Benemérita
Universidad Autónoma de Puebla

Facultad de Ingeniería

Colegio de Ingeniería Mecánica y Eléctrica

“Programación de PLCs en FluidSIM”

Tesina

Que para obtener el Titulo de

Ingeniero Mecánico y Eléctrico

Presenta:

Marco Antonio Toledo Arellano

Asesor: M.C. Guillermo Flores Martínez

PUEBLA, PUE. ENERO 2014

Agradecimientos

A mis Padres, hacerlos sentir orgullosos es lo único que me motivó, este logro es de Ustedes.

A mis hermanas, Fátima y Carolina, que me aconsejaron y apoyaron con todo lo que a ellas les faltó durante su formación académica.

Al resto de mi familia, que con sus palabras de aliento me indicaban que iba por el camino correcto.

Al M.C. Guillermo Flores Martínez, por su completo e incondicional apoyo para realizar este documento.

Gracias por hacer esto posible.

Índice

Introducción

El presente documento titulado “Programación de PLCs en FluidSIM” es una guía para poder realizar programaciones de un PLC en el software FluidSIM[1] de Festo[2].

Tiene el propósito de explicar de forma introductoria una herramienta para el aprendizaje de la automatización mediante PLC.

El primer capítulo nos brinda todas las definiciones y teoría básica relacionada con automatización moderna, también se muestran los lenguajes de programación más ocupados en un PLC, con sus respectivas características, con el fin de saber diferenciar una técnica de otra.

El segundo apartado es dedicado completamente al software FluidSIM, explicando sus características más esenciales como: nombres de herramientas, opciones de edición, biblioteca de componentes, etc. Después se detallarán las opciones más avanzadas de simulación, edición, configuración de los componentes disponibles y programación de un PLC dentro del software.

En el tercer capítulo se realizaran ejemplos con aplicaciones reales de la industria, donde se ocupara todo lo explicado en apartados anteriores.

Este tercer capítulo será un tanto comparativo, ya que los ejemplos los realizaremos de dos formas, una con las técnicas clásicas de automatización y la otra con los métodos modernos que utilizan el PLC como eje principal de control.

Finalmente podremos analizar los resultados con base a la comparativa realizada, las ventajas y desventajas de cada método quedarán expuestas y de esta forma se validará lo propuesto durante todo el documento.

[1] FluidSIM es una marca registrada y está protegida bajo derechos de Autor
[2] Festo es una marca registrada y está protegida bajo derechos de Autor

Capítulo I: Programación PLC

1.1 ¿Qué es un PLC?

Un Controlador Lógico Programable, más conocido por sus siglas en inglés PLC (Programmable Logic Controller), también conocido como “Autómata”, es una computadora utilizada en la industria destinada a gobernar máquinas o procesos lógicos y/o secuenciales.

Surgen como solución más eficiente para reemplazar los sistemas de control basados en circuitos eléctricos con relevadores, interruptores y otros componentes comúnmente utilizados para el control de los sistemas de lógica combinacional.

En el automatismo clásico, la función de un proceso reside en el cableado de los elementos de maniobra, es decir, poniendo en serie o en paralelo contactos de cierre o apertura. Por lo tanto, una modificación exige nuevos componentes, cambios de cableado, trabajos de montaje e incremento económico. Todo esto sin contar las pérdidas de tiempo en las pruebas y sincronización. (Cembranos Nistal, 1998)

En un PLC, esta función se realiza por programa o algoritmo, es decir, mediante una serie de instrucciones que le dicen que contactos debe abrir, cuáles debe cerrar, retardos, contadores, etc. Todo el proceso de mando está almacenado en la memoria del PLC. (Cembranos Nistal, 1998)

De igual forma como se introdujo los comandos de programación, podemos realizar cualquier modificación de la misma, sin cambiar cableado, en el mismo lugar de la instalación e incluso, durante el proceso de trabajo.

La estructura de los PLCs es diferente en cada fabricante, sin embargo, existen elementos base en todos ellos. En la figura 1.1 se puede observar el aspecto dos PLCs, uno marca Siemens[3] y el otro Allen Bradley[4].

[3] Siemens es una marca registrada y está protegida bajo derechos de autor
[4] Allen Bradley es una marca registrada y está protegida bajo derechos de autor

Está formado por una carcasa, generalmente, de plástico que alberga todos los componentes electrónicos, estos son (Cembranos Nistal, 1998):

- **MICROPROCESADOR:** Es el encargado de dirigir el funcionamiento del PLC, recibe las órdenes del programa y lo ejecuta.
- **MEMORIA:** Es el lugar donde se almacena el programa, este se irá leyendo instrucción a instrucción a medida que se ejecute.
- **ENTRADAS/SALIDAS:** Conexiones para comunicarse con el exterior, reciben señales externas para procesar o activar salidas en función del proceso del programa.
- **FUENTE DE ALIMENTACIÓN:** Es el que energiza todo el conjunto.

Figura 1.1. Aspecto de un PLC, SIEMENS izquierda (AutomatykaOnline), Allen Bradley derecha (Southland Electrical).

A pesar de que el funcionamiento interno, la transmisión de información y la programación de los PLCs se efectúa en forma digital, un PLC nos lee señales tanto digitales (un interruptor en ON/OFF) como analógicas (un sensor de temperatura).

Además puede enviar órdenes de mando a: contactores de motores, válvulas magnéticas, frenos electromagnéticos, lámparas de señalización, etc. Independientemente puede contar impulsos, almacenar señales, prefijar desarrollos temporales, etc., y todo esto conectado una red informática para enviar y recibir datos de la red.

La información o señales en forma digital utiliza variables discretas, esto es, variables que solo pueden tomar un numero finito de valores (en este caso 1 o 0), al contrario de la forma analógica que puede utilizar un número infinito de valores posibles (un rango de temperatura)

Para muchas aplicaciones, la utilización de señales e información de tipo digital proporciona una mayor precisión que la analógica, además que son menos sensibles al ruido electromagnético, las aplicaciones electrónicas son más sencillas de realizar y con un tamaño mucho menor que las analógicas. (Cembranos Nistal, 1998)

Todas estas características le permiten ser incluido en tareas de automatización en todas las ramas de la industria, tales como la electrónica, mecánica, automoción, alimentación, petroquímica, construcción de máquinas, etc.

1.2Técnicas de programación

La programación de un PLC se realiza por medio de periféricos que se conectan directamente a la memoria del aparato para grabar las instrucciones deseadas. En la actualidad la mayoría de la programación se lleva a cabo mediante una PC, en la cual se utiliza el software designado de la marca del PLC, Aunque a la fecha se siguen utilizando pequeñas consolas de programación para dicha tarea.

Independientemente del periférico que se utilice para la programación, hay pasos y técnicas que se tienen que respetar, por practicidad y por estandarización.

Una vez que se tenga planeado lo que se va a automatizar, se tiene que elaborar un diagrama de flujo o pasos que vamos a seguir durante la programación, esto con la finalidad de lograr simplificar los comandos utilizados.

Teniendo lo anterior se tiene que elegir uno de los tres lenguajes de programación más comunes:

- Diagrama de Contactos
- Lista de Instrucciones
- Diagrama de Bloque de Función

1.2.1 Diagrama de Contactos

Mejor conocido por sus siglas en alemán KOP (Kontaktplan), LD en ingles (Ladder Diagram) o Diagrama de escalera, es un lenguaje de programación grafico muy popular dentro de los PLCs debido a que está basando en diagramas eléctricos de control.

Los primeros PLCs fueron programados en KOP con la finalidad de integrar más rápidamente a los técnicos a las tareas de programación, evitando contratar personal especializado en esta disciplina.

“Su principal ventaja es que los símbolos básicos están normalizados según el estándar IEC (International Electrotechnical Commission) y son empleados por todos los fabricantes.” (Universidad Nacional de La Plata)

Los símbolos básicos son:

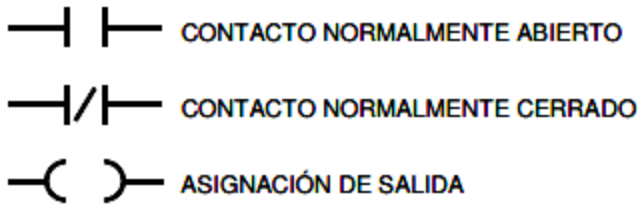

Figura 1.2. Símbolos Básicos (Universidad Nacional de La Plata)

En estos diagramas la línea vertical a la izquierda representa un conductor conectado a tensión, y la línea vertical a la derecha representa la conexión a tierra aunque algunos software no ilustran esta ultima dando por hecho que el último elemento a la derecha estará conectado a tierra.

Con este tipo de diagramas se describe la operación eléctrica de distintos tipos de máquinas, y puede utilizarse para sintetizar un sistema de control y, con las herramientas de software adecuadas, realizar la programación del PLC.

A diferencia de un diagrama eléctrico, donde todas las acciones ocurren simultáneamente, un programa de PLC las realiza en forma secuencial, siguiendo el orden en el que los "escalones" o “pasos” fueron escritos, y que al contrario de los relés y contactos reales (cuyo número está determinado por la implementación física de estos elementos), en el PLC se puede considerar que existen infinitos contactos auxiliares para cada entrada, salida, relé auxiliar, temporizador, etc. (Universidad Nacional de La Plata)

Los elementos que evalúan si activar o no las salidas en determinado "escalón", son variables lógicas o binarias, que pueden tomar solo dos estados: 1 ó 0 (HI-LO, Alto-Bajo). Estos estados provienen desde las entradas físicas al PLC o relés internos del mismo.

En el lenguaje de Diagrama de Contactos, estas variables están representadas justamente por contactos, que pueden estar en solo dos estados: abierto o cerrado.

Los contactos, dependiendo de la marca comercial del PLC, se pueden representar con la letra “I” o "E" y dos números que indicaran el Módulo al cual pertenecen y el borne de la entrada a la cual están asociados.

Ejemplo: I0.3 =Entrada del Módulo “0”, Borne “3”.

Cabe aclarar que los contactos abiertos se cerraran al ser activados, y de forma inversa, los contactos cerrados se abrirán, como se muestra en la Figura 1.3.

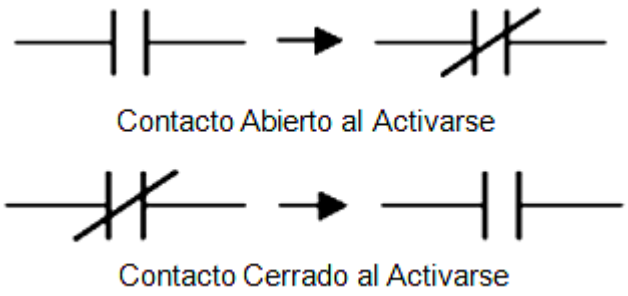

Figura 1.3. Comportamiento de los Contactos. (Modificado de Universidad Nacional de La Plata)

Las salidas de un programa en lenguaje KOP son equivalentes a las cargas (bobinas de relés, lámparas, etc.) en un circuito eléctrico. Se les designa la letra "Q", "O" u otra letra, dependiendo de los fabricantes, y dos números que indicaran el Módulo al cual pertenecen y el borne de la salida al cual están asociados.

Ejemplo: Q0.1 = Salida del Módulo "0", Borne "1"

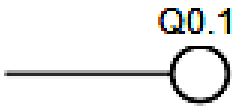

Figura 1.4. Salida en lenguaje KOP (Autoría Propia)

En el programa del PLC, no solo se toman las salidas que el equipo posee físicamente, sino también los "Relés Internos o Marcas". Los relés internos son variables lógicas virtuales que se pueden usar, por ejemplo, para memorizar estados o como acumuladores de resultados.

Se las identifica con la letra "M" y un número el cual servirá para asociarla con algún otro escalón del programa.

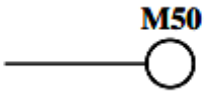

Figura 1.5. Marcas o Relevadores Internos. (Universidad Nacional de La Plata)

Ejemplo, en la Figura 1.6 se puede apreciar que el estado de la salida Q0.0 depende directamente de la entrada M1, la cual a su vez depende de la salida M1 que se activa mediante la entrada I0.0.

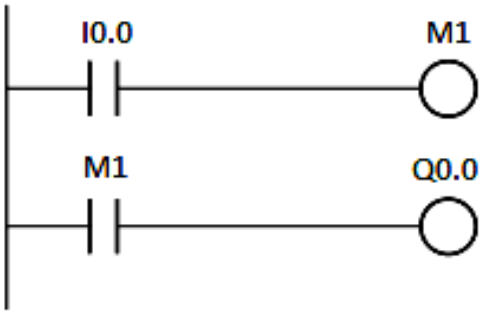

Figura 1.6. Utilización de Marcas. (Modificado de Universidad Nacional de La Plata)

Otras instrucciones importantes dentro del lenguaje KOP son las de “SET” y “RESET”. La instrucción “SET” activa la salida correspondiente cada vez que enviamos un impulso, y sólo se desactivará al enviar otro a la instrucción “RESET”. Podemos activar tanto salidas como marcas internas. Para cada “SET” corresponde un “RESET”. Dependiendo de la aplicación, es una forma más segura de programar el enclavamiento eléctrico.

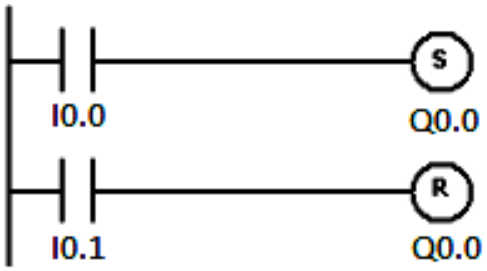

Figura 1.7. Configuración de función SET-RESET. (Modificado de Universidad Nacional de La Plata)

Las funciones lógicas más complejas como temporizadores, contadores, registros de desplazamiento, etc., se representan en formato de bloques. Estos no están normalizados, aunque guardan una gran similitud entre sí para distintos fabricantes.

Según la Universidad Nacional de La Plata, en estos bloques se define la base de los tiempos y el tiempo final en el caso de temporizadores, el módulo de contaje y condiciones de paro y restablecimiento en el caso de contadores.

También existen bloques con funciones complejas que permiten la manipulación de datos y las operaciones con variables digitales. Solo los PLC de gama alta pueden manejar todas las posibilidades de programación con contactos.

Durante la programación de un PLC, hay reglas que seguir, por el simple hecho que de no hacerlo, el programa no funcionará.

Los nombramientos de las salidas no se pueden repetir en los diferentes “escalones” del programa. Al contrario de las salidas, los nombramientos de entradas podrán repetirse siempre que lo ameriten.

Las salidas pueden ir precedidas de contactos, pero no pueden estar seguidas por ninguno. Ver Figura 1.7.

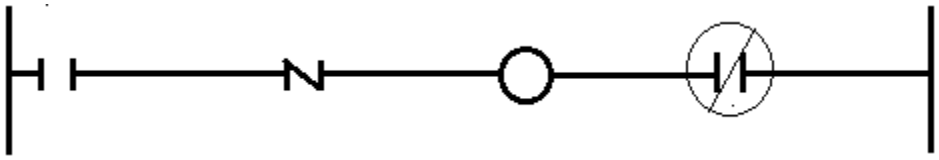

Figura 1.7. Configuración de Contacto Errónea. (Autoría Propia)

Lo mismo se aplica a los bloques Función, como un bloque de transferencia o un Temporizador, ya que se comportan como salidas.

Sin embargo hay una conexión que es posible en el Lenguaje de Escalera pero imposible en un tablero. Las salidas pueden ser conectadas en Serie comportándose en forma similar que si estuvieran en paralelo. Ver Figura 1.8.

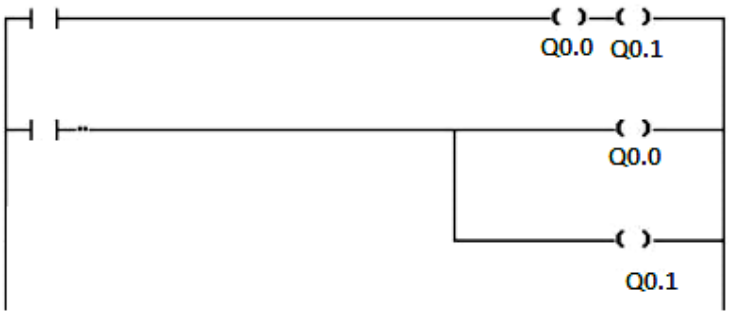

Figura 1.8. Bobinas en Serie y Paralelo. (Autoría Propia)

Se debe prestar atención al orden en que se ubican las salidas respecto a las entradas que las activan.

Por ejemplo:

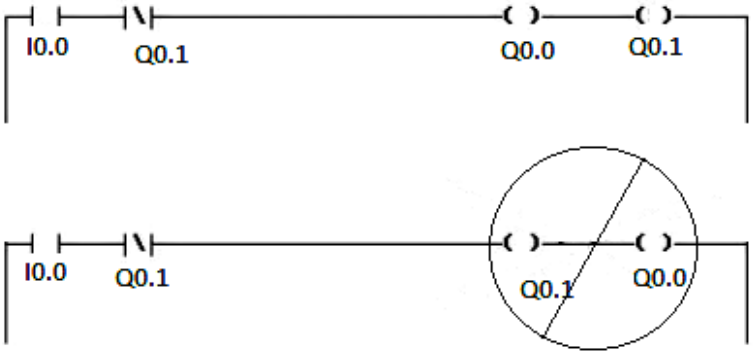

Figura 1.9. Orden de Bobinas. (Autoría Propia)

En la Figura 1.9, si adoptáramos la segunda alternativa de conexión, sucederá que una vez actuada la salida Q0.1, ya nunca se activaría la salida Q0.0, dado que el contacto cerrado de la salida Q0.1 quedaría abierto.

Si las salidas son conectadas directamente a tensión, entonces se les considera permanentemente activadas. En algunos PLC esto está prohibido.

Por último, no hay que olvidar que ningún elemento de la programación, sea contacto, bobina de salida, contador, etc., podrá quedarse sin nombramiento, de lo contrario producirá una inconsistencia dentro del algoritmo, causando a su vez una falla en el proceso. Ciertos software de programación no permiten dejar sin nombramiento ningún elemento.

1.2.2 Lista de instrucciones

Mejor conocido por sus siglas en alemán AWL (Anweisungsliste) o IL en ingles (Instruction List), es nivel más fundamental de lenguajes de programación definidos por los estándares del IEC. Consiste en elaborar un listado de instrucciones "Booleanas" escritas (Operaciones Lógicas con Bits) que se asocian a los símbolos y contactos de un diagrama eléctrico de control, y las cuales representarán la combinación lógica que exista entre dichos contactos. (Navarro G., 2001)

En realidad, internamente todos los PLCs trabajan con AWL sin importar bajo que lenguaje haya sido programado, esto permite crear programas con optimización de tiempo de ejecución y espacio de almacenamiento.

Las Operaciones Lógicas con bits operan con dos dígitos, 1 y 0. Que son la base del sistema numérico binario. Interpretan los estados de señal 1 y 0 (HI-LO o Alto- Bajo), y los combinan de acuerdo con la lógica de Boole. (Siemens AG, 2010)

De acuerdo con Siemens, estas combinaciones producen un 1 ó un 0 como resultado y se denominan "Resultado Lógico" (RLO). Eléctricamente hablando, un 1 significa activado (conductor) y un 0 significa desactivado (no conductor).

Las operaciones básicas para las operaciones lógicas con bits son:

- “A” – Que es una abreviación de “And”, en ocasiones se llega a ocupar “U” de su abreviación “Und”, una del Inglés y la otra del Alemán respectivamente, ambas significan “Y” en español.
- “AN”- Es la abreviación de “And Not” que en español significa “Y negada”, también se llega a ocupar “UN” al igual que el caso anterior.
- “O”- Es la abreviación de “Or”, que español significa “O”, se refiere a “una opción u otra”.
- “ON”- Abreviación de “Or Not”, en español se puede interpretar como “O negada”.

Existen otros operadores no lógicos que son muy importantes dentro de este lenguaje de programación:

- “LD”- Es la abreviación de “Load”, que en español significa “Cargar”. Todo programa, acción o paso dentro del algoritmo de programación debe iniciar con este operador. Desde el punto de vista de diagrama en escalera, este comando inserta un contacto normalmente abierto.
- “LDN”- “Load Not”, “Cargar Negado” en español, es muy similar a “LD” solo que este operador inserta un contacto normalmente cerrado.
- “=” – Denota una asignación, esta operación implica que el RLO se almacene en el operando.

Los programas creados en AWL se componen de los siguientes elementos (Cano):

- Etiqueta (Opcional)
- Operador
- Modificador (Si se requiere)
- Operando (Entradas, Salidas, Marcas)
- Comentario (Opcional)

Etiqueta	Operador y Modificador	Operando	Comentario
Start	**LD**	**I0.0**	**Botón Pulsador**
	=	**Q0.0**	**Encender Motor**

Tabla 1.1. Elementos del Lenguaje AWL. (Modificado de Cano)

Estas sencillas instrucciones ejemplifican el arranque de un motor con un contacto normalmente abierto y su salida. Para nombrar las entradas, salidas, marcas, etc., se utilizan el mismo sistema que en KOP .Las instrucciones anteriores en lenguaje de Diagrama de Contactos se representan de la siguiente manera:

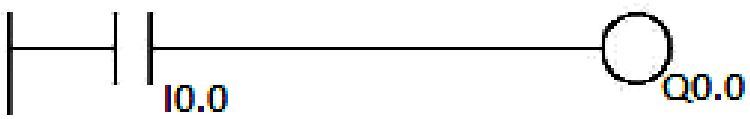

Figura 1.10. Equivalencia de la Tabla 1.1. (Autoría Propia)

Cada uno de los operadores lógicos tiene una función diferente, la cual emula la relación que tienen los contactos en un diagrama eléctrico, es por eso que todo programa hecho con lista de instrucciones puede tener un equivalente en diagrama de escalera. A continuación de describe las funciones de los operadores lógicos básicos:

- AND: Implica que se va a realizar una operación dependiente entre el operando de la instrucción anterior y el operando que se le asigne al mismo operador AND, es decir, establecerá una condición donde se necesitarán ambos operandos activos para realizar una tarea. Desde el punto de vista del Diagrama de Contactos, AND agrega un contacto normalmente abierto.

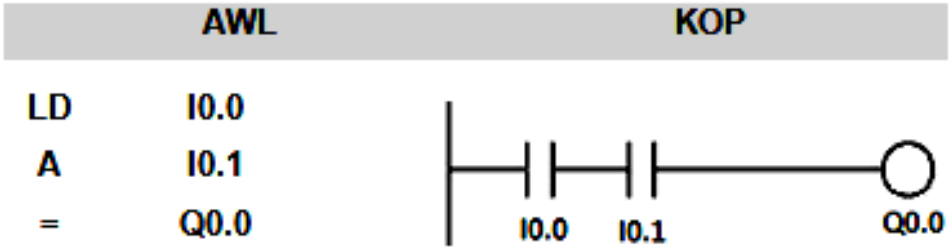

Figura 1.11. Operación AND. (Autoría Propia)

- AND NOT: Básicamente realiza la misma función que “AND”, pero al contrario de la anterior, esta agrega un contacto normalmente cerrado.

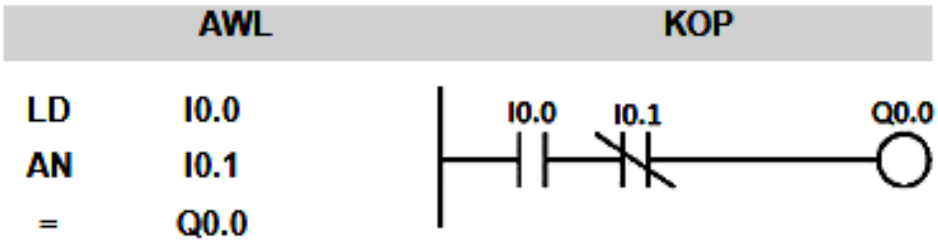

Figura 1.12. Operación AND NOT. (Autoría Propia)

- OR: Implica que se va a realizar una operación no dependiente entre el operando de la instrucción anterior y el operando que se le asigne al mismo operador OR, es decir, establecerá una condición donde se necesitarán uno u otro de los operandos activos para realizar una tarea.

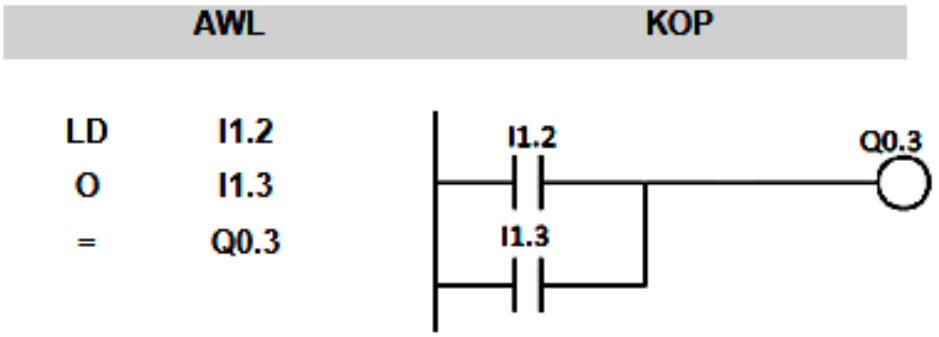

Figura 1.13. Operación OR. (Autoría Propia)

- OR NOT: Establece las mismas condiciones que la compuerta OR con la excepción que esta será con un contacto normalmente cerrado.

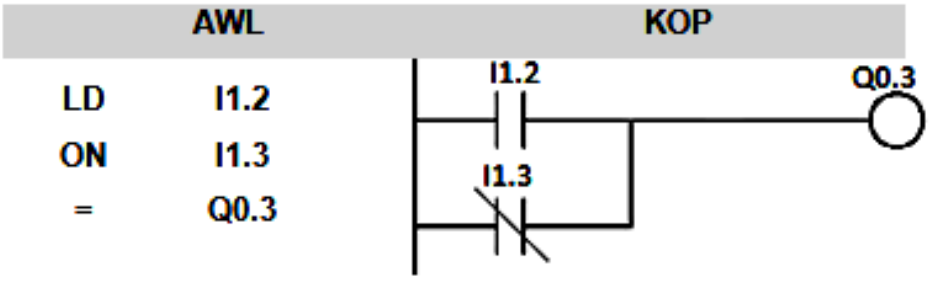

Figura 1.14. Operador OR NOT. (Autoría Propia)

Si a esta información se le da un enfoque puramente eléctrico, podríamos llegar a la conclusión que los operadores “AND” y “AND NOT” establecen condiciones en “serie” de los contactos de un diagrama eléctrico, mientras que “OR” y “OR NOT” establecen las condiciones en “paralelo”.

En los ejemplos de la Figura 1.15 se puede apreciar cómo se combinan estos operadores para crear instrucciones más complejas.

Para estos casos, no hay una regla absoluta que nos indique como combinar los operadores, en ocasiones hay más de una forma de programar la misma instrucción. A pesar de esto, vamos a plantear un método para las ya mencionadas combinaciones de los operadores lógicos.

Para instrucciones que requieran varios contactos, se recomienda dividirlos en grupos, estos grupos empezaran con el operador “LD” y se usaran los otros 4 operadores lógicos para agregar los contactos necesarios.

En el último ejemplo de la Figura 1.15 se puede ver que se usó “LD” para insertar un contacto normalmente abierto, en seguida el operador “A” insertó un contacto normalmente abierto en serie con el anterior, de esa forma se cerró el primer grupo.

Dentro de la lista de instrucciones se volvió a escribir “LD” para insertar un nuevo contacto normalmente abierto, que indicó la apertura de un nuevo grupo, después se escribió el operador “A” que insertó otro contacto normalmente abierto en serie con el anterior.

De esta forma se obtuvo dos grupos de dos contactos conectados en serie, pero para poder emular lo que se tiene en lenguaje KOP estos dos grupos tienen que estar en paralelo, es por esta razón que se ocupo el modificador “OLD” que aplica la operación “OR” los grupos “LD”. Finalmente se asignó estas operaciones a las salidas.

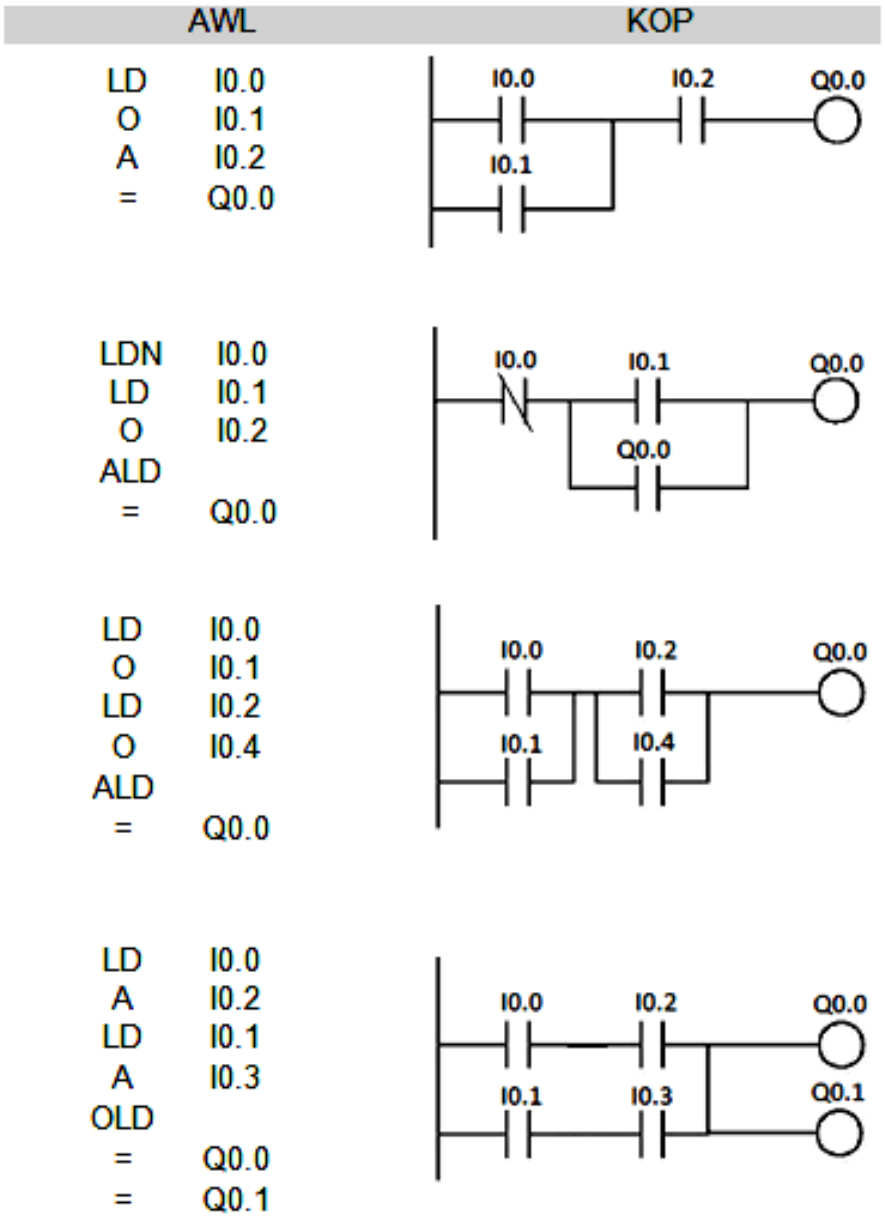

Figura 1.15. Combinación de Operadores. (Modificado de Navarro G., 2001)

Otro modificador ocupado en los ejemplos es “ALD” que aplica la operación “AND” a los grupos “LD” formados. Tanto “OLD” como “ALD” pueden ir más de una vez en una lista de instrucciones. También cabe aclarar que los grupos antes mencionados pueden ser formados por más de dos contactos (normalmente abiertos o cerrados).

Con el lenguaje lista de instrucciones también se pueden representar instrucciones más complejas, como contadores, temporizadores, trasferencias de datos, etc., pero estas no están normalizadas y la forma de ingresarlos en los programas dependerá de cada fabricante, en la Figura 1.16 se ejemplifica para un software de Siemens:

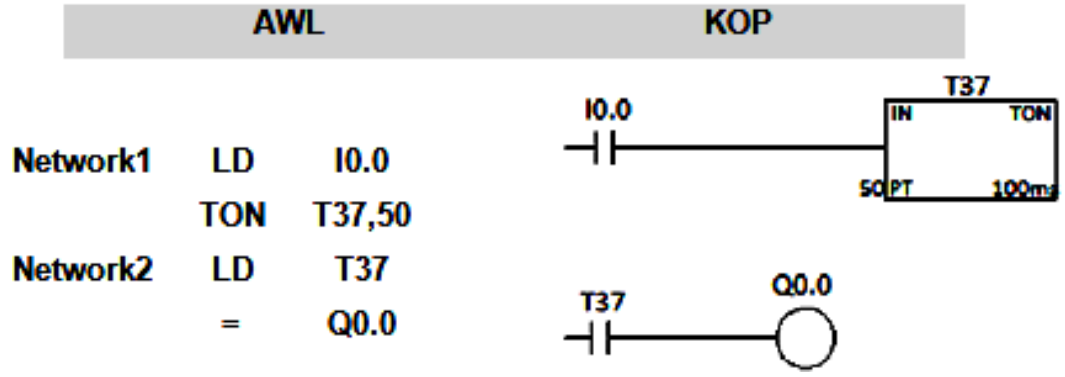

Figura 1.16. Temporizador en Software de Siemens. (Autoría Propia)

En este caso especifico, la función del temporizador y el accionamiento que este va a realizar se escriben en dos escalones diferentes del programa, “TON” es un temporizador a la conexión, “T37” es el nombre que se le da para asignar la unidad de tiempo, el numero “50” es la magnitud de tiempo, el contacto normalmente abierto “I0.0” envía un impulso a la entrada “IN” del temporizador para activarlo, mientras que el contacto “T37” será activado por el mismo temporizador al finalizar su conteo.

En este lenguaje básicamente se tienen las mismas restricciones que en el lenguaje de diagrama de escalera, puesto que los programas hechos bajo AWL tienen equivalente en KOP.

1.2.3 Diagrama de Bloques de Función

Más popular por sus siglas en alemán FUP (Funktionsplan) o en ingles FBD (Function Block Diagram), es un lenguaje de programación orientado a gráficos.

Es una representación grafica de dispositivos físicos que trabajan mediante redes, donde cada una de ellas representa una expresión lógica o aritmética, permite construir procedimientos complejos mediante la unión de bloques funcionales prediseñados que aparecen como circuitos integrados.

Generalmente utilizan Compuertas Lógicas para representar al bloque de función. Es muy común en aplicaciones que implican flujo de información o datos entre componentes de control.

El diagrama Bloques de Función, resulta especialmente cómodo de utilizar a técnicos habituados a trabajar con circuitos de Compuertas Lógicas, ya que la simbología usada en ambos es equivalente. (Universidad Nacional Abierta y a Distancia)

El objetivo de este documento exige que este lenguaje se explique de manera más detallada, es por eso que se extenderá las características del mismo en el siguiente apartado.

1.3 Compuertas Lógicas como Lenguaje de Programación

La teoría base del lenguaje de programación mediante Bloques de Función es la misma correspondiente a las Compuertas Lógicas, por lo que es necesario comprenderla para realizar una correcta programación en este lenguaje.

Es importante aclarar que a pesar que los nombres de las Compuertas Lógicas son iguales a los operadores del lenguaje de Lista de Instrucciones, su equivalente en KOP no es necesariamente el mismo, o dicho de manera, el software que se ocupe para programar leerá de distinta manera una instrucción “AND” en FUP que una en AWL aun cuanto estas teóricamente son lo mismo.

Como se ha hecho referencia anteriormente, las compuertas lógicas se basan en el algebra de Boole, la cual maneja variables que representan proposiciones que pueden adoptar dos valores, uno verdadero y otro falso. El nombre de “Circuitos Lógicos”, “Compuertas Lógicas” o “Puertas Lógicas” generalmente se aplica a los elementos que funcionan bajo este principio. (Cembranos Nistal, 1998)

El álgebra de Boole opera entonces con variables que admiten exclusivamente dos valores que, basados en sistema binario, se designan por 0 y 1. Hay que tener presente que estos símbolos no representan números en este caso, sino dos estados que puede tener un dispositivo: HI-LO, Alto-Bajo o Verdadero-Falso.

La referencia anterior explica que en un sistema eléctrico digital, un bit (0 ó 1) se caracteriza por representar uno de los dos niveles de tensión. Si la tensión más positiva es representada por 1 y la otra por 0, se dice que el sistema maneja lógica positiva. De forma inversa, si la tensión más negativa se representa con 1 y la más positiva con 0 se dice que emplea lógica negativa. Para definiciones y ejemplos de este documento se ocupo lógica positiva.

Cembranos Nistal aclara no confundir los valores absolutos de las tensiones ocupadas con los estados del dispositivo, por ejemplo, el estado 0 de un dispositivo no corresponde necesariamente a un nivel 0 volts de tensión, aunque en algunos casos pueda coincidir.

Dentro del álgebra de Boole se definen tres tipos de operaciones básicas:

- **Adición**: Utilizando las variables A y B, se representaría como: A+B.
- **Producto**: Supóngase A y B como variables, se representaría como: AB
- **Complementación**: Usada en una sola variable. Si la variable es A, se representa como: A’

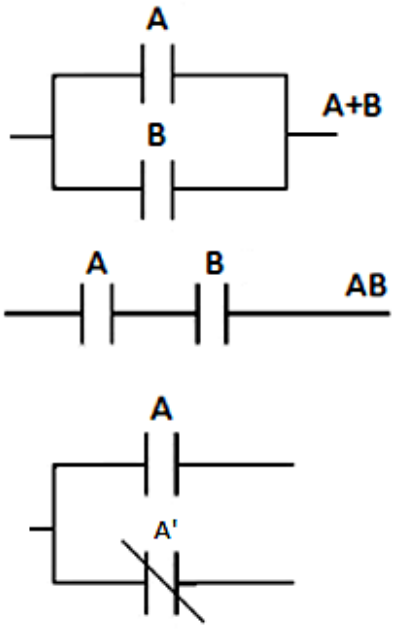

Figura 1.17. Representación de Operaciones Básicas mediante contactos. (Autoría Propia)

Con el objetivo de visualizar de manera más familiar estas operaciones, podemos ver las variables A y B como el estado de dos contactos (Ver Figura 1.17). Si un contacto se encuentra en estado 1 representa que está cerrado; y si el estado es 0 el contacto estará abierto.

El arreglo en paralelo de la figura anterior simboliza la “Adición”, esto quiere decir que existirá continuidad en el extremo derecho si A o B valen 1(están cerrados). El segundo arreglo que está en serie representa el “Producto” ya que, para que exista continuidad en el extremo derecho, las variables A y B tienen que valer 1. Finalmente la complementación se simboliza mediante la forma de contactos complementarios de un mismo interruptor, de modo que, si uno está cerrado, el complementario estará abierto. (Cembranos Nistal, 1998)

1.3.1 ¿Qué son las Compuertas Lógicas?

Una Compuerta Lógica, o Puerta Lógica, es un dispositivo electrónico físico que expresa a un operador booleano en la lógica de conmutación. Cada Compuerta Lógica consiste en una red de dispositivos interruptores que cumplen las condiciones booleanas para un operador en particular. Son esencialmente un conjunto de transistores integrados en un chip.

“Hay disponible una gran variedad de compuertas estándar, cada una con un comportamiento perfectamente definido, y es posible combinarlas entre sí para obtener funciones nuevas. Desde un punto de vista práctico, podemos considerar a cada compuerta como una caja negra, en la que se introducen valores digitales en sus entradas, y el valor del resultado lógico aparece en la salida”. (Palazzesi, 2007)

Cada compuerta tiene asociada una tabla de verdad, que es una forma de representar la función lógica a la que corresponde, se expresa en forma de lista los estados de su salida para cada combinación posible de estados en la(s) entrada(s).

Tienen dos principales simbologías, una de ellas es la tradicional la cual es la más ocupada a pesar de no estar normalizada y la otra es la que establece la IEC, que es menos ocupada a pesar de su carácter oficial.

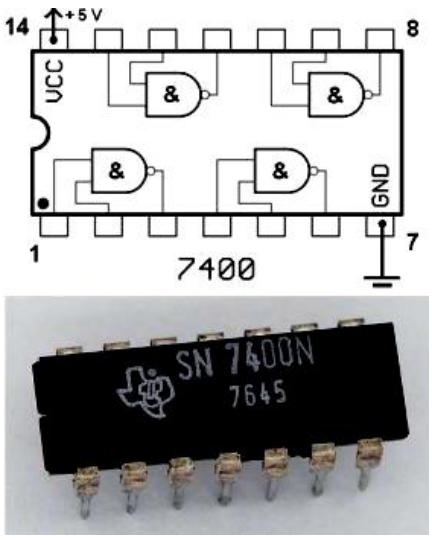

Figura 1.18. Circuito integrado con cuatro compuertas NAND. (Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado)

1.3.2 Compuertas Lógicas Básicas

Existen tres compuertas Lógicas Básicas que representan a las tres operaciones básicas del álgebra Booleana. Realizan las operaciones más elementales que son la base de todas las demás Compuertas existentes.

Antes de abordar las antes mencionadas, es necesario mencionar la existencia de una compuerta igual de importante, con la diferencia de que esta no es “Lógica”.

La compuerta “**IF**” o “**Buffer**” realiza la función booleana de la igualdad. Se simboliza mediante un triangulo, cuya base corresponde a la entrada, y el vértice opuesto la salida.

Entrada A	Salida A
0	**0**
1	**1**

Tabla1.2. Tabla de Verdad de la compuerta “IF”. (Autoría Propia)

La interpretación de su tabla de verdad es que la salida toma siempre el valor de la entrada. Esto significa que si en su entrada hay un estado en 1, también lo estará en su salida; y si la entrada se encuentra en un estado 0, su salida también estará en ese estado. Ver Tabla 1.2.

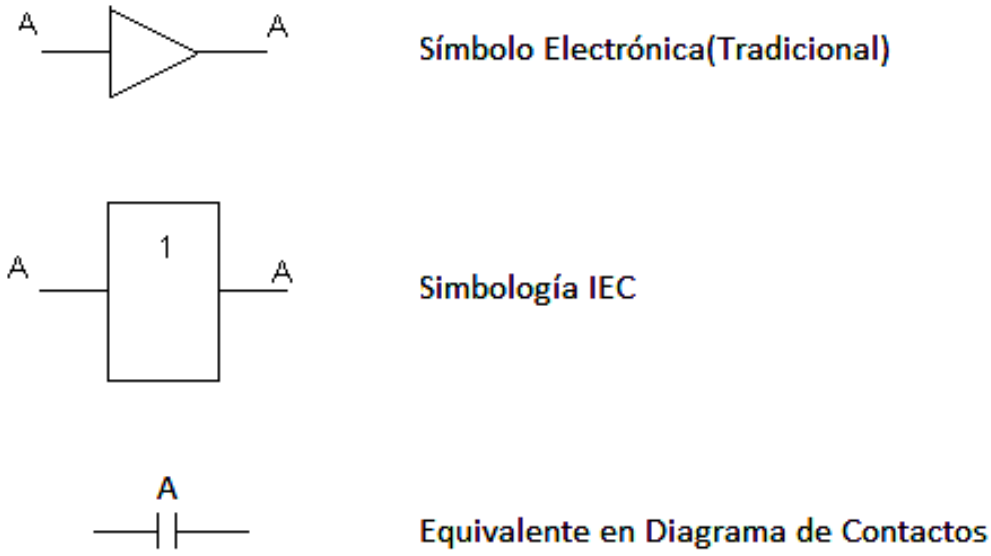

Figura 1.19. Simbología Compuerta "IF". (Autoría Propia)

Hablando sobre la programación utilizando Compuertas, es muy importante entender que, a pesar que "IF" podría representarse como un contacto normalmente abierto, este por sí solo no realiza ninguna tarea, es solo la forma en que podemos representar la estructura interna de la Compuerta por medio de contactos.

Es por eso que dentro del software de programación hay que agregar otros elementos para que la compuerta pueda realizar la función deseada.

En el caso de "IF", hay que agregar otro contacto normalmente abierto en la entrada, que es el que podremos cerrar (1) y abrir (0) forzando al contacto interno a cambiar a su estado inverso, la señal resultante que se refleje en la salida de la compuerta se conecta a una salida del propio PLC que es la que activará otros dispositivos. (Ver diagrama 1 de la Figura 1.20).

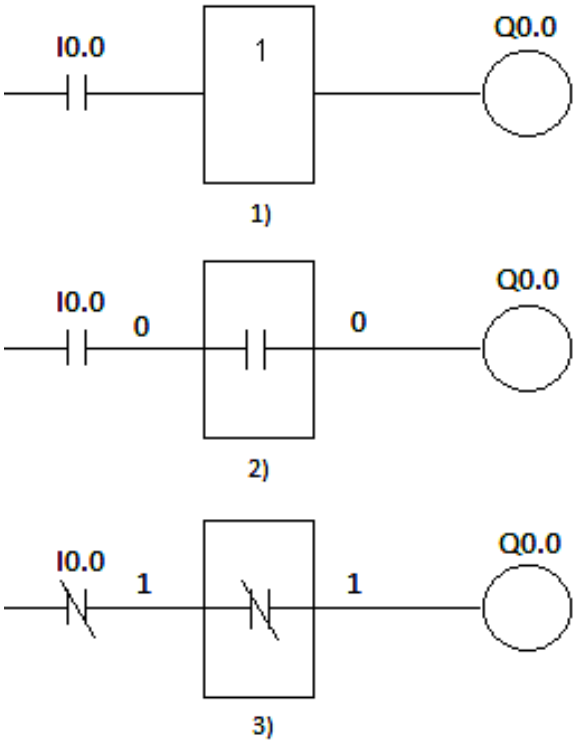

Figura 1.20. Arreglo y funcionamiento de la Compuerta "IF. (Autoría Propia)

El diagrama 2 de la figura anterior, muestra que el contacto correspondiente a la entrada del I0.0 se encuentra abierto, es decir, en estado 0, dicho estado interactúa con el arreglo interno de la compuerta, dando como resultado un estado 0 en la salida del la misma, en contraste, el diagrama 3 muestra que cuando I0.0 se manda a cerrar (estado 1), la estructura interna cambia a su estado inverso 1 , dicho de otra manera, en el momento que se manda a cerrar I0.0 el contacto interno de la compuerta se cierra también, dando un resultado 1 en la salida, de esta forma se cumple lo establecido en la Tabla 1.2.

Para aplicaciones de automatización, muchas veces el estado 0 significa un nivel de tensión 0 volts, mientras que el estado 1 representa un nivel de tensión que permita realizar accionamientos, normalmente 24 volts.

La Compuerta Lógica "**NOT**" representa la operación Booleana "Complementación" que nos indica que cada contacto tendrá un complementario que estará en estado opuesto, es por eso que también se le conoce como "Inversor" y su función es la negación.

Esta compuerta tiene en su salida un valor que es el inverso al presente en su única entrada (Tabla 1.3). Se usa cuando es necesario tener un valor lógico opuesto a uno dado. Su símbolo es el mismo que la compuerta “IF”, con un pequeño círculo agregado en su salida, que representa la negación. Ver Figura 1.21.

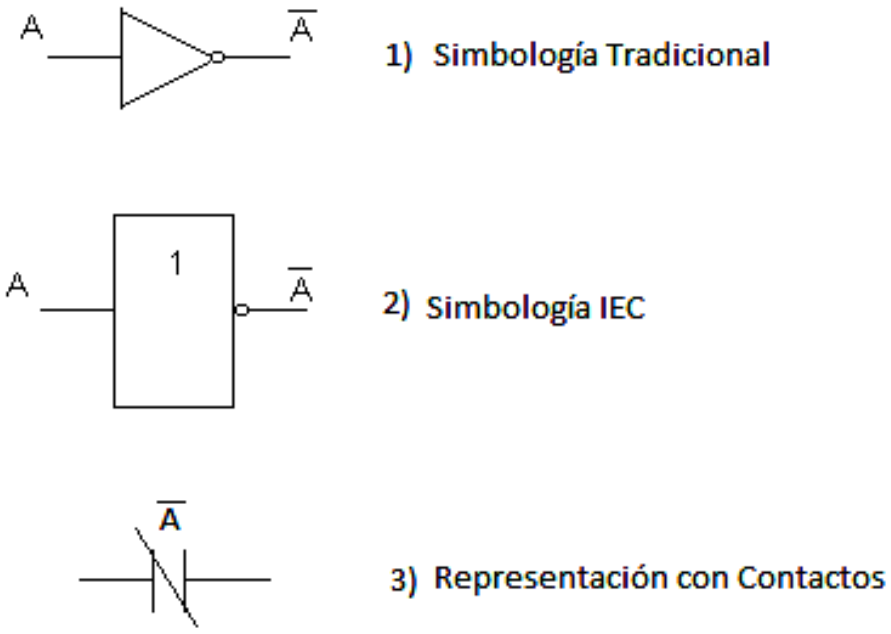

Figura 1.21. Compuerta Lógica “NOT. (Autoría Propia)

Entrada	Salida
0	**1**
1	**0**

Tabla 1.3. Tabla de Verdad de la Compuerta “NOT”. (Autoría Propia)

Para agregar esta Compuerta a un programa de PLC en lenguaje FUP, al igual que “IF”, es necesario incluir los elementos que la harán funcionar. Son necesarios un contacto normalmente abierto en su entrada y en su salida un conexión a la salida o una Marca del PLC como se muestra en el diagrama 1 de la Figura 1.22.

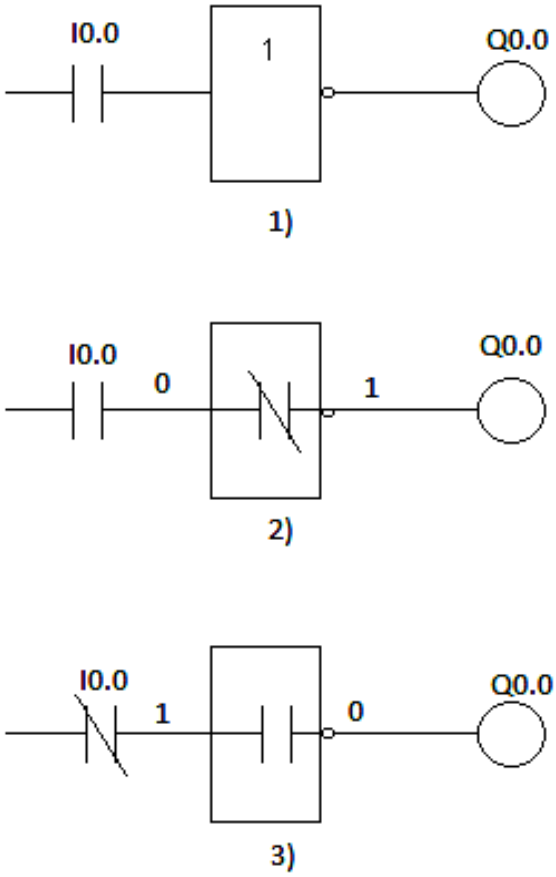

Figura 1.22. Arreglo y funcionamiento de Compuerta “NOT”. (Autoría Propia)

Los diagrama 2 y 3 de la figura anterior son una representación de cómo interactúa la compuerta, en ellos se puede observar que se cumple la tabla de verdad de “NOT”, ya que cuando el estado de la entrada I0.0 es 0 (abierto), la compuerta ya tiene el contacto interno cerrado, de tal forma que 0 se invierte a 1 y cuando el contacto I0.0 se cierra (estado 1), este excita al contacto interno de la compuerta haciendo que se abra (estado 0), así el resultado lógico en la salida será el opuesto al que entró.

La compuerta “**AND**” es la segunda de las Compuertas Lógicas básicas, representa la operación Booleana de “Producto Lógico” utilizando el 1 y 0 del sistema binario, que en este caso es igual al producto de 1 y 0 de los números reales.

Esta compuerta puede tener dos o más entradas, con una sola salida, donde el valor será 1 si y sólo si todas las entradas están en estado 1, esta condición se basa en multiplicar las entradas entre sí para obtener los valores de salida mostrados en la siguiente tabla.

Entrada A	Entrada B	Salida
0	0	0
0	1	0
1	0	0
1	1	1

Tabla 1.4. Tabla de Verdad de la Compuerta “AND” con dos entradas. (Autoría Propia)

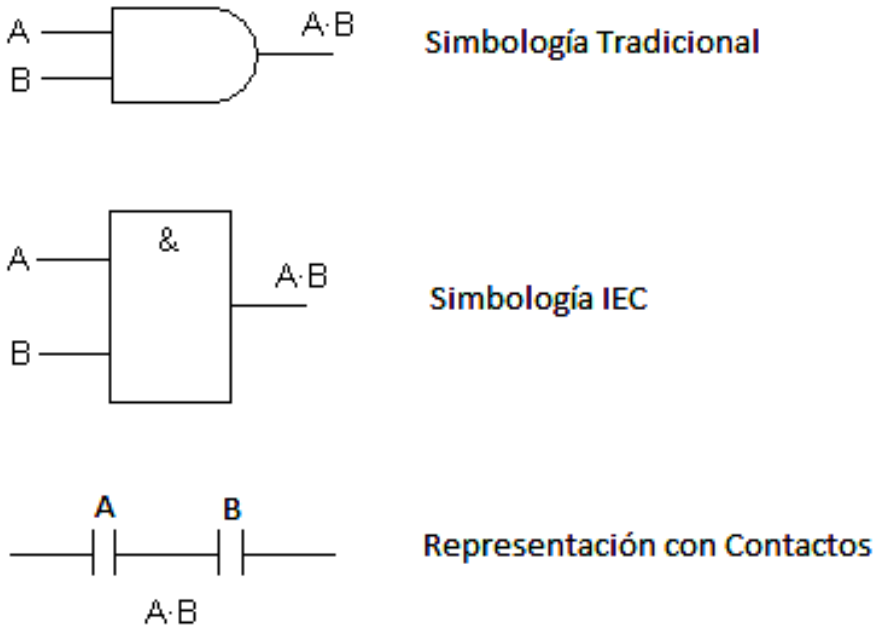

Figura 1.23. Simbología Compuerta “AND”. (Autoría Propia)

La forma de ingresar esta compuerta a un programa de PLC, es agregando un contacto normalmente abierto a cada una de las entradas, las cuales interactúan con la compuerta de forma similar a la compuerta “IF” y al igual que las compuertas anteriores, si se desea que el resultado lógico en la salida se vea reflejado en un accionamiento se tendrá que conectar a una salida o una Marca interna del PLC (diagrama 1 de la Figura 1.24).

Los diagramas del 2 al 5 representan el funcionamiento de la compuerta en sus cuatro posibles combinaciones, tal como lo propone la tabla de verdad.

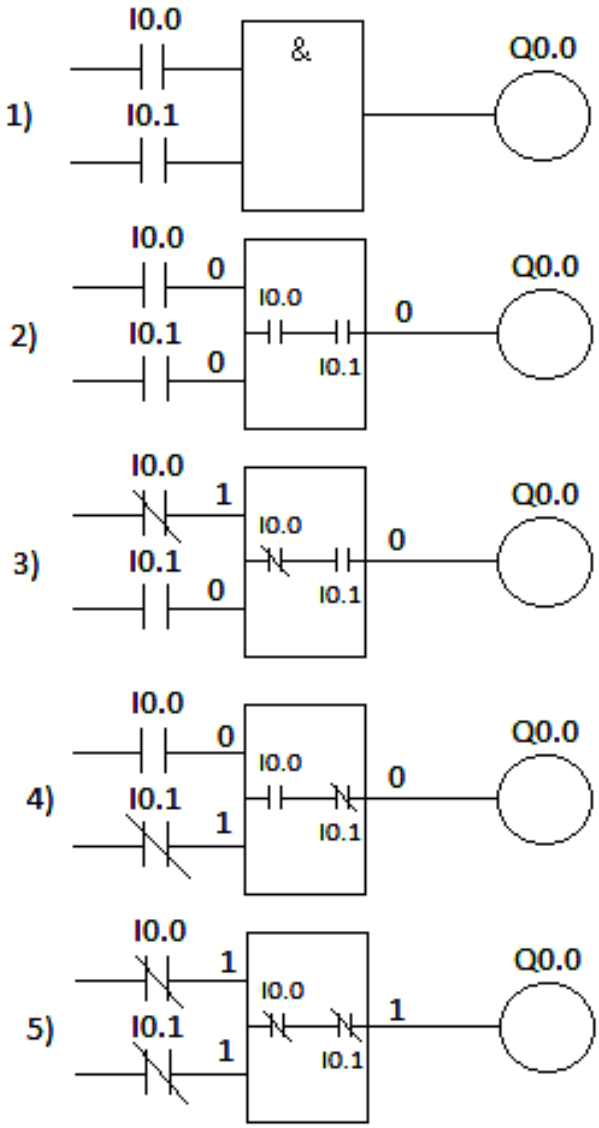

Figura 1.24. Arreglo y funcionamiento de la Compuerta “AND”. (Autoría Propia)

La Compuerta “**OR**” realiza la operación booleana “Adición” o “Suma”, esta compuerta presenta un estado 1 en su salida cuando al menos una de sus entradas también se encuentra en estado 1. En cualquier otro caso, la salida será 0. Al igual quelas compuertas AND, el número de entradas puede ser mayor a dos. (Palazzesi, 2007)

Su tabla de verdad (ver Tabla 1.5) presenta una peculiaridad ya que, al contrario de la multiplicación, la suma de números binarios es muy diferente a la suma de números reales, esta diferencia se hace presente al sumar “1+1” que resultara en “1” dentro del sistema binario. En la Figura 1.25 se presenta su simbología.

Entrada A	Entrada B	Salida
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**1**

Tabla 1.5. Tabla de Verdad de la Compuerta “OR”, con dos entradas. (Autoría Propia)

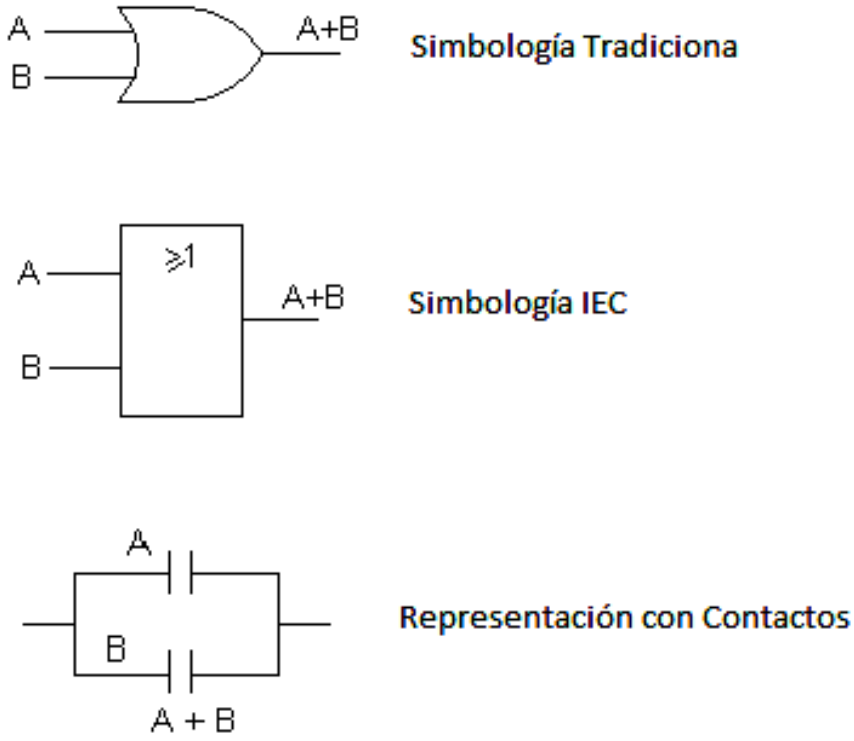

Figura 1.25. Simbología de la Compuerta “OR”. (Autoría Propia)

Para que un software de programación reconozca a esta Compuerta como funcional, necesita dos contactos normalmente abiertos en sus entradas, y de ser necesario conectar la salida a una salida del PLC.

En el diagrama 1 de la siguiente figura se muestra la forma de ingresar la compuerta a un programa de PLC, y en el resto de los diagramas se hace la representación de cómo reacciona la Compuerta a todas las combinaciones posibles de sus entradas, tal como en su tabla de verdad.

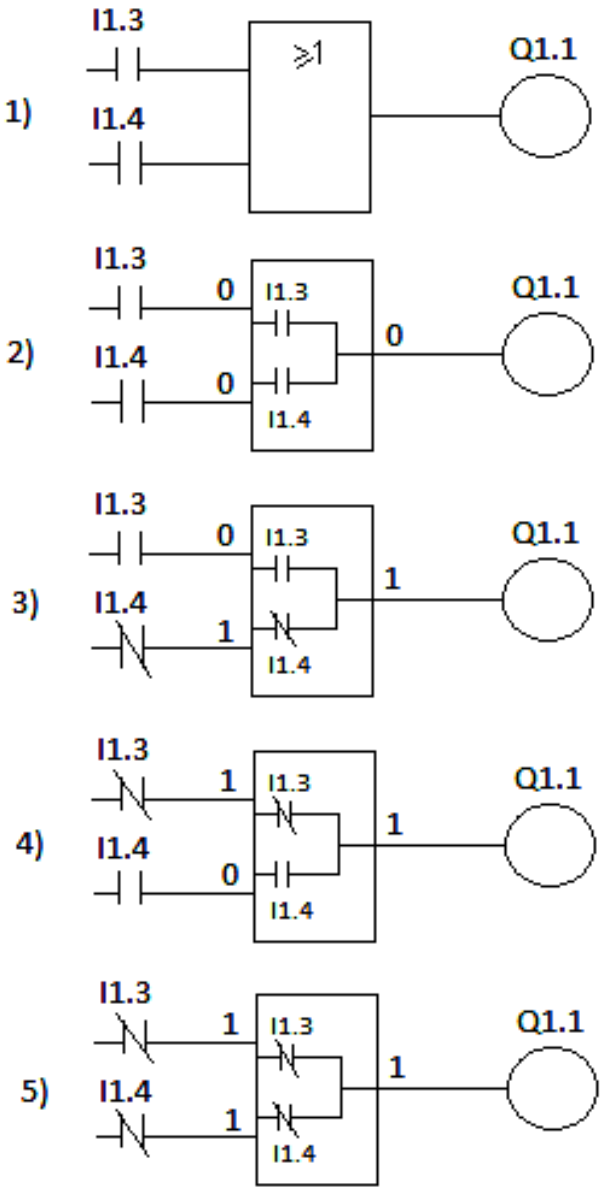

Figura 1.26. Arreglo y Funcionamiento de Compuerta “OR”. (Autoría Propia)

1.3.3 Compuertas lógicas Dual y Especiales

Las Compuertas Lógicas “Dual” o “Negadas”, se refieren a aquellas que hacen una función completamente opuesta a otra, estas niegan (invierten) el resultado lógico de otra Compuerta.

Eléctricamente hablando, todos los elementos se asocian en pares llamados duales, para que puedan ser duales entre sí, sus formulas o características tienen que recíprocas o inversas, por ejemplo:

- Tensión- Corriente
- Resistencia- Conductancia
- Impedancia- Admitancia

Para fines de este documento, nos interesa saber que el dual de un arreglo en “Serie” es uno en “Paralelo”, mientras que para un contacto “Normalmente Abierto”, su dual es un contacto “Normalmente Cerrado”. Esta condición cambiara la representación interna de cada Compuerta.

La Compuerta “**NAND**” es la negación de la compuerta “AND”, “esto modifica su tabla de verdad, de hecho la invierte, quedando que la salida solo será un 0 cuando todas sus entradas estén en 1”. (Palazzesi, 2007)

Entrada A	Entrada B	Salida
0	**0**	**1**
0	**1**	**1**
1	**0**	**1**
1	**1**	**0**

Tabla 1.6. Tabla de Verdad de Compuerta “NAND” con dos entradas. (Autoría Propia)

El número de entradas podrá ser mayor a 2. Esta compuerta podría emularse conectando la salida de una compuerta “AND” a la entrada de una “NOT”, de esta forma se estaría negando el Resultado Lógico de “AND”, pero su uso es tan común que se opta por compuertas internamente negadas.

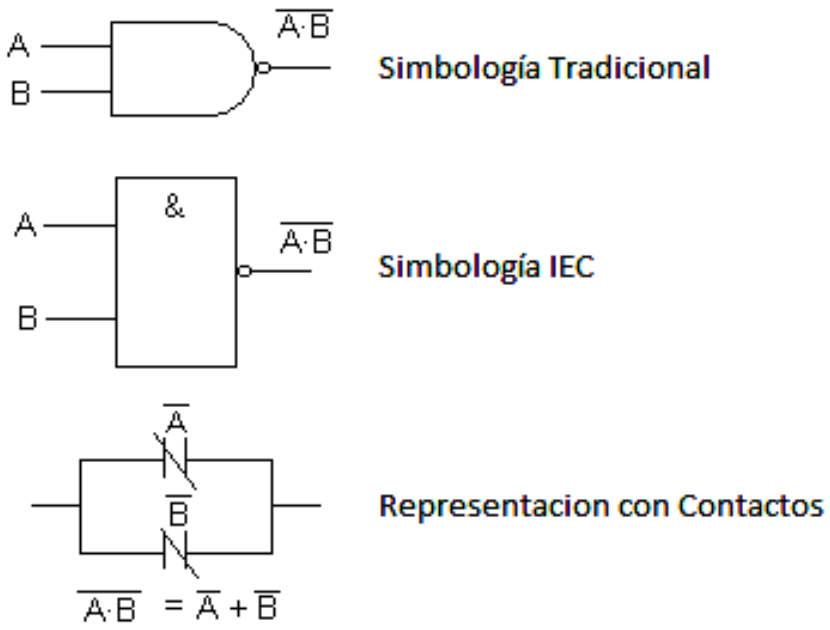

Figura 1.27. Simbología de la Compuerta "NAND". (Autoría Propia)

La forma de ingresar esta compuerta dentro de un programa de PLC en lenguaje FUP será igual que las otras: se agregan tantos contactos abiertos como entradas tenga y la salida conectada a una salida del PLC (Ver diagrama 1 de la figura 1.28).

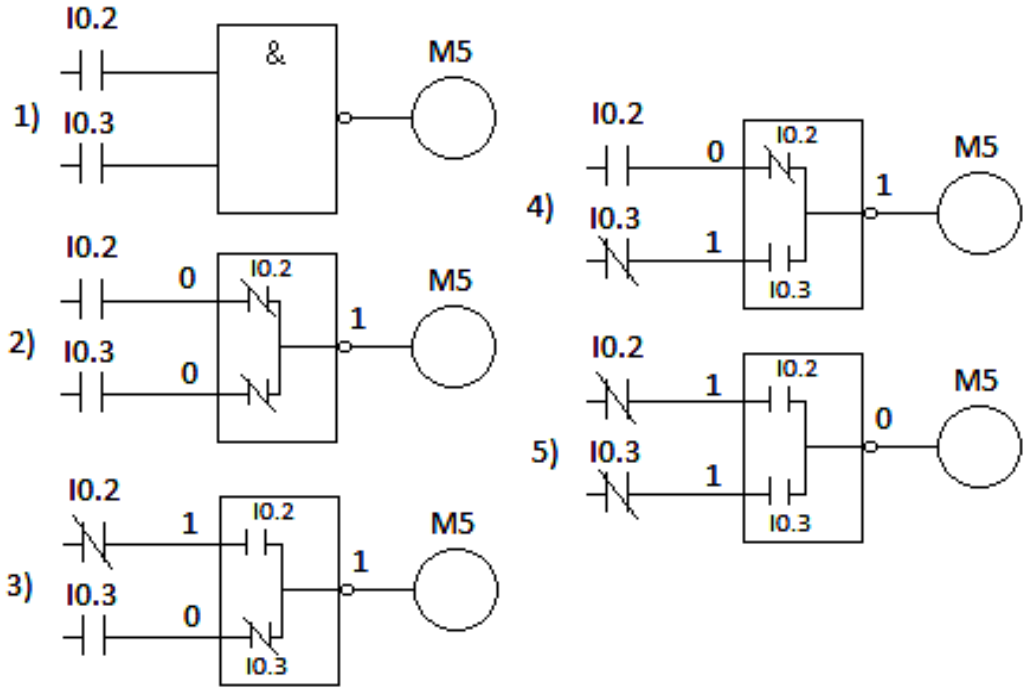

Figura 1.28. Arreglo y Funcionamiento de Compuerta "NAND". (Autoría Propia)

Los diagramas del 2 al 5 de la figura anterior, muestran cómo reacciona la Compuerta a las distintas combinaciones que se pueden formar en sus entradas, donde los contactos dispuestos en ese lugar envían impulsos para cambiar de estado los contactos internos, en este caso lo hace de forma similar a “NOT”.

La Dualidad Eléctrica de “NAND” y “AND” se encuentra en la representación interna, donde “AND” son un par de contactos normalmente abiertos en serie, mientras que “NAND” son un par de contactos normalmente cerrados en paralelo.

La Compuerta “**NOR**” es la negación de una compuerta OR, esto quiere decir que su tabla de verdad se invierte, teniendo valor 1 en su salida sólo cuando en sus entradas hay un estado 0.

Entrada A	Entrada B	Salida
0	**0**	**1**
0	**1**	**0**
1	**0**	**0**
1	**1**	**0**

Tabla 1.7. Tabla de Verdad de Compuerta “NOR” con dos entradas. (Autoría Propia)

También es posible realizar la función de “NOR” conectando la salida de una compuerta “OR” a la entrada de una “NOT”. Simbológicamente hablando, la negación se expresa mediante un círculo en la salida. El número de entradas puede ser mayor a dos. (Palazzesi, 2007)

A continuación se muestra la forma de ingresar esta compuerta en un programa de PLC (Diagrama 1, Figura 1.30), seguido del funcionamiento de la compuerta en base a las posibles combinaciones de sus entradas, así como la representación interna mediante contactos (Diagrama 2-5, Figura 1.30).

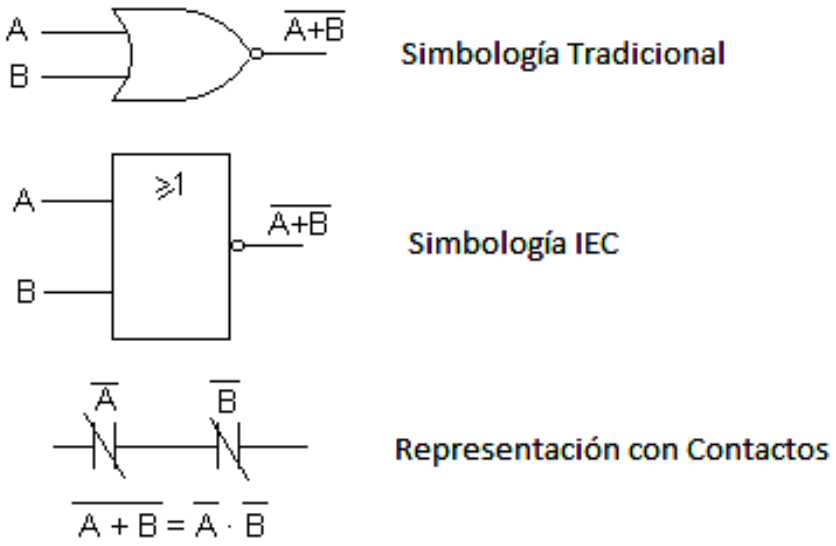

Figura 1.29. Simbología de la Compuerta “NOR”. (Autoría Propia)

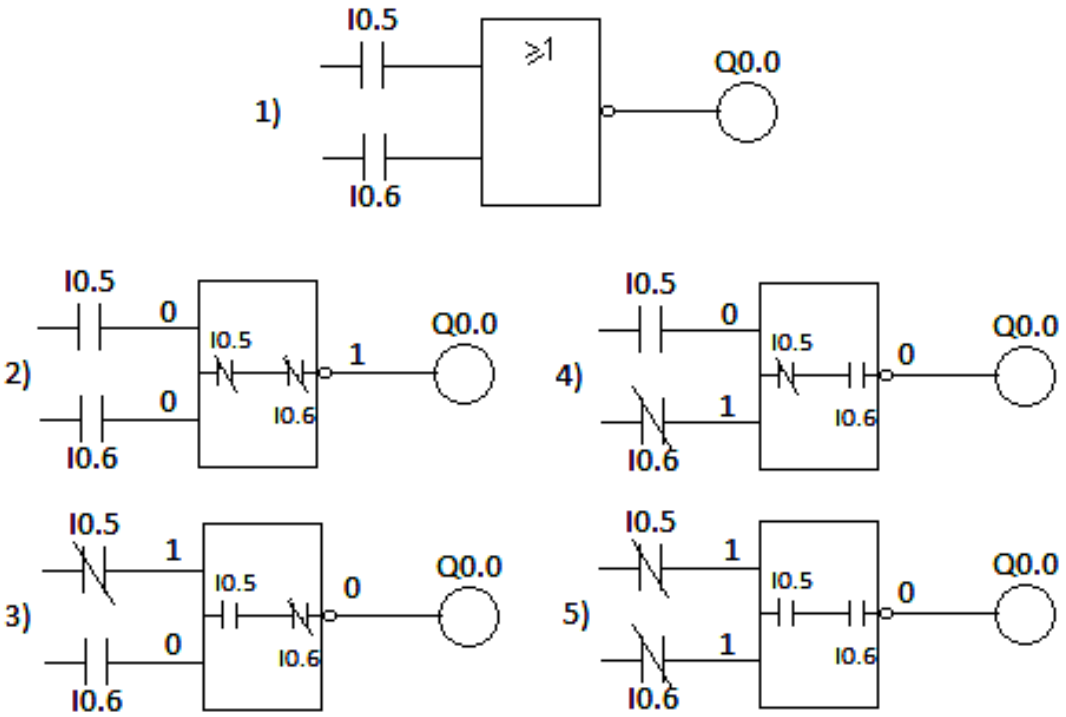

Figura 1.30. Arreglo y Funcionamiento de la Compuerta “NOR”. (Autoría Propia)

En ocasiones, las funciones de negación como las de “NAND” y “NOR” no son suficientes para realizar ciertas tareas, es por eso que son necesarias otras compuertas que brinden más “restricciones” en su funcionamiento.

Un ejemplo de lo anterior es la Compuerta “**XOR**” (OR Exclusiva). Comparándola con la Compuerta “OR” que realiza una operación lógica “inclusiva”, donde basta que una entrada este en 1 para que la salida sea 1, en “XOR” la salida será 1 exclusivamente cuando una entrada este en 1, es decir, las entradas tienen que ser diferentes entre sí, cuando las dos llegan a estar en el mismo estado (0 ó 1), la salida será 0. (Palazzesi, 2007)

En su tabla de verdad se puede apreciar estas condiciones (Tabla 1.8). Esta compuerta sólo puede tener dos entradas.

Entrada A	Entrada B	Salida
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**0**

Tabla 1.8. Tabla de Verdad de la Compuerta “XOR” con dos entradas. (Autoría Propia)

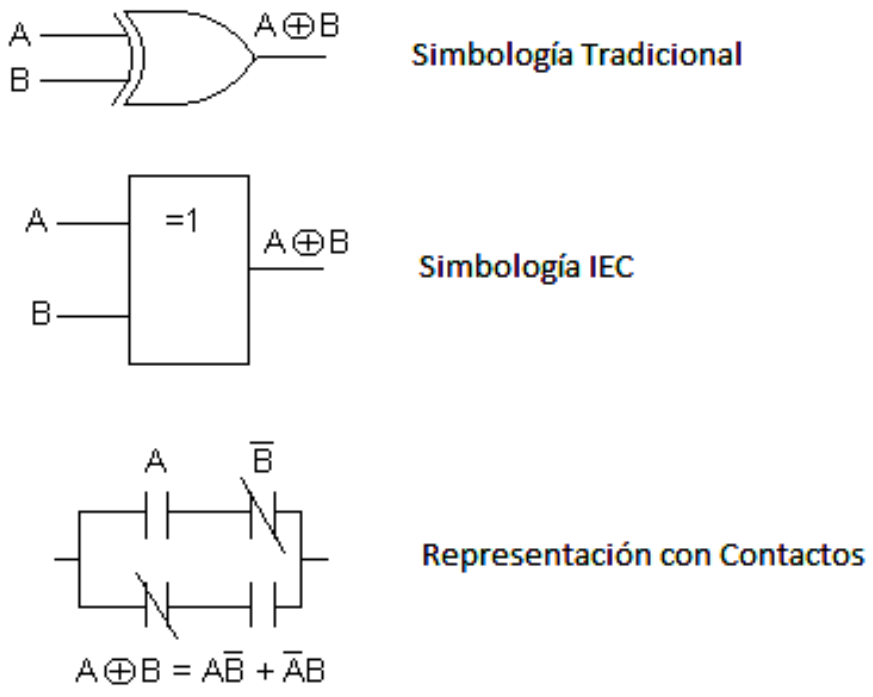

Figura 1.31. Simbología de la Compuerta “XOR”. (Autoría Propia)

Al igual que todas las Compuertas anteriores, “XOR” puede ser representada mediante un esquema de contactos, lo que facilita su comprensión para aplicarlo dentro de un programa de PLC.

En este caso la presencia de un estado (0 ó 1) causara una apertura y un cierre de dos contactos al mismo tiempo, por lo que es muy importante no perder de vista los contactos pertenecientes a cada entrada. Ver Figura 1.32.

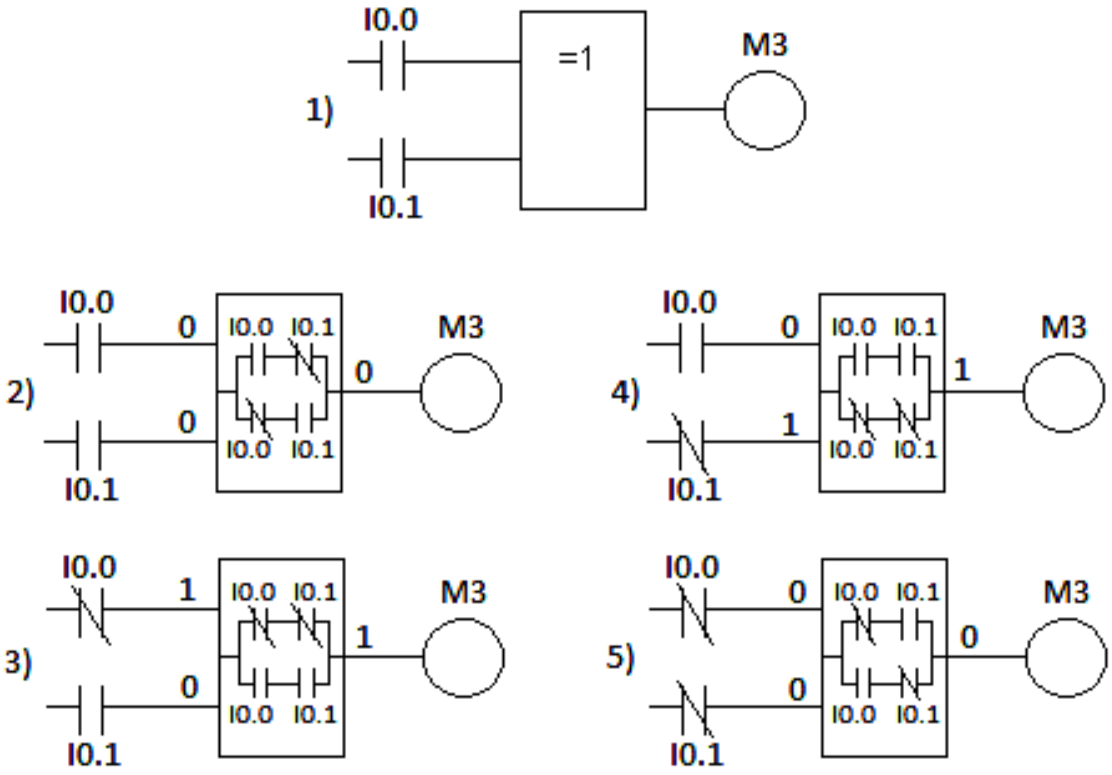

Figura 1.32. Arreglo y Funcionamiento de la Compuerta “XOR”. (Autoría Propia)

En la Figura anterior se muestra el arreglo necesario para un programa de PLC en lenguaje FUP (Diagrama 1), los Diagramas 2, 3, 4 y 5 son el comportamiento de la Puerta respecto a sus posibles combinaciones en las entradas de acuerdo con su tabla de verdad.

La Compuerta “**XNOR**” no es más que la negación de “XOR”, esta Compuerta, podría ser sustituida por una “XOR” conectada a la entrada de una Puerta “NOT” y realizaría la misma función.

La Tabla de Verdad de “XNOR” es precisamente la inversa de “XOR”, donde su salida estará en estado 1 solamente cuando sus entradas son iguales, y en estado 0 para las demás combinaciones posibles. Esta compuerta sólo puede tener dos entradas.

Entrada A	Entrada B	Salida
0	**0**	**1**
0	**1**	**0**
1	**0**	**0**
1	**1**	**1**

Tabla 1.9. Tabla de Verdad de la Compuerta “XNOR”. (Autoría Propia)

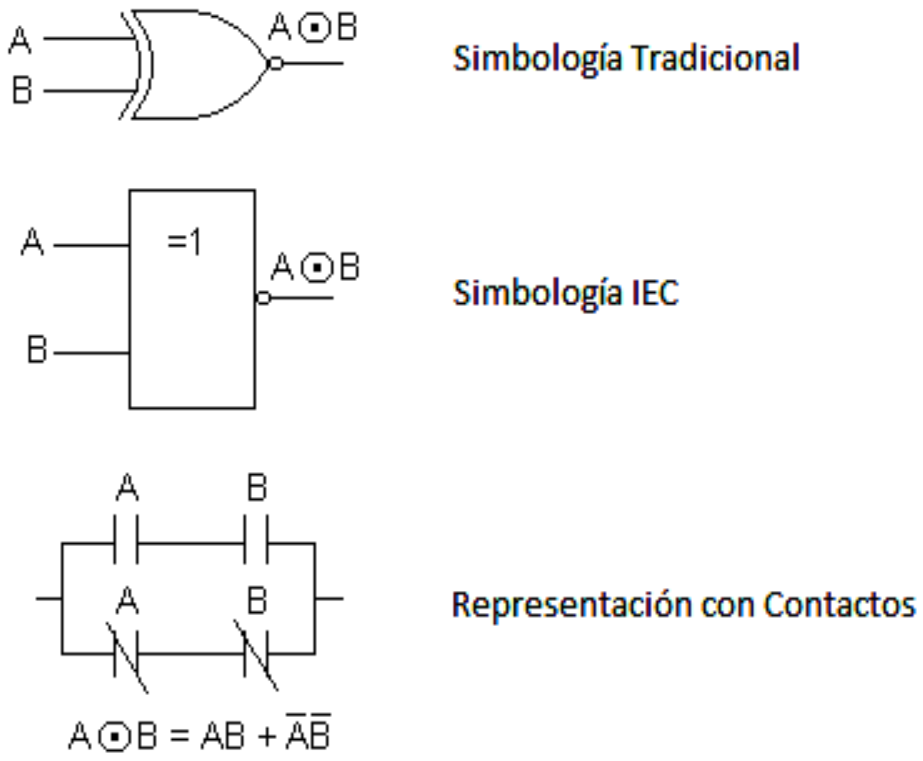

Figura 1.33. Simbología de la Compuerta “XNOR”. (Autoría Propia)

A continuación de ilustra la forma de representar la Compuerta dentro de un programa PLC en lenguaje FUP (Diagrama 1,Figura 1.34), además de la representación interna de la Compuerta con las posibles combinaciones en las entradas donde se corrobora lo mostrado en su Tabla de Verdad.

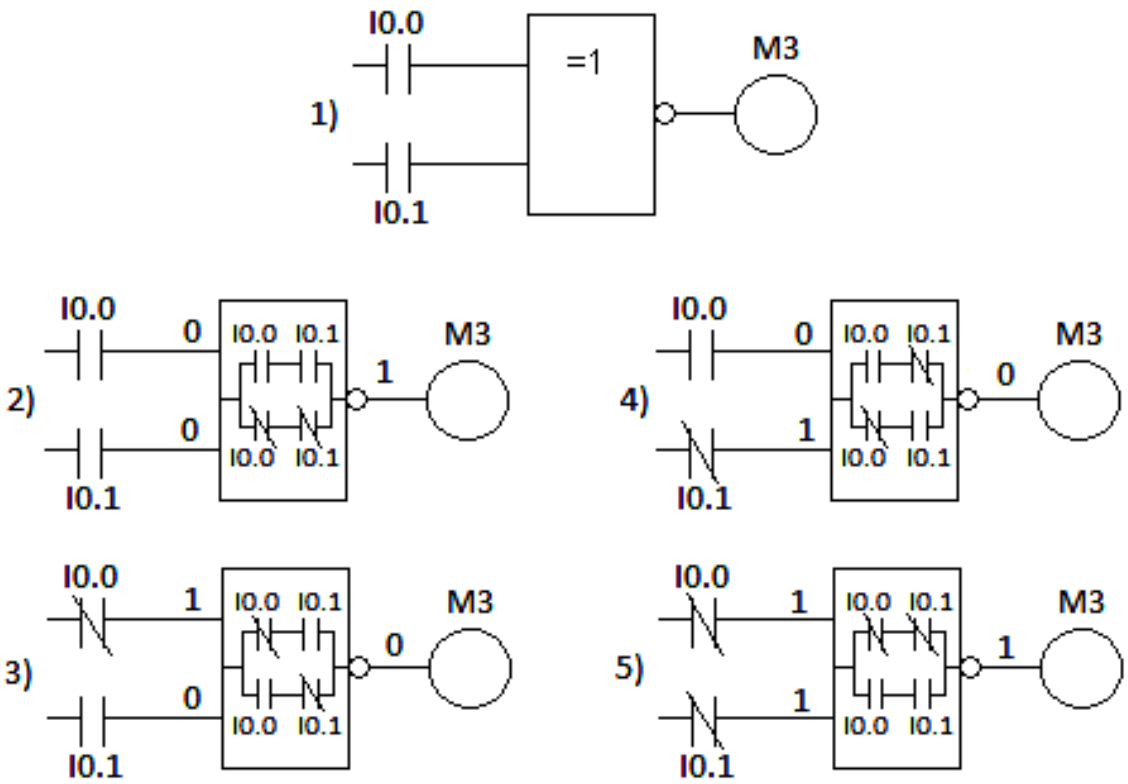

Figura 1.34. Arreglo y Funcionamiento de la Compuerta “XNOR”. (Autoría Propia)

1.3.4 Otros elementos de programación

Como se menciono anteriormente, las Compuertas se pueden combinar entre sí para formar otras, ya que en ocasiones se tendrá número limitado de ellas y es importante saber cómo conectarlas para llegar al resultado deseado.

Dentro de un Software no se tiene este problema ya que podemos disponer de infinitas compuertas de distintos tipos, sin embargo sigue siendo necesario saber algunas de estas combinaciones para poder crear un Programa de PLC.

Un ejemplo de esto es reducir el número de entradas de una Compuerta. Puede ser reducido conectando dos (o más) entradas juntas. La Figura 1.35 muestra una Compuerta “AND” de 3 entradas operando como una puerta de 2 entradas. (Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado)

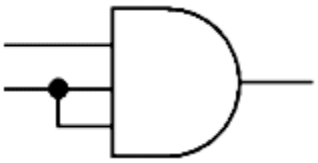

Figura 1.35. Reducción de Entradas. (Modificado de Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado)

En el documento “Electrónica Digital 3” se explica que cualquier puerta puede ser construida partiendo de una “NAND” o una “NOR”. Por ejemplo una “AND” es una “NAND” seguida de una NOT (para deshacer la función inversión). Otro ejemplo es crear una Puerta “NOT” partiendo de una “NAND” (Ver Figura 1.36).

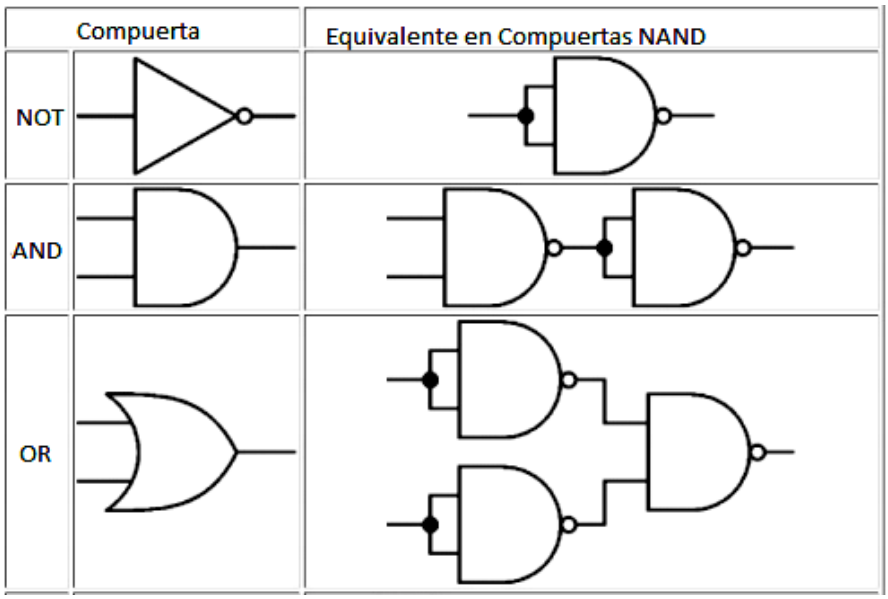

Figura 1.36. Compuertas mediante “NAND”. (Modificado de Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado)

También podemos definir que para cambiar una Compuerta se puede realizar dos modificaciones, invertir la(s) entrada(s) conectando una Puerta “NOT” en ella(s) o invertir la salida. Conectando una Puerta “NOT” en ella. Por ejemplo una compuerta OR puede ser construida invirtiendo las entradas de una puerta NAND.

Capítulo II: FluidSIM como Plataforma de Programación

2.1 ¿Qué es FluidSIM?

En el último apartado se profundizo en el lenguaje de programación FUP y las Compuertas Lógicas puesto que en el Software “FluidSIM” es precisamente el único lenguaje con el que se puede programar un PLC.

FluidSIM es una herramienta de simulación para la obtención de los conocimientos básicos de neumática. Es desarrollado por FESTO, empresa líder en automatización neumática y electroneumática.

“Una característica importante es su estrecha relación con la función y simulación CAD (Diseño Asistido por Computadora). FluidSIM permite crear el esquema del circuito de un fluido según DIN (Instituto Alemán de Estandarización); también posibilita la ejecución sobre la base de descripciones de componentes físicos de una simulación plenamente explicativa. Con esto se establece una división entre la elaboración de un esquema y la simulación de un dispositivo práctico”. (Art Systems, Festo Didactic, 2007)

También permite verificar si ciertas conexiones entre componentes son realmente posibles.

“Otra característica es su bien pensado concepto didáctico, FluidSIM soporta el aprendizaje, la formación y la visualización de los conceptos de la técnica neumática. Los componentes neumáticos se explican con descripciones textuales, figuras y animaciones que ilustran los principios de funcionamiento subyacentes; ejercicios y vídeos didácticos aportan conocimientos sobre los circuitos esenciales y el uso de los componentes neumáticos”. (Art Systems, Festo Didactic, 2007)

Existe también FluidSIM que trabaja bajo teoría de la Hidráulica, a pesar que básicamente es el mismo software este se tiene que adquirir por separado. La versión de FluidSIM Neumática con la que se ilustrara el presente apartado es la 4.2 en Ingles aunque en el mercado ya se encuentra la versión 5 que es la más reciente.

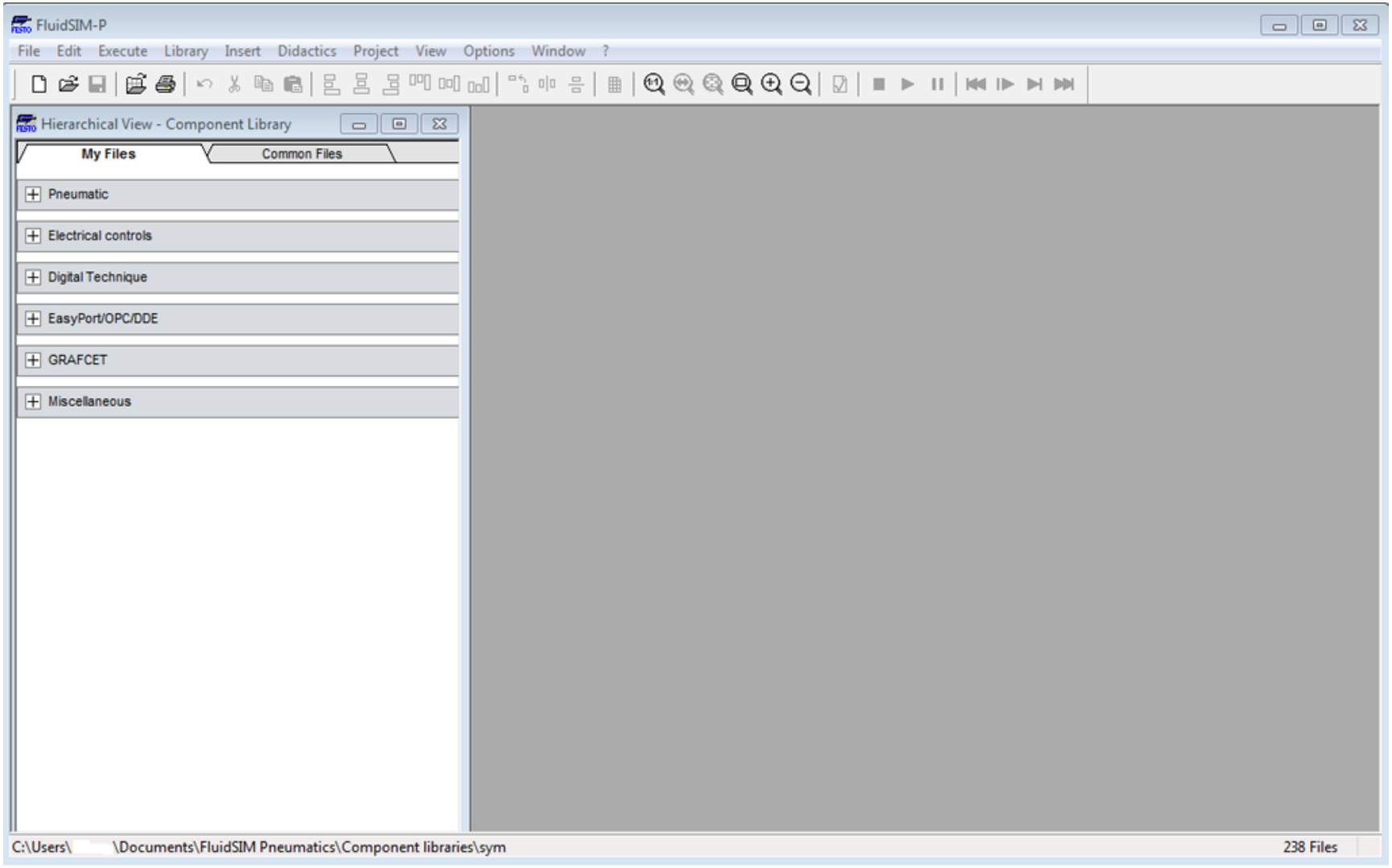

Figura 2.1. Pantalla Principal. (Hecha mediante FluidSIM)

Cabe señalar que muchos de los ejemplos y pasos que se describirán en este apartado pueden variar para versiones distintas a la 4.2. También es importante señalar que las licencias de estudiante y demostración del Software pueden no tener todas las funciones que se mencionaran en este documento.

2.2 Navegación en FluidSIM

Partiremos del hecho que el Software ya se encuentra instalado, donde al iniciarlo podrimos visualizar la pantalla principal o área de trabajo, tal y como se muestra en la Figura 2.1.

En la parte izquierda podemos observar la biblioteca de componentes organizada de forma jerárquica. Esta contiene componentes neumáticos, eléctricos y digitales para el diseño de circuitos.

Utilizando la barra de menú de la parte superior de la ventana (Figura 2.2), puede acceder a todas las funciones necesarias para el diseño y simulación de circuitos.

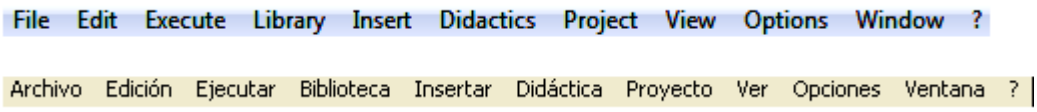

Figura 2.2. Barra de Menú, en inglés y español. (Hecha Mediante FluidSIM)

La barra de herramientas inferior le permite el cómodo acceso a las funciones de menú utilizadas frecuentemente, para conocer sus funciones basta con pasar el puntero por encima de cada icono, se organizan en 10 grupos mostrados en la Figura 2.3.

Estas herramientas no estarán habilitadas todo el tiempo, solo estarán seleccionables en ciertos procesos del programa, por ejemplo, no se podrán alinear elementos mientras una simulación está corriendo. Todos los demás elementos del programa funcionan igual que lo hacen otros bajo la plataforma de Microsoft Windows, como las barras de desplazamiento, funciones de “guardar”, “guardar como”, “abrir”, etc.

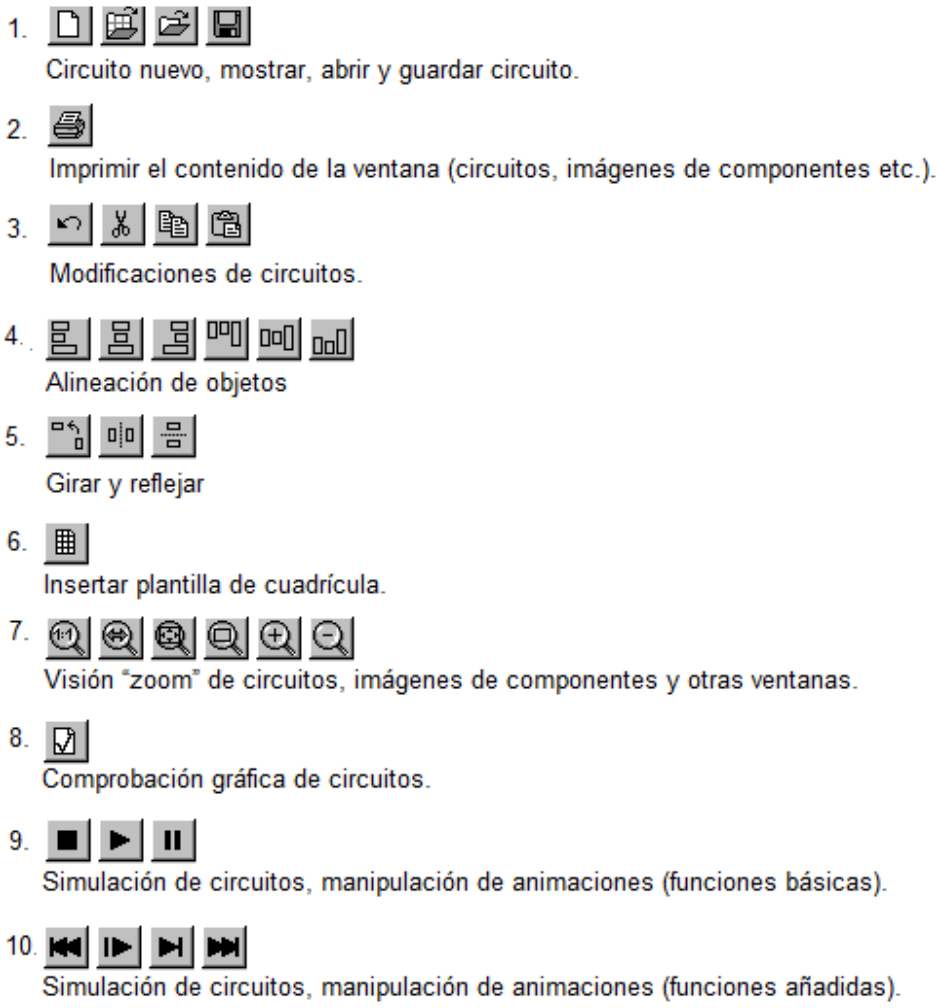

Figura 2.3. Barra de Herramientas. (Art Systems, Festo Didactic, 2007)

La biblioteca de componentes contiene todos los objetos que podremos usar para realizar simulaciones de circuitos, cada elemento en la biblioteca corresponde a un elemento físico existente en el mercado, lo que permite llevar a cabo los circuitos en la vida real.

La biblioteca está dividida en seis categorías:

- **Neumática** (Pneumatic): Básicamente contiene todo aquel componente que se puede agregar en un circuito Neumático o Electroneumático.
- **Componentes Eléctricos** (Electrical Controls): Como el nombre sugiere, contiene todo componente puramente eléctrico.

- **Técnica Digital** (Digital Technique): Esta categoría contiene los elementos que se utilizaran para la programación de un PLC, es decir, contiene elementos de Electrónica Digital.
- **EasyPort/OPC/DDE**: Son herramientas que nos permiten establecer una comunicación a dispositivos externos de FluidSIM.
- **GRAFCET**: Brinda las herramientas para crear y simular diagramas bajo este lenguaje.
- **Otros** (Miscellaneous): Contiene elementos que podrían ser necesarios, como cuadros de texto, figuras predeterminadas, logos, etc.

Cabe mencionar que las Versiones anteriores a FluidSIM 4.2 no organizan la biblioteca de la forma anterior.

La categoría Neumática a su vez está dividida en sub-categorías para un mejor acceso a sus componentes, quedando le la siguiente manera:

- Elementos de Alimentación (Supply Elements)
- Actuadores (Actuators)
- Válvulas (Valves)
- Grupo de Válvulas (Valve Groups)
- Sensores e Instrumentos de Medida (Measuring Instruments and Sensors)

La pestaña de Componentes Eléctricos también se encuentra subdividida de la siguiente manera:

- Actuadores
- Alimentación de Tensión (Power Supply)
- Sensores e Instrumentos de Medida
- Relés (Relay)
- Interruptores (Switches)
- Regulador (Controller)
- Símbolos de Diagrama de Ladder (Ladder Symbols)

Las demás categorías generales no tienen otra subdivisión, por lo que al abrirlas sus componentes quedaran visibles. Todos los componentes mostrados en la biblioteca son símbolos estandarizados por DIN, por lo que cualquier persona familiarizada con simbología eléctrica encontrara muy intuitiva la búsqueda de elementos dentro de la biblioteca.

FluidSIM también cuenta con alternativas para personas que no tengan conocimientos amplios en el tema, por esa razón todos los componentes poseen una descripción técnica en una página de ayuda que contiene el símbolo del diagrama para el componente según el estándar DIN, una descripción breve de la función del componente, la designación de las conexiones y un listado de parámetros configurables, además mostrará una fotografía de la pieza tal como es en realidad. (Art Systems, Festo Didactic, 2007)

Festo Didactic aclara que en el caso de que un componente no exista de forma independiente en un sistema real, FluidSIM muestra una foto del grupo de montaje al que pertenece, por ejemplo, relés, interruptores y la fuente de alimentación eléctrica, los componentes que no existan en la realidad, simplemente no tienen foto. Ejemplos de ello son los textos de los componentes y la regla de recorrido.

Es esta la razón por la que no se describe cada uno de los componentes es este documento. Toda esta información la podremos encontrar al dar Clic Derecho sobre el símbolo deseado y elegir la opción “Descripción del Componente” (Component Description).

Para iniciar un proyecto nuevo de FluidSIM, es necesario hacer Clic en el primer icono de la barra de herramientas, “Nuevo” o “New”, entonces en la zona de color gris de la Figura 2.1 aparecerá una ventana en blanco, la cual será la Zona de Trabajo. Otra forma de realizar este paso es entrar en el Menú “Archivo” (File) y enseguida seleccionar “Nuevo”.

En la parte superior de la ventana que contiene la Zona de Trabajo se encuentra la dirección y nombre actual del Nuevo Circuito (“noname” será el nombre por defecto con extensión “.ct”), nombre que podrá modificarse al momento de guardar cambios.

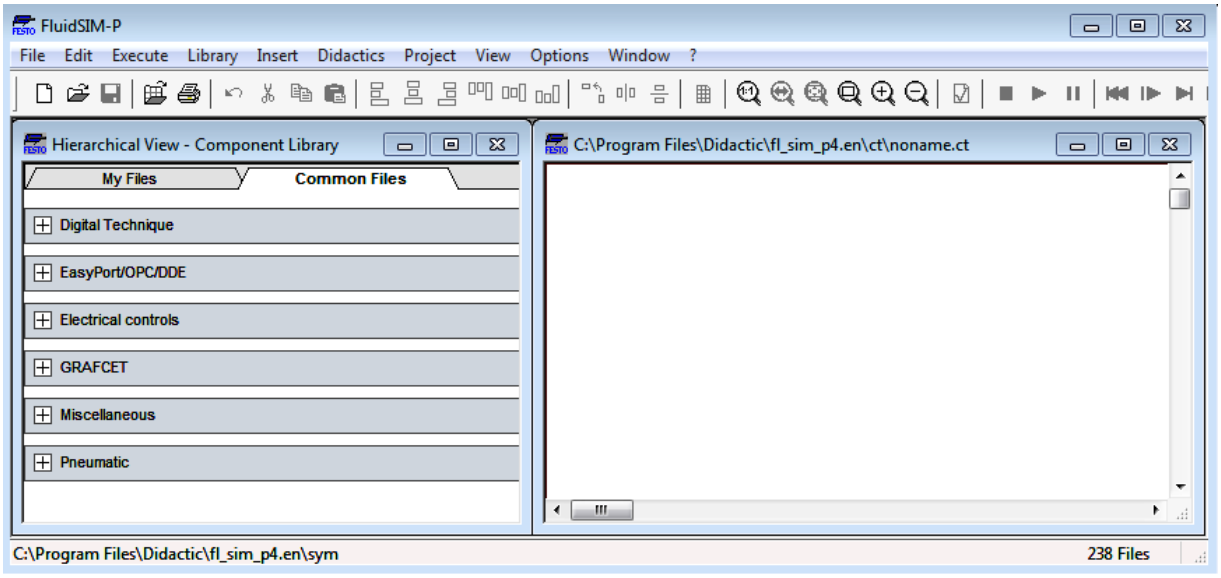

Figura 2.4 Circuito Nuevo. (Hecha mediante FluidSIM)

Ya con la zona de trabajo desplegada, podremos empezar la construcción de nuestro circuito, para lo cual solo es necesario arrastrar los componentes deseados desde la biblioteca.

Otro lugar donde podremos encontrar y usar todos los componentes de la biblioteca, es el menú Insertar (Insert), localizado en la barra de menú, estarán ordenados en las categorías ya mencionadas pero de forma escrita, de cualquier forma, al navegar entre los nombres de los componentes se irá mostrando el símbolo correspondiente en una extremo de la pantalla.

Hay que tener en cuenta que FluidSIM es un simulador de circuitos que podrían llevarse a la realidad, por esta razón, dentro del software hay que incluir todos los componentes y respetar todas la reglas que tendría cualquier circuito real, por ejemplo, incluir una fuente de poder, conectar a línea neutra los componentes incluidos, respetar el orden en que se conectan los componentes para realizar ciertas tareas, etc.

Ejemplificaremos lo anterior mediante un circuito para encender y apagar una lámpara. Como se menciono anteriormente, primero tenernos que desplegar una zona de trabajo, enseguida arrastraremos los componentes necesarios (Ver Figura 2.5), que en este caso son:

- **Fuente de Tensión a 24 v** (Electrical Connection).

  Se podrán encontrar en la biblioteca de componentes siguiendo la ruta: Componentes Eléctricos/Alimentación de Tensión. Cabe destacar que la única fuente que se podrá encontrar es la de 24 volts, ya que ese el voltaje en el que comúnmente realizan los circuitos de control industrial.

- **Fuente de Tensión a 0v** (Electrical Connection)

  Se encuentra en la misma ruta de la Fuente de Tensión a 24v.

- **Interruptor Obturador** (Detent Switch Make)

  La podremos hallar en la biblioteca de componentes, siguiendo la ruta: Componentes Eléctricos/Interruptores/Manualmente Operados. Este es un Interruptor Normalmente Abierto, con enclavamiento mecánico.

- **Indicador Luminoso** (Indicator Light)

  Se encuentra localizada en la siguiente ruta: Componentes Eléctricos/Sensores e Instrumentos de Medida.

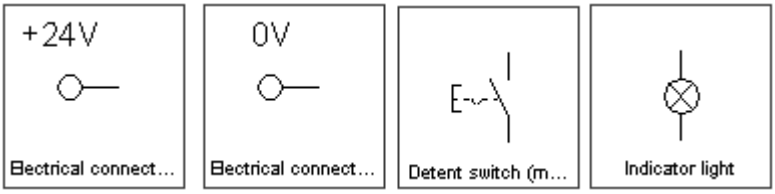

Figura 2.5. Elementos para Circuito de Lámpara. (Hechas Mediante FluidSIM)

Al momento de arrastrar los componentes a la zona de trabajo, estos no toman una posición automática, se quedarán en el lugar donde dejemos de arrastrar, por esta razón hay que alinearlos de forma manual.

A pesar que podrían ordenarse de la forma que quisiéramos, es importante seguir los estándares de DIN para diagramas eléctricos, donde los pasos (escalones) de los diagramas eléctricos de control deberán dibujarse de izquierda a derecha y los elementos de dichos pasos de arriba hacia abajo, teniendo la Fuente de Tensión a 24v en la parte superior y la de 0v en la parte inferior (Ver Figura 2.6).

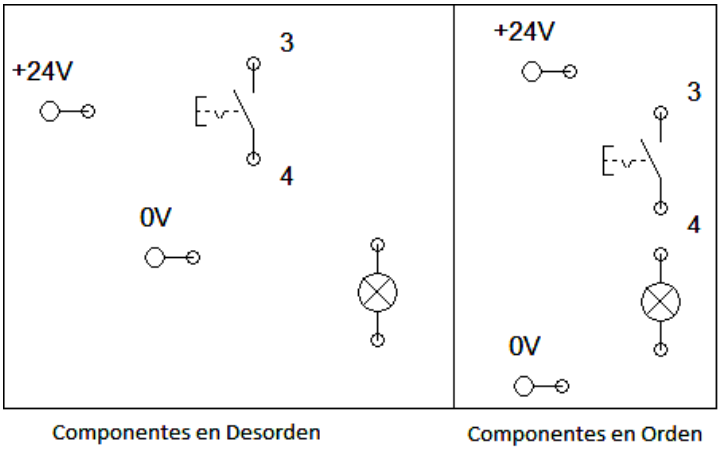

Figura 2.6. Orden Correcto de Componentes. (Hecha Mediante FluidSIM)

Finalmente tenemos que conectar los componentes entre sí, para esto hay que colocar el puntero en alguna de las terminales de algún componente, si esta se pone en color verde significa que es elegible, entonces damos Clic y sin soltar arrastramos hasta la terminal donde deseemos conectarla, la cual también tendrá que ponerse de color verde, indicando que es una terminal disponible para la conexión, de caso contrario, pasara a color rojo. También se podrá realizar la conexión de una terminal a una línea, al igual que las terminales, las líneas se tendrán que iluminar en color verde.

Después de realizar todas las conexiones, nuestro diagrama quedara tal y como en la Figura 2.7.

Ciertos elementos tienen parámetros o propiedades que podemos cambiar, en el caso del interruptor solo podremos ponerle una etiqueta, con la cual se hará distinción entre otros y para la lámpara se brinda la opción del color que tendrá cuando esta encienda.

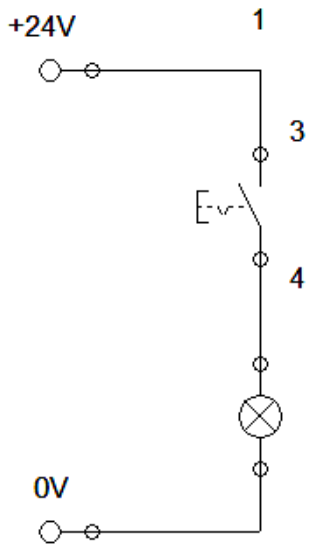

Figura 2.7. Componentes de Lámpara Conectados. (Hecha Mediante FluidSIM)

Ambas opciones se pueden encontrar al presionar Clic derecho sobre un componente y elegir “propiedades”, o también presionando doble Clic sobre el elemento deseado.

En este punto ya podemos empezar la simulación, la cual nos permitirá observar cómo se comportan los elementos en diferentes circunstancias. También nos permitirá observar el flujo de la corriente eléctrica a través del circuito.

Para iniciar, nos tenemos que dirigir a las herramientas de simulación (funciones básicas) antes presentadas, y presionar la herramienta “Iniciar” o “Start” representada por un triangulo.

Al iniciar la simulación, la fuente de tensión libera la corriente, permitiendo que pase por los elementos, en este caso, el interruptor está abierto, por lo que la corriente no pasará, pero si desplazamos el puntero por encima del interruptor en plena simulación, este pasará a tener la forma de una mano, indicando que podemos presionar ese interruptor.

Cuando activamos el interruptor, este cierra el circuito y enciende la lámpara, en ese instante será visible como la línea cambia a color rojo, indicando que por ese segmento está circulando corriente eléctrica (Ver Figura 2.8).

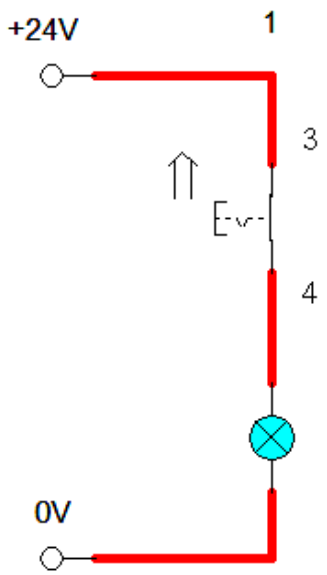

Figura 2.8. Lámpara encendida y Flujo de corriente. (Hecha Mediante FluidSIM)

En ocasiones la complejidad del circuito no permite analizar fácilmente un componente en particular, por lo que se vuelven necesarias las funciones añadidas de las herramientas de simulación, que nos dan la opción de realizar la simulación en forma no instantánea, dejando observar los cambios de estado y cómo se desplaza la corriente.

Otra situación muy común dentro de la simulación, es que en el instante de iniciarla, FluidSIM realiza una verificación sobre el diagrama que queremos simular, y si algo estuviera mal nos despliega una alerta sobre la falla que el diagrama presenta.

Los errores más comunes son: líneas sobrepuestas, Conexiones Abiertas, Etiquetas no Validas, Cortos Circuitos, etc.

Muchas de esas fallas son por el mal arreglo del circuito, en otras ocasiones son por usar elementos que no corresponden, de igual forma FluidSIM nos da la opción iniciar la simulación a pesar de la alerta. A veces se correrá con la suerte de que la falla detectada realmente no afecte el desempeño deseado del circuito, pero podría no ser así, por lo que se recomienda iniciar la simulación hasta corregir todas las fallas que el mismo software nos indique.

Una función muy utilizada en el diseño de circuitos de FluidSIM son las “Etiquetas”, una etiqueta posee un nombre concreto y un componente puede disponer de ella. Si dos componentes tienen la misma etiqueta, estarán conectados entre sí, aunque no se haya dibujado una conexión visible entre ellos. (Art Systems, Festo Didactic, 2007). Para asignar una etiqueta hay que presionar Doble Clic sobre el componente deseado (máximo treinta y dos caracteres por etiqueta).

Para ejemplificar esto, haremos una modificación al diagrama anterior, sustituyendo la lámpara por un Relé (Relay) y en otro escalón colocaremos un “Interruptor Común Obturador” (Make Switch) seguido de la lámpara. En la siguiente imagen se muestran dos posibles arreglos.

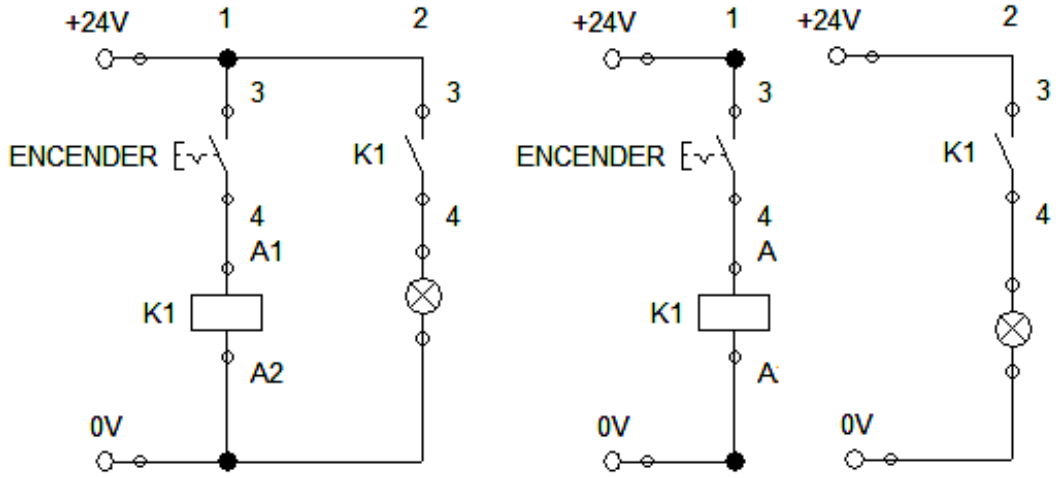

Figura 2.9. Uso de etiquetas. (Hecha Mediante FluidSIM)

En el diagrama nombramos al relé como “K1” y ese mismo nombre se lo asignamos al interruptor común, de esta forma la apertura o cierre del interruptor dependerá de la activación del relé K1, el cual a su vez depende del Interruptor que nombramos como “ENCENDER”. Ver Figura 2.10.

Por medio de un relé pueden conmutarse varios interruptores a la vez. Para ello es necesario acoplar todos los interruptores deseados mediante las etiquetas.

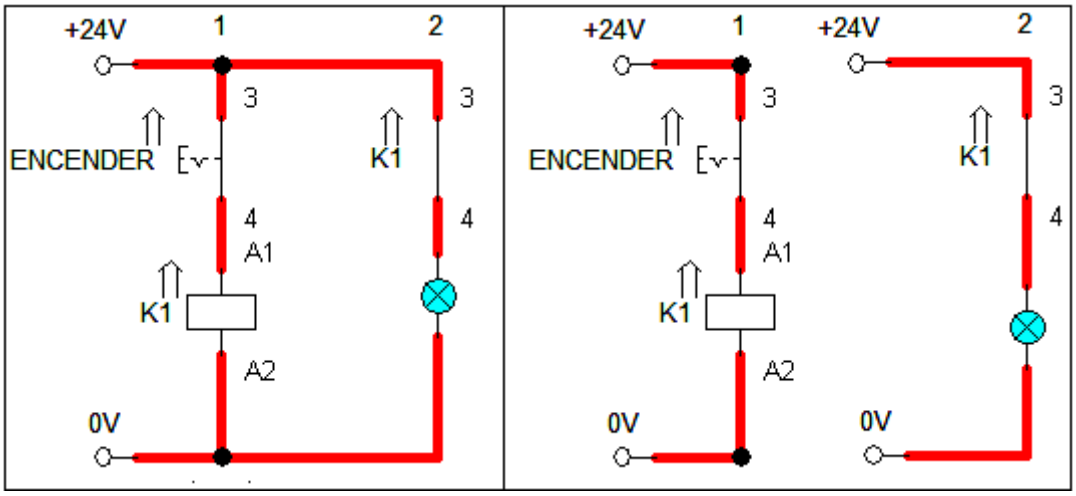

Figura 2.10. Uso de etiquetas 2. (Hecha Mediante FluidSIM)

Junto a los relés simples, existen también relés temporizadores a la conexión, a la desconexión y contadores. Estos se encargan de que los interruptores acoplados sean activados con retardo o cuando se hayan producido un determinado número de señales (impulsos). En este tipo de relés también aparece una ventana de diálogo (tras un doble clic) para la inclusión de los valores. (Art Systems, Festo Didactic, 2007)

El Manual de Usuario de FluidSIM indica que el acoplamiento con interruptores mecánicos (o de accionamiento manual) se realiza también mediante la asignación de una etiqueta. Si varios interruptores mecánicos poseen la misma etiqueta, al accionar uno de ellos, todos los demás serán igualmente activados.

Otro tipo de circuitos que podemos crear en FluidSIM son los Neumáticos, básicamente se crean de la misma forma que los Eléctricos. La diferencia más notable es que los conductos se iluminaran de color azul, indicando el flujo de aire, además de pequeñas flechas que indican la dirección de ese flujo.

A continuación realizaremos un circuito básico de Neumática, para lo que dispondremos de tres elementos:

- **Cilindro de Simple Efecto** (Single Acting Cylinder)

  En la biblioteca de componentes se encuentra siguiendo la ruta: Neumática/Actuadores.
- **Fuente de Aire Comprimido** (Compressed Air Supply)

  La ruta a seguir para hallar este componente es Neumática/Elementos de Alimentación.
- **Válvula de 3/n Vías** (3/n Way Valve)

  La ruta es Neumática/Válvulas/Válvulas de Vías Configurables.

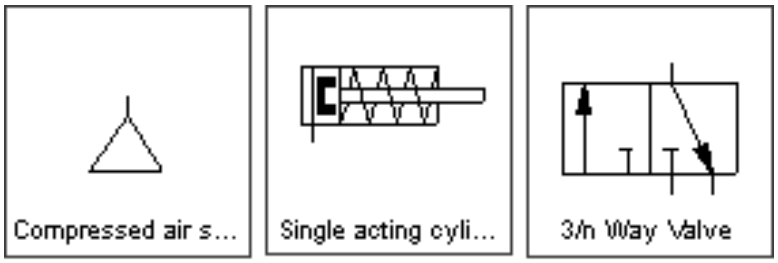

Figura 2.11. Componentes de Diagrama Neumático. (Hecha Mediante FluidSIM)

Después de conectarlos quedaran de la siguiente forma:

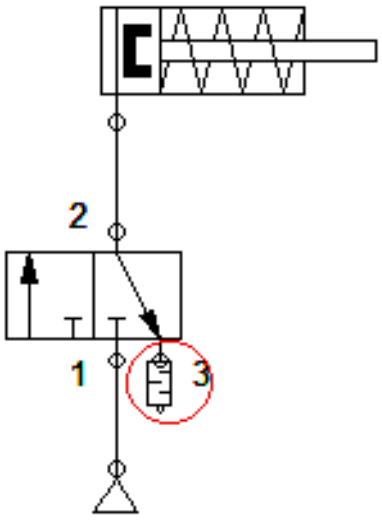

Figura 2.12. Diagrama Neumático. (Hecha Mediante FluidSIM)

El tipo de conexión entre la Fuente y la válvula mostrado en la Figura 2.12 hará que el cilindro entre en movimiento solo cuando la válvula sea activada. El elemento dentro del círculo rojo es un cierre de conexión, en la realidad, una válvula de este tipo necesita un conducto de escape de aire una vez que el cilindro regrese a su posición original, en este conducto podremos conectar distintos accesorios como filtros, silenciadores o simples tapones, la elección de uno de ellos dependerá de la aplicación del sistema Neumático.

En FluidSIM podríamos prescindir de ellos sin ninguna afectación a nuestra simulación, pero como ya se mencionó, esto generaría una alerta, por eso optamos por poner un cierre de conexión. Esto lo podremos hacer dando Doble Clic sobre la conexión deseada de la válvula.

Regresando al ejemplo, una vez que empecemos la simulación podremos observar que existen dos tonos de color azul, el más obscuro indicará presión alta de aire, mientras que la de tono claro presión baja. La siguiente figura muestra el comportamiento de los elementos incluidos, antes y después de activar la válvula (Dar Clic sobre la Válvula para activarla).

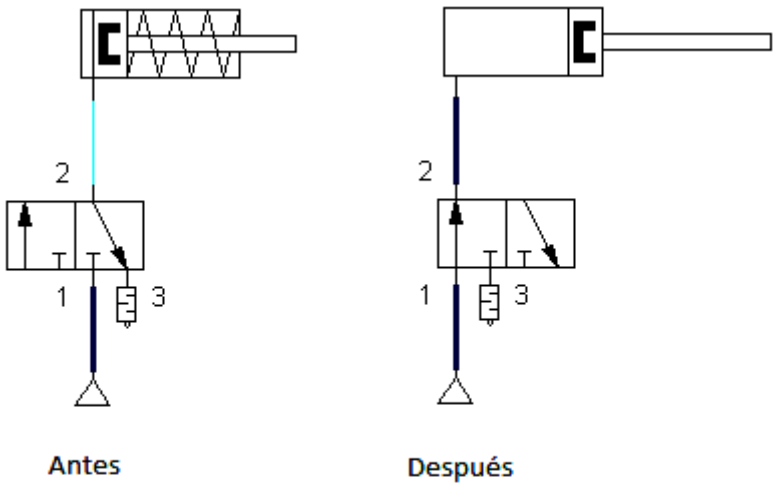

Figura 2.13. Comportamiento Circuito Neumático. (Hecha Mediante FluidSIM)

Finalmente, realizaremos un ejemplo que represente cómo interactúa un diagrama eléctrico con uno neumático, para esto, tomaremos el ejemplo anterior y le acoplaremos un circuito eléctrico.

El circuito eléctrico necesitará: Fuentes de Tensión, Un Interruptor Obturador manualmente operado y un Solenoide de Válvula (Valve Solenoid), este último se encuentra en el apartado de Relés en la Biblioteca de Componentes.

Después de insertar y conectar todos los componentes necesarios, el diagrama quedara como en la Figura 2.14.

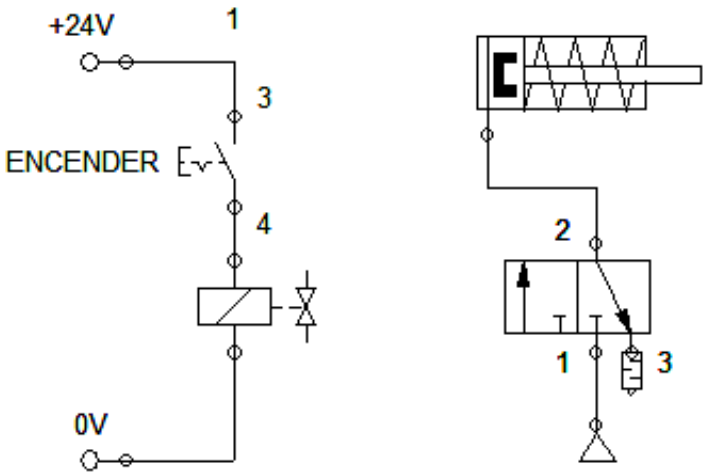

Figura 2.14. Circuito Electroneumático. (Hecha Mediante FluidSIM)

Ahora necesitamos conectarlos de tal forma que la activación de la válvula dependa directamente del Interruptor, para esto hay que configurar la Válvula (Doble Clic sobre la válvula). Ver Figura 2.15.

Dentro de las numerosas opciones que nos ofrece esta ventana de configuración, nos ubicaremos en las opciones de la columna “Left Actuation” (Accionamiento Izquierdo). Como el nombre sugiere, estas son las opciones que nos permiten realizar configuraciones del lado izquierdo de la válvula, nos posibilita atribuirle accionamientos manuales, mecánicos y neumáticos/eléctricos (Manually, Mechanically, Pneumatically/ Electrically), en este caso, nosotros queremos que la válvula sea activada por un circuito eléctrico, entonces desplegamos las opciones de Pneumatically/Electrically y seleccionamos la primera opción.

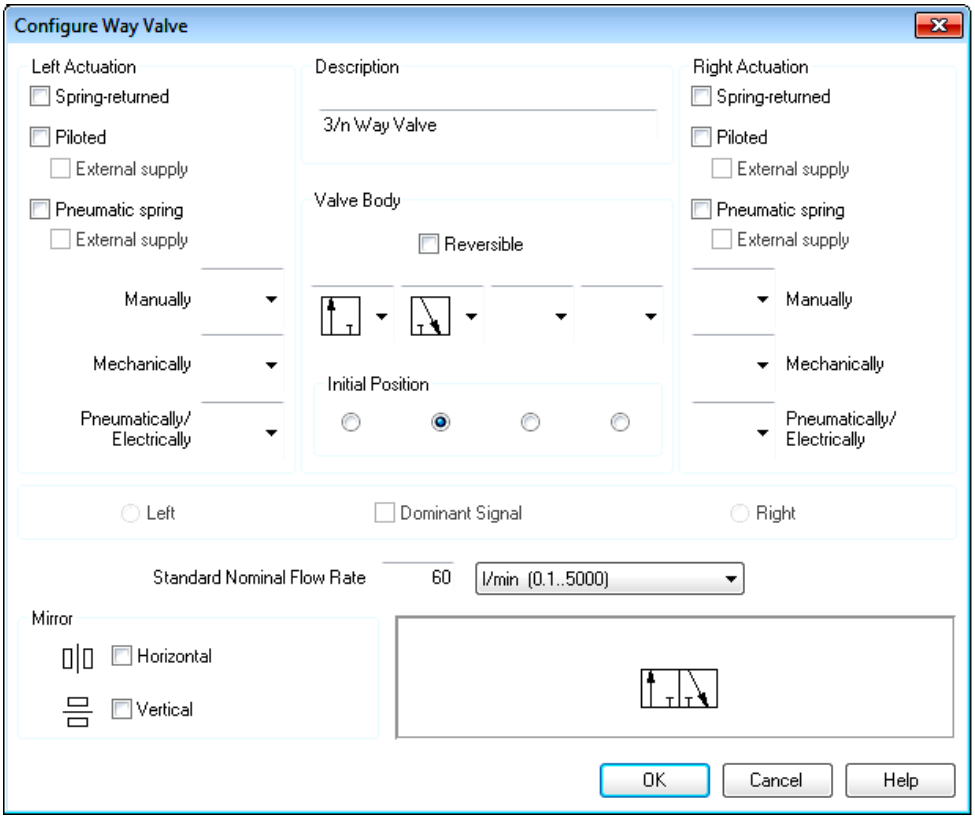

Figura 2.15. Ventana de Configuración de la Válvula. (Hecha Mediante FluidSIM)

Para el lado derecho, solamente activaremos la opción “Retorno de muelle” (Spring Returned) que hará que la válvula regrese a su posición original por medio de un resorte una vez que desactivemos el lado izquierdo, Presionamos el botón “Aceptar” (Ok) y la Válvula quedara de la siguiente manera:

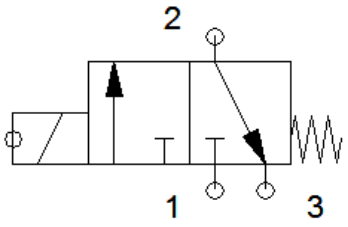

Figura 2.16. Válvula Configurada. (Hecha Mediante FluidSIM)

Lo único que resta es poner una etiqueta dando doble Clic sobre la conexión del elemento que se añadió del lado izquierdo, el nombre que le pongamos (p.e Y1) se le pondrá también al solenoide de válvula en el circuito eléctrico, entonces podremos empezar la simulación y analizar el comportamiento.

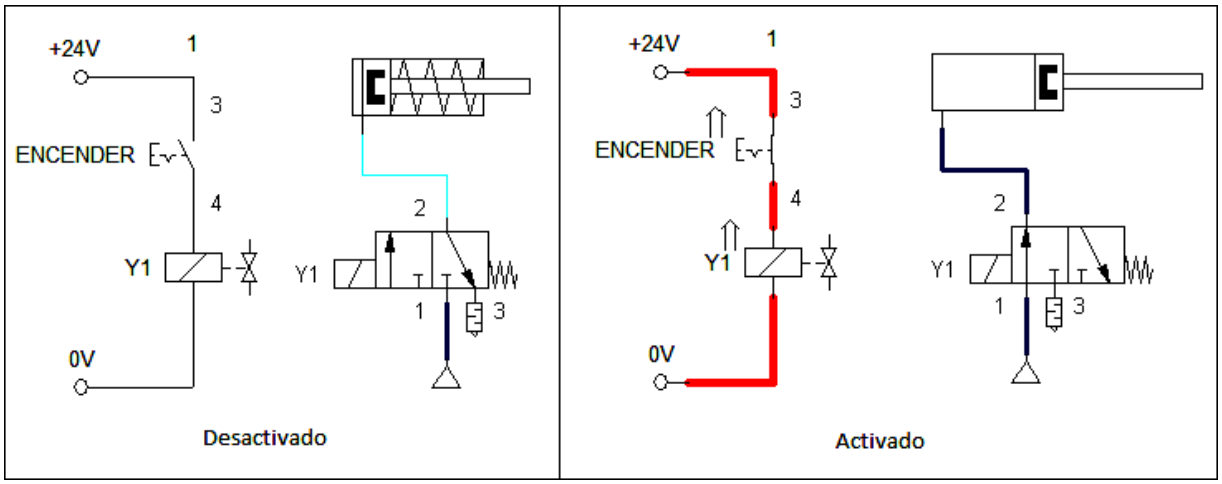

Figura 2.17. Simulación Circuito Electroneumático. (Hecha Mediante FluidSIM)

Todos los ejemplos anteriores, nos muestran cómo funciona FluidSIM a grandes rasgos. Para facilitar las explicaciones se usaron diagramas muy elementales que no permiten explotar todas las características y alcances del software, aunque se ser deseado, los usuarios podrían encontrar dentro de FluidSIM toda clase de ejemplos didácticos que le permitan adentrarse dentro de este simulador.

Estos ejemplos están dentro del Menú “Archivo”, al dar Clic en la opción “Vista Previa del Circuito” (Circuit Preview), o en la barra de herramientas, al dar Clic en el icono que está entre los iconos de “Guardar” e “Imprimir”.

2.3 PLC en FluidSIM

Ahora que conocemos cómo funciona FluidSIM, es momento de adentrarnos en las herramientas de programación de PLC.

Anteriormente mencionamos que, “Técnica Digital” o “Digital Technique” es la categoría en la Biblioteca de Componentes que alberga todas las herramientas y elementos de programación de un PLC.

Estas herramientas, el modo de ingresarlas y la forma de conectarlas unas con otras no es precisamente idéntica a como lo sería en un software dedicado a la programación de PLCs reales, pero nos proporciona la teoría que comparten todos los software de programación y nos deja experiencia que usaremos al momento de trabajar con equipos reales.

Iniciemos aclarando que el dispositivo que vamos a programar en FluidSIM es un PLC virtual, el cual podremos acoplar a los Circuitos Eléctricos y Electroneumáticos.

Esta característica le da a FluidSIM una ventaja sobre un software de programación especializada, al menos en el aspecto didáctico, ya que FluidSIM nos da la opción de simular el arreglo externo del PLC, como los Interruptores, Electroválvulas, Actuadores, es decir, en FluidSIM no solo se simula el control de un proceso, sino también los dispositivos que este va a controlar.

La Categoría de “Técnica Digital” tiene disponible veintiocho componentes, de los cuales, dos de ellos son “Módulos Digitales” (Logic Module), que son la forma de representar el PLC en FluidSIM.

La única diferencia entre estos dos módulos es que uno tiene ocho entradas y ocho salidas digitales, mientras que el otro tiene dieciséis entradas y dieciséis salidas digitales que transfieren sus estados a un circuito de conmutación digital en la parte interna del mismo módulo.

Se designa la letra “I” para las entradas, y la letra “Q” para las salidas, ambas acompañadas con un numero.

Haciendo doble Clic con el botón izquierdo del ratón sobre el Módulo Digital, se pasa a la zona de trabajo del circuito digital en la parte interior, que puede ser tratado del modo habitual.

Dentro del Módulo Digital sólo pueden insertarse componentes digitales (Componentes de “Técnica Digital”). Además, no es posible insertar Módulos Digitales adicionales dentro de un Módulo. Sin embargo, pueden utilizarse varios Módulos Digitales en un circuito Electroneumático.

Un Módulo Digital sólo funciona correctamente si se establecen las correspondientes conexiones en las alimentaciones eléctricas, dicho de otra manera, también hay que conectarlo a las fuentes de tensión (24v y 0v). (Art Systems, Festo Didactic, 2007)

Otra función incluida en FluidSIM es la opción de simular el circuito interno del Módulo Digital independiente a la simulación del Circuito Electroneumático con el fin de analizar el comportamiento de los elementos digitales por separado.

En la siguiente figura se muestran todas los componentes de “Técnica Digital”, incluidos los Módulos Digitales:

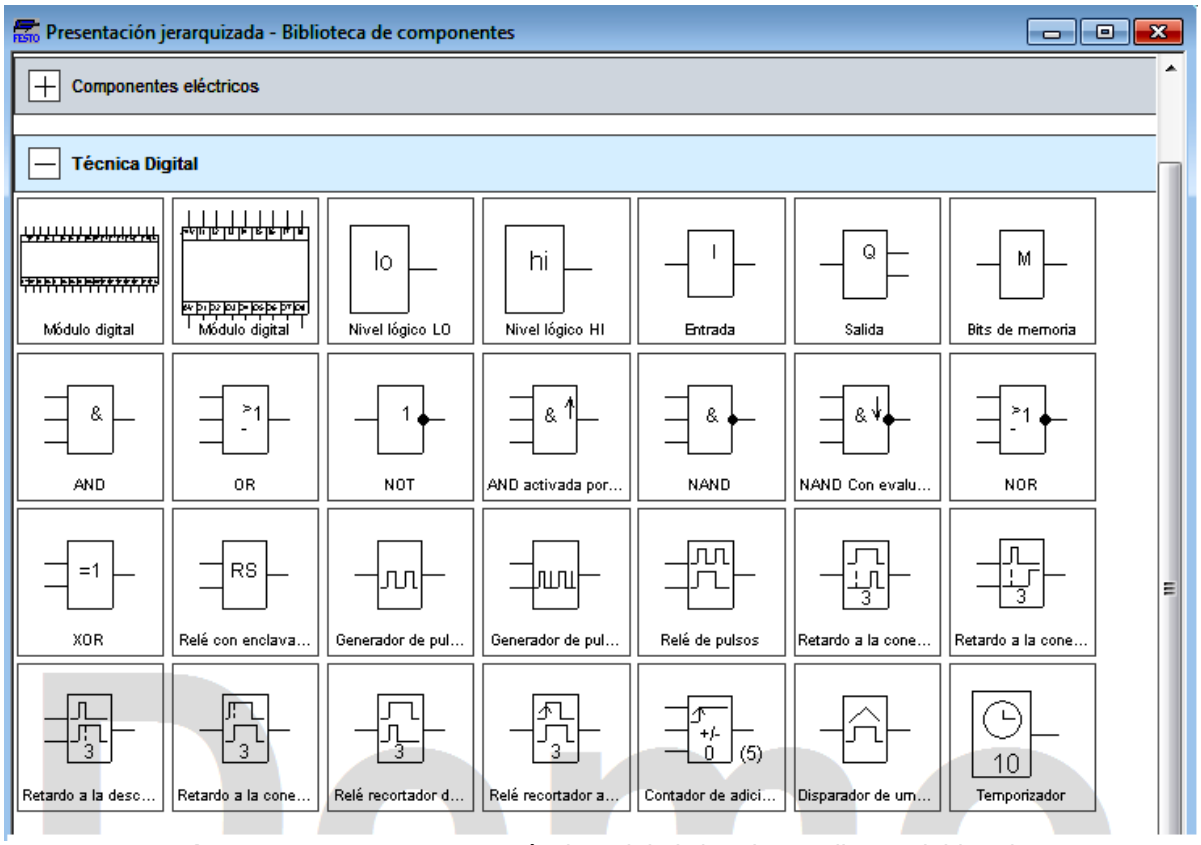

Figura 2.18. Componentes Técnica Digital. (Hecha Mediante FluidSIM)

Los componentes anteriores, se pueden dividir en tres grupos:

- **Funciones Básicas**

  AND, AND activada por flancos, NAND, NAND Con evaluación de flancos, OR, NOR, XOR, NOT.

- **Funciones Especiales**

  Módulo Digital, Retardo a la Conexión, Retardo a la Desconexión, Retardo a la Conexión/Desconexión, Retardo a la Conexión con Retención, Relé con Enclavamiento, Relé de Pulsos, Relé Recortador del Pulso de Salida, Relé Recortador Accionado por Flancos, Temporizador, Contador de Adición/Substracción, Generador de Pulsos Simétrico, Generador de Pulsos Asíncrono, Disparador de Umbral de Frecuencia

- **Constantes/Conectores**

  Entrada Digital, Salida Digital, Bits de Memoria, Nivel Lógico HI, Nivel Lógico LO, Conexión (digital), Conducto (digital), Distribuidor en T (digital).

Las Funciones Básicas no son más que las Compuertas Lógicas que hemos explicado en capítulos anteriores, las cuales con ayuda de la Funciones Especiales y Constantes permiten crear un sinfín de aplicaciones.

Al igual que en apartados anteriores, explicaremos con ejemplos prácticos el modo de conectar los componentes digitales dentro del Módulo Digital (PLC virtual) y cómo este interactúa con un circuito eléctrico.

Una vez más tomaremos el ejempló de la lámpara, solo que en esta ocasión necesitamos que dos interruptores estén cerrados para que la luz encienda. Su circuito eléctrico sin Módulo Digital seria como en la siguiente figura:

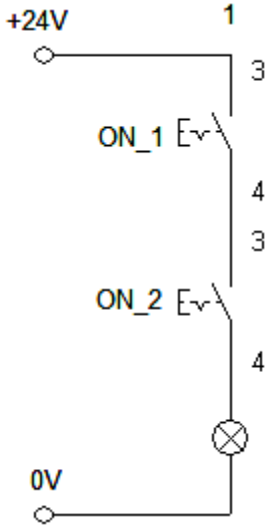

Figura 2.19. Ejemplo sin PLC. (Hecha Mediante FluidSIM)

El circuito con Módulo Digital quedaría tal como la Figura 2.20, donde se usaron las entradas I1 y I2 para conectar los interruptores, mientras que se uso la salida Q1 para alimentar la lámpara.

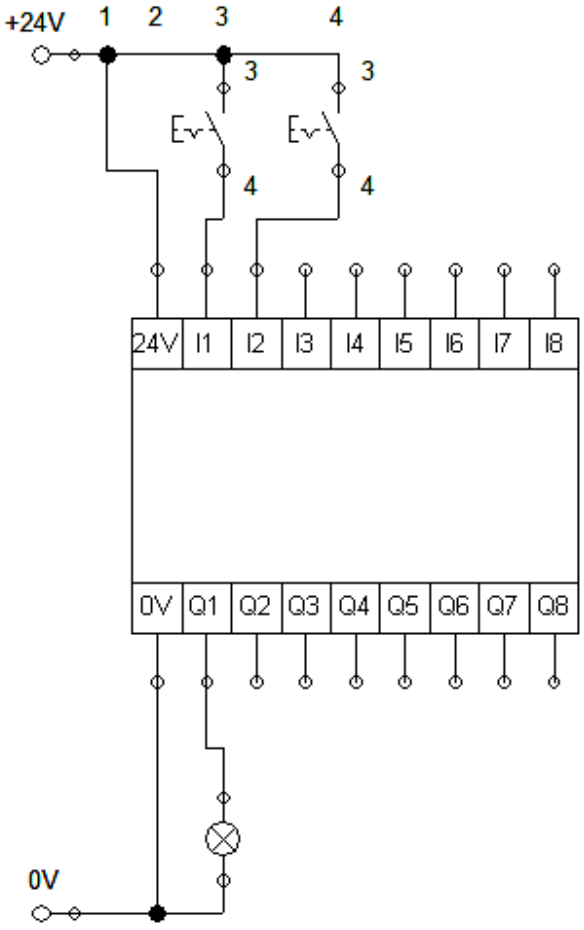

Figura 2.20. Ejemplo con PLC. (Hecha Mediante FluidSIM)

Ahora solo resta armar el circuito interno del Módulo Digital (Dar Doble Clic sobre el mismo), donde la función deseada la podremos implementar mediante una compuerta lógica “AND”, conectada como se muestra a continuación:

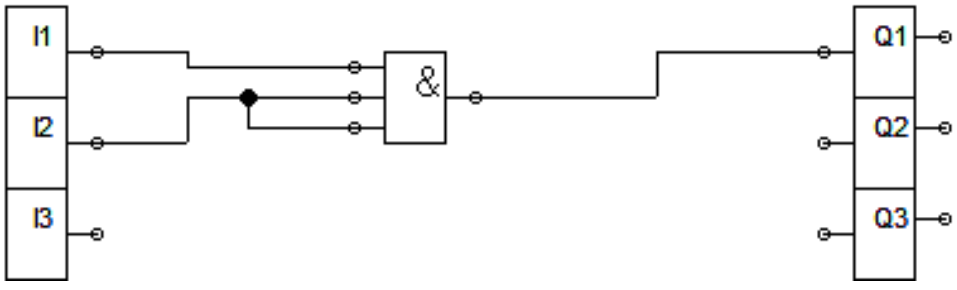

Figura 2.21. Conexión AND. (Hecha Mediante FluidSIM)

Cabe aclarar que AND, así como la mayoría de las Compuertas Lógicas en FluidSIM tienen tres entradas por defecto, pero a veces solo será necesario usar dos, cuando esto suceda conectaremos la entrada sobrante a alguna de las otras dos, de lo contrario FluidSIM indicara falla por “Entradas Abiertas”, de ser omitida esa alerta se considerará que en esa entrada hay un estado “1” o “HI” (explicado anteriormente).

Una vez terminado el arreglo interno, simplemente cerramos la ventana correspondiente a este circuito y pasaremos a la simulación, verificando que tienen que cerrarse los dos interruptores para encender la lámpara.

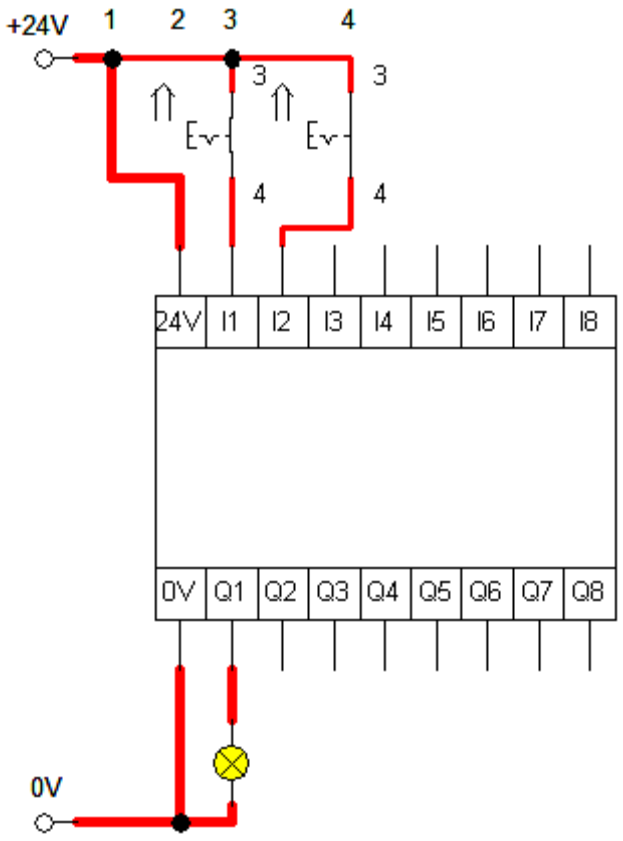

Figura 2.22. Simulación Módulo Digital. (Hecha Mediante FluidSIM)

El ejemplo anterior podría parecer que complica la tarea deseada, esto causado por la simplicidad del mismo, pero nos sirvió para mostrar como acoplar un PLC virtual dentro del circuito eléctrico.

Ahora realizaremos un ejemplo que nos deje incluir distintos tipos de Compuertas Lógicas y además simplifique el arreglo general del circuito.

Supongamos un circuito para activar un cilindro neumático de simple efecto por medio de un botón pulsador y un botón de paro que permita regresar al cilindro a su posición original. Para este tipo de circuito hay que usar enclavamiento eléctrico ya que se usaran botones pulsadores. A continuación se muestra el circuito sin el uso de Módulo Digital:

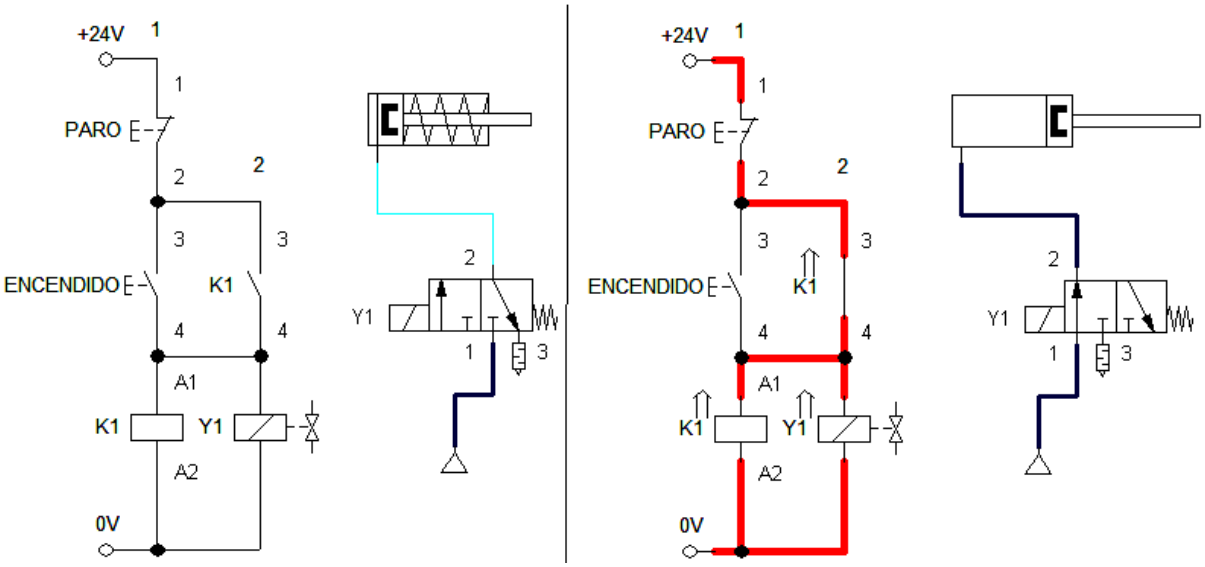

Figura 2.23. Circuito simulado sin PLC. (Hecha Mediante FluidSIM)

En la figura anterior podemos observar un ejemplo muy similar al que se encuentra en la Figura 2.17, con la única modificación de que esta usa botones pulsadores (sin enclavamiento mecánico), es por eso que se implemento el uso de un Relé para mantener Solenoide de Válvula energizado al momento que el pulsador se abra, finalmente se colocan las etiquetas necesarias para el correcto funcionamiento.

Nótese que para “apagar” el circuito se ocupó un segundo pulsador pero normalmente cerrado, que se coloca al principio del circuito con el fin de cortar la energía en el momento deseado.

Si incluimos un Módulo Digital, el arreglo antes mostrado se cambia por el siguiente (Figuras 2.24 y 2.25):

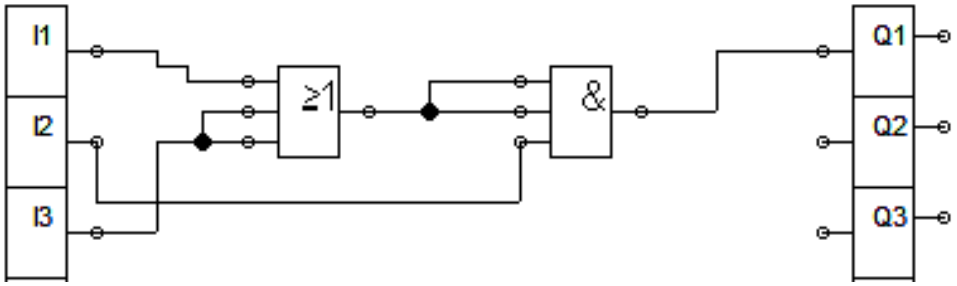

Figura 2.24. Arreglo Interno. (Hecha Mediante FluidSIM)

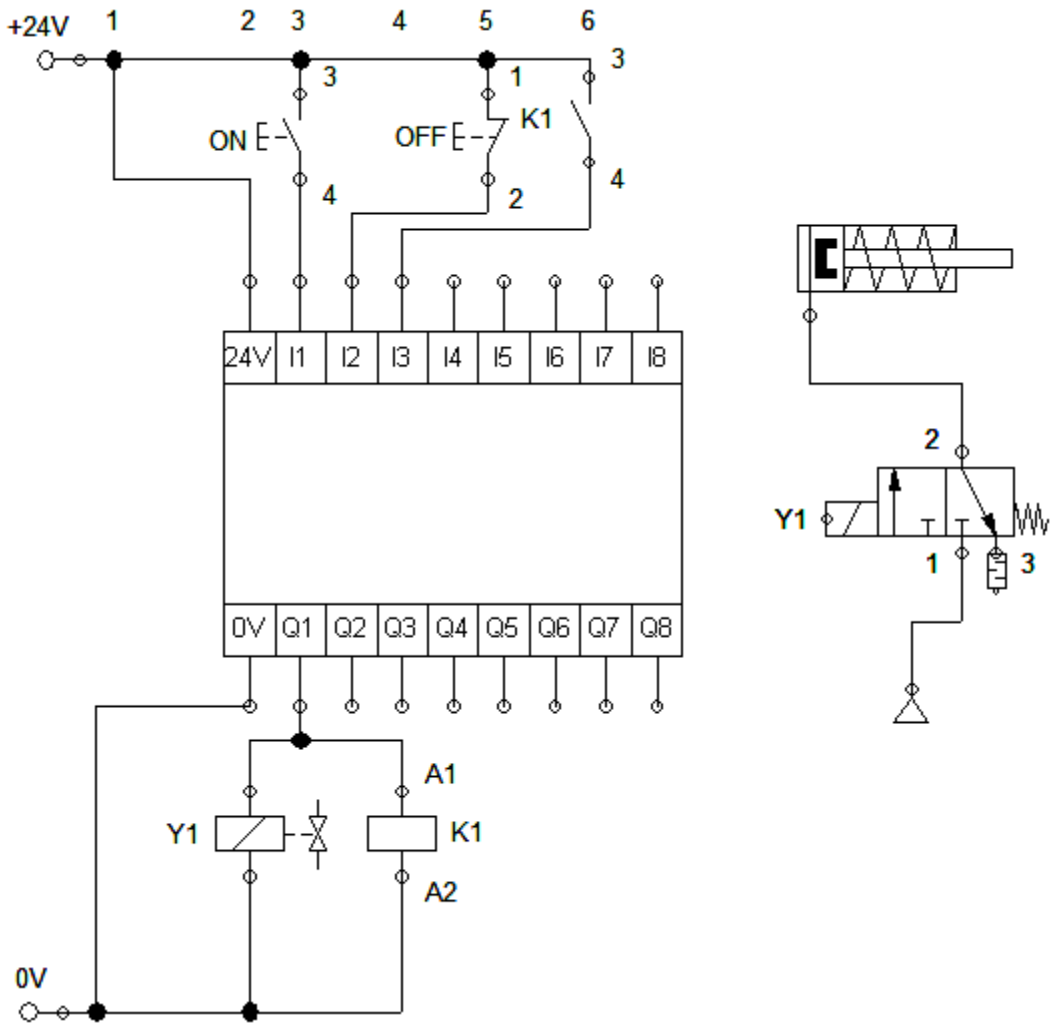

Figura 2.25. Arreglo Externo. (Hechas Mediante FluidSIM)

En el arreglo interno, podemos aclarar que la Compuerta OR se ocupa para “encender”, esta será activada por el pulsador “ON” o por el contacto “K1”, mientras que la compuerta AND tiene la función de “apagar”. Simule para comprobar estas funciones.

La ventaja de la implementación del Módulo Digital es que, entre más experiencia tenga el programador, mejores serán las optimizaciones, en el ejemplo anterior es válido pensar que no hay una simplificación del circuito, pero después de analizar mejor las herramientas disponibles se puede cambiar los arreglos de las figuras 2.24 y 2.25 por el de la Figura 2.26.

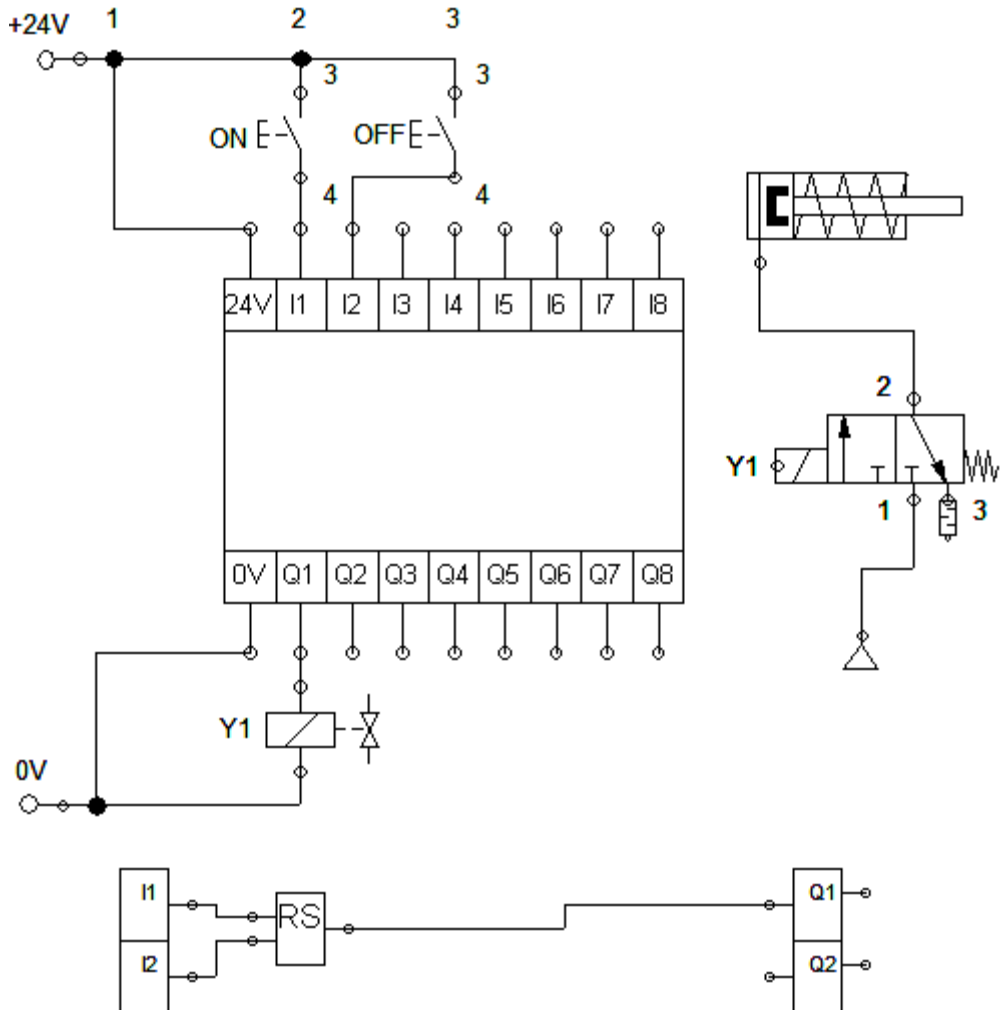

Figura 2.26. Arreglo Alternativo Simplificado. (Hechas Mediante FluidSIM)

En este ultimo arreglo, la disminución de componentes reside en el circuito interno, donde se inserta un Relé con enclavamiento o como en el capitulo anterior llamamos “SET-RESET”, elemento que con un impulso manda a su salida un estado “1” o “HI” y con otro impulso lo cambia a “0” o “LO”, que es lo que activa y desactiva el Solenoide de Válvula.

Es muy importante comprender que el hecho de que tenga enclavamiento interno vuelve prescindible el Relé “K1” que se tenía en el diagrama anterior, por lo tanto el Contacto Auxiliar “K1” también lo tenemos que eliminar. Cabe resaltar que en el arreglo de la Figura 2.26 el pulsador “OFF” es normalmente abierto con el fin de poder enviar los impulsos.

Solo resta simular el arreglo de la Figura 2.26 para comprobar que efectivamente realiza la misma tarea que el Circuito Eléctrico de la Figura 2.23.

De esta forma concluimos la explicación básica de la incorporación del Módulo Digital dentro de los circuitos Eléctricos y Electroneumáticos, tal como se sugirió anteriormente, la opción “Vista Previa del Circuito” ofrece una categoría dedicada a “Técnica Digital” donde se encuentran ejemplos más completos de los arreglos internos del PLC virtual, los cuales muestran el uso de todos los componentes digitales para distintos tipos de aplicaciones.

Capítulo III: Aplicación

3.1 Proyecto Práctico

En este capítulo abordaremos dos ejercicios en particular, respaldados bajo circunstancias reales, que deje aclarar aún más el funcionamiento y alcances del Módulo Digital, pero sobre todo dejar resaltada la importancia de incluir el PLC en los Circuitos de Control Industrial.

3.1.1 Empacadora de Rollos

Se necesita un circuito de control para dos cilindros neumáticos que forman parte de una maquina Empacadora de Rollos, la cual tiene que llenar cajas con cuatro rollos cada una. Nombraremos como cilindro “A” aquel que se encarga de suplir de cajas vacías y como cilindro “B” a el que llena las cajas. La secuencia cíclica de movimiento es: A+/A-/B+/B-/B+/B-/B+/B-/B+/B-, donde el signo “+” indica que el cilindro “Sale”, mientras que “-” indica el regreso del cilindro. La Figura 3.1 muestra un esquema de la maquina.

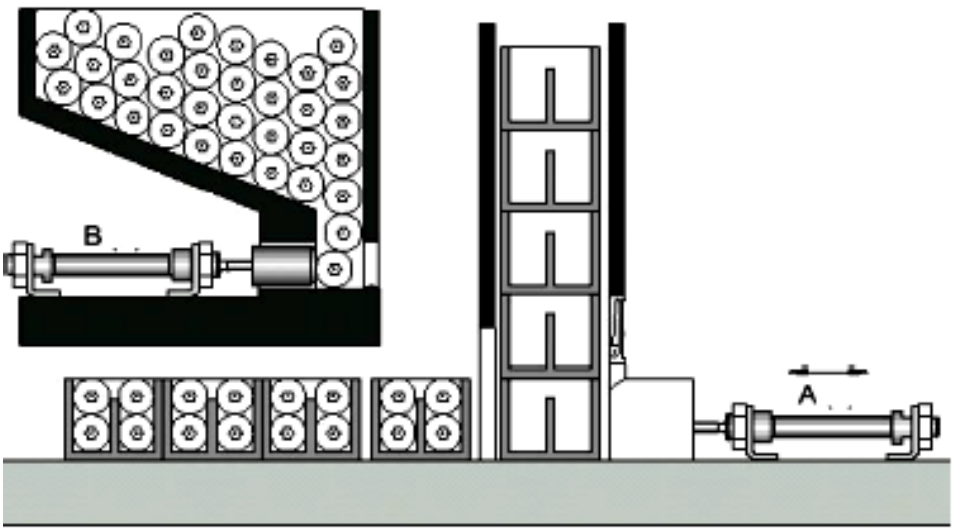

Figura 3.1. Empacadora de Rollos. (Modificado de Piñeros Calderón, 2011)

Primero mostraremos el Circuito de Control hecho bajo teorías del Automatismo Clásico, donde el control reside en el cableado y la disposición de Relés con sus Contactos Auxiliares (Ver Figura 3.2).

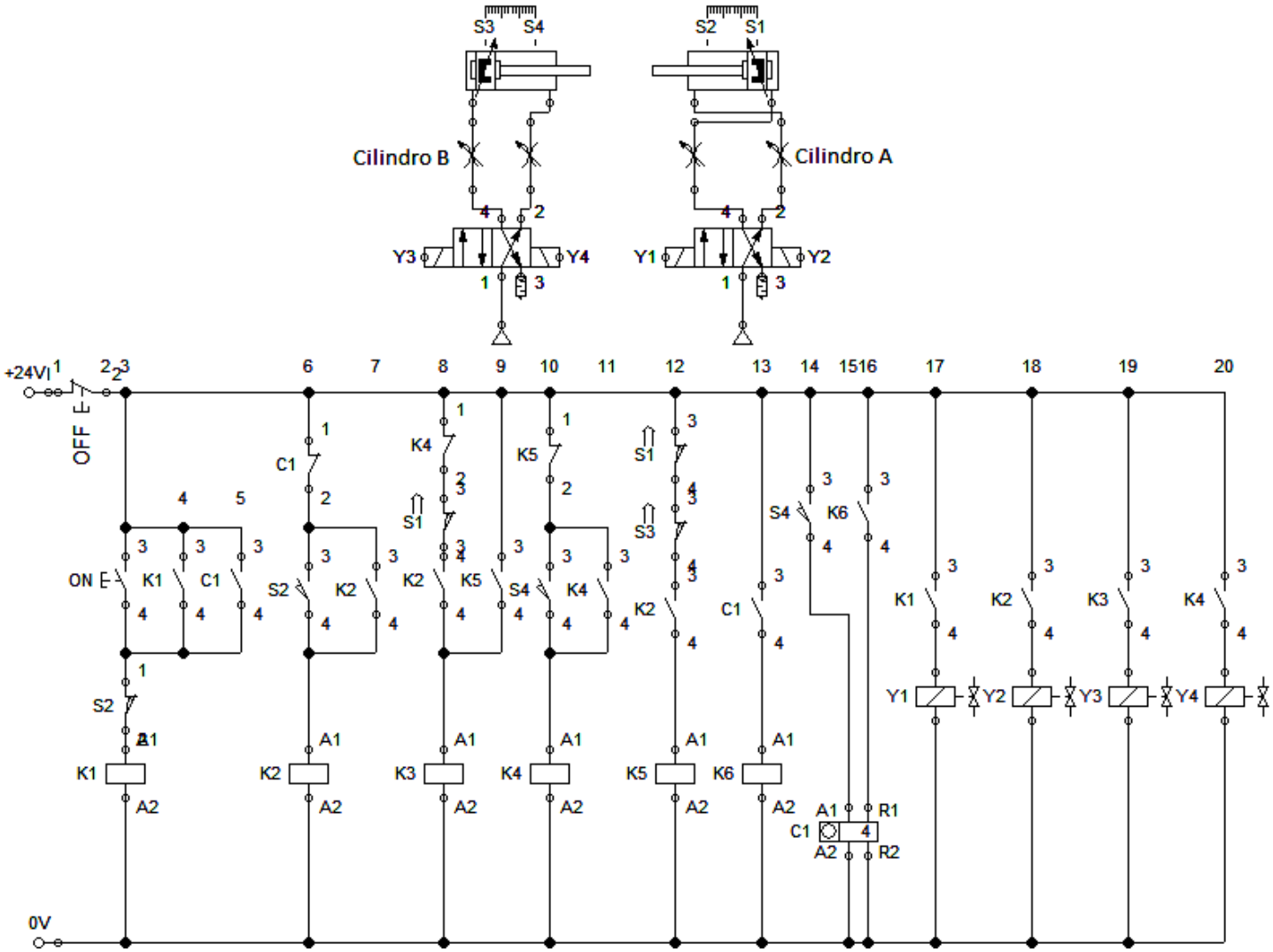

Figura 3.2. Diagrama Electroneumático de Control de Empacadora de Rollos. (Hecha Mediante FluidSIM)

Como podemos observar, una aplicación de esta magnitud requiere un número considerable de Contactos Auxiliares, numerosos Relés además de los Solenoides de Válvulas esto sin contar que el espacio necesario para conectarlos también creció.

Con base en este ejemplo podríamos calcular lo que implica una aplicación más compleja. Ahora presentamos un circuito de control para la misma aplicación pero con la diferencia que se incluye el PLC. Ver Figura 3.3.

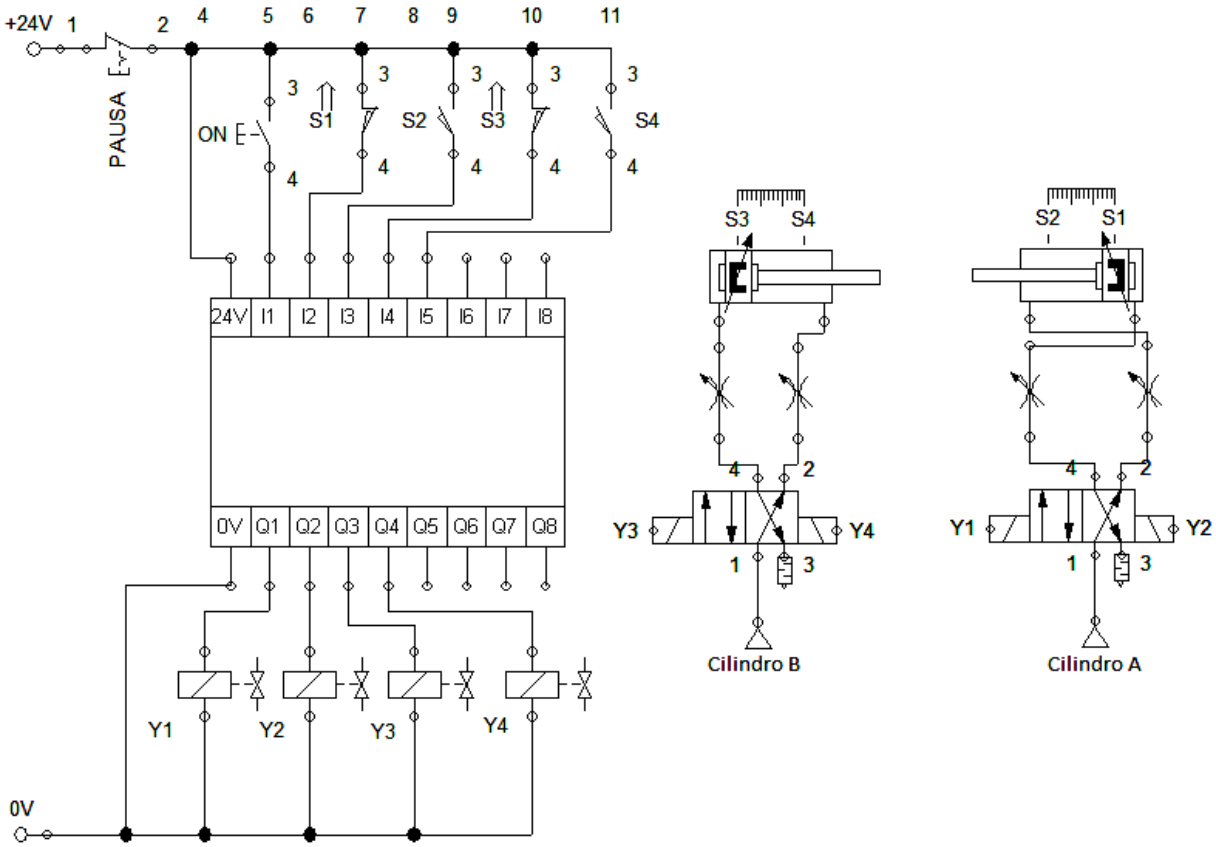

Figura 3.3. Circuito para Empacadora de Rollos, con PLC. (Hecha Mediante FluidSIM)

Es notable cómo se reduce el número de componentes en el circuito gracias al acoplamiento del PLC. Ahora la operación del circuito funciona por medio del arreglo interno de los Componentes Digitales, el cual se muestra en la Figura 3.4.

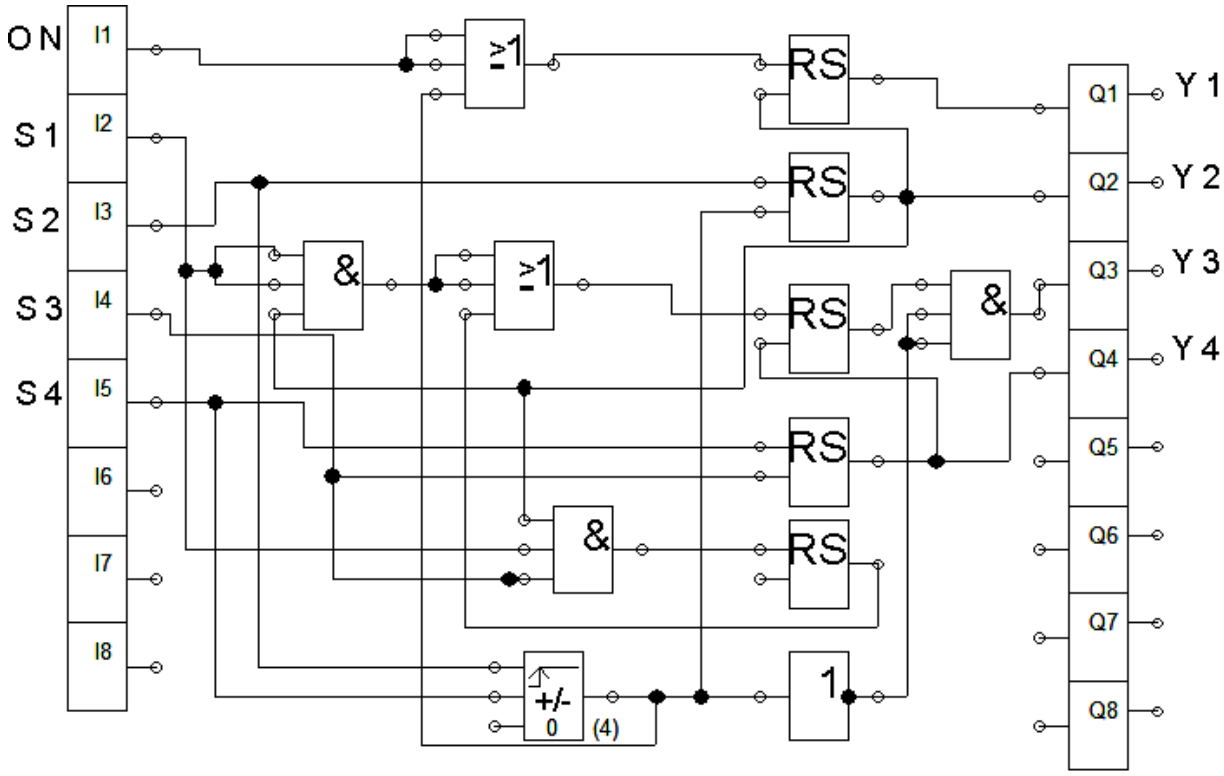

Figura 3.4. Circuito Interno del PLC de la Empacadora de Rollos. (Hecha Mediante FluidSIM)

El arreglo interno del PLC parece más complicado de lo que realmente es, la conexión y desconexión de los Solenoides de Válvula se realiza mediante los elementos “RS” (explicado anteriormente), entonces las restricciones de conexión se establecen por las Compuertas Lógicas.

La única desventaja que algunos usuarios podrían llegar a encontrar es que todas las conexiones son físicas entre los elementos, causando una saturación visual en el circuito, pero después de un corto tiempo de práctica ese problema se resuelve.

Otros componentes que valen la pena explicar son las “Reglas de Distancia” (Distance Rule), se localizan sobre los cilindros, dentro de la configuración de los cilindros se encuentra la opción para introducirlas o dentro de la Biblioteca de Componentes, junto a los actuadores (arrastrar hasta colocarla encima del cilindro).

En cada regla de recorrido podemos insertar “Marcas” o “Etiquetas”, cada una con una posición definida en el recorrido del cilindro.

Mediante estas configuraciones estaríamos ingresando “Interruptores en Cilindro”, es decir, al ingresar una marca estamos introduciendo un Interruptor de Posición, el cual ocupamos para crear las restricciones de activación de los Solenoides de Válvula. En esta aplicación ocupamos dos Interruptores de Posición por cilindro, uno en cada extremo del recorrido.

3.1.2 Dobladora de Barras Metálicas

Para la siguiente aplicación necesitamos un circuito de control de una maquina dobladora de barras metálicas, que está compuesta por tres cilindros neumáticos. Un cilindro sujetara la barra metálica mientras que las otras dos realizan los dobleces. La secuencia de funcionamiento es la siguiente: A+/B+/B-/C+/C-/A-. La Figura 3.5 muestra un esquema de la maquina.

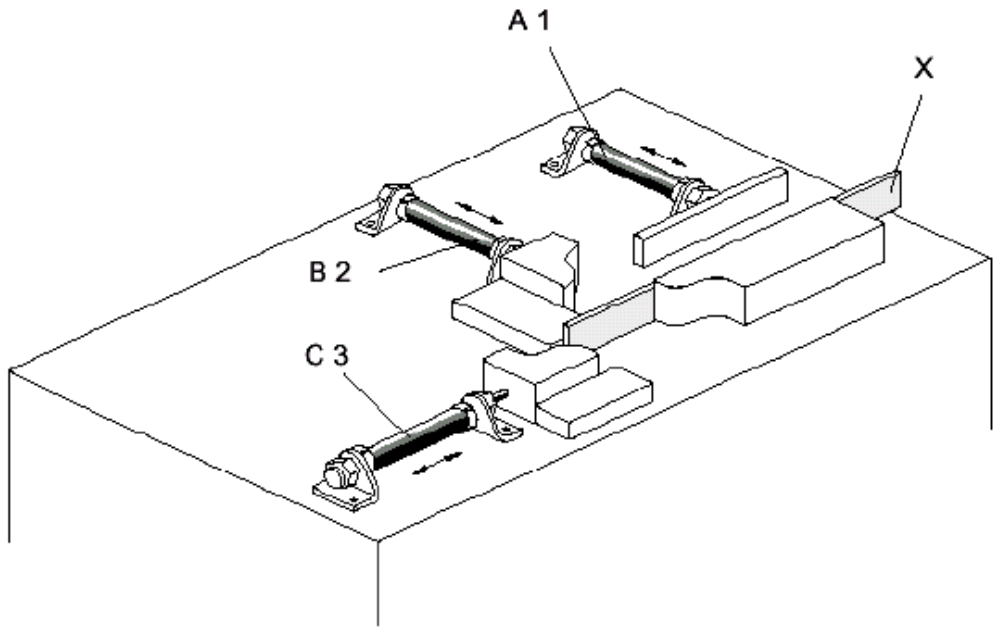

Figura 3.5. Dobladora de Barras Metálicas. (Modificado de Piñeros Calderón, 2011)

Así como en la Empacadora de Rollos, primero mostraremos el circuito de control sin el uso del PLC. Ver Figuras 3.6 y 3.7.

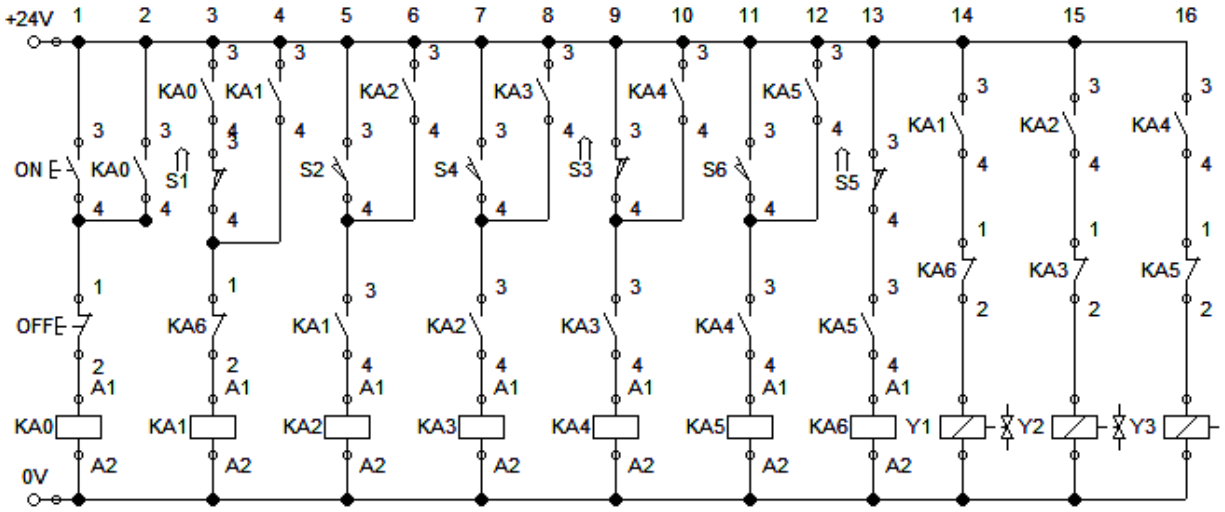

Figura 3.6. Diagrama Eléctrico de Dobladora Barras sin PLC. (Hecha Mediante FluidSIM)

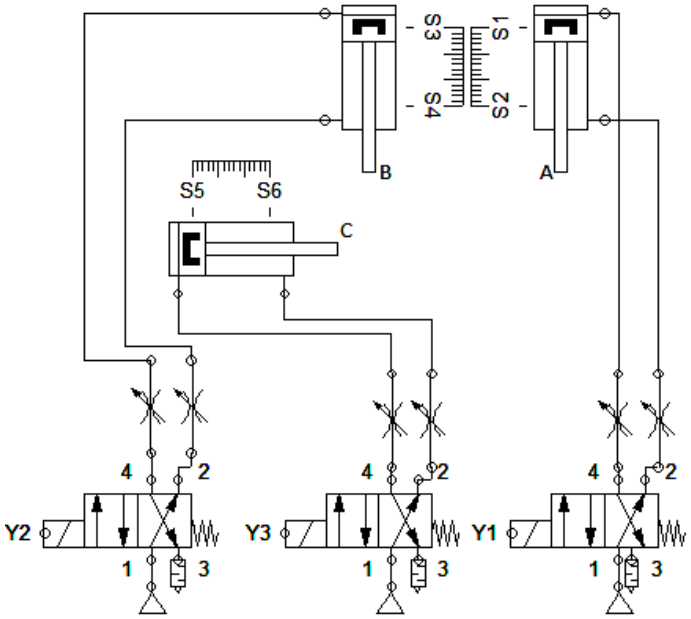

Figura 3.7. Diagrama Neumático de Dobladora de Barras sin PLC. (Hecha Mediante FluidSIM)

Al igual que el circuito de la Empacadora de Rollos, la cantidad de componentes necesarios es muy grande. Veamos qué pasa cuando se hace uso del PLC para el control de la misma máquina. Ver Figura 3.8.

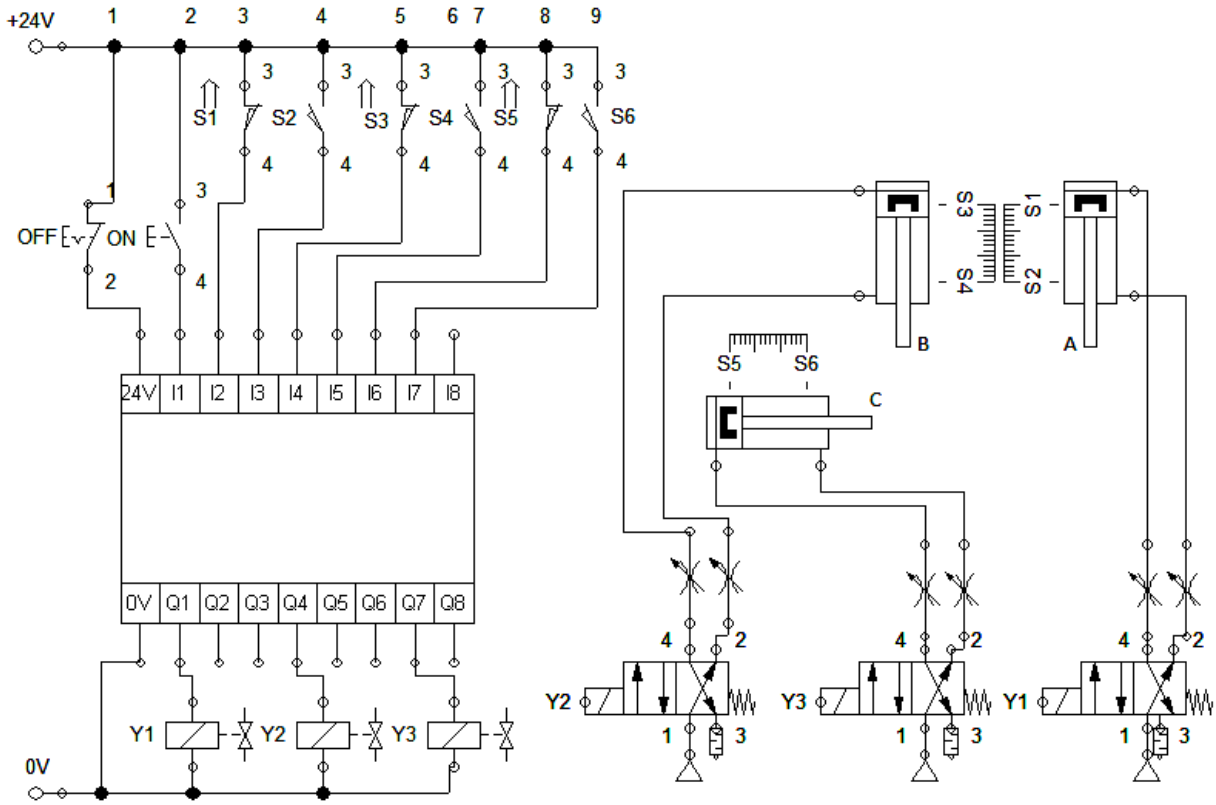

Figura 3.8. Circuito de Dobladora de Barras con PLC. (Hecha Mediante FluidSIM)

Es evidente la disminución de componentes físicos ocupados, los más notables son el número de Contactos Auxiliares, que bajo de veinticinco a seis y que los Relés se vuelven prescindibles ya que todo el control está dentro del PLC, donde se usan elementos virtuales. Ver Figura 3.9.

De esta forma se demuestra una vez más los grandes beneficios que tiene el integrar un PLC a los Circuitos de Control Industrial.

3.2 Resultados

Tanto en la Empacadora de Rollos como en la Dobladora de Barras, quedan evidenciadas las deficiencias que tiene un Circuito sin PLC contra un que si lo incluye, principalmente en el número de componentes que utilizan.

Por un momento dejemos de pensar que FluidSIM es un simulador, e imaginemos que los circuitos del capítulo anterior los teníamos que hacer con componentes reales.

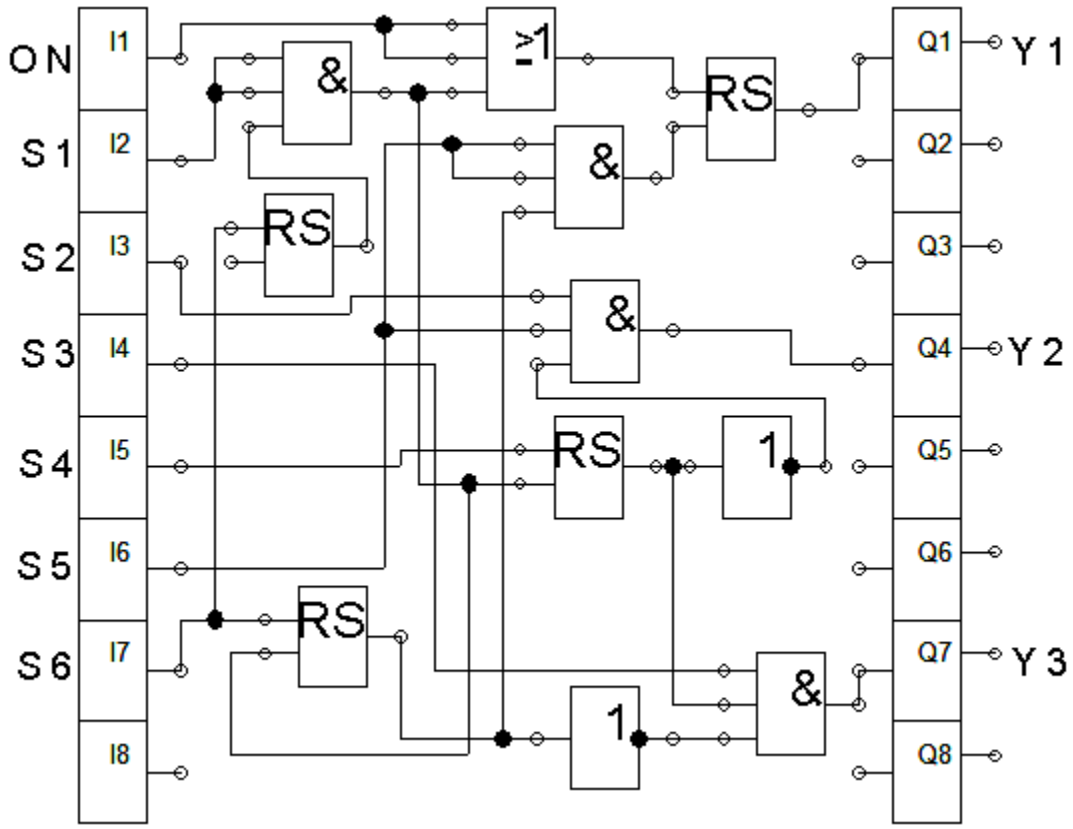

Figura 3.9. Circuito interno Dobladora de Barras. (Hecha Mediante FluidSIM)

Si se hubieran hecho de la forma clásica, el presupuesto para los componentes sería elevado y esos componentes difícilmente podrían ocuparse en otra aplicación, ya que fueron comprados para las funciones de un circuito en específico, entonces, un circuito hecho bajo el Automatismo Clásico es más caro y en caso de necesitarse tiene pocas probabilidades de reutilizar sus componentes.

Ahora, si el eje principal de nuestro circuito es un PLC, el costo de todos los componentes necesarios sería mucho menor, y aún más importante, el PLC es 100% reutilizable en una infinidad de aplicaciones, incluso podría albergar más de un circuito de control. Entonces, en la vida real, el circuito de la Dobladora de Barras pudo haberse hecho reutilizando el PLC que se usó en la Empacadora de Rollos puesto que solo tendríamos que reprogramarlo, de hecho, los Solenoides de Válvula, Botones Pulsadores, incluso el cable conductor se podrían reutilizar.

Conclusión

Podríamos definir cinco ventajas de incluir los PLCs a los circuitos de control, mismas ventajas que siempre se están buscando en todos los campos de la Ingeniería Aplicada:

- Economía
- Reducción de Tiempo
- Optimización de Espacios
- Eficiencia
- Versatilidad

Es un hecho que las tecnologías se vuelven obsoletas con el paso del tiempo. Los campos de la ingeniería aplicada no están exentos de esta situación, es por eso que todos los ingenieros deben dejar de sentirse ajenos a las nuevas tecnologías.

Concretamente, el campo del control industrial, no es exclusivo de una industria moderna, sin darnos cuenta se ha vuelto parte indispensable de nuestro diario vivir, a tal grado que el más sencillo grupo de semáforos es controlado por un PLC.

Son estas razones las que evidencian que la automatización ya no es un campo exclusivo de la Ingeniería Electrónica o Mecatrónica, y todas las personas involucradas dentro de la industria deben conocer al menos los métodos más elementales de automatización mediante PLC y así evitar el rezago en un mundo que avanza rápidamente.

Bibliografía

Art Systems, Festo Didactic. (2007). *Festo FluidSIM 4, Manual de Usuario.*

AutomatykaOnline. (s.f.). Recuperado el Noviembre de 2013, de http://automatykaonline.pl/wp-content/uploads/2013/09/Siemens-6ED1052-1FB00-0BA7-31.jpg

Cano, G. A. (s.f.). *Aprende en Linea.* Recuperado el Noviembre de 2013, de Universidad de Antioquia: http://aprendeenlinea.udea.edu.co/revistas/index.php/coltrem/article/viewFile/7287/6731

Cembranos Nistal, F. J. (1998). *Sistemas de Control Secuencial.* Madrid: EDICIONES PARANINFO, S.A.

Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado. (s.f.). Recuperado el Noviembre de 2013, de http://recursostic.educacion.es/secundaria/edad/4esotecnologia/quincena6/quincena6_contenidos_1a.htm

Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado. (s.f.). Recuperado el Noviembre de 2013, de http://roble.pntic.mec.es/jlop0164/archivos/electronica-digital-3.pdf

Navarro G., D. A. (2001). *Controlador Lógico Programable (PLC).* Curso Tutorial, Universidad de Oriente,Escuela de Ingeniería y Ciencias Aplicadas, Puerto La Cruz,Venezuela.

Palazzesi, A. (21 de marzo de 2007). *ucontrol.* Recuperado el Noviembre de 2013, de http://www.ucontrol.com.ar/Articulos/Compuertas/compuertas.htm

Piñeros Calderón, J. J. (2011). ELECTRONEUMATICA. *Tech Time* (1).

Siemens AG. (2010). *Lista de instrucciones (AWL) para S7-300 y S7-400.* Manual de Referencia, Nürnberg.

Southland Electrical. (s.f.). Recuperado el Noviembre de 2013, de http://www.southlandelectrical.com/mmSOUTHLAND/Images/PLC/ALLEN_BRADLEY/1761-L16BWA_D_(1000).JPG

Universidad Nacional Abierta y a Distancia. (s.f.). Recuperado el Noviembre de 2013, de http://datateca.unad.edu.co/contenidos/2150512/ContenidoLinea/leccin_1632__diagrama_de_bloques_funcionales_function_block_diagram__fbd.html

Universidad Nacional de La Plata. (s.f.). *Facultad de Ingenieria, Universidad Nacional de La Plata.* Recuperado el Noviembre de 2013, de http://www.ing.unlp.edu.ar/electrotecnia/procesos/apuntes/Diagrama%20Escalera.pdf