, como lo vimos anteriormente con una compuerta lógica XOR.

El raciocinio nos permite percibir claramente que la única diferencia significativa entre una compuerta XOR y una compuerta XOR NEGADA es que la salida tiene PREINVERSIÓN. Por lo tanto, todos los ejercicios realizados para una compuerta lógica XOR aplican de la misma manera a una compuerta XOR NEGADA, teniendo en cuenta que en todos los casos que habrá de utilizarse otro renglón de nuestro programa para realizar la correspondiente PREINVERSIÓN de la SALIDA.

Para hacer énfasis en esta idea se muestra la figura A.92 de una compuerta lógica XOR NEGADA de DOS entradas realizada en el diagrama de ESCALERA:

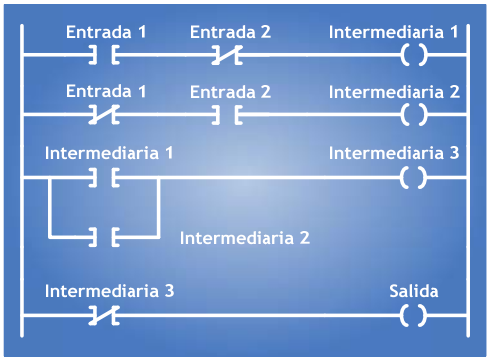


Figura A.92 – Compuerta Lógica XOR de dos Entradas en el Diagrama de Escalera.

El diagrama mostrado en la figura A.92 se puede también ejemplificar mediante las siguientes ecuaciones:

$$Salida = \overline{(Entrada\,1 \cdot \overline{Entrada\,2} \quad + \quad \overline{Entrada\,1} \cdot Entrada\,2)} \quad \ldots\ldots(1)$$

Donde:

$$Intermediaria\,1 = Entrada\,1 \cdot \overline{Entrada\,2}$$

$$Intermediaria\,2 = \overline{Entrada\,1} \cdot Entrada\,2$$

$$Intermediaria\,3 = Intermediaria\,1 \quad + \quad Intremediaria\,2$$

$$Salida = \overline{Intermediaria\,3}$$

Esta última ecuación muestra la PREINVERSIÓN de la SALIDA, que fue representada a través de la NEGACIÓN de la variable booleana intermediaria: Intermediaria 3.

A.3.11 Arquitectura De Compuertas Lógicas

En este apartado se describe el uso y ventajas de las herramientas “CONECTORES AL INICIO” y “CONECTORES AL FINAL”. Hasta ahora estos dos objetos se han limitado a construir estructuras simples de compuertas lógicas OR y XOR, pero su verdadero potencial radica en su uso conjunto.

Por ejemplo, ¿qué sucede si se quiere INCRUSTAR una estructura como la mostrada en la figura A.93?

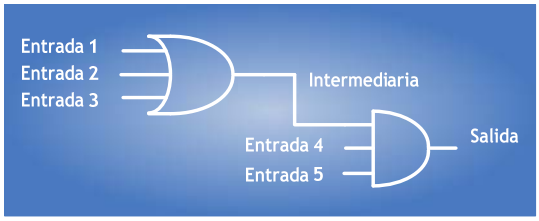


Figura A.93 – Compuerta Lógica OR de tres entradas cuya Salida es la Entrada de una Compuerta Lógica AND.

La Figura A.93 muestra una compuerta lógica OR de TRES entradas cuya salida es una de las TRES entradas de una compuerta lógica AND. Por lo realizado hasta el momento el construir esta arquitectura conduciría a la utilización de varios renglones en la ZONA DE TRABAJO.

Para esta compuerta en específico se utilizarían CUATRO renglones, TRES para la construcción de la compuerta lógica OR y UNO más para la compuerta lógica AND. Normalmente lo más adecuado sería dibujar la compuerta OR e inmediatamente después la compuerta AND, sin utilizar el último renglón ya mencionado. La figura A.94 muestra esta idea:

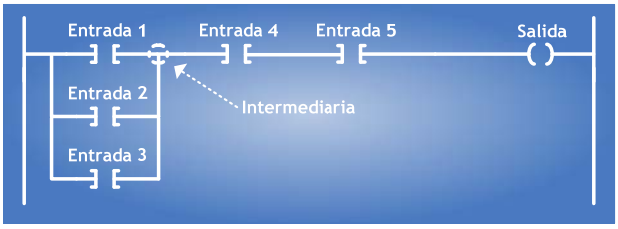


Figura A.94 – Compuerta Lógica OR de tres entradas cuya Salida es la Entrada de una Compuerta Lógica AND,

En la figura A.94 se puede observar que la SALIDA de la compuerta lógica OR cuyas entradas son Entrada 1, Entrada 2 y Entrada 3 es una variable INTERMEDIARIA. En la figura A.94 se puede apreciar que la SALIDA de la compuerta OR es el punto de convergencia entre la Entrada 1, la Entrada 2, y la Entrada 3. Este punto de convergencia será tomado por el programa como la SALIDA de esta compuerta y le asignará una variable INTERMEDIARIA con un NOMBRE Ixxx que **NO** haya sido utilizado por el usuario. Esta variable INTERMEDIARIA no será mostrada en la ZONA DE TRABAJO por lo que el usuario sólo la “verá” cuando se ha realizado la correspondiente traducción del proyecto a LENGUAJE SIIL1.

Esta variable INTERMEDIARIA generada por el programa fungirá ahora como UNA de las TRES entradas de la compuerta lógica AND.

Para aclarar lo siguiente habrá que realizar el siguiente ejemplo. Hacer clic en el botón CONECTORES AL INICIO, en el cuadro de diálogo seleccionar la opción “Tipo 2” con un clic izquierdo del ratón para insertar un Conector de tres entradas. Enseguida se deberá dar clic en el botón “Insertar” de la misma ventana.

Seleccionar el renglón 1 (R1) para este fin. Enseguida aparecerá la siguiente ventana de diálogo (figura A.95), elegimos “NO”:

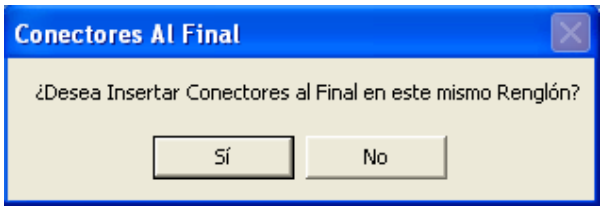


Figura A.95 - Ventana de Confirmación para insertar la estructura “Conectores al Final”

Una vez insertado el CONECTOR AL INICIO dar clic en el botón ENTRADA, seleccionar como VBE las entradas E00 en el renglón 1 (R1) y en la Columna 3 (C3), E01 en el renglón 2 (R2) y en la Columna 3 (C3) y E02 en el renglón 3 (R3) y en la Columna 3 (C3).

Una vez completada la estructura de la compuerta OR continuar con la inserción de la compuerta AND. Dando clic en el botón ENTRADA, seleccionar E03 para la SEGUNDA ENTRADA (HAY QUE RECORDAR QUE LA PRIMERA ENTRADA FUE LA VARIABLE INTERMEDIARIA GENERADA POR EL PROGRAMA) e ingresarla en el renglón 1 (R2) y en la Columna 5 (C5). Seleccionar E04 como tercer entrada en el renglón 1 (R1) y en la Columna 7 (C7).

Seleccionar S00 como salida e ingresarla en el renglón 1 (R1) y en la columna 9 (C9). Realizar el proceso de UNIÓN entre las ENTRADAS y la SALIDA. Recordando que este proceso se lleva a cabo con el botón CABLE. Finalmente esta estructura de COMPUERTAS utilizará cuatro renglones de la ZONA DE TRABAJO mostrados en la figura A.96:

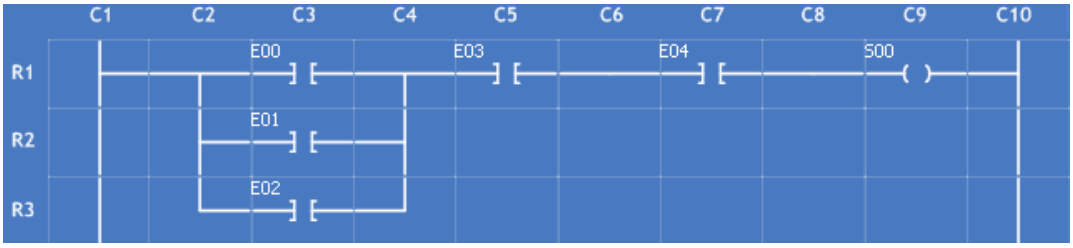


Figura A.96 – Compuerta Lógica OR de tres entradas cuya Salida es la Entrada de una Compuerta Lógica AND.

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

OR3#1 E00,E01,E02,IXXX,111;
AND3#1 IXXX,E03,E04,S00,111;

Donde XXX son los números utilizados por el programa para nombrar a las variables intermediarias.

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

El ejemplo anterior sólo muestra un pequeño panorama de lo que pueden realizar las herramientas "CONECTORES AL INICIO" Y "CONECTORES AL FINAL". La utilización de dichas herramientas permite una gran gama de posibilidades, que quedará en manos del usuario, para su explotación.

A.3.12 Flip-Flop Asíncrono Rs

El PLM puede realizar módulos tipo *latch*, que en la nomenclatura del mismo se denominan como Flip-Flops asíncronos R-S (FFARS), teniéndose para este ML, la capacidad de predefinir el nivel de verificación de las entradas "S" y "R", además de poder predefinir el nivel que ha de tener la salida cuando ambas entradas se verifican y el valor que se desea que tome la misma al iniciar el programa en SIIL1 que ejecuta el PLM en un momento dado.

Un FLIP-FLOP ASÍNCRONO RS (FFARS) está representado por la figura A.97 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un FFARS):

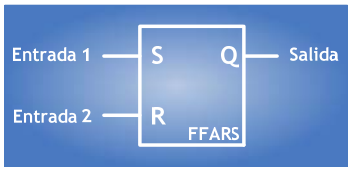


Figura A.97 – Flip-Flop Asíncrono RS.

La figura A.98 es la representación en diagrama de escalera.

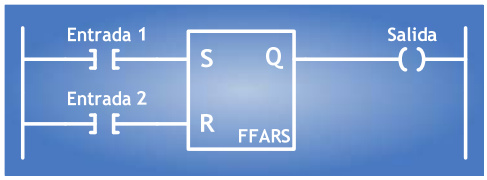


Figura A.98 – Flip-Flop Asíncrono RS en el Diagrama de Escalera.

Para incrustar un FFARS en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón “Flip-Flop RS”, el cual se encuentra ubicado en la esquina superior izquierda de la ventana. Aparecerá el siguiente cuadro de diálogo:

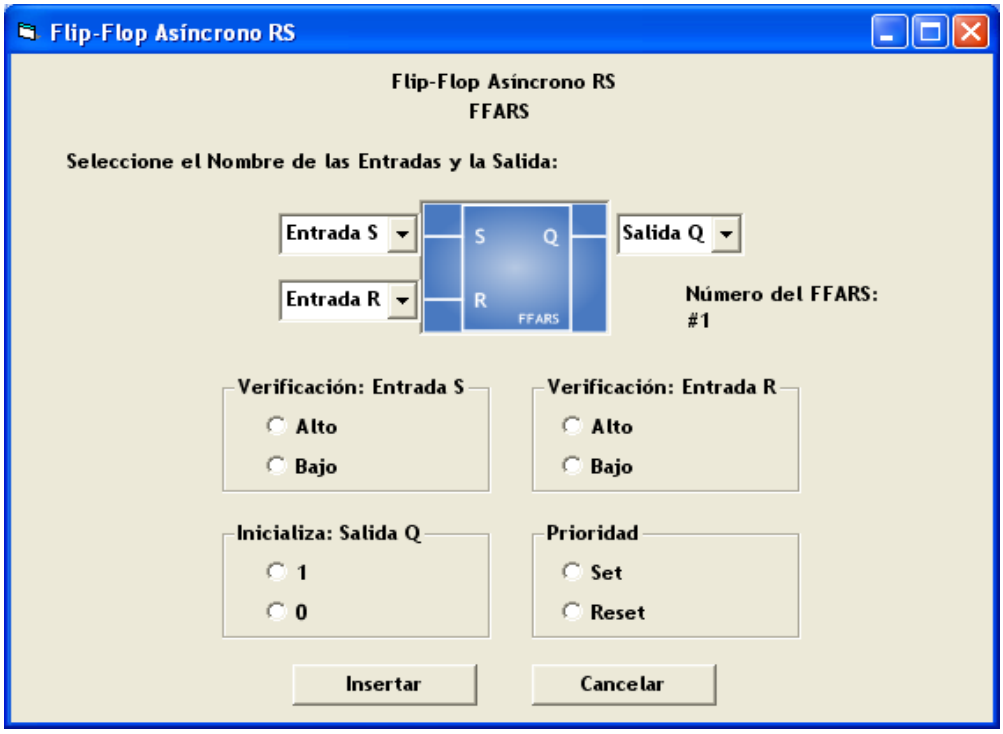


Figura A.99 – Ventana de Configuración para el Flip-Flop Asíncrono RS.

La figura A.99 muestra la ventana que permitirá configurar los parámetros de un FFARS. Lo primero es elegir las VBE (Entrada S y Entrada R) del módulo lógico. La sintaxis de las variables es la misma que la utilizada para configurar las ENTRADAS y SALIDAS de las COMPUERTAS LÓGICAS por lo que si la sintaxis de alguna de estas es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón “Insertar” de esta misma ventana (ver figura A.100):

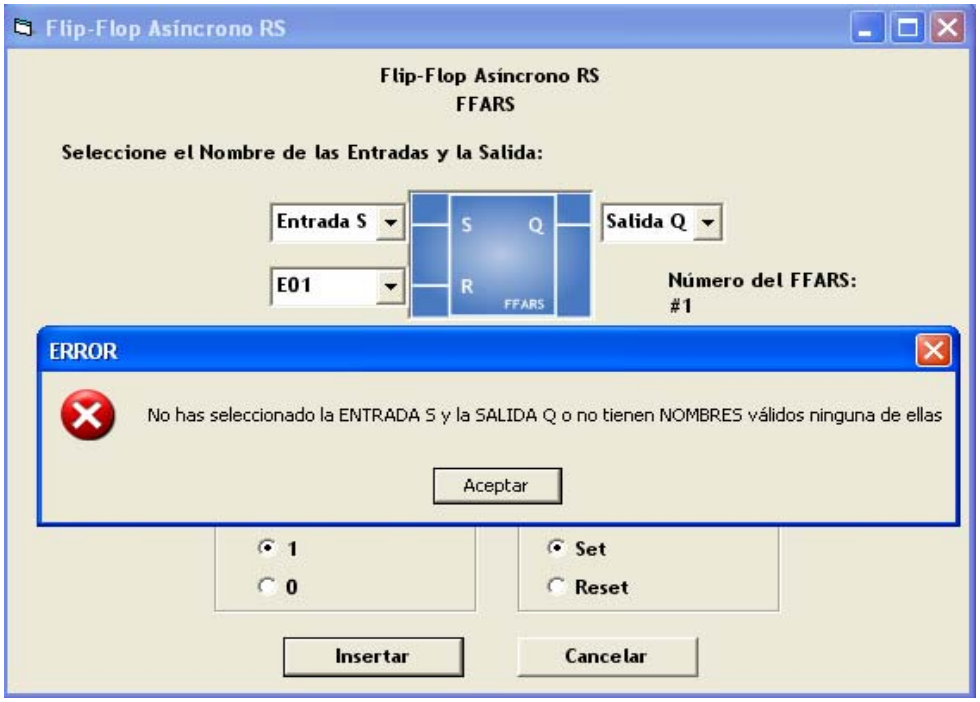


Figura A.100 – Ventana de Error al configurar los nombres de entradas y/o salidas en el Flip-Flop Asíncrono RS.

El cuadro de error menciona que no se han seleccionado la ENTRADA y la SALIDA Q o no tienen nombres válidos ninguna de ellas. Esto ocurre porque al dar clic en el botón "Insertar" el programa toma como nombres los textos actuales que aparecen en los campos de las correspondientes entradas y la salida. Lo que sucedió en este caso es que no se seleccionó la sintaxis adecuada en los nombres de la ENTRADA S y la SALIDA Q y se dejaron como texto los nombres DEFAULT que se les asignaron (Entrada S y Salida Q). El usuario deberá elegir la variable correcta en cada caso, si esto no acontece el mensaje de ERROR seguirá apareciendo y no se podrá configurar el módulo lógico y en consecuencia no se incrustará en la ZONA DE TRABAJO.

El siguiente paso en la configuración es seleccionar una de las dos opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Verificación: Entrada S**

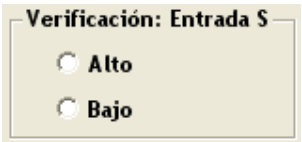


Figura A.101 – Caja para configurar el tipo de Verificación de la Entrada "S" del Flip-Flop Asíncrono RS.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “S” sea cero, en otro caso deberá ser Alto.

- **Verificación: Entrada R**

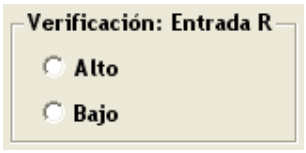


Figura A.102 – Caja para configurar el tipo de Verificación de la Entrada “R” del Flip-Flop Asíncrono RS.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “R” sea cero, en otro caso deberá ser Alto.

- **Inicializa: Salida Q**

Figura A.103 – Caja para configurar el tipo de Inicio de la Salida “Q” del Flip-Flop Asíncrono RS.

Habrá de ser cero, si se desea la Salida “Q” se inicialice en cero, en otro caso deberá ser uno.

- **Prioridad**

Figura A.104 – Caja para configurar la Prioridad del Flip-Flop Asíncrono RS.

El hecho de que la Entrada SET (Entrada S) tenga prioridad implica que si ambas entradas SET y RESET se verifican simultáneamente la Salida Q será uno lógico, por otro lado, prioridad para la entrada RESET significa que al verificarse ambas entradas del Flip-Flop la Salida Q será puesta en cero lógico.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el siguiente mensaje de ERROR:

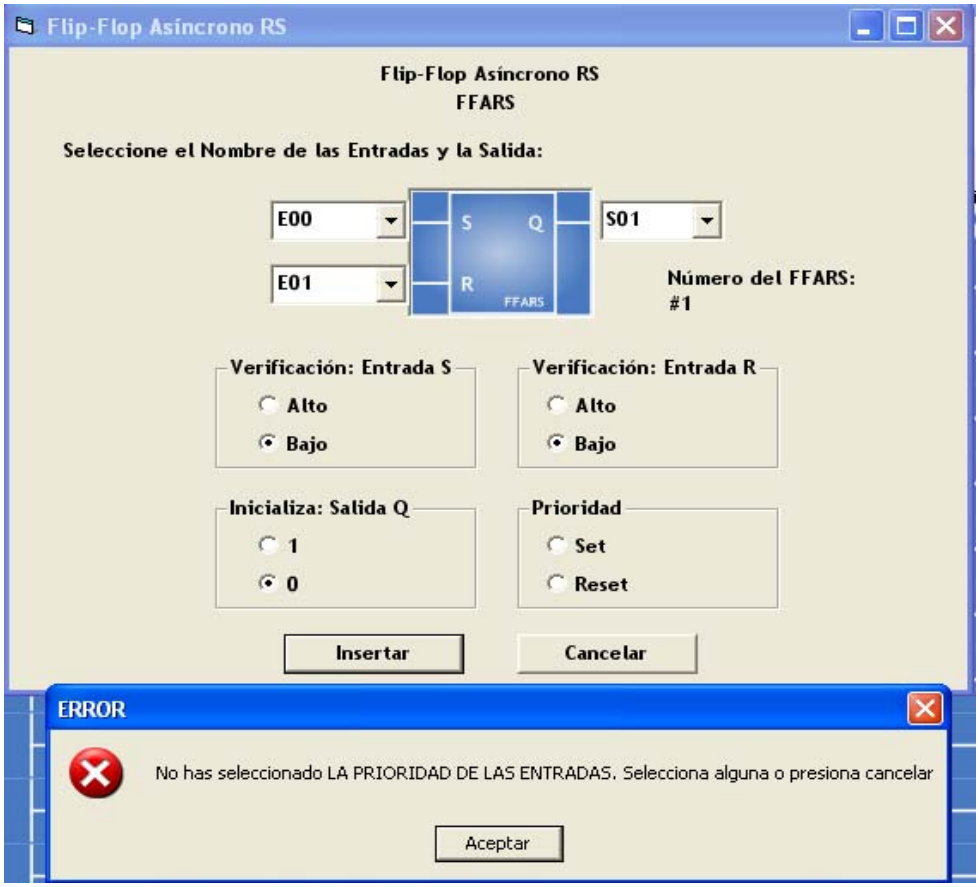


Figura A.105 – Ventana de Error al configurar la Prioridad de las entradas en el Flip-Flop Asíncrono RS.

La figura A.105 muestra un ERROR donde se menciona que no se ha seleccionado la PRIORIDAD DE LAS ENTRADAS. Lo que sucedió es que no se seleccionó alguna de las dos opciones en el recuadro PRIORIDAD. El usuario deberá escoger una opción para cada RECUADRO, si esto no acontece el mensaje de ERROR seguirá apareciendo y no se podrá configurar el módulo lógico y en consecuencia no se incrustará en la ZONA DE TRABAJO.

El único parámetro sobre el cual no se tiene control es el número del FFARS:

Figura A.106 – Número asignado por el programa para el Flip-Flop Asíncrono RS.

La Figura A.106 muestra la leyenda “Número del FFARS: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el FFARS, se deberá insertar dando clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Para insertarlo en el renglón 2 (R2) habrá que dar clic en cualquier COLUMNA del RENGLÓN 2. Este módulo utilizará dos renglones de la ZONA DE TRABAJO tal como lo muestra la figura A.107:

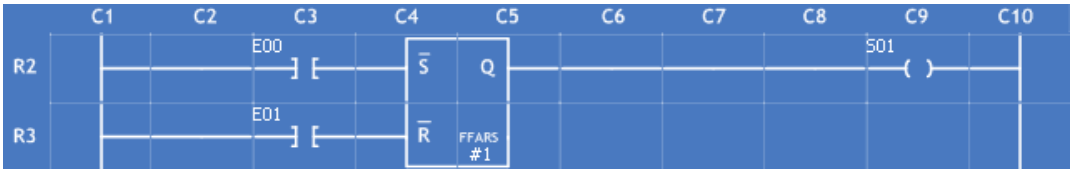


Figura A.107 - Flip-Flop Asíncrono RS insertado en los renglones R2 y R3 de la Zona de Trabajo.

Como se puede observar en la figura A.108 el FFARS muestra los parámetro configurado, como lo son: variables de ENTRADAS y de SALIDA, así como la verificación de las ENTRADAS "S" Y "R", además del número del módulo lógico que se a insertando, en este caso el #1.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA "S": E00
ENTRADA "R": E01
SALIDA "Q": S01.
VERIFICACIÓN: ENTRADA S: BAJO, que esta representado a través del primer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
VERIFICACIÓN: ENTRADA R: BAJO, que esta representado a través del segundo dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
INICIALIZA: SALIDA Q: 0, que esta representado a través del tercer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
PRIORIDAD: SET, que esta representado a través del cuarto dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

FFARS#1 E00,E01,S01,0110;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.13 Contador De Eventos

El PLM puede realizar módulos CONTADORES DE EVENTOS ascendentes o descendentes, siendo los valores de la cuenta comprendidos en un determinado intervalo, pudiendo el usuario definir tanto la cuenta inicial como la final, este ML tiene tres entradas y una salida, véase la figura A.108 las entradas son: entrada "D" sensible a flancos que hacen que se modifique la cuenta, entrada de restablecimiento (RESET) que al verificarse hace que la cuenta retorne a su valor inicial desverificándose la salida (TF), y entrada de congelamiento que al verificarse hace que el contador conserve la cuenta sin responder a los niveles lógicos presentes en las otras dos entradas; la salida de este módulo (TF) se verifica cuando la cuenta ha llegado a su valor final.

Para este ML se tiene la capacidad de predefinir los límites del intervalo de cuenta, el tipo de flanco que incrementa o decrementa la cuenta, el nivel de verificación de las entradas de RESET y congelamiento, el nivel de verificación de la salida testigo de cuenta final y el tipo de cuenta (ascendente o descendente) a efectuar.

Cabe señalar, que para la correcta operación del contador de eventos, se requiere que el intervalo de tiempo entre dos flancos consecutivos sea mayor de 10 ms.

Un CONTADOR DE EVENTOS (CONTA) está representado por la figura A.108 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un CONTADOR DE EVENTOS):

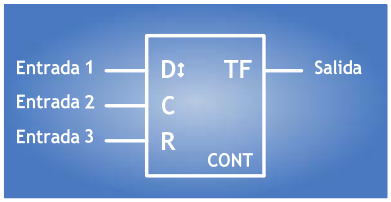


Figura A.108 – Contador de Eventos.

La figura A.109 muestra un contador de eventos mediante un diagrama de escalera.

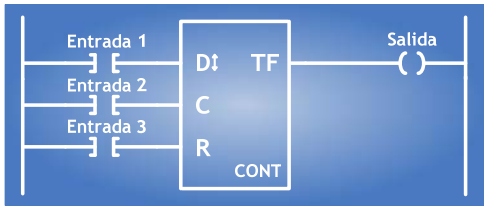


Figura A.109 – Contador de Eventos en el Diagrama de Escalera.

Para incrustar un CONTADOR DE EVENTOS en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón "Contador", el cual se encuentra ubicado en la parte superior central de la ventana. Aparecerá un cuadro de diálogo como el de la figura A.110:

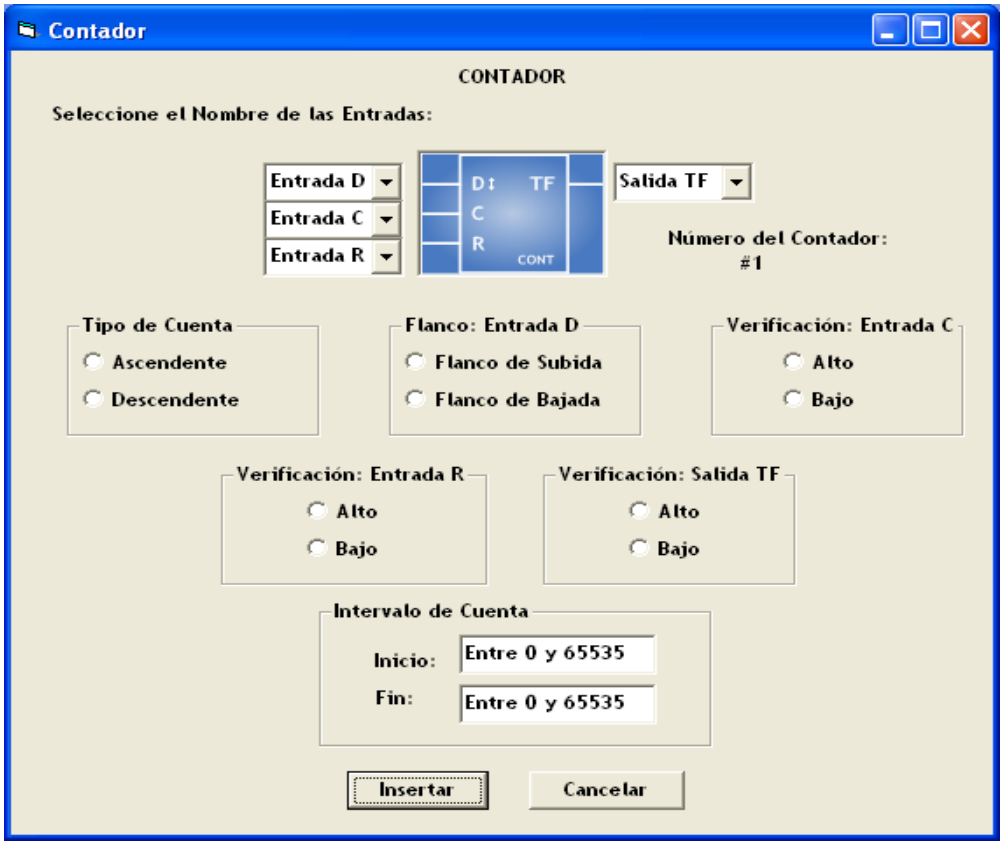


Figura A.110 – Ventana de Configuración para el Contador de Eventos.

La figura A.110 muestra la ventana que permitirá configurar los parámetros de un CONTADOR DE EVENTOS. Primeramente se eligen las VBE (Entrada D, Entrada C y Entrada R) y la SALIDA (Salida TF) del módulo lógico. La sintaxis de las variables de entrada y salida ya fue anteriormente descrita.

Si la configuración es errónea aparecerá un mensaje de ERROR (figura A.111) cuando se dé clic en el botón “Insertar” de esta misma ventana:

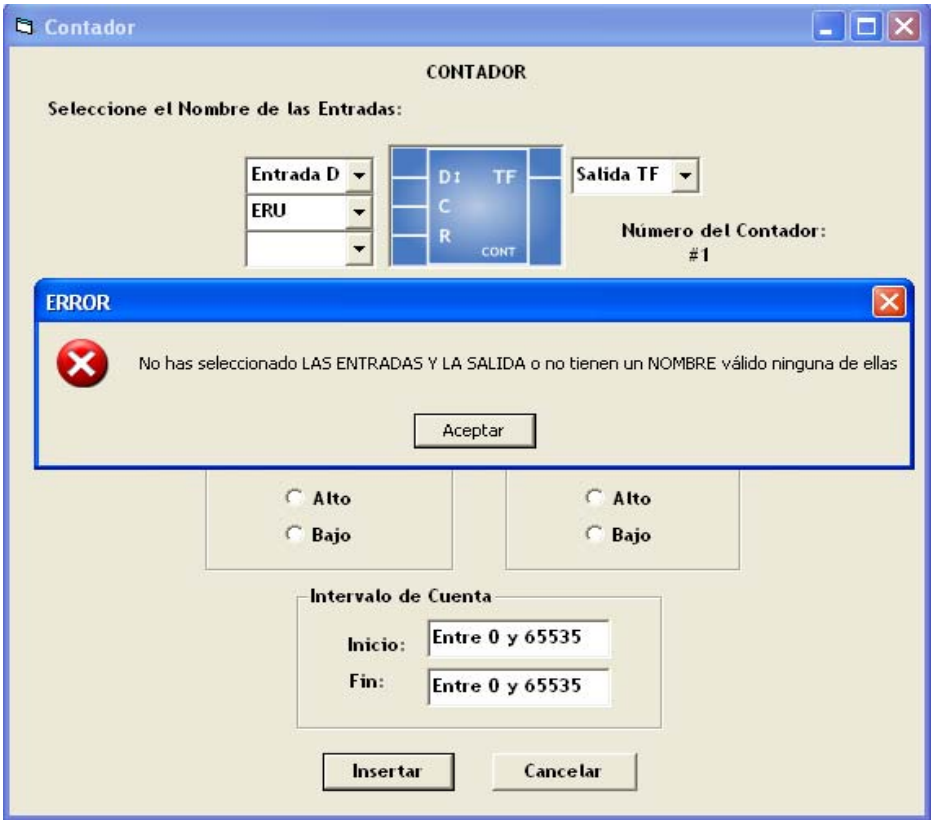


Figura A.111 – Ventana de Error al configurar los nombres de entradas y/o salidas en el Contador de Eventos.

El cuadro de error menciona que no se han seleccionado las ENTRADAS y la SALIDA o no tienen nombres válidos ninguna de ellas. Esto ocurre porque al dar clic en el botón "Insertar" el programa toma como nombres los textos actuales que aparecen en los campos de las correspondientes entradas y la salida.

Lo que sucedió en este caso es que no se seleccionó la sintaxis adecuada en las VBE; en la ENTRADA D se dejo como texto la entrada default (Entrada D), en la ENTRADA C se eligió una sintaxis que no coincide con la establecida, en la ENTRADA R se dejo el campo vacío y la SALIDA Q tiene como texto el nombre DEFAULT que se le asignó (Salida Q).

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Tipo de Cuenta**

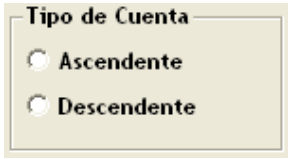


Figura A.112 – Caja para configurar el tipo Cuenta del Contador de Eventos.

Si se desea que el módulo contador de eventos realice una cuenta ASCENDENTE se deberá seleccionar la opción: Ascendente, en otro caso deberá ser: Descendente.

- **Flanco: Entrada D**

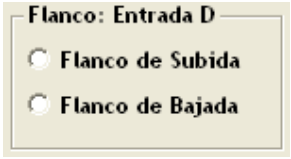


Figura A.113 – Caja para configurar el tipo de Flanco de la Entrada “D” del Contador de Eventos.

Si se desea que se modifique la cuenta para flancos de subida en la entrada de disparo “D” se deberá seleccionar la opción: Flanco de Subida, en otro caso deberá ser: Flanco de Bajada.

- **Verificación: Entrada C**

Figura A.114 – Caja para configurar el tipo de Verificación de la Entrada “C” del Contador de Eventos.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “C” sea cero, en otro caso deberá ser Alto.

- **Verificación: Entrada R**

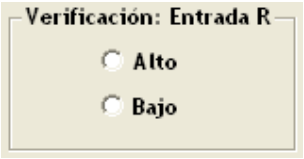


Figura A.115 – Caja para configurar el tipo de Verificación de la Entrada “D” del Contador de Eventos.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “R” sea uno, en otro caso deberá ser Alto.

- **Verificación: Salida TF**

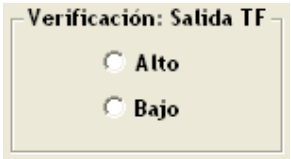


Figura A.116 – Caja para configurar el tipo de Verificación de la Salida “TF” del Contador de Eventos.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Salida “TF” (testigo fin de carrera) sea cero, en otro caso deberá ser Alto.

- **Intervalo de Cuenta**

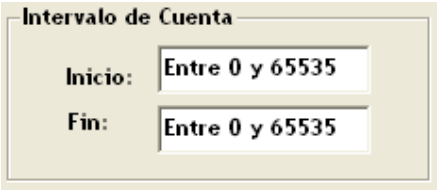


Figura A.117 – Caja para configurar el Intervalo de Cuenta del Contador de Eventos.

Inicio: denota al valor de la cuenta inicial, debiendo el mismo estar comprendido entre 0 y 65535, este valor debe ser menor que el correspondiente a la cuenta final si el contador es ascendente, en otro caso el valor declarado para la cuenta inicial deberá ser mayor que la cuenta final.
Fin: denota el valor de la cuenta final, debiendo el mismo estar comprendido entre 0 y 65535.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR. Un caso especial es el mensaje de ERROR que se envía en el recuadro: Intervalo de Cuenta.

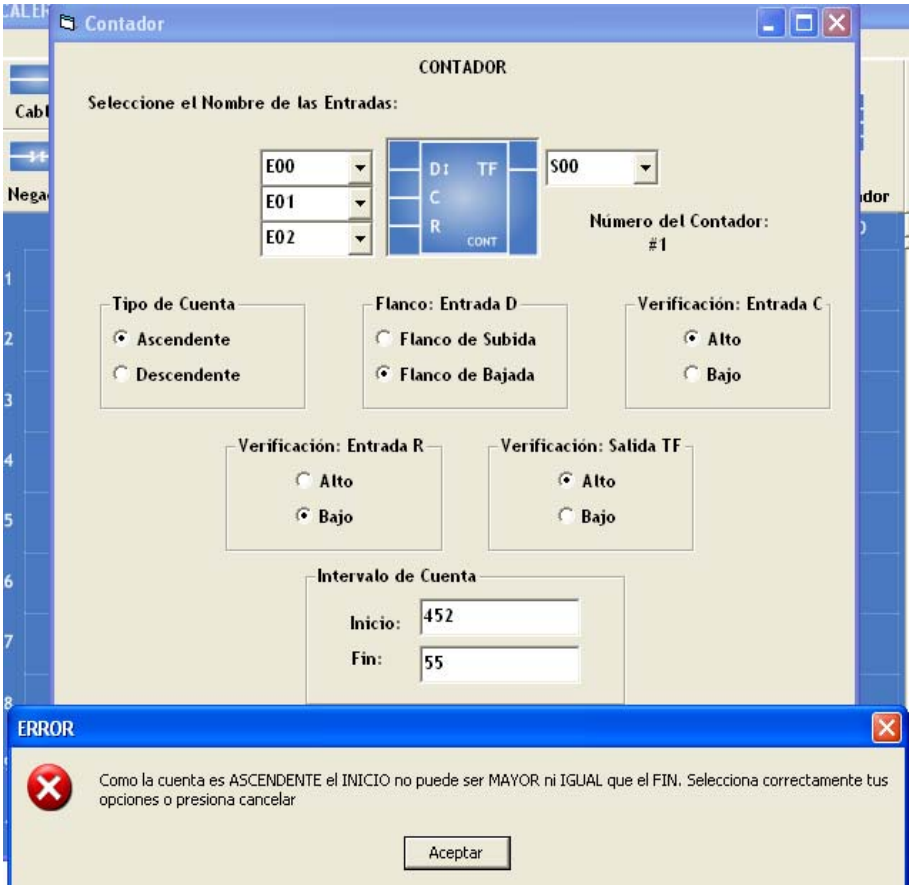


Figura A.118 – Ventana de Error al configurar el Tipo de Cuenta y su Intervalo en el Contador de Eventos.

El mensaje de ERROR mostrado en la figura A.118 nos indica que “Como la cuenta es ASCENDENTE el INICIO no puede ser MAYOR ni IGUAL que el FIN”, esto es evidente al observar los campos Inicio y Fin de la misma figura. Un mensaje de ERROR parecido se enviaría si la cuenta fuera DESCENDENTE y el FIN fuera MAYOR al INICIO. Otros mensajes de ERROR se mostrarían si los campos INICIO Y FIN estuvieran vacíos o si alguno de ellos se encontrará fuera el intervalo.

El único parámetro sobre el cual no se tiene control es el número del CONTADOR DE EVENTOS. Mostrado en la ventana como: “Número del Contador: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el CONTADOR DE EVENTOS, la tarea siguiente es dar clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 1 (R1) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 1. Este módulo utilizará tres renglones de la ZONA DE TRABAJO (figura A.119):

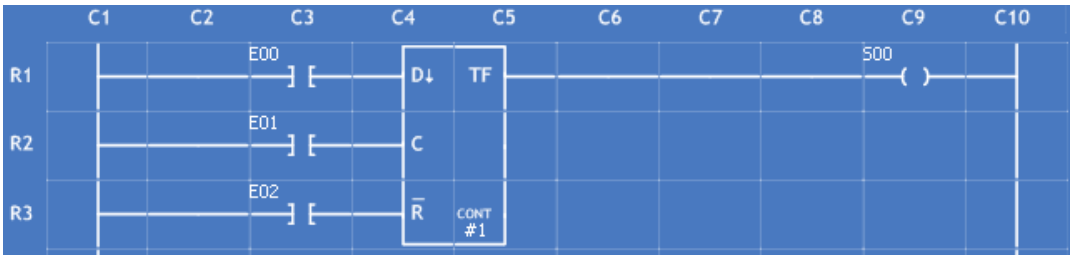


Figura A.119 – Contador de Eventos insertado en los renglones R1, R2 y R3 de la Zona de Trabajo del “PUMA ESCALERA”.

El CONTADOR DE EVENTOS muestra varios datos que configuramos, como las variables de ENTRADA y las de SALIDA, la verificación de las ENTRADAS “C” Y “R” y la SALIDA “TF”, así como el flanco de disparo de la ENTRADA “D”, además del número del módulo lógico que se está insertando, en este caso el #1. Queda como tarea del usuario conservar los respectivos datos de cada módulo lógico insertado para al final compararlos con la TRADUCCIÓN resultante.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA “D”: E00
ENTRADA “C”: E01
ENTRADA “R”: E02
SALIDA “TF”: S00
FLANCO: ENTRADA D: BAJADA, que esta representado a través del primer dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
VERIFICACIÓN: ENTRADA C: ALTO, que esta representado a través del segundo dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1

VERIFICACIÓN: ENTRADA R: BAJO, que esta representado a través del tercer dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
TIPO DE CUENTA: ASCENDENTE, que esta representado a través del cuarto dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
VERIFICACIÓN: SALIDA TF: ALTO, que esta representado a través del quinto dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

CONTA#1 E00,E01,E02,S00,452,700,01111;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.14 Secuenciador De Estados De Nbxne

El PLM puede realizar módulos SECUENCIADORES DE ESTADOS, el número de bits de la palabra de estado puede ir de uno a ocho, tanto el número de estados como los diferentes valores que presenten los mismos, así como su sucesión son programables por el usuario, las declaraciones de los valores deseados para los estados pueden hacerse en formato binario o hexadecimal. El software de traducción limita a 1000 el número de estados asociados con un secuenciador, esto aún cuando en la mayoría de las aplicaciones prácticas este tope sería sensiblemente menor.

Un secuenciador presentará tres entradas y NB+1 salidas, donde NB es el número de bits en la palabra de estado; la primera entrada es sensible a flancos y se denomina como “D”, al detectarse un flanco en ella se coloca en las salidas asociadas el siguiente estado de la lista que el usuario haya declarado, si el estado colocado es el último de la lista se verifica una salida denominada como “TF” (testigo de fin de carrera); la segunda entrada, se denomina “C” (entrada de congelamiento) y al verificarse, hace que el secuenciador permanezca en el estado presente sin responder a las otras dos entradas; la tercera entrada se denomina “R” (RESET) y al verificarse, hace que el secuenciador presente el estado inicial y desverifique la salida “TF”, la primera vez que se ejecuta el código se invoca el accionamiento de RESET; ver figura A.14.1

Una vez que se coloca el estado final el secuenciador no responde a los flancos, para reiniciar el ciclo hay que verificar la entrada de RESET.

Un SECUENCIADOR DE ESTADOS (SECNB) está representado por la figura A.120 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un SECNB):

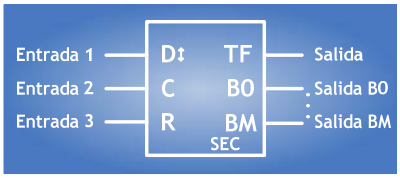


Figura A.120 – Secuenciador de Estados de NBxNE.

M = NB-1; donde NB es el número de bits de la palabra de estado.

La figura A.121 muestra un secuenciador de estados mediante un diagrama de escalera

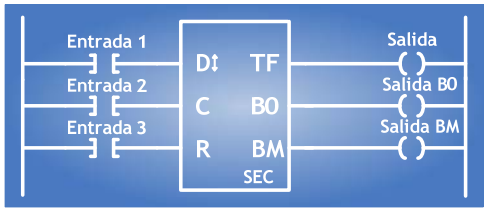


Figura A.121 – Secuenciador de Estados de NBxNE en el Diagrama de Escalera.

Para incrustar un SECUENCIADOR DE ESTADOS en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón “Secuenciador”, el cual se encuentra ubicado en la parte superior derecha de la ventana. Aparecerá el siguiente cuadro de diálogo:

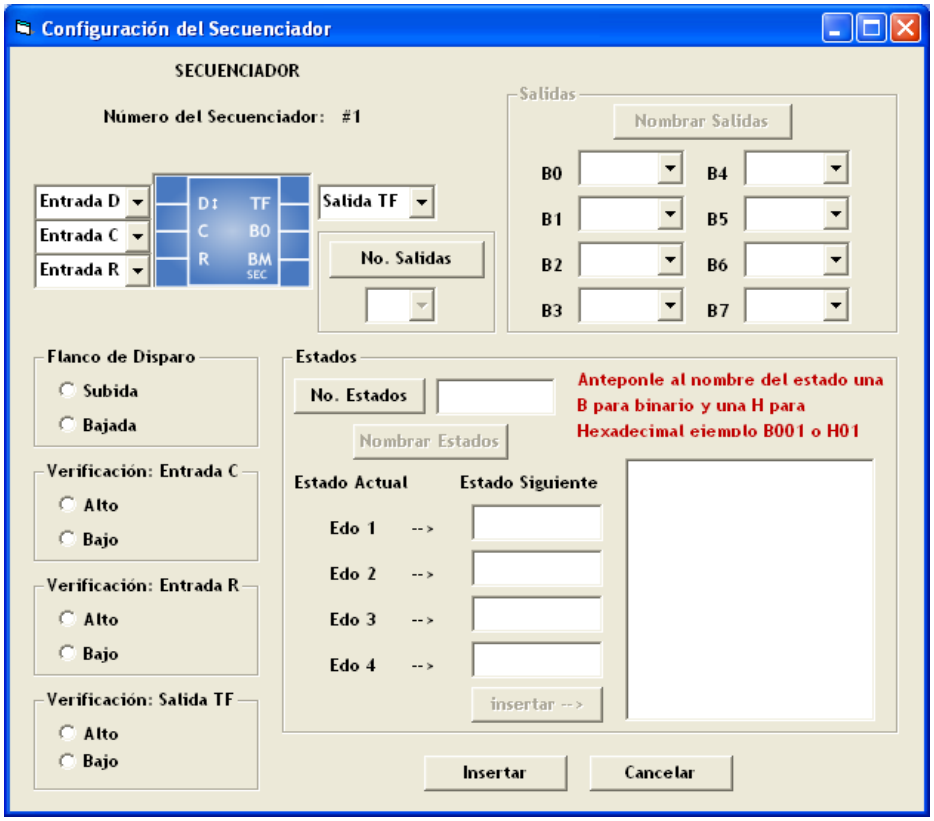


Figura A.122 – Ventana de Configuración del Secuenciador de Estados de NBxNE.

La figura A.122 muestra la ventana que permitirá configurar los parámetros de un SECUENCIADOR DE ESTADOS. Primeramente se eligen las VBE las ENTRADAS (Entrada D, Entrada C y Entrada R) y la SALIDA (Salida TF) del módulo lógico. La sintaxis de las variables de entrada y salida ya fue anteriormente descrita.

Si la sintaxis de alguna de estas es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón "Insertar" de esta misma ventana. Estos envíos de ERROR ya fueron analizados y resueltos en la configuración del FFARS y el CONTADOR DE EVENTOS por si es necesario tomar alguna referencia de éstos.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Flanco de Disparo**

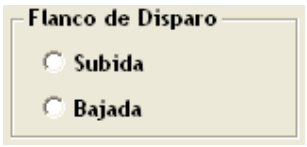


Figura A.123 – Caja para configurar el Flanco de Disparo de la Entrada "D" del Secuenciador de Estados.

Si se desea que se coloque el estado siguiente para flancos de subida en la entrada de disparo "D" se deberá seleccionar la opción: Subida, en otro caso deberá ser: Bajada.

- **Verificación: Entrada C**

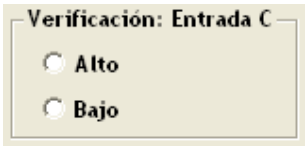


Figura A.124 – Caja para configurar el Tipo de Verificación de la Entrada "D" del Secuenciador de Estados.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada "C" sea cero, en otro caso deberá ser Alto.

- **Verificación: Entrada R**

Figura A.125 – Caja para configurar el Tipo de Verificación de la Entrada "R" del Secuenciador de Estados.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada "R" sea uno, en otro caso deberá ser Alto.

- **Verificación: Salida TF**

Figura A.126 – Caja para configurar el Tipo de Verificación de la Salida "TF" del Secuenciador de Estados.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Salida "TF" (testigo fin de carrera) sea cero, en otro caso deberá ser Alto.

- **No. Salidas**

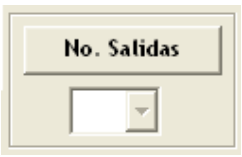


Figura A.127 – Caja para configurar el No. de Salidas del Secuenciador de Estados.

Denota el número de bits (número de salidas) de la palabra de estado, debiendo el mismo estar comprendido entre uno y ocho, esto definido por el usuario. Para activarlo hay que dar clic sobre el botón "No. Salidas"

- **Salidas**

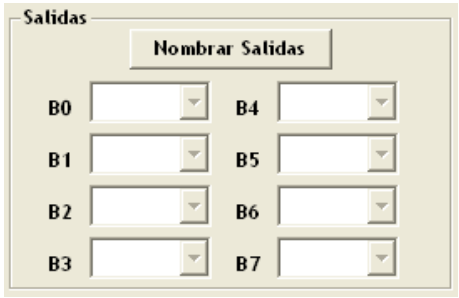


Figura A.128 – Caja para configurar el Nombre de las Salidas del Secuenciador de Estados.

Para poder elegir las Salidas tiene que estar ya definido el número de salidas. Una vez hecho esto, habrá que dar clic en el botón "Nombrar Salidas" (mostrado en la figura A.14.9) para poder habilitar las salidas correspondientes. Las salidas tendrán la sintaxis correspondiente a **Variables Booleanas de Salida (VBS).** Y **Variables Booleanas Intermediarias (VBI)**. Previamente descritas.

- **Estados**

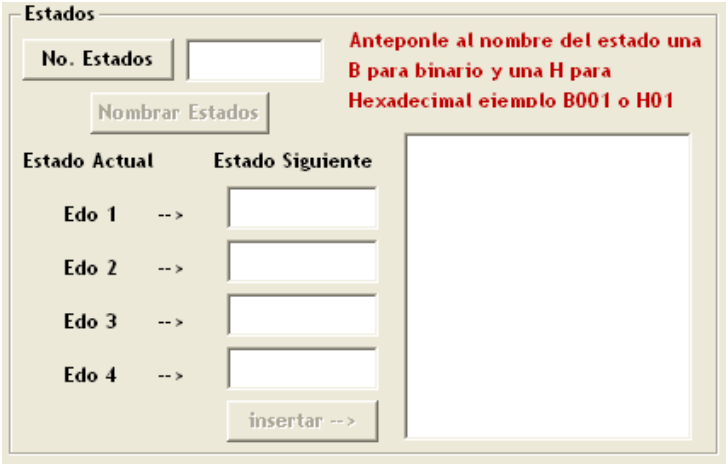


Figura A.129 – Caja para configurar el Número de Estados, Nombres y hacia que Estados se dirigen respectivamente.

Para poder elegir los ESTADOS hay que declarar primero el número de estados. Para activarlo sólo hay que dar clic sobre el botón "No. Estados". El software de traducción limita el número de estados a un intervalo determinado, que va **desde 2 hasta 1000** en un secuenciador. Hecho lo anterior hay que activar el botón "Nombrar Estados" dando un clic sobre él. Las declaraciones de los valores deseados para los estados pueden hacerse en formato binario o hexadecimal. Por ejemplo, **B000, B010, B110 o B111** para el formato binario y **H07, H00, H01 o H02** para el formato hexadecimal.

Los estados se elegirán en grupos de cuatro y serán validados cuando se dé clic en el botón "Insertar→" y estos aparezcan en el cuadro que aparece a la derecha en la figura A.14.10. Así por ejemplo, si el número de estados es siete se elegirán los 4 primeros estados, a continuación se dará clic en "Insertar→" y si tienen la correcta estructura aparecerán en el cuadro mencionado, enseguida los campos de texto quedarán vacíos para poder elegir los tres estados restantes. Cuando se nombren estos estados correctamente, aparecerán en el cuadro derecho enseguida de los cuatro estados anteriormente creados, cuando se dé clic en el botón "Insertar→".

La correcta estructura de los estados implican las siguientes características:

- Se deberá respetar el número de estados seleccionado a la hora de elegir estos, es decir, no se permitirá elegir estados que excedan el número de estados elegido, ni dejar vacíos los campos disponibles para elegirlos. El software reconocerá incongruencias a este respecto.
- Tener en cuenta en todo momento el número de salidas del secuenciador que se haya seleccionado. Ya que estas determinarán el rango de valores adecuado para elegir cada estado. Así por ejemplo, si el número de salidas es 4 el rango de valores irá desde el B0000 hasta el B1111 en formato binario y H00 hasta el H0A en formato hexadecimal.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR.

Vale la pena analizar algunos de ellos:

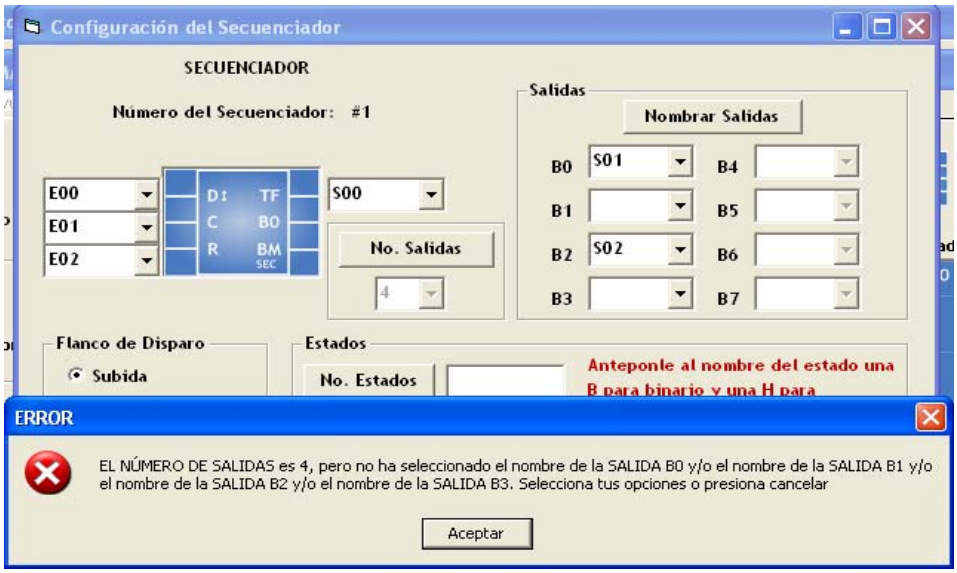


Figura A.130 – Ventana de Error al configurar el Nombre de las Salidas en el Secuenciador de Estados.

El mensaje de ERROR mostrado en la figura A.130 indica que “EL NÚMERO DE SALIDAS es 4, pero no ha seleccionado el nombre de la SALIDA B0 y/o el nombre de la SALIDA B1 y/o el nombre de la SALIDA B2 y/o el nombre de la SALIDA B3”. Esto sucede porque se seleccionó 4 como número de salidas pero las SALIDAS B1 y B3 no fueron elegidas. Como en casos anteriores el mensaje de ERROR no desaparecerá hasta que se seleccionen las opciones correctas.

Otro punto a destacar en los mensajes de ERROR es el relacionado con las SALIDAS, ya que si alguna salida coincide con una o más salidas se enviará un ERROR mencionando esta incongruencia. Esto también incluye a la “Salida TF” ya que la estructura de la misma corresponde con la de la salidas B0 hasta la salida BM.

Analicemos ahora un caso de ERROR relacionado con los ESTADOS:

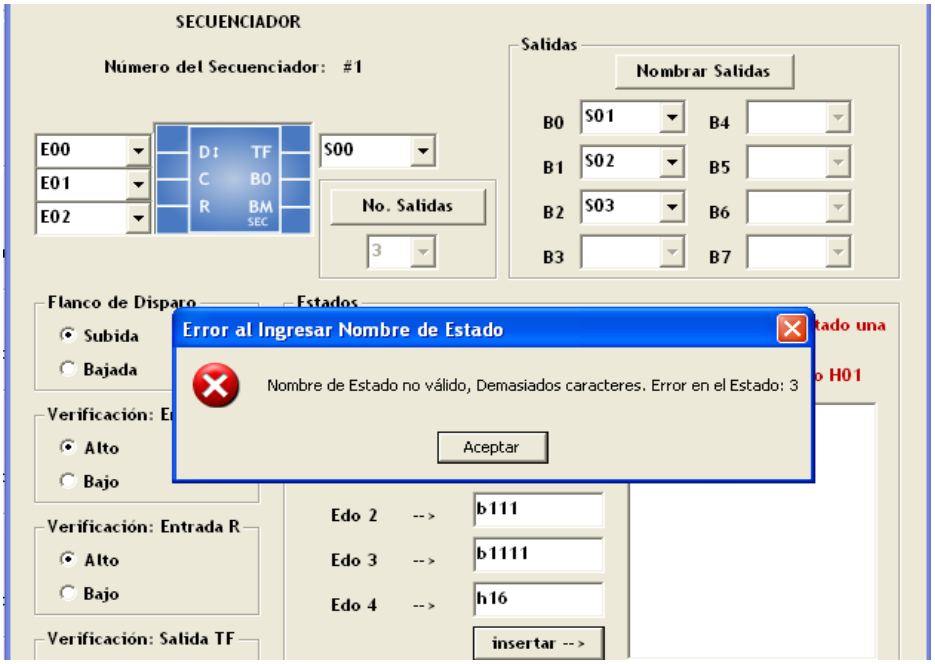


Figura A.131 – Ventana de Error al configurar el Nombre del Estado 3 en el Secuenciador de Estados.

El mensaje de ERROR mostrado en la figura A.131 indica: “Nombre de Estado no válido, Demasiados caracteres. Error en el Estado: 3”. Este error es enviado porque se ha seleccionado 3 como el número de SALIDAS en el SECUENCIADOR, por lo que el rango de valores aceptado va desde el b000 hasta el b111 en binario y desde el h00 hasta el h07 en hexadecimal. Y dado que en el Estado 3 se ha seleccionado que se dirigirá al estado b1111, el cual no existe en el intervalo mencionado.

La figura A.131 muestra a su vez que el estado 4 se dirigirá al estado h16 (Edo 4 → h16), esta declaración enviará el siguiente mensaje de ERROR:

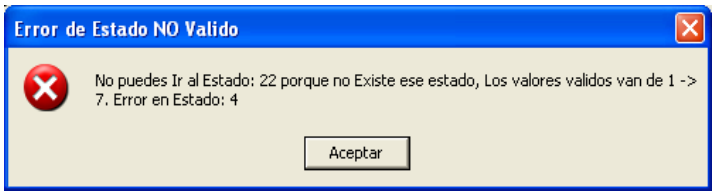


Figura A.132 – Ventana de Error al configurar al Estado donde se dirige el Estado 4 en el Secuenciador de Estados.

El mensaje de ERROR mostrado en la figura A.132 indica: “No puedes Ir al Estado: 22 porque no Existe ese estado, Los valores válidos van de 1 → 7. Error en el Estado: 4”. Este error es enviado porque se ha seleccionado 3 como el número de SALIDAS en el SECUENCIADOR, por lo que el rango de valores aceptado va desde el b000 hasta el b111 en binario y desde el h00 hasta el h07 en hexadecimal. Y dado que en el Estado 4 se ha seleccionado que se dirigirá al estado h16, el cual no existe en el intervalo mencionado.

El único parámetro sobre el cual no se tiene control es el número del SECUENCIADOR DE ESTADOS. Mostrado en la ventana como: “Número del Secuenciador: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número

permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el SECUENCIADOR DE ESTADOS, la tarea siguiente es dar clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 1 (R1) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 1. Este módulo utilizará tres renglones de la ZONA DE TRABAJO como la que se muestra en la figura A.133:

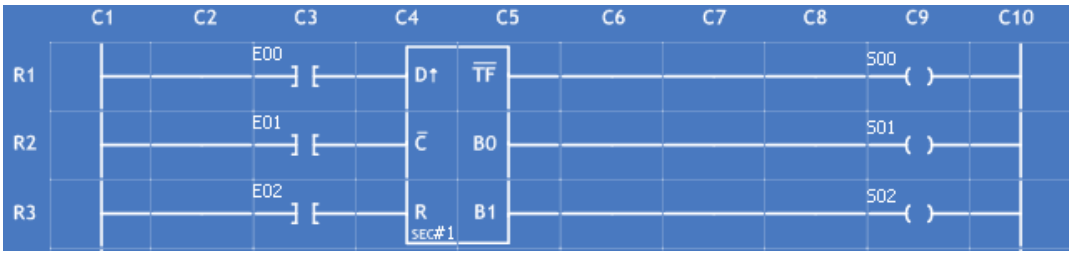


Figura A.133 – Secuenciador de Estados insertado en los renglones R1, R2 y R3 de la Zona de Trabajo del “PUMA ESCALERA”.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA “D”: E00
ENTRADA “C”: E01
ENTRADA “R”: E02
SALIDA “TF”: S00
SALIDA “B0”: S01
SALIDA “B1”: S02
NÚMERO DE SALIDAS: 2, mostrado enseguida de “SEC” de la sentencia en lenguaje SIIL1 mostrada abajo.
NÚMERO DE ESTADOS: 2, mostrado enseguida de la salida “S01” de la sentencia en lenguaje SIIL1 mostrada abajo.
FLANCO: ENTRADA D: SUBIDA, que esta representado a través del primer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
VERIFICACIÓN: ENTRADA C: BAJO, que esta representado a través del segundo dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
VERIFICACIÓN: ENTRADA R: ALTO, que esta representado a través del tercer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
VERIFICACIÓN: SALIDA TF: BAJO, que esta representado a través del cuarto dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
ESTADO 1: B01
ESTADO 2: B10

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

SEC2#1 E00,E01,E02,S00,S02,S01,2,1000;
 # B01;
B10;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.15 Temporizador Monodisparo Tipo 1 (TEMPOA)

El PLM puede realizar dos tipos de temporizadores de tipo monodisparo, aquí se describe lo concerniente al temporizador monodisparo de tipo uno, mostrándose en la figura A.134

El intervalo de tiempo correspondiente lo especifica el usuario en la declaración sintáctica correspondiente, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos, como se aprecia en la figura A.134 Tanto el disparo como el restablecimiento (RESET), pueden ser por flanco de subida o bajada, teniéndose además otra entrada denominada como H (habilitación). En la figura A.134 se aprecia que este temporizador tiene capacidad de redisparo.

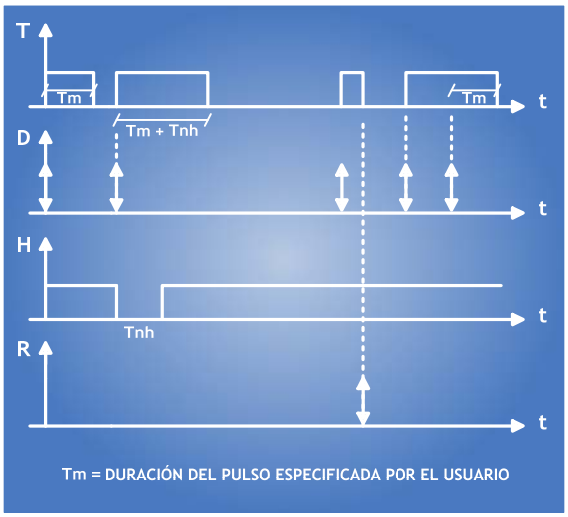


Figura A.134 - Diagrama de tiempos asociado al Temporizador Monodisparo Tipo 1.

Mediante la entrada H se habilita el funcionamiento del temporizador, en caso de que la misma se verifique este ML responde a las otras dos entradas normalmente, si H no se verifica no habrá respuesta a las entradas, si tal verificación ocurre durante el intervalo de verificación de la salida se suspende la cuenta de tiempo asociada, permaneciendo verificada la salida.

La entrada R responde a flancos, que al detectarse desverifican la salida y restablecen a cero el contador de tiempo asociado.

Un TEMPORIZADOR MONODISPARO TIPO 1 (TEMPOA) está representado por la figura A.135 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un TEMPOA):

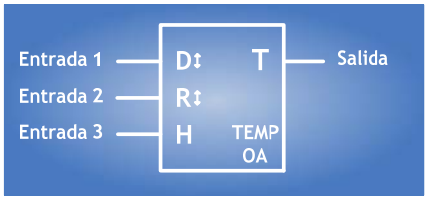


Figura A.135 – Temporizador Monodisparo Tipo 1.

La figura A.136 muestra la misma representación pero en diagrama de escalera.

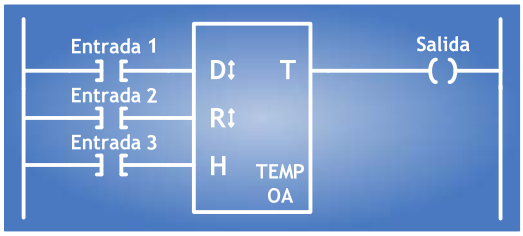


Figura A.136 – Temporizador Monodisparo Tipo 1 en el Diagrama de Escalera.

Para incrustar un TEMPOA en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón “Temporizador OA”, el cual se encuentra ubicado en el costado superior izquierdo de la ventana. Aparecerá el siguiente cuadro de diálogo:

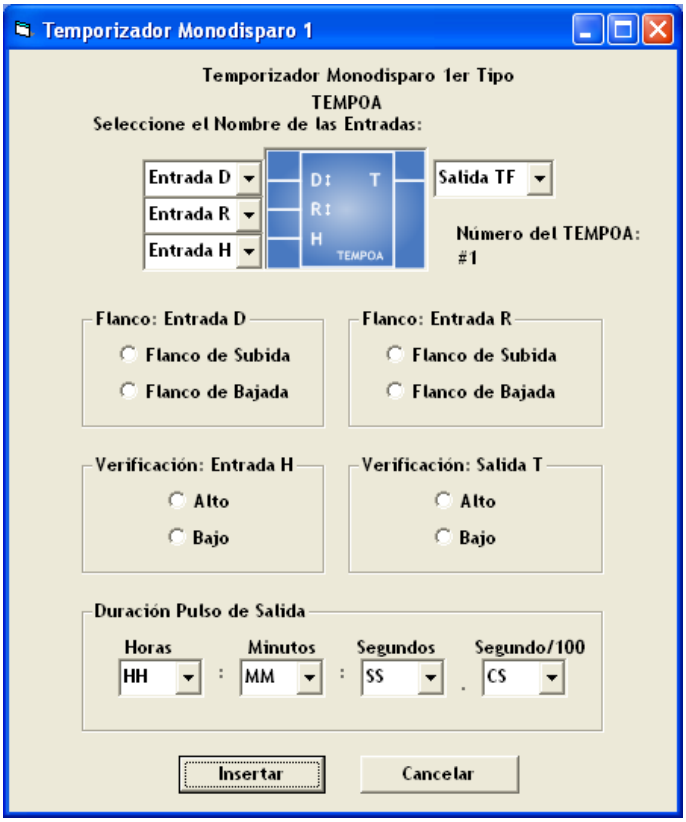


Figura A.137 - Ventana de Configuración del Temporizador Monodisparo Tipo 1.

La figura A.137 muestra la ventana que permitirá configurar los parámetros de un TEMPOA. Primeramente se eligen las VBE las ENTRADAS (Entrada D, Entrada R y Entrada H) y la SALIDA (Salida T) del módulo lógico. La sintaxis de las variables de entrada y salida ya fue anteriormente descrita.

Si la sintaxis de alguna de estas es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón "Insertar" de esta misma ventana. Estos envíos de ERROR ya fueron analizados y resueltos en la configuración del FFARS y el CONTADOR DE EVENTOS por si es necesario tomar alguna referencia de éstos.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Flanco: Entrada D**

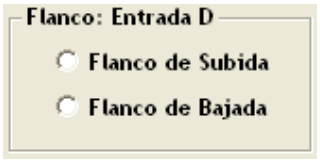


Figura A.138 – Caja para configurar el Flanco de Disparo de la Entrada "D" del Temporizador Monodisparo Tipo 1.

Si se desea que el temporizador se dispare para flancos de subida en la entrada de disparo “D” se deberá seleccionar la opción: Flanco de Subida, en otro caso deberá ser: Flanco de Bajada.

- **Flanco: Entrada R**

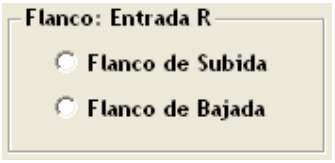


Figura A.139 – Caja para configurar el Flanco de Disparo de la Entrada “R” del Temporizador Monodisparo Tipo 1.

Si se desea que el temporizador se reestablezca para flancos de subida en la entrada de restablecimiento “D” se deberá seleccionar la opción: Flanco de Subida, en otro caso deberá ser: Flanco de Bajada.

- **Verificación: Entrada H**

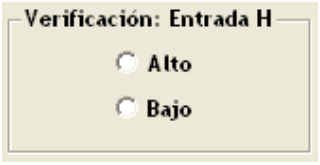


Figura A.140 – Caja para configurar el Flanco de Disparo de la Entrada “H” del Temporizador Monodisparo Tipo 1.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “H” sea cero, en otro caso deberá ser Alto.

- **Verificación: Salida T**

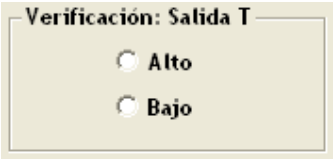


Figura A.141 – Caja para configurar el Tipo de Verificación de la Salida “T” del Temporizador Monodisparo Tipo 1.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Salida “T” sea cero, en otro caso deberá ser Alto.

- **Duración Pulso de Salida**

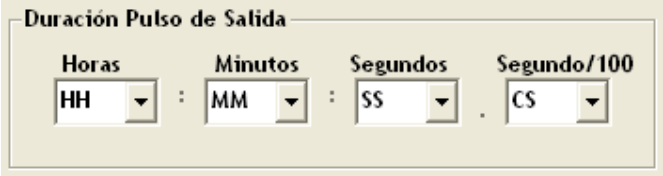


Figura A.142 – Caja para configurar la Duración del Pulso de Salida del Temporizador Monodisparo Tipo 1.

La duración del Pulso de Salida correspondiente lo especifica el usuario, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos.

- HH denota un par de dígitos que especifican el número de horas en el tiempo Tm, siendo 47 el valor máximo aceptado.
- MM denota un par de dígitos que especifican el número de minutos en Tm.
- SS denota un par de dígitos que especifican el número de segundos en Tm.
- CS denota un par de dígitos que especifican las centésimas de segundo en Tm.

Donde Tm es la duración del pulso especificada por el usuario.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR. Donde destaca el enviado por el recuadro: “Duración Pulso de Salida”:

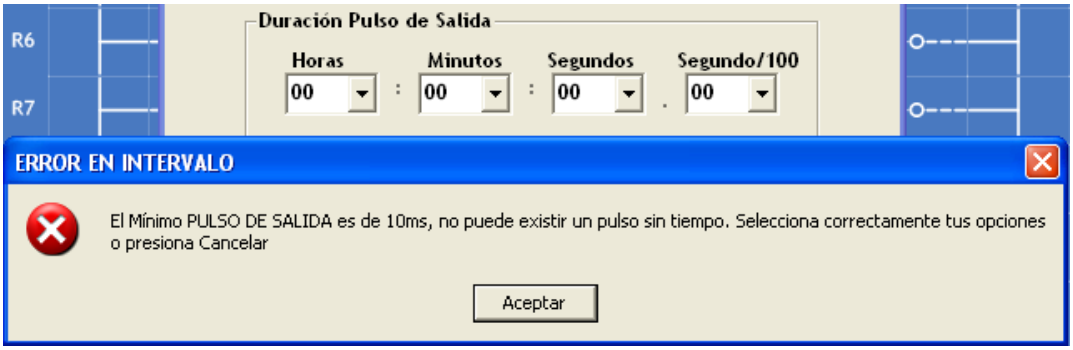


Figura A.143 – Ventana de Error al configurar un Pulso de Salida Sin Tiempo en el Temporizador Monodisparo Tipo 1.

El mensaje de ERROR mostrado en la figura A.143 indica que “El Mínimo PULSO DE SALIDA es de 10ms, no puede existir un pulso sin tiempo”. Esto sucede porque se seleccionaron en todos los campos del Pulso de Salida la opción “00”, no hay que olvidar que el mínimo Pulso de Salida requerido debe ser de al menos 10 milisegundos. El caso contrario lo muestra la siguiente figura A.144:

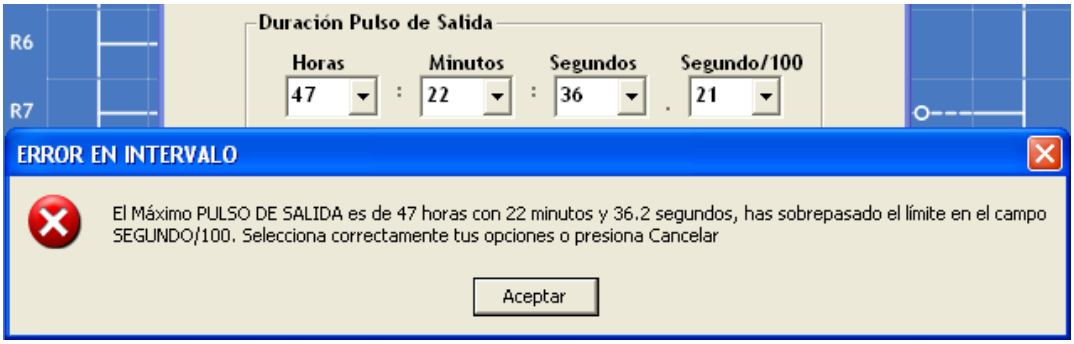


Figura A.144 – Ventana de Error al configurar un Pulso de Salida que sobrepasa el límite permitido en el Temporizador Monodisparo Tipo 1.

Hay que tener muy presente estos dos límites ya que al sobrepasar alguno de ellos se enviarán sendos mensajes de ERROR.

El único parámetro sobre el cual no se tiene control es el número del TEMPOA. Mostrado en la ventana como: "Número del TEMPOA: #1", que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el TEMPOA, la tarea siguiente es dar clic en el botón "Insertar". La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 3 (R3) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 3. Este módulo utilizará tres renglones de la ZONA DE TRABAJO como la que se muestra en la figura A.145:

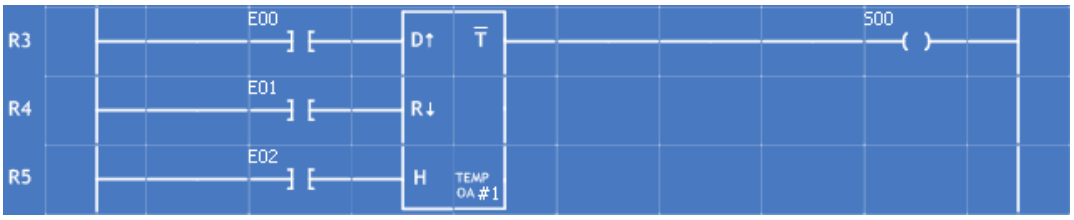


Figura A.145 – Temporizador Monodisparo Tipo 1 insertado en los renglones R3, R4 y R5 de la Zona de Trabajo del "PUMA ESCALERA".

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA "D": E00
ENTRADA "R": E01
ENTRADA "H": E02
SALIDA "T": S00
DURACIÓN PULSO DE SALIDA: 20:05:44.06
FLANCO: ENTRADA D: SUBIDA, que esta representado a través del primer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1

FLANCO: ENTRADA R: BAJADA, que esta representado a través del segundo dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
VERIFICACIÓN: ENTRADA H: ALTO, que esta representado a través del tercer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
VERIFICACIÓN: SALIDA T: BAJO, que esta representado a través del cuarto dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

TEMPOA#1 E00,E01,E02,S00,20:05:44.06,1010;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.16 Temporizador Monodisparo Tipo 2 (One Shot): TEMPOC

El PLM puede realizar dos tipos de temporizadores de tipo monodisparo, aquí se describe lo concerniente al temporizador monodisparo de tipo dos, mostrándose en la figura A.146

El intervalo de tiempo correspondiente lo especifica el usuario en la declaración sintáctica correspondiente, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos, como se aprecia en la figura A.146 el disparo puede ser por flanco de subida o bajada, teniéndose además otra entrada denominada “R” (RESET), que al verificarse coloca a este módulo en su condición de espera de disparo, con su salida no verificada. Al igual que el temporizador monodisparo de tipo uno, este temporizador tiene capacidad de redisparo.

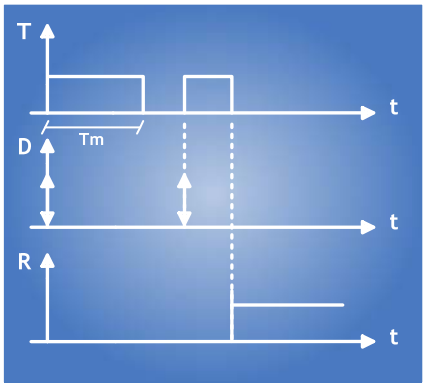


Figura A.146 - Diagrama de tiempos asociado (RESET verificado en alto) al Temporizador Monodisparo Tipo 2.

La entrada R responde al nivel, y al verificarse se desverifica la salida y se restablece a cero el contador de tiempo asociado.

Un TEMPORIZADOR MONODISPARO TIPO 2 (ONE SHOT): TEMPOC está representado por la figura A.147 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un TEMPOC):

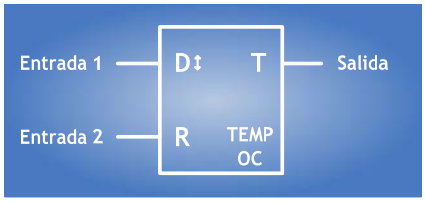


Figura A.147 - Temporizador Monodisparo Tipo 2.

La figura A.148 muestra la misma representación pero en diagrama de escalera.

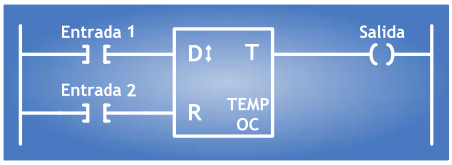


Figura A.148 Temporizador Monodisparo Tipo 2 en el Diagrama de Escalera.

Para incrustar un TEMPOC en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón "Temporizador OC", el cual se encuentra ubicado en el costado izquierdo de la ventana. Aparecerá el siguiente cuadro de diálogo:

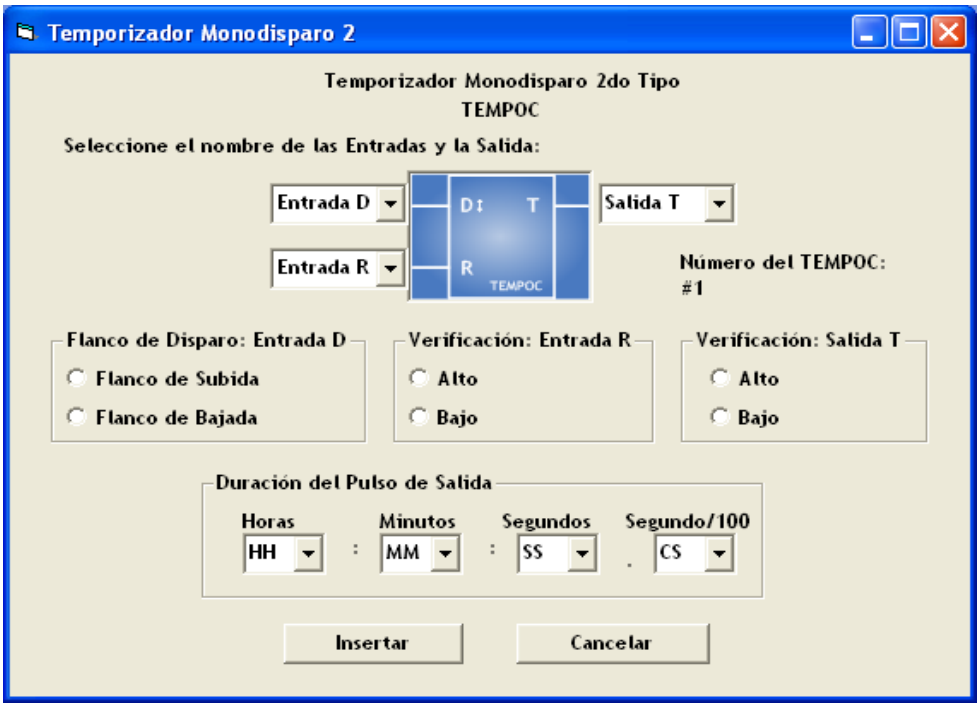


Figura A.149 - Ventana de Configuración del Temporizador Monodisparo Tipo 2.

La figura A.149 muestra la ventana que permitirá configurar los parámetros de un TEMPOC. Primeramente se eligen las VBE las ENTRADAS (Entrada D y Entrada R) y la SALIDA (Salida T) del módulo lógico. La sintaxis de las variables de entrada y salida ya fue anteriormente descrita.

Si la sintaxis de alguna de estas es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón "Insertar" de esta misma ventana. Estos envíos de ERROR ya fueron analizados y resueltos en la configuración del FFARS y el CONTADOR DE EVENTOS por si es necesario tomar alguna referencia de éstos.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Flanco de Disparo: Entrada D**

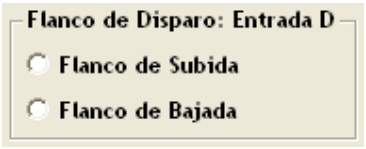


Figura A.150 – Caja para configurar el Flanco de Disparo de la Entrada "D" del Temporizador Monodisparo Tipo 2.

Si se desea que el temporizador se dispare para flancos de subida en la entrada de disparo "D" se deberá seleccionar la opción: Flanco de Subida, en otro caso deberá ser: Flanco de Bajada.

- **Verificación: Entrada R**

Figura A.151 – Caja para configurar el Tipo de Verificación de la Entrada "R" del Temporizador Monodisparo Tipo 2.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada "R" sea uno, en otro caso deberá ser Alto.

- **Verificación: Salida T**

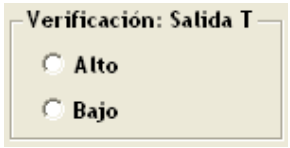


Figura A.152 – Caja para configurar el Tipo de Verificación de la Salida "T" del Temporizador Monodisparo Tipo 2.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Salida "T" sea cero, en otro caso deberá ser Alto.

- **Duración del Pulso de Salida**

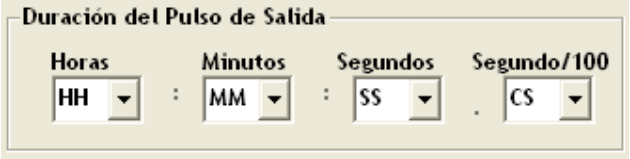


Figura A.153 – Caja para configurar la Duración del Pulso de Salida del Temporizador Monodisparo Tipo 2.

La duración del Pulso de Salida correspondiente lo especifica el usuario, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos.

- HH denota un par de dígitos que especifican el número de horas en el tiempo Tm, siendo 47 el valor máximo aceptado.
- MM denota un par de dígitos que especifican el número de minutos en Tm.
- SS denota un par de dígitos que especifican el número de segundos en Tm.
- CS denota un par de dígitos que especifican las centésimas de segundo en Tm.

Donde Tm es la duración del pulso especificada por el usuario.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR. Como en el TEMPOA, el mínimo PULSO DE SALIDA es de 10ms, así como el máximo permitido es 47 horas con 22 minutos y 36.2 segundos. Hay que tener muy presente estos dos límites ya que al sobrepasar alguno de ellos se enviarán sendos mensajes de ERROR.

El único parámetro sobre el cual no se tiene control es el número del TEMPOC. Mostrado en la ventana como: “Número del TEMPOC: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el TEMPOC, la tarea siguiente es dar clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 2 (R2) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 2. Este módulo utilizará dos renglones de la ZONA DE TRABAJO como la que se muestra en la figura A.154:

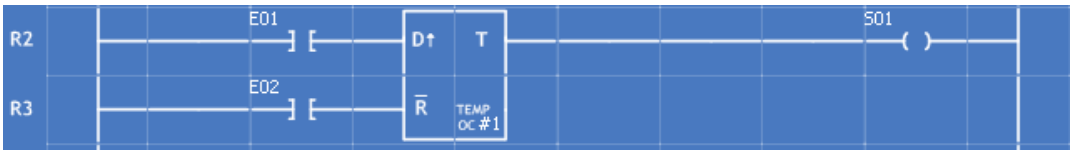


Figura A.154 – Temporizador Monodisparo Tipo 2 insertado en los renglones R2 y R3 de la Zona de Trabajo del “PUMA ESCALERA”.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA "D": E01
ENTRADA "R": E02
SALIDA "T": S01
DURACIÓN PULSO DE SALIDA: 30:03:05.04
FLANCO: ENTRADA D: SUBIDA, que esta representado a través del primer dígito (izquierda a derecha) de los 3 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
VERIFICACIÓN: ENTRADA R: BAJO, que esta representado a través del segundo dígito (izquierda a derecha) de los 3 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
VERIFICACIÓN: SALIDA T: ALTO, que esta representado a través del tercer dígito (izquierda a derecha) de los 3 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

TEMPOC#1 E01,E02,S01,30:03:05.04,111;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.17 Temporizador Con Retardo A La Activación (On-Delay) O Con Retardo A La Desactivación (Off-Delay): TEMPOD

El PLM puede realizar temporizadores con retardo a la activación (RA) o a la desactivación (RD), esto se logra a partir de un solo ML. En la figura A.157 se muestra la representación como bloque de este módulo. En las figuras A.155 y A.156 se muestran los diagramas de tiempo asociados.

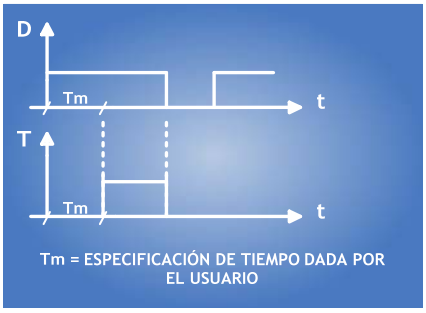


Figura A.155 - Respuesta con retardo a la activación (ON-DELAY)

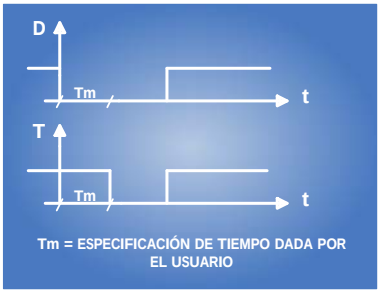


Figura A.156 - Respuesta con retardo a la desactivación (OFF-DELAY)

Al verificarse la entrada de restablecimiento (R) la salida pasa a su nivel no verificado (cero para on-delay, uno para off-delay) inicializándose el contador descendente asociado.

El intervalo de tiempo correspondiente lo especifica el usuario en la declaración sintáctica correspondiente, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos; la entrada R responde al nivel, y al verificarse se desverifica la salida y se inicializa el contador asociado.

Un TEMPORIZADOR MONODISPARO CON RETARDO A LA ACTIVACIÓN O CON RETARDO A LA DESACTIVACIÓN: TEMPOD está representado por la figura A.157 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un TEMPOD):

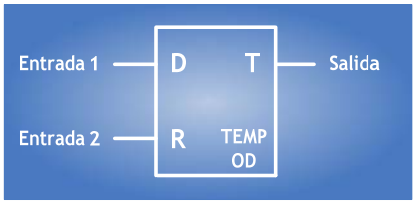


Figura A.157 – Temporizador Monodisparo con Retardo a la Activación o Desactivación.

La figura A.158 muestra la misma representación pero en diagrama de escalera.

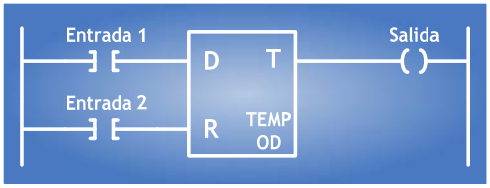


Figura A.158 – Temporizador Monodisparo con Retardo a la Activación o Desactivación en el Diagrama de Escalera.

Para incrustar un TEMPOD en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón “Temporizador OD”, el cual se encuentra ubicado en el costado izquierdo de la ventana. Aparecerá el siguiente cuadro de diálogo:

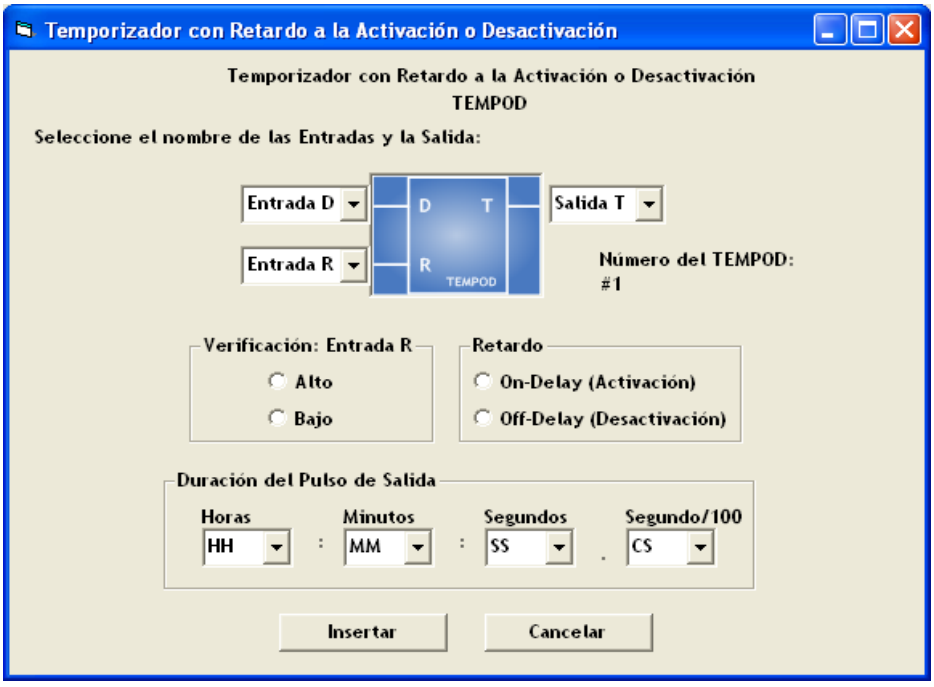


Figura A.159 – Ventana de Configuración del Temporizador Monodisparo con Retardo a la Activación o Desactivación.

La figura A.159 muestra la ventana que permitirá configurar los parámetros de un TEMPOD. Primeramente se eligen las VBE las ENTRADAS (Entrada D y Entrada R) y la SALIDA (Salida T) del módulo lógico. La sintaxis de las variables de entrada y salida ya fue anteriormente descrita.

Si la sintaxis de alguna de estas es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón “Insertar” de esta misma ventana. Estos envíos de ERROR ya fueron analizados y resueltos en la configuración del FFARS y el CONTADOR DE EVENTOS por si es necesario tomar alguna referencia de éstos.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Verificación: Entrada R**

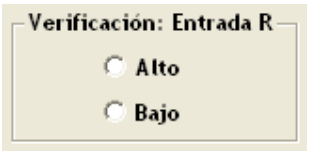


Figura A.160 – Caja para configurar el Tipo de Verificación de la Entrada “R” del Temporizador Monodisparo con Retardo a la Activación o Desactivación.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “R” sea uno, en otro caso deberá ser Alto.

- **Retardo**

Figura A.161 Caja para configurar el Retardo del Temporizador Monodisparo con Retardo a la Activación o Desactivación.

Si se desea que el temporizador presente retardo a la desactivación se tendrá que seleccionar la opción: Off-Delay, en otro caso deberá ser On Delay.

- **Duración del Pulso de Salida**

Figura A.162 – Caja para configurar la Duración del Pulso de Salida del Temporizador Monodisparo con Retardo a la Activación o Desactivación.

La duración del Pulso de Salida correspondiente lo especifica el usuario, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos.

- HH denota un par de dígitos que especifican el número de horas en el tiempo Tm, siendo 47 el valor máximo aceptado.
- MM denota un par de dígitos que especifican el número de minutos en Tm.
- SS denota un par de dígitos que especifican el número de segundos en Tm.
- CS denota un par de dígitos que especifican las centésimas de segundo en Tm.

Donde Tm es la duración del pulso especificada por el usuario.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR. Como en el TEMPOA y el TEMPOC, el mínimo PULSO DE SALIDA es de 10ms, así como el máximo permitido es 47 horas con 22 minutos y 36.2 segundos. Hay que tener muy presente estos dos límites ya que al sobrepasar alguno de ellos se enviarán sendos mensajes de ERROR.

El único parámetro sobre el cual no se tiene control es el número del TEMPOD. Mostrado en la ventana como: “Número del TEMPOD: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el TEMPOD, la tarea siguiente es dar clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 2 (R2) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 2. Este módulo utilizará dos renglones de la ZONA DE TRABAJO como la que se muestra en la figura A.163:

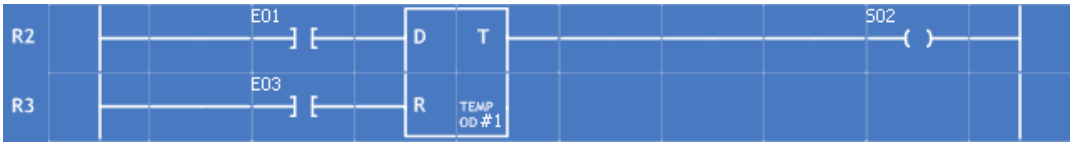


Figura A.163 – Temporizador Monodisparo con Retardo a la Activación o Desactivación insertado en los renglones R2 y R3 de la Zona de Trabajo del "PUMA ESCALERA".

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA "D": E01
ENTRADA "R": E03
SALIDA "T": S02
DURACIÓN PULSO DE SALIDA: 27:05:43.66
RETARDO: OFF-DELAY, que esta representado a través del primer dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
VERIFICACIÓN: ENTRADA R: ALTO, que esta representado a través del segundo dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

TEMPOD#1 E01,E03,S02,27:05:43.66,00;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.18 Temporizador Astable: TEMPOE

El PLM puede realizar temporizadores astables que generan señales cuadradas con ciclo de trabajo que puede ser fijado por el usuario. En la figura A.166 se muestra la representación como bloque de este módulo. En las figuras A.164 y A.165 se muestran los diagramas de tiempo asociados, el nivel de arranque puede ser uno o cero, esto definido por el interesado.

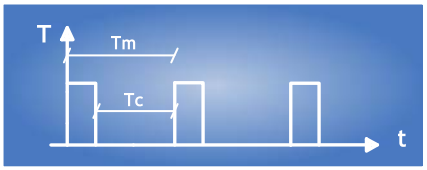


Figura A.164 - Temporizador Astable con arranque en uno.

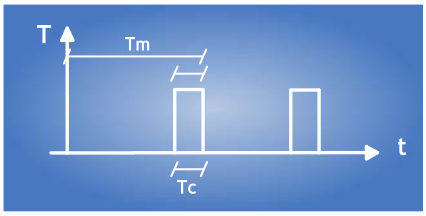


Figura A.165 - Temporizador Astable con arranque en cero.

Los tiempos Tc y Tm pueden estar comprendidos entre 10ms y 47 horas con 22 minutos y 36.2 segundos, debiendo siempre el tiempo Tc ser menor que el tiempo Tm; la entrada R responde al nivel, y al verificarse se coloca en la salida el nivel de arranque y se inicializa el contador asociado.

Un TEMPORIZADOR ASTABLE: TEMPOE está representado por la figura A.166 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un TEMPOE):

Figura A.166 – Temporizador Astable.

La figura A.167 muestra la misma representación pero en diagrama de escalera.

Figura A.167 – Temporizador Astable en el Diagrama de Escalera.

Para incrustar un TEMPOE en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón “Temporizador OE”, el cual se encuentra ubicado en el costado izquierdo de la ventana. Aparecerá el siguiente cuadro de diálogo:

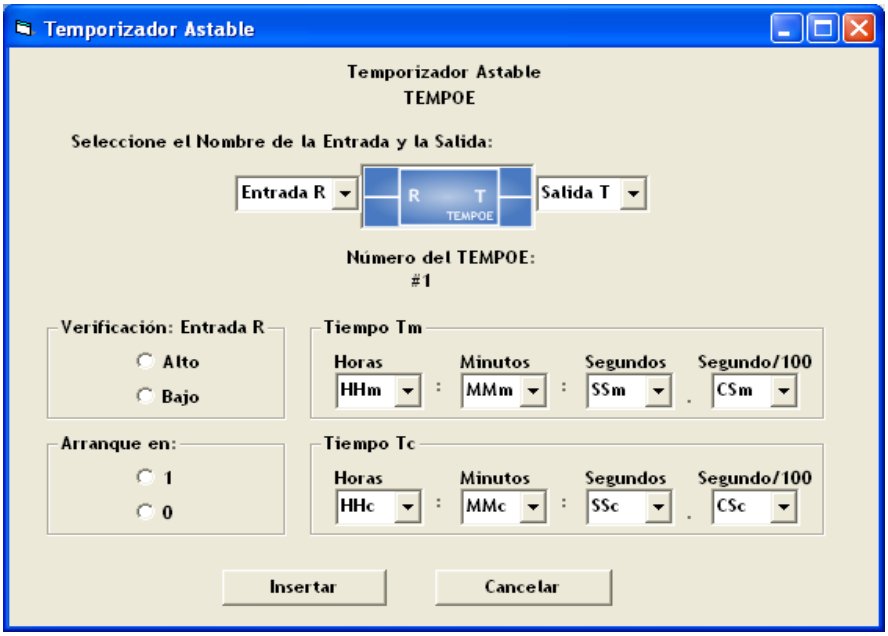


Figura A.168 – Ventana de Configuración para el Temporizador Astable.

La figura A.168 muestra la ventana que permitirá configurar los parámetros de un TEMPOE. Primeramente se eligen las VBE la ENTRADA (Entrada R) y la SALIDA (Salida T) del módulo lógico. La sintaxis de las variables de entrada y salida ya fue anteriormente descrita.

Si la sintaxis de alguna de estas es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón “Insertar” de esta misma ventana. Estos envíos de ERROR ya fueron analizados y resueltos en la configuración del FFARS y el CONTADOR DE EVENTOS por si es necesario tomar alguna referencia de éstos.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Verificación: Entrada R**

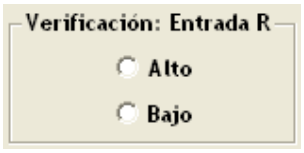


Figura A.169 – Caja para configurar el Tipo de Verificación de la Entrada “R” del Temporizador Astable.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “R” sea uno, en otro caso deberá ser Alto.

- **Arranque en:**

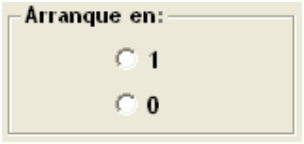


Figura A.170 – Caja para configurar el Tipo de Arranque del Temporizador Astable.

Habrá de ser cero si se desea que el temporizador arranque en cero, en otro caso deberá ser uno.

- **Tiempo Tm:**

Figura A.171 – Caja para configurar el Tiempo Tm del Temporizador Astable.

La duración del Tiempo Tm correspondiente lo especifica el usuario, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos.

- HHm denota un par de dígitos que especifican el número de horas en el tiempo Tm, siendo 47 el valor máximo aceptado.
- MMm denota un par de dígitos que especifican el número de minutos en Tm.
- SSm denota un par de dígitos que especifican el número de segundos en Tm.
- CSm denota un par de dígitos que especifican las centésimas de segundo en Tm.

Donde Tm es la duración del pulso especificada por el usuario.

- **Tiempo Tc:**

Figura A.172 – Caja para configurar el Tiempo Tc del Temporizador Astable.

La duración del Tiempo Tc correspondiente lo especifica el usuario, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos.

- HHc denota un par de dígitos que especifican el número de horas en el tiempo Tc, siendo 47 el valor máximo aceptado.
- MMc denota un par de dígitos que especifican el número de minutos en Tc.
- SSc denota un par de dígitos que especifican el número de segundos en Tc.
- CSc denota un par de dígitos que especifican las centésimas de segundo en Tc.

Donde Tc es la duración del pulso especificada por el usuario.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR.

El mínimo PULSO DE SALIDA es de 10ms, así como el máximo permitido es 47 horas con 22 minutos y 36.2 segundos. Estos límites deben ser respetados en ambos tiempos (Tm y Tc), debiendo siempre el Tiempo Tc ser menor que el Tiempo Tm, de lo contrario aparecerá el siguiente MENSAJE DE ERROR.

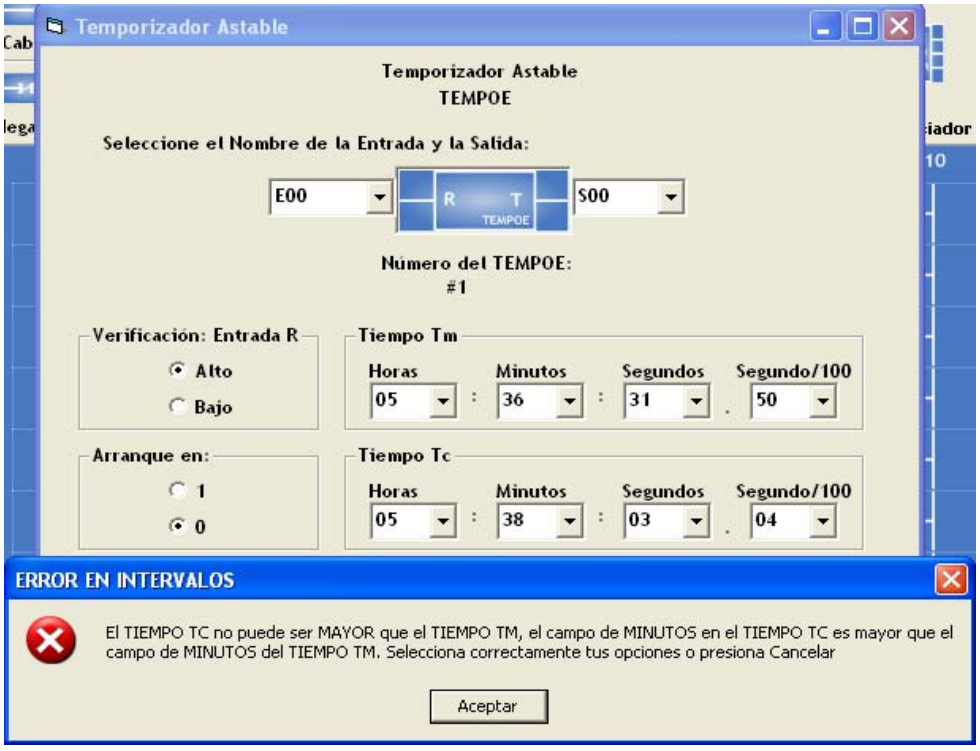


Figura A.173 – Ventana de Error al configurar los tiempos Tm y Tc en el Temporizador Astable.

El mensaje de ERROR mostrado en la figura A.173 indica que “El TIEMPO TC no puede ser mayor que el tiempo TM, el campo de MINUTOS en el TIEMPO TC es mayor que el campo de MINUTOS del TIEMPO TM”. Este tipo de mensajes enviarán ERRORES específicos, donde se señalarán los campos en los cuales se haya sobrepasado el límite permitido. El usuario deberá elegir un parámetro correcto en cada caso, si esto no acontece el envío de ERROR seguirá apareciendo y no se podrá configurar el módulo lógico y en consecuencia no se incrustará en la ZONA DE TRABAJO.

El único parámetro sobre el cual no se tiene control es el número del TEMPOE. Mostrado en la ventana como: “Número del TEMPOE: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el TEMPOE, la tarea siguiente es dar clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 1 (R1) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 1. Este módulo utilizará un renglón de la ZONA DE TRABAJO como la que se muestra en la figura A.174:

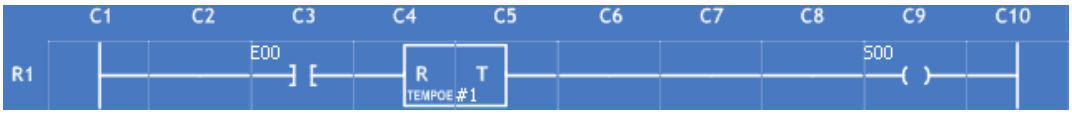


Figura A.174 – Temporizador Astable insertado en el renglón R1 de la Zona de Trabajo del “PUMA ESCALERA”.

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA “R”: E00
SALIDA “T”: S00
DURACIÓN TIEMPO Tm: 18:46:06.07
DURACIÓN TIEMPO Tc: 10:00:00.00
VERIFICACIÓN: ENTRADA R: ALTO, que esta representado a través del primer dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
ARRANQUE EN: 0, que esta representado a través del segundo dígito (izquierda a derecha) de los 2 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

TEMPOE#1 E00,S00,18:46:06.07,10:00:00.00,00;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.19 Temporizador Con Capacidad Para Generar N Pulsos A Intervalos De Tiempo Especificados Por El Usuario: TEMPOG

El PLM puede realizar temporizadores denominados de tipo *multipulso* que generan N pulsos de anchura fija, a intervalos de tiempo específicos, tanto el número N como los intervalos de tiempo implicados son definidos por el usuario, una vez que ha transcurrido el flanco que lleva a la verificación del último pulso la salida permanece en ese nivel, verificándose otra salida del módulo denominada “testigo de fin de carrera”, este módulo cuenta con dos entradas, una es para restablecimiento y la otra es una entrada denominada “entrada de congelamiento”, al verificarse esta última el estado de la salida de pulsos permanece invariable, en las figuras A.175 y A.176 se muestran

respectivamente la representación como bloque de este módulo y los diagramas de tiempo asociados, el nivel de verificación de los pulsos es definido por el interesado.

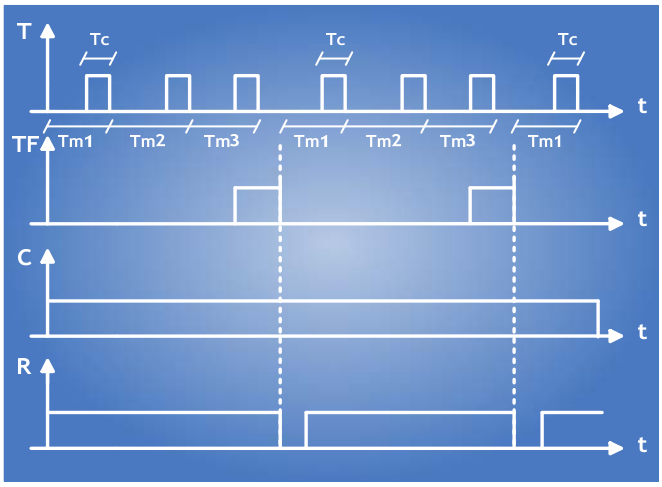


Figura A.175 – Diagrama de Tiempo Asociado

Para la figura A.175 el nivel de verificación de las dos entradas del módulo es bajo, el nivel de verificación, tanto de los pulsos como del testigo de fin de carrera es alto y el número de pulsos es tres.

Los tiempos Tc y Tmi (i = 1, 2, 3, ...N), pueden estar comprendidos entre 10ms y 47 horas con 22 minutos y 36.2 segundos, debiendo siempre el tiempo Tc ser menor que todos los tiempos Tmi; la entrada R responde al nivel, y al verificarse se coloca en la salida el nivel de arranque y se inicializa el contador asociado.

Un TEMPORIZADOR CON CAPACIDAD PARA GENERAR N PULSOS A INTERVALOS DE TIEMPO ESPECIFICADOS POR EL USUARIO: TEMPOG está representado por la figura A.176 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un TEMPOG):

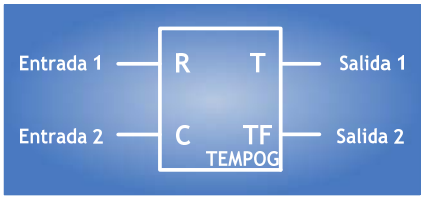


Figura A.176 – Temporizador Generador de N Pulsos.

La figura A.177 muestra la misma representación pero en diagrama de escalera.

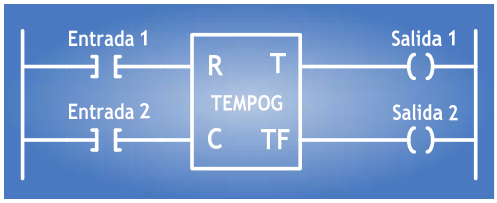


Figura A.177 – Temporizador Generador de N Pulsos en el Diagrama de Escalera.

Para incrustar un TEMPOG en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón "Temporizador OG", el cual se encuentra ubicado en la parte superior derecha de la ventana. Aparecerá el siguiente cuadro de diálogo:

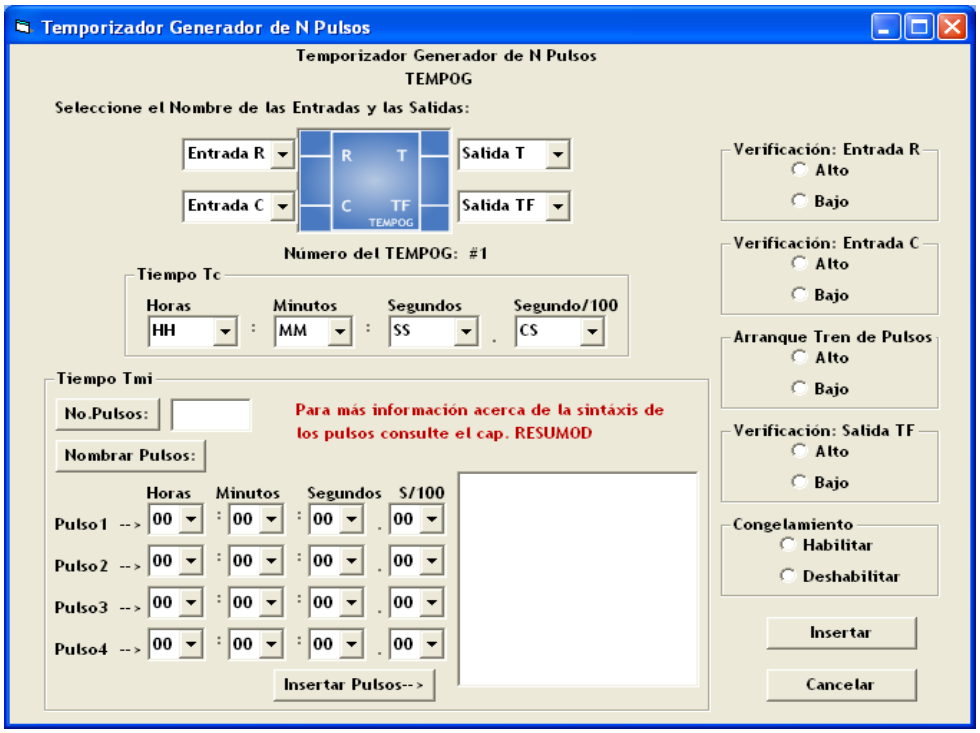


Figura A.178 – Ventana de Configuración para el Temporizador Generador de N Pulsos.

La figura A.178 muestra la ventana que permitirá configurar los parámetros de un TEMPOG. Primeramente se eligen las VBE las ENTRADAS (Entrada R y la entrada C) y las SALIDAS (Salida T y la Salida TF) del módulo lógico. La sintaxis de las variables de entrada y salida ya fue anteriormente descrita.

Si la sintaxis de alguna de estas es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón "Insertar" de esta misma ventana. Estos envíos de ERROR ya fueron analizados y resueltos en la configuración del FFARS y el CONTADOR DE EVENTOS por si es necesario tomar alguna referencia de éstos.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Verificación: Entrada R**

Figura A.179 – Caja para configurar el Tipo de Verificación de la Entrada "R" del Temporizador Generador de N Pulsos.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “R” sea uno, en otro caso deberá ser Alto.

- **Verificación: Entrada C**

Figura A.180 – Caja para configurar el Tipo de Verificación de la Entrada “C” del Temporizador Generador de N Pulsos.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada “C” sea cero, en otro caso deberá ser Alto.

- **Arranque Tren de Pulsos**

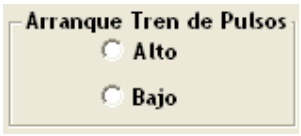


Figura A.181 – Caja para configurar el Arranque del Tren de Pulsos del Temporizador Generador de N Pulsos.

Si se desea que el arranque del tren de pulsos sea generado en bajo (pulsos verificados en alto) se seleccionará la opción Bajo, en otro caso deberá ser Alto.

- **Verificación: Salida TF**

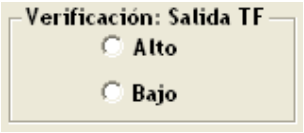


Figura A.182 – Caja para configurar el Tipo de Verificación de la Salida “TF” del Temporizador Generador de N Pulsos.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Salida “TF” sea cero, en otro caso deberá ser Alto.

- **Congelamiento**

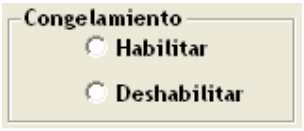


Figura A.183 – Caja para configurar el Congelamiento del Temporizador Generador de N Pulsos.

Si se desea Deshabilitar la entrada de Congelamiento se seleccionará la opción Deshabilitar, en otro caso deberá ser Habilitar.

Es importante señalar que aún y cuando la entrada de congelamiento se encuentre deshabilitada, deberá colocarse la especificación de una Variable Booleana, de no hacerse el programa traductor de SIIL1 a código ejecutable por el PLM indicará un ERROR de sintaxis.

- **Tiempo Tc:**

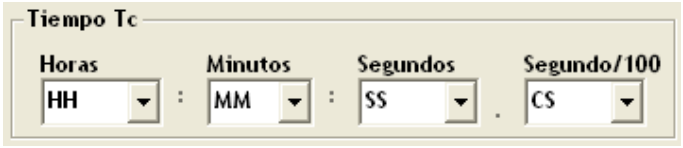


Figura A.184 – Caja para configurar el Tiempo Tc del Temporizador Generador de N Pulsos.

La duración del Tiempo Tc correspondiente lo especifica el usuario, pudiendo el mismo estar comprendido entre 10ms y 47 horas con 22 minutos y 36.2 segundos.

- HHc denota un par de dígitos que especifican el número de horas en el tiempo Tc, siendo 47 el valor máximo aceptado.
- MMc denota un par de dígitos que especifican el número de minutos en Tc.
- SSc denota un par de dígitos que especifican el número de segundos en Tc.
- CSc denota un par de dígitos que especifican las centésimas de segundo en Tc.

Donde Tc es la duración del pulso especificada por el usuario.

- **Tiempo Tmi:**

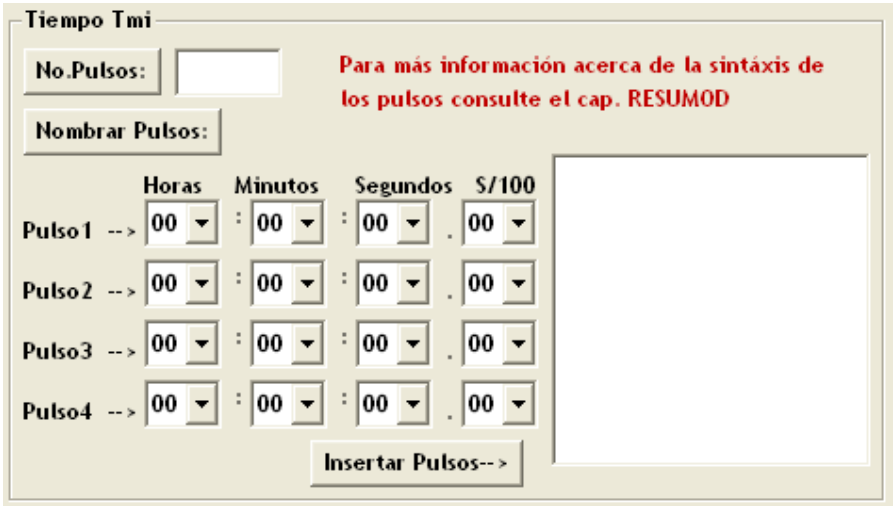


Figura A.185 – Caja para configurar el Tiempo Tmi, el Número y Nombre de los Pulsos del Temporizador Generador de N Pulsos.

Para poder nombrar los PULSOS de los tiempos Tmi hay que declarar primero el número de Pulsos. Para activarlo sólo hay que dar clic sobre el botón “No. Pulsos”. El software de traducción limita a 50 el número de pulsos asociados con un TEMPOG. Hecho lo anterior hay que activar el botón “Nombrar Pulsos” dando un clic sobre él. Las declaraciones de los valores deseados para los pulsos deben tener la siguiente estructura:

La duración de los Tiempos Tm correspondientes lo especifica el usuario, pudiendo los mismos estar comprendidos entre 10ms y 47 horas con 22 minutos y 36.2 segundos.

- Horas denota un par de dígitos que especifican el número de horas en los tiempos Tm, siendo 47 el valor máximo aceptado.
- Minutos denota un par de dígitos que especifican el número de minutos en los tiempos Tm.
- Segundos denota un par de dígitos que especifican el número de segundos en los tiempos Tm.
- S/100 denota un par de dígitos que especifican las centésimas de segundo en los tiempos Tm.

Donde Tm es la duración del pulso especificada por el usuario.

Los pulsos se nombrarán en grupos de cuatro y serán validados cuando se dé clic en el botón "Insertar Pulsos→" y estos aparezcan en el cuadro que aparece a la derecha en la figura A.19.10. Así por ejemplo, si el número de pulsos es siete se nombrarán los 4 primeros pulsos, a continuación se dará clic en "Insertar Pulsos→" y si tienen la correcta estructura aparecerán en el cuadro mencionado, enseguida los campos de texto quedarán vacíos para poder nombrar los tres pulsos restantes. Cuando se nombren estos pulsos aparecerán en el cuadro derecho enseguida de los cuatro pulsos anteriormente creados.

Importante: Los tiempos Tm no podrán ser configurados hasta que no se defina algún Tiempo Tc.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR.

El mínimo PULSO DE SALIDA es de 10ms, así como el máximo permitido es 47 horas con 22 minutos y 36.2 segundos. Estos límites deben ser respetados en ambos tiempos (Tm y Tc), debiendo siempre el Tiempo Tc ser menor que todos los Tiempos Tm, de lo contrario aparecerá el siguiente MENSAJE DE ERROR:

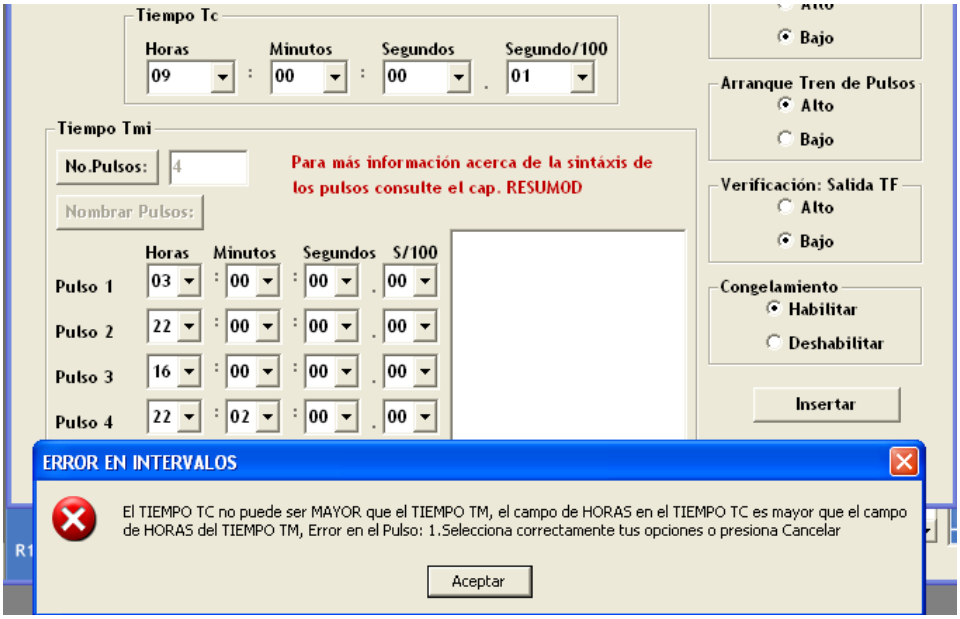


Figura A.186 – Ventana de Error al configurar los tiempos Tmi y Tc en el Temporizador Generador de N Pulsos.

El mensaje de ERROR mostrado en la figura A.186 indica que “El TIEMPO TC no puede ser mayor que el tiempo TM, el campo de HORAS en el TIEMPO TC es mayor que el campo de HORAS del TIEMPO TM, Error en el Pulso: 1”. Este tipo de mensajes enviarán ERRORES específicos, donde se señalarán los campos en los cuales se haya sobrepasado el límite permitido. El usuario deberá elegir un parámetro correcto en cada caso, si esto no acontece el envío de ERROR seguirá apareciendo y no se podrá configurar el módulo lógico y en consecuencia no se incrustará en la ZONA DE TRABAJO.

El único parámetro sobre el cual no se tiene control es el número del TEMPOG. Mostrado en la ventana como: “Número del TEMPOG: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el TEMPOG, la tarea siguiente es dar clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 1 (R1) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 1. Este módulo utilizará dos renglones de la ZONA DE TRABAJO como la que se muestra en la figura A.187:

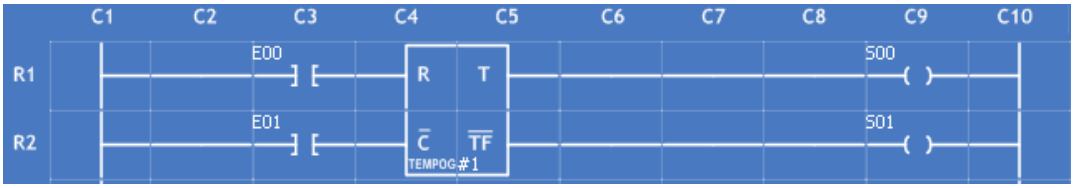


Figura A.187 – Temporizador Generador de N Pulsos insertado en los renglones R1 y R2 de la Zona de Trabajo del "PUMA ESCALERA".

Para este ejemplo se seleccionaron los siguientes parámetros:

ENTRADA "R": E00
ENTRADA "C": E01
SALIDA "T": S00
SALIDA "TF": S01
NÚMERO DE PULSOS A GENERAR: 4
DURACIÓN PULSO DEL TIEMPO Tc: 00:01:00.01
VERIFICACIÓN: ENTRADA R: ALTO, que esta representado a través del primer dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
ARRANQUE DEL TREN DE PULSOS: ALTO, que esta representado a través del segundo dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
VERIFICACIÓN: ENTRADA C: BAJO, que esta representado a través del tercer dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
VERIFICACIÓN: SALIDA TF: BAJO, que esta representado a través del cuarto dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
CONGELAMIENTO: HABILITAR, que esta representado a través del quinto dígito (izquierda a derecha) de los 5 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
PULSOS DEL TIEMPO Tm:
1ero.: 01:00:00.00
2do.: 02:00:00.00
3ero.: 03:00:00.00
4to.: 04:00:00.00

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

TEMPOG#1 E00,E01,S00,S01,4,00:01:00.01,01001;
 # 01:00:00.00, 02:00:00.00, 03:00:00.00;
04:00:00.00;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.20 Temporizador Con Capacidad De Generación De Pulsos En Instantes De Acuerdo Al Estado Del Reloj Del Tiempo Real (Gprtr): TEMPOB

El PLM puede realizar temporizadores denominados de tipo *GPRTR*, con capacidad para generar pulsos en instantes preestablecidos de acuerdo con el RTR, la duración de los mismos es aproximadamente de 100 ms, de acuerdo con la forma en que se especifiquen los instantes en que se desea se produzcan los pulsos los temporizadores GPRTR pueden ser de seis clases.

Un TEMPORIZADOR CON CAPACIDAD DE GENERACIÓN DE PULSOS EN INSTANTES DE ACUERDO AL ESTADO DEL RELOJ DEL TIEMPO REAL (GPRTR): TEMPOB está representado por la figura A.188 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un TEMPOB):

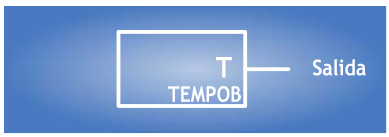


Figura A.188 – Temporizador Generador de Pulsos con Reloj en Tiempo Real.

La figura A.189 muestra la misma representación pero en diagrama de escalera.

Figura A.189 - Temporizador Generador de Pulsos con Reloj en Tiempo Real en el Diagrama de Escalera.

Para incrustar un TEMPOB en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón “Temporizador OB”, el cual se encuentra ubicado en el costado izquierdo de la ventana. Aparecerá el siguiente cuadro de diálogo:

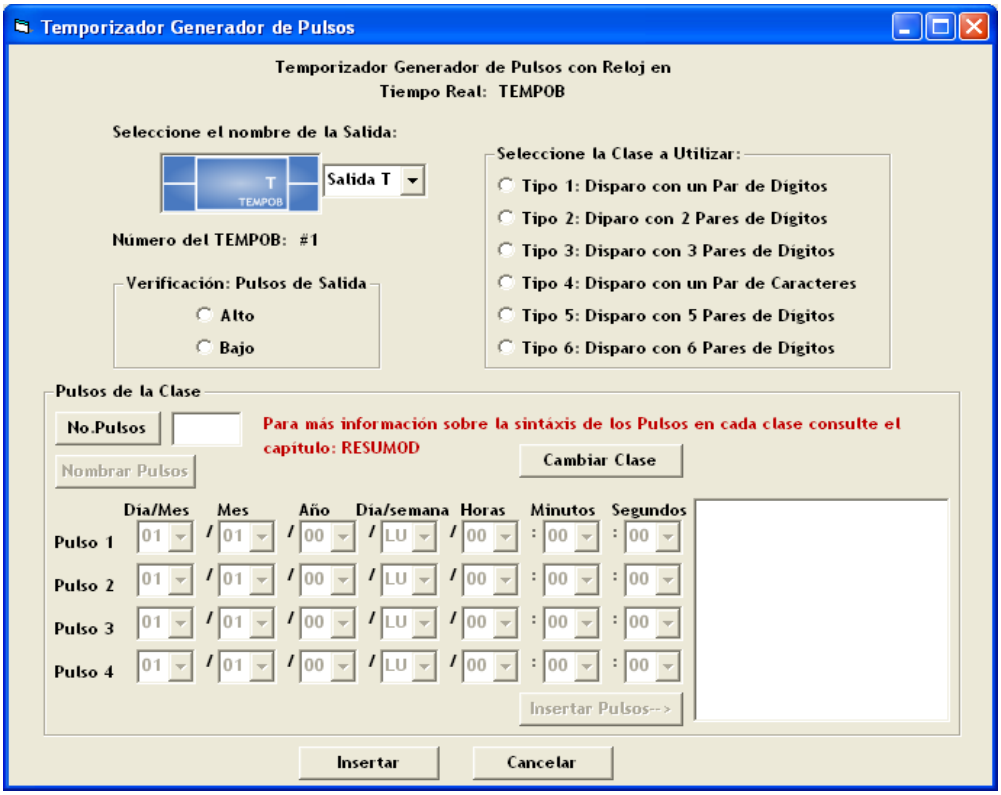


Figura A.190 – Ventana de Configuración para el Temporizador Generador de Pulsos con Reloj en Tiempo Real.

La figura A.190 muestra la ventana que permitirá configurar los parámetros de un TEMPOB. Primeramente se eligen las VBE la SALIDA (Salida T) del módulo lógico. La sintaxis de los nombres de las salidas ya fue anteriormente descrita.

Si la sintaxis de esta es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón “Insertar” de esta misma ventana. Estos envíos de ERROR ya fueron analizados y resueltos en la configuración del FFARS y el CONTADOR DE EVENTOS por si es necesario tomar alguna referencia de éstos.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Verificación: Pulsos de Salida**

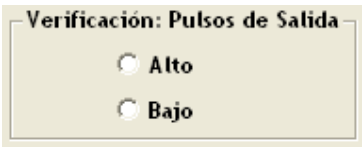


Figura A.191 – Caja para configurar el Tipo de Verificación de los Pulsos de Salida del Temporizador Generador de Pulsos con Reloj en Tiempo Real.

Si se desea que los pulsos de salida sean verificados en Bajo se seleccionará la opción Bajo, en otro caso deberá ser Alto.

- **Seleccione la Clase a Utilizar:**

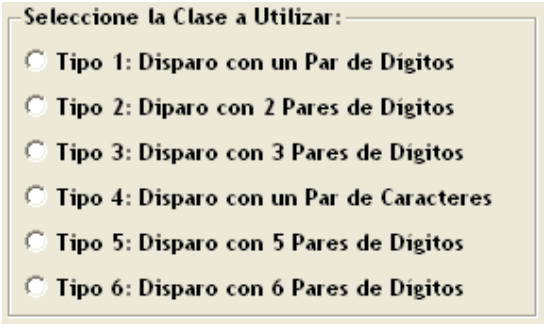


Figura A.192 – Caja para configurar el Tipo de Disparo de la clase a Utilizar del Temporizador Generador de Pulsos con Reloj en Tiempo Real.

Tipo 1: Temporizador GPRTR de clase uno, en este caso se especifican los instantes de disparo con un par de dígitos, que indican los segundos en los que se desea que aparezcan los pulsos; por ejemplo, 43 sería una especificación de disparo para el segundo cuarenta y tres de cada minuto.

Tipo 2: Temporizador GPRTR de clase dos, en este caso se especifican los instantes de disparo con dos pares de dígitos, que indican los minutos y segundos en los que se desea que aparezcan los pulsos; por ejemplo, 37:15 sería una especificación de disparo para el minuto treinta y siete con quince segundos de cada hora.

Tipo 3: Temporizador GPRTR de clase tres, en este caso se especifican los instantes de disparo con tres pares de dígitos, que indican en que hora, minuto y segundo se desea que aparezcan los pulsos; por ejemplo, 13:23:10 sería una especificación de disparo para las trece horas con veintitrés minutos y diez segundos.

Tipo 4: Temporizador GPRTR de clase cuatro, en este caso se especifican los instantes de disparo con un par de caracteres que indican un día de la semana seguidos por tres pares de dígitos, de esta forma cada especificación indica en que hora minuto y segundo y día de la semana se desea que aparezcan los pulsos; por ejemplo, VI/00:12:08 sería una especificación de disparo para las cero horas con doce minutos y ocho segundos de un día viernes.

El par de caracteres empleados para denotar a cada uno de los días de la semana es: “LU” para el lunes, “MA” para el martes, “MI” para el miércoles, “JU” para el jueves, “VI” para el viernes, “SA” para el sábado y “DO” para el domingo.

Tipo 5: Temporizador GPRTR de clase cinco, en este caso se especifican los instantes de disparo con cinco pares de dígitos, que indican en que número de día, mes, hora, minuto y segundo se desea que aparezcan los pulsos; por ejemplo, 08/05/12:30:02 sería una especificación de disparo para las doce horas con treinta minutos y dos segundos de un día ocho de mayo.

Tipo 6: Temporizador GPRTR de clase seis, en este caso se especifican los instantes de disparo con seis pares de dígitos, que indican en que día de mes, mes, año, hora, minuto y segundo se desea que aparezcan los pulsos; por ejemplo,

03/31/02/15:12:13 sería una especificación de disparo para las quince horas con doce minutos y trece segundos del treinta y uno de marzo del año dos mil dos

- **Pulsos de la Clase**

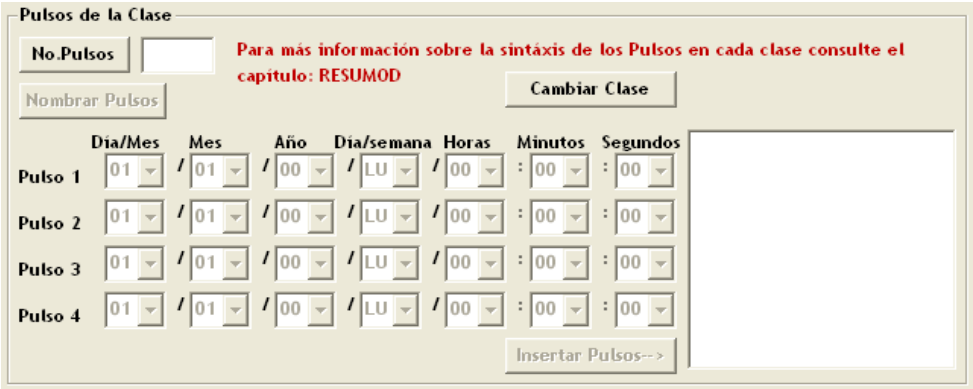


Figura A.193 – Caja para configurar el Número y Nombre de los Pulsos del Temporizador Generador de Pulsos con Reloj en Tiempo Real.

Para poder nombrar los PULSOS de la clase seleccionada hay que declarar primero el número de Pulsos. Para activarlo sólo hay que dar clic sobre el botón "No. Pulsos". El software de traducción limita a 50 el número de pulsos asociados con un TEMPOB. Hecho lo anterior hay que activar el botón "Nombrar Pulsos" dando un clic sobre él. Las declaraciones de los valores deseados para los pulsos se ajustarán según la Clase escogida en el punto anterior:

- Tipo 1: Se activará el campo: Segundos.
- Tipo 2: Se activarán los campos: Minutos y Segundos.
- Tipo 3: Se activarán los campos: Horas, Minutos y Segundos.
- Tipo 4: Se activarán los campos: Día/Semana, Horas, Minutos y Segundos.
- Tipo 5: Se activarán los campos: Día/Mes, Mes, Horas, Minutos y Segundos.
- Tipo 6: Se activarán los campos: Día/Mes, Mes, Año, Horas, Minutos y Segundos.

Los pulsos se nombrarán en grupos de cuatro y serán validados cuando se dé clic en el botón "Insertar Pulsos→" y estos aparezcan en el cuadro que aparece a la derecha en la figura A.20.6. Así por ejemplo, si el número de pulsos es siete se nombrarán los 4 primeros pulsos, a continuación se dará clic en "Insertar Pulsos→" y si tienen la correcta estructura aparecerán en el cuadro mencionado, enseguida los campos de texto quedarán vacíos para poder nombrar los tres pulsos restantes. Cuando se nombren estos pulsos aparecerán en el cuadro derecho enseguida de los cuatro pulsos anteriormente creados.

Importante: Los Pulsos de la Clase no podrán ser configurados hasta que no se defina algún Tipo de Clase.

El Botón "Cambiar Clase" tiene la finalidad de cambiar la clase seleccionada por una nueva. Esto implica que se borrarán todos los pulsos ya configurados de la clase sustituida.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR. Como se muestran a continuación:

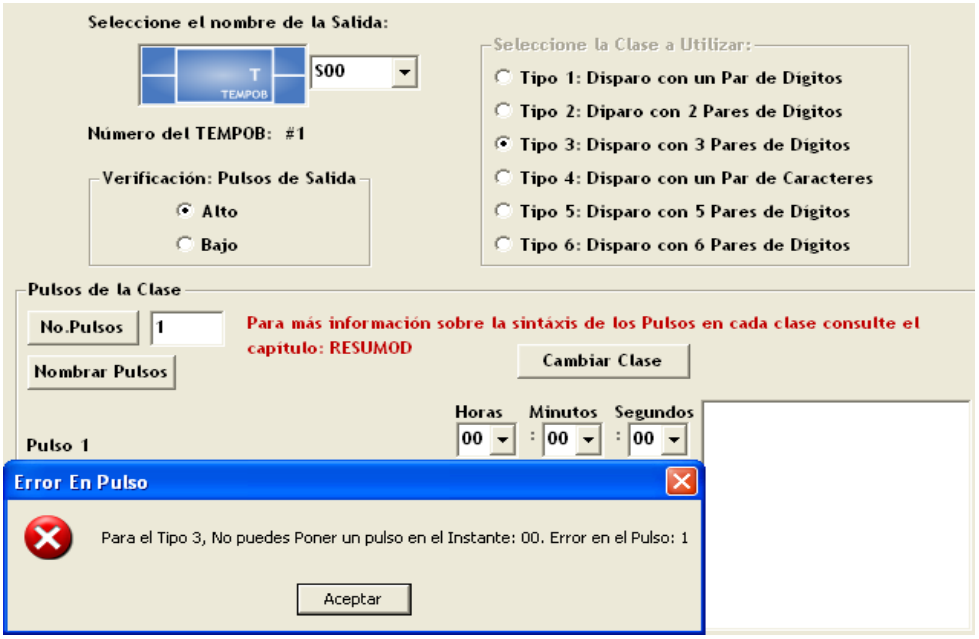


Figura A.194 – Ventana de Error al configurar la Clase Tipo 3 en los Pulsos de la Clase en el Temporizador Generador de Pulsos con Reloj en Tiempo Real.

El mensaje de ERROR mostrado en la figura A.194 indica que “Para el Tipo 3, No puedes Poner un pulso en el instante: 00. Error en el Pulso: 1”. Este tipo de mensajes enviarán ERRORES específicos, donde se señalarán los campos en los cuales se haya definido alguna incongruencia. Este Mensaje de ERROR se enviará para las Clases 1, 2 y 3 ya que es incoherente especificar un instante: 00.

Otro mensaje de ERROR que se puede presentar frecuentemente es el siguiente:

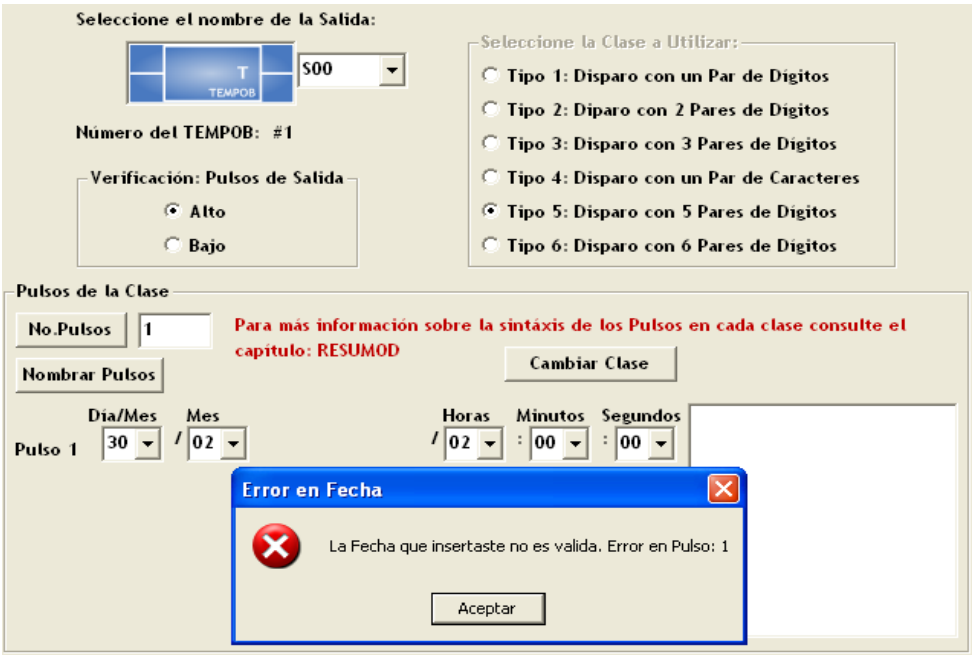


Figura A.195 – Ventana de Error al configurar la Clase Tipo 5 en los Pulsos de la Clase en el Temporizador Generador de Pulsos con Reloj en Tiempo Real.

El mensaje de ERROR mostrado en la figura A.195 indica que “La Fecha que insertaste no es válida. Error en el Pulso: 1”. Este tipo de mensajes enviarán ERRORES específicos, donde se señalarán los campos en los cuales se haya definido alguna incongruencia. Para este caso se seleccionó el día 30 de febrero, el cual no existe. También hay que tener precaución al seleccionar la CLASE 6 y escoger una fecha que contenga un AÑO BISIESTO ya que febrero tiene 29 días, por lo que aparecerá un mensaje de ERROR si la fecha no concuerda con alguno de estos años.

El usuario deberá elegir un parámetro correcto en cada caso, si esto no acontece el envío de ERROR seguirá apareciendo y no se podrá configurar el módulo lógico y en consecuencia no se incrustará en la ZONA DE TRABAJO.

El único parámetro sobre el cual no se tiene control es el número del TEMPOB. Mostrado en la ventana como: “Número del TEMPOB: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el TEMPOB, la tarea siguiente es dar clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 1 (R1) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 1. Este módulo utilizará un renglón de la ZONA DE TRABAJO como la que se muestra en la figura A.196:

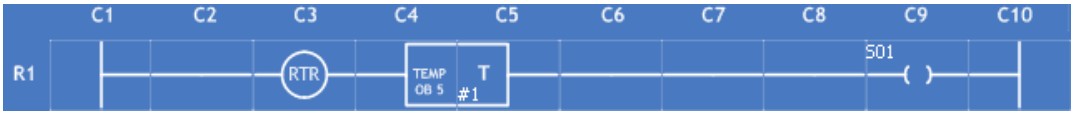


Figura A.196 – Temporizador Generador de Pulsos con Reloj en Tiempo Real insertado en el renglón R1 de la Zona de Trabajo del "PUMA ESCALERA".

Para este ejemplo se seleccionaron los siguientes parámetros:

SALIDA "T": S01
CLASE: 5
NÚMERO DE PULSOS A GENERAR: 4
VERIFICACIÓN: PULSOS DE SALIDA: ALTO, que esta representado a través del primer dígito (izquierda a derecha) del último caracter de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
PULSOS DE LA CLASE 5:
1ero.: 01/02/01:18:03
2do.: 05/07/04:05:00
3ero.: 14/08/03:27:32
4to.: 28/02/04:00:00

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

TEMPOB#1 S01,5,4,1;
 # 01/02/01:18:03, 05/07/04:05:00, 14/08/03:27:32;
28/02/04:00:00;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.21 Módulo Auxiliar Con Capacidad Para Generar Texto Estático Y/O Dinámico En La Ud Del PLM: Mensajero

El PLM puede realizar módulos denominados de tipo *MENSAJERO*, con capacidad para generar texto, predefinido por el usuario, dicho texto podría ser el alusivo a una condición de alarma, o bien cualquier otro tipo de mensaje que el usuario deseara desplegar, en la figura A.198 se muestra la pantalla de la UD(Unidad Desplegadora), ahí se aprecia que la misma puede desplegar dos renglones de dieciséis caracteres cada uno; la posición de cada caracter se especifica por un par de números que denotan en que renglón y columna se encuentra el mismo.

Al texto a desplegar se le denomina mensaje, teniendo este módulo la capacidad para desplegar mensajes fijos y/o mensajes móviles, este último será visible en una ventana delimitada por dos columnas, cuyo número es definido por el usuario y se desplazará entrando por la columna que define el extremo derecho de la ventana, saliendo por el extremo izquierdo de la misma; el usuario puede especificar en la declaración correspondiente, en cual renglón de la UD se desplegarán cada uno de los dos tipos de

mensaje aquí mencionados, el tamaño en caracteres del mensaje móvil es definido por el usuario, a cada uno de los mensajeros involucrados en una determinada aplicación se le ha de asignar un número, el cual es limitado a 32 por el software de traducción.

Dado el reducido tamaño de la pantalla de la UD, este módulo está diseñado de modo que en una aplicación que involucre más de un mensajero, sólo esté activo uno a la vez, por defecto el mensajero que es activo al iniciar la ejecución de un programa en el PLM, es el número cero, existen dos módulos con capacidad para cambiar el número de mensajero activo y estos son el módulo MANDESP y el módulo ALARMA.

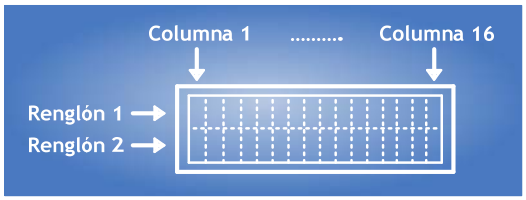


Figura A.198 – Estructura del Mensajero Generador de Texto estático y/o dinámico.

Un MÓDULO AUXILIAR CON CAPACIDAD PARA GENERAR TEXTO ESTÁTICO Y/O DINÁMICO EN LA UD DEL PLM: MENSAJERO está representado por la figura A.199 (se da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un MENSAJERO):

Figura A.199 – Mensajero Generador de Texto estático y/o dinámico.

La figura A.200 muestra la misma representación pero en diagrama de escalera.

Figura A.200 – Mensajero Generador de Texto estático y/o dinámico en el Diagrama de Escalera.

Para incrustar un MENSAJERO en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón “Mensajero”, el cual se encuentra ubicado en la parte superior derecha de la ventana. Aparecerá el siguiente cuadro de diálogo:

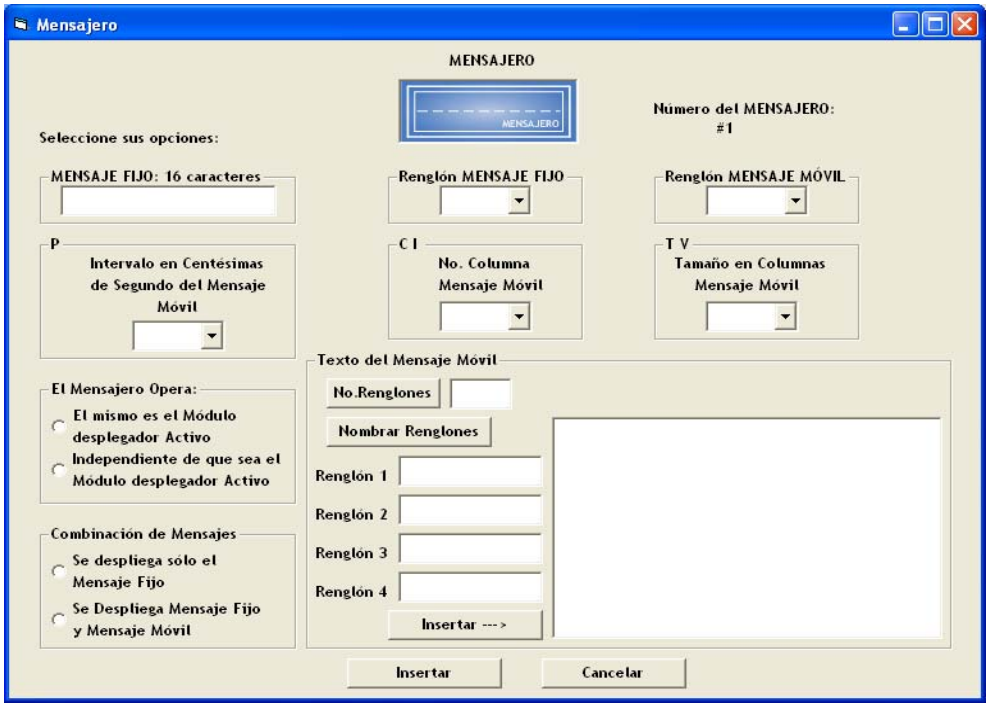


Figura A.201 – Ventana de Configuración del Mensajero Generador de Texto estático y/o dinámico.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **MENSAJE FIJO: 16 caracteres**

Figura A.202 – Caja para configurar el Mensaje Fijo.

MENSAJE FIJO es una cadena de un máximo de dieciséis caracteres, que es colocada a partir de la columna uno del renglón que el usuario especifique, si el usuario deseara que no haya mensaje fijo, denotaría esto con dejando el campo vacío.

- **Renglón MENSAJE FIJO**

Figura A.203 – Caja para configurar el Renglón del Mensaje Fijo.

Habrá de ser uno, si se desea que el mensaje fijo se despliegue en el renglón uno, en otro caso deberá ser dos para ser mostrado en el renglón dos.

- **Renglón MENSAJE MÓVIL**

Figura A.204 – Caja para configurar el Renglón del Mensaje Móvil.

Habrá de ser uno, si se desea que el mensaje móvil se despliegue en el renglón uno, en otro caso deberá ser dos para ser mostrado en el renglón dos.

- **P**

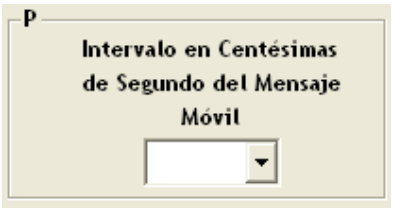


Figura A.205 – Caja para configurar el Intervalo del Mensaje Móvil.

P es un número comprendido entre 1 y 255, que representa el intervalo en centésimas de segundo entre dos posiciones subsecuentes del mensaje móvil.

- **C I**

Figura A.206 – Caja para configurar el No. de Columna del Mensaje Móvil.

CI denota el número de columna, que delimita el extremo izquierdo de la ventana donde se desplegará el mensaje móvil.

- **T V**

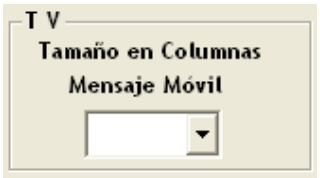


Figura A.207 – Caja para configurar el Tamaño en Columnas del Mensaje Móvil.

TV representa el tamaño en columnas del mensaje móvil.

- **El Mensajero Opera:**

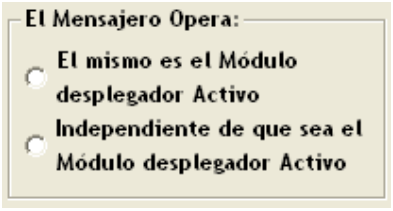


Figura A.208 – Caja para configurar la Operación del Mensajero.

Si se desea que el mensajero opere sólo si el mismo es el módulo desplegador activo seleccionar la primera opción. Si el mensajero opera independientemente del hecho de que el mismo sea el módulo desplegador activo seleccionar la segunda opción, desde luego que el usuario sería responsable de que en el último caso no se produjera una colisión entre dos mensajeros, que intentarán escribir texto en una misma zona de la pantalla de la UD.

- **Combinación de Mensajes**

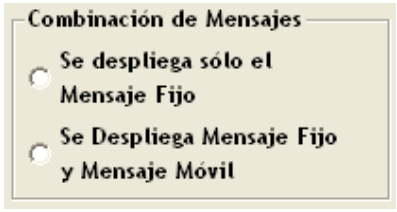


Figura A.209 – Caja para configurar la Combinación de Mensajes en el Mensajero.

Si se desea que únicamente se despliegue el mensaje fijo seleccionar la primera opción. Si se selecciona la segunda opción la operación será normal, desplegándose tanto el mensaje fijo como el móvil.

- **Texto del Mensaje Móvil**

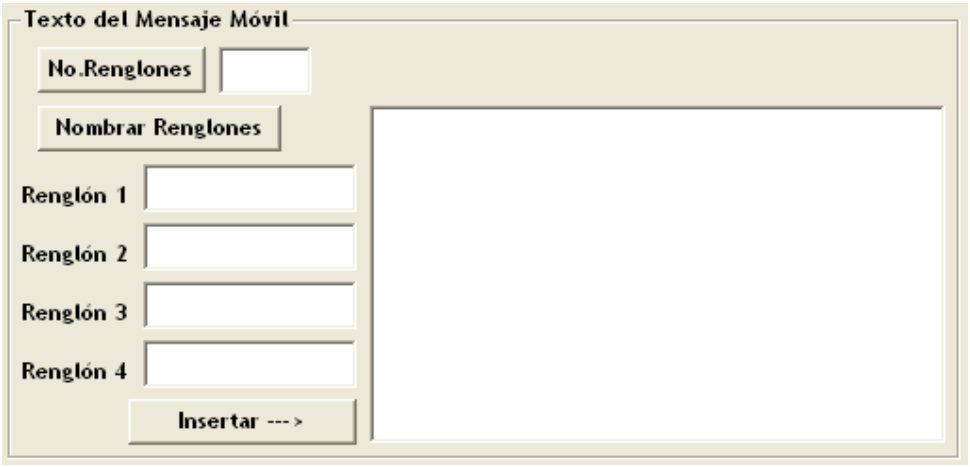


Figura A.210 Caja para configurar el Texto del Mensaje Móvil.

Para poder nombrar los RENGLONES hay que declarar primero el número de Renglones a utilizar. Para activarlo sólo hay que dar clic sobre el botón “No. Renglones”. El software de traducción limita a 100 el número de renglones asociados con un MENSAJERO. Hecho lo anterior hay que activar el botón “Nombrar Renglones” dando un clic sobre él. Las declaraciones de los renglones se ajustarán según el usuario, estando limitado a 100 caracteres por renglón.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR.

El único parámetro sobre el cual no se tiene control es el número del MENSAJERO. Mostrado en la ventana como: “Número del MENSAJERO: #1”, que es el número del módulo lógico en turno que se está insertando. Para este caso fue #1 porque es el primer módulo que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente el MENSAJERO, la tarea siguiente es dar clic en el botón “Insertar”. La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 2 (R2) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 2. Este módulo utilizará un renglón de la ZONA DE TRABAJO como la que se muestra en la figura A.211:

Figura A.211 – Mensajero Generador de Texto estático y/o dinámico insertado en el renglón R2.

Para este ejemplo se seleccionaron los siguientes parámetros:

MENSAJE FIJO: HOLA
CI (Número de Columna): 4
TV (Tamaño en Columnas): 6
P (Intervalo): 124
RENGLÓN Mensaje Fijo: 1, que esta representado a través del primer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
RENGLÓN Mensaje Móvil: 2, que esta representado a través del segundo dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
El Mensaje Opera: EL MISMO ES EL MÓDULO DESPLEGADOR ACTIVO, que esta representado a través del tercer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
Combinación de Mensajes: SE DESPLIEGA TANTO EL MENSAJE FIJO COMO EL MÓVIL, que esta representado a través del cuarto dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
MENSAJE MÓVIL: Esta es una prueba para determinar si funciona la traducción

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

MENSAJERO#1 "HOLA",4,6,124,1001;
 # Esta es una prueba para determinar si|
funciona la traducción

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.22 Módulo Auxiliar Con Capacidad Para Generar Mensajes De Alarma En La Ud Del PLM: Alarma

El PLM puede realizar módulos denominados de tipo *ALARMA*, con capacidad para generar mensajes en la Unidad Desplegadora (UD), este módulo cuenta con una sola entrada, que al verificarse, hace que el texto asociado con un determinado módulo mensajero sea desplegado en la UD. En la figura A.213 muestra la representación como bloque de este tipo de módulo.

Cada módulo de alarma tendrá asociado el número de módulo mensajero, que contiene el texto alusivo al la misma, el orden de colocación de la declaración correspondiente en el programa fuente será el orden de prioridad del mensaje de alarma asociado; de esta manera, el primer módulo de alarma colocado tendrá máxima prioridad, mientras que el último tendrá la prioridad mínima, esto quiere decir que si se verifica más de una entrada de alarma en sendos módulos de este tipo, se desplegará en la UD únicamente el mensaje asociado con el módulo de mayor prioridad, cuando la entrada de alarma asociada con este último módulo se desverifica se despliega el mensaje asociado con el módulo de alarma de mayor prioridad cuya entrada permanezca verificada.

Cuando no están verificadas las entradas asociadas, con cada uno de los módulos de alarma declarados en un programa en SIIL1, el texto que se despliega es el correspondiente con el módulo mensajero número cero; así, el texto que indique que no se ha dado ninguna condición de alarma, habrá de ser el asociado con un módulo mensajero, al que se deberá asignar el número cero; de este modo, al diseñar un sistema de mensajes de alarma apoyándose en los módulos mensajeros y de alarma, siempre deberá existir un módulo mensajero con el número cero.

Un MÓDULO AUXILIAR CON CAPACIDAD PARA GENERAR MENSAJES DE ALARMA EN LA UD DEL PLM: ALARMA está representado por la figura A.1212 da por un hecho que el usuario tiene previos conocimientos y habilidades para conocer un ALARMA):

Figura A.212 – Alarma Generadora de Mensajes.

La figura A.1 muestra la misma representación pero en diagrama de escalera.

Figura A.213 – Alarma Generadora de Mensajes en el Diagrama de Escalera.

Para incrustar una ALARMA en la ZONA DE TRABAJO habrá que situarse en la interfaz principal y se deberá dar clic en el botón “Alarma”, el cual se encuentra ubicado en la parte superior derecha de la ventana. Aparecerá el siguiente cuadro de diálogo:

Figura A.214 – Ventana de Configuración para la Alarma Generadora de Mensajes.

La figura A.214 muestra la ventana que permitirá configurar los parámetros de una ALARMA. Primeramente se eligen las VBE la ENTRADA (Entrada A) del módulo lógico. La sintaxis de los nombres de las entradas ya fue anteriormente descrita.

Si la sintaxis de esta es errónea aparecerá un mensaje de ERROR cuando se dé clic en el botón “Insertar” de esta misma ventana. Estos envíos de ERROR ya fueron analizados y resueltos en la configuración del FFARS y el CONTADOR DE EVENTOS por si es necesario tomar alguna referencia de éstos.

El siguiente paso en la configuración es seleccionar las opciones que se encuentran en cada recuadro de clasificación. La descripción de cada recuadro es la siguiente:

- **Mensajero Asociado**

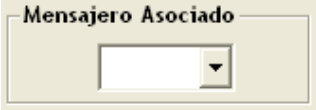


Figura A.215 – Caja para configurar el Mensajero Asociado a la Alarma.

Denota el número de módulo de tipo alarma, que debe coincidir con el número de módulo mensajero asociado con el mensaje a desplegar, cuando se verifique la variable definida como entrada al módulo de alarma.

- **Verificación: Entrada A**

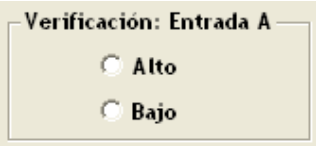


Figura A.216 – Caja para configurar el Tipo de Verificación de la Entrada "A" de la Alarma Generadora de Mensajes.

Habrá de ser Bajo, si se desea que el nivel de verificación de la Entrada "A" sea cero, en otro caso deberá ser Alto.

Si algún parámetro no se selecciona en cada uno de éstos recuadros aparecerá el correspondiente mensaje de ERROR. Como se muestran a continuación:

Figura A.217 - Ventana de Error al configurar el número de Mensajero Asociado en la Alarma Generadora de Mensajes.

El mensaje de ERROR mostrado en la figura A.217 indica que "No has seleccionado el número del mensajero asociado o aún no existe un mensajero al que le puedas asociar la alarma". Esto significa que no se ha seleccionado algún número que identifique a la alarma y a su vez se asocie a un módulo MENSAJERO o quizá se quiere configurar un

módulo ALARMA sin que previamente se haya configurado algún módulo mensajero, esto no es posible. El usuario deberá elegir un parámetro correcto en cada caso, si esto no acontece el envío de ERROR seguirá apareciendo y no se podrá configurar el módulo lógico y en consecuencia no se incrustará en la ZONA DE TRABAJO.

Este es el único módulo sobre el que se tiene control del número de identificación. Para este caso fue #1 porque es el primer módulo auxiliar que se incrusta en la ZONA DE TRABAJO. Este número permite la identificación del módulo cuando se realice la traducción a LENGUAJE SIIL1.

Una vez que se haya configurado correctamente la ALARMA, la tarea siguiente es dar clic en el botón "Insertar". La ventana de configuración desaparecerá y será posible incrustar el módulo lógico en la ZONA DE TRABAJO.

Se utilizará el renglón 2 (R2) para insertar este módulo lógico, para esto se deberá dar clic en cualquier COLUMNA del RENGLÓN 2. Este módulo utilizará un renglón de la ZONA DE TRABAJO como la que se muestra en la figura A.218:

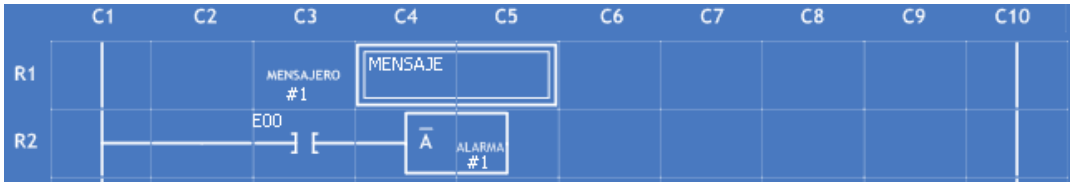


Figura A.218 – Alarma Generadora de Mensajes junto con el Mensajero Asociado en el renglón R1.

La Figura A.218 no sólo muestra el módulo ALARMA, sino que también se insertó un módulo MENSAJERO para poder ejemplificar claramente su uso. El #1 es el número asociado que tendrá la alarma con el mensajero, que a su vez activará el mensaje asociado correspondiente.

Para este ejemplo se seleccionaron los siguientes parámetros:

Para el módulo MENSAJERO:

MENSAJE FIJO: MENSAJE
CI (Número de Columna): 1
TV (Tamaño en Columnas): 14
P (Intervalo): 150
RENGLÓN Mensaje Fijo: 1, que esta representado a través del primer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
RENGLÓN Mensaje Móvil: 2, que esta representado a través del segundo dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
El Mensaje Opera: EL MISMO ES EL MÓDULO DESPLEGADOR ACTIVO, que esta representado a través del tercer dígito (izquierda a derecha) de los 4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0
Combinación de Mensajes: SE DESPLIEGA TANTO EL MENSAJE FIJO COMO EL MÓVIL, que esta representado a través del cuarto dígito (izquierda a derecha) de los

4 últimos caracteres de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 1
MENSAJE MÓVIL: Esta es un mensaje para probar la Alarma
Para el módulo ALARMA:

Número de Mensajero Asociado: #1
ENTRADA "A": E00
VERIFICACIÓN: ENTRADA A: BAJO, que esta representado a través del primer dígito (izquierda a derecha) del último caracter de la sentencia en lenguaje SIIL1 mostrada abajo. Su valor para este caso es: 0

La traducción que se obtendrá para este ejemplo en lenguaje SIIL1 será:

MENSAJERO#1 "MENSAJE",1,14,150,1001;
 # Este es un mensaje para probar|
la Alarma
ALARMA#1 E00,0;

Nota: La sintaxis de la traducción para cada módulo lógico se verá más adelante en la sección A.3.6 de este mismo Anexo.

A.3.23 Dos Módulos No Pueden Ocupar El Mismo Espacio

Por diversos motivos suelen suceder accidentes a la hora de insertar los módulos, uno de ellos es la **SUPERPOSICIÓN** de dos de ellos en un mismo ESPACIO. Por ejemplo, un módulo CONTADOR DE EVENTOS se encuentra ubicado a partir del renglón 3 (R3) y hasta el renglón 5 (R5), por su estructura este módulo necesita TRES renglones de la ZONA DE TRABAJO para su configuración. Intentemos ahora colocar un módulo FFARS en el renglón 2 (R2); este módulo ocupará DOS renglones de la ZONA DE TRABAJO, lo que significa que abarcará desde el renglón 2 (R2) hasta el renglón 3 (R3), el cual ya se encuentra ocupado por el módulo CONTADOR DE EVENTOS.

Este tipo de colisiones provocarán que se envíen MENSAJES DE ERROR como el siguiente:

Figura A.219 – Ventana de Error al tratar de insertar un módulo en un renglón que ya esta ocupado por otro módulo.

La figura A.219 muestra un mensaje de ERROR que indica: “No hay espacio suficiente para que insertes este bloque en el renglón que seleccionaste. Este bloque necesita dos renglones vacíos”. Refiriéndose claramente al ejemplo expuesto anteriormente. Por lo tanto, para que el usuario evite la aparición de estos ERRORES deberá tomar en cuenta la cantidad de renglones que ocupa cada módulo en la ZONA DE TRABAJO:

MÓDULOS LÓGICOS: COMPUERTAS

- SEGUIDOR LÓGICO: Un Renglón
- INVERSOR LÓGICO: Un Renglón
- COMPUERTA LÓGICA AND:
  - Dos Entradas: Un Renglón
  - Tres Entradas: Un Renglón
  - Cuatro Entradas: Un Renglón
- COMPUERTA LÓGICA OR (Al inicio o Al Final):
  - Dos Entradas: Dos Renglones
  - Tres Entradas: Tres Renglones
  - Cuatro Entradas: Cuatro Renglones

ESTRUCTURA DE COMPUERTAS: COMPUERTAS AND Y OR

- COMPUERTA LÓGICA OR de 2 ENTRADAS (Al Inicio o Al Final) CON COMPUERTA LÓGICA AND de 1,2 o 3 ENTRADAS: 2 Renglones.
- COMPUERTA LÓGICA OR de 3 ENTRADAS (Al inicio o Al Final) CON COMPUERTA LÓGICA AND de 1,2 o 3 ENTRADAS: 3 Renglones.
- COMPUERTA LÓGICA OR de 4 ENTRADAS (Al Inicio o Al Final) CON COMPUERTA LÓGICA AND de 1,2 o 3 ENTRADAS: 4 Renglones.
- COMPUERTA LÓGICA OR de 2 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 2 ENTRADAS Al Final: 2 Renglones.
- COMPUERTA LÓGICA OR de 2 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 3 ENTRADAS Al Final: 3 Renglones.
- COMPUERTA LÓGICA OR de 2 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 4 ENTRADAS Al Final: 4 Renglones.

- COMPUERTA LÓGICA OR de 3 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 2 ENTRADAS Al Final: 3 Renglones.
- COMPUERTA LÓGICA OR de 3 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 3 ENTRADAS Al Final: 3 Renglones.
- COMPUERTA LÓGICA OR de 3 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 4 ENTRADAS Al Final: 4 Renglones.
- COMPUERTA LÓGICA OR de 4 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 2 ENTRADAS Al Final: 4 Renglones.
- COMPUERTA LÓGICA OR de 4 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 3 ENTRADAS Al Final: 4 Renglones.
- COMPUERTA LÓGICA OR de 4 ENTRADAS Al Inicio CON COMPUERTA LÓGICA OR de 4 ENTRADAS Al Final: 4 Renglones.
- COMPUERTA LÓGICA OR de 2 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 2 ENTRADAS Al Inicio: 2 Renglones.
- COMPUERTA LÓGICA OR de 2 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 3 ENTRADAS Al Inicio: 3 Renglones.
- COMPUERTA LÓGICA OR de 2 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 4 ENTRADAS Al Inicio: 4 Renglones.
- COMPUERTA LÓGICA OR de 3 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 2 ENTRADAS Al Inicio: 3 Renglones.
- COMPUERTA LÓGICA OR de 3 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 3 ENTRADAS Al Inicio: 3 Renglones.
- COMPUERTA LÓGICA OR de 3 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 4 ENTRADAS Al Inicio: 4 Renglones.
- COMPUERTA LÓGICA OR de 4 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 2 ENTRADAS Al Inicio: 4 Renglones.
- COMPUERTA LÓGICA OR de 4 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 3 ENTRADAS Al Inicio: 4 Renglones.
- COMPUERTA LÓGICA OR de 4 ENTRADAS Al Final CON COMPUERTA LÓGICA OR de 4 ENTRADAS Al Inicio: 4 Renglones.

MÓDULOS LÓGICOS

- FLIP-FLOP ASINCRONO SR: Dos Renglones
- CONTADOR DE EVENTOS: Tres Renglones
- SECUENCIADOR DE ESTADOS: Tres Renglones
- TEMPOA: Tres Renglones
- TEMPOB: Un Renglón
- TEMPOC: Dos Renglones
- TEMPOD: Dos Renglones
- TEMPOE: Un Renglón
- TEMPOG: Dos Renglones

MÓDULOS AUXILIARES

- MENSAJERO: Un Renglón.
- ALARMA: Un Renglón.

Queda manifiesto a través de estas correspondencias el adecuado uso de los renglones de la ZONA DE TRABAJO por parte del usuario.

A.4 La Herramienta Borrar

Esta herramienta permitirá "BORRAR" de manera individual compuertas lógicas, módulos lógicos y módulos auxiliares. Existiendo también la posibilidad de borrar todos los módulos existentes en la ZONA DE TRABAJO.

La naturaleza del ser humano impide llegar a la perfección, el cometer ERRORES nos ayuda de alguna manera alcanzar un poco de ella. Este programa contiene una función que permite enmendar ERRORES cometidos a la hora de configurar alguna compuerta lógica, algún módulo lógico u auxiliar.

Esta función la realiza el botón "Borrar", el cual se encuentra ubicado en el costado derecho central de la ventana principal, al dar clic en él aparecerá la siguiente ventana:

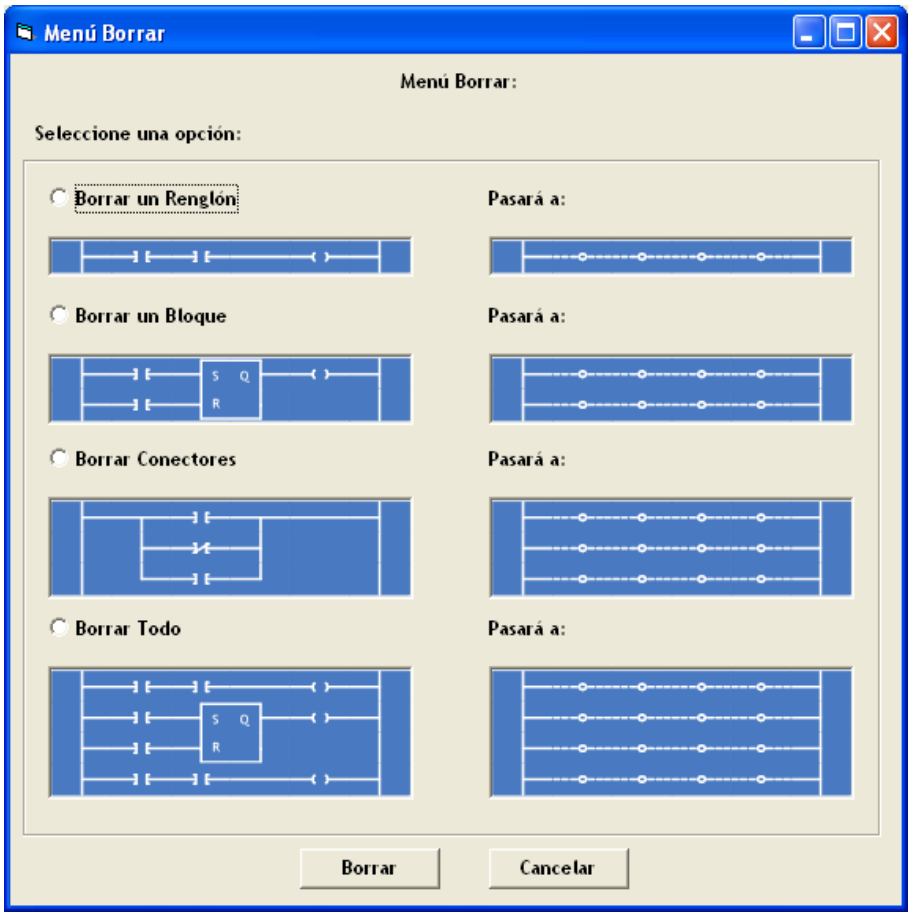


Figura A.220 – Ventana para configurar la forma de borrado en la Zona de Trabajo del "PUMA ESCALERA".

La Figura A.220 muestra la interfaz que permitirá seleccionar sobre cuatro opciones de borrado:

1. **Borrar un Renglón**
2. **Borrar un Bloque**
3. **Borrar Conectores**
4. **Borrar Todo**

A.4.1 Borrar Un Renglón

Esta opción servirá para borrar módulos lógicos que ocupen un renglón en la ZONA DE TRABAJO, como SEGUIDORES, NEGADORES y COMPUERTAS LÓGICAS AND. Esta opción excluye módulos lógicos como el TEMPOB, TEMPOE, MENSAJERO Y ALARMA.

Imagínese que se desea borrar una compuerta lógica AND de TRES entradas, la cual se encuentra ubicada en el renglón 2 como la mostrada en la figura A.221:

Figura A.221 – Compuerta Lógica AND insertada en el renglón R2 de la Zona de Trabajo.

Al elegir la opción “Borrar un Renglón” de la ventana Menú Borrar:

Figura A.222 – Opción para borrar un renglón de la Zona de Trabajo.

En la figura A.222 se puede observar como quedará el renglón después de ser borrado, esto se hace con el fin de mostrarle al usuario los resultados antes de que tome alguna decisión. Una vez que ya se haya seleccionado la opción “Borrar un Renglón” habrá que dar clic con el botón izquierdo del ratón en el botón “Borrar” de la misma ventana. En seguida aparecerá el siguiente cuadro de diálogo:

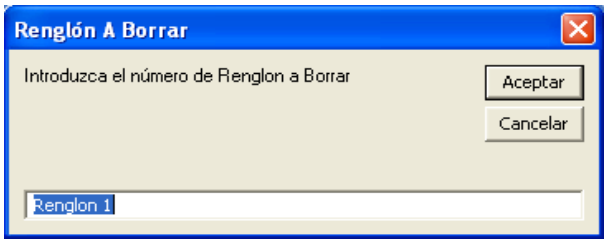


Figura A.223 – Ventana para seleccionar el Renglón a Borrar en la Zona de Trabajo.

El cuadro de diálogo que muestra la figura A.223 solicita explícitamente el número del renglón a borrar. El usuario deberá introducir el número del renglón donde se encuentra el módulo lógico que desea borrar. Por default se ha colocado el Renglón 1, como una

guía de sintaxis para el usuario, sin embargo, también será posible escribir R1, Ren 1, Reng 1, etc.; incluso será posible introducir sólo el número del renglón. Lo importante es escribir alguna sentencia que mantenga siempre el número en el último carácter, o los últimos dos caracteres si el número es de dos dígitos.

En este ejemplo se introdujo R2, enseguida dar clic en el botón “Aceptar”. El resultado será el siguiente:

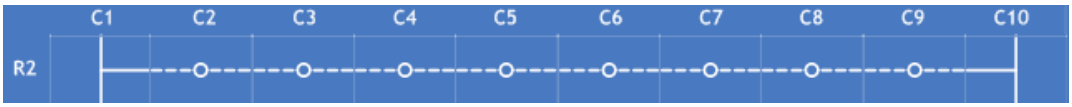


Figura A.224 – Renglón R2 después del borrado de la Compuerta Lógica AND.

La figura A.224 muestra un renglón “BORRADO”, por lo cual el renglón 2 está listo para ser utilizado de nuevo.

A.4.2 Borrar Un Bloque

Esta opción servirá para borrar módulos lógicos o BLOQUES que ocupen DOS o TRES renglones en la ZONA DE TRABAJO, como FFARS, TEMPOA, TEMPOB, TEMPOC, TEMPOD, TEMPOE, TEMPOG, CONTADORES Y SECUENCIADORES. Además de módulos auxiliares como el MENSAJERO y la ALARMA.

Imagine que se desea borrar un módulo lógico CONTADOR DE EVENTOS, el cual se encuentra ubicado en el renglón 1:

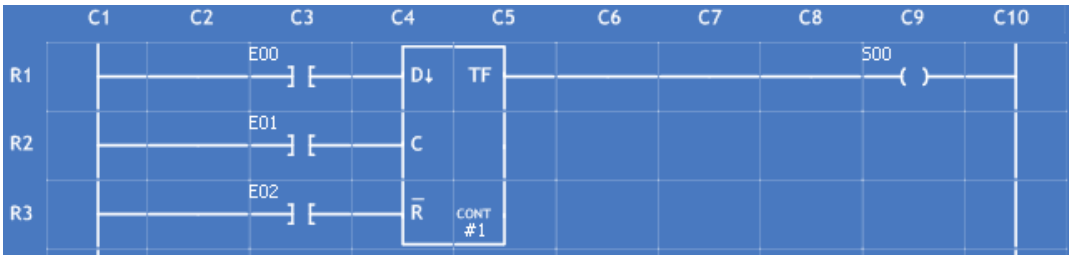


Figura A.225 – Contador de Eventos insertado en los renglones R1, R2 y R3 de la Zona de Trabajo del “PUMA ESCALERA”.

Elegir la opción “Borrar un Bloque” de la ventana Menú Borrar:

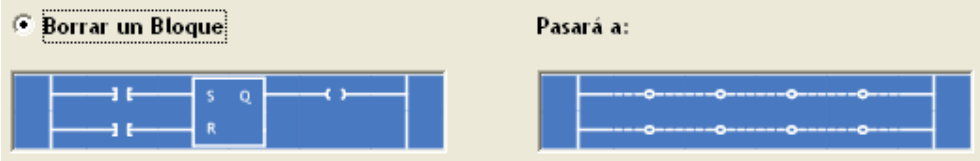


Figura A.226 – Opción para borrar un Bloque de la Zona de Trabajo.

En la figura A.226 se puede observar como quedarán los renglones después de ser borrados, esto se hace con el fin de mostrarle al usuario los resultados antes de que tome alguna decisión. Una vez que ya se haya seleccionado la opción “Borrar un Bloque”

habrá que dar clic con el botón izquierdo del ratón en el botón “Borrar” de la misma ventana. En seguida aparecerá el siguiente cuadro de diálogo:

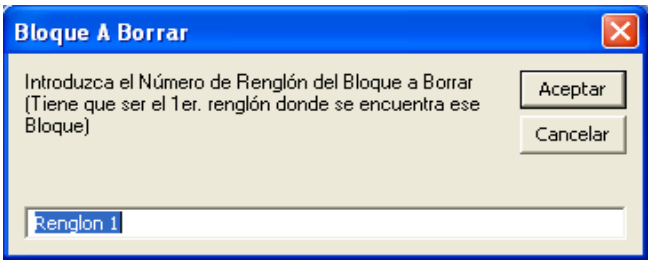


Figura A.227 – Ventana para seleccionar el Renglón del Bloque a Borrar en la Zona de Trabajo.

El cuadro de diálogo que muestra la figura A.227 solicita explícitamente el número del renglón del BLOQUE a borrar. El usuario deberá introducir el número del PRIMER renglón donde se encuentra el BLOQUE que desea borrar. Por default se ha colocado el Renglón 1, como una guía de sintaxis para el usuario, sin embargo, también será posible escribir R1, Ren 1, Reng 1, etc.; incluso será posible introducir sólo el número del renglón. Lo importante es escribir alguna sentencia que mantenga siempre el número en el último carácter, o los últimos dos caracteres si el número es de dos dígitos.

En este ejemplo se introdujo R1, que es el PRIMER RENGLÓN donde se encuentra el BLOQUE. Enseguida dar clic en el botón “Aceptar”. El resultado será el siguiente:

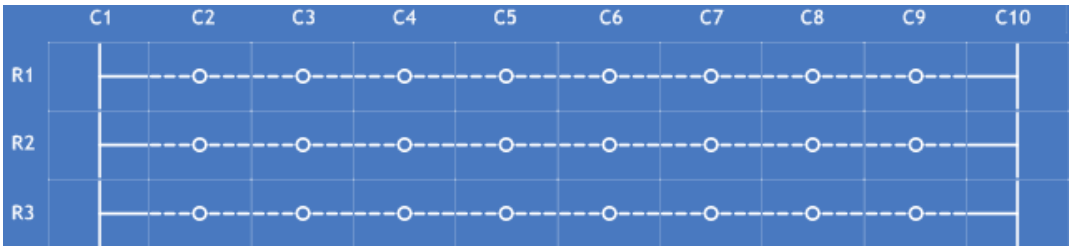


Figura A.228 – Renglones R1, R2 y R3 después del borrado del Contador de Eventos.

La figura A.228 muestra tres renglones “BORRADOS”, que son los mismos que ocupaba el bloque CONTADOR DE EVENTOS, por lo cual los renglones 1, 2 y 3 están en disposición para ser utilizados de nuevo.

A.4.3 Borrar Conectores

Esta opción servirá para borrar única y exclusivamente COMPUERTAS LÓGICAS OR Y AND. Para ser más específicos módulos lógicos creados mediante las herramientas “CONECTORES AL INICIO” U “CONECTORES AL FINAL”. Esto quiere decir entonces, que se borrará toda la estructura creada, no sólo una parte de ella.

Imagine que se desea borrar el siguiente módulo: DOS compuertas OR cuyas salidas son las primeras dos entradas de la compuerta lógica AND cuya salida es S00, la tercera entrada de esta compuerta AND es una entrada Preinvertida (E07), toda esta estructura se encuentra ubicada a partir del renglón 1:

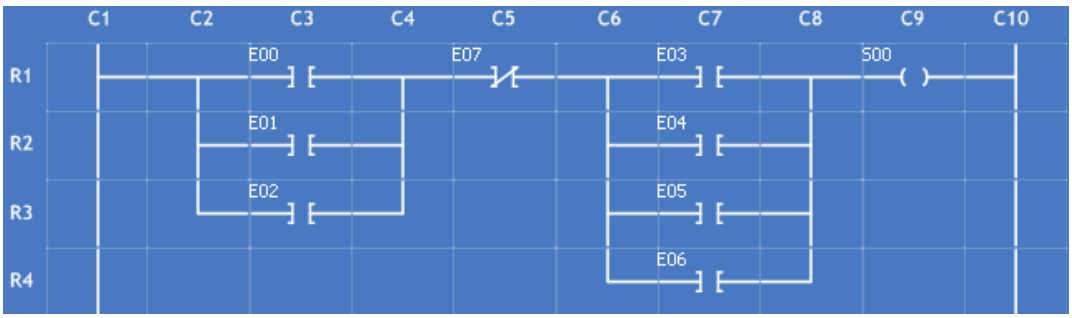


Figura A.229 – Compuertas Lógicas OR insertadas en los renglones R1, R2, R3 y R4 de la Zona de Trabajo del "PUMA ESCALERA".

Elegir la opción "Borrar Conectores" de la ventana Menú Borrar:

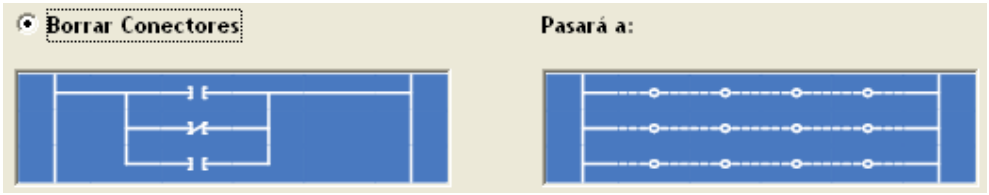


Figura A.230 – Opción para borrar Conectores de la Zona de Trabajo.

En la figura A.230 se puede observar como quedarán los renglones después de ser borrados, esto se hace con el fin de mostrarle al usuario los resultados antes de que tome alguna decisión. Una vez que ya se haya seleccionado la opción "Borrar Conectores" habrá que dar clic con el botón izquierdo del ratón en el botón "Borrar" de la misma ventana. En seguida aparecerá el siguiente cuadro de diálogo:

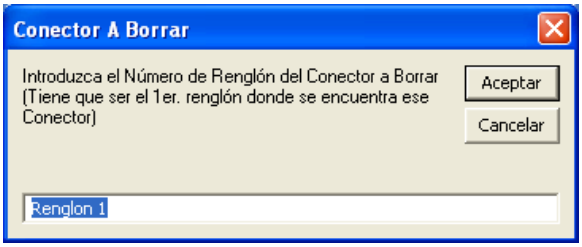


Figura A.231 – Ventana para seleccionar el Renglón del Conector a Borrar en la Zona de Trabajo.

El cuadro de diálogo que muestra la figura A.231 Solicita explícitamente el número del renglón del CONECTOR a borrar. El usuario deberá introducir el número del PRIMER renglón donde se encuentra el CONECTOR que desea borrar. Por default se ha colocado el Renglón 1, como una guía de sintaxis para el usuario, sin embargo, también será posible escribir R1, Ren 1, Reng 1, etc.; incluso será posible introducir sólo el número del renglón. Lo importante es escribir alguna sentencia que mantenga siempre el número en el último carácter, o los últimos dos caracteres si el número es de dos dígitos.

En este ejemplo se introdujo R1, que es el PRIMER RENGLÓN donde se encuentra el CONECTOR. Enseguida dar clic en el botón "Aceptar". El resultado será el siguiente:

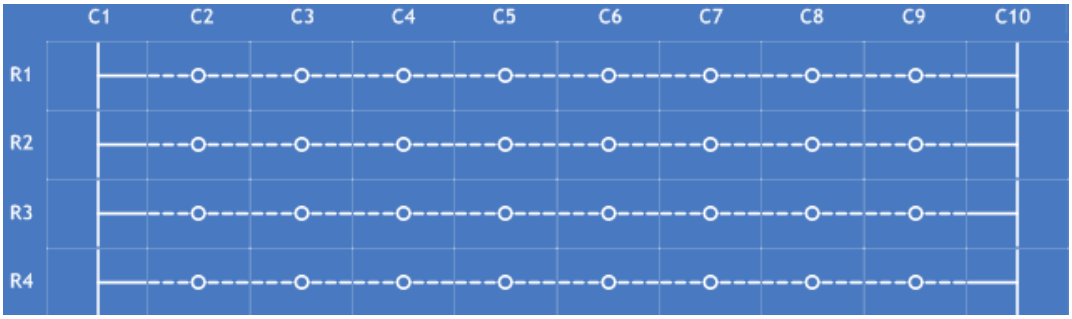


Figura A.232 – Renglones R1, R2, R3 y R4 después del borrado de los Conectores.

La figura A.232 muestra cuatro renglones “BORRADOS”, que son los mismos que ocupaba TODA LA ESTRUCTURA DEL CONECTOR, por lo cual los renglones 1, 2, 3 Y 4 están en disposición para ser utilizados de nuevo.

A.4.4 Borrar Todo

Esta opción servirá para borrar TODA LA ZONA DE TRABAJO. Esto implica que se eliminará todo módulo creado hasta el momento, incluyendo COMPUERTAS LÓGICAS, MÓDULOS LÓGICOS Y MÓDULOS AUXILIARES.

Esta es una herramienta que se debe utilizar en casos extremos, cuando la cantidad de ERRORES no pueda ser resuelta a través de las otras opciones de BORRADO.

BORRAR TODO conlleva la eliminación de todos los parámetros configurados para cada uno de los módulos creados. En otras palabras, se encontrarán nuevamente disponibles todas las variables boolenas de ENTRADA, SALIDA e INTERMEDIARIAS que se utilizaron en la elaboración de estas estructuras. Se borrarán parámetros como ESTADOS, PULSOS, VERIFICACIONES DE ENTRADAS O SALIDAS, TEXTOS DE MENSAJEROS, ETC.

Imagine que se desea borrar los siguientes módulos lógicos en la ZONA DE TRABAJO:

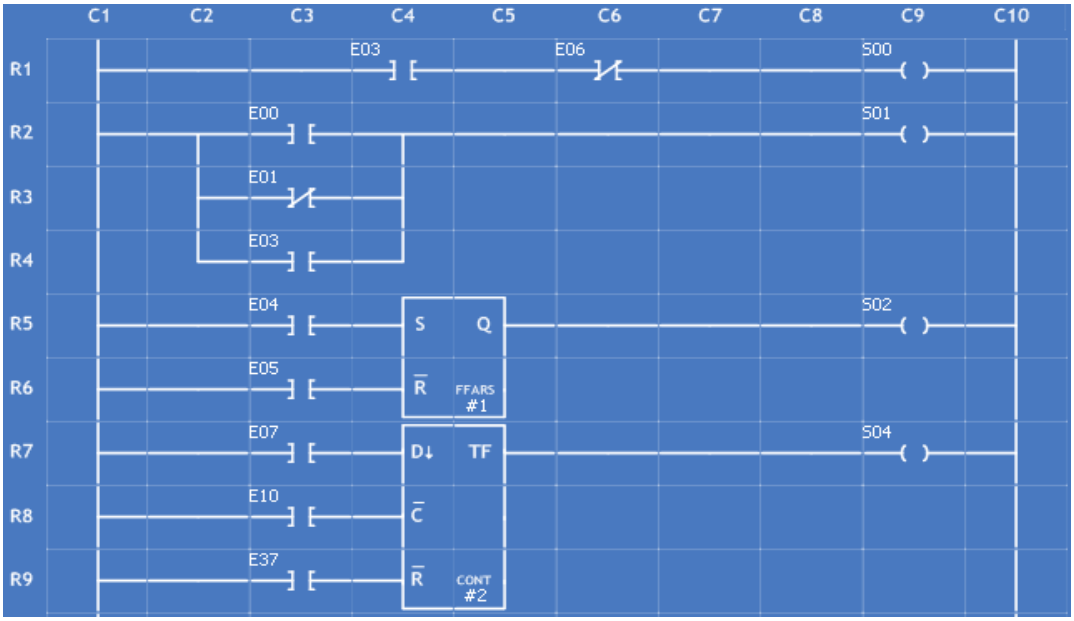


Figura A.233 – Módulos Lógicos Insertados desde el renglón R1 hasta el renglón R9 en la Zona de Trabajo del "PUMA ESCALERA".

Elegir la opción "Borrar Todo" de la ventana Menú Borrar:

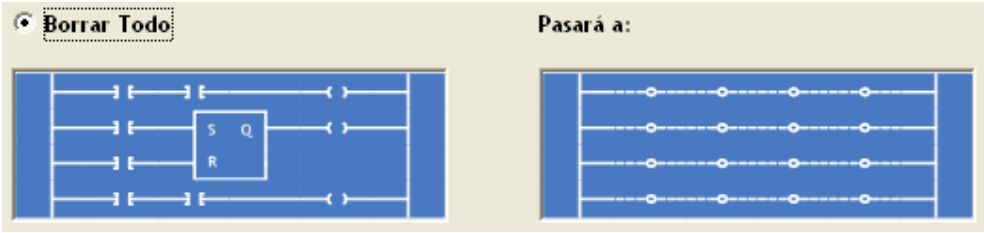


Figura A.234 – Opción para borrar Toda la Zona de Trabajo.

En la figura 4.15 se puede observar como quedarán los renglones después de ser borrados, esto se hace con el fin de mostrarle al usuario los resultados antes de que tome alguna decisión. Una vez que ya se haya seleccionado la opción "Borrar Todo" habrá que dar clic con el botón izquierdo del ratón en el botón "Borrar" de la misma ventana. En seguida aparecerá el siguiente cuadro de diálogo:

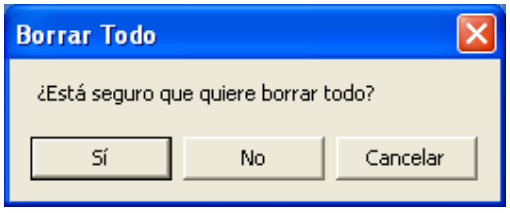


Figura A.235 – Venta de Confirmación para Borrar toda la Zona de Trabajo.

El cuadro de diálogo mostrado en la figura 4.16 es una forma de evitar que el usuario tome la decisión de BORRAR TODO. Si se responde afirmativamente aparecerá la siguiente ventana:

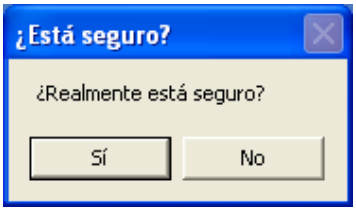


Figura A.236 – Última ventana de confirmación para borrar toda la Zona de Trabajo.

El cuadro de diálogo de la figura A.236 es una última oportunidad para el arrepentimiento. Si el usuario responde que "Sí" el resultado será el siguiente:

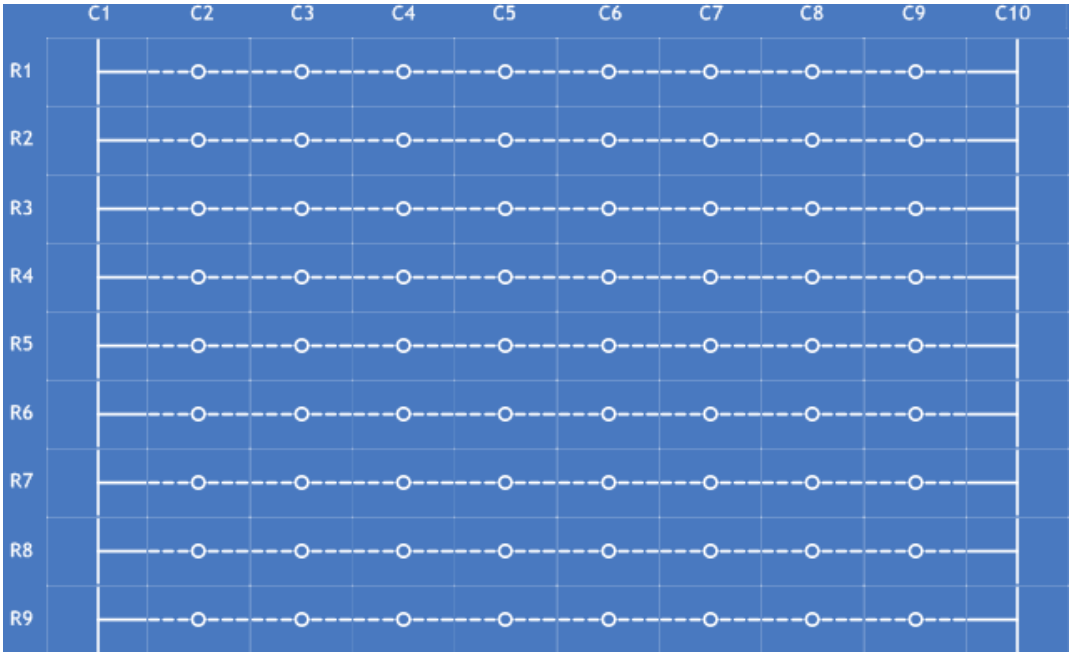


Figura A.237 – Zona de Trabajo después del Borrado de todos los módulos lógicos insertados en ella.

La figura A.237 muestra nueve renglones "BORRADOS", que son los mismos que ocupaban TODOS LOS MÓDULOS CREADOS, por lo cual todos estos renglones están en disposición para ser utilizados de nuevo. Por obvias razones no se muestra toda la ZONA DE TRABAJO, ya que sólo fueron utilizados nueve renglones para este pequeño proyecto.

A.4.5 Errores Comunes A La Hora De Borrar

Las primeras tres opciones de borrado (Borrar un Renglón, Borrar un Bloque y Borrar Conectores) mostrarán comúnmente este tipo de ERRORES:

Figura A.238 – Ventana de Error al tratar de Borrar un Conector en la Zona de trabajo.

Este tipo de ERROR es enviado por los siguientes motivos:

- Porque se ha seleccionado un renglón que no coincide con el tipo de opción especificada, es decir, si se seleccionó la opción "Borrar un Renglón" y se ha optado por borrar un renglón donde se encuentra algún módulo lógico u auxiliar. Cabe recordar que para borrar módulos lógicos y auxiliares se utiliza la opción "Borrar un Bloque".

- Porque se ha seleccionado un renglón que no existe, como: -1 ó 51. Hay que recordar que el intervalo va desde 1 hasta 50.

- Porque se ha seleccionado un renglón que se ya se encuentra vacío. De forma accidental se pudo haber escogido un renglón que no tenía relación alguna con la estructura que se quería elegir para ser borrada.

- Porque se ha seleccionado un renglón que no coincide con el primer renglón donde se encuentra un bloque ó algún Conector, es decir, si un bloque comienza a partir del renglón 3 y se ha elegido el renglón 4 para borrarlo se enviará el mensaje de ERROR.

A.5 El Proceso De Compilación

El proceso más importante en este software es el de la COMPILACIÓN. La compilación suele utilizarse para la traducción de cualquier representación simbólica de alto nivel a un formato simbólico de bajo nivel o comprensible para una máquina. El PUMA ESCALERA es un programa que realiza esta función, efectúa la traducción de un diagrama de escalera (alto nivel) a un formato comprensible para el Programador Lógico Modular (PLM).

Cualquier programa que se haga llamar COMPILADOR o INTÉRPRETE debe ser capaz de traducir una secuencia de instrucciones para que puedan ser procesadas sistemáticamente por un ordenador o computadora que reciben únicamente señales BINARIAS.

El PUMA ESCALERA es un COMPILADOR especial, ya que en lugar de tomar una secuencia de instrucciones de alto nivel para traducirlas a un lenguaje de bajo nivel (unos y ceros) CREA Y ANALIZA un diagrama de escalera y lo traduce en

instrucciones de lenguaje SIIL1 para ser procesadas por el Programador Lógico Modular (PLM).

Pero toda COMPILACIÓN antes de ser iniciada, debe cumplir ciertos requerimientos para su correcta INTERPRETACIÓN, TRADUCCIÓN Y FUNCIONAMIENTO. En el caso del PUMA ESCALERA todos estos requisitos son aplicados al DIAGRAMA DE ESCALERA, el cual se encuentra ubicado en su totalidad en la ZONA DE TRABAJO de nuestro programa. Antes de compilar tenemos que tomar en cuenta lo siguiente:

- Tener la plena seguridad de que se ha finalizado la etapa de la construcción del diagrama de escalera. Esto implica que se han introducido correctamente todos los datos en cada uno de los módulos lógicos, incluyendo compuertas lógicas y módulos auxiliares.

- **MUY IMPORTANTE**: Haber guardado el proyecto con algún nombre que lo identifique, ya que todos los archivos creados a partir de la compilación se formarán a partir de la sintaxis de este nombre.

Para guardar un proyecto se deberá dar clic en Archivo del Menú Herramientas de la ventana principal:

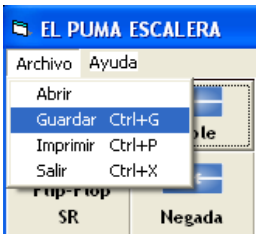


Figura A.239 – Opción “Archivo” en el Menú de Herramientas.

Se deberá dar clic nuevamente en Guardar del Menú Archivo. Aparecerá la siguiente ventana:

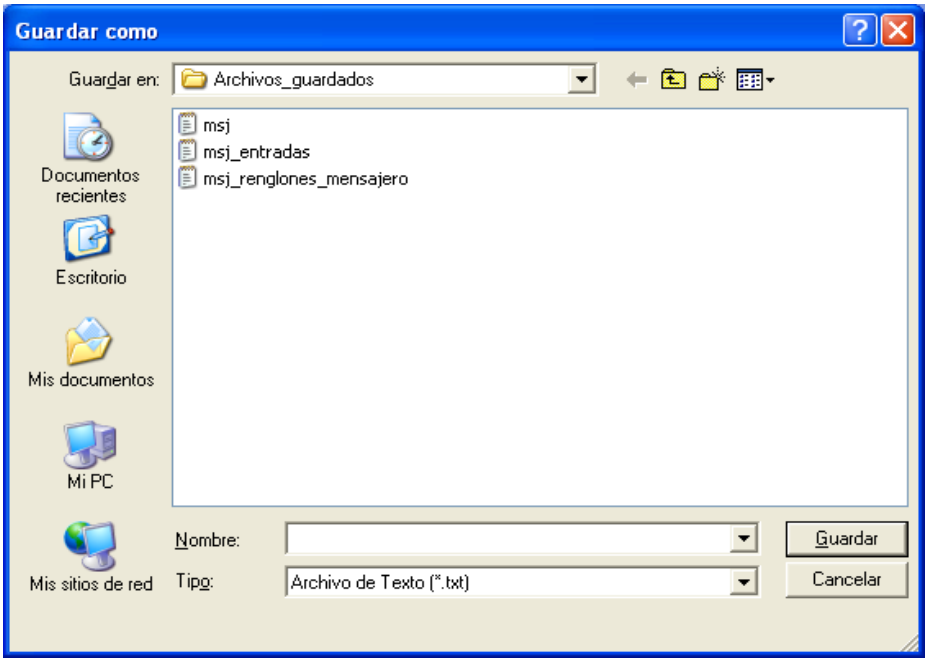


Figura A.240 – Ventana de Interacción para guardar todos los proyectos creados.

El cuadro de diálogo mostrado en la figura A.240 da por default una carpeta llamada "Archivos_guardados", en donde guardaremos **OBLIGATORIAMENTE** todos los proyectos. Una vez seleccionado algún nombre para él proyecto se deberá dar clic en el botón guardar, entonces se almacenará como un archivo de texto.

Una vez que se han verificado satisfactoriamente los requerimientos mencionados podremos iniciar la compilación. Para esto se deberá dar clic en el botón "Compilar", el cual se encuentra ubicado en el costado central derecho de la interfaz principal del programa:

Figura A.241 – Botón Compilar.

Cuando se da clic en este botón comienza una serie de procedimientos que va desde una revisión total de la ZONA DE TRABAJO para buscar renglones vacíos dejados por el usuario, pasando por el uso de dos autómatas para la identificación de ERRORES y de las diferentes estructuras de los MÓDULOS, hasta llegar a la traducción de todo el proyecto a través de diversos algoritmos de INTERPRETACIÓN. Todos estos procedimientos son descritos a continuación:

A.5.1 Llenado De Renglones Vacíos

La PRIMERA ETAPA de la compilación comienza con el "LLENADO DE RENGLONES VACÍOS", un renglón vacío es aquel renglón que no fue utilizado durante la construcción del diagrama de escalera. Su estructura es la siguiente:

Figura A.242 – Renglón vacío en la Zona de Trabajo.

La ubicación de estos renglones es por obviedad aleatoria, ya que se encontrarán comúnmente entre las estructuras de los diferentes módulos, ya sean compuertas lógicas, módulos lógicos o auxiliares. El otro lugar donde sustancialmente se encontrarán estos renglones será a partir del renglón donde se termine el proyecto; por ejemplo, si terminamos nuestro proyecto en el renglón 24 (R24), los renglones vacíos comenzarán a partir del renglón 25 (R25) hasta el renglón 50 (R50).

De esta manera, una vez que el software identificó todos los renglones vacíos de la ZONA DE TRABAJO, iniciará su "LLENADO" con la colocación de sendas LÍNEAS HORIZONTALES en cada una de las columnas de cada renglón vacío. Así, la estructura de cada renglón vacío quedará:

Figura A.243 – Llenado de un renglón vacío con cables después de la primera etapa de compilación.

La estructura mostrada en la figura A.243 le indicará al programa compilador que se trata de un renglón vacío. Esta arquitectura mostrará al software que es tiempo de pasar a la SEGUNDA ETAPA DE COMPILACIÓN.

A.5.2 El Autómata Identificador De Errores

La SEGUNDA ETAPA de la compilación es realizada por un el AUTÓMATA IDENTIFICADOR DE ERRORES, el cual se encarga de la revisión renglón a renglón de la ARQUITECTURA de cada módulo, ya sean compuertas lógicas, módulos lógicos o auxiliares.

El autómata comienza la revisión en el renglón 1 (R1) y en la columna 1 (C1). En este punto el software ya habrá identificado de qué estructura se trata. Cuando el programa compilador determine que el módulo ya ha terminado de revisarse decidirá si el módulo tiene o no la arquitectura correcta. Si la respuesta es negativa, se mostrará el correspondiente mensaje de ERROR en una nueva ventana. Se muestra a continuación un ejemplo:

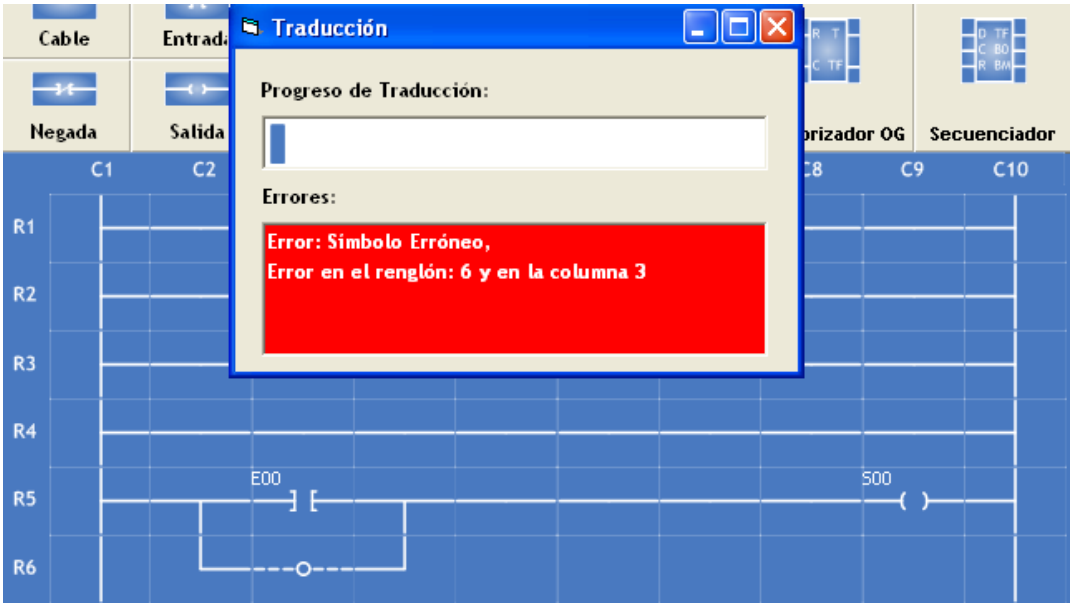


Figura A.244 – Ventana que muestra el proceso de Compilación, que después de un proceso de análisis muestra un Error en el renglón R6 y la columna C3.

La figura A.244 muestra una ventana donde se ha enviado el siguiente mensaje de ERROR: “Error: Símbolo Erróneo, Error en el renglón: 6 y en la columna 3”. Como se puede ver los renglones R5 y R6, tienen en sus columnas la estructura de una compuerta lógica OR, esta compuerta tiene una singularidad y es que carece de una ENTRADA en su arquitectura. El software detecta esta característica y envía el mensaje de ERROR correspondiente. Cabe señalar que este mensaje indica explícitamente la posición donde se encuentra el ERROR, señalando el número de RENGLÓN y el número de COLUMNA de este.

IMPORTANTE: El software detendrá el proceso de COMPILACIÓN cuando encuentre un ERROR en la arquitectura de algún módulo construido. Una vez que el software encuentre el ERROR el usuario tendrá la obligación de corregirlo, para iniciar nuevamente la COMPILACIÓN mediante el botón COMPILAR.

La figura A.244, además muestra las dos primeras etapas de compilación. Los primeros cuatro renglones fueron “LLENADOS” con líneas horizontales en todas sus columnas, lo mismo sucedió con los renglones vacíos no mostrados en la imagen. Una vez que el programa compilador detectó esta característica en los renglones vacíos, siguió con la ejecución de la segunda etapa, la cual detectó una compuerta lógica OR en el renglón 5 (R5). Cuando el software descubrió una anomalía en la ARQUITECTURA de la compuerta envió el mensaje de ERROR correspondiente.

No es menester de este manual mostrar cada uno los ERRORES que se enviarán cuando el programa compilador encuentre en el análisis de cada módulo alguna singularidad que no le permita continuar con la COMPILACIÓN. Todo mensaje de ERROR indicará explícitamente su causa así como su posición en la ZONA DE TRABAJO.

Si en la segunda etapa de compilación no se encontró ERROR alguno en cada uno de los módulos construidos en la ZONA DE TRABAJO, entonces el programa COMPILADOR estará en condiciones de ejecutar la tercera etapa de compilación.

A.5.3 El Autómata Identificador De Clases

La TERCERA ETAPA de la compilación es realizada por un el AUTÓMATA IDENTIFICADOR DE CLASES, el cual se encarga de la revisión renglón a renglón de la CLASE a la cual pertenece el módulo identificado en éste análisis. Una CLASE es el conjunto de características que identifican a un módulo determinado. Así, cada compuerta lógica pertenecerá a un tipo de CLASE, cada módulo lógico pertenecerá a una específica CLASE, lo mismo sucederá con los módulos auxiliares.

El análisis comenzará en el renglón 1 (R1) y en la Columna 1 (C1). Cuando el programa compilador identifique la CLASE a la que pertenece el módulo que ha revisado, mandará a llamar una función que hará la traducción correspondiente, la cual creará la sentencia en LENGUAJE SIIL1. Cada CLASE tiene asociada una función que realizará esta traducción. Es así, que habrá tantas funciones de traducción como el número de CLASES que existan.

Este proceso no es acumulativo, esto quiere decir que el programa compilador no reúne todas las CLASES encontradas para comenzar el llamado de funciones traductoras. En cuanto encuentra una CLASE manda a llamar su correspondiente función de traducción, continuando enseguida con el análisis del siguiente renglón. Hará la misma tarea hasta que termine la revisión de toda la ZONA DE TRABAJO.

Afortunadamente para el usuario no verá este proceso en pantalla. El objetivo no es abrumarlo con cúmulos y cúmulos de información. Sin embargo, es importante que conozca todos los procesos que realiza el programa compilador para lograr la traducción del proyecto creado.

Cuando el software haya terminado de reconocer todas las CLASES, en consecuencia habrá finalizado también el llamado de todas las funciones de traducción asociadas a esas CLASES. Cuando cada CLASE llama a su función traductora se esta iniciando una cuarta etapa de COMPILACIÓN, la etapa de TRADUCCIÓN. Entonces, cuando la última CLASE analizada llame a su correspondiente función de traducción se habrá completado esta penúltima etapa. Quedara así, como última etapa la formación de las sentencias en LENGUAJE SIIL1.

A.6 La Traducción Al Lenguaje SIIL1

La creación de las sentencias en lenguaje SIIL1 es la última etapa en el proceso de COMPILACIÓN. Las funciones de traducción son las encargadas de realizar esta última tarea.

Una vez que alguna función traductora ha recibido la señal de la creación de una sentencia en LENGUAJE SIIL1, comenzará a tomar datos que tiene almacenados el programa compilador como las variables de entrada y salida, las verificaciones de éstas o quizá el número de estados, pulsos o renglones que el módulo utilizará durante su

funcionamiento. Todos estos datos dependerán del módulo que haya sido detectado por la CLASE, que a su vez llamó a la función traductora para que fueran utilizados.

Cuando alguna sentencia en lenguaje SIIL1 es creada por una función de traducción es escrita en el **ARCHIVO FINAL DE TRADUCCIÓN**, el cual será un **ARCHIVO DE TEXTO (.TXT)** que contendrá un el programa en lenguaje SIIL1. El programa estará conformado por dos sectores fundamentales: el **SUBPROGRAMA PRINCIPAL** y el **SUBPROGRAMA TEMPORIZADO**. El algoritmo de cada función traductora decidirá si la sentencia irá al subprograma principal o al subprograma temporizado. Esta decisión depende del módulo que en ese momento de este traduciendo.

Cada sentencia en lenguaje SIIL1 sigue una sintaxis según el módulo que este representando. La sintaxis para cada módulo es la siguiente:

Módulos Lógicos (Ml)

Nota: Los ejemplos expuestos en cada módulo lógico tienen por objetivo mostrar la sintaxis que se utilizó en el programa "EL PUMA ESCALERA" para la creación de sentencias en el lenguaje SIIL1. Su única función es mostrar al usuario un panorama inmediato acerca de la construcción de la sentencia que se esté construyendo en ese momento.

A.6.1 Descripción Del Módulo De Seguimiento Lógico.

Este módulo lógico es declarado por el programa compilador en el subprograma principal, siendo la sintaxis para declararlo la siguiente:

SEG#N XEIEJE,XSISJS;

Donde:

- N denota el número de SEGUIDOR.
- XE podrá ser la letra "e", "s" o "i" mayúscula o minúscula dependiendo esto de que la variable de entrada al seguidor sea una VBE, VBS o VBI.
- IE denota el número de grupo que corresponda a la VB declarada como entrada al seguidor.
- JE denota el número de bit dentro del grupo Ie, asociado a la variable de entrada al seguidor.
- XS podrá ser la letra "s" o "i" mayúscula o minúscula dependiendo esto de que la variable de salida al seguidor sea una VBS o VBI.
- IS denota el número de grupo que corresponda a la VB declarada como salida del seguidor.
- JS denota el número de bit dentro del grupo IS, asociado a la variable de salida del seguidor.

Ejemplo: Se desea realizar con el PLM un seguidor lógico, requiriéndose que la entrada y salida al mismo sean respectivamente las VB E03 e I24; la declaración sintáctica sería:
SEG#4 E03, I24

A.6.2 Descripción Del Módulo De Inversión Lógica.

Este módulo lógico es declarado por el programa compilador en el subprograma principal, siendo la sintaxis para declararlo la siguiente:

NOT#N XEIEJE,XSISJS

Donde:

- N denota el número de INVERSOR.
- XE podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada al seguidor sea una VBE, VBS o VBI.
- IE denota el número de grupo que corresponda a la VB declarada como entrada al seguidor.
- JE denota el número de bit dentro del grupo Ie, asociado a la variable de entrada al seguidor.
- XS podrá ser la letra “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de salida al seguidor sea una VBS o VBI.
- IS denota el número de grupo que corresponda a la VB declarada como salida del seguidor.
- JS denota el número de bit dentro del grupo IS, asociado a la variable de salida del seguidor.

Ejemplo: Se desea realizar con el PLM un inversor lógico, requiriéndose que la entrada y la salida al mismo sean respectivamente las VB E12 e I67, la declaración sintáctica sería:
NOT#7 E12, I67

A.6.3 Descripción de Modulos lntermedios que realizan compuertas de dos entradas

El PUMA ESCALERA puede realizar dos tipos de compuertas lógicas de dos entradas y estas son de tipo: AND y OR, teniéndose además la capacidad de preinversión en las entradas que el usuario desee. Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

COMP#N X0I0J0, X1I1J1, XSISJS, AB

Donde:

- COMP es una cadena que puede ser AND2 u OR2 esto de acuerdo al tipo de compuerta que se desee realizar.
- N denota el número de compuerta, para cada uno de los dos tipos de compuertas posibles se ha de llevar una numeración independiente.
- X0 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E0 a la compuerta sea una VBE, VBS o VBI.
- I0 denota el número de grupo que corresponda a la VB declarada como entrada E0 a la compuerta.
- J0 denota el número de bit dentro del grupo I0, asociado a la variable de entrada E0.
- X1 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E1 a la compuerta sea una VBE, VBS o VBI.
- I1 denota el número de grupo que corresponda a la VB declarada como entrada E1 a la compuerta.
- J1 denota el número de bit dentro del grupo I1, asociado a la variable de entrada E1.
- XS podrá ser la letra “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de salida “S” de la compuerta sea una VBS o VBI.
- IS denota el número de grupo que corresponda a la VB declarada como salida de la compuerta.
- JS denota el número de bit dentro del grupo IS, asociado a la variable de salida de la compuerta.
- A es un dígito binario, que habrá de ser cero, si se desea que la entrada “E1” tenga preinversión, en otro caso el dígito “A” deberá ser uno.
- B es un dígito binario, que habrá de ser cero, si se desea que la entrada “E0” tenga preinversión, en otro caso el dígito “B” deberá ser uno.

Ejemplo: Se desea realizar con el PLM una compuerta AND de dos entradas, para la cual se desea que las entradas E0 y E1 y la salida S sean respectivamente las VB E01, I24, y S13, requiriéndose que la entrada E0 tenga preinversión, donde el número de asignación es 4, siendo la sintaxis para declararlo la siguiente:

AND2#4 E01, I24, S13, 10;

A.6.4 Descripción De Los Ml Que Realizan Compuertas De Tres Entradas

El PUMA ESCALERA puede realizar dos tipos de compuertas lógicas de tres entradas y estas son de tipo: AND y OR, teniéndose además la capacidad de preinversión en las entradas que el usuario desee. Este módulo lógico es declarado por el programa compilador en el subprograma principal, siendo la sintaxis para declararlo la siguiente:

COMP#N X0I0J0, X1I1J1, X2I2J2, XSISJS, ABC

Donde:

- COMP es una cadena que puede ser AND3 u OR3 esto de acuerdo al tipo de compuerta que se desee realizar.
- N denota el número de compuerta, para cada uno de los dos tipos de compuertas posibles se ha de llevar una numeración independiente.
- X0 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E0 a la compuerta sea una VBE, VBS o VBI.
- I0 denota el número de grupo que corresponda a la VB declarada como entrada E0 a la compuerta.
- J0 denota el número de bit dentro del grupo I0, asociado a la variable de entrada E0.
- X1 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E1 a la compuerta sea una VBE, VBS o VBI.
- I1 denota el número de grupo que corresponda a la VB declarada como entrada E1 a la compuerta.
- J1 denota el número de bit dentro del grupo I1, asociado a la variable de entrada E1.
- X2 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E2 a la compuerta sea una VBE, VBS o VBI.
- I2 denota el número de grupo que corresponda a la VB declarada como entrada E2 a la compuerta.
- J2 denota el número de bit dentro del grupo I2, asociado a la variable de entrada E2.
- XS podrá ser la letra “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de salida “S” de la compuerta sea una VBS o VBI.
- IS denota el número de grupo que corresponda a la VB declarada como salida de la compuerta.
- JS denota el número de bit dentro del grupo IS, asociado a la variable de salida de la compuerta.
- A es un dígito binario, que habrá de ser cero, si se desea que la entrada “E2” tenga preinversión, en otro caso el dígito “A” deberá ser uno.
- B es un dígito binario, que habrá de ser cero, si se desea que la entrada “E1” tenga preinversión, en otro caso el dígito “B” deberá ser uno.
- C es un dígito binario, que habrá de ser cero, si se desea que la entrada “E0” tenga preinversión, en otro caso el dígito “C” deberá ser uno.

Ejemplo: Se desea realizar con el PLM una compuerta OR de tres entradas para la cual se desea que las entradas E0, E1 y E2 y la salida S sean respectivamente las VB E14, I03, E17 y S17, requiriéndose que la entrada E1 tenga preinversión, donde el número de asignación es 7, siendo la sintaxis para declararlo la siguiente:

OR3#7 E14, I03, E17, S17, 101;

A.6.5 Descripción De Los Ml Que Realizan

Compuertas De Cuatro Entradas

El PUMA ESCALERA puede realizar dos tipos de compuertas lógicas de cuatro entradas y estas son de tipo: AND y OR, teniéndose además la capacidad de preinversión en las entradas que el usuario desee. Este módulo lógico es declarado por el programa compilador en el subprograma principal, siendo la sintaxis para declararlo la siguiente:

COMP#N X0I0J0, X1I1J1, X2I2J2, X3I3J3, XSISJS, ABCD

Donde:

- COMP es una cadena que puede ser AND3 u OR3 esto de acuerdo al tipo de compuerta que se desee realizar.
- N denota el número de compuerta, para cada uno de los dos tipos de compuertas posibles se ha de llevar una numeración independiente.
- X0 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E0 a la compuerta sea una VBE, VBS o VBI.
- I0 denota el número de grupo que corresponda a la VB declarada como entrada E0 a la compuerta.
- J0 denota el número de bit dentro del grupo I0, asociado a la variable de entrada E0.
- X1 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E1 a la compuerta sea una VBE, VBS o VBI.
- I1 denota el número de grupo que corresponda a la VB declarada como entrada E1 a la compuerta.
- J1 denota el número de bit dentro del grupo I1, asociado a la variable de entrada E1.
- X2 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E2 a la compuerta sea una VBE, VBS o VBI.
- I2 denota el número de grupo que corresponda a la VB declarada como entrada E2 a la compuerta.
- J2 denota el número de bit dentro del grupo I2, asociado a la variable de entrada E2.
- X3 podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada E3 a la compuerta sea una VBE, VBS o VBI.
- I3 denota el número de grupo que corresponda a la VB declarada como entrada E3 a la compuerta.
- J3 denota el número de bit dentro del grupo I3, asociado a la variable de entrada E3.
- XS podrá ser la letra “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de salida “S” de la compuerta sea una VBS o VBI.
- IS denota el número de grupo que corresponda a la VB declarada como salida de la compuerta.
- JS denota el número de bit dentro del grupo IS, asociado a la variable de salida de la compuerta.

- A es un dígito binario, que habrá de ser cero, si se desea que la entrada “E3” tenga preinversión, en otro caso el dígito “A” deberá ser uno.
- B es un dígito binario, que habrá de ser cero, si se desea que la entrada “E2” tenga preinversión, en otro caso el dígito “B” deberá ser uno.
- C es un dígito binario, que habrá de ser cero, si se desea que la entrada “E1” tenga preinversión, en otro caso el dígito “C” deberá ser uno.
- D es un dígito binario, que habrá de ser cero, si se desea que la entrada “E0” tenga preinversión, en otro caso el dígito “D” deberá ser uno.

Ejemplo: Se desea realizar con el PLM una compuerta AND de cuatro entradas para la cual se desea que las entradas E0, E1, E2 y E3 y la salida S sean respectivamente las VB E12, E14, I16, E06 y S03, requiriéndose que las entradas E1 y E0 tengan preinversión, donde el número de asignación es 8, siendo la sintaxis para declararlo la siguiente:

AND4#8 E12, E14, I16, E06, S03, 1100;

A.6.6 Descripción De Los Ml Que Realizan Flip-Flops R-S Asíncronos

Este módulo lógico es declarado por el programa compilador en el subprograma principal, siendo la sintaxis para declararlo la siguiente:

FFARS#N XSISJS, XRIR1JR, XQIQJQ, ABCD;

Donde:

- N denota el número de Flip-Flop.
- XS podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada EST (S) al Flip-Flop sea una VBE, VBS o VBI.
- IS denota el número de grupo que corresponda a la VB declarada como entrada “S” al Flip-Flop.
- JS denota el número de bit dentro del grupo IS, asociado a la variable de entrada “S”.
- XR podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada RESET (R) al Flip-Flop una VBE, VBS o VBI.
- IR denota el número de grupo que corresponda a la VB declarada como entrada “R” al Flip-Flop.
- JR denota el número de bit dentro del grupo IR, asociado a la variable de entrada “R”.
- XQ podrá ser la letra “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de salida “Q” del Flip-Flop sea una VBS o VBI.
- IQ denota el número de grupo que corresponda a la VB declarada como salida del Flip-Flop.
- JQ denota el número de bit dentro del grupo IQ, asociado a la variable de salida del Flip-Flop.

- A es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la entrada “S” sea bajo, en otro caso el dígito “A” deberá ser uno.
- B es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la entrada “R” sea bajo, en otro caso el dígito “B” deberá ser uno.
- C es un dígito binario, que habrá de ser uno si se desea que la VB de entrada SET tenga prioridad, en otro caso (prioridad para la VB de entrada RESET), “C” deberá ser cero. El hecho de que la entrada SET tenga prioridad implica que si ambas entradas SET y RESET se verifican simultáneamente la salida Q será uno lógico, por otro lado, prioridad para la entrada RESET significa que al verificarse ambas entradas del Flip-Flop la salida Q será puesta en cero lógico.
- D es un dígito binario, que habrá de ser cero, si se desea que la salida Q se inicialice en cero lógico, en otro caso el dígito “D” deberá ser uno.

Ejemplo: Se desea realizar con el PLM un módulo tipo LATCH, para el cual se desea que la entrada S tenga prioridad, deseándose que las entradas S, R y la salida Q sean respectivamente las VB E13, E14 y S06, requiriéndose que ambas entradas S y R tengan verificación en bajo y que el estado inicial de la salida Q sea uno, donde el número de asignación es 22, siendo la sintaxis para declararlo la siguiente:

FFARS#22 E13, E14, S06, 0011;

A.6.7 Descripción Del Ml Que Realiza Un Contador De Eventos

Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

CONTA#N XDIDJD, XCICJC, XRIRJR, XFIFJF,CUENTAI, CUENTAF, ABCDE;

Donde:

- N denota el número de contador de eventos.
- XD podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada D al contador sea una VBE, VBS o VBI.
- ID denota el número de grupo que corresponda a la VB declarada como entrada “D” al contador.
- JD denota el número de bit dentro del grupo ID, asociado a la variable de entrada “D”.
- XC podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de congelamiento (C) al contador sea una VBE, VBS o VBI.
- IC denota el número de grupo que corresponda a la VB declarada como entrada “C” al contador.

- JC denota el número de bit dentro del grupo IC, asociado a la variable de entrada "C".
- XR podrá ser la letra "e", "s" o "i" mayúscula o minúscula, dependiendo esto de que la variable de entrada RESET (R) al contador sea una VBE, VBS o VBI.
- IR denota el número de grupo que corresponda a la VB declarada como entrada "R" al contador.
- JR denota el número de bit dentro del grupo IR, asociado a la variable de entrada "R".
- XF podrá ser la letra "s" o "i" mayúscula o minúscula, dependiendo esto de que la variable de salida "TF" del contador sea una VBS o VBI.
- IF denota el número de grupo que corresponda a la VB declarada como salida del contador.
- JF denota el número de bit dentro del grupo IF, asociado a la variable de salida del contador.
- CUENTAI denota al valor de la cuenta inicial, debiendo el mismo estar comprendido entre cero y 65535, debiendo este valor ser menor que el correspondiente a la cuenta final si el contador es ascendente, en otro caso el valor declarado para la cuenta inicial deberá ser mayor que la cuenta final.
- CUENTAF denota el valor de la cuenta final, debiendo el mismo estar comprendido entre cero y 65535.
- A es un dígito binario, que habrá de ser cero, si se desea que se modifique la cuenta para flancos de bajada en la entrada de disparo "D", en otro caso el dígito "A" deberá ser uno.
- B es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la entrada "C" sea bajo, en otro caso el dígito "B" deberá ser uno.
- C es un dígito binario, que habrá de ser uno si se desea que el nivel de verificación de la entrada "R" sea bajo, en otro caso, "C" deberá ser cero.
- D es un dígito binario, que habrá de ser uno, si se desea que la cuenta sea ascendente, en otro caso el dígito "D" deberá ser cero.
- E es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la salida "TF" sea bajo, en otro caso el dígito "E" deberá ser uno.

Ejemplo: Se desea realizar con el PLM un módulo CONTADOR DE EVENTOS, con intervalo de cuenta comprendido entre cero y siete y testificación de fin de cuenta en nivel bajo, se requiere que los niveles de verificación de las entradas "R" y "C" sean en alto y que la entrada de disparo "D" sea sensible a flancos de bajada, se desea que las entradas D, C, R y la salida TF sean respectivamente las VB E17, I14, E12 y S01, donde el número de asignación es 14, siendo la sintaxis para declararlo la siguiente:

CONTA#14 E17,I14,E12,S01,0,7,01010;

Para entender mejor lo anterior es conveniente mostrar los diagramas de tiempo asociados con el contador de eventos, los cuales se pueden ver en la figura A.245:

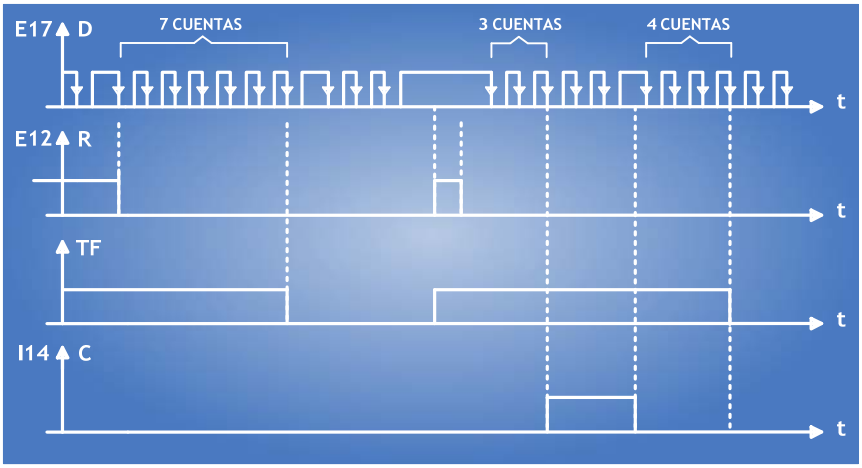


Figura A.245 – Diagrama de tiempo asociado al contador de eventos.

A.6.8 Descripción Del Ml Secuenciador De Estados De Nbxne

Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

SECNB#N XDIDJD, XCICJC, XRIRJR, XFIFJF, EVBPE, NE, ABCD;
 # [EF1][ESTADO1], [EF2][ESTADO2], [EF3][ESTADO3], ...[EFq][ESTADOq];
 # [EFp][ESTADOp], ...[EFr][ESTADOr];
.
.
[EFu][ESTADOu],.....[EFne][ESTADOne];

A continuación se explica el significado de las literales que aparecen en el primer renglón de la declaración genérica de un módulo secuenciador:

- NB denota el número de bits de la palabra de estado, debiendo el mismo estar comprendido entre uno y ocho, esto definido por el usuario.
- N denota el número de secuenciador.
- XD podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada D al secuenciador sea una VBE, VBS o VBI.
- ID denota el número de grupo que corresponda a la VB declarada como entrada “D” al secuenciador.
- JD denota el número de bit dentro del grupo ID, asociado a la variable de entrada “D”.
- XC podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de congelamiento (C) al secuenciador sea una VBE, VBS o VBI.
- IC denota el número de grupo que corresponda a la VB declarada como entrada “C” al secuenciador.

- JC denota el número de bit dentro del grupo IC, asociado a la variable de entrada "C".
- XR podrá ser la letra "e", "s" o "i" mayúscula o minúscula, dependiendo esto de que la variable de entrada RESET (R) al secuenciador sea una VBE, VBS o VBI.
- IR denota el número de grupo que corresponda a la VB declarada como entrada "R" al secuenciador.
- JR denota el número de bit dentro del grupo IR, asociado a la variable de entrada "R".
- XF podrá ser la letra "s" o "i" mayúscula o minúscula, dependiendo esto de que la variable de salida "TF" del secuenciador sea una VBS o VBI.
- IF denota el número de grupo que corresponda a la VB declarada como salida del secuenciador.
- JF denota el número de bit dentro del grupo IF, asociado a la variable de salida del secuenciador.
- EVBPE es un vector de NB elementos separados por comas, que especifica que variables booleanas se desea que sean cada uno de los bits de la palabra de estado, por lo tanto, EVBPE presentará la siguiente forma:

  XPIPJP,.....XLILJL,.....X0I0J0
  con P=NB-1 y:

  - XL (L=0, 1,.....NB-1), podrá ser la letra "s" o "i" mayúscula o minúscula, dependiendo esto de que el bit L de la palabra de estado del secuenciador sea una VBS o VBI.
  - IL (L=0, 1,.....NB-1), denota el número de grupo que corresponda a la VB declarada como bit L de la palabra de estado del secuenciador.
  - JL (L=0, 1,.....NB-1), denota el número de bit dentro del grupo IL, asociado a con el bit L de la palabra de estado.

- NE denota el número de estados que se desea presente el secuenciador y el mismo deberá ser mayor o igual que dos.
- A es un dígito binario, que habrá de ser cero, si se desea que se coloque el estado siguiente para flancos de bajada en la entrada de disparo "D", en otro caso el dígito "A" deberá ser uno.
- B es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la entrada "C" sea bajo, en otro caso el dígito "B" deberá ser uno.
- C es un dígito binario, que habrá de ser uno si se desea que el nivel de verificación de la entrada "R" sea bajo, en otro caso, "C" deberá ser cero.
- D es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la salida testigo de fin de carrera (TF) sea bajo, en otro caso el dígito "D" deberá ser uno.

En lo que toca a los renglones donde se declaran los valores que se desea tomen los estados que presentará el secuenciador, a continuación se explica el significado de los términos genéricos que los mismos contienen:

EFi (i=1, 2,.....NE), deberá ser la letra “B” si se desea que la especificación del estado “i” sea en formato binario, en caso de que se desee que la misma sea en formato hexadecimal EFi deberá ser la letra H; si el usuario escoge el formato binario deberá escribir la palabra de estado correspondiente limitándose al número de bits de la misma, por otro lado si el formato escogido fue el hexadecimal, el usuario deberá escribir un byte, en caso de que la longitud de la palabra de estado sea menor a ocho bits el valor binario de los bits no usados será irrelevante (don’t care).

- La literal “q”, denota el número correspondiente al último estado declarado en el primer renglón de especificación de valores de estados.
- La literal “p”, denota el número correspondiente al primer estado declarado en el segundo renglón de especificación de valores de estados.
- La literal “r”, denota el número correspondiente al último estado declarado en el segundo renglón de especificación de valores de estados.
- La literal “u”, denota el número correspondiente al primer estado declarado en el último renglón de especificación de valores de estados.
- Las literales “ne”, denotan el número correspondiente al último estado de la secuencia deseada.
- ESTADOi (i=1, 2,.....NE), denota el valor deseado para el estado i de la secuencia que se desea presente el secuenciador.

Cada renglón de especificación de valores de estados deberá tener un caracter “#” en la primera columna, para especificar el último renglón de datos el mismo habrá de iniciarse con dos caracteres “#” seguidos, colocándose el primero de ellos en la primera columna.

Nótese que el número de renglones a emplear, para especificar la secuencia de estados deseada, es variable, ya que se podrá especificar desde un solo estado en cada renglón hasta los que puedan contenerse en el ancho de la pantalla.

Ejemplo: Se desea realizar con el PLM un módulo SECUENCIADOR DE ESTADOS de tres bits por doce estados, el número de asignación es 9; se desea que las entradas de disparo, congelamiento y RESET sean respectivamente las VB E02, I00, y E00, para la salida testigo de fin de carrera ha de usarse la VB S17, las salidas asociadas con los tres bits de la palabra de estado en orden decreciente de significancia, deberán ser las VB I34, S03 y S01; se requiere que la entrada de disparo responda a flancos de bajada, las entradas de congelamiento y RESET y la salida TF sean verificadas en bajo; la lista de estados a secuenciar se muestra a continuación en formato binario:

Estado 01	**000**	Estado 07	**111**
Estado 02	**010**	Estado 08	**000**
Estado 03	**110**	Estado 09	**001**
Estado 04	**111**	Estado 10	**010**
Estado 05	**001**	Estado 11	**110**
Estado 06	**010**	Estado 12	**111**

Para fines ilustrativos, se usará el formato binario para declarar los primeros cinco estados de la lista, empleándose el formato hexadecimal para los demás, colocándose ceros en las posiciones de bit irrelevantes, una posible forma para declarar este secuenciador es la siguiente:

SEC3#9 E02, I00, E00, S17, I34, S03, S01, 12, 0010;

 # B000, B010, B110, B111;
 # B001, H02, H07, H00, H01, H02;
H06, H07

A.6.9 Descripción Del Ml Temporizador Monodisparo (One Shot) Del Primer Tipo: TEMPOA

Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

TEMPOA#N XDIDJD, XRIRJR, XHIHJH, XTITJT, HH:MM:SS.CS,ABCD;

Donde:

- N denota el número del TEMPOA.
- XD podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada D al temporizador sea una VBE, VBS o VBI.
- ID denota el número de grupo que corresponda a la VB declarada como entrada “D” al temporizador.
- JD denota el número de bit dentro del grupo ID, asociado a la variable de entrada “D”.
- XH podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de congelamiento (H) al temporizador sea una VBE, VBS o VBI.
- IH denota el número de grupo que corresponda a la VB declarada como entrada “H” al temporizador.
- JH denota el número de bit dentro del grupo IH, asociado a la variable de entrada “H”.
- XR podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada RESET (R) al temporizador sea una VBE, VBS o VBI.
- IR denota el número de grupo que corresponda a la VB declarada como entrada “R” al temporizador.
- JR denota el número de bit dentro del grupo IR, asociado a la variable de entrada “R”.
- XT podrá ser la letra “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de salida “T” del temporizador sea una VBS o VBI.
- IT denota el número de grupo que corresponda a la VB declarada como salida del temporizador.

- JT denota el número de bit dentro del grupo IT, asociado a la variable de salida del temporizador.
- HH denota un par de dígitos que especifican el número de horas en el tiempo Tm, siendo 47 el valor máximo aceptado.
- MM denota un par de dígitos que especifican el número de minutos en Tm.
- SS denota un par de dígitos que especifican los segundos en Tm.
- CS denota un par de dígitos que especifican las centésimas de segundo en Tm.
- A es un dígito binario, que habrá de ser cero, si se desea que el temporizador se dispare para flancos de bajada en la entrada de disparo "D", en otro caso el dígito "A" deberá ser uno.
- B es un dígito binario, que habrá de ser cero, si se desea que el temporizador se restablezca para flancos de bajada en la entrada de restablecimiento "R", en otro caso el dígito "B" deberá ser uno.
- C es un dígito binario, que habrá de ser uno si se desea que el nivel de verificación de la entrada "H" sea alto, en otro caso, "C" deberá ser cero.
- D es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la salida (T) sea bajo, en otro caso el dígito "D" deberá ser uno.

Ejemplo: Se desea realizar con el PLM un módulo TEMPOA, de manera que las entradas de disparo, restablecimiento y habilitación sean respectivamente las entradas físicas E00, E01 y E02, requiriéndose que la salida T sea la VBS S02, es necesario que el pulso de salida sea verificado en bajo y tenga una duración de dos minutos con treinta segundos, tanto el disparo como el restablecimiento deben ser por flanco de bajada y la señal de habilitación (H) debe ser verificada en alto. El número de asignación es 1, siendo la sintaxis para declararlo la siguiente:

TEMPOA#1 E00, E01, E02, S02, 00:02:30.00, 0010;

En la figura A.246 se presentan los diagramas de tiempo asociados con el ejemplo anterior:

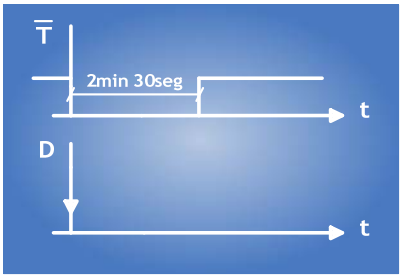


Figura A.246 – Diagrama de tiempo asociado al Temporizador Monodisparo Tipo 1 del ejemplo anterior.

A.6.10 Descripción Del Ml Temporizador Monodisparo (One Shot) Del Segundo Tipo: TEMPOC

Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

TEMPOC#N XDIDJD, XRIRJR, XTITJT, HH:MM:SS.CS,ABC;

Donde:

- N denota el número del TEMPOC.
- XD podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada D al temporizador sea una VBE, VBS o VBI.
- ID denota el número de grupo que corresponda a la VB declarada como entrada “D” al temporizador.
- JD denota el número de bit dentro del grupo ID, asociado a la variable de entrada “D”.
- XR podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de restablecimiento (R) al temporizador sea una VBE, VBS o VBI.
- IR denota el número de grupo que corresponda a la VB declarada como entrada “R” al temporizador.
- JR denota el número de bit dentro del grupo IR, asociado a la variable de entrada “R”.
- XT podrá ser la letra “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de salida “T” del temporizador sea una VBS o VBI.
- IT denota el número de grupo que corresponda a la VB declarada como salida del temporizador.
- JT denota el número de bit dentro del grupo IT, asociado a la variable de salida del temporizador.
- HH denota un par de dígitos que especifican el número de horas en el tiempo Tm, siendo 47 el valor máximo aceptado.
- MM denota un par de dígitos que especifican el número de minutos en Tm.
- SS denota un par de dígitos que especifican los segundos en Tm.
- CS denota un par de dígitos que especifican las centésimas de segundo en Tm.
- A es un dígito binario, que habrá de ser cero, si se desea que el temporizador se dispare para flancos de bajada en la entrada de disparo “D”, en otro caso el dígito “A” deberá ser uno.
- B es un dígito binario, que habrá de ser cero, si se desea que el temporizador sea restablecido por nivel alto, en otro caso el dígito “B” deberá ser uno.
- C es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la salida (T) sea bajo, en otro caso el dígito “C” deberá ser uno.

Ejemplo: Se desea realizar con el PLM un módulo TEMPOC, de manera que las entradas de disparo y restablecimiento sean respectivamente las entradas físicas E00 y E03, requiriéndose que la salida T sea la VBS S03, es necesario que el pulso de salida sea verificado en bajo y tenga una duración de treinta segundos, el disparo debe ser por

flanco de bajada y el restablecimiento debe ser por nivel bajo. El número de asignación es 2, siendo la sintaxis para declararlo la siguiente:

TEMPOC#2 E00, E03, S03, 00:00:30.00, 010;

En la figura A.2 se presentan los diagramas de tiempo asociados con el ejemplo anterior:

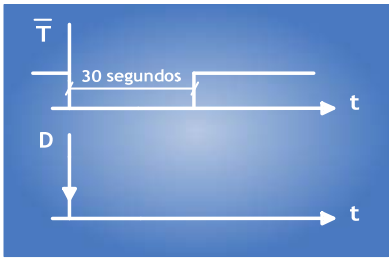


Figura A.247 – Diagrama de tiempo asociado al Temporizador Monodisparo Tipo 2 del ejemplo anterior.

A.6.11 Descripción Del Ml Que Realiza Temporizadores Con Retardo A La Activación (On-Delay) O Con Retardo A La Desactivación (Off-Delay): TEMPOD

Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

TEMPOD#N XDIDJD, XRIRJR, XTITJT, HH:MM:SS.CS,AB;

Donde:

- N denota el número del TEMPOD.
- XD podrá ser la letra “e”, “s” o “i” mayúscula o minúscula dependiendo esto de que la variable de entrada D al temporizador sea una VBE, VBS o VBI.
- ID denota el número de grupo que corresponda a la VB declarada como entrada “D” al temporizador.
- JD denota el número de bit dentro del grupo ID, asociado a la variable de entrada “D”.
- XR podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de restablecimiento (R) al temporizador sea una VBE, VBS o VBI.
- IR denota el número de grupo que corresponda a la VB declarada como entrada “R” al temporizador.
- JR denota el número de bit dentro del grupo IR, asociado a la variable de entrada “R”.
- XT podrá ser la letra “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de salida “T” del temporizador sea una VBS o VBI.
- IT denota el número de grupo que corresponda a la VB declarada como salida del temporizador.

- JT denota el número de bit dentro del grupo IT, asociado a la variable de salida del temporizador.
- HH denota un par de dígitos que especifican el número de horas en el tiempo Tm, siendo 47 el valor máximo aceptado.
- MM denota un par de dígitos que especifican el número de minutos en Tm.
- SS denota un par de dígitos que especifican los segundos en Tm.
- CS denota un par de dígitos que especifican las centésimas de segundo en Tm.
- A es un dígito binario, que habrá de ser cero, si se desea que el temporizador presente retardo a la desactivación (off-delay), en otro caso (A=1), el temporizador presentará retardo a la activación (on-delay).
- B es un dígito binario, que habrá de ser cero, si se desea que el temporizador sea restablecido por nivel alto, en otro caso el dígito "B" deberá ser uno.

Ejemplo: Se desea realizar con el PLM dos temporizadores, uno con retardo a la activación y el otro con retardo a la desactivación. Para el primero se requiere que el retardo a la activación sea de 7 segundos, siendo necesaria que la entrada "R" sea verificada en bajo, cuyo número de asignación es 3, las entradas de disparo y restablecimiento han de ser las VB E02 y E03, la salida debe ser la VB S04.

Para el segundo temporizador de este ejemplo, se requiere que presente un retardo a la desactivación de 10 segundos, con restablecimiento en nivel bajo, cuyo número de asignación es 4, las entradas de disparo y restablecimiento han de ser las VB E04 y E05, la salida debe ser la VB S05.

Una declaración para estos temporizadores podría ser la siguiente:

TEMPOD#3 E02, E03, S04, 00:00:07.00, 11;
TEMPOD#4 E04, E05, S05, 00:00:10.00, 01;

En las figuras A.248 y A.249 se presentan los diagramas de tiempo asociados con el ejemplo anterior:

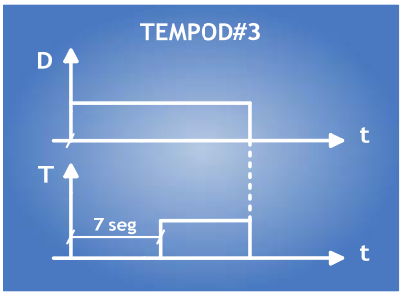


Figura A.248 – Diagrama de tiempo asociado al Primer Temporizador con Retardo a la Activación o Desactivación del ejemplo anterior.

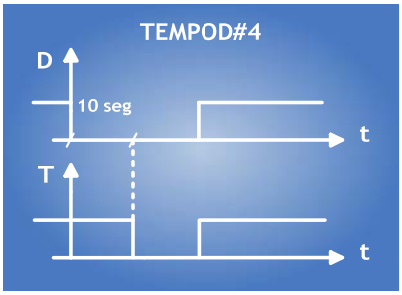


Figura A.249 – Diagrama de tiempo asociado al Segundo Temporizador con Retardo a la Activación o Desactivación del ejemplo anterior.

A.6.12 Descripción Del Ml Que Realiza Temporizadores Astables: TEMPOE

Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

TEMPOE#N XRIRJR, XTITJT, HHm:MMm:SSm.CSm, HHc:MMc:SSc.CSc, AB;

Donde:

- N denota el número de TEMPOE.
- XR podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de restablecimiento (R) al temporizador sea una VBE, VBS o VBI.
- IR denota el número de grupo que corresponda a la VB declarada como entrada “R” al temporizador.
- JR denota el número de bit dentro del grupo IR, asociado a la variable de entrada “R”.
- XT podrá ser la letra “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de salida “T” del temporizador sea una VBS o VBI.
- IT denota el número de grupo que corresponda a la VB declarada como salida del temporizador.
- JT denota el número de bit dentro del grupo IT, asociado a la variable de salida del temporizador.
- HHm denota un par de dígitos que especifican el número de horas en el tiempo Tm, siendo 47 el valor máximo aceptado.
- MMm denota un par de dígitos que especifican el número de minutos en Tm.
- SSm denota un par de dígitos que especifican los segundos en Tm.
- CSm denota un par de dígitos que especifican las centésimas de segundo en Tm.
- HHc denota un par de dígitos que especifican el número de horas en el tiempo Tc, siendo 47 el valor máximo aceptado.
- MMc denota un par de dígitos que especifican el número de minutos en Tc.
- SSc denota un par de dígitos que especifican los segundos en Tc.
- CSc denota un par de dígitos que especifican las centésimas de segundo en Tc.

- A es un dígito binario, que habrá de ser cero, si se desea que el temporizador se restablezca por nivel alto, en otro caso (A=1), el temporizador se restablecerá por nivel bajo.
- B es un dígito binario, que habrá de ser cero, si se desea que el temporizador arranque en cero, en otro caso el dígito "B" deberá ser uno.

Ejemplo: Se desea realizar con el PLM dos temporizadores astables, uno con arranque en uno y el otro con arranque en cero. Para el primero se requiere que los tiempos Tm y Tc sean respectivamente dos segundos y 250 ms, el nivel de restablecimiento ha de ser bajo y la VB asociada debe ser E06, la salida debe ser la VB S06, siendo el número de asignación de este temporizador el número cinco.

Para el segundo temporizador de este ejemplo, se requiere que los tiempos Tm y tc sean respectivamente un segundo y 300 ms, el nivel de restablecimiento ha de ser bajo siendo E07 la VB asociada, la salida debe ser la vb S07, siendo el número de asignación de este temporizador el número seis.

Una declaración para estos temporizadores podría ser la siguiente:

TEMPOE#5 E06, S06, 00:00:02.00, 00:00:00.25, 11;
TEMPOE#6 E07, S07, 00:00:01.00, 00:00:00.30, 10;

En las figuras A.250 y A.251 se presentan los diagramas de tiempo asociados con el ejemplo anterior:

Figura A.250 – Diagrama de tiempo asociado al Primer Temporizador Astable del ejemplo anterior.

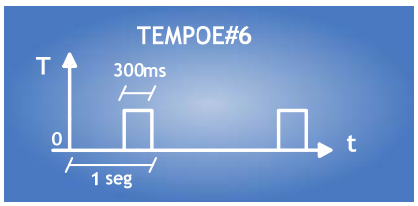


Figura A.251 – Diagrama de tiempo asociado al Primer Temporizador Astable del ejemplo anterior.

A.6.13 Descripción Del Ml Que Realiza Temporizadores Con Capacidad Para Generar N Pulsos A Intervalos De Tiempo Especificados Por El Usuario: TEMPOG

Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

TEMPOG#N XRIRJR, XCICJC, XTITJT, XFIFJF, NP, HH:MM:SS.CS, ABCDE;

 # Renglón de datos uno;
 # Renglón de datos dos;
.
.
.
último renglón de datos;

Donde:

- N denota el número de TEMPOG.
- NP denota el número de pulsos a generar.
- XR podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de restablecimiento (R) al temporizador sea una VBE, VBS o VBI.
- IR denota el número de grupo que corresponda a la VB declarada como entrada “R” al temporizador.
- JR denota el número de bit dentro del grupo IR, asociado a la variable de entrada “R”.
- XC podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de congelamiento (C) al temporizador sea una VBE, VBS o VBI.
- IC denota el número de grupo que corresponda a la VB declarada como entrada “C” al temporizador.
- JC denota el número de bit dentro del grupo IC, asociado a la variable de entrada “C”.
- XT podrá ser la letra “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de salida “T” del temporizador sea una VBS o VBI.
- IT denota el número de grupo que corresponda a la VB declarada como salida del temporizador.
- JT denota el número de bit dentro del grupo IT, asociado a la variable de salida del temporizador.
- XF podrá ser la letra “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de salida “TF” del temporizador sea una VBS o VBI.
- IF denota el número de grupo que corresponda a la VB declarada como salida testigo de fin de carrera del temporizador.
- JF denota el número de bit dentro del grupo IF, asociado a la variable de salida del temporizador.
- HH denota un par de dígitos que especifican el número de horas en el tiempo Tc, siendo 47 el valor máximo aceptado.
- MM denota un par de dígitos que especifican el número de minutos en Tc.
- SS denota un par de dígitos que especifican los segundos en Tc.
- CS denota un par de dígitos que especifican las centésimas de segundo en Tc.
- A es un dígito binario, que habrá de ser cero, si se desea que el temporizador se restablezca por nivel alto, en otro caso (A=1), el temporizador se restablecerá por nivel bajo.

- B es un dígito binario, que habrá de ser cero, si se desea que el arranque del tren de pulsos generado sea en bajo (pulsos verificados en alto), en otro caso el dígito “B” deberá ser uno.
- C es un dígito binario, que habrá de ser cero, si se desea que el nivel de verificación de la entrada de congelamiento sea bajo, en otro caso el dígito “C” deberá ser uno.
- D es un dígito binario, que habrá de ser cero si se desea que la salida testigo de fin de carrera sea verificada en bajo, en otro caso el dígito “D” deberá ser uno.
- E es un dígito binario, que habrá de ser cero si se desea deshabilitar la entrada de congelamiento, en otro caso (operación normal), el dígito “E” deberá ser uno; es importante señalar aquí que aún cuando la entrada de congelamiento sea deshabilitada, en la posición que corresponda a esta entrada en la declaración del módulo, deberá colocarse la especificación de una VB, de no hacerse esto el programa traductor de SIIL1 a código ejecutable por la CC del PLM, indicará un error de sintaxis.

En los renglones de datos habrán de colocarse, separados por comas, las especificaciones de tiempo que correspondan a cada uno de los Tmi (i = 1, 2, 3,....., N) implicados. Cada renglón de datos, excepto el último, deberá iniciar con el caracter “#” en la primera columna seguido por un espacio, el último renglón de datos deberá iniciar con dos caracteres “#” en la primera y segunda columna seguidos por un espacio, todos los renglones de datos deberán finalizar con el caracter “;”.

Ejemplo: Se desea realizar con el PLM un temporizador multidisparo que genere seis pulsos, es necesario que el arranque sea en cero (pulsos verificados en alto), se requiere que el tiempo Tc sea dos segundos, el nivel de restablecimiento ha de ser bajo y la VB asociada debe ser E06, la entrada de congelamiento “C” habrá de ser la VB E07 con un nivel de verificación en bajo, la salida de pulsos “T” deberá ser la VB S06 y la salida “TF” habrá de ser la VB S07 con verificación en alto, cuyo número de asignación es el número siete.

Los seis intervalos de tiempo (Tmi) implicados deberán presentarse en el orden que se indican a continuación: treinta segundos, un minuto con treinta segundos, veinte segundos, cinco segundos con 250 ms, dos minutos y un minuto con dos segundos.

La declaración de este temporizador multidisparo podría ser:

TEMPOG#7 E06, E07, S06, S07, 6, 00:00:02.00, 10011;
 # 30.00, 01:30.00, 20.00, 05.25;
02:00.00, 01:02.00;

En la figura A.252 se presentan los diagramas de tiempo asociados con el ejemplo anterior:

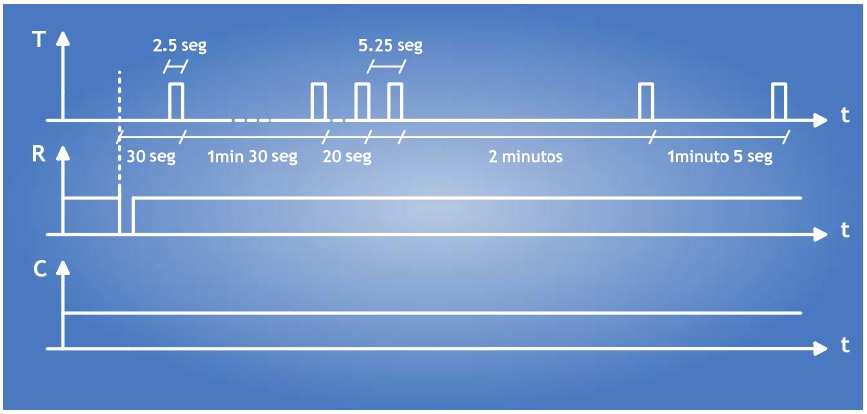


Figura A.252 – Diagrama de tiempo asociado Temporizador Generador de N Pulsos del ejemplo anterior.

A.6.14 Descripción Del Ml Que Realiza Temporizadores Con Capacidad De Generación De Pulsos En Instantes De Acuerdo Al Estado Del Reloj De Tiempo Real (Rtr): TEMPOB

Este módulo lógico es declarado por el programa compilador en el subprograma principal, siendo la sintaxis para declararlo la siguiente:

TEMPOB#N XTITJT, CLASE, NT, A;
 # Renglón de datos uno;
 # Renglón de datos dos;
.
.
.
último renglón de datos;

Donde:

- N denota el número de TEMPOB; cabe señalar aquí el hecho de que la numeración para módulos GPRTR es independiente de la correspondiente a la correspondiente con los otros cinco temporizadores que puede realizar el PLM.
- CLASE es un dígito que denota la clase de temporizador GPRTR que se desea realizar.
- NT denota el número de pulsos a generar, el cual es limitado a cincuenta por módulo.
- XT podrá ser la letra “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de salida “T” del temporizador sea una VBS o VBI.
- IT denota el número de grupo que corresponda a la VB declarada como salida del temporizador.
- JT denota el número de bit dentro del grupo IT, asociado a la variable de salida del temporizador.

- A es un dígito binario, que habrá de ser cero, si se desea que los pulsos de salida sean verificados en bajo, en otro caso (pulsos de salida verificados en alto) “A” deberá ser uno.

En los renglones de datos habrán de colocarse, separados por comas, las especificaciones de disparo deseadas, cada renglón de datos, excepto el último, deberá iniciar con el caracter “#” en la primera columna seguido por un espacio, el último renglón de datos deberá iniciar con dos caracteres “#” en la primera y segunda columna seguidos por un espacio, todos los renglones de datos deberán finalizar con el caracter “;”.

Ejemplo: Se desea realizar con el PLM un temporizador de tipo GPRTR que genere cinco pulsos verificados en alto, los primeros dos habrán de presentarse los días lunes a las ocho de la mañana en punto y a las once horas con cinco minutos, los otros tres se deberán presentar los días miércoles jueves y sábado a las cinco de la tarde; la salida “T” deberá ser la VBS S12, cuyo número de asignación es el número nueve; el temporizador GPRTR obviamente sería de clase cuatro.

La declaración de este temporizador GPRTR podría ser:

TEMPOB#9 S12, 4, 5, 1;
 # LU/08:00:00, LU/11:05:00, MI/17:00:00;
JU/17:00:00, SA/17:00:00;

Módulos Auxiliares (MA)

Existen módulos para el PLM denominados como auxiliares, que sirven para manejar facilidades que no son propiamente funciones lógicas, para varios de ellos la declaración correspondiente no requiere de operandos, algunos de estos módulos se deberán colocar en el subprograma principal y otros en el temporizado.

Nota: Los ejemplos expuestos en cada módulo auxiliar tienen por objetivo mostrar la sintaxis que se utilizó en el programa “EL PUMA ESCALERA” para la creación de sentencias en el lenguaje SIIL1. Su única función es mostrar al usuario un panorama inmediato acerca de la construcción de la sentencia que se esté construyendo en ese momento.

A.6.15 Descripción Del Módulo Auxiliar (Ma) Con Capacidad Para Generar Texto Estático Y/O Dinámico En La Ud Del PLM: Mensajero

Este módulo lógico es declarado por el programa compilador en el subprograma temporizado, siendo la sintaxis para declararlo la siguiente:

MENSAJERO#N “MENSAJE FIJO”, CI, TV, P, ABCD;

 # Renglón de datos uno
 # Renglón de datos dos
.
.
.
último renglón de datos

Donde:

- N denota el número de MENSAJERO, el valor máximo permitido es 32.
- MENSAJE FIJO es una cadena de un máximo de dieciséis caracteres, que es colocada a partir de la columna uno del renglón que el usuario especifique, nótese el empleo de comillas para delimitar el texto del mensaje fijo.
- CI denota el número de columna, que delimita el extremo izquierdo de la ventana donde se desplegará el mensaje móvil.
- TV representa el tamaño en columnas del mensaje móvil.
- P es un número comprendido entre 1 y 255, que representa el intervalo en centésimas de segundo entre dos posiciones subsecuentes del mensaje móvil.
- A es un dígito binario, que habrá de ser uno, si se desea que el mensaje fijo se despliegue en el renglón uno, en otro caso "A" deberá ser cero.
- B es un dígito binario, que deberá ser uno, si se desea que el mensaje móvil se despliegue en el renglón uno, en otro caso "B" deberá ser cero.
- C es un dígito binario, que deberá ser cero, si se desea que el mensajero opere sólo si el mismo es el módulo desplegador activo, si "C" es uno, el mensajero opera independientemente del hecho de que el mismo sea el módulo desplegador activo, desde luego que el usuario sería responsable de que en el último caso no se produjera una colisión entre dos mensajeros, que intentarán escribir texto en una misma zona de la pantalla de la UD.
- D es un dígito binario, que deberá ser cero, si se desea que únicamente se despliegue el mensaje fijo, si "D" se pone en uno la operación será normal, desplegándose tanto el mensaje fijo como el móvil.

En los renglones de datos habrá de colocarse el texto del mensaje móvil, cada renglón de datos, excepto el último, deberá iniciar con el caracter "#" en la primera columna seguido por un espacio, el último renglón de datos deberá iniciar con dos caracteres "#" en la primera y segunda columna seguidos por un espacio, a diferencia de los otros módulos que involucran declaraciones de datos, al final de cada renglón de datos no deberá colocarse el caracter ";".

En caso de que un renglón de datos terminara con una palabra completa, deberá colocarse el caracter "|" como último caracter del mismo, de no hacerse esto, en el mensaje móvil aparecerán, sin espaciamiento entre ellas, la última palabra del renglón en cuestión y la primera del renglón subsecuente, por cada caracter "|" colocado al final de un renglón de datos aparecerá, en el mensaje móvil, un espacio entre el último caracter de un renglón de datos y el primero del siguiente, los caracteres espacio que queden en alguna posición intermedia, en los renglones que declaran el contenido del texto móvil, se colocan de manera normal.

Ejemplo: Se desea realizar con el PLM un módulo de tipo mensajero, el cual deberá activarse al iniciar la ejecución en el PLM del programa escrito para el caso, debiendo

ser el texto fijo el siguiente: “PROG1”, el texto móvil debe ser: “Se inició exitosamente la ejecución de PROG1 en el PLM, mientras aparezca este mensaje, no se ha generado ninguna condición de alarma”.

Tanto el texto fijo como el móvil deberán aparecer en el renglón uno, la ventana para el mensaje móvil deberá estar comprendida entre las columnas 7 y 16 (TV = 10), siendo 25 centésimas de segundo el intervalo deseado entre posiciones subsecuentes del mensaje móvil; debiendo el mensajero operar sólo cuando el número que se le asigne coincida con el del módulo desplegador activo (C = 0).

Dado que se requiere que el mensajero inicie su operación al ejecutarse el programa de la aplicación, se le deberá asignar el número cero; por lo tanto, la declaración de este mensajero podría ser:

MENSAJERO#0 “PROG1”, 7, 10, 25, 1101;
 # Se inició exitosamente la ejecución de PROG1|
 # en el PLM, mientras aparezca este mensaje, no se ha generado ninguna condición de|
alarma

En la figura A.253 se presenta la pantalla de UD, 150 centésimas de segundo después de iniciarse la ejecución del programa:

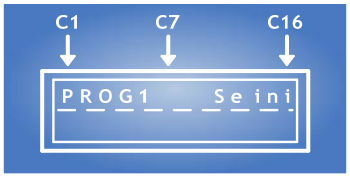


Figura A.253 – Pantalla UD después de la ejecución del programa.

A.6.16 Descripción Del Módulo Auxiliar (MA) Con Capacidad Para Generar Mensajes De Alarma En La Ud Del PLM: Alarma

Este módulo lógico es declarado por el programa compilador en el subprograma principal, siendo la sintaxis para declararlo la siguiente:

ALARMA#NMG XAJAIA, A;

Donde:

- NMG denota el número de módulo de tipo alarma, que debe coincidir con el número de módulo mensajero asociado con el mensaje a desplegar, cuando se verifique la variable definida como entrada al módulo de alarma.
- XA podrá ser la letra “e”, “s” o “i” mayúscula o minúscula, dependiendo esto de que la variable de entrada de activación de mensaje de alarma (A), sea una VBE, VBS o VBI.
- IA denota el número de grupo que corresponda a la VB declarada como entrada “A”.

- JA denota el número de bit dentro del grupo IA, asociado a la variable de entrada “A”.
- A es un dígito binario, que habrá de ser uno, si se desea que el nivel de verificación de la entrada sea alto, en otro caso “A” deberá ser cero.

Ejemplo: Se desea realizar con el PLM un sistema de mensajes de alarma con tres mensajes testigo, todos ellos móviles sin mensaje fijo y desplegados en el primer renglón de la UD a partir de la columna uno, con un tamaño de ventana de 10 caracteres, de modo que el mensaje de mayor prioridad tenga el siguiente texto: “El motor del compresor principal se ha detenido, si no se reinicia su operación, habrá que parar la planta por 24 horas”; el texto del siguiente mensaje en orden de prioridad deberá ser: “La temperatura del interior de la autoclave 4 no es la adecuada, checar el controlador asociado.”; finalmente, el texto del mensaje de mínima prioridad ha de ser: “La pintura en el dosificador A, se ha agotado; por lo tanto, el suministro de pintura se ha conmutado al dosificador B.”; se requiere que el mensaje la condición de no alarma sea: “Operación normal, no se ha dado ninguna condición de alarma”, debiendo el mismo ser móvil, desplegado al igual que los mensajes de alarma, en el renglón uno a partir de la columna uno con un tamaño de ventana de 10 caracteres; se desea que el intervalo entre posiciones subsecuentes de los mensajes sea de 30 centésimas de segundo.

Las VB que testificarían las condiciones de alarma serían en orden de prioridad las siguientes: E23, E12, e I12, siendo las dos primeras verificadas en alto y la última en bajo, los módulos mensajeros tienen asignados, de acuerdo al orden de prioridad los números 3, 5, y 7.

Este ejemplo implica a varios módulos, cuatro mensajeros y tres activadores de mensajes de alarma, de acuerdo a lo explicado en párrafos anteriores los módulos activadores de alarma de este ejemplo se deben colocar en el subprograma principal mientras que los módulos mensajeros deben ser parte del subprograma temporizado a continuación se muestra una posible forma de hacer las declaraciones correspondientes:

Declaraciones de los módulos de tipo de alarma (deben estar en el subprograma principal):

ALARMA#3 E23, 1; Prioridad 1
ALARMA#5 E12, 1; Prioridad 2
ALARMA#7 I12, 0; Prioridad 3

Declaraciones de los módulos mensajeros (deben estar en el subprograma temporizado):

MENSAJERO#5 “”, 1, 10, 30, 1001; Mensaje asociado con la señal de alarma E12.
 # La temperatura del interior de la autoclave 4 no es la adecuada, |
checar el controlador asociado.

MENSAJERO#3 “”, 1, 10, 30, 1001; Mensaje asociado con la señal de alarma E23.
 # El motor del compresor principal se ha detenido, si no se reinicia su operación, |
habrá que parar la planta por 24 horas

MENSAJERO#7 “”, 1, 10, 30, 1001; Mensaje asociado con la señal de alarma I12.

 # La pintura en el dosificador A, se ha agotado; por lo tanto, el suministro de pintura |
se ha conmutado al dosificador B.

MENSAJERO#0 “”, 1, 10, 30, 1001; Mensaje testigo de no condición de alarma.
 # Operación normal, no se ha dado ninguna condición de alarma

Nótese que las declaraciones de los módulos mensajeros, pueden ser hechas no necesariamente en el que corresponda, a la prioridad de los mensajes que portan.

A.7 Resultados Después De La Compilación

Una vez que el proceso de compilación ha terminado, el usuario tendrá como resultado la traducción de su proyecto en lenguaje SIIL1 a través de un archivo con extensión ***.sil**. La interpretación de todo el trabajo realizado se verá reflejado en archivos de texto (incluido el archivo sil), cada uno con una extensión que identifica plenamente su contenido. Estos archivos serán enviados a la carpeta “Archivos_Guardados”, el cual se encuentra ubicado en la carpeta donde se ubico el paquete de instalación.

Los resultados variarán dependiendo de los módulos incrustados en la ZONA DE TRABAJO. Sin embargo, en todos los proyectos se generarán por default dos archivos principales, con extensión ***.mod** y ***.ent** respectivamente, el primero de ellos contiene la información acerca de todos los módulos incrustados en la ZONA DE TRABAJO. El segundo tiene los nombres de las entradas y salidas de los módulos lógicos, así como el número asignado a ellos y en ciertos casos la duración de los pulsos o estados que definen su configuración.

A la hora de guardar un proyecto, ciertos módulos lógicos, por su estructura, **generan por si mismos un archivo**. Esta organización implica que si se generan varios módulos del mismo tipo, todos ellos se enviarán al archivo con la extensión que les corresponda. Para una mayor comprensión la tabla A.254 muestra los módulos que generan sus propios archivos:

MÓDULO LÓGICO	ARCHIVO GENERADO
Secuenciador	“Nombre del Archivo”.sec
TEMPOB	“Nombre del Archivo”.tmb
TEMPOE	“Nombre del Archivo”.tme
TEMPOG	“Nombre del Archivo”.tmg
Mensajero	“Nombre del Archivo”.msj

Tabla A.254 Extensiones de los archivos generados

Así por ejemplo, un proyecto que contenga en la ZONA DE TRABAJO una COMPUERTA AND de dos entradas, un módulo lógico TEMPOA, un módulo SECUENCIADOR, dos módulos lógicos TEMPOB, dos módulos MENSAJEROS y un módulo ALARMA tendrá como resultado los archivos mostrados en la tabla A.255:

MÓDULO LÓGICO	ARCHIVO GENERADO
Todos los Módulos	“Nombre del Archivo”.sil
Todos los Módulos	“Nombre del Archivo”.mod
Compuerta AND	“Nombre del Archivo”.ent
TEMPOA	“Nombre del Archivo”.ent
Secuenciador	“Nombre del Archivo”.sec
TEMPOB#1	“Nombre del Archivo”.tmb
TEMPOB#2	“Nombre del Archivo”.tmb
MENSAJERO#1	“Nombre del Archivo”.msj
MENSAJERO#2	“Nombre del Archivo”.msj
ALARMA	“Nombre del Archivo”.ent

Tabla A.255 Extensiones de los archivos generados

Como se muestra en la tabla A.255 en primera instancia se generan los tres archivos default **(*.sil, *.mod y *.ent**). El archivo ***.sil** generado tiene la traducción del proyecto. El archivo ***.mod** posee la constitución física de los ocho módulos incrustados. El archivo ***.ent** tiene las VB de las entradas y salidas de todos los módulos incrustados, así como algunos datos de configuración de determinados módulos, para este caso, la compuerta AND, el módulo TEMPOA y el módulo ALARMA. Estos módulos se incluyen en este archivo porque sus datos de definición son muy pequeños y no necesitan la creación de un archivo propio.

El archivo ***.sec** tiene los datos de configuración del secuenciador (número de estados). El archivo ***.tmb** tiene los datos de configuración de los módulos TEMPOB (número de pulsos de cada módulo). El archivo ***. msj** tiene los datos de configuración de los módulos TEMPOB (número de pulsos de cada módulo).

BIBLIOGRAFÍA

1. Salva Calleja A. (1999) “Programador Lógico Modular” Tesis M. en I. México DF Universidad Nacional Autónoma de México, Facultad de Ingeniería 339 p.

2. Dunning G. (1998) “Introduction to Programable Logic controllers” Delmar Publishers, United States of America.433 p.

3. http://www.mailxmail.com/curso/informatica/controladores/capitulo3.htm 20 de septiembre de 2007

4. http://www.ing.uc.edu.ve/~emescobar/automat_I/contenido_menu/Unidad_IV/Contenido/pagina3/pagina3.htm 3 de octubre de 2007

5. http://www.mailxmail.com/curso/informatica/controladores/capitulo3.htm 5 de octubre de 2007